Linear Algebraic Characterization of Logic Programs*

Chiaki Sakama[1], Katsumi Inoue[2], and Taisuke Sato[3]

[1] Wakayama University, Japan
sakama@sys.wakayama-u.ac.jp
[2] National Institute of Informatics, Japan
inoue@nii.ac.jp
[3] AI research center AIST, Japan
satou.taisuke@aist.go.jp

Abstract. This paper introduces a novel approach for computing logic programming semantics based on multilinear algebra. First, a propositional Herbrand base is represented in a vector space and if-then rules in a program are encoded in a matrix. Then we provide methods of computing the least model of a Horn logic program, minimal models of a disjunctive logic program, and stable models of a normal logic program by algebraic manipulation of higher-order tensors. The result of this paper exploits a new connection between linear algebraic computation and symbolic computation, which has potential to realize logical inference in huge scale of knowledge bases.

1 Introduction

Logic programming (LP) provides languages for declarative problem solving and symbolic reasoning, while proof-theoretic computation like Prolog turns inefficient in real-world applications. Recent studies have developed efficient solvers for *answer set programming* (ASP)—LP under the stable model semantics [1]. In this paper, we take a different approach and introduce a new method of computing LP semantics in vector spaces. There are several reasons for considering linear algebraic computation of LP. First, linear algebra is at the core of many applications of scientific computation, and integrating linear algebraic computation and symbolic computation is considered a challenging topic in AI [13]. Second, linear algebraic computation has potential to cope with Web scale symbolic data, and several studies develop scalable techniques to process huge relational knowledge bases [10, 11, 18]. Since relational KBs consist of ground atoms, the next challenge is applying linear algebraic techniques to LP and deductive DBs. Third, it would enable us to use efficient (parallel) algorithms of numerical linear algebra for computing LP. Moreover, matrix/tensor factorization techniques would be useful for approximation and optimization in LP.

Several studies attempt to realize logical reasoning using linear algebra. Grefenstette [5] introduces tensor-based predicate calculus in which elements of tensors represent truth values of domain objects and logical operations are realized by third-order tensor contractions. Yang, *et al.* [18] introduce a method of mining Horn clauses from relational facts represented in a vector space. Serafini and Garcez [16] introduce logic tensor networks that integrate logical deductive reasoning and data-driven relational

* This work is supported by NII Collaborative Research Program.

learning. Sato [14] formalizes Tarskian semantics of first-order logic in vector spaces, and shows how tensorization realizes efficient computation of Datalog [15]. These studies realize linear algebraic computation of predicate calculus over unary/binary relations on a finite domain, while they do not target computing LP semantics. There are studies that compute LP semantics in other computational paradigms (e.g. [8]). To the best of our knowledge, however, there is no study that realizes LP using linear algebra.

In this paper, we develop a theory of LP based on multilinear algebra. First, a propositional Herbrand base is represented in a vector space and if-then rules in a program are encoded in a matrix. Then the least model of a (propositional) Horn logic program is computed using matrix products. Next disjunctive logic programs are represented in third-order tensors and their minimal models are computed by algebraic manipulation of tensors. Normal logic programs are also represented by third-order tensors in terms of disjunctive logic programs, and their stable models are computed using tensor products. The rest of this paper is organized as follows. Section 2 reviews notions in multilinear algebra. Section 3 formulates LP semantics in vector spaces. Section 4 discusses related issues and Section 5 concludes the paper.

2 Preliminaries

This section reviews basic notions of tensors used in this paper. The following definitions are from [6]. An *N-th order tensor* is an element of the tensor product of N vector spaces. A first-order tensor is a vector $\boldsymbol{v} \in \mathbb{R}^I$, a second-order tensor is a matrix $\boldsymbol{M} \in \mathbb{R}^{I \times J}$, and a third-order tensor is a three-way array $\boldsymbol{T} \in \mathbb{R}^{I \times J \times K}$. The i-th element of a vector is denoted by a_i, an element (i, j) of a matrix is denoted by a_{ij}, and an element (i, j, k) of a third-order tensor is denoted by a_{ijk}. A tensor or matrix is often written as $\boldsymbol{T} = (a_{ijk})$ or $\boldsymbol{M} = (a_{ij})$, respectively. A *column vector* is an $m \times 1$ matrix which is represented as $(a_1, \ldots, a_m)^\mathsf{T}$ $(m \geq 1)$ where T is the transpose operation. In this paper, a vector means a column vector unless stated otherwise. *Slices* are two-dimensional sections of a tensor, defined by fixing all but two indices. A third-order tensor $\boldsymbol{T} \in \mathbb{R}^{I \times J \times K}$ is decomposed into (frontal) slices by fixing an index k $(1 \leq k \leq K)$. The k-th slice of a third-order tensor $\boldsymbol{T}$ is a matrix and is written as $\boldsymbol{T}_{::k}$. Let $\boldsymbol{T}_{::1}, \ldots, \boldsymbol{T}_{::K}$ be the slices of a third-order tensor $\boldsymbol{T} \in \mathbb{R}^{I \times J \times K}$ such that

$$\begin{pmatrix} a_{111} & \cdots & a_{1J1} \\ \vdots & \ddots & \vdots \\ a_{I11} & \cdots & a_{IJ1} \end{pmatrix} \cdots \begin{pmatrix} a_{11K} & \cdots & a_{1JK} \\ \vdots & \ddots & \vdots \\ a_{I1K} & \cdots & a_{IJK} \end{pmatrix}$$

Then $\boldsymbol{T}$ is *flattened* into the $I \times (J \times K)$ matrix as

$$\begin{pmatrix} a_{111} & \cdots & a_{1J1} & & a_{11K} & \cdots & a_{1JK} \\ \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ a_{I11} & \cdots & a_{IJ1} & & a_{I1K} & \cdots & a_{IJK} \end{pmatrix}$$

To distinguish slices in a matrix, we often introduce a vertical line between blocks representing slices $\boldsymbol{T}_{::k}$ and $\boldsymbol{T}_{::k+1}$.

Example 2.1 Consider a third-order tensor $\boldsymbol{T} \in \mathbb{R}^{2 \times 3 \times 2}$ which is decomposed into two slices

$$\begin{pmatrix} a_1 & b_1 & c_1 \\ d_1 & e_1 & f_1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} a_2 & b_2 & c_2 \\ d_2 & e_2 & f_2 \end{pmatrix}$$

Then $\boldsymbol{T}$ is flattened into the 2×6 matrix as

$$\left(\begin{array}{ccc|ccc} a_1 & b_1 & c_1 & a_2 & b_2 & c_2 \\ d_1 & e_1 & f_1 & d_2 & e_2 & f_2 \end{array}\right)$$

The (2-*mode*) *product* of a tensor $\boldsymbol{T} \in \mathbb{R}^{I \times J \times K}$ with a vector $\boldsymbol{v} \in \mathbb{R}^J$ is denoted by $\boldsymbol{T} \bullet \boldsymbol{v}$. The result is the $I \times K$ matrix that has the element $(\boldsymbol{T} \bullet \boldsymbol{v})_{ik} = \sum_{j=1}^{J} x_{ijk} v_j$ where $\boldsymbol{T} = (x_{ijk})$ and $\boldsymbol{v} = (v_1, \ldots, v_J)^\mathsf{T}$. For example, let $\boldsymbol{T} \in \mathbb{R}^{2 \times 3 \times 2}$ be the tensor of Example 2.1. Given the vector $\boldsymbol{v} = (\alpha, \beta, \gamma)^\mathsf{T}$, $\boldsymbol{T} \bullet \boldsymbol{v}$ becomes

$$\begin{pmatrix} a_1\alpha + b_1\beta + c_1\gamma & a_2\alpha + b_2\beta + c_2\gamma \\ d_1\alpha + e_1\beta + f_1\gamma & d_2\alpha + e_2\beta + f_2\gamma \end{pmatrix}$$

When $\boldsymbol{M} \in \mathbb{R}^{I \times J}$ is a matrix and $\boldsymbol{v} \in \mathbb{R}^J$, the product $\boldsymbol{M} \bullet \boldsymbol{v}$ is the standard matrix multiplication that becomes a vector in $\mathbb{R}^I$.

