シフトスケジューリング問題における量子アニーリングマシンの初期評価

浜田　捺希[1,a)]　斎藤　和広[2,1,b)]　川島　英之[3,c)]

概要：シフトスケジューリング問題はコールセンターのアルバイトなどのシフトの組み合わせ最適化問題である．この問題はナーススケジューリング問題などの必要な人員数が定められ，複数の制約の元で人員を割り当てなければならないスケジューリングを行う現実問題に応用することが可能な問題である．この問題は考慮する人数や期間が増えるほど組合せ数が膨大となり，厳密解を求めることが困難となる．量子アニーリングは量子コンピュータの一種として，このような組み合わせ最適化問題を現実時間で解くことが期待されている．本論文では目的関数や制約式を量子アニーリングに適用できる QUBO(Quadratic Unconstrained Binary Optimization) で定式化し，D-wave 社の量子アニーリングマシンを用いてシフトスケジューリングにおける量子アニーリング適用の効果を評価た．更に，量子アニーリングシミュレータである OpenJij と既存手法として制約充足問題のソルバーである The Z3 theorem prover と比較し，量子アニーリングマシンの有効性を検証した．

Initial Evaluation of a Quantum Annealing Machine for the Shift Scheduling Problem

1. はじめに

シフトスケジューリング問題 [1] はコールセンターのアルバイトなどのシフトの組み合わせを最適化する組み合わせ最適化問題である．応用例としてナース・スケジューリング問題などがあり，自動化にむけて様々な研究が行われてきた [2][3][4]．現状として，シフトスケジューリングは手作業で行われていることが多く，パートの希望シフトと照らし合わせた上で，時間帯毎に必要な人員数など，様々な条件を加味してシフトを組む．これらの制約を加味したシフトスケジューリング作業は，手作業で行われていることが多く，時間と労力を要する作業である．従って，シフトスケジューリング問題を自動化できることは重要である．

[1] 慶應義塾大学政策メディア研究科
Keio University,5322,Endo,Fujisawa,Kanagawa252-0882,Japan
[2] 株式会社　KDDI 総合研究所
KDDI Research, Inc.,Fujimino,Saitama356-8502,Japan
[3] 慶應義塾大学環境情報学部
Keio University,5322,Endo,Fujisawa,Kanagawa252-0882,Japan
[a)] hnatsu@sfc.keio.ac.jp
[b)] ku-saitou@kddi-research.jp
[c)] river@sfc.keio.ac.jp

シフトスケジューリング問題はシフト条件を制約とする組み合わせ最適化問題として定式化が可能である．この問題は NP 完全 [5] に分類され，最適解を出すことが困難であり，古典的なコンピュータでは多項式時間で問題を解くことが難しい問題とされている．

近年多項式時間で計算が困難な組み合わせ最適化問題の解決に量子アニーリング [6] の活用が期待されている．量子アニーリングは，二つの量子ビット σ_i, σ_j の相互作用係数 J_{ij}，及び量子ビット σ_i に影響する外部磁場エネルギー係数 h_i から，ハミルトニアン $\mathscr{H}$ を最小化する量子ビット σ_i のスピン方向 $(+1/-1)$ を決定する．この仕組みはイジングモデルとして以下のように表現できる．

$$\mathscr{H} = \sum_{i,j} J_{ij}\sigma_i\sigma_j + \sum_i h_i\sigma_i \tag{1}$$

組合せ最適化問題を目的関数の最小化問題としてイジングモデルに定式化することで，量子アニーリングによって解くことが可能となる．実問題を解く際には定式化を直感的に行うことが求められるため，以下のように量子ビット σ_i をバイナリ変数 $x_i \in \{0, 1\}$ に変換する．

$$x_i = \frac{\sigma_i + 1}{2} \tag{2}$$

量子アニーリング利用時は，これを数式 (1) のイジングモデルに適用して以下のような QUBO (Quadratic Unconstrained Binary Optimization）問題とする [7][8][9][10].

$$\mathscr{H} = \sum_{i,j} Q_{ij} x_i x_j \tag{3}$$

したがってシフトスケジューリング問題の目的関数, 制約を定式化し，数式 (3) の QUBO 式に適用することによってシフトスケジューリング問題を量子アニーリングに適用することが可能となる.

そこで本研究ではコールセンターのシフトスケジューリングを想定したシフトスケジューリング問題の定式化を行い，QUBO 式を構築して量子アニーリングにおける性能を評価した．評価には，D-wave systems 社 [11] の量子アニーリングマシンと量子アニーリングのシミュレータとしてソフトウェア実装された OpenJij[12] を使用し，既存手法と比較する.

本論文の構成は以下の通りである．2 節においてシフトスケジューリング問題の問題設定と，量子アニーリングに適用するための定式化と QUBO 式における定式化を記述し，3 節において D-wave systems 社の量子アニーリングマシンを用いて各種パラメータをチューニングし，評価した後，量子アニーリングシミュレータと既存研究と比較する．4 節で議論し，5 節で結論を述べる.

