# Agda 中的证明，从三到四

2017, Nov 9 by Tesla Ice Zhang

这篇文章我们来说说模式匹配的另一种特殊情况。 到目前为止，很可能部分字符无法在一些字体下正常显示。

这篇文章主要讲 `rewrite` 。

注意，本文的函数命名依旧骚如白皮猪，请非 Agda / Lisp 用户不要在家里进行模仿，谢谢【

## 前置知识

- 上一篇文章

以及，由于 Agda 语言的特殊性，本文将继续使用 LaTeX 和代码块来共同展示代码。 代码块唯一的作用在于便于复制，主要的呈现途径为 LaTeX 。

## Dot pattern

我们来看模式匹配中的另一种特殊情况，叫 dot pattern 。

当模式匹配时，一个模式在另一个模式匹配的情况下会变得唯一（也就是说，条件性地唯一），我们可以把这个模式前面加一个 `.` ，比如：

$$\begin{aligned}&\mathrm{data}\ F : \mathbb{N} \to \mathrm{Set}\ \mathrm{where}\\&\quad \mathrm{ff} : (n : \mathbb{N}) \to F\ n\end{aligned}$$

$$\begin{aligned}&\mathrm{proof}_0 : \{n : \mathbb{N}\} \to F\ n \to F\ n \to F\ n\\&\mathrm{proof}_0\ (\mathrm{ff}\ n)\ (\mathrm{ff}\ .n) = \mathrm{ff}\ n\end{aligned}$$

```
data F : ℕ → Set where
  ff : (n : ℕ) → F n

proof₀ : {n : ℕ} → F n → F n → F n
proof₀ (ff n) (ff .n) = ff n
```

在这个例子当中，参数中的两个 `n` 相等，因此当其中一个参数确定时，第二个参数肯定也就确定了，因此可以使用 `.n` 来代替第二个参数（而不是使用另一个变量来把它匹配出来）。

这个很简单，也很好理解，我就不多说了。

## rewrite

看完这章你就能看懂很多定理证明代码所使用的语法了。。。

首先，我们定义这么一个 `rev` 函数，就是和 Haskell 的 `reverse` 一样的：

$$\begin{aligned}&\mathrm{rev} : \forall\{n\ m\}\{A : \mathrm{Set}\ n\} \to \mathrm{Vec}\ A\ m \to \mathrm{Vec}\ A\ m\\&\mathrm{rev}\ \ [] = []\\&\mathrm{rev}\ \ (x :: \mathrm{xs}) = \mathrm{rev}\ \ \mathrm{xs}\ ::^r x\end{aligned}$$

```
rev : ∀ {n m} {A : Set n} → Vec A m → Vec A m
rev [] = []
rev (x ∷ xs) = rev xs ∷ʳ x
```

其中，$::^r$ 的作用是将一个元素放在链表的尾巴上。

这里说明一下，我的博客使用的 MathJax 不支持 `\squaredots` 这个 LaTeX 命令，因此我只能使用两个 `:` 代替，请读者谅解。

然后我们可以写一些证明。

## 证明一：关于 rev

我们要证明的，是这样的逻辑：

> $\forall v \to \mathrm{rev}(\mathrm{rev}\, v) \equiv v$

蓝后我们可以稍微思考一下。 要证明这个东西，我们可以先证明它的一个部分，即

> $\forall a, v \to \mathrm{rev}\ \ (v ::^r a) \equiv a :: \mathrm{rev}\, v$

。这个逻辑应该很容易就可以看懂，我就不再用自然语言解释一遍了。

我们可以先把这个命题使用 Agda 的类型表达出来（比前文还骚的函数命名出现了！）：

$$\mathrm{rev\$v{:}a{=}a{:}rev\$v} : \forall\{n\ m\}\{A : \mathrm{Set}\ n\}(a : A)(v : \mathrm{Vec}\ A\ m) \to \mathrm{rev}(v ::^r a) \equiv a :: \mathrm{rev}$$

```
rev$v:a=a:rev$v : ∀ {n m} {A : Set n} (a : A) (v : Vec A m) →
```

```
rev (v ::ʳ a) ≡ a :: rev v
```

然后我们开始证明。 写下模式匹配的结果：

$$\begin{aligned}&\mathrm{rev\$v{:}a{=}a{:}rev\$v}\ a\ [] = \ ?\\&\mathrm{rev\$v{:}a{=}a{:}rev\$v}\ a\ (\_ :: \mathrm{xs}) = \ ?\end{aligned}$$

```
rev$v:a=a:rev$v a [] = ?
rev$v:a=a:rev$v a (x :: xs) = ?
```

第一个情况很显然，直接用 `refl` 表达 $a \equiv a$ 就好了，即得易见平凡。 但是我们如何证明第二种情况呢？

## 证明思路

我们可以用这个思路来证明（为了方便看，我在此处就直接使用 `lemma` 代替上文的 $\mathrm{rev\$v{:}a{=}a{:}rev\$v}$ 了）：

$$\forall a, x, xs \to \mathrm{lemma}(a, \mathrm{xs}) \to \mathrm{rev}(\mathrm{xs}\ ::a) \equiv (a ::^r \mathrm{rev}(\mathrm{xs})) \to \mathrm{lemma}(a, x :: \mathrm{xs})$$

这个逻辑放在我们正常人眼中就是一个看起来有点点流氓但实际上非常有道理的逻辑。
首先，我们摆出了这样一个事实： 既然我们这个证明是归纳证明，而且我们已经证明了第一种情况（就是上面那个即得易见平凡）， 因此我们可以在后面的证明中引用前面的情况，在上面的例子中，就是 $\mathrm{lemma}(a, \mathrm{xs})$ 这种情况。
由于 $\mathrm{lemma}(a, xs)$ 成立，我们可以知道 $\mathrm{rev}(\mathrm{xs}\ ::a) \equiv (a ::^r \mathrm{rev}(\mathrm{xs}))$，从而得出 $\mathrm{lemma}(a, x :: \mathrm{xs})$。
这就是我们的证明过程。

