An Approximate, Efficient Solver for LP Rounding

Srikrishna Sridhar[1], **Victor Bittorf**[1], **Ji Liu**[1], **Ce Zhang**[1]
Christopher Ré[2], **Stephen J. Wright**[1]
[1]Computer Sciences Department, University of Wisconsin-Madison, Madison, WI 53706
[2]Computer Science Department, Stanford University, Stanford, CA 94305
{srikris,vbittorf,ji-liu,czhang,swright}@cs.wisc.edu
chrismre@cs.stanford.edu

Abstract

Many problems in machine learning can be solved by rounding the solution of an appropriate linear program (LP). This paper shows that we can recover solutions of comparable quality by rounding an approximate LP solution instead of the exact one. These approximate LP solutions can be computed efficiently by applying a parallel stochastic-coordinate-descent method to a quadratic-penalty formulation of the LP. We derive worst-case runtime and solution quality guarantees of this scheme using novel perturbation and convergence analysis. Our experiments demonstrate that on such combinatorial problems as vertex cover, independent set and multiway-cut, our approximate rounding scheme is up to an order of magnitude faster than Cplex (a commercial LP solver) while producing solutions of similar quality.

1 Introduction

A host of machine-learning problems can be solved effectively as approximations of such NP-hard combinatorial problems as set cover, set packing, and multiway-cuts [8, 11, 16, 22]. A popular scheme for solving such problems is called LP rounding [22, chs. 12-26], which consists of the following three-step process: (1) construct an integer (binary) linear program (IP) formulation of a given problem; (2) relax the IP to an LP by replacing the constraints $x \in \{0, 1\}$ by $x \in [0, 1]$; and (3) round an optimal solution of the LP to create a feasible solution for the original IP problem. LP rounding is known to work well on a range of hard problems, and comes with theoretical guarantees for runtime and solution quality.

The Achilles' heel of LP-rounding is that it requires solutions of LPs of possibly extreme scale. Despite decades of work on LP solvers, including impressive advances during the 1990s, commercial codes such as Cplex or Gurobi may not be capable of handling problems of the required scale. In this work, we propose an approximate LP solver suitable for use in the LP-rounding approach, for very large problems. Our intuition is that in LP rounding, since we ultimately round the LP to obtain an approximate solution of the combinatorial problem, a crude solution of the LP may suffice. Hence, an approach that can find approximate solutions of large LPs quickly may be suitable, even if it is inefficient for obtaining highly accurate solutions.

This paper focuses on the theoretical and algorithmic aspects of finding approximate solutions to an LP, for use in LP-rounding schemes. Our three main technical contributions are as follows: First, we show that one can approximately solve large LPs by forming convex quadratic programming (QP) approximations, then applying stochastic coordinate descent to these approximations. Second, we derive a novel convergence analysis of our method, based on Renegar's perturbation theory for linear programming [17]. Finally, we derive bounds on runtime as well as worst-case approximation ratio of our rounding schemes. Our experiments demonstrate that our approach, called Thetis, produces solutions of comparable quality to state-of-the-art approaches on such tasks as noun-phrase chunking and entity resolution. We also demonstrate, on three different classes of combinatorial problems, that Thetis can outperform Cplex (a state-of-the-art commercial LP and IP solver) by up to an order of magnitude in runtime, while achieving comparable solution quality.

Related Work. Recently, there has been some focus on the connection between LP relaxations and maximum a posteriori (MAP) estimation problems [16, 19]. Ravikumar et. al [16] proposed rounding schemes for iterative LP solvers to facilitate MAP inference in graphical models. In contrast, we propose to use stochastic descent methods to solve a QP relaxation; this allows us to take advantage of recent results on asynchronous parallel methods of this type [12, 14]. Recently, Makari et. al [13] propose an intriguing parallel scheme for packing and covering problems. In contrast, our results apply to more general LP relaxations, including set-partitioning problems like multiway-cut. Additionally, the runtime of our algorithm is less sensitive to approximation error. For an error ε, the bound on runtime of the algorithm in [13] grows as ε^{-5}, while the bound on our algorithm's runtime grows as ε^{-2}.

2 Background: Approximating NP-hard problems with LP Rounding

In this section, we review the theory of LP-rounding based approximation schemes for NP-hard combinatorial problems. We use the vertex cover problem as an example, as it is the simplest nontrivial setting that exposes the main ideas of this approach.

Preliminaries. For a minimization problem Φ, an algorithm ALG is an *α-factor approximation for* Φ, for some $\alpha > 1$, if any solution produced by ALG has an objective value at most α times the value of an optimal (lowest cost) solution. For some problems, such as vertex cover, there is a constant-factor approximation scheme ($\alpha = 2$). For others, such as set cover, the value of α can be as large as $O(\log N)$, where N is the number of sets.

An LP-rounding based approximation scheme for the problem Φ first *constructs* an IP formulation of Φ which we denote as "P". This step is typically easy to perform, but the IP formulation P is, in theory, as hard to solve as the original problem Φ. In this work, we consider applications in which the only integer variables in the IP formulation are binary variables $x \in \{0, 1\}$. The second step in LP rounding is a *relax / solve* step: We relax the constraints in P to obtain a linear program $LP(P)$, replacing the binary variables with continuous variables in $[0, 1]$, then solve $LP(P)$. The third step is to *round* the solution of $LP(P)$ to an integer solution which is feasible for P, thus yielding a candidate solution to the original problem Φ. The focus of this paper is on the relax / solve step, which is usually the computational bottleneck in an LP-rounding based approximation scheme.

Example: An Oblivious-Rounding Scheme For Vertex Cover. Let $G(V, E)$ denote a graph with vertex set V and undirected edges $E \subseteq (V \times V)$. Let c_v denote a nonnegative cost associated with each vertex $v \in V$. A vertex cover of a graph is a subset of V such that each edge $e \in E$ is incident to at least one vertex in this set. The *minimum-cost* vertex cover is the one that minimizes the sum of terms c_v, summed over the vertices v belonging to the cover. Let us review the "construct," "relax / solve," and "round" phases of an LP-rounding based approximation scheme applied to vertex cover.

In the "construct" phase, we introduce binary variables $x_v \in \{0, 1\}, \forall v \in V$, where x_v is set to 1 if the vertex $v \in V$ is selected in the vertex cover and 0 otherwise. The IP formulation is as follows:

$$\min_x \sum_{v \in V} c_v x_v \text{ s.t. } x_u + x_v \geq 1 \text{ for } (u, v) \in E \text{ and } x_v \in \{0, 1\} \text{ for } v \in V. \quad (1)$$

Relaxation yields the following LP

$$\min_x \sum_{v \in V} c_v x_v \text{ s.t. } x_u + x_v \geq 1 \text{ for } (u, v) \in E \text{ and } x_v \in [0, 1] \text{ for } v \in V. \quad (2)$$

A feasible solution of the LP relaxation (2) is called a "fractional solution" of the original problem. In the "round" phase, we generate a valid vertex cover by simply choosing the vertices $v \in V$ whose fractional solution $x_v \geq \frac{1}{2}$. It is easy to see that the vertex cover generated by such a rounding scheme costs no more than twice the cost of the fractional solution. If the fractional solution chosen for rounding is an optimal solution of (2), then we arrive at a 2-factor approximation scheme for vertex cover. We note here an important property: The rounding algorithm can generate feasible integral solutions while being *oblivious* of whether the fractional solution is an optimal solution of (2). We formally define the notion of an oblivious rounding scheme as follows.

Definition 1. *For a minimization problem Φ with an IP formulation P whose LP relaxation is denoted by LP(P), a γ-factor 'oblivious' rounding scheme converts any feasible point $x_f \in$ LP(P) to an integral solution $x_I \in P$ with cost at most γ times the cost of LP(P) at x_f.*

Problem Family	Approximation Factor	Machine Learning Applications
Set Covering	$\log(N)$ [20]	Classification [3], Multi-object tracking [24].
Set Packing	$es + o(s)$ [1]	MAP-inference [19], Natural language [9].
Multiway-cut	$3/2 - 1/k$ [5]	Computer vision [4], Entity resolution [10].
Graphical Models	Heuristic	Semantic role labeling [18], Clustering [21].

Figure 1: LP-rounding schemes considered in this paper. The parameter N refers to the number of sets; s refers to s-column sparse matrices; and k refers to the number of terminals. e is the Euler's constant.

Given a γ-factor *oblivious algorithm* ALG to the problem Φ, one can construct a γ-factor approximation algorithm for Φ by using ALG to round an *optimal* fractional solution of LP(P). When we have an approximate solution for LP(P) that is feasible for this problem, rounding can produce an α-factor approximation algorithm for Φ for a factor α slightly larger than γ, where the difference between α and γ takes account of the inexactness in the approximate solution of LP(P). Many LP-rounding schemes (including the scheme for vertex cover discussed in Section 2) are oblivious. We implemented the oblivious LP-rounding algorithms in Figure 1 and report experimental results in Section 4.

3 Main results

In this section, we describe how we can solve LP relaxations approximately, in less time than traditional LP solvers, while still preserving the formal guarantees of rounding schemes. We first define a notion of approximate LP solution and discuss its consequences for oblivious rounding schemes. We show that one can use a regularized quadratic penalty formulation to compute these approximate LP solutions. We then describe a stochastic-coordinate-descent (SCD) algorithm for obtaining approximate solutions of this QP, and mention enhancements of this approach, specifically, asynchronous parallel implementation and the use of an augmented Lagrangian framework. Our analysis yields a worst-case complexity bound for solution quality and runtime of the entire LP-rounding scheme.

3.1 Approximating LP Solutions

Consider the LP in the following standard form

$$\min\ c^T x \ \text{ s.t. } Ax = b,\ \ x \geq 0, \tag{3}$$

where $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and $A \in \mathbb{R}^{m\times n}$ and its corresponding dual

$$\max\ b^T u \ \text{ s.t. }\ c - A^T u \geq 0. \tag{4}$$

Let x^* denote an optimal primal solution of (3). An approximate LP solution $\hat{x}$ that we use for LP-rounding may be infeasible and have objective value different from the optimum $c^T x^*$. We quantify the inexactness in an approximate LP solution as follows.

Definition 2. *A point $\hat{x}$ is an (ϵ, δ)-approximate solution of the LP (3) if $\hat{x} \geq 0$ and there exists constants $\epsilon > 0$ and $\delta > 0$ such that*

$$\|A\hat{x} - b\|_\infty \leq \epsilon \quad \text{and} \quad |c^T\hat{x} - c^T x^*| \leq \delta |c^T x^*|.$$

Using Definitions 1 and 2, it is easy to see that a γ-factor oblivious rounding scheme can round a $(0, \delta)$ approximate solution to produce a feasible integral solution whose cost is no more than $\gamma(1 + \delta)$ times the optimal solution of the P. The factor $(1 + \delta)$ arises because the rounding algorithm does not have access to an optimal fractional solution. To cope with the infeasibility, we convert an (ϵ, δ)-approximate solution to a $(0, \hat{\delta})$ approximate solution where $\hat{\delta}$ is not too large. For vertex cover (2), we prove the following result in Appendix C. (Here, $\Pi_{[0,1]^n}(\cdot)$ denotes projection onto the unit hypercube in $\mathbb{R}^n$.)

Lemma 3. *Let $\hat{x}$ be an (ε, δ) approximate solution to the linear program (2) with $\varepsilon \in [0, 1)$. Then, $\tilde{x} = \Pi_{[0,1]^n}((1-\varepsilon)^{-1}\hat{x})$ is a $(0, \delta(1-\varepsilon)^{-1})$-approximate solution.*

Since $\tilde{x}$ is a feasible solution for (2), the oblivious rounding scheme in Section 2 results in an $2(1 + \delta(1-\varepsilon)^{-1})$ factor approximation algorithm. In general, constructing $(0, \hat{\delta})$ from (ϵ, δ) approximate solutions requires reasoning about the structure of a particular LP. In Appendix C, we establish statements analogous to Lemma 3 for packing, covering and multiway-cut problems.

3.2 Quadratic Programming Approximation to the LP

We consider the following regularized quadratic penalty approximation to the LP (3), parameterized by a positive constant β, whose solution is denoted by $x(\beta)$:

$$x(\beta) := \arg\min_{x \geq 0} f_\beta(x) := c^T x - \bar{u}^T(Ax - b) + \frac{\beta}{2}\|Ax - b\|^2 + \frac{1}{2\beta}\|x - \bar{x}\|^2, \tag{5}$$

where $\bar{u} \in \mathbb{R}^m$ and $\bar{x} \in \mathbb{R}^n$ are arbitrary vectors. (In practice, $\bar{u}$ and $\bar{x}$ may be chosen as approximations to the dual and primal solutions of (3), or simply set to zero.) The quality of the approximation (5) depends on the *conditioning* of underlying linear program (3), a concept that was studied by Renegar [17]. Denoting the data for problem (3) by $d := (A, b, c)$, we consider perturbations $\Delta d := (\Delta A, \Delta b, \Delta c)$ such that the linear program defined by $d + \Delta d$ is primal infeasible. The primal condition number δ_P is the infimum of the ratios $\|\Delta d\|/\|d\|$ over all such vectors Δd. The dual condition number δ_D is defined analogously. (Clearly both δ_P and δ_D are in the range $[0, 1]$; smaller values indicate poorer conditioning.) We have the following result, which is proven in the supplementary material.

Theorem 4. *Suppose that δ_P and δ_D are both positive, and let (x^*, u^*) be any primal-dual solution pair for (3), (4). If we define $C_* := \max(\|x^* - \bar{x}\|, \|u^* - \bar{u}\|)$, then the unique solution $x(\beta)$ of (5) satisfies*

$$\|Ax(\beta) - b\| \leq (1/\beta)(1 + \sqrt{2})C_*, \quad \|x(\beta) - x^*\| \leq \sqrt{6}C_*.$$

If in addition the parameter $\beta \geq \frac{10C_*}{\|d\| \min(\delta_P, \delta_D)}$, then we have

$$|c^T x^* - c^T x(\beta)| \leq \frac{1}{\beta}\left[\frac{25C_*}{2\delta_P\delta_D} + 6C_*^2 + \sqrt{6}\|\bar{x}\|C_*\right].$$

In practice, we solve (5) approximately, using an algorithm whose complexity depends on the threshold $\bar{\epsilon}$ for which the objective is accurate to within $\bar{\epsilon}$. That is, we seek $\hat{x}$ such that

$$\beta^{-1}\|\hat{x} - x(\beta)\|^2 \leq f_\beta(\hat{x}) - f_\beta(x(\beta)) \leq \bar{\epsilon},$$

where the left-hand inequality follows from the fact that f_β is strongly convex with modulus β^{-1}. If we define

$$\bar{\epsilon} := \frac{C_{20}^2}{\beta^3}, \quad C_{20} := \frac{25C_*}{2\|d\|\delta_P\delta_D}, \tag{6}$$

then by combining some elementary inequalities with the results of Theorem 4, we obtain the bounds

$$|c^T\hat{x} - c^T x^*| \leq \frac{1}{\beta}\left[\frac{25C_*}{\delta_P\delta_D} + 6C_*^2 + \sqrt{6}\|\bar{x}\|C_*\right], \quad \|A\hat{x} - b\| \leq \frac{1}{\beta}\left[(1 + \sqrt{2})C_* + \frac{25C_*}{2\delta_P\delta_D}\right].$$

The following result is almost an immediate consequence.

Theorem 5. *Suppose that δ_P and δ_D are both positive and let (x^*, u^*) be any primal-dual optimal pair. Suppose that C_* is defined as in Theorem 4. Then for any given positive pair (ϵ, δ), we have that $\hat{x}$ satisfies the inequalities in Definition 2 provided that β satisfies the following three lower bounds:*

$$\beta \geq \frac{10C_*}{\|d\| \min(\delta_P, \delta_D)},$$
$$\beta \geq \frac{1}{\delta|c^T x^*|}\left[\frac{25C_*}{\delta_P\delta_D} + 6C_*^2 + \sqrt{6}\|\bar{x}\|C_*\right],$$
$$\beta \geq \frac{1}{\epsilon}\left[(1 + \sqrt{2})C_* + \frac{25C_*}{2\delta_P\delta_D}\right].$$

For an instance of vertex cover with n nodes and m edges, we can show that $\delta_P^{-1} = O(n^{1/2}(m + n)^{1/2})$ and $\delta_D^{-1} = O((m + n)^{1/2})$ (see Appendix D). The values $\bar{x} = \mathbf{1}$ and $\bar{u} = \vec{0}$ yield $C_* \leq \sqrt{m}$. We therefore obtain $\beta = O(m^{1/2}n^{1/2}(m + n)(\min\{\epsilon, \delta|c^T x^*|\})^{-1})$.

Algorithm 1 SCD method for (5)

Choose $x_0 \in \mathbb{R}^n$; $j \leftarrow 0$
loop
  Choose $i(j) \in \{1, 2, \ldots, n\}$ randomly with equal probability;
  Define x_{j+1} from x_j by setting $[x_{j+1}]_{i(j)} \leftarrow \max(0, [x_j]_{i(j)} - (1/L_{\max})[\nabla f_\beta(x_j)]_{i(j)})$, leaving other components unchanged;
  $j \leftarrow j + 1$;
end loop

3.3 Solving the QP Approximation: Coordinate Descent

We propose the use of a stochastic coordinate descent (SCD) algorithm [12] to solve (5). Each step of SCD chooses a component $i \in \{1, 2, \ldots, n\}$ and takes a step in the ith component of x along the partial gradient of (5) with respect to this component, projecting if necessary to retain nonnegativity. This simple procedure depends on the following constant $L_{\max}$, which bounds the diagonals of the Hessian in the objective of (5):

$$L_{\max} = \beta(\max_{i=1,2,\ldots,n} A_{:i}^T A_{:i}) + \beta^{-1}, \tag{7}$$

where $A_{:i}$ denotes the ith column of A. Algorithm 1 describes the SCD method. Convergence results for Algorithm 1 can be obtained from [12]. In this result, $\mathbb{E}(\cdot)$ denotes expectation over all the random variables $i(j)$ indicating the update indices chosen at each iteration. We need the following quantities:

$$l := \frac{1}{\beta}, \quad R := \sup_{j=1,2,\ldots n} \|x_j - x(\beta)\|_2, \tag{8}$$

where x_j denotes the jth iterate of the SCD algorithm. (Note that R bounds the maximum distance that the iterates travel from the solution $x(\beta)$ of (5).)

Theorem 6. *For Algorithm 1 we have*

$$\mathbb{E}\|x_j - x(\beta)\|^2 + \frac{2}{L_{\max}}\mathbb{E}(f_\beta(x_j) - f_\beta^*) \le \left(1 - \frac{l}{n(l + L_{\max})}\right)^j \left(R^2 + \frac{2}{L_{\max}}(f_\beta(x_0) - f_\beta^*)\right),$$

where $f_\beta^* := f_\beta(x(\beta))$. We obtain high-probability convergence of $f_\beta(x_j)$ to f_β^* in the following sense: For any $\eta \in (0,1)$ and any small $\bar{\epsilon}$, we have $P(f_\beta(x_j) - f_\beta^* < \bar{\epsilon}) \ge 1 - \eta$, provided that

$$j \ge \frac{n(l + L_{\max})}{l} \left| \log \frac{L_{\max}}{2\eta\bar{\epsilon}} \left(R^2 + \frac{2}{L_{\max}}(f_\beta(x_0) - f_\beta^*)\right)\right|.$$

Worst-Case Complexity Bounds. We now combine the analysis in Sections 3.2 and 3.3 to derive a worst-case complexity bound for our approximate LP solver. Supposing that the columns of A have norm $O(1)$, we have from (7) and (8) that $l = \beta^{-1}$ and $L_{\max} = O(\beta)$. Theorem 6 indicates that we require $O(n\beta^2)$ iterations to solve (5) (modulo a log term). For the values of β described in Section 3.2, this translates to a complexity estimate of $O(m^3 n^2/\epsilon^2)$.

