Structural Analysis of Narratives with the Coq Proof Assistant

Anne-Gwenn Bosser[1], Pierre Courtieu[2], Julien Forest[3], and Marc Cavazza[1]

[1] University of Teesside, School of Computing
http://ive.scm.tees.ac.uk/
[2] Conservatoire National des Arts et Métiers, Laboratoire CEDRIC, Equipe CPR
http://cedric.cnam.fr/
[3] École nationale supérieure d'informatique pour l'industrie et l'entreprise, Laboratoire CEDRIC, Equipe CPR
http://cedric.cnam.fr/

Abstract. This paper proposes a novel application of Interactive Proof Assistants for studying the formal properties of Narratives, building on recent work demonstrating the suitability of Intuitionistic Linear Logic as a conceptual model. More specifically, we describe a method for modelling narrative resources and actions, together with constraints on the story endings in the form of an ILL sequent. We describe how well-formed narratives can be interpreted from cut-free proof trees of the sequent obtained using Coq. We finally describe how to reason about narratives at the structural level using Coq: by allowing to prove 2nd order properties on the set of all the proofs generated by a sequent, Coq assists the verification of structural narrative properties traversing all possible variants of a given plot.

Keywords: Applications of Theorem Provers, Linear Logic, Formal Models of Narratives

1 Introduction

The formalisation of narratives has attracted interest from researchers from many disciplines, not solely for their role as knowledge structures [24], but also for the challenges that their structural properties pose to logical representations [15, 17]. Narratives extend the logic of actions to provide a framework in which causal, temporal and resource consumption aspects are intertwined. Whilst the logical formalisation of actions has become a standard topic in philosophical logic and formal semantics, comparatively little work has addressed the structure of narratives. Initial hopes of developing computational narratology on the same basis as computational linguistics using narrative models developed in the field of humanities [2, 13] have failed due to narratology's formalisms being mostly content ontologies rather than logical or computational formalisms [3].

Addressing this problem from a new perspective, we have recently described in [1] how Linear Logic (LL) [10], and in particular Intuitionistic Linear Logic

(ILL) [11] can provide a suitable conceptual model for Narratives on a structural basis. Narratives are modelled as proofs of a sequent written in Linear Logic which describes initial resources and possible narrative actions. This allows to naturally express key properties for Narratives while supporting a return to first principles of narrative action representation (causality and competition for resources). This was not merely an attempt at logically encoding a given narrative: on the contrary the logical formulation supports the description of possible variants of the narrative, including alternative endings, which would be logically consistent. In other words, the ILL formalisation captures the essence of the narrative logic, not simply the accidentality of a given story. Since the manual exploration of proofs to discover story variants can be both tedious and error prone, we decided to support this exploration with proof assistants.

We propose here a first step towards the automation of the structural analysis of narratives using the Coq Proof-Assistant. We describe how to specify narratives on a structural basis only in the form of an ILL sequent, and a dedicated ILL encoding into Coq [4], with tactics allowing the discovery of proofs of such a sequent. We describe how such proofs are interpreted as well-formed narratives. Our encoding of ILL into Coq supports, as has previous work, the assisted generation of ILL proofs, but also assists reasoning about the properties of proofs and on all the possible proofs of an ILL sequent. This allows us to explore second order structural properties, traversing all the narratives which can be generated from a description of initial and atomic narrative actions and resources.

2 Related Works

2.1 Logical Approaches to Narratology

While most of the research in computational narratology has developed empirically, there have been a few logical and formal approaches to narratology, some of which are reviewed here.

A formal grammar for describing narratives has been proposed by [17], supporting the implementation of a system generating linear narratives and relying on temporal logic. While such generated narratives are not able to support an open-world assumption or to take into account the point of view of more than one protagonist, the approach shares with ours the emphasis on narrative causality description which is here embedded in the heart of the formalism.

Grasbon and Braun [12] have used standard logic programming to support the generation of narratives. However their system still relied on a narrative ontology (inspired from Vladimir Propps narrative funtions [23]), rather than on logical properties as first principles. Logic Programming has also been used in [26] for the generation of logically consistent stories. This character-based approach relies on argumentation models developed for autonomous agent systems

[4] Source available:
`http://cedric.cnam.fr/~courtiep/downloads/ill_narrative_coq.tgz`

for resolving the conflicts experienced by protagonists, while our more generic approach relies only on the description of narrative on the structural fundamentals which are action representation and competition for resources.

The concept of narrative action and its impact on the narrative environment is generally considered by narrative theories as the fundamental building block. Therefore AI formalisms dedicated to action semantic representation have been used previously for narrative action description, such as the situation calculus [21]. Linear Logic provides a very elegant solution to the frame problem by allowing the description of narrative actions using an action-reaction paradigm, avoiding the need to specify additional frame axioms.

The only previous use of LL in a closely related application has first been reported by [4], where the multiplicative fragment of LL is used for scenario validation. Their approach aims at a priori game/scenario design validation, through compilation into Petri Nets, with an emphasis on evidencing reachable states and dead-ends. While providing a relatively friendly computational model, such a fragment is not expressive enough for our purpose.

2.2 Related Applications of Linear Logic

Recent research in computational models of narratives has converged on the use of planning systems: typically, a planner is used to generate a sequence of actions which will constitute the backbone of the narrative [27]. On the other hand, Linear Logic has typically been used for action representation, and Masseron et al. [19, 20] have established how LL formalisation could support planning and how the fundamental properties of LL allows a proof in LL to be equated to a plan. While the Intuitionistic fragment of Linear Logic is undecidable, Dixon et al. [8,9] use proof-assistant technologies to build and validate plans for dialogues between agents in a Multi-Agents System. The approach we propose here goes, however, beyond the generation of a course of actions as we are interested in studying and verifying second order structural properties, transcending all the narratives which can be generated from a given specification relying on ILL.

