Graph Partitioning into Hamiltonian Subgraphs on a Quantum Annealer

Eugenio Cocchi,1 Edoardo Tignone,1 and Davide Vodola1, 2, 3

1Quantum Glare Ltd, London, United Kingdom
2Dipartimento di Fisica e Astronomia dell’Università di Bologna, I-40127 Bologna, Italy
3INFN, Sezione di Bologna, I-40127 Bologna, Italy

We demonstrate that a quantum annealer can be used to solve the NP-complete problem of graph partitioning into subgraphs containing Hamiltonian cycles of constrained length. We present a method to find a partition of a given directed graph into Hamiltonian subgraphs with three or more vertices, called vertex 3-cycle cover. We formulate the problem as a quadratic unconstrained binary optimisation and run it on a D-Wave Advantage quantum annealer. We test our method on synthetic graphs constructed by adding a number of random edges to a set of disjoint cycles. We show that the probability of solution is independent of the cycle length, and a solution is found for graphs up to 4000 vertices and 5200 edges, close to the number of physical working qubits available on the quantum annealer.

I. INTRODUCTION

Many combinatorial optimisation problems arising in practical applications are notoriously hard to solve with classical methods [1]. Recently, quantum annealers have been considered as potentially faster alternatives for finding solutions to this class of problems in different domains. For example, in logistics they have been employed for job shop scheduling [2], traffic flow optimisation [3–5], and airport gate assignment [6]. In telecommunications quantum annealers have been used for satellite coverage [7], and in chemistry for protein folding [8]. In finance, use cases range from portfolio optimisation [9, 10] to prediction of financial crashes [11].

Finding Hamiltonian cycles, i.e. cycles that visit each vertex exactly once, is another type of combinatorial optimisation problem having several applications, for example in kidney and lung exchange [12–14], house allocation [15], branch selection for cadets [16], and, more generally, good exchange [17]. However, so far quantum annealers have not been used to solve these problems. Here, we consider the partitioning of a directed graph into subgraphs containing Hamiltonian cycles. More specifically, we focus on the case where the cycle length is required to be at least three, which makes the problem NP-complete [1]. This problem is also known as the vertex 3-cycle cover decision problem for directed graphs (3-DCC) [18].

II. THE PROBLEM

For solving the problem on the quantum annealer, we cast its cost function and the corresponding constraints as a Quadratic Unconstrained Binary Optimisation (QUBO) problem. We consider graphs containing disjoint cycles and random edges added and we analyse the probability of finding a vertex 3-cycle cover in a single run on the quantum annealer as a function of the size of the input graph as well as the number of edges added. We find solutions for input graphs up to 4000 vertices and 5200 edges, close to the number of physical working qubits available on the quantum annealer. We find that the dependence of the probability on the system size is stronger when the relative number of random edges added is higher, while it does not depend on the length of the cycles in the input graph.

This paper is structured as follows. In Sec. II we introduce the relevant definitions and formally define the problem. In Sec. III we show the procedure used to rewrite the problem as a QUBO. In Sec. IV we describe the quantum annealing protocol and the procedure used to test the solutions found. In Sec. V we present the results obtained with D-Wave Advantage quantum annealer. In Sec. VI we outline possible future developments.
vertices be partitioned into disjoint sets \( V_1, V_2, \ldots, V_k \) for some \( k \) such that each \( V_i \) contains at least three vertices and induces a subgraph \( G \) that contains a Hamiltonian cycle?

Figure 1 illustrates the problem for a graph composed of a set of \( N_V = 7 \) vertices \( V = \{1, \ldots, 7\} \), and \( N_E = 11 \) edges \( E = \{(1, 2), (2, 5), (2, 6), (3, 4), (4, 3), (4, 7), (5, 1), (5, 6), (6, 3), (6, 7), (7, 6)\} \) (Fig. 1a). A solution exists for this graph because it does have a partition into Hamiltonian subgraphs with one cycle of length three \( V_1 = \{1, 2, 5\} \), and one cycle of length four \( V_2 = \{3, 4, 6, 7\} \) (Fig. 1b). We stress that, since the problem requires the cycle length to be at least three, the cycles of length two do not appear in the solution.