In order to obtain the desired accuracy in terms of feasibility and function value of the LP (captured by ϵ) we need to solve the QP to within the different, tighter tolerance $\bar{\epsilon}$ introduced in (6). Both tolerances are related to the choice of penalty parameter β in the QP. Ignoring here the dependence on dimensions m and n, we note the relationships $\beta \sim \epsilon^{-1}$ (from Theorem 5) and $\bar{\epsilon} \sim \beta^{-3} \sim \epsilon^3$ (from (6)). Expressing all quantities in terms of ϵ, and using Theorem 6, we see an iteration complexity of ϵ^{-2} for SCD (ignoring log terms). The linear convergence rate of SCD is instrumental to this favorable value. By contrast, standard variants of stochastic-gradient descent (SGD) applied to the QP yield poorer complexity. For diminishing-step or constant-step variants of SGD, we see complexity of ϵ^{-7}, while for robust SGD, we see ϵ^{-10}. (Besides the inverse dependence on $\bar{\epsilon}$ or its square in the analysis of these methods, there is a contribution of order ϵ^{-2} from the conditioning of the QP.)

3.4 Enhancements

We mention two important enhancements that improve the efficiency of the approach outlined above. The first is an asynchronous parallel implementation of Algorithm 1 and the second is the use of an augmented Lagrangian framework rather than "one-shot" approximation by the QP in (5).

				Thetis				Gibbs Sampling			
Task	Formulation	PV	NNZ	**P**	**R**	**F1**	Rank	**P**	**R**	**F1**	Rank
CoNLL	Skip-chain CRF	25M	51M	.87	.90	.89	10/13	.86	.90	.88	10/13
TAC-KBP	Factor graph	62K	115K	.79	.79	.79	6/17	.80	.80	.80	6/17

Figure 2: Solution quality of our LP-rounding approach on two tasks. PV is the number of primal variables and NNZ is the number of non-zeros in the constraint matrix of the LP in standard form. The *rank* indicates where we would been have placed, had we participated in the competition.

Asynchronous Parallel SCD. An asynchronous parallel version of Algorithm 1, described in [12], is suitable for execution on multicore, shared-memory architectures. Each core, executing a single thread, has access to the complete vector x. Each thread essentially runs its own version of Algorithm 1 independently of the others, choosing and updating one component $i(j)$ of x on each iteration. Between the time a thread reads x and performs its update, x usually will have been updated by several other threads. Provided that the number of threads is not too large (according to criteria that depends on n and on the diagonal dominance properties of the Hessian matrix), and the step size is chosen appropriately, the convergence rate is similar to the serial case, and near-linear speedup is observed.

Augmented Lagrangian Framework. It is well known (see for example [2, 15]) that the quadratic-penalty approach can be extended to an augmented Lagrangian framework, in which a sequence of problems of the form (5) are solved, with the primal and dual solution estimates $\bar{x}$ and $\bar{u}$ (and possibly the penalty parameter β) updated between iterations. Such a "proximal method of multipliers" for LP was described in [23]. We omit a discussion of the convergence properties of the algorithm here, but note that the quality of solution depends on the values of $\bar{x}$, $\bar{u}$ and β at the last iteration before convergence is declared. By applying Theorem 5, we note that the constant C_* is smaller when $\bar{x}$ and $\bar{u}$ are close to the primal and dual solution sets, thus improving the approximation and reducing the need to increase β to a larger value to obtain an approximate solution of acceptable accuracy.

4 Experiments

Our experiments address two main questions: (1) Is our approximate LP-rounding scheme useful in graph analysis tasks that arise in machine learning? and (2) How does our approach compare to a state-of-the-art commercial solver? We give favorable answers to both questions.

4.1 Is Our Approximate LP-Rounding Scheme Useful in Graph Analysis Tasks?

LP formulations have been used to solve MAP inference problems on graphical models [16], but general-purpose LP solvers have rarely been used, for reasons of scalability. We demonstrate that the rounded solutions obtained using Thetis are of comparable quality to those obtained with state-of-the-art systems. We perform experiments on two different tasks: entity linking and text chunking. For each task, we produce a factor graph [9], which consists of a set of random variables and a set of factors to describe the correlation between random variables. We then run MAP inference on the factor graph using the LP formulation in [9] and compare the quality of the solutions obtained by Thetis with a Gibbs sampling-based approach [26]. We follow the LP-rounding algorithm in [16] to solve the MAP estimation problem. For entity linking, we use the TAC-KBP 2010 benchmark[1]. The input graphical model has 12K boolean random variables and 17K factors. For text chunking, we use the CoNLL 2000 shared task[2]. The factor graph contained 47K categorical random variables (with domain size 23) and 100K factors. We use the training sets provided by TAC-KBP 2010 and CoNLL 2000 respectively. We evaluate the quality of both approaches using the official evaluation scripts and evaluation data sets provided by each challenge. Figure 2 contains a description of the three relevant quality metrics, precision (P), recall (R) and F1-scores. Figure 2 demonstrates that our algorithm produces solutions of quality comparable with state-of-the-art approaches for these graph analysis tasks.

4.2 How does our proposed approach compare to a state-of-the-art commercial solver?

We conducted numerical experiments on three different combinatorial problems that commonly arise in graph analysis tasks in machine learning: vertex cover, independent set, and multiway cuts. For

[1] `http://nlp.cs.qc.cuny.edu/kbp/2010/`
[2] `http://www.cnts.ua.ac.be/conll2000/chunking/`

each problem, we compared the performance of our LP solver against the LP and IP solvers of Cplex (v12.5) (denoted as Cplex-LP and Cplex-IP respectively). The two main goals of this experiment are to: (1) compare the quality of the integral solutions obtained using LP-rounding with the integral solutions from Cplex-IP and (2) compare wall-clock times required by Thetis and Cplex-LP to solve the LPs for the purpose of LP-rounding.

Datasets. Our tasks are based on two families of graphs. The first family of instances (*frb59-26-1* to *frb59-26-5*) was obtained from Bhoslib[3] (Benchmark with Hidden Optimum Solutions); they are considered difficult problems [25]. The instances in this family are similar; the first is reported in the figures of this section, while the remainder appear in Appendix E. The second family of instances are social networking graphs obtained from the Stanford Network Analysis Platform (SNAP)[4].

System Setup. Thetis was implemented using a combination of C++ (for Algorithm 1) and Matlab (for the augmented Lagrangian framework). Our implementation of the augmented Lagrangian framework was based on [6]. All experiments were run on a 4 Intel Xeon E7-4450 (40 cores @ 2Ghz) with 256GB of RAM running Linux 3.8.4 with a 15-disk RAID0. Cplex used 32 (of the 40) cores available in the machine, and for consistency, our implementation was also restricted to 32 cores. Cplex implements presolve procedures that detect redundancy, and substitute and eliminate variables to obtain equivalent, smaller LPs. Since the aim of this experiment is compare the algorithms used to solve LPs, we ran both Cplex-LP and Thetis on the reduced LPs generated by the presolve procedure of Cplex-LP. Both Cplex-LP and Thetis were run to a tolerance of $\epsilon = 0.1$. Additional experiments with Cplex-LP run using its default tolerance options are reported in Appendix E. We used the barrier optimizer while running Cplex-LP. All codes were provided with a time limit of 3600 seconds excluding the time taken for preprocessing as well as the runtime of the rounding algorithms that generate integral solutions from fractional solutions.

Tasks. We solved the vertex cover problem using the approximation algorithm described in Section 2. We solved the maximum independent set problem using a variant of the $es + o(s)$-factor approximation in [1] where s is the maximum degree of a node in the graph (see Appendix C for details). For the multiway-cut problem (with $k = 3$) we used the $3/2 - 1/k$-factor approximation algorithm described in [22]. The details of the transformation from approximate infeasible solutions to feasible solutions are provided in Appendix C. Since the rounding schemes for maximum-independent set and multiway-cut are randomized, we chose the best feasible integral solution from 10 repetitions.

Instance	**Minimization problems**								**Maximization problems**			
	VC				**MC**				**MIS**			
	PV	NNZ	S	Q	PV	NNZ	S	Q	PV	NNZ	S	Q
frb59-26-1	0.12	0.37	2.8	1.04	0.75	3.02	53.3	1.01	0.12	0.38	5.3	0.36
Amazon	0.39	1.17	8.4	1.23	5.89	23.2	-	0.42	0.39	1.17	7.4	0.82
DBLP	0.37	1.13	8.3	1.25	6.61	26.1	-	0.33	0.37	1.13	8.5	0.88
Google+	0.71	2.14	9.0	1.21	9.24	36.8	-	0.83	0.71	2.14	10.2	0.82

Figure 3: Summary of wall-clock speedup (in comparison with Cplex-LP) and solution quality (in comparison with Cplex-IP) of Thetis on three graph analysis problems. Each code is run with a time limit of one hour and parallelized over 32 cores, with '-' indicating that the code reached the time limit. PV is the number of primal variables while NNZ is the number of nonzeros in the constraint matrix of the LP in standard form (both in millions). S is the speedup, defined as the time taken by Cplex-LP divided by the time taken by Thetis. Q is the ratio of the solution objective obtained by Thetis to that reported by Cplex-IP. For minimization problems (**VC** and **MC**) lower Q is better; for maximization problems (**MIS**) higher Q is better. For MC, a value of $Q < 1$ indicates that Thetis found a better solution than Cplex-IP found within the time limit.

Results. The results are summarized in Figure 3, with additional details in Figure 4. We discuss the results for the vertex cover problem. On the Bhoslib instances, the integral solutions from Thetis were within 4% of the documented optimal solutions. In comparison, Cplex-IP produced

[3] `http://www.nlsde.buaa.edu.cn/~kexu/benchmarks/graph-benchmarks.htm`
[4] `http://snap.stanford.edu/`

VC (min)	Cplex IP			Cplex LP			Thetis		
	t (secs)	BFS	Gap (%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	-	1475	0.67	2.48	767	1534	0.88	959.7	1532
Amazon	85.5	1.60×10^5	-	24.8	1.50×10^5	2.04×10^5	2.97	1.50×10^5	1.97×10^5
DBLP	22.1	1.65×10^5	-	22.3	1.42×10^5	2.08×10^5	2.70	1.42×10^5	2.06×10^5
Google+	-	1.06×10^5	0.01	40.1	1.00×10^5	1.31×10^5	4.47	1.00×10^5	1.27×10^5
MC (min)	Cplex IP			Cplex LP			Thetis		
	t (secs)	BFS	Gap (%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	72.3	346	-	312.2	346	346	5.86	352.3	349
Amazon	-	12	NA	-	-	-	55.8	7.28	5
DBLP	-	15	NA	-	-	-	63.8	11.7	5
Google+	-	6	NA	-	-	-	109.9	5.84	5
MIS (max)	Cplex IP			Cplex LP			Thetis		
	t (secs)	BFS	Gap (%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	-	50	18.0	4.65	767	15	0.88	447.7	18
Amazon	35.4	1.75×10^5	-	23.0	1.85×10^5	1.56×10^5	3.09	1.73×10^5	1.43×10^5
DBLP	17.3	1.52×10^5	-	23.2	1.75×10^5	1.41×10^5	2.72	1.66×10^5	1.34×10^5
Google+	-	1.06×10^5	-	44.5	1.11×10^5	9.39×10^4	4.37	1.00×10^5	8.67×10^4

Figure 4: Wall-clock time and quality of fractional and integral solutions for three graph analysis problems using Thetis, Cplex-IP and Cplex-LP. Each code was given a time limit of one hour, with '-' indicating a timeout. BFS is the objective value of the best integer feasible solution found by Cplex-IP. The gap is defined as (BFS−BB)/BFS where BB is the best known solution bound found by Cplex-IP within the time limit. A gap of '-' indicates that the problem was solved to within 0.01% accuracy and NA indicates that Cplex-IP was unable to find a valid solution bound. LP is the objective value of the LP solution, and RSol is objective value of the rounded solution.

integral solutions that were within 1% of the documented optimal solutions, but required an hour for each of the instances. Although the LP solutions obtained by Thetis were less accurate than those obtained by Cplex-LP, the rounded solutions from Thetis and Cplex-LP are almost exactly the same. In summary, the LP-rounding approaches using Thetis and Cplex-LP obtain integral solutions of comparable quality with Cplex-IP — but Thetis is about three times faster than Cplex-LP.

We observed a similar trend on the large social networking graphs. We were able to recover integral solutions of comparable quality to Cplex-IP, but seven to eight times faster than using LP-rounding with Cplex-LP. We make two additional observations. The difference between the optimal fractional and integral solutions for these instances is much smaller than frb59-26-1. We recorded unpredictable performance of Cplex-IP on large instances. Notably, Cplex-IP was able to find the optimal solution for the *Amazon* and *DBLP* instances, but timed out on *Google+*, which is of comparable size. On some instances, Cplex-IP outperformed even Cplex-LP in wall clock time, due to specialized presolve strategies.

5 Conclusion

We described Thetis, an LP rounding scheme based on an approximate solver for LP relaxations of combinatorial problems. We derived worst-case runtime and solution quality bounds for our scheme, and demonstrated that our approach was faster than an alternative based on a state-of-the-art LP solver, while producing rounded solutions of comparable quality.

Acknowledgements

SS is generously supported by ONR award N000141310129. JL is generously supported in part by NSF awards DMS-0914524 and DMS-1216318 and ONR award N000141310129. CR's work on this project is generously supported by NSF CAREER award under IIS-1353606, NSF award under CCF-1356918, the ONR under awards N000141210041 and N000141310129, a Sloan Research Fellowship, and gifts from Oracle and Google. SJW is generously supported in part by NSF awards DMS-0914524 and DMS-1216318, ONR award N000141310129, DOE award DE-SC0002283, and Subcontract 3F-30222 from Argonne National Laboratory. Any recommendations, findings or opinions expressed in this work are those of the authors and do not necessarily reflect the views of any of the above sponsors.

References

[1] Nikhil Bansal, Nitish Korula, Viswanath Nagarajan, and Aravind Srinivasan. Solving packing integer programs via randomized rounding with alterations. Theory of Computing, 8(1):533–565, 2012.

[2] Dimitri P. Bertsekas. Nonlinear Programming. Athena Scientific, 1999.

[3] Jacob Bien and Robert Tibshirani. Classification by set cover: The prototype vector machine. arXiv preprint arXiv:0908.2284, 2009.

[4] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26:1124–1137, 2004.

[5] Gruia Călinescu, Howard Karloff, and Yuval Rabani. An improved approximation algorithm for multiway cut. In Proceedings of the thirtieth annual ACM symposium on Theory of Computing, pages 48–52. ACM, 1998.

[6] Jonathan Eckstein and Paulo JS Silva. A practical relative error criterion for augmented lagrangians. Mathematical Programming, pages 1–30, 2010.

[7] Dorit S Hochbaum. Approximation algorithms for the set covering and vertex cover problems. SIAM Journal on Computing, 11(3):555–556, 1982.

[8] VK Koval and MI Schlesinger. Two-dimensional programming in image analysis problems. USSR Academy of Science, Automatics and Telemechanics, 8:149–168, 1976.

[9] Frank R Kschischang, Brendan J Frey, and H-A Loeliger. Factor graphs and the sum-product algorithm. Information Theory, IEEE Transactions on, 47(2):498–519, 2001.

[10] Taesung Lee, Zhongyuan Wang, Haixun Wang, and Seung-won Hwang. Web scale entity resolution using relational evidence. Technical report, Microsoft Research, 2011.

[11] Victor Lempitsky and Yuri Boykov. Global optimization for shape fitting. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR '07), pages 1–8. IEEE, 2007.

[12] Ji Liu, Stephen J. Wright, Christopher Ré, and Victor Bittorf. An asynchronous parallel stochastic coordinate descent algorithm. Technical report, University of Wisconsin-Madison, October 2013.

[13] F Manshadi, Baruch Awerbuch, Rainer Gemulla, Rohit Khandekar, Julián Mestre, and Mauro Sozio. A distributed algorithm for large-scale generalized matching. Proceedings of the VLDB Endowment, 2013.

[14] Feng Niu, Benjamin Recht, Christopher Ré, and Stephen J. Wright. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. arXiv preprint arXiv:1106.5730, 2011.

[15] Jorge Nocedal and Stephen J Wright. Numerical Optimization. Springer, 2006.

[16] Pradeep Ravikumar, Alekh Agarwal, and Martin J Wainwright. Message-passing for graph-structured linear programs: Proximal methods and rounding schemes. The Journal of Machine Learning Research, 11:1043–1080, 2010.

[17] J. Renegar. Some perturbation theory for linear programming. Mathenatical Programming, Series A, 65:73–92, 1994.

[18] Dan Roth and Wen-tau Yih. Integer linear programming inference for conditional random fields. In Proceedings of the 22nd International Conference on Machine Learning, pages 736–743. ACM, 2005.

[19] Sujay Sanghavi, Dmitry Malioutov, and Alan S Willsky. Linear programming analysis of loopy belief propagation for weighted matching. In Advances in Neural Information Processing Systems, pages 1273–1280, 2007.

[20] Aravind Srinivasan. Improved approximation guarantees for packing and covering integer programs. SIAM Journal on Computing, 29(2):648–670, 1999.

[21] Jurgen Van Gael and Xiaojin Zhu. Correlation clustering for crosslingual link detection. In IJCAI, pages 1744–1749, 2007.

[22] Vijay V Vazirani. Approximation Algorithms. Springer, 2004.

[23] Stephen J. Wright. Implementing proximal point methods for linear programming. Journal of Optimization Theory and Applications, 65(3):531–554, 1990.

[24] Zheng Wu, Ashwin Thangali, Stan Sclaroff, and Margrit Betke. Coupling detection and data association for multiple object tracking. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pages 1948–1955. IEEE, 2012.

[25] Ke Xu and Wei Li. Many hard examples in exact phase transitions. Theoretical Computer Science, 355(3):291–302, 2006.

[26] Ce Zhang and Christopher Ré. Towards high-throughput gibbs sampling at scale: A study across storage managers. In SIGMOD Proceedings, 2013.

Supplementary Material

A Perturbation Results

We discuss here the perturbation results for the quadratic approximation (5) to the linear program (3). These results constitute a proof of Theorem 5.

We note for future reference that the optimality (KKT) conditions for the primal-dual pair of LPs (3) and (4) are

$$Ax = b, \quad 0 \le c - A^T u \;\perp\; x \ge 0. \tag{9}$$

The QP approximation (5) is equivalent to the following monotone linear complementarity problem (LCP):

$$0 \le x \;\perp\; F_\beta(x) := c - A^T \bar{u} + \beta A^T(Ax - b) + \frac{1}{\beta}(x - \bar{x}). \tag{10}$$

Here we rely on Renegar's theory [17] which requires not only that primal and dual are both solvable, but also that they are still solvable after we make arbitrary small perturbations to the data (A, b, c). This includes cases in which the basis has fewer nonzeros than there are equality constraints (a situation known as "primal degeneracy"). We assume throughout that A has full row rank m. If A were row rank deficient, then even if the primal-dual pair had a solution, we would be able to find an arbitrarily small perturbation that renders the primal infeasible.

In accordance with Renegar, we use $d := (A, b, c)$ to denote the data for the problems (3) and (4). We denote by Pri$\emptyset$ the set of data d for which the primal (3) is infeasible, and define Dual$\emptyset$ analogously for the dual (4). Renegar uses the "distance to infeasibility" to define a condition number for the primal and dual. Specifically, defining

$$\delta_P := \frac{\text{dist}(d, \text{Pri}\emptyset)}{\|d\|}, \quad \delta_D := \frac{\text{dist}(d, \text{Dual}\emptyset)}{\|d\|}, \tag{11}$$

the quantities $1/\delta_P$ and $1/\delta_D$ capture the sensitivity of the optimal objective value for the problem (3) to perturbations in b and c. Note that both δ_P and δ_D lie in the interval $[0, 1]$.

We assume $\delta_P > 0$ and $\delta_D > 0$ throughout the analysis below. This implies that the primal and dual are both feasible, hence by strong duality both have solutions x^* and u^* (not necessarily unique).

