Algorithms for Context-Aided Variable Elimination

Inigo Incer
Albert Benveniste
Richard M. Murray
Alberto L. Sangiovanni-Vincentelli
Sanjit A. Seshia

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2023-15
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2023/EECS-2023-15.html

January 28, 2023

Algorithms for Context-Aided Variable Elimination

Inigo Incer[1,3], Albert Benveniste[2], Richard M. Murray[3], Alberto Sangiovanni-Vincentelli[1], and Sanjit A. Seshia[1]

[1] University of California, Berkeley, USA
[2] INRIA/IRISA, Rennes, France
[3] California Institute of Technology, USA

Deriving system-level specifications from component specifications usually involves the elimination of variables that are not part of the interface of the top-level system. This paper presents algorithms for eliminating variables from formulas by computing refinements or abstractions of these formulas in a context. We discuss a connection between this problem and optimization and give efficient algorithms to compute refinements and abstractions of linear inequality constraints.

1 Introduction

In the setting of formal system design using assume-guarantee specifications [2,4,6], we come across the need to eliminate variables from a formula by computing refinements or abstractions in a context. Let ϕ be a formula containing some variables that must be eliminated. These will be called *irrelevant variables*, and the set of such variables will be denoted Y. In order to carry out the elimination, suppose we can use information from a set of formulas Γ called the *context*. We will consider the problems of synthesizing missing formulas in the expressions

$$\Gamma \land \ ? \ \models \phi \quad \text{and} \quad \Gamma \land \phi \models \ ?$$

such that the result lacks forbidden variables. We will call the first problem *antecedent synthesis*, and the second *consequent synthesis*. If ψ is a solution to the antecedent synthesis problem, we will say that ψ is a refinement (or an antecedent) of ϕ in the context Γ. If ψ is a solution to the consequent synthesis problem, we will say that ψ is an abstraction (or a consequent) of ϕ in the context Γ. Before we formalize the problem, we consider two examples.

Figure 1 shows two components connected in series, M_1 and M_2. The first has input i and output o, and the second has input o and output o'. Each component comes with its assumptions and guarantees. The nature of M_1 and M_2 is left abstract; they could be routines executing in order, or they could be physical systems that interact through their input and output ports. Our problem is to obtain a specification for the entire system using the specifications of the subsystems in such a way that only the top-level input and output variables

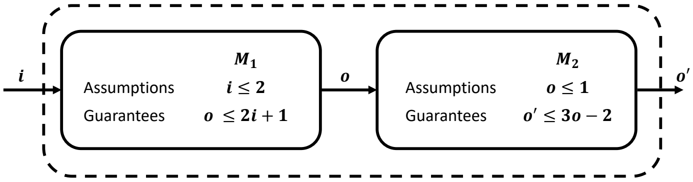

Fig. 1: Two components connected in series. We wish to compute the specification of the top-level system composed of these two elements.

i and o' appear in the final answer. In other words, the top-level specification should not mention the internal variable o.

We would like to operate the system in such a way that the assumptions of the two components hold. This would mean that we can rely on the two subsystems to deliver their guarantees. Thus, the top-level system should assume $(i \leq 2) \wedge (o \leq 1)$. This cannot be the top-level specification because the second formula involves the *irrelevant variable* o. We would like to find a term only depending on i that somehow ensures that the assumptions $o \leq 1$ of M_2 are satisfied. For this, we make use of the knowledge that M_1 guarantees $o \leq 2i + 1$ when $i \leq 2$. We want to transform the constraint $o \leq 1$ into a constraint ψ on the input i with the property that, given the guarantees of M_1, ψ implies $o \leq 1$. That is, this new constraint should satisfy $\psi \wedge (o \leq 2i + 1) \rightarrow (o \leq 1)$, which means that ψ should be a refinement (an antecedent) of $o \leq 1$ in the context of the guarantees of M_1. We observe that $\psi\colon i \leq 0$ satisfies this requirement. Thus, we transform the term $o \leq 1$ into the term $i \leq 0$. The top-level assumptions become $i \leq 0$. We can verify that these top-level assumptions ensure that subsystems M_1 and M_2 have their assumptions met.

Similarly, the guarantees for the system are $(o \leq 2i + 1) \wedge (o' \leq 3o - 2)$. Again, the variable o is not welcome in the final answer, giving us two options: we could eliminate both terms and have no guarantees—which is right, but not useful—or we could relax (compute the consequent of) one of the terms in the context of the other term. We find out, for example, that $(o \leq 2i + 1) \wedge (o' \leq 3o - 2) \rightarrow (o' \leq 6i + 1)$. The constraint $o' \leq 6i + 1$ is an acceptable promise for the system specification.

By computing antecedents and consequents, we concluded that the top-level system guarantees $o' \leq 6i + 1$ as long as the input satisfies $i \leq 0$.

This example shows that the computation of antecedents and consequents plays a key role in the identification of pre/post conditions. One may be tempted to link antecedents to assumptions and consequents to guarantees. This is not always so. Figure 2 show a situation in which we again have two components connected in series, M_1 and M_2, with inputs and outputs as before. Now we are given the top level assumptions and guarantees, and we also know the assump-

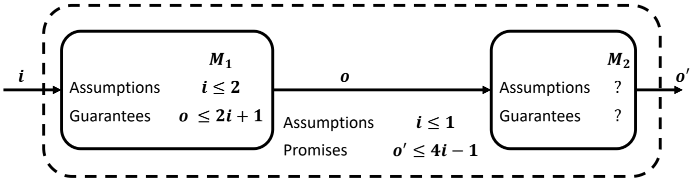

Fig. 2: Two components connected in series. We are given the specification of the top-level system and the specification of M_1. The problem is to find the pre/post conditions of M_2 in order to obtain the given system-level specifications.

tions and guarantees of M_1. The problem is to find the pre/post conditions of M_2 using this data.