While the computational properties of the fragment of Linear Logic we consider are an obstacle for the automation or semi-automation of proof-search (see [18] for a survey of decidability and complexity results for various fragments), the subset-language we use provides some restrictions and additional properties. This is similar to previous use of LL in the field of computational linguistics: [14] identifies usage patterns of LL in meaning assembly analysis [7] ensuring better complexity results than the full considered fragment.

2.3 Proof Assistants Support for LL

Previous work has proposed various encodings of fragments and variants of Linear Logic for proof assistants. In [22], authors present a shallow embedding of ILL in Coq and perform some simple generic proofs using induction. In [25] authors present a shallow embedding of the sequent calculus of classical modal linear logic and performs some *epistemic* proofs. In [16] an efficient and easy to use implementation of ILL rules in the Isabelle framework is presented. However our development focuses on properties of proofs (interpreted as narratives) themselves, not just on provability of sequents. As in these previous works we provide some (limited) automation for proving *closed sequents*, but we also provide reasoning lemmas and tactics for reasoning on properties of proofs and even on *all possible proofs of a sequent.*

More recently, Dixon et al [8] have proposed a formalisation of ILL in Isabelle focusing on the generation of verified *plans*. This is certainly the approach that is closest to ours, as it allows reasoning on plans themselves. A notable difference, due to the use of Isabelle, is that plans are added inside ILL rules formalisation. We do not need this artefact in our formalisation: narratives can be directly interpreted from pure ILL proof-terms. The relation between the shape of a proof and the properties of the corresponding narrative is, to our knowledge an original use of the proof-as-term paradigm.

3 ILL as a Representational Theory for Narratology

Our approach is based on a formal specification of narrative resources (including narrative actions), initial conditions, and possible ending states in the form of an ILL sequent. We then interpret a given proof of such a sequent as a narrative taking place in an open-world assumption. A sequent may have multiple proofs. It may therefore specify multiple narratives sharing the same initial resources and narrative actions. When interpreting the proof as a narrative, we look for a trace of the use of the $\multimap$ left rule. This rule is interpreted as the execution of a narrative action. Other rules have an interpretation reflecting the structure of the narrative, such as an external branching choice in an open-world assumption (for instance, end-user interaction), or a concurrency relationship between different subsets of the narrative with independent resource requirements.

3.1 Modelling of Narratives Specification through an ILL Sequent

The subset language of ILL we define here allows the description of the initial resources of the narrative, the available narrative actions, and constraints on the possible ending states of the narrative. Key to our interpretation, narrative actions are modelled using $\multimap$ which allows a precise description of their impact on the narrative environment. As we work in an open world assumption, external impact on the narrative (for instance user interaction) is modelled by using the $\oplus$ connector for describing choices between possible narrative actions, and by using & for describing a choice between two possible ending states.

Such a specification of narratives encompasses the description of the available resources and states of the narratives, the description of the semantic of narrative actions through their impact on the context of execution, and the possible ending states of the narrative. The initial sequent thus takes the form $\Gamma, \Delta \vdash \texttt{Goal}$, where Γ is a multiset representing resources and initial conditions, Δ is a multiset representing the possible narrative actions, and *Goal* a formula representing the possible ending state of the narrative. A sequent thus provides the knowledge representation base of a set of narratives.

While the description of the sequent places us in the whole ILL fragment, the modeling language we use for this paper imposes restrictions on the structure of formulae. This will in turn enforce properties on the generated proofs which can be verified using Coq. Figure 1 describes the subset-language of ILL used for describing narratives in this paper.

$\texttt{Res}::=1|\texttt{atom}|\texttt{Res}\&\texttt{Res}|\texttt{Res}\otimes\texttt{Res}|!\texttt{Res}$
$\texttt{Act}::=1|\texttt{CRes}\multimap\texttt{Context}|\texttt{Act}\oplus\texttt{Act}|\texttt{Act}\&\texttt{Act}|!\texttt{Act}$
$\texttt{Goal}::=1|\texttt{atom}|\texttt{Goal}\otimes\texttt{Goal}|\texttt{Goal}\oplus\texttt{Goal}|\texttt{Goal}\&\texttt{Goal}$

$\texttt{CRes}::=1|\texttt{atom}|\texttt{CRes}\otimes\texttt{CRes}$
$\texttt{Context}::=\texttt{Res}|\texttt{Act}|\texttt{Context}\otimes\texttt{Context}$

Fig. 1. Resources ($\texttt{Res} \in \Gamma$), Actions ($\texttt{Act} \in \Delta$) and Goal description for the $\Gamma, \Delta \vdash \texttt{Goal}$ sequent specifying a narrative. Actions consume resources of the form `RCons` and produce context formulae of the form `Context`

Resources of a Narrative Part of the left-hand side of the sequent consists of the description of the available resources and initial states. A resource can be an atom, 1, or composed. The formula $\texttt{Res}_1\&\texttt{Res}_2$, expresses the availability of one of the resources. One only of $\texttt{Res}_\texttt{i}$ will be used and the choice depends on the proof search, and can vary depending on the branches of the proof. This allows us to describe how the initial conditions can adapt to a given unfolding of the story. The formula $\texttt{Res}_1 \otimes \texttt{Res}_2$ allows to express the availability of both resources. The formula `!Res` allows to express the unbounded availability of the resource `Res`.

Narrative Actions Representation A simple narrative action is of the form $\texttt{CRes} \multimap \texttt{Context}$, where `Cres` is a finite resource description and `Context` a multiplicative conjunction of resources and actions. Its semantic is thus precisely defined in terms of how it affects the execution environment: to the execution of a narrative action in the narrative corresponds the application of the $\multimap$ left rule in the proof, consuming the resources modelled by `CRes` and introducing in the sequent context the formula `Context` which models resources and actions made available by this execution.

