Phometa — a visualised proof assistant that builds a formal system and proves its theorems using derivation trees

Gun Pinyo

Imperial College London

June 21, 2016

Derivation systems can reason many formal systems.

$$\text{IMPLY-ELIM}\ \frac{\Gamma \vdash (A \to B) \qquad \Gamma \vdash A}{\Gamma \vdash B}$$

$$\dfrac{\dfrac{}{p, p \to q \vdash p}\ \text{ASS} \qquad \dfrac{\dfrac{}{p, p \to q \vdash p \to q}\ \text{ASS} \qquad \dfrac{}{p, p \to q \vdash p}\ \text{ASS}}{p, p \to q \vdash q}\ \text{IMPLY-ELIM}}{p, p \to q \vdash p \wedge q}\ \text{AND-INTRO}$$

Derivation systems can reason many formal systems.

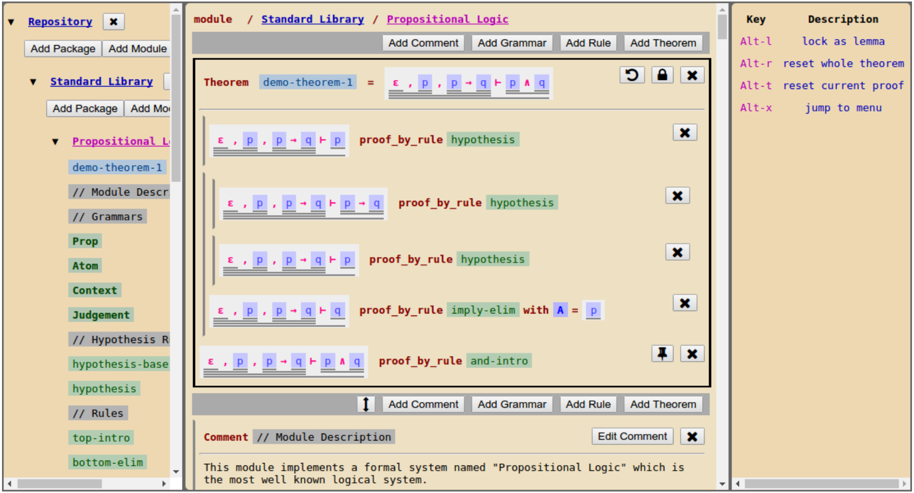
$$
\dfrac{\dfrac{\dfrac{}{(w \times (x + y)) = ((w \times x) + (w \times y))}\ \textsc{Dist-Left}}{((w \times x) + (w \times y)) = (w \times (x + y))}\ \textsc{Eq-Symm} \qquad \dfrac{}{z = z}\ \textsc{Eq-Refl}}{(((w \times x) + (w \times y)) \times z) = ((w \times (x + y)) \times z)}\ \textsc{Mult-Intro}
$$

$$
\dfrac{\dfrac{\dfrac{}{y : A, x : A \vdash x : A}\ \textsc{Assumption}}{y : A \vdash \lambda x.x : A \to A}\ \textsc{Arrow-Intro} \qquad \dfrac{}{y : A \vdash y : A}\ \textsc{Assumption}}{y : A \vdash (\lambda x.x)y : A}\ \textsc{Arrow-Elim}
$$

Problem of drawing a derivation tree.

- Its width grows exponentially to its height.
- One drawing mistake might need major correction of the entire tree.
- (Meta) is being rewritten by unification.
- Cannot systematically check that the tree has no errors.

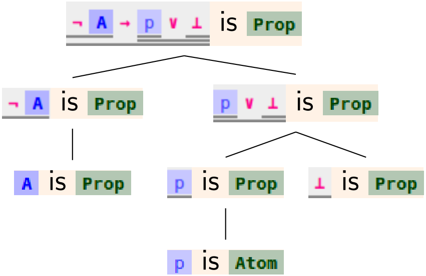
Screenshot of "Phometa" in action

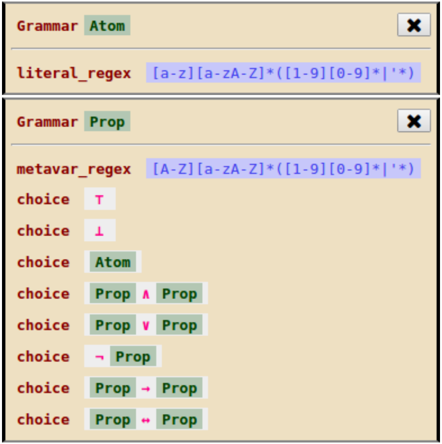


Formal system ingredient - Grammar (Backus-Nour Form)

```
<Prop> ::= '⊤' | '⊥' | <Atom>
    | '('<Prop> '∧' <Prop>')'
    | '('<Prop> '∨' <Prop>')'
    | '( ¬' <Prop>')'
    | '('<Prop> '→' <Prop>')'
    | '('<Prop> '↔' <Prop>')'
    | meta-variable with regex
/[A-Z][a-zA-Z]*([1-9][0-9]*|'*)/
```

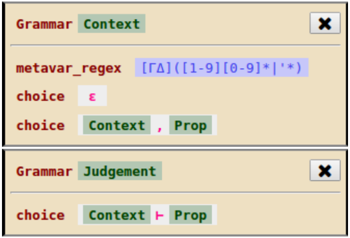
```
<Atom> ::= literal with regex
/[a-z][a-zA-Z]*([1-9][0-9]*|'*)/
```

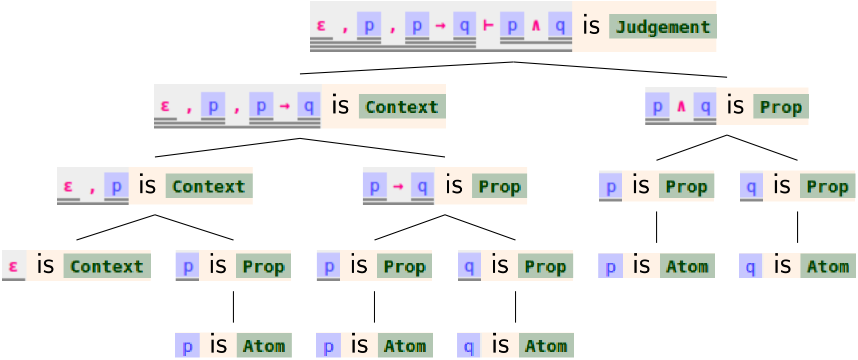




Grammar (Backus-Nour Form)

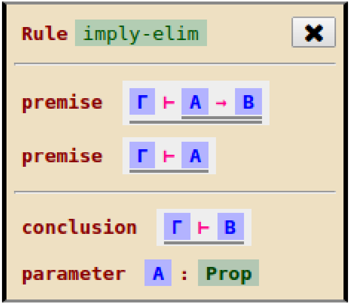
```
<Context> ::= 'ϵ'
  | <Context> ',' <Prop>
  | meta-variables comply with regex
     /[ΓΔ]([1-9][0-9]*|'*)/

<Judgement> ::= <Context> '⊢' <Prop>
```

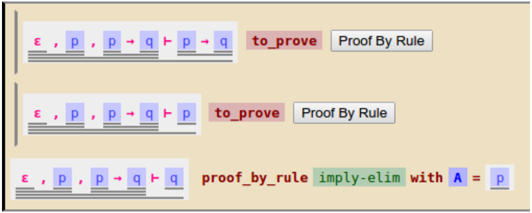




Formal system ingredient - Rule (Derivation Rule)

$$\text{IMPLY-ELIM}\ \frac{\Gamma \vdash (A \rightarrow B) \qquad \Gamma \vdash A}{\Gamma \vdash B}$$





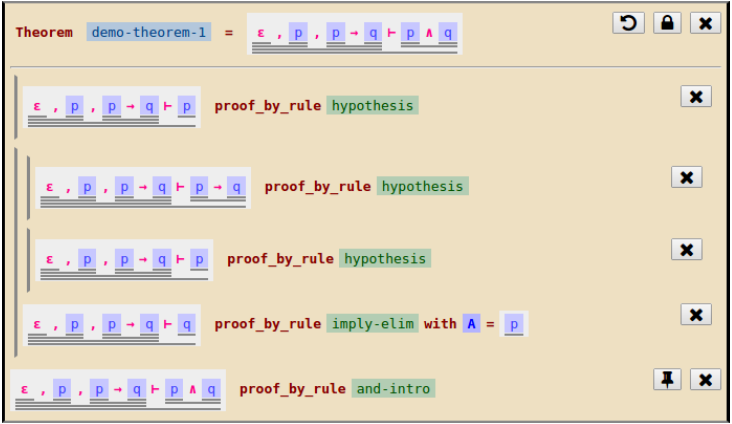
Pattern Matching Γ = ε , p , p → q

B = q

Parameter(s) A = p

Formal system ingredient - Theorem (Derivation Tree)

$$\dfrac{\dfrac{}{p, p \to q \vdash p}\,\text{ASS} \qquad \dfrac{\dfrac{}{p, p \to q \vdash p \to q}\,\text{ASS} \qquad \dfrac{}{p, p \to q \vdash p}\,\text{ASS}}{p, p \to q \vdash q}\,\text{IMPLY-ELIM}}{p, p \to q \vdash p \wedge q}\,\text{AND-INTRO}$$



Demonstration

- Overview of Phometa's user interface.
- Prove that $\varepsilon, p, p \to q \vdash p \wedge q$ is a valid `Judgement`.
- Extend Propositional Logic to support equivalence.

```
<Equivalence> ::= <Prop> '≡' <Prop>
```

$$\textsc{Equiv-Intro}\ \frac{A \vdash B \qquad B \vdash A}{A \equiv B}$$

- Prove that $\neg P \vee Q \equiv \neg P \wedge \neg Q$ is a valid `Equivalence`.

