Lean 言語による書籍執筆支援ツール

井上亜星 (Proxima Technology)

要旨

- Lean で書籍を作成するためのツールとして、mdgen [6] と mk-exercise [7] というソフトウェアを開発した。
- mdgen は Lean ファイルから markdown ファイルを生成するシンプルなツールであり、ソースコードとテキストを別々に管理する必要がなくなり手間が減らせるほか、記述の正しさをコードとして検証できるのも利点である。
- mdgen はコメントを地の文に、コメント以外の部分をコードブロックに変換するだけのシンプルなツールであるため、専用の構文やライブラリがあまり必要ない。Web 上でテキストを公開する際に「Web Playground にジャンプして実行する」という機能をファイル単位で実装できる。
- mdgen で生成された markdown ファイルからテキストを生成するための mdbook [2] テンプレートも開発した。
- 演習問題の問題と解答を同期させるためのツールとして、mk-exercise も開発した。
- 他の Lean によるドキュメンテーションツールと比較すると、シンプルさと、シンプルさゆえに実現できた「Web Playground [1] へジャンプして実行する機能」が長所である。

導入

Lean 言語とは

Lean 言語は、2013 年ごろから現在まで活発に開発が続けられている定理証明支援系である。数学理論の形式化や、自動証明の研究に用いられている。

Lean の特徴: プログラミング言語でもある

Rocq(旧 Coq) と同様に依存型に依拠しているが、Lean は関数型プログラミング言語でもある。特に、停止しない再帰関数も定義できる。これには以下のような利点がある。

- アルゴリズムの正しさの証明を実装とシームレスに行うことができる。
- 数学の計算と証明がシームレスに行える。

Lean の特徴: 対話的実行機能

Lean は対話的実行機能がすぐれており、ターミナルに移動したり jupyter notebook のような特別な環境を用意したりしなくても、エディタ上で即座にコードのフィードバックが得られる。

```
/-- リストの順番を逆にする関数 -/
def reverse : List α → List α
  | [] ⇒ []
  | x :: xs ⇒ reverse xs ++ [x]

/- [5, 4, 3, 2, 1] -/
#eval reverse [1, 2, 3, 4, 5]
```

Figure 1. #eval によって値をその場で計算できる

Lean の特徴: 拡張性の高さ

Lean は強力なメタプログラミングフレームワークを備えており、パーサ、エラボレータ、タクティク、プリティプリンタ、コードジェネレータをユーザが（Lean 自体のソースコードを一切変更することなく）変更できる。

```
/-- `Expr` のための構文カテゴリ -/
declare_syntax_cat expr

/-- `Expr` を見やすく定義するための構文 -/
syntax "expr!{" expr "}" : term

-- 数値リテラルは数式
syntax:max num : expr

-- 数式を `+` または `*` で結合したものは数式
-- `+` と `*` のパース優先順位を指定しておく
syntax:30 expr:30 " + " expr:31 : expr
syntax:35 expr:35 " * " expr:36 : expr

-- 数式を括弧でくくったものは数式
syntax:max "(" expr ")" : expr

macro_rules
  | `(expr!{$n:num}) ⇒ `(Expr.val $n)
  | `(expr!{$l:expr + $r:expr}) ⇒ `(Expr.app Op.add expr!{$l} expr!{$r})
  | `(expr!{$l:expr * $r:expr}) ⇒ `(Expr.app Op.mul expr!{$l} expr!{$r})
  | `(expr!{($e:expr)}) ⇒ `(expr!{$e})

#guard
  let expected := app Op.add (app Op.add (val 1) (val 2)) (val 3)
  let actual := expr!{1 + 2 + 3}
  actual = expected
```

Figure 2. macro_rules コマンドでマクロとして構文を用意する

Lean の応用

上記の特徴を考えると、Lean は教科書を書くのに向いているのではないか。

- アルゴリズムの記述と実行、性質の証明が全部 Lean で完結して便利
- 学習者が Lean で実際に実行できるので対話性が高い
- Lean は構文が読みやすくて簡潔

方針

Lean で教科書を書くためのツールを作成したい。

設計方針

Lean ファイルから markdown を生成する

markdown から本を作るツールとして、既に mdbook [2] などが存在する。したがって Lean ファイルから markdown ファイルを生成できればよい。Lean のコメント部分を markdown の地の文に、コード部分を markdown のコードブロックに変換する方針を採用した。

