令和6年度 卒業論文

ペトリネットに基づく量子最適化の効率化

Efficient quantum optimization based on Petri nets

琉球大学工学部工学科
知能情報コース

215752H　上地 涼太
指導教員 名嘉村 盛和

要旨

近年，量子最適化技術が最適化問題の解決手段として注目を集めている．特に，量子アニーリングや量子近似最適化アルゴリズム (QAOA) は、組合せ最適化問題に適用される手法として研究が進められている．しかし，QUBO や Ising モデルへの定式化が困難であり，大規模かつ複雑な制約を持つ問題に対して高品質な解を得るための定式化手法が求められている．

本研究では，ペトリネットを用いた量子最適化の効率化を目的とし，ペトリネットモデリングに基づく問題の定式化手法を開発する．具体的には，多資源フローショップスケジューリング問題 (MRFSSP) を対象とし，ペトリネットモデルを活用することで，スケジューリング問題の QUBO モデルへの変換を容易にする．

また，モデル変換による変数削減手法を提案する．これは二段階最適化に基づくもので最初に与えられた組合せ最適化問題に対してモデル化したペトリネットモデルを一段目の最適化計算でネットモデル変換を行う．変換後のペトリネットから生成される QUBO モデルは変数の数が削減される．モデル変換の前後の比較実験により，待機時間の縮小を目的とした最適解の探索が可能であることを示し，より効果的なスケジューリングが可能であることを確認した．

本研究の結果は，大規模な組合せ最適化問題における量子計算の実用化に向けた一歩となるものであり，今後の展望としては，リバース量子アニーリングの導入や多目的最適化を維持したモデル変換による高品質な解の探索の改善，有効性を確認したい．

Abstract

In recent years, quantum optimization techniques have attracted much attention as a means of solving optimization problems. In particular, quantum annealing and quantum approximate optimization algorithms (QAOA) have been studied as methods applied to combinatorial optimization problems. However, their formulation into QUBO and Ising models is difficult, and there is a need for formulation methods to obtain high-quality solutions for problems with large and complex constraints.

This study aims to improve the efficiency of quantum optimization using Petri nets, and to develop a formulation method for the problem based on Petri net modeling. Specifically, we focus on the multi-resource flow store scheduling problem (MRFSSP) and utilize the Petri net model to facilitate the conversion of the scheduling problem to the QUBO model.

We also propose a method of variable reduction through model transformation. This is based on two-stage optimization, in which a Petri net model modeled for a given combinatorial optimization problem is transformed into a net model in the first stage of optimization computation. The QUBO model generated from the transformed Petri net reduces the number of variables. Comparison experiments before and after the model transformation show that it is possible to find the optimal solution for the purpose of reducing the waiting time, and that more effective scheduling is possible.

The results of this study are a step toward the practical application of quantum computation in large-scale combinatorial optimization problems. Future prospects include the introduction of reverse quantum annealing and the improvement and effectiveness of searching for high-quality solutions by model transformation while maintaining multi-objective optimization.

目 次

第 1 章　序論 **1**
1.1　背景と目的 1
1.2　論文の構成 1

第 2 章　基礎概念 **2**
2.1　ペトリネット 2
2.2　量子アニーリング 2
2.3　多資源フローショップスケジューリング問題 3
2.4　TPE(Tree-structured Parzen Estimator) 3

第 3 章　ペトリネットに基づく変換前の多目的量子最適化 **4**
3.1　多資源フローショップスケジューリング問題 4
3.1.1　ペトリネットモデル 4
3.1.2　CPNTools によるモデル生成 5
3.1.3　QUBO 定式化 9
3.2　評価実験 10
3.3　考察 13

第 4 章　ペトリネット変換に基づく二段階最適化 **14**
4.1　効率よく解を探索するためのマシンリソース配置 14
4.2　評価実験 15
4.2.1　タスク処理時間を最適化したペトリネットモデル変換 (戦略 1) . . . 16
4.2.2　リソースコストを最適化したペトリネットモデル変換 (戦略 2) . . . 18
4.2.3　ペトリネット変換前と戦略 1 の比較 20
4.2.4　ペトリネット変換前と戦略 2 の比較 21
4.3　考察 21

第 5 章　結論 **23**
5.1　まとめ 23
5.2　今後の展望 23

図 目 次

3.1 フローショップシステムのペトリネットモデル 4
3.2 ペトリネット変換前の MRFSSP のパレート解 11
3.3 5 個のジョブに対するスケジュール 12
3.4 計算できる最大規模数 (job 数が 6) のパレート解 12

4.1 変数削減のためのペトリネット変換 14
4.2 タスクの処理時間を最適化したパレート解 16
4.3 戦略 1 の計算できる最大規模数 (job 数が 7) のパレート解 17
4.4 リソースコストを最適化したパレート解 18
4.5 戦略 2 の計算できる最大規模数 (job 数が 7) のパレート解 19
4.6 ペトリネット変換前と戦略 1 のパレート解 20
4.7 ペトリネット変換前と戦略 2 のパレート解 21

表 目 次

3.1 XML ファイルから取得した情報 9
3.2 エネルギー関数内の変数 9
3.3 各巡回における実行可能解と個数および平均値 11
3.4 最大規模数の各巡回における実行可能解と個数および平均値 13

4.1 各巡回における実行可能解と個数および平均値 (戦略 1) 16
4.2 戦略 1 の最大規模数の実行可能解と個数および平均値 18
4.3 各巡回における実行可能解と個数および平均値 19
4.4 戦略 2 の最大規模数の実行可能解と個数および平均値 19

第1章　序論

1.1　背景と目的

近年，最適化問題に対するアプローチとして量子最適化技術が注目を集めている．特に，量子アニーリングと量子近似最適化アルゴリズム (QAOA) は，量子コンピュータのために設計された最適化アルゴリズムでとして広く認知されている [1], [2]．量子最適化の特徴は，従来の古典的アルゴリズムでは解決が困難な問題に対して，高速かつ効率的に最適解を求める潜在能力がある点である．一方で，量子最適化計算の技術は進展しているものの，実用化および普及に向けては多くの課題が存在する．特に，最適化問題をQUBO(Quadratic Unconstrained Binary Optimaization) や Ising モデルで定式化するプロセスが容易ではなく実用化が困難である．また，大規模な問題，制約や目的関数の複雑性による解の質の劣化が実用化における障壁となっている．

本研究では，ペトリネット理論に基づく QUBO や Ising モデルの定式化技術の開発を行う．具体的には，ドメイン知識に基づくモデルベースの定式化手法と効率的な探索を実現するための最適化問題の変換技術を組み合わせることにより，大規模かつ複雑な制約を含む問題に対しても高品質な解が得られる枠組みの構築を目指す．さらに，提案する手法の有効性を確認するために，問題の規模を拡大した場合における性能を評価し従来の手法との解の質の比較を行う．

1.2　論文の構成

本論文は，第 1 章に序論を述べ，第 2 章で本実験で使用した基礎概念，第 3 章で本実験で扱う技術を学ぶために実施した関連研究と評価実験に関して記述する．第 4 章で本実験の効率化の手法に関する詳細を述べ，最後の第 5 章にてまとめと今後の課題に関して記述する．

第2章　基礎概念

2.1　ペトリネット

ペトリネット $PN = (N, M_o)$ は，プレース，トランジションの 2 種類のノードからなる有向 2 部グラフ $N = (P, T, Pre, Post)$ と初期マーキング M_0 で表される [3],[4].

$P = p_1, p_2, ..., p_n$ はプレースの集合，$T = t_1, t_2, ..., t_n$ はトランジションの集合である．プレースにトークンを配置することにより現在のシステムの状態を表すことができる．$Pre(p, t)$ はトランジション t に入る入力プレース p を接続するアークの重みを表している．$Post(p, t)$ はトランジション t から出ている出力プレース p を接続するアークの重みを表している．各トランジション t の入力プレースに重み以上のトークンが配置されている時トランジション t は発火可能であると言う．トランジションの発火は事象の生起を表し，トークンの分布変化が起こる．具体的には，トランジションが発火すると入力プレースから重みの分だけトークンが消費され，出力プレースに重みの分だけトークンが生成されることによってシステムの状態が変化する．

2.2　量子アニーリング

量子アニーリングは量子コンピュータを用いた一種の最適化アルゴリズムであり，最適化問題を QUBO モデルや Ising モデルとしてエネルギー関数の形で表現し，エネルギーを最小化することで最適解に近い解を高速に求めることが期待されている．量子アニーリングは，D-Wave 社が開発した商用量子アニーリングマシンにより数万量子ビットの計算が可能であり，実用化が進んでいる．さらに，デジタル方式で QUBO モデルや Ising モデルに基づく最適化処理を行う量子インスパイヤードシステムや，GPU を用いてアニーリングを模倣するシステムも提案されており，これらは新たな組み合わせ最適化プラットフォームとして注目を集めている．

QUBO モデルは (2.1) 式のように表すことができる．

$$H = \sum_{i,j} Q_{i,j} q_i q_j \tag{2.1}$$

ここで $Q_{i,j} \in \mathbb{R}$ は，バイナリ変数 ($q_i \in \{0, 1\}$) の係数を表している．また，Ising モデルは (2.2) 式のように表すことができる．

$$H = \sum_{i<j} J_{i,j} s_i s_j + \sum_i h_i s_i \tag{2.2}$$

ここで $J_{i,j} \in \mathbb{R}$ は，スピン変数 $(s_i = \pm 1)$ の間の相互作用を表している．また，$h_i \in \mathbb{R}$ はスピン s_i に作用する外部磁場を表している．

2.3 多資源フローショップスケジューリング問題

多資源フローショップスケジューリング問題 (MRFSSP) は，複数の工程からなる複数のジョブを処理するスケジューリング問題の一種である [5]．各ジョブは複数の工程によって構成され，各工程は指定された共有資源の利用を必要とする．ジョブ内の各工程はタスクと呼ばれる．

本論文で扱っている MRFSSP を以下にまとめる．

1. ジョブの数は N 個，それぞれのジョブは M 個のタスク (工程に対応) で構成される．
2. 各マシンの単位時間当たりの稼働コストが予め与えられている．
3. 各マシンの単位作業時間当たりの処理時間が予め与えられている．
4. 各ジョブは M 個のタスクを決められた順番で処理する必要があり，その順番は全てのジョブで同一である．
5. 各タスクは予め指定されたマシンリソース群の中から一台のマシンを利用して処理がなされる．一度スタートした処理は中断なく完了まで行われる．
6. 各タスクは同じジョブの直前タスクの完了以前には処理は開始できない (先行制約)．
7. 各ジョブの全てのタスクに対して，マシンが割り当てられている (完了制約)．
8. 任意の二つのタスク間でマシンの競合が発生しない (競合制約)．

また，目的関数として次の二つを考える．

9. 各タスクのリソースの待ち時間の総和を最小化する．
10. リソースコストの総和を最小化する．

2.4 TPE(Tree-structured Parzen Estimator)

TPE とは，Parzen 推定 (カーネル密度推定) に基づいてパラメータ最適化を効率的に行うための手法である．Parzen 推定は観測データに基づいて目的関数の値が小さい値を出すパラメータを良い解，大きい値を出すパラメータを悪い解としそれらに対する確率分布を学習する．良い解に近いパラメータを選びやすくするためにこれらの確率分布を利用し次に試すパラメータを探索する [6]．これにより，無駄な探索を避け効率的に最適解を見つけることができる [7]．

第3章　ペトリネットに基づく変換前の多目的量子最適化

量子インスパイアード最適化の論文 [8] の定式化を参考に MRFSSP の各タスクのリソースコストとタスク間の待機時間の最適化を図る検証を行った.

3.1　多資源フローショップスケジューリング問題

3.1.1　ペトリネットモデル

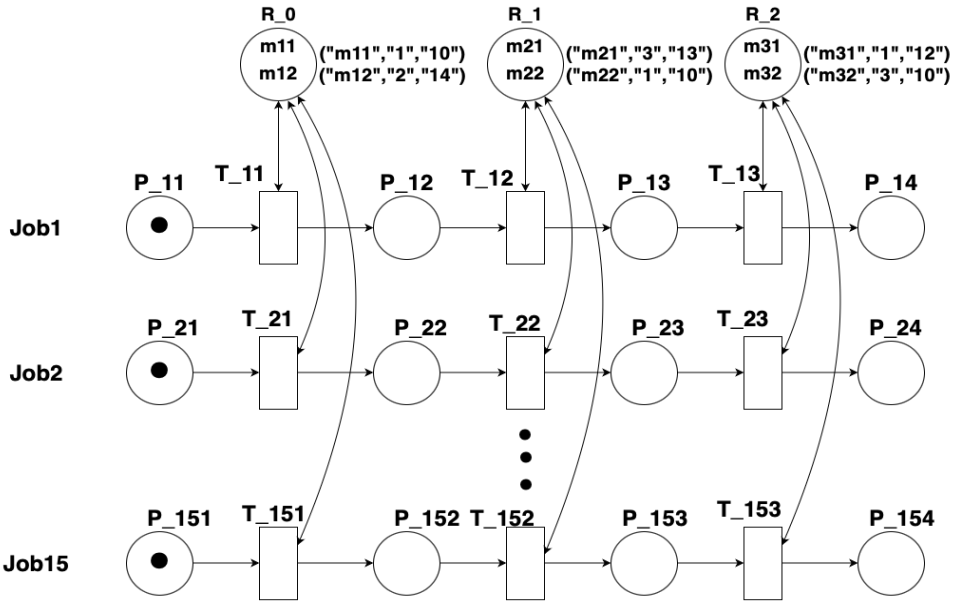

図 3.1: フローショップシステムのペトリネットモデル

MRFSSP に対するペトリネットモデルは，図 3.1 のように表現できる．図 3.1 では，15 個のジョブを省略した形で記載している．各ジョブはプレース，トランジションの交互列となるパスで表されている．例えば，Job1 に対するパス $P_{1,1}, T_{1,1}, P_{1,2}, ..., P_{1,4}$ は，3 種類のタスク (工程)$T_{1,1}, T_{1,2}, T_{1,3}$ から構成されており，それぞれ，R_0, R_1, R_2 のマシンリソースを必要としている．3 種類のマシンリソースは，カラートークンとしてマシン ID，処理速度，マシンコストの情報を含む文字列で表されている．このシステムの目的は，ペトリネットにおける繊維が順序通りに発火することにより最終的に全てのトークンがモデル

の最右端に到達することである．これはスケジューリングの完了を意味する．この過程においては，システムに課された全ての制約が満たされることが前提となり，加えて目的関数の最適化が達成されることが重要である．すなわち，最適化されたスケジューリングにおいては，リソースの利用効率や処理時間などの複数の要素がバランスよく最適化される必要がある．このペトリネットモデルのようにドメイン知識さえあれば，わずかなペトリネットの記述のルールに基づいてペトリネットモデルを作成することができる．

3.1.2 CPNTools によるモデル生成

CPNTools は，カラードペトリネットのモデリングおよびシミュレーションを支援するツールである．本研究では，CPNTools を用いて生成したペトリネットモデルを XML 形式でエクスポートすることで，プレース，アーク，トランジションに関する構造情報および動作情報を取得する．

プレース，トランジション，アークの XML ファイルの一部を以下に示す．

Listing 3.1: プレースの XML ファイル

```
<place id="ID1412324394">
        <posattr x="-284.000000"
                 y="42.000000"/>
        <fillattr colour="White"
                  pattern=""
                  filled="false"/>
        <lineattr colour="Black"
                  thick="1"
                  type="Solid"/>
        <textattr colour="Black"
                  bold="false"/>
        <text>p_11</text>
        <ellipse w="60.000000"
                 h="40.000000"/>
        <token x="-10.000000"
               y="0.000000"/>
        <marking x="0.000000"
                 y="0.000000"
                 hidden="false">
          <snap snap_id="0"
                anchor.horizontal="0"
                anchor.vertical="0"/>
        </marking>
        <type id="ID1412324395">
          <posattr x="-244.000000"
                   y="18.000000"/>
          <fillattr colour="White"
                    pattern="Solid"
```

```
                    filled="false"/>
          <lineattr colour="Black"
                    thick="0"
                    type="Solid"/>
          <textattr colour="Black"
                    bold="false"/>
          <text tool="CPN Tools"
                version="4.0.1">UNIT</text>
        </type>
        <initmark id="ID1412324396">
          <posattr x="-255.000000"
                   y="65.000000"/>
          <fillattr colour="White"
                    pattern="Solid"
                    filled="false"/>
          <lineattr colour="Black"
                    thick="0"
                    type="Solid"/>
          <textattr colour="Black"
                    bold="false"/>
          <text tool="CPN Tools"
                version="4.0.1">1`()</text>
        </initmark>
      </place>
```

Listing 3.1 は，プレースに関する情報であり＜ place id ＞はアークとの関連付けに使用するためのユニークな ID である．＜ text ＞プレース＜/text ＞はプレースのラベルが指定されており人間が理解しやすい形になっている．＜ initmark id ＞はトークンのユニークな ID であり＜ text tool ＞トークンの数 ‘ () ＜/text ＞で初期状態でのトークンの数を表している．

Listing 3.2: トランジションの XML ファイル

```
<trans id="ID1412324537"
            explicit="false">
        <posattr x="-138.000000"
                 y="42.000000"/>
        <fillattr colour="White"
                  pattern=""
                  filled="false"/>
        <lineattr colour="Black"
                  thick="1"
                  type="solid"/>
        <textattr colour="Black"
                  bold="false"/>
        <text>t_11</text>
        <box w="60.000000"
             h="38.000000"/>
```

```
        <binding x="7.200000"
                 y="-3.000000"/>
        <cond id="ID1412324538">
          <posattr x="-177.000000"
                   y="72.000000"/>
          <fillattr colour="White"
                    pattern="Solid"
                    filled="false"/>
          <lineattr colour="Black"
                    thick="0"
                    type="Solid"/>
          <textattr colour="Black"
                    bold="false"/>
          <text tool="CPN Tools"
                version="4.0.1"/>
        </cond>
        <time id="ID1412324539">
          <posattr x="-93.500000"
                   y="72.000000"/>
          <fillattr colour="White"
                    pattern="Solid"
                    filled="false"/>
          <lineattr colour="Black"
                    thick="0"
                    type="Solid"/>
          <textattr colour="Black"
                    bold="false"/>
          <text tool="CPN Tools"
                version="4.0.1"/>
        </time>
        <code id="ID1412324540">
          <posattr x="-73.500000"
                   y="-9.000000"/>
          <fillattr colour="White"
                    pattern="Solid"
                    filled="false"/>
          <lineattr colour="Black"
                    thick="0"
                    type="Solid"/>
          <textattr colour="Black"
                    bold="false"/>
          <text tool="CPN Tools"
                version="4.0.1"/>
        </code>
        <priority id="ID1412324542">
          <posattr x="-206.000000"
                   y="12.000000"/>
```

```
63          <fillattr colour="White"
64                    pattern="Solid"
65                    filled="false"/>
66          <lineattr colour="Black"
67                    thick="0"
68                    type="Solid"/>
69          <textattr colour="Black"
70                    bold="false"/>
71          <text tool="CPN Tools"
72                version="4.0.1"/>
73        </priority>
74      </trans>
```

Listing 3.2 は，トランジションに関する情報でありプレース同様にプレースとアークの関連付けに使用するためのユニークな ID が付与されている．また，text 要素にはトランジションのラベルが指定されている．さらに，＜ cond ＞はトランジションの発火条件，＜ priority ＞は発火の優先順位，＜ time ＞トランジションの処理時間を記載する要素である．ただし，本研究では＜ cond ＞，＜ priority ＞，および＜ time ＞には設定を行っていない．

Listing 3.3: アークの XML ファイル

```
1  <arc id="ID1412324579"
2             orientation="PtoT"
3             order="1">
4          <posattr x="0.000000"
5                   y="0.000000"/>
6          <fillattr colour="White"
7                    pattern=""
8                    filled="false"/>
9          <lineattr colour="Black"
10                   thick="1"
11                   type="Solid"/>
12         <textattr colour="Black"
13                   bold="false"/>
14         <arrowattr headsize="1.200000"
15                    currentcyckle="2"/>
16         <transend idref="ID1412324537"/>
17         <placeend idref="ID1412324394"/>
18         <annot id="ID1422288249">
19           <posattr x="-211.000000"
20                    y="53.000000"/>
21           <fillattr colour="White"
22                     pattern="Solid"
23                     filled="false"/>
24           <lineattr colour="Black"
25                     thick="0"
```

```
26                    type="Solid"/>
27          <textattr colour="Black"
28                    bold="false"/>
29          <text tool="CPN Tools"
30                version="4.0.1">1`()</text>
31        </annot>
32      </arc>
```

Listing 3.3 は，アークに関する情報でありユニークな ID が付与されている．< arc orientation >では，アークの接続方向の指定がされている．PtoT はプレースからトランジション，TtoP はトランジションからプレースの接続を意味する．

上記の XML ファイルから得られた情報を表にまとめる．

表 3.1: XML ファイルから取得した情報

リスト	取得情報
resource_m	マシンリソースのラベルとリソースの複数の性能
machine_processing_time	単位時間あたりの処理時間
machine_cost	単位時間あたりのコスト
job	各ジョブのタスクを順番に並べた構造

3.1.3 QUBO 定式化

基本的な QUBO 定式化の方式に基づいて MRFSSP のペトリネットモデルからの定式化を行う．これらの式で用いた変数を表 3.2 にまとめる．

表 3.2: エネルギー関数内の変数

変数	定義
rc^r	単位時間あたりのマシンリソース r の使用コスト
fd^r	マシンリソース r を使ってタスクを処理する場合の処理時間
t_i^j	ジョブ j の i 番目のタスク
$x_k^r(t_i^j)$	時刻 k において t_i^j がマシンリソース r を使用していれば $x_k^r(t_i^j) = 1$，そうでなければ 0 を示すバイナリ変数

ペトリネットの発火規則に従うためには，以下のエネルギー関数をゼロにする必要がある．すなわち，E_{c1} 及び E_{c2} ともトランジションズの発火にはその入力プレース全てにトークンが配置されている必要があり，かつ，同じ入力プレースで同じトークンを前提に発火を行うことは禁止されているため，その競合状態にある場合には，一つのトランジションの発火のみに当該トークンが利用される必要がある．このようにペトリネットの発火規則から以下のエネルギー関数が生成できる．

$$E_{c1} = \sum_{k_1,k_2} \sum_{r} \sum_{(j_1,j_2)} \sum_{i} x^r_{k_1}(t^{j_1}_i) \cdot x^r_{k_2}(t^{j_2}_i) \tag{3.1}$$

$$E_{c2} = \sum_{k_1,k_2} \sum_{r_1,r_2} \sum_{j} \sum_{i} x^{r_1}_{k_1}(t^{j}_i) \cdot x^{r_2}_{k_2}(t^{j}_{i+1}) \tag{3.2}$$