Cascade Premise — Hypothesis Rule

```
Rule hypothesis
─────────────────────────────────────
cascade
│ hypothesis-base  exact_match  Γ , B ⊢ A
│ hypothesis  Γ ⊢ A
─────────────────────────────────────
conclusion  Γ , B ⊢ A
```

```
Rule hypothesis-base
─────────────────────────────────────
conclusion  Γ , A ⊢ A
```

Meta-Reduction — Context Manipulation

Rule context-commutative

premise Γ , B , A

conclusion Γ , A , B

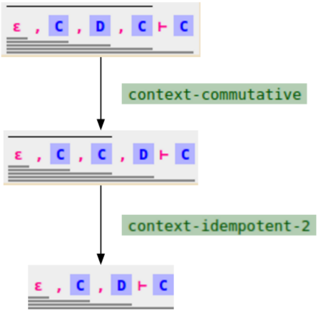
allow_reduction

Rule context-idempotent-2

premise Γ , A

conclusion Γ , A , A

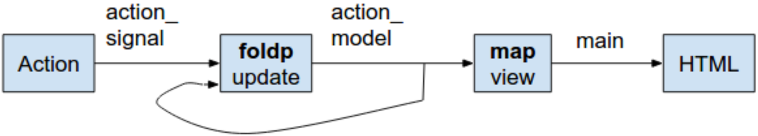
allow_reduction



Implementation — Elm language and its Signals


```
Mouse.isDown : Signal Bool

Signal.map  : (a -> b) -> Signal a -> Signal b

view : Bool -> Html
view is_clicked =
  if is_clicked then text "bar"
                else text "foo"

main : Signal Html
main = Signal.map view Mouse.isDown
```

Implementation — Model-Controller-View (MCV)




```
Signal.foldp : (a -> b -> b) -> b -> Signal a -> Signal b

update : Action -> Model -> Model
view   : Model -> Html

action_signal : Signal Action
action_signal = Signal.merge mailbox.signal keyboard_signal

model_signal : Signal Model
model_signal = Signal.foldp update init_model action_signal

main : Signal Html
main = Signal.map view model_signal
```

Remark: this is the *simplified* version of Phometa main entry.

Strengths, Limitation, and Future Work

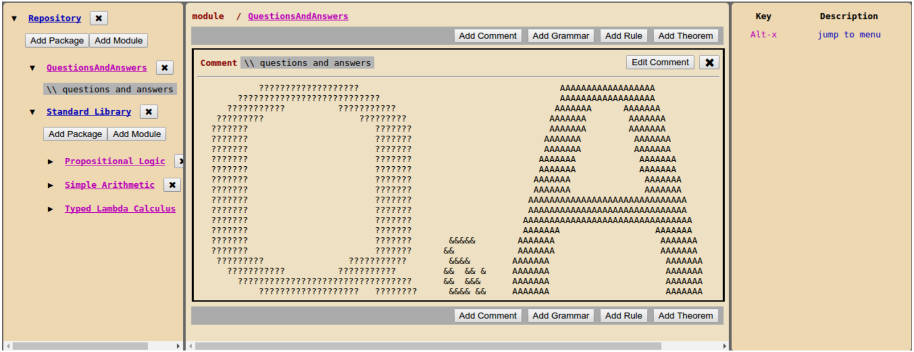
- Has extra features over the traditional derivation system such as prove by lemma, cascade premise, and meta-reduction.
- Less learning curve than mainstream proof assistants.
- The program always in consistent state
  i.e. impossible to create an invalid proof.

- Phometa need to load the entire proof repository when start.
- Theorem building process should be more automatic.
- Nodes should be able to be exported into LaTeX source code.
- Modules should be able to import nodes from other modules.

Achievement & Conclusion

- Finished designing Phometa specification in such a way to keep it simple yet be able to produce a complex proof.
- Finished implementing Phometa, as you have seen in the demo.
- Encoded these formal systems in the standard library.
  - Propositional Logic
  - Simple Arithmetic
  - Typed Lambda Calculus
- Wrote a tutorial for newcomers to use Phometa (Chapters 3, 4, A, B, C in the report).
- Phometa is ready to be used as a replacement of the traditional derivation system.

Questions & Answers



Thank You