IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 7 September 2023, accepted 9 October 2023, date of publication 13 October 2023, date of current version 20 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3324378

RESEARCH ARTICLE

Problem Relaxation Methods for Quantum Minimum Fill-in Algorithm

TOMOKO KOMIYAMA AND TOMOHIRO SUZUKI
Integrated Graduate School of Medicine, Engineering, and Agricultural Sciences, University of Yamanashi, Kofu 400-8511, Japan
Corresponding author: Tomoko Komiyama (tkomiee@gamil.com)

This work was supported by the Japan Science and Technology Agency through the project "Establishment of university fellowships toward the creation of science technology innovation," under Grant JPMJFS2117.

ABSTRACT Current quantum annealing and quantum-inspired annealing devices have many usage limitations and are difficult to apply to real-scale problems. In particular, a major hardware limitation is the limited number of available variables (qubits). This paper proposes a problem relaxation method for the Quantum Minimum Fill-in (QMF) algorithm. QMF finds the ordering of matrix rows and columns that minimizes the incidence of fill-ins that occur when a direct solver is used to solve linear equations with sparse matrices. In general, the first few steps in the forward elimination of sparse matrices add the most fill-ins. Therefore, QMF was relaxed to order the rows and columns in only the first few steps. The results obtained using the Fixstars Amplify Annealing Engine, a quantum-inspired annealing device, show that the problem relaxation can be computed with 20 % – 60 % of the qubits for the original problem and that relaxation can be applied to larger problems. Furthermore, it is found that more solutions that satisfy the constraints can be obtained with problem relaxation and that the number of fill-ins is reduced. These results confirm that the proposed problem relaxation is effective for QMF.

INDEX TERMS Quantum annealing, sparse matrix, fill-in reduction, ordering, relaxation method.

I. INTRODUCTION

Quantum annealing (QA) is an optimization technique that utilizes transitions between states due to quantum fluctuation effects to search for solutions [1], [2]. Since the development of QA machines with superconducting integrated circuits [3], [4], QA has attracted attention as a metaheuristic solution method for combinatorial optimization problems. The combinatorial optimization problems can be formulated as a quadratic unconstrained binary optimization (QUBO):

$$E(x) = x^T Q x, \tag{1}$$

where x is a vector of $x_i \in \{0, 1\}$ and Q is a matrix whose elements Q_{ij} are the interactions between x_i and x_j. QA finds x that minimizes $E(x)$, which is called the cost function. QA introduces quantum fluctuations into the simulated annealing [5] process to speed up the search for the optimal solution [1]. By gradually lowering the system's temperature over a sufficiently long period, simulated annealing is guaranteed to reach its optimal solution. However, in many cases, the time simulated annealing requires is impractically long. Simulated annealing is often applied with an insufficient search time, resulting in suboptimal solutions (local solutions). In contrast, QA introduces a quantum tunneling process into the state transition. QA may avoid falling into local solutions and can more quickly reach the optimal solution [1].

In addition to superconducting integrated circuit machines, quantum-inspired annealing (QIA) machines (also called Ising machines) are available from some companies [6], [7], [8]. QIA machines can handle more quantum bits (qubits) at a lower cost compared with machines that utilize superconducting integrated circuits. Since the number of qubits corresponds to the number of variables in a problem that can be handled, QIA is expected to solve real-world problems. However, both QA and QIA have limitations in their use, such as the limited number of qubits available in current machines and the difficulty of applying them to

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Huang.

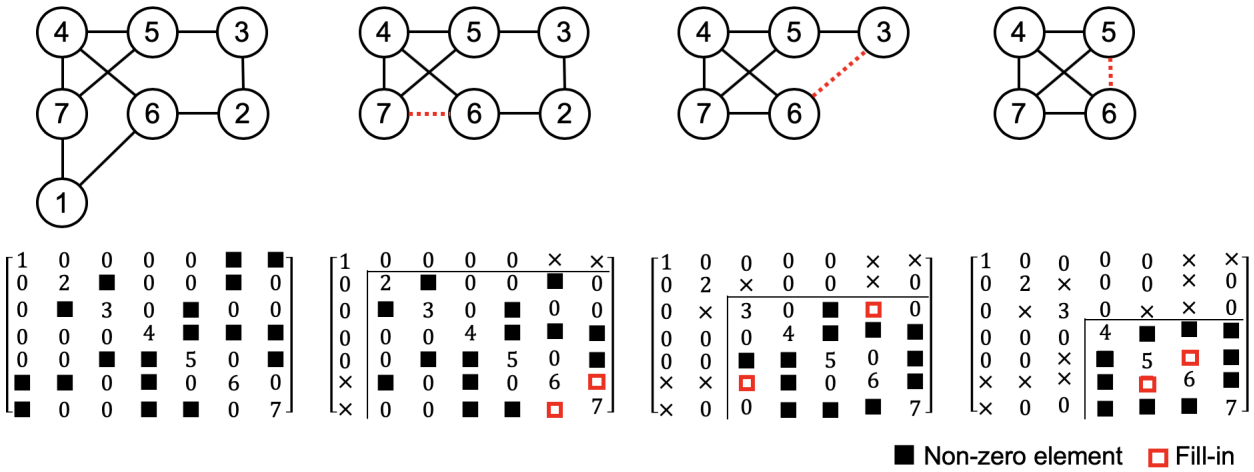


FIGURE 1. Example of MD algorithm. A symmetric matrix and the corresponding elimination graph for each elimination step are shown. The elimination proceeds from left to right. In this example, 2 × 3 fill-ins are generated.

real-scale problems. An effective method for using these machines is expected to significantly contribute to the application of QA to practical problems.

