2004 年 8 月 7 日
Lightweight Language Weekend 2004 ／君ならどう書く
須藤功平（COZMIXNG）<kou@cozmixng.org>／ Gauche 担当

1 n-Queens ゲーム

1.1 方針

自分のターンではない時でも，現在の盤面の状態から到達可能な手をリストアップし続ける．つまり，少しでも多くの手を計算しておけば有利なのではないかという（誰でも考え付く）「下手な鉄砲数うちゃ当たる作戦」である．

そのために，複数のプロセスを起動し[1]，数にものを言わせて手をリストアップする．各プロセスはタプルスペースを用いて通信を行う．

1.2 出演者

今回のプログラムにおける各出演者の関係を図 1 に示す．

プレイヤー n-Queens サーバと通信し，実際に n-Queens ゲームをプレイする．

プレイヤーはどこに Queen をおけばよいかは考えずに，単に現在の盤面の状態をタプルスペースに書き，解答者が解いてくれるのを待つ．自分のターンになったら解答者が解いた解答をタプルスペースからかき集め，その中から適当な解答を選択する．自分のターンに置く Queen はその解答を元に決める．

適当な解答は以下のように選択する．上にあるものほど優先順位が高い．

1. 次のターンで終了する解答
2. 次の自分のターンが来るまでに終了する解答
3. 自分のターンでは終了しない解答
4. かき集めた中の最初の解答

出題者 プレイヤーがタプルスペースに書いた盤面の状態をサブ問題に分割し，それをタプルスペースに書く．

出題者は以下のようなサブ問題を作成する．最終状態とはもうこれ以上 Queen が置けなくなる盤面の状態のことである．

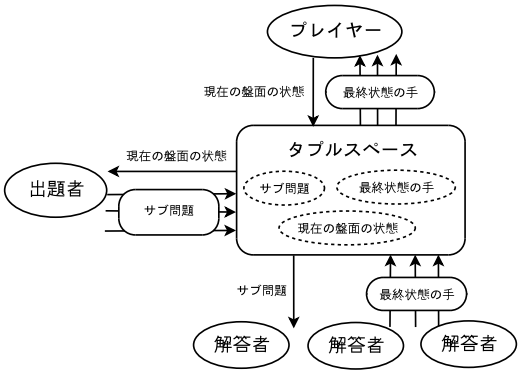

図 1: システム構成

[1] Gauche の用いている Boehm GC は FreeBSD 上での pthread と相性が悪いようで，スレッドは使えない．

- 元の盤面の状態から始めて，最終状態を全て求める問題．
- 元の盤面の状態に（Queen を置ける場所に）適当に Queen をひとつ置いた状態から始めて，最終状態を全て求める問題．

解答者 出題者が作成したサブ問題を解いて，解答をタプルスペースに書いていく．システムには複数の解答者が存在する．

解答者は以下のどちらかの条件を満たすまでサブ問題を解き続ける．

- 現在の盤面の状態からどんな手順を踏んでも，与えられたサブ問題と同じ状態にならない．
- 解答を全て生成した．

タプルスペース プレイヤー，出題者，解答者が通信するための場所を提供する．プレイヤー，出題者，解答者がどのように通信しているかには一切関知しない．

2 プログラム

今回のプログラムで一番大事な部分はタプルスペースを用いた「夏休み終了直前のこどもの宿題を家族総出で片付けよう作戦」であるが，これは Scheme っぽくないのでプログラム中で使用している継続[2]について解説する．

以下は解答者オブジェクト（実体は手続き）を作成するコードである．`terminated-hands` は順次もうこれ以上 Queen が置けなくなる盤面の状態を求め，それを引数として，`terminated-hands` の第 4 引数で指定された手続き（以下のコードでは `(lambda (hand) ...)`）を呼び出す手続きである．

```
(define (make-answerer queens width height)
  (define return #f)

  (define (next)
    (terminated-hands queens width height
                      (lambda (hand)
                        (let/cc restart
                          (set! next (lambda () (restart 'do-next)))
                          (return hand))))
    (return #f))

  (lambda ()
    (let/cc cont
      (set! return cont)
      (next))))
```

以下のように使用する．

```
(define answerer (make-answerer 盤面の状態 盤面の幅 盤面の高さ))
(answerer) ;; -> 最初に見付けた解
...
(answerer) ;; -> 最後に見付けた解
(answerer) ;; -> #f
...
```

む，やはり解説は当日までとっておくことにしよう．それを書くにはこの余白はあまりにも狭すぎる．

[2] こっちの方が Scheme っぽいでしょ？