An ASP Approach
to Generate Minimal Countermodels
in Intuitionistic Propositional Logic

Camillo Fiorentini

DI, Univ. degli Studi di Milano, Milano, Italy

IJCAI 2019
Macao, August 14th, 2019
Intuitionistic Propositional Logic

Intuitionistic Propositional Logic (IPL) is a constructive non-classical logic.

- **Non-classical**: some classical tautologies are not valid in IPL
  \[ A \lor \neg A \quad (A \to B) \lor (B \to A) \quad \neg A \lor (A \to B) \]

- **Constructive**: IPL enjoys the Disjunction Property:
  \[ A \lor B \in IPL \implies A \in IPL \text{ or } B \in IPL \]

IPL is closely related to Propositional Classical Logic (CPL):

- IPL ⊂ CPL
- IPL can be embedded in CPL:
  \[ A \in IPL \implies \neg \neg A \in IPL \]

Thus, the following principles are valid in IPL:

\[ \neg \neg (A \lor \neg A) \quad \neg \neg ((A \to B) \lor (B \to A)) \quad \neg \neg (\neg A \lor (A \to B)) \]
Semantics

- **CPL**
  
  An interpretation $\mathcal{I}$ is a set of propositional variables.

  The validity of a formula w.r.t. $\mathcal{I}$ is defined according to the classical meaning of logical connectives (truth tables):
  - $\mathcal{I} \models p$ iff $p \in \mathcal{I}$, for $p$ a propositional variable
  - $\mathcal{I} \models A \land B$ iff $\mathcal{I} \models A$ and $\mathcal{I} \models B$
  - ...  

  CPL is the set of formulas valid in all the interpretations.

- **IPL**
  
  To get a sound semantics for IPL, we need a more refined semantics.
  A model is a set of classical interpretations, called worlds
  
  ✓ Each world represents a knowledge state
  ✓ Worlds are ordered by a partial order relation $\leq$
  ✓ Validity is represented by forcing relation $\models$ between worlds and formulas
  ✓ Forcing is preserved by $\leq$:

  $$w_1 \models A \land w_1 \leq w_2 \implies w_2 \models A$$

  This leads to Kripke frame semantics.
A *Kripke model* is a structure $\mathcal{K} = \langle P, \leq, V \rangle$, where:

- $P$ is a **finite** nonempty set of worlds
- $\leq$ is a partial order between worlds
- $V$ assigns to each world a classical interpretation, obeying truth preservation:

$$w_1 \leq w_2 \implies I(w_1) \subseteq I(w_2)$$

- The forcing relation $\models$ between worlds and formulas is inductively defined as follows:

  - $w \not\models \bot$
  - $w \models p$ iff $V(w) \models p$ ($V(w)$ is the interpretation related to $w$)
  - $w \models A \land B$ iff $w \models A$ and $w \models B$
  - $w \models A \lor B$ iff $\alpha \models A$ or $w \models B$
  - $w \models \neg A$ iff, for every $w' \geq w$, $w' \not\models A$
  - $w \models A \to B$ iff, for every $w' \geq w$, $w' \not\models A$ or $w' \models B$

For formulas $\neg A$ and $A \to B$, forcing at $w$ depends on the successors of $w$
IPL is complete with respect to Kripke semantics, namely:

- \( A \in \text{IPL} \) iff \( A \) is forced in every world of every Kripke model

Accordingly, if \( A \not\in \text{IPL} \), there exists a model \( K \) and a world \( w \) in \( K \) such that \( A \) is not forced at \( w \).

We call \( K \) a countermodel for \( A \)
Example

A countermodel for $p \lor \neg p$ is

$$\begin{align*}
  w_2 & : p \\
  w_1 & : \\
  V(w_1) & = \emptyset \quad v(w_2) = \{p\} \\
  w_1 & \not\vDash p \quad \text{since } p \notin V(w_1) \\
  w_1 & \not\vDash \neg p \quad \text{since } w_1 \leq w_2 \text{ and } w_2 \vDash p \ (p \in V(w_2)) \\
  w_1 & \not\vDash p \lor \neg p \quad \text{since } w_1 \not\vDash p \text{ and } w_1 \not\vDash \neg p
\end{align*}$$

At $w_1$, $p$ is not forced.

The world $w_1$ is followed by a world $w_2$ and $p$ is forced at $w_1$, thus $\neg p$ is not forced at $w_2$. Since forcing must be preserved through $\leq$, $\neg p$ is not forced at $w_1$.

We conclude that $p \lor \neg p$ is not forced at $w_1$. 
Let $G$ be a goal formula

- The **validity** of $G$ in IPL can be witnessed by a **derivation** in a sound calculus for IPL
  - *Hilbert calculus, natural deduction, tableaux/sequent, ...*

- The **non-validity** of $G$ can be witnessed by a **countermodel**

Typically, the emphasis is on derivations and countermodels are obtained as a result of a failed proof-search for a derivation of $G$.

For almost all the known tableaux/sequent calculi for IPL, we can define a proof-search procedure $\text{ProofSearch}$ such that:

$$
\text{ProofSearch}(G) = \begin{cases} 
\text{A derivation of } G & \text{if } G \in \text{IPL} \\
\text{A countermodel for } G & \text{otherwise}
\end{cases}
$$
A countermodel can be understood as a certificate witnessing the non-validity of the goal formula $G$.

Countermodels can be used for diagnosis, to analyze why some property fails or to fix errors in formal specifications (see Property-Based Testing).

It is critical that countermodels are minimal so as to convey a plain and concise representation of non-validity.

This issue has been scarcely investigated in the literature. Many proof-search procedures have been introduced, but all fail to build small countermodels.

