```
#lang racket
(require racket/trace)

;; Auxiliary function
(define (add_prefix pre lst)
  (map (lambda (x) (cons pre x)) lst))

; Example:
;> (add_prefix 0 '((0) (1)))
;'((0 0) (0 1))

;; The algorithm
(define (hamiltonian_cycle_on_cube n)
  (cond
    [(<= n 1) '((0) (1))]
    [else
     (let* ((inner_cube (hamiltonian_cycle_on_cube (- n 1)))
            (zerolist (add_prefix 0 inner_cube))
            (onelist (add_prefix 1 (reverse inner_cube))))
       (append zerolist onelist))]))

; Example:
(trace hamiltonian_cycle_on_cube)
(hamiltonian_cycle_on_cube 3)
;>(hamiltonian_cycle_on_cube 3)
;> (hamiltonian_cycle_on_cube 2)
;> >(hamiltonian_cycle_on_cube 1)
;< <'((0) (1))
;< '((0 0) (0 1) (1 1) (1 0))
;<'((0 0 0) (0 0 1) (0 1 1) (0 1 0) (1 1 0) (1 1 1) (1 0 1) (1 0 0))
;'((0 0 0) (0 0 1) (0 1 1) (0 1 0) (1 1 0) (1 1 1) (1 0 1) (1 0 0))
```