A Verified Reduction Algorithm from MLSSmf to MLSS

Yiran Duan, Lukas Stevens

February 6, 2026

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 a verified decision procedure for **MLSSmf**.

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

definition $size_m :: ('v, 'f)$ *MLSSmf-clause* $\Rightarrow$ *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*:
$card\ (restriction\text{-}on\text{-}InterOfVars\ vs) \leq 2 * length\ vs$
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-restriction-on-UnionOfVars*:
$card\ (restriction\text{-}on\text{-}UnionOfVars\ vs) \leq Suc\ (length\ vs)$
⟨*proof*⟩

theorem (**in** *normalized-MLSSmf-clause*) *size-introduce-v*:
$card\ introduce\text{-}v \leq (3 * card\ V + 2) * (2\ \widehat{}\ card\ V)$
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-restriction-on-UnionOfVennRegions*:
$card\ (restriction\text{-}on\text{-}UnionOfVennRegions\ \alpha s) \leq Suc\ (length\ \alpha s)$
⟨*proof*⟩

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

lemma (**in** *normalized-MLSSmf-clause*) *length-F-list*:
$length\ F\text{-}list = card\ F$
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-introduce-UnionOfVennRegions*:
$card\ introduce\text{-}UnionOfVennRegions \leq Suc\ (2\ \widehat{}\ card\ V) * 2\ \widehat{}\ 2\ \widehat{}\ card\ V$
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *length-choices-from-lists*:
$\forall\, choice \in set\ (choices\text{-}from\text{-}lists\ xss).\ length\ choice = length\ xss$
⟨*proof*⟩

lemma (**in** *normalized-MLSSmf-clause*) *size-introduce-w*:
$\forall\, clause \in introduce\text{-}w.\ card\ clause \leq 2\ \widehat{}\ (2 * 2\ \widehat{}\ card\ V) * card\ F$

⟨*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*$_m$ *C*
⟨*proof*⟩

theorem (**in** *normalized-MLSSmf-clause*) *size-reduced-dnf*:
  ∀ *clause* ∈ *reduced-dnf*. *card clause* ≤
    2 ^ (*2* ∗ *2* ^ (*3* ∗ *size*$_m$ *C*)) ∗ (*2* ∗ *size*$_m$ *C*) +
    (*3* ∗ (*3* ∗ *size*$_m$ *C*) + *2*) ∗ (*2* ^ (*3* ∗ *size*$_m$ *C*)) +
    Suc (*2* ^ (*3* ∗ *size*$_m$ *C*)) ∗ *2* ^ *2* ^ (*3* ∗ *size*$_m$ *C*) +
    2 ^ (*Suc* (*2* ∗ *2* ^ (*3* ∗ *size*$_m$ *C*))) ∗ *size*$_m$ *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 C **for** *C* :: ('*v*, '*f*) *MLSSmf-clause* +
    fixes *M*$_v$ :: '*v* ⇒ *hf*
      and *M*$_f$ :: '*f* ⇒ *hf* ⇒ *hf*
  assumes *model-for-C*: *I*$_{cl}$ *M*$_v$ *M*$_f$ *C*
begin

interpretation *proper-Venn-regions V M*$_v$
  ⟨*proof*⟩

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

termination
⟨*proof*⟩

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

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

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

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

lemma *soundness-introduce-UnionOfVennRegions*:
∀ *lt* ∈ *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 ∃ *lt* ∈ *set* (*restriction-on-FunOfUnionOfVennRegions l' m' f*). *interp* I_{sa} $\mathcal{M}$ *lt*
⟨*proof*⟩

lemma *soundness-introduce-w*:
∃ *clause* ∈ *introduce-w*. ∀ *lt* ∈ *clause*. *interp* I_{sa} $\mathcal{M}$ *lt*
⟨*proof*⟩

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

lemma *soundness-reduce-cl*:
∀ *fml* ∈ *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 $at_p'\text{-}def$ [*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* $\Rightarrow$ *hf*
assumes $\mathcal{B}$: *is-model-for-reduced-dnf* $\mathcal{B}$

fixes Λ :: *hf* $\Rightarrow$ $'v$ *set set*
assumes Λ-*subset-V*: $\Lambda\ x \subseteq P^+\ V$
and Λ-*preserves-zero*: $\Lambda\ 0 = \{\}$
and Λ-*inc*: $a \leq b \Longrightarrow \Lambda\ a \subseteq \Lambda\ b$
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 $discretize_v$ :: $(('v, 'f)\ Composite \Rightarrow hf) \Rightarrow ('v \Rightarrow hf)$ **where**
$discretize_v\ \mathcal{M} = \mathcal{M} \circ Solo$

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

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

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