3 Tensor Logic Programming

We consider a language $\mathcal{L}$ that contains a finite set of propositional variables and the logical connectives $\neg$, $\wedge$, $\vee$ and $\leftarrow$. $\mathcal{L}$ contains $\top$ and $\bot$ representing *true* and *false*, respectively. Given a logic program P, the set of all propositional variables appearing in P is the *Herbrand base* of P (written B_P). We assume $\{\top, \bot\} \subseteq B_P$.

3.1 Horn Logic Programs

A *Horn (logic) program* is a finite set of *rules* of the form:

$$h \leftarrow \ b_1 \wedge \cdots \wedge b_m \quad (m \geq 0) \tag{1}$$

where h and b_i are propositional variables (also called atoms) in $\mathcal{L}$. In particular, the rule (1) is a *fact* if "$h \leftarrow \top$" and (1) is a *constraint* if "$\bot \leftarrow b_1 \wedge \cdots \wedge b_m$". The fact and the constraint are simply written as "$h \leftarrow$" and "$\leftarrow b_1 \wedge \cdots \wedge b_m$", respectively. For each rule r of the form (1), the left-hand side of $\leftarrow$ is the *head* and the right-hand side is the *body*. We write $head(r) = h$ and $body(r) = \{b_1, \ldots, b_m\}$. We assume that every program (implicitly) contains the rule $\top \leftarrow \top$. A *definite program* is a Horn program that contains no constraints. For a Horn program P, a set I satisfying $\{\top\} \subseteq I \subseteq B_P$ is an *interpretation* of P. An interpretation I is a *model* of P if $\{b_1, \ldots, b_m\} \subseteq I$ implies $h \in I$ for every rule (1) in P and $\bot \notin I$. We often omit $\top$ in a model, i.e., any model I is semantically identified with $I \setminus \{\top\}$. A model I is the *least model* of P if $I \subseteq J$ for any model J of P. P is *inconsistent* if it has no model. A mapping $T_P : 2^{B_P} \to 2^{B_P}$ is defined as $T_P(I) = \{\, h \mid h \leftarrow b_1 \wedge \cdots \wedge b_m \in P \text{ and } \{b_1, \ldots, b_m\} \subseteq I \,\}$ if $\bot \notin I$. Otherwise, $T_P(I) = B_P$.[4] The powers of T_P are defined as: $T_P^{k+1}(I) = T_P(T_P^k(I))$ $(k \geq 0)$ and $T_P^0(I) = I$. Given $\{\top\} \subseteq I \subseteq B_P$, there is a *fixpoint* $T_P^{n+1}(I) = T_P^n(I)$ $(n \geq 0)$. For a definite program P, the fixpoint $T_P^n(\{\top\})$ coincides with the least model of P [17].[5]

[4] The operator T_P of [17] is applied to Horn programs by viewing each constraint as a rule with the head $\bot$. In this setting, every atom in B_P is derived in $T_P(I)$ if I is inconsistent, i.e., $\bot \in I$. Note that $\top \in T_P(I)$ by $(\top \leftarrow \top) \in P$.

[5] $I = \{\top\}$ is semantically identified with $I = \{\}$.

In this paper, we consider a Horn program P such that for any two rules r_1 and r_2 in P, $head(r_1) = head(r_2)$ implies $|body(r_1)| \leq 1$ and $|body(r_2)| \leq 1$ (called the *multiple definitions (MD) condition*). That is, if two different rules have the same head, those rules contain at most one atom in their bodies. Every Horn program is converted to this class of programs by a simple program transformation. Given a Horn program P, define a set $Q_p \subseteq P$ such that $Q_p = \{ r \in P \mid head(r) = p \text{ and } |body(r)| > 1 \}$. Let $Q_p = \{ p \leftarrow \Gamma_1, \ldots, p \leftarrow \Gamma_k \}$ where Γ_i $(1 \leq i \leq k)$ is the conjunction in the body of each rule. Then Q_p is transformed to $Q'_p = \{ p_1 \leftarrow \Gamma_1, \ldots, p_k \leftarrow \Gamma_k \} \cup \{ p \leftarrow p_i \mid 1 \leq i \leq k \}$ where $p_i (\neq p)$ are new propositional variables associated with p and $p_i \neq p_j$ if $i \neq j$. Let P' be the program obtained from P by replacing $Q_p \subseteq P$ with Q'_p for every $p \in B_P$ satisfying the condition: $|\{r \in P \mid head(r) = p\}| > 1$ and $Q_p \neq \emptyset$. Then P has the least model M iff P' has the least model M' such that $M' \cap B_P = M$.

Example 3.1 Consider $P = \{ p \leftarrow q \wedge r,\ p \leftarrow r \wedge s,\ p \leftarrow t,\ r \leftarrow t,\ s \leftarrow,\ t \leftarrow \}$ where $B_P = \{p, q, r, s, t\}$. Then $P' = \{ p_1 \leftarrow q \wedge r,\ p_2 \leftarrow r \wedge s,\ p \leftarrow t,\ r \leftarrow\ t,\ s \leftarrow,\ t \leftarrow,\ p \leftarrow p_1,\ p \leftarrow p_2 \}$ has the least model $M' = \{p, p_2, r, s, t\}$ and $M' \cap B_P = \{p, r, s, t\}$ is the least model of P.

As such, every Horn program P is transformed to a semantically equivalent program P' that satisfies the MD condition. We then consider programs satisfying the MD condition without loss of generality. Next we represent interpretations/programs in vector spaces.

Definition 3.1 (vector representation of interpretations) Let P be a Horn program and $B_P = \{p_1, \ldots, p_n\}$. Then an interpretation $I \subseteq B_P$ of P is represented by a vector $\boldsymbol{v} = (a_1, \ldots, a_n)^\mathsf{T}$ where each element a_i represents the truth value of the proposition p_i such that $a_i = 1$ if $p_i \in I$ $(1 \leq i \leq n)$; otherwise, $a_i = 0$. The vector representing $I = \{\top\}$ is written by $\boldsymbol{v}_0$. We write $\mathsf{row}_i(\boldsymbol{v}) = p_i$.

Definition 3.2 ($\leq$) Let P be a Horn program and $B_P = \{p_1, \ldots, p_n\}$. Suppose two vectors $\boldsymbol{v} = (a_1, \ldots, a_n)^\mathsf{T}$ and $\boldsymbol{w} = (b_1, \ldots, b_n)^\mathsf{T}$ representing interpretations $I \subseteq B_P$ and $J \subseteq B_P$, respectively. Then $\boldsymbol{v} \leq \boldsymbol{w}$ if $a_i \leq b_i$ for every $1 \leq i \leq n$.

A vector $\boldsymbol{v} = (a_1, \ldots, a_n)^\mathsf{T}$ representing an interpretation I is *minimal* if $\boldsymbol{w} \leq \boldsymbol{v}$ implies $\boldsymbol{w} = \boldsymbol{v}$ for any $\boldsymbol{w}$ representing an interpretation J.

Proposition 3.1 *For $\boldsymbol{v} = (a_1, \ldots, a_n)^\mathsf{T}$ and $\boldsymbol{w} = (b_1, \ldots, b_n)^\mathsf{T}$, $\boldsymbol{v} \leq \boldsymbol{w}$ iff $a_i = a_i * b_i$ for $i = 1, \ldots, n$ where $*$ means algebraic multiplication.*

Definition 3.3 (matrix representation of Horn programs) Let P be a Horn program and $B_P = \{p_1, \ldots, p_n\}$. Then P is represented by a matrix $\boldsymbol{M}_P \in \mathbb{R}^{n \times n}$ such that for each element a_{ij} $(1 \leq i, j \leq n)$ in $\boldsymbol{M}_P$, (i) $a_{ij} = 1$ if $p_i = \top$ or $p_j = \bot$; (ii) $a_{ij_k} = \frac{1}{m}$ $(1 \leq k \leq m;\ 1 \leq i, j_k \leq n)$ if $p_i \leftarrow p_{j_1} \wedge \cdots \wedge p_{j_m}$ is in P; (iii) otherwise, $a_{ij} = 0$. We write $\mathsf{row}_i(\boldsymbol{M}_P) = p_i$ and $\mathsf{col}_j(\boldsymbol{M}_P) = p_j$.

In $\boldsymbol{M}_P$ the i-th row corresponds to the atom p_i appearing in the head of a rule, and the j-th column corresponds to the atom p_j appearing in the body of a rule. The first condition (i) says that every element in the i-th row is 1 if $\mathsf{row}_i(\boldsymbol{M}_P) = \top$ and every element in the j-th column is 1 if $\mathsf{col}_i(\boldsymbol{M}_P) = \bot$. The second condition (ii) says if there is a rule $p_i \leftarrow p_{j_1} \wedge \cdots \wedge p_{j_m}$ in P, then each a_{ij_k} in the i-th row and the j_k-th column has the value $\frac{1}{m}$. The remaining elements are set to $a_{ij} = 0$ in (iii).

Example 3.2 Consider $P = \{\, p \leftarrow q,\ \ p \leftarrow r,\ \ q \leftarrow r \wedge s,\ \ r \leftarrow,\ \ \leftarrow q \,\}$ with $B_P = \{p, q, r, s, \top, \bot\}$. Then $\boldsymbol{M}_P \in \mathbb{R}^{6\times 6}$ is the matrix (right). The 1st row "011001" represents $p \leftarrow q$, $p \leftarrow r$ and $p \leftarrow \bot\,(\equiv \top)$. The 2nd row "00 $^1/_2$ $^1/_2$ 01" represents $q \leftarrow r \wedge s$ and $q \leftarrow \bot\,(\equiv \top)$. The 3rd row "000011" represents $r \leftarrow \top\,(\equiv r \leftarrow)$ and $r \leftarrow \bot\,(\equiv \top)$. The 4th row "000001" represents $s \leftarrow \bot\,(\equiv \top)$. The 5th row "111111" represents $\top \leftarrow p$, $\top \leftarrow q$, $\top \leftarrow r$, $\top \leftarrow s$, $\top \leftarrow \top$, and $\top \leftarrow \bot$, which are all equivalent to $\top$. The 6th row "010001" represents $\bot \leftarrow q\,(\equiv \leftarrow q)$ and $\bot \leftarrow \bot\,(\equiv \top)$.

$$\begin{array}{c} \\ p \\ q \\ r \\ s \\ \top \\ \bot \end{array} \begin{array}{c} \begin{array}{cccccc} p & q & r & s & \top & \bot \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1/2 & 1/2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Suppose a matrix $\boldsymbol{M}_P \in \mathbb{R}^{n\times n}$ representing a Horn program. Given a vector $\boldsymbol{v} \in \mathbb{R}^n$ representing an interpretation $I \subseteq B_P$, let $\boldsymbol{M}_P \bullet \boldsymbol{v} = (a_1, \ldots, a_n)^\mathsf{T}$. We transform $\boldsymbol{M}_P \bullet \boldsymbol{v}$ to a vector $\boldsymbol{w} = (a'_1, \ldots, a'_n)^\mathsf{T}$ where $a'_i = 1$ $(1 \leq i \leq n)$ if $a_i \geq 1$; otherwise, $a'_i = 0$. We write $\boldsymbol{w} = \boldsymbol{M}_P \underline{\bullet}\, \boldsymbol{v}$.

Example 3.3 Consider $\boldsymbol{M}_P$ of Example 3.2. Given $\boldsymbol{v} = (0, 1, 1, 0, 1, 0)^\mathsf{T}$ representing the interpretation $I = \{q, r, \top\}$, it becomes $\boldsymbol{M}_P \bullet \boldsymbol{v} = (2, 1/2, 1, 0, 3, 1)^\mathsf{T}$. Then $\boldsymbol{w} = \boldsymbol{M}_P \underline{\bullet}\, \boldsymbol{v} = (1, 0, 1, 0, 1, 1)^\mathsf{T}$ represents $J = \{p, r, \top, \bot\}$.

In Example 3.3, we can see that $J = T_P(I)$ is computed by $\boldsymbol{w} = \boldsymbol{M}_P \underline{\bullet}\, \boldsymbol{v}$. Formally, the next result holds.

Proposition 3.2 *Let P be a Horn program and $\boldsymbol{M}_P \in \mathbb{R}^{n\times n}$ its matrix representation. Let $\boldsymbol{v} \in \mathbb{R}^n$ be a vector representing an interpretation $I \subseteq B_P$. Then $\boldsymbol{w} \in \mathbb{R}^n$ is a vector representing $J = T_P(I)$ iff $\boldsymbol{w} = \boldsymbol{M}_P \underline{\bullet}\, \boldsymbol{v}$.*

Proof. Suppose $\boldsymbol{w} = \boldsymbol{M}_P \underline{\bullet}\, \boldsymbol{v}$. Then for $\boldsymbol{w} = (x'_1, \ldots, x'_n)^\mathsf{T}$, $x'_k = 1$ $(1 \leq k \leq n)$ iff $x_k \geq 1$ in $\boldsymbol{M}_P \bullet \boldsymbol{v}$. Let $\boldsymbol{M}_P = (a_{ij})$ and $\boldsymbol{v} = (y_1, \ldots, y_n)^\mathsf{T}$. Then $a_{k1}y_1 + \cdots + a_{kn}y_n = x_k \geq 1$. If $\mathsf{row}_k(\boldsymbol{M}_P) = \top$ then $a_{kj} = 1$ $(1 \leq j \leq n)$. Otherwise, let $\{b_1, \ldots, b_m\} \subseteq \{a_{k1}, \ldots, a_{kn}\}$ such that $b_i \neq 0$ $(1 \leq i \leq m)$ and $b_i \neq a_{kj}$ for $\mathsf{col}_j(\boldsymbol{M}_P) = \bot$. Two cases are considered. (i) $b_i = 1$ $(1 \leq i \leq m)$ and $b_1 y_{j_1} + \cdots + b_m y_{j_m} \geq 1$ $(1 \leq j_l \leq n,\ 1 \leq l \leq m)$. Then there are rules $p_k \leftarrow p_i$ in P such that $p_i = \mathsf{col}_j(\boldsymbol{M}_P)$ for $b_i = a_{kj}$ $(1 \leq i \leq m)$ and $p_i \in I$ implies $p_k \in T_P(I)$ where $p_k = \mathsf{row}_k(\boldsymbol{w})$. (ii) $b_i = 1/m$ and $b_1 y_{j_1} + \cdots + b_m y_{j_m} = 1$ $(1 \leq j_l \leq n,\ 1 \leq l \leq m)$. Then there is a rule $p_k \leftarrow p_1 \wedge \cdots \wedge p_m$ in P such that $p_i = \mathsf{col}_j(\boldsymbol{M}_P)$ for $b_i = a_{kj}$ $(1 \leq i \leq m)$ and $\{p_1, \ldots, p_m\} \subseteq I$ implies $p_k \in T_P(I)$ where $p_k = \mathsf{row}_k(\boldsymbol{w})$. By putting $J = \{\, \mathsf{row}_k(\boldsymbol{w}) \mid x'_k = 1 \}$, $J = T_P(I)$ holds. Conversely, suppose $J = T_P(I)$. Construct a matrix $\boldsymbol{M}_P = (a_{ij}) \in \mathbb{R}^{n\times n}$ as Def. 3.3. For $\boldsymbol{v} = (y_1, \ldots, y_n)^\mathsf{T}$ representing I, $\boldsymbol{M}_P \bullet \boldsymbol{v} = (x_1, \ldots, x_n)^\mathsf{T}$ is a vector such that $x_k \geq 1$ $(1 \leq k \leq n)$ iff $\mathsf{row}_k(\boldsymbol{M}_P \bullet \boldsymbol{v}) = \top$ or $\mathsf{row}_k(\boldsymbol{M}_P \bullet \boldsymbol{v}) \in T_P(I)$. Define $\boldsymbol{w} = (x'_1, \ldots, x'_n)^\mathsf{T}$ such that $x'_k = 1$ $(1 \leq k \leq n)$ iff $x_k \geq 1$ in $\boldsymbol{M}_P \bullet \boldsymbol{v}$. Then $\boldsymbol{w}$ represents $J = T_P(I)$. Hence, $\boldsymbol{w} = \boldsymbol{M}_P \underline{\bullet}\, \boldsymbol{v}$. □

