Structure of TCE

So Hirata

Pacific Northwest National Laboratory

SciDAC
Scientific Discovery through Advanced Computing

Pacific Northwest National Laboratory
Operated by Battelle for the U.S. Department of Energy

This presentation will address

- How TCE works
- How TCE-generated programs work
- How TCE can be used
- How TCE can be enhanced
- Other sources of information

How TCE works —— overview

- TCE derives working equations for a second quantized many-electron theory
  - TCE is written in Python® programming language
- TCE implements the working equations into parallel programs
  - For the NWChem© QC software suite
  - Using ParSoft© Library

How TCE works —— supported models

$$\langle Q|[(L_0 + L_1 + \ldots)(H_1 + H_2)\exp(T_1 + T_2 + \ldots)(R_0 + R_1 + \ldots)]_{C/L}|P\rangle$$

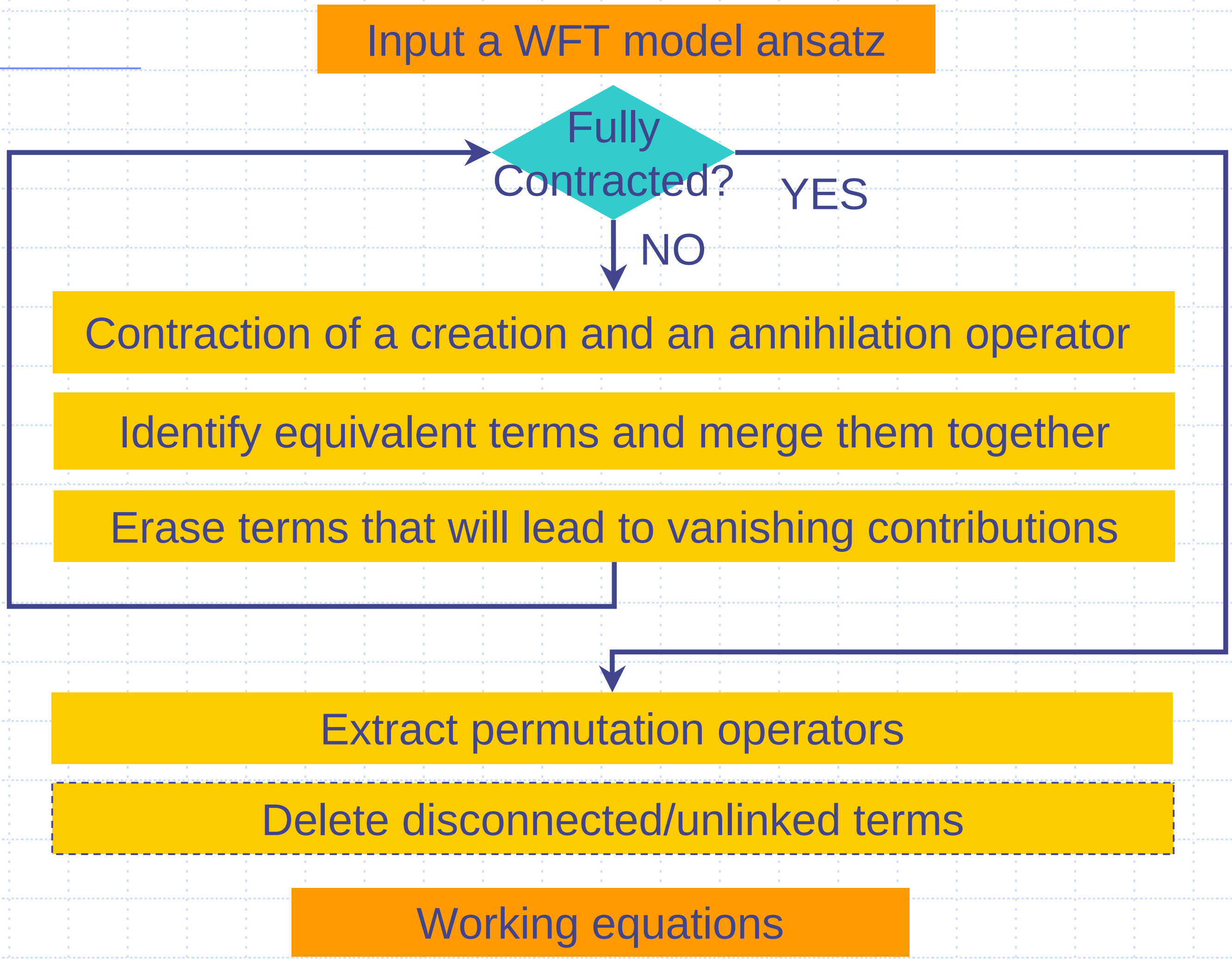
- CISD $\langle D|(H_1 + H_2)(R_1 + R_2)|O\rangle = E\langle D|(R_1 + R_2)|O\rangle$
- MBPT(2) $\langle D|(H_1)(R_1 + R_2)|O\rangle + \langle D|(H_2)|O\rangle = 0$
- CCSDT $\langle T|[(H_1 + H_2)\exp(T_1 + T_2 + T_3)]_C|O\rangle = 0$
- EOM-CCSD $\langle D|[(H_1 + H_2)\exp(T_1 + T_2)(R_1 + R_1)]_C|O\rangle = 0$
- CCSD lambda equation