- G. Corsi and G. Tassi. Intuitionistic logic freed of all metarules. JSL, 2007
- M. Ferrari, C. Fiorentini, and G. Fiorino. Contraction-free linear depth sequent calculi for intuitionistic propositional logic with the subformula property and minimal depth counter-models. JAR, 2013.
- ...
Countermodels

Example

\[ G = (p_1 \rightarrow p_2) \lor (p_2 \rightarrow p_1) \lor (q_1 \rightarrow q_2) \lor (q_2 \rightarrow q_1) \]

Countermodel generated by ProofSearch(G) [Ferrari et al., TOCL, 2015] generating countermodels of minimal depth

The model has minimal height, but it is not minimal in the number of worlds. A minimum countermodel is:

Note that we cannot shrink the first model to get a minimum one!
Countermodel generation

Main contribution

We present a procedure to generate minimal countermodels:

- given a goal formula $G$, we try to build a countermodel for $G$ by a model-search procedure guided by semantics.

A naive implementation of the process immediately blows-up; even for small goal formulas, model generation is not terminating.

We need a clever formalization of the problem.
Countermodel generation

- Model formalization
  
  We follow the approach of R. Goré et al. [IJCAR 2012, 2014]:
  
  - Worlds of models are represented by sets $\mathcal{W}$ of atomic subformulas $H$ of $G$, namely:
    
    $\begin{align*}
    H & ::= p \mid \neg A \mid A \rightarrow B 
    \end{align*}$
    
    $p$: propositional variable
  
  - We do not consider all possible sets $\mathcal{W}$ of atomic subformulas, but only the sets $\mathcal{W}$ satisfying some closure properties, we call $p$-worlds (possible worlds)
  
  For instance:
  
  $\begin{align*}
  \mathcal{W}_1 &= \{ p, \neg p \} & \mathcal{W}_2 &= \{ p, p \rightarrow q \} 
  \end{align*}$
  
  $\mathcal{W}_1$ must be discarded since it is inconsistent
  $\mathcal{W}_1$ must be discarded since it is not closed under modus ponens ($q \not\in \mathcal{W}_2$)
Countermodel generation

- The first selected p-world \( \mathcal{W}_0 \) is a putative world falsifying \( G \).
- To get a well-defined Kripke model, we have to guarantee that atomic subformulas of \( G \) not belonging to \( \mathcal{W}_0 \) are not valid in \( \mathcal{W}_0 \), for instance:

\[
A \rightarrow B \notin \mathcal{W}_0 \implies \exists \mathcal{W}_1 ( \mathcal{W}_0 \subseteq \mathcal{W}_1 \land A \in \mathcal{W}_1 \land B \notin \mathcal{W}_1 )
\]

\( \mathcal{W}_1 \) is needed to witness the non-validity of \( A \rightarrow B \) in \( \mathcal{W}_0 \).

This triggers a saturation process which successfully ends when all the needed witnesses have been generated, thus yielding a countermodel for \( G \).

\( \mathcal{W}_0 \not\models G \)

\( \mathcal{W}_1 \not\models B \)

\( \mathcal{W}_1 \models A \)
Countermodel generation

- Computation engine
  - We formalize the search problem in Answer Set Programming (ASP) [Baral 2010].
    - √ ASP is a form of declarative programming based on the stable model semantics (answer sets),
    - √ ASP enables to solve hard search problems (in $NP$ and in $NP^{NP}$) in a uniform way
  - We define an ASP program $\Pi_G$ such that an answer set of $\Pi_G$ corresponds to a countermodel for $G$.
    - If no answer exists, there is no countermodel for $G$, meaning that $G$ is valid (in IPL).
  - To compute answer sets, we exploit the Potassco tool clingo [Gebser et al., 2012].
  - The minimization of models is delegated to clingo; however, it is critical to encode the problem so that even the first computed model is small, otherwise the minimization engine gets stuck.
Countermodel generation

Differently from other declarative formalisms, ASP allows for a quite modular formalization:

\[ \Pi_G = \text{Gen} + \text{Goal}(G) \]

- \text{Gen} encodes the generator and is independent of the goal formula
- \text{Goal}(G) encodes the goal formula

The generator can be easily extended to deal with other intermediate logics where the frame conditions can be expressed in ASP, such as:

- The Gödel-Dummett logic [Dummett, 59], characterized by linear frames
- The logic of bound-depth frames
- Here and There logic [Pearce, 97], well-known in ASP
Countermodel generation

Frame conditions can be freely composed:

- **lin.lp** encodes the constraint “the model is linear”
  
  ```
  :- world(W1), world(W2), W1 <> W2 , not le(W1,W2), not le(W2,W1).
  ```

- **bd2.lp** encodes the constraint “the model has depth at most 2”
  
  ```
  :- world(W1), world(W2), world(W3), W1 <> W2, W1 <> W3, W2 <> W3, 
  le(W1,W2), le(W2,W3).
  ```

```c
clingo gen.lp goal.lp lin.lp  // linear countermodels
clingo gen.lp goal.lp bd2.lp  // depth <=2 countermodels
clingo gen.lp goal.lp lin.lp  bd2.lp
  // linear AND depth <=2 countermodels
```

This kind of modularity is not possible with derivations!
Countermodel generation

The program is efficiente with formulas containing few propositional variables.
For instance, let us consider the non-valid Nishimura formulas:

\[ N_1 = p \quad N_2 = \neg p \]
\[ N_{2n+3} = N_{2n+1} \lor N_{2n+2} \quad N_{2n+4} = N_{2n+3} \rightarrow N_{2n+1} \]

The countermodel for \( N_{17} \) is computed in few seconds: