Theory CFGCI

Vijay Gopal Chilkuri

January 12, 2021

Contents

1 Introduction

There are three main aspects while using configuration based representation

2 Algorithms

2.1 Small function to convert int to bit

Convert input CFG to an array of bits containing 0's and 1's representing the position of SOMOs and DOMOs.

```
void int_to_bin_digit(unsigned int in, int count, int* out)
{
    /* assert: count <= sizeof(int)*CHAR_BIT */
    unsigned int mask = 1U << (count-1);
    int i;
    for (i = 0; i < count; i++) {
        out[i] = (in & mask) ? 1 : 0;
        in <<= 1;
    }
}
```

2.2 Print all CFGs

The only input required is the selected list of CFGs at a given CIPSI iteration I.

```
#include <stdio.h>

void printcfglist_(int *inpcfgs, int *inpNint, int *inpNcfgs){
  int *cfglist =  inpcfgs;
  int Ncfgs    = *inpNcfgs;
  int N_int    = *inpNint;
  int stepInt  = sizeof(int);
  int digit[18];
  int cfg;
  printf("In printcfglist\n");
  printf("Ncfgs = %d Nint=%d size=%d\n",Ncfgs, N_int,stepInt);
  printf(" 1-- %d \n -- %d \n",cfglist[0*(2*Ncfgs) + 0*(Ncfgs) + 0], cfglist[0*(2*Ncfg
  for(int i = 2; i < 28; i+=stepInt){
      cfg = cfglist[0*(2*Ncfgs) + 0*(Ncfgs) + i];
      printf("%d> %d\n",i,cfg);
      int_to_bin_digit(cfg,18,digit);
      for(int j=0;j<18;j++)
          printf("%d ",digit[j]);
      printf("\n");
  }
}
```

2.3 Calculate the orthogonalization matrix

The orthogonalization matrix gives the orthonormalized vectors in bonded-function (BF) (or determinant) basis which are eigenfunctions of S^2 c.f. $\mathbf{O}_i$ matrices.

```
void getOrthoMatrix(int *cfg, Matrix &orthoMat);
```

2.4 Make the prototype matrices

jThe prototype matrices give the matrix-elements (MEs) for a given type of excitation $p- > q$ of a specific type between two CFGs I, J.

These matrices are independent of the MOs and only depend on the total number of electrons nel, total number of orbitals $norb$, and the total spin S.

```
void makePrototypeMatrices(int nel, int norb, double spin);
```

2.5 Functions required for calculating MEs

2.5.1 Calculate Overlap between two BFs

The overlap between two bonded functions is based on the derivations by Cooper and McWeeney[?] and Sutcliffe[?]. They are based on Rumer diagrams. Here, we shall briefly outline the algorithm for the calculation of the overlap between two BFs.

In order to calculate the Overlap (S) between two bonded functions V_r and V_s, there are two steps which are as follows:

1. Permutations of the strings to bring V_r, V_s into maximum overlap configuration. This incurs a phase (-1 for each permutation) $(-1)^r$.
2. The calculation of the number of Islands (i), the number of Open chains (O), and the number of E chains.

The description of the three types of diagrams is described below:

1. Islands

   Once the two BFs are brought into maximum overlap, the number of islands can be calculated. An island is defined as the total number of closed polygons formed by joining the common indices in V_r and V_s. The pairs in each BF V_r and V_s are also joind by an arc. Each island has two primitive spin-functions. A primitive spin-function is defined as a product of $\alpha - \beta$ pair in the two BFs. The two primitives originate from assigning α or β to the head and tail of the closed polygon or vice-versa.

```
#include <stdio.h>

void calculateislands_(int *inpbfvr, int *inpbfvs, int *inpNMO, double *me){
  int *bfvr  =  inpbfvr;
  int *bfvs  =  inpbfvs;
  int NMO      = *inpNMO;
  int stepInt  = sizeof(int);
}
```

2. Open chains (O)

   The open chains are constituted of open polygons which have an odd number of sides. These are made up of BFs which consist of unpaired spins such as $2^{-1/2}\left[\alpha(i)\beta(j)+\alpha(j)\beta(i)\right]\alpha(k)$ for V_r and $\alpha(i) - 2^{-1/2}\left[\alpha(j)\beta(k)+\alpha(k)\beta(j)\right]$ for V_s respectively. These contribute a factor of 1 to the MEs.

3. E type chains

   The E type chains originate from BFs which contain different indices which are mutually exclusive. The presence of mutually exclusive indices results in a vanishing ME between such BFs.

   Finally, the BFs contribute to the ME as follows:

   $\langle V_r|V_s\rangle = \delta_{SS'}\delta_E 2^{i-s}(-1)^r$

   Where S and S' are the spins for the V_r and V_s respectively and s represents the total number of pairs in V_r and V_s. The total number of pairs are the same in V_r and V_s if they belong to the same spin subspace.

2.6 Calculate the Operator MEs

The operator matrix-elements are calculated using the bonded-function (or determinant) basis and are called A^{pq}_{IK}, where p, q are the two molecular orbital indices and I, K are the two CFGs.

Note that this function simply returns the value from a prototype lookup table which contains the pretabulated values for a given p, q excitation of a specified type which is one of the four:

1. SOMO $->$ VMO
2. SOMO $->$ SOMO
3. DOMO $->$ VMO
4. DOMO $->$ SOMO

```
void getOneElOperatorMatrix(int *cfgI, int *cfgK);
```

2.7 Sigma-Vector I

The one-electron part $\sum_{pq} \tilde{h}_{pq} < \Psi|\hat{E}_{pq}|\Psi >$.

2.8 Sigma-Vector II

The two-electron part $\frac{1}{2}\sum_{pq,rs} g(pq,rs) < \Psi|\hat{E}_{pq}\hat{E}_{rs}|\Psi >$

2.8.1 Function to calculate Sigma-Vector

```
<<getOrthoMatrix>>
void calcSigma(double *coeff, double *Gpqrs);
```

3 Bibliography

bibliography:biblio.org