Scalable Automated Proving of Information Theoretic Inequalities with Proximal Algorithms

Lin Ling, Chee Wei Tan, Siu-Wai Ho, and Raymond W. Yeung
linling2-c@my.cityu.edu.hk, cheewtan@cityu.edu.hk, siuwai.ho@adelaide.edu.au, whyeung@ie.cuhk.edu.hk

***Abstract*—Proving or disproving linear information theoretic inequalities is a fundamental task in information theory, and it has also been proved to be important in fields like cryptography and quantum communication theory. Manually proving information inequalities involving more than a few random variables can often be tedious or even intractable. In 1997, Yeung proposed a linear programming framework for verifying information inequalities, which was later extended to construct analytical proofs and disproofs. However, in practice this framework can be very slow for inequalities involving more than ten random variables, thus it is impossible to be applied to a wide range of practical problems. In this paper, we further extend this optimization-theoretic framework by reformulating the LPs and applying the Alternating Direction Method of Multipliers (ADMM) technique, where all the subproblems have closed-form solutions and thus can be solved efficiently. The proposed algorithm is also parallelizable so the performance can be further improved by running it on a GPU. An online web service is developed to allow users to prove or disprove their problem-specific inequalities without installing any software package or dependency.**

I. Introduction

The importance of computers as a mathematical tool for automated reasoning cannot be overstated as is evident from computer-assisted proofs to derive theorems (e.g., proving all theorems in Principia Mathematica [1]) or to validate well-known mathematical conjectures such as the four color theorem in graph theory by Heesch–Appel–Haken and the Kepler conjecture in geometry by Hales [2]. It is thus imperative to understand how computers can play a similar role of automated reasoning in the realm of information theory. In this paper, we present a theoretical and algorithmic framework to generate computer-aided proofs for information inequalities as well as demonstrating its scalable software implementation as a proof-of-concept.

Proving or disproving information inequalities is a fundamental task to prove converse theorems in information theory, and it also finds important applications in many other fields. A practical application of information inequalities can be found in [3, Example 14.8], where a bound in threshold secret sharing was proved. In [4], by proving an information inequality involving 16 random variables, C. Tian settled a conjecture for exact-repair regenerating codes. Recently, in [5], the authors showed that in database theory, the output size bound of a query can be proved by proving a corresponding information inequality.

In the following subsections, we give a brief review of the framework developed by the authors in [6], [7] and point out the scalability issue. This motivates a reformulation of the optimization-theoretic framework in [6] and [7], thus opening the door to new algorithmic developments.

A. The Linear Programming Framework

Any linear information inequality can be written in its canonical form, where all the information measures are represented as linear combinations of joint-entropies. For an inequality involving n random variables, there are $k = 2^n - 1$ joint-entropy terms, and naturally these terms can be viewed as the optimization variables in mathematical optimization. We group these terms into a column vector $\mathbf{h} \in \mathbb{R}^k$.

Any probability distribution of n random variables can be represented by a point in $\mathbb{R}^k$, but the converse is not true. Let $N = \{1, 2, \cdots, n\}$, one necessary condition is the non-negativity of the set of elemental inequalities, namely

$$H(X_i|X_{N-i}) \geq 0$$
$$I(X_i; X_j|X_K) \geq 0 \text{ where } i \neq j \text{ and } K \subset N - \{i, j\}$$

For an inequality involving n random variables, there are $m = n + \binom{n}{2} \times 2^{n-2}$ such elemental inequalities, which can be written in vector form as $\mathbf{Dh} \geq \mathbf{0}$, where $\mathbf{D} \in \mathbb{R}^{m \times k}$. In practice, we often need to prove information inequalities under some additional problem-specific constraints, which can be done by adding an additional constraint $\mathbf{Eh} = \mathbf{0}$. Thus, to verify an information inequality, we can construct $\mathbf{b}$, $\mathbf{D}$ and $\mathbf{E}$, and solve the following linear program (LP) and check the sign of the optimal objective value p^*.

$$\begin{aligned} \min \quad & p = \mathbf{b}^T\mathbf{h} \\ \text{s.t.} \quad & \mathbf{Dh} \geq \mathbf{0} \\ & \mathbf{Eh} = \mathbf{0} \\ \text{var.} \quad & \mathbf{h}. \end{aligned} \tag{LP-Primal}$$

Lin Ling and Chee Wei Tan are with the Department of Computer Science, City University of Hong Kong. Siu-Wai Ho is with the Teletraffic Research Centre, University of Adelaide, Australia. R. W. Yeung is with the Institute of Network Coding and the Department of Information Engineering, The Chinese University of Hong Kong, N.T., Hong Kong.

