MathLang

Fairouz Kamareddine
Heriot-Watt University, Edinburgh

January 2011

Common Mathematical Language of mathematicians: CML

+ CML is *expressive*: it has linguistic categories like *proofs* and *theorems*.

+ CML has been refined by intensive use and is rooted in *long traditions*.

+ CML is *approved* by most mathematicians as a communication medium.

+ CML *accommodates many branches* of mathematics, and is adaptable to new ones.

– Since CML is based on natural language, it is *informal* and *ambiguous*.

– CML is *incomplete:* Much is left implicit, appealing to the reader's intuition.

– CML is *poorly organised:* In a CML text, many structural aspects are omitted.

– CML is *automation-unfriendly:* A CML text is a plain text and cannot be easily automated.

A CML-text

From chapter 1, § 2 of E. Landau's *Foundations of Analysis* (Landau 1930, 1951).

Theorem 6. [Commutative Law of Addition]

$$x + y = y + x.$$

Proof Fix y, and let $\mathfrak{M}$ be the set of all x for which the assertion holds.
I) We have

$$y + 1 = y',$$

and furthermore, by the construction in the proof of Theorem 4,

$$1 + y = y',$$

so that

$$1 + y = y + 1$$

and 1 belongs to $\mathfrak{M}$.
II) If x belongs to $\mathfrak{M}$, then

$$x + y = y + x,$$

Therefore

$$(x + y)' = (y + x)' = y + x'.$$

By the construction in the proof of Theorem 4, we have

$$x' + y = (x + y)',$$

hence

$$x' + y = y + x',$$

so that x' belongs to $\mathfrak{M}$. The assertion therefore holds for all x. □

The problem with formal logic

- No logical language is an alternative to CML
  - A logical language does not have *mathematico-linguistic* categories, is *not universal* to all mathematicians, and is *not a good communication medium*.
  - Logical languages make fixed choices (*first versus higher order, predicative versus impredicative, constructive versus classical, types or sets*, etc.). But different parts of mathematics need different choices and there is no universal agreement as to which is the best formalism.
  - A logician reformulates in logic their *formalization* of a mathematical-text as a formal, complete text which is structured considerably *unlike* the original, and is of little use to the *ordinary* mathematician.
  - Mathematicians do not want to use formal logic and have *for centuries* done mathematics without it.

- *So, mathematicians kept to CML.*

- We would like to find an alternative to CML which avoids some of the features of the logical languages which made them unattractive to mathematicians.

What are the options for computerization?

Computers can handle mathematical text at various levels:

- Images of pages may be stored. While useful, this is not a good representation of *language* or *knowledge*.
- Typesetting systems like LaTeX, TeXmacs, can be used.
- Document representations like OpenMath, OMDoc, MathML, can be used.
- Formal logics used by theorem provers (Coq, Isabelle, HOL, Mizar, Isar, etc.) can be used.

We are gradually developing a system named Mathlang which we hope will eventually allow building a bridge between the latter 3 levels.

This talk aims at discussing the motivations rather than the details.

The issues with typesetting systems

+ A system like LaTeX, TeXmacs, provides good defaults for visual appearance, while allowing fine control when needed.

+ LaTeX and TeXmacs support commonly needed document structures, while allowing custom structures to be created.

– Unless the mathematician is amazingly disciplined, the *logical structure of symbolic formulas is not represented* at all.

– The *logical structure of mathematics as embedded in natural language text is not represented.* Automated discovery of the semantics of natural language text is still too primitive and requires human oversight.

LaTeX example

draft documents	✓
public documents	✓
computations and proofs	✗

```
\begin{theorem}[Commutative Law of Addition]\label{theorem:6}

  $$x+y=y+x.$$

\end {theorem}

\begin{proof}

  Fix $y$, and $\mathfrak{M}$ be the set of all $x$ for which
  the assertion holds.

  \begin{enumerate}

  \item We have $$y+1=y',$$
    and furthermore, by the construction in
    the proof of Theorem~\ref{theorem:4}, $$1+y=y',$$

    so that $$1+y=y+1$$
    and $1$ belongs to $\mathfrak{M}$.
```

```
\item If $x$ belongs to $\mathfrak{M}$, then $$x+y=y+x,$$

  Therefore
  $$(x+y)'=(y+x)'=y+x'.$$

  By the construction in the proof of
  Theorem~\ref{theorem:4}, we have $$x'+y=(x+y)',$$

  hence
  $$x'+y=y+x',$$
  so that $x'$ belongs to $\mathfrak{M}$.

\end{enumerate}

The assertion therefore holds for all $x$.

\end{proof}
```

Full formalization difficulties: choices

A CML-text is structured differently from a fully formalized text proving the same facts. *Making the latter involves extensive knowledge and many choices:*

- The choice of the *underlying logical system.*
- The choice of *how concepts are implemented* (equational reasoning, equivalences and classes, partial functions, induction, etc.).
- The choice of the *formal foundation*: a type theory (dependent?), a set theory (ZF? FM?), a category theory? etc.
- The choice of the *proof checker*: Automath, Isabelle, Coq, PVS, Mizar, HOL, ...

An issue is that one must in general commit to one set of choices.

Full formalization difficulties: informality

Any informal reasoning in a CML-text will cause various problems when fully formalizing it:

- A single (big) step may need to expand into a (series of) syntactic proof expressions. *Very long expressions can replace a clear* CML*-text.*

- The entire CML-text may need *reformulation* in a fully *complete* syntactic formalism where every detail is spelled out. New details may need to be woven throughout the entire text. The text may need to be *turned inside out.*

- Reasoning may be obscured by *proof tactics*, whose meaning is often *ad hoc* and implementation-dependent.

Regardless, ordinary mathematicians do not find the new text useful.

Coq example

	draft documents	✗
	public documents	✗
	computations and proofs	✓

From Module `Arith.Plus` of Coq standard library (`http://coq.inria.fr/`).

```
Lemma plus_sym: (n,m:nat)(n+m)=(m+n).

Proof.

Intros n m ; Elim n ; Simpl_rew ; Auto with arith.

Intros y H ; Elim (plus_n_-Sm m y) ; Simpl_rew ; Auto with arith.

Qed.
```

Mathlang's Goal: Open borders between mathematics, logic and computation

- Ordinary mathematicians *avoid* formal mathematical logic.
- Ordinary mathematicians *avoid* proof checking (via a computer).
- Ordinary mathematicians *may use* a computer for computation: there are over 1 million people who use Mathematica (including linguists, engineers, etc.).
- Mathematicians may also use other computer forms like Maple, LaTeX, etc.
- But we are not interested in only *libraries* or *computation* or *text editing*.
- We want *freedeom of movement* between mathematics, logic and computation.
- At every stage, we must have *the choice* of the level of formalilty and the depth of computation.

Aim for Mathlang? (Kamareddine and Wells 2001, 2002)

Can we formalise a mathematical text, avoiding as much as possible the ambiguities of natural language, while still guaranteeing the following four goals?

1. The formalised text looks very much like the original mathematical text (and hence the content of the original mathematical text is respected).

2. The formalised text can be fully manipulated and searched in ways that respect its mathematical structure and meaning.

3. Steps can be made to do computation (via computer algebra systems) and proof checking (via proof checkers) on the formalised text.

4. This formalisation of text is not much harder for the ordinary mathematician than LaTeX. *Full formalization down to a foundation of mathematics is not required*, although allowing and supporting this is one goal.

(No theorem prover's language satisfies these goals.)

	draft documents	✓
Mathlang	**public documents**	✓
	computations and proofs	✓

- A Mathlang text captures the grammatical and reasoning aspects of mathematical structure for further computer manipulation.

- A *weak type system* checks Mathlang documents at a grammatical level.

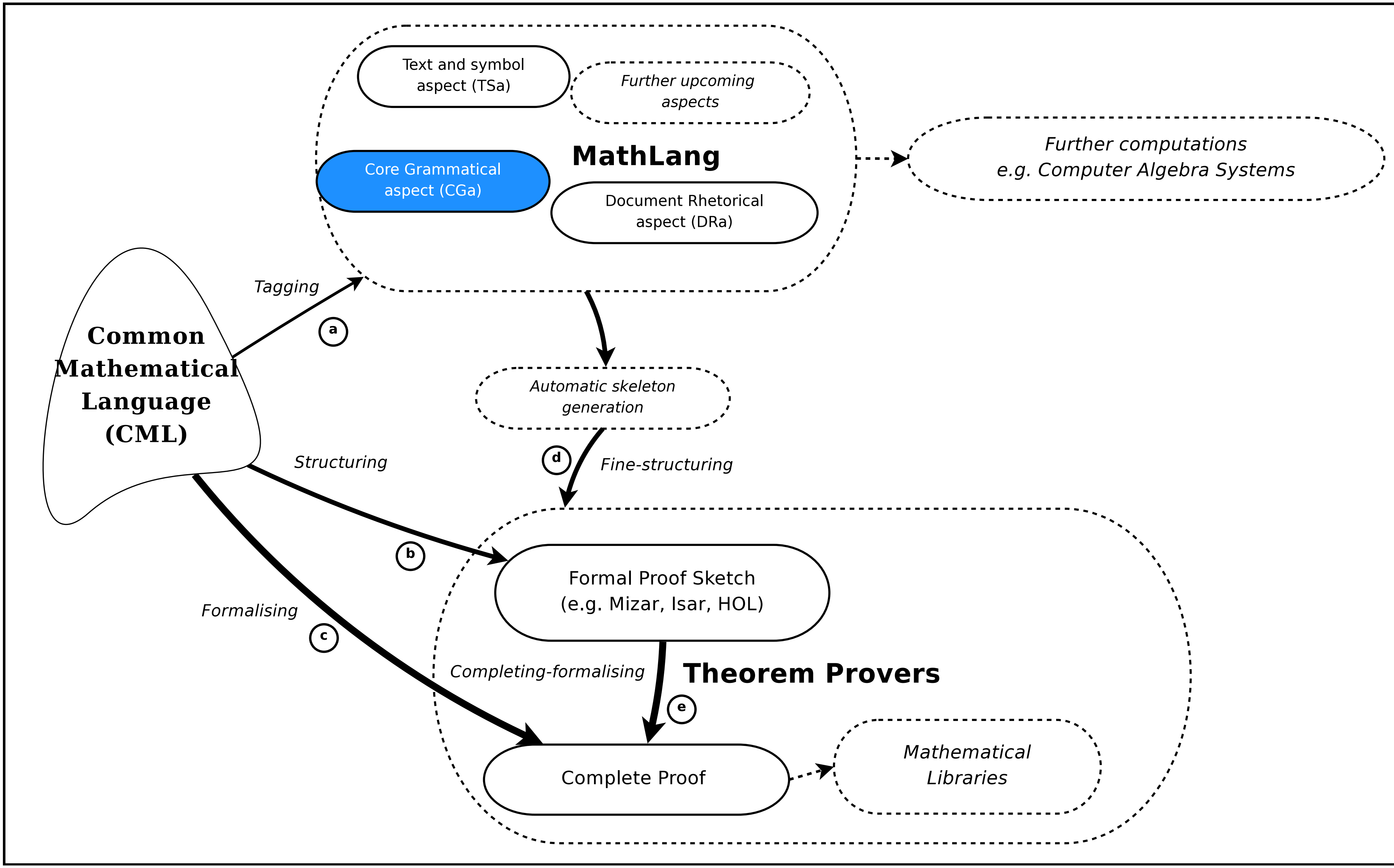
- A Mathlang text remains *close* to its CML original, allowing confidence that the CML has been captured correctly.

- We have been developing ways to weave natural language text into Mathlang.

- Mathlang aims to eventually support *all encoding uses*.

- The CML view of a Mathlang text should match the mathematician's intentions.

- The formal structure should be suitable for various automated uses.


Text and symbol aspect (TSa)
Further upcoming aspects
Core Grammatical aspect (CGa)
MathLang
Document Rhetorical aspect (DRa)
Further computations e.g. Computer Algebra Systems
Tagging
a
Common Mathematical Language (CML)
Automatic skeleton generation
d
Fine-structuring
Structuring
b
Formal Proof Sketch (e.g. Mizar, Isar, HOL)
Formalising
c
Completing-formalising
Theorem Provers
e
Complete Proof
Mathematical Libraries

What is CGa? (Maarek's PhD thesis)

- CGa is a formal language derived from MV (N.G. de Bruijn 1987) and WTT (Kamareddine and Nederpelt 2004) which aims at expliciting the grammatical role played by the elements of a CML text.

- The structures and common concepts used in CML are captured by CGa with a finite set of grammatical/linguistic/syntactic categories: *Term* "$\sqrt{2}$", *set* "$\mathbb{Q}$", *noun* "number", *adjective* "even", *statement* "$a = b$", *declaration* "Let a be a number", *definition* "An even number is..", *step* "a is odd, hence $a \neq 0$", *context* "Assume a is even".
  term set noun adjective statement declaration definition step context .

- Generally, each syntactic category has a corresponding *weak type*.

- CGa's type system derives typing judgments to check whether the reasoning parts of a document are coherently built.

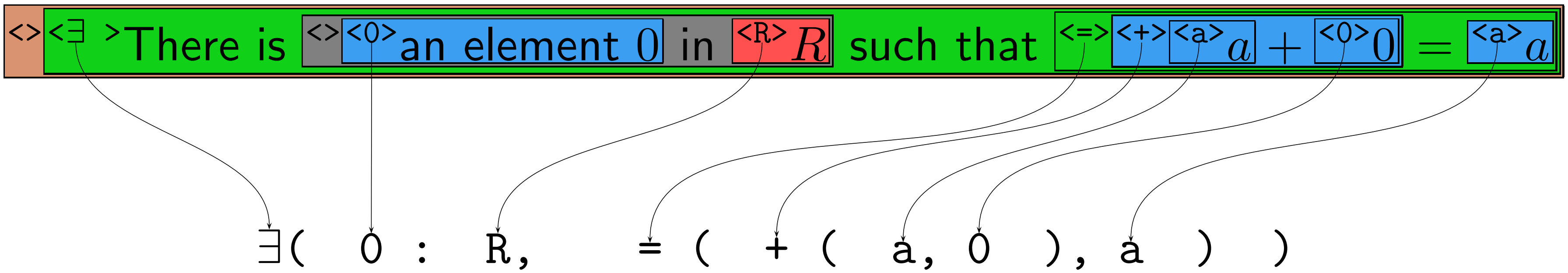


Figure 1: Example of CGa encoding of CML text

Weak Type Theory

In Weak Type Theory (or $\mathrm{W}\textsc{tt}$) we have the following linguistic categories:

- On the *atomic* level: *variables*, *constants* and *binders*,
- On the *phrase* level: *terms* $\mathcal{T}$, *sets* $\mathbb{S}$, *nouns* $\mathcal{N}$ and *adjectives* $\mathcal{A}$,
- On the *sentence* level: *statements* P and *definitions* $\mathcal{D}$,
- On the *discourse* level: *contexts* $\mathbb{I}\Gamma$, *lines* $\mathbf{l}$ and *books* $\mathbf{B}$.

Categories of syntax of WTT

Other category	abstract syntax	symbol
expressions	$\mathcal{E} = T \vert \mathbb{S} \vert \mathcal{N} \vert P$	E
parameters	$\mathcal{P} = T \vert \mathbb{S} \vert P$ (note: $\overrightarrow{\mathcal{P}}$ is a list of $\mathcal{P}$s)	P
typings *declarations*	$\mathbf{T} = \mathbb{S} : \text{SET} \vert \mathcal{S} : \text{STAT} \vert T : \mathbb{S} \vert T : \mathcal{N} \vert T : \mathcal{A}$ $\mathcal{Z} = \mathtt{V}^S : \text{SET} \vert \mathtt{V}^P : \text{STAT} \vert \mathtt{V}^T : \mathbb{S} \vert \mathtt{V}^T : \mathcal{N}$	T Z

level	category	abstract syntax	symbol
atomic	*variables*	$\mathtt{V} = \mathtt{V}^T \vert \mathtt{V}^S \vert \mathtt{V}^P$	x
	constants	$\mathtt{C} = \mathtt{C}^T \vert \mathtt{C}^S \vert \mathtt{C}^N \vert \mathtt{C}^A \vert \mathtt{C}^P$	c
	binders	$\mathtt{B} = \mathtt{B}^T \vert \mathtt{B}^S \vert \mathtt{B}^N \vert \mathtt{B}^A \vert \mathtt{B}^P$	b
phrase	*terms*	$\mathcal{T} = \mathtt{C}^T(\overrightarrow{\mathcal{P}}) \vert \mathtt{B}^T_{\mathcal{Z}}(\mathcal{E}) \vert \mathtt{V}^T$	t
	sets	$\mathbb{S} = \mathtt{C}^S(\overrightarrow{\mathcal{P}}) \vert \mathtt{B}^S_{\mathcal{Z}}(\mathcal{E}) \vert \mathtt{V}^S$	s
	nouns	$\mathcal{N} = \mathtt{C}^N(\overrightarrow{\mathcal{P}}) \vert \mathtt{B}^N_{\mathcal{Z}}(\mathcal{E}) \vert \mathcal{A}\mathcal{N}$	n
	adjectives	$\mathcal{A} = \mathtt{C}^A(\overrightarrow{\mathcal{P}}) \vert \mathtt{B}^A_{\mathcal{Z}}(\mathcal{E})$	a
sentence	*statements*	$P = \mathtt{C}^P(\overrightarrow{\mathcal{P}}) \vert \mathtt{B}^P_{\mathcal{Z}}(\mathcal{E}) \vert \mathtt{V}^P$	S
	definitions	$\mathcal{D} = \mathcal{D}^\varphi \vert \mathcal{D}^P$	D
		$\mathcal{D}^\varphi = \mathtt{C}^T(\overrightarrow{V}) := \mathcal{T} \vert \mathtt{C}^S(\overrightarrow{V}) := \mathbb{S} \vert$	
		$\mathtt{C}^N(\overrightarrow{V}) := \mathcal{N} \vert \mathtt{C}^A(\overrightarrow{V}) := \mathcal{A}$	
		$\mathcal{D}^P = \mathtt{C}^P(\overrightarrow{V}) := P$	
discourse	*contexts*	$\Gamma = \emptyset \mid \Gamma, \mathcal{Z} \mid \Gamma, P$	Γ
	lines	$\mathbf{l} = \Gamma \rhd P \mid \Gamma \rhd \mathcal{D}$	l
	books	$\mathbf{B} = \emptyset \mid \mathbf{B} \circ \mathbf{l}$	B

Derivation rules

(1) B is a weakly well-typed book: $\vdash\ B :: \texttt{book}$.

(2) Γ is a weakly well-typed context relative to book B: $B\ \vdash\ \Gamma :: \texttt{cont}$.

(3) t is a weakly well-typed term, etc., relative to book B and context Γ:

$$\begin{array}{lll} B;\Gamma\ \vdash\ t :: T, & B;\Gamma\ \vdash\ s :: S, & B;\Gamma\ \vdash\ n :: N, \\ B;\Gamma\ \vdash\ a :: A, & B;\Gamma\ \vdash\ p :: P, & B;\Gamma\ \vdash\ d :: D \end{array}$$

$OK(B;\Gamma)$. stands for: $\vdash\ B :: \texttt{book}$, *and* $B\ \vdash\ \Gamma :: \texttt{cont}$

Examples of derivation rules

- $\mathtt{dvar}(\emptyset) = \emptyset \qquad \mathtt{dvar}(\Gamma', x : W) = \mathtt{dvar}(\Gamma'), x \qquad \mathtt{dvar}(\Gamma', P) = \mathtt{dvar}(\Gamma')$

$$\frac{OK(B;\Gamma), \quad x \in \mathtt{V}^{\mathtt{T/S/P}}, \quad x \in \mathtt{dvar}(\Gamma)}{B;\Gamma \ \vdash \ x :: T/S/P} \quad (var)$$

$$\frac{B;\Gamma \quad \vdash \quad n :: N, \quad B;\Gamma \quad \vdash \quad a :: A}{B;\Gamma \quad \vdash \quad an :: N} \quad (adj{-}noun)$$

$$\frac{}{\vdash \quad \emptyset :: \mathtt{book}} \quad (emp{-}book)$$

$$\frac{B;\Gamma \quad \vdash \quad p :: P}{\vdash \quad B \ \circ \ \Gamma \triangleright p :: \mathtt{book}} \qquad \frac{B;\Gamma \quad \vdash \quad d :: D}{\vdash \quad B \ \circ \ \Gamma \triangleright d :: \mathtt{book}} \quad (book{-}ext)$$

Properties of WTT

- *Every variable is declared* If $B;\Gamma \vdash \Phi :: \mathbf{W}$ then $FV(\Phi) \subseteq \mathtt{dvar}(\Gamma)$.
- *Correct subcontexts* If $B \vdash \Gamma :: \mathtt{cont}$ and $\Gamma' \subseteq \Gamma$ then $B \vdash \Gamma' :: \mathtt{cont}$.
- *Correct subbooks* If $\vdash B :: \mathtt{book}$ and $B' \subseteq B$ then $\vdash B' :: \mathtt{book}$.
- *Free constants are either declared in book or in contexts* If $B;\Gamma \vdash \Phi :: \mathbf{W}$, then $FC(\Phi) \subseteq \mathtt{prefcons}(B) \cup \mathtt{defcons}(B)$.
- *Types are unique* If $B;\Gamma \vdash A :: \mathbf{W_1}$ and $B;\Gamma \vdash A :: \mathbf{W_2}$, then $\mathbf{W_1} \equiv \mathbf{W_2}$.
- *Weak type checking is decidable* there is a decision procedure for the question $B;\Gamma \vdash \Phi :: \mathbf{W}\ ?$.
- *Weak typability is computable* there is a procedure deciding whether an answer exists for $B;\Gamma \vdash \Phi :: ?$ and if so, delivering the answer.

Definition unfolding

- Let $\vdash B$ :: $\texttt{book}$ and $\Gamma \rhd c(x_1, \ldots, x_n) := \Phi$ a line in B.

- We write $B \vdash c(P_1, \ldots, P_n) \xrightarrow{\delta} \Phi[x_i := P_i]$.

- *Church-Rosser* If $B \vdash \Phi \overset{\delta}{\twoheadrightarrow} \Phi_1$ and $B \vdash \Phi \overset{\delta}{\twoheadrightarrow} \Phi_2$ then there exists Φ_3 such that $B \vdash \Phi_1 \overset{\delta}{\twoheadrightarrow} \Phi_3$ andf $B \vdash \Phi_2 \overset{\delta}{\twoheadrightarrow} \Phi_3$.

- *Strong Normalisation* Let $\vdash B$ :: $\texttt{book}$. For all subformulas Ψ occurring in B, relation $\xrightarrow{\delta}$ is strongly normalizing (i.e., definition unfolding inside a well-typed book is a well-founded procedure).

CGa Weak Type Checking

Let $\mathfrak{M}$ be a set,

y and x are natural numbers,

if x belongs to $\mathfrak{M}$

then $x + y = y + x$

CGa Weak Type checking detects grammatical errors

Let $\mathfrak{M}$ be a set ,

y and x are natural numbers ,

if x belongs to $\mathfrak{M}$

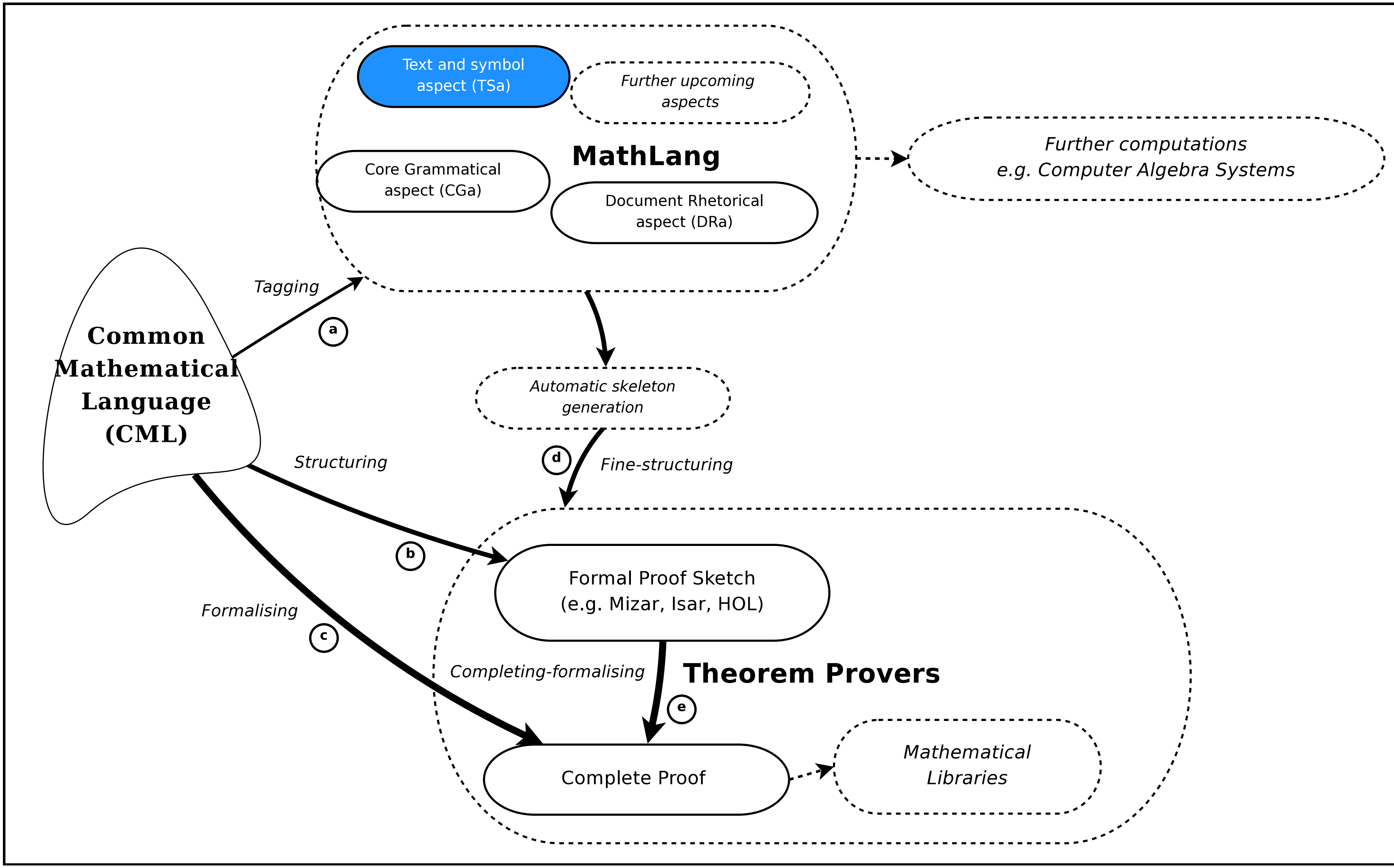
then $x + y$ $\Leftarrow$ **error**

How complete is the CGa?

- CGa is quite advanced but remains under development according to new translations of mathematical texts. Are the current CGa categories sufficient?

- The metatheory of WTT has been established in (Kamareddine and Nederepelt 2004). That of CGa remains to be established. However, since CGa is quite similar to WTT, its metatheory might be similar to that of WTT.

- The type checker for CGa works well and gives some useful error messages. Error messages should be improved.


Text and symbol aspect (TSa)
Further upcoming aspects
Core Grammatical aspect (CGa)
MathLang
Document Rhetorical aspect (DRa)
Further computations e.g. Computer Algebra Systems
Common Mathematical Language (CML)
Tagging
a
Automatic skeleton generation
d
Fine-structuring
Structuring
b
Formal Proof Sketch (e.g. Mizar, Isar, HOL)
Formalising
c
Completing-formalising
Theorem Provers
e
Complete Proof
Mathematical Libraries

What is TSa? Lamar's PhD thesis

- TSa builds the bridge between a CML text and its grammatical interpretation and adjoins to each CGa expression a string of words and/or symbols which aims to act as its CML representation.
- TSa plays the role of a user interface
- TSa can flexibly represent natural language mathematics.
- The author wraps the natural language text with boxes representing the grammatical categories (as we saw before).
- The author can also give interpretations to the parts of the text.

Interpretations

There is ${}^{\mathtt{0}}$an element 0 in ${}^{\mathtt{R}}R$ such that ${}^{\mathtt{eq}}\,{}^{\mathtt{plus}}\,{}^{\mathtt{a}}a + {}^{\mathtt{0}}0 = {}^{\mathtt{a}}a$.

```
{  0  :  R;     eq (  plus (  a, 0  ), a  );  };
```

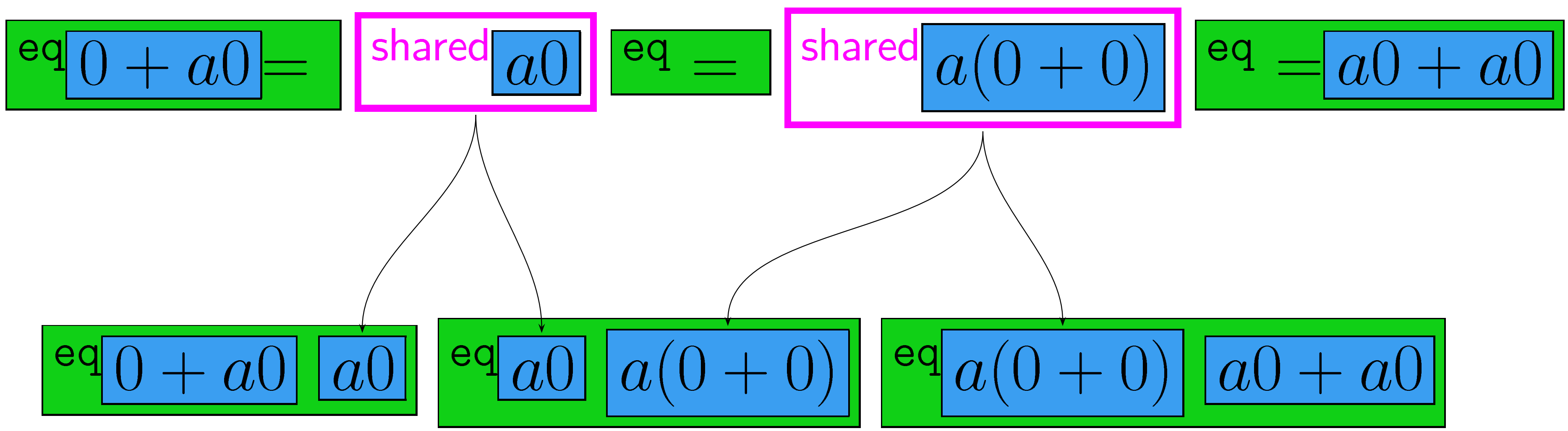
There is ${}^{\mathtt{0}}$an element 0 in ${}^{\mathtt{R}}R$ such that ${}^{\mathtt{eq}}\,{}^{\mathtt{plus}}\,{}^{\mathtt{a}}a + {}^{\mathtt{0}}0 = {}^{\mathtt{a}}a$.

There is ${}^{\mathtt{0}}$an element 0 in ${}^{\mathtt{R}}R$ such that ${}^{\mathtt{eq}}\,{}^{\mathtt{plus}}\,{}^{\mathtt{a}}a + {}^{\mathtt{0}}0$ equals ${}^{\mathtt{a}}a$.

${}^{\mathtt{0}}0 \in {}^{\mathtt{R}}R$, ${}^{\mathtt{eq}}\,{}^{\mathtt{plus}}\,{}^{\mathtt{a}}a + {}^{\mathtt{0}}0 = {}^{\mathtt{a}}a$.

Rewrite rules enable natural language representation

Take the example $0 + a0 = a0 = a(0 + 0) = a0 + a0$



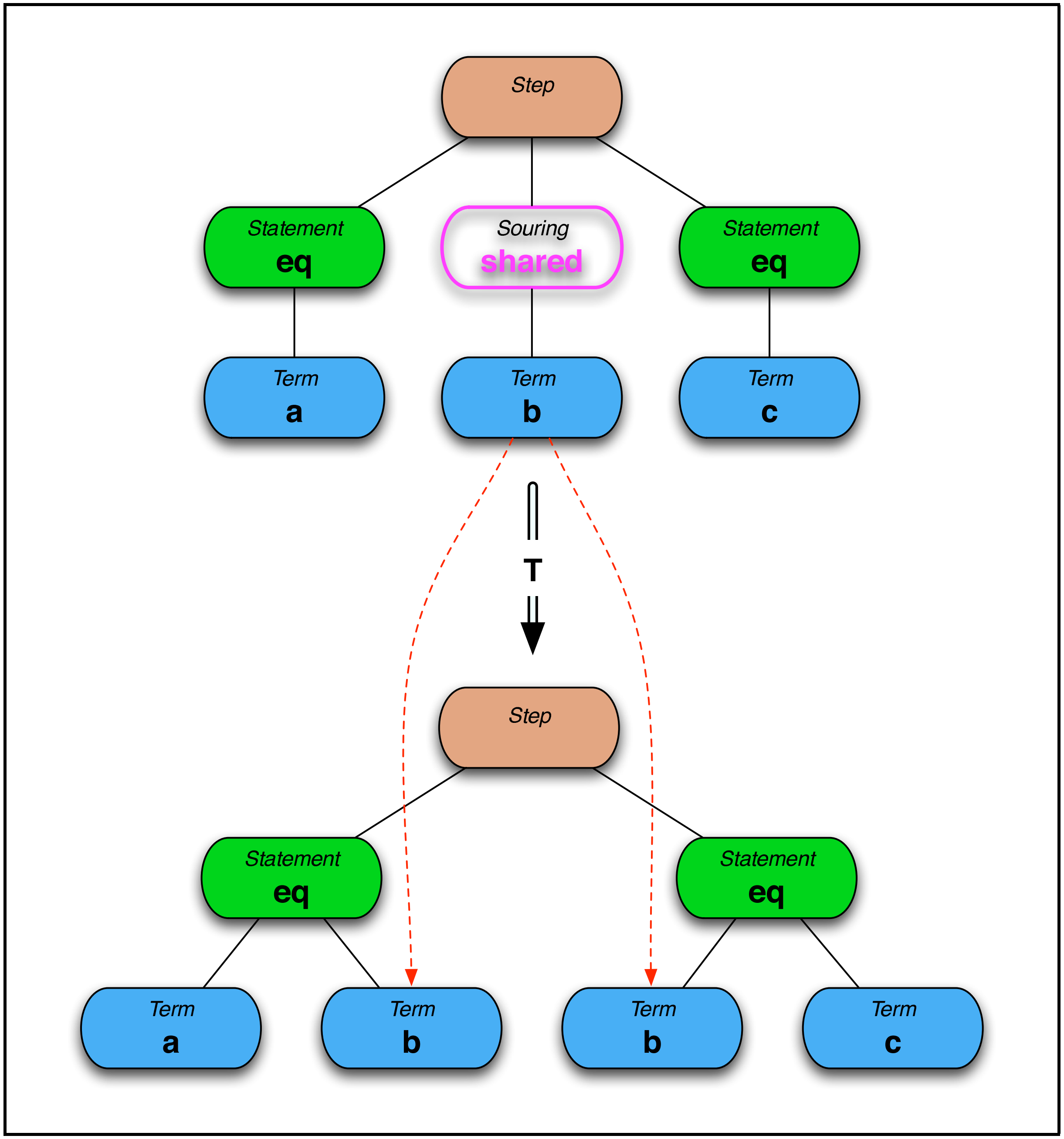
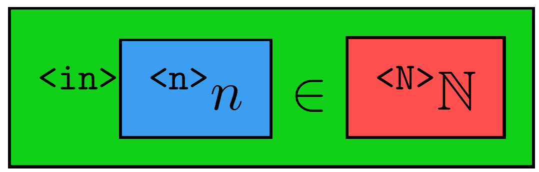
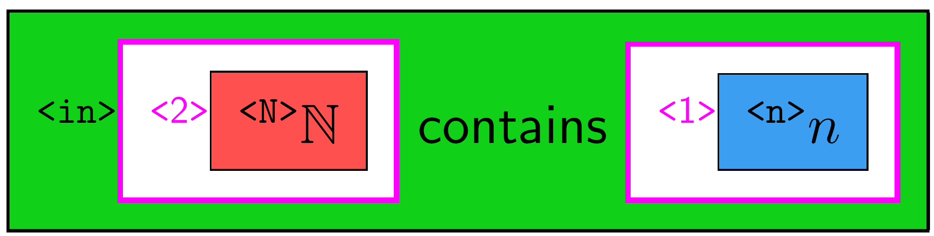