一方，本スケジューリング問題は，全てのジョブが完了するまでタスクの実行計画を作成することであるので，全てのトランジションがただ一度ずつ発火する必要がある．これより，以下のエネルギー関数が生成できる．

$$E_{c3} = \left(1 - \sum_{k} \sum_{r} \sum_{j} \sum_{i} x^r_k(t^j_i)\right)^2 \tag{3.3}$$

目的関数として，リソースの総コストの最小化とタスクの処理の待機時間の最小化となるため，ペトリネットの振る舞いモデルより以下のエネルギー関数が導かれる．

$$E_{o1} = \sum_{k} \sum_{r} \sum_{j} \sum_{i} rc^r \cdot fd^r \cdot x^r_k(t^j_i) \tag{3.4}$$

$$E_{o2} = \sum_{i} \sum_{k_1,k_2} \sum_{r_1,r_2} \sum_{j} \sum_{i} (k_1 - fd^{r_2}) \cdot x^{r_2}_{k_2}(t^j_{i+1}) \cdot x^{r_1}_{k_1}(t^j_i) \tag{3.5}$$

式 (3.1), (3.2), (3.3), (3.4), (3.5) に重み A, B, C, D, E を乗じてまとめた全体のエネルギー関数は以下のようになる．

$$E = A \cdot E_{c1} + B \cdot E_{c2} + C \cdot E_{c3} \ + D \cdot E_{o1} + E \cdot E_{o2} \tag{3.6}$$

3.2 評価実験

ペトリネットモデルから QUBO モデルに定式化したエネルギー関数の最適化を行う．本実験では，OpenJij[9] を用いて最適化計算を行う．問題設定は以下に示す．

1. ジョブの数は 5 個.
2. パラメータ A,B,C は制約の重み.
3. パラメータ D は resource cost の重み.
4. パラメータ E は waiting time の重み.
5. 最適化計算を OpenJij で 100 回行いベイズ最適化でパラメータチューニングを行う.

6. 5. を 5 回繰り返す.

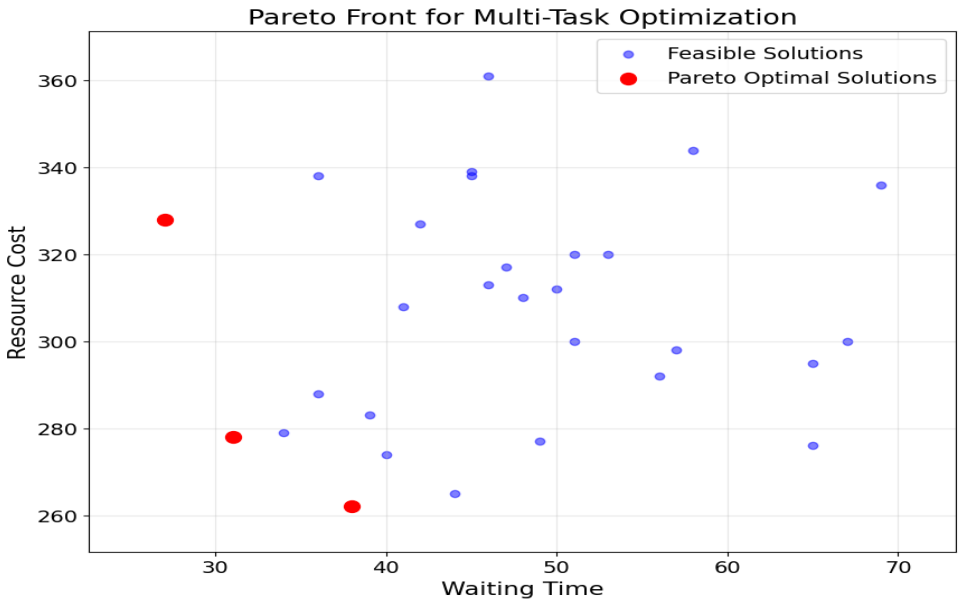

図 3.2: ペトリネット変換前の MRFSSP のパレート解

表 3.3: 各巡回における実行可能解と個数および平均値

巡回	個数	Resource Cost の平均値	Waiting Time の平均値
第 1 巡	9	312.6	47
第 2 巡	8	302.4	49.4
第 3 巡	6	300	44.7
第 4 巡	2	328	55.5
第 5 巡	4	297.5	44.8
全体の平均	5.8	308.1	48.3

表 3.3 は一度の最適化計算で得られた実行可能解の個数と，それらの Resource Cost および Waiting Time の平均値を示している．平均値は実行可能解で得られた数で算出している．また，図 3.2 はペトリネット変換前の MRFSSP の最適化計算を行い，得られたパレート解を可視化したグラフになっている．

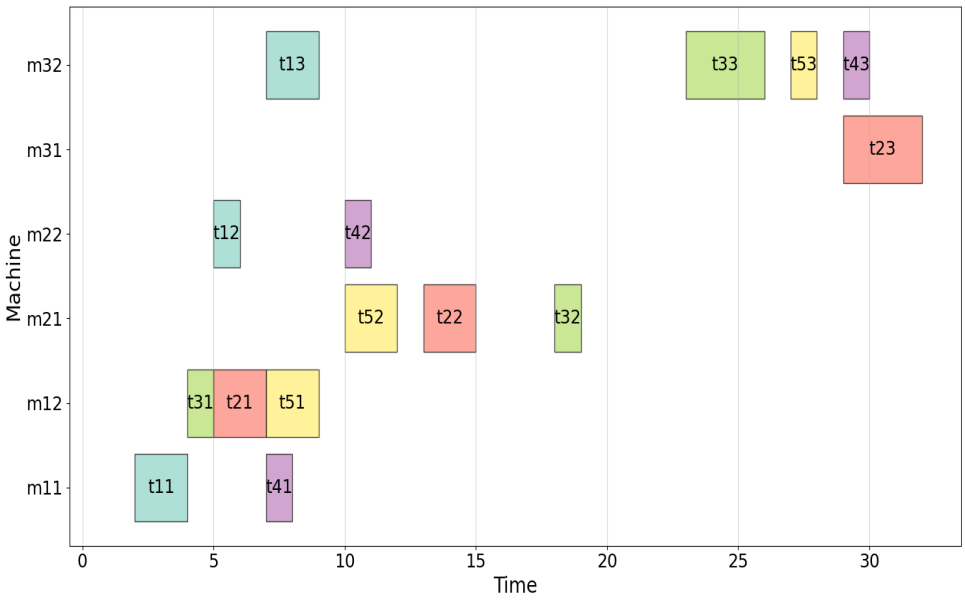

図 3.3: 5 個のジョブに対するスケジュール

図 3.3 は，計算結果の一例をガントチャートに表したものである．本ガントチャートでは，色分けによりジョブごとのタスクが区別されており，ジョブ全体の進行状況が視覚的に把握できる．また，各ジョブ内でタスクの工程順序が正確に守られており，依存関係が反映されたスケジュールになっていることが確認できる．さらに，全てのタスクが適切なマシンに割り当てられ，未処理のタスクが存在しないためリソースの競合や制約違反が発生していないことがわかる．

