A Verified Reduction Algorithm from MLSSmf to MLSS

Yiran Duan, Lukas Stevens

September 1, 2025

Abstract

Multi-level syllogistic with monotone functions (**MLSSmf**) is a sublanguage of set theory introduced by Cantone et al. [1], involving set-to-set functions and their monotonicity, additivity, and multiplicativity. It is an extension of *multi-level syllogistic with singleton* (**MLSS**), which involves the predicates membership, set equality, set inclusion, and the operators union, intersection, set difference, and singleton.

In this work we formalize the reduction algorithm from **MLSSmf** to **MLSS**, and verify the correctness proof originally presented by Cantone et al. [1]. Combined with the verified decision procedure for **MLSS** formalized by Stevens [2], this yields an executable and verified decision procedure for **MLSSmf**.

theory *MLSSmf-to-MLSS-Complexity*
imports *MLSSmf-to-MLSS*
begin

definition $size_m$ :: ('v, 'f) *MLSSmf-clause* ⇒ *nat* **where**
$size_m\ \mathcal{C} \equiv card\ (set\ \mathcal{C})$

lemma (**in** *normalized-MLSSmf-clause*) *card-V-upper-bound*:
$card\ V \leq 3 * size_m\ \mathcal{C}$
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *card-F-upper-bound*:
$card\ F \leq 2 * size_m\ \mathcal{C}$
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-restriction-on-InterOfVars*:
card (*restriction-on-InterOfVars vs*) ≤ *2* ∗ *length vs*
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-restriction-on-UnionOfVars*:
card (*restriction-on-UnionOfVars vs*) ≤ *Suc* (*length vs*)
⟨*proof*⟩

theorem (**in** *normalized-MLSSmf-clause*) *size-introduce-v*:
card introduce-v ≤ (*3* ∗ *card V* + *2*) ∗ (*2* ^ *card V*)
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-restriction-on-UnionOfVennRegions*:
card (*restriction-on-UnionOfVennRegions* αs) ≤ *Suc* (*length* αs)
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *length-all-V-set-lists*:
length all-V-set-lists = *2* ^ *card* (P^+ *V*)
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *length-F-list*:
length F-list = *card F*
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-introduce-UnionOfVennRegions*:
card introduce-UnionOfVennRegions ≤ *Suc* (*2* ^ *card V*) ∗ *2* ^ *2* ^ *card V*
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *length-choices-from-lists*:
∀ *choice* ∈ *set* (*choices-from-lists xss*). *length choice* = *length xss*
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-introduce-w*:
∀ *clause* ∈ *introduce-w*. *card clause* ≤ *2* ^(*2* ∗ *2* ^ *card V*) ∗ *card F*

⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *card-P-P-V-ge-1*:
card (*Pow* (P^+ *V*) × *Pow* (P^+ *V*)) ≥ *1*
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-reduce-norm-literal*:
assumes *norm-literal lt*
shows *card* (*reduce-literal lt*) ≤ *2* ∗ *card* (*Pow* (P^+ *V*) × *Pow* (P^+ *V*))
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-reduce-clause*:
card reduce-clause ≤ *2* ^ (*Suc* (*2* ∗ *2* ^ *card V*)) ∗ $size_m$ $\mathcal{C}$
⟨*proof*⟩

theorem (**in** *normalized-MLSSmf-clause*) *size-reduced-dnf*:
∀ *clause* ∈ *reduced-dnf*. *card clause* ≤
2 ^ (*2* ∗ *2* ^ (*3* ∗ $size_m$ $\mathcal{C}$)) ∗ (*2* ∗ $size_m$ $\mathcal{C}$) +
(*3* ∗ (*3* ∗ $size_m$ $\mathcal{C}$) + *2*) ∗ (*2* ^ (*3* ∗ $size_m$ $\mathcal{C}$)) +
Suc (*2* ^ (*3* ∗ $size_m$ $\mathcal{C}$)) ∗ *2* ^ *2* ^ (*3* ∗ $size_m$ $\mathcal{C}$) +
2 ^ (*Suc* (*2* ∗ *2* ^ (*3* ∗ $size_m$ $\mathcal{C}$))) ∗ $size_m$ $\mathcal{C}$
⟨*proof*⟩

end
theory *MLSSmf-to-MLSS-Soundness*
imports *MLSSmf-to-MLSS MLSSmf-Semantics Proper-Venn-Regions MLSSmf-HF-Extras*
begin

locale *satisfiable-normalized-MLSSmf-clause* =
normalized-MLSSmf-clause $\mathcal{C}$ **for** $\mathcal{C}$ :: (′*v*, ′*f*) *MLSSmf-clause* +
fixes M_v :: ′*v* ⇒ *hf*
and M_f :: ′*f* ⇒ *hf* ⇒ *hf*
assumes *model-for-*$\mathcal{C}$: I_{cl} M_v M_f $\mathcal{C}$
begin

interpretation *proper-Venn-regions V* M_v
⟨*proof*⟩

function $\mathcal{M}$ :: (′*v*, ′*f*) *Composite* ⇒ *hf* **where**
$\mathcal{M}$ (*Solo x*) = M_v *x*
| $\mathcal{M}$ (v_α) = *proper-Venn-region* α
| $\mathcal{M}$ (*UnionOfVennRegions xss*) = ⨆ *HF* (($\mathcal{M}$ ∘ *VennRegion*) ‘ *set xss*)
| $\mathcal{M}$ (w_{fl}) = (M_f *f*) ($\mathcal{M}$ (*UnionOfVennRegions* (*var-set-set-to-var-set-list l*)))
| $\mathcal{M}$ (*UnionOfVars xs*) = ⨆ *HF* (M_v ‘ *set xs*)
| $\mathcal{M}$ (*InterOfVars xs*) = ⨅ *HF* (M_v ‘ *set xs*)
| $\mathcal{M}$ (*MemAux x*) = *HF* {M_v *x*}
| $\mathcal{M}$ (*InterOfWAux f l m*) = $\mathcal{M}$ w_{fl} − $\mathcal{M}$ w_{fm}
| $\mathcal{M}$ (*InterOfVarsAux xs*) = M_v (*hd xs*) − $\mathcal{M}$ (*InterOfVars* (*tl xs*))
⟨*proof*⟩

termination
⟨*proof*⟩

lemma *soundness-restriction-on-InterOfVars*:
assumes *set xs* $\in P^+ \ V$
shows $\forall a \in$ *restriction-on-InterOfVars xs.* I_{sa} $\mathcal{M}$ *a*
⟨*proof*⟩

lemma *soundness-restriction-on-UnionOfVars*:
assumes *set xs* $\in$ *Pow V*
shows $\forall a \in$ *restriction-on-UnionOfVars xs.* I_{sa} $\mathcal{M}$ *a*
⟨*proof*⟩

lemma *soundness-introduce-v*:
$\forall$ *fml* $\in$ *introduce-v. interp* I_{sa} $\mathcal{M}$ *fml*
⟨*proof*⟩

lemma *soundness-restriction-on-UnionOfVennRegions*:
assumes *set* $\alpha s \in$ *Pow* (*Pow V*)
shows $\forall a \in$ *restriction-on-UnionOfVennRegions* αs. I_{sa} $\mathcal{M}$ *a*
⟨*proof*⟩

lemma *soundness-introduce-UnionOfVennRegions*:
$\forall$ *lt* $\in$ *introduce-UnionOfVennRegions. interp* I_{sa} $\mathcal{M}$ *lt*
⟨*proof*⟩

lemma *soundness-restriction-on-FunOfUnionOfVennRegions*:
assumes l'-l: $l' =$ *var-set-set-to-var-set-list l*
and m'-m: $m' =$ *var-set-set-to-var-set-list m*
shows $\exists$ *lt* $\in$ *set* (*restriction-on-FunOfUnionOfVennRegions* l' m' *f*). *interp* I_{sa} $\mathcal{M}$ *lt*
⟨*proof*⟩

lemma *soundness-introduce-w*:
$\exists$ *clause* $\in$ *introduce-w.* $\forall$ *lt* $\in$ *clause. interp* I_{sa} $\mathcal{M}$ *lt*
⟨*proof*⟩

lemma *soundness-reduce-literal*:
assumes *lt* $\in$ *set* $\mathcal{C}$
shows $\forall$ *fml* $\in$ *reduce-literal lt. interp* I_{sa} $\mathcal{M}$ *fml*
⟨*proof*⟩

lemma *soundness-reduce-cl*:
$\forall$ *fml* $\in$ *reduce-clause. interp* I_{sa} $\mathcal{M}$ *fml*
⟨*proof*⟩