$$\langle O|[(1 + L_1 + L_2)[(H_1 + H_2)\exp(T_1 + T_2)]_C]_L|D\rangle = 0$$

- CCSD dipole moments

$$\langle O|(1 + L_1 + L_2)[(H_1)\exp(T_1 + T_2)]_C|O\rangle = \langle H_1\rangle$$

How TCE works —— formula derivation



How TCE works —— program synthesis

Input WFT working equations

↓

Strength reduction: X=ABCD → X=((BC)A)D

Factorization: X=AB+AC → X=A(B+C)

Intermediate reuse: X=AB+C(AB) → Y=AB, X=Y+CY

↓

Operation Tree

↓

Determine permutation symmetry of intermediates

Analyze the use of permutation symmetry in contraction

Generate Fortran77/Fortran90 programs

How TCE-generated programs work —— challenges

- Use of spin & spatial symmetry in tensor storage & contraction
- Use of permutation symmetry in tensor storage & contraction
- Permutation symmetry of intermediates
- Abstraction of theories
- Memory management
- Scalablility to massively parallel regime

How TCE-generated programs work —— permutation symmetry

- Reduced storage size

$$t_{ij}^{ab} \Rightarrow t_{i<j}^{a<b}$$

- Reduced operation cost

$$\sum_{cd} t_{ij}^{cd} v_{cd}^{ab} \Rightarrow 2 \sum_{c<d} t_{i<j}^{c<d} v_{c<d}^{a<b}$$

How TCE-generated programs work —— intermediates

- With no de-excitation (*L*) operator

$$i^{g_1<g_2<g_3<\cdots<g_n,\ell_1<\ell_2<\ell_3<\cdots<\ell_p}_{g'_1<g'_2<g'_3<\cdots<g'_m,\ell'_1<\ell'_2<\ell'_3<\cdots<\ell'_q}$$

- With a de-excitation (*L*) operator

$$i^{g_1<g_2<g_3<\cdots<g_n,h_1<h_2<h_3<\cdots<h_p,p_1<p_2<p_3<\cdots<p_p}_{g'_1<g'_2<g'_3<\cdots<g'_m,h'_1<h'_2<h'_3<\cdots<h'_q,p'_1<p'_2<p'_3<\cdots<p'_q}$$

How TCE-generated programs work —— spin & spatial symmetries

- Spin symmetry

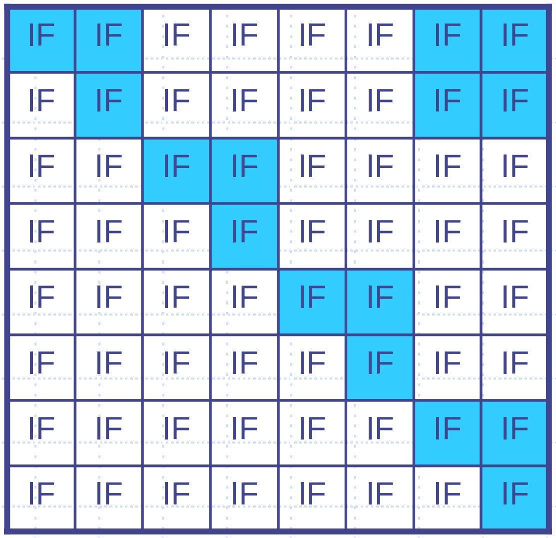
$$\underset{\text{Covariant indices}}{\sum_{p}} s_p = \underset{\text{Contravariant indices}}{\sum_{q}} s_q$$