メッセージの内容も検証

Lean は活発に開発が続いているのでバージョン変更で記述内容が古くなることが懸念されるが、mdgen で書いている本は記述内容をコードで検証できる。たとえば、#guard_msgs コマンドによりコマンドからの出力メッセージを検証できる。

```
/-- `s : String` をパースして `Syntax` の項を得る。`cat`
 は構文カテゴリ。-/
def parse (cat : Name) (s : String) : MetaM Syntax := do
  ofExcept ◁ runParserCategory (← getEnv) cat s

-- 最初は `#greet` などというコマンドは定義されていないので
-- そもそも Lean の合法な構文として認められない。
/-- error: <input>:1:0: expected command -/
#guard_msgs (error) in
  #eval parse `command "#greet"
```

Figure 3. コマンドで出力メッセージを検証できる

mdbook と組み合わせて、Web Playground での実行も可能に

markdown から本を作るツールとしては mdbook を主に想定し、Lean のシンタックスハイライトなどの機能を実装したテンプレート lean-book [4] を作成した。

GitHub 上で本を公開している場合、Lean のソースファイルが GitHub 上で入手できる。そのコードをそのまま Lean の Web Playground [1] に貼り付けることで、「このファイルのコードを実行する」というボタンも実装できた。

mk-exercise で演習問題と解答を同期する

教科書を作る際には演習問題を用意したいことがあるが、演習問題の解答と問題を同期できず、片方だけを更新してしまうという問題が発生しがちである。この問題を解決するため、mk-exercise [7] というツールを作成した。これは、指定した範囲を sorry タクティクで置き換える、シンプルなツールである。

他のツールとの比較

lean2md との比較

既に「Lean ファイルのコメント部分を markdown では地の文に変換し、コードを markdown のコードブロックに変換する」という同一のコンセプトを持つツールとして、lean2md [9] という Python スクリプトが開発されていた。mdgen は lean2md と比較すると以下のような長所を持つ。

- パーサの実装がより適切で、ネストされたブロックコメントも扱える。
- 指定したディレクトリのサブディレクトリに対しても再帰的に実行する。
- 特定の記号で囲った部分を出力から隠すことができる。
- ドキュメントコメントをブロックコメントに変換できるので、#guard_msgs コマンドによる出力メッセージの検証を隠すことができる。
- 純粋な Lean で実装されているので、プロジェクトに依存関係を増やさずに済む。

Verso との比較

Lean によるドキュメント作成ツールとしては、公式が作成している Verso [3] も存在する。Verso と比較すると、mdgen には以下のような利点があると考えている。

- mdgen には特別な構文がなくシンプルなので、覚えることが少なく直観的に使える。
- mdgen の構文はシンプルなので Web Playground へのジャンプ機能を簡単に実装できる。
- markdown への生成だけを行うので、どんなツールで本にするかはユーザが選択できる。

利用例

mdgen または mk-exercise が使用されている例として、以下の書籍が挙げられる。

- Lean by Example [5] 著者が執筆している日本語リファレンス。
- Metaprogramming in Lean 4 [10] Lean でのメタプログラミングのやり方を説明する書籍。
- 数学系のための Lean 勉強会 [8] Lean で数学をどのように実装するのかを解説する勉強会の資料。

参考

[1] Lean 4 web. https://live.lean-lang.org/.

[2] mdbook. https://github.com/rust-lang/mdBook, version 0.4.35.

[3] The Lean Developers. verso. https://github.com/leanprover/verso.

[4] Asei Inoue. lean-book. https://github.com/Seasawher/lean-book.

[5] Asei Inoue. Lean by Example. https://lean-ja.github.io/lean-by-example/.

[6] Asei Inoue. mdgen. https://github.com/Seasawher/mdgen.

[7] Asei Inoue. mk-exercise. https://github.com/Seasawher/mk-exercise.

[8] Yuma Mizuno and Haruhisa Enomoto. 数学系のための lean 勉強会. https://github.com/yuma-mizuno/lean-math-workshop.

[9] Arthur Paulino. lean2md. https://github.com/arthurpaulino/lean2md.

[10] Arthur Paulino, Damiano Testa, Edward Ayers, and et al. Metaprogramming in lean 4. https://leanprover-community.github.io/lean4-metaprogramming-book/.