lemma $\mathcal{M}$*-is-model-for-reduced-dnf*: *is-model-for-reduced-dnf* $\mathcal{M}$
⟨*proof*⟩

end

lemma *MLSSmf-to-MLSS-soundness*:
  assumes $\mathcal{C}$*-norm*: *norm-clause* $\mathcal{C}$
    and $\mathcal{C}$*-has-model*: $\exists\, M_v\ M_f.\ I_{cl}\ M_v\ M_f\ \mathcal{C}$
  shows $\exists\, M.$ *normalized-MLSSmf-clause.is-model-for-reduced-dnf* $\mathcal{C}$ M
⟨*proof*⟩

end
theory *Reduced-MLSS-Formula-Singleton-Model-Property*
  imports *Syntactic-Description Place-Realisation MLSSmf-to-MLSS*
begin

locale *satisfiable-normalized-MLSS-clause-with-vars-for-proper-Venn-regions* =
  satisfiable-normalized-MLSS-clause $\mathcal{C}$ $\mathcal{A}$ **for** $\mathcal{C}$ $\mathcal{A}$ +
  fixes $U :: 'a\ set$
    — The collection of variables representing the proper Venn regions of the "original" variable set of the MLSSmf clause
  assumes *U-subset-V*: $U \subseteq V$
    and *no-overlap-within-U*: $[\![u_1 \in U;\ u_2 \in U;\ u_1 \neq u_2]\!] \Longrightarrow \mathcal{A}\ u_1 \sqcap \mathcal{A}\ u_2 = 0$
    and *U-collect-places-neq*: $AF\ (Var\ x =_s\ Var\ y) \in \mathcal{C} \Longrightarrow$
      $\exists\, L\ M.\ L \subseteq U \wedge M \subseteq U \wedge \mathcal{A}\ x = \bigsqcup HF\ (\mathcal{A}\ `\ L) \wedge \mathcal{A}\ y = \bigsqcup HF\ (\mathcal{A}\ `\ M)$
    and *U-collect-places-single*: $AT\ (Var\ x =_s\ Single\ (Var\ y)) \in \mathcal{C} \Longrightarrow$
      $\exists\, L\ M.\ L \subseteq U \wedge M \subseteq U \wedge \mathcal{A}\ x = \bigsqcup HF\ (\mathcal{A}\ `\ L) \wedge \mathcal{A}\ y = \bigsqcup HF\ (\mathcal{A}\ `\ M)$
begin

interpretation $\mathfrak{B}$: *adequate-place-framework* $\mathcal{C}$ PI at_p
  ⟨*proof*⟩

lemma *fact-1*:
  assumes $u_1 \in U$
    and $u_2 \in U$
    and $u_1 \neq u_2$
    and $\pi \in PI$
  shows $\neg\ (\pi\ u_1 \wedge \pi\ u_2)$
⟨*proof*⟩

fun *place-eq* :: $('a \Rightarrow bool) \Rightarrow ('a \Rightarrow bool) \Rightarrow bool$ **where**
  place-eq $\pi_1\ \pi_2 \longleftrightarrow (\forall\, x \in V.\ \pi_1\ x = \pi_2\ x)$

fun *place-sim* :: $('a \Rightarrow bool) \Rightarrow ('a \Rightarrow bool) \Rightarrow bool$ (**infixl** $\sim$ *50*) **where**
  place-sim $\pi_1\ \pi_2 \longleftrightarrow$ *place-eq* $\pi_1\ \pi_2 \vee (\exists\, u \in U.\ \pi_1\ u \wedge \pi_2\ u)$

abbreviation *rel-place-sim* $\equiv \{(\pi_1, \pi_2) \in PI \times PI.\ \pi_1 \sim \pi_2\}$

lemma *place-sim-rel-equiv-on-PI*: *equiv PI rel-place-sim*
⟨*proof*⟩

lemma *refl-sim*:

assumes $a \in PI$
and $b \in PI$
and $a \sim b$
shows $b \sim a$
⟨*proof*⟩

lemma *trans-sim*:
assumes $a \in PI$
and $b \in PI$
and $c \in PI$
and $a \sim b$
and $b \sim c$
shows $a \sim c$
⟨*proof*⟩

lemma *fact-2*:
assumes $x \in V$
and *exL*: $\exists L \subseteq U.\ \mathcal{A}\ x = \bigsqcup HF\ (\mathcal{A}\ `\ L)$
and $\pi_1 \in PI$
and $\pi_2 \in PI$
and $\pi_1 \sim \pi_2$
shows $\pi_1\ x \longleftrightarrow \pi_2\ x$
⟨*proof*⟩

lemma *U-collect-places-single'*: $y \in W \Longrightarrow \exists L.\ L \subseteq U \wedge \mathcal{A}\ y = \bigsqcup HF\ (\mathcal{A}\ `\ L)$
⟨*proof*⟩

definition PI' :: $('a \Rightarrow bool)\ set$ **where**
$PI' \equiv (\lambda \pi s.\ SOME\ \pi.\ \pi \in \pi s)\ `\ (PI\ //\ rel\text{-}place\text{-}sim)$

definition *rep* :: $('a \Rightarrow bool) \Rightarrow ('a \Rightarrow bool)$ **where**
$rep\ \pi = (SOME\ \pi'.\ \pi' \in rel\text{-}place\text{-}sim\ ``\ \{\pi\})$

lemma *range-rep*:
assumes $\pi \in PI$
shows $rep\ \pi \in PI'$
⟨*proof*⟩

lemma *PI'-eq-image-of-rep-on-PI*: $PI' = rep\ `\ PI$
⟨*proof*⟩

lemma *rep-sim*:
assumes $\pi \in PI$
shows $\pi \sim rep\ \pi$
and $rep\ \pi \sim \pi$
⟨*proof*⟩

lemma *PI'-subset-PI*: $PI' \subseteq PI$
⟨*proof*⟩

lemma *sim-self*:
  assumes $\pi \in PI'$
    and $\pi' \in PI'$
    and $\pi \sim \pi'$
  shows $\pi' = \pi$
⟨*proof*⟩

fun $at_p\text{-}f'$ :: $'a \Rightarrow ('a \Rightarrow bool)$ **where**
  $at_p\text{-}f'\ w = rep\ (at_p\text{-}f\ w)$

definition $at_p{}' = \{(y,\ at_p\text{-}f'\ y)|y.\ y \in W\}$
declare $at_p{}'$*-def* [*simp*]

lemma *range-*at_p*-*f':
  assumes $w \in W$
  shows $at_p\text{-}f'\ w \in PI'$
⟨*proof*⟩

lemma *rep-at*:
  assumes $\pi \in PI$
    and $(y,\ \pi) \in at_p$
  shows $(y,\ rep\ \pi) \in at_p{}'$
⟨*proof*⟩

interpretation $\mathfrak{B}'$: *adequate-place-framework* $\mathcal{C}\ PI'\ at_p{}'$
⟨*proof*⟩

lemma *singleton-model-for-normalized-reduced-literals*:
  $\exists\mathcal{M}.\ \forall lt \in \mathcal{C}.\ interp\ I_{sa}\ \mathcal{M}\ lt \wedge (\forall u \in U.\ hcard\ (\mathcal{M}\ u) \leq 1)$
⟨*proof*⟩

end

theorem *singleton-model-for-reduced-MLSS-clause*:
  assumes *norm-*$\mathcal{C}$: *normalized-MLSSmf-clause* $\mathcal{C}$
    and V: $V = vars_m\ \mathcal{C}$
    and $\mathcal{A}$*-model*: *normalized-MLSSmf-clause.is-model-for-reduced-dnf* $\mathcal{C}\ \mathcal{A}$
  shows $\exists\mathcal{M}.$ *normalized-MLSSmf-clause.is-model-for-reduced-dnf* $\mathcal{C}\ \mathcal{M} \wedge$
      $(\forall \alpha \in P^+\ V.\ hcard\ (\mathcal{M}\ v_\alpha) \leq 1)$
⟨*proof*⟩

end
theory *MLSSmf-to-MLSS-Completeness*
  imports *MLSSmf-Semantics MLSSmf-to-MLSS MLSSmf-HF-Extras*
    Proper-Venn-Regions Reduced-MLSS-Formula-Singleton-Model-Property
begin

locale *MLSSmf-to-MLSS-complete* =

normalized-MLSSmf-clause $\mathcal{C}$ **for** $\mathcal{C}$ :: ('v, 'f) *MLSSmf-clause* +
fixes $\mathcal{B}$ :: ('v, 'f) *Composite* ⇒ *hf*
assumes $\mathcal{B}$: *is-model-for-reduced-dnf* $\mathcal{B}$

fixes Λ :: *hf* ⇒ *'v set set*
assumes Λ-*subset-V*: $\Lambda\ x \subseteq P^+\ V$
and Λ-*preserves-zero*: Λ *0* = {}
and Λ-*inc*: $a \le b \Longrightarrow \Lambda\ a \subseteq \Lambda\ b$
and Λ-*add*: $\Lambda\ (a \sqcup b) = \Lambda\ a \cup \Lambda\ b$
and Λ-*mul*: $\Lambda\ (a \sqcap b) = \Lambda\ a \cap \Lambda\ b$
and Λ-*discr*: $l \subseteq P^+\ V \Longrightarrow$
$a = \bigsqcup HF\ ((\mathcal{B} \circ VennRegion)\ `\ l) \Longrightarrow a = \bigsqcup HF\ ((\mathcal{B} \circ VennRegion)\ `\ (\Lambda\ a))$
begin

fun $discretize_v$ :: (('v, 'f) *Composite* ⇒ *hf*) ⇒ ('v ⇒ *hf*) **where**
$discretize_v\ \mathcal{M} = \mathcal{M} \circ Solo$

fun $discretize_f$ :: (('v, 'f) *Composite* ⇒ *hf*) ⇒ ('f ⇒ *hf* ⇒ *hf*) **where**
$discretize_f\ \mathcal{M} = (\lambda f\ a.\ \mathcal{M}\ w_{f\Lambda\ a})$

interpretation *proper-Venn-regions V* $discretize_v\ \mathcal{B}$
⟨*proof*⟩

lemma *all-literal-sat*: $\forall\, lt \in set\ \mathcal{C}.\ I_l\ (discretize_v\ \mathcal{B})\ (discretize_f\ \mathcal{B})\ lt$
⟨*proof*⟩

lemma $\mathcal{C}$-*sat*: $I_{cl}\ (discretize_v\ \mathcal{B})\ (discretize_f\ \mathcal{B})\ \mathcal{C}$
⟨*proof*⟩

end

lemma (**in** *normalized-MLSSmf-clause*) *MLSSmf-to-MLSS-completeness*:
assumes *is-model-for-reduced-dnf M*
shows $\exists\, M_v\ M_f.\ I_{cl}\ M_v\ M_f\ \mathcal{C}$
⟨*proof*⟩

end
theory *MLSSmf-to-MLSS-Correctness*
imports *MLSSmf-to-MLSS-Soundness MLSSmf-to-MLSS-Completeness*
begin

fun *reduce* :: ('v, 'f) *MLSSmf-clause* ⇒ ('v, 'f) *Composite pset-fm set set* **where**
reduce $\mathcal{C}$ = *normalized-MLSSmf-clause.reduced-dnf* $\mathcal{C}$

fun *interp-DNF* :: (('v, 'f) *Composite* ⇒ *hf*) ⇒ ('v, 'f) *Composite pset-fm set set* ⇒ *bool* **where**
interp-DNF $\mathcal{M}$ *clauses* ⟷ ($\exists\, clause \in clauses.\ \forall\, lt \in clause.\ interp\ I_{sa}\ \mathcal{M}\ lt$)

corollary *MLSSmf-to-MLSS-correct*:
  assumes *norm-clause* $\mathcal{C}$
    shows $(\exists M_v\ M_f.\ I_{cl}\ M_v\ M_f\ \mathcal{C}) \longleftrightarrow (\exists \mathcal{M}.\ \mathit{interp\text{-}DNF}\ \mathcal{M}\ (\mathit{reduce}\ \mathcal{C}))$
⟨*proof*⟩

end

References

[1] Domenico Cantone, Jacob T. Schwartz, and Calogero G. Zarba. A decision procedure for a sublanguage of set theory involving monotone additive and multiplicative functions, ii. the multi-level case. *Le Matematiche; Vol 60, No 1 (2005); 133-162*, 60, 01 2006.

[2] Lukas Stevens. Mlss decision procedure. *Archive of Formal Proofs*, May 2023. ISSN 2150-914x. https://isa-afp.org/entries/MLSS_Decision_Proc.html, Formal proof development.