Vector Addition Tree Automata & Multiplicative Exponential Linear Logic

Philippe de Groote, Bruno Guillaume, Sylvain Salvati

LORIA & INRIA Lorraine

September 24, 2004

Overview

- Introduction
- An example
- The automata side
- The linear logic side
- Bridging the two sides

Introduction

Introduction

- There is an encoding of a fragment (!-Horn) of MELL in Petri Nets or Vector Addition System with States (Kanovich).

Introduction

- There is an encoding of a fragment (!-Horn) of MELL in Petri Nets or Vector Addition System with States (Kanovich).
- Our goal is to extend this encoding to full [I]MELL.

Introduction

- There is an encoding of a fragment (!-Horn) of MELL in Petri Nets or Vector Addition System with States (Kanovich).
- Our goal is to extend this encoding to full [I]MELL.
- This yields a reformulation of the (open) problem of the decidability of [I]MELL in an equivalent problem in automata theory.

Introduction

- There is an encoding of a fragment (!-Horn) of MELL in Petri Nets or Vector Addition System with States (Kanovich).
- Our goal is to extend this encoding to full [I]MELL.
- This yields a reformulation of the (open) problem of the decidability of [I]MELL in an equivalent problem in automata theory.

$$
\begin{tikzcd}
\text{!-Horn fragment} \arrow[r, dash] \arrow[d, dash] & \begin{array}{c}\text{Vector Addition}\\ \text{System with States}\\ \simeq \text{Petri Nets}\end{array} \arrow[r, dash] \arrow[d, dash] & \text{Finite State Automata} \arrow[d, dash] \\
\text{IMELL} \arrow[r, dash] & \begin{array}{c}\text{Vector Addition}\\ \text{Tree Automata}\end{array} \arrow[r, dash] & \text{Tree Automata}
\end{tikzcd}
$$

An example

Find a proof in IMELL of

$$!((A \multimap B \multimap S) \multimap S), !(S \multimap S \multimap S), !(A \multimap S), !(B \multimap S) \vdash S$$

An example

Find a proof in IMELL of

$$!((A \multimap B \multimap S) \multimap S), !(S \multimap S \multimap S), !(A \multimap S), !(B \multimap S) \vdash S$$

is equivalent to find a closed linear λ-term of type S on the signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

An example

Find a proof in IMELL of

$$!((A \multimap B \multimap S) \multimap S), !(S \multimap S \multimap S), !(A \multimap S), !(B \multimap S) \vdash S$$

is equivalent to find a closed linear λ-term of type S on the signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

Let's go...

An example

Find a proof in IMELL of

$$!((A \multimap B \multimap S) \multimap S), !(S \multimap S \multimap S), !(A \multimap S), !(B \multimap S) \vdash S$$

is equivalent to find a closed linear λ-term of type S on the signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

Let's go...

$$\begin{array}{rcl} \mathcal{T}_S & ::= & f\ (\lambda xy.\mathcal{T}_S[x : A, y : B]) \\ & \mid & g\ \mathcal{T}_S\ \mathcal{T}_S \end{array}$$

An example

Find a proof in IMELL of

$$!((A \multimap B \multimap S) \multimap S), !(S \multimap S \multimap S), !(A \multimap S), !(B \multimap S) \vdash S$$

is equivalent to find a closed linear λ-term of type S on the signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

Let's go...

$$\begin{array}{rcl}
\mathcal{T}_S & ::= & f\ (\lambda xy.\mathcal{T}_S[x : A, y : B]) \\
& | & g\ \mathcal{T}_S\ \mathcal{T}_S \\
\mathcal{T}_S[x : A, y : B] & ::= & f\ (\lambda x'y'.\mathcal{T}_S[x : A, y : B, x' : A, y' : B]) \\
& | & g\ \mathcal{T}_S\ \mathcal{T}_S[x : A, y : B] \\
& | & g\ \mathcal{T}_S[x : A]\ \mathcal{T}_S[y : B] \\
& | & g\ \mathcal{T}_S[y : B]\ \mathcal{T}_S[x : A] \\
& | & g\ \mathcal{T}_S[x : A, y : B]\ \mathcal{T}_S
\end{array}$$

An example

Find a proof in IMELL of

$$!((A \multimap B \multimap S) \multimap S), !(S \multimap S \multimap S), !(A \multimap S), !(B \multimap S) \vdash S$$

is equivalent to find a closed linear λ-term of type S on the signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

Let's go...

