Noname manuscript No.
(will be inserted by the editor)

Worst-case analysis of clique MIPs

Mohammad Javad Naderi · Austin Buchanan · Jose L. Walteros

Received: date / Accepted: date

Abstract The usual integer programming formulation for the maximum clique problem has several undesirable properties, including a weak LP relaxation, a quadratic number of constraints and nonzeros when applied to sparse graphs, and poor guarantees on the number of branch-and-bound nodes needed to solve it. With this as motivation, we propose new mixed integer programs (MIPs) for the clique problem that have more desirable worst-case properties, especially for sparse graphs. The smallest MIP that we propose has just $O(n+m)$ nonzeros for graphs with n vertices and m edges. Nevertheless, it ensures a root LP bound of at most $d+1$, where d denotes the graph's degeneracy (a measure of density), and is solved in $O(2^d n)$ branch-and-bound nodes. Meanwhile, the strongest MIP that we propose visits fewer nodes, $O(1.62^d n)$. Further, when a best-bound node selection strategy is used, $O(2^g n)$ nodes are visited, where $g = (d+1) - \omega$ is the clique-core gap. Often, g is so small that it can be treated as a constant in which case $O(n)$ nodes are visited. Experiments are conducted to understand their performance in practice.

Keywords integer program · branch-and-bound · fixed-parameter tractability · clique · k-core · degeneracy · clique-core gap

M.J. Naderi
Oklahoma State University, Stillwater, OK, USA
E-mail: mnaderi@okstate.edu

A. Buchanan
Oklahoma State University, Stillwater, OK, USA
E-mail: buchanan@okstate.edu

J.L. Walteros
University at Buffalo, The State University of New York, Buffalo, NY, USA
E-mail: josewalt@buffalo.edu

1 Introduction

The maximum clique problem over graph $G = (V, E)$ can be expressed as the following integer program. Letting $n = |V|$ and $m = |E|$, it uses n binary variables, as well as a conflict constraint for each of the $\binom{n}{2} - m$ edges of the complement graph $\bar{G} = (V, \bar{E})$.

$$\omega(G) = \max \sum_{i \in V} x_i \tag{1a}$$

$$\text{(conflict)} \qquad x_i + x_j \leq 1 \qquad \forall \{i, j\} \in \bar{E} \tag{1b}$$

$$x_i \in \{0, 1\} \qquad \forall i \in V. \tag{1c}$$

While this *conflict* model is simple to state, it can be impractical, especially when the graph is sparse. One reason is the large number of conflict constraints (1b). For example, for trees there are $\binom{n}{2} - (n-1)$ conflicts, which can make it difficult to build or solve the root LP relaxation for graphs with thousands of vertices. A second reason is its poor linear programming (LP) relaxation; the all-half vector $(\frac{1}{2}, \dots, \frac{1}{2})$ is always feasible, meaning that the root LP bound (at least $n/2$) can be far from the optimal IP objective ω (which equals 2 for trees). Third, this formulation generally provides little guarantee on how many branch-and-bound nodes will be needed.

Consequently, many have sought to improve this formulation—or, more generally, MIPs that have conflict constraints or set packing constraints. For example, one can generate valid inequalities to strengthen the LP relaxation [3, 23, 40, 63, 67]. These inequalities can be added initially, or on-the-fly as cutting planes. Both approaches are employed by open-source MIP solvers like CBC [16] and proprietary MIP solvers like Gurobi [1]. While these procedures significantly impact solve time in practice, they do not lead to nontrivial worst-case bounds on the number of branch-and-bound nodes, to our knowledge.

An alternative model proposed by [27, 58] is below, cf. stronger versions by [49, 60] and a generalization for k-plex (which permits each vertex to have up to k nonneighbors in the cluster) [7]. It is obtained by aggregating the conflict constraints, giving the big-M constraints (2b) where $M_i = n - |N(i)| - 1$ and $N(i)$ is the subset of vertices that neighbor vertex i.

$$\omega(G) = \max \sum_{i \in V} x_i \tag{2a}$$

$$\text{(dense)} \qquad \sum_{j \in V \setminus N[i]} x_j \leq M_i (1 - x_i) \qquad \forall i \in V \tag{2b}$$

$$x_i \in \{0, 1\} \qquad \forall i \in V. \tag{2c}$$

We call this the *dense* model because it uses approximately $n^2 - 2m$ nonzeros, much like the conflict model. However, the number of nonzeros can be reduced to $O(n+m)$ with a new variable $z = \sum_{i \in V} x_i$, giving the *sparse* model.

$$
\begin{aligned}
& & \omega(G) = {} & \max z & & \text{(3a)}\\
& & z = {} & \sum_{i \in V} x_i & & \text{(3b)}\\
&\text{(sparse)} & z - \sum_{j \in N[i]} x_j \leq {} & M_i(1 - x_i) & \forall i \in V & \text{(3c)}\\
& & x_i \in {} & \{0,1\} & \forall i \in V. & \text{(3d)}
\end{aligned}
$$

Because this model is small, its LP relaxation is likely easier to solve than that of the conflict model. However, this comes at the cost of a weaker LP relaxation, which likely leads to more branch-and-bound nodes being visited.

Contributions for Existing MIPs. In Section 3, we observe that MIPs like these can perform quite poorly, even when applied to straightforward instances like co-lollipops, which are the complements of the lollipop graphs $L_{3,p}$ shown in Figure 1. Namely, the co-lollipops are solved in $\Theta(\varphi^n)$ branch-and-bound nodes when using the conflict model, where $\varphi \approx 1.61803$ is the golden ratio.

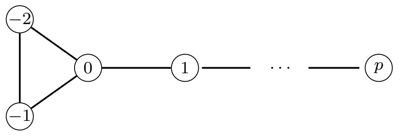


Fig. 1: The lollipop graph $L_{3,p}$.

In response, we seek alternative MIPs with better worst-case properties. We are particularly interested in exploiting the characteristics of real-life graphs, such as small (graph) degeneracy d and small clique-core gap $g := (d+1) - \omega$. Table 1 illustrates that the degeneracy and clique-core gap of real-life graphs are much smaller than the number of nodes. (The notions of degeneracy and clique-core gap are defined in Section 2.2.)

Table 1: Walteros and Buchanan [71] observe that real-life graphs have small clique number ω, degeneracy d, and clique-core gap g compared to the number of nodes n.

graph	n	m	ω	d	g
as-22july06	22,963	48,436	17	25	9
citationCiteseer	268,495	1,156,647	13	15	3
ldoor	952,203	22,785,136	21	34	14
in-2004	1,382,908	13,591,473	489	488	0
cage15	5,154,859	47,022,346	6	25	20
uk-2002	18,520,486	261,787,258	944	943	0

Contributions for New MIPs. In Section 4, we conceive of four new MIPs for the maximum clique problem that take inspiration from the literature on cliques and disjunctive extended formulations. Each has a root LP objective of at most $d+1$, which is typically much smaller than the root LP of the conflict model. In Section 5, we show that the weakest of these four MIPs is solved in $O(2^d n)$ branch-and-bound nodes. Meanwhile, the strongest of the four MIPs is solved in $O(\varphi^d n)$ nodes, improving on the $O(\varphi^n)$ bound obtained by the conflict and sparse models; moreover, we show that the analysis is tight by providing a class of graphs (related to the co-lollipop graphs) for which $\Omega(\varphi^d n)$ nodes are visited. Finally, we show that if branch-and-bound follows a best-bound node selection strategy, then a different (usually better) bound of $O(2^g n)$ holds—a virtue not shared by the conflict model. In Section 6, we experiment with the MIPs to understand their performance in practice. We conclude and offer directions for future research in Section 7.

2 Background and Literature Review

Below, we review basic terminology from graph theory, the maximum clique problem, MIP techniques for clique finding, and disjunctive formulations.

2.1 Graph terminology

We consider a simple graph $G=(V,E)$ with vertex set V and edge set $E \subseteq \binom{V}{2}$, where $\binom{V}{2}$ denotes the collection of 2-vertex subsets of V. The subgraph of G induced by a vertex subset $S \subseteq V$ is denoted by $G[S] := (S, E \cap \binom{S}{2})$. The (open) neighborhood of vertex $v \in V$ is denoted $N(v) := \{u \in V \mid \{u,v\} \in E\}$. Meanwhile, the closed neighborhood $N[v] := N(v) \cup \{v\}$ also includes v.

A graph $G=(V,E)$ is *complete* if it has all possible edges, i.e., if $E = \binom{V}{2}$. A subset of vertices S is a *clique* if its induced subgraph $G[S]$ is complete. The complete graph on n vertices is denoted by K_n. A clique is *maximal* if no proper superset of it is also a clique. A clique is *maximum* if it has largest size among all cliques. A maximum clique is necessarily maximal, but not vice versa. The clique number, denoted $\omega(G)$, is the size of a maximum clique. As an example, $S=\{-2,-1,0\}$ is a maximum (and thus maximal) clique of the lollipop graph $L_{3,p}$ in Figure 1, implying that $\omega(L_{3,p})=3$. Also, $S'=\{0,1\}$ is a maximal (but not maximum) clique.

The complement of $G=(V,E)$, denoted by $\bar{G}=(V,\bar{E})$, has the same vertex set as G but has the complementary edge set $\bar{E} = \binom{V}{2} \setminus E$. An independent set (a.k.a. stable set) is a subset of vertices S such that $G[S]$ has no edges. Easily, S is a clique in G if and only if S is an independent set in $\bar{G}$. Accordingly, the independence number $\alpha(G)$, which is the size of a largest independent set, satisfies $\omega(G)=\alpha(\bar{G})$. A vertex cover is a subset of vertices S that hits every edge, i.e., $S \cap e \neq \emptyset$ for all $e \in E$. Easily, S is an independent set if and only if $V \setminus S$ is a vertex cover. Accordingly, the vertex cover number $\tau(G)$, which is the size of a smallest vertex cover, satisfies $\tau(G)+\alpha(G)=n$.

2.2 The clique problem

Cliques were originally introduced to model clusters in networks [53], and many clique-like structures have been proposed over the years [64]. The maximum clique problem, which asks for a clique of largest size, is a well-known NP-hard problem [13, 24, 41]. It is notoriously hard to approximate; namely, getting a $n^{1-\varepsilon}$-approximation for any constant $\varepsilon > 0$ is NP-hard [37, 73], while an n-approximation is trivial to achieve by picking a single vertex.

With these observations, there is little hope for clique MIPs to have nice worst-case properties. In fact, even if P=NP, it would still be impossible to always construct small perfect (or approximate) MIPs for clique [14, 15, 31, 35].

As a compromise, one could ask for a fixed-parameter tractable (FPT) algorithm, which is one that runs in time $f(k)n^{O(1)}$, where k is the parameter of choice and f is a computable function that depends only on k. For example, one could seek an algorithm that determines whether a graph has a clique of size k in time $O(2^k n^2)$. However, there are reasons to believe such algorithms do not exist [26, Ch. 13-14].

Fortunately, there is some good news for a different parameter. Eppstein et al. [29] show that all maximal cliques can be enumerated in time $O(dn3^{d/3})$, where d denotes the graph's degeneracy. Intuitively, graph degeneracy is a measure of density. For example, forests have $d = 1$; cycle graphs have $d = 2$; planar graphs have $d \leq 5$; and complete graphs have $d = n - 1$. As Table 1 in the introduction shows, many real-life graphs have small degeneracy, making the algorithm of [29] practical in many cases. Degeneracy is defined as follows.

Definition 1 (Lick and White [50]) A graph G is k-degenerate if each of its subgraphs G' has minimum degree $\delta(G') \leq k$. The degeneracy d of a graph is the smallest k for which it is k-degenerate.

As observed by [50], if a graph has degeneracy d, then it admits a vertex ordering $(v_1, v_2, \ldots, v_n)$ in which each vertex v_i has at most d neighbors to its right. The converse also holds. Such an ordering can be found by applying the following algorithm to $G' \leftarrow G$:

```
1. for $i = 1, 2, \ldots, n$ do:
     – let $v_i$ be a minimum degree vertex of $G'$
     – remove $v_i$ from $G'$
2. return $(v_1, v_2, \ldots, v_n)$.
```

This algorithm can be implemented to run in $O(m + n)$ time [56]. Although the original implementation used linked lists, array-based implementations are faster in practice [9, 71].

A key insight used in many clique algorithms is as follows. For each maximal clique S of the graph G, there is a (unique) vertex v_i for which S is contained in its *closed right-neighborhood*, i.e.,

$$S \subseteq N[v_i] \cap \{v_i, v_{i+1}, \ldots, v_n\} =: V_i.$$

This allows for clique problems to be decomposed into n subproblems[1], $G[V_1]$, $G[V_2]$, …, $G[V_n]$. Each has at most $d+1$ vertices by the degeneracy ordering. This leads to an $O(dn3^{d/3})$ algorithm for listing maximal cliques [29] and an $O((n-d)2^{d/4})$ algorithm for maximum clique [19, 54].

Given that these times are exponential in the degeneracy, these algorithms can be slow when d exceeds 100. However, Walteros and Buchanan [71] observe that for many real-life graphs, the clique number ω is quite close to the upper bound $d+1$, differing by just 0, 1, or 2 units on half of their instances. For cases like this where the clique-core gap $g := (d+1)-\omega$ can be treated as a constant, the maximum clique problem can be solved in time $O(dm) = O(m^{1.5})$, as shown by [71]. This algorithm, which applies FPT vertex cover algorithms to the subproblems, is often fast in practice.

2.3 MIP techniques for clique

The clique problem has a long history in the MIP literature, as do related problems like independent set and vertex cover [61, 62, 63]; see [65] for a more recent survey. This is partly because many practical problems have set packing constraints that can be conveniently modeled in terms of cliques or independent sets; techniques developed for the stylized problems can then be extended to the real problem. These abstractions have led to the discovery of valid inequalities like clique inequalities, odd-hole inequalities, and others that are used in modern-day MIP solvers to strengthen the LP relaxation [1]. Notable concepts in this area include *conflict generation* [67], used to detect implicit conflicts between variable assignments, and *conflict graphs*, which are data structures used to store these conflicts [3, 16], cf. implication graphs [2].

Due to the complexity of the clique and independent set problems, these MIP techniques generally lack desirable worst-case guarantees. Exceptions do arise when restricted to particular classes of graphs. For example, there are polynomial-size *perfect* MIPs for independent set in comparability graphs and chordal graphs [72] and in graphs with small treewidth [10, 18, 45, 47], which are essentially best-possible [30]. By *perfect*, we mean that their LP relaxation's feasible region equals (or orthogonally projects to) the convex hull of feasible solutions. There exist perfect MIPs for vertex covers of size k that use $O(1.47^k + kn)$ inequalities [17], which can be seen as a polyhedral analogue to FPT algorithms for vertex cover [20, 21]. Other MIPs for clique, such as those given by [55] are not known to provide worst-case guarantees. Basu et al. [8] show that branch-and-bound algorithms that use variable disjunction will visit $\Omega(3^{n/3})$ nodes when applied to the conflict model for independent set in Moon-Moser/Miller-Muller graphs [57, 59] (the disjoint union of triangles).

MIP solvers typically apply branch-and-cut, which is a combination of LP-based branch-and-bound [46] and the cutting plane method [34, 42]. Consequently, the running time is largely dependent on the number of branch-and-

[1] Actually, $(n-d)+1$ subproblems suffice: $n-d$ subproblems $G[V_1]$, $G[V_2]$, …, $G[V_{n-d}]$, and a final subproblem $G[\{v_q, v_{q+1}, \dots, v_n\}]$ where $q = n-d+1$.

bound nodes visited during its execution, as well as the time spent to solve the LP relaxations at each node. Generally speaking, one expects that stronger LP bounds will lead to fewer branch-and-bound nodes being explored. However, just because the gap is small, this does not *imply* a small or polynomial number of branch-and-bound nodes. For example, consider an instance of clique (G, k) in which we are to determine if graph G has a clique of size k. From this, one could quickly create an equivalent instance of clique (G', k) in which $G' = (V', E')$ is the disjoint union of G and a complete graph on $k-1$ vertices (i.e., K_{k-1}). Now consider the following IP:

$$z_{IP} = \max \sum_{i \in V'} x_i \tag{4a}$$

$$x_i + x_j \leq 1 \qquad \forall \{i, j\} \in \binom{V'}{2} \setminus E' \tag{4b}$$

$$\sum_{i \in V'} x_i \leq k \tag{4c}$$

$$x_i \in \{0, 1\} \qquad \forall i \in V'. \tag{4d}$$

This IP has a feasible solution of size $k-1$, since G' has K_{k-1} as a subgraph. Moreover, the root LP bound z_{LP} is at most k by constraint (4c). So, $k-1 \leq z_{IP} \leq z_{LP} \leq k$, and so the (absolute) integrality gap $z_{LP} - z_{IP}$ is at most one, yet solving this IP amounts to solving the clique problem over G. Thus, despite the small integrality gap, we expect this IP to be hard. Moreover, since the k-clique problem is not believed to be FPT, this IP should not be FPT either (with respect to k). In contrast, the strongest MIP that we propose has a small integrality gap $z_{LP} - z_{IP} \leq (d + 1) - \omega = g$ and is solved in $O(2^g n)$ branch-and-bound nodes under a best-bound node selection strategy.

2.4 Disjunctive extended formulations

When constructing MIPs, a helpful modeling primitive is the disjunctive constraint $x \in \cup_{j=1}^{k} P^j$, where each $P_j = \{x \in \mathbb{R}^n \mid A^j x \leq b^j\}$ is a rational polytope. While the disjunctive constraint itself is not permitted in a MIP, it can be modeled as a MIP with new variables y_j, indicating whether x belongs to P^j, and copies of the x variables $w^1, w^2, \ldots, w^k$, as Proposition 1 shows, see [70]. This is convenient for us, as it allows us to decompose the clique problem into multiple subproblems and then write a MIP for the disjunction.

Proposition 1 ([4, 6, 39, 52]) *Given k nonempty polytopes*[2] *$P^j = \{x \in \mathbb{R}^n | A^j x \leq b^j\}$, for $j \in [k]$, the disjunctive constraint $x \in \cup_{j=1}^{k} P^j$ can be*

[2] Something like Proposition 1 holds for unbounded polyhedra, but we will not need this.

modeled as the set of all $(w^1, w^2, \dots, w^k, x, y)$ *satisfying*

$$\text{(Balas)} \quad \begin{aligned} & A^j w^j \le b^j y_j && \forall j \in [k] && \text{(5a)} \\ & \sum_{j=1}^{k} w^j = x && && \text{(5b)} \\ & \sum_{j=1}^{k} y_j = 1 && && \text{(5c)} \\ & w^j \in \mathbb{R}^n && \forall j \in [k] && \text{(5d)} \\ & x \in \mathbb{R}^n && && \text{(5e)} \\ & y \in \{0,1\}^k. && && \text{(5f)} \end{aligned}$$

Moreover, the LP feasible region Q *of model* (5) *is perfect, i.e., satisfies* $\operatorname{proj}_x(Q) = \operatorname{conv}\left(\cup_{j=1}^{k} P^j\right)$*. Further, every extreme point of* Q *has binary* y*.*

One drawback of model (5) is the large number of w variables (5d) and nonzeros in constraints (5b) and (5c). However, in some sense this is unavoidable if we require the convex hull and full generality [22].

In the special case where the polytopes P^j are defined by box constraints, there is hope. Suppose each polytope P^j takes the form

$$P^j = \{x \in \mathbb{R}^n \mid l_i^j \le x_i \le u_i^j, \ \forall i \in [n]\}. \tag{6}$$

In this case, we can write the MIP:

$$\text{(Jeroslow)} \quad \begin{aligned} & \sum_{j=1}^{k} l_i^j y_j \le x_i \le \sum_{j=1}^{k} u_i^j y_j && \forall i \in [n] && \text{(7a)} \\ & \sum_{j=1}^{k} y_j = 1 && && \text{(7b)} \\ & y \in \{0,1\}^k. && && \text{(7c)} \end{aligned}$$

This model (7) is nice because the w variables are not needed. It is also sharp; its LP feasible region projects to $\operatorname{conv}(\cup_{j=1}^{k} P^j)$, [38, Section 3.1]. More general forms of this model hold for polyhedra with the same left-hand-side, $A = A^j$, and have been studied by [5, 11, 38] and more recently [70].

This MIP (7) will be helpful for us, because each of the n different clique subproblems is defined by having some variables x_i being: fixed to one ($l_i^j = u_i^j = 1$); fixed to zero ($l_i^j = u_i^j = 0$); or unfixed ($l_i^j = 0$ and $u_i^j = 1$). Fortunately, model (7) will have just $O(n + m)$ nonzeros when applied to the clique subproblems, even though it may appear to have $\Theta(n^2)$ nonzeros at first glance.

3 Worst-Case Analysis for Existing Clique MIPs

To understand the worst-case behavior of the existing clique MIPs, we consider the simple branch-and-bound algorithm given in Algorithm 1, which directly applies to the conflict model (1) and the dense model (2). With a small change to account for the z variable, it applies to the sparse model (3). In its pseudocode, (F_0, F_1) denotes a node of the branch-and-bound tree, where F_0 and F_1 are the sets of variables fixed to zero and to one at that node, respectively.

Algorithm 1 Branch and bound

```
1: initialize $\mathcal{B} \leftarrow \{(\emptyset, \emptyset)\}$ and $x^* \leftarrow \emptyset$
2: **while** $\mathcal{B} \neq \emptyset$ **do**
3:     select and remove a node $(F_0, F_1)$ from $\mathcal{B}$
4:     solve the LP relaxation $\text{LP}(F_0, F_1)$
5:     **if** $\text{LP}(F_0, F_1)$ is infeasible **then**
6:         continue
7:     let $\bar{x}$ be an optimal (extreme point) solution to $\text{LP}(F_0, F_1)$, with objective $\bar{x}(V)$
8:     **if** $\bar{x}(V) \leq x^*(V)$ **then**
9:         continue
10:    **if** $\bar{x} \in \{0,1\}^n$ **then**
11:        $x^* \leftarrow \bar{x}$
12:        continue
13:    pick a variable $x_i$ with $0 < \bar{x}_i < 1$
14:    add $(F_0 \cup \{x_i\}, F_1)$ and $(F_0, F_1 \cup \{x_i\})$ to $\mathcal{B}$
15: **return** $x^*$
```

In line 1, the root node $(\emptyset, \emptyset)$ is added to the collection of branch-and-bound nodes $\mathcal{B}$. The incumbent solution x^* is initialized as null. In line 2, the algorithm continues as long as some node remains open. Line 3 is *node selection* in which an open node (F_0, F_1) is explored. The associated linear programming relaxation $\text{LP}(F_0, F_1)$ is solved. If the LP is infeasible, the node is pruned by infeasibility in line 6. If the LP's objective value $\bar{x}(V) = \sum_{i \in V} \bar{x}_i$ is no better than that of x^*, the node is pruned by bound in line 9. Next, if the LP solution $\bar{x}$ is integer, then the incumbent x^* is updated, and the node is pruned by integrality in line 12. Finally, a variable x_i is selected for branching. Line 14 creates the left child $(F_0 \cup \{x_i\}, F_1)$ and right child $(F_0, F_1 \cup \{x_i\})$.

For the worst-case analysis, recall the following definitions about rooted trees from [25]. A *rooted tree* is a tree with a special node called the *root*. For each node v of the rooted tree, there is a unique (simple) path from the root r to v; the nodes along this path (including v) are the *ancestors* of v. If u is an ancestor of v, then v is a *descendant* of u. If u and v are neighbors and u is an ancestor of v, then u is the *parent* of v and v is a *child* of u. If two nodes have the same parent, they are *siblings*. If a node has zero descendants besides itself, then it is a *leaf*. The *depth* of a node is its distance from the root.

When branch-and-bound is applied to an integer program, there is a corresponding rooted tree which we denote by $\mathcal{T}$. The nodes of $\mathcal{T}$ are the pairs (F_0, F_1) of fixed variables encountered during the algorithm, and the root of

$\mathcal{T}$ is the node $(\emptyset, \emptyset)$. The depth of node (F_0, F_1) is equal to $|F_0|+|F_1|$. If a variable x_i is branched on at node (F_0, F_1), then (F_0, F_1) has two children $(F_0 \cup \{x_i\}, F_1)$ and $(F_0, F_1 \cup \{x_i\})$.

Proposition 2 *When branch-and-bound is applied to the conflict, dense, and sparse models, it visits $O(\varphi^n)$ nodes, where $\varphi < 1.6181$ is the golden ratio.*

Proof At each branch-and-bound node (F_0, F_1), the number of "free" variables is $k = n - |F_0| - |F_1| - |I_0(F_0, F_1)|$, where

$$I_0(F_0, F_1) = \{x_v \mid x_v \notin F_0 \text{ and } v \text{ has a nonneighbor } u \text{ with } x_u \in F_1\} \tag{8}$$

is the set of variables that are *implicitly* fixed to zero by F_1 and conflict constraints (1b) or big-M constraints (2b) or (3c). If $T(k)$ denotes the number of its descendants, we have the recurrence

$$T(k) \leq \begin{cases} 1 + T(k-2) + T(k-1) & \text{if } k \geq 2 \\ 1 & \text{if } k \leq 1. \end{cases}$$

The reason is as follows. Suppose (F_0, F_1) were not pruned, then it had an LP solution $\bar{x}$ and Algorithm 1 chose to branch on some free variable, say x_i, that was fractional, i.e., $0 < \bar{x}_i < 1$. See that i has a nonneighbor j with x_j free, since otherwise $\bar{x}_i$ can be increased to one and still be LP feasible. So, in the left child there is one fewer free variable (x_i), while in the right child there are at least two fewer free variables (x_i and x_j), so $T(k) \leq 1 + T(k-1) + T(k-2)$. Solving the recurrence gives $T(n) = O(\varphi^n)$, see [32, Chapter 2.1]. □

Below we show that the analysis in Proposition 2 is tight. That is, the conflict model (which is stronger than the dense and sparse models) sometimes visits $\Theta(\varphi^n)$ branch-and-bound nodes. In fact, this occurs on the co-lollipops $\bar{L}_{3,p}$, which are the complements of the lollipops $L_{3,p}$ depicted in Figure 1.

Theorem 1 *When branch-and-bound is applied to the conflict model, it can visit $\Theta(\varphi^n)$ nodes for the co-lollipops $\bar{L}_{3,p}$.*

Proof It suffices to show that Algorithm 1 visits $\Theta(\varphi^n)$ branch-and-bound nodes when the *lollipops* $L_{3,p}$ are solved with the *stable set* conflict model, whose LP relaxation is the fractional stable set polytope:

$$\text{FSTAB}(G) = \{x \in [0,1]^n \mid x_i + x_j \leq 1, \ \forall \{i,j\} \in E\}.$$

We will suppose that Algorithm 1 makes the following choices:

1. at each node, pick an LP solution with the most fractional coordinates;
2. among the fractional variables, branch on the one x_t with largest index t;
3. for the node-selection strategy, prioritize nodes with fractional solutions.

First, we claim that the all-half vector $(\frac{1}{2}, \frac{1}{2}, \dots, \frac{1}{2})$ is an optimal extreme point of FSTAB$(L_{3,p})$. It is extreme because it is feasible and satisfies the n linearly independent constraints $x_i + x_j \leq 1$ at equality. Now, when p is even, optimality follows because any LP-feasible solution $\bar{x}$ satisfies

$$\begin{aligned}\bar{x}(V) &= (\bar{x}_{-1} + \bar{x}_0) + (\bar{x}_1 + \bar{x}_2) + \cdots + (\bar{x}_{p-1} + \bar{x}_p)\\ &\leq 1 + 1 + \cdots + 1 = 1 + \frac{p}{2} = \frac{n}{2},\end{aligned}$$

and the all-half vector meets this bound. Meanwhile, when p is odd,

$$\begin{aligned}\bar{x}(V) &= \frac{1}{2}(\bar{x}_{-1} + \bar{x}_0) + \frac{1}{2}(\bar{x}_{-1} + \bar{x}_1) + \frac{1}{2}(\bar{x}_0 + \bar{x}_1)\\ &\quad + (\bar{x}_2 + \bar{x}_3) + (\bar{x}_4 + \bar{x}_5) + \cdots + (\bar{x}_{p-1} + \bar{x}_p)\\ &\leq \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + 1 + \cdots + 1 = 1.5 + \frac{p-1}{2} = \frac{n}{2}.\end{aligned}$$

Now, consider the execution of Algorithm 1. First, the root LP is solved, giving the all-half vector (by choice 1). By choice 2, the algorithm branches on x_p. In the right child, the fixing $x_p = 1$ forces $x_{p-1} = 0$; the remaining LP is equivalent to that for $L_{3,p-2}$. In the left child, where $x_p = 0$, the remaining LP is equivalent to that for $L_{3,p-1}$. In this way, all (nontrivial) LPs that are encountered are over lollipop graphs with fractional LP solutions. Consider a node (F_0, F_1) in the branch-and-bound tree and let k denote the number of variables not (explicitly or implicitly) fixed to a binary value. If $T(k)$ denotes the number of its descendants, we have the recurrence

$$T(k) = \begin{cases} 1 + T(k-2) + T(k-1) & \text{if } k \geq 3\\ 1 & \text{if } k \leq 2.\end{cases}$$

By choice 3, there is no incumbent with which to prune until all fractional nodes are processed. Solving the recurrence gives $T(p) = \Theta(\varphi^p) = \Theta(\varphi^n)$. $\square$

The co-lollipops $\bar{L}_{3,p}$ are quite unlike real-life graphs. For example, they have clique-core gaps g approximately equal to $n/2$, much larger than that of the real-life graphs in Table 1. One may wonder—is branch-and-bound quick when g is small? Theorem 2 shows this is false. The proof holds for any node selection strategy; even a best-bound strategy visits $\Omega(1.2599^n)$ nodes.

Theorem 2 *When branch-and-bound is applied to the conflict model, it can visit exponentially many nodes even when the clique-core gap g is zero.*

Proof To prove the claim, consider graphs of the form $G_q = (V_q, E_q)$, where

$$V_q = \{1, 1', 1''\} \cup \{2, 2', 2''\} \cup \cdots \cup \{q, q', q''\} \tag{9}$$

and E_q is such that $C = \{1, 2, \dots, q\}$ is a clique, and each of its vertices $c \in C$ neighbors all vertices of V except for c' and c''. Figure 2 depicts the case $q = 4$.

The graph G_q has degeneracy $d = q-1$; if $a \frown b$ denotes the concatenation of a and b, a degeneracy ordering is $(1'', 2'', \dots, q'') \frown (1', 2', \dots, q') \frown (1, 2, \dots, q)$.

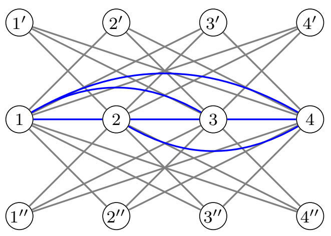


Fig. 2: The graph G_q when $q = 4$.

The clique number ω equals q, because C is a clique of size q and because V can be partitioned into q independent sets of the form $\{c, c', c''\}$. So, the clique-core gap g equals zero, as $g := (d+1) - \omega = (q-1+1) - q = 0$.

The following claim says that, if only variables x_c with $c \in C$ have been branched on, all "free" variables can be set to one-half. In fact, this is optimal and extreme. The claim uses the notation $I_0(F_0, F_1)$, as defined previously (8).

Claim 1 *Suppose branch-and-bound node (F_0, F_1) has $F_0 \cup F_1 \subseteq \{x_c \mid c \in C\}$. Then, x^*, defined below, is an optimal extreme point of* $\mathrm{LP}(F_0, F_1)$.

$$x_i^* := \begin{cases} 1 & \textit{if } x_i \in F_1 \\ 0 & \textit{if } x_i \in F_0 \cup I_0(F_0, F_1) \\ \frac{1}{2} & \textit{if otherwise} \end{cases}$$

Proof of Claim By the graph's structure, any variable x_c that is fixed to one forces two variables $x_{c'}$ and $x_{c''}$ to zero, so $|I_0(F_0, F_1)| = 2|F_1|$. The variables x_c that are fixed to zero have $c \in C$, which is disjoint from $I_0(F_0, F_1)$, implying that $|F_0 \cup I_0(F_0, F_1)| = |F_0| + |I_0(F_0, F_1)|$. Thus, the point x^* has objective value

$$\begin{aligned} x^*(V) &= 1 \cdot |F_1| + 0 \cdot |F_0 \cup I_0(F_0, F_1)| + \frac{1}{2} \cdot \Big(3q - |F_1| - |F_0 \cup I_0(F_0, F_1)|\Big) \\ &= |F_1| + \frac{1}{2}\Big(3q - |F_1| - |F_0| - 2|F_1|\Big) = \frac{3}{2}q - \frac{1}{2}|F_0 \cup F_1|, \end{aligned}$$

which is optimal as any $\bar{x}$ that is feasible for the LP relaxation $\mathrm{LP}(F_0, F_1)$ has:

$$\bar{x}(V) = \sum_{c \in C} (\bar{x}_c + \bar{x}_{c'} + \bar{x}_{c''}) \tag{10a}$$

$$= \sum_{c \in C : x_c \in F_0} (\bar{x}_{c'} + \bar{x}_{c''}) + \sum_{c \in C : x_c \in F_1} 1 + \sum_{c \in C : x_c \notin F_0 \cup F_1} (\bar{x}_c + \bar{x}_{c'} + \bar{x}_{c''}) \tag{10b}$$

$$\leq |F_0 \cup F_1| + \sum_{c \in C : x_c \notin F_0 \cup F_1} \frac{1}{2} \left((\bar{x}_c + \bar{x}_{c'}) + (\bar{x}_c + \bar{x}_{c''}) + (\bar{x}_{c'} + \bar{x}_{c''})\right) \tag{10c}$$

$$\leq |F_0 \cup F_1| + \frac{3}{2}(q - |F_0 \cup F_1|) = \frac{3}{2}q - \frac{1}{2}|F_0 \cup F_1| = x^*(V). \tag{10d}$$

Meanwhile, x^* is extreme, because it is the unique point satisfying the following n constraints from $\mathrm{LP}(F_0, F_1)$ at equality.

$$\begin{aligned}
&x_c \geq 0 && \forall x_c \in F_0 && \text{(11a)}\\
&x_c \leq 1 && \forall x_c \in F_1 && \text{(11b)}\\
&x_c + x_{c'} \leq 1 && \forall c \in C : x_c \notin F_0 \cup F_1 && \text{(11c)}\\
&x_{c'} + x_{(c+1)'} \leq 1 && \forall c \in \{1, 2, \dots, q-1\} && \text{(11d)}\\
&x_{c''} + x_{(c+1)''} \leq 1 && \forall c \in \{1, 2, \dots, q-1\} && \text{(11e)}\\
&x_{1'} + x_{1''} \leq 1 && && \text{(11f)}\\
&x_{1'} + x_{2''} \leq 1. && && \text{(11g)}
\end{aligned}$$

To see uniqueness, consider $x_{1''} + x_{2''} \leq 1$ from (11e) and inequalities (11f) and (11g), which together form a triangle of conflicts. For them to hold at equality, we must have $x_{1'} = x_{1''} = x_{2''} = 1/2$. Then, the conflicts (11d) and (11e) propagate 1/2 values along the paths $(1', 2', \dots, q')$ and $(2'', \dots, q'')$, and then into the "free" vertices from C via (11c). The 0-1 bounds (11a) and (11b) complete the solution x^*. ■

By Claim 1, the root node's LP objective is $\frac{3}{2}q$. Meanwhile, the optimal IP objective is far away at q. Finally, we observe that branching on variables x_c with $c \in C$ leads to slow, predictable progress in the LP bound. Specifically, by Claim 1, branching on a variable x_c with $c \in C$ creates two child nodes, each having an LP objective that is one-half less than that of its parent: in the left child, the previously "free" x_c is added to F_0, while other variables remain unchanged; in the right child, x_c is added to F_1, forcing $x_{c'}$ and $x_{c''}$ into I_0. In both cases, the LP objective decreases by one-half.

In this way, if Algorithm 1 prioritizes variables x_c with $c \in C$ for branching, then no node whose depth $|F_0 \cup F_1|$ is *less* than q will be pruned. This follows since its LP objective will be greater than q by (10d). So, all $2^{q+1} - 1$ nodes with depth *at most* q will be visited, and $2^{q+1} - 1 > 2^q = 2^{n/3} > 1.2599^n$. □

4 New Clique MIPs

Here, we conceive of four new MIPs for the clique problem. They all rely on a vertex ordering $(v_1, v_2, \dots, v_n)$, which we take to be a degeneracy ordering, see Section 2.2. The MIPs differ depending on the choice of base model (conflict vs. sparse) and the choice of extended formulation (Balas vs. Jeroslow). For brevity, we only explicitly define and analyze the two extremes: the weakest model (sparse-Jeroslow) and the strongest model (conflict-Balas).

The new MIPs rely on the decomposition given in Lemma 1 which allows us to break the clique problem into n subproblems. Nicely, if a degeneracy ordering is used, then each of these subproblems has at most d free variables.

Lemma 1 (clique decomposition, folklore) *Suppose the vertices of a graph $G = (V, E)$ are ordered $(v_1, v_2, \dots, v_n)$. Then, every nonempty clique $Q \subseteq V$*

of graph G satisfies the following inclusions for some (unique) $i \in [n]$.

$$\{v_i\} \subseteq Q \subseteq N[v_i] \cap \{v_i, v_{i+1}, \dots, v_n\}. \tag{12}$$

Proof Consider a clique $Q = \{v_{i_1}, v_{i_2}, \dots, v_{i_q}\}$ indexed so that its vertices follow the ordering, i.e., $i_1 < i_2 < \cdots < i_q$. Then,

$$\{v_{i_1}\} \subseteq Q \subseteq N[v_{i_1}] \cap \{v_{i_1}, v_{i_1+1}, \dots, v_n\},$$

as desired. To prove uniqueness, observe that vertices v that do not belong to Q will not satisfy $\{v\} \subseteq Q$, thus violating the first inclusion of (12). Meanwhile, vertices $v_{i_j} \in Q$ with $j \geq 2$, will not satisfy $Q \subseteq N[v_{i_j}] \cap \{v_{i_j}, v_{i_j+1}, \dots, v_n\}$ as v_{i_1} will belong to the left side but not the right, thus violating the second inclusion of (12). So, $i_1 \in [n]$ is the unique index satisfying (12). □

4.1 Sparse-Jeroslow model

By Lemma 1, we can break the clique problem into n subproblems. In each subproblem $j \in [n]$, one vertex is fixed in the clique, some vertices are "free", and others are fixed out of the clique. The associated subsets of vertices are:

$$\begin{aligned} S_j^+ &:= \{v_j\} \\ S_j &:= N(v_j) \cap \{v_{j+1}, v_{j+2}, \dots, v_n\} \\ S_j^- &:= V \setminus (S_j^+ \cup S_j). \end{aligned}$$

In the sparse-Jeroslow model, we introduce a binary variable y_j for each subproblem j. These variables indicate which subproblem will select our clique.

$$\text{(sparse-Jeroslow)} \qquad \begin{aligned} \max \quad & z && && \text{(13a)} \\ & z = \sum_{i \in V} x_i && && \text{(13b)} \\ & z - \sum_{j \in N[i]} x_j \leq M_i(1 - x_i) && \forall i \in V && \text{(13c)} \\ & \sum_{j \,:\, i \in S_j^+} y_j \leq x_i && \forall i \in V && \text{(13d)} \\ & x_i \leq \sum_{j \,:\, i \in S_j^+ \cup S_j} y_j && \forall i \in V && \text{(13e)} \\ & \sum_{j=1}^{n} y_j \leq 1 && && \text{(13f)} \\ & (x, y) \in \{0,1\}^{n+n}. && && \text{(13g)} \end{aligned}$$

This is the same as the sparse model, except for the addition of Jeroslow's constraints (13d), (13e), (13f), which come from model (7). Notice that the bound constraints (13d) and (13e) are sparser than in Jeroslow's model; the reason is that some of the bounds are zero and can be omitted. Also, notice that the constraint $\sum_{j=1}^n y_j = 1$ has been relaxed to $\sum_{j=1}^n y_j \leq 1$ to allow for the empty solution (where $x = \mathbf{0}$ and $y = \mathbf{0}$).

The sparse-Jeroslow model has $2n+1$ variables and $3n+2$ constraints. It has $\Theta(n+m)$ nonzeros; the sparse model already had $\Theta(n+m)$ nonzeros, and Jeroslow's constraints add $\Theta(n+m)$ nonzeros, in part because

$$\sum_{j=1}^{n} |S_j| = \sum_{j=1}^{n} |N(v_j) \cap \{v_j, v_{j+1}, \dots, v_n\}| = m. \tag{14}$$

Additionally, the model itself can be constructed in $\Theta(n+m)$ time, including the degeneracy ordering. Finally, by [38], we know that if constraints (13c) were omitted from the sparse-Jeroslow model (13), then its LP relaxation would have integer extreme points. Of course, the addition of constraints (13c) destroys this integrality. Nevertheless, the root LP bound is still strong. Proposition 3 shows it is at most $d+1$. Consequently, its (absolute) integrality gap is at most g, by $z_{LP} - z_{IP} \leq (d+1) - \omega = g$.

Proposition 3 *The LP objective of the sparse-Jeroslow model is at most* $d+1$.

Proof If $(\bar{x}, \bar{y}, \bar{z})$ is LP feasible, then letting $\text{rdeg}(v_j) = |N(v_j) \cap \{v_{j+1}, \dots, v_n\}|$,

$$\sum_{i \in V} \bar{x}_i \leq \sum_{i \in V} \sum_{j : i \in S_j^+ \cup S_j} \bar{y}_j = \sum_{j=1}^{n} (\text{rdeg}(v_j) + 1)\bar{y}_j \leq \sum_{j=1}^{n} (d+1)\bar{y}_j \leq d+1,$$

where the first inequality holds by (13e), the equality by rearrangement, the next inequality by the degeneracy ordering, and the last by (13f). □

4.2 Conflict-Balas model

We now turn to the conflict-Balas model, which is obtained by writing the conflict model for each subproblem j:

$$\begin{aligned} P^j := \{x \in \mathbb{R}^n \mid x_u + x_v &\leq 1, && \forall \{u,v\} \in \bar{E}(S_j); \\ x_i &= 0, && \forall i \in S_j^-; \\ x_i &= 1, && \forall i \in S_j^+; \\ 0 \leq x_i &\leq 1, && \forall i \in S_j\}, \end{aligned}$$

and taking their union with Balas's model (5). It uses new variables w_i^j that equal one when vertex $i \in V$ is picked from subproblem $j \in [n]$. Although there are n^2 many w variables, most of them will equal zero by Balas's constraints, so model (16) omits these variables, w_i^j with $i \in S_j^-$. After this reduction, the conflict-Balas model has $\Theta(n+m)$ variables (recalling (14)), $O(nd^2)$ constraints, and $O(nd^2)$ nonzeros. While the x and y variables could be projected out via constraints (16c) and (16d), we leave them in for analysis.

$$
\text{(conflict-Balas)} \quad
\begin{aligned}
\max \quad & \sum_{i\in V} x_i && && \text{(16a)}\\
& w_u^j + w_v^j \le y_j && \forall \{u,v\} \in \bar{E}(S_j),\ j\in[n] && \text{(16b)}\\
& \sum_{j:\ i\in S_j^+\cup S_j} w_i^j = x_i && \forall i \in V && \text{(16c)}\\
& w_i^j = y_j && \forall i \in S_j^+,\ j\in[n] && \text{(16d)}\\
& 0 \le w_i^j \le y_j && \forall i \in S_j,\ j\in[n] && \text{(16e)}\\
& \sum_{j=1}^n y_j \le 1 && && \text{(16f)}\\
& w, y \text{ binary.} && && \text{(16g)}
\end{aligned}
$$

The following remark states that the x variables, although "unrestricted" in the conflict-Balas model (16), will be binary in any MIP-feasible solution.