2. シフトスケジューリング問題の量子アニーリング適用

2.1 問題設定

本論文におけるシフトスケジューリング問題は，コールセンターのシフトスケジューリングを想定する．コールセンターにはブースが存在し，そのブースに作業員を割り当てる．日によって必要なブース数が定められており，そのブース数に近づくようにそれぞれのタームに人を割り振る.

目的関数は 2 つ存在し，P_1, P_2 とし，制約条件も 2 つで C_1, C_2 とする．P_1 は，一日のシフトを数ターム（朝番，昼番，夜番など）に分け，そのそれぞれのタームにおいて割り振りたい人数を設定する．その人数と実際に割り振った人数との差を最小化することである。P_2 は，あらかじめ指定された日数の中で作業員の希望ターム数を定め，それぞれの作業員の割り振られたターム数と希望ターム数との差を最小化することである．これにより，ある特定の作業員にシフトが集中し，他の作業員がシフトに入ることができなくなる状態を回避できる.

C_1 は作業員それぞれが具体的にどのタームに入りたいのかという希望シフトを提出してもらい，希望シフトに沿うように割り振ることである．C_2 は一緒にシフトを組んで欲しい作業員をグループにまとめ，そのグループに属している作業員を全員同じシフトに割り振ることである.

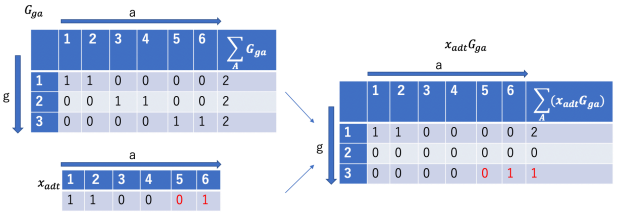


図 1: (6) 式の制約項の一例
Fig. 1 An example of a constraint of (6)

2.2 最適化問題の定式化

定式化の前に、それに用いる記号を**表 1** で説明する。

2.1 節で説明したように本研究での目的関数は二つ存在する.

$$P_1 = \sum_d \sum_t (\sum_a |x_{adt} - S_{dt}|) \tag{4}$$

(4) 式は d 日目におけるターム t に割り振った x_{adt} を合計し，もともと設定されていた希望ブース数 S_{dt} から引いた値の絶対値の総和である．この式を最小に近づけることで，それぞれのターム毎に設定された必要な作業員数に近づくように作業員を割り振ることができる.

$$P_2 = \sum_a (\sum_d \sum_t |x_{adt} - R_a|) \tag{5}$$

(5) 式は作業員 a 毎に割り振られた x_{adt} を合計し, 設定されたそれぞれの作業員の R_a から引いた差の絶対値の総和である．R_a は全作業員の希望シフトのターム合計数であるので，この総和を最小化することで作業員の希望の通りにシフトを組むことが可能になる．またシフトが特定の作業員に偏ることなく割り振ることができるようになる.

次に制約式 C_1, C_2 を定式化する.

$$\forall\ a \in A, \forall d \in D, \forall t \in T$$
$$x_{adt} = 0, if\ r_{adt} = 0 \tag{6}$$

(6) 式は制約条件 C_1 であり，作業員が希望していないターム ($r_{adt} = 0$) ではシフトを割り振らない制約を表す.

$$\forall a \in A, \forall d \in D, \forall g$$
$$\sum_a x_{adt} G_{ga} = (0\ or\ \sum_a G_{ga}) \tag{7}$$

(7) 式は制約条件 C_2 であり，同じグループに割り振られた作業員全員が同じシフトに割り振られるようにするもので，その一例を**図 1** で示す．図 1 の G_{ga} は作業員 1 と 2 がグループ 1 に所属し，作業員 3 と 4 はグループ 2 に，作業員 5 と 6 はグループ 3 に所属していることを表している．この条件下で作業日 d とターム t を固定してシフトを割り振った結果を x_{adt} で示している．$x_{adt}G_{ga}$ はそれぞれの a,g で掛けたものになっている．本制約では同じグループに所属する作業員は全員同じシフトにならなくてはならな

表 1: 定式化で用いる記号
Table 1 Symbols used in formulations

記号		説明
A	$=\{$ 作業員 $1,$作業員 $2,\cdots,$作業員 $a\}$	作業員の集合
D	$=\{1$ 日$,2$ 日$,\cdots,d$ 日 $\}$	スケジュール対象日の集合
T	$=\{$ ターム $1,$ターム $2,\cdots,$ターム $t\}$	1 日のターム（午前，午後，夕方など）の集合
S_{dt}	$=\{s\|s$ は d 日のターム t 毎に必要なブース数 $\}$	
r_{adt}	$=\{0,1\}$	作業員 a が d 日のターム t へ割り当て可能なら 1，不可なら 0
R_a	$=\{r\|r$ は作業員 a の希望割り当てシフト数 $\}$	$\sum_A R_a \geq \sum_{D,T} S_{dt}$ となるように設定する
G_{ga}	$=\{0,1\}$	グループ g に作業員 a が所属していれば 1，いなければ 0
x_{adt}	$=\{0,1\}$	問題を解いた際に作業員 a が d 日のターム t に割り振られたら 1，割り振られなければ 0

い．従ってグループ 3 に所属している作業員 5 と 6 は制約違反している．グループ 1 は $\sum_a(x_{adt}G_{ga}) = \sum_a G_{ga} = 2$ で (7) 式を満たしている．グループ 2 は $\sum_a(x_{adt}G_{ga}) = 0$ で (7) 式を満たしている．しかし制約違反しているグループ 3 では $\sum_a(x_{adt}G_{ga}) \neq \sum_a G_{ga}$ で (7) 式を満たしていない．

2.3 QUBO 式における定式化

量子アニーリングに適用するために QUBO 式を構築する．2.2 節の 4,5,6,7 式からハミルトニアン $\mathscr{H}$ を (8) 式に定義する．

$$\mathscr{H} = \lambda_i\mathscr{H}_i + \lambda_j\mathscr{H}_j + w_k\mathscr{H}_k + w_l\mathscr{H}_l \tag{8}$$

$$\mathscr{H}_i = \sum_d\sum_t\{(\sum_a x_{adt}) - S_{dt}\}^2 \tag{9}$$

$$\mathscr{H}_j = \sum_a\{(\sum_d\sum_t x_{adt}) - R_a\}^2 \tag{10}$$

$$\mathscr{H}_k = \sum_a\sum_d\sum_t\{(1 - r_{adt})x_{adt}\} \tag{11}$$

$$\mathscr{H}_l = \sum_d\sum_t\sum_g[\{(\sum_a G_{ga}) - (\sum_a x_{adt}G_{ga})\} \cdot(\sum_a x_{adt}G_{ga})] \tag{12}$$

(8) 式の $\mathscr{H}$ が全体として最小となるよう，アニーリングシミュレータが量子アニーリングで最適解に近い値になるよう演算を行う．(8) 式の λ_i, λ_j は目的関数 $\mathscr{H}_i$，$\mathscr{H}_j$ のそれぞれの係数である．$\mathscr{H}_i$ の値が小さい場合は各タームで割り振られた作業員数が必要ブース数に近いことを表し，$\mathscr{H}_j$ の値が小さい場合は各作業員の割り振りがそれぞれの合計希望シフト数に近いことを表している．係数 λ_i，λ_j の値を他方と差をつけて設定することで求解における目的関数の優先度を調整できる．例えば $\lambda_i = 1, \lambda_j = 2$ と設定した場合，$\mathscr{H}$ 全体を最小化するよう量子アニーリングが実行されるため $\mathscr{H}_i$ に比べて $\mathscr{H}_j$ の値が小さくなるよう解を得られやすくなる．

同様に W_k, W_l は制約項 $\mathscr{H}_k, \mathscr{H}_l$ それぞれのペナルティ係数となっている．$\mathscr{H}_k, \mathscr{H}_l$ は制約項となっており，それぞれの制約を満たしていれば値は 0 になり，制約違反をすれば値が正の値を取るように定式化されている．よって目的関数項の値を十分超える大きさの W_k, W_l の値を設定することで制約違反が起こらないように量子アニーリングで解を導き出すことができる．

(9) 式は (4) 式を QUBO 式に適用したものである． d 日目におけるターム t に割り振った x_{adt} を合計し，もともと設定されていた希望ブース数 S_{dt} から引いた値の二乗の総和となる．二乗によりそれぞれの差を正の値として受け取ることができる．

(10) は (5) を QUBO 式に適用したものである．作業員 a 毎に割り振られた x_{adt} を合計し，設定されたそれぞれの作業員の R_a から引いた差の二乗の総和となる．(9) 式と同様に二乗することでそれぞれの差を正の値として受け取っている．

(11) は (6) の制約条件を QUBO 式として定義した式である．作業員が希望していないターム ($r_{adt} = 0$) でシフトを割り振る ($x_{adt} = 1$) と制約違反となり，値が $(1-r_{adt})x_{adt} = 1$ となり制約項が正の値をとる．それ以外の場合全てにおいて $(1 - r_{adt})x_{adt} = 0$ となるように定義している．

(12) 式は (7) 式の制約条件を QUBO 式として定義した式であり，図 1 はその一例である．$\sum_a G_{ga} \geq \sum_a(x_{adt}G_{ga})$ であるため，この制約式が負の値を取ることはなく，制約違反した場合は必ず正の値を取る．

