International Journal of Information Technology & Decision Making
Vol. 21, No. 1 (2022) 59–81
© World Scientific Publishing Company
DOI: 10.1142/S0219622021500504

Solving Clique Partitioning Problems: A Comparison of Models and Commercial Solvers

Yu Du[*,‖], Gary Kochenberger[*], Fred Glover[†], Haibo Wang[‡], Mark Lewis[§], Weihong Xie[¶] and Takeshi Tsuyuguchi[‡]

[*]*University of Colorado at Denver, USA*

[†]*University of Colorado at Boulder, USA*

[‡]*Texas A&M International University, USA*

[§]*Missouri State University, USA*

[¶]*Guangdong University of Technology, P. R. China*

[‖]*yu.du@ucdenver.edu*

Received 16 October 2020
Revised 10 June 2021
Accepted 10 June 2021
Published 11 August 2021

Finding good solutions to clique partitioning problems remains a computational challenge. With rare exceptions, finding optimal solutions for all but small instances is not practically possible. However, choosing the most appropriate modeling structure can have a huge impact on what is practical to obtain from exact solvers within a reasonable amount of run time. Commercial solvers have improved tremendously in recent years and the combination of the right solver and the right model can significantly increase our ability to compute acceptable solutions to modest-sized problems with solvers like CPLEX, GUROBI and XPRESS. In this paper, we explore and compare the use of three commercial solvers on modest sized test problems for clique partitioning. For each problem instance, a conventional linear model from the literature and a relatively new quadratic model are compared. Extensive computational experience indicates that the quadratic model outperforms the classic linear model as problem size grows.

Keywords: Clique partitioning; combinatorial optimization; quadratic integer programming.

1. Introduction

Consider a graph $G = (V, E)$ with n vertices and unrestricted edge weights. The clique partitioning problem (CPP) consists of partitioning the graph into cliques such that the sum of the edge weights over all cliques formed is as large as possible. This is an NP-hard problem with applications reported in such diverse areas as VLSI layout design[1–3] (MacGregor, 1978),[31] program design for paged computer memory[1,4] (MacGregor, 1978),[31] group technology analysis[5] (Oosten, 2001), image analysis,[6] and cluster analysis.[7,8] Related to cluster analysis, and with strong connections

[‖]Corresponding author.

to machine learning, several authors have proposed using clique partitioning models for community detection in networks. For these and related discussions see the papers by Agarwal and Kempe,[9] Ailon *et al.*,[10] Alosie *et al.*, 2010, Bruckner *et al.*,[11] Dinh and Thai,[12] Miyauchi and Sukegawa,[13] Berg and Jarvisalo,[14] and Miyauchi *et al.*[15] In addition, Ref. 16. reported modeling the airport gate scheduling as a CPP. We comment here that the formation of alliances among countries as well as strategic alliances and coalitions among companies as discussed in the work of Axelrod *et al.*,[17] can also be modeled and analyzed as a CPP.

The main objective of this work is to test alternative models to see if the model form, independent of the solver employed, makes a difference in finding good if not optimal solutions for CPPs as considered here. Specifically, we test and compare a quadratic alternative to the standard linear model for clique partitioning found in the literature.

Most of the literature on solving CPP focuses on heuristic methods due to the computational burden that CPP poses. Nonetheless, some work on exact methods has been reported. Exact methods such as that proposed by Jaehn and Pesch,[18] have performed well on modest-sized problems with certain characteristics. Aloise *et al.*,[19] discuss special column and row generation methods for optimally solving clique partitioning models for community detection in networks. These methods proved to be successful for certain small to medium-sized problems. In contrast to such specially crafted methods, our focus here is on commercial methods, as represented by products such as CPLEX, Gurobi, and Xpress, which have improved substantially in recent years and offer a readily available alternative to special specially crafted methods.

Another of our objectives in this work is to test these three exact solvers to see how they perform relative to each other and to understand what is currently possible in terms of finding optimal solutions for modest sized CPPs of the type considered in this study.

The sections below present the models we are comparing along with a description of the test problems and a discussion of the results obtained. The three exact solvers (CPLEX, Gurobi and Xpress) are launched via the AMPL platform. All our AMPL models and our test problems are available from the authors.

2. Basic Clique Partitioning: Alternative Models

We first describe the standard linear model for clique partitioning and then present the quadratic alternative model we will be testing. For each model, we compare the performance of CPLEX, Gurobi, and XPRESS on a set of test problems. Finally, we compare the two competing models and identify the model/solver combination that provided the best performance over the test bed considered.

2.1. *Standard linear model*

The standard model for clique partitioning from the literature (Oosten, 2001) and Ref. 20 is a large 0/1 linear program we denote here as Model 1.

$$\text{Model 1} : \max x_0 = \sum_{i,j>i} w_{ij} x_{ij} \tag{1}$$

s.t.

$$\begin{aligned} x_{ij} + x_{jk} - x_{ik} &\leq 1 \quad 1 \leq i < j < k \leq n, \\ x_{ij} - x_{jk} + x_{ik} &\leq 1 \quad 1 \leq i < j < k \leq n, \\ -x_{ij} + x_{jk} + x_{ik} &\leq 1 \quad 1 \leq i < j < k \leq n, \end{aligned} \tag{2}$$

where $x_{ij} = 1$ if nodes i and j are in the same clique, n is the number of nodes in the graph, and the w_{ij} are unrestricted edge weights. The constraints of (2) are the well-known clique inequality constraints and they ensure that the nodes are in fact partitioned into cliques. Note that this is an edge-based model and thus even modest-sized problems will have many variables and constraints. For complete graphs, Model 1 will have $n(n-1)/2$ variables and $3\binom{n}{3}$ constraints. We comment that if we start with a graph that is not complete, edges can be added as needed, with penalty edge weights, to produce a complete graph. The penalty edge weights are chosen such that the optimizer will never choose the new edges.

As an example, consider the graph shown in Fig. 1.

Model 1 has 10 variables and 30 constraints. Solving the model gives the solution $x_0 = 31$ with $x_{13} = x_{14} = x_{34} = 1, x_{25} = 1$. Thus, we have two cliques, one with nodes 1, 3, and 4 and the other with nodes 2 and 5. In the graph above, edges (1,2), (1,5), and (3,5) illustrate the addition of new edges with penalty edge weights required to produce a complete graph.

2.2. *A quadratic alternative model*

Model 1 above is the standard model for clique partitioning from the literature. An alternative quadratic model can be developed as follows: Associating variables with nodes instead of edges, we obtain the following model, which we will refer to as Model 2:

$$\text{Model 2} : \max x_0 = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_{ij} \sum_{k=1}^{k\max} x_{ik} x_{jk} \tag{3}$$

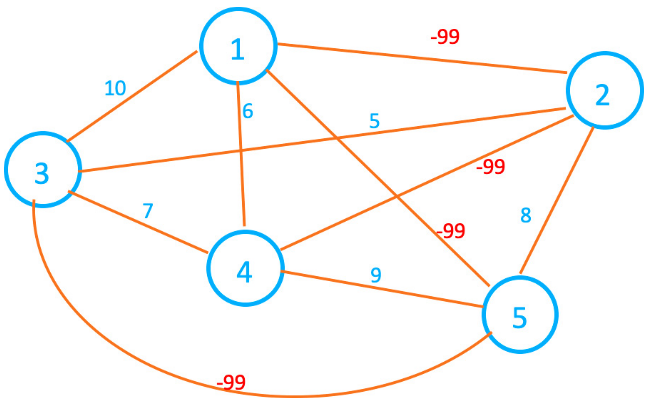

Fig. 1. A small example for demonstration.

s.t.

$$\sum_{k=1}^{k\max} x_{ik} = 1 \quad i = 1, n, \tag{4}$$

where $x_{ik} = 1$ if node i is assigned to clique k, 0 otherwise, and $k_{\max}$ = the maximum number of cliques allowed, some of which might prove to be empty. Here, the constraints require that each node is assigned to one of the cliques that are formed. As with Model 1, the objective function sums up the weights over all of the cliques formed, which we seek to maximize. The parameter $k_{\max}$ is set based on domain knowledge and/or an educated guess.

