2016 年 4 月

2016-04-30 (Sat)

[Haskell] Bunny: trExpr

昨日気付いた問題を直した。Pattern Match コンパイラが生成した Lambda 式の仮引数を as に加えて trExpr を呼ぶように修正。いっしゅん、as が状態になっているので State モナドにしないといけないかなと思ったんだけど、仮引数のスコープは局所的なので、再帰の呼び出し側には影響なくてよいので、ＯＫだった。

680f0431d15c9ab3bf8613e32d7a0783581197b4[*1]

2016-04-29 (Fri)

[Haskell] Bunny

今日も少しだけ書いたよ[*2] という。

Pattarn.Expression, Core.Expr どっちにおいても、case 式の構造の意味がよくわからんまま、苦し紛れにとりあえず書いた。

ま、中間言語の仕様は、実際につぎの形式に変換しようとしないとわからないところがあるので、ある程度仕方がないかな。

もうひとつ、今日書いてみて気付いたのは、Pattern Match コンパイラが新しく導入した変数の扱いのために、trExpr ではなく、ローカル環境の trpat にしてたんだけど、これは間違いだった。新しく導入した変数の情報は as :: [Assump] につっこんで、trExpr を再帰的につかうべきだった。。こちらは、次に直そう。

2016-04-27 (Wed)

[Haskell] Bunny: TrCore

今日もちょっと書くぞということで、某ルノワールではすける。

昨日は sample1 をどうにか Core 変換したので、次は sample4 なわけだけど…。Pattern.Expression と Core.Expr を見比べても、なんだ？よくわからないぞ？

それもそのはず、パターンマッチの「平坦化」が間違えていた。

```
f True = "true message"
f False = "false message"
```

は、

```
f = \u1 -> case u1 of
```

[*1] https://github.com/unnohideyuki/bunny/commit/680f0431d15c9ab3bf8613e32d7a0783581197b4

[*2] https://github.com/unnohideyuki/bunny/commit/b87db92ec01ea5901921181d9c7f8979fc6b37cb

```
True -> "true message"
False -> "false message"
```

のように平坦化されるべきなんだけど、lambda 式でくるむのをわすれていて（参考にした論文はその部分について省略されていたんだと思う）。なもんだから、いざ Core に変換しようとして、なぞの自由変数が出現してしまっていて「んん？？」となったというわけでした。

そういうわけで、今日はとりあえず、Pattern.Expression に Lambda を足して、そいつを使うように変更[*3]。Desugar が Desugar.hs と Pattern.hs に分散しているのはきしょいが。そういうのは、あと。あと。

[*3] https://github.com/unnohideyuki/bunny/commit/f97e91e677ade9b5eead356bc5ce928ab35f5ddc