This work was supported in part by the Research Grants Council of Hong Kong RGC 11207615, in part by the Science, Technology, and Innovation Commission of Shenzhen Municipality under Project JCYJ20170818094955771, and in part by the National Natural Science Foundation of China under Grant 61771017.

If $p^* \geq 0$, we conclude that the inequality is true. If $p^* < 0$, the inequality is either not true, or the inequality is a non-Shannon-type inequality, which is yet to be completely characterized (see [6, V], [8] and [3, p. 361] for more information).

Now to actually construct the proof or disproof, we need to look at the dual of the problem

$$\begin{aligned} \max \quad & 0 \\ \text{s.t.} \quad & \mathbf{D}^T\mathbf{y} = \mathbf{b} + \mathbf{E}^T\boldsymbol{\mu} \\ & \mathbf{y} \geq \mathbf{0} \\ \text{var.} \quad & \mathbf{y}, \boldsymbol{\mu}, \end{aligned} \tag{LP-Dual}$$

which would be feasible if the inequality is true, and in this case we can obtain the optimal dual variables $\mathbf{y}^*$ and $\boldsymbol{\mu}^*$. Since they are dual feasible, we have

$$\mathbf{b} = \mathbf{D}^T\mathbf{y}^* - \mathbf{E}^T\boldsymbol{\mu}^*.$$

For any primal feasible $\mathbf{h}$, we have

$$\begin{aligned} \mathbf{b}^T\mathbf{h} &= \mathbf{y}^{*T}\mathbf{D}\mathbf{h} - \boldsymbol{\mu}^{*T}\mathbf{E}\mathbf{h} \\ &\geq 0, \end{aligned} \tag{1}$$

where the inequality follows as $\mathbf{y} \geq \mathbf{0}$, $\mathbf{D}\mathbf{h} \geq \mathbf{0}$ and $\mathbf{E}\mathbf{h} = \mathbf{0}$. Note that (1) can be viewed as a certificate of the non-negativity of $\mathbf{b}^T\mathbf{h}$ for any feasible $\mathbf{h}$, and an analytical proof can be constructed from it [1].

Example I.1. *To prove the inequality*

$$H(A, B) \geq I(A; B),$$

one can rewrite it to the canonical form

$$-H(A) - H(B) + 2H(A, B) = \mathbf{b}^T\mathbf{h} \geq 0,$$

where $\mathbf{b} = \begin{bmatrix} -1 & -1 & 2 \end{bmatrix}^T$ *and* $\mathbf{h} = \begin{bmatrix} H(A) & H(B) & H(A, B) \end{bmatrix}^T$.

The inequality involves 2 *random variables, so there are* $m = 3$ *elemental inequalities*

$$\begin{aligned} \mathbf{D}\mathbf{h} &= \begin{bmatrix} 0 & -1 & 1 \\ -1 & 0 & 1 \\ 1 & 1 & -1 \end{bmatrix} \mathbf{h} \\ &= \begin{bmatrix} -H(B) + H(A, B) \\ -H(A) + H(A, B) \\ H(A) + H(B) - H(A, B) \end{bmatrix} \\ &\geq \mathbf{0}. \end{aligned}$$

One possible optimal solution to (LP-Dual) *is* $\mathbf{y}^* = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^T$*, so according to* (1)*, we can construct the proof as*

$$\begin{aligned} & -H(A) - H(B) + 2H(A, B) \\ &= [-H(B) + H(A, B)] + [-H(A) + H(A, B)] \\ &\geq 0, \end{aligned}$$

where the inequality follows as both expressions in the brackets are rows in $\mathbf{D}\mathbf{h}$ *while* $\mathbf{D}\mathbf{h} \geq \mathbf{0}$.