Since a solution to Model 2 assigns each node to a clique, it also corresponds to a solution for Model 1. It follows that for the extreme case when $k_{\max} = n$, an optimal solution to Model 2 corresponds to an optimal solution for Model 1. That is, the models have equivalent solutions when $k_{\max} = n$. Based on extensive computational experience, however, we find that Model 1 and Model 2 have corresponding optimal solutions for $k_{\max}$ values much smaller than n, leading us to comment in general that Model 1 and Model 2 are equivalent for a sufficiently large $k_{\max} \leq n$. Guided by domain knowledge, the issue of setting $k_{\max}$ has not been a problem in any of the testing we have undertaken. Further discussion about the relationship of Model 1 and Model 2 can be found in Appendix A.

It is clear that the choice of $k_{\max}$ can have major impact on the performance of Model 2. If $k_{\max}$ is set lower than the optimal number of cliques, the optimal solution derived from Model 2 will be sub-optimal compared to the optimal solution to Model 1. If $k_{\max}$ is set equal to or great than the optimal number of cliques needed, the optimal solutions derived from the two models will be equivalent. However, the computational advantage derived from Model 2 decreases rapidly as $k_{\max}$ gets unnecessarily large compared to the optimal number of cliques. An example illustrating the impact of $k_{\max}$ on Model 2 performance can be found in Appendix B.

Since our decision variables in Model 2 are node-based rather than edge-based as in Model 1, Model 2 typically has many fewer variables (and many fewer constraints) than Model 1. As problem size grows, the discrepancy between Models 1 and 2 in terms of number of variables and number of constraints grows rapidly to increasingly favor the quadratic model on both dimensions.

For the very small example considered above, we observe that the size difference is negligible. If we take $k_{\max}$ to be three, meaning that we allow for as many as three cliques to be formed, Model 2 has 15 variables and 5 constraints. Solving this quadratic model yields the same result as we obtained via Model 1 above: $x_0 = 31$ with $x_{11} = x_{31} = x_{41} = 1, x_{22} = x_{52} = 1$ showing that nodes 1, 3, and 4 are assigned to clique #1 and nodes 2 and 5 are assigned to clique # 2. Potential clique number 3 is empty. (Recall that the variables, x_{ij}, have different definitions in the two models.)

We comment that in earlier work, the authors have employed Model 2 with specially crafted heuristic solution methods for solving certain clustering problems, some with special properties. See for instance Refs. 5 and 8. Our focus in this paper is to test the performance of exact commercial solvers on the two models, as opposed to heuristic methods, on CPPs of varying structures and sizes.

To test the alternative models and solvers, three sets of test problems were employed, one set of new randomly generated problems and two sets from the literature referenced as follows:

- Set 1: A set of randomly generated test problems
- Set 2: A set of test problems from the paper by Jaehn and Pesch,[18]
- Set 3: A set of problems from the paper by Aloise *et al.*,[19]

All the results reported in the following tables came from a Dell PC with 2.4 GHz and 16 GB RAM. All computations were carried out using the AMPL modeling and solution system.

3. Computational Experiments Regarding Problem Set 1

For our initial testing, we generated a set of random problems of size 25, 35, 45, 50, 65 and 100 nodes. All test problems correspond to complete graphs with unrestricted edge weights. Eighty percent of the edge weights were randomly generated from $U(0,100)$ and 20% of the edges were given edge weights equal to -99. For each problem size, four instances were generated, giving a total of 24 basic problems which formed the core of our computational testing for this section.

3.1. *Comparing the exact solvers on Model 1*

Table 1 presents the results obtained from Model 1 from each of the exact solvers Gurobi version 7.5.0, CPLEX version 12.7.1 and XPRESS version 31.01. In all cases, the default settings were used. For each problem, the table gives the best solution found by each solver, along with the percent difference between a given result and the best of the three, along with the time to best for each. All times are in seconds and with the exception of Table 2, all results presented are based on a time limit of 1 h for each problem. Looking at the CPLEX results for problem 45-1, for example, an objective function value of 10,822, the best value the solver found within 3600 s, was found at time 2532.

Examining Table 1, we see that all three solvers were quickly able to find optimal solutions for the four 25 node problems. However, XPRESS was not able to prove optimality for any of the four problems. CPLEX proved optimality for three of the four problems while Gurobi proved optimality for all four problems. Likewise, for the $n = 35$ node problems, all three solvers found the optimal solution within the allotted time of 1 h. Both Gurobi and CPLEX proved optimality for these four problems. XPRESS found the optimal solutions but failed to establish optimality for any of the four problems in the time allotted.

For the $n = 45$, $n = 50$, $n = 65$ and $n = 100$ node problems, none of the solvers were able to prove optimality in the allotted time limit of 1 h. Looking across the

Table 1. Model 1 results (1 h time limit for each problem).

Model 1	Gurobi			Cplex			Xpress		
ID	Objective	Diff (%)	Time	Objective	Diff (%)	Time	Objective	Diff (%)	Time
25-1	4535	0%	8	4636	0%	12	4636	0%	10
25-2	4023	0%	3	4023	0%	12	4023	0%	6
25-3	5043	0%	10	5043	0%	10	5043	0%	11
25-4	4554	0%	11	4564	0%	10	4564	0%	11
35-1	7837	0%	496	7837	0%	497	7837	0%	682
35-2	7215	0%	403	7215	0%	661	7215	0%	448
35-3	7633	0%	95	7633	0%	398	7633	0%	101
35-4	7652	0%	292	7652	0%	278	7652	0%	404
45-1	11545	0%	1305	10882	−6%	2532	9415	−18%	474
45-2	12137	0%	2492	11903	−2%	2464	10497	−14%	465
45-3	11672	0%	2325	11521	−1%	2500	10100	−13%	674
45-4	10338	0%	2615	10080	−2%	2411	9919	−4%	451
50-1	12373	0%	3173	11200	−9%	2000	9840	−20%	1038
50-2	13543	0%	3011	12172	−10%	2405	11724	−13%	999
50-3	12191	0%	2553	10935	−10%	1856	11024	−10%	127
50-4	13009	0%	2544	12242	−6%	2141	11500	−12%	121
65-1	11073	−30%	3399	15865	0%	3600	14978	−6%	170
65-2	11273	−28%	3348	15764	0%	3600	12773	−19%	41
65-3	11298	−23%	3336	14652	0%	3600	10468	−29%	144
65-4	12351	−17%	3550	14811	0%	3600	13081	−12%	145
100-1	16809	−18%	18	20387	0%	3600	18395	−10%	1220
100-2	15456	−22%	648	12668	−36%	3600	19742	0%	829
100-3	15749	−24%	17	11345	−45%	3600	20760	0%	565
100-4	12738	−24%	836	7861	−53%	3600	16809	0%	236

entire test bed of 24 problems, Gurobi produced the best solution (of those found by the three methods) 16 out of the 24 times, CPLEX produced the best solution 13 times, and XPRESS matched the best solution 11 times. It is interesting to note that while XPRESS lagged behind Gurobi and CPLEX in terms of number of best solutions found across all 24 problems, its relative performance picked up with increasing problem size and in fact XPRESS produced 3 of the best solutions found for the $n = 100$ node problems, outperforming Gurobi and CPLEX by a substantial

Table 2. Longer Gurobi runs on Model 1.