$$\begin{array}{rcl}
\mathcal{T}_S & ::= & f\ (\lambda xy.\mathcal{T}_S[x : A, y : B]) \\
& | & g\ \mathcal{T}_S\ \mathcal{T}_S \\
\mathcal{T}_S[x : A, y : B] & ::= & f\ (\lambda x'y'.\mathcal{T}_S[x : A, y : B, x' : A, y' : B]) \\
& | & g\ \mathcal{T}_S\ \mathcal{T}_S[x : A, y : B] \\
& | & g\ \mathcal{T}_S[x : A]\ \mathcal{T}_S[y : B] \\
& | & g\ \mathcal{T}_S[y : B]\ \mathcal{T}_S[x : A] \\
& | & g\ \mathcal{T}_S[x : A, y : B]\ \mathcal{T}_S \\
\mathcal{T}_S[x : A] & ::= & a\ x \\
& | & \ldots
\end{array}$$

An example

The signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

In fact, <u>only the number of free variables matters.</u>

An example

The signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

In fact, only the number of free variables matters.
Let $\mathbf{x} = (n_1, n_2) \in \mathbb{N}^2$, $\mathcal{T}_S[\mathbf{x}]$ stands for the set of terms of type S with n_1 free variables of type A and n_2 free variables of type B.

An example

The signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

In fact, only the number of free variables matters.
Let $\mathbf{x} = (n_1, n_2) \in \mathbb{N}^2$, $\mathcal{T}_S[\mathbf{x}]$ stands for the set of terms of type S with n_1 free variables of type A and n_2 free variables of type B.

$$\begin{array}{rcll}
\mathcal{T}_S[\mathbf{x}] & ::= & f\ (\lambda_ : A\ \lambda_ : B\ \mathcal{T}_S[\mathbf{x} + (1,1)]) & \\
& | & g\ \mathcal{T}_S[\mathbf{x}_1]\ \mathcal{T}_S[\mathbf{x}_2] & \mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 \\
& | & a\ _ & \text{if } \mathbf{x} = (1,0) \\
& | & b\ _ & \text{if } \mathbf{x} = (0,1)
\end{array}$$

An example

The signature:

$$f : (A \multimap B \multimap S) \multimap S \qquad g : S \multimap S \multimap S \qquad a : A \multimap S \qquad b : B \multimap S$$

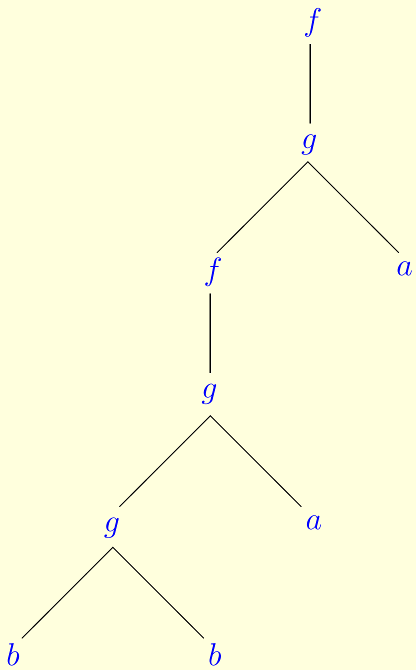
In fact, only the number of free variables matters.
Let $\mathbf{x} = (n_1, n_2) \in \mathbb{N}^2$, $\mathcal{T}_S[\mathbf{x}]$ stands for the set of terms of type S with n_1 free variables of type A and n_2 free variables of type B.

$$\begin{array}{rcll}
\mathcal{T}_S[\mathbf{x}] & ::= & f\ (\lambda_ : A\ \lambda_ : B\ \mathcal{T}_S[\mathbf{x} + (1,1)]) & \\
& \mid & g\ \mathcal{T}_S[\mathbf{x}_1]\ \mathcal{T}_S[\mathbf{x}_2] & \mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 \\
& \mid & a\ _ & \text{if } \mathbf{x} = (1,0) \\
& \mid & b\ _ & \text{if } \mathbf{x} = (0,1)
\end{array}$$

Putting it as a (tree automata like) rewriting system:

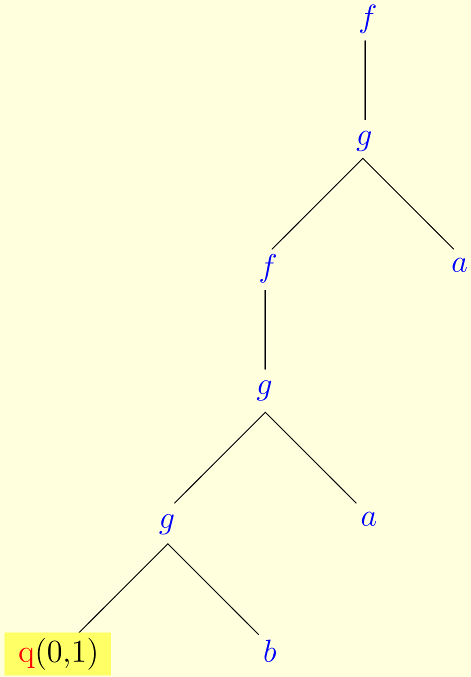
$$\begin{array}{rcl}
a & \to & q[(1,0)] \\
b & \to & q[(0,1)] \\
f(q[\mathbf{x}]) & \to & q[\mathbf{x} - (1,1)] \\
g(q[\mathbf{x}_1], q[\mathbf{x}_2]) & \to & q[\mathbf{x}_1 + \mathbf{x}_2]
\end{array}$$