Narrative actions can be composed for offering two types of choices. A composed action $\texttt{Act}_1 \oplus \texttt{Act}_2$ corresponds to a choice between two possible actions, with both possibilities leading to well-formed alternative narratives. This is used for modelling the impact of events external to the narrative in an open-world assumption (for instance, user interaction). When such a formula is decomposed using the $\oplus$ rule, the two sub-proofs, which require proving the sequent with each of the sub-formula replacing the composed action, are interpreted as the two possible unfolding of the story. The proof thus ensures that each possible subsequent narrative is well-formed. A composed action $\mathtt{Act_1}\&\mathtt{Act_2}$ corresponds to a choice depending on the proof-search, and might differ depending on the branch of the proof. If both choices successfully produce a different proof of the sequent, this will be interpreted as two different narratives.

Narrative Ending States The resulting state of a narrative is modelled in the subset of ILL composed of $\otimes$ and $\oplus$ and &. $\mathtt{Goal_1} \otimes \mathtt{Goal_2}$ expresses that both $Goal_i$ states are accessible at the end of the narrative. $\mathtt{Goal_1} \oplus \mathtt{Goal_2}$ expresses that either $\mathtt{Goal_i}$ state is accessible, and the choice depends on the proof search and might differ depending on the unfolding branch of the narrative.$\mathtt{Goal_1}\&\mathtt{Goal_2}$ expresses that either state should be accessible, and this choice depends on an event external to the narrative, such as user interaction for instance. Similar to the situation with composed actions $\mathtt{Act_1} \oplus \mathtt{Act_2}$, the sub-proofs following the application of the & right rule will allow ensuring that the corresponding narratives are well-formed stories.

Stability of the Representation Given an ILL sequent respecting the grammar described on figure 1, all the sequents appearing in the proof will be of the form $\Gamma \vdash G$, where $\forall F \in \Gamma$, F is a `Context` formula and G is a `Goal` formula. In other words, modulo the application of the $\otimes$ left rule, all the sequents appearing in such a proof will be composed of a context describing resources and actions of a narrative, and of a right-hand side formula representing constraints on the ending state of the narrative.

More formally, we define the following properties on sequents and proofs:

Definition 1. *Let s be a sequent of the form $\Gamma \vdash G$, we say that s is* well formed *if $G \in$ `Goal` and $\forall F \in \Gamma, f \in$ `Context`. We shall write* $\mathrm{WF}(s)$.

Definition 2. *A property P on sequents is said to* hold for a proof *h of a sequent s if it holds for all sequents of the proof h above s. We shall note* $\mathrm{WF}(h)$.

Definition 3. WF *is stable for ILL, that is for any sequent s such that* $\mathrm{WF}(s)$ *and any proof h of s,* $\mathrm{WF}(h)$.

The proof of this property in Coq is described later in section 4.4.

3.2 Interpreting a Proof as a Narrative

Narratives are interpreted from proofs, from a structured trace of execution of the $\multimap$ left rule. Other ILL rules of particular significance for this interpretation are the $\oplus$ left, and $\otimes$ and & right rules. Narratives are obtained from proofs using the ν function described on figure 2. They are thus described using simple narrative actions (modelled in the initial sequent using the $\multimap$ connector), and the following list of operators:

- $\succ$ is a precedence relationship, defining a partial order on the execution of narrative actions: $\nu = \nu_1 \succ \nu_{action} \succ \nu_2$ is a narrative where the narrative ν_1 precedes the narrative action ν_{action} which precedes the narrative ν_2.
- ∇ is a branching of the narrative in an open-world assumption: $\nu = \nu_1 \nabla \nu_2$ is a narrative where both sub-narratives ν_1 and ν_2 are possible, but only one will actually be unfolded, depending on an external event in an open-world assumption (such as user interaction for instance).
- $\|$ represents a concurrency relationship between two narratives: $\nu = \nu_1 \| \nu_2$ is a narrative consisting of both ν_1 and ν_2 where the two sub-narratives will be unfolded concurrently and independently.

$$\frac{\Gamma \vdash A : \nu_1 \qquad \Delta, B \vdash C : \nu_2}{\Gamma, \Delta, A \multimap B \vdash C : \nu_1 \succ \nu_{A \multimap B} \succ \nu_2}\ (\multimap_{left}) \qquad \frac{}{\Gamma \vdash A : \emptyset}\ (\mathit{Leaf\ rule})$$

$$\frac{\Gamma \vdash A : \nu_1 \qquad \Delta \vdash B : \nu_2}{\Gamma, \Delta \vdash A \otimes B : \nu_1 \| \nu_2}\ (\otimes_{right}) \qquad \frac{\Gamma \vdash A : \nu}{\Gamma' \vdash A' : \nu}\ (\mathit{Unary\ rule})$$

$$\frac{\Gamma, A \vdash C : \nu_1 \qquad \Gamma, B \vdash C : \nu_2}{\Gamma, A \oplus B \vdash C : \nu_1 \nabla \nu_2}\ (\oplus_{left}) \qquad \frac{\Gamma \vdash A : \nu_1 \qquad \Gamma \vdash B : \nu_2}{\Gamma \vdash A \& B : \nu_1 \nabla \nu_2}\ (\&_{right})$$

Fig. 2. Proof to Narrative Interpretation Function ν: the function is defined recursively on sub-proofs from the last applied rule. $\nu_{A \multimap B}$ is the narrative action initially specified using the formula $A \multimap B$

3.3 From Narrative Specification to Narrative: an Example

We can now illustrate our approach through a complete example of narrative generation based on a specification of initial narrative resources and actions. This process is described in Figure 3. We use an extract of Flaubert's classical *Madame Bovary* novel [5]. We start from an identification of atomic resources and simple narrative actions: we add alternatives to some of the narrative actions occuring in the novel, inspired by each of the character's possible choices. Based on this identification, we model narrative context and goals as an ILL sequent. Composed actions and resources are defined, in order to reflect branching narrative possibilities depending on the impact of external events in an open-world assumption (for instance, the composed action $(G \multimap 1) \oplus (G \multimap S)$), to allow for the generation of different courses of actions (for instance, the action $(E \multimap A) \& 1$ can potentially generate a narrative where the narrative action corresponding to $E \multimap A$ occurs or not), and so on.

