The efficient implementation of the multi-reference perturbation theories at second order

Alex A. Granovsky

Laboratory of Chemical Cybernetics, M.V. Lomonosov Moscow State University, Moscow, Russia

September 21, 2005

Canonical single-reference MP

- MP2:

$$E_{mp2} = \frac{1}{4} \sum_{ijab} \frac{|< ij || ab >|^2}{\Delta_{ijab}}$$

  - Parameters: N, N_{occ}, N_{vir}
  - Integral transformation - N^5 step
  - Only minor overhead due to PT power series summation itself (N^4 step)

- MP3 and above:
  - Integral transformation - N^5 step
  - Intermediate quantities (amplitudes entering into numerators of the individual terms of the PT series) calculations - N^6 and above
  - As in the case of MP2, PT summation itself has better scaling (e.g., N^4 for MP3)

Multi-reference (MR) MBPT theories

- Additional parameters:
  - N_{act}, N_{det} (N_{csf}), N_{eff}
- More complex expressions both for energy correction itself and for computational costs
  - Third and higher orders of various formulations of the multi-reference (MR) MBPT
    - Calculation of various intermediates is the most computationally-demanding stage
  - Non-contracted and partially contracted MR-MBPT theories at second order
    - Most of the computational efforts are typically due to summation of the individual terms of the PT series themselves, especially for the case of large active spaces

Horrible MCQDPT2 example (H. Nakano, 1993)

Ordering the generators in Eq. (34) to normal products with only active orbital labels, we obtain

$$
\begin{aligned}
\langle\alpha|\mathscr{H}_{\mathrm{eff}}^{(2)}|\beta\rangle = \frac{1}{2}\sum_{AB} C_{A\alpha}C_{B\beta}\Bigg[-\delta_{AB}\Bigg(& \sum_{ia'} \frac{2u_{ia'}u_{a'i}}{\epsilon_{a'}-\epsilon_i+\Delta E_{B\beta}} + \sum_{ija'b'} \frac{(ia'|jb')[2(a'i|b'j)-(a'j|b'i)]}{\epsilon_{a'}-\epsilon_i+\epsilon_{b'}-\epsilon_j+\Delta E_{B\beta}} \\
& + \sum_{pq}\langle A|E_{pq}|B\rangle\Bigg[\sum_i \frac{u_{iq}u_{pi}}{\epsilon_p-\epsilon_i+\Delta E_{B\beta}} - \sum_e \frac{u_{pe}u_{eq}}{\epsilon_e-\epsilon_q+\Delta E_{B\beta}} - \sum_{ia'} \frac{u_{ia'}[2(a'i|pq)-(a'q|pi)]}{\epsilon_{a'}-\epsilon_i+\epsilon_p-\epsilon_q+\Delta E_{B\beta}} \\
& - \sum_{ia'} \frac{[2(ia'|pq)-(iq|pa')]u_{a'i}}{\epsilon_{a'}-\epsilon_i+\Delta E