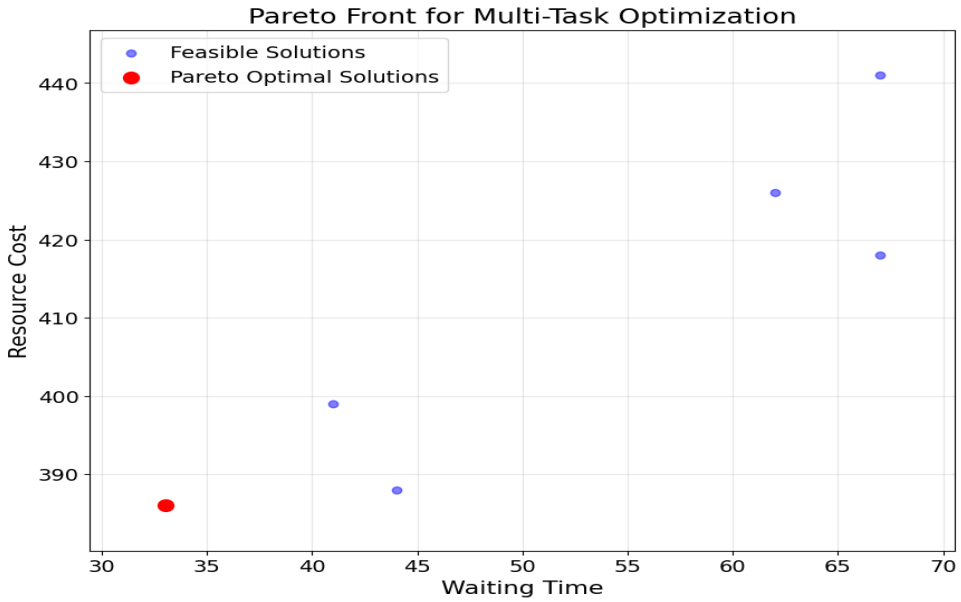

図 3.4: 計算できる最大規模数 (job 数が 6) のパレート解

表 3.4: 最大規模数の各巡回における実行可能解と個数および平均値

巡回	個数	Resource Cost の平均値	Waiting Time の平均値
第 1 巡	2	403	55.5
第 2 巡	1	426	62
第 3 巡	0	-	-
第 4 巡	2	420	54
第 5 巡	1	386	33
全体の平均	1.5	408.8	51.1

表 3.4 と図 3.4 はペトリネット変換前の MRFSSP の計算できる最大規模 (job 数が 6) の時の結果になっている．job 数が 5 の時と比較してパレート解の個数が減少していることから効率的な解を見つけることが困難であることがわかる．また，規模の拡大に伴いそれぞれの目的関数の値の増加も見られる．

3.3 考察

現時点での結果として得られたパレートフロントの形状から，複数の目的関数がトレードオフの関係になっていることが明らかとなった．具体的には，Waiting Time を短縮すると Resource Cost が増加し，逆に Resource Cost を抑制すると Waiting Time が増加する傾向が確認される．この結果は，どの目的関数を優先するかによって選択される解が異なることを示しており，パレートフロント上の解を選択することで効率的かつ実用的な意思決定を行えると考えられる．実行可能解の個数が少ない原因としては，各パラメータの範囲に対する制限の厳しさが影響している可能性があると考えられる．また，探索回数を 100 回に限定しているため，解空間の十分な探索が行われず，実行可能解が少なくあった可能性も示唆される．この結果は，パラメータ範囲の設定や探索回数の増加が解空間の拡大および実行可能解の増加に寄与する可能性があると思われる．

さらに，ジョブ数が 5 から 6 に増加した場合，Resource Cost は約 100，Waiting Time は約 3 増加することが確認された．Resource Cost の増加は，ジョブ数の増加に伴いリソースの割り当てが増加したことに起因すると考えられる．一方，Waiting Time の増加は，リソースの競合が激化しタスク処理の待機時間が延びたためと推測される．ジョブ数の増加により探索空間が指数関数的に拡大し探索効率が低下することが課題として挙げられる．この課題を克服するためには，解の多様性を維持しつつ探索効率を向上させるアルゴリズムの導入が必要である．特に，大規模問題への適用においては探索空間を効率的に絞り込む手法などの調整が求められる．

第4章　ペトリネット変換に基づく二段階最適化

多くの組合せ最適化問題の多くは NP-hard に分類され．その結果，問題の規模が増大するにつれて解候補の数が指数関数的に増加する．このため，制限された計算時間内での最適解を求めることが困難となる．この課題を軽減するために，解空間を効果的に縮小することで計算の複雑度を最小化するアルゴリズムや，最適解に近似した実現可能な解を効率的に特定するヒューリスティクスおよびメタヒューリスティクスが広く研究されている．本章では，量子アニーリングを効果的に活用するため，新たな手法としてペトリネット変換と問題分割による問題サイズの削減手法を提案する．さらに，提案手法の有効性を評価するために，解の質や計算可能な問題サイズの上限について検証を行う．

4.1　効率よく解を探索するためのマシンリソース配置

量子アニーリングは，変数の数が少ない場合により効率的に最適解を探索できる特性がある．この特性，問題の解探索に必要な量子ビットの数が変数の数に依存するためである．変数が少ないほど解の探索空間が小さくなり最適解に到達しやすくなる．このため，量子アニーリングを用いる場合は，対象問題を可能な限りコンパクトに表現し変数の数を削減することが重要である．特に，大規模問題を効率的に解くためには，ペトリネット変換や問題分割などの手法を用いて問題のサイズを適切に縮小することが必要である．

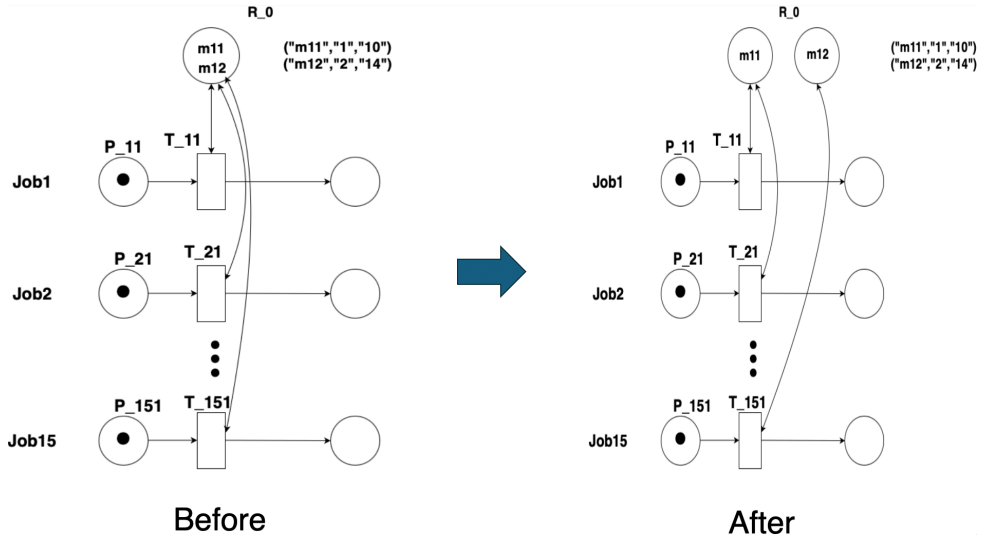

図 4.1: 変数削減のためのペトリネット変換

図 4.1 は対象問題のペトリネットモデルを変換した図になる．このモデル変換は以下の二段階のプロセスで構成される．

1. リソースの事前割り当てを適用することによりペトリネットモデルを変換する．ペトリネットモデルを変換するためのエネルギー関数が (4.1) 式である．

$$min \sum_{m \neq m_i}^{M} \left(\sum_{t}^{T} a_{m_i,t} \cdot x_{m_i,t} - \sum_{t}^{T} a_{m,t} \cdot x_{m,t} \right)^2 + \left(1 - \sum_{t}^{T} x_{m,t} \right)^2 \tag{4.1}$$