Proposition 3.3 *Let P be a Horn program and $\boldsymbol{M}_P \in \mathbb{R}^{n\times n}$ its matrix representation. Let $\boldsymbol{v} \in \mathbb{R}^n$ be a vector representing an interpretation I. Then I is a model of P iff (i) $\boldsymbol{w} = \boldsymbol{M}_P \underline{\bullet}\, \boldsymbol{v}$ and $\boldsymbol{w} \leq \boldsymbol{v}$, and (ii) $a_i = 1$ implies $\mathsf{row}_i(\boldsymbol{v}) \neq \bot$ for any element a_i in $\boldsymbol{v}$.*

Proof. I is a model of a Horn program P iff $T_P(I) \subseteq I$ and $\bot \notin I$ ([9]). Then the result holds by Proposition 3.2. □

Given a matrix $\boldsymbol{M}_P \in \mathbb{R}^{n\times n}$ and a vector $\boldsymbol{v} \in \mathbb{R}^n$, define

$$\boldsymbol{M}_P \mathbin{\underline{\bullet}}^{k+1} \boldsymbol{v} = \boldsymbol{M}_P \mathbin{\underline{\bullet}} (\boldsymbol{M}_P \mathbin{\underline{\bullet}}^k \boldsymbol{v}) \quad \text{and} \quad \boldsymbol{M}_P \mathbin{\underline{\bullet}}^1 \boldsymbol{v} = \boldsymbol{M}_P \mathbin{\underline{\bullet}} \boldsymbol{v} \quad (k \geq 1).$$

When $\boldsymbol{M}_P \mathbin{\underline{\bullet}}^{k+1} \boldsymbol{v} = \boldsymbol{M}_P \mathbin{\underline{\bullet}}^k \boldsymbol{v}$ for some $k \geq 1$, we write $\mathsf{FP}(\boldsymbol{M}_P \mathbin{\underline{\bullet}} \boldsymbol{v}) = \boldsymbol{M}_P \mathbin{\underline{\bullet}}^k \boldsymbol{v}$.

Theorem 3.4 *Let P be a Horn program and $\boldsymbol{M}_P \in \mathbb{R}^{n\times n}$ its matrix representation. Then $\boldsymbol{m} \in \mathbb{R}^n$ is a vector representing the least model of P iff $\boldsymbol{m} = \mathsf{FP}(\boldsymbol{M}_P \mathbin{\underline{\bullet}} \boldsymbol{v}_0)$ and $a_i = 1$ implies $\mathsf{row}_i(\boldsymbol{m}) \neq \bot$ for any element a_i in $\boldsymbol{m}$.*

Proof. Since the least model of P is computed by the fixpoint of $T_P^k(\{\top\})$ $(k \geq 0)$, the result holds by Proposition 3.2. □

Corollary 3.5 *Let P be a Horn program and $\boldsymbol{M}_P \in \mathbb{R}^{n\times n}$ its matrix representation. Then P is inconsistent iff a vector $\boldsymbol{w} = \boldsymbol{M}_P \mathbin{\underline{\bullet}}^k \boldsymbol{v}_0$ $(k \geq 1)$ has an element $a_i = 1$ $(1 \leq i \leq n)$ such that $\mathsf{row}_i(\boldsymbol{w}) = \bot$.*

Corollary 3.5 is used for query-answering by refutation in a definite program.

3.2 Disjunctive Logic Programs

A *disjunctive (logic) program* is a finite set of *rules* of the form:

$$h_1 \vee \cdots \vee h_l \leftarrow b_1 \wedge \cdots \wedge b_m \quad (l, m \geq 0) \tag{2}$$

where h_i and b_j are propositional variables in $\mathcal{L}$. A rule (2) is called a *disjunctive rule* if $l > 1$. In particular, the rule is a *(disjunctive) fact* if the body of (2) is $\top$ and it is a *constraint* if the head of (2) is $\bot$. A disjunctive fact is simply written as "$h_1 \vee \cdots \vee h_l \leftarrow$". For each rule r of the form (2), we write $head(r) = \{h_1, \ldots, h_l\}$ and $body(r) = \{b_1, \ldots, b_m\}$. A disjunctive program reduces to a Horn program if $l \leq 1$ for every rule (2) in a program. We assume that every program (implicitly) contains the rule $\top \leftarrow \top$ as before. An interpretation $\{\top\} \subseteq I \subseteq B_P$ is a *model* of a disjunctive program P if $body(r) \subseteq I$ implies $head(r) \cap I \neq \emptyset$ for every rule r in P and $\bot \notin I$. A model I is a *minimal model* of P if there is no model J of P such that $J \subset I$.

We consider a disjunctive program P satisfying the *MD condition*: for any two rules r_1 and r_2 in P, $head(r_1) \cap head(r_2) \neq \emptyset$ implies $|body(r_1)| \leq 1$ and $|body(r_2)| \leq 1$. That is, if two different rules share the same atom in their heads, those rules contain at most one atom in their bodies. The condition coincides with the MD condition of Horn programs when P contains no disjunctive rules. Every disjunctive program is converted to this class of programs by a simple program transformation as the case of Horn programs. Given a disjunctive program P, define a set $Q_p \subseteq P$ such that $Q_p = \{\, r \in P \mid p \in head(r) \text{ and } |body(r)| > 1 \,\}$. Let $Q_p = \{\, \Sigma_1 \leftarrow \Gamma_1, \ldots, \Sigma_k \leftarrow \Gamma_k \,\}$ where Σ_i (resp. Γ_i) $(1 \leq i \leq k)$ is the disjunction (resp. conjunction) in the head (resp. body) of each rule. Then Q_p is transformed to $Q'_p = \{\, \Sigma'_1 \leftarrow \Gamma_1, \ldots, \Sigma'_k \leftarrow \Gamma_k \,\} \cup \{\, p \leftarrow p_i \mid 1 \leq i \leq k \,\}$ where $p_i\,(\neq p)$ are new propositional variables associated with p and $p_i \neq p_j$ if $i \neq j$. Σ'_i is obtained from Σ_i by replacing p in Σ_i by p_i. Let P'_p be the program obtained from P by replacing $Q_p \subseteq P$ with Q'_p for an atom $p \in B_P$ satisfying the condition: $|\{r \in P \mid p \in head(r)\}| > 1$ and $Q_p \neq \emptyset$. Repeat such replacement one by one until there is no atom $p \in B_P$ satisfying the above condition. Let P' be the resulting program. Then P has a minimal model M iff P' has a minimal model M' such that $M' \cap B_P = M$.

Example 3.4 Consider $P = \{\, p \vee q \leftarrow r \wedge s,\quad p \vee r \leftarrow t,\quad r \leftarrow s,\quad s \leftarrow \,\}$ where $B_P = \{p, q, r, s, t\}$. Then $P' = \{\, p_1 \vee q \leftarrow r \wedge s,\ \ p \vee r \leftarrow t,\ \ r \leftarrow s,\ \ s \leftarrow,\ \ p \leftarrow p_1 \,\}$ has two minimal models $M'_1 = \{p, p_1, r, s\}$ and $M'_2 = \{q, r, s\}$ which correspond to the minimal models $M_1 = \{p, r, s\}$ and $M_2 = \{q, r, s\}$ of P.

As such, every disjunctive program P is transformed to a semantically equivalent program P' that satisfies the MD condition. We then consider disjunctive programs satisfying the MD condition without loss of generality. Given a disjunctive program P, its *split program* is a Horn program obtained from P by replacing each disjunctive rule of the form (2) in P with a Horn rule: $h_i \leftarrow\ b_1 \wedge \cdots \wedge b_m$ $(1 \leq i \leq l)$. By definition, P has multiple split programs in general. When P has k split programs, it is written as $SP_1, \ldots, SP_k$ $(k \geq 1)$.

Example 3.5 $P = \{\, p \vee r \leftarrow s,\quad q \vee r \leftarrow,\quad s \leftarrow \,\}$ has the four split programs: $SP_1 = \{\, p \leftarrow s,\quad q \leftarrow,\quad s \leftarrow \,\}$, $SP_2 = \{\, p \leftarrow s,\quad r \leftarrow,\quad s \leftarrow \,\}$, $SP_3 = \{\, r \leftarrow s,\ \ q \leftarrow,\ \ s \leftarrow \,\}$, and $SP_4 = \{\, r \leftarrow s,\ \ r \leftarrow,\ \ s \leftarrow \,\}$.