3. 評価

3.1 ワークロード

本節ではハミルトニアン $\mathscr{H}$ を energy とし，W_k, W_l はそれぞれ w_desire,w_group と表すこととする．本評価では，制約を限定した基礎的なシフトスケジューリング問題として，2 節で述べた問題設定のシフトスケジューリング問題に量子アニーリングを適用した際の実行時間と energy，実行可能解の関係を評価する．また，評価の前に，量子アニーリングで使用されるペナルティ係数をチューニングする．これにより評価において制約違反の解の数と実行時間

表 2: 評価で利用するデータと変数の個数
Table 2 The number of variable and data in evaluation

作業員 a	日数 d	ターム数 t	変数の個数
5	4	3	60
6	5	3	90
6	7	3	126

を大幅に削減する．本評価では量子アニーリングマシンで実行可能な規模のデータを用意した．ターム数は固定し，作業員数と日数を増やすことで変数の個数を変化させた．**表 2** が本評価で利用したデータとそれぞれの変数の個数である．

3.2 評価環境

本研究では D-wave systems 社がクラウドサービスで提供している量子アニーリングマシンをハードウェアで実装した Advantage system1.1 を利用した．このアニーリングマシンの各種パラメータ値の決定過程は 3.3 節で述べる．

また量子アニーリングシュミレータとして OpenJij を利用した．OpenJij は，オープンソースソフトウェアとして，量子アニーリングをシミュレートする Simulated Quantum Annealing (SQA) の Python 用 API を提供している．本評価ではバージョン 0.1.1 の SQA を利用した．SQA の設定は,beta=10，gamma=1.0，trotter=10，num reads=100 とした．

既存手法として用いたのは Microsoft 社がオープンソースソフトウェアとして提供する SMT(Satisfiability Modulo Theories) ソルバーの The Z3 Thorem Prover[13] の Python 用 API[14] を利用した．

OpenJij と The Z3 Thorem Prover を使用する環境は Amazon Web Service の Sage Maker notebook を用いた．サーバーのインスタンスは ml.t3.medium で vCPU は 2 つ，メモリは 4GiB で，Python 3.6.13 バージョンを使用した．

3.3 パラメータ評価

ここでは，量子アニーリングマシンのシフトスケジューリング問題におけるエネルギーを低くするために D-wave systems 社が提供する量子アニーリングマシンの各種パラメータと (8) 式の W_k, W_l を調整する．本研究では $\lambda_i = 1, \lambda_j = 1$ とする．またパラメータ調整時のアニーリング処理回数 (num_reads) は全て 1,000 に設定し，Advantage における量子ビットの結合関係の実装（ペガサスグラフ）に全結合の QUBO を埋め込む際に発生するパラメータ (chain_strength) は自動に設定してある．

3.3.1 ペナルティ係数の比率決定

まずは (8) 式の W_k, W_l の比率を決定する．比率を変えることにより，制約項 (11)(12) 式がどちらかの式の値だけが大きくなることがなく，バランスを取るようにチューニングした．

表 3: w_desire と w_group の比率と base 値
Table 3 The ratio of w_desire and w_group and value of base

変数の個数	w_desire の比率	w_group の比率	base
60	15	24	0.5
90	27	12	0.6
126	55	26	0.3

表 4: チューニング後の各パラメータの値
Table 4 Each value of parameter after tuning

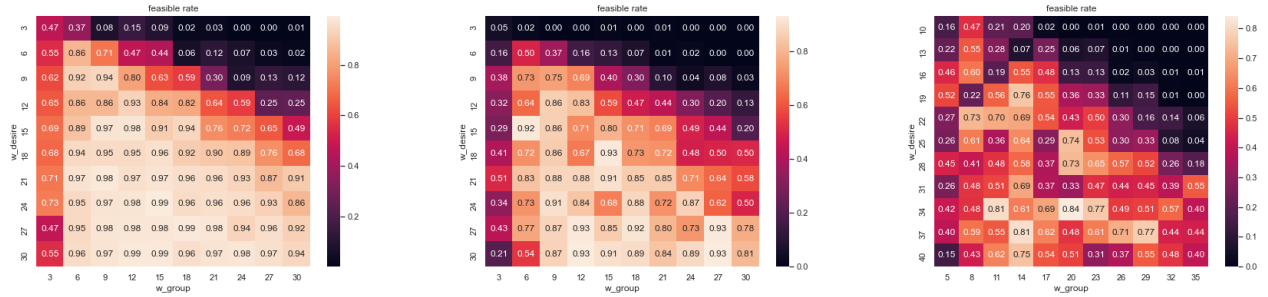
変数の個数	w_desire	w_group	chain_strength
60	7.5	12	4.4
90	16.2	7.2	3.6
126	16.5	7.8	3.8