To start with, we know that the top level assumes that $i \leq 1$. Under these assumptions, M_1 guarantees $o \leq 2i + 1$. The assumptions of M_2 should be met when the top-level system is operating within its assumptions. Thus, the assumptions of M_2 should be implied by the data $(i \leq 1) \wedge (o \leq 2i + 1)$. Since the assumptions of M_2 should only depend on o, we obtain the expression $o \leq 3$.

Now we look for the guarantees of M_2, which we call ψ. The guarantees of M_1 and M_2 together must imply the top-level guarantees. Thus, we have the expression $\psi \wedge (o \leq 2i + 1) \to (o' \leq 4i - 1)$. In other words, ψ is an antecedent of $o \leq 2i + 1$ in the context $o' \leq 4i - 1$. We require ψ to only refer to variables o and o' and observe that $o' \leq 2o - 3$ is an acceptable promise.

We conclude that M_2 should assume $o \leq 3$ and promise $o' \leq 2o - 3$.

The examples just described motivate us to study automated mechanisms for the computation of antecedents and consequents of formulas in a given context with the objective of removing dependencies on irrelevant variables. We first consider this problem in the setting where the model of the theory is a complete partial order. We treat antecedents and consequents in a unified manner and formulate both notions as optimization problems. We then specialize our considerations to formulas expressed as linear constraints in a context of linear inequalities and provide efficient algorithms to address this problem. Our previous discussion shows that this problem is of relevance to formal system design.

2 Computing antecedents and consequents in partial orders

In this section, we consider the computation of antecedents and consequents of atomic formulas from a first-order theory in a structure endowed with a partial order. We establish a link between this problem and optimization. Our attention will be on atomic formulas of the form $f(x, g(y)) \leq K$, where y is an array of irrelevant variables and f is monotonic in the second argument. In other words, we assume that the irrelevant variables can be separated from the rest of the variables in the formula. We use a context Γ to bound $g(y)$ using an expression that does not depend on y. This allows us to eliminate irrelevant variables from the original formula. We show that this framework supports formula refinement/abstraction for propositional logic and linear inequalities.

Our description of formal languages borrows notation from [1]. Let $(\mathcal{F}, \mathcal{R})$ be the signature of the first order language L and let V be a set of variables. That is, let $\mathcal{F}$ and $\mathcal{R}$ be sets of function and relation symbols of various arities. Let $\mathcal{M} = (\mathcal{D}_\mathcal{M}, \mathcal{F}^\mathcal{M}, \mathcal{R}^\mathcal{M})$ be a structure in which we interpret L. Here $\mathcal{D}_\mathcal{M}$ is a set called the domain. For each function symbol $f \in \mathcal{F}$ of arity n, there is $f^\mathcal{M} \in \mathcal{F}^\mathcal{M}$ such that $f^\mathcal{M} \colon \mathcal{D}_\mathcal{M}^n \to \mathcal{D}_\mathcal{M}$; and for each m-ary relation symbol $R \in \mathcal{R}$, we have $R^\mathcal{M} \in \mathcal{R}^\mathcal{M}$ such that $R^\mathcal{M} \subseteq \mathcal{D}_\mathcal{M}^m$.

We will assume that $\mathcal{D}_\mathcal{M}$ is a bounded, complete partial order and that $\mathcal{R}$ contains a binary relation, denoted $\leq$, whose interpretation $\leq_\mathcal{M}$ in $\mathcal{M}$ is the partial order relation of $\mathcal{D}_\mathcal{M}$.

Let $X = \{x_i\}_{i=1}^m \subseteq V$ and $Y = \{y_i\}_{i=1}^n \subseteq V$ be disjoint sets of variables, where the elements of Y will be called *irrelevant variables*. Suppose $K \in \mathcal{F}$ is a 0-ary (a constant) function symbol, f is an $(m+1)$-ary term generated by $\mathcal{F}$ and $X \cup Y$, and g is an n-ary term generated by $\mathcal{F}$ and Y. Moreover, suppose that the function $f^\mathcal{M}$ is monotonic in its last argument. This is the problem we want to solve:

Given an atomic formula ϕ of the form

$$\phi\colon \quad f(x_1, \ldots, x_m, g(y_1, \ldots, y_n)) \leq K, \tag{1}$$

a set of formulas Γ, and a structure $\mathcal{M}$, synthesize an antecedent or consequent of the atomic formula ϕ in the context Γ when the formulas are interpreted in $\mathcal{M}$. The resulting formulas cannot contain irrelevant variables.

We assume that the context Γ contains, in addition to irrelevant variables and X variables, a set of variables $Z = \{z_i\}_{i=1}^o$ disjoint from X and Y. From now on, we will keep the following notation: m denotes the number of variables that do not need to be eliminated from ϕ, n the number of irrelevant variables in ϕ, o the number of Z variables in Γ, and N the number of formulas in Γ.

