#/usr/bin/python

import itertools as it

import numpy as np

from scipy import linalg as LA

import time

import numba

from numba import jit, njit,config, threading\_layer, prange, cuda

from numba.typed import List

cuda.select\_device(0)

#set the threading layer before any parallel target compilation

config.THREADING\_LAYER = 'threadsafe'

# function optimized to run on gpu

@njit(parallel=True)

def s2(nSite, basis, vecm, sProduct, length):

sSquare = 0

for zz in prange (length):

s1Square = 0

szVal = 0

sxyVal = 0

c1 = vecm[zz]

c2 = 0

for idx, ix in enumerate(sProduct):

if (ix[0] == ix[1]):

s1Square += (0.75)\*c1\*c1

if (ix[0] != ix[1]):

if (basis[zz][ix[0]]) == (basis[zz][ix[1]]):

szVal += 0.25\*c1\*c1

if (basis[zz][ix[0]]) != (basis[zz][ix[1]]):

szVal -= 0.25\*c1\*c1

basis1 = list(basis[zz])

basis1[ix[0]], basis1[ix[1]] = basis1[ix[1]], basis1[ix[0]]

basis2 = ''.join(basis1)

exists = basis2 in basis

if exists == True:

basis1Key = basis.index(basis2)

c2 = vecm[basis1Key]

sxyVal += 0.5 \*c1\*c2

sSquare += s1Square + szVal + sxyVal

#print(sSquare)

return round(sSquare, 4)

###########################################################

####################The End#################################