If the input inequality is not true or if it is a non-Shannon-type inequality, (LP-Primal) would be unbounded and (LP-Dual) would be infeasible. To construct a disproof (assuming the inequality is Shannon-type), one would need to bound the primal variables $\mathbf{h}$ so they do not go to infinity. The easiest way to do so would be to bound $H(X_N)$, which is the last entry of $\mathbf{h}$. For example, we can solve the following LP

$$\begin{aligned} \min \quad & p = \mathbf{b}^T\mathbf{h} \\ \text{s.t.} \quad & \mathbf{D}\mathbf{h} \geq \mathbf{0} \\ & \mathbf{E}\mathbf{h} = \mathbf{0} \\ & \mathbf{e}^T\mathbf{h} = 1 \\ \text{var.} \quad & \mathbf{h}, \end{aligned} \tag{LP-Disproof-Primal}$$

where $\mathbf{e} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix}^T$, and similar to the construction of proofs, its dual also gives us hints to construct a disproof. The disproof construction is not in the scope of this paper, and interested readers are referred to [7, IV] for the details.

B. The Scalability Issue

From (1), we see that the number of elemental inequalities used in the proof or disproof equals the cardinality of the optimal dual variables used. For both proving and disproving, the dual LP has infinitely many solutions, thus it is important to find a sparse dual solution to ensure that the proof/disproof is as concise and elegant as possible.

In [7] as well as in software implementations like [9], [10], the simplex method is used to solve the LPs. It is well known that the simplex method always gives vertex solutions, which are solutions at the vertex of the feasible region, where only a small subset of constraints are active, thus the dual solutions obtained by the simplex method are in fact often sparse. However, in practice, the performance of these simplex-based software implementations deteriorates for relatively large values of n ($n \geq 10$), due to the following reasons:

1) The worst-case running time of the simplex method is exponential, and the constraint matrix $\mathbf{D}$ grows exponentially with n.
2) The $\mathbf{D}$ matrix is very sparse, but the simplex method takes no advantage of the sparsity of the constraint matrix.
3) The LPs are highly degenerate (with infinitely many optimal dual solutions), and the simplex method is usually quite inefficient in solving such problems.

Note that when applied to some practical problems, the LPs in the linear programming framework can exhibit symmetric structures, allowing us to reduce the size of $\mathbf{D}$ dramatically. In [4], C. Tian formulated an LP which can be viewed as (LP-Primal) with 16 random variables, plus a few extra constraints. The problem has 65535 variables and 1966112 constraints, which is intractable even for commercial LP

[1] The idea of constructing analytical proofs using the dual solutions was discovered simultaneously in [7] and [4].

solvers. Fortunately, by exploiting the symmetric structure introduced by the extra constraints, many constraints and variables can be shown to be redundant and thus can be removed from the LP, leaving only 76 variables and 6152 constraints. Similar techniques were also used in [11], [12]. However, when proving general inequalities, the size of the LPs cannot be further reduced, as the minimality of $\mathbf{D}$ has been proved in [3, p.353].

The performance issue mentioned above prevents the LP framework in [6], [7] from being readily applied to large scale problems, thus a more scalable and efficient extension to the framework is needed. In Section II, we reformulate the LPs and introduce an ADMM-based algorithm to solve them efficiently in parallel. In Section IV, we introduced the software implementation we developed as a web service to allow users to submit and solve their problem-specific information inequalities without installing any software package or dependency.

II. Parallel Computation with ADMM

A. Generic ADMM algorithm

Consider the optimization problem

$$\begin{array}{ll} \min & f(\mathbf{x})+g(\mathbf{y}) \\ \text{s.t.} & \mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}=\mathbf{c} \\ \text{var.} & \mathbf{x},\mathbf{y}, \end{array} \tag{2}$$

with the ρ-augmented Lagrangian

$$L_\rho = f(\mathbf{x})+g(\mathbf{y})+\boldsymbol{\lambda}^T(\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}-\mathbf{c})+\frac{\rho}{2}||\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}-\mathbf{c}||^2,$$

where ρ is a hyper-parameter that we can choose [2]. A generic ADMM algorithm is given in Algorithm 1.