Scientific and engineering computations often involve solving parse matrix linear systems. Solving such equations using a direct solver frequently results in fill-ins, where the zero elements of the sparse matrix become nonzero. Fill-ins degrade the sparsity of the original matrix and increase memory and computational costs. The number of fill-ins can be reduced by pre-ordering the matrix elements. It is known that finding the ordering that minimizes the number of fill-ins is an NP-complete problem [9]. Various algorithms have been studied for this problem, including heuristic methods and exact solution methods such as the divide-and-conquer method.

In [10], the problem of finding orderings to reduce the number of fill-ins for a positive definite symmetric sparse matrix was formulated as a combinatorial optimization problem, which was solved using QA. This ordering method is called Quantum Minimum Fill-in (QMF). However, since current QA/QIA machines have limited available qubits, QMF can only be applied to very-small-size matrix problems. The solutions that satisfy the problem's constraint were obtained only for problems up to 9×9 or 10×10 matrices. Problems with such small matrices have few fill-ins and thus even conventional methods can produce orderings with the minimum number of fill-ins.

In this paper, a method for handling problems with large matrices, achieved by relaxing QMF, is proposed. With relaxation, the ordering of only the first few steps is found for larger matrices. The results show that relaxed QMF can find the orderings obtained by QMF and can sometimes find orderings with fewer fill-ins than those obtained using conventional ordering methods.

The remainder of this paper is organized as follows. The ordering of sparse matrices is described in Section II. A conventional ordering method based on graph theory and QMF is also described. A problem relaxation method for QMF is proposed in Section III. The results of the proposed method are presented in Section IV. Lastly, conclusions are presented in Section V.

II. FILL-IN REDUCTION ORDERING FOR SPARSE MATRICES

Large-scale numerical simulations often involve solving linear equations with large sparse matrices. Memory consumption can be reduced by storing only the nonzero elements of the sparse matrix. However, when a direct solver is used to solve linear equations with sparse matrices, fill-ins frequently occur, where elements that were zero become nonzero. Fill-ins degrade the sparsity of sparse matrices, increasing computational cost and memory consumption. A technique called fill-in reduction ordering is utilized to address this problem. Fill-in reduction ordering pre-orders the rows and columns of matrices to minimize the occurrence of fill-ins. However, finding such an ordering is an NP-complete problem and thus to difficult to achieve in a feasible time on a classical computer. Therefore, some heuristic ordering methods [11], [12], [13], [14], [15], [16], [17], [18] have been proposed.

A positive definite symmetric sparse matrix can be expressed as a graph by setting the diagonal elements as nodes and the nonzero elements as edges, as shown in Fig. 1. This graph, called the elimination graph, can be used for ordering because the order in which nodes are eliminated in the elimination graph is equivalent to ordering and the edges added when nodes are eliminated are equivalent to fill-ins. An edge is added so that a complete graph is formed by edges connected to nodes adjacent to the eliminated node. The conventional ordering method based on this elimination graph is described in II-A. In II-B, the problem of finding the ordering that minimizes fill-ins using the elimination graph as a 0-1 integer programming problem was formulated. This problem was then solved using QA.

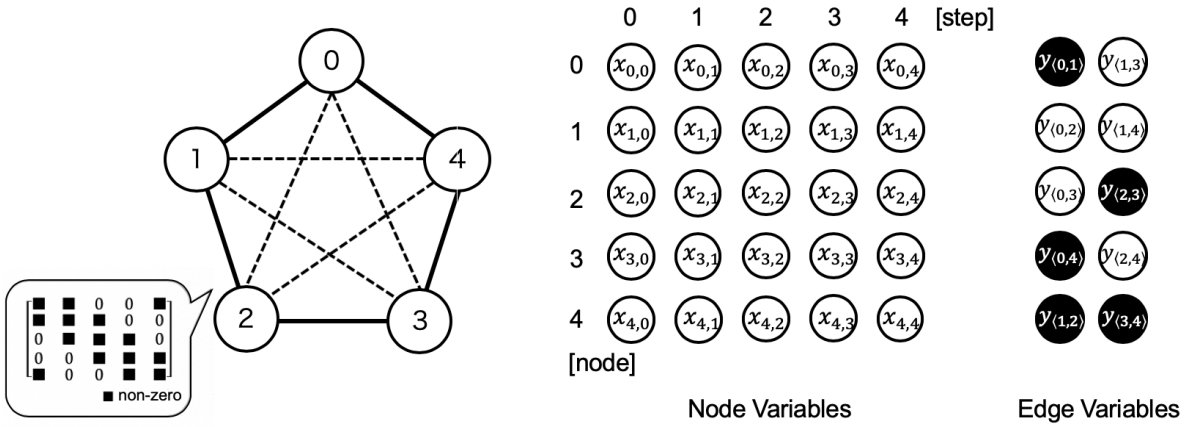


FIGURE 2. Example of variable setting for QMF algorithm. The matrix elimination graph is shown on the left, the variable set x, which indicates the elimination steps for nodes 0 to 4, is shown in the center, and the variable set y, which indicates the existence or non-existence of edges, is shown on the right.

A. CONVENTIONAL FILL-IN REDUCTION ORDERING METHODS

Here, some conventional methods based on elimination graphs are presented.

One of the most commonly used conventional methods is the minimum degree (MD) algorithm [11], [12]. This algorithm minimizes the number of fill-ins in each step by eliminating the node with the lowest degree (lowest number of edge connections) in the elimination graph in each step [13].

Fig. 1 shows an example of applying the MD algorithm to a 7×7 matrix. Note that the MD algorithm and forward elimination remove nodes in the same order in this matrix. In this figure, we first eliminate node 1, which has the lowest degree in the elimination graph (diagram on the left). This is equivalent to eliminating the first row and first column in the original matrix. When node 1 is eliminated, an edge must be added to connect all the neighbors of node 1. In this figure, an edge that connects nodes 6 and 7 is added (second diagram from the left). The addition of the edge is equivalent to a fill-in and causes a fill-in for the elements at row 6/column 7 and row 7/column 6. If this process is continued for the subsequent steps, the elimination graph becomes complete in step 4 (diagram on the right). After this step, regardless of which order the nodes are eliminated, no fill-in occurs. The elimination graph also shows that 3×2 fill-ins are generated in this matrix by the MD algorithm.

The MD algorithm can effectively order small matrices. It does not generate fill-ins for problems where the elimination graph has no cycles [14]. The algorithm has been improved in many studies. Some studies have reduced the run time by computing approximate node degrees [15], [16].

Another commonly used algorithm is the nested dissection algorithm [17]. This algorithm recursively divides the matrix of a mesh structure into multiple smaller matrices, which can then be processed in parallel to quickly find solutions to linear equations. This approach has been improved to handle common graph structures [18]. The nested dissection algorithm is applied to handle large matrices in parallel, not to minimize the number of fill-ins. In this study, the MD algorithm is referred to as the conventional ordering method for fill-in reduction.

B. QUANTUM MINIMUM FILL-IN ALGORITHM

The QMF algorithm was proposed as a method for finding the order with the minimum number of fill-ins as a combinatorial optimization problem. It uses an elimination graph to find the elimination order of vertices (ordering) that minimizes edge addition (fill-ins).

The QMF formulation introduces the binary variable $x_{n,s} \in \{0, 1\}$, which represents the order of node elimination. The variable $x_{n,s}$ represents the elimination of node n in step s when $x_{n,s} = 1$. The binary variable $y_{\langle u,v\rangle} \in \{0, 1\}$ is introduced to represent the existence or non-existence of an edge between nodes u and v. If the edge exists, $y_{\langle u,v\rangle} = 1$. Thus, if an edge is not present in the initial elimination graph, it represents an added edge (fill-in). An example of this variable setup is shown in Fig. 2.

This method minimizes the number of edges added with node elimination. The function H_{cost}, which minimizes the number of added edges, is defined in (2).

$$H_{cost} = \sum_{\langle u,v\rangle \in \bar{E}} y_{\langle u,v\rangle}, \tag{2}$$

where $\bar{E}$ is the set of edges that do not exist in the first elimination graph. If there are no fill-ins, H_{cost} has a minimum value of 0.

There are three constraints in this approach.

Constraint I: All nodes are eliminated

The function (3) for this constraint takes the minimum value 0 when the sum of the variables $x_{n,s}$ that represent the node n equals one. This means that each node is eliminated

only once.

$$H_1 = \sum_{n=0}^{N-1}\left(\sum_{s=0}^{N-1} x_{n,s} - 1\right)^2, \tag{3}$$

where N is the matrix size and $x_{s,n} = 1$ means that the n-th node is eliminated in the s-th step.

Constraint II: Only one node can be eliminated in each step

The function (4) for this constraint takes the minimum value 0 when the sum of the variables $x_{n,s}$ that represent the same elimination step s is one. This means that only one node is eliminated in each elimination step s.

$$H_2 = \sum_{s=0}^{N-1}\left(\sum_{n=0}^{N-1} x_{n,s} - 1\right)^2. \tag{4}$$

Constraint III: Edges are added with node elimination In each step, edges are added so that all nodes connected to the node to be eliminated are joined. Fig. 2 is used as an example to explain this constraint. When node 0 is eliminated, an edge between nodes 1 and 4 is added to make adjacent nodes 1 and 4 fully connected. The addition of an edge has the following conditions.

- node n is eliminated in step s ($x_{n,s} = 1$; e.g., $x_{0,0} = 1$)
- node i that is not eliminated in the step before step s($\sum_{k=0}^{s} x_{u,k} = 0$; e.g., $\sum_{k=0}^{0} x_{1,k} = 1$) is eliminated
- node j is not eliminated in the step before step s($\sum_{k=0}^{s} x_{v,k} = 0$; e.g., $\sum_{k=0}^{0} x_{4,k} = 1$)
- there exists an edge that connects nodes n and i to be eliminated ($y_{\langle n,u\rangle} = 1$; $y_{\langle 0,1\rangle} = 1$)
- there exists an edge that connects nodes n and j to be eliminated ($y_{\langle n,v\rangle} = 1$; $y_{\langle 0,4\rangle} = 1$)

When all the above conditions are satisfied, the required edge (the edge between nodes 1 and 4 in Fig. 2) is added ($y_{\langle u,v\rangle} = 1$; e.g., $y_{\langle 1,4\rangle} = 1$). The function that satisfies this constraint is formulated using two methods, one based on higher-order functions and the other based on inequalities.