- Spatial symmetry

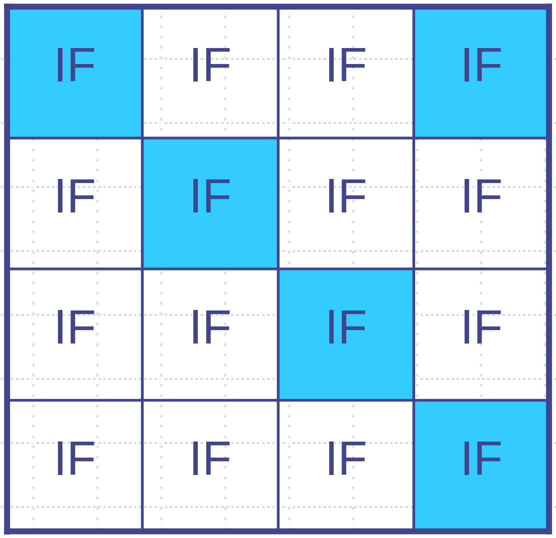
$$\underset{\text{Iindices}}{\prod_{p}} \Gamma_p = \Gamma_0$$

How TCE-generated programs work —— tiling algorithm

```
DO i
 DO j
  IF (i>j)
  IF X(ij) IS NONZERO
   ARITHMETIC
```

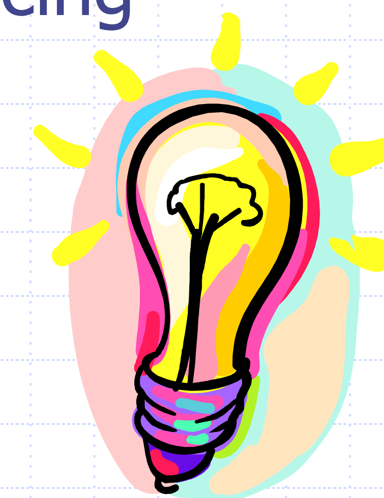



```
DO I (i1~i2)
 DO J (j1~j2)
  IF (I>J)
  IF X(IJ) IS NONZERO
  DO i
   DO j
    ARITHMETIC
```



How TCE-generated programs work —— tiling algorithm

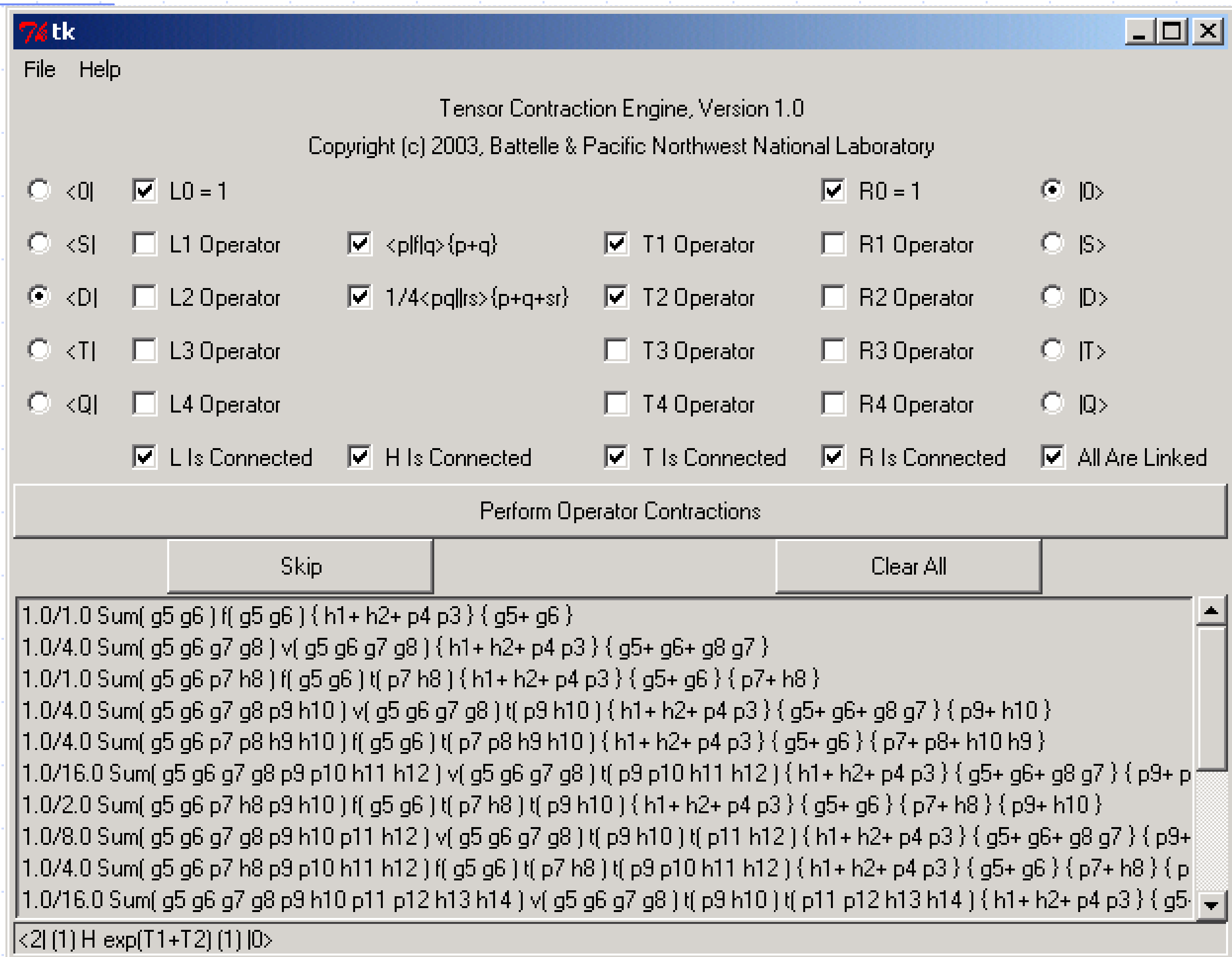
- Spin, spatial, and permutation symmetries
- Near-operation-minimum
- Runtime adjustment of peak memory size
- Granularity of dynamic load balancing parallelism