Problem ID	Objective	Time	Gap (Extended runs)	Gap (at 1 h)
45-1	11,545	1,305	0.52 % (8 h)	5.7%
45-2	12,137	2,492	0.74% (8 h)	1.95%
45-3	11,880	7,595	0.09% (8 h)	7.8%
45-4	10,506	4,280	1.35% (8 h)	8.3%
50-1	13,453	46,616	6.21% (24 h)	17.6%
50-2	14,080	18,831	3.54% (24 h)	17.4%
50-3	13,034	50,630	7.45% (24 h)	21.3%
50-4	13,728	69,040	7.03% (24 h)	35.4%

margin on these largest problems in the time allotted. However, the results produced by all three solvers on the $n = 65$ and $n = 100$ node problems are far from optimal as revealed in Table 3 presented later in this section.

Note that a time of 3600 in the tables denotes that the algorithm ran into the time limit before moving beyond the root node calculations.

Based on the results given in Table 1 for Model 1, Gurobi gave the best overall performance, followed in order by CPLEX and then XPRESS. As expected, the difficulty of proving optimality can be mitigated by going to longer run times. Table 2, for instance, shows the results of running Gurobi using Model 1 on the $n = 45$ node problems for an extended time period of 8 h and on the $n = 50$ node problems for an extended limit of 24 h. For reference, we also show the optimality gaps reported at the end of the 1 h runs.

Looking at the Table 2 results for the 45 node problems, we see that increasing the time limit from 1 h to 8 h substantially reduced the final gaps while modestly improving the objective function values, as shown in Table 1, for problems 45-3 and 45-4. The objective function values for 45-1 and 45-2 were not changed by increasing the run times. The improved gaps for all 4 of the 45 node problems suggest that the results shown in Table 2 are most likely optimal. Longer runs for a limit of 16 h for these four problems confirmed that this is the case. Nevertheless, we emphasize that these are relatively small problems.

The results shown in Table 2 for the 50 node problems, where the time limit was raised from 1 h to 24 h, indicating how difficult these problems are to solve to optimality. At the end of 24 h of computational time, the objective function values for all four problems have improved modestly over the previous 1 h results and the final gaps, while much improved, suggest that the solutions in the table are likely of high quality but not necessarily optimal. We note that the results reported in Table 2, obtained from Gurobi, are illustrative of the behavior of the other solvers as well.

The difficulty in solving these problems to optimality is not unexpected due to the size of Model 1 as we go to larger graphs. Branch and Cut methodologies, while greatly improved, are nonetheless challenged to find good solutions from Model 1 as the size of the graphs being analyzed grows.

3.2. *Comparing exact solvers on Model 2*

Table 3 presents the results obtained from running the solvers on the quadratic formulation of Model 2, the alternative to Model 1. For the $n = 25$, $n = 35$, and $n = 45$ node problems, $k_{\max}$ was set to 7. A value of $k_{\max} = 10$ was used for the larger problems with $n > 45$. In all cases, these values for $k_{\max}$ proved to be more than sufficient to provide high-quality solutions.

All three solvers have the capability, in principle, of solving quadratic models of the type represented by Model 2. In fact, both Gurobi and CPLEX offer two options: Solve the quadratic problem by first linearizing the quadratic objective function, converting the problem to a large MIP; or, solve without linearizing by employing

Table 3. Results from Model 2 (1 h time limit).

Model 2	Gurobi (Lin)			Gurobi (Quad)			Cplex (Lin)			Cplex (Quad)			Xpress		
ID	Objective	Diff (%)	Time	Objective	Diff (%)	Time	Objective	Diff (%)	Time	Objective	Diff (%)	Time	Objective	Diff (%)	Time
25-1	4636	0%	50	4501	−3%	30	4636	0%	35	4636	0%	3239	4636	0%	33
25-2	4023	0%	56	4023	0%	1590	4023	0%	25	4023	0%	18	4023	0%	43
25-3	5043	0%	87	4931	−2%	1572	5043	0%	37	5043	0%	265	5043	0%	90
25-4	4564	0%	36	4481	−2%	3	4564	0%	21	4564	0%	9	4106	−10%	17
35-1	7837	0%	2911	7629	−3%	10	7837	0%	1200	7837	0%	3300	7464	−5%	3265
35-2	7215	0%	1405	6917	−4%	105	7215	0%	1700	7215	0%	6	7135	−1%	2154
35-3	7633	0%	135	7633	0%	141	7377	−3%	510	7633	0%	25	6776	−11%	3045
35-4	7652	0%	1445	7301	−5%	20	7652	0%	253	7652	0%	21	7409	−3%	327
45-1	10972	−5%	1858	10470	−9%	117	10898	−6%	3198	11542	0%	3583	9823	−15%	1675
45-2	11292	−7%	1778	11796	−3%	552	11202	−8%	618	12137	0%	3600	10461	−4%	1019
45-3	11689	−1%	485	11759	0%	130	11595	−2%	3496	11796	0%	90	9836	−16%	3530
45-4	9715	−7%	221	10030	−4%	1182	9960	−5%	1140	10474	0%	3347	8158	−22%	1693
50-1	12757	−3%	79	12966	−1%	215	13132	0%	3331	13107	0%	189	11768	−10%	2030
50-2	13526	−4%	836	13247	−6%	38	13258	−6%	1825	14056	0%	3410	11411	−19%	1769
50-3	12507	−2%	2420	12125	−5%	36	11935	−6%	15	12762	0%	3407	10022	−21%	986
50-4	12649	−5%	614	12985	−3%	35	12395	−7%	3151	13323	0%	149	11116	−17%	2279
65-1	17697	−8%	1812	17932	−6%	840	16007	−16%	3436	19156	0%	302	9373	−51%	104
65-2	18657	−4%	1689	18149	−7%	702	15784	−19%	3600	19413	0%	1493	1481	−92%	2214
65-3	18088	−6%	2915	18458	−4%	968	16852	−13%	2139	19286	0%	3521	−6173	−132%	317
65-4	17133	−11%	2079	18547	−4%	1421	16140	−16%	1175	19285	0%	3461	12012	−38%	3109
100-1	30828	−8%	2197	31995	−5%	3069	27846	−17%	357	33588	0%	3578	−313469	−1033%	43
100-2	30401	−11%	2012	33558	−2%	2034	24655	−28%	117	34157	0%	3475	−421905	−1335%	13
100-3	31445	−12%	3600	35602	0%	3188	29494	−17%	121	35424	0%	3600	−234985	−763%	51
100-4	28557	−17%	2216	34416	0%	3351	24605	−29%	118	34293	0%	1889	−351948	−1126%	60

continuous quadratic relaxations. XPRESS, in contrast, does not offer the "quadratic relaxation" option and only proceeds by first linearizing the problem. The linearizations used by CPLEX and Gurobi are based on the standard technique of replacing a product term with a new binary variable and adding linear constraints to ensure the new linear representation is equivalent to the original quadratic formulation (see Glover and Woolsey, 1974).

In keeping with these options, we denote, in Table 3, the Gurobi options by Gurobi (Lin) and Gurobi (Quad). Similar designations are used for CPLEX. All the other parameters are left as default. All told, then, we have 5 solutions to report for each problem as shown in the table.

An examination of Table 3 shows that Gurobi (Lin), CPLEX (Lin), and CPLEX (Quad) did well on the $n = 25$ and $n = 35$ node problems. Gurobi (Lin) and CPLEX (Quad) found the same best solutions on all eight problems. CPLEX (Lin) was close behind with seven best solutions. For these two problem sets, XPRESS was only able to match the other solvers on three of the eight problems. Moving on to the larger problems in Table 3, we see that CPLEX (Quad) completely dominates the other 4 solvers, finding best solutions on 14 of the remaining 16 problems.

Across the entire set of 24 problems, CPLEX (Quad) consistently performed well, producing best solutions for 22 of the 24 problems in the allotted time of 1 h. In no case, however, did it or any other method terminate naturally for the runs depicted in Table 3. On each problem, each solver terminated with an "out of time" message, being unable to prove optimality and terminate naturally within the allotted time limit of 1 h.

3.3. *Comparing Models 1 and 2*

Comparing the results from Model 1 (Table 1) and those from Model 2 (Table 3), we see that Model 1 and Model 2 results, across all solvers, are fairly close in quality for the $n = 25$, $n = 35$, and $n = 45$ node problems. For the larger problems, however, several of the Model 2 solvers found solutions much superior to the corresponding results produced from Model 1. For example, for the problem instance 100-1, the best solution produced from Model 1 (see Table 1) was 16809 while all 5 solutions obtained from Model 2 were substantially better than this and the solution obtained from Model 2 via CPLEX (Quad) was 34,293, more than twice the best Model 1 result.

Looking across all 24 problems and all 8 solvers represented in Tables 1 and 3, we see that Model 2 solved by CPLEX (Quad) clearly dominates the others, finding best solutions for 21 out of the 24 problems. None of the other model/solver combinations comes close to this performance. All told, Model 2 clearly dominates Model 1 in terms of producing good solutions as the graphs scale in size.

The dominant performance of Model 2 as problem size scales is in large part explained by the vast difference in size between Model 1 and Model 2. As noted previously, a problem with 100 nodes produces an instance of Model 1 with 4,950 variables and 485,100 constraints while Model 2 on the same problem and $k_{\max} = 10$ has 1,000 variables and 100 constraints.

It is worth noting that the use of Model 2, particularly with the quadratic solutions option, often produced good solutions fairly quickly and then made slow progress toward closing the optimality gap. For example, for Model 2 and problem 100-4, CPLEX (Quad) produced the solution of 34,293 within the 1 h time limit as shown in Table 3.

4. Computational Experiments Regarding Problem Set 2

The paper by Jaehn and Pesch,[18] presented computational experience on additional CPPs of varying structures, including clustering problems originally due to Ref. 21, and group technology problems due to Ref. 22. Since these problems are fairly modest in size, we included, as part of problem set # 2, three large group technology problems due to Ref. 5. Our computational experience with these problems is presented in the following Tables 4–6. Note in Table 4, the first six problems (wild cats down to UNO2a) are clustering problems. The remaining problems are group technology applications.

Results of our testing are shown in Table 4 for the linear model (Model 1) using all three solvers and in Table 5 for Model 2, the quadratic alternative model. For each problem and solution method, a time limit of one hour was imposed.

Examining Table 4, we see that, with one exception, all three solvers were able to quickly terminate naturally with optimal solutions for the first 13 problems within the allotted time of 1 h. The one exception to this is for XPRESS which terminated with a slightly inferior solution on one problem. All three solvers produced non-optimal solutions for Wang 250. Moreover, all three terminated with an "out of memory "error without giving a solution for test problems Wang 800 and Wang 1150. The size of the linear model (i.e., Model 1) for the Wang problems precludes producing good solutions, or solutions at all. Considering all 16 problems in Table 4, Gurobi turned in the best performance although there is not much difference between the performances of the solvers on the 13 small problems.

Jaehn and Pesch,[18] presented a branch and bound algorithm specialized for CPPs and reported computational experience with their algorithm on the first 13 problems in Table 4. The first six problems (wild cats through UNO2a) were optimally solved very quickly by their method, most likely faster than we report in Table 4 considering we used a newer computer. These clustering problems were very easy for their method as was also the case for all three commercial methods shown in the Table 4. However, the group technology problems (Kumar down to sule) were difficult for specialized method of Jaehn and Pesch and they were not able to solve these group technology problems to optimality within the time limit they allowed of one-half hour. In contrast, all three commercial methods we report on in Table 4 quickly solved all 13 of these problems. Table 4 reveals, as we have seen earlier, that Model 1

Table 4. Results from Model 1 (1 h time limit).

Model 1		Gurobi			Cplex			Xpress		
ID	Nodes	Objective	Diff(%)	Time	Objective	Diff(%)	Time	Objective	Diff(%)	Time
wild cats	30	1304	0%	0	1304	054	0.08	1304	0%	0
cars	33	1501	0%	0.01	1501	0%	0.13	1501	0%	0
workers	34	964	0%	0	964	054	0.22	964	0%	1
UNO	54	778	0%	0	778	0%	0.83	778	0%	0
UNO1a	158	12197	0%	0.48	12197	0%	120.72	12197	0%	30
UN02a	158	72820	0%	0.47	72820	0%	58.5	72776	0%	23
kumar	24	23	0%	0	23	0%	0.05	23	0%	0
boc	59	67	0%	157	67	0%	300.2	67	0%	160
groover	43	55	0%	48	55	0%	162.39	55	0%	113
leskowsky	38	30	0%	1	30	0%	3.36	30	0%	4
mccormick	39	43	0%	1	43	0%	5.03	43	0%	9
seifoddini	33	54	0%	0	54	0%	0.25	54	0%	11
sule	31	46	0%	0	46	0%	0.39	46	0%	0
Wang 250	250	363	0%	2239	189	−48%	3331.47	259	−29%	414
Wang 800	800	NA	NA	NA	NA	NA	NA	NA	NA	NA
Wang 1150	1150	NA	NA	NA	NA	NA	NA	NA	NA	NA

has limited utility as problem size scales upward as shown by the results for the Wang problems.

4.1. *Comparing exact solvers on Model 2*

Table 5 presents the results obtained from running the solvers on the quadratic formulation of Model 2. We set the $k_{\max}$ parameter in all cases so that $k_{\max}$ proved to be more than sufficient for the number of cliques actually produced for each of the 16 problems of Table 5. The $k_{\max}$ value for clustering instances in section is the default clique values reported from Refs. 18 and 21. The $k_{\max}$ value for the group technology instances in Sec. 4.1 is the number of machines or parts, whatever is smaller. As before, a time limit of one hour was imposed in each case.