Remark 1 If $(\bar{w},\bar{x},\bar{y})$ is LP feasible for the conflict-Balas model (16), then $\bar{x}\in[0,1]^n$. Further, if $(\bar{w},\bar{x},\bar{y})$ is MIP feasible, then $\bar{x}$ is binary.

Proof If $(\bar{w},\bar{x},\bar{y})$ is LP feasible, then $\bar{x}_i = \sum_{j:\ i\in S_j^+\cup S_j} \bar{w}_i^j$ by (16c) and

$$0 \le \sum_{j:\ i\in S_j^+\cup S_j} \bar{w}_i^j \le \sum_{j: i\in S_j^+\cup S_j} \bar{y}_j \le 1,$$

where the middle inequality holds by (16d) and (16e), and the last inequality holds by (16f) and $\bar{y}\ge 0$. Further, $\bar{x}_i$ is integer (and thus binary) since it is written as the sum of binary values $\bar{w}_i^j$. □

We note that the conflict-Balas model is at least as strong as the sparse-Jeroslow model, as any point $(\bar{w},\bar{x},\bar{y})$ feasible to the conflict-Balas model immediately gives the related point $(\bar{x},\bar{y},\bar{z})$ with $\bar{z}=\bar{x}(V)$ that is feasible for the sparse-Jeroslow model. To see this, observe that $\bar{x}$ is also LP feasible for the conflict model over G, so $(\bar{x},\bar{z})$ is LP feasible for the (weaker) sparse model over G. The sparse model and sparse-Jeroslow model differ only in constraints (13d), (13e), (13f), which $(\bar{x},\bar{y},\bar{z})$ satisfies (through (16c)) by (16d), (16e), and (16f), respectively. Moreover, this inclusion can be strict. For example, the conflict-Balas model is perfect when applied to the cycle on vertices $\{1,2,3,4,5\}$ as Proposition 4 below guarantees and gives an LP bound of 2, but the sparse-Jeroslow model gives an LP bound of 2.5. Since the conflict-Balas model is at least as strong as the sparse-Jeroslow model, it also gives an LP bound of at most $d+1$ and (absolute) integrality gap at most g, see Proposition 3. Additionally, Proposition 4 identifies when the conflict-Balas model is perfect.

Proposition 4 *The conflict-Balas model is perfect if and only if each subproblem's S_j can be partitioned into two cliques (i.e., each $G[S_j]$ is co-bipartite).*

Proof For arbitrary graphs G', the conflict model for $\omega(G')$ is perfect if and only if G' is co-bipartite [68, Theorem 19.7, p. 319]. So, if each subproblem's S_j induces a co-bipartite subgraph, then each P^j is integral and is thus the convex hull of cliques from $G[S_j]$. So, $\text{conv}(\cup_j P^j)$ is integral and by Lemma 1 is thus the convex hull of all cliques from G, i.e., $\text{conv}(\cup_j P^j)$ is the clique polytope of G. Then, by Proposition 1, the conflict-Balas model (16) projects to $\text{conv}(\cup_j P^j)$, the clique polytope, and so the conflict-Balas model is perfect.

Meanwhile, if some subproblem's S_j does not induce a co-bipartite subgraph, then it contains an odd antihole $H \subseteq S_j$. That is, H has an odd number of vertices and its induced subgraph $G[H]$ is the complement of a cycle graph. Then see that there is a point $(\bar{w}, \bar{x}, \bar{y})$ that belongs to the LP of conflict-Balas model, but $\bar{x}$ lies outside the clique polytope because it violates the valid inequality $x(H) \leq \lfloor |H|/2 \rfloor$. Specifically, set $\bar{w}_i^j = 1$ for $i \in S_j^+$, $\bar{w}_i^j = 1/2$ for $i \in H$, and other w's to zero; set $\bar{x}$ and $\bar{y}$ by (16c) and (16d). □

By Proposition 4, the conflict-Balas model (16) is perfect and has linear size $O(n)$ when the degeneracy is at most two; this includes the class of series-parallel graphs [12]. For such graphs, each subproblem's S_j can trivially be partitioned into two cliques, as $|S_j| \leq d \leq 2$, so Proposition 4 applies. Since $d \leq 2$, there are $O(nd^2) = O(n)$ nonzeros, and thus $O(n)$ variables and constraints.

5 Worst-Case Analysis for New Clique MIPs

Now, we analyze the worst-case performance of the sparse-Jeroslow and conflict-Balas models when solved with simple branch-and-bound algorithms. The algorithms are essentially the same as Algorithm 1, but with minor and straightforward changes because of the additional variables w, y, and z. We will also consider the effects of other changes, like a best-bound node selection strategy.

5.1 Shrunk trees

When conducting our analysis, we use the rooted tree terminology from Section 3. For example, associated with the algorithm's execution, there is rooted tree $\mathcal{T}$ whose nodes are pairs (F_0, F_1) of fixed variables. Associated with each clique subproblem $j \in \{1, 2, \dots, n\}$ there is also a *shrunk tree* $\mathcal{T}_j$ which includes the portions of the (full) branch-and-bound tree that relate to subproblem j. The *lowest common ancestor* (LCA) of a subset of nodes is the node that is an ancestor to all of them that has the largest depth.

Definition 2 The *shrunk tree* for subproblem j is denoted by $\mathcal{T}_j$. Its node set $V(\mathcal{T}_j)$ consists of two types of nodes:

(A) nodes from $V(\mathcal{T})$ whose LP solution has $\bar{y}_j = 1$, and

(B) (other) nodes from $V(\mathcal{T})$ that are LCAs of type A node pairs.

Consider two distinct nodes u and v from $V(\mathcal{T}_j)$. If the simple path between them in $\mathcal{T}$ crosses no other nodes from $V(\mathcal{T}_j)$, then the edge $\{u, v\}$ belongs to $E(\mathcal{T}_j)$. The root of $\mathcal{T}_j$ is the LCA of $V(\mathcal{T}_j)$ in $\mathcal{T}$.

Figure 3 illustrates the branch-and-bound tree $\mathcal{T}$ and shrunk trees $\mathcal{T}_1$ and $\mathcal{T}_2$ for the conflict-Balas model. Figure 3(a) depicts graph G with degeneracy ordering $1, 2, 3, \ldots, 9$. Figure 3(d) shows the branch-and-bound tree $\mathcal{T}$. At its root, the LP solution $(\bar{w}, \bar{x}, \bar{y})$ has objective 3.5, which is achieved, say, by setting $\bar{y}_1 = 1$, $\bar{w}_1^1 = 1$ (indicated by black fill), and $\bar{w}_2^1 = \bar{w}_4^1 = \bar{w}_6^1 = \bar{w}_8^1 = \bar{w}_9^1 = 1/2$ (gray fill) from the first subproblem. Down the tree, we see solutions in which some w variables equal zero (indicated by white fill). Finally, Figure 3(e) and Figure 3(f) depict the shrunk trees $\mathcal{T}_1$ and $\mathcal{T}_2$. They illustrate some ways in which shrunk trees are different than branch-and-bound trees. For example, while the nodes of $\mathcal{T}$ all have zero or two children, $\mathcal{T}_2$ has a node with one child. Also, the nodes of shrunk trees are not necessarily contiguous in $\mathcal{T}$, as $\mathcal{T}_1$ shows. Nevertheless, shrunk trees do have some helpful traits.

Lemma 2 *If u and v are nodes of the shrunk tree $\mathcal{T}_j$, then their LCA in $\mathcal{T}$ also belongs to the shrunk tree.*

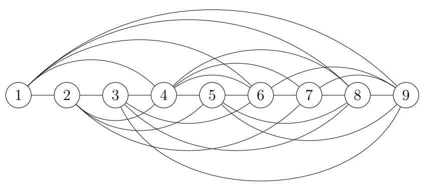
Proof This is immediate when u and v are of type A. Generally, observe that every node of $\mathcal{T}_j$ has a descendant of type A. So, if u and v are nodes of $\mathcal{T}_j$ then the type A descendants of theirs, u' and v', have a LCA t' that belongs to $V(\mathcal{T}_j)$ by definition of shrunk tree, and t' is also the LCA of u and v. □

Proposition 5 *The shrunk tree $\mathcal{T}_j$ is indeed a tree.*

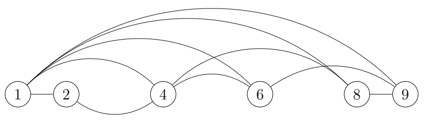
Proof The shrunk tree $\mathcal{T}_j$ is a forest because it is a minor of the tree $\mathcal{T}$. So, to show it is a tree, we just need it to be connected. Consider arbitrary nodes u and v from $V(\mathcal{T}_j)$. By Lemma 2, their LCA t also belongs to $V(\mathcal{T}_j)$, so it suffices to show that there is a u,t-path in $\mathcal{T}_j$. Consider the path from u to t in $\mathcal{T}$, and let $u = i_0, i_1, i_2, \ldots, i_q = t$ be the vertices along this path (in sequence) that belong to $V(\mathcal{T}_j)$. By definition of shrunk tree, each pair $\{i_k, i_{k+1}\}$ of consecutive nodes in this sequence is an edge in $E(\mathcal{T}_j)$, thus giving a path from u to t in $\mathcal{T}_j$. □

Proposition 6 *All shrunk tree nodes of the conflict-Balas model have type A.*

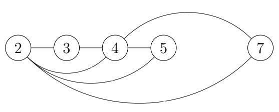
Proof For contradiction purposes, suppose that t is a type B node from the shrunk tree $\mathcal{T}_j$ and that it is the LCA of type A nodes u and v. At node t, the algorithm branched on a variable. It was not an x variable, as they are defined to be continuous. It was not a y variable (nor the w equivalent (16d)), because the extreme points of the conflict-Balas model have binary y, see Proposition 1. So, it must have been a variable of the form w_i^k with $i \in S_k$. In the first case, where $k = j$, the LP solution $(\bar{w}, \bar{x}, \bar{y})$ at node t has $0 < \bar{w}_i^j \leq \bar{y}_j$ by (16e) and $\bar{y}_j$ is binary, so t is actually a type A node, a contradiction. In the other case, where $k \neq j$, the branch with $w_i^k = 1$ forces $y_k = 1$ by (16e), contradicting the assumption that LP solutions at u and v have $y_j = 1$. □



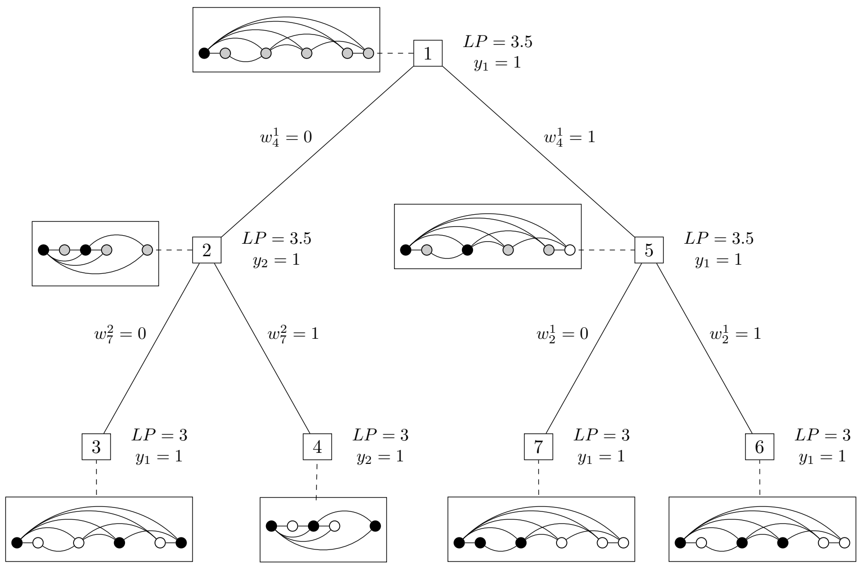
(a) Example graph G.



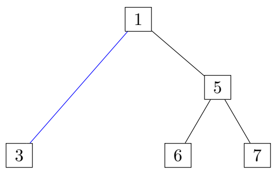
(b) The subgraph $G[S_1 \cup S_1^+]$.

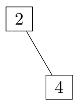


(c) The subgraph $G[S_2 \cup S_2^+]$.



(d) The branch-and-bound tree $\mathcal{T}$.





(e) The shrunk tree $\mathcal{T}_1$. (f) The shrunk tree $\mathcal{T}_2$.

Fig. 3: Illustration of shrunk trees. The vertices of graph G are round, while the nodes of the "full" tree $\mathcal{T}$ and shrunk tree $\mathcal{T}_j$ are square. Note that the y_j values next to the *nodes* indicate the LP relaxation solution (not variable fixings), and the w_i^j values next to the *tree edges* denote variable fixings.

5.2 Analysis for sparse-Jeroslow

For our analysis with the sparse-Jeroslow model, we observe that the nodes visited by branch-and-bound can be partitioned into three sets:

N_1 : nodes (F_0, F_1) for which $\mathrm{LP}(F_0, F_1)$ is infeasible;
N_2 : nodes (F_0, F_1) whose LP solution $(\bar{x}, \bar{y}, \bar{z})$ has fractional $\bar{y} \notin \{0,1\}^n$;
N_3 : nodes (F_0, F_1) whose LP solution $(\bar{x}, \bar{y}, \bar{z})$ has integral $\bar{y} \in \{0,1\}^n$.

Denote their sizes by $t_1 = |N_1|$, $t_2 = |N_2|$, and $t_3 = |N_3|$.

In the following lemma, we prioritize the y variables for branching. That is, if the LP solution $(\bar{x}, \bar{y}, \bar{z})$ has fractional $\bar{y}$, then one of its variables y_j must be selected for branching. Most MIP solvers allow users to set the branching priority of variables.

Lemma 3 *If the y variables are prioritized for branching, then all nodes visited for the sparse-Jeroslow model are LP feasible (i.e., $t_1 = 0$), and the number of nodes with fractional $\bar{y}$ is not too big ($t_2 \leq 1 + 3t_3$), so $t_1 + t_2 + t_3 \leq 1 + 4t_3$.*

Proof First we show $t_1 = 0$. The root node is clearly LP feasible, so consider a non-root node (F_0, F_1) encountered by the algorithm. Its parent (F_0', F_1') is LP feasible, say at $(\bar{x}, \bar{y}, \bar{z})$. See that $Q := \{v \in V \mid \bar{x}_v = 1\}$ is a clique. Additionally, $Q \cup \{v\}$ is a clique for each v with $\bar{x}_v > 0$, because otherwise v has a nonneighbor q in Q with $\bar{x}_q = 1$ giving the contradiction

$$0 < \bar{x}_v \leq \sum_{j \in \bar{N}(q)} \bar{x}_j = \bar{z} - \sum_{j \in N[q]} \bar{x}_j \leq M_q(1 - \bar{x}_q) = 0,$$

where the last inequality holds by (13c). At parent node (F_0', F_1'), the algorithm branched on a variable, fixing it to zero or one, giving four cases for the child:

1. some variable x_v was fixed to zero, $(F_0, F_1) = (F_0' \cup \{x_v\}, F_1')$;
2. some variable x_v was fixed to one, $(F_0, F_1) = (F_0', F_1' \cup \{x_v\})$;
3. some variable y_k was fixed to zero, $(F_0, F_1) = (F_0' \cup \{y_k\}, F_1')$;
4. some variable y_k was fixed to one, $(F_0, F_1) = (F_0', F_1' \cup \{y_k\})$.

For each case, we construct a feasible point $(\hat{x}, \hat{y}, \hat{z})$ for $\mathrm{LP}(F_0, F_1)$.

Case 1: $x_v = 0$. Since x_v was branched on, $\bar{y} \in \{0,1\}^n$ holds by the branching priority. Let $\hat{y} = \bar{y}$, $\hat{x} = x^Q$ be the characteristic vector of clique Q, and $\hat{z} = \sum_{i \in V} \hat{x}_i$ be its cardinality. Clearly, $(\hat{x}, \hat{y}, \hat{z})$ satisfies constraints (13b), (13c), (13f), and the 0-1 fixings required by (F_0, F_1). Now we show constraints (13d) and (13e) hold; recall that they require each x_i variable to lie between two expressions:

$$\sum_{j : i \in S_j^+} y_j \leq x_i \leq \sum_{j : i \in S_j^+ \cup S_j} y_j.$$

Observe that each of the sums $\sum_{j:i\in S_j^+} \hat{y}_j$ and $\sum_{j:i\in S_j^+\cup S_j} \hat{y}_j$ take binary values as $\hat{y} \in \{0,1\}^n$ and $\sum_{j=1}^n \hat{y}_j \leq 1$. So, if $i \in V \setminus Q$, then

$$\sum_{j:i\in S_j^+} \hat{y}_j = \sum_{j:i\in S_j^+} \bar{y}_j = 0 \leq 0 = \hat{x}_i = 0 \leq \sum_{j:i\in S_j^+\cup S_j} \hat{y}_j.$$

Note that $\sum_{j:i\in S_j^+} \bar{y}_j = 0$, as otherwise $1 = \sum_{j:i\in S_j^+} \bar{y}_j \leq \bar{x}_i < 1$. If $i \in Q$, then

$$\sum_{j:i\in S_j^+} \hat{y}_j \leq \sum_{j=1}^n \hat{y}_j \leq 1 = \hat{x}_i = 1 \leq 1 = \sum_{j:i\in S_j^+\cup S_j} \bar{y}_j = \sum_{j:i\in S_j^+\cup S_j} \hat{y}_j.$$

Note that $1 = \sum_{j:i\in S_j^+\cup S_j} \bar{y}_j$, as otherwise $0 < \bar{x}_i \leq \sum_{j:i\in S_j^+\cup S_j} \bar{y}_j = 0$.

Case 2: $x_v = 1$. The proof is the same as Case 1, using $Q \cup \{v\}$ instead of Q.

Case 3: $y_k = 0$. If $Q = \emptyset$ then $(\hat{x}, \hat{y}, \hat{z}) = (\mathbf{0}, \mathbf{0}, 0)$ proves the claim, so suppose $Q \neq \emptyset$. There exists $k' \neq k$ for which $\bar{y}_{k'} > 0$, as otherwise picking any vertex $i \in Q$ gives the contradiction

$$1 = \bar{x}_i \leq \sum_{j:i\in S_j^+\cup S_j} \bar{y}_j \leq \sum_{j=1}^n \bar{y}_j = \bar{y}_k < 1.$$

Let v be the (only) vertex of $S_{k'}^+$. Observe that $\bar{y}_{k'} \leq \bar{x}_v$ by constraints (13d), so $x_v \notin F_0'$ and also $Q \cup \{v\}$ is a clique. Let $\hat{y}$ be the binary vector that has $\hat{y}_j = 1$ if and only if $j = k'$, $\hat{x} = x^{Q\cup\{v\}}$ be the characteristic vector of $Q \cup \{v\}$, and $\hat{z} = \hat{x}(V)$ be its cardinality. Clearly, $(\hat{x}, \hat{y}, \hat{z})$ satisfies constraints (13b), (13c), (13f), and the 0-1 fixings required by (F_0, F_1). Now we show that constraints (13d) and (13e) hold for every vertex $i \in V$. If $i \in V \setminus (Q \cup \{v\})$, then $i \notin S_{k'}^+$ and

$$\sum_{j:i\in S_j^+} \hat{y}_j = 0 \leq 0 = \hat{x}_i = 0 \leq \sum_{j:i\in S_j^+\cup S_j} \hat{y}_j.$$

Observe that every vertex $i \in Q$ also belongs to $S_{k'}^+ \cup S_{k'}$ because otherwise

$$1 = \bar{x}_i \leq \sum_{j:i\in S_j^+\cup S_j} \bar{y}_j \leq \sum_{j=1}^n \bar{y}_j - \bar{y}_{k'} < 1.$$

So, every vertex $i \in Q \cup \{v\}$ belongs to $S_{k'}^+ \cup S_{k'}$ and thus satisfies

$$\sum_{j:i\in S_j^+} \hat{y}_j \leq 1 = \hat{x}_i = 1 = \hat{y}_{k'} \leq \sum_{j:i\in S_j^+\cup S_j} \hat{y}_j.$$

Case 4: $y_k = 1$. Let v be the (only) vertex that belongs to S_k^+. Let $\hat{y}$ be the vector whose only nonzero entry is $\hat{y}_k = 1$, $\hat{x} = x^{Q \cup \{v\}}$ be the characteristic vector of clique $Q \cup \{v\}$, and $\hat{z} = \hat{x}(V)$ be its cardinality. Note that $Q \cup \{v\}$ is a clique as $0 < \bar{y}_k \leq \bar{x}_v$. Clearly, $(\hat{x}, \hat{y}, \hat{z})$ satisfies constraints (13b), (13c), (13f), and the 0-1 fixings required by (F_0, F_1). Now we show constraints (13d) and (13e) hold for every $i \in V$. If $i \in V \setminus (Q \cup \{v\})$, then $i \notin S_k^+$ and

$$\sum_{j : i \in S_j^+} \hat{y}_j = 0 \leq 0 = \hat{x}_i = 0 \leq \sum_{j : i \in S_j^+ \cup S_j} \hat{y}_j.$$

As in Case 3, every $i \in Q \cup \{v\}$ belongs to $S_k^+ \cup S_k$. So, if $i \in Q \cup \{v\}$, then

$$\sum_{j : i \in S_j^+} \hat{y}_j \leq 1 = \hat{x}_i = 1 \leq 1 = \sum_{j : i \in S_j^+ \cup S_j} \hat{y}_j.$$

Now we show $t_2 \leq 1 + 3t_3$. Let $r = (\emptyset, \emptyset)$ denote the root node. We create a function $f : N_2 \setminus \{r\} \to N_3$ that maps non-root nodes from N_2 to nodes from N_3. Let v be a node of N_2 that is not the root and thus has a parent p. In the first case, where the algorithm branched on an x variable at p, node p belongs to N_3; let $f(v) = p$. In the other case, where the algorithm branched on a variable y_j at p, v's sibling s belongs to N_3, since all nodes are LP feasible and its sibling has $y_j = 1$; let $f(v) = s$. In this way, a node w from N_3 has preimage $f^{-1}(w)$ consisting of at most three nodes from N_2 (its sibling and its two children), so $|f^{-1}(w)| \leq 3$. Thus, we have

$$t_2 = |N_2| \leq 1 + |N_2 \setminus \{r\}| \leq 1 + \sum_{w \in N_3} |f^{-1}(w)| \leq 1 + 3|N_3| = 1 + 3t_3.$$

□

Theorem 3 *When branch-and-bound is applied to the sparse-Jeroslow model, it visits $O(2^d n)$ nodes if the y variables are prioritized for branching.*

Proof We bound the total number of branch-and-bound nodes as follows:

$$t_1 + t_2 + t_3 \leq 1 + 4t_3 \leq 1 + 4 \left(1 + \sum_{j=1}^{n} |V(\mathcal{T}_j)| \right) = O(2^d n). \tag{17}$$

The first inequality holds by Lemma 3. The second inequality holds as follows. Consider an arbitrary branch-and-bound node. If its LP solution $(\bar{x}, \bar{y}, \bar{z})$ has integer $\bar{y}$, then either $\bar{y} = 0$ or $\bar{y} = e_j$ for some j. At most one node can have $\bar{y} = 0$ because $\bar{y} = 0$ implies $\bar{x} = 0$ and $\bar{z} = 0$, and no two nodes can have the same LP solution. In the latter case, where $\bar{y} = e_j$ for some j, the node belongs to the shrunk tree $\mathcal{T}_j$. So, to prove (17), it remains to show that each shrunk tree has $O(2^d)$ nodes.

At each branch-and-bound node (F_0, F_1) of the shrunk tree $\mathcal{T}_j$, the number of "free" x variables (from subproblem j) is

$$k = |S_j| - |F_0 \cap X_j| - |F_1 \cap X_j| - |I_0(F_0, F_1) \cap X_j|,$$

where $X_j = \{x_i \mid i \in S_j\}$ and $I_0(F_0, F_1)$ is defined as (8). If $T(k)$ denotes the number of its descendants in $\mathcal{T}_j$, we have the recurrence

$$T(k) \le \begin{cases} 1 + 2T(k-1) & \text{if } k \ge 2 \\ 1 & \text{if } k \le 1. \end{cases}$$

The reason is as follows. Suppose node (F_0, F_1) from $\mathcal{T}_j$ was not pruned, then it had an LP solution $(\bar{x}, \bar{y}, \bar{z})$ with integer $\bar{y}$, and the algorithm chose to branch on some free x variable, say x_s, that was fractional, i.e., $0 < \bar{x}_s < 1$. So, any left child of (F_0, F_1) in $\mathcal{T}_j$, if one exists, has at least one fewer free variable (x_s). Similarly, any right child of (F_0, F_1) in $\mathcal{T}_j$, if one exists, has at least one fewer free variable (x_s). So, solving the recurrence gives $T(k) = O(2^k)$, and the root of the shrunk tree has $k \le |S_j| \le d$, giving $|V(\mathcal{T}_j)| = O(2^d)$, as desired. □

Proposition 7 *If the clique-core gap is zero, then when branch-and-bound is applied to the sparse-Jeroslow model it visits at most* $2n$ *nodes if a best-bound node selection strategy is used and the y variables are prioritized for branching.*

Proof When the clique-core gap g is zero, there is a MIP solution with objective $d+1$, and this is LP optimal by Proposition 3. Suppose the root LP solution (x^*, y^*, z^*) has binary y^*, say with $y_k^* = 1$. Then, x^* is binary as well; in fact, it must be the characteristic vector of $S_k^+ \cup S_k$ as

$$d + 1 = \sum_{i \in V} x_i^* \le \sum_{i \in S_k^+ \cup S_k} x_i^* + 0 \le d + 1,$$

where the middle inequality holds by (13e). So, the node is pruned by feasibility. Otherwise, y^* is fractional, and the algorithm branches on some y_j. In the branch where $y_j = 1$, the LP solution either has objective $d+1$ in which case it again has binary x and is pruned by feasibility, or it has objective less than $d+1$ in which case it produces no children by the best-bound strategy. □

5.3 Analysis for conflict-Balas

We have seen that the sparse-Jeroslow model (13) visits $O(2^d n)$ nodes, provided that y variables are prioritized for branching. For the conflict-Balas model (16), we are able to prove a better bound, $O(\varphi^d n)$, which is achieved. That is, there are classes of graphs for which $\Omega(\varphi^d n)$ nodes are visited. They are obtained by modifying the lollipop graphs.

Specifically, we define the *dessert* graph $D_{p,q} = L_{3,p} \cup K_{q+3}$ as the disjoint union of the lollipop $L_{3,p}$ and the complete "cotton candy" graph K_{q+3}. As shown in Figure 4, it has $p + q + 6$ vertices and is defined for $p \ge 1$ and $q \ge 1$.

Lemma 4 *The dessert complement* $\overline{D}_{p,q}$ *admits the degeneracy ordering*

$$(1', 2', \dots, q')^\frown(0, 0')^\frown(2, 4, \dots, p_e)^\frown(-1', -1, -2', -2)^\frown(1, 3, \dots, p_o),$$

where p_e *and* p_o *are the largest even and odd integers, respectively, that do not exceed* p. *Its degeneracy* $d(\overline{D}_{p,q})$ *is* $p + 3$.

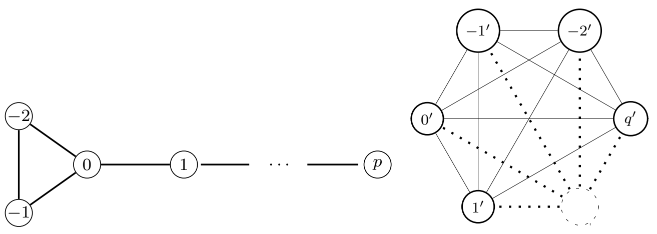


Fig. 4: The dessert graph $D_{p,q} := L_{3,p} \cup K_{q+3}$.

Proof This is a maximum degree ordering of $D_{p,q}$, thus a degeneracy ordering of $\overline{D}_{p,q}$. Vertex $1'$ has the largest right-neighborhood, with $p+3$ vertices. □

Without loss of generality, suppose that each subproblem j of $\overline{D}_{p,q}$ has $S_j^+ = \{j\}$ and is thus represented by the variable y_j. Now, for each subproblem defined by vertex $u \in \{1', 2', \dots, q'\}$, we construct a point $(\bar{w}, \bar{x}, \bar{y})$ given by

$$\bar{y}_v = \begin{cases} 1 \text{ if } v = u \\ 0 \text{ if } v \neq u; \end{cases} \qquad \bar{w}_i^v = \begin{cases} \frac{1}{2} \text{ if } v = u,\ i \in \{-2,-1,0,1,2,\dots,p\} \\ 1 \text{ if } v = u \text{ and } i = u \\ 0 \text{ if otherwise.} \end{cases}$$

and let each vertex i have $\bar{x}_i = \sum_{j:\ i \in S_j^+ \cup S_j} \bar{w}_i^j$. We argue that all such points $(\bar{w}, \bar{x}, \bar{y})$ constructed in this way are LP optimal.

Lemma 5 *Assume the degeneracy ordering given in Lemma 4. For each vertex $u \in \{1', 2', \dots, q'\}$, the point $(\bar{w}, \bar{x}, \bar{y})$ defined above is an optimal extreme point of the LP relaxation of the conflict-Balas model* (16).

Proof Observe that $(\bar{w}, \bar{x}, \bar{y})$ is LP feasible with objective value $(p+5)/2$. First, we show LP optimality. By Proposition 1, it suffices to show that the objective value over each conflict model subproblem P^j (defined in Section 4.2) is at most $(p+5)/2$. When $j \in \{-2', -1', 0', 1', 2', \dots, q'\}$, the conflict model subproblem is over a universal vertex j (contributing one to the objective) and a subgraph of the co-lollipop $\overline{L}_{3,p}$ (contributing at most $(p+3)/2$ as in the proof of Theorem 1), giving LP objective at most $(p+5)/2$. When $j = 0$, the conflict model subproblem is over the universal vertex 0, the co-path $(2, 3, \dots, p)$, and the co-triangle on $\{-2', -1', 0'\}$, giving LP objective $1 + \lfloor p/2 \rfloor + (3/2) \leq (p+5)/2$. Finally, for any other $j \in \{-2, -1\} \cup \{1, 2, \dots, p\}$, the conflict model subproblem is over a subset of the vertices $\{-2, -1\} \cup \{1, 2, \dots, p\} \cup \{-2', -1'\}$, giving an LP objective of at most $1 + \lceil p/2 \rceil + 1 \leq (p+5)/2$.

Now, to show $(\bar{w}, \bar{x}, \bar{y})$ is extreme, see that it is the unique point satisfying the following $3n + m$ constraints from (16) at equality, where n and m refer to the number of vertices and edges of $\overline{D}_{p,q}$, respectively. The number of inequalities is reported in the left-most column in parentheses.

$$\begin{aligned}
&(n-1) && y_j \geq 0 && \forall j \neq u && \text{(18a)}\\
&(1) && \sum_j y_j \leq 1 && && \text{(18b)}\\
&(m-(p+3)) && w_i^j \geq 0 && \forall i \in S_j,\ \forall j \neq u && \text{(18c)}\\
&(n-1) && w_i^j = y_j && \forall i \in S_j^+,\ \forall j \neq u && \text{(18d)}\\
&(1) && w_i^j = y_j && \forall i \in S_j^+,\ \forall j = u && \text{(18e)}\\
&(p+3) && w_r^u + w_s^u \leq y_j && \forall \{r,s\} \in \bar{E}(S_u) && \text{(18f)}\\
&(n) && \sum_{j:\ i \in S_j^+ \cup S_j} w_i^j = x_i && \forall i. && \text{(18g)}
\end{aligned}$$

For uniqueness, see that if constraints (18a) and (18b) hold at equality then y is binary with one nonzero $y_u = 1$. Constraints (18c) and (18d) then force the associated w_i^j variables to zero, and $w_i^u = 1$ by (18e). The conflict constraints (18f) associated with the triangle from the lollipop then force $w_r^u = 1/2$ for triangle nodes r; the other conflicts from the lollipop propagate the $1/2$ values down the path $(1, 2, \ldots, p)$. Constraints (18g) uniquely determine x. □

Theorem 4 *When branch-and-bound is applied to the conflict-Balas model, it can visit $\Omega(\varphi^d n)$ nodes.*

Proof We show that branch-and-bound visits $\Theta(n\varphi^d)$ nodes when applied to the conflict-Balas model, for co-dessert graphs $\overline{D}_{p,q}$ under the degeneracy ordering from Lemma 4. Namely, we exhibit a branch-and-bound tree with q' subtrees, each with $\Omega(\varphi^{d-2})$ nodes.

The degeneracy of $\overline{D}_{p,q}$ is $d = p + 3$, as shown in Lemma 4. The root LP solution can be taken as $(\bar{w}, \bar{x}, \bar{y})$ for $u = 1'$, per Lemma 5. This solution has $\bar{y}_{1'} = 1$ and $\bar{w}_p^{1'} = 1/2$. Suppose we branch on $w_p^{1'}$. In the right child, where $w_p^{1'} = 1$, constraints (16b) force $y_{1'} = 1$, so the model reduces to the conflict model for $L_{3,p-2}$; so, by Theorem 1, this subtree consisting of the right child and its descendants has $\Theta(\varphi^{p+1}) = \Theta(\varphi^{d-2})$ nodes. Meanwhile, in the left child of the root, where $w_p^{1'} = 0$, we can take the LP solution to be $(\bar{w}, \bar{x}, \bar{y})$ for $u = 2'$, which is LP optimal and extreme by Lemma 5; similar to before, this solution has $\bar{y}_{2'} = 1$ and $\bar{w}_p^{2'} = 1/2$, and branching on $w_p^{2'}$ leads to a right subtree with $\Theta(\varphi^{d-2})$ nodes. Repeating this scheme for $u = \{3', 4', \ldots, q'\}$ down the left side of the tree gives q subtrees, each with $\Theta(\varphi^{d-2})$ nodes. This shows that the whole branch-and-bound tree has $\Omega(q\varphi^{d-2})$ nodes, which is $\Omega(n\varphi^d)$ over the class of co-desserts $\{\overline{D}_{q,q} \mid q \in \mathbb{Z}_+\}$. □

For further analysis with the conflict-Balas model, we again observe that the nodes visited by branch-and-bound can be partitioned into three sets:

N_1 : nodes (F_0, F_1) for which $\text{LP}(F_0, F_1)$ is infeasible;

N_2 : nodes (F_0, F_1) whose LP solution $(\bar{w}, \bar{x}, \bar{y})$ has fractional $\bar{y} \notin \{0,1\}^n$;

N_3 : nodes (F_0, F_1) whose LP solution $(\bar{w}, \bar{x}, \bar{y})$ has integral $\bar{y} \in \{0,1\}^n$.

As before, let $t_1 = |N_1|$, $t_2 = |N_2|$, and $t_3 = |N_3|$ denote their sizes.

Lemma 6 *When the conflict-Balas model is solved with branch-and-bound, all visited nodes are LP feasible (i.e., $t_1 = 0$) and have binary $\bar{y}$ (i.e., $t_2 = 0$).*

Proof By Proposition 1, all extreme points $(\bar{w}, \bar{x}, \bar{y})$ of the conflict-Balas LP have binary $\bar{y}$. Consequently, the same holds for *faces* of the conflict-Balas LP. This includes the LPs encountered at branch-and-bound nodes, as they are induced by valid inequalities of the form $w_i^j \geq 0$ and $w_i^j \leq 1$. Thus, $t_2 = 0$.

The root node is clearly LP feasible, so consider a non-root node (F_0, F_1) encountered by the algorithm. Its parent (F_0', F_1') is LP feasible, say at $(\bar{w}, \bar{x}, \bar{y})$. As we know, $\bar{y}$ is binary, say, with $\bar{y}_j = 1$. See that $Q := \{v \in V \mid \bar{w}_v^j = 1\}$ is a clique. Additionally, $Q \cup \{v\}$ is a clique for each v with $\bar{w}_v^j > 0$, because otherwise v has a nonneighbor q in Q with $\bar{w}_q^j = 1$ giving the contradiction

$$0 < \bar{w}_v^j \leq \bar{y}_j - \bar{w}_q^j = 1 - 1 = 0,$$

where the second inequality holds by (16b). At parent node (F_0', F_1'), the algorithm branched on a variable, fixing it to zero or one. This branching variable must take the form w_v^j with $v \in S_j$, as x is defined continuous, $\bar{y}$ is binary, and $\bar{w}_i^j = \bar{y}_j$ for $i \in S_j^+$ by (16d). This gives two cases for the child; in both we construct a feasible point $(\hat{w}, \hat{x}, \hat{y})$ for $\text{LP}(F_0, F_1)$. In the first case, w_v^j was fixed to zero, i.e., $(F_0, F_1) = (F_0' \cup \{w_v^j\}, F_1')$. Let $\hat{y} = \bar{y} \in \{0,1\}^n$ and let $\hat{w}^j$ represent clique Q. Set other $\hat{w}$'s to zero and $\hat{x}_i := \sum_{j:\ i \in S_j^+ \cup S_j} \hat{w}_i^j$ for all $i \in [n]$. Clearly, $(\hat{w}, \hat{x}, \hat{y})$ satisfies all constraints in model (16) and the 0-1 fixings required by (F_0, F_1). In the other case, w_v^j was fixed to one, i.e., $(F_0, F_1) = (F_0', F_1' \cup \{w_v^j\})$. The construction follows the previous case, but using $Q \cup \{v\}$ instead of Q. □

Theorem 5 *When branch-and-bound is applied to the conflict-Balas model, it visits $O(\varphi^d n)$ nodes.*

Proof We bound the total number of branch-and-bound nodes as follows:

$$t_1 + t_2 + t_3 = t_3 \leq 1 + \sum_{j=1}^{n} |V(\mathcal{T}_j)| = O(\varphi^d n). \tag{19}$$

The first equality holds by Lemma 6. The second inequality holds because if the node's LP solution $(\bar{w}, \bar{x}, \bar{y})$ has integer $\bar{y}$, then either $\bar{y} = 0$ or $\bar{y} = e_j$ for some j. At most one node can have $\bar{y} = 0$ because $\bar{y} = 0$ implies $\bar{w} = 0$ and $\bar{x} = 0$, and no two nodes can have the same LP solution. In the latter case, where $\bar{y} = e_j$ for some j, the node belongs to the shrunk tree $\mathcal{T}_j$. So, to prove (19), it remains to show that each shrunk tree has $O(\varphi^d)$ nodes.

At each node (F_0, F_1) of the shrunk tree $\mathcal{T}_j$, the number of "free" w variables from subproblem j is

$$k = |S_j| - |F_0 \cap W_j| - |F_1 \cap W_j| - |I_0^j(F_0, F_1) \cap W_j|, \tag{20}$$

where $W_j := \{w_i^j \mid i \in S_j\}$ is the set of w variables from subproblem j, and $I_0^j(F_0, F_1)$ is the set of w variables from subproblem j that are *implicitly* fixed to zero by the extended conflict constraints (16b):

$$I_0^j(F_0, F_1) := \left\{ w_v^j \in W_j \setminus F_0 \;\middle|\; v \text{ has a nonneighbor } u \text{ with } w_u^j \in F_1 \right\}. \quad (21)$$

Now, if $T(k)$ is the number of its descendants in $\mathcal{T}_j$, we have the recurrence

$$T(k) \leq \begin{cases} 1 + T(k-2) + T(k-1) & \text{if } k \geq 2 \\ 1 & \text{if } k \leq 1. \end{cases}$$

The reason is as follows. Suppose (F_0, F_1) from $\mathcal{T}_j$ were not pruned, then it had an LP solution $(\bar{w}, \bar{x}, \bar{y})$ with integer $\bar{y}$, and the algorithm chose to branch on some free w variable, say w_s^j, that was fractional, i.e., $0 < \bar{w}_s^j < 1$. See that s has a nonneighbor t from S_j with w_t^j free since otherwise $\bar{w}_s^j$ can be increased to one and still be LP feasible. So, any left child of (F_0, F_1) in $\mathcal{T}_j$, if one exists, has one fewer free variable (w_s^j added to F_0). Meanwhile, any right child of (F_0, F_1) in $\mathcal{T}_j$, if one exists, has at least two fewer free variables (w_s^j added to F_1, and w_t^j added to I_0^j), so $T(k) \leq 1 + T(k-1) + T(k-2)$. So, solving the recurrence gives $T(k) = O(\varphi^k)$, and the root of the shrunk tree has $k \leq |S_j| \leq d$, giving $|V(\mathcal{T}_j)| = O(\varphi^d)$, as desired. □

Recall that the number of branch-and-bound nodes visited by the sparse-Jeroslow model was $O(2^d n)$ back in Theorem 3, but for the conflict-Balas model we have a stronger bound of $O(\varphi^d n)$. The reason for the different base of the exponent is that when a variable w_s^j is branched on in the conflict-Balas model, s has a nonneighbor t that also belongs to S_j; in the branch where $w_s^j = 1$, the conflict constraint will force $w_t^j = 0$. Meanwhile, if a variable x_s from the sparse-Jeroslow model is branched upon, there exists a nonneighbor of s that is "free" but it may lie outside the particular subproblem S_j.

Theorem 6 *When branch-and-bound is applied to the conflict-Balas model, it visits $\mathcal{O}(2^g n)$ nodes if a best-bound node selection strategy is used.*

Proof By the proof of Theorem 5 it suffices to show that $|V(\mathcal{T}_j)| = O(2^g)$. First, recall the notations W_j and $I_0^j(F_0, F_1)$ used for Theorem 5. At each node (F_0, F_1) of the shrunk tree, the number of w variables from subproblem j that are explicitly or implicitly fixed to zero is $|F_0 \cap W_j| + |I_0^j(F_0, F_1) \cap W_j|$. So, the LP at node (F_0, F_1) exceeds ("beats") the IP objective ω by at most

$$b = (|S_j| + 1) - |F_0 \cap W_j| - |I_0^j(F_0, F_1) \cap W_j| - \omega.$$

Crucially, if $b < 0$, then (F_0, F_1) produces no children by the best-bound node selection strategy. Now, if $T(b)$ is the number of its descendants in $\mathcal{T}_j$, we have

$$T(b) \leq \begin{cases} 1 + T(b-1) + T(b-1) & \text{if } b \geq 0 \\ 1 & \text{if } b < 0. \end{cases}$$

The reason is as follows. Suppose (F_0, F_1) from $\mathcal{T}_j$ were not pruned, then it had an LP solution $(\bar{w}, \bar{x}, \bar{y})$ with integer $\bar{y}$, and the algorithm branched on some free w variable, say w_s^j, that was fractional. As in Theorem 5's proof, s must have a nonneighbor t from S_j with w_t^j free. So, any left child of (F_0, F_1) in $\mathcal{T}_j$, if one exists, has another variable explicitly fixed to zero (w_s^j added to F_0). Meanwhile, any right child of (F_0, F_1) in $\mathcal{T}_j$, if one exists, has another variable implicitly fixed to zero (w_t^j added to I_0^j), so $T(b) \leq 1 + T(b-1) + T(b-1)$. Solving the recurrence gives $T(b) = O(2^b)$, and the root of the shrunk tree has $b \leq (|S_j|+1) - \omega \leq (d+1) - \omega = g$, giving $|V(\mathcal{T}_j)| = O(2^g)$, as desired. □

6 Experiments

To better understand the practical performance of the clique MIPs, we conduct some experiments. Our aim is shed light on two broad questions:

1. How does the practical performance compare to the worst-case analysis? How well do the parameterizations based on d and g match the actual number of branch-and-bound nodes visited?
2. In light of size/strength tradeoffs, which clique MIPs perform best in practice? Do the new clique MIPs outperform existing MIPs on the sparse, real-life instances that motivated them? What size graphs are within reach?

For our experiments, we consider instances from the 10th DIMACS Implementation Challenge on Graph Partitioning and Graph Clustering [28], as well as from the Stanford Large Network Dataset Collection [48, SNAP]. These repositories include real-life graphs from various applications (e.g., social networks, citation networks, web graphs, road networks, biological networks), and are used for benchmarking graph algorithms, including maximum clique algorithms [29, 69, 71]. Like [44] and [33], we select a subset of instances that are not too easy, not too hard, and come from diverse applications. We select instances with diverse values of degeneracy d and clique-core gap g to illustrate their relationship with MIP solve time. For comparison purposes, we also consider three hamming graphs from the 2nd DIMACS Implementation Challenge relating to coding theory. These graphs are dense and structurally different.

Our computer is a Dell Precision Tower 7000 Series (7810) machine running Windows 10 enterprise, x64, with Intel Xeon Processor E52630 v4 and 32 GB memory. The MIPs are implemented in Python 3.9.0 for Gurobi 9.1.1 [36]. To mimic the simplicity of Algorithm 1 (branch and bound), we turn off presolve, heuristics, and cuts, yielding a *bare-bones* Gurobi MIP solver. For the sparse-Jeroslow model, y variables are prioritized for branching. We impose a time limit of 3600 seconds. The code, data, and results are available at `https://github.com/MohNaderi/Worst-case-analysis-of-clique-MIPs`.

In Table 2, we report results obtained from applying *bare-bones* Gurobi to the conflict (c), sparse (s), conflict-Balas (cB), and sparse-Jeroslow (sJ) models. The first several columns report the graph name, the number of vertices and edges (n and m), the clique number (ω), the degeneracy (d), and the clique-core gap (g). The graphs vary in size from small (e.g., polbooks with 105 vertices) to large (e.g., belgium with over a million vertices). As [71] observe, the clique-core gaps of the DIMACS10 and SNAP graphs are often very small–much smaller than that of the hamming graphs from DIMACS2.

The next two portions of the table report the number of branch-and-bound nodes and the time required to solve the MIPs. For the conflict, sparse, and sparse-Jeroslow models, we can easily calculate the number of nonzeros before building them. If the number of nonzeros exceeds 10,000,000, the instance is skipped to avoid a memory crash, and we report DNA (did not attempt). If the MIP is not built within 3600 seconds, we report DNFB (did not finish building). If the MIP was built but not solved within the 3600-second time-limit, we report the time as TLR (time limit reached). If the instance was attempted but not solved, we report the number of nodes so far ($\geq$ bbnodes).

First see that the conflict model is typically too large to be built, as expected. There are also several cases where the sparse and sparse-Jeroslow models cannot be built. Perhaps surprisingly, the conflict-Balas model can be built for all DIMACS10 and SNAP instances, despite providing a worst-case size bound $O(nd^2)$ that is seemingly no better than that of the sparse and spare-Jeroslow models, $O(n+m)$. An explanation is that the subgraphs $G[S_j]$ are typically dense and have few extended conflict constraints (16b) compared to the worst-case bound $\binom{d}{2}$, and other constraints contribute $O(n+m)$ nonzeros.

To our astonishment, the conflict-Balas model solves all but a few of the DIMACS10 and SNAP instances *at the root node*! While we expected that some of the instances with $g = 0$ would solve at the root given that they guarantee an integrality gap of zero, by $z_{LP} - z_{IP} \leq (d+1) - \omega = g = 0$, we did not expect this phenomenon to be so widespread. In fact, even the instance Cit-HepTh is solved at the root, despite having a clique-core gap of 15. At first, we suspected this to be an error, but a deeper analysis confirms the result; the root LP solution is indeed integral. Figure 5 shows the (complement) subgraph $\overline{G}[S_j]$ associated with the subproblem j that the LP selects. The top row of vertices, which form a clique in $G[S_j]$, are selected by the LP, and the bottom vertices are not. Depicted in the figure are vertical edges that form a matching between the *head* (bottom row) and the *crown* (top row). The conflict constraints associated with this matching show that the crown is optimal for the subproblem's LP. For more, we refer the reader to the influential paper [62] and the textbook [26] for more on crown decomposition.

Inspecting the MIP solve times, we see that the conflict-Balas model is the clear winner on the DIMACS10 and SNAP instances. It is followed by the sparse-Jeroslow model in (a distant) second place, the sparse model in third, and the conflict model in (a distant) fourth place. Meanwhile, on the DIMACS2 instances, the situation is much different, almost the reverse, with the conflict model in first place, and the conflict-Balas model in last place. To explain this behavior, recall that the clique MIPs developed in this paper are designed to exploit sparsity, but the hamming graphs are quite dense. They have lost their size advantage and with little gain in terms of strength. Figure 6 illustrates the models' constraint matrices when applied to email and hamming8-4.

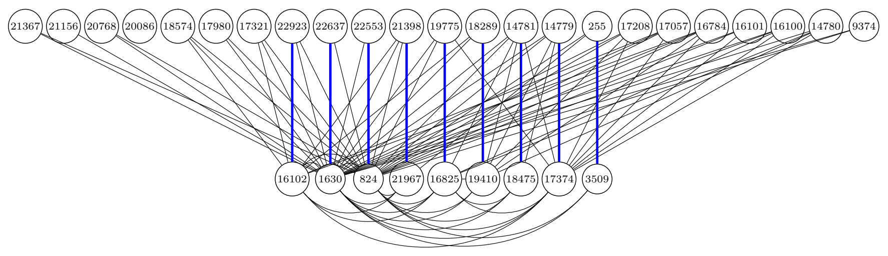


Fig. 5: The LP relaxation of the conflict-Balas model on `Cit-HepTh` gives an integral solution even though $g = 15$. Depicted is the complement subgraph $\overline{G}[S_j]$ for the chosen subproblem j. Only the top row of vertices were selected.

Finally, we observe that even though the new clique MIPs outperform previously existing MIPs, they cannot compete with combinatorial algorithms for the maximum clique problem like those of [66], [69], and [71]. For example, Walteros and Buchanan [71] report solving the instance `web-NotreDame` in 0.07 seconds. Meanwhile, the conflict-Balas model takes 201.28 seconds to solve, and the conflict, sparse, and sparse-Jeroslow models cannot even be built in the one-hour time limit. The MIP overhead is too costly.

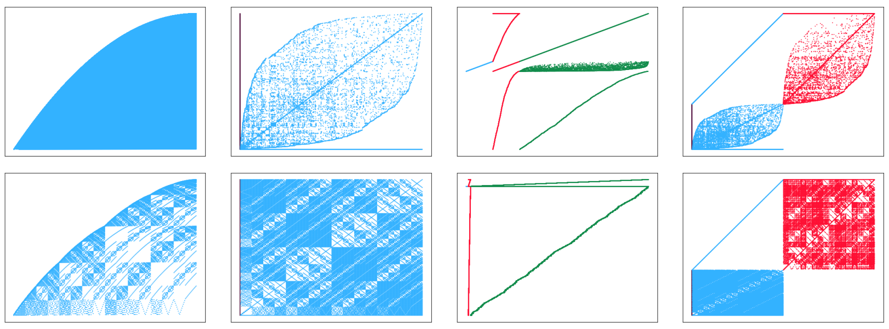


Fig. 6: Constraint matrices for the conflict, sparse, conflict-Balas, and sparse-Jeroslow models on the graphs `email` (top row) and `hamming8-4` (bottom).

Table 2: MIP results when applying *bare-bones* Gurobi to the conflict (c), sparse (s), conflict-Balas (cB), and sparse-Jeroslow (sJ) models. If the number of nonzeros is known to exceed 10,000,000, then the instance is skipped, and we report DNA (did not attempt). If the MIP is not built within 3600 seconds, then we report DNFB (did not finish building). If the MIP was built but not solved, we report the time as TLR (time limit reached). If the instance was attempted but not solved within 3600 seconds, then we report the number of branch-and-bound nodes visited so far ($\geq$ `bbnodes`).

						Number of B&B nodes				MIP solve time			
Graph	n	m	ω	d	g	c	s	cB	sJ	c	s	cB	sJ
polbooks	105	441	6	6	1	184	257	0	16	1.15	0.12	0.07	13.99
adjnoun	112	425	5	6	2	203	531	0	87	1.26	0.18	0.06	0.55
celegans_metabolic	453	2,025	9	10	2	868	1,306	0	39	60.16	0.69	0.11	0.64
delaunay_n10	1,024	3,056	4	4	1	DNFB	2,490	0	936	DNFB	6.34	0.14	12.35
email	1,133	5,451	12	11	0	DNFB	1,965	0	0	DNFB	4.16	0.33	0.91
polblogs	1,490	16,715	20	36	17	DNFB	288,255	3	7,867	DNFB	209.73	24.80	42.70
delaunay_n11	2,048	6,127	4	4	1	DNFB	5,201	0	2,135	DNFB	71.18	0.24	120.75
power	4,941	6,594	6	5	0	DNA	695	0	0	DNA	21.18	0.25	7.32
PGPgiantcompo	10,680	24,316	25	31	7	DNA	4,913	0	692	DNA	177.93	1.36	84.60
astro-ph	16,706	121,251	57	56	0	DNA	1,592	0	0	DNA	339.94	21.39	272.29
cond-mat	16,726	47,594	18	17	0	DNA	1,969	0	0	DNA	339.31	1.82	88.43
as-22july06	22,963	48,436	17	25	9	DNA	7,266	0	1,747	DNA	856.66	2.71	308.61
Cit-HepTh	27,770	352,285	23	37	15	DNA	$\geq$3,557	0	$\geq$0	DNA	TLR	317.14	TLR
cond-mat-2003	31,163	120,029	25	24	0	DNA	5,762	0	0	DNA	1,021.08	10.93	288.41
Cit-HepPh	34,546	420,877	19	30	12	DNA	$\geq$3,507	3	$\geq$0	DNA	TLR	653.71	TLR
cond-mat-2005	40,421	175,691	30	29	0	DNA	9,553	0	0	DNA	2,974.00	29.33	683.29
preferentialAttachment	100,000	499,985	6	5	0	DNA	$\geq$0	0	DNFB	DNA	TLR	844.32	DNFB
smallworld	100,000	499,998	6	7	2	DNA	$\geq$0	0	DNFB	DNA	TLR	2,254.89	DNFB
G_n_pin_pout	100,000	501,198	4	7	4	DNA	$\geq$0	$\geq$0	DNFB	DNA	TLR	TLR	DNFB
luxembourg	114,599	119,666	3	2	0	DNA	$\geq$2	0	0	DNA	TLR	3.64	1,501.11
cnr-2000	325,557	2,738,969	84	83	0	DNA	DNFB	0	DNFB	DNA	DNFB	2,505.75	DNFB
web-NotreDame	325,729	1,090,108	155	155	1	DNA	DNFB	0	DNFB	DNA	DNFB	201.28	DNFB
belgium	1,441,295	1,549,970	3	3	1	DNA	DNFB	0	DNFB	DNA	DNFB	56.63	DNFB
hamming8-2	256	31,616	128	247	120	0	0	0	0	0.98	0.24	3.65	0.49
hamming8-4	256	20,864	16	163	148	3,468,318	6,410,637	$\geq$3,704	5,993,645	1,184.02	2,496.82	TLR	2,233.63
hamming10-2	1,024	518,656	512	1,013	502	0	0	DNFB	0	0.44	5.00	DNFB	6.92

7 Conclusion and Future Research

In this paper, we observe that the usual *conflict*-based IP formulation for the maximum clique problem has several undesirable properties. Specifically, when it is applied to sparse, real-life graphs, it has a weak LP relaxation, it involves a large number of constraints, and it can take $\Omega(\varphi^n)$ branch-and-bound nodes to solve, where $\varphi \approx 1.618$ denotes the golden ratio. Even on easy instances with zero clique-core gap $g := (d+1) - \omega$, it can visit $\Omega(1.2599^n)$ nodes, regardless of the node-selection strategy. An alternative *sparse* formulation, which aggregates the conflict constraints into big-M constraints, is shown to be smaller, $\Theta(n+m)$, but still lacks other desirable worst-case properties.

In response, we conceive of four new MIPs for the clique problem that take inspiration from the literature on disjunctive extended formulations and clique finding. Of these four MIPs, two are detailed and analyzed. The *sparse-Jeroslow* model, which is the smallest and weakest of the four, requires $\Theta(n+m)$ nonzeros. It achieves an LP bound of at most $d+1$, implying that its (absolute) integrality gap is at most the clique-core gap. With a simple branch-and-bound algorithm, it is solved in $O(2^d n)$ nodes. Further, when the clique-core gap is zero, it is solved in at most $2n$ nodes. The *conflict-Balas* model is the strongest and (possibly) largest of the four new MIPs. We prove that the simplest of branch-and-bound algorithms solves it in $O(\varphi^d n)$ nodes, and also exhibit a class of instances for which $\Omega(\varphi^d n)$ nodes are visited. Moreover, when it is solved with a best-bound node selection strategy, we show that it visits $O(2^g n)$ nodes. Walteros and Buchanan [71] observed that on sparse, real-life graphs, the clique-core gap g can often be treated as a (small) constant—over half of their instances have $g \in \{0,1,2\}$. In this case, the conflict-Balas model visits just $O(n)$ nodes. Experiments on real-life instances show that the conflict-Balas model indeed visits very few nodes in practice; in fact, the LP solution is integral in most cases.

Nevertheless, the clique MIPs proposed here cannot compete with purely combinatorial methods, largely due to the overhead required for the LP relaxations. An alternative approach to ours would be to construct and solve n different MIPs, one for each of the (small) subproblems identified in Lemma 1, and report the best solution found. By Proposition 2, this would visit a combined $O(\varphi^d n)$ branch-and-bound nodes if using the conflict, dense, or sparse models. One practical disadvantage to such an approach is that it is not known up front which subproblem will contain the best solution. In an unlucky scenario, one may spend considerable effort on the first subproblem, only to find out the princess is in another castle; meanwhile, the princess is waiting unguarded in the n-th castle to be visited. To avoid such a scenario, one may want to go back-and-forth between the subproblems, working a little on whichever one is currently most promising. This is the idea behind the best-bound node selection strategy that leads to a $O(2^g n)$ bound for the conflict-Balas model. Such a back-and-forth strategy would be nontrivial to implement if using the MIP solver as a black box. However, we are able to mimic this with the conflict-Balas model in one black-box call to the MIP solver.

The new clique MIPs may also be practically relevant for variants and extensions that are not pure maximum clique problems, but still have clique-like sub-structures. In analogous examples, the shortest path problem and minimum spanning tree problem can be solved very quickly with specialized combinatorial algorithms, but the ability to express them via an integer program is nevertheless useful. When other constraints are added to them, a general-purpose MIP solver can still be be used without much trouble.

In future work, it would be interesting to see whether known classes of valid inequalities that are useful in practice for the clique problem lead to nontrivial bounds on the number of branch-and-bound nodes. Additionally, what guarantees, if any, come from a branch-and-bound algorithm where the Lovász theta semidefinite program is used as the bounding mechanism instead of an LP relaxation? This would be particularly interesting, as this relaxation "includes" all maximal independent set inequalities [43, 51]. Similar questions would be interesting for other problems besides clique. For a given problem, what is the best-performing formulation in practice? Does it come with worst-case branch-and-bound guarantees? If the associated worst-case guarantees are inadequate, can an alternative formulation be developed that performs better in theory? Is it more practical than existing MIPs?

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant Nos. 1662757 and 1942065, and by the Office of Naval Research under contract No. N00014-20-1-2242. We thank Hamidreza Validi for helpful comments.

References

1. Tobias Achterberg, Robert E Bixby, Zonghao Gu, Edward Rothberg, and Dieter Weninger. Presolve reductions in mixed integer programming. *INFORMS Journal on Computing*, 32(2):473–506, 2020.
2. Bengt Aspvall, Michael F Plass, and Robert Endre Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979.
3. Alper Atamtürk, George L Nemhauser, and Martin WP Savelsbergh. Conflict graphs in solving integer programming problems. *European Journal of Operational Research*, 121(1):40–55, 2000.
4. Egon Balas. Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. *SIAM Journal on Algebraic Discrete Methods*, 6(3):466–486, 1985.
5. Egon Balas. On the convex hull of the union of certain polyhedra. *Operations Research Letters*, 7(6):279–283, 1988.
6. Egon Balas. Disjunctive programming: Properties of the convex hull of feasible points. *Discrete Applied Mathematics*, 89(1-3):3–44, 1998.
7. Balabhaskar Balasundaram, Sergiy Butenko, and Illya V Hicks. Clique relaxations in social network analysis: The maximum k-plex problem. *Operations Research*, 59(1):133–142, 2011.

8. Amitabh Basu, Michele Conforti, Marco Di Summa, and Hongyi Jiang. Complexity of cutting planes and branch-and-bound in mixed-integer optimization. *arXiv preprint arXiv:2003.05023*, 2020.
9. V. Batagelj and M. Zaversnik. An $O(m)$ algorithm for cores decomposition of networks. *arXiv preprint arXiv:cs/0310049v1*, 2003.
10. Daniel Bienstock and Nuri Ozbay. Tree-width and the Sherali–Adams operator. *Discrete Optimization*, 1(1):13–21, 2004.
11. Charles Blair. Representation for multiple right-hand sides. *Mathematical Programming*, 49(1-3):1–5, 1990.
12. Hans L Bodlaender, T Wolle, and Arie MCA Koster. Contraction and treewidth lower bounds. *Journal of Graph Algorithms and Applications*, 10(1):5–49, 2006.
13. Immanuel M Bomze, Marco Budinich, Panos M Pardalos, and Marcello Pelillo. The maximum clique problem. In *Handbook of Combinatorial Optimization*, pages 1–74. Springer, 1999.
14. Gábor Braun, Samuel Fiorini, Sebastian Pokutta, and David Steurer. Approximation limits of linear programs (beyond hierarchies). *Mathematics of Operations Research*, 40(3):756–772, 2015.
15. Mark Braverman and Ankur Moitra. An information complexity approach to extended formulations. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 161–170, 2013.
16. Samuel S Brito and Haroldo G Santos. Preprocessing and cutting planes with conflict graphs. *Computers & Operations Research*, 128:105176, 2021.
17. Austin Buchanan. Extended formulations for vertex cover. *Operations Research Letters*, 44(3):374–378, 2016.
18. Austin Buchanan and Sergiy Butenko. Tight extended formulations for independent set. Manuscript available on Optimization Online at `http://www.optimization-online.org/DB_FILE/2014/09/4540.pdf`, 2015.
19. Austin Buchanan, Jose L Walteros, Sergiy Butenko, and Panos M Pardalos. Solving maximum clique in sparse graphs: an $O(nm+n2^{d/4})$ algorithm for d-degenerate graphs. *Optimization Letters*, 8(5):1611–1617, 2014.
20. Jianer Chen, Iyad A Kanj, Jie Meng, Ge Xia, and Fenghui Zhang. Parameterized top-k algorithms. *Theoretical Computer Science*, 470:105–119, 2013.
21. Jianer Chen, Iyad A Kanj, and Ge Xia. Improved upper bounds for vertex cover. *Theoretical Computer Science*, 411(40-42):3736–3756, 2010.
22. Michele Conforti, Marco Di Summa, and Yuri Faenza. Balas formulation for the union of polytopes is optimal. *Mathematical Programming*, 180(1):311–326, 2020.
23. Stefano Coniglio and Stefano Gualandi. Optimizing over the closure of rank inequalities with a small right-hand side for the maximum stable set problem via bilevel programming. *INFORMS Journal on Computing*, 2021. To appear.
24. Stephen A Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, 1971.

25. Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT press, 2009.
26. Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
27. Federico Della Croce and Roberto Tadei. A multi-KP modeling for the maximum-clique problem. *European Journal of Operational Research*, 73(3):555–561, 1994.
28. DIMACS. 10th DIMACS Implementation Challenge - Graph Partitioning and Graph Clustering. `http://www.cc.gatech.edu/dimacs10/downloads.shtml`, 2020. Accessed: 2020-12-30.
29. David Eppstein, Maarten Löffler, and Darren Strash. Listing all maximal cliques in large sparse real-world graphs. *Journal of Experimental Algorithmics (JEA)*, 18:3–1, 2013.
30. Yuri Faenza, Gonzalo Muñoz, and Sebastian Pokutta. New limits of treewidth-based tractability in optimization. *Mathematical Programming*, 2020. To appear.
31. Samuel Fiorini, Serge Massar, Sebastian Pokutta, Hans Raj Tiwary, and Ronald De Wolf. Exponential lower bounds for polytopes in combinatorial optimization. *Journal of the ACM (JACM)*, 62(2):1–23, 2015.
32. Fedor V Fomin and Dieter Kratsch. *Exact Exponential Algorithms*. Springer Science & Business Media, 2010.
33. Ambros Gleixner, Gregor Hendel, Gerald Gamrath, Tobias Achterberg, Michael Bastubbe, Timo Berthold, Philipp M Christophel, Kati Jarck, Thorsten Koch, Jeff Linderoth, et al. MIPLIB 2017: Data-driven compilation of the 6th mixed-integer programming library. Optimization Online preprint available at: `http://www.optimization-online.org/DB_HTML/2019/07/7285.html`, 2019.
34. Ralph Gomory. An algorithm for the mixed integer problem. Technical Report P-1885, The RAND Corporation, Santa Monica, CA, 1960.
35. Mika Goos, Rahul Jain, and Thomas Watson. Extension complexity of independent set polytopes. *SIAM Journal on Computing*, 47(1):241–269, 2018.
36. Gurobi Optimization, Inc. Gurobi Optimizer Reference Manual, 2020.
37. Johan Håstad. Clique is hard to approximate within $O(n^{1-\varepsilon})$. *Acta Mathematica*, 182(1):105–142, 1999.
38. Robert G Jeroslow. A simplification for some disjunctive formulations. *European Journal of Operational Research*, 36(1):116–121, 1988.
39. Robert G Jeroslow and James K Lowe. Modelling with integer variables. In *Mathematical Programming at Oberwolfach II*, pages 167–184. Springer, 1984.
40. Ellis L Johnson and Manfred W Padberg. Degree-two inequalities, clique facets, and biperfect graphs. In *North-Holland Mathematics Studies*, volume 66, pages 169–187. Elsevier, 1982.
41. Richard M Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Springer, 1972.

42. James E Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.
43. Donald E Knuth. The sandwich theorem. *The Electronic Journal of Combinatorics*, 1(1):A1, 1994.
44. Thorsten Koch, Tobias Achterberg, Erling Andersen, Oliver Bastert, Timo Berthold, Robert E Bixby, Emilie Danna, Gerald Gamrath, Ambros M Gleixner, Stefan Heinz, et al. MIPLIB 2010. *Mathematical Programming Computation*, 3(2):103, 2011.
45. Petr Kolman and Martin Koutecký. Extended formulation for CSP that is compact for instances of bounded treewidth. *The Electronic Journal of Combinatorics*, 22(4):P4–30, 2015.
46. AH Land and AG Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960.
47. Monique Laurent. Sums of squares, moment matrices and optimization over polynomials. In *Emerging applications of algebraic geometry*, pages 157–270. Springer, 2009.
48. Jure Leskovec and Rok Sosič. SNAP: A general-purpose network analysis and graph-mining library. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):1–20, 2016.
49. Adam N Letchford, Fabrizio Rossi, and Stefano Smriglio. The stable set problem: Clique and nodal inequalities revisited. *Computers & Operations Research*, 123:105024, 2020.
50. Don R Lick and Arthur T White. k-degenerate graphs. *Canadian Journal of Mathematics*, 22(5):1082–1096, 1970.
51. László Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information theory*, 25(1):1–7, 1979.
52. James Kenneth Lowe. Modelling with integer variables, Ph.D. thesis. Technical report, Georgia Institute of Technology, 1984.
53. R Duncan Luce and Albert D Perry. A method of matrix analysis of group structure. *Psychometrika*, 14(2):95–116, 1949.
54. G. Manoussakis. The clique problem on inductive k-independent graphs. *arXiv preprint arXiv:1410.3302*, 2014.
55. Pedro Martins. Extended and discretized formulations for the maximum clique problem. *Computers & Operations Research*, 37(7):1348–1358, 2010.
56. D. W. Matula and L. L. Beck. Smallest-last ordering and clustering and graph coloring algorithms. *Journal of the ACM*, 30(3):417–427, 1983.
57. RE Miller and DE Muller. A problem of maximum consistent subsets. Technical Report RC-240, IBM TJ Watson Research Center, New York, 1960.
58. I Douglas Moon and Sohail S Chaudhry. An analysis of network location problems with distance constraints. *Management Science*, 30(3):290–307, 1984.
59. John W Moon and Leo Moser. On cliques in graphs. *Israel journal of Mathematics*, 3(1):23–28, 1965.

60. Alan T Murray and Richard L Church. Facets for node packing. *European Journal of Operational Research*, 101(3):598–608, 1997.
61. George L Nemhauser and Leslie Earl Trotter. Properties of vertex packing and independence system polyhedra. *Mathematical Programming*, 6(1):48–61, 1974.
62. George L Nemhauser and Leslie Earl Trotter. Vertex packings: structural properties and algorithms. *Mathematical Programming*, 8(1):232–248, 1975.
63. Manfred W Padberg. On the facial structure of set packing polyhedra. *Mathematical Programming*, 5(1):199–215, 1973.
64. Jeffrey Pattillo, Nataly Youssef, and Sergiy Butenko. On clique relaxation models in network analysis. *European Journal of Operational Research*, 226(1):9–18, 2013.
65. Steffen Rebennack, Gerhard Reinelt, and Panos M Pardalos. A tutorial on branch and cut algorithms for the maximum stable set problem. *International Transactions in Operational Research*, 19(1-2):161–199, 2012.
66. Ryan A Rossi, David F Gleich, and Assefaw H Gebremedhin. Parallel maximum clique algorithms with applications to network analysis. *SIAM Journal on Scientific Computing*, 37(5):C589–C616, 2015.
67. Martin WP Savelsbergh. Preprocessing and probing techniques for mixed integer programming problems. *ORSA Journal on Computing*, 6(4):445–454, 1994.
68. Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*. Springer Science & Business Media, Berlin, 2003.
69. Anurag Verma, Austin Buchanan, and Sergiy Butenko. Solving the maximum clique and vertex coloring problems on very large sparse networks. *INFORMS Journal on computing*, 27(1):164–177, 2015.
70. Juan Pablo Vielma. Mixed integer linear programming formulation techniques. *SIAM Review*, 57(1):3–57, 2015.
71. Jose L Walteros and Austin Buchanan. Why is maximum clique often easy in practice? *Operations Research*, 68(6):1866–1895, 2020.
72. Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43(3):441–466, 1991.
73. David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the Thirty-eighth Annual ACM Symposium on Theory of Computing*, STOC '06, pages 681–690, 2006.