```
#lang scheme

(require "formula.scm")

(define (sudoku-var i j k ) (+ (* i 100) (* j 10) k))

(define (sudoku-neg-var i j k ) (* -1 (sudoku-var i j k)))

(define N (list 1 2 3 4 5 6 7 8 9))

(define (phi-matrix-1)
  (for*/list ([i N]
              [j N])
    (for*/list ([k N])
      (sudoku-var i j k))))

(define (phi-matrix-2)
  (for*/list ([i N]
              [j N]
              [k1 N]
              [k2 N]
              #:when (not (= k1 k2)))
      (list (sudoku-neg-var i j k1) (sudoku-neg-var i j k2))))

(define (sudoku-ref s i j)
  (list-ref (list-ref s (- i 1)) (- j 1)))

(define (phi-s s)
  (for*/list ([i N]
              [j N]
              [k N]
              #:when (= (sudoku-ref s i j) k))
      (list (sudoku-var i j k))))

(define sudoku-instance (list
  (list 0 0 0  0 0 0  0 1 0)
  (list 4 0 0  0 0 0  0 0 0)
  (list 0 2 0  0 0 0  0 0 0)
  (list 0 0 0  0 5 0  4 0 7)
  (list 0 0 8  0 0 0  3 0 0)
  (list 0 0 1  0 9 0  0 0 0)
  (list 3 0 0  4 0 0  2 0 0)
  (list 0 5 0  1 0 0  0 0 0)
  (list 0 0 0  8 0 6  0 0 0)
))
```