図 2 はそれぞれの個数の変数の時に得られた実行可能解の確率のヒートマップ である．ヒートマップを見ることで実行可能回の確率の均衡の取れている値を選択する．どの変数の個数の時にも共通して片方の値が大きく，片方の値が小さくなると実行可能解を得られる確率が下がっていることが分かる．特に w_group の値を大きくしすぎると実行可能解を得られる確率が低くなっている．**表 3** はヒートマップ からそれぞれの変数の個数における w_desire と w_group の比率を決めたものである．

3.3.2 ペナルティ係数に掛ける値の決定

次に 3.3.1 節で決定したペナルティ係数の比率に掛け合わせる値 base を決定し，ペナルティ係数の値を決定する．制約違反を全く起こさず，かつ energy が低くなる base 値を設定することが目的である．ここでは，base 値を変化させ，実行可能解の確率と energy を計測した．

図 3 は右側の縦軸が base 値を変化させた時に実行可能解を得られる確率を折れ線グラフで計測したものであり，左側の縦軸はそれぞれの base 値において、得られた num_reads 分（1000 個分）の解におけるエネルギーの最小値と最大値をエラーバーの下限上限で，平均値を折れ線グラフで表したものである．base 値が小さい時はエネルギーは低い値を安定して取るが，実行可能解を得られる確率も低くなってしまう．逆に base 値を大きくすると実行可能解を得られる確率は高くなるが，エネルギーは大きくなってしまうため，トレードオフの関係の関係にあると言える．そこで base 値は実行可能解を得られる確率が緩やかに収束し始める値のなるべく小さい値に決定した．表 3 でそれぞれの個数における base 値を載せた．そして 3.3.1 節で決定したそれぞれのペナルティ係数の比率と base 値を掛けて，チューニングにより決定したペナルティ係数の値を**表 4** に示した．

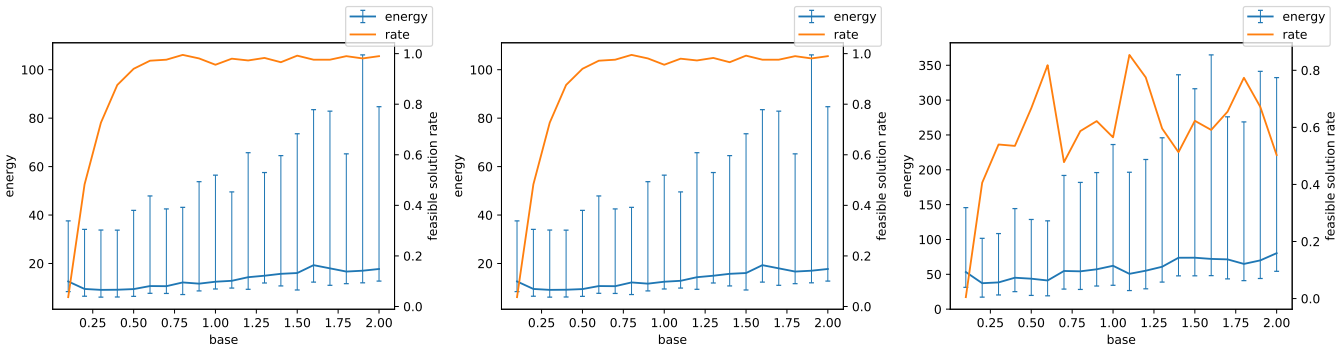


(a) 変数 60 個
20 variables

(b) 変数 90 個
90 variables

(c) 変数 126 個
20 variables

図 2: パラメータを変化させた時のヒートマップ
Fig. 2 A heatmap as each parameter changes



(a) 変数 60 個
20 variables

(b) 変数 90 個
90 variables

(c) 変数 126 個
20 variables

図 3: base 値を変化させた時の実行可能解とエネルギー
Fig. 3 Feasible solution rate and energy as value of base changes

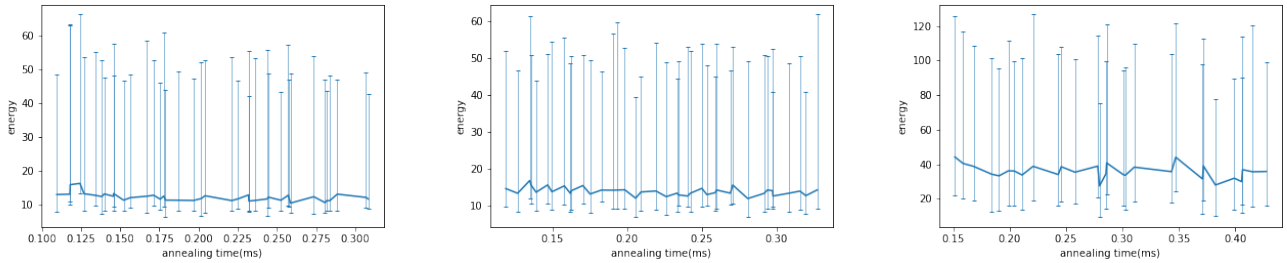
3.4 性能評価

イジングマシンにおけるシフトスケジューリング問題の QUBO 式を評価するため，表 4 のパラメータを用いて実行した D-wave systems 社の量子アニーリングマシンの実行結果と，OpenJij の量子アニーリングシミュレータと既存手法を比較する．

