Type Classes in Coq

이계식

한경대학교

12회 ROSAEC 워크숍
2015년 1월 27일

동기

- 수학의 정형화
- 기존 Coq 라이브러리 탐색 및 연구
- 기존 연구에 응용 가능: GMeta 등등.

하스켈에서의 타입 클래스

- 오버로딩 지원
  - 자연수에서의 덧셈, 정수에서의 덧셈 등을 하나의 기호($*$)로 나타낼 수 있다.
- 또는 특정 타입에 대해서만 한정된 polymorphism(다형성)

하스켈에서의 타입 클래스

- 예를 들어, 동일성을 담는 클래스 Eq를 아래와 같이 선언할 수 있다.

```
class Eq a where
  (==) :: a -> a -> Bool
```

- 이제는 Eq a를 만족하는 타입 a에 대해서만 등일성(==) 관련 성질 묘사 가능.

```
=/= :: Eq a => a -> a -> Bool
  x =/= y = not (x == y)
```

- Bool 상에 정의된 등일성 선언:

```
instance Eq Bool where
  x == y = if x then y else (not y)
```

First-Class Type Classes in Coq

- 상대적으로 최근에 구현됨.
- Matthieu Sozeau and Nicolas Oury, TPHOLs 2008.
- 하스켈에서와는 달리 콕 시스템이 지원하는 언어/기술적 능력만을 활용함.
  - record type, dependent type theory, type inference, unification 등등

콕에서의 타입 클래스

- 콕에서:

```
Class EqDec (A : Type) := {
  eqb : A -> A -> bool ;
  eqb_leibniz : forall x y,
                  eqb x y = true -> x = y }.


Instance eq_bool : EqDec bool :=
 { eqb x y := if x then y else negb y }.

 Proof. .... Qed.
```

콕에서의 타입 클래스

- First-class라서 가능한 표현 1 (한정 polymorphism):

```
Definition negb {A} {eqa : EqDec A} (x y : A) :=
  negb (eqb x y).
```

콕에서의 타입 클래스

- First-class라서 가능한 표현 2 (매개변수 인스턴스):

```
Instance prod_eqb `(EA : EqDec A, EB : EqDec B) :
 EqDec (A * B) :=
 { eqb x y := match x, y with
   | (la, ra), (lb, rb) =>
       andb (eqb la lb) (eqb ra rb)
   end }.
```

상, 하위 클래스 예제 1

- 클래스의 계층을 만들 수 있다.

```
Class Ord `(E : EqDec A) :=
  { le : A -> A -> bool }.
```

- 아래 하스켈 정의에 대응:

```
Class `(E : EqDec A) => Ord A :=
  { le : A -> A -> bool }.
```

상, 하위 클래스 예제 2

- 수학에서 상, 하위 클래스 개념이 많이 사용된다.
- 예를 들어 preorder를 정의하려면 아래 성질들이 필요하다.

```
Class Reflexive (A : Type) (R : relation A) :=
  reflexivity : forall x, R x x.

Class Transitive (A : Type) (R : relation A) :=
  transitivity :
    forall x y z, R x y -> R y z -> R x z.
```

- Preorder는 두 클래스의 하위 클래스이다.

```
Class PreOrder (A : Type) (R : relation A) :=
 { PreOrder_Reflexive :> Reflexive A R ;
   PreOrder_Transitive :> Transitive A R }.
```

맨해튼에서 길찾기

- Castéran and Sozeau: “A Gentle Introduction to Type Classes and Relations in Coq”, 2014
- 뉴욕 맨해튼에서 길을 물으면...

맨해튼에서 길찾기

- 길안내1:

```
North::North::East::South::South::South::
West::West::nil
```

- 길안내2:

```
South::West::nil
```

맨해튼에서 길찾기

- 길안내1과 길안내2가 동일하지는 않지만 동치이다.
  - 두 길안내가 동치 := 어느 지점에서 출발해도 동일한 지점에 도착
- 두 개의 길안내가 서로 동치라는 것을 증명하는 데에 타입 클래스를 활용할 수 있다.
  - r과 r'이 동치이고 s와 s'이 동치이면 r++s와 r'++s'도 동치이다.

맨해튼에서 길찾기

- 기본 정의 몇 개:

```
Inductive direction : Type :=
  North | East | South | West.

Definition route := list direction.

Record Point : Type :=
  {Point_x : Z; Point_y : Z}.

Definition Point_O := Build_Point 0 0.
```

맨해튼에서 길찾기

- 길안내 따라가기:

```
Definition translate (dx dy:Z) (P : Point) :=
  Build_Point (Point_x P + dx) (Point_y P + dy).

Fixpoint move (r:route) (P:Point) : Point :=
  match r with
  | nil => P
  | North :: r' => move r' (translate 0 1 P)
  | East :: r' => move r' (translate 1 0 P)
  | South :: r' => move r' (translate 0 (-1) P)
  | West :: r' => move r' (translate (-1) 0 P)
  end.
```

맨해튼에서 길찾기

- 길안내 동치 정의

```
Definition route_equiv : relation route :=
  fun r r' => forall P, move r P = move r' P.

Infix "=r=" := route_equiv (at level 70).

Example Ex1 :
  East::North::West::South::East::nil
    =r= East::nil.
Proof. ... Qed.
```