Once the sequent has been specified, we use Coq with simple tactics to generate a proof (sketched in Figure 3). The proof can then be interpreted as a given narrative.

[5] The plot of Madame Bovary is one naturally inclined to contain key decisions and prone to suggest what-if analyses

1. Narrative Resources and Actions:

Atomic resources	
P	Poison
R	A discussion with Rodolphe
B	A discussion with Binet
G	A discussion with Guillaumin

Atomic goals	
A	Emma is alive
M	Emma is dead

Simple Narrative Actions	
S⊸A	Emma sells herself which saves her life
E⊸A	Emma escapes with Rodolphe (which saves her life)
P⊸M	Emma ingests poison and dies
R⊸1	Emma has a conversation with Rodolphe. This does not alter her situation (non productive).
R⊸E	Emma talks to Rodolphe. They agree to escape together.
G⊸1	Emma has a conversation with Guillaumin. This does not alter her situation (non productive).
G⊸S	Emma discusses her situation with Guillaumin. As a result, Emma accepts to sell herself in exchange for Guillaumin's help.
B⊸1	Emma has a conversation with Binet. This does not alter her situation (non productive).
B⊸S	Emma discusses her situation with Binet. As a result, Emma accepts to sell herself in exchange for Binet's help..

2. Sequent Description

Initial Resources $\mathcal{R}$	$P\&1, R, G, B\&1$
Narrative actions $\mathcal{A}$	$!(S \multimap A), (E \multimap A)\&1, (P \multimap M)\&1, (R \multimap 1)\&(R \multimap E), (G \multimap 1) \oplus (G \multimap S), 1 \oplus ((B \multimap S)\&(B \multimap 1))$

3. Sketch of the proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\multimap_{left}\colon P \multimap M}{\multimap_{left}\colon R \multimap 1}}{\multimap_{left}\colon B \multimap 1} \quad \dfrac{\multimap_{left}\colon E \multimap A}{\multimap_{left}\colon R \multimap E}}{\oplus left : 1 \oplus ((B \multimap S)\&(B \multimap 1))}}{\multimap_{left}\colon G \multimap 1} \quad \dfrac{\dfrac{\dfrac{\multimap_{left}\colon S \multimap A \quad \dfrac{\multimap_{left}\colon S \multimap A}{\multimap_{left}\colon B \multimap 1}}{\oplus left : 1 \oplus ((B \multimap S)\&(B \multimap 1))}}{\multimap_{left}\colon R \multimap 1}}{\multimap_{left}\colon G \multimap S}}{\oplus_{left} : (G \multimap 1) \oplus (G \multimap S)}}{\mathcal{R}, \mathcal{A} \vdash A \oplus M}$$

4. Interpreted narrative:
$(\nu_{G\multimap 1} \succ ((\nu_{B\multimap 1} \succ \nu_{R\multimap 1} \succ \nu_{P\multimap M})\triangledown(\nu_{R\multimap E} \succ \nu_{E\multimap A})))\triangledown$
$(\nu_{G\multimap S} \succ \nu_{R\multimap 1} \succ (\nu_{S\multimap A}\triangledown(\nu_{B\multimap 1} \succ \nu_{\multimap A}))$

Fig. 3. Specification of a narrative environment into an ILL sequent, a corresponding proof obtained with Coq, and the corresponding interpreted narrative: The end of Emma Bovary

4 Using the Coq Proof Assistant for Narrative Properties Analysis

In this section, we will first describe how, based on our interpretation of proof as narratives and our ILL encoding into Coq, a proof assistant supports the building of coherent narratives from initial specifications.

This naturally leads one to wonder, given an initial specification, what are the characteristics of a *well-formed* narrative. In order to answer this, we need to be able to express properties regarding the set of all possible proofs of a given sequent, and to formally evidence structural properties which are verified by all the narratives generated by a given specification: we need to be able to express and prove properties by reasoning about proofs and sets of proofs.

As a proof assistant based on the Curry-Howard isomorphism, Coq is the tool of choice for studying such properties, as it allows to consider proofs as first class objects. We thus discuss in this section how we have been taking advantage of this proof-as-term paradigm in order to verify properties regarding all the proofs corresponding to narrative specifications as defined on figure 1, and to verify an example of structural property on the set of all the narratives generated by a given sequent specification.