ALGORITHM 1: Generic ADMM Algorithm

repeat

1. $\mathbf{x}$-update: $\mathbf{x}^{k+1} = \arg\min\{L_\rho(\mathbf{x},\mathbf{y}^k,\boldsymbol{\lambda}^k)\}$
2. $\mathbf{y}$-update: $\mathbf{y}^{k+1} = \arg\min\{L_\rho(\mathbf{x}^{k+1},\mathbf{y},\boldsymbol{\lambda}^k)\}$
3. $\boldsymbol{\lambda}$-update: $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho(\mathbf{A}\mathbf{x}^{k+1}+\mathbf{B}\mathbf{y}^{k+1}-\mathbf{c})$

until *Convergence*;

The ADMM algorithm and its convergence analysis have been introduced in great details in [13] and [14, 4.4].

B. Problem Reformulation

Consider the following LP and its dual

$$\begin{array}{ll} \min & \mathbf{b}^T\mathbf{h} \\ \text{s.t.} & \mathbf{0}\le\mathbf{D}\mathbf{h}\le\mathbf{1} \\ & \mathbf{E}\mathbf{h}=\mathbf{0} \\ \text{var.} & \mathbf{h}, \end{array} \tag{P-Merge}$$

[2]In our software implementation and the online web service in Session III and IV, we use $\rho = 2$.

$$\begin{array}{ll} \max & -\mathbf{1}^T\boldsymbol{\lambda}_2 \\ \text{s.t.} & \mathbf{b}-\mathbf{D}^T\boldsymbol{\lambda}_1+\mathbf{D}^T\boldsymbol{\lambda}_2+\mathbf{E}^T\boldsymbol{\mu}=\mathbf{0} \\ & \boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2\ge\mathbf{0} \\ \text{var.} & \boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2,\boldsymbol{\mu}. \end{array} \tag{D-Merge}$$

Theorem 1. *A set of optimal solutions to* (D-Merge)*, $\{\boldsymbol{\lambda}_1^*,\boldsymbol{\lambda}_2^*,\boldsymbol{\mu}^*\}$, can be used to construct a proof or disproof for the corresponding Shannon-type information inequality.*

Proof. When the inequality is true, it is easy to see that in both (LP-Primal) and (P-Merge), we have $\mathbf{h}^* = \mathbf{0}$ and $p^* = 0$. Thus the extra constraint in (P-Merge), $\mathbf{D}\mathbf{h}\le\mathbf{1}$, is redundant. Redundant constraint has no effect in both primal and dual, so we conclude that, when the inequality is indeed true, (P-Merge) and (D-Merge) are equivalent to (LP-Primal) and (LP-Dual).

If the inequality is not true, (P-Merge) and (D-Merge) are not equivalent to (LP-Primal) and (LP-Dual), but the solutions $\{\boldsymbol{\lambda}_1^*,\boldsymbol{\lambda}_2^*,\boldsymbol{\mu}^*\}$ can still be used to construct a disproof. Specifically, assuming that the input inequality is not true, if there exists a valid entropy vector $\tilde{\mathbf{h}}$ satisfying

$$\boldsymbol{\lambda}_1^{*T}\mathbf{D}\tilde{\mathbf{h}} = \boldsymbol{\lambda}_2^{*T}(\mathbf{D}\tilde{\mathbf{h}}-\mathbf{1}) = \boldsymbol{\mu}^{*T}\mathbf{E}\tilde{\mathbf{h}} = \mathbf{0}, \tag{3}$$

using (3) together with the stationarity of Lagrangian

$$\mathbf{b}-\mathbf{D}^T\lambda_1+\mathbf{D}^T\lambda_2+\mathbf{E}^T\boldsymbol{\mu}=\mathbf{0},$$

we can show that $\tilde{\mathbf{h}}$ can be used as a counter-example to disprove the inequality, as

$$\begin{aligned} \mathbf{b}^T\tilde{\mathbf{h}} &= \mathbf{b}^T\tilde{\mathbf{h}} - \boldsymbol{\lambda}_1^{*T}\mathbf{D}\tilde{\mathbf{h}} + \boldsymbol{\mu}^{*T}\mathbf{E}\tilde{\mathbf{h}} \\ &= -\mathbf{D}^T\boldsymbol{\lambda}_2^* \\ &= -\mathbf{1}^T\boldsymbol{\lambda}_2^*. \end{aligned}$$