2. (4.1) 式をタスクの処理時間またはリソースコストの最適化を行うことで効率的なリソース配置をする．

(4.1) 式の M はマシンリソースの集合であり，T はタスクの集合である．第一項の目的関数は特定のタスクにおけるあるマシンリソースを基準として他のリソースとの間に差異が生じないようにすることを目的としている．具体的には，タスクの処理時間またはリソースコストの最適化を目的関数とした時に，各リソースの割り当てが均等であることを求めており，リソース間の不均衡を最小化することで全体的なリソースの利用効率を向上させることを目指している．この項は，リソース配分の不均等性による効率的ではない動作を防ぐために最適化において重要な役割を果たす．第二項は制約条件を表しており，1 つのタスクに対して 2 つ以上のマシンリソースが割り当てられないように制限している．この制約は，リソースの重複した割り当てを防ぎ各タスクにおけるリソースの一貫した分配を確保すつことを目的としている．

変数の数は $r \times k \times t$(ここで，r はマシンリソース数，k は制限の最大時刻，t はタスク数) で表される．だが，ペトリネットモデルの変換を行うことにより変数の数が $k \times t$ に削減される．この削減により，最適化計算の規模が縮小し計算効率が向上する．

4.2 評価実験

ペトリネットモデルの変換後，3.2 と同様な実験を行い比較検証をする．

4.2.1 タスク処理時間を最適化したペトリネットモデル変換 (戦略 1)

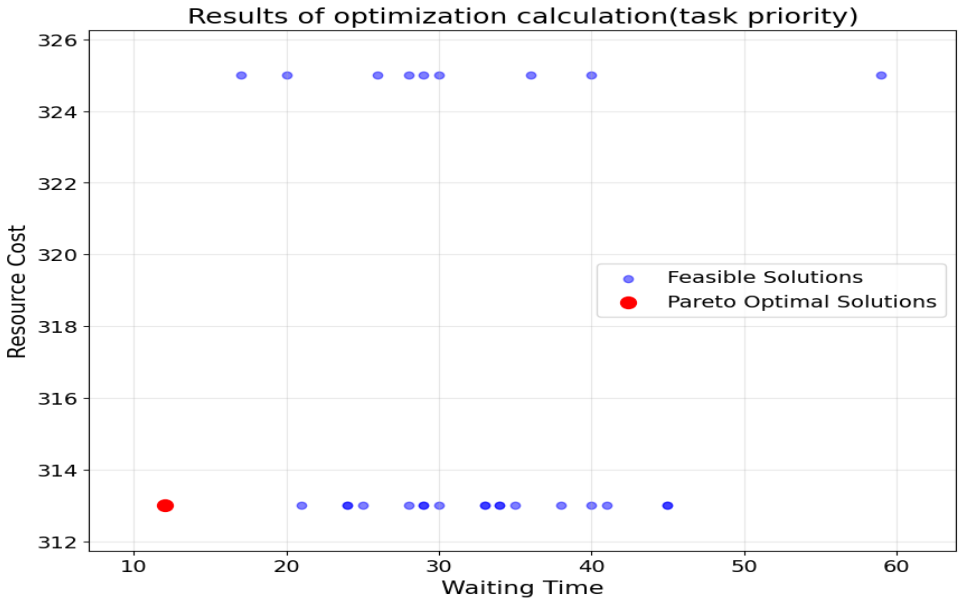

図 4.2: タスクの処理時間を最適化したパレート解

表 4.1: 各巡回における実行可能解と個数および平均値 (戦略 1)

巡回	個数	Resource Cost の平均値	Waiting Time の平均値
第 1 巡	7	313	46.6
第 2 巡	4	313	32
第 3 巡	6	313	36.5
第 4 巡	1	313	20
第 5 巡	4	325	33.1
全体の平均	4.4	315.4	33.6

表 4.1 は表 3.3 と同様に実行可能解の解の分布と解が得られた時のデータを示している．また，図 4.2 はタスク処理時間を最適化した後の最適化計算を行い，得られたパレート解を可視化したグラフになっている．第一段階の最適化において，タスクに使用するマシンリソースの分割を行い，多目的から単目的に変換されるため Resource Cost が確定される．この段階で分割されたリソースが固定され，第二段階の最適化では Resource Cost が一定の値を保ったまま，Waiting Time の短縮に向けた探索が行われるため解空間上で解が横方向に広がる結果が得られる．ただし，リソースの分割も含めて再計算を行うため全ての解が完全に同一の Resource Cost を持つわけではなく，一部の解において Resource Cost の変動が見られる．

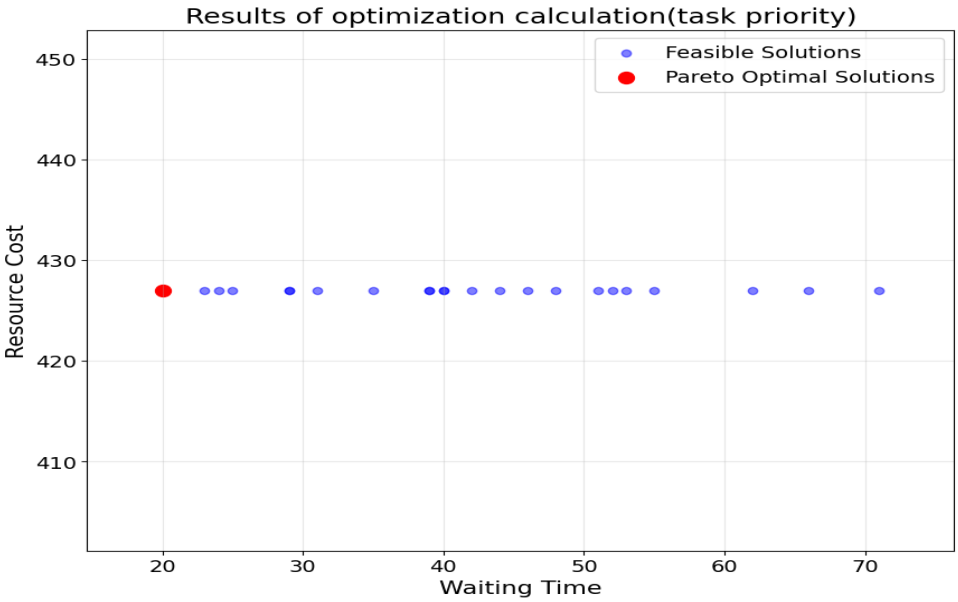

図 4.3: 戦略 1 の計算できる最大規模数 (job 数が 7) のパレート解

表 4.2: 戦略 1 の最大規模数の実行可能解と個数および平均値

巡回	個数	Resource Cost の平均値	Waiting Time の平均値
第 1 巡	3	427	33
第 2 巡	4	427	39.8
第 3 巡	8	427	44
第 4 巡	4	427	50
第 5 巡	4	427	38.5
全体の平均	4.6	427	41.1

表 4.2 と図 4.3 は戦略 1 の計算できる最大規模数 (job 数が 7) の時の結果になっている．job 数が 5 の時と比較して実行可能解が増加している結果は，提案手法が拡大した探索空間を効率的に探索していることを示唆している．モデルの変換前と同様に規模の拡大に伴い目的関数の値の増加が見られる．

4.2.2 リソースコストを最適化したペトリネットモデル変換 (戦略 2)

