

プログラミング言語Egisonのコンパイラの開発

一超強力なパターンマッチプログラミング

江木聡志(東京大学大学院 情報理工学系研究科)

- Egisonは強力なパターンマッチをプログラマが表現できる新しい言語
 - 従来のプログラミング言語では集合や多重集合のような1つの定まった形を持たないデータ型に対する直感的なパターンマッチは不可能だった。
 - プログラマが集合や多重集合のような(正規形を持たない)データ型のパターンマッチの方法を定義する。それを使用してプログラムを直感的に記述することができる。
 - 処理系がHackageのパッケージとしてフリーで配布されている。
 - ソースコードはGithubで公開されている(オープンソース)。
- Egisonのパターンマッチの特徴
 - 複数のパターンマッチの結果を扱える。
 - パターンマッチの際、パターンの左側で得られた束縛を、それより右側のパターンで参照できる。
 - Loopパターン(パラメータの値によって変わってくるパターンの繰り返しを表現するためのパターン)、Notパターン(パターンマッチに成功しない場合にマッチするパターン)、Macroパターン(パターンをモジュール化して再利用するための機構)、Cutパターン(パターンマッチの際のバックトラックを制御するためのパターン)などパターンの表現力を上げるための独自パターンがある。
- Egisonのコンパイラを作成、ライブラリ関数の整理などを行い、速度が最高で7倍近く速くなった！
- Egisonリンク
 - <http://hagi.is.s.u-tokyo.ac.jp/~egi/egison>
 - <https://github.com/egisatoshi/egison2>
 - <http://hackage.haskell.org/package/egison>

プログラミング言語Egisonのコンパイラの開発

一超強力なパターンマッチプログラミングー

江木聡志(東京大学大学院 情報理工学系研究科)

- Egisonで書いたN-Queenの解にマッチするパターンマッチ

– <http://hagi.is.s.u-tokyo.ac.jp/~egi/egison/manual/n-queen.html>

The image shows a snippet of Egison code for solving the N-Queen problem, with several annotations explaining specific features:

- Multiple pattern matches:** A callout points to `(match-all (between 1 n) (Multiset Integer))`, stating: "複数のパターンマッチ全ての結果についてマッチ節の式を実行する" (Execute the expression in the match clause for all results of multiple pattern matches).
- Multiset:** A callout points to `(Multiset Integer)`, stating: "整数の多重集合としてパターンマッチを行う" (Perform pattern matching as a multiset of integers).
- Cons and loop:** A callout points to `[<cons $a_1`, stating: "添字付き変数" (Indexed variable).
- Loop pattern:** A callout points to `(loop $l $i (between 2 n))`, stating: "Loopパターン" (Loop pattern).
- And pattern:** A callout points to `(& ^, (- a_i1 (- i i1))`, stating: "Andパターン" (And pattern).
- Not pattern:** A callout points to `^, (+ a_i1 (- i i1))`, stating: "Notパターン" (Not pattern).
- Value pattern:** A callout points to `$a_i)`, stating: "Valueパターン: ターゲットとマッチ可能な値ならマッチ可能な値ならマッチする" (Value pattern: Match if the target is a value that can match).

```
(define $n-queen
  (lambda [$n]
    (match-all (between 1 n) (Multiset Integer)
      [<cons $a_1
        (loop $l $i (between 2 n)
          <cons (loop $l1 $i1 (between 1 (- i 1))
            (& ^, (- a_i1 (- i i1))
              ^, (+ a_i1 (- i i1))
                l1)
              $a_i)
            l>
          <nil>)>
        (loop $l $i (between 1 n) {a_i @l} {}))])))
```