An examination of Table 5 shows that all five solution methods did well on the first 13 problems, finding and proving optimal solutions quickly in all but one case. Note that, with one exception, all five solvers reported solutions for each of the three Wang problems within the one-hour limit. However, the quality of these solutions varies widely with Gurobi (Quad) turning in the best performance by far. Considering all the problems in Table 5, Gurobi (Quad) produced the best solutions for all the 16 problem instances.

4.2. *Comparing Models 1 and 2*

Comparing the results from Table 4 (Model 1) with those from Table 5 (Model 2), we see that the Model 1 and Model 2 results, across all solvers, are fairly close in quality for the first 13 problems. For the larger group technology problems, however, several of the Model 2 solvers found solutions far superior to the corresponding results produced from Model 1. For the two largest Wang problems, all five solvers for Model 2 produced solutions within the one-hour time limit, in contrast to the fact that none of the three solvers applied to Model 1 were able to report a solution (at all) within this time limit.

The results, shown in Tables 4 and 5, highlight again how difficult it is for exact methods to find good solutions to CPPs as they scale in size. This difficulty is particularly on display for the Wang problems where the linear model struggled to produce solutions at all in the allotted time limit of 1 h. The quadratic model (Table 5) generally produced solutions but the gaps at the 1 h mark are enormous, suggesting that the "1-h" solutions are of low quality.

To get a sense of the time-quality tradeoff, Wang 250 was solved again via Gurobi with an increased time limit of 24 h on Model 1. Moreover, all three Wang problems were solved again using Gurobi (Quad) on Model 2, allowing for the new time limit of 24 h. Results from these runs are shown in Table 6, where both the 1 and 24 h results are reported to enable easy comparisons.

Examining three Wang problems, we see that increasing the time limit from 1 h to 24 h significantly improved both the gaps and the objective function values. Note that for Wang 250, the 24 hour result obtained from Model 1 is now slightly better

Table 5. Results from Model 2 (1 h time limit).

Model 2			Gurobi (Lin)			Gurobi (Quad)			Cplex (Lin)			Cplex (Quad)			Xpress		
ID	Node	Kmax	Objective	Diff(%)	Time	Objective	Diff(%)	Time	Objective	Diff(%)	Time	Objective	Diff(%)	Time	Objective	Diff(%)	Time
wild cats	30	4	1304	0%	0	1304	0%	0	1304	0%	0.09	1304	0%	0.02	1304.00	0%	0
cars	33	3	1501	0%	0	1501	0%	0	1501	0%	0.13	1501	0%	0.03	1501.00	0%	1
workers	34	4	964	0%	0	964	0%	0	964	0%	1.78	964	0%	0.09	964.00	0%	4
UNO	54	5	778	0%	2	778	0%	6	778	0%	16.27	778	0%	4.2	778.00	0%	19
UNO1a	158	6	12197	0%	154.1	12197	0%	260	12197	0%	120.73	12197	0%	120.66	12197.00	0%	61
UNO2a	158	3	72820	0%	22	72820	0%	60	72820	0%	4	72820	0%	0.34	72820.00	0%	3
kumar	24	6	23	0%	0	23	0%	0	23	0%	0.08	23	0%	0.06	23.00	0%	0
boc	59	10	67	0%	14	67	0%	3	67	0%	2.47	67	0%	28	65.00	−2.99%	305
groover	43	8	55	0%	88	55	0%	4	55	0%	46.91	55	0%	5.17	55.00	0%	311
leskowsky	38	10	30	0%	1	30	0%	211	30	0%	2.7	30	0%	5.05	30.00	0%	3
mccormick	39	9	43	0%	2	43	0%	0	43	0%	1.06	43	0%	5.22	43.00	0%	5
seifoddini	33	5	54	0%	0	54	0%	0	54	0%	0.14	54	0%	0.11	54.00	0%	1
sule	31	5	46	0%	0	46	0%	0	46	0%	0.13	46	0%	0.02	46.00	0%	1
Wang 250	250	14	275	−25%	3600	368	0%	3600	288	−22%	3357.05	337	−8%	3600	284.00	−23%	2085
Wang 800	800	9	488	−7.05%	490	525	0%	3600	195	−62.86%	1301.99	43	−91.81%	2743.5	−383.00	−172.95%	307
Wang 1150	1150	11	143	−81.78%	3600	785	0%	3335	0	−100.00%	3603.2	61	−92.23%	3311.36	−6852.00	−972.87%	205

Table 6. Longer Gurobi runs on Models 1 and 2.

Models 1 and 2	Model 1 Gurobi (24 h)			Model 1 Gurobi (1 h)			Model 2 Gurobi (Quad) 24 h			Model 2 Gurobi (Quad) 1 h		
ID	Objective	Gap	Time	Objective	Gap	Time	Objective	Gap	Time	Objective	Gap	Time
Wang 250	403	25.50%	62066	363	53%	2239	395	1990%	25590	368	2224%	3600
Wang 800	NA	NA	NA	NA	NA	NA	1098	3933%	77594	525	8413%	3602
Wang 1150	NA	NA	NA	NA	NA	NA	1540	410%	33820	785	975%	3335

than the 24 h solution obtained from Model 2. However, the objective function values obtained from Model 2 on each of the larger problems approximately doubled as the time limit was raised from 1 to 24 h.

Table 6 shows that the Model 2 gaps, with the increase time limit, are much improved but still very large, suggesting that even longer run times may produce further improved solutions. All told, the results shown in Table 6 indicate once again that Model 2, the quadratic model, is a viable alternative to Model 1 for producing solutions from problem instances as problem size scales upward.

5. Computational Experiments Regarding Problem Set 3

In recent years, modularity maximization has become a widely used criterion for detecting communities in networks. Since the initial work by Newmann and Girvan (2004) several authors have proposed models and solutions methods intended to carry out such analysis. The work by Aloise *et al.*[19] proposed and tested four different exact methods for solving the modularity maximization problem, three of which were based on the standard linear clique partitioning model, i.e., Model 1 considered in this paper. Their computational experience was based on a set of test problems from the literature and they were the first to report proven optimal solutions for all of the test problems considered. Below we report on these same problems.

Our results are presented below in Table 7 where we show the reported optimal solution from the Aloise *et al.*,[19] paper along with the results we obtained by running CPLEX on Model 1 and CPLEX(Quad) on Model 2. We point out that while Aloise *et al.*,[19] tested special row and column generation methods for solving Model 1, these efforts did not produce optimal solutions to all problems within the allotted time limit of 100,000 seconds. Instead, they employed a special column generating method on an alternative formulation (i.e., not Model 1) which was able to provide optimal solutions for all their test problems. It is these results that we list as best known solution (BKN) in Table 7 to provide a benchmark for comparison with our models. In the table, n denotes the number of vertices in the graph, the "$k_{\max}$" value for clustering instances in Sec. 5 is the default clique values reported from Ref. 19. "objective" denotes the best objective function found, and the timing information denotes the "time to best." The first six problems were given a time limit of 3600 s and the last four, following the lead of Ref. 19, were given a time limit or 100,000 s. The results in the table highlighted in yellow are proven optimal results obtained from our Models 1 and 2.