Since we assumed that the inequality is not true, we know that the optimal objective value of (D-Merge), $-\mathbf{1}^T\boldsymbol{\lambda}_2^*$, should be strictly negative, thus $\mathbf{b}^T\tilde{\mathbf{h}} = -\mathbf{1}^T\boldsymbol{\lambda}_2^* < 0$. Therefore, such $\tilde{\mathbf{h}}$ can be used as a counter-example to disprove the inequality. □

Note that the optimal solution to (P-Merge), $\mathbf{h}^*$, also satisfies (3), but we cannot use it directly as a counter-example, as it might not have a corresponding distribution of r.v.s (see the discussion of non-Shannon-type inequalities in I-A). Instead, we can only provide hints that the counter-example $\tilde{\mathbf{h}}$ should satisfy (3), and leave the job of finding the exact $\tilde{\mathbf{h}}$ to the users. This is also what the authors of [7] do (see [7, IV]).

Now consider the following LP and its dual

$$\begin{array}{ll} \min & \mathbf{b}^T\mathbf{h} \\ \text{s.t.} & \mathbf{D}\mathbf{h}-\mathbf{u}=\mathbf{0} \\ & \mathbf{D}\mathbf{h}+\mathbf{v}=\mathbf{1} \\ & \mathbf{E}\mathbf{h}=\mathbf{0} \\ & \mathbf{u},\mathbf{v}\ge\mathbf{0} \\ \text{var.} & \mathbf{h},\mathbf{u},\mathbf{v}, \end{array} \tag{P-ADMM}$$

$$\begin{aligned} \max \quad & -\mathbf{1}^T\boldsymbol{\nu}_2 \\ \text{s.t.} \quad & \mathbf{b}+\mathbf{D}^T\boldsymbol{\nu}_1+\mathbf{D}^T\boldsymbol{\nu}_2+\mathbf{E}^T\boldsymbol{\mu}=\mathbf{0} \\ & \boldsymbol{\nu}_1\le\mathbf{0},\boldsymbol{\nu}_2\ge\mathbf{0}, \\ \text{var.} \quad & \boldsymbol{\nu}_1,\boldsymbol{\nu}_2,\boldsymbol{\mu}, \end{aligned} \tag{D-ADMM}$$

where (P-ADMM) is obtained by introducing slack variables $\mathbf{u}$ and $\mathbf{v}$ to (P-Merge). Comparing (D-ADMM) and (D-Merge), it is easy to see that if we have a set of optimal solutions to (D-ADMM), $\{\boldsymbol{\nu}_1^*,\ \boldsymbol{\nu}_2^*\}$, we also obtain the optimal solutions to (D-Merge) as $\{-\boldsymbol{\nu}_1^*,\ \boldsymbol{\nu}_2^*\}$ "for free". Therefore, solving (P-ADMM) and (D-ADMM) is enough to prove or disprove any given Shannon-type information inequality.

Finally, we rewrite (P-ADMM) to the following form so that it is more compact and easier to work with

$$\begin{aligned} \min \quad & \mathbf{b}^T\mathbf{h} \\ \text{s.t.} \quad & \mathbf{B}\mathbf{h}+\mathbf{y}=\mathbf{c} \\ \text{var.} \quad & \mathbf{h},\mathbf{y}, \end{aligned} \tag{P-ADMM}$$

where $\mathbf{B}=\begin{bmatrix}\mathbf{D}\\ \mathbf{D}\\ \mathbf{E}\end{bmatrix}$, $\mathbf{y}=\begin{bmatrix}-\mathbf{u}\\ \mathbf{v}\\ \mathbf{0}\end{bmatrix}$, $\mathbf{u},\mathbf{v}\ge 0$ and $\mathbf{c}=\begin{bmatrix}\mathbf{0}\\ \mathbf{1}\\ \mathbf{0}\end{bmatrix}$.

C. ADMM-Based Algorithm

The ρ-augmented Lagrangian of (P-ADMM) is

$$L_\rho=\mathbf{b}^T\mathbf{h}+\boldsymbol{\nu}^T(\mathbf{B}\mathbf{h}+\mathbf{y}-\mathbf{c})+\frac{\rho}{2}||\mathbf{B}\mathbf{h}+\mathbf{y}-\mathbf{c}||^2,$$

where $\boldsymbol{\nu}$ is the Lagrangian multiplier of the constraint $\mathbf{B}\mathbf{h}+\mathbf{y}=\mathbf{c}$. With ADMM, we can solve (P-ADMM) using Algorithm 2.

ALGORITHM 2: ITIP Algorithm with ADMM

repeat

1. $\mathbf{h}$-update: $\mathbf{h}^{k+1}=\arg\min\{L_\rho(\mathbf{h},\mathbf{y}^k,\boldsymbol{\nu}^k)\}$
2. $\mathbf{y}$-update: $\mathbf{y}^{k+1}=\arg\min\{L_\rho(\mathbf{h}^{k+1},\mathbf{y},\boldsymbol{\nu}^k)\}$
3. $\boldsymbol{\nu}$-update: $\boldsymbol{\nu}^{k+1}=\boldsymbol{\nu}^k+\rho(\mathbf{B}\mathbf{h}^{k+1}+\mathbf{y}^{k+1}-\mathbf{c})$

until *Convergence*;

In [13, Appendix A], the authors proved that ADMM converges under the following two assumptions:

1) the objective functions f and g as in Algorithm 1 are closed proper convex.

2) the unaugmented Lagrangian L_0 has a saddle point.

Assumption 1) is obviously true since the objective function is linear in our case. Since our problem is LP, assumption 2) is equivalent to saying that both the primal and the dual problem have an optimal solution, which is also the case. Therefore Algorithm 2 does converge.

The $\mathbf{h}$-update is an unconstrained QP, so we can further get the following closed-form expression:

$$\mathbf{h}^{k+1}=-\frac{1}{\rho}(\mathbf{B}^T\mathbf{B})^{-1}(\mathbf{b}+\mathbf{B}^T\boldsymbol{\nu}^k+\rho\mathbf{B}^T\mathbf{y}^k-\rho\mathbf{B}^T\mathbf{c}). \tag{$\mathbf{h}$-update}$$

Note that in ($\mathbf{h}$-update) we use the inversion of $\mathbf{B}^T\mathbf{B}$ only for showing the closed-form solution. In practice inversing a large matrix is both inefficient and numerically unstable. Instead, we apply the Cholesky factorization on $\mathbf{B}^T\mathbf{B}$ to get a lower-triangular matrix $\mathbf{L}$ such that $\mathbf{B}^T\mathbf{B}=\mathbf{L}\mathbf{L}^T$, and directly solve the linear system via back substitution.

Since $\mathbf{y}=\begin{bmatrix}-\mathbf{u}\\ \mathbf{v}\end{bmatrix}$, the $\mathbf{y}$-update can be decomposed into $\mathbf{u}$-update and $\mathbf{v}$-update.

The $\mathbf{u}$-update is a constrained QP, but luckily the KKT system can be directly solved, giving us the closed-form solution

$$\mathbf{u}^{k+1}=(\mathbf{D}\mathbf{h}^{k+1}+\frac{1}{\rho}\boldsymbol{\nu}_u^k)_+, \tag{$\mathbf{u}$-update}$$

where $\boldsymbol{\nu}_u^k$ is the upper half of $\boldsymbol{\nu}^k$. Similarly, for $\mathbf{v}$-update, we have

$$\mathbf{v}^{k+1}=(\mathbf{1}-\mathbf{D}\mathbf{h}^{k+1}-\frac{1}{\rho}\boldsymbol{\nu}_v^k)_+. \tag{$\mathbf{v}$-update}$$

We can see that every subproblem in our ADMM algorithm has a closed-form solution, thus it can be solved very efficiently. Notice that ADMM is a numerical algorithm, so the $\boldsymbol{\nu}^*$ we get might not be sparse. After its convergence, we need to do "crossover", which is essentially taking the numerical primal and dual solutions and supply them to a simplex solver as the starting point to "hot-start" a simplex procedure, which is much faster than solving the LP from scratch [15]. This "crossover" feature is in fact built-in in most commercial LP solvers.