This constraint can be formulated using a higher-order function as follows.

$$H_3 = \sum_{\langle u,v\rangle\in\bar{E}}\sum_{n\notin\{u,v\}}\sum_{s=0}^{N-1}(1 - y_{\langle u,v\rangle})x_{n,s}\left(1 - \sum_{k=0}^{s} x_{u,j}\right) \times \left(1 - \sum_{k=0}^{s} x_{v,j}\right) y_{\langle n,u\rangle} y_{\langle n,v\rangle}. \tag{5}$$

QA can optimize problems expressed as a QUBO. Since QA cannot handle functions with an order higher than three, auxiliary variables are introduced for order reduction. The auxiliary variable $\mathbf{b}$ is defined as follows.

$$b_{\langle u,v\rangle} = (1 - y_{\langle u,v\rangle}),\ b_{u,j} = \left(1 - \sum_{k=0}^{s} x_{u,j}\right),\ b_{v,j} = \left(1 - \sum_{k=0}^{s} x_{v,j}\right) \tag{6}$$

With the auxiliary variable $\mathbf{b}$, the higher-order function (4) can be formulated as a monomial of the sixth degree.

$$H'_{3a} = \sum_{\langle u,v\rangle\in\bar{E}}\sum_{n\notin\{u,v\}}\sum_{s=0}^{N-1} b_{\langle u,v\rangle}\, x_{n,s}\, b_{u,j}\, b_{v,j}\, y_{\langle n,u\rangle}\, y_{\langle n,v\rangle}. \tag{7}$$

This function is converted to a QUBO using the Ishikawa method [19]. The number of auxiliary variables introduced at this time can be expressed as $2N(N-2)|\bar{E}|$ for a problem size N and a total number of edges $|\bar{E}|$ that are not present in the elimination graph.

This constraint can also be formulated using an inequality as follows.

$$(1 - y_{\langle u,v\rangle}) + x_{n,s} + \left(1 - \sum_{k=0}^{s} x_{u,j}\right) + \left(1 - \sum_{k=0}^{s} x_{v,j}\right) + y_{\langle n,u\rangle} + y_{\langle n,v\rangle} \le 5 \tag{8}$$

The left-hand side of (8) takes integer values from 0 to 6. The constraint H_3 is satisfied when the value is less than or equal to 5. Since QA cannot handle inequality constraints, the inequalities are converted into equalities by adding auxiliary variables. Therefore, this constraint can be formulated using the inequality (8) and auxiliary variables y_l as follows.

$$H'_{3b} = \sum_{\langle u,v\rangle\in\bar{E}}\sum_{n\notin\{u,v\}}\sum_{s=0}^{N-1}\left\{(1 - y_{\langle u,v\rangle}) + x_{n,s} + \left(1 - \sum_{k=0}^{s} x_{u,j}\right) + \left(1 - \sum_{k=0}^{s} x_{v,j}\right) + y_{\langle n,u\rangle} + y_{\langle n,v\rangle} - \sum_{l=0}^{4} y_l\right\}^2. \tag{9}$$

The function H to be optimized with QA is formulated by adding the weights δ and ϵ to each of the above functions.

$$H = H_{cost} + \delta(H_1 + H_2) + \epsilon H'_3 \tag{10}$$

Functions with larger weights are more likely to be minimized (i.e., more likely to satisfy the constraints). However, if the weights applied to the constraints are too large, it is difficult to minimize (maximize) the cost function. Conversely, if the weights are too small, finding a solution that satisfies the constraints becomes difficult. Therefore, it is essential to apply appropriate weights in order to obtain good-quality solutions. The method for setting these weights is explained in Section IV.

III. RELAXATION OF QMF ALGORITHM

Due to hardware limitations, current QA/QIA machines are limited in terms of the number of variables available. Therefore, it is difficult to obtain orderings for large matrices using QMF with the current machines. In order to obtain orderings for such matrices, the number of variables in the problem must be significantly reduced. In the forward elimination of the sparse direct solver, many fill-ins occur in the first few steps. Fig. 3 shows the average number of fill-ins for the first and last half of steps when QMF is applied to

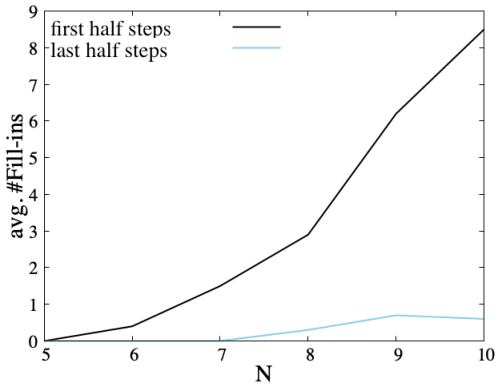


FIGURE 3. Number of fill-ins for first and last half of steps of QMF. The average number of fill-ins when the QMF is applied to 10 $N \times N$ matrices is shown. The vertical axis is the average number of fill-ins and the horizontal axis is the problem size N.

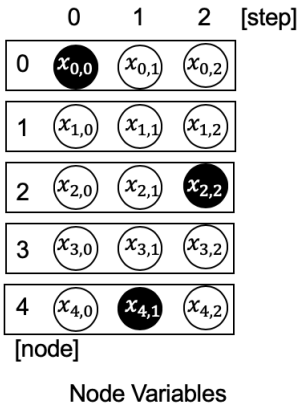


FIGURE 4. Modified Constraint I for node elimination with problem relaxation.

10 $N \times N$ matrices in which nonzero elements occur with a probability of 1/3. This figure shows that the majority of fill-ins occur in the first half of the steps. In other words, by finding the order of only the first few steps, the number of fill-ins can be reduced. This can significantly reduce the number of variables in the problem and increase the size of the target matrix.

To find the ordering of only the first few steps in QMF, the function H_1 for Constraint I is changed as follows.

$$H_1' = \sum_{n=0}^{N-1} \sum_{s=0}^{S-1} x_{n,s} \left(\sum_{s=0}^{S} x_{n,s} - 1 \right), \tag{11}$$

where N is the matrix size and S is the number of steps to be ordered. When ordering only the first few steps, there are nodes that are eliminated and nodes that are not eliminated, as shown in Fig. 4. Therefore, it can be formulated as a function that takes a minimum value when the sum of the variables $x_{n,s}$ representing node n is equal to 0 or 1.

Furthermore, since only the rows and columns selected in the first few steps are swapped, the other rows and columns are eliminated without being swapped. This is equivalent to eliminating the remaining nodes in ascending order of node number (i.e., row and column number) in the elimination graph.

IV. EXPERIMENTS AND RESULTS

This section presents the results of evaluating the size of matrices that can be ordered using QMF with problem relaxation. The variation of the number of fill-ins for QMF with problem relaxation is also investigated.

The number of fill-ins generated during forward elimination after QMF ordering is applied for only the first three steps is measured. In the proposed method, ordering only the first step is almost the same as the MD algorithm, and the number of fill-ins cannot be reduced compared with that for the MD algorithm. In addition, for the tested problem sizes, a larger variable reduction is not expected with more than five steps and the number of fill-ins generated in later steps is small. Therefore, three steps were applied in this experiment.

In this experiment, QMF is performed using the Fixstars Amplify Annealing Engine [6], a GPU-based annealing machine that utilizes parallel processing. This QIA machine is capable of fast annealing of a fully connected graph with over 100,000 qubits. To facilitate parameter tuning, the parameters δ and ϵ in (10) were normalized so that the maximum coefficient of each constraint function had the same magnitude. The annealing time (i.e., time taken to solve the problem) was set to 2000 ms.

The parameters δ and ϵ in (10) were set as follows.

$$\delta = \epsilon \, max(H_3'), \tag{12}$$

where $max(H')$ is the maximum coefficient value. The value of ϵ was varied in the range of 1.0 to 10.0. The values that gave the most orderings with the fewest fill-ins were used.

For the experiments, 100 matrices were used. We used a random matrix in which nonzero elements occurred with a probability of 1/3. We compared the number of required qubits and the number of fill-ins for the same matrices without ordering, with MD ordering applied, and with QMF ordering applied in all steps.

A. NUMBER OF QUBITS

Here, the number of qubits required for the calculation is evaluated. Fig.5 shows a comparison of the average number of qubits for ordering only the first three steps and all steps. Both QMFs using (7) and (9) show a significant reduction in the number of qubits and smaller problem sizes due to problem relaxation. In particular, a larger decrease in qubits was achieved for a larger problem size (N). It was found that $N = 15$ requires only about 20 % of the qubits of the original problem (before relaxation). There is no significant difference in the number of variables that would be required using either formulation.

Thus, even QA/QIA devices with a limited number of qubits are capable of solving large-size problems via problem relaxation. In the experiment described below (see Section IV-B), we solved problems with up to 160,000 qubits.

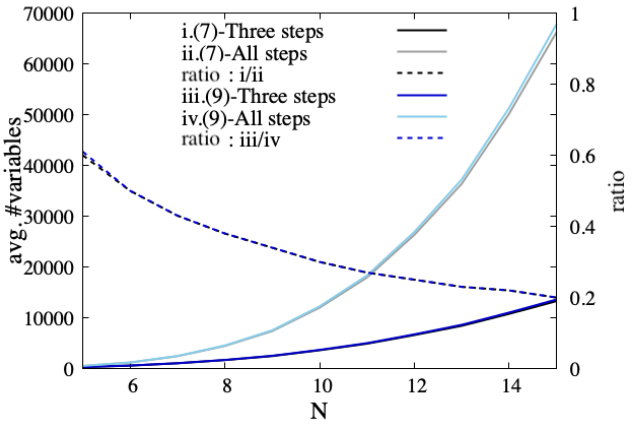


FIGURE 5. Average number of variables. It is almost the same when using QUBO based on the higher-order formulation and the inequality-based formulation. Note that the lines "i" and "iii", "ii" and "iv", "ratio: i/ii" and "ratio: iii/iv" overlap each other.

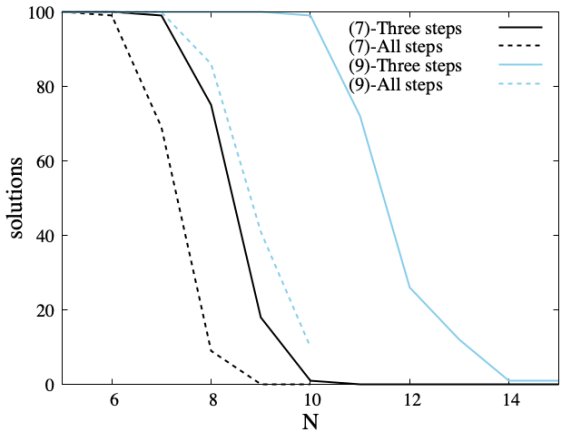


FIGURE 6. Results for total number of solutions that satisfy constraints when QMF is executed.

TABLE 1. Comparison of results for number of fill-ins between QMF using (7) and other methods.

N	QMF-NoOrder Fill-ins						QMF-MD Fill-ins					
	Three steps			All steps			Three steps			All steps		
	<	=	>	<	=	>	<	=	>	<	=	>
5	36	64	0	35	65	0	0	100	0	0	99	1
6	54	40	6	47	37	15	0	77	23	0	63	36
7	50	42	7	44	17	8	0	44	55	0	35	34
8	53	19	3	7	1	1	0	26	49	0	3	6
9	16	2	0				0	2	16			
10	1	0	0				0	0	1			

It is possible to conduct experiments with up to $N = 15$ when ordering only the first three steps and up to $N = 10$ when ordering all steps. For these problem sizes, adding one step increases the number of variables by approximately $O(N^3)$.

B. NUMBER OF FILL-INS

Here, the number of fill-ins when ordering and forward elimination are applied is evaluated. QMF compared two formulations, (7) and (9). For each matrix size $N \times N$, the following results are presented. 1) the total number of solutions that satisfy the constraints when QMF is executed, 2) the average number of fill-ins for each method (avg. Fill-ins) and 3) the comparison results for the number of fill-ins between QMF and each method (i.e., between QMF and method without ordering (NoOrder), and between QMF and MD). The comparison of the number of fill-ins between QMF and each method is based on the number of solutions that satisfy the constraints where the number of fill-ins is QMF $<$ NoOrder (or MD), QMF=NoOrder (or MD), and QMF $>$ NoOrder (or MD). For example, in Table 1, in the "Three steps" column for QA-NoOrder fill-ins, for $N = 5$, QMF yielded 36 solutions with fewer fill-ins than those obtained without ordering and 64 solutions with the same number of fill-ins.

First, the results of 1) are shown in Fig. 6. Fig. 6 shows that it becomes harder to obtain solutions that satisfy the constraints as the matrix size increases for all QMF methods. With QUBO based on the higher-order formulation (7), the solutions that satisfy the constraints can be obtained for up to 10×10 matrices when ordering only the first three steps, and for up to 8×8 matrices when ordering all steps. With QUBO based on the inequality-based formulation (9), the solutions that satisfy the constraints can be obtained for up to 15×15 matrices when ordering the first three steps, and for up to 10×10 matrices when ordering all steps. A larger matrix can be used when ordering the first three steps because the actual problem size to be solved is smaller due to problem relaxation. Fig. 6 also shows that it is easier to obtain a solution that satisfies the constraints when using QUBO with (9) than when using QUBO with (7). Even when the problem size was reduced to give a computable number of variables in Amplify AE, no solution satisfying the constraint was obtained in QUBO with (7).

Next, the results for QMF with QUBO based on the higher-order formulation (7) are discussed in terms of the number of fill-ins. The results for 2) and 3) for QMF using QUBO with the higher-order formulation (7) are shown in Fig. 7 and Table 1, respectively. Fig. 7 shows that when ordering only the first three steps or all steps, the average number of fill-ins is smaller than that for NoOrder, but larger than that for MD. It can also be seen that the average number of fill-ins is almost the same when ordering the first three steps or all steps. On the other hand, a comparison of the number of fill-ins between QMF and the conventional method (MD) in Table 1 indicates that ordering only the first three steps tends to produce more solutions with the same number of fill-ins than those obtained with the conventional method. Note that in this experiment, this QUBO did not yield a solution with fewer fill-ins than MD.

Lastly, the results for QMF with QUBO based on the inequality-based formulation (9) are discussed in terms of the number of fill-ins. The results for 2) and 3) for QMF are shown in Fig. 8 and Table 2, respectively. Fig. 8 shows that ordering the first three steps or all steps gives a smaller

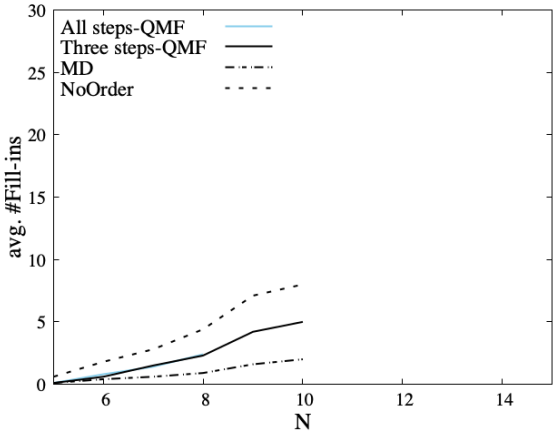


FIGURE 7. Average number of fill-ins for QUBO using higher-order function (7). It is almost the same when ordering the first three steps or all steps. Note the overlapping lines "All steps-QMF" and "Three steps-QMF."