Figure 2: Example for a simple shared souring

reordering/position Souring



$ann =$



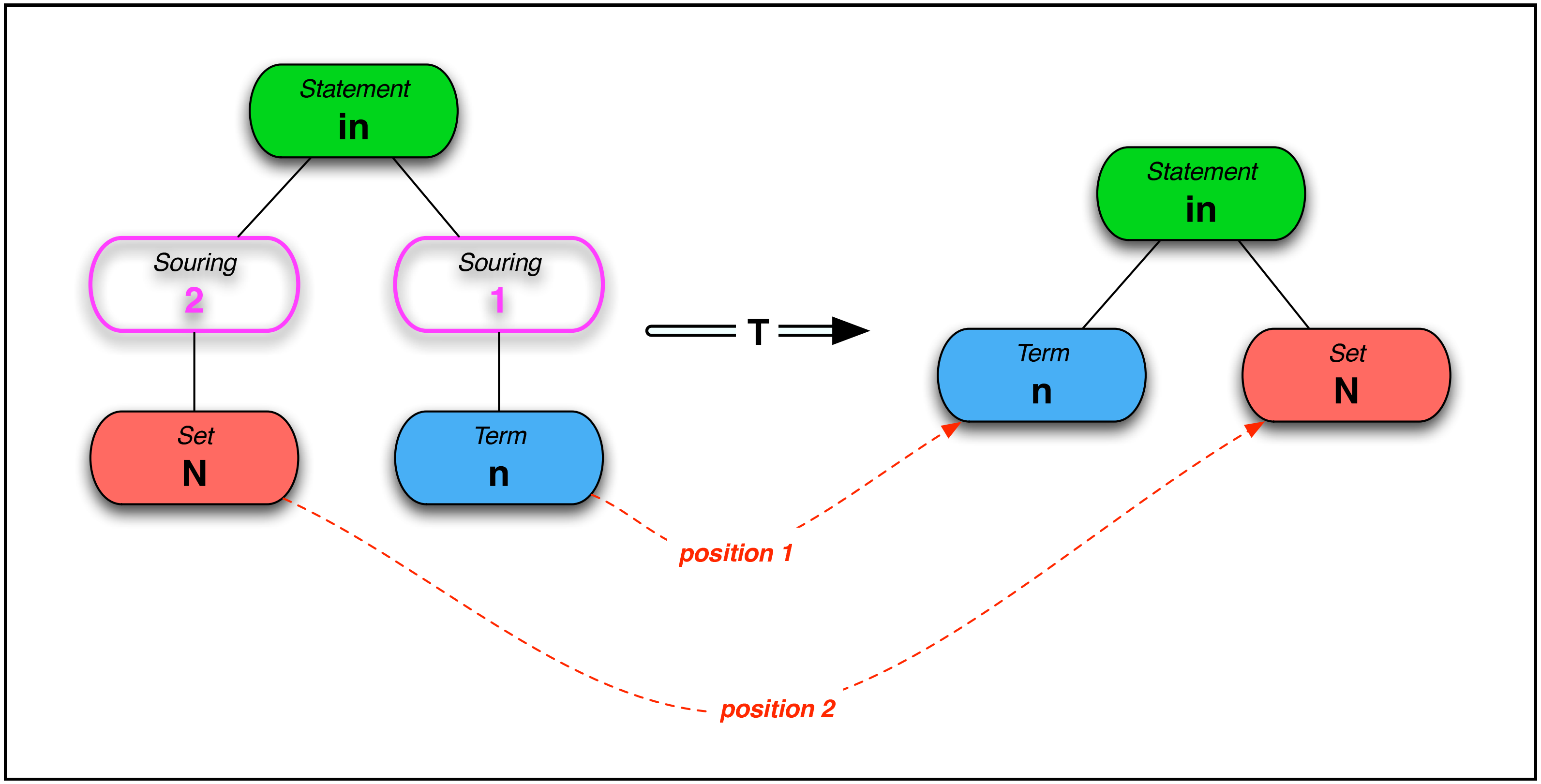
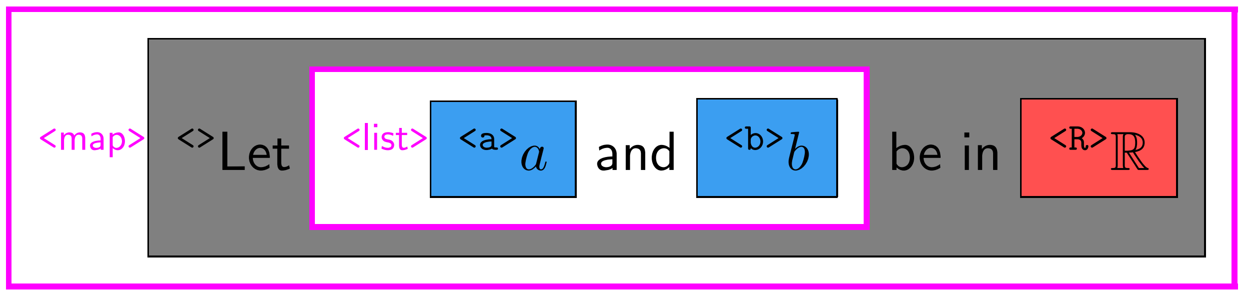


Figure 3: Example for a position souring

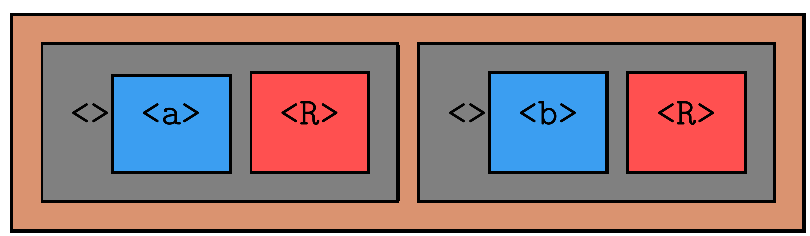
map souring

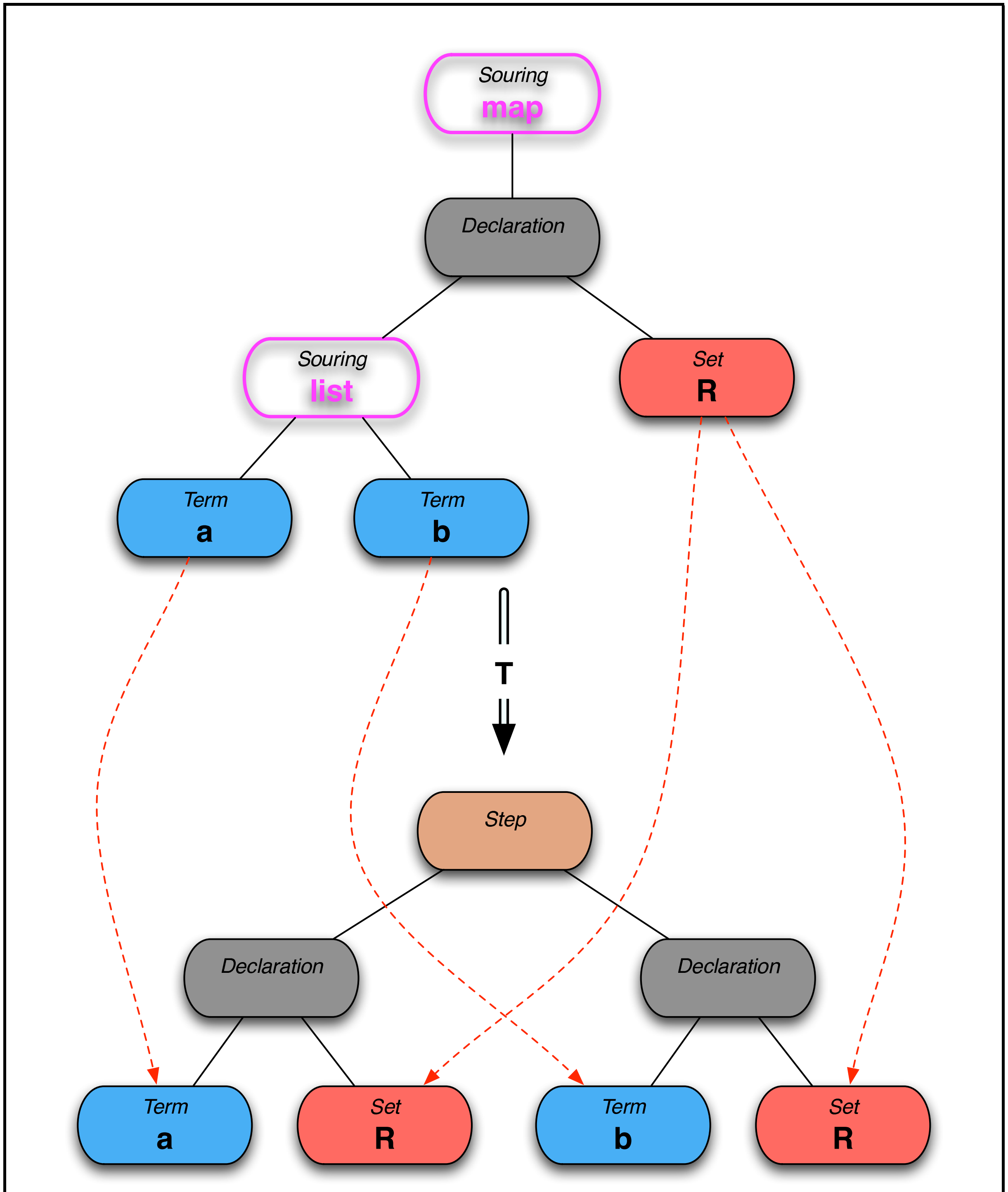
$$ann =$$



This is expanded to

$$\mathbf{T}\,(ann) =$$



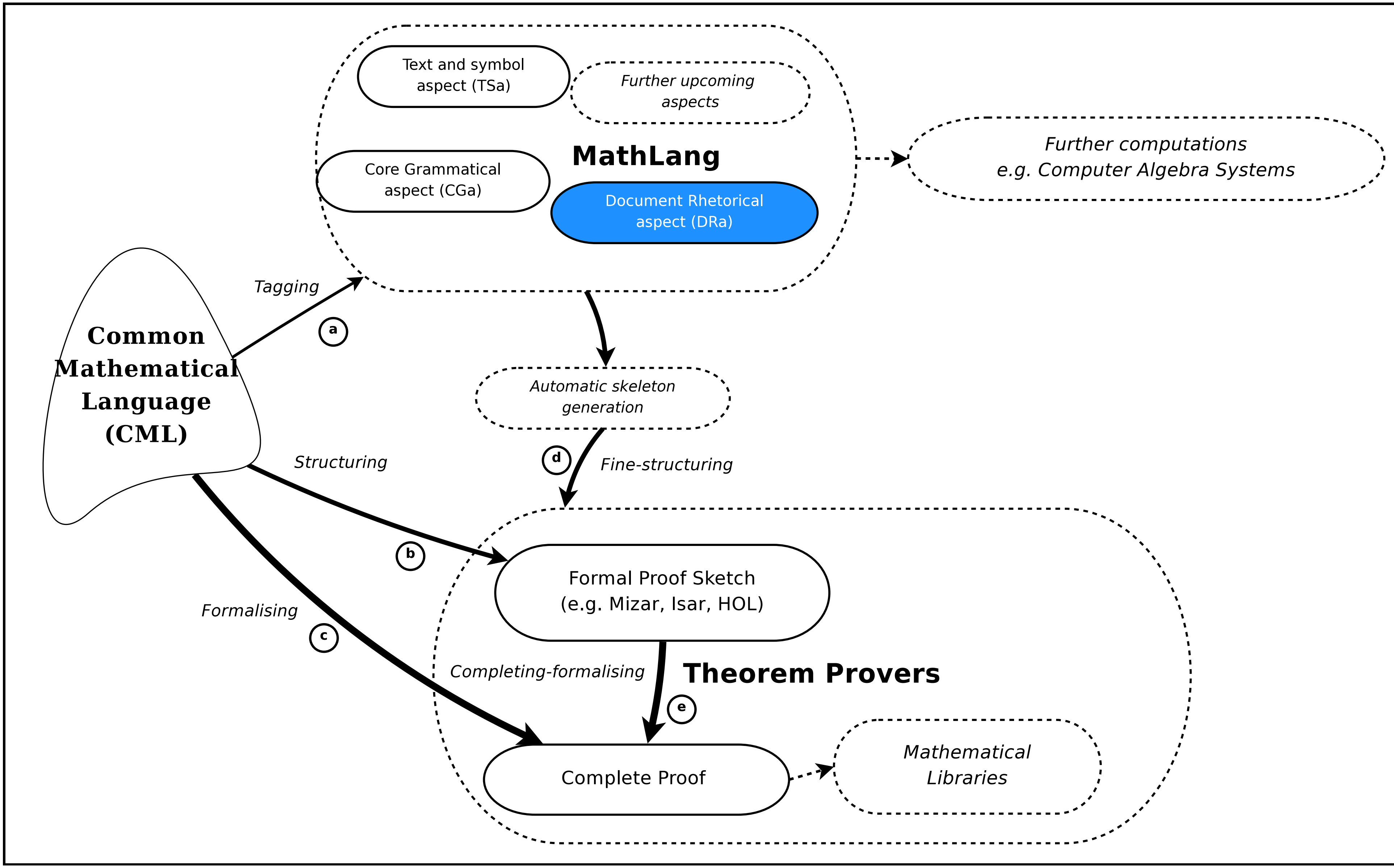

Souring
map
Declaration
Souring
list
Set
R
Term
a
Term
b
T
Step
Declaration
Declaration
Term
a
Set
R
Term
b
Set
R

How complete is TSa?

- TSa provides useful interface facilities but it is still under development.

- So far, only simple rewrite (souring) rules are used and they are not comprehensive. E.g., unable to cope with things like $\overbrace{x = \ldots = x}^{n \text{ times}}$.

- The TSa theory and metatheory need development.


Text and symbol aspect (TSa)
Further upcoming aspects
Core Grammatical aspect (CGa)
MathLang
Document Rhetorical aspect (DRa)
Further computations e.g. Computer Algebra Systems
Common Mathematical Language (CML)
Tagging
a
Automatic skeleton generation
d
Fine-structuring
Structuring
b
Formal Proof Sketch (e.g. Mizar, Isar, HOL)
Formalising
c
Completing-formalising
Theorem Provers
e
Complete Proof
Mathematical Libraries

What is DRa? Retel's PhD thesis

- DRa Document Rhetorical structure aspect.
- **Structural components of a document** like *chapter, section, subsection, etc.*
- **Mathematical components of a document** like *theorem, corollary, definition, proof, etc.*
- **Relations** between above components.
- These enhance readability, and ease the navigation of a document.
- Also, these help to go into more formal versions of the document.

Relations

Description
Instances of the StructuralRhetoricalRole *class:* preamble, part, chapter, section, paragraph, *etc.*
Instances of the MathematicalRhetoricalRole *class:* lemma, corollary, theorem, conjecture, definition, axiom, claim, proposition, assertion, proof, exercise, example, problem, solution, *etc.*

Relation
Types of relations: relatesTo, uses, justifies, subpartOf, inconsistentWith, exemplifies

What does the mathematician do?

- The mathematician wraps into boxes and uniquely names chunks of text

- The mathematician assigns to each box the structural and/or mathematical rhetorical roles

- The mathematician indicates the relations between wrapped chunks of texts

Lemma 1. For $m, n \in \mathbb{N}$ one has: $m^2 = 2n^2 \implies m = n = 0$.
Define on $\mathbb{N}$ the predicate:

$$P(m) \iff \exists n.m^2 = 2n^2 \ \& \ m > 0.$$

Claim. $P(m) \implies \exists m' < m.P(m')$. Indeed suppose $m^2 = 2n^2$ and $m > 0$. It follows that m^2 is even, but then m must be even, as odds square to odds. So $m = 2k$ and we have

$$2n^2 = m^2 = 4k^2 \implies n^2 = 2k^2$$

Since $m > 0$, if follows that $m^2 > 0, n^2 > 0$ and $n > 0$. Therefore $P(n)$. Moreover, $m^2 = n^2 + n^2 > n^2$, so $m^2 > n^2$ and hence $m > n$. So we can take $m' = n$.

By the claim $\forall m \in \mathbb{N}.\neg P(m)$, since there are no infinite descending sequences of natural numbers.

Now suppose $m^2 = 2n^2$ with $m \neq 0$. Then $m > 0$ and hence $P(m)$. Contradiction. Therefore $m = 0$. But then also $n = 0$.
Corollary 1. $\sqrt{2} \notin \mathbb{Q}$.
Suppose $\sqrt{2} \in \mathbb{Q}$, i.e. $\sqrt{2} = p/q$ with $p \in \mathbb{Z}, q \in \mathbb{Z} - \{0\}$. Then $\sqrt{2} = m/n$ with $m = |p|, n = |q| \neq 0$. It follows that $m^2 = 2n^2$. But then $n = 0$ by the lemma. Contradiction shows that $\sqrt{2} \notin \mathbb{Q}$.

Lemma 1.

For $m, n \in \mathbb{N}$ one has: $m^2 = 2n^2 \implies m = n = 0$

Proof.

Define on $\mathbb{N}$ the predicate:

$$P(m) \iff \exists n. m^2 = 2n^2 \ \& \ m > 0.$$

Claim. $P(m) \implies \exists m' < m. P(m')$.

Indeed suppose $m^2 = 2n^2$ and $m > 0$. It follows that m^2 is even, but then m must be even, as odds square to odds. So $m = 2k$ and we have $2n^2 = m^2 = 4k^2 \implies n^2 = 2k^2$ Since $m > 0$, if follows that $m^2 > 0$, $n^2 > 0$ and $n > 0$. Therefore $P(n)$. Moreover, $m^2 = n^2 + n^2 > n^2$, so $m^2 > n^2$ and hence $m > n$. So we can take $m' = n$.

By the claim $\forall m \in \mathbb{N}. \neg P(m)$, since there are no infinite descending sequences of natural numbers.

Now suppose $m^2 = 2n^2$

with $m \neq 0$. Then $m > 0$ and hence $P(m)$. Contradiction.

Therefore $m = 0$. But then also $n = 0$. □

Corollary 1. $\sqrt{2} \notin \mathbb{Q}$

Proof. Suppose $\sqrt{2} \in \mathbb{Q}$, i.e. $\sqrt{2} = p/q$ with $p \in \mathbb{Z}, q \in \mathbb{Z} - \{0\}$. Then $\sqrt{2} = m/n$ with $m = |p|, n = |q| \neq 0$. It follows that $m^2 = 2n^2$. But then $n = 0$ by the lemma. Contradiction shows that $\sqrt{2} \notin \mathbb{Q}$. □

(A, hasMathematicalRhetoricalRole, *lemma*)
(E, hasMathematicalRhetoricalRole, *definition*)
(F, hasMathematicalRhetoricalRole, *claim*)
(G, hasMathematicalRhetoricalRole, *proof*)
(B, hasMathematicalRhetoricalRole, *proof*)
(H, hasOtherMathematicalRhetoricalRole, *case*)
(I, hasOtherMathematicalRhetoricalRole, *case*)
(C, hasMathematicalRhetoricalRole, *corollary*)
(D, hasMathematicalRhetoricalRole, *proof*)

(B, justifies, A)
(D, justifies, C)
(D, uses, A)
(G, uses, E)
(F, uses, E)
(H, uses, E)
(H, subpartOf, B)
(H, subpartOf, I)

Lemma 1.

For $m, n \in \mathbb{N}$ one has: $m^2 = 2n^2 \Longrightarrow m = n = 0$ $\boxed{\mathbf{A}}$

Proof.