4.1 ILL Encoding into Coq

Formulae, proofs and corresponding convenient (Unicode) notations are defined as follows. Type `env` is an instance of multisets equipped with an equivalence relation == and Vars.t (type of atomic propositions) is implemented[6] as ℕ in the following:

```
Inductive formula : Type :=
| Proposition : Vars.t → formula | Implies : formula → formula → formula
| Otimes : formula → formula → formula | One : formula
| Oplus : formula → formula → formula | Zero : formula | Top : formula
| Bang : formula → formula | And : formula → formula  → formula.

Notation "A ⊸ B" := (Implies A B).
(* ...Other connectives... *)
Notation  "x :: Γ" := (add x G). (* Environment operation *)
Notation  "x \ Γ" := (remove x G). (* Environment operation *)

Inductive ILL_proof: env → formula → Prop:=
|Id: ∀ Γ p, Γ == {p} → Γ ⊢ p
|Impl_R: ∀ Γ p q, p::Γ ⊢ q → Γ ⊢ p ⊸ q
|Impl_L: ∀ΓΔΔ'pqr,(p ⊸ q) ∈ Γ → (Γ\(p ⊸ q))==Δ ∪ Δ' → Δ⊢ p → q:: Δ'⊢ r → Γ ⊢ r
|Times_R: ∀ Γ Δ Δ’ p q, Γ == Δ ∪ Δ’ → Δ ⊢ p → Δ’ ⊢ q → Γ ⊢ p ⊗ q
|Times_L: ∀ Γ p q r, (p ⊗ q) ∈ Γ → q::p::(Γ\(p ⊗ q)) ⊢ r → Γ ⊢ r
|One_R: ∀ Γ, Γ == ∅ → Γ ⊢ 1
|One_L: ∀ Γ p, 1 ∈ Γ → (Γ\1) ⊢ p → Γ ⊢ p
```

[6] by functorial application

```
|And_R: ∀ Γ p q, Γ ⊢ p → Γ ⊢ q → Γ ⊢ (p & q)
|And_L_1: ∀ Γ p q r, (p & q) ∈ Γ → p::(Γ\(p&q)) ⊢ r → Γ ⊢ r
|And_L_2: ∀ Γ p q r, (p & q) ∈ Γ → q::(Γ\(p&q)) ⊢ r → Γ ⊢ r
|Oplus_L: ∀Γpqr, (p ⊕ q) ∈ Γ → p::(Γ\(p ⊕ q)) ⊢ r → q::(Γ\(p ⊕ q)) ⊢ r → Γ ⊢ r
|Oplus_R_1: ∀ Γ p q, Γ ⊢ p → Γ ⊢ p ⊕ q
|Oplus_R_2: ∀ Γ p q, Γ ⊢ q → Γ ⊢ p ⊕ q
|T_: ∀ Γ, Γ ⊢ ⊤
|Zero_: ∀ Γ p, 0 ∈ Γ → Γ ⊢ p
|Bang_D: ∀ Γ p q, !p ∈ Γ → p::(Γ\(!p)) ⊢ q → Γ ⊢ q
|Bang_C: ∀ Γ p q, !p ∈ Γ → !p::Γ ⊢ q → Γ ⊢ q
|Bang_W: ∀ Γ p q, !p ∈ Γ → Γ\(!p) ⊢ q → Γ ⊢ q
where " x ⊢ y ":= (ILL_proof x y).
```

Notice the use of the form "$\phi \in \Gamma \to \Gamma \vdash \ldots$" instead of "$\phi, \Gamma \vdash \ldots$". This formulation avoids tedious manipulations on the environment to match rules. Simple tactics allow to apply rules and premisses of the form $\phi \in \Gamma$ are discharged automatically (on closed environments) by reduction. As we are looking for a trace of the execution of the narrative actions through the application of the $\multimap$ left rule, we are only searching for cut-free proofs and thus do not provide the Cut rule.

The Coq command `Program Fixpoint` allows to define rather easily dependently typed fixpoints and pattern matchings on terms of type `x ⊢ y`. For instance one can define the ν function described in section 3.2 as follows:

```
Program Fixpoint ν Γ φ (h: Γ ⊢ φ) {struct h}: narrative :=
match h with
|Id _ _ p ⇒ ∅
|Impl_R Γ p q x ⇒ ν _ _ x
|Impl_L Γ Δ Δ' p q r _ _ x x0 ⇒ (ν Δ p x) ≻ ([Implies p q] ≻ (ν (q::Δ') r x0))
|Times_R Γ Δ Δ' p q heq x x0 ⇒ (ν Δ p x) || (ν Δ' q x0)
   ...
end.
```

4.2 Well-Formed Narrative Generation: Proving an ILL Sequent in Coq

Our encoding of ILL into Coq can be used simply with the aim of generating a *well-formed* story, from a sequent specification. We provide a set of simple tactics assisting the user in unfolding a proof, thus constructing a proof-term which will subsequently be interpreted as a narrative.

As an example, let us consider the sequent given below presented in figure 3, corresponding to the end of Emma Bovary.

```
Goal Emma: {P&1, R, G, B&1, !(S ⊸ A), (E ⊸ A)&1, (P ⊸ M)&1,
              (R ⊸ 1)&(R ⊸ E), (G ⊸ 1) ⊕ (G ⊸ S), 1⊕(B ⊸ S)&(B ⊸ 1)} ⊢ A ⊕ M.
```

One can, for example, apply the $\oplus_L$ rule to consider the alternative offered by external choice `(G ⊸ 1) ⊕ (G ⊸ S)`. This is achieved by tactic: `oplus_l (G ⊸ 1) (G ⊸ S)` that leaves with two subgoals. The first one is `{G ⊸ 1, P&1, R, G, ... } ⊢ A ⊕ M` and allows for rule $\multimap_l$ rule to perform a narrative action consuming `G` using tactic: `impl_l G 1`.

The succesful proof of this sequent unravels the narrative structure by only producing the set of alternative actions consistent with the baseline plot description.

4.3 Stability of an ILL Narrative Sequent: Well-Formed Sequents

The subset-language we defined in ILL for modelling narratives (figure 1) defines strong properties regarding any proof of the sequent. In fact, as we have briefly mentioned in section 3.1 when defining formally property 3, all the sequents appearing in the proof will respect the same subset-language, modulo the use of the $\otimes$ connector on the left. Our model is stable, reflecting narrative composition as each sequent appearing in the proof actually corresponds to a narrative specification, and the subset language of ILL we have defined earlier on figure 1 can thus be used as a heuristics. This is an important property as it allows to disregard the use of certain ILL rules (for instance $\multimap$ right) when our subset-language is used for the modelling, and has implications in terms of sub-formula properties as well. It can thus simplify the verification of narrative properties.

In order to use this fact, we provide a proof of stability of WF for ILL (property 3). To this end, we prove the stability of WF for each rule as follows: first the grammar of figure 1 is defined by the follwing (mutual) inductive properties:

```
Inductive Act : formula → Prop := (* Act *)
| A1: Act 1
| A2:∀ ϕ1 ϕ2, Cres ϕ1 → Context ϕ2 → Act (ϕ1 ⊸ ϕ2)
| A3: ∀ ϕ1 ϕ2, Act ϕ1 → Act ϕ2 → Act (ϕ1 ⊕ ϕ2)
| A4: ∀ ϕ1 ϕ2, Act ϕ1 → Act ϕ2 → Act (ϕ1 & ϕ2)
| A5: ∀ ϕ, Act ϕ → Act (!ϕ)
with Cres: formula → Prop:= (* CRes *)
| Cres1: Cres 1
| Cres2: ∀ n, Cres (Proposition n)
| Cres3: ∀ ϕ1 ϕ2, Cres ϕ1 → Cres ϕ2 → Cres (ϕ1 ⊗ ϕ2)
with Context: formula → Prop:= (* Context *)
| Context1:∀ ϕ, Act ϕ → Context ϕ
| Context2:∀ ϕ, Res ϕ → Context ϕ
| Context3: ∀ ϕ1 ϕ2, Context ϕ1 → Context ϕ2 → Context (ϕ1 ⊗ ϕ2)
with Res: formula → Prop:= (* Res *)
  R1: Res 1
| R2: ∀ n, Res (Proposition n)
| R3: ∀ ϕ, Res ϕ → Res (!ϕ)
| R4: ∀ ϕ1 ϕ2, Res ϕ1 → Res ϕ2 → Res (ϕ1 ⊗ ϕ2)
| R5: ∀ ϕ1 ϕ2, Res ϕ1 → Res ϕ2 → Res (ϕ1 & ϕ2).

Inductive Goal : formula → Prop :=
| G1: Goal 1
| G2: ∀ n, Goal (Proposition n)
| G3: ∀ ϕ1 ϕ2, Goal ϕ1 → Goal ϕ2 → Goal (ϕ1 ⊗ ϕ2)
| G4: ∀ ϕ1 ϕ2, Goal ϕ1 → Goal ϕ2 → Goal (ϕ1 ⊕ ϕ2)
| G5: ∀ ϕ1 ϕ2, Goal ϕ1 → Goal ϕ2 → Goal (ϕ1 & ϕ2).
```

```
Definition Contextall Γ f (_:Γ⊢f):= Goal f ∧∀ g:formula, g∈Γ → Context g.
```

then the stability for each rule is given by an inductive property `Istable` mirroring the type of proofs, stating that a property `pred` holds for all premisses sequents of all rules.

```
(** This property is true when pred holds for all nodes above the root
   of h (it has not to hold for the root itself). *)
Inductive Istable: ∀ {e} {f} (h: e ⊢ f) , Prop :=
| IId: ∀ Γ p heq, Istable (Id Γ p heq)
| IImpl_R: ∀ Γ p q h, pred h → Istable h → Istable (Impl_R Γ p q h)
| IImpl_L: ∀ Γ Δ Δ' p q r hin heq h h', pred h → pred h'
       → Istable h → Istable h' → Istable (Impl_L Γ Δ Δ' p q r hin heq h h')
| ...
```

The stability of the grammar is then stated as the following lemma:

```
Lemma Grammar_Stable : ∀ Γ φ (h:Γ ⊢ φ), Contextall h → Istable Contextall h.
```

It is proved by induction on `h`.

4.4 Second Order Analysis of Narratives Specification

We consider in this section the reachability of a given ending state regardless of the impact of external events in an open-world assumption, as an example of interesting structural property. When considering a given proof (and narrative), this property is not difficult to check. We build a (dependently typed) function checking this property for a closed proof. That is, at least one branch contains an application of the $\oplus_{R1}$ rule with `... ⊢ A` on the right premise:

```
Program Fixpoint check φl φr {e} {f} (h: e ⊢ f) {struct h}: boolP :=
match h with
 | One_R _ _ ⇒ falseP
 | One_L Γ p _ x ⇒ check φl φr x
 | Oplus_R_1 Γ p q x ⇒
    if andP (p ?= φl) (q ?= φr) then trueP else check φl φr x
 | ...
end.
Eval vm_compute in check A M Emma. (* true *)
```

where `trueP`, `falseP`, `andP` etc are boolean over sort `Prop` and related operations, and `?=` is the equality decision over formulae.

What would be much more interesting from a structural analysis point of view would be to prove that this property is valid regardless of the proof of the sequent (`check` should return `trueP` for *any proof of* `Emma`). In our interpretation, this would mean that for every narrative possibly generated by the initial specification, a given end state is reachable. This is much more difficult to prove using Coq as there is a potentially infinite set of such proofs. We tackle this problem using a variety of means. First, we define a notion of equivalence between proofs. Second, we define an incremental method to avoid proving several times the same property. A description of these two techniques follows.