D. Efficient Matrix Factorization in $\mathbf{h}$-update

The bottleneck of Algorithm 2 is in the $\mathbf{h}$-update, where at each iteration we need to factorize a large matrix ($\mathbf{B}^T\mathbf{B}\in\mathbb{R}^{k\times k}$). As $\mathbf{B}$ is fixed throughout the algorithm, an obvious approach is to factorize the matrix at the beginning of the algorithm, and directly use the factorization in later iterations. However, this is still slow as Cholesky factorization generally has a time complexity of $O(n^3)$. In this section, we develop an efficient method to compute the factorization.

Recall that $\mathbf{B}=\begin{bmatrix}\mathbf{D}\\ \mathbf{D}\\ \mathbf{E}\end{bmatrix}$, where the only problem-dependent component is $\mathbf{E}$. In other words, different input inequalities can share the same $\mathbf{B}$ matrix, if they involve the same number of random variables n, and they do not contain problem-specific constraints ($\mathbf{E}$ does not exist). Therefore, we can pre-factorize the $\mathbf{B}^T\mathbf{B}$ matrix for different n values and cache the factorization on disk. If the input inequality has no problem-specific constraint, the factorization can be loaded directly from disk to speed up the computation.

If the input inequality does contain user-constraints, one can still obtain the corresponding Cholesky factorization from the cache. Let $\mathbf{e}_i^T$ be the i-th row of the matrix $\mathbf{E}$, where $\mathbf{E}\in\mathbb{R}^{l\times k}$, and let $\tilde{\mathbf{B}}$ be $\begin{bmatrix}\mathbf{D}\\ \mathbf{D}\end{bmatrix}$, we have

$$\mathbf{B}^T\mathbf{B} = \tilde{\mathbf{B}^T}\tilde{\mathbf{B}} + \sum_{i=1}^{l} \mathbf{e}_i\mathbf{e}_i^T,$$

i.e., $\mathbf{B}^T\mathbf{B}$ can be constructed from $\tilde{\mathbf{B}^T}\tilde{\mathbf{B}}$ via a series of rank-1 updates. Since we have the cache of factorization of $\tilde{\mathbf{B}^T}\tilde{\mathbf{B}}$, the factorization of $\mathbf{B}^T\mathbf{B}$ can be efficiently obtained with complexity $O(n^2l)$, as discussed in [16]. In practice, the number of problem-specific constraints is usually small, thus $l \ll k$, and therefore this method is much faster than running the factorization from scratch.

III. Numerical Experiment

The proposed Algorithm 2 is implemented in C++. Gurobi [17], a state-of-the-art commercial LP solver, is used in the crossover step after the convergence of ADMM. The original simplex-based algorithm is also implemented as a baseline using Gurobi.

Since all the sub-problems in Algorithm 2 have closed-form solutions, the algorithm is essentially a series of (sparse) linear algebra operations, which can be easily parallelized to boost the performance. To fully unleash the power of this algorithm, it is also implemented using the CUDA toolkits (cuSPARSE and cuBLAS) to be executed on the GPU [18].

The average running time over 10 randomly generated inequalities for the original simplex-based algorithm, our proposed ADMM-based algorithm on CPU and GPU, as well as crossover are listed in the Table I below. The numbers outside of parentheses are the average running time in seconds, and the numbers in parentheses are the number of instances (out of 10) the algorithm failed to solve within the 3600 seconds time limit. All the CPU computations are done on a Linux machine with two 6-core Intel® Xeon® CPUs, and the GPU computations are done on a machine with one Nvidia® V100 GPU installed.

TABLE I
Experiment results

n	Original	ADMM (CPU)	ADMM (GPU)	Crossover
10	1.88 (0)	1.99 (0)	0.30 (0)	1.22 (0)
11	20.08 (0)	11.94 (0)	0.91 (0)	5.44 (0)
12	269.32 (0)	49.13 (1)	8.06 (0)	35.09 (0)
13	3409.55 (7)	231.81 (4)	11.81 (0)	248.46 (0)
14	(10)	522.82 (5)	54.06 (0)	1430.12 (0)

From Table I, we can see that for large n values ($n \geq 11$), CPU ADMM with crossover runs faster and is more reliable than the original simplex-based method. With access to a GPU, GPU ADMM with crossover is significantly faster and more robust than any other methods.

IV. Conclusions

As a proof-of-concept, the proposed algorithm has been implemented as a web service available at `https://itip.algebragame.app`. With the service, users can easily prove their information inequalities, without the need of installing software packages on their machines and having access to powerful computers. The web service can also be useful for educators and students of information theory, providing them with a quick way to verify inequalities with human-readable analytical proofs.

In this work we propose a new reformulation of the linear programming framework developed in [6], [7] and propose a scalable algorithmic framework based on ADMM. Unlike previous works [4], [11], [12] where structures of specific problems are exploited to reduce the problem size, our work is more general in the sense that it aims directly at the scalability and computational aspect of proving or disproving any Shannon-type information inequality, allowing it to be used on a much wider range of problems.

References

[1] H. Wang, "Computer theorem proving and artificial intelligence," *Automated Theorem Proving: After 25 years, American Mathematical Society, Contemporary Mathematics*, vol. 29, pp. 49–70, 1984.

[2] T. C. Hales, "Linear programs for the Kepler conjecture," in *International Congress on Mathematical Software*. Springer, 2010, pp. 149–151.

[3] R. W. Yeung, *Information theory and network coding*. Springer Science & Business Media, 2008.

[4] C. Tian, "Characterizing the rate region of the (4, 3, 3) exact-repair regenerating codes," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 967–975, 2014.

[5] M. Abo Khamis, H. Q. Ngo, and D. Suciu, "What do Shannon-type inequalities, submodular width, and disjunctive datalog have to do with one another?" in *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. ACM, 2017, pp. 429–444.

[6] R. W. Yeung, "A framework for linear information inequalities," *IEEE Transactions on Information Theory*, vol. 43, no. 6, pp. 1924–1934, 1997.

[7] S.-W. Ho, C. W. Tan, and R. W. Yeung, "Proving and disproving information inequalities," in *2014 IEEE International Symposium on Information Theory (ISIT)*. Citeseer, 2014, pp. 2814–2818.

[8] Z. Zhang and R. W. Yeung, "A non-Shannon-type conditional inequality of information quantities," *IEEE Transactions on Information Theory*, vol. 43, no. 6, pp. 1982–1986, 1997.

[9] R. W. Yeung and Y. O. Yan, *Information Theoretic Inequality Prover (ITIP), Matlab program software package available at:* `http://home.ie.cuhk.edu.hk/~ITIP`, 1999.

[10] R. Pulikkoonattu, E. Perron, and S. Diggavi, *XITIP, Cross-platform C software package available at:* `http://xitip.epfl.ch/`, 2008.

[11] S. Thakor, T. Chan, and A. Grant, "A minimal set of Shannon-type inequalities for functional dependence structures," in *2017 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2017, pp. 679–683.

[12] S. Thakor, A. Grant, and T. Chan, "On complexity reduction of the LP bound computation and related problems," in *2011 International Symposium on Network Coding (NetCod)*. IEEE, 2011, pp. 1–6.

[13] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[14] N. Parikh, S. Boyd *et al.*, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[15] R. E. Bixby and M. J. Saltzman, "Recovering an optimal LP basis from an interior point solution," *Operations Research Letters*, vol. 15, no. 4, pp. 169–178, 1994.

[16] R. Van Der Merwe and E. A. Wan, "The square-root unscented Kalman filter for state and parameter-estimation," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing Proceedings.(ICASSP'01)*, vol. 6. IEEE, 2001, pp. 3461–3464.

[17] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2018. [Online]. Available: http://www.gurobi.com

[18] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with CUDA," in *ACM SIGGRAPH 2008 classes*. ACM, 2008, p. 16.