An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

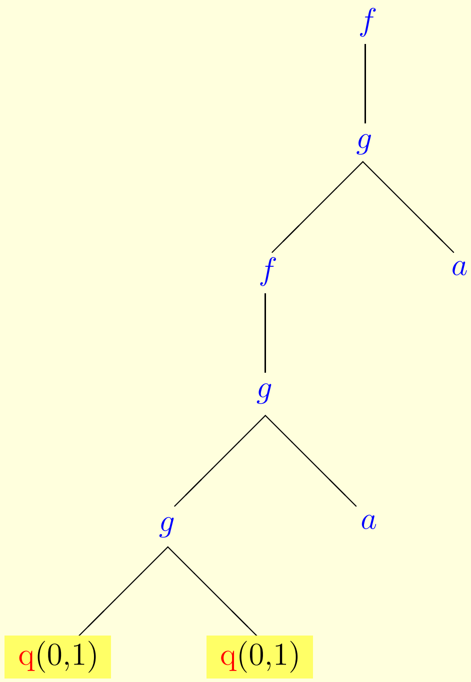
An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$\dfrac{}{!\Gamma, B \vdash S}\ (b)$$

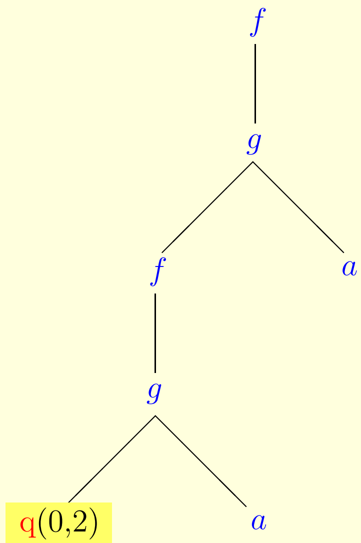
An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$\dfrac{\dfrac{}{!\Gamma, B \vdash S}\,(b) \qquad \dfrac{}{!\Gamma, B \vdash S}\,(b)}{}$$

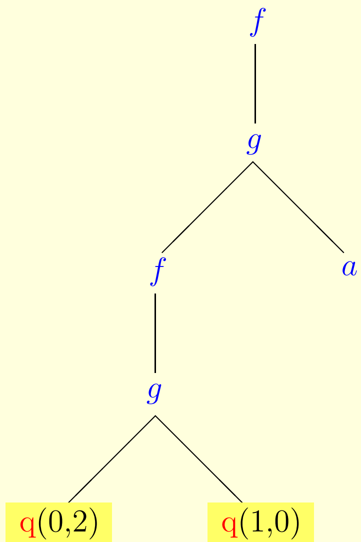
An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$\dfrac{\dfrac{\dfrac{}{!\Gamma, B \vdash S}\,(b) \qquad \dfrac{}{!\Gamma, B \vdash S}\,(b)}{!\Gamma, B, B \vdash S}\,(g)}{}$$

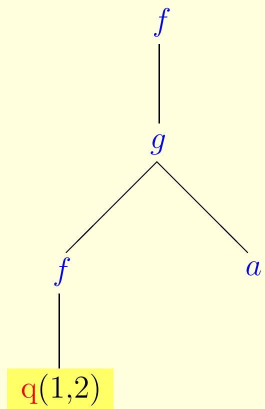
An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$\dfrac{\dfrac{\dfrac{}{!\Gamma, B \vdash S}\,(b) \qquad \dfrac{}{!\Gamma, B \vdash S}\,(b)}{!\Gamma, B, B \vdash S}\,(g) \qquad \dfrac{}{!\Gamma, A \vdash S}\,(a)}{}$$

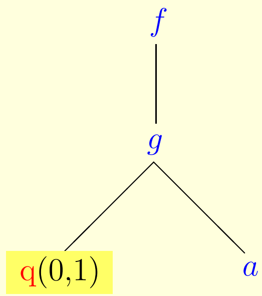
An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$\dfrac{\dfrac{\dfrac{}{!\Gamma, B \vdash S}\,(b) \qquad \dfrac{}{!\Gamma, B \vdash S}\,(b)}{!\Gamma, B, B \vdash S}\,(g) \qquad \dfrac{}{!\Gamma, A \vdash S}\,(a)}{\underline{!\Gamma, A, B, B \vdash S}}\,(g)$$

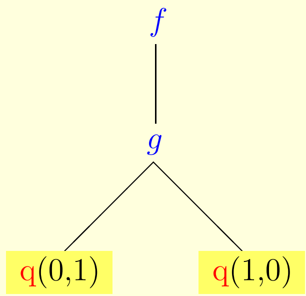
An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{!\Gamma, B \vdash S}\,(b) \qquad \dfrac{}{!\Gamma, B \vdash S}\,(b)}{!\Gamma, B, B \vdash S}\,(g) \qquad \dfrac{}{!\Gamma, A \vdash S}\,(a)}{!\Gamma, A, B, B \vdash S}\,(g)}{\underline{!\Gamma, B \vdash S}}\,(f)$$

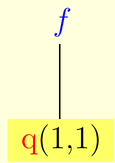
An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{!\Gamma, B \vdash S}\,(b) \qquad \dfrac{}{!\Gamma, B \vdash S}\,(b)}{!\Gamma, B, B \vdash S}\,(g) \qquad \dfrac{}{!\Gamma, A \vdash S}\,(a)}{!\Gamma, A, B, B \vdash S}\,(g)}{!\Gamma, B \vdash S}\,(f) \qquad \dfrac{}{!\Gamma, A \vdash S}\,(a)}{}$$

An example



$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{!\Gamma, B \vdash S}\,(b) \quad \cfrac{}{!\Gamma, B \vdash S}\,(b)}{!\Gamma, B, B \vdash S}\,(g) \quad \cfrac{}{!\Gamma, A \vdash S}\,(a)}{!\Gamma, A, B, B \vdash S}\,(g)}{!\Gamma, B \vdash S}\,(f) \quad \cfrac{}{!\Gamma, A \vdash S}\,(a)}{!\Gamma, A, B \vdash S}\,(g)}{}$$

An example

q(0,0)

$$\Gamma = (A \multimap B \multimap S) \multimap S, S \multimap S \multimap S, A \multimap S, B \multimap S$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{!\Gamma, B \vdash S}\,(b) \quad \dfrac{}{!\Gamma, B \vdash S}\,(b)}{!\Gamma, B, B \vdash S}\,(g) \quad \dfrac{}{!\Gamma, A \vdash S}\,(a)}{!\Gamma, A, B, B \vdash S}\,(g)}{!\Gamma, B \vdash S}\,(f) \quad \dfrac{}{!\Gamma, A \vdash S}\,(a)}{!\Gamma, A, B \vdash S}\,(g)}{!\Gamma \vdash S}\,(f)
$$

Vector Addition Tree Automata

A k-VATA is a quadruple $\langle \mathcal{F}, Q, C_f, \Delta \rangle$ where:

- $\mathcal{F}$ is a ranked alphabet;
- Q is a finite set of states;
- C_f is a finite set of accepting configurations (i.e. elements of $Q \times \mathbb{N}^k$);
- Δ is a finite set of transition rules of the form:

$$f(q_0[\mathbf{x}_0], \ldots, q_{n-1}[\mathbf{x}_{n-1}]) \rightarrow q\left[\sum_{i \in n} (\mathbf{x}_i - \mathbf{c}_i) + \mathbf{c}\right]$$

Vector Addition Tree Automata

A k-VATA is a quadruple $\langle \mathcal{F}, Q, C_f, \Delta \rangle$ where:

- $\mathcal{F}$ is a ranked alphabet;
- Q is a finite set of states;
- C_f is a finite set of accepting configurations (i.e. elements of $Q \times \mathbb{N}^k$);
- Δ is a finite set of transition rules of the form:

$$f(q_0[\mathbf{x}_0], \ldots, q_{n-1}[\mathbf{x}_{n-1}]) \to q\left[\sum_{i \in n} (\mathbf{x}_i - \mathbf{c}_i) + \mathbf{c}\right]$$

where:

Vector Addition Tree Automata

A k-VATA is a quadruple $\langle \mathcal{F}, Q, C_f, \Delta \rangle$ where:

- $\mathcal{F}$ is a ranked alphabet;
- Q is a finite set of states;
- C_f is a finite set of accepting configurations (i.e. elements of $Q \times \mathbb{N}^k$);
- Δ is a finite set of transition rules of the form:

$$f(q_0[\mathbf{x}_0], \ldots, q_{n-1}[\mathbf{x}_{n-1}]) \to q\left[\sum_{i \in n} (\mathbf{x}_i - \mathbf{c}_i) + \mathbf{c}\right]$$

where:

$f \in \mathcal{F}$ is a functional symbol of arity n;

$q_0 \ldots q_{n-1} \in Q$;

$\mathbf{c}, \mathbf{c}_0, \ldots \mathbf{c}_{n-1} \in \mathbb{N}^k$ are given vectors, proper to the transition rule;

$\mathbf{x}_0, \ldots \mathbf{x}_{n-1}$ are variables in $\mathbb{N}^k$.

Vector Addition Tree Automata

A k-VATA is a quadruple $\langle \mathcal{F}, Q, C_f, \Delta \rangle$ where:

- $\mathcal{F}$ is a ranked alphabet;
- Q is a finite set of states;
- C_f is a finite set of accepting configurations (i.e. elements of $Q \times \mathbb{N}^k$);
- Δ is a finite set of transition rules of the form:

$$f(q_0[\mathbf{x}_0], \ldots, q_{n-1}[\mathbf{x}_{n-1}]) \rightarrow q\left[\sum_{i \in n}(\mathbf{x}_i - \mathbf{c}_i) + \mathbf{c}\right]$$

where:

$f \in \mathcal{F}$ is a functional symbol of arity n;

$q_0 \ldots q_{n-1} \in Q$;

$\mathbf{c}, \mathbf{c}_0, \ldots \mathbf{c}_{n-1} \in \mathbb{N}^k$ are given vectors, proper to the transition rule;

$\mathbf{x}_0, \ldots \mathbf{x}_{n-1}$ are variables in $\mathbb{N}^k$.

The rewriting relation is induced by the transition rules with the constraint that $\mathbf{x}_i - \mathbf{c}_i \in \mathbb{N}^k$ (this corresponds to the positivity condition in VASS).

Vector Addition Tree Automata under Normal Form

A k-VATA is a normal form if

- it is deterministic;

Vector Addition Tree Automata under Normal Form

A k-VATA is a normal form if

- it is deterministic;
- $C_f = (q_f, \mathbf{0})$;

Vector Addition Tree Automata under Normal Form

A k-VATA is a normal form if

- it is deterministic;
- $C_f = (q_f, \mathbf{0})$;
- each transition rule is in one of the three forms:

$$\begin{array}{rcll} f & \to & q[\mathbf{e}_i] & \text{for some } i \in k \\ f(q_0[\mathbf{x}_0]) & \to & q[\mathbf{x}_0 - \mathbf{e}_i] & \text{for some } i \in k \\ f(q_0[\mathbf{x}_0], q_1[\mathbf{x}_1]) & \to & q[\mathbf{x}_0 + \mathbf{x}_1] & \end{array}$$

Vector Addition Tree Automata under Normal Form

A k-VATA is a normal form if

- it is deterministic;
- $C_f = (q_f, \mathbf{0})$;
- each transition rule is in one of the three forms:

$$
\begin{array}{rcll}
f & \to & q[\mathbf{e}_i] & \text{for some } i \in k \\
f(q_0[\mathbf{x}_0]) & \to & q[\mathbf{x}_0 - \mathbf{e}_i] & \text{for some } i \in k \\
f(q_0[\mathbf{x}_0], q_1[\mathbf{x}_1]) & \to & q[\mathbf{x}_0 + \mathbf{x}_1] &
\end{array}
$$

Proposition 1 *For any k-VATA $\mathcal{A}$ there exists a k-VATA $\mathcal{A}'$ in normal form such that $\mathcal{L}_{\mathcal{A}} = \emptyset$ iff $\mathcal{L}_{\mathcal{A}'} = \emptyset$.*

Linear Logic

IMELL

$$\mathcal{F} \quad ::= \quad \mathbf{1} \mid \mathcal{A} \mid \mathcal{F} \otimes \mathcal{F} \mid \mathcal{F} \multimap \mathcal{F} \mid\ !\mathcal{F}$$

Linear Logic

IMELL

$$\mathcal{F} \quad ::= \quad \mathbf{1} \mid \mathcal{A} \mid \mathcal{F} \otimes \mathcal{F} \mid \mathcal{F} \multimap \mathcal{F} \mid\ !\mathcal{F}$$

IMELL$_0$

$$\begin{array}{lcl} \mathcal{F}_0 & ::= & \mathcal{M} \mid\ !\mathcal{M} \\ \mathcal{M} & ::= & \mathbf{1} \mid \mathcal{A} \mid \mathcal{M} \otimes \mathcal{M} \mid \mathcal{M} \multimap \mathcal{M} \end{array}$$

Linear Logic

IMELL

$$\mathcal{F} \quad ::= \quad \mathbf{1} \mid \mathcal{A} \mid \mathcal{F} \otimes \mathcal{F} \mid \mathcal{F} \multimap \mathcal{F} \mid !\mathcal{F}$$

IMELL_0

$$\begin{array}{lcl} \mathcal{F}_0 & ::= & \mathcal{M} \mid !\mathcal{M} \\ \mathcal{M} & ::= & \mathbf{1} \mid \mathcal{A} \mid \mathcal{M} \otimes \mathcal{M} \mid \mathcal{M} \multimap \mathcal{M} \end{array}$$

$\mathrm{IMELL}_0^{\multimap}$

$$\begin{array}{lcl} \mathcal{F}_0^{\multimap} & ::= & \mathcal{M} \mid !\mathcal{M} \\ \mathcal{M} & ::= & \mathcal{A} \mid \mathcal{M} \multimap \mathcal{M} \end{array}$$

Linear Logic

IMELL

$$\mathcal{F} \quad ::= \quad \mathbf{1} \mid \mathcal{A} \mid \mathcal{F} \otimes \mathcal{F} \mid \mathcal{F} \multimap \mathcal{F} \mid !\mathcal{F}$$

IMELL$_0$

$$\begin{array}{lcl} \mathcal{F}_0 & ::= & \mathcal{M} \mid !\mathcal{M} \\ \mathcal{M} & ::= & \mathbf{1} \mid \mathcal{A} \mid \mathcal{M} \otimes \mathcal{M} \mid \mathcal{M} \multimap \mathcal{M} \end{array}$$

IMELL$_0^{\multimap}$

$$\begin{array}{lcl} \mathcal{F}_0^{\multimap} & ::= & \mathcal{M} \mid !\mathcal{M} \\ \mathcal{M} & ::= & \mathcal{A} \mid \mathcal{M} \multimap \mathcal{M} \end{array}$$

s-IMELL$_0^{\multimap}$

$$\text{s-}\mathcal{F}_0^{\multimap} \quad ::= \quad \mathcal{A} \mid !(\mathcal{A} \multimap \mathcal{A}) \mid !(\mathcal{A} \multimap \mathcal{A} \multimap \mathcal{A}) \mid !((\mathcal{A} \multimap \mathcal{A}) \multimap \mathcal{A})$$

From IMELL to IMELL$_0$

Proposition 2 *IMELL is decidable iff IMELL$_0$ is decidable.*

From IMELL to IMELL$_0$

Proposition 2 *IMELL is decidable iff IMELL$_0$ is decidable.*

Proof

Let $\Gamma \vdash A$ an IMELL sequent.

From IMELL to IMELL$_0$

Proposition 2 *IMELL is decidable iff IMELL$_0$ is decidable.*

Proof

Let $\Gamma \vdash A$ an IMELL sequent.

Define $\Gamma^* \vdash A^*$ to be the sequent obtained by replacing each exponential subformula $!F$ by a fresh atomic proposition p_F.

From IMELL to IMELL$_0$

Proposition 2 *IMELL is decidable iff IMELL$_0$ is decidable.*

Proof

Let $\Gamma \vdash A$ an IMELL sequent.

Define $\Gamma^* \vdash A^*$ to be the sequent obtained by replacing each exponential subformula $!F$ by a fresh atomic proposition p_F.

Add Σ, the following set of formulas to the antecedent of the sequent:

$!(p_F \multimap F^*)$,

$!(p_F \multimap p_F \otimes p_F)$,

$!(p_F \multimap \mathbf{1})$,

$!(p_{F_1} \multimap \cdots \multimap p_{F_n} \multimap p_F)$ whenever $p_{F_1} \multimap \cdots \multimap p_{F_n} \multimap F^*$ is provable.

From IMELL to IMELL$_0$

Proposition 2 *IMELL is decidable iff IMELL$_0$ is decidable.*

Proof

Let $\Gamma \vdash A$ an IMELL sequent.

Define $\Gamma^* \vdash A^*$ to be the sequent obtained by replacing each exponential subformula $!F$ by a fresh atomic proposition p_F.

Add Σ, the following set of formulas to the antecedent of the sequent:

$!(p_F \multimap F^*)$,

$!(p_F \multimap p_F \otimes p_F)$,

$!(p_F \multimap \mathbf{1})$,

$!(p_{F_1} \multimap \cdots \multimap p_{F_n} \multimap p_F)$ whenever $p_{F_1} \multimap \cdots \multimap p_{F_n} \multimap F^*$ is provable.

- If IMELL$_0$ is decidable then the construction is effective.

From IMELL to IMELL$_0$

Proposition 2 *IMELL is decidable iff IMELL$_0$ is decidable.*

Proof

Let $\Gamma \vdash A$ an IMELL sequent.

Define $\Gamma^* \vdash A^*$ to be the sequent obtained by replacing each exponential subformula $!F$ by a fresh atomic proposition p_F.

Add Σ, the following set of formulas to the antecedent of the sequent:

$!(p_F \multimap F^*)$,

$!(p_F \multimap p_F \otimes p_F)$,

$!(p_F \multimap \mathbf{1})$,

$!(p_{F_1} \multimap \cdots \multimap p_{F_n} \multimap p_F)$ whenever $p_{F_1} \multimap \cdots \multimap p_{F_n} \multimap F^*$ is provable.