This problem can be tackled by associating to each edge \((i, j) \in E\) a binary variable \( x_{ij} \in \{0, 1\} \), that equals 1 when vertices \( i \) and \( j \) are connected in the solution (see green arrows in Fig. 1b), 0 otherwise. We note that since the number \( N_E \) of existing edges is in general smaller than the number \( N_V(N_V - 1) \) of possible edges among all the vertices, associating variables \( x_{ij} \) only to the existing edges makes the size of the problem as small as possible.

One can then formulate the problem as:

\[
\text{maximise} \sum_{ij} x_{ij} \quad (1)
\]

subject to the following constraints:

\[
\sum_{j} x_{ij} = N_V, \quad (2)
\]

\[
\sum_{j} x_{ij} \leq 1 \quad \forall i \in V, \quad (3)
\]

\[
\sum_{i} x_{ij} \leq 1 \quad \forall j \in V, \quad (4)
\]

\[
x_{ij} + x_{ji} \leq 1 \quad \forall (i, j), (j, i) \in E. \quad (5)
\]

Constraint (2) guarantees that the number of edges is equal to the number of vertices. Constraint (3) guarantees that for every vertex there is more than one outgoing edge, likewise does constraint (4) for the ingoing edges. Constraints (2)-(4) guarantee that the solution will be a partition into Hamiltonian subgraphs. Finally, constraint (5) ensures that two vertices can be connected to each other by maximum one edge, meaning that cycles of length two are forbidden.

### III. QUBO FORMULATION

The optimisation problem described by Eqs. (1)-(5) can be rewritten using the QUBO formalism, which is suitable for a quantum annealer [20]. In the QUBO formalism we look for a configuration \( \mathbf{x} \), where \( \mathbf{x} \) is a vector with components \( x_{ij} \), that minimizes the following cost function:

\[
J(\mathbf{x}) = F(\mathbf{x}) + P(\mathbf{x}). \quad (6)
\]

The first term of Eq. (6) is

\[
F(\mathbf{x}) = -\sum_{ij} x_{ij} \quad (7)
\]

and corresponds to Eq. (1).

The second term of Eq. (6) is a penalty term

\[
P(\mathbf{x}) = P_{\text{out}}(\mathbf{x}) + P_{\text{in}}(\mathbf{x}) + P_{\text{no pairs}}(\mathbf{x}), \quad (8)
\]

where

\[
P_{\text{out}}(\mathbf{x}) = \sum_{i} a_i \sum_{j, j' > j} x_{ij} x_{ij'}, \quad (9)
\]

\[
P_{\text{in}}(\mathbf{x}) = \sum_{j} b_j \sum_{i, i' > i} x_{ij} x_{i'j} \quad (10)
\]

\[
P_{\text{no pairs}}(\mathbf{x}) = c \sum_{i, j > i} x_{ij} x_{ji} \quad (11)
\]

The quantities in Eqs. (9)-(11) implement the constraints of Eqs. (3), (4) and (5), given that they will be zero when the configuration \( \mathbf{x} \) is allowed and positive when the constraint is violated, thus penalising forbidden configurations. Equations (9)-(11) are based on the fact that for any two binary variables \( y \) and \( z \), the constraint \( y + z \leq 1 \) is equivalent to \( y \cdot z = 0 \). We note that it is not necessary to encode constraint (2) as a penalty term, since it has the same functional form as Eq. (1), and checking the solution found will be sufficient.

When translating the constraints into penalties one needs to choose the penalty constants \( a_i, b_j, c \) large enough compared to the strength of the term \( \sum_{ij} x_{ij} \). However, due to the hardware implementation, these cannot be chosen arbitrarily large. An optimal choice for the penalty constants is presented in Appendix A.

### IV. IMPLEMENTATION ON A QUANTUM ANNEALER

#### A. Quantum annealing

The constructed QUBO problem is solved on a D-Wave Advantage quantum annealer, containing 5436 physical working qubits. The starting point for the quantum routine used is a quantum state that corresponds to the ground state of a drive Hamiltonian

\[
H_0 = -\sum_{\ell} \sigma_\ell^z. \quad (12)
\]

This Hamiltonian is slowly changed to the problem Hamiltonian \( H_1 \) whose ground state represents the state with lowest energy for the QUBO problem. Its general form is that of an Ising-like Hamiltonian:

\[
H_1 = \sum_{\ell} h_\ell \sigma_\ell^z + \sum_{\ell \ell'} t_{\ell \ell'} \sigma_\ell^z \sigma_{\ell'}^z. \quad (13)
\]
In $H_0$ and $H_1$, $\ell$ and $\ell'$ are indices denoting the position of the physical qubits in the hardware, $\sigma_{x,y}^\ell$ are Pauli operators, and the parameters $h_\ell$ and $t_{\ell\ell'}$ are set by the QUBO problem.