Lemma 7. *Suppose that $\delta_P > 0$ and $\delta_D > 0$, and let x^* be any solution of (3) and u^* be any solution of (4), and define*

$$C_* := \max(\|x^* - \bar{x}\|, \|u^* - \bar{u}\|).$$

Then the unique solution $x(\beta)$ of (5) satisfies the following inequalities:

$$\begin{aligned} \|Ax(\beta) - b\| &\le \beta^{-1} \left[\|u^* - \bar{u}\| + \sqrt{\|u^* - \bar{u}\|^2 + \|x^* - \bar{x}\|^2} \right] \\ &\le \beta^{-1}(1 + \sqrt{2}) C_*, \\ \|x(\beta) - \bar{x}\| &\le \left[2\|u^* - \bar{u}\| \left[\|u^* - \bar{u}\| + \sqrt{\|u^* - \bar{u}\|^2 + \|x^* - \bar{x}\|^2} \right] + \|x^* - \bar{x}\|^2 \right]^{1/2} \\ &\le \sqrt{6} C_*. \end{aligned}$$

Proof. Note that x^* is a feasible point for (5), so we have by optimality of $x(\beta)$ that $f_\beta(x(\beta)) \le f_\beta(x^*)$, that is,

$$c^T x(\beta) - \bar{u}^T(Ax(\beta) - b) + \frac{\beta}{2}\|Ax(\beta) - b\|^2 + \frac{1}{2\beta}\|x(\beta) - \bar{x}\|^2 \le c^T x^* + \frac{1}{2\beta}\|x^* - \bar{x}\|^2,$$

and thus

$$\frac{\beta}{2}\|Ax(\beta) - b\|^2 + \frac{1}{2\beta}\|x(\beta) - \bar{x}\|^2 \le c^T(x^* - x(\beta)) + \bar{u}^T(Ax(\beta) - b) + \frac{1}{2\beta}\|x^* - \bar{x}\|^2.$$

Note from $x(\beta) \ge 0$ and (9) that

$$0 \le x(\beta)^T(c - A^T u^*) \;\Rightarrow\; -c^T x(\beta) \le -(u^*)^T A x(\beta).$$

We also have from (9) that $c^T x^* = (u^*)^T A x^*$. By combining these observations, we obtain

$$\frac{\beta}{2}\|Ax(\beta) - b\|^2 + \frac{1}{2\beta}\|x(\beta) - \bar{x}\|^2 \le (u^* - \bar{u})^T A(x^* - x(\beta)) + \frac{1}{2\beta}\|x^* - \bar{x}\|^2. \qquad (12)$$

By dropping the second term on the left-hand side of this expression, multiplying by β, and using Cauchy-Schwartz and $Ax^* = b$, we obtain

$$\frac{\beta^2}{2}\|Ax(\beta) - b\|^2 \le \|u^* - \bar{u}\|\beta\|Ax(\beta) - b\| + \frac{1}{2}\|x^* - \bar{x}\|^2.$$

Denoting $e_\beta := \beta\|Ax(\beta) - b\|$, this inequality reduces to the condition

$$\frac{1}{2}e_\beta^2 - \|u^* - \bar{u}\|e_\beta - \frac{1}{2}\|x^* - \bar{x}\|^2 \le 0.$$

Solving this quadratic for e_β, we obtain

$$e_\beta \le \|u^* - \bar{u}\| + \sqrt{\|u^* - \bar{u}\|^2 + \|x^* - \bar{x}\|^2},$$

proving the first claim.

For the second claim, we return to (12), dropping the first term on the left-hand side, to obtain

$$\frac{1}{2\beta}\|x(\beta) - \bar{x}\|^2 \le \|u^* - \bar{u}\|\|Ax(\beta) - b\| + \frac{1}{2\beta}\|x^* - \bar{x}\|^2.$$

By substituting the bound on $\|Ax(\beta) - b\|$ just derived, multiplying by 2β and taking the square root, we obtain the result. □

Fixing β and $x(\beta)$, we now consider the following perturbed linear program

$$\min\ c_\beta^T x \ \text{ s.t. } \ Ax = b_\beta, \ \ x \ge 0, \qquad (13)$$

and its dual

$$\max\ b_\beta^T u \ \text{ s.t. } \ A^T u \le c_\beta, \qquad (14)$$

where

$$b_\beta := Ax(\beta), \quad c_\beta := c + \frac{1}{\beta}(x(\beta) - \bar{x}).$$

From Lemma 7, we have

$$\|b - b_\beta\| \le \frac{1}{\beta}(1 + \sqrt{2})C_* \le \frac{2.5C_*}{\beta}, \quad \|c - c_\beta\| \le \frac{1}{\beta}\sqrt{6}C_* \le \frac{2.5C_*}{\beta}. \qquad (15)$$

KKT conditions for (13), (14) are

$$0 \le \hat{x} \perp c + \frac{1}{\beta} - A^T\hat{u} \ge 0, \quad A\hat{x} = Ax(\beta).$$

It is easy to check, by comparing with (10), that these conditions are satisfied by

$$\hat{x} = x(\beta), \quad \hat{u} = \bar{u} - \beta A(x(\beta) - x^*).$$

Hence $\hat{x} = x(\beta)$ is a solution of (13). There may be other solutions, but they will have the same objective value, of course.

We now use the following result, which follows immediately from [17, Theorem 1, part (5)].[5]

Theorem 8. *Let $d = (A, b, c)$ be the data defining the primal-dual pair (3) and (4), and suppose that δ_P and δ_D defined by (11) are both positive. Consider the following perturbation applied to the b and c components: $\Delta d := (0, \Delta b, \Delta c)$, and assume that*

$$\frac{\|\Delta d\|}{\|d\|} \le \delta_P, \quad \frac{\|\Delta d\|}{\|d\|} \le \delta_D.$$

Then, denoting the solution of (3) by x^* and the solution of the linear program with perturbed data $d + \Delta d$ by x^*_Δ, we have

$$|c^T x^* - (c + \Delta c)^T x^*_\Delta| \le \frac{\|\Delta b\|}{\delta_D}\frac{\|c\| + \|\Delta c\|}{\text{dist}(d, \text{Pri}\emptyset) - \|\Delta d\|} + \frac{\|\Delta c\|}{\delta_P}\frac{\|b\| + \|\Delta b\|}{\text{dist}(d, \text{Dual}\emptyset) - \|\Delta d\|}.$$

[5] Note that Renegar appears to use a different formulation for the linear program, namely $Ax \le b$ rather than $Ax = b$. However, his inequality represents a complete ordering with respect to a closed convex cone C_Y, and when we set $C_Y = \{0\}$, we recover $Ax = b$.

Our main theorem is obtained by applying this result with the perturbations

$$\Delta b := b_\beta - b = Ax(\beta) - b, \quad \Delta c := c_\beta - c = \frac{1}{\beta}(x(\beta) - \bar{x}). \tag{16}$$

We have the following result.

Theorem 9. *Suppose that*

$$\beta \geq \bar{\beta} := \frac{10C_*}{\|d\| \min(\delta_P, \delta_D)}.$$

We then have the following bound on the difference between the optimal values of (3) and (13):

$$|c^T x^* - c_\beta^T x(\beta)| \leq \frac{1}{\beta} \frac{25C_*}{2\delta_P \delta_D}.$$

Proof. Note first that from (15) and

$$\|\Delta d\| \leq \|\Delta b\| + \|\Delta c\| \leq \frac{5C_*}{\beta}.$$

From our assumption on β, we have

$$\frac{\|\Delta d\|}{\|d\|} \leq \frac{5C_*}{\beta \|d\|} \leq \frac{1}{2} \min(\delta_P, \delta_D),$$

so that the assumptions of Theorem 8 are satisfied. We have moreover from the definitions (11) that

$$\text{dist}(d, \text{Pri}\emptyset) - \|\Delta d\| = \|d\| \left[\delta_P - \frac{\|\Delta d\|}{\|d\|} \right] \geq \frac{1}{2} \|d\| \delta_P,$$

and similarly $\text{dist}(d, \text{Dual}\emptyset) \geq (1/2)\|d\|\delta_D$. By substituting into the inequality of Theorem 8, and using the bounds just derived together with (15), we obtain

$$|c^* x^* - c_\beta^T x(\beta)| \leq \frac{2.5\beta^{-1} C_*}{\delta_D} \frac{(\|c\| + 2.5\beta^{-1} C_*)}{.5\|d\|\delta_P} + \frac{2.5\beta^{-1} C_*}{\delta_P} \frac{(\|b\| + 2.5\beta^{-1} C_*)}{.5\|d\|\delta_D}.$$

Since

$$\|c\| \leq \|d\|, \quad \|b\| \leq \|d\|, \quad \frac{2.5C_*}{\beta} \leq \frac{1}{4} \min(\delta_P, \delta_D) \|d\| \leq \frac{1}{4} \|d\|,$$

we have

$$|c^* x^* - c_\beta^T x(\beta)| \leq \frac{2.5\beta^{-1} C_* (2.5) \|d\|}{(1/2)\|d\| \delta_P \delta_D} = \frac{1}{\beta} \frac{25C_*}{2\delta_P \delta_D},$$

completing the proof. ☐

The following corollary is almost immediate.

Corollary 10. *Suppose the conditions of Theorem 9 are satisfied. Then*

$$|c^T x^* - c^T x(\beta)| \leq \frac{1}{\beta} \left[\frac{25C_*}{2\delta_P \delta_D} + 6C_*^2 + \sqrt{6} \|\bar{x}\| C_* \right].$$

Proof. We have from the definition of c_β that

$$\begin{aligned}
|c^T x^* - c^T x(\beta)| &\leq |c^T x^* - c_\beta^T x(\beta)| + \frac{1}{\beta} x(\beta)^T (x(\beta) - \bar{x}) \\
&= |c^T x^* - c_\beta^T x(\beta)| + \frac{1}{\beta} \|x(\beta) - \bar{x}\|^2 + \frac{1}{\beta} \bar{x}^T (x(\beta) - \bar{x}) \\
&\leq \frac{1}{\beta} \left[\frac{25C_*}{2\delta_P \delta_D} + 6C_*^2 + \sqrt{6} \|\bar{x}\| C_* \right].
\end{aligned}$$

where the final inequality follow from Lemma 7 and Theorem 9. ☐

B Details of Rounding Schemes

In this section, we provide details of known LP-rounding schemes for covering, packing and multiway-cut problems. (Vazirani [22] provides a comprehensive survey on the theory and algorithms for LP-rounding.). We then discuss how these algorithms can be extended to round (ϵ, δ) optimal solutions.