## 使用 with

相信大家还记得 Agda 中的模式匹配，其中一种便是 `with` （这个和 Haskell的 `case of` 是有很大区别的，但是其实也是模式匹配）。

这个证明写成 Agda 代码，就是：

$$\begin{aligned}&\mathrm{rev\$v{:}a{=}a{:}rev\$v} : \forall\{n\ m\}\{A : \mathrm{Set}\ n\}(a : A)(v : \mathrm{Vec}\ A\ m) \to \mathrm{rev}(v ::^r a) \equiv a :: \mathrm{rev}\\&\mathrm{rev\$v{:}a{=}a{:}rev\$v}\ \_\ [] = \mathrm{refl}\\&\mathrm{rev\$v{:}a{=}a{:}rev\$v}\ a\ (\_ :: \mathrm{xs})\ \mathrm{with}\ \ \mathrm{rev}(\mathrm{xs}\ ::^r a) \mid \mathrm{rev\$v{:}a{=}a{:}rev\$v}\ a\ \ \mathrm{xs}\\&\ldots \qquad\qquad\qquad\qquad\qquad\qquad \mid .(a :: \mathrm{rev\ xs}) \mid \mathrm{refl} = \mathrm{refl}\end{aligned}$$

```
rev$v:a=a:rev$v : ∀ {n m} {A : Set n} (a : A) (v : Vec A m) →
                  rev (v ::ʳ a) ≡ a :: rev v
rev$v:a=a:rev$v _ [] = refl
rev$v:a=a:rev$v a (x :: xs) with rev (xs ::ʳ a) | rev$v:a=a:rev$v a xs
...                            | .(a :: rev xs) | refl = refl
```

可以看到，我们使用模式匹配摆出了前面的证明成立的事实，然后使用了一个 dot pattern 来把前面的证明所说明的东西给摆出来。

这个前面的证明，指 $\mathrm{rev\$}v{:}a{=}a{:}\mathrm{rev\$}v\ a\ \mathrm{xs}$。

这个证明所说明的，是 $\mathrm{rev}(\mathrm{xs}\ ::^r\ a)$ 可以用 $.(a :: \mathrm{rev\ xs})$ 给匹配出来。

我们可以发现，这个证明写成一个通用的形式，就是这个形状：

$$\begin{aligned}&\mathrm{proof}_0\ \mathrm{params}\ \mathrm{with}\ \mathrm{lhs} \mid \mathrm{proof}_0\ \mathrm{params}\\ &\ldots \qquad\qquad\quad \mid .\mathrm{rhs} \mid \mathrm{refl} = \mathrm{proof}_1\ \mathrm{params}\end{aligned}$$

```
proof₀ params with lhs | proof₀ params
...              | .rhs | refl = proof₁ params
```

而 `lhs` 和 `rhs` 具体是什么，都可以通过前文所说的“前面的证明”给推导出来。因此，这种写法本身是相当啰嗦的。

## 使用 rewrite

为了简化这种常用的结构，各种证明语言都提供了一个叫 `rewrite` 的语法糖来代替它，可以省去 `lhs` 和 `rhs` 的书写。 上面的使用 `with` 的代码通用结构，可以写成这种形式：

$$\begin{aligned}&\mathrm{proof}_0\ \mathrm{params}\\ &\quad \mathrm{rewrite}\ \mathrm{proof}_0\ \ \mathrm{params}\\ &\qquad = \mathrm{proof}_1\ \mathrm{params}\end{aligned}$$

```
proof₀ params
  rewrite proof₀ params
        = proof₁ params
```

最后使用 `rewrite` 的完整带签名版本就是这样啦：

$$\begin{aligned}
&\mathrm{rev\$v{:}a{=}a{:}rev\$v} : \forall\{n\ m\}\{A : \mathrm{Set}\ n\}(a : A)(v : \mathrm{Vec}\ A\ m) \to \mathrm{rev}(v ::^r a) \equiv a :: \mathrm{rev} \\
&\mathrm{rev\$v{:}a{=}a{:}rev\$v}\ _\ [] = \mathrm{refl} \\
&\mathrm{rev\$v{:}a{=}a{:}rev\$v}\ a\ (_ ::\ \mathrm{xs}) \\
&\quad \mathrm{rewrite}\ \mathrm{rev\$v{:}a{=}a{:}rev\$v}\ a\ \ \mathrm{xs} \\
&\qquad = \mathrm{refl}
\end{aligned}$$

```
rev$v:a=a:rev$v : ∀ {n m} {A : Set n} (a : A) (v : Vec A m) →
                  rev (v ::ʳ a) ≡ a :: rev v
rev$v:a=a:rev$v _ [] = refl
rev$v:a=a:rev$v a (_ :: xs)
  rewrite rev$v:a=a:rev$v a xs
        = refl
```

可以看到， `rewrite` 使我们可以略去一些琐碎的证明中间过程，然后通过引用更小的情况直接 QED 。 每个更大的情况都是更小的情况的推论。 因此我们可以这么说：

> 反之亦然同理，推论自然成立。略去过程 QED ，由上可知证毕。

## 作业

使用 $\mathrm{rev\$v{:}a{=}a{:}rev\$v}$ 证明 $\mathrm{rev}(\mathrm{rev}\ v) \equiv v$ 。

## 结束

是的，我说完了。

$$\downarrow$$
$$\bot$$

---

Tweet this

Top

创建一个 issue 以申请评论

Create an issue to apply for commentary

---