Looking at Table 7, we see that Model 1, as we have seen earlier in this paper, performs well on small problems, quickly finding and proving optimality for the first six problem. As noted in Ref. 19, for the last four problems, the MIPs represented by Model 1 are too large to be solved in any reasonable time if at all. For instance, for the A01 problem, Model 1 has more than 7 million rows and 30 thousand columns. For the largest problem, Circuit, Model 1 has more than 27 million rows and

Table 7. Test problems for modularity maximization.

				Model 1		Model 2	
ID	n	kmax	BKN	Objective	Time	Objective	Time
Karate	34	4	0.4198	0.4198	<1	0.4198	<1
Dolphins	62	5	0.5285	0.5285	14	0.5285	63
Les mis	77	6	0.5600	0.5600	5	0.5600	33
A00	83	9	0.5309	0.5309	74	0.5309	236
Books	105	5	0.5272	0 5273	191	0.5273	87
Football	115	10	0.6046	0.6046	53	0.6046	453
A01	249	14	0.6329	NA	NA	0.6272	12098
US Air	332	6	0.3682	NA	NA	0.3682	21180
Netscience	379	19	0.8486	NA	NA	0.8476	96814
Circuit	512	12	0.8194	NA	NA	0.8369	45700

71 thousand columns. Given these sizes, Model 1 was unable to produce solutions for these problems.

Model 2 also found optimal solutions to the first six problems but was only able to prove optimality for the first two problems in the time allotted. With the exception of the fifth problem, Books, the "time to best" for Model 2 was generally longer than that of Model 1 for these same six problems. Note, however, that Model 2 was able to perform rather well on the larger, last four problems, finding, but not proving, the optimal solution for the US Air problem and providing near-optimal solutions for the others. We note the Model 2 result for the last problem, Circuit, is greater than the result reported as optimal in Ref. 19. We suspect that this is due to an error in our data or a reporting error in their paper. We further note that while not reported in the table, a sample of results obtained from Gurobi on these problems proved to be similar to those we report here obtained from CPLEX.

All told, the results shown in Table 7 indicate once again the advantage of Model 2 over Model 1 in terms of providing good solutions to moderate sized CPPs as problem size scales upward and Model 1 fails to be functional. We note that the last four problems were fairly difficult for the specialized method used by Aloise *et al.*,[19] Yet, Model 2, and CPLEX's standard quadratic 0/1 optimizer, without any customization, was able to find very high quality solutions in a reasonable amount of time for these problems.

6. Summary and Conclusions

The computational experience reported in Secs. 3, 4 and 5 highlights the difficulty commercial solvers have in finding optimal or even near-optimal solutions to CPPs as problem size scales upward. For Model 1 even solving the root node problem for the linear model can become problematic as problem size grows due to the enormous size of the associated relaxations. For example, for a problem with 500 nodes, Model 1 would have roughly 125,000 variables and more than 62,000,000 constraints.

By contrast, the quadratic model scales much more agreeably. Model 2, for example, on a graph with 500 nodes and a $k_{\max}$ value of 10, would have just 5000 variables and 500 constraints, offering a clear size advantage by comparison to the linear model.

The main contribution of this paper is to highlight the advantage offered by the quadratic formulation compared to the standard linear model when solutions to CPPs are sought using commercial exact solvers. Considering the various model/solver combinations tested in this paper, the quadratic model turned in the best performance across the entire test bed of problems considered here by a wide margin.

Our computational experiments indicate that the ability to find good solutions from the commercial codes using the standard linear model is limited to graphs of size 50 nodes or less for problems with the characteristic considered here. In contrast to this, adopting the alternative quadratic model can extend the size for CPPs that can be effectively solved by exact methods like CPLEX, Gurobi and XPRESS considerably, as highlighted in our tables. As exact methods for solving quadratic binary models continue to advance, the attractiveness of Model 2 relative to the linear model will most likely grow even larger.

In addition to highlighting the superiority of the quadratic model, we have provided a comparison of the relative performance of CPLEX, Gurobi, and XPRESS, on both the linear and the quadratic models. Looking only at the linear models, Gurobi turned in the best performance followed by CPLEX and XPRESS in that order. On the other hand, focusing on the quadratic models, CPLEX turned in the best performance on the random problems of Table 3 while Gurobi gave the best results on the structured problems of Table 5. XPRESS lagged behind in performance across the board. It's interesting that the paper by Lima and Grossmann (2017), while working on problems other than clique partitioning, reported that CPLEX far out performed Gurobi and XPRESS when directly solving the quadratic models they considered. This highlights that the issue of which solver is best for solving quadratic models directly is still an open question.

Our results on the comparative performance of CPLEX, Gurobi and XPRESS are fairly consistent with other recent comparisons given by Anand *et al.*,[23] and Hvattum *et al.*,[24] on combinatorial problems. On a wide variety of problem testing, these papers concluded, as we have, that all three solvers are greatly improved but that CPLEX and Gurobi generally outperform XPRESS and that neither CPLEX nor Gurobi dominates the other across all problems and performance measures.

It is interesting to reflect on traditional practices that are commonly adopted when solving optimization problems. These practices have long placed a premium on linear model representations over nonlinear models whenever a choice between the two model forms is an issue. In fact, it is often the case that when one is confronted with a nonlinear model, the first step in seeking a solution is to re-formulate the model in a linear form. The experience we report in this paper illustrates that doing just the opposite has proved to be beneficial in the context of CPP. Opting for linearity, as we show, is not the best choice for finding high quality solutions to CPPs when using solvers such as CPLEX, Gurobi, and XPRESS. This superior performance of a quadratic model over an equivalent linear model is likely to be realized in general for optimization problems on graphs where one can go from a linear model with edge-related variables to a quadratic model with node-related variables as demonstrated in the work presented here.

Given the improvement in commercial solvers as tested here for solving quadratic models in binary variables and the growing awareness of the attractiveness of working directly with such quadratic models, the traditional practice of first looking for a suitable linearization is being challenged. Noteworthy recent articles supporting this adoption of quadratic models in binary variables in settings other than the clique partitioning are reported by Wang and Alidaee (2018); Carrizosa *et al.* (2018) and Matsypura *et al.* (2018).

We note that the improvement in commercial solvers opens the door for realistic applications in other problem domains as can be found in the recent papers.[25–27]

This current project has opened up several avenues for future research that we intend to pursue and report on in future papers. For instance, we have the following:

- The computational testing carried out in this paper on the quadratic version on the clique partitioning model employed, as one solution option, the standard off-the shelf linearization methods embedded in CPLEX, Gurobi and Xpress. Certainly, other linearizations, as highlighted in the recent papers by Forrester and Hunt-Isaak (2020), Furini and Traversi (2018), and Lima and Grossmann (2017) hold promise for improved performance of Model 2. These important papers give extensive computational and theoretical insights into alternative linearizations of quadratic models and provide guidance for potential improved performance of the quadratic model considered here for clique partitioning. Testing such alternative linearizations is a priority for our future work.
- The performance of both Models 1 and 2 could potentially be enhanced by augmenting the models with additional constraints. For instance, facets as discussed by Grotschel and Wakabayashi,[21] and by Oosten *et al.*,[22] could be implemented in Model 1. Moreover, adopting ideas from Ref. 28, symmetry breaking constraints, addressing a potential problem with Model 2, could be added to Model 2. We intend to explore the impact of such changes on the ultimate comparison of the two models as part of our future work.
- Regarding the performance of Model 1, ideas coming from the work of Dinh and Thai,[12] Miyauchi and Sukegawa,[13] and Miyauchi *et al.*,[15] on removing unnecessary constraints from this model may lead to improved performance by reducing the problem size. We plan to explore the implementation of these ideas in our future work.
- Our focus in this paper has been on comparing the performance of off-the-shelf exact solvers on the quadratic alternative to the standard linear model for clique partitioning. An interesting alternative approach, with potential application to the problems we consider here, is to encode the clique partitioning model (Model 1) as a MaxSat instance to be solved optimally by exact MaxSat solvers. Such an

approach was successfully adopted by Berg and Jarvisalo (2015) in the context of Correlation Clustering. Their success motivates us to consider this approach as part of our future research.