The total evolution can then be modelled via the Hamiltonian $H(s) = A(s)H_0 + B(s)H_1$, where $A(s)$ and $B(s)$ are a decreasing and increasing function of the dimensionless schedule parameter $s \in [0,1]$, respectively. The $A(s)$ and $B(s)$ functions are fixed by the hardware, while the time variation of $s$ is controlled by the programmed schedule. In our experiment we use a schedule having a total duration of 300 $\mu$s, composed of an initial annealing where the parameter $s$ is linearly ramped up from 0 to 0.4 in 80 $\mu$s, followed by a pause at $s = 0.4$ lasting for 100 $\mu$s, and a second part of the annealing consisting of a linear ramp of $s$ from 0.4 to 1 in 120 $\mu$s.

The mapping of the QUBO problem in Eq. (6) to the Ising-like Hamiltonian in Eq. (13) is done by identifying the two states $\{0,1\}$ of the variables $x_{ij}$ with the two eigenstates of the $\sigma_z^\ell$ operator of the qubit $\ell$. Since the physical qubits in the quantum processor are not fully connected to each other, each logical qubit is embedded into a chain of one or more physical qubits. To find such an embedding, we use the minorminer algorithm [21] provided by D-Wave, with its default parameters.

**B. Protocol for solving the partitioning problem**

In this section we present the steps for the protocol we use to solve the problem on the quantum annealer: (i) graph construction, (ii) problem submission, (iii) check of the output.

(i) We start by generating a graph $G_0$ composed of $n$ disjoint cycles of length $L$ (Fig. 2a). This contains $nL$ vertices and $nL$ edges. A new graph $G$ is generated starting from $G_0$, by introducing noise, i.e., adding $N_{\text{noise}}$ new randomly chosen edges that connect the existing vertices (Fig. 2b). The so-constructed graph $G$ will always admit $G_0$ as solution, even though additional solutions might also appear when the amount of noise is large.

(ii) The graph $G$ is then transformed into a QUBO problem as explained in the previous sections and submitted to the quantum annealer. The annealing schedule is run 100 times, and the frequency of the final states obtained is computed.

(iii) Using Algorithm 1 we check if the lowest energy state (or states in the degenerate case) corresponds to a partition of $G$ into Hamiltonian subgraphs containing cycles of length three or more. If that is the case, the probability $P_{\text{sol}}$ of finding a solution is equal to frequency of the lowest energy state (or the sum of the frequencies of the lowest energy states in the degenerate case), if not $P_{\text{sol}} = 0$. We note that the solution check algorithm runs in polynomial time. The only step that is proportional to the size of the problem is line 6 in Algorithm 1, which is $O(N_E)$.

**Algorithm 1 Solution check.** The pseudocode describes the function used to verify whether a graph $G' = (V,E')$, with $E' \subseteq E$, is a partition of a graph $G = (V,E)$ into Hamiltonian subgraphs with three or more vertices.