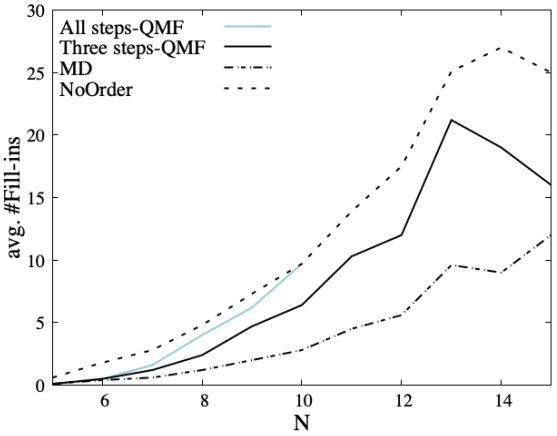


FIGURE 8. Average number of fill-ins for QUBO using inequality function (9).

TABLE 2. Comparison of number of fill-ins between QMF using (9) and other methods.

N	QMF-NoOrder Fill-ins						QMF-MD Fill-ins					
	Three steps			All steps			Three steps			All steps		
	<	=	>	<	=	>	<	=	>	<	=	>
5	36	64	0	36	64	0	0	100	0	0	100	0
6	56	44	0	54	42	4	0	90	10	0	86	14
7	64	33	3	56	21	23	0	60	40	0	41	59
8	74	22	4	42	13	31	0	35	65	0	11	75
9	76	17	7	27	1	13	0	11	89	0	3	38
10	76	13	10	4	0	6	0	7	92	0	0	10
11	60	8	4				0	1	71			
12	21	2	3				0	0	26			
13	10	0	2				0	0	12			
14	1	0	0				0	0	1			
15	1	0	0				0	0	1			

average number of fill-ins than NoOrder, but a larger number than MD. This is almost the same result as for QMF with QUBO based on (7). However, the average number of fill-ins is about the same regardless of whether the first three steps or all steps are ordered. In addition, a comparison of QMF and the conventional method shows that there is a tendency for many solutions to have the same number of fill-ins when the first three steps are ordered. Consequently, problem relaxation using QUBO based on (9) is more effective than using QUBO based on (7). For matrices larger than 10×10, although solutions that satisfy the constraints are obtained, most have a higher number of fill-ins than that for MD. The solution with the minimum number of fill-ins was not obtained because only a few steps were ordered. It is also highly possible that the optimal solution was not obtained with QA because the number of fill-ins tends to be larger than that for MD when the problem size is larger even when all steps are ordered. Since MD is capable of ordering effectively for the problem sizes in this experiment, the effect of not obtaining an optimal solution with QA is more significant.

These results suggest that problem relaxation using the proposed method is effective for QMF. However, the average number of fill-ins with QMF for both methods is smaller than that without ordering and larger than that with MD. This is due to the fact that QA does not always provide the optimal solution, and therefore does not provide better ordering than the MD method, which can be effective for smaller sizes. Therefore, when large matrices can be solved with QA in the future, it may be possible to order more effectively than MD with QMF. When it is possible to use larger matrix sizes, the change in the number of fill-ins due to ordering only a few steps will be discussed in more detail.

V. CONCLUSION

Current QA/QIA devices have many usage limitations and are difficult to apply to real-scale problems. Here, a problem relaxation method for QMF was proposed and the results of applying and evaluating QMF for larger problems were presented. Generally, the first few steps are the ones where most fill-ins are added in the forward elimination of a sparse matrix. Therefore, QMF was relaxed to order only the first few steps.

From the results of the number of qubits required for computation, it was found that a problem can be computed with 20 % – 60 % of the qubits of the original problem when problem relaxation is applied. This effect is larger for larger problem sizes. It was found that large matrices can be ordered even using current QA/QIA devices. Furthermore, the orderings were evaluated using the number of fill-ins in the matrices to which the orderings were applied. It was found that QA can obtain more solutions that satisfy the constraints and fewer fill-ins for orderings with problem relaxation than those obtained without it. In this evaluation, the results were also compared with those of the conventional method (MD) and the case without ordering. It was found that the case with problem relaxation obtained more orderings with the same or lower number of fill-ins than the conventional method.

These results suggest that the effect of the problem size on the difficulty of obtaining optimal solutions is larger than the effect of problem relaxation, confirming that problem relaxation for QMF is effective.

For the matrix sizes considered in the experiment, the effect of the difficulty in obtaining an optimal solution with QA was more significant than the effect of ordering only the first few steps. In the future, when larger matrix sizes can be used for QA/QIA devices, the change in the number of fill-ins due to ordering only a few steps will be studied in more detail.

REFERENCES

[1] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse Ising model," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 58, no. 5, pp. 5355–5363, Nov. 1998.