For a set $\mathcal{I}$ of interpretations, define the set of minimal elements as $min(\mathcal{I}) = \{\, I \mid I \in \mathcal{I}$ and there is no $J \in \mathcal{I}$ such that $J \subset I \,\}$.

Proposition 3.6 *Let P be a disjunctive program and $SP_1, \ldots, SP_k$ its split programs. Also let $\mathcal{LM}$ be the set of least models of $SP_1, \ldots, SP_k$. Then $min(\mathcal{LM})$ coincides with the set $\mathcal{MM}$ of minimal models of P.*

Proof. Let M be the least model of some split program SP_j. Then for each rule r : $h_i \leftarrow b_1 \wedge \cdots \wedge b_m$ in SP_j, there is a rule r' : $h_1 \vee \cdots \vee h_l \leftarrow\ b_1 \wedge \cdots \wedge b_m$ in P such that $h_i \in head(r')$. Since M satisfies r, M satisfies r'. Thus, M is a model of P. Then a minimal set M in $min(\mathcal{LM})$ is a minimal model of P. Hence, $min(\mathcal{LM}) \subseteq \mathcal{MM}$. Conversely, let $M \in \mathcal{MM}$. Then for each rule r : $h_1 \vee \cdots \vee h_l \leftarrow\ b_1 \wedge \cdots \wedge b_m$ in P, $\{b_1, \ldots, b_m\} \subseteq M$ implies $h_i \in M$ for some i $(1 \leq i \leq l)$. In this case, there is a split program SP_j of P in which r is replaced by $h_i \leftarrow b_1 \wedge \cdots \wedge b_m$. Then M is the least model of SP_j. Since M is a minimal among models in $\mathcal{LM}$, $\mathcal{MM} \subseteq min(\mathcal{LM})$. □

Example 3.6 Consider the program P in Example 3.5. The set of least models of split programs is $\mathcal{LM} = \{\{\, p, q, s \,\}, \{\, p, r, s \,\}, \{\, q, r, s \,\}, \{\, r, s \,\}\}$. Then $min(\mathcal{LM}) = \{\{p, q, s\}, \{r, s\}\}$, which is the set of minimal models of P.

By definition, if a disjunctive program satisfies the MD condition, its split program satisfies the MD condition for Horn programs. Next we introduce tensor representation of a disjunctive program in terms of its split programs.

Definition 3.4 (tensor representation of disjunctive programs) Let P be a disjunctive program that is split into $SP_1, \ldots, SP_k$ and $B_P = \{p_1, \ldots, p_n\}$. Then P is represented by a third-order tensor $\boldsymbol{U}_P \in \mathbb{R}^{n \times n \times k}$ as follows.

1. Each slice $U_{::h}$ $(1 \leq h \leq k)$ of $\boldsymbol{U}_P$ is a matrix $\boldsymbol{M}_h \in \mathbb{R}^{n \times n}$ representing a split program SP_h.
2. Each matrix $\boldsymbol{M}_h \in \mathbb{R}^{n \times n}$ has an element a_{ij} $(1 \leq i, j \leq n)$ such that
   (a) $a_{ij} = 1$ if $p_i = \top$ or $p_j = \bot$.
   (b) $a_{ij_l} = \frac{1}{m}$ $(1 \leq l \leq m;\ 1 \leq i, j_l \leq n)$ if $p_i \leftarrow p_{j_1} \wedge \cdots \wedge p_{j_m}$ is in SP_h.
   (c) Otherwise, $a_{ij} = 0$.

$U_{::h}$ represents a slice that is obtained by fixing an index h ($1 \leq h \leq k$). The index h represents the h-th split program SP_h of P. In this way, a disjunctive program is represented by a third-order tensor by introducing an additional dimension to the matrix representation of Horn programs.

Suppose a third-order tensor $\boldsymbol{U}_P \in \mathbb{R}^{n\times n\times k}$ of Def. 3.4. Given a vector $\boldsymbol{v} \in \mathbb{R}^n$ representing an interpretation $I \subseteq B_P$, the (2-mode) product $\boldsymbol{U}_P \bullet \boldsymbol{v}$ produces a matrix $(a_{ij}) \in \mathbb{R}^{n\times k}$. We transform $\boldsymbol{U}_P \bullet \boldsymbol{v}$ to a matrix $\boldsymbol{W}_P = (a'_{ij}) \in \mathbb{R}^{n\times k}$ in a way that $a'_{ij} = 1$ $(1 \leq i \leq n; 1 \leq j \leq k)$ if $a_{ij} \geq 1$ in $\boldsymbol{U}_P \bullet \boldsymbol{v}$; otherwise, $a_{ij} = 0$. We write $\boldsymbol{W}_P = \boldsymbol{U}_P \underline{\bullet}\, \boldsymbol{v}$.

Example 3.7 The program P in Example 3.5 is represented by the 3rd-order tensor $\boldsymbol{U}_P \in \mathbb{R}^{5\times 5\times 4}$ such that[6]

$$U_{::1} = \begin{array}{c} \\ p \\ q \\ r \\ s \\ \top \end{array}\!\!\begin{array}{c} \begin{array}{ccccc} p & q & r & s & \top \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array} \quad U_{::2} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad U_{::3} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad U_{::4} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

where $U_{::1}$, $U_{::2}$, $U_{::3}$ and $U_{::4}$ represent SP_1, SP_2, SP_3, and SP_4, respectively. Given the vector $\boldsymbol{v}_0 = (0,0,0,0,1)^\mathsf{T}$ representing $I = \{\top\}$, the product $\boldsymbol{U}_P \bullet \boldsymbol{v}_0$ becomes the matrix in $\mathbb{R}^{5\times 4}$ where $\boldsymbol{W}_P = \boldsymbol{U}_P \underline{\bullet}\, \boldsymbol{v}_0 = \boldsymbol{U}_P \bullet \boldsymbol{v}_0$.

In the matrix $\boldsymbol{W}_P$ (right), the 1st column $(0,1,0,1,1)^\mathsf{T}$ represents the interpretation $T_{SP_1}(\{\top\}) = \{q, s, \top\}$, the 2nd column $(0,0,1,1,1)^\mathsf{T}$ represents $T_{SP_2}(\{\top\}) = \{r, s, \top\}$, the 3rd column $(0,1,0,1,1)^\mathsf{T}$ represents $T_{SP_3}(\{\top\}) = \{q, s, \top\}$, and the 4th column $(0,0,1,1,1)^\mathsf{T}$ represents $T_{SP_4}(\{\top\}) = \{r, s, \top\}$.

$$\begin{array}{c} \\ p \\ q \\ r \\ s \\ \top \end{array}\!\!\begin{array}{c} \begin{array}{cccc} & & & \end{array} \\ \left(\begin{array}{c|c|c|c} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array}\right) \end{array}$$

Given a matrix $\boldsymbol{M} = (a_{ij})$ $(1 \leq i \leq m; 1 \leq j \leq n)$, $(a_{1j}, \ldots, a_{mj})^\mathsf{T}$ is said a *column vector in* $\boldsymbol{M}$. Let $\boldsymbol{U}_P \in \mathbb{R}^{n\times n\times m}$ and $\boldsymbol{M} \in \mathbb{R}^{n\times m}$. The product $\boldsymbol{U}_P \bullet \boldsymbol{M}$ is then defined as the matrix in $\mathbb{R}^{n\times m}$ such that each column vector in $\boldsymbol{U}_P \bullet \boldsymbol{M}$ is $U_{::k} \bullet (a_{1k}, \ldots, a_{nk})^\mathsf{T}$ $(1 \leq k \leq m)$ where $(a_{1k}, \ldots, a_{nk})^\mathsf{T}$ is a column vector in $\boldsymbol{M}$. Then $\boldsymbol{U}_P \underline{\bullet}\, \boldsymbol{M}$ is the matrix obtained from $\boldsymbol{U}_P \bullet M$ by replacing each element a_{ij} in $\boldsymbol{U}_P \bullet M$ by $a'_{ij} = 1$ if $a_{ij} \geq 1$; otherwise, replacing a_{ij} by $a'_{ij} = 0$. Given a tensor $\boldsymbol{U}_P \in \mathbb{R}^{n\times n\times m}$ and a vector $\boldsymbol{v} \in \mathbb{R}^n$, define

$$\boldsymbol{U}_P \underline{\bullet}^{k+1}\, \boldsymbol{v} = \boldsymbol{U}_P \underline{\bullet}\, (\boldsymbol{U}_P \underline{\bullet}^{k}\, \boldsymbol{v}) \quad \text{and} \quad \boldsymbol{U}_P \underline{\bullet}^{1}\, \boldsymbol{v} = \boldsymbol{U}_P \underline{\bullet}\, \boldsymbol{v} \quad (k \geq 1).$$