図 4 の縦軸はそれぞれの変数の個数の時に 1 回の読み取り毎のアニーリング時間 (annealing_time) を 10 μ s〜200 μ s の範囲で 5 μ s ごとに変化させてそれぞれの annealing_time 値において、得られた num_reads 分（1000 個分）の解におけるエネルギーの最小値と最大値をエラーバーの下限上限で表した．横軸は 1 回の量子アニーリングマシンが解を出すまでの時間 (sampling time) にしたものである．annealing_time を長くすればそれだけ安定して解を得られやすくなる．しかし変数が 60 個の時と変数が 90 個の時は 0.1ms 程度で平均値が最小値とほぼ近く，安定してエネルギーの低い解を得られていることが分かる．変数が 126 個の時も 0.18ms 付近で安定して低いエネルギーを得られている．

図 5 はハードウェアで量子アニーリングマシンである D-wave advantage 1.1 と古典コンピュータのソフトウェアで実装された量子アニーリングシミュレータの OpenJij を比較し，さらにシフトスケジューリング問題を QUBO 式を基に制約充足問題として定義し，制約充足問題のソルバーの一つの SMT(Satisfiability Modulo Theories) ソルバーのソフトウェア，The Z3 Thorem Prover との解と比較したものである．赤色のエラーバーが D-wave advantage 1.1 , 青色が OpenJij , 緑色が The Z3 Thorem Prover である．また図 5 の D-wave の実行結果は図 4 の実行結果と全く同じものを使用している．

実行時間の観点では，変数が 60 個の場合で低いエネルギーの解を得られ始めるのは D-wave は 0.1ms 近辺，OpenJij は 5ms 近辺である．Z3 は 9ms で解を得られている．よって D-wave は OpenJij の 50 倍，Z3 の 90 倍高速化している．変数が 90 個の場合で低いエネルギーの解を得られ始めるのは D-wave は 0.1ms 近辺，OpenJij は 6.5ms 近辺，Z3 は 10.5ms で解を得られている．よって D-wave は OpenJij の 65 倍，Z3 の 105 倍高速化している．変数が 126 個の場合で低いエネルギーの解を得られ始めるのは D-wave は 0.18ms 近辺，OpenJij は 10ms 近辺，Z3 は 12ms で解を

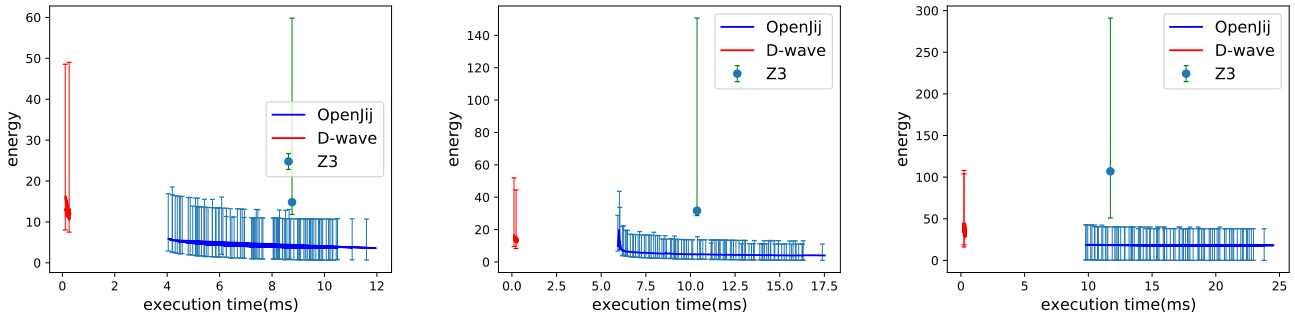


(a) 変数 60 個
20 variables

(b) 変数 90 個
90 variables

(c) 変数 126 個
20 variables

図 4: annealing_time を変化させた時のエネルギーの変化
Fig. 4 Energy as annealing_time changes



(a) 変数 60 個
20 variables

(b) 変数 90 個
90 variables

(c) 変数 126 個
20 variables

図 5: 量子アニーリングと既存手法の比較
Fig. 5 Compare quantum annealing to existing method

得られている．よって D-wave は OpenJij の 56 倍，Z3 の 67 倍高速化している．よってハードウェアで実装した量子アニーリングマシンは他 2 つの手法に比べて低いエネルギーの解を高速で得ることができる．