How TCE can be used — preliminary

- Use a Windows or LINUX machine
- Download the latest Python® from www.python.org (free)
  - TCE codes (aee.py, ccc.py, oce.py, tce.py) will run
- Obtain the latest NWChem development version for interfacing and running the TCE-generated codes
  - Minor modification of TCE's code generator & development of interface will permit other QC software suites than NWChem

How TCE can be used —— GUI based



How TCE can be used —— text based

cf) Auer's presentation

Shortcut to python.exe

```
Python 2.2.1 (#34, Apr  9 2002, 19:34:33) [MSC 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import oce
>>> ccd_t2 = oce.readfromfile("ccd_t2.in")
>>> print ccd_t2

[ + 0.25 ] * Sum ( g5 g6 g7 g8 ) * v ( g5 g6 g7 g8 ) * <0| ( h1+ h2+ p4 p3 ) ( g5+ g6+ g8 g7 ) |0>
[ + 0.25 ] * Sum ( g5 g6 p7 p8 h9 h10 ) * f ( g5 g6 ) * t ( p7 p8 h9 h10 ) * <0| ( h1+ h2+ p4 p3 ) ( g5+ g6 )
( p7+ p8+ h10 h9 ) |0>
[ + 0.0625 ] * Sum ( g5 g6 g7 g8 p9 p10 h11 h12 ) * v ( g5 g6 g7 g8 ) * t ( p9 p10 h11 h12 ) * <0| ( h1+ h2+ p
4 p3 ) ( g5+ g6+ g8 g7 ) ( p9+ p10+ h12 h11 ) |0>
[ + 0.0078125 ] * Sum ( g5 g6 g7 g8 p9 p10 h11 h12 p13 p14 h15 h16 ) * v ( g5 g6 g7 g8 ) * t ( p9 p10 h11 h12
) * t ( p13 p14 h15 h16 ) * <0| ( h1+ h2+ p4 p3 ) ( g5+ g6+ g8 g7 ) ( p9+ p10+ h12 h11 ) ( p13+ p14+ h16 h15 )
 |0>

>>> ccd_t2 = ccd_t2.performfullcontraction()
[ + 0.25 ] * Sum ( g5 g6 g7 g8 ) * v ( g5 g6 g7 g8 ) * <0| ( h1+ h2+ p4 p3 ) ( g5+ g6+ g8 g7 ) |0>
 ... commencing full operator contraction
 ... iteration = 1, number of terms = 1
 ... iteration = 2, number of terms = 1
 ... iteration = 3, number of terms = 1
 ... iteration = 4, number of terms = 1
[ + 0.25 ] * Sum ( h9 h10 p7 p8 g5 g6 ) * f ( g5 g6 ) * t ( p7 p8 h9 h10 ) * <0| ( h1+ h2+ p4 p3 ) ( g5+ g6 )
( p7+ p8+ h10 h9 ) |0>
 ... commencing full operator contraction
 ... iteration = 1, number of terms = 2
 ... iteration = 2, number of terms = 3
 ... iteration = 3, number of terms = 6
 ... iteration = 4, number of terms = 7
 ... iteration = 5, number of terms = 4
[ + 0.0625 ] * Sum ( h7 h8 p5 p6 g9 g10 g11 g12 ) * t ( p5 p6 h7 h8 ) * v ( g9 g10 g11 g12 ) * <0| ( h1+ h2+ p
4 p3 ) ( g9+ g10+ g12 g11 ) ( p5+ p6+ h8 h7 ) |0>
 ... commencing full operator contraction
 ... iteration = 1, number of terms = 2
 ... iteration = 2, number of terms = 4
 ... iteration = 3, number of terms = 8
 ... iteration = 4, number of terms = 14
 ... iteration = 5, number of terms = 10