동치관계 클래스

- =r=가 정말로 동치관계(equivalence relation)이다.

```
Lemma route_equiv_refl : Reflexive route_equiv.
Proof. ... Qed.

Lemma route_equiv_sym : Symmetric route_equiv.
Proof. ... Qed.

Lemma route_equiv_trans : Transitive route_equiv.
Proof. ... Qed.
```

- 그런데 ....

```
Goal forall r, r =r= r.
  Proof. intro; reflexivity.
    Error:....
```

동치관계 클래스

- 동치관계 클래스와 인스턴스 선언

```
Class Equivalence {A} (R : relation A) : Prop := {
  Equivalence_Reflexive :> Reflexive R ;
  Equivalence_Symmetric :> Symmetric R ;
  Equivalence_Transitive :> Transitive R }.

Instance route_equiv_Equiv : Equivalence
Proof. ... Qed.


Goal forall r, r =r= r.
  Proof. intro; reflexivity. Qed.
```

Proper 함수

- rewrite 택틱 사용하기

```
Lemma route_cons :
  forall r r' d, r =r= r' -> d::r =r= d::r'.
Proof.
  intros r r' d H P; rewrite H.

Error: ....
Abort.
```

Proper 함수

- cons 함수가 proper 함수임을 Proper 클래스의 인스턴스로 확인해 주어야 함.

```
Instance cons_route_Proper (d:direction) :
  Proper (route_equiv ==> route_equiv) (cons d).
Proof. ... Qed.

Lemma route_cons :
  forall r r' d, r =r= r' -> d::r =r= d::r'.
Proof.
  intros r r' d H P; rewrite H; reflexivity.
Qed.
```

Proper 함수

- move와 app 함수도 proper이다.

```
Instance move_Proper :
  Proper (route_equiv ==> eq ==> eq) move.
Proof. ... Qed.

Instance app_route_Proper :
  Proper
    (route_equiv ==> route_equiv ==> route_equiv)
    (@app direction).
```

Proper 함수

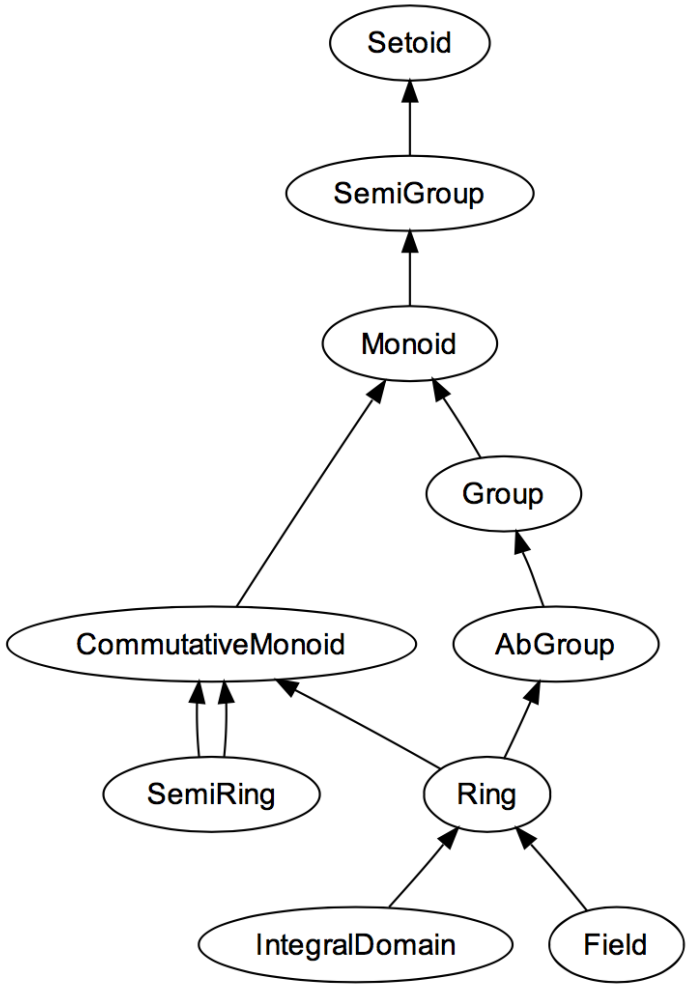
- 예제:

```
Example Ex2 :
  forall r, North::East::South::West::r =r= r.
Proof. ... Qed.

Example Ex3 : forall r r', r =r= r' ->
  North::East::South::West::r =r= r'.
Proof. intros r r' H. now rewrite Ex2. Qed.
```

수학에서 활용

- C-CORN 라이브러리 구조 (Univ. of Nijmegen)



할 수 있는 것들

- 변수 묶기(Variable binding) 관련 라이브러리 개발에 연구 활용 가능
- 함수의 오버로딩 뿐만아니라 증명의 오버로딩 가능.
  - Gonthier et al., “How to make ad hoc proof automation less ad hoc”, ICFP 2011.
- 제네릭 프로그래밍 적용 가능
  - GMeta 등등

감사합니다.

할 수 있는 것들

- 변수 묶기(Variable binding) 관련 라이브러리 개발에 연구 활용 가능
- 함수의 오버로딩 뿐만아니라 증명의 오버로딩 가능.
  - Gonthier et al., “How to make ad hoc proof automation less ad hoc”, ICFP 2011.
- 제네릭 프로그래밍 적용 가능
  - GMeta 등등

감사합니다.