We define functions $g_+^\mathcal{M}, g_-^\mathcal{M} \colon \mathcal{D}_\mathcal{M}^{m+o} \to \mathcal{D}_\mathcal{M}$ as follows:

$$g_+^\mathcal{M}(a_1, \ldots, a_m, c_1, \ldots, c_o) = \begin{cases} \underset{b_1, \ldots, b_n \in \mathcal{D}_\mathcal{M}}{\text{maximize}} & g^\mathcal{M}(b_1, \ldots, b_n) \\ \text{subject to} & \mathcal{M}, [x := a, y := b, z := c] \models \Gamma \end{cases} \tag{2}$$

and

$$g_-^\mathcal{M}(a_1, \ldots, a_m, c_1, \ldots, c_o) = \begin{cases} \underset{b_1, \ldots, b_n \in \mathcal{D}_\mathcal{M}}{\text{minimize}} & g^\mathcal{M}(b_1, \ldots, b_n) \\ \text{subject to} & \mathcal{M}, [x := a, y := b, z := c] \models \Gamma, \end{cases} \tag{3}$$

where the notation $\mathcal{M}, [x := a, y := b, z := c]$ means that the formula is satisfied in the structure $\mathcal{M}$ after substituting the variables x_i by a_i, y_i by b_i, and z_i by c_i. $g_+^{\mathcal{M}}$ and $g_-^{\mathcal{M}}$ are well defined because $\mathcal{D}_{\mathcal{M}}$ is complete and bounded.

We assume that the term algebra generated by $\mathcal{F}$ and $X \cup Z$ contains $(m+o)$-ary terms g_+ and g_- whose interpretations in $\mathcal{M}$ are $g_+^{\mathcal{M}}$ and $g_-^{\mathcal{M}}$, respectively. We have the following result:

Theorem 1. *Let ϕ' and ϕ'' be*

$$\phi' \colon \quad f(x_1, \ldots, x_m, g_+(x_1, \ldots, x_m, z_1, \ldots, z_o)) \leq K \tag{4}$$

and

$$\phi'' \colon \quad f(x_1, \ldots, x_m, g_-(x_1, \ldots, x_m, z_1, \ldots, z_o)) \leq K. \tag{5}$$

These formulas satisfy $\mathcal{M} \models (\Gamma \wedge \phi') \Rightarrow \phi$ and $\mathcal{M} \models (\Gamma \wedge \phi) \Rightarrow \phi''$.

Proof. Let $a_i, b_j, c_k \in \mathcal{D}_{\mathcal{M}}$ for $i \leq m$, $j \leq n$, and $k \leq o$. First we will show that

$$\mathcal{M} \models \Gamma \Rightarrow (g(y_1, \ldots y_n) \leq g_+(x_1, \ldots, x_m, z_1, \ldots z_o)). \tag{6}$$

If $\mathcal{M}, [x := a, y := b, z := c] \not\models \Gamma$, the result holds vacuously. If $\mathcal{M}, [x := a, y := b, z := c] \models \Gamma$, then from (2) we have

$$g^{\mathcal{M}}(b_1, \ldots, b_n) \leq_{\mathcal{M}} g_+^{\mathcal{M}}(a_1, \ldots, a_m, c_1, \ldots, c_o),$$

showing that (6) holds.

Suppose $\mathcal{M}, [x := a, y := b, z := c] \models (\Gamma \wedge \phi')$. We have from (6)

$$\mathcal{M}, [x := a, y := b, z := c] \models (g(y_1, \ldots y_n) \leq g_+(x_1, \ldots, x_m, z_1, \ldots z_o)).$$

It follows that

$$\mathcal{M}, [x := a, y := b, z := c] \models \phi' \wedge (g(y_1, \ldots y_n) \leq g_+(x_1, \ldots, x_m, z_1, \ldots z_o)),$$

which means that

$$\begin{aligned}&(f^{\mathcal{M}}(a_1, \ldots, a_m, g_+^{\mathcal{M}}(a_1, \ldots, a_m, c_1, \ldots, c_o)) \leq_{\mathcal{M}} K^{\mathcal{M}}) \wedge \\ &(g^{\mathcal{M}}(b_1, \ldots b_n) \leq_{\mathcal{M}} g_+^{\mathcal{M}}(a_1, \ldots, a_m, c_1, \ldots c_o)).\end{aligned}$$

Since $f^{\mathcal{M}}$ is monotonic in the last argument, we obtain

$$f^{\mathcal{M}}(a_1, \ldots, a_m, g^{\mathcal{M}}(b_1, \ldots b_n)) \leq_{\mathcal{M}} K^{\mathcal{M}}.$$

Thus, $\mathcal{M}, [x := a, y := b, z := c] \models \phi$, proving the first part of the theorem.

We will prove that

$$\mathcal{M} \models \Gamma \Rightarrow (g_-(x_1, \ldots, x_m, z_1, \ldots z_o) \leq g(y_1, \ldots y_n)). \tag{7}$$

Suppose $\mathcal{M}, [x := a, y := b, z := c] \models \Gamma$, then from (3) we have

$$g_-^{\mathcal{M}}(a_1, \ldots, a_m, c_1, \ldots, c_o) \leq_{\mathcal{M}} g^{\mathcal{M}}(b_1, \ldots, b_n),$$

showing that (7) holds.