When $\boldsymbol{U}_P \underline{\bullet}^{k+1}\, \boldsymbol{v} = \boldsymbol{U}_P \underline{\bullet}^{k}\, \boldsymbol{v}$ for some $k \geq 1$, we write $\mathsf{FP}(\boldsymbol{U}_P \underline{\bullet}\, \boldsymbol{v}) = \boldsymbol{U}_P \underline{\bullet}^{k}\, \boldsymbol{v}$.

Theorem 3.7 *Let P be a disjunctive program and $\boldsymbol{U}_P \in \mathbb{R}^{n\times n\times k}$ its tensor representation of Def. 3.4. Let $\boldsymbol{M}_P = \mathsf{FP}(\boldsymbol{U}_P \underline{\bullet}\, \boldsymbol{v}_0)$ be a matrix where $\boldsymbol{v}_0$ represents $I = \{\top\}$. Then a vector $\boldsymbol{m} \in \mathbb{R}^n$ in $\boldsymbol{M}_P$ represents a minimal model of P iff $\boldsymbol{m}$ is minimal among all column vectors in $\boldsymbol{M}_P$ and $a_i = 1$ implies $\mathsf{row}_i(\boldsymbol{m}) \neq \bot$ for any element a_i in $\boldsymbol{m}$.*

Proof. The result holds by Theorem 3.4 and Proposition 3.6. □

[6] Here we omit the row and the column representing $\bot$. When a program contains no constraints, the row and the column representing $\bot$ can be removed (see Section 4).

Example 3.8 In Example 3.7, $\boldsymbol{M}_P = \boldsymbol{U}_P \,\underline{\bullet}^k\, \boldsymbol{v}_0$ $(k \geq 2)$ becomes the matrix (below). The 1st column $(1,1,0,1,1)^\mathsf{T}$ represents the minimal model $\{p,q,s\}$ of SP_1, the 2nd column $(1,0,1,1,1)^\mathsf{T}$ represents the minimal model $\{p,r,s\}$ of SP_2, the 3rd column $(0,1,1,1,1)^\mathsf{T}$ represents the minimal model $\{q,r,s\}$ of SP_3, and the 4th column $(0,0,1,1,1)^\mathsf{T}$ represents the minimal model $\{r,s\}$ of SP_4.

$$\begin{array}{c} p \\ q \\ r \\ s \\ \top \end{array} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Then two vectors $(1,1,0,1,1)^\mathsf{T}$ and $(0,0,1,1,1)^\mathsf{T}$ are minimal in $\boldsymbol{M}_P$, which represents two minimal models $\{p,q,s\}$ and $\{r,s\}$ of P.[7]

3.3 Normal Logic Programs

A *normal (logic) program* is a finite set of *rules* of the form:

$$h \leftarrow\ b_1 \wedge \cdots \wedge b_m \wedge \neg\, b_{m+1} \wedge \cdots \wedge \neg b_n \quad (n \geq m \geq 0) \tag{3}$$

where h and b_i are propositional variables in $\mathcal{L}$. h and b_i $(1 \leq i \leq m)$ are *positive literals*, and $\neg b_j$ $(m+1 \leq j \leq n)$ are *negative literals*. As before, a rule is a *fact* if the body of (3) is $\top$ and it is a *constraint* if the head of (3) is $\bot$. A program implicitly contains $\top \leftarrow \top$. For each rule r of the form (3), we write $head(r) = h$, $body^+(r) = \{\, b_1, \ldots, b_m\}$ and $body^-(r) = \{\, b_{m+1}, \ldots, b_n \,\}$. A normal program reduces to a Horn program if it contains no negative literals. An interpretation $\{\top\} \subseteq I \subseteq B_P$ is a *model* of a normal program P if $body^+(r) \subseteq I$ and $body^-(r) \cap I = \emptyset$ imply $head(r) \in I$ for every rule r in P and $\bot \notin I$. Given a normal program P, an interpretation I is a *stable model* of P [4] if it coincides with the least model of the Horn program: $P^I = \{\, h \leftarrow b_1 \wedge \cdots \wedge b_m \mid\ h \leftarrow\ b_1 \wedge \cdots \wedge b_m \wedge \neg\, b_{m+1} \wedge \cdots \wedge \neg b_n \in P$ and $\{b_{m+1}, \ldots, b_n\} \cap I = \emptyset \,\}$. A stable model coincides with the least model if P is a Horn program. A normal program may have no, one, or multiple stable models in general. A program is *consistent* if it has at least one stable model; otherwise, the program is *inconsistent*.

In this paper, we consider a normal program P satisfying the following *MD condition*: for two rules r_1 and r_2 in P, (i) $head(r_1) = head(r_2)$ implies $|body^+(r_1)| \leq 1$ and $|body^+(r_2)| \leq 1$, and (ii) $body^-(r_1) \cap body^-(r_2) \neq \emptyset$ implies $|body^+(r_1)| \leq 1$ and $|body^+(r_2)| \leq 1$. The condition reduces to the one of Horn programs when P contains no negative literals. Every normal program is converted to this class of programs by a simple program transformation. Given a normal program P, define a set $Q_p^+, Q_q^- \subseteq P$ such that $Q_p^+ = \{\, r \in P \mid head(r) = p$ and $\mid body^+(r) \mid > 1 \,\}$ and $Q_q^- = \{\, r \in P \mid q \in body^-(r)$ and $\mid body^+(r) \mid > 1 \,\}$. Let $Q_p^+ = \{\, p \leftarrow \Gamma_1, \ldots, p \leftarrow \Gamma_k \,\}$ and $Q_q^- = \{\, h_1 \leftarrow \Upsilon_1 \wedge not\, q, \ldots, h_l \leftarrow \Upsilon_l \wedge not\, q \,\}$ where Γ_i $(1 \leq i \leq k)$ or Υ_j $(1 \leq j \leq l)$ is the conjunction in the body of each rule. Then Q_p^+ and Q_q^- are respectively transformed to $R_p^+ = \{\, p_1 \leftarrow \Gamma_1, \ldots, p_k \leftarrow \Gamma_k \,\} \cup \{\, p \leftarrow p_i \mid 1 \leq i \leq k \,\}$ and $R_q^- = \{\, h_1 \leftarrow \Upsilon_1 \wedge not\, q_1, \ldots, h_l \leftarrow \Upsilon_l \wedge not\, q_l \,\} \cup \{\, q_j \leftarrow q \mid 1 \leq j \leq l \,\}$ where p_i $(\neq p)$ and q_j $(\neq q)$ are new propositional variables respectively associated with p and q, and $p_i \neq p_j$ $(q_i \neq q_j)$ if $i \neq j$. Let P' be the program obtained from P by replacing $Q_p^+ \subseteq P$ with R_p^+ for every $p \in B_P$ satisfying the condition: $|\,\{r \in P \mid head(r) = p\}\,| > 1$ and

[7] Here we omit $\top$ in each model.

$Q_p^+ \neq \emptyset$. Next let P''_q be the program obtained from P' by replacing $Q_q^- \subseteq P'$ with R_q^- for an atom $q \in B_P$ satisfying the condition: $|\{r \in P' \mid q \in body^-(r)\}| > 1$ and $Q_q^- \neq \emptyset$. Repeat the replacement one by one until there is no atom $q \in B_P$ satisfying the above condition. Let P'' be the resulting program. Then, P has a stable model M iff P'' has a stable model M' such that $M' \cap B_P = M$.