 ... iteration = 6, number of terms = 6
[ + 0.0078125 ] * Sum ( h7 h8 h11 h12 p5 p6 p9 p10 g13 g14 g15 g16 ) * t ( p5 p6 h7 h8 ) * t ( p9 p10 h11 h12
) * v ( g13 g14 g15 g16 ) * <0| ( h1+ h2+ p4 p3 ) ( g13+ g14+ g16 g15 ) ( p5+ p6+ h8 h7 ) ( p9+ p10+ h12 h11 )
 |0>
 ... commencing full operator contraction
 ... iteration = 1, number of terms = 3
 ... iteration = 2, number of terms = 6
 ... iteration = 3, number of terms = 12
 ... iteration = 4, number of terms = 23
 ... iteration = 5, number of terms = 28
 ... iteration = 6, number of terms = 12
 ... iteration = 7, number of terms = 16
 ... iteration = 8, number of terms = 8
>>> print ccd_t2

[ + 1.0 ] * v ( p3 p4 h1 h2 )
[ - 1.0 ] * Sum ( h5 ) * f ( h5 h1 ) * t ( p3 p4 h5 h2 )
[ + 1.0 ] * Sum ( h5 ) * f ( h5 h2 ) * t ( p3 p4 h5 h1 )
```

How TCE can be enhanced —— limitations of TCE

- Tested for ...
  - CC up to CCSDTQ, CI up to CISDTQ, ‘generalized’ MBPT up to MBPT(4), CC Λ up to CCSDTQ, EOM-CC (right & left) up to EOM-CCSDTQ
- Anything else will probably need some work
  - But far less than a full manual implementation would!
  - cf) Sadayappan’s presentation

How TCE can be enhanced —— collaborations

- Join our effort!
  - Partner with existing members (quantum chemists or computer scientists) to realize your new methods much more quickly
  - ex) **Professor Rodney Bartlett & Mr. Igor Schweigert** —— CC analytical derivatives
  - ex) **Professor Piotr Piecuch, Dr. Karol Kowalski, Professor Mark S. Gordon** —— Novel CC & EOM-CC methods
  - ex) **Professor Kimihiko Hirao, Dr. Takeshi Yanai** —— Relativistic correlation theories

Other sources of information

- Papers
  - G. Baumgartner *et al*. Proc. Supercomp. (2002) [Performance optimization aspects]
  - D. Cociorva *et al*. Lect. Notes Comput. Sci. **2228**, 237 (2001). [Performance optimization aspects]
  - S. Hirata, J.Phys.Chem.A **107**, 9887 (2003) [TCE specs, and CISDTQ, CCSDTQ, MBPT(4) applications]
  - S. Hirata *et al*., J.Chem.Phys. **120**, 3297 (2004) [DK3 relativistic CCSDTQ applications]
  - S. Hirata, submitted [EOM-CCSDTQ applications]
- Webpage
  - http://www.cis.ohio-state.edu/~gb/TCE/
- TCE Manual

Acknowledgements

- U.S. Department of Energy
- National Science Foundation

- Professor Rodney J. Bartlett
- Dr. Wibe A. de Jong
- Dr. Michel Dupuis
- Professor Kimihiko Hirao
- Dr. Takahito Nakajima
- Dr. Jarek Nieplocha
- Dr. Theresa L. Windus
- Dr. Takeshi Yanai