- Finally, we are curious about the interaction of exact and heuristic methods for the models considered here, perhaps exploiting information quickly obtained from heuristic methods to enhance the performance of the exact methods. These and related issues are also on our list for future investigation.

Over all, we expect additional insights about models and algorithms will be provided by further testing on CPP problem variants and other graph problems related to clique partitioning. We plan to report on our findings in these domains in future papers.

Appendix A. Relationship Between Model 1 and Model 2

The relationships between Model 1 and Model 2 can be stated as follows:

For the extreme case when $k_{\max} = \#$ of vertices, an optimal solution to Model 2 corresponds to an optimal solution for Model 1. That is, the models have equivalent solutions when $k_{\max} = \#$ of vertices. However, when $k_{\max}$ is less than the number of vertices, it is possible that the two models are not equivalent, meaning that the optimal solution to Model 2 may not correspond to an optimal solution to Model 1. We comment that based on extensive computational experience over a wide variety of CPPs, and often with quite small values of $k_{\max}$, Model 2 has consistently delivered high quality and often optimal solutions. Nonetheless, for a given problem, if $k_{\max}$ is set too low, it is possible that an optimal solution to Model 2 might not be optimal for Model 1. It will always be a feasible solution, but not necessarily an optimal solution. We illustrate how this could happen with the following example.

Given a graph with $2n$ vertices, which are denoted by $\{1, 2, \ldots, n, 1', 2', \ldots, n'\}$. Vertices $\{1, \ldots, n\}$ denote a complete subgraph with edge weights 1 and similarly, vertices $\{1', \ldots, n'\}$ also denote a complete subgraph with edge weights 1. Moreover, there are $n - 1$ further edges to which we will refer as "connected edges". Vertex pairs $(1, 1')$, $(2, 2')$, …, $(n - 1, (n - 1)')$ are connected by edges (but NOT (n, n')). These edges have edge weight

$n + 1/(n - 1)$ (see Fig. 2). As the input of CPP has to be a complete graph, we assume that all edges not depicted in the figure are weighted with a very big negative value.

We will analyze the optimal solution of this instance by a distinction of cases. We will distinguish how many connected edges are considered in the solution (note that due to the symmetry of the graph, we need not decide which connected edges are considered). Given a certain number of connected edges, both end nodes of a connected edge define their own clique (as otherwise, edges with a negative value have to be added). The remaining vertices of each complete subgraph will all end up in the same clique. This observation can be summarized in the following cases.

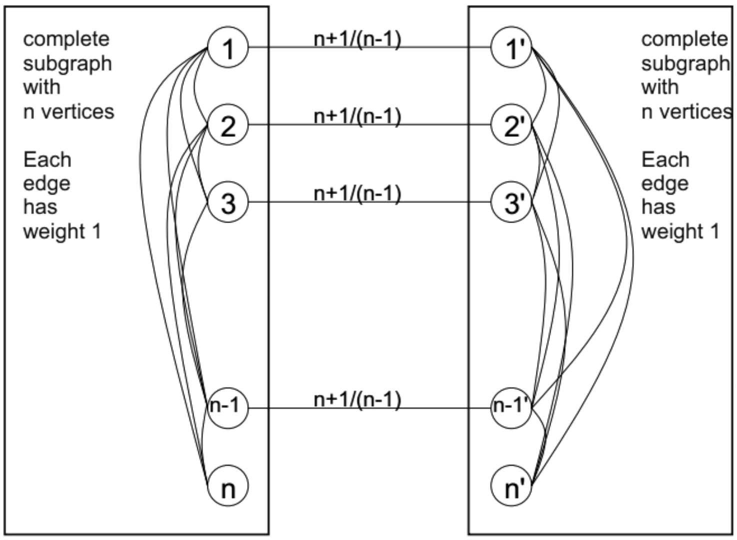

Fig. 2. Counter example.

Case 1 # connected edges: 0

- Best solution: $\{1, 2, \ldots, n\}$, $\{1', 2', \ldots, n'\}$,
- # cliques: 2,
- Objective value: $2\frac{n(n-1)}{2} = n^2 - n$,

Case 2 # connected edges: $1 \leq x \leq n - 2$

- Best solutions: $\{1, 1'\}$, $\{2, 2'\}$,..., $\{x, x'\}$, $\{x + 1, \ldots, n\}, \{(x + 1)', \ldots, n'\}$,
- # cliques: $x + 2$,
- Objective values: $2\frac{(n-x)(n-x-1)}{2} + x(n + \frac{1}{n-1}) = n^2 - n + x^2 + x - xn + \frac{x}{n-1}$,

Case 3 # connected edges: $n - 1$

- Best solutions: $\{1, 1'\}$, $\{2, 2'\}$, ..., $\{n - 1, (n - 1)\ '\}$, $\{n\}, \{n'\}$,
- # cliques: $n + 1$,
- Objective values: $(n - 1)(n + \frac{1}{n-1}) = n^2 - n + 1$.

We can see that the objective function value in the second case is always lower than the objective function value in the first case (as $x^2 + x - xn + \frac{x}{n-1}$ is always negative for $1 \leq x \leq n - 2$). Thus, if $k_{\max} \leq n$, a solution with $n - 1$ connected edges is excluded and Model 2 will always deliver a solution with only two cliques. However, the optimal solution contains $n + 1$ cliques. This proves that Model 2 is not equivalent to Model 1 with significantly less cliques than $k_{\max}$.

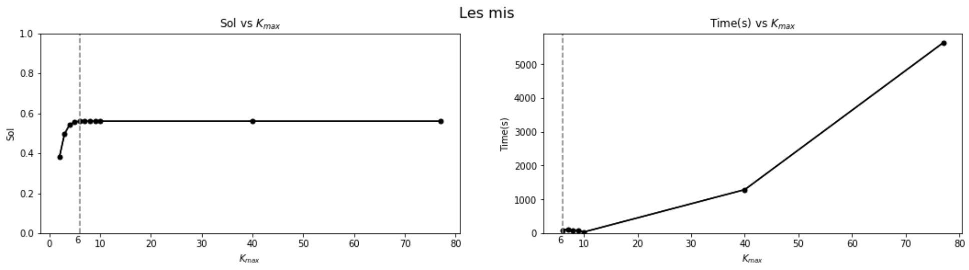

Fig. 3. Computational impact of $k_{\max}$ for Les mis.

Appendix B. Computational Impact of $k_{\max}$

We provide additional analysis of the impact of $k_{\max}$ on the computational performance of quadratic Model 2 using the sample data set (Les mis) in Table 7. The computational burden goes up substantially due to the larger models created as $k_{\max}$ is increased. We have clarified this with a brief discussion and charts illustrating this behavior on the following example.