Example 3.9 Consider $P = \{\, p \leftarrow q \wedge r \wedge not\, s,\ \ p \leftarrow r \wedge t \wedge not\, s,\ \ q \leftarrow t,\ \ r \leftarrow,\ \ t \leftarrow \}$. First, it is converted to $P' = \{\, p_1 \leftarrow q \wedge r \wedge not\, s,\ \ p_2 \leftarrow r \wedge t \wedge not\, s,\ \ q \leftarrow t,\ \ r \leftarrow,\ \ t \leftarrow,\ \ p \leftarrow p_1,\ \ p \leftarrow p_2 \}$. Next, it is converted to $P'' = \{\, p_1 \leftarrow q \wedge r \wedge not\, s_1,\ \ p_2 \leftarrow r \wedge t \wedge not\, s_2,\ \ q \leftarrow t,\ \ r \leftarrow,\ \ t \leftarrow,\ \ p \leftarrow p_1,\ \ p \leftarrow p_2,\ \ s_1 \leftarrow s,\ \ s_2 \leftarrow s \}$. Then P'' has the single stable model $M' = \{p, p_1, p_2, q, r, t\}$ which corresponds to the stable model $M = \{p, q, r, t\}$ of P.

As such, every normal program P is transformed to a semantically equivalent program P'' that satisfies the MD condition. We then assume normal programs satisfying the MD condition without loss of generality. A normal program P is transformed to a disjunctive program with *integrity constraints* as follows [3].

$$
\begin{aligned}
P^\varepsilon = \{\, & h \vee \varepsilon b_{m+1} \vee \cdots \vee \varepsilon b_n \leftarrow\ b_1 \wedge \cdots \wedge b_m \mid \\
& (h \leftarrow\ b_1 \wedge \cdots \wedge b_m \wedge \neg b_{m+1} \wedge \cdots \wedge \neg b_n) \in P \,\} \cup \{\, \varepsilon p \leftarrow p \mid p \in B_P \setminus \{\top, \bot\} \,\}, \\
IC_P = \{\, & \varepsilon p \Rightarrow p \mid p \in B_P \setminus \{\top, \bot\} \,\}
\end{aligned}
$$

where εp is a new atom associated with p. Let B_{P^ε} be the Herbrand base of P^ε such that $\{\top, \bot\} \subseteq B_{P^\varepsilon}$. An interpretation $\{\top\} \subseteq I \subseteq B_{P^\varepsilon}$ *satisfies* IC_P iff $\varepsilon p \in I$ implies $p \in I$ for every $\varepsilon p \Rightarrow p$ in IC_P.

Proposition 3.8 [3] *Let P be a normal program. M is a stable model of P iff $M \cup \varepsilon M$ is a minimal model of P^ε satisfying IC_P where $\varepsilon M = \{\, \varepsilon p \mid p \in M \}$.*

Example 3.10 Consider $P = \{\, p \leftarrow \neg q,\ \ q \leftarrow \neg p \,\}$. Then $P^\varepsilon = \{\, p \vee \varepsilon q \leftarrow,\ q \vee \varepsilon p \leftarrow,\ \varepsilon p \leftarrow p,\ \ \varepsilon q \leftarrow q \,\}$ and $IC_P = \{\, \varepsilon p \Rightarrow p,\ \ \varepsilon q \Rightarrow q \,\}$. The program P^ε has three minimal models: $M_1 = \{p, \varepsilon p\}$, $M_2 = \{q, \varepsilon q\}$, and $M_3 = \{\varepsilon p, \varepsilon q\}$. Of which M_1 and M_2 satisfy IC_P. Then $\{p\}$ and $\{q\}$ are two stable models of P.

By definition, if a normal program P satisfies the MD condition, P^ε satisfies the MD condition for disjunctive programs. Suppose that a disjunctive program P^ε is split into $SP_1^\varepsilon, \ldots, SP_k^\varepsilon$. Then P^ε is represented by a third-order tensor $\boldsymbol{U}_{P^\varepsilon} \in \mathbb{R}^{n \times n \times k}$ as in Def. 3.4, and we can compute stable models of P in terms of minimal models of P^ε.

Definition 3.5 ($\boldsymbol{v}_{B_P}$, $\boldsymbol{v}_{\varepsilon B_P}$) Let $\boldsymbol{v} \in \mathbb{R}^n$ be a vector representing $\{\top\} \subseteq I \subseteq B_{P^\varepsilon}$. Then $\boldsymbol{v}$ is divided into two vectors $\boldsymbol{v}_{B_P} \in \mathbb{R}^m$ and $\boldsymbol{v}_{\varepsilon B_P} \in \mathbb{R}^m$ $(m \leq n)$ where $\boldsymbol{v}_{B_P}$ represents elements of $B_P = \{p_1, \ldots, p_m\}$ $(\top, \bot \in B_P)$ and $\boldsymbol{v}_{\varepsilon B_P}$ represents elements of $(B_{P^\varepsilon} \setminus B_P) \cup \{\bot, \top\}$. Two vectors satisfy the condition: $\mathsf{row}_i(\boldsymbol{v}_{B_P}) = \mathsf{row}_i(\boldsymbol{v}_{\varepsilon B_P})$ iff $\mathsf{row}_i(\boldsymbol{v}_{B_P}) = \top$ or $\bot$; otherwise, $p_i = \mathsf{row}_i(\boldsymbol{v}_{B_P})$ iff $\varepsilon p_i = \mathsf{row}_i(\boldsymbol{v}_{\varepsilon B_P})$ $(1 \leq i \leq m)$.

Proposition 3.9 *Let P be a normal program and $\boldsymbol{v} \in \mathbb{R}^n$ a vector representing $\{\top\} \subseteq I \subseteq B_{P^\varepsilon}$. Then I satisfies IC_P iff $\boldsymbol{v}_{\varepsilon B_P} \leq \boldsymbol{v}_{B_P}$.*

Proof. Let $\boldsymbol{v}_{B_P} = (a_1, \ldots, a_m)^\mathsf{T}$ and $\boldsymbol{v}_{\varepsilon B_P} = (b_1, \ldots, b_m)^\mathsf{T}$ $(m \leq n)$. Then, $\boldsymbol{v}_{\varepsilon B_P} \leq \boldsymbol{v}_{B_P}$ iff $b_i = 1$ implies $a_i = 1$ $(1 \leq i \leq m)$ for any $\mathsf{row}_i(\boldsymbol{v}_{\varepsilon B_P}) = \varepsilon p$ and $\mathsf{row}_i(\boldsymbol{v}_{B_P}) = p$ iff $\varepsilon p \in I$ implies $p \in I$. Hence, the result holds. □

Theorem 3.10 *Let P be a normal program and $B_{P^\varepsilon} = \{p_1, \ldots, p_n\}$. Also let $\boldsymbol{U}_{P^\varepsilon} \in \mathbb{R}^{n\times n\times k}$ be a tensor representation of a disjunctive program P^ε where k is the number of split programs of P^ε. Then*

1. *$\boldsymbol{m} = \boldsymbol{v}_{B_P}$ represents a stable model of P iff $\boldsymbol{v} \in \mathbb{R}^n$ represents a minimal model of P^ε and $\boldsymbol{v}_{\varepsilon B_P} \leq \boldsymbol{v}_{B_P}$.*
2. *P is inconsistent iff $\boldsymbol{v}_{\varepsilon B_P} \not\leq \boldsymbol{v}_{B_P}$ for any $\boldsymbol{v} \in \mathbb{R}^n$ representing a minimal model of P^ε.*

Proof. Since minimal models of P^ε are computed by Theorem 3.7, the result holds by Propositions 3.8 and 3.9. □