Identify Proofs Corresponding to the Same Tree We identify proofs that differ only by the way side premisses (like `p` $\in$ Γ or Γ $\in$ Δ ∪ Δ') are proven. We then prove that `check` and other definitions are compatible with this equivalence.

```
Inductive eq: ∀ Γ Γ' f, (Γ ⊢ f) → (Γ' ⊢ f) → Prop :=
| EQId: ∀ Γ1 Γ2 f heq heq', eq (Id Γ1 f heq) (Id Γ2 f heq')
| EQImpl_R:∀ Γ1 Γ2 p q h h', eq h h' → eq (Impl_R Γ1 p q h) (Impl_R Γ2 p q h')
| EQTimes_R: ∀ Γ1 Δ1 Δ1' Γ2 Δ2 Δ2' p q heq heq' h1 h1' h2 h2',
  eq h1 h1' → eq h2 h2' →
  eq (Times_R Γ1 Δ1 Δ1' p q heq h1 h2) (Times_R Γ2 Δ2 Δ2' p q heq' h1' h2')
| ...

Lemma eq_compat_check : ∀ f1 f2 Γ Γ' ϕ (h1:Γ⊢ϕ) (h2:Γ'⊢ϕ),
                                      eq h1 h2 → check f1 f2 h1 = check f1 f2 h2.
```

An interesting consequence is that one can substitute an environment with a provably equal one in our proofs:

```
Lemma eq_env_compat_check : ∀ f1 f2 Γ Γ' ϕ (h1:Γ⊢ϕ) (h2:Γ'⊢ϕ),
                                      eq Γ Γ' → check f1 f2 h1 = check f1 f2 h2.
```

This lemma is heavily used in our automated tactics described in the next section.

Property Validation on All the Proofs of a Sequent We can also prove that some property holds for all proofs of a given closed sequent. We developed a method for this kind of proofs which can be automated using an external tool. This method should work for properties than one can define as a boolean function over ILL proofs (i.e. of type $\forall \Gamma \phi, \Gamma \vdash \phi \to$ `boolP`). This method involves intricate lemmas and tactics allowing to explore all possible proofs of a sequent. This amounts in particular to detect unprovable goals as soon as possible. This is made possible in some cases by generic lemmas about the unprovability of a sequent $\Gamma \vdash \phi$. For instance the following lemma is proved and used:

Lemma 1 (`unusable_var_in_env`). *If a variable $v \in \Gamma$ does not appear in the left-hand side of a $\multimap$ in any (sub-)formula of Γ and do not appear in ϕ, then $\Gamma \vdash \phi$ has no proof.*

Our tactics detect such patterns in the hypothesis of a goal g and discharge g by absurdity. The proof strategy applies to a goal G of the form: $\forall h : \Gamma \vdash \phi, f\ h =$ `trueP` and proceeds by building a database of previously proved lemmas as described in algorithm 1. The use of the lemmas database prevents proving the same lemma twice. The application of previous lemmas is eased by the use of `eq_env_compat_check` (described in previous section). Using this method we manage to prove several non trivial properties, including the reachability property mentioned earlier.

We have modelled the following simple narrative actions, which give another perspective on the end of Madame Bovary:

B⊸S	A discussion with Binet: Emma accepts the idea of selling herself
B⊸R	A discussion with Binet: Emma decides to go and see Rodolphe
G⊸B	A discussion with Guillaumin: Emma decides to go and see Binet
G⊸S	A discussion with Guillaumin: Emma accepts the idea of selling herself

```
$M(\Gamma \vdash \phi)$:
foreach rule $r$ applicable to $\Gamma \vdash \phi$ do
    foreach sequent $\Delta \vdash \psi$ of the premisses of $r$ do
        if f h = trueP then OK;
        else if $\Delta \vdash \psi \in DB$ then apply $DB(\Delta \vdash \psi)$ and OK;
        else if unprovability tactics applies on $\Delta \vdash \psi$ then OK by absurdity;
        else
            prove new lemma $l : \forall h\!:\!\Delta \vdash \psi, f\ h =$ trueP using $M(\Delta \vdash \psi)$;
            store $l$ in $DB$; apply $l$;
        end
    end
end
```

Algorithm 1: Proof method for properties of the form: $\forall h\!:\!\Gamma \vdash \phi, f\ h =$ `trueP`.

These actions, together with initial resources, can be used to model the following narrative specification: the outcome of the discussion with Binet will be determined by the proof-search, while an external event (in an open-world assumption) which decides between the two possible outcomes of the discussion with Guillaumin.

Two possible ending states are specified for this narrative: Emma is ready to sell herself to improve her situation (S) or prepared to have a discussion with Rodolphe. We want to prove that whatever the narrative generated by this specification, there is always a possible sub-narrative in the open-world assumption which allows for the ending state S to be reached. The corresponding sequent modelled in Coq is:

$s =$`{G,((B⊸S)&(B⊸R))&1,(G⊸B)⊕(G⊸S)⊢S⊕R}`

We have proved that this sequent is such that $\forall$ `h:(s), check s = trueP`. This proof uses 47 auxiliary lemmas, while the sequent only offers a low-level of generativity. Each lemma is proved automatically but currently the lemmas are stated by hand. We discuss briefly how we plan to automate the lemmas generation in the conclusion of this paper.

In order to show that provided adequate automation our technique can scale on sequents offering a high level of generativity, we proved a similar reachability property on the following sequent which uses narrative actions described on figure 3:

$s_1 =$`{P\&1,(S⊸A)\&1,(E⊸A)\&1,(P⊸M)\&1,S}) ⊢ (A⊕M)`

As the sequent is more generative, this proof uses 283 auxiliary lemmas.

5 Conclusion

In this paper, we have shown that the Coq proof assistant is a powerful tool for studying and verifying structural properties of narratives modelled using Intuitionistic Linear Logic.

We have provided a method for encoding narratives specifications into an ILL sequent, encompassing narrative actions and initial resources description, and described an encoding of ILL into Coq which allows to built well-formed narratives from proofs of such a sequent.

The encoding we have proposed makes use of the proof-as-term paradigm and allows us to verify structural properties of narratives transcending all narratives generated by a specification. This allows to study resource-sensitive and causality relationships emerging from the initial specification. From a low-level description of the semantic of narrative actions, we are thus able to obtain high-level semantic properties regarding the narrative.