[2] S. Tanaka, K. Tanahashi, T. Motohashi, and S. Takayanagi, "Basics and applications of quantum annealing," *Teion Kogaku*, vol. 53, no. 5, pp. 287–294, Sep. 2018.

[3] S. Boixo, T. F. Rønnow, S. V. Isakov, Z. Wang, D. Wecker, D. A. Lidar, J. M. Martinis, and M. Troyer, "Evidence for quantum annealing with more than one hundred qubits," *Nature Phys.*, vol. 10, no. 3, pp. 218–224, Feb. 2014.

[4] R. Hamerly, "Experimental investigation of performance differences between coherent Ising machines and a quantum annealer," *Sci. Adv.*, vol. 5, no. 5, May 2019, Art. no. eaau0823.

[5] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[6] *Fixstars Amplify*. Accessed: Oct. 17, 2023. [Online]. Available: https://amplify.fixstars.com/

[7] S. Matsubara, M. Takatsu, T. Miyazawa, T. Shibasaki, Y. Watanabe, K. Takemoto, and H. Tamura, "Digital annealer for high-speed solving of combinatorial optimization problems and its applications," in *Proc. 25th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2020, pp. 667–672.

[8] M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki, and H. Mizuno, "20k-spin Ising chip for combinational optimization problem with CMOS annealing," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 303–309, Mar. 2015.

[9] M. Yannakakis, "Computing the minimum fill-in is NP-complete," *SIAM J. Algebr. Discrete Methods*, vol. 2, no. 1, pp. 77–79, Mar. 1981, doi: 10.1137/0602010.

[10] T. Komiyama and T. Suzuki, "Sparse matrix ordering method with a quantum annealing approach and its parameter tuning," in *Proc. IEEE 14th Int. Symp. Embedded Multicore/Many-Core Syst. Chip (MCSoC)*, Dec. 2021, pp. 258–264, doi: 10.1109/MCSoC51149.2021.00045.

[11] W. F. Tinney and J. W. Walker, "Direct solutions of sparse network equations by optimally ordered triangular factorization," *Proc. IEEE*, vol. 55, no. 11, pp. 1801–1809, Nov. 1967, doi: 10.1109/PROC.1967.6011.

[12] A. George and J. W. H. Liu, "The evolution of the minimum degree ordering algorithm," *SIAM Rev.*, vol. 31, no. 1, pp. 1–19, Mar. 1989, doi: 10.1137/1031001.

[13] T. A. Davis, *Direct Methods for Sparse Linear Systems*. Pennsylvania, PA, USA: Siam, 2006.

[14] I.S. Duff, A. M. Erisman, and J. K. Reid, "Capter," in *Direct Methods for Sparse Matrices*. New York, NY, USA: Oxford, 2017.

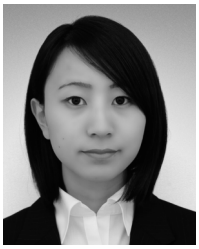
[15] P. R. Amestoy, T. A. Davis, and I. S. Duff, "An approximate minimum degree ordering algorithm," *SIAM J. Matrix Anal. Appl.*, vol. 17, no. 4, pp. 886–905, Oct. 1996, doi: 10.1137/S0895479894278952.

[16] M. Fahrbach, G. L. Miller, R. Peng, S. Sawlani, J. Wang, and S. C. Xu, "Graph sketching against adaptive adversaries applied to the minimum degree algorithm," in *Proc. IEEE 59th Annu. Symp. Found. Comput. Sci. (FOCS)*, Oct. 2018, pp. 101–112, doi: 10.1109/FOCS.2018.00019.

[17] A. George, "Nested dissection of a regular finite element mesh," *SIAM J. Numer. Anal.*, vol. 10, no. 2, pp. 345–363, Apr. 1973, doi: 10.1137/0710032.

[18] A. George and J. W. H. Liu, "An automatic nested dissection algorithm for irregular finite element problems," *SIAM J. Numer. Anal.*, vol. 15, no. 5, pp. 1053–1069, Oct. 1978, doi: 10.1137/0715069.

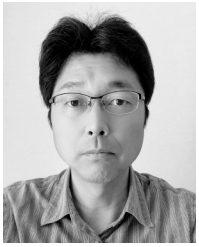
[19] H. Ishikawa, "Transformation of general binary MRF minimization to the first-order case," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 6, pp. 1234–1249, Jun. 2011, doi: 10.1109/TPAMI.2010.91.



TOMOKO KOMIYAMA received the B.E. and M.E. degrees in engineering from the University of Yamanashi, Kofu, Japan, in 2021 and 2023, respectively, where she is currently pursuing the Ph.D. degree in engineering.

Her research interest includes quantum computing, especially the application of quantum annealing and mathematical optimization.

Ms. Komiyama is a member of the Information Processing Society of Japan (IPSJ).



TOMOHIRO SUZUKI received the B.E. and M.E. degrees from the University of Yamanashi, Japan, in 1989 and 1991, respectively, and the Ph.D. degree from Nagoya University, in 2003. He is currently a Professor with the University of Yamanashi. He is a member of the ACM, the Japan Society for Industrial and Applied Mathematics (JSIAM), and the Information Processing Society of Japan (IPSJ). He received the *Japan Journal of Industrial and Applied Mathematics* Best Paper Award, in 2000.