- If IMELL$_0$ is decidable then the construction is effective.
- $\Gamma \vdash A$ is provable if and only if $\Sigma, \Gamma^* \vdash A^*$ is provable. $\square$

From $\mathbf{IMELL}_0$ to $\mathbf{IMELL}_0^{\multimap}$

Proposition 3 *$IMELL_0$ is decidable iff $IMELL_0^{\multimap}$ is decidable.*

From $\mathbf{IMELL}_0$ to $\mathbf{IMELL}_0^{\multimap}$

Proposition 3 *$IMELL_0$ is decidable iff $IMELL_0^{\multimap}$ is decidable.*

Proof Let b a fresh atomic proposition, we define two translation on multiplicative formulas:

$A^+ = A^- \multimap b$, for any formula A

$\mathbf{1}^- = b$

$a^- = a \multimap b$

$(A \otimes B)^- = A^+ \multimap (B^+ \multimap b)$

$(A \multimap B)^- = (A^+ \multimap B^+) \multimap b$

and extend it to IMELL_0 with $(!A)^+ = !(A^+)$.

From IMELL_0 to $\mathrm{IMELL}_0^{\multimap}$

Proposition 3 *IMELL_0 is decidable iff $\mathrm{IMELL}_0^{\multimap}$ is decidable.*

Proof Let b a fresh atomic proposition, we define two translation on multiplicative formulas:

$A^+ = A^- \multimap b$, for any formula A

$\mathbf{1}^- = b$

$a^- = a \multimap b$

$(A \otimes B)^- = A^+ \multimap (B^+ \multimap b)$

$(A \multimap B)^- = (A^+ \multimap B^+) \multimap b$

and extend it to IMELL_0 with $(!A)^+ = !(A^+)$.

- $\Gamma \vdash A$ an IMELL_0 sequent is provable iff $\Gamma^+ \vdash A^+$ is provable. $\square$

From $\mathbf{IMELL}_0^{\multimap}$ to s-$\mathbf{IMELL}_0^{\multimap}$

Proposition 4 *$IMELL_0^{\multimap}$ is decidable iff s-$IMELL_0^{\multimap}$ is decidable.*

From IMELL$_0^{\multimap}$ to s-IMELL$_0^{\multimap}$

Proposition 4 *IMELL$_0^{\multimap}$ is decidable iff s-IMELL$_0^{\multimap}$ is decidable.*

Proof

Lemma 1 *Let $\Gamma \vdash A$ an IMELL sequent.*

$$\underbrace{\ldots F \ldots F \ldots}_{\Gamma} \vdash \underbrace{\ldots F \ldots}_{A} \iff !(p \multimap F), !(F \multimap p), \underbrace{\ldots p \ldots p \ldots}_{\Gamma} \vdash \underbrace{\ldots p \ldots}_{A}$$

From IMELL$_0^{\multimap}$ to s-IMELL$_0^{\multimap}$

Proposition 4 *IMELL$_0^{\multimap}$ is decidable iff s-IMELL$_0^{\multimap}$ is decidable.*

Proof

Lemma 1 *Let $\Gamma \vdash A$ an IMELL sequent.*

$$\underbrace{\ldots F \ldots F \ldots}_{\Gamma} \vdash \underbrace{\ldots F \ldots}_{A} \iff\ !(p \multimap F), !(F \multimap p), \underbrace{\ldots p \ldots p \ldots}_{\Gamma} \vdash \underbrace{\ldots p \ldots}_{A}$$

$$!\Sigma, \Gamma \vdash A \text{ is a provable IMELL}_0^{\multimap} \text{ sequent}$$
$$\Updownarrow$$
$$!\Sigma', \underbrace{\Gamma'}_{atomic} \vdash a \text{ is a provable IMELL}_0^{\multimap} \text{ sequent}$$
$$\Updownarrow$$
$$!\underbrace{\Sigma''}_{\leq 2 `\multimap`}, \underbrace{\Gamma'}_{atomic} \vdash a \text{ is a provable IMELL}_0^{\multimap} \text{ sequent}$$

$\square$

From VATA to IMELL

Let $\mathcal{A} = \langle \mathcal{F}, Q, \{(q_f, \mathbf{0})\}, \Delta \rangle$ a k-VATA in normal form.