エネルギーの観点では，変数が 60 個の際のエネルギーの範囲は D-wave は 8〜60，OpenJij は低いエネルギーが安定して得られている部分では 2〜10，Z3 は 10〜60 であり，OpenJij が得られるエネルギーの範囲もその値も小さく，D-wave と Z3 はほぼ同じエネルギーの範囲でその値も等しい．変数が 90 個の際のエネルギーの範囲は D-wave は 8〜60，OpenJij は低いエネルギーが安定して得られている部分では 2〜20，Z3 は 25〜150 であり，OpenJij が得られるエネルギーの範囲もその値 3 つの中で一番小さく，次いで D-wave，そして Z3 は他二つの範囲とその値よりも大きい．変数が 126 個の際のエネルギーの範囲は D-wave は 8〜120，OpenJij は 8〜50，Z3 は 55〜290 であり，変数 90 個の時と同様に OpenJij が得られるエネルギーの範囲もその値 3 つの中で一番小さく，次いで D-wave，そして Z3 の順番になる．よって変数が大きくなるにつれて量子アニーリングと既存手法ではエネルギーの範囲で差が広がることが分かる．

4. 議論

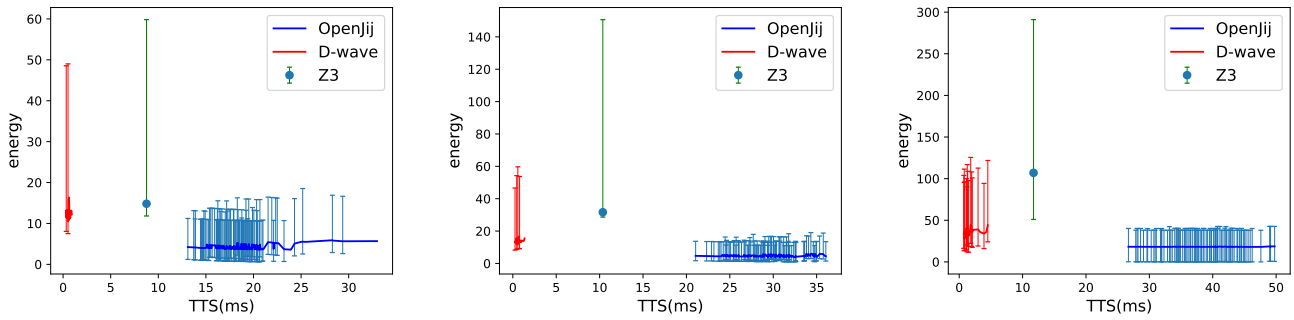
3 節で示した評価から，QUBO 式で定式化したシフトスケジューリング問題を量子アニーリングマシンで実行することで，既存手法より高速かつエネルギーの低い解を得られることを示した．しかし量子アニーリングの実行時間の解釈には注意をする必要がある．量子アニーリングマシンは 1 回の実行では実行可能解を保証できないので，実行可能解を得ることを保証するために実行可能解が得られるまで実行回数を重ねる必要がある．アニーリング処理回数を増やした場合の実行可能解が得られる確率について考える．一回のアニーリング処理で実行可能解が得られる確率 r_1 に対して，m 回のアニーリング処理で実行可能解が得られる確率 r_m は以下となる．

$$r_m = 1 - (1 - r_1)^m \tag{13}$$

この (13) 式より，確率 r_m で実行可能解を得るために必要なアニーリング処理の回数 m は以下となる．

$$m = \lceil \frac{log(1 - r_m)}{log(1 - r_1)} \rceil \tag{14}$$

例えば実行可能解を得る確率が 50%($r_1 = 0.50$) で実行



(a) 変数 60 個
20 variables

(b) 変数 90 個
90 variables

(c) 変数 126 個
20 variables

図 6: TTS での既存手法との比較
Fig. 6 Comparison with existing methods in TTS

時間が τ のアニーリング処理の場合，99%($r_m = 0.99$) 以上の確率で実行可能解を得るための TTS は (15) 式にそれぞれの値を代入し，$m = 7$ 回を実行する必要がある．

この (14) 式の m に一回の実行時間 τ を掛けることで TTS(Time to solution) が得られる．

$$TTS(\tau, p_M) = \tau \lceil \frac{ln(1 - r_m)}{ln(1 - r_1)} \rceil \tag{15}$$

1 回あたりの実行時間 τ が短くとも実行可能解を得られる確率 r_1 が小さければ，(14) 式の実行回数 m が多くなり，結果として TTS が大きくなってしまう可能性がある．逆に一回あたりの実行時間が長くても実行可能解を得られる確率が大きければ，実行回数 m が少なくて済むため，TTS が小さく済む可能性もある．よって TTS を計測することで 99%で実行可能解を得ることが保証できる時間を算出することができる．