Suppose $\mathcal{M}, [x := a, y := b, z := c] \models (\Gamma \wedge \phi)$. We have from (7)

$$\mathcal{M}, [x := a, y := b, z := c] \models (g_-(x_1, \dots, x_m, z_1, \dots z_o) \leq g(y_1, \dots y_n)).$$

It follows that

$$\mathcal{M}, [x := a, y := b, z := c] \models \phi \wedge (g_-(x_1, \dots, x_m, z_1, \dots z_o) \leq g(y_1, \dots y_n)),$$

which means that

$$\begin{gathered}(f^{\mathcal{M}}(a_1, \dots, a_m, g^{\mathcal{M}}(b_1, \dots b_n)) \leq_{\mathcal{M}} K^{\mathcal{M}}) \wedge \\ (g_-^{\mathcal{M}}(a_1, \dots, a_m, c_1, \dots c_o) \leq_{\mathcal{M}} g^{\mathcal{M}}(b_1, \dots b_n)).\end{gathered}$$

Since $f^{\mathcal{M}}$ is monotonic in the last argument, we conclude that

$$f^{\mathcal{M}}(a_1, \dots, a_m, g_-^{\mathcal{M}}(a_1, \dots, a_m, c_1, \dots, c_o)) \leq_{\mathcal{M}} K^{\mathcal{M}}.$$

Thus, $\mathcal{M}, [x := a, y := b, z := c] \models \phi''$, proving the second part. □

Theorem 1 gives us antecedents and consequents of atomic formulas ϕ of the form (1) such that the result lacks irrelevant variables.

Example 1. Suppose $(\mathcal{F}, \mathcal{R})$ is the signature of a propositional language and $\mathcal{R}$ only contains a binary relation $\leq$. We will interpret this language in the model $\mathcal{M}$ with domain $\{0, 1\}$, where we will assume that $0 \leq_{\mathcal{M}} 1$. Suppose we want to compute antecedents and consequents of the formula

$$\phi\colon (p \wedge q) \vee r,$$

in the context $\Gamma\colon s \Rightarrow q$, where q is an irrelevant variable. We can apply Theorem 1 as follows: We observe that $\Rightarrow$ and $\leq$ have the same semantics in $\mathcal{M}$. Thus, we can write ϕ as $\phi\colon (\neg p \vee \neg q) \leq r$. Let $g(q) = \neg q$ and $f(p, x) = \neg p \vee x$. Then $f^{\mathcal{M}}$ is monotonic in its last argument. To apply Theorem 1, we compute $g_+^{\mathcal{M}}$:

$$g_+^{\mathcal{M}}(a, c) = \begin{cases} \underset{b \in \{0,1\}}{\text{maximize}} & \neg q \\ \text{subject to} & \mathcal{M}, [p := a, q := b, s := c] \models s \leq q \end{cases} = \neg c.$$

Thus, $g_+(p, s) = \neg s$. By Theorem 1, we conclude that $f(p, g_+(p, s)) \leq r$ is an antecedent of ϕ in the given context, i.e., we compute

$$\neg p \vee \neg s \leq r,$$

which is equivalent to $(p \wedge s) \vee r$. ■

3 Linear inequality constraints

Now we apply the results of Section 2 to the situation when formulas are expressed as linear inequalities. In this section, we interpret formulas in only one structure, allowing us to relax the distinction between formulas and their interpretations. Consider the formula ϕ given by

$$\phi\colon \quad \sum_{i=1}^{m} p_i x_i + \sum_{i=1}^{n} q_i y_i \le r, \tag{8}$$

where r and the p_i and q_i are constant symbols. $X = \{x_i\}_{i=1}^{m}$ and $Y = \{y_i\}_{i=1}^{n}$ are sets of variables. Y is the set of irrelevant variables. We also have a context $\mathit{\Gamma}$, which is a set of linear inequalities of the form

$$\mathit{\Gamma} = \left\{ \sum_{j=1}^{m} \alpha_{ij} x_i + \sum_{j=1}^{n} \beta_{ij} y_i + \sum_{j=1}^{o} \gamma_{ij} z_i \le K_i \right\}_{i=1}^{N}, \tag{9}$$

where the K_j, α_i^j, β_i^j, and γ_i^j are constant symbols, and $Z = \{z_i\}_{i=1}^{o}$ is a set of variables disjoint from X and Y.

We interpret these formulas in the extended real line $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$, which is a complete, bounded partial order. In this setting, the term f is given by $f(x_1, \dots, x_n, w) = \sum_{i=1}^{n} \alpha_i x_i + w$, the interpretation of which is clearly monotonic in the last argument. Therefore, we can eliminate irrelevant variables from ϕ by using Theorem 1.

After interpreting the formulas, the constant symbols become real numbers. Let $A = (\alpha_{ij}) \in \mathbb{R}^{N \times m}$, $B = (\beta_{ij}) \in \mathbb{R}^{N \times n}$, $C = (\gamma_{ij}) \in \mathbb{R}^{N \times o}$, $K \in \mathbb{R}^{N}$, $p \in \mathbb{R}^{m}$, and $q \in \mathbb{R}^{n}$. We also let $x = (x_i)$, $y = (y_i)$, $z = (z_i)$ be m-, n-, and o-dimensional vectors of variables, respectively.

Our problem is to compute antecedents/consequents of

$$\phi\colon \quad p^{\mathsf{T}} x + q^{\mathsf{T}} y \le r$$

in the context

$$\mathit{\Gamma}\colon \quad Ax + By + Cz \le K$$

such that the result lacks irrelevant variables.