**Input:** Graphs $G' = (V,E')$ and $G = (V,E)$

**Output:** True if $G'$ is a partition of $G$ into Hamiltonian subgraphs with three or more vertices, False otherwise

**Procedure:**

1: repeat
2: Initialise two empty sets $C_V = \{\}$ and $C_E = \{\}$
3: Pick a vertex $v_{\text{start}} \in V$
4: Assign $v_{\text{from}} := v_{\text{start}}$
5: repeat
6: Find the edges $E_{\text{from}} \subseteq E'$ going out of $v_{\text{from}}$
7: if $E_{\text{from}}$ has exactly one element then
8: Add the vertex $v_{\text{from}}$ to $C_V$
9: Add the edge $e$ in $E_{\text{from}}$ to $C_E$
10: else
11: return False
12: end if
13: Assign $v_\ell$ to the vertex pointed at by $e$
14: if ($v_\ell \neq v_{\text{start}}$) and ($v_\ell \in C_V$ or $C_E = E'$) then
15: return False
16: end if
17: Assign $v_{\text{from}} := v_\ell$
18: until $v_\ell = v_{\text{start}}$
19: if $C_V$ has exactly 2 elements then
20: return False
21: else if $C_V$ has 3 elements or more then
22: Remove from $V$ the vertices contained in $C_V$
23: Remove from $E'$ the edges contained in $C_E$
24: end if
25: until no vertices are left in $V$ and no edges are left in $E'$
26: return True
FIG. 3. Single-run solution probability as a function of the number of vertices. (a) $P_{\text{sol}}$ is shown as a function of $N_V$ for cycles of length 4 for different $p_{\text{noise}}$. For all curves the probability to find a solution in a single run on the quantum annealer decreases as the system size increases. With noise, the dependence on the system size is stronger: for $N_V = 4200$ and $p_{\text{noise}} = 0.5 \times 10^{-4}$ (i.e. $N_{\text{noise}} = 882$) the probability is $P_{\text{sol}} = 0.104(9)$, and for $N_V = 3600$ and $p_{\text{noise}} = 1 \times 10^{-4}$ (i.e. $N_{\text{noise}} = 1295$), the probability is $P_{\text{sol}} = 0.055(8)$. (b) $P_{\text{sol}}$ is shown as a function of $N_V$ for $p_{\text{noise}} = 0.5 \times 10^{-4}$ for different cycle lengths. The probability exhibits the same behaviour as a function of the problem size regardless of the cycle length used in the input graph configuration. In both panels, the error bars show the standard error on the mean.

To collect statistics on $P_{\text{sol}}$, we repeat the steps (i) to (iii) 50 times and average $P_{\text{sol}}$ over the 50 repetitions to obtain $P_{\text{sol}}$, that represents the average probability to find a solution with a single run (i.e. a single annealing schedule) on the quantum annealer.

V. RESULTS

We start by fixing the fraction $p_{\text{noise}}$ of the maximum allowed number of additional edges for the given number of vertices $N_V$, and set $N_{\text{noise}} = \text{round}(p_{\text{noise}} N_V (N_V - 2))$. In the simplest scenario, i.e. $p_{\text{noise}} = 0$, the quantum annealer easily finds a solution regardless of the problem size (grey points in Fig. 3a). Even when considering a problem with a very large number of vertices and edges, $N_V = N_E = 5400$, where the graph size is very close to the total number of working physical qubits available (5436), the probability to find a solution with a single run on the quantum annealer is $P_{\text{sol}} = 75(1)\%$. We observe that $P_{\text{sol}}$ decreases with $N_V$ with a slope that strongly depends on $p_{\text{noise}}$, and the quantum annealer finds a solution up to $N_V = 4200$ for $p_{\text{noise}} = 0.5 \times 10^{-4}$ and up to $N_V = 3600$ for $p_{\text{noise}} = 1 \times 10^{-4}$ (blue and orange points in Fig. 3a).

Figure 3b shows that the single-run solution probability $P_{\text{sol}}$ does not depend on the cycle length. This can be explained by the fact that the dimension of the combinatorial space depends on the number of all the possible paths in the input graph, which is determined only by $N_V$ and $p_{\text{noise}}$.

To further investigate the dependence on the combinatorial complexity, we now fix the number of vertices $N_V$ and vary the number of added edges $N_{\text{noise}}$. To give an intuition on how the complexity of the problem scales with $N_{\text{noise}}$, we can think of a classical approach based on Algorithm 1. Without any edges added, there is only one possible simple path. When we add one edge of noise, there will be one vertex with two outgoing edges: this bifurcation gives rise to two different simple paths. Each of them could be then fed into Algorithm 1 to check whether it is a solution. However, when the number of added edges increases, the number of simple paths to be checked can be up to $2^{N_{\text{noise}}}$, and therefore finding a solution with this classical procedure becomes exponentially hard.

In Fig. 4 we show the results obtained with the quantum annealer. For a small amount of added edges ($N_{\text{noise}} \lesssim 100$) the single-run solution probability is higher for the smaller system size explored (Fig. 4a). However, as $N_{\text{noise}}$ is increased, for a given number of added edges it becomes easier to find a solution for a system where the size is larger. We note that, regardless of the value of the single-run solution probability, the same problem can be submitted multiple times in order to find a solution at least once with arbitrarily high probability. Fixing the desired probability to 99%, the time needed to find a solution is given by [22]

$$TTS = (t_{\text{anneal}} + t_{\text{pause}}) \frac{\log(1 - 0.99)}{\log(1 - P_{\text{sol}})},$$

where the first term is the single-run time (i.e. the sum of the times used in the schedule of the quantum annealer.
For the exponential fit, the parameters obtained from the fits to the data shown in Fig. 4b.

| Fit parameters          | $a$  | $b$  |
|-------------------------|------|------|
| power law fit $N_V = 1000$ | -34(3) | 5.9(6) |
| power law fit $N_V = 4000$ | -15(1) | 2.6(2) |
| exponential fit $N_V = 1000$ | -3.8(2) | 0.0120(4) |
| exponential fit $N_V = 4000$ | -0.8(3) | 0.0035(3) |

**TABLE I.** Fit parameters. The table reports the parameters obtained from the fits to the data shown in Fig. 4b. For the exponential fit, the parameter $b$ is much less than log 2, expected for the $2^{N_{\text{noise}}}$ classical scaling described in section V.

$t_{\text{anneal}} = 200 \mu s$ and $t_{\text{pause}} = 100 \mu s$ and the second term is the number of necessary runs to find a solution with the desired probability.

In Fig. 4b we show the behaviour of the time to solution TTS as a function of the number of added random edges. We fit the large $N_{\text{noise}}$ behaviour of TTS with an exponential and power law functions. The fit parameters are reported in Table I. It is clear from the plots that for the range explored TTS is compatible with either fit. However, even in the exponential case, the scaling is much slower than the one of the classical procedure explained earlier, that scales as $\exp (N_{\text{noise}} \log 2)$.

**VI. OUTLOOK**

Possible extensions of the problem presented here can be considered for graphs where weights are assigned to the edges, or where different constraints on the cycle length are present.

We point out that, if self-loops are included in the construction of the problem and the constraint on the cycle length is lifted, our method could be used to compute the permanent of a graph is equivalent to computing the permanent of its adjacency matrix [23].

**Additional information:** The code used to generate the results presented in this paper is available at [https://github.com/quantumglare/quantum_cycle](https://github.com/quantumglare/quantum_cycle). Further information can be requested at info@quantumglare.com.

**Appendix A: Constraints**

In order to properly set the penalty constants in Eqs. (9)-(11) we proceed with an analysis of the different cost terms in Eq. (6), that allows us to choose the penalty constants as small as possible.

We require the cost $J(x_a)$ of an allowed configuration $x_a$ that satisfies all constraints to be lower than the cost $J(x)$ of any configuration $x$ that violates at least one constraint, i.e.

$$J(x) > J(x_a) \quad \forall x, x_a.$$  \hspace{1cm} (A1)

Let us first consider the constraint on the number of outgoing edges given in Eq. (3), whose corresponding penalty is given in Eq. (9). Any configuration that violates only that constraint can be decomposed as $x = x_a + x'$, where $x'$ is a vector whose only elements equal to 1 are those corresponding to the additional edges. From equation (A1) it follows that, for every vertex $i$,

$$a_i \sum_{j,j' \geq j} x_{ij}x_{ij'} > \sum_J x'_{ij},$$  \hspace{1cm} (A2)

Equation (A2) is satisfied by setting:

$$a_i = \max_{x,x'} \left( \frac{\sum_{j,j' \geq j} x_{ij}x_{ij'}}{\sum_{j,j' \geq j} x_{ij}x_{ij'}} + \epsilon \right),$$  \hspace{1cm} (A3)

where $\epsilon$ is an arbitrarily small positive constant, which makes $a_i$ an optimal choice. The number $N_{x',i}$ of non-zero elements in $x'$ varies from 1 to the total number $N_{\text{out},i}$ of outgoing edges from vertex $i$ in the original graph. In terms of $N_{x',i}$ Eq. (A3) becomes

$$a_i = \max_{N_{x',i}} \left( \frac{N_{x',i}}{(N_{x',i}+1)2} + \epsilon \right)$$  \hspace{1cm} (A4)

where in the denominator the round brackets denote the binomial coefficient. The maximum is achieved for $N_{x',i} = 1$, giving $a_i = 1 + \epsilon$ for all vertices $i$ of the graph that might violate the constraint. For all other vertices we simply set it to zero, leading to

$$a_i = \begin{cases} 1 + \epsilon & \text{if } N_{\text{out},i} > 1, \\ 0 & \text{otherwise.} \end{cases}$$  \hspace{1cm} (A5)

Likewise, for the constraint on the number of ingoing edges in Eq. (4), which corresponds to the penalty in Eq. (10), we set

$$b_i = \begin{cases} 1 + \epsilon & \text{if } N_{\text{in},i} > 1, \\ 0 & \text{otherwise,} \end{cases}$$  \hspace{1cm} (A6)

where $N_{\text{in},i}$ is the number of ingoing edges to vertex $i$ in the original graph.

