076: showlist2.hs が再びエラー、dictionary not found

↑ up

- issued: 2020-05-15
- 分類： A サンプルコードが fail
- status: Closed (2020-10-01)

概要

035 対処により、以前通っていた showlist2.hs がエラーするようになってしまったので、sample194 から再度降格。

現象：

```
$ ./trun testcases/showlist2.hs
# 1. tcompile
source file: testcases/showlist2.hs
dst dir: /showlist2
doCompile ... done.
implicitPrelude ... done.
doCompile ... bunnyc: Error: dictionary not found: ("Prelude.Show",Tyvar "t40" Star,[(IsIn "Prelude.
CallStack (from HasCallStack):
  error, called at src/DictPass.hs:261:25 in main:DictPass
```

調査ログ

2020-09-30 (Wed)

現象の絞り込み

問題を特定するために、showlist2.hs を小さくしていく。

以下のようにすると、エラーせずに実行できてしまった：

```
main = do
  putStrLn $ showlist' [1]
    where
      showlist' []  = "[]"
      showlist' [x] =  "[" ++ show x ++ "]"
```

以下では再現：

```
main = do
  putStrLn $ showlist' [1]
    where
      showlist' [x] =  "[" ++ show x ++ "]"
      showlist' (x:xs) = "[" ++ foldl (\s t -> s ++ "," ++ show t) (show x) xs ++ "]"
```

もうすこし簡単にしてみた（showlist2b.hs):

```
showlist' [x]    = show x
showlist' (x:xs) = showlist' xs
main = putStrLn $ showlist' [1]
```

これをコンパイルすると、次のようになる。

```
$ ./trun testcases/showlist2b.hs
# 1. tcompile
source file: testcases/showlist2b.hs
dst dir: /showlist2b
doCompile ... done.
implicitPrelude ... done.
doCompile ... bunnyc: Error: dictionary not found: ("Prelude.Show",Tyvar "t20" Star,[(IsIn "Prelude.
CallStack (from HasCallStack):
  error, called at src/DictPass.hs:261:25 in main:DictPass
```

--ddump-core0 の結果は次のとおり (showlist' 部のみ)：

```
(Main.showlist' :: ([Prelude.Show t22] :=> ([t22] -> [Prelude.Char]))) =
    \(_Main.showlist'.U1 :: ([Prelude.Show t7] :=> [t7])) ->
        case (_Main.showlist'.U1 :: ([Prelude.Show t7] :=> [t7])) (_Main.showlist'.U1b :: ([Prelude.
            Prelude.: (_Main.showlist'.U2 :: t10) (_Main.showlist'.U3 :: [t10]) :: (t10 -> ([t10] ->
                case (_Main.showlist'.U3 :: [t10]) (_Main.showlist'.U3b :: [t10]) of
                    Prelude.: (_Main.showlist'.U3 :: t18) (_Main.showlist'.U4 :: [t18]) :: (t18 -> (
                        ((Main.showlist' :: ([Prelude.Show t20] :=> ([t20] -> [Prelude.Char])))
                          (_Main.showlist'.U3 :: t18))
```

```
                    Prelude.[]  :: [t12] ->
                        ((Prelude.show :: ([Prelude.Show t16] :=> (t16 -> [Prelude.Char])))
                          (_Main.showlist'.U2 :: t10))

            Prelude.[]  :: [t8] ->
                (Prim.FAIL :: t9)
```

showlist' xs に相当する部分が((Main.showlist' :: ([Prelude.Show t20] :=> ([t20] -> [Prelude.Char]))) (
なのだが、型引数が [t20] なのに対して、引数の型が t18 となっている（前者が型変数のリストだが、後者はリストでない)。

いっぽう、これによくにたプログラム（showlist2c.hs) ではエラーは発生しない。この core0 は以下のとおり：

```
(Main.showlist' :: ([Prelude.Show t17] :=> ([t17] -> [Prelude.Char]))) =
    \(_Main.showlist'.U1 :: ([Prelude.Show t7] :=> [t7])) ->
        case (_Main.showlist'.U1 :: ([Prelude.Show t7] :=> [t7])) (_Main.showlist'.U1b :: ([Prelude.
            Prelude.: (_Main.showlist'.U2 :: t10) (_Main.showlist'.U3 :: [t10]) :: (t10 -> ([t10] ->
                (((Prelude.++ :: ([t12] -> ([t12] -> [t12])))
                    ((Prelude.show :: ([Prelude.Show t13] :=> (t13 -> [Prelude.Char])))
                      (_Main.showlist'.U2 :: t10)))
                  ((Main.showlist' :: ([Prelude.Show t15] :=> ([t15] -> [Prelude.Char])))
                    (_Main.showlist'.U3 :: [t10])))
            Prelude.[]  :: [t8] ->  ""
```

同じ部分に着目すると、((Main.showlist' :: ([Prelude.Show t15] :=> ([t15] -> [Prelude.Char]))) (_Main.sho
のようになっていて、[t15] と [t10] が対応している。

このあたりに着目してみるとよさそう。

原因調査

よくみてみると、型付けうんぬんのまえに、局所変数の解決がおかしい。

(x:xs) -> showlist' xs が、: U3 U4 -> showlist' U3 になっている。どこで狂った？

ああ、ちがうな。showlist2b.hs は次のような「平坦な case 式」に展開されるはずなのだが、

```
showlist' xs = case xs of
  (y:ys) -> case ys of
    (z:zs) -> showlist' ys
    []     -> show y
```

```
main = do putStrLn $ showlist' [1]
          putStrLn $ showlist' [1, 2, 3]
```

上記における ys と、z がいずれも U3 になってしまっている。

パターンマッチングコンパイラ（Pattern.ns) が入れ子になっているときの α 変換でしくっている（重複しておなじ文字番号をわりあててしまっている）ようだ。

2020-10-01 (Thu)

Pattern.hs において、Case 式が深くなるときに prefix にレベル相当を足してやればよかった。簡単に以下のように修正：

```
--- a/compiler/src/Pattern.hs
+++ b/compiler/src/Pattern.hs
@@ -93,7 +93,7 @@ reduceMatch ci n0 k0 vs0 qs0 def0 =

         matchClause c k (_:us) qs def =
           Clause c us' (match
-                        n
+                        (n ++ "._c")
                         (k + k)
                         (us' ++ us)
                         [(ps' ++ ps, e) | (PCon _ ps':ps, e) <- qs]
```

showlist2.hs は sample282.hs としてクローズ。