Let $b(x, z) = K - Ax - Cz$. We obtain the following corollary from Theorem 1.

Corollary 1. *Let ϕ and $\mathit{\Gamma}$ be as above. Let*

$$g_{+}(x, z) = \begin{cases} \underset{y \in \mathbb{R}^n}{\mathit{maximize}} & q^{\mathsf{T}} y \\ \mathit{subject\ to} & By \le b(x, z) \end{cases} \tag{10}$$

and

$$g_-(x,z) = \begin{cases} \mathit{minimize}_{y \in \mathbb{R}^n} & q^\intercal y \\ \mathit{subject\ to} & By \leq b(x,z). \end{cases} \tag{11}$$

Then the formula $p^\intercal x + g_+(x,z) \leq r$ is an antecedent of ϕ in the context Γ and $p^\intercal x + g_-(x,z) \leq r$ is a consequent from ϕ in the context Γ.

Example 2. Suppose we wish to eliminate variables y_1 and y_2 from $2x + y_1 - 2y_2 \leq 5$ through antecedent computation, using the context $\{x - 2y_1 + y_2 + z \leq 1, 3y_1 - 4y_2 \leq 6\}$. We compute

$$g_+(x,z) = \begin{cases} \text{maximize}_{y_1,y_2 \in \mathbb{R}} & y_1 - 2y_2 \\ \text{subject to} & x - 2y_1 + y_2 + z \leq 1 \\ & 3y_1 - 4y_2 \leq 6 \end{cases} = 4 - \frac{2}{5}(x+z).$$

The antecedent formula is $2x + 4 - \frac{2}{5}(x+z) \leq 5$, which becomes $8x - 2z \leq 5$. ■

Example 3. Suppose we wish to eliminate variables y_1 and y_2 from $x + 5y_1 - 2y_2 \leq 5$ by the computation of a consequent, using the context $\{x - 2y_1 + y_2 + z \leq 1, 3y_1 - 4y_2 \leq 6\}$. We compute

$$g_-(x,z) = \begin{cases} \text{minimize}_{y_1,y_2 \in \mathbb{R}} & 5y_1 - 2y_2 \\ \text{subject to} & x - 2y_1 + y_2 + z \leq 1 \\ & 3y_1 - 4y_2 \leq 6 \end{cases} = -4 + \frac{14}{5}(x+z).$$

The consequent is $x - 4 + \frac{14}{5}(x+z) \leq 5$, or $19x + 14z \leq 45$. ■

3.1 Solving the symbolic optimization problems

The next issue we face is the computation of (10) and (11). Both are linear programs, but their solutions are symbolic due to the presence of $b(x,z)$. We observe that if we have a context Γ' such that $\Gamma = \Gamma' \wedge \Gamma''$, and if ϕ' is an antecedent/consequent of ϕ in the context Γ' then ϕ' is also an antecedent/consequent in the context Γ. First, we will discuss conditions required for solving linear programs with symbolic constraints when the context has as many constraints as optimization variables (i.e., when $N = n$). Then we will discuss approaches for selecting from Γ a set of formulas that meets these requirements. We consider two selection criteria: a method based on positive solutions to linear equations and a method based on linear programming.

Optimization in a subset of the context. A linear program achieves its optimal value on the boundary of its constraints. If the context Γ contains N constraints and is a bounded polyhedron, then the optimal value of the linear program will occur at one of the $\binom{N}{n}$ possible vertices. We will look for ways to choose n constraints from Γ such that the optimization problems achieve optimal values at the vertex determined by those n constraints. First, we focus on solving symbolic LPs when the context contains n constraints. The following definition will be useful:

Definition 1. *Let $M \in \mathbb{R}^{n\times n}$ and $\nu \in \mathbb{R}^n$. We say that (M,ν) is a refining pair if M is invertible and $(M^\intercal)^{-1}\nu$ has nonnegative entries. We say that the pair (M,ν) is an abstractive pair if M is invertible and $-(M^\intercal)^{-1}\nu$ has nonnegative entries.*

As the next result shows, these conditions are sufficient to solve the problems (10) and (11) when there are as many context formulas as irrelevant variables (i.e., when $N = n$). Suppose $J \subseteq \{1,\ldots,N\}$ has cardinality n. We let $B_J = (\beta_{J_i,j})_{i,j=1}^n$ and $b_J = (b_{J_i})_{i=1}^n$ be the J-indexed rows of B and b, respectively.

Lemma 1. *Suppose (B_J, q) is a refining pair. Then*

$$\begin{cases} \textit{maximize}\limits_{y\in\mathbb{R}^n} & q^\intercal y \\ \textit{subject to} & B_J y \le b_J(x,z) \end{cases} = q^\intercal B_J^{-1} b(x,z).$$

Suppose (B_J, q) is an abstractive pair. Then

$$\begin{cases} \textit{minimize}\limits_{y\in\mathbb{R}^n} & q^\intercal y \\ \textit{subject to} & B_J y \le b_J(x,z) \end{cases} = q^\intercal B_J^{-1} b_J(x,z).$$

Proof. Let (B_J, q) be a refining pair. We consider the first problem and its Lagrange dual (see [3], Section 5.2.1):

$$\text{primal}\begin{cases} \text{minimize}\limits_{y} & -q^\intercal y \\ \text{subject to} & B_J y \le b_J(x,z) \end{cases} \qquad \text{dual}\begin{cases} \text{maximize}\limits_{\lambda} & -b_J^\intercal \lambda \\ \text{subject to} & B_J^\intercal \lambda - q = 0 \\ & \lambda \ge 0 \end{cases}$$

The dual problem only admits the solution $\lambda^\star = (B_J^\intercal)^{-1}q$ if $\lambda^\star \ge 0$, which is the case, as (B_J, q) is a refining pair. Thus, the optimal value of the dual problem is $v^\star = -q^\intercal(B_J^{-1}b_J)$. As strong duality holds for any linear program (see [3], Section 5.2.4), $v^\star$ is also the optimal value of the primal problem. The statement of the theorem follows.