Now that we have shown that our encoding and our proof method allows for automated heuristics, we plan to implement certifying external procedures in a similar fashion than previous work of authors [5, 6]. More precisely we plan to 1) implement a Coq script generator that will generate the lemmas statements and proof following ideas of section 4.4 and 2) prove more unprovability results in order to tame a bit more the combinatorial explosion of ILL proofs. The need to prove properties on proofs themselves forces the use of dependently typed programming style, which happens to be uncommon, especially on sort Prop on which elimination is limited. The experience was however successful.

This work therefore opens new perspectives on the design and understanding of computational models of narratives. A particularly interesting avenue to explore concerns the search for normalised forms of narratives, for instance offering the highest possible degree of sub-narratives parallelism relying on resource independence. Such normalisation procedures can rely on dedicated proof-search algorithms, complementing our existing encoding. This work is also a first step towards the assessment of story variance on a structural and formal basis: based on the definition of equivalence relationships between proofs, and further, between their narrative interpretations, we plan to investigate formally what makes story differ and propose metrics which would allow to evaluate narrative specifications.

References

1. Anne-Gwenn Bosser, Marc Cavazza, and Ronan Champagnat. Linear logic for non-linear storytelling. In *Proceedings of ECAI 2010 - 19th European Conference on Artificial Intelligence*, 2010.
2. Claude Brémond. *Logique du Récit.* Seuil, 1973.
3. Marc Cavazza and David Pizzi. Narratology for interactive storytelling: A critical introduction. In *Proceedings of the Third International Conference on the Technologies for Interactive Digital Storytelling and Entertainment (TIDSE)*, Lecture Notes in Computer Science. Springer, 2006.
4. Frédéric Collé, Ronan Champagnat, and Armelle Prigent. Scenario analysis based on linear logic. In *Advances in Computer Entertainment Technology (ACE)*, 2005.
5. Évelyne Contejean, Pierre Courtieu, Julien Forest, Andrei Paskevich, Olivier Pons, and Xavier Urbain. A3PAT, an Approach for Certified Automated Termination Proofs. In *ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation (PEPM 10)*, pages 63–72. ACM, 2010.
6. Évelyne Contejean, Pierre Courtieu, Julien Forest, Olivier Pons, and Xavier Urbain. The CiME Rewriting Toolbox, Version 3.

7. Mary Dalrymple, John Lamping, and Fernando Pereira. Linear logic for meaning assembly. In *Proceedings of Computational Logic for Natural Language Processing*, 1995.
8. Lucas Dixon, Alan Smaill, and Alan Bundy. Verified planning by deductive synthesis in intuitionistic linear logic. In *ICAPS Workshop on Verification and Validation of Planning and Scheduling Systems*, 2009.
9. Lucas Dixon, Alan Smaill, and Tracy Tsang. Plans, actions and dialogues using linear logic. *Journal of Logic, Language and Information*, 18(2):251–289, 2009.
10. Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–102, 1987.
11. Jean-Yves Girard and Yves Lafont. Linear logic and lazy computation. In *Proceedings of the International Joint Conference on Theory and Practice of Software Development (TAPSOFT '87)*. Springer-Verlag, 1987.
12. Dieter Grasbon and Norbert Braun. A morphological approach to interactive storytelling. In *Proceedings of the Conference on Artistic, Cultural and Scientific Aspects of Experimental Media Spaces (cast01)*, 2001.
13. Algirdas Julien Greimas. *Sémantique structurale: recherche et méthode*. Larousse, 1966.
14. Vineet Gupta and John Lamping. Efficient linear logic meaning assembly. In *Proceedings of the 17th international conference on Computational linguistics*. Association for Computational Linguistics, 1998.
15. Antonios Kakas and Rob Miller. A simple declarative language for describing narratives with actions. *The journal of Logic Programming*, 31:157–200, 1997.
16. Sara Kalvala and Valeria De Paiva. Mechanizing linear logic in isabelle. In *In 10th International Congress of Logic, Philosophy and Methodology of Science*, 1995.
17. R. Raymond Lang. A declarative model for simple narratives. In *AAAI*, 1999.
18. Patrick Lincoln. Deciding provability of linear logic formulas. In *Advances in Linear Logic*, pages 109–122. Cambridge University Press, 1994.
19. M. Masseron. Generating plans in linear logic: I i. a geometry of conjunctive actions. *Theoretical Computer Science*, 113(2):371–375, 1993.
20. M. Masseron, Christophe Tollu, and Jacqueline Vauzeilles. Generating plans in linear logic: I. actions as proofs. *Theoretical Computer Science*, 113(2):349–370, 1993.
21. Rob Miller and Murray Shanahan. Narratives in the situation calculus. *Journal of Logic and Computation*, 4:513–530, 1994.
22. James Power and Caroline Webster. Working with coq in linear logic. In *Proceedings of the 12th International Conference on Theorem Proving in Higher Order Logics (TPHOL)*, 1999.
23. Vladimir Propp. *Morphology of the Folktale*. University of Texas Press, 1968.
24. Ray Reiter. Narratives as programs. In *Proc. of the Seventh International Conference on Knowledge Representation and Reasoning (KR2000)*, 2000.
25. Mehrnoosh Sadrzadeh. Modal linear logic in higher order logic: An experiment with coq. In *In Emerging Trends TPHOLS '03*, pages 75–93, 2003.
26. Michael Schroeder. How to tell a logical story. In *AAAI*, 1999.
27. R. Michael Young. Notes on the use of plan structures in the creation of interactive plot. In *Narrative Intelligence: Papers from the AAAI Fall Symposium.* AAAI Press, 1999.