Pattern matching dependent types in Coq

@mbrcknl

May 2015

```
match e as v in T i return R v i with
  C a => B a
end
```

For the branch

- substitute **C a** for **v**
- match the type of **C a** with **T i**

Then **B a** must have type **R v i**

For the whole **match** expression

- substitute **e** for **v**
- match the type of **e** with **T i**

Then the type of the **match** is **R v i**

A pattern match only refines its return type

Defer the introduction of arguments so their types appear in return annotations

Thread explicit evidence through the return annotation to recover relationships between things inside and outside a match (convoy pattern)

A pattern match must cover all constructors

Match on type indices in the return annotation to convert nonsense cases to a trivial type (unit)

How would you like pattern-matching to behave?

Write functions which provide that behaviour

Code

m.brck.nl/ylj15

Bibliography

adam.chlipala.net/cpdt