Now suppose (B_J, q) is an abstractive pair. We consider the second problem and its dual:

$$\text{primal}\begin{cases} \text{minimize}\limits_{y} & q^\intercal y \\ \text{subject to} & B_J y \le b_J(x,z) \end{cases} \qquad \text{dual}\begin{cases} \text{maximize}\limits_{\lambda} & -b_J^\intercal \lambda \\ \text{subject to} & B_J^\intercal \lambda + q = 0 \\ & \lambda \ge 0 \end{cases}$$

The dual only admits the solution $\lambda^\star = -(B_J^\intercal)^{-1}q$ if $\lambda^\star \ge 0$, which is the case because (B_J, q) is an abstractive pair. The optimal value of the dual problem is $v^\star = q^\intercal(B_J^{-1}b)$. Due to strong duality, $v^\star$ is also the optimal value of the primal problem. □

As a consequence of Corollary 1 and Lemma 1, we obtain the following

Corollary 2. *With all definitions as above, if (B_J, q) is a refining pair, then $p^\intercal x + q^\intercal B_J^{-1} b_J(x,z) \leq r$ is an antecedent of $p^\intercal x + q^\intercal y \leq r$ in the context $By \leq b(x,z)$. If (B_J, q) is an abstractive pair, then $p^\intercal x + q^\intercal B_J^{-1} b_J(x,z) \leq r$ is a consequent of $p^\intercal x + q^\intercal y \leq r$ in the context $By \leq b(x,z)$.*

Corollary 2 gives explicit formulas for computing antecedents/consequents of a formula in a context. This result is missing methods for computing J, the set of the indices of formulas in Γ, in such a way that it yields refining or abstractive pairs (B_J, q), as needed. We consider two methods to identify J.

Computing J by seeking positive solutions to linear equations. Our first method is based on identifying constraints yielding linear systems of equations whose solutions are guaranteed to be nonnegative. We will use the following result.

Theorem 2 (Kaykobad [5]). *Let $M = (\mu_{ij}) \in \mathbb{R}^{n \times n}$ and $\nu \in \mathbb{R}^n$. Suppose the entries of M are nonnegative, its diagonal entries are positive, the entries of ν are positive, and $\nu_i > \sum_{j \neq i} \mu_{ij} \frac{\nu_j}{\mu_{jj}}$ for all $i \leq n$. Then M is invertible and $M^{-1}\nu$ has positive entries.*

A pair (M, ν) satisfying the conditions of Theorem 2 we will call a *Kaykobad pair*. We have the following result.

Lemma 2. *Let Q be an $n \times n$ diagonal matrix whose i-th entry is $\textsc{sign}(q_i)$. Let $\bar{B}_J = B_J Q$ and $\bar{q} = Qq$. If $(\bar{B}_J^\intercal, \bar{q})$ is a Kaykobad pair, then (B_J, q) is a refining pair. If $(-\bar{B}_J^\intercal, \bar{q})$ is a Kaykobad pair, then (B_J, q) is an abstractive pair.*

Proof. Suppose $(\bar{B}_J^\intercal, \bar{q})$ is a Kaykobad pair. Then $\bar{B}_J^\intercal$ is invertible. We have $B_J(\bar{B}_J Q)^{-1} = B_J Q (\bar{B}_J)^{-1} = I$ and $(\bar{B}_J Q)^{-1} B_J = Q(\bar{B}_J)^{-1}(B_J Q)Q = I$, so B_J is invertible. Moreover, we have

$$0 < (\bar{B}_J^\intercal)^{-1}\bar{q} = (QB_J^\intercal)^{-1}(Qq) = (B_J^\intercal)^{-1}q,$$

which means that (B_J, q) is a refining pair.

If $(-\bar{B}_J^\intercal, \bar{q})$ is a Kaykobad pair, then $\bar{B}_J^\intercal$ is invertible, which means that so is B_J. Moreover,

$$0 < -(\bar{B}_J^\intercal)^{-1}\bar{q} = -(QB_J^\intercal)^{-1}(Qq) = -(B_J^\intercal)^{-1}q.$$

Thus, (B_J, q) is an abstractive pair. □

Corollary 2 and Lemma 2 yield a method for computing antecedents and consequents of formulas lacking irrelevant variables. To use it, we must construct J such that (B_J, q) meets the corollary's conditions. We construct J by choosing n formulas from the context Γ; these formulas must meet the conditions of a Kaykobad pair. One advantage of the Kaykobad condition is that it allows

Algorithm 1 Antecedents and consequents for linear inequality constraints by identifying systems of equations with positive solutions

```
**Input:** Term to transform $p^\intercal x + q^\intercal y \leq r$, context $\varGamma$,
        transform instruction $s$ (**true** for antecedents and **false** for consequents)
**Output:** Transformed term $t'$ lacking any $y$ variables
MatrixRowTerms $\leftarrow \emptyset$                    ▷ Rows of the context matrix $A$
PartialSums $\leftarrow \textsc{zeros}(\textsc{length}(y))$
TCoeff $\leftarrow -1$
**if** s **then**
    TCoeff $\leftarrow 1$
**for** $i = 1$ to $i = \textsc{length}(y)$ **do**          ▷ One iteration per row of context matrix
    IthRowFound $\leftarrow$ **false**          ▷ Indicate whether we could add the $i$-th row
    **for** $\gamma \in \varGamma \setminus$ MatrixRowTerms **do**
        ▷ 1. Verifying Kaykobad pair: sign of nonzero matrix terms
        TermIsInvalid $\leftarrow$ **false**
        **for** $j = 1$ to $j = \textsc{length}(y)$ **do**
            **if** $\textsc{coeff}(\gamma, y_j) \neq 0$ and $\textsc{sign}(\textsc{coeff}(\gamma, y_j)) \neq \textsc{sign}(q_j) \cdot$ TCoeff **then**
                TermIsInvalid $\leftarrow$ **true**
                **break**
        ▷ 2. Verifying Kaykobad pair: matrix diagonal terms
        **if** $\textsc{coeff}(\gamma, y_i) = 0$ or TermIsInvalid **then**
            **next**
        ▷ 3. Verifying Kaykobad pair: relationship between matrix and vector entries
        Residuals $\leftarrow$ zeros($\textsc{length}(y)$)
        **for** $j = 1$ to $j = \textsc{length}(y)$ **do**
            **if** $j \neq i$ **then**
                Residuals[$j$] $\leftarrow \textsc{sign}(q_j) \cdot$ TCoeff $\cdot\, \textsc{coeff}(\gamma, y_j) \cdot \frac{q_i}{\textsc{coeff}(\gamma, y_i)}$
            **if** $|q_j| \cdot$ TCoeff $\leq$ PartialSums[$j$] + Residuals[$j$] **then**
                TermIsInvalid $\leftarrow$ **true**
                **break**
        **if not** TermIsInvalid **then**
            ▷ Resulting matrix is meeting Kaykobad pair conditions at $i$-th row
            IthRowFound $\leftarrow$ **true**
            **for** $j = 1$ to $j = \textsc{length}(y)$ **do**
                PartialSums[$j$] $\leftarrow$ PartialSums[$j$] + Residuals[$j$]
            MatrixRowTerms.append($\gamma$)
            **break**
    **if not** IthRowFound **then**
        **return** Error: Cannot transform term
$B \leftarrow \textsc{MatrixFromTerms}(\text{MatrixRowTerms}, y)$
$b \leftarrow \textsc{VectorFromTerms}(\text{MatrixRowTerms}, y)$
**return** $p^\intercal x + q^\intercal B^{-1} b \leq r$
```

us to incrementally identify suitable constraints to add to the context Γ', i.e., we don't have to select n constraints before we run the verification. That is, when we have identified $k < n$ constraints, we can easily verify whether a candidate $(k+1)$-th formula would be acceptable for constructing a Kaykobad pair. Algorithm 1 computes antecedents and consequents for linear inequality constraints based on Corollary 2. Lines 6–30 search the context Γ for n constraints meeting the Kaykobad conditions. The rest of the algorithm computes the antecedents/consequents. If there are n variables to be eliminated, and $N = |\Gamma|$ constraints in the context Γ, the algorithm has complexity $O(n^2 N + N^3)$. The function $\textsc{coeff}(\gamma, y_j)$ extracts the coefficient of the variable y_j from the term γ. The call $\textsc{MatrixFromTerms}(\text{MatrixRowTerms}, y)$ extracts all coefficients of the y variables contained in MatrixRowTerms and makes these coefficients the rows of the resulting matrix. The call $\textsc{VectorFromTerms}(\text{MatrixRowTerms}, y)$ returns a vector of all expressions contained in MatrixRowTerms with their y variables removed. These are the elements of $b(x, z)$. Finally, $\textsc{diag}(v)$ returns a diagonal matrix whose entries are the vector v.

Computing *J* via linear programming. Now we will build J by numerically solving (10) and (11) for fixed values of x and z.

Lemma 3. *Let $a \in \mathbb{R}^m$ and $c \in \mathbb{R}^o$.*

- *Suppose $g_+(a, c)$ is finite and the optimum of the LP* (10) *(with $x = a$ and $z = c$) is attained at $y^\star$. Let $J = \left\{ i \;\middle|\; b_i(a, c) - \sum_{j=1}^n \beta_{ij} y_j^\star = 0 \right\}$ and assume that $|J| = n$, where n is the number of optimization variables y in g_+. If B_J is invertible, then (B_J, q) is a refining pair.*
- *Similarly, suppose $g_-(a, c)$ is finite and the optimum of the LP* (11) *(with $x = a$ and $z = c$) is attained at $y^\star$. Let $J = \left\{ i \;\middle|\; b_i(a, c) - \sum_{j=1}^n \beta_{ij} y_j^\star = 0 \right\}$ and assume that $|J| = n$. If B_J is invertible, then (B_J, q) is an abstractive pair.*