Example 3.11 The program P^ε of Example 3.10 has four split programs $SP^\varepsilon_1 = \{\, p \leftarrow,\ q \leftarrow \} \cup R$, $SP^\varepsilon_2 = \{\, p \leftarrow,\ \varepsilon p \leftarrow \} \cup R$, $SP^\varepsilon_3 = \{\, \varepsilon q \leftarrow,\ q \leftarrow \} \cup R$, and $SP^\varepsilon_4 = \{\, \varepsilon q \leftarrow,\ \varepsilon p \leftarrow \} \cup R$ where $R = \{\, \varepsilon p \leftarrow p,\ \varepsilon q \leftarrow q \,\}$. Then P^ε is represented by the tensor $\boldsymbol{U}_{P^\varepsilon} \in \mathbb{R}^{5\times 5\times 4}$. $\mathsf{FP}(\boldsymbol{U}_{P^\varepsilon} \underline{\bullet} \boldsymbol{v}_0)$ is the matrix where $\boldsymbol{v} = (1,0,1,0,1)^\mathsf{T}$, $\boldsymbol{v}' = (0,1,0,1,1)^\mathsf{T}$ and $\boldsymbol{v}'' = (0,0,1,1,1)^\mathsf{T}$ are minimal (right). Then $\boldsymbol{v}_{B_P} = (1,0,1)^\mathsf{T}$ (representing p, q, $\top$) and $\boldsymbol{v}_{\varepsilon B_P} = (1,0,1)^\mathsf{T}$ (representing εp, εq, $\top$) satisfy $\boldsymbol{v}_{\varepsilon B_P} \leq \boldsymbol{v}_{B_P}$. Also, $\boldsymbol{v}'_{B_P} = (0,1,1)^\mathsf{T}$ (representing p, q, $\top$) and $\boldsymbol{v}'_{\varepsilon B_P} = (0,1,1)^\mathsf{T}$ (representing εp, εq, $\top$) satisfy $\boldsymbol{v}'_{\varepsilon B_P} \leq \boldsymbol{v}'_{B_P}$. By contrast, $\boldsymbol{v}''$ does not satisfy $\boldsymbol{v}''_{\varepsilon B_P} \leq \boldsymbol{v}''_{B_P}$. Then $\boldsymbol{v}_{B_P} = (1,0,1)^\mathsf{T}$ and $\boldsymbol{v}'_{B_P} = (0,1,1)^\mathsf{T}$ represent two stable models $\{p\}$ and $\{q\}$ of P.

$$\begin{array}{c} p \\ q \\ \varepsilon p \\ \varepsilon q \\ \top \end{array} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

The result of this section is extended to *normal disjunctive programs* (i.e., programs containing both disjunction and negation) by combining techniques of Sec. 3.2 and 3.3.

4 Discussion

It is known that the least model of a Horn program is computed in $O(N)$ time and space where N is the size of the program, i.e., the total number of occurrences of literals in the program [2]. The proposed method requires $O(n^2)$ space and $O(n^4)$ time (i.e., $O(n^3)$ for matrix multiplication and at most n-iteration of powers) in the worst case where n is the number of propositional variables in B_P. Since the size of a matrix is independent of the size of a program, the proposed method would be advantageous in a large knowledge base on a fixed language. Several optimization techniques are considered for efficient computation in practice. When every element in the i-th row of a matrix $\boldsymbol{M}_P$ is zero except the element $a_{ij} = 1$ for $\mathsf{col}_j(\boldsymbol{M}_P) = \bot$, there is no rule deriving p_i in P. In this case, the i-th row and the i-th column can be removed from $\boldsymbol{M}_P$. For instance, let $\boldsymbol{M}'_P$ be the matrix obtained by removing the 4th row and the 4th column from $\boldsymbol{M}_P$ in Example 3.2. Then $\mathsf{FP}(\boldsymbol{M}'_P \underline{\bullet} \boldsymbol{v}_0)$ produces the least model of P. In particular, when a program P contains no facts (resp. constraints), the row and the column representing $\top$ (resp. $\bot$) could be removed from $\boldsymbol{M}_P$. In this way, we can reduce the size of vector spaces when a program is represented in a sparse matrix/tensor. A technique of dividing a program into subprograms [7] also helps to reduce the size of matrices/tensors in large scale of programs. Tensor LP can also be extended to representing *weight constraints* [12] for ASP. For instance, a weight constraint rule $p_i \leftarrow l \leq \{p_{j_1} = w_{j_1}, \ldots, p_{j_m} = w_{j_m}\}$ ($w_{j_k} \geq 0$; $1 \leq k \leq m$) is represented by a matrix $\boldsymbol{M}_P \in \mathbb{R}^{n\times n}$ such that $\mathsf{row}_i(\boldsymbol{M}_P) = p_i$,

$\mathsf{col}_j(\boldsymbol{M}_P) = p_j$ $(1 \leq i, j \leq n)$, and $a_{ij_k} = w_{j_k}/l$. Given a vector $\boldsymbol{v} = (b_1, \ldots, b_n)^\mathsf{T}$ representing an interpretation, p_i becomes true if the i-th element of $\boldsymbol{M}_P \bullet \boldsymbol{v}$ has a value $b_{j_1} w_{j_1}/l + \cdots + b_{j_m} w_{j_m}/l \geq 1$, thereby computed by $\boldsymbol{M}_P \underset{\bullet}{} \boldsymbol{v}$. In this way, weight constraints are effectively translated into linear algebraic calculation.

5 Conclusion

This paper introduced linear algebraic characterization of logic programs. The result of this paper bridges logic programming and linear algebraic approaches, which would contribute to a step for realizing logical inference in huge scale of knowledge bases. This paper focuses on theoretical aspects of LP in vector spaces. Future research includes implementation and evaluation of the proposed approach as well as development of techniques for robust and scalable inference in a huge scale of programs.

References

1. Brewka, G. Eiter, T., Truszczynski, M. (eds.): Special issue on answer set programming. AI Magazine **37**(3), Fall (2016)
2. Dowling, W. F., Gallier, J. H.: Linear-time algorithms for testing the satisfiability of propositional Horn formulae. J. Logic Program. **1**(3), 267–284 (1984)
3. Fernandez, J. A., Lobo, J., Minker, J., Subrahmanian, V. S.: Disjunctive LP + integrity constraints = stable model semantics. Annals of Mathematics and AI **8**(3-4), 449–474 (1993)
4. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Proc. 5th Int. Conf. and Symp. Logic Programming, pp. 1070–1080, MIT Press (1988)
5. Grefenstette, E.: Towards a formal distributional semantics: simulating logical calculi with tensors. In: Proc. 2nd Joint Conf. Lexical and Computational Semantics, pp. 1–10 (2013)
6. Kolda, T. G., Bader, B. W.: Tensor decompositions and applications. SIAM Review **51**(3), 455–500 (2009)
7. Lifschitz, V., Turner, H.: Splitting a logic program. In: Proc. 5th Int. Conf. Logic Programming, pp. 23–37, MIT Press (1994)
8. Liu, G., Janhunen, T., Niemelä, I.: Answer set programming via mixed integer programming. In: Proc. 13th Int. Conf. Knowledge Representation and Reasoning, pp. 32–42 (2012)
9. Lloyd, J. W.: Foundations of Logic Programming (2nd ed.), Springer, Heidelberg (1987)
10. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. In: Proc. IEEE **104**(1), 11–33 (2016)
11. Rocktaschel, T., Bosnjak, M., Singh, S., Riedel, S.: Low-dimensional embeddings of logic. In: Proc. ACL 2014 Workshop on Semantic Parsing, pp. 45–49 (2014)
12. Simons, P., Niemela, I., Soininen, T.: Extending and implementing the stable model semantics. Artif. Intell. **138**:181–234 (2002)
13. Saraswat, V.: Reasoning 2.0 or machine learning and logic–the beginnings of a new computer science. Data Science Day, Kista Sweden (2016)
14. Sato, T.: Embedding Tarskian semantics in vector spaces. In: Proc. AAAI-17 Workshop on Symbolic Inference and Optimization (2017)
15. Sato, T.: A linear algebraic approach to Datalog evaluation. TPLP, **17**(3), 244–265 (2017)
16. Serafini, L., d'Avila Garcez, A.: Learning and reasoning with logic tensor networks. Advances in Artificial Intelligence, LNCS, vol. 10037, pp. 334–348, Springer, Heidelberg (2016)
17. van Emden, M. H., Kowalski, R. A.: The semantics of predicate logic as a programming language. JACM **23**(4), 733–742 (1976)
18. Yang, B., Yih, W.-t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: Proc. Int. Conf. Learning Representations (2015)