Define on $\mathbb{N}$ the predicate: $\boxed{\mathbf{E}}$

$$P(m) \iff \exists n. m^2 = 2n^2 \; \& \; m > 0.$$

uses justifies

Claim. $P(m) \Longrightarrow \exists m' < m. P(m').$ $\boxed{\mathbf{F}}$

uses

Indeed suppose $m^2 = 2n^2$ and $m > 0$. It follows that m^2 is even, but then m must be even, as odds square to odds. So $m = 2k$ and we have $2n^2 = m^2 = 4k^2 \Longrightarrow n^2 = 2k^2$ Since $m > 0$, it follows that $m^2 > 0$, $n^2 > 0$ and $n > 0$. Therefore $P(n)$. Moreover, $m^2 = n^2 + n^2 > n^2$, so $m^2 > n^2$ and hence $m > n$. So we can take $m' = n$. $\boxed{\mathbf{G}}$ $\boxed{\mathbf{B}}$

justifies uses uses

By the claim $\forall m \in \mathbb{N}. \neg P(m)$, since there are no infinite descending sequences of natural numbers.

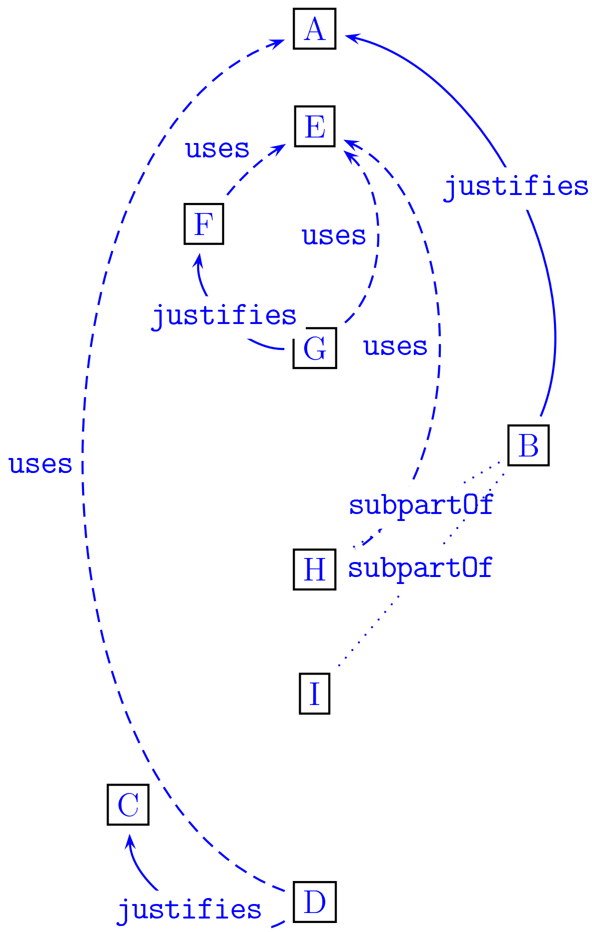
subpartOf subpartOf

Now suppose $m^2 = 2n^2$

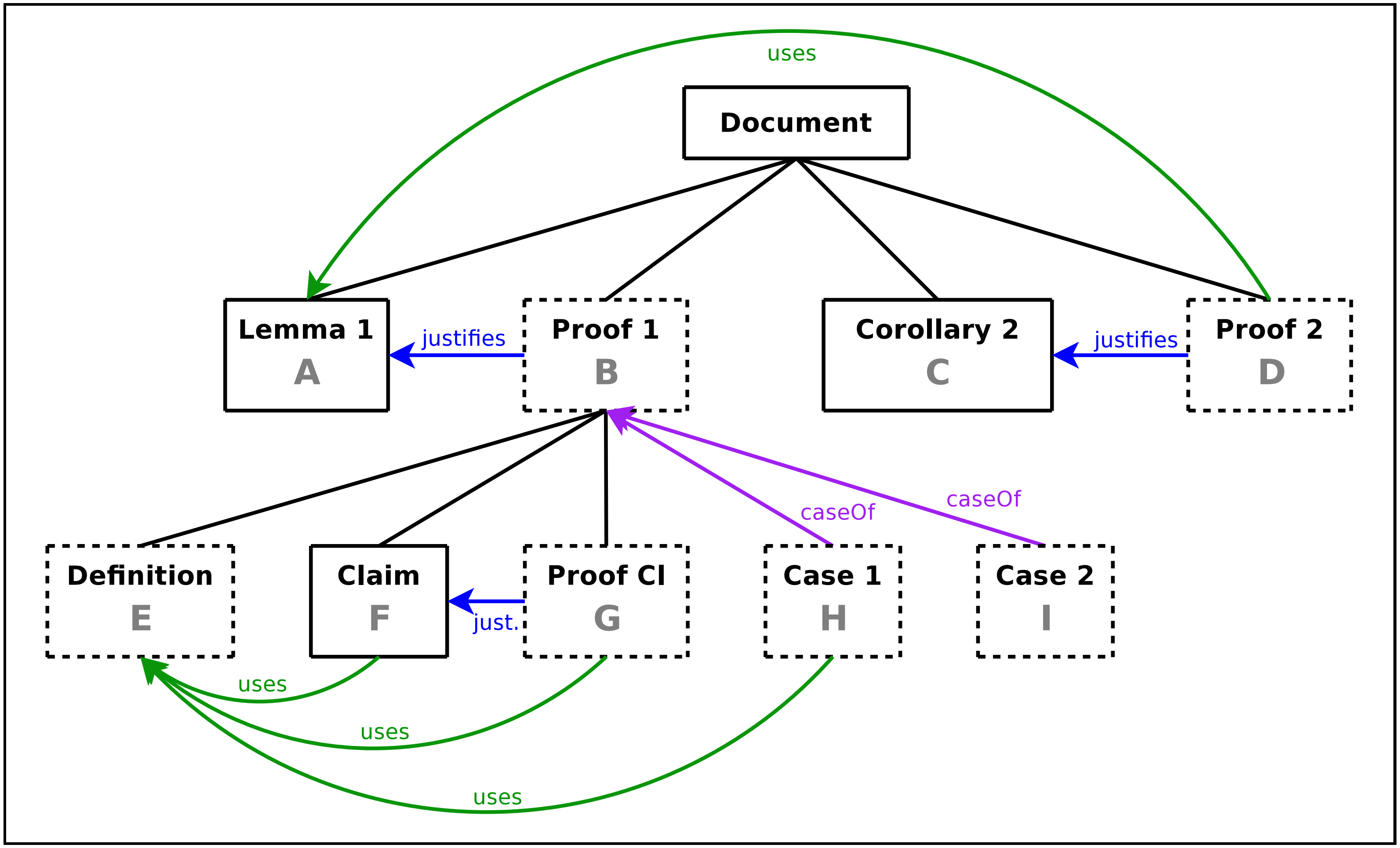
with $m \neq 0$. Then $m > 0$ and hence $P(m)$. Contradiction. $\boxed{\mathbf{H}}$

The automatically generated dependency Graph

Dependency Graph (DG)



An alternative view of the DRa (Zengler's thesis)



The Graph of Textual Order: GoTO
Zengler's thesis

- To be able to examine the proper structure of a DRa tree we introduce the concept of textual order between two nodes in the tree.
- Using textual orders, we can transform the dependency graph into a GoTO by transforming each edge of the DG.
- So far there are two reasons why the GoTO is produced:
  1. Automatic Checking of the GoTO can reveal errors in the document (e.g. loops in the structure of the document).
  2. The GoTO is used to automatically produce a proof skeleton for a prover (we use a variety: Isabelle, Mizar, Coq).
- We automatically transform a DG into GoTO and automatically check the GoTO for errors in the document:

1. Loops in the GoTO (error)
2. Proof of an unproved node (error)
3. More than one proof for a proved node (warning)
4. Missing proof for a proved node (warning)

- To achieve this we define for each vertex v of the tree:
  - $\mathcal{ENV}v$ is the environment of all mathematical statements that occur before the statements of v (from the root vertex).
  - Introduced symbols':
    $\mathcal{IN}v := \mathcal{DF}v \cup \mathcal{DC}v \cup \{s|s \in \mathcal{ST}v \wedge s \notin \mathcal{ENV}v\} \cup \bigcup_{c\ childOf\ v} \mathcal{IN}c$
  - Used symbol: $\mathcal{USE}v := \mathcal{T}v \cup \mathcal{S}v \cup \mathcal{N}v \cup \mathcal{A}v \cup \mathcal{ST}v \cup \bigcup_{c\ childOf\ v} \mathcal{USE}c$

- Strong textual order $\prec$: $B \prec A := \exists x(x \in \mathcal{IN}B \wedge x \in \mathcal{USE}A)$

- Weak textual order $\preceq$: $A \preceq B := \mathcal{IN}A \subseteq \mathcal{IN}B \wedge \mathcal{USE}A \subseteq \mathcal{USE}B$

- Common textual order $\leftrightarrow$: $A \leftrightarrow B := \exists x(x \in \mathcal{USE}A \wedge x \in \mathcal{USE}B)$

Graph of Textual Order

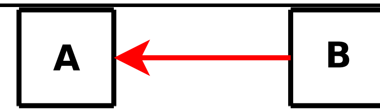
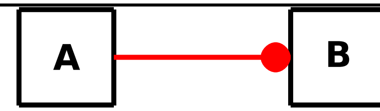
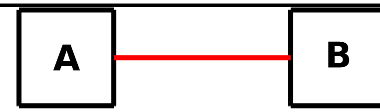
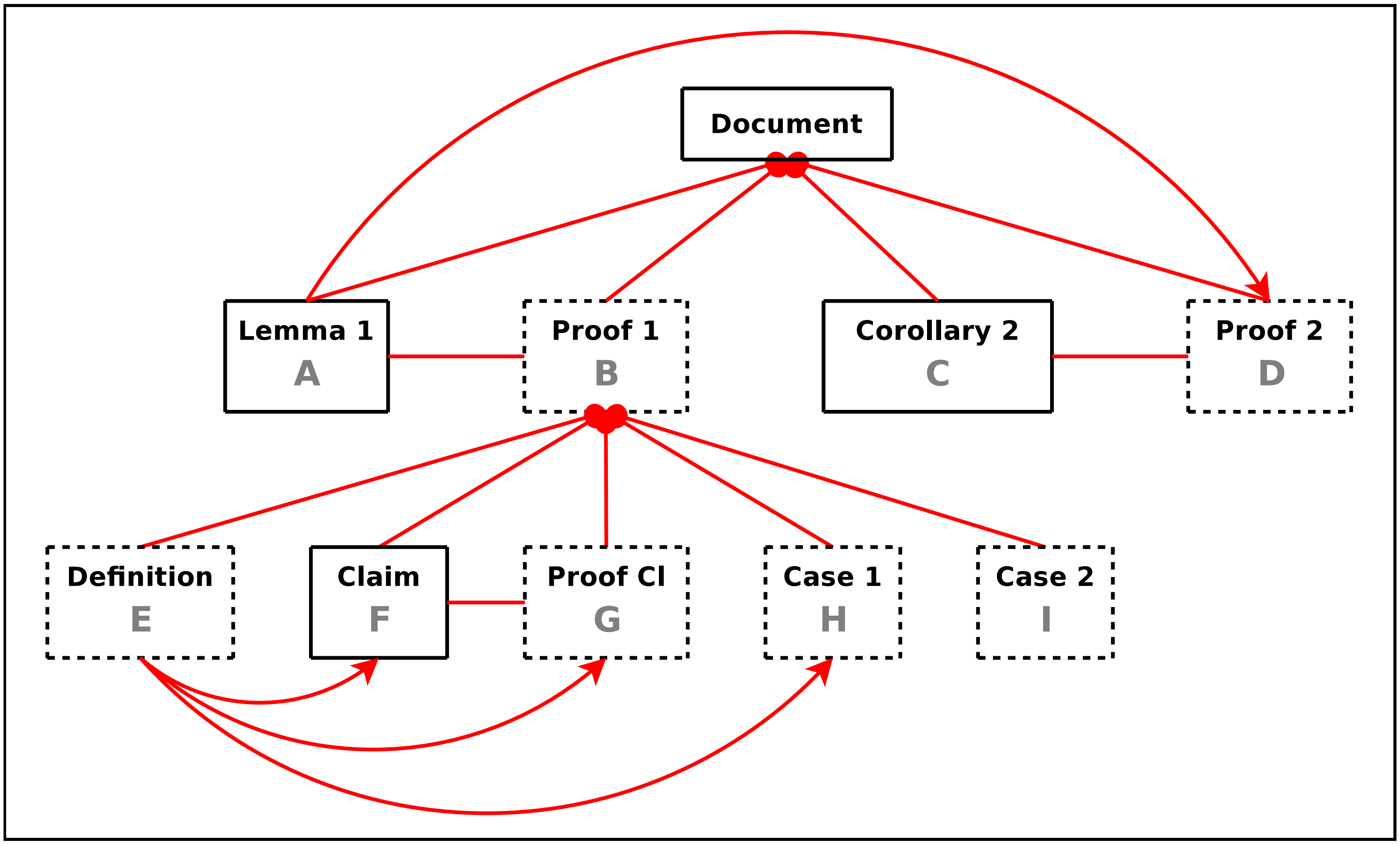
(A, uses, B)	$A \succ B$	
(A, caseOf, B)	$A \preceq B$	
(A, justifies, B)	$A \leftrightarrow B$	

Table 1: Graphical representation of edges in the GoTO

The GoTO can be generated automatically from the DG and therefore (since the DG can be produced automatically from an annotated document) automatically from an annotated document.

Graph of Textual Order for the DRa tree example



How complete is DRa?

- The dependency graph can be used to check whether the logical reasoning of the text is coherent and consistent (e.g., no loops in the reasoning).

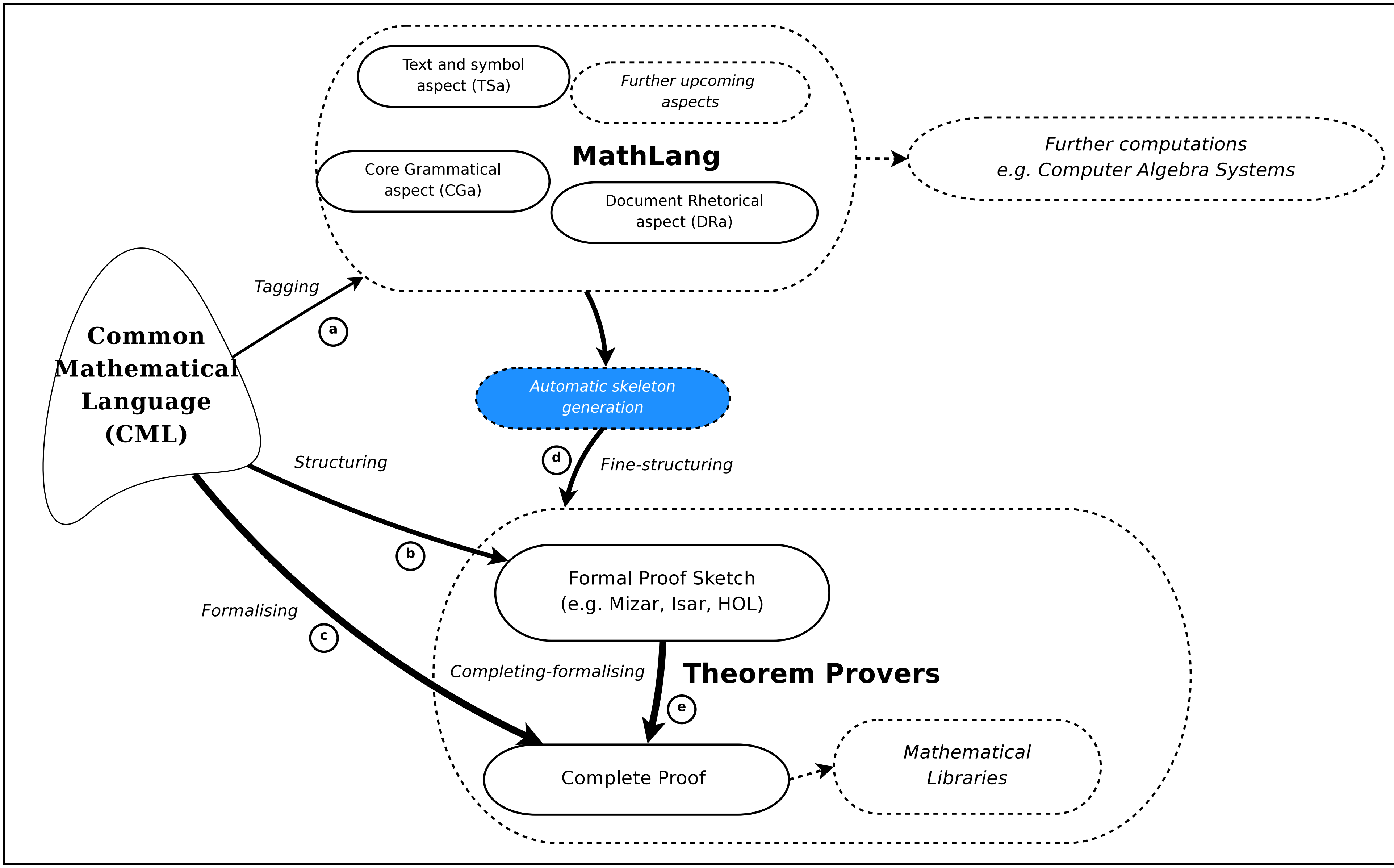
- However, both the DRa language and its implementation need more experience driven tests on natural language texts.

- Also, the DRa aspect still needs a number of implementation improvements (the automation of the analysis of the text based on its DRa features).

- Extend TSa to also cover DRa (in addition to CGa).

- Extend DRa depending on further experience driven translations.

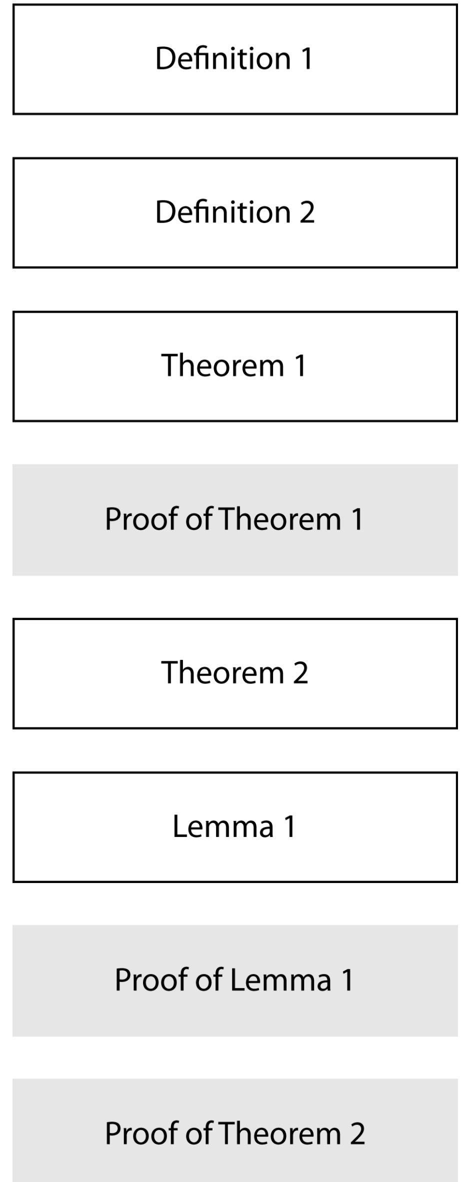
- Establish the soundness and completeness of DRa for mathematical texts.


Text and symbol aspect (TSa)
Further upcoming aspects
MathLang
Core Grammatical aspect (CGa)
Document Rhetorical aspect (DRa)
Further computations e.g. Computer Algebra Systems
Common Mathematical Language (CML)
Tagging
a
Automatic skeleton generation
d
Fine-structuring
Structuring
b
Formal Proof Sketch (e.g. Mizar, Isar, HOL)
Formalising
c
Completing-formalising
Theorem Provers
e
Complete Proof
Mathematical Libraries

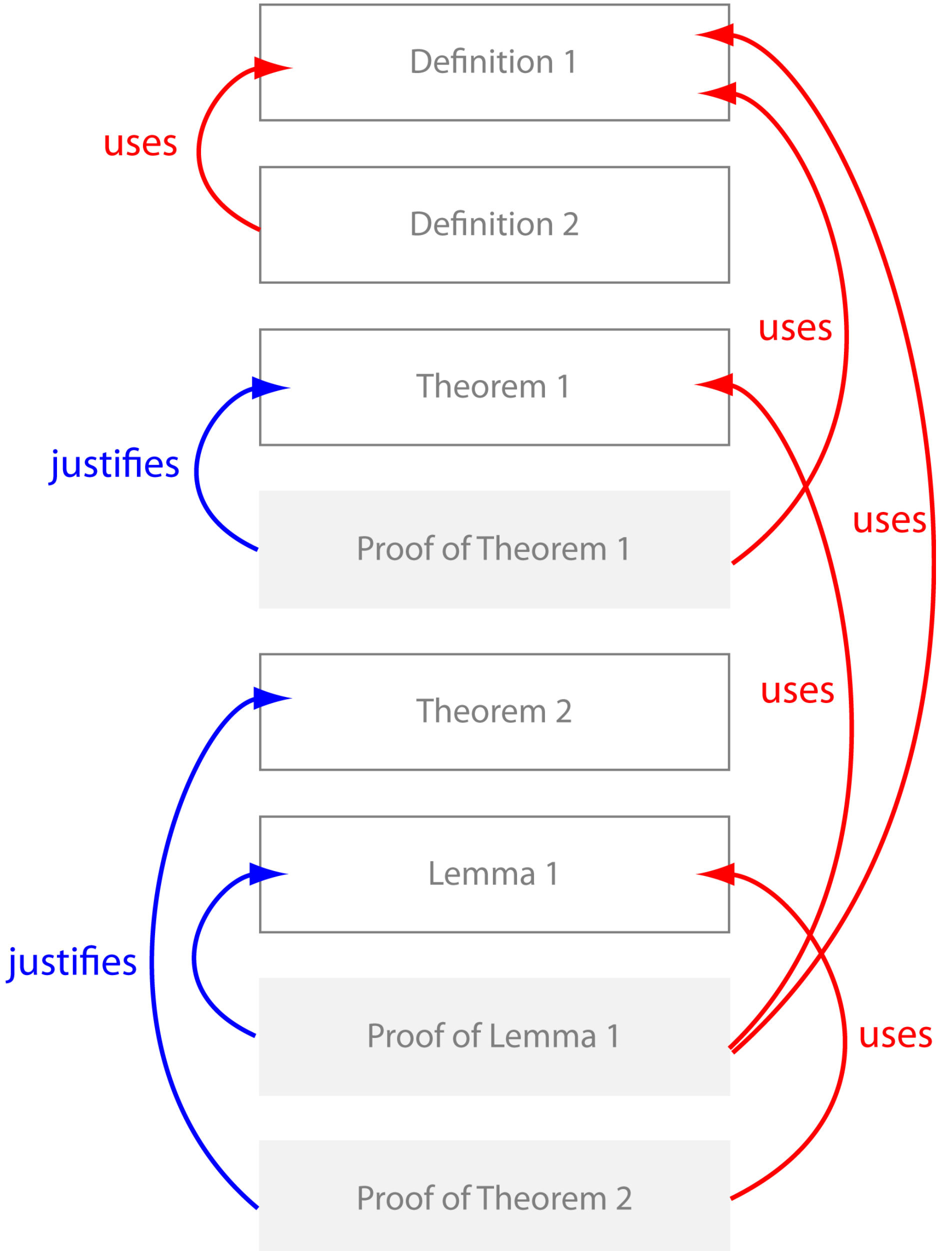
Different provers have

- different syntax
- different requirements to the structure of the text
  e.g.
  - no nested theorems/lemmas
  - only backward references
  - ...
- Aim: Skeleton should be as close as possible to the mathematician's text but with re-arrangements when necessary

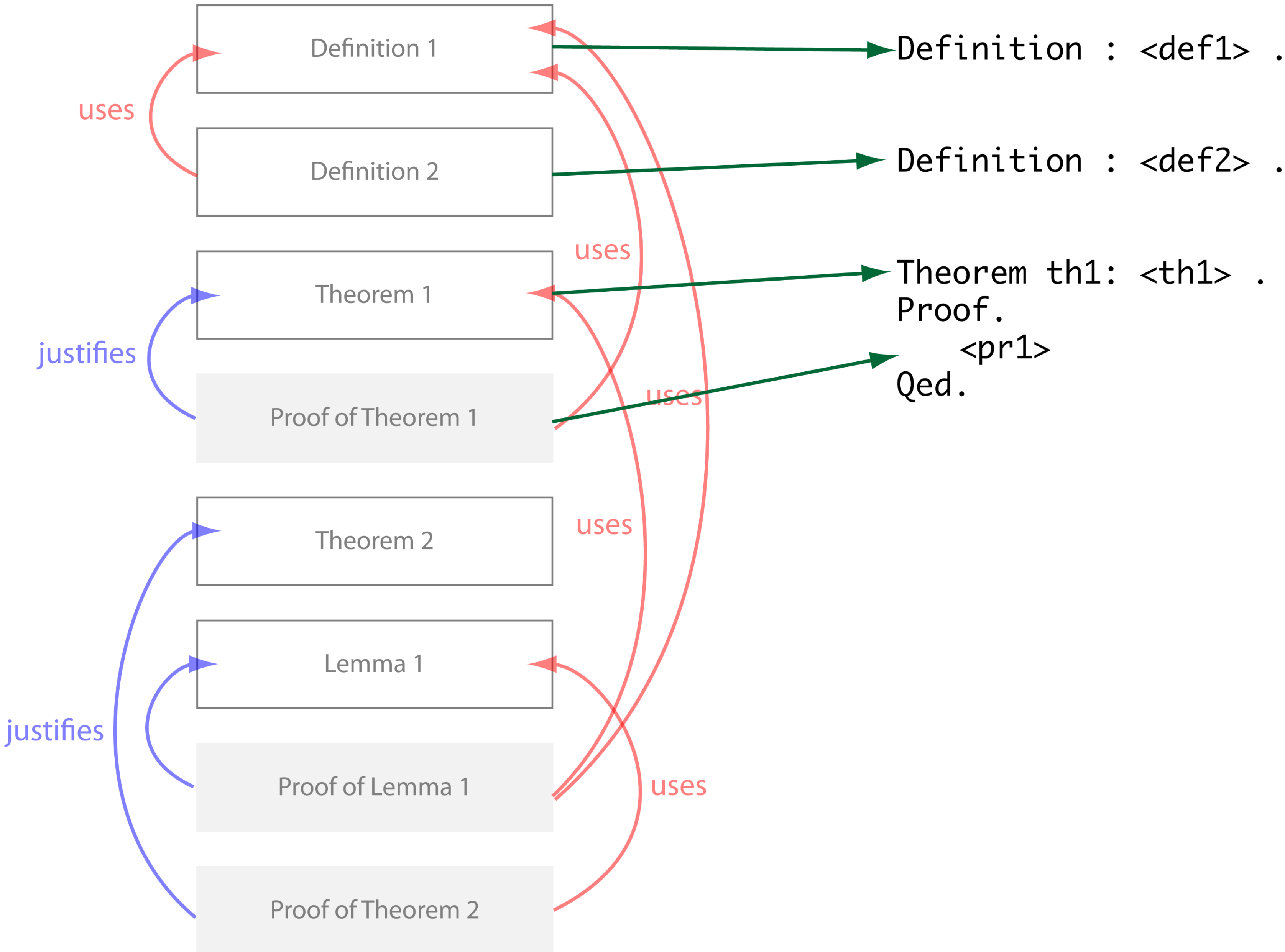
Example of nested theorems/lemmas (Moller, 03, Chapter III,2)



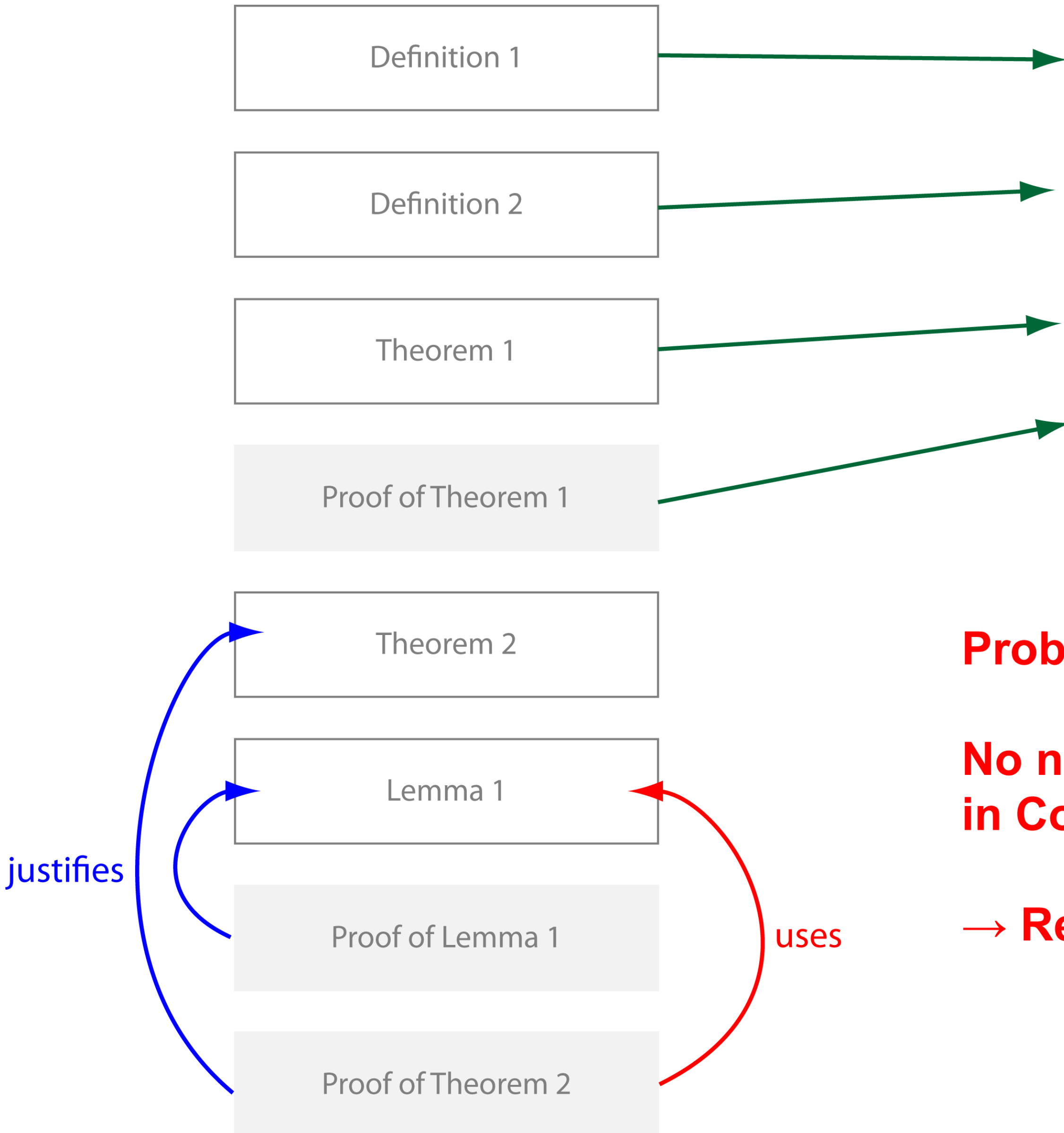
The automatic generation of a proof skeleton



The DG for the example



Straight-forward translation of the first part



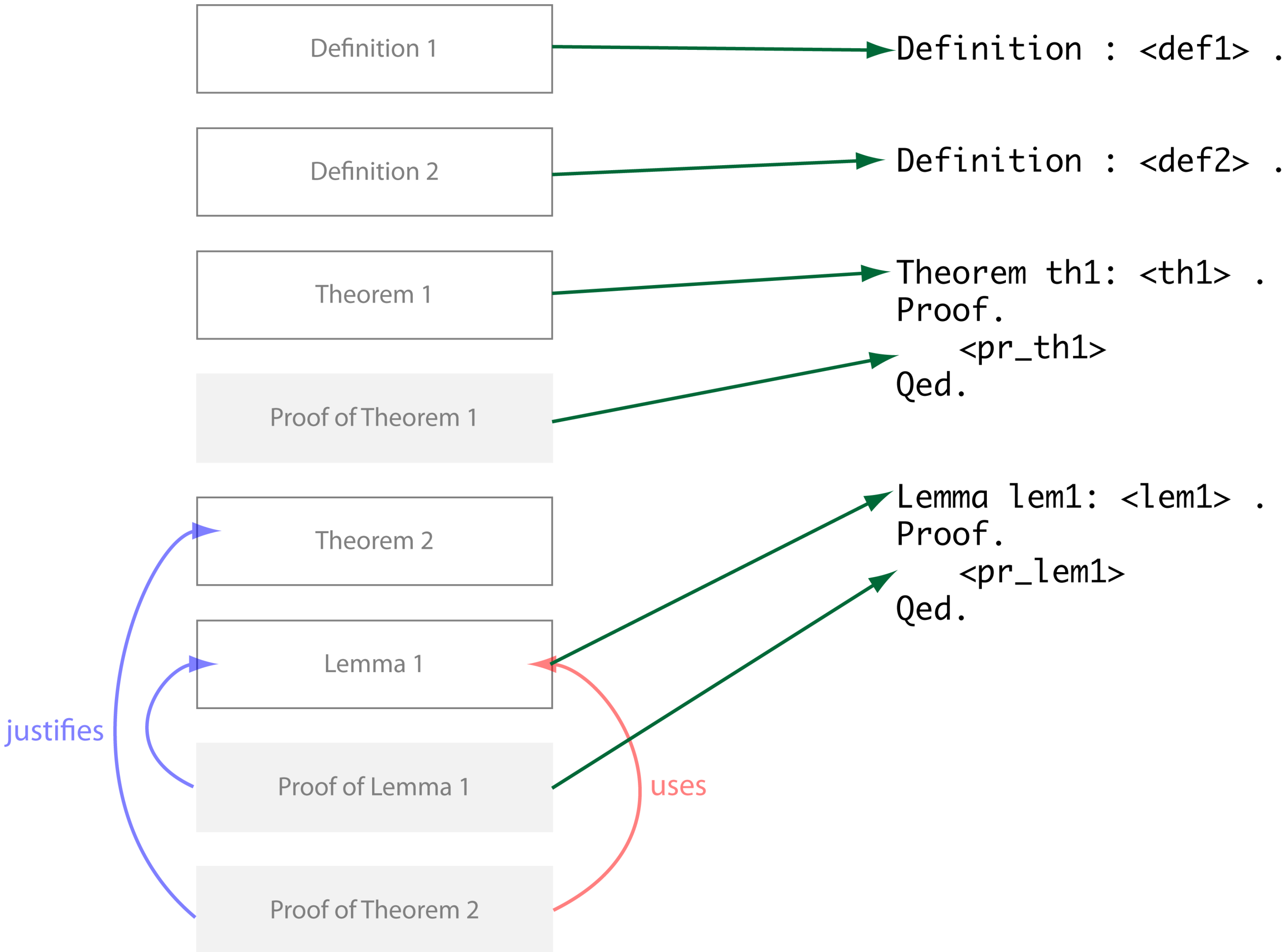

```
Definition : <def1> .

Definition : <def2> .

Theorem th1: <th1> .
Proof.
    <pr1>
Qed.
```

Problem

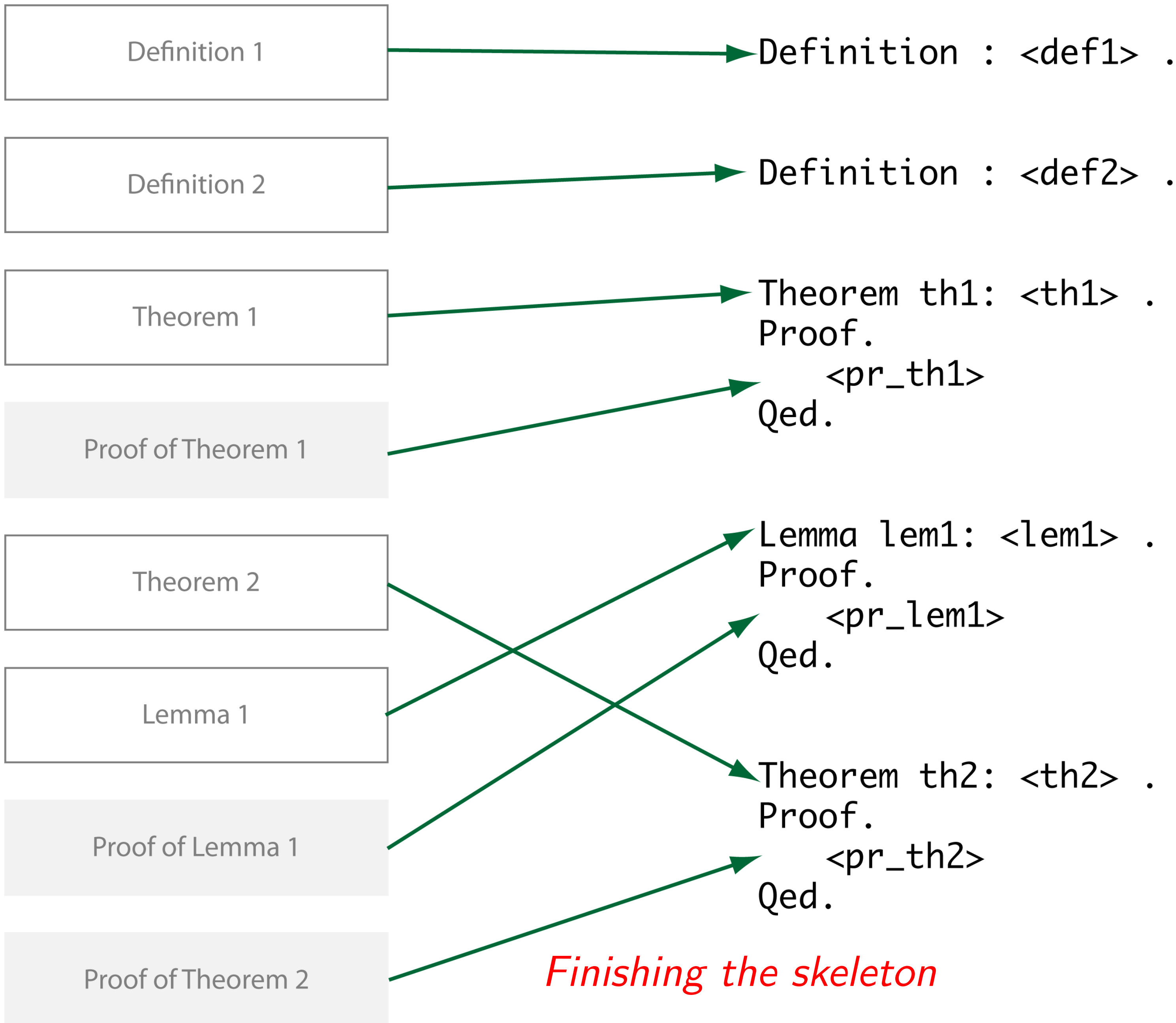
No nested theorems/lemmas in Coq e.g.

→ Re-Ordering !

Problem: nested theorems

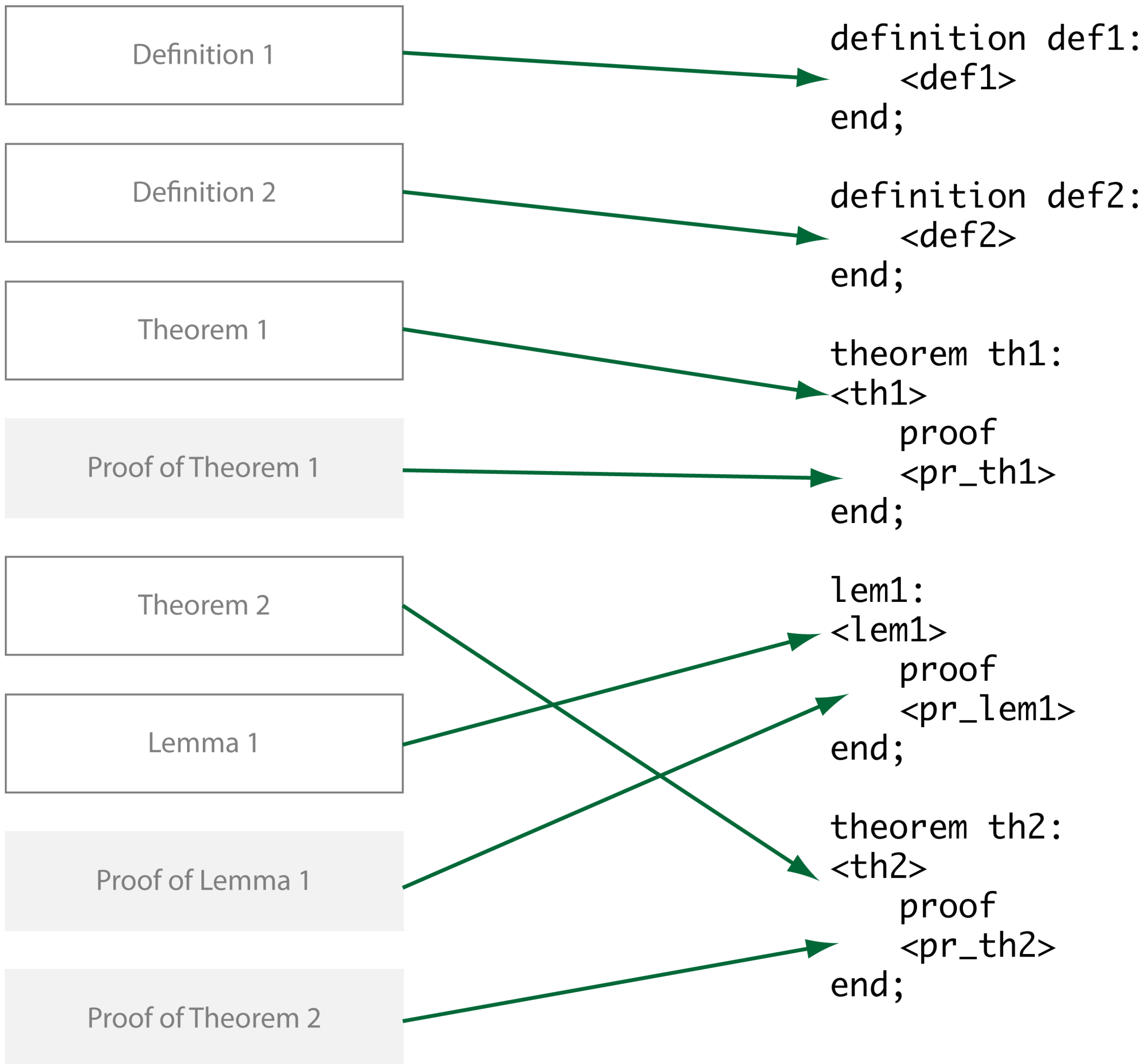


Solution: Re-ordering

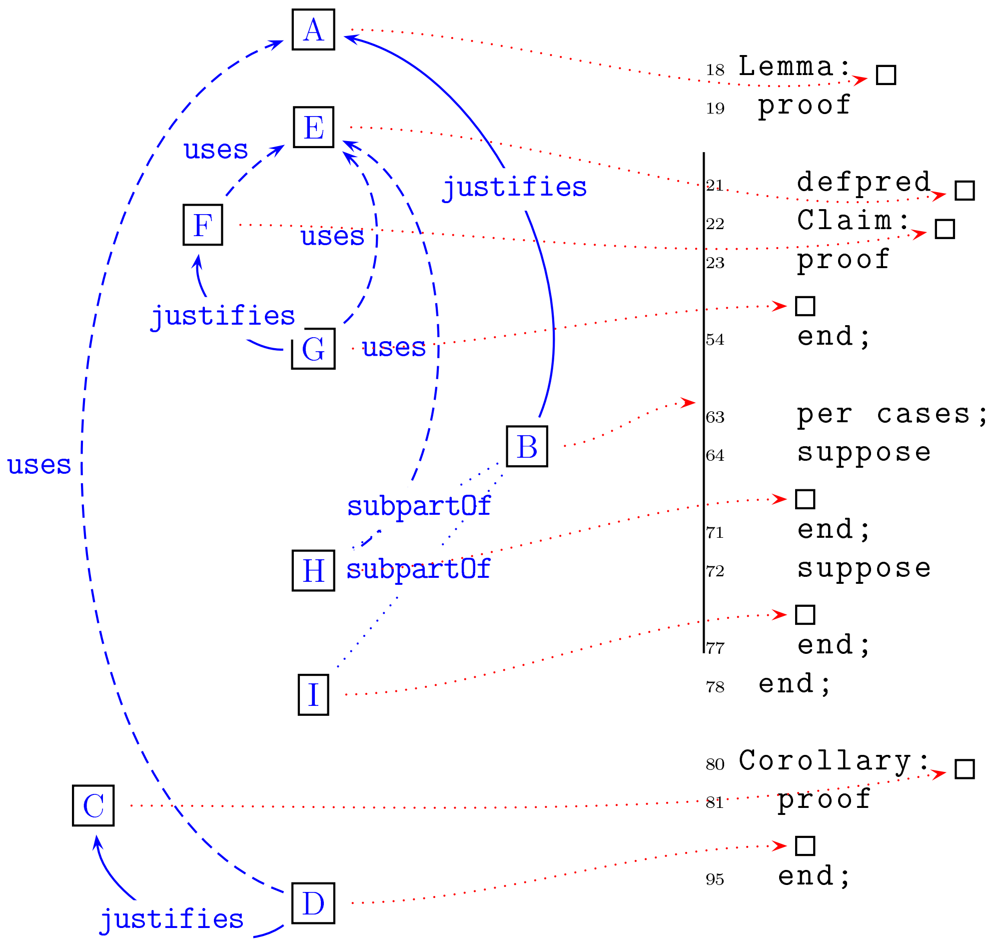


Finishing the skeleton

Skeleton for Mizar


Definition 1
Definition 2
Theorem 1
Proof of Theorem 1
Theorem 2
Lemma 1
Proof of Lemma 1
Proof of Theorem 2
definition def1:
<def1>
end;
definition def2:
<def2>
end;
theorem th1:
<th1>
proof
<pr_th1>
end;
lem1:
<lem1>
proof
<pr_lem1>
end;
theorem th2:
<th2>
proof
<pr_th2>
end;

DRa annotation into Mizar skeleton for Barendregt's example (Retel's PhD thesis)

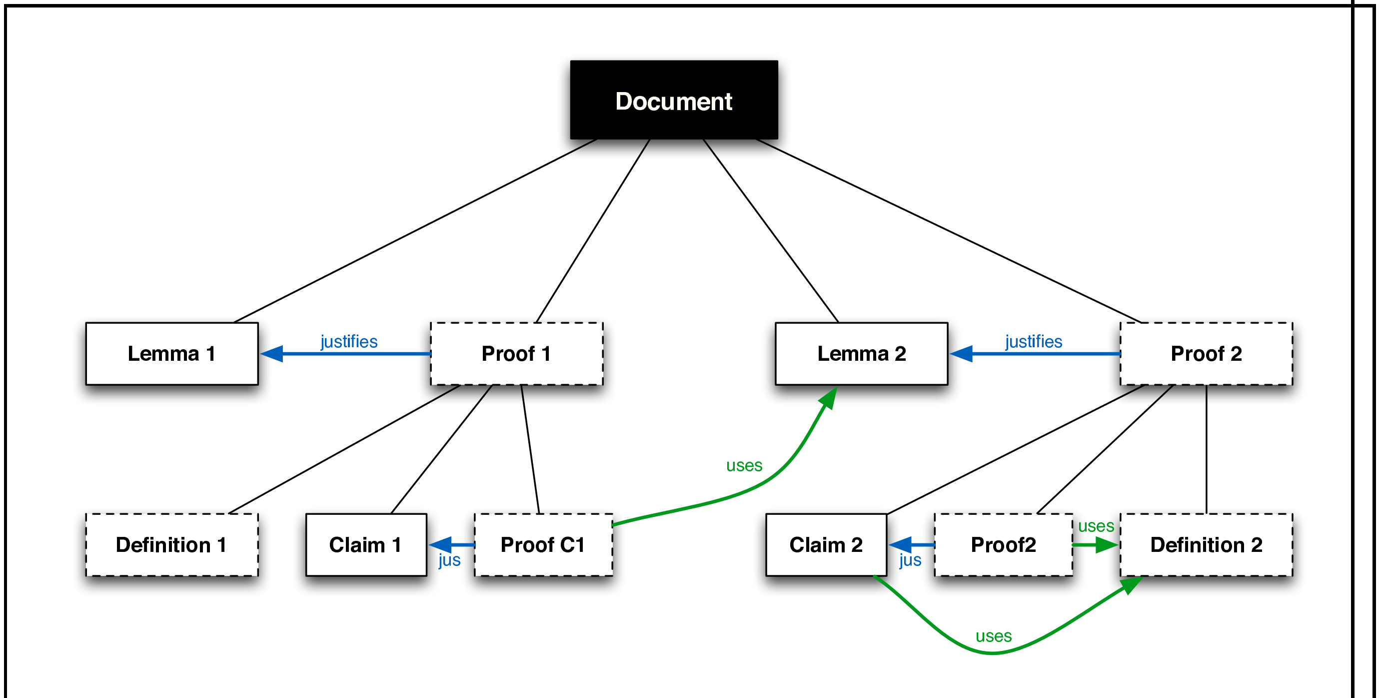


The generic algorithm for generating the proof skeleton (SGa, Zengler's thesis)

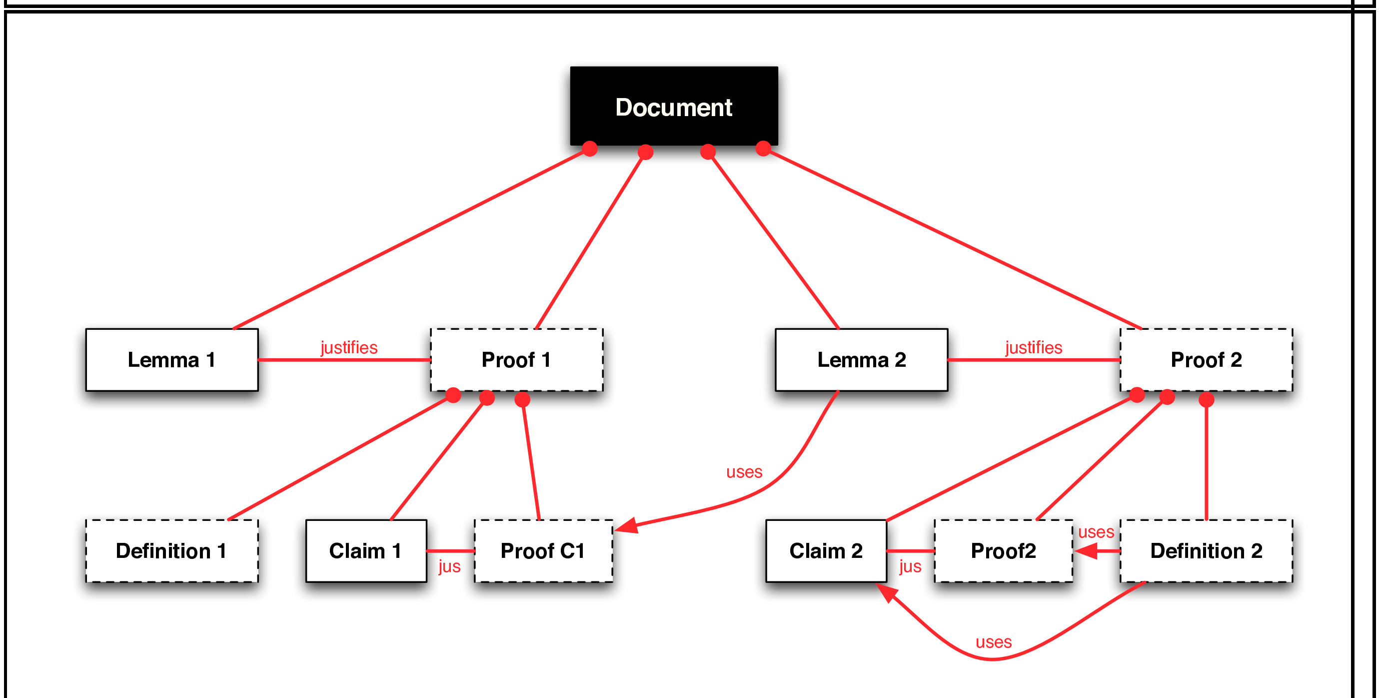
A vertex is ready to be processed iff:

- it has no incoming $\prec$ edges (in the GoTO) of unprocessed (white) vertices
- all its children are ready to be processed
- if the vertex is a proved vertex: its proof is ready to be processed

Consider the DG and GoTO of a (typical and not well structured) mathematical text:



Document
Lemma 1
justifies
Proof 1
Lemma 2
justifies
Proof 2
uses
Definition 1
Claim 1
jus
Proof C1
Claim 2
jus
Proof2
uses
Definition 2
uses



Document
Lemma 1
justifies
Proof 1
Lemma 2
justifies
Proof 2
uses
Definition 1
Claim 1
jus
Proof C1
Claim 2
jus
Proof2
uses
Definition 2
uses

The final order of the vertices is:

```
Lemma 2
Proof 2
   Definition 2
   Claim 2
   Proof C2
Lemma 1
Proof 1
   Definition 1
   Claim 1
   Proof C1
```

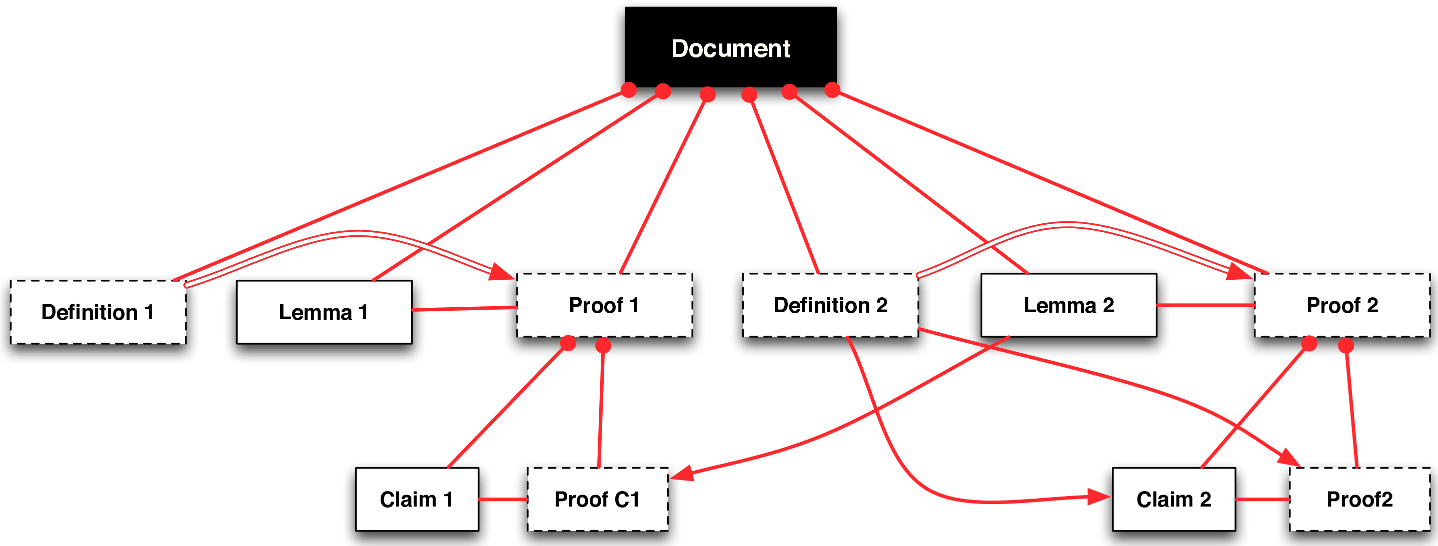


Figure 6: A flattened graph of the GoTO of figure 5 without nested definitions

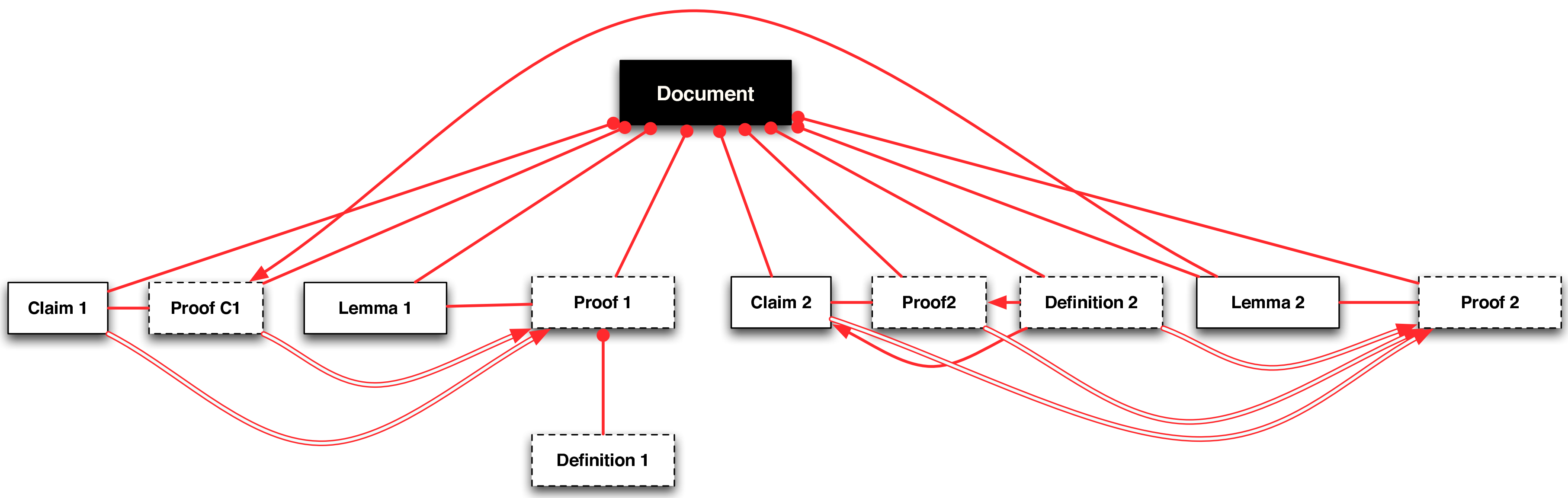


Figure 7: A flattened graph of the GoTO of figure 5 without nested claims

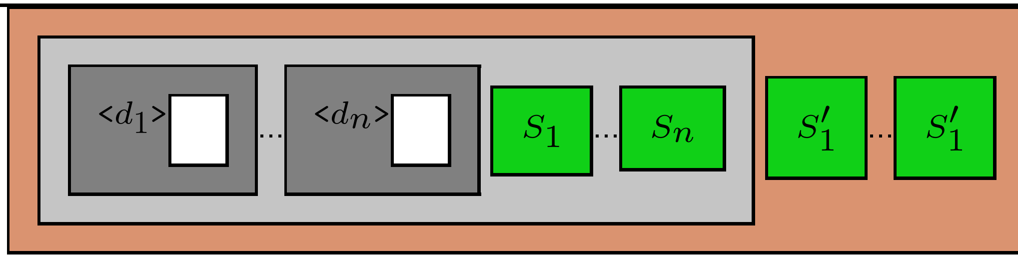
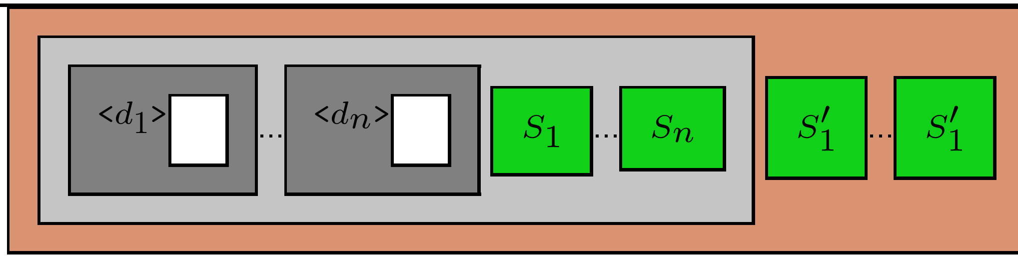
The Mizar and Coq rules for the dictionary

Role	Mizar rule	Coq rule
axiom	`%name : %body ;`	`Axiom %name : %body .`
definition	`definition %name : %nl %body %nl end;`	`Definition : %body .`
theorem	`theorem %name: %nl %body`	`Theorem %name %body .`
proof	`proof %nl %body  %nl end;`	`Proof %name : %body .`
cases	`per cases; %nl`	`%body`
case	`suppose %nl %body %nl end;`	`%body`
existencePart	`existence %nl %body`	`%body`
uniquenessPart	`uniqueness %nl %body`	`%body`

Rich skeletons for Coq

Rule N^O	Annotation ann	Coq translation $\mathcal{S}_{Coq}(ann)$
coq1)	<#>	`Set`
coq2)	<#>	`Prop`
coq3)	$<id>$ $<N>$	`id : N`
coq4)	$<id>$ $<S>$	`id : S`
coq5)	$<id>$	`id`
coq6)	$<id>$ $\boxed{p_1}$ ... $\boxed{p_n}$ $<N>$	`id :` $\mathcal{S}_{Coq}\left(\boxed{p_1}\right)$ `->` ... `->` $\mathcal{S}_{Coq}\left(\boxed{p_n}\right)$ `-> N`
coq7)	$<id>$ $\boxed{p_1}$ ... $\boxed{p_n}$ $<S>$	`id :` $\mathcal{S}_{Coq}\left(\boxed{p_1}\right)$ `->` ... `->` $\mathcal{S}_{Coq}\left(\boxed{p_n}\right)$ `-> S`

coq8)	<id> $\boxed{p_1}$ … $\boxed{p_n}$	`id :` $\mathcal{S}_{Coq}\left(\boxed{p_1}\right)$ `->` … `->` $\mathcal{S}_{Coq}\left(\boxed{p_n}\right)$ `-> Prop`
coq9)	<id> $\boxed{p_1}$ … $\boxed{p_n}$	`id :` $\mathcal{S}_{Coq}\left(\boxed{p_1}\right)$ `->` … `->` $\mathcal{S}_{Coq}\left(\boxed{p_n}\right)$ `-> Set`
coq10)	<id> $\boxed{p_1}$ … $\boxed{p_n}$	`(id` $\mathcal{S}_{Coq}\left(\boxed{p_1}\right)$ … $\mathcal{S}_{Coq}\left(\boxed{p_n}\right)$ `)`
coq11)	<id> $\boxed{p_1}$ … $\boxed{p_n}$	`(id` $\mathcal{S}_{Coq}\left(\boxed{p_1}\right)$ … $\mathcal{S}_{Coq}\left(\boxed{p_n}\right)$ `)`
coq12)	<id> $\boxed{p_1}$ … $\boxed{p_n}$	`(id` $\mathcal{S}_{Coq}\left(\boxed{p_1}\right)$ … $\mathcal{S}_{Coq}\left(\boxed{p_n}\right)$ `)`
coq13)	<id>	`id`
coq14)	<id> $\boxed{<id_1>}$ … $\boxed{<id_n>}$ $\boxed{e}$	`id id_1 ... id_n :=` $\mathcal{S}_{Coq}\left(\boxed{e}\right)$

coq15)	 for a surrounding unproved DRa annotation	`forall` $\mathcal{S}_{Coq}\left(\boxed{\langle d_1 \rangle}\right)$ ... $\mathcal{S}_{Coq}\left(\boxed{\langle d\right.$ `...` `/\` $\mathcal{S}_{Coq}\left(\boxed{S_n}\right)$ `->` $\mathcal{S}_{Coq}\left(\boxed{S'_1}\right.$
coq16)	 for a surrounding proved DRa annotation	$\mathcal{S}_{Coq}\left(\boxed{\langle d_1 \rangle}\right)$ ... $\mathcal{S}_{Coq}\left(\boxed{\langle d_n \rangle}\right)$ `/\` $\mathcal{S}_{Coq}\left(\boxed{S_n}\right)$ `->` $\mathcal{S}_{Coq}\left(\boxed{S'_1}\right)$ /

With these rules almost every axiom, definition and theorem can be translated in a way that it is immediately usable in Coq.

<definition><subsetDef>

<defSubSet><><subset><A> $A \subset$ <B> $B \iff$ <forall>$\forall$<><x> x, <impl><in><x> $x \in$ <A> $A \Rightarrow$ <in><x> $x \in$ <B> B

the left hand side of the definition is translated according to rule (coq14)) with `subset A B`.

The right hand side is translated with the rules coq5), coq10), coq11) and coq12) and the result is

```
forall x (impl (in x A) (in x B))
```

Putting left hand and right hand side together and taking the outer DRa annotation we get the translation

```
Definition subset A B := forall x (impl (in x A) (in x B))
```

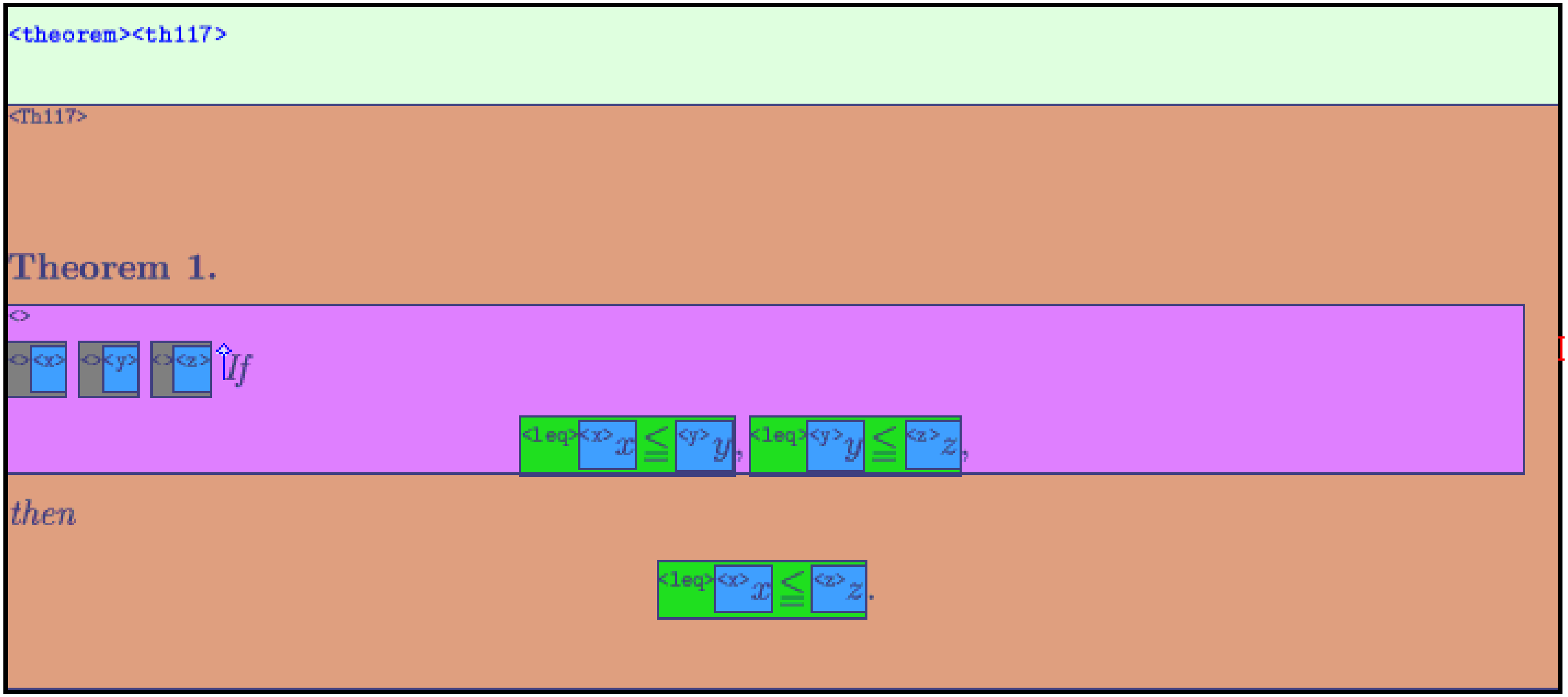
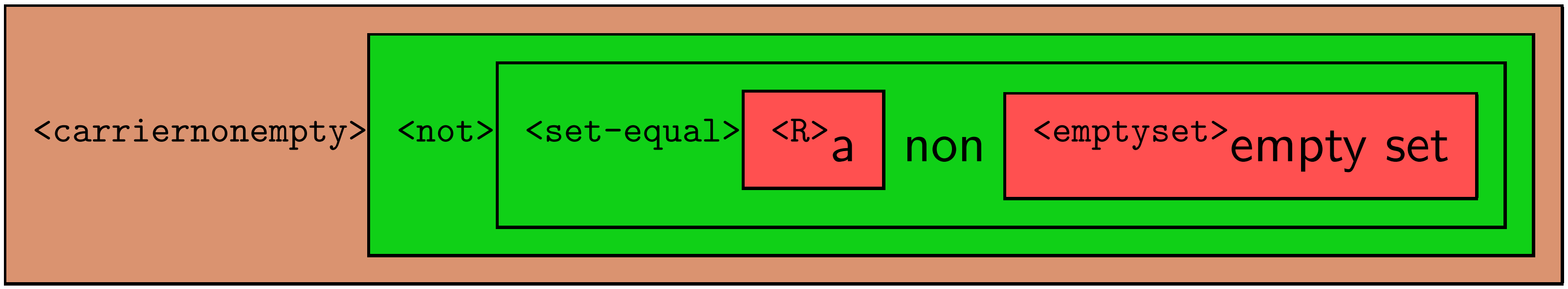


Figure 8: Theorem 17 of Landau's "Grundlagen der Analysis"

The automatic translation is:

```
Theorem th117 x y z : (leq x y /\ leq y z) -> leq x z .
```

Rich skeletons for Isabelle



The corresponding translation into Isabelle is:

```
assumes carriernonempty: "not (set-equal R emptyset)"
```

An example of a full formalisation in Coq via MathLang

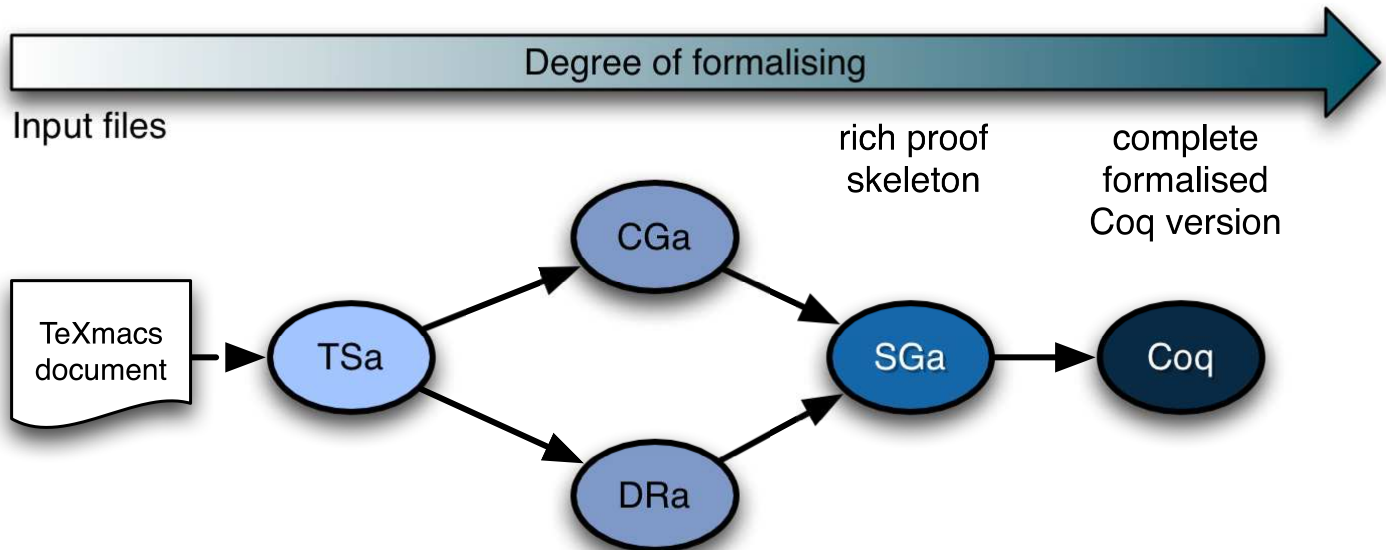


Figure 9: The path for processing the Landau chapter

<Th16>

Theorem 1.6. Commutative Law of Addition <><><x> <><y>

<eq><plus><x> x + <y> y = <plus><y> y + <x> x.

Figure 10: Simple theorem of the second section of Landau's first chapter

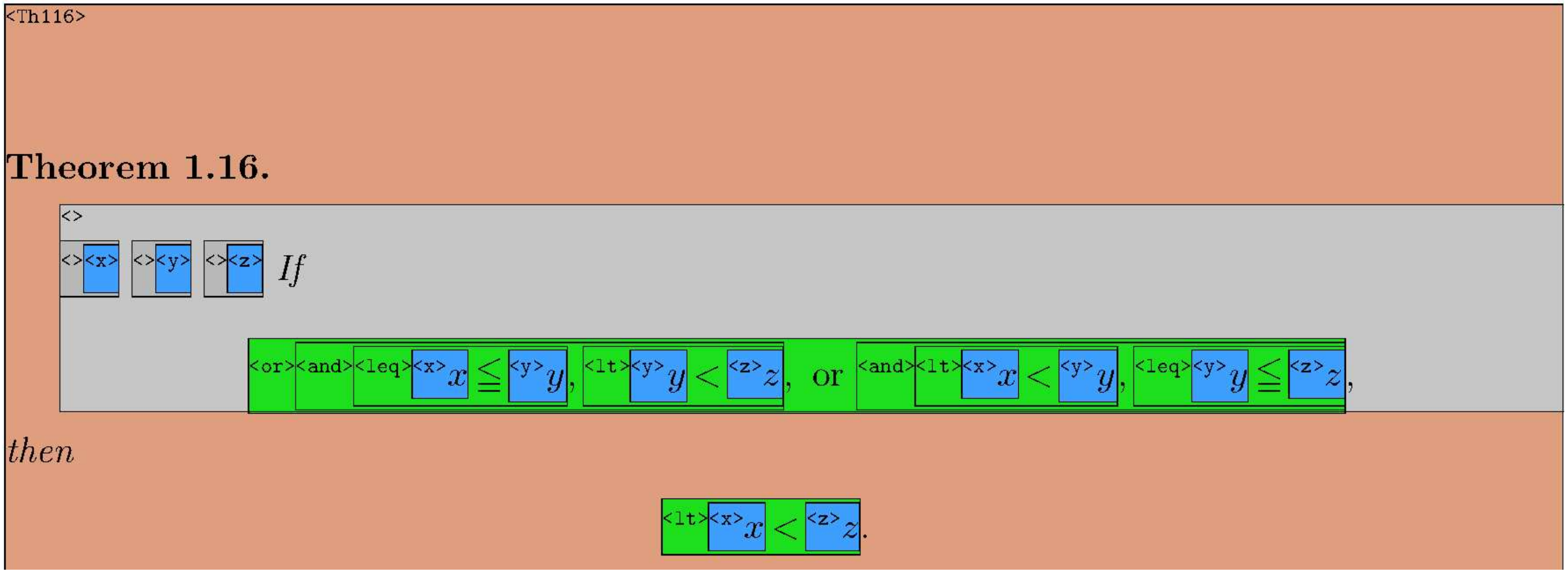


Figure 11: The annotated theorem 16 of the Landau's first chapter

Chapter 1

Natural Numbers

<><forall>$\forall$<#><#>. <#> <><exists>$\exists$<#><#>. <#> <><exists_one>$\exists!$<#><#>. <#> <><isa><#> <#> <><1> <><and><#> $\wedge$ <#>

<><or><#> $\vee$ <#> <><impl><#> $\Longrightarrow$ <#> <><succ><#> <><in><#> $\in$ <#> <><subset><#> $\subset$ <#> <><Set>$\{$<#><#> $|$ <#>$\}$

<><seteq><#><#> <><setneq><#><#> <><index><#><#> <><xor><#> $\oplus$ <#> <><emptyset>$\emptyset$

1.1 Axioms

We assume the following to be given:

<><N>A set (i.e. totality) of objects called <><natural_numbers>natural numbers, possessing the properties - called axioms- to be listed below.

Before formulating the axioms we make some remarks about the symbols $=$ and $\neq$ which be used.

Unless otherwise specified, small italic letters will stand for natural numbers throughout this book.

<>

<>If <><x>x is given and <><y>y is given, then either <><eq> <#>x and <#>y are the same number; this may be written

$$x = y$$

($=$ to be read "equals"); or <><neq><#>x and <#>y are not the same number; this may be written

$$1x \neq y$$

($\neq$ to be read "is not equal to").

Accordingly, the following are true on purely logical grounds:

<><forall><2><eq><x>$x =$ <x>x for every <1><><x>x

<><>if <><x> <><y> <eq><x>$x =$ <y>y then <eq><y>$y =$ <x>x

<><>If <><x> <><y> <><z> <eq><x>$x =$ <y>y, <eq><y>$y =$ <z>z then <eq><x>$x =$ <z>z

1

Chapter 1 of Landau:

- 5 axioms which we annotate with the mathematical role “axiom”, and give them the names“ax11” - “ax15”.
- 6 definitions which we annotate with the mathematical role “definition”, and give them names “def11” - “def16”.
- 36 nodes with the mathematical role “theorem”, named “th11” - “th136” and with proofs “pr11” - “pr136”.
- Some proofs are partitioned into an existential part and a uniqueness part.
- Other proofs consist of different cases which we annotate as unproved nodes with the mathematical role “case”.

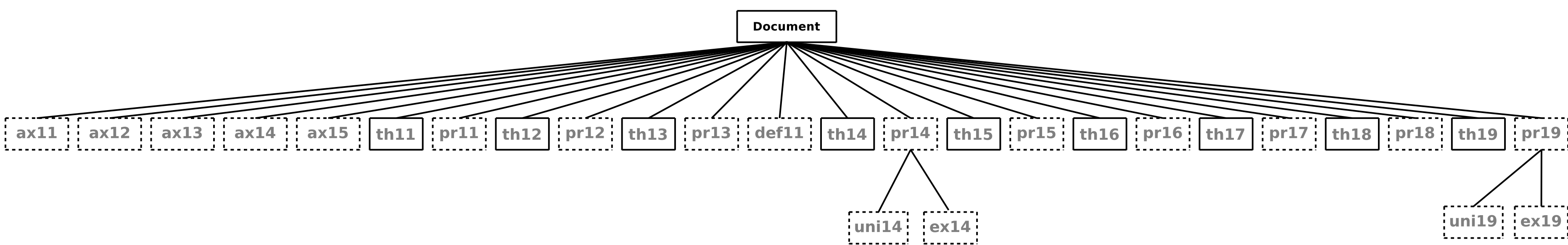


Figure 12: The DRa tree of sections 1 and 2 of chapter 1 of Landau’s book

- The relations are annotated in a straightforward manner.
- Each proof *justifies* its corresponding theorem.
- Axiom 5 (“ax15”) is the axiom of induction. So every proof which uses induction, *uses* also this axiom.
- Definition 1 (“def11”) is the definition of addition. Hence every node which uses addition also *uses* this definition.
- Some theorems *use* other theorems via texts like: “By Theorem ...”.
- In total we have 36 *justifies* relations, 154 uses relations, 6 caseOf, 3 existencePartOf and 3 uniquenessPartOf relations.
- The DG and GoTO are automatically generated.
- The GoTO is automatically checked and no errors result. So, we proceed to the next stage: automatically generating the SGa.

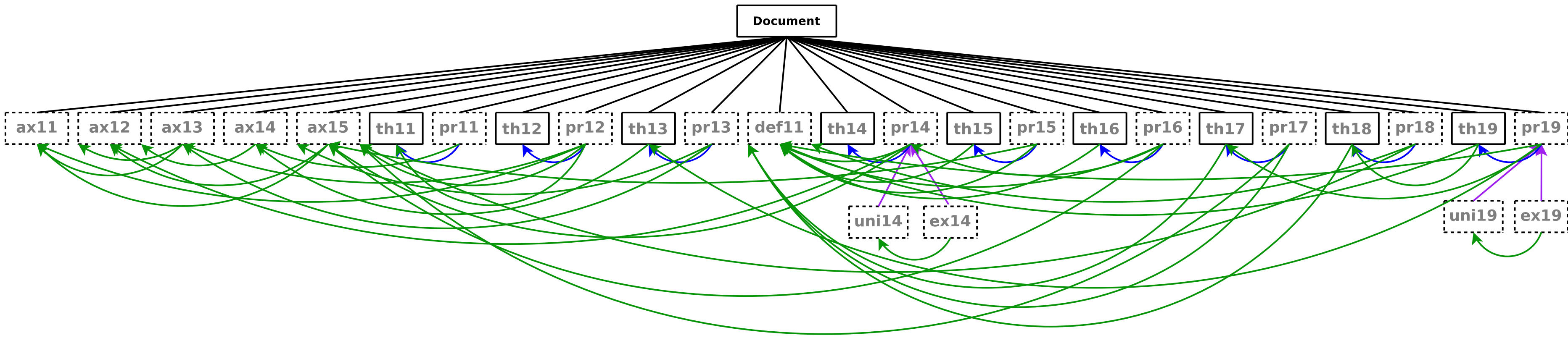
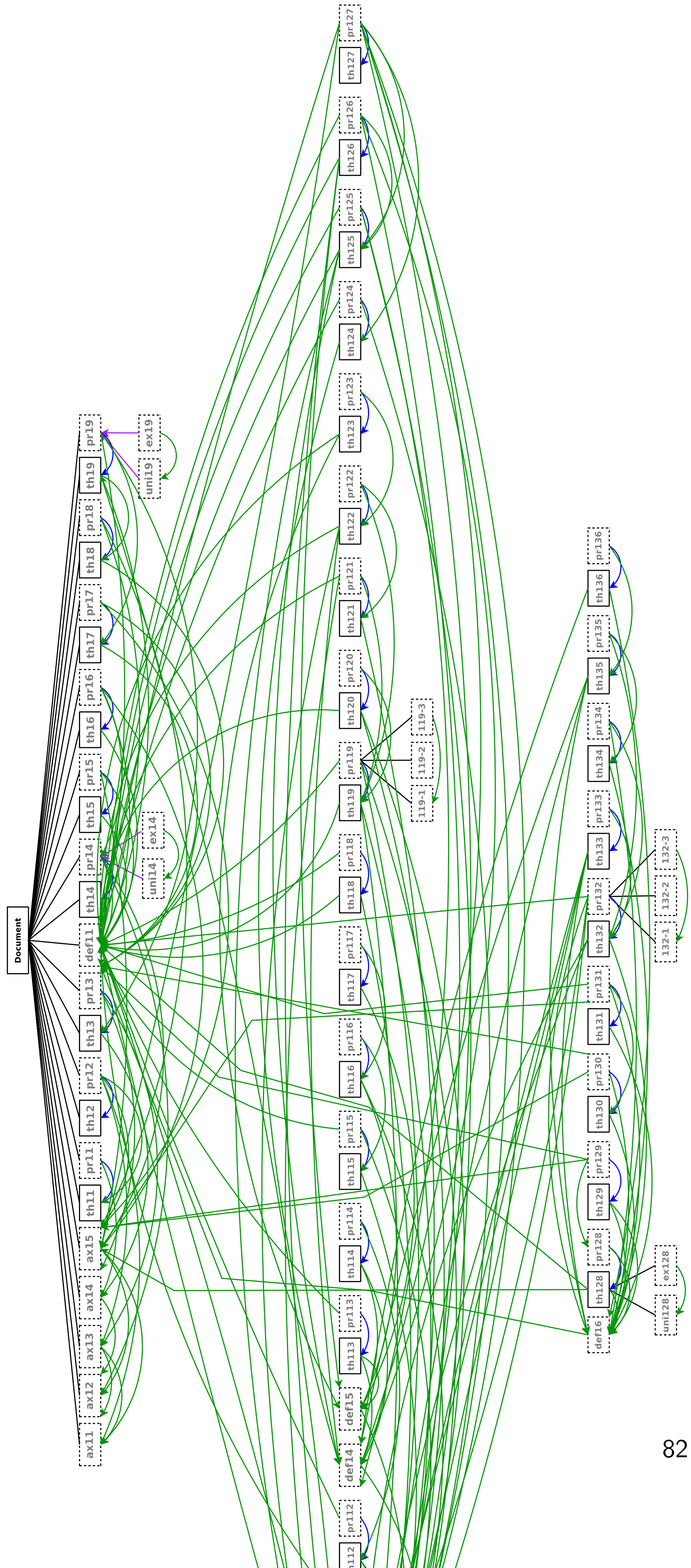
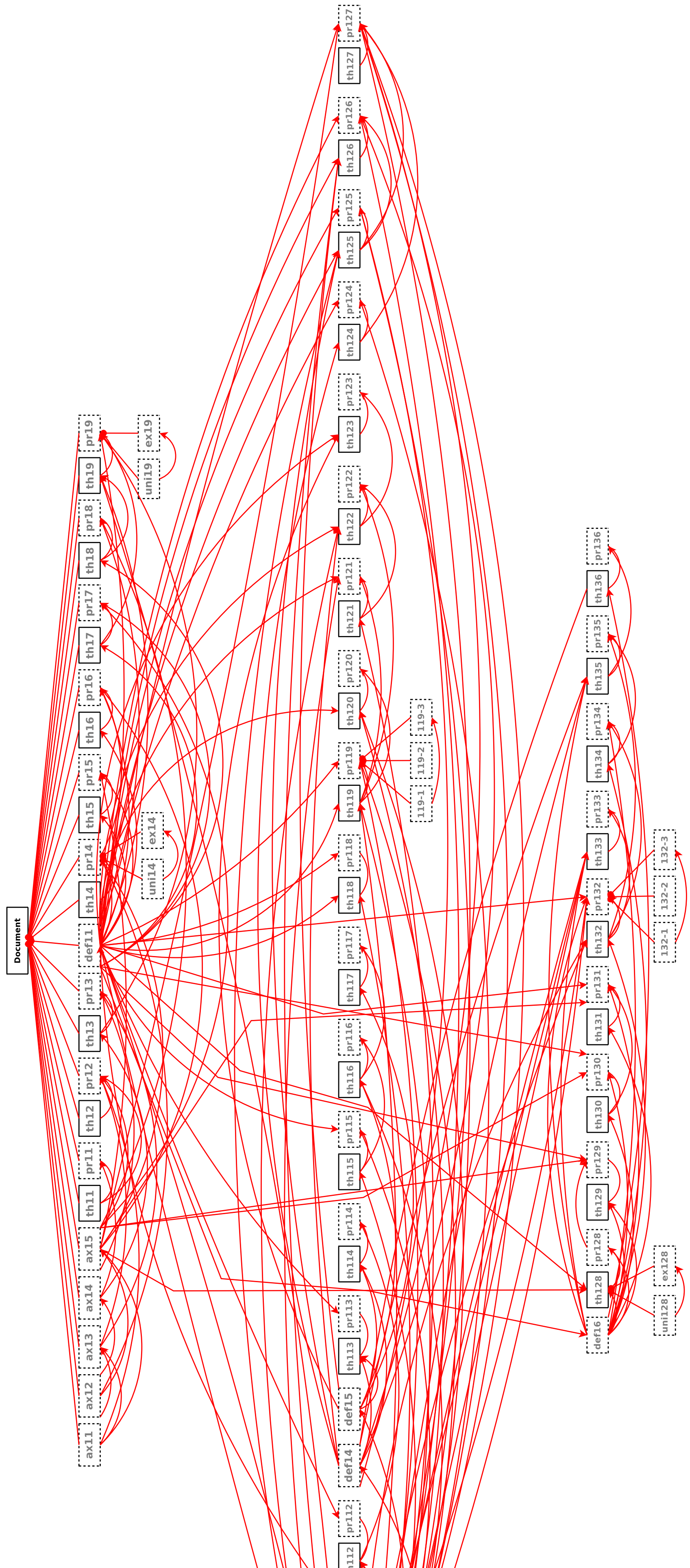


Figure 13: The DG of sections 1 and 2 of chapter 1 of Landau's book


Document
ax11
ax12
ax13
ax14
ax15
th11
pr11
th12
pr12
th13
pr13
def11
th14
pr14
uni14
ex14
th15
pr15
th16
pr16
th17
pr17
th18
pr18
th19
pr19
uni19
ex19
pr127
th127
pr126
th126
pr125
th125
pr124
th124
pr123
th123
pr122
th122
pr121
th121
pr120
th120
pr119
119-3
119-2
119-1
th119
pr118
th118
pr117
th117
pr116
th116
pr115
th115
pr114
th114
pr113
th113
def15
def14
pr112
pr136
th136
pr135
th135
pr134
th134
pr133
th133
pr132
132-3
132-2
132-1
th132
pr131
th131
pr130
th130
pr129
th129
pr128
th128
ex128
uni128
def16

The GoTO of section 1 - 4


Document
pr19
th19
pr18
th18
pr17
th17
pr16
th16
pr15
th15
pr14
th14
def11
pr13
th13
pr12
th12
pr11
th11
ax15
ax14
ax13
ax12
ax11
ex19
uni19
ex14
uni14
pr127
th127
pr126
th126
pr125
th125
pr124
th124
pr123
th123
pr122
th122
pr121
th121
pr120
th120
pr119
th119
pr118
th118
pr117
th117
pr116
th116
pr115
th115
pr114
th114
pr113
th113
def15
def14
pr112
119-3
119-2
119-1
pr136
th136
pr135
th135
pr134
th134
pr133
th133
pr132
th132
pr131
th131
pr130
th130
pr129
th129
pr128
th128
def16
132-3
132-2
132-1
ex128
uni128

An extract of the automatically generated rich skeleton

```
Definition geq x y := (or (gt x y) (eq x y)).
Definition leq x y := (or (lt x y) (eq x y)).

Theorem th113 x y : (impl (geq x y) (leq y x)).
Proof.
...

Qed.

Theorem th114 x y : (impl (leq x y) (geq y x)).
Proof.
...

Qed.

Theorem th115 x y z : (impl (impl (lt x y) (lt y z)) (lt x z)).
Proof.
...

Qed.
```

Completing the proofs in Coq

- We defined the natural numbers as an inductive set - just as Landau does in his book.

```
Inductive nats : Set :=
 | I : nats
 | succ : nats -> nats
```

- The encoding of theorem 2 of the first chapter in Coq is

```
theorem th12 x : neq (succ x) x .
```

- Landau proves this theorem with induction. He first shows, that $1' \neq 1$ and then that with the assumption of $x' \neq x$ it also holds that $(x')' \neq x'$.

- We do our proof in the Landau style. We introduce the variable x and eliminate it, which yields two subgoals that we need to prove. These subgoals are exactly the induction basis and the induction step.

```
Proof.
intro x. elim x.

2 subgoals
x : nats
______________________________________(1/2)
neq (succ I) I

______________________________________________________________(
forall n : nats, neq (succ n) n -> neq (succ (succ n)) (succ n)
```

Landau proved the first case with the help of Axiom 3 (for all x, $x' \neq 1$).

```
apply ax13.

1 subgoal
x : nats
______________________________________________________________(
forall n : nats, neq (succ n) n -> neq (succ (succ n)) (succ n)
```

The next step is to introduce n as natural number and to introduce the induction hypothesis:

```
intros n H.

1 subgoal
x : nats
n : nats
H : neq (succ n) n
______________________________________(1/1)
neq (succ (succ n)) (succ n)
```

We see that this is exactly the second case of Landau's proof. He proved this case with Theorem 1 - we do the same:

```
apply th11.

1 subgoal
x : nats
n : nats
```

```
H : neq (succ n) n
______________________________________(1/1)
neq (succ n) n
```

And of course this is exactly the induction hypotheses which we already have as an assumption and we can finish the proof:

```
assumption.
Proof completed.
```

The complete theorem and its proof in Coq finally look like this:

```
Theorem th12 (x:nats) : neq (succ x) x .
Proof.
intro x. elim x.
apply ax13.
intros n H.
apply th11.
assumption.
Qed.
```

With the help of the CGa annotations and the automatically generated rich proof skeleton, Zengler (who was not familiar with Coq) completed the Coq proofs of the whole of chapter one in a couple of hours.

Some points to consider

- We do not at all assume/prefer one type/logical theory instead of another.
- The formalisation of a language of mathematics should separate the questions:
  - *which type/logical theory is necessary for which part of mathematics*
  - *which language should mathematics be written in*.
- Mathematicians don't usually know or work with type/logical theories.
- Mathematicians usually *do* mathematics (manipulations, calculations, etc), but are not interested in general in reasoning *about* mathematics.
- The steps used for computerising books of mathematics written in English, as we are doing, can also be followed for books written in Arabic, French, German, or any other natural language.

- MathLang aims to support non-fully-formalized mathematics practiced by the ordinary mathematician as well as work toward full formalization.

- MathLang aims to handle mathematics as expressed in natural language as well as symbolic formulas.

- MathLang aims to do some amount of type checking even for non-fully-formalized mathematics. This corresponds roughly to grammatical conditions.

- MathLang aims for a formal representation of CML texts that closely corresponds to the CML conceived by the ordinary mathematician.

- MathLang aims to support automated processing of mathematical knowledge.

- MathLang aims to be independent of any foundation of mathematics.

- MathLang allows anyone to be involved, whether a mathematician, a computer engineer, a computer scientist, a linguist, a logician, etc.