Proof. We prove the first part. Consider the following problems:

$$\text{(P)} \begin{cases} \underset{y}{\text{minimize}} & -q^\intercal y \\ \text{subject to} & By \le b(a, c) \end{cases} \qquad \text{(D)} \begin{cases} \underset{\lambda}{\text{maximize}} & -b(a, c)^\intercal \lambda \\ \text{subject to} & B^\intercal \lambda - q = 0 \\ & \lambda \ge 0 \end{cases}$$

Since $g_+(a, c)$ is finite, the primal is feasible. By strong duality, so is the dual. Let $\lambda^\star$ be the value of λ where the dual attains its optimum. Then $\lambda^\star \ge 0$ and $0 = B^\intercal \lambda^\star - q = B_J^\intercal \lambda_J^\star + B_{\hat{J}}^\intercal \lambda_{\hat{J}}^\star - q$, where $\hat{J} = \{1, \ldots, N\} \setminus J$. Due to complementary slackness, we know that $\lambda_{\hat{J}}^\star = 0$. Thus, $0 = B_J^\intercal \lambda_J^\star - q$. By assumption, B_J is invertible. Then (B_J, q) is a refining pair. The proof of the second part is similar. □

Lemma 2 allows us to obtain the solution to a linear programming problem with symbolic constraints $By \le b(x, z)$ in a reduced context $B_J y \le b_J(x, z)$,

Algorithm 2 Antecedents and consequents for linear inequality constraints through linear programming

```
Input: Term to transform $p^\intercal x + q^\intercal y \leq r$, context $\Gamma$, $a \in \mathbb{R}^m$, $c \in \mathbb{R}^o$,
       transform instruction $s$ (true for antecedents and false for consequents)
Output: Transformed term $t'$ lacking any $y$ variables
$B \leftarrow$ MatrixFromTerms($\Gamma, y$)
$b \leftarrow$ VectorFromTerms($\Gamma, y$)
$b_e \leftarrow$ Evaluate($b, a, c$)
if s then
    (success, $y^\star$) $\leftarrow$ LinearProgramming($-q, B, b_e$)
else
    (success, $y^\star$) $\leftarrow$ LinearProgramming($q, B, b_e$)
if not success then
    return Error: LP obtained after evaluation at $(a, c)$ is unfeasible
$S \leftarrow b_e - By^\star$
$J \leftarrow \emptyset$
for $j = 1$ to $j =$ length($b$) do
    if $S_j = 0$ then
        $J \leftarrow J \cup \{j\}$
(success, $\hat{B}_J$) $\leftarrow$ MatrixInv($B_J$)
if not success then
    return Error: cannot invert $B_J$
return $p^\intercal x + q^\intercal \hat{B}_J b_J \leq r$
```

where we identify J by solving a numerical LP. Lemma 2 and Corollary 2 yield a method for computing antecedents and consequents. This method is reflected in Algorithm 2. As before, MatrixFromTerms(Γ, y) and VectorFromTerms(Γ, y) extract from the context Γ the matrix B and symbolic vector $b(x, z)$ of the constraints $By \leq b(x, z)$. Evaluate(b, a, c) returns the vector $b(a, c) \in \mathbb{R}^N$. LinearProgramming(q, B, b_e) solves the LP $\underset{y}{\text{min.}}\ q^\intercal y$ subject to $By \leq b_e$ and returns a success variable and the value $y^\star$ where the minimum is attained. The success variable is true when the LP is feasible and has a finite solution. MatrixInv computes matrix inverses. Its success variable is false when the matrix is not invertible.

4 Conclusions

We considered the problem of eliminating variables from a formula by computing refinements and abstractions in a context. First we treated the problem in the setting of a partial order. Then the results were applied to linear inequality constraints. Progress in two areas would extend the reach of the techniques obtained for linear inequalities.

- Methods to efficiently select a set of linear equations such that their solution is nonnegative.

- Solving linear programming problems with symbolic constraints.

The main theorem we presented can be directly extended to handle nonlinear constraints and modal logic.

Acknowledgements

This work was supported by the DARPA LOGiCS project under contract FA8750-20-C-0156 and by NSF and ASEE through an eFellows postdoctoral fellowship. Its contents are solely the responsibility of the authors and do not necessarily represent the views of the sponsors.

References

1. Barbenchon, P., Pinchinat, S., and Schwarzentruber, F. *Logique : Fondements et Applications.* Dunod, 2022.
2. Benveniste, A., Caillaud, B., Nickovic, D., Passerone, R., Raclet, J.-B., Reinkemeier, P., Sangiovanni-Vincentelli, A., Damm, W., Henzinger, T. A., and Larsen, K. G. Contracts for system design. *Foundations and Trends® in Electronic Design Automation 12*, 2-3 (2018), 124–400.
3. Boyd, S., Boyd, S. P., and Vandenberghe, L. *Convex Optimization.* Cambridge university press, 2004.
4. Incer, I. *The Algebra of Contracts.* PhD thesis, EECS Department, University of California, Berkeley, May 2022.
5. Kaykobad, M. Positive solutions of positive linear systems. *Linear Algebra and its Applications 64* (1985), 133–140.
6. Sangiovanni-Vincentelli, A. L., Damm, W., and Passerone, R. Taming Dr. Frankenstein: Contract-based design for cyber-physical systems. *Eur. J. Control 18*, 3 (2012), 217–238.