A similar reasoning for the constraint to forbid pairs in Eq. (5), which corresponds to the penalty in Eq. (11), leads to

$$c = 2 + \epsilon.$$  \hspace{1cm} (A7)
[1] D. S. J. Michael R. Garey, Computer and intractability: a guide to the theory of NP-completeness, A Series of books in the mathematical sciences (W. H. Freeman and Co Ltd, 1979).
[2] D. Venturelli, D. J. J. Marchand, and G. Rojo, arXiv e-prints (2015), 1506.08479 [quant-ph].
[3] F. Neukart, G. Compostella, C. Seidel, D. von Dollen, S. Yarkoni, and B. Parney, arXiv e-prints (2017), arXiv:1708.01625 [quant-ph].
[4] T. Stollenwerk, B. O’Gorman, D. Venturelli, S. Mandrà, O. Rodionova, H. K. Ng, B. Sridhar, E. G. Rieffel, and R. Biswas, arXiv e-prints (2017), arXiv:1711.04889 [quant-ph].
[5] D. Inoue, A. Okada, T. Matsumori, K. Aihara, and H. Yoshida, arXiv e-prints (2020), arXiv:2003.07527 [math.OC].
[6] T. Stollenwerk, E. Lobe, and M. Jung, arXiv e-prints (2018), arXiv:1811.09465 [quant-ph].
[7] G. Bass, C. Tomlin, V. Kumar, P. Rihaczek, and I. Dulny, Joseph, Quantum Science and Technology 3, 024010 (2018), arXiv:1709.05381 [quant-ph].
[8] A. Perdomo-Ortiz, N. Dickson, M. Drew-Brook, G. Rose, and A. Aspuru-Guzik, Scientific Reports 2, 571 (2012).
[9] D. Venturelli and A. Kondratyev, Quantum Machine Intelligence 1, 17 (2019).
[10] J. Cohen, A. Khan, and C. Alexander, arXiv e-prints (2020), arXiv:2007.01430 [q-fin.GN].
[11] Y. Ding, L. Lamata, J. D. Martín-Guerrero, E. Lizaso, S. Mugel, X. Chen, R. Orús, E. Solano, and M. Sanz, “Towards prediction of financial crashes with a d-wave quantum computer,” (2019), arXiv:1904.05808 [quant-ph].
[12] M. Constantino, X. Klimentova, A. Viana, and A. Rais, European Journal of Operational Research 231, 57 (2013).
[13] S. Luo and P. Tang, in Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI’15 (AAAI Press, 2015) pp. 209–215.
[14] R. Anderson, I. Ashlagi, D. Gamarnik, and A. E. Roth, Proceedings of the National Academy of Sciences 112, 663 (2015).
[15] A. Abdulkadiroglu and T. Sönmez, Journal of Economic Theory 88, 233 (1999).
[16] T. Sönmez and T. B. Switzer, Econometrica 81, 451 (2013).
[17] W. Fang, P. Tang, and S. Zuo, in Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016, edited by S. Kambhampati (IJCAI/AAAI Press, 2016) pp. 264–270.
[18] M. Bläser and B. Siebert, in Algorithms — ESA 2001, edited by F. M. auf der Heide (Springer Berlin Heidelberg, Berlin, Heidelberg, 2001) pp. 368–379.
[19] J. Edmonds and E. L. Johnson, “Matching: A well-solved class of integer linear programs,” in Combinatorial Optimization — Eureka, You Shrink!: Papers Dedicated to Jack Edmonds 5th International Workshop Aussois, France, March 5–9, 2001 Revised Papers, edited by M. Jünger, G. Reinelt, and G. Rinaldi (Springer Berlin Heidelberg, Berlin, Heidelberg, 2003) pp. 27–30.
[20] F. Glover, G. Kochenberger, and Y. Du, “A tuto-