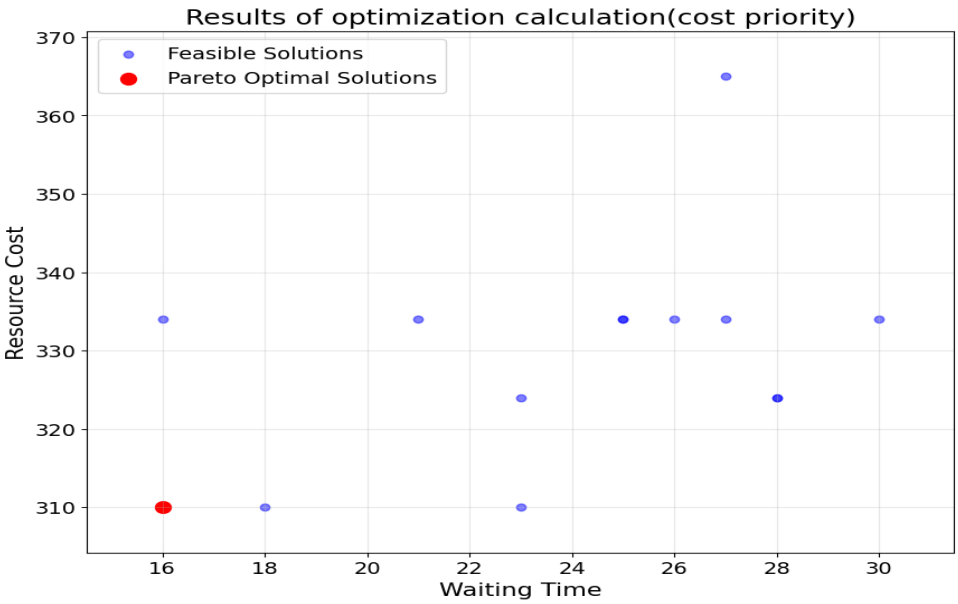

図 4.4: リソースコストを最適化したパレート解

表 4.3: 各巡回における実行可能解と個数および平均値

巡回	個数	Resource Cost の平均値	Waiting Time の平均値
第 1 巡	1	365	27
第 2 巡	3	310	19
第 3 巡	3	324	26.3
第 4 巡	0	-	-
第 5 巡	7	334	24.3
全体の平均	3.5	333.3	24.2

表 4.3 は表 4.1 と同様に実行可能解の解の分布と解が得られた時のデータを示している．また，図 4.5 はリソースコストの最適化を初段階として行い，解の分布が戦略 1 で得られた分布と類似していることがわかる．

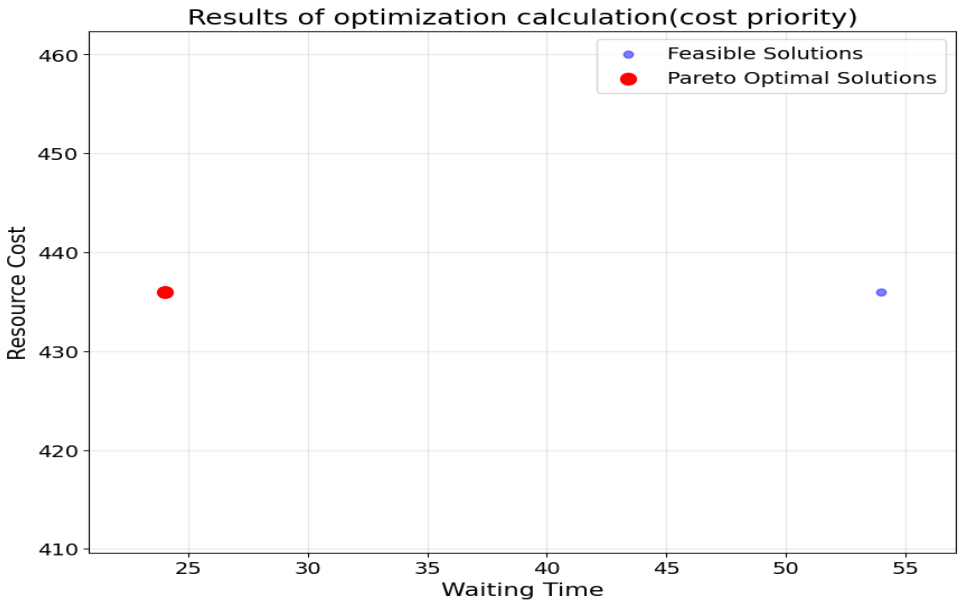

図 4.5: 戦略 2 の計算できる最大規模数 (job 数が 7) のパレート解

表 4.4: 戦略 2 の最大規模数の実行可能解と個数および平均値

巡回	個数	Resource Cost の平均値	Waiting Time の平均値
第 1 巡	0	-	-
第 2 巡	0	-	-
第 3 巡	0	-	-
第 4 巡	0	-	-
第 5 巡	2	436	39
全体の平均	2	436	39

表 4.4 と図 4.5 は戦略 2 の計算できる最大規模数 (job 数が 7) の時の結果になっている．job 数が 5 の時と比較すると実行可能解が減少しているが，戦略 1 同様規模を拡大しても解を探索することができることより効率的に探索していることを示唆している．

4.2.3　ペトリネット変換前と戦略 1 の比較

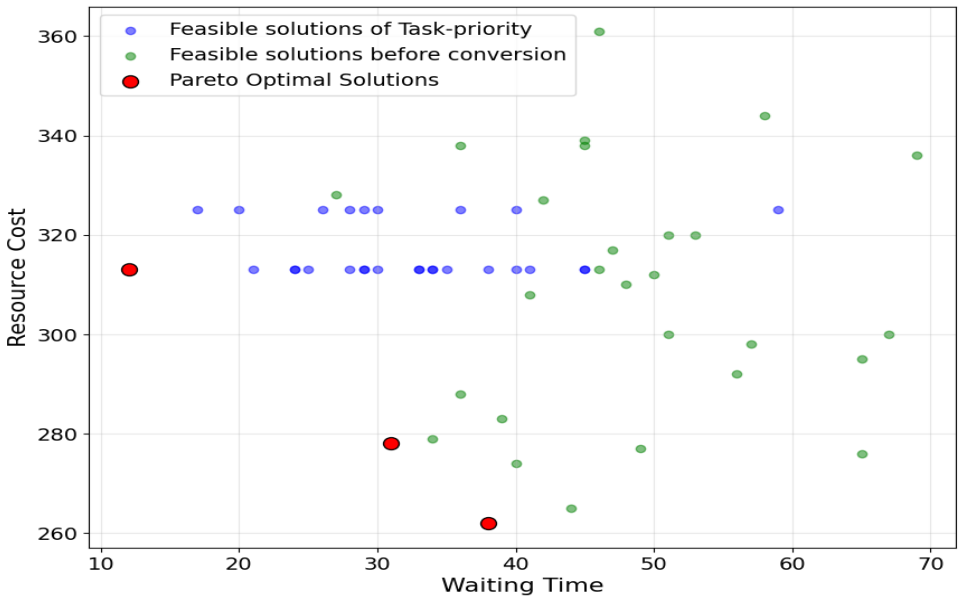

図 4.6: ペトリネット変換前と戦略 1 のパレート解

図 4.6 は，変換を行わない手法と戦略 1 に基づく最適化計算の結果を同時にプロットしたものである．青の点が戦略 1 で緑の点が変換前である．この図に示された 3 つのパレート解のうち，1 つは戦略 1 により得られた解であり，残りの 2 つは変換を行わない手法による解である．このことから，戦略 1 においても効率的な解を見つけることが可能であることが示された．また，表 3.3 と表 4.1 を比較すると，戦略 1 では Resource Cost の値が比較的安定している一方で，変換を行わない手法の結果と比較すると若干劣る傾向が見られる．しかしながら，Waiting Time に関しては，戦略 1 の結果において変換を行わない手法の結果と比較して明らかに減少している傾向が確認された．これらの結果は，戦略 1 が Waiting Time の最適化に寄与しつつ，Resource Cost の安定性を維持できる手法であることを示唆している．