For data set Les mis, the number of nodes is 77, k_{opt} is 6, and the optimal objective function value is 0.56. To provide this illustration, we solved this problem several times with $k_{\max}$ values ranging from 2 to 77. Figure 3 shows the sub-optimal solutions obtained for $k_{\max}$ less than the k_{opt} value of 6 and the optimal value (0.56) for $k_{\max}$ values of 6 and greater. Figure 3 shows the rapidly increasing run times as $k_{\max}$ grows.

References

1. B. W. Kernighan and S. Lin, An efficient heuristic procedure for partitioning graphs, *The Bell System Technical Journal* **49** (1970) 291–307.
2. C. H. Chen, Graph partitioning for concurrent test scheduling in VLSI circuit, in *Proc. 28th ACM/IEEE Design Automation Conf* (ACM, 1991), pp. 287–390.
3. T. A. Feo and M. Khellaf, A class of bounded approximation algorithms for graph partitioning, *Networks* **20** (1990) 181–195.
4. J. Kral, To the problem of segmentation of a program, *Information Processing Machines* **2** (1965) 116–127.
5. H. Wang, B. Alidaee, F. Glover and G. Kochenberger, Solving group technology problems via clique partitioning, *International Journal of Flexible Manufacturing Systems* **18**(2) (2006) 77–97.
6. G. Dahl, G. Storvik and A. Fadnes, Large-scale integer programs in image analysis, *Operations Research* **50**(3) (2002) 490–500.
7. M. A. T. Mehrotra, Clique and clustering: A combinatorial approach, *Operations Research Letters* **22** (1998) 1–12.
8. H. Wang, T. Obremski, B. Alidaee and G. Kochenberger, Clique partitioning model for clustering: A comparison with K-means and latent class analysis, *Communications in Statistics- Simulation and Computation* **37**(1) (2008) 1–13.
9. G. Agarwal and D. Kempe, Modularity-maximizing graph communities via mathematical programming, *The European Physical Journal B* **66**(3) (2008) 409–418.

10. N. Ailon, M. Charikar and A. Newman, Aggregating inconsistent information: Ranking and clustering, *Journal of the ACM (JACM)* **55**(5) (2008) 23.
11. S. Bruckner, F. Huffner, C. Komusiewicz and R. Niedermeier, Evaluation of ILP-based approaches for partitioning into colorful components, in *Int. Symp. Experimental Algorithms* (Springer, Berlin, 2013), pp. 176–187.
12. T. N. Dinh and M. T. Thai, Toward optimal community detection: From trees to general weighted networks, *Internet Mathematics* **11**(3) (2015) 181–200.
13. Miyauchi and N. Sukegawa, Redundant constraints in the standard formulation for the clique partitioning problem, *Optimization Letters* **9**(1) (2015) 199–207.
14. J. Berg and M. Jarvisalo, Cost-optimal constrained correlation clustering via weighted partial maximum satisfiability, *Artificial Intelligence* **244** (2017) 110–142.
15. Miyauchi, T. Sonobe, and N. Sukegawa, Exact clustering via integer programming and maximum satisfiability, in *Thirty-Second AAAI Conf. Artificial Intelligence* (2018).
16. U. Dorndorf, F. Jaehn and E. Pesch, Flight gate assignment and recovery strategies with stochastic arrival and departure times, *OR Spectrum* **39** (2017) 65–93.
17. R. Axelrod, W. Mitchell, R. E. Thomas, D. S. Bennett and E. Bruderer, Coalition formation in standard setting alliances, *Management Science* **41**(9) (1995) 1493–1508.
18. F. Jaehn and E. Pesch, New bounds and constraint propagation techniques for the clique partitioning problem, *Discrete Applied Mathematics* **161** (2013) 2025–2203.
19. D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron and L. Liberti, Column generation algorithms for exact modularity maximization in networks, *Physical Review E* **82**(4) (2010) 046112.
20. S. Chopra and M. R. Rao, The partition problem, *Mathematical Programming* **59** (1993) 87.
21. M. Grotschel and Y. Wakabayashi, A cutting plane algorithm for a clustering problem, *Mathematical Programming* **45** (1989) 59–96.
22. M. Oosten, J. H. G. C. Rutten and F. C. R. Spieksma, The clique partitioning problem: Facets and patching facets. *Networks* **38**(4) (2001) 209–226.
23. R. Anand, D. Aggarwal and V. Kumar, A comparative analysis of optimization solvers, *Journal of Statistics and Management Systems* **20**(4) (2017) 623–635.
24. L. M. Hvattum, A. Lokketangen and F. Glover, Comparisons of commercial MIP solvers and an adaptive memory(tabu search) procedure for a class of 0-1 integer programming problems, *Algorithmic Operations Research* **7** (2012) 12–31.
25. G. Kou and C. Lin, A cosine maximization method for the priority vector derivation in AHP, *European Journal of Operational Research* **235**(1) (2014) 225–232.
26. G. Kou, H. Xiao, M. Cao and L. H. Lee, Optimal computing budget allocation for the vector evaluated genetic algorithm in multi-objective simulation optimization, *Automatica* **129** (2021) 109599.
27. J. Zhang, G. Kou, Y. Peng and Y. Zhang, Estimating priorities from relative deviations in pairwise comparison matrices, *Information Sciences* **552** (2021) 310–327, ISSN 0020-0255.
28. E. Malaguti and P. Toth, A survey on vertex coloring problems, *International Transactions in Operational Research* **17**(1) (2010) 1–34.
29. M. Lewis, H. Wang and G. Kochenberger, Exact solutions to the capacitated clustering problem: A comparison of two models, *Annals of Data Science*, **1**(1) (2014) 15–23.
30. X. Li and J. E. Mitchell, Branch-and-price-and-cut on the clique partitioning problem with minimum clique size requirement, *Discrete Optimization* **4**(1) (2007) 87–102.
31. R. M. MacGregor, *On Partitioning a Graph.* PhD Dissertation, University of California (1978).

32. Newman, Mark and Girvan, Michelle, Finding and evaluating community structure in networks. *Physical Review E, Statistical, Nonlinear, and Soft Matter Physics* **69** (2004) 026113, 10.1103/PhysRevE.69.026113.
33. R. M. Lima and I. E. Grossmann, On the solution of nonconvex cardinality Boolean quadratic programming problems: A computational study. *Comput Optim Appl* **66** (2017) 1–37.
34. H. Wang and B. Alidaee, Unrelated parallel machine selection and job scheduling with the objective of minimizing total workload and machine fixed costs, *IEEE Transactions on Automation Science and Engineering* **15**(4) (2018) 1955–1963.
35. E. Carrizosa V. Guerrero and D. R. Morales, On mathematical optimization for the visualization of frequencies and adjacencies as rectangular Maps, *EJOR* **265** (2018) 290–302.
36. D. Matsypura, O. A. Prokopyev and A. Zahara, Wildfire fuel management: Network-based models and optimization of prescribed burning, *EJOR* **264**(2) (2018) 774–706.
37. Forrester, Richard J. and Noah Hunt-Isaak, Computational comparison of exact solution methods for 0-1 quadratic programs: Recommendations for practitioners, *Journal of Applied Mathematics*, Article ID 5974820 (2020) 21.
38. F. Furini and E. Traversi, Theoretical and computational study of several linearization techniques for binary quadratic problems, *Annals of OR* **279**(2) (2019).