From VATA to IMELL

Let $\mathcal{A} = \langle \mathcal{F}, Q, \{(q_f, \mathbf{0})\}, \Delta \rangle$ a k-VATA in normal form.
We define the set of atomic types $A = Q \cup \{a_0, \ldots, a_{k-1}\}$ and Σ by:

$$\begin{array}{rcl}
\Delta & & \Sigma \\
f \rightarrow q[\mathbf{e}_i] & \rightsquigarrow & a_i \multimap q \\
f(q_0[\mathbf{x}_0]) \rightarrow q[\mathbf{x}_0 - \mathbf{e}_i] & \rightsquigarrow & (a_i \multimap q_0) \multimap q \\
f(q_0[\mathbf{x}_0], q_1[\mathbf{x}_1]) \rightarrow q[\mathbf{x}_0 + \mathbf{x}_1] & \rightsquigarrow & q_0 \multimap q_1 \multimap q
\end{array}$$

From VATA to IMELL

Let $\mathcal{A} = \langle \mathcal{F}, Q, \{(q_f, \mathbf{0})\}, \Delta \rangle$ a k-VATA in normal form.
We define the set of atomic types $A = Q \cup \{a_0, \ldots, a_{k-1}\}$ and Σ by:

$$\begin{array}{rcl}
\Delta & & \Sigma \\
f \to q[\mathbf{e}_i] & \rightsquigarrow & a_i \multimap q \\
f(q_0[\mathbf{x}_0]) \to q[\mathbf{x}_0 - \mathbf{e}_i] & \rightsquigarrow & (a_i \multimap q_0) \multimap q \\
f(q_0[\mathbf{x}_0], q_1[\mathbf{x}_1]) \to q[\mathbf{x}_0 + \mathbf{x}_1] & \rightsquigarrow & q_0 \multimap q_1 \multimap q
\end{array}$$

Proposition 5 $\mathcal{L}(\mathcal{A}) \neq \varnothing$ *iff* $!\Sigma \vdash q_f$.

From IMELL to VATA

Let $!\Sigma, \Gamma \vdash a_0$ an s-IMELL$_0^{\multimap}$ sequent and $\{a_0, \ldots, a_{k-1}\}$ an enumeration of the atomic formulas of the sequent.

From IMELL to VATA

Let $!\Sigma, \Gamma \vdash a_0$ an s-IMELL$_0^{\multimap}$ sequent and $\{a_0, \ldots, a_{k-1}\}$ an enumeration of the atomic formulas of the sequent. We define the set of state $Q = \{q_0, \ldots, q_{k-1}\}$, the only final configuration $(q_0, |\Gamma|)$ and the set of transitions:

$$
\begin{array}{rcl}
\Sigma & & \Delta \\
 & \rightsquigarrow & c_i \longrightarrow q_i[\mathrm{e}_i] \\
a_j \multimap a_l & \rightsquigarrow & f(q_j[\mathbf{x}]) \longrightarrow q_l[\mathbf{x}] \\
(a_j \multimap a_l) \multimap a_m & \rightsquigarrow & g(q_l[\mathbf{x}]) \longrightarrow q_m[\mathbf{x} - \mathrm{e}_j] \\
a_j \multimap a_l \multimap a_m & \rightsquigarrow & h(q_j[\mathbf{x}_0], q_l[\mathbf{x}_1]) \longrightarrow q_m[\mathbf{x}_0 + \mathbf{x}_1]
\end{array}
$$

From IMELL to VATA

Let $!\Sigma, \Gamma \vdash a_0$ an s-IMELL$_0^{\multimap}$ sequent and $\{a_0, \ldots, a_{k-1}\}$ an enumeration of the atomic formulas of the sequent. We define the set of state $Q = \{q_0, \ldots, q_{k-1}\}$, the only final configuration $(q_0, |\Gamma|)$ and the set of transitions:

$$
\begin{array}{rcl}
\Sigma & & \Delta \\
 & \rightsquigarrow & c_i \longrightarrow q_i[\mathrm{e}_i] \\
a_j \multimap a_l & \rightsquigarrow & f(q_j[\mathbf{x}]) \longrightarrow q_l[\mathbf{x}] \\
(a_j \multimap a_l) \multimap a_m & \rightsquigarrow & g(q_l[\mathbf{x}]) \longrightarrow q_m[\mathbf{x} - \mathrm{e}_j] \\
a_j \multimap a_l \multimap a_m & \rightsquigarrow & h(q_j[\mathbf{x}_0], q_l[\mathbf{x}_1]) \longrightarrow q_m[\mathbf{x}_0 + \mathbf{x}_1]
\end{array}
$$

Proposition 6 $!\Sigma, \Gamma \vdash a_0$ *is provable in s-IMELL$_0^{\multimap}$ iff* $\mathcal{L}(\mathcal{A}) \neq \varnothing$.

Conclusion and future work

Conclusion and future work

- We have so far:

  Decidability of MELL is equivalent to decidability of reachability in VATA

Conclusion and future work

- We have so far:

  Decidability of MELL is equivalent to decidability of reachability in VATA

- Future work:

  Prove the decidability of MELL