図 6 は図 5 と同様にハードウェアで量子アニーリングマシンである D-wave advantage 1.1 と古典コンピュータのソフトウェアで実装された量子アニーリングシミュレータの OpenJij の結果から TTS を算出した．さらにシフトスケジューリング問題を QUBO 式を基に制約充足問題として定義し，制約充足問題のソルバーの一つの SMT(Satisfiability Modulo Theories) ソルバーのソフトウェア，The Z3 Thorem Prover との解と比較したものである．赤色のエラーバーが D-wave advantage 1.1 , 青色が OpenJij , 緑色が The Z3 Thorem Prover である．図 6 は図 5 と違い，横軸は 99%で実行可能解が得られる時の TTS(ms) となっているので，既存手法である The Z3 Theorem Prover と実行可能解をほぼ確実に得られるという点でなるべく条件を揃えた比較をすることが出来る．

どの変数の個数でも D-wave の TTS は 5ms にも満たず，ハードウェア実装した量子アニーリングマシンは他二つの実行時間よりも短い TTS で解を得られることが分かる．しかし図 5 では OpenJij で安定して解を得られた際の実行時間は Z3 よりも早かった，または同等であったのに対し，TTS で比較をすると Z3 よりも遅くなってしまっている．よって，TTS の観点では量子アニーリングシミュレータは既存手法に劣るが，エネルギーの観点ではどの手法よりも安定して低いエネルギーを得ることができることで差別化できている．

5. 結論

本論文では制約を限定した基礎的なシフトスケジューリング問題の QUBO 式を構築した．その後，量子アニーリングをハードウェアで実装した D-wave systems 社の量子アニーリングマシンを用いて小規模のシフトスケジューリング問題のペナルティ係数をチューニングし，実行可能解を得られる確率，エネルギーの推移を示した．変数が 60 個，90 個の際には 0.1ms で，変数が 126 個の際には 0.2ms で最小値に近いエネルギー値を安定して得ることができた．そして本問題を量子アニーリングシミュレータである OpenJij と既存手法の SMT ソルバーの The Z3 theorem Prover との実行結果と比較し，そのどちらよりも高速に解を得ることができた．また，量子アニーリングが既存手法よりエネルギーの低い解を安定して得られることを示した．

現在のハードウェアで量子アニーリングマシンを実現しているものでは量子ビット数が限られており，その結合方式も全ての量子ビット間で行われているものではないため，Embedding の操作で実際のイジングモデルに活用できる量子ビット数は少なくなってしまう．これは解くことができる問題の規模が小さくなってしまうことを意味している．しかし実際のシフトスケジューリング問題は作業員が 100 人，日数が 30 日，ターム数が 4 などの変数が 10,000 以上のものが想定されているため，現実にはまだ実現できない．よって 1 回の実行ではなく，問題分割を行うことでそれを実装できればと考えている．

謝辞 This paper is based on results obtained from

a project, JPNP16007, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

参考文献

[1] 茨木俊秀. ”組合せ最適化とスケジューリング問題: 新解法とその動向.” 計測と制御 34.5 (1995): 340-346.
[2] 池上敦子. ”ナース・スケジューリング-調査・モデル化・アルゴリズム.” (2005).
[3] Miller, Holmes E., William P. Pierskalla, and Gustave J. Rath. ”Nurse scheduling using mathematical programming.” Operations Research 24.5 (1976): 857-870.
[4] 鈴木邦成, 村山要司, and 若林敬造. ”ナーススケジューリング問題の現状と展望.” 第 77 回全国大会講演論文集 2015.1 (2015): 213-214.
[5] Ullman, Jeffrey D. ”NP-complete scheduling problems.” Journal of Computer and System sciences 10.3 (1975): 384-393.
[6] Kadowaki, Tadashi, and Hidetoshi Nishimori. ”Quantum annealing in the transverse Ising model.” Physical Review E 58.5 (1998): 5355.
[7] E. G. Rieffel, D. Venturelli, B. O’Gorman, M. B. Do, E. M. Prystay, and V. N. Smelyanskiy, “A case study in programming a quantum annealer for hard operational planning problems,” Quantum Inf. Process., vol. 14, no. 1, 2014.
[8] Choi, Vicky. ”Minor-embedding in adiabatic quantum computation: I. The parameter setting problem.” Quantum Information Processing 7.5 (2008): 193-209.
[9] Bian, Zhengbing, et al. ”Discrete optimization using quantum annealing on sparse Ising models.” Frontiers in Physics 2 (2014): 56.
[10] Lucas, Andrew. ”Ising formulations of many NP problems.” Frontiers in Physics 2 (2014): 5.
[11] D-wave systems ,https://www.dwavesys.com/
[12] OpenJij，https://github.com/OpenJij/OpenJij
[13] De Moura, Leonardo, and Nikolaj Bjørner. ”Z3: An efficient SMT solver.” International conference on Tools and Algorithms for the Construction and Analysis of Systems. Springer, Berlin, Heidelberg, 2008.
[14] The Z3 Theorem Prover ,https://github.com/Z3Prover/z3