4.2.4　ペトリネット変換前と戦略 2 の比較

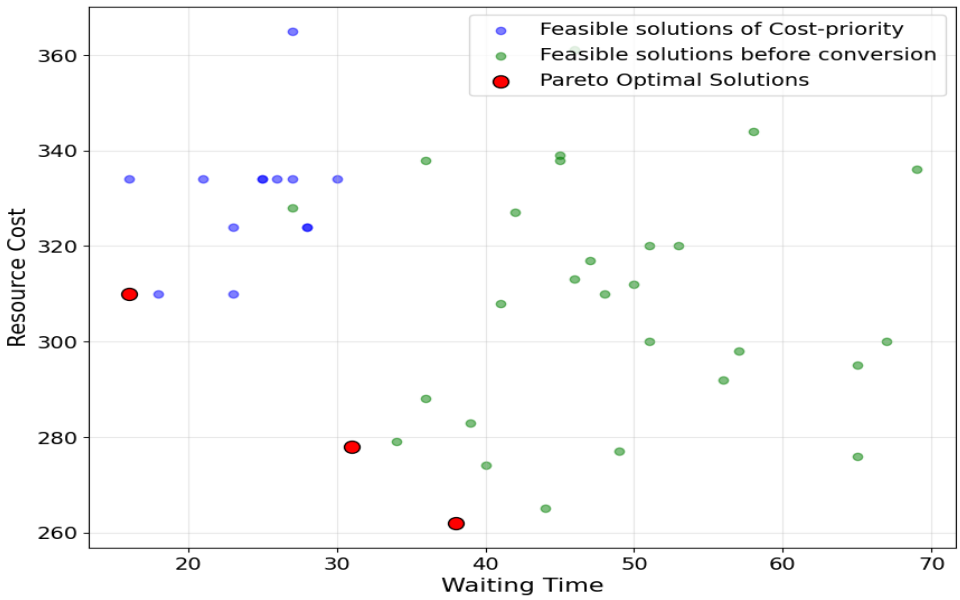

図 4.7: ペトリネット変換前と戦略 2 のパレート解

図 4.7 は，変換を行わない手法と戦略 1 に基づく最適化計算の結果を同時にプロットしたものである．戦略 1 と同様な結果が得られた．このことから，戦略 2 においても効率的な解を見つけることが可能であることが示された．また，表 3.3 と表 4.3 を比較すると，戦略 2 は Waiting Time の短縮において顕著な効果を示していることが明らかとなった．一方で，Resource Cost が増加し，実行可能解の個数が減少している点は，課題として考慮する必要がある．

4.3　考察

変換手法の戦略 1 および戦略 2 において観察された実行可能解の減少は，モデル変換によって導入された付加的な制約条件に起因するものと考えられる．この実行可能解の減少が一見すると解の多様性の損失として捉えられる可能性があるが，むしろ質的な観点からは有意義な減少である可能性があると示唆される．具体的には，戦略 1 および戦略 2 の適用により，システムにおける Waiting Time が統計的に有意な減少を示したことが確認されている．このことから，スケジューリング戦略が制約条件を厳格化することで低品質な解を排除し，より効率的で質の高い解を優先するように探索空間を調整したことを示唆している．一方で，実行可能解の減少に伴う探索空間の縮小が Resource Cost の増加と関連している可能性も考えられる．制約条件の厳格化により，低コストな解が探索区間から排除された結果 Resource Cost が増加したと推測される．

さらに，本研究ではモデル変換を適用することで問題の規模が大きくなった場合でも計算を実行可能であることが示された．この結果は，変数数の削減や探索効率の向上を通じ

てモデル変換がスケーラビリティの向上に寄与していることを示唆していると思われる．また，モデル変換後においてリソースコストを優先した場合よりもタスクの処理時間を優先した場合の方が実行可能解の個数が多かった理由として，マシンリソースの性能とコストのばらつきが影響していると考えられる．具体的には，タスクの処理時間を優先する場合，性能の高いマシンが優先的に活用されることでスケジュールが柔軟に調整され多くの実行可能解が得られたと推測される．一方で，リソースコストを優先する場合，探索空間に厳格な制約が課されその結果，より効率的なスケジューリングが可能な良質な解が排除されてしまい実行可能解の個数が減少したと考えられる。

また，本研究で提案された戦略 1 および戦略 2 は，特に Waiting Time の最小化が重要視される運用システムにおいて有力なスケジューリング戦略となり得る．このことは，システムの要求事項や運用目的に応じた柔軟で適応的なアプローチの重要性を強調している．一方で，Resource Cost の増加が一定の課題として残るため，より広い解空間を確保する探索手法の導入や，制約条件の緩和を検討する余地がある．

第5章　結論

5.1　まとめ

本研究では，複数のリソースを対象としたスケジューリング問題を組合せ最適化問題としてQUBOモデルで定式化し，量子アニーリングを用いて最適なスケジューリングを探索する実験を行った．また，ベイズ最適化を活用することで，制約条件のパラメータおよび目的関数のパラメータを自動で調整し，多様な解を効率的に得ることが可能であることを示した．

さらに，効率的な解の探索を目的として，最適化計算を二段階に分けペトリネットモデルへの変換を行う手法を提案した．このモデル変換により，問題の変数数を削減し，良質な解を効率的に探索することが可能となった．特定の目的を優先する場合においては，変換前よりも効率的な解を求める可能性を示唆する結果が得られた。

5.2　今後の展望

本研究では，モデル変換を多目的から単目的に変換する手法を採用した．しかし，この変換により特定の目的関数の値が固定されることで解空間が過度に狭まり，探索の多様性が制限された可能性があると考えられる．今後は，多目的性を維持したままモデル変換を行う手法を検討することで解の質のさらなる向上が期待される．

また，解の探索方法としてリバース量子アニーリングを導入することにより実用的で高品質な解を得られる可能性がある．この手法の採用により，探索精度の向上と計算効率の改善が期待され，現実のスケジューリング問題への適用性がさらに高まると考えられる。

謝辞

本研究の遂行にあたり，多くの方々にご指導を賜りました．指導教官である名嘉村盛和教授には，御多忙の中，研究の方向性の決定か結果の考察に至るまで貴重な助言をいただきました．深く感謝いたします．また，合同ゼミで貴重な助言をしていただいた仲地孝之教授，城間政司助教にも深く感謝いたします．おかげで国際会議で研究成果を発表することができました．共に研究活動を頑張った仲地研と城間研の同期に感謝いたします．最後に，物心両面で支えてくれた家族に深く感謝いたします．

2025 年 1 月
上地 涼太

参考文献

[1] Bingzhi Zhang, Akira Sone, and Quntao Zhuang. Quantum computational phase transition in combinatorial problems. *npj Quantum Information*, Vol. 8, No. 1, pp. 87, 2022.

[2] Tadashi Kadowaki and Hidetoshi Nishimori. Quantum anneaing in the transverse ising model. *Phys. Rev. E*, Vol. 58, pp. 5355-5363, Nov 1998.

[3] T.Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, Vol. 77, No. 4, pp. 541-580, 1989.

[4] Kurt Jenson, Lars Michael Kristensen, and Lisa Wells. Coloured Petri Nets and CPN Tools for modeling and validation of concurrent systems. *International Journal on Software Tools for Technology Transfer*, Vol. 9, No. 3, pp. 213-254, 2007.

[5] Houssem Eddine Nouri, Olfa Belkahla Driss, and Khaled Ghedira. Solving the flexible job shop problem by hybrid metahcuristics-based multiagent model. *Journal on Software Tools fot Technology Transfar*, Vol. 9, No. 3, pp. 213-254, 2007.

[6] 大阪大学医学部 Python 会. Parameter Tuning - 大阪大学医学部 Python 会. `https://oumpy.github.io/blog/2018/09/parameter_tuning.html`.

[7] Bergstra, James, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. *International conference on machine learning*, pp. 115-123, 2013.

[8] 新城巧也，名嘉村盛和，猪谷宜彦. 信学技報, 段替え作業を考慮したフローショップスケジューリング問題のペトリネットモデリングと QUBO 定式化. vol. 122, no. 78, MSS2022-17, pp. 90-95, 2022 年 6 月

[9] OpenJij.OpenJij - Optimization Library. `https://www.openjij.org`.