B.1 Set Cover

Given a universe U with N elements, a collection of subsets $\mathcal{S} = \{S_1, S_2 \dots S_k\}$ each associated with a positive cost function $c : S \to \mathbb{R}^+$. In the set cover problem, we must identify a minimum cost sub-collection of sets S that covers all elements in U. The set cover problem can be formulated as the following IP:

$$\min \sum_{s \in \mathcal{S}} c_s x_s \quad \text{subject to} \quad \sum_{s:a \in s} x_s \geq 1 \;\; \forall a \in U, \; x_s \in \{0,1\} \; \forall s \in \mathcal{S}. \tag{17}$$

A simple way to convert a solution x_s^* of the LP relaxation to an integral solution is to pick all sets x_s where $x_s^* > 1/f$, where f is a bound on the maximum number of sets in which a single element is present. Such an algorithm achieves an f-factor approximation [7]. An alternative approximation scheme is a randomized scheme due to [20]. In this scheme, we put $s \in \mathcal{S}$ into the set cover with probability equal to the optimal fractional solution x_s^*. In expectation, this approximation scheme is a $O(\log N)$-factor approximation, and is a valid set cover with probability $1/2$.

B.2 Set Packing

Using the same notation for U, N, $\mathcal{S}$, and x_s, $\forall s \in \mathcal{S}$ as above, the set packing problem is to identify the lowest cost collection of mutually disjoint sets. It can be formulated as the following IP:

$$\max \sum_{s \in \mathcal{S}} c_s x_s \quad \text{subject to} \quad \sum_{s:a \in s} w_{a,s} x_s \leq 1 \;\; \forall a \in U, \; x_s \in \{0,1\} \; \forall s \in \mathcal{S}, \tag{18}$$

where $w_{a,s}$ is the weight of element $a \in U$ in set $s \in \mathcal{S}$.

Bansal et al. [1] proposed an $ek + o(k)$-factor approximation (see Algorithm B.2) for the special case of k-column sparse set packing where the maximum number of sets containing each element is at most k. They use the following stronger formulation of the set packing problem:

$$\begin{aligned} & \max \sum_{s \in \mathcal{S}} c_s x_s && && (19) \\ \text{subject to} \quad & \sum_{s:a \in s} w_{a,s} x_s \leq 1 && \forall a \in U, \\ & \sum_{a \in B(s)} x_s \leq 1 && \forall a \in U, \\ & x_s \in \{0,1\} && \forall s \in \mathcal{S}, \end{aligned}$$

where $w_{a,s} = 1$ if the element $a \in U$ is present in set $s \in S$, c_s is the cost of set $s \in S$ and $B(s) := \{a \in U | w_{a,s} > 1/2\}$.

Algorithm 2 A $ek + o(k)$-factor randomized LP-rounding algorithm for set packing

1: Find any feasible solution $\hat{x}$ to the LP relaxation of (19).
2: Choose set $s \in \mathcal{S}$ with probability $\hat{x}_s/(k\theta)$. Let $\mathcal{C} \subseteq \mathcal{S}$ denote the chosen sets.
3: For each set $s \in \mathcal{C}$ and element $a \in U$, let $E_{a,s}$ denote the event that the sets $\{s_2 \in \mathcal{C} : w_{a,s_2} > w_{a,s}\}$ have a total weight (with respect to element a) exceeding 1. Mark s for deletion if $E_{a,s}$ occurs for any $a \in s$.
4: Delete all sets from $s \in \mathcal{C}$ that are marked for deletion.

B.3 Multiway-Cuts

Given a graph $G(V, E)$ and a set of terminals $V_1, V_2, \dots V_k$, a k-way cut partitions the set of vertices V into k mutually disjoint sets. The cost of the k-way cut is the sum of the costs of all the edges that run across the partitions. A k-way cut of minimum cost is the solution to the following problem:

$$\begin{aligned} \min \frac{1}{2} \sum_{u,v \in E} c_{u,v} \sum_{i=1}^{k} |x_u^i - x_v^i| \\ \text{subject to} \quad x_v \in \Delta_k \quad \forall v \in V \\ x_v \in \{0,1\}^k \quad \forall v \in V, \end{aligned} \tag{20}$$

where $\Delta_k := \{x \in \mathbb{R}^k : \sum_{i=1}^k x_i = 1,\ x \geq 0\}$ is the set of simplex constraints in k dimensions. Although it might appear that the formulation in (20) is non-linear, one can easily linearize (20) to

$$\begin{aligned} &\min \frac{1}{2} \sum_{u,v \in E} c_{u,v} \sum_{i=1}^{k} x_{uv}^i \\ \text{subject to} \quad & x_v \in \Delta_k && \forall v \in V \\ & x_{uv}^i \geq x_v^i - x_u^i && \forall u,v \in E, i \in \{1 \dots k\} \\ & x_{uv}^i \geq x_u^i - x_v^i && \forall u,v \in E, i \in \{1 \dots k\} \\ & x_{uv}^i \in [0,1] && \forall u,v \in E, i \in \{1 \dots k\} \\ & x_v^i \in \{0,1\} && \forall v \in V, i \in \{1 \dots k\} \end{aligned}$$

There is a $3/2 - 1/k$ factor approximation for multiway-cut using the region-growing algorithm due to [5]. The details of the algorithm are laid out in [22, Algorithm 19.4].

C Rounding Infeasible Solutions

In this section, we briefly describe how we can extend known LP-rounding algorithms to infeasible (ϵ, δ)-approximate solutions. We discuss how one can go from an (ϵ, δ)-approximate solution to a feasible $(0, f(\epsilon, \delta))$-approximate solution, for some positive function $f(\cdot, \cdot)$. The arguments in this section are based on simple ideas of scaling and projection.

As is the case in the main manuscript, we illustrate our approach using vertex cover. Let $\hat{x}$ be an (ϵ, δ)-approximate solution of the following vertex cover LP:

$$\min_{x \in [0,1]^n} 1^T x \quad \text{subject to } \ x_i + x_j \geq 1 \text{ for } (i,j) \in E,$$

so that in particular, $x_i \in [0,1]$ for all i, and $x_i + x_j \geq 1 - \epsilon$ for all $(i,j) \in E$. We claim that the point

$$z := \Pi_{[0,1]^n}(x/(1-\epsilon))$$

is a $(0, \delta/(1-\varepsilon))$-approximate solution. To check feasibility, suppose for contradiction that $z_i + z_j < 1$ for some $(i,j) \in E$. We thus have $z_i < 1$ and $z_j < 1$, so that $z_i = x_i/(1-\epsilon)$ and $z_j = x_j/(1-\epsilon)$. Therefore, $z_i + z_j = (x_i + x_j)/(1-\epsilon) \geq 1$, a contradiction.

C.1 Rounding for Coverings

We consider a covering program $P = (A, b, c)$ with positive integer data, that is, $(A, b, c) \geq 0$ and $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, and $c \in \mathbb{Z}^n$. Suppose that there are also $[0,1]$ bound constraints on each component of x. The problem formulation is as follows:

$$\min_{x \in [0,1]^n} c^T x \quad \text{subject to } Ax \geq b. \qquad [P(A,b,c)]$$

To obtain a formulation closer to the standard form (3), we can introduce slack variables and write

$$\min_{x \in [0,1]^n, z \in [0,\infty)^m} c^T x \quad \text{subject to } Ax - z = b,\ z \geq 0.$$

We can always set $z = \max\{Ax - b, 0\}$ to translate between feasible solutions of the two programs.

The following quantity $q(P)$ defines a minimum infeasibility measure over all infeasible, integral solutions to P:

$$q(P) = \min_{j=1,\dots,m} \min_{x \in \{0,1\}^n : A_{j\cdot} x < b_j} b_j - A_{j\cdot} x,$$

where $A_{j\cdot}$ denotes the jth row of A. Notice for $q(P) \geq 1$ for any non-trivial covering program P, by integrality alone.

Lemma 11. *Let P be a covering program with a nonempty solution set. Let $\hat{x}$ be an (ϵ, δ)-approximate solution of P, and suppose that $\epsilon/q(P) \le 1$. Then there is a $(0, \delta/(1-\alpha))$-approximate solution $\tilde{x}$ defined as*

$$\tilde{x} = \Pi_{[0,1]^n}((1-\alpha)^{-1}\hat{x}),$$

where $\alpha \in [\epsilon/q(P), 1)$.

Proof. We first show that $\tilde{x}$ is feasible. Without loss of generality, assume that $z_j = \max(A_{j\cdot}\hat{x} - b_j, 0)$ for $j = 1, \ldots, m$. Since $\hat{x}$ is a (ϵ, δ) solution, we have $\|A\hat{x} - z - b\|_\infty \le \epsilon$. With z defined as in our formula, this bound implies that

$$A\hat{x} = b \ge -\epsilon \mathbf{1}, \tag{21}$$

where $\mathbf{1}$ is the all-ones vector in $\mathbb{R}^n$. After scaling by $\hat{x}$ by $(1-\alpha)^{-1}$, some components may exceed 1. Hence, we partition the indices into two sets $\Omega_1 = \{i \mid \hat{x}_i \ge 1-\alpha\}$ and $\Omega_{<1} = \{1, 2, \ldots, n\} \backslash \Omega_1$. For any $\Omega \subseteq [n]$, we define the following projection operator:

$$\pi_\Omega(x) := \begin{cases} x_i & \text{if } i \in \Omega \\ 0 & \text{otherwise.} \end{cases}$$

We can then write $\tilde{x}$ as follows:

$$\tilde{x} = \pi_{\Omega_1}\mathbf{1} + (1-\alpha)^{-1}\pi_{\Omega_{<1}}\hat{x}.$$

Assume for contradiction that $\tilde{x}$ is infeasible. Then there must be some constraint j for which $A_{j\cdot}\tilde{x} < b_j$. Using the decomposition above and the fact that $\alpha \in (0, 1)$, we have

$$A_{j\cdot}\pi_{\Omega_{<1}}\hat{x} < (b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1})(1-\alpha). \tag{22}$$

On the other hand, by (21), we have

$$A_{j\cdot}(\pi_{\Omega_1}\mathbf{1} + \pi_{\Omega_{<1}}\hat{x}) \ge A_{j\cdot}(\pi_{\Omega_1}\hat{x} + \pi_{\Omega_{<1}}\hat{x}) \ge b_j - \epsilon.$$

and so

$$A_{j\cdot}\pi_{\Omega_{<1}}\hat{x} \ge b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1} - \epsilon \tag{23}$$

By combining (22) and (23), we obtain

$$(b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1})(1-\alpha) > (b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1}) - \epsilon$$

Since $b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1} \ge b_j - A_{j\cdot}\tilde{x} > 0$, we can divide by $b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1}$ without changing signs to obtain

$$\frac{\epsilon}{b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1}} > \alpha \quad \Rightarrow \quad b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1} < \epsilon/\alpha. \tag{24}$$

We have by using the definition of α that $b_j - A_{j\cdot}\pi_{\Omega_1}\mathbf{1} \ge q(P) \ge \epsilon/\alpha$, since $\pi_\Omega \mathbf{1}$ is an integral but infeasible point for (P). This fact contradicts (24), so we have proved feasibility of $\tilde{x}$ for (P).

We now bound the difference between $c^*\tilde{x}$ and c^*x^*, where x^* is the optimal solution of (P). Since $\tilde{x}$ is feasible, we have that $c^T x^* \le c^T\tilde{x}$. For the upper bound, we have

$$c^T\tilde{x} - c^T x^* \le (1-\alpha)^{-1}c^T\hat{x} - c^T x^* \le (1-\alpha)^{-1}(c^T\hat{x} - c^T x^*) \le \frac{\delta}{1-\alpha}c^T x^*.$$

The first inequality follows from $c^T z \ge c^T(\Pi_{[0,1]^n} z)$ since $c \ge 0$; the second inequality is from $\alpha \in (0, 1)$; and the third inequality follows from the fact that $\hat{x}$ is a (ϵ, δ) approximation. □

In our experiments, we set $\alpha = \epsilon/q(P)$, which is computed using the approximate (ϵ, δ) optimal fractional solution.

C.2 Rounding for Packing

A packing problem is a maximization linear program $P(A, b, c)$ where $A, b, c \geq 0$ and $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, and $c \in \mathbb{Z}^n$ along with bound constraints $[0, 1]$ on all variables. That is,

$$\max_{u \in [0,1]^m} u^T b \quad \text{subject to} \quad A^T u \leq c. \qquad [P(A, b, c)]$$

In this class of problems, we can assume without loss of generality that $c \geq \mathbf{1}$. The equality constrained formulation of this problem is

$$\max_{u \in [0,1]^m, z \in \mathbb{R}^n} u^T b \quad \text{subject to} \quad A^T u + z = c, \ z \geq 0.$$

(We can set $z = \max(c - A^T u, 0)$ to obtain the equivalence.)

We use $A_{\cdot i}$ to denote the ith column of A in the discussion below.

Lemma 12. *Let P be a packing program. Let $\hat{u}$ be an (ϵ, δ)-approximate solution of P, then there is a $(0, \frac{\delta+\alpha}{1+\alpha})$-approximate solution $\tilde{u}$ defined as*

$$\tilde{u} = \hat{u}/(1 + \alpha)$$

provided that $\hat{u} \in [0, 1]^m$ where $\alpha \geq \epsilon / (\min_{i=1,2,\ldots,n} c_i)$.

Proof. We observe first that $\tilde{u} \in [0, 1]^m$. To prove that $A^T \tilde{u} \leq c$, note that since $\hat{u}$ is an (ϵ, δ)-approximate solution, we have

$$A^T \hat{u} \leq c + \epsilon \mathbf{1} \leq c + \alpha \left(\min_{l=1,2,\ldots,n} c_l \right) \mathbf{1} \leq (1 + \alpha) c,$$

proving the claim.

Let u_* be an optimal solution of $P(A, b, c)$. Since $\tilde{u}$ is feasible and this is a maximization problem, we have $u_*^T b \geq \tilde{u}^T b \geq 0$. For the other bound, we have

$$u_*^T b - \tilde{u}^T b = u_*^T b - \frac{1}{1+\alpha} \hat{u}^T b \leq u_*^T b - \frac{1-\delta}{1+\alpha} u_*^T b = \frac{\delta+\alpha}{1+\alpha} u_*^T b,$$

completing the proof. ☐

A quick examination of the proof suggests that we can take $\alpha := \left(\max_{i=1,2,\ldots,n} \frac{A_{\cdot i}^T \hat{u} - c_i}{c_i} \right)_+$, which is never larger than α as defined above. In our experiments, we set α using this tighter bound and $\theta = \frac{1}{k}$ in algorithm B.2. We note that the algorithm is sensitive to the value of θ. Any positive value of $\theta k \geq 1$ will always return a valid independent set. The proofs in [1] require that θ must be greater or equal to 1, but we found that $\theta = \frac{1}{k}$ works much better in practice.

C.3 Rounding for Multiway-Cuts

Since we enforce the simplex constraints in the SCD solve, every solution obtained by our quadratic relaxation is automatically feasible for our linear program.

D Linear Programming Condition Numbers

In this section, we describe estimates of (δ_P, δ_D) in detail for vertex cover, and sketch the ideas for estimating these quantities for the other relaxations that we consider in this paper.

D.1 Vertex Cover: The Bounds in Detail

Consider vertex cover with a graph $G = (V, E)$, where $|V| = n$ and $|E| = m$. The LP relaxation is as follows

$$\min_{x \in \mathbb{R}^n_+} \mathbf{1}^T x \quad \text{subject to} \quad x_v + x_w \geq 1 \text{ for all } (v, w) \in E \text{ and } x_v \leq 1 \text{ for all } v \in V. \qquad (25)$$

The dual of this program is

$$\max_{u \in \mathbb{R}^m_+, z \in \mathbb{R}^+} u^T \mathbf{1} - z^T \mathbf{1} \quad \text{subject to} \quad \sum_{e : e \ni v} u_e - z_v \leq 1 \text{ for each } v \in V.$$

Computing $\|d\|$. Define $\|d\| = \max\{\|A\|_F, \|b\|_2, \|c\|_2\}$ for this problem, where (A, b, c) are the data defining (25). We have

$$\|A\|_F = \sqrt{2m+n}, \quad \|b\|_2 = \sqrt{m+n} \quad \|c\| = \sqrt{n}$$

Hence, $\|d\| = \sqrt{2m+n}$.

Primal Bound. We define $x = \frac{2}{3}\mathbf{1}$, and figure how large a perturbation $(\Delta A, \Delta b, \Delta c)$ is needed to problem data (A, b, c) to make this particular point infeasible. The norm of this quantity will give a lower bound on the distance to infeasibility.

By construction of x, we have that $Ax - b = \frac{1}{3}\mathbf{1}$. For infeasibility with respect to one of the cover constraints, we would need for some i that

$$|(\Delta A)_{i\cdot}x - \Delta b_i| \geq \frac{1}{3},$$

which, given our definition of x, would require

$$\frac{2}{3}\sum_{j=1}^{n}|\Delta A_{ij}| + |\Delta b_i| > \frac{1}{3}. \tag{26}$$

We must therefore have that

$$\sum_{j=1}^{n}|\Delta A_{ij}| \geq \frac{1}{4} \ \text{ and/or } \ |\Delta b_i| > \frac{1}{6}.$$

In the first case, noting that

$$\frac{1}{4}n^{-1/2} = \min_{z\in[0,1]^n}\|z\|_2 \quad \text{subject to } \ z^T\mathbf{1} \geq \frac{1}{4},$$

we would have that $\|\Delta A\|_F \geq \|(\Delta A)_{i\cdot}\|_2 \geq n^{-1/2}/4$. In the second case, we would have $\|\Delta b\|_2 \geq |\Delta b_i| \geq 1/6$.

Suppose that the infeasibility happens instead with respect to one of the $x \leq \mathbf{1}$ constraints. A similar argument for the violated constraint would lead to the same necessary condition (26) and the same bounds.

In either case, assuming that $n \geq 3$, we have

$$\|(\Delta A, \Delta b, \Delta c)\| \geq n^{-1/2}/4,$$

so that

$$\delta_P \geq \|d\|^{-1}n^{-1/2}/4.$$

Dual Bound. We consider here a fixed vector $(u, z) = 0$. For infeasibility, we would need $\Delta c_i < -1$ for some i, and therefore $\|\Delta d\| \geq 1$.We thus have

$$\delta_D \geq \|d\|^{-1}$$

Putting the primal and dual bounds together, and using our bound on $\|d\|$, we obtain

$$\frac{1}{\delta_P\delta_D} = O(\|d\|^2 n^{1/2}) = O((m+n)n^{1/2}).$$

D.2 Packing and Covering Programs

Suppose we have a covering program with data $(A, b, c) \geq 0$, with $[0, 1]$ bound constraints on each variable. That is,

$$\min_{x\in\mathbb{R}^n_+} c^T x \quad \text{subject to } \ Ax \geq b, x \leq \mathbf{1},$$

its dual is a packing program:

$$\max_{u\in\mathbb{R}^m_+, z\in\mathbb{R}^n_+} u^T b - z^T\mathbf{1} \quad \text{subject to } \ A^T u - z \leq c.$$

Generalizing our argument above, we find a point that has the most slack from each constraint. Defining the following measure of slack:

$$s(A,b,c) = \max_{x \in \mathbb{R}^n_+ : Ax \geq b, x \leq \mathbf{1}} \min\{ \min_{i=1,\ldots,n} 1 - x_i, \min_{j=1,\ldots,m} b_j - A_{j\cdot}x \},$$

we can obtain a lower bound $\delta_P \geq \|d\|^{-1} n^{-1/2} s(A,b,c)/2$, as follows. Suppose that x_S is the point that achieves the maximum slack. We need that one of the following conditions holds for at least one constraint i: $\Delta A_{i\cdot} x_S > s(P)/2$ or $|\Delta b_i| \geq s(P)/2$. Observe that

$$\Delta A_{i\cdot} x_S \leq \|x_S\|_2 \|\Delta A_{i\cdot}\|_2 \leq n^{1/2} \|\Delta A_i\|_2.$$

(The second inequality follows from $0 \leq x_S \leq \mathbf{1}$.) Thus, in this case, $\|\Delta A_i\|_2 > s(A,b,c) n^{-1/2}/2$. Using a similar argument to the previous subsection, we have

$$\delta_P \geq \|d\|^{-1} \|\Delta d\| \geq \|d\|^{-1} s(A,b,c) n^{-1/2}/2.$$

Since $(u,z) = (0,0)$ is feasible for the dual, we have by a similar argument to the previous subsection that infeasibility occurs only if $|\Delta c_i| \geq c_i$ for at least one i. We therefore have $\|\Delta d\| \geq \min_{i=1,2,\ldots,n} c_i$, so that

$$\delta_D \geq \|d\|^{-1} \min_{i=1,2,\ldots,n} c_i.$$

Putting the bounds on δ_P and δ_D together, we have

$$\frac{1}{\delta_P \delta_D} \leq \|d\|^2 \frac{1}{s(A,b,c) \min_{i=1,2,\ldots,n} c_i} O(n^{1/2}).$$

E Extended Experimental Results

In this section, we elaborate our discussion on the experimental results in Section 4.2 and provide additional evidence to support our claims. Figures 5 and 6 compare the performance of Thetis with Cplex-IP and Cplex-LP on all tested instances of vertex cover, independent set, and multiway-cut. In all three formulations, we used unit costs in the objective function. The results in Figure 6 were obtained by using default tolerance on Cplex-LP, while Figure 5 uses the same tolerance setting as the main manuscript.

Maximum Independent Set. We observed that the rounded feasible solutions obtained using Thetis were of comparable quality to those obtained by rounding the more accurate solutions computed by Cplex-LP. The integral solutions obtained from Cplex-IP were only marginally better than that obtained by LP-rounding, but at a cost of at least an order of magnitude more time.

Multiway Cuts. The number of variables in the multiway-cut problem is $O((|E| + |V|) \times k)$ where $|E|$ is the number of edges, $|V|$ is the number of vertices and k is the number of terminals. The terminals were chosen randomly to be in the same connected component of the graph. All the test instances, excepting Google+, were fully connected. For Google+, 201949 (of 211186 vertices) were connected to the terminals. For all instances, including Google+, all codes were run on (20) built using the entire graph.

We solved the QP-approximation of (20) using a block-SCD method, which is variant of Algorithm 1, in which an update step modifies a block of co-ordinates of size k. For the blocks corresponding to variables x_v, $\forall v \in V$, we performed a projection on to the k-dimensional simplex Δ_k. The simplex projection was necessary to ensure that the approximate LP solution is always feasible for (20). We disabled presolve for Thetis to prevent the simplex constraints from being eliminated or altered. We did not disable presolve for Cplex-LP or Cplex-IP.

Our results demonstrate that Thetis is much more scalable than both Cplex-IP and Cplex-LP. Thetis was an order of magnitude faster than Cplex-LP on the Bhoslib instances while generating solutions of comparable quality. Both Thetis and Cplex-LP recovered the optimal solution on some of the instances. On the SNAP instances, both Cplex-IP and Cplex-LP failed to complete within an hour on any of the instances. Cplex-IP was able to generate feasible solutions using its heuristics, but was able to unable to solve the root-node relaxation on any of the SNAP instances.

VC	Cplex IP			Cplex LP			Thetis		
(min)	t (secs)	BFS	Gap(%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	-	1475	0.7	2.48	767.0	1534	0.88	959.7	1532
frb59-26-2	-	1475	0.6	3.93	767.0	1534	0.86	979.7	1532
frb59-26-3	-	1475	0.5	4.42	767.0	1534	0.89	982.9	1533
frb59-26-4	-	1475	0.5	2.65	767.0	1534	0.89	983.6	1531
frb59-26-5	-	1475	0.5	2.68	767.0	1534	0.90	979.4	1532
Amazon	85.5	1.60×10^5	-	24.8	1.50×10^5	2.04×10^5	2.97	1.50×10^5	1.97×10^5
DBLP	22.1	1.65×10^5	-	22.3	1.42×10^5	2.08×10^5	2.70	1.42×10^5	2.06×10^5
Google+	-	1.06×10^5	0.01	40.1	1.00×10^5	1.31×10^5	4.47	1.00×10^5	1.27×10^5
MC	Cplex IP			Cplex LP			Thetis		
(min)	t (secs)	BFS	Gap(%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	72.3	346	-	312.2	346	346	5.86	352.3	349
frb59-26-2	561.1	254	-	302.9	254	254	5.82	262.3	254
frb59-26-3	27.7	367	-	311.6	367	367	5.86	387.7	367
frb59-26-4	65.4	265	-	317.1	265	265	5.80	275.7	265
frb59-26-5	553.9	377	-	319.2	377	377	5.88	381.0	377
Amazon	-	12	NA	-	-	-	55.8	7.3	5
DBLP	-	15	NA	-	-	-	63.8	11.7	5
Google+	-	6	NA	-	-	-	109.9	5.8	5
MIS	Cplex IP			Cplex LP			Thetis		
(max)	t (secs)	BFS	Gap(%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	-	50	18.0	4.65	767	15	0.88	447.7	18
frb59-26-2	-	50	18.0	4.74	767	17	0.88	448.6	17
frb59-26-3	-	52	13.4	3.48	767	19	0.87	409.2	19
frb59-26-4	-	53	11.3	4.41	767	18	0.90	437.2	17
frb59-26-5	-	51	15.6	4.43	767	18	0.88	437.0	18
Amazon	35.4	1.75×10^5	-	23.0	1.85×10^5	1.56×10^5	3.09	1.73×10^5	1.43×10^5
DBLP	17.3	1.52×10^5	-	23.2	1.75×10^5	1.41×10^5	2.72	1.66×10^5	1.34×10^5
Google+	-	1.06×10^5	0.02	44.5	1.11×10^5	9.39×10^4	4.37	1.00×10^5	8.67×10^4

Figure 5: Wall-clock time and quality of fractional and integral solutions for three graph analysis problems using Thetis, Cplex-IP and Cplex-LP. Each code was given a time limit of one hour, with '-' indicating a timeout. BFS is the objective value of the best integer feasible solution found by Cplex-IP. The gap is defined as (BFS−BB)/BFS where BB is the best known solution bound found by Cplex-IP within the time limit. A gap of '-' indicates that the problem was solved to within 0.01% accuracy and NA indicates that Cplex-IP was unable to find a valid solution bound. LP is the objective value of the LP solution, and RSol is objective value of the rounded solution.

VC	Cplex-IP			Cplex-LP (default tolerances)			Thetis		
(min)	t (secs)	BFS	Gap(%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	-	1475	0.7	4.59	767.0	1534	0.88	959.7	1532
frb59-26-2	-	1475	0.6	4.67	767.0	1534	0.86	979.7	1532
frb59-26-3	-	1475	0.5	4.76	767.0	1534	0.89	982.9	1533
frb59-26-4	-	1475	0.5	4.90	767.0	1534	0.89	983.6	1531
frb59-26-5	-	1475	0.5	4.72	767.0	1534	0.90	979.4	1532
Amazon	85.5	1.60×10^5	-	21.6	1.50×10^5	1.99×10^5	2.97	1.50×10^5	1.97×10^5
DBLP	22.1	1.65×10^5	-	23.7	1.42×10^5	2.07×10^5	2.70	1.42×10^5	2.06×10^5
Google+	-	1.06×10^5	0.01	60.0	1.00×10^5	1.30×10^5	4.47	1.00×10^5	1.27×10^5
MC	Cplex-IP			Cplex-LP (default tolerances)			Thetis ($\epsilon = 0.1$)		
(min)	t (secs)	BFS	Gap(%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	72.3	346	-	397.9	346	346	5.86	352.3	349
frb59-26-2	561.1	254	-	348.1	254	254	5.82	262.3	254
frb59-26-3	27.7	367	-	386.6	367	367	5.86	387.7	367
frb59-26-4	65.4	265	-	418.9	265	265	5.80	275.7	265
frb59-26-5	553.9	377	-	409.6	377	377	5.88	381.0	377
Amazon	-	12	NA	-	-	-	55.8	7.28	5
DBLP	-	15	NA	-	-	-	63.8	11.70	5
Google+	-	6	NA	-	-	-	109.9	5.84	5
MIS	Cplex-IP			Cplex-LP (default tolerances)			Thetis ($\epsilon = 0.1$)		
(max)	t (secs)	BFS	Gap(%)	t (secs)	LP	RSol	t (secs)	LP	RSol
frb59-26-1	-	50	18.0	4.88	767	16	0.88	447.7	18
frb59-26-2	-	50	18.0	4.82	767	16	0.88	448.6	17
frb59-26-3	-	52	13.4	4.85	767	16	0.87	409.2	19
frb59-26-4	-	53	11.3	4.67	767	15	0.90	437.2	17
frb59-26-5	-	51	16.6	4.82	767	16	0.88	437.0	18
Amazon	35.4	1.75×10^5	-	25.7	1.85×10^5	1.58×10^5	3.09	1.73×10^5	1.43×10^5
DBLP	17.3	1.52×10^5	-	24.0	1.75×10^5	1.41×10^5	2.72	1.66×10^5	1.34×10^5
Google+	-	1.06×10^5	0.02	68.8	1.11×10^5	9.40×10^4	4.37	1.00×10^5	8.67×10^4

Figure 6: Wall-clock time and quality of fractional and integral solutions for three graph analysis problems using Thetis, Cplex-IP and Cplex-LP (run to default tolerance). Each code was given a time limit of one hour, with '-' indicating a timeout. BFS is the objective value of the best integer feasible solution found by Cplex-IP. The gap is defined as (BFS$-$BB)/BFS where BB is the best known solution bound found by Cplex-IP within the time limit. A gap of '-' indicates that the problem was solved to within 0.01% accuracy and NA indicates that Cplex-IP was unable to find a valid solution bound. LP is the objective value of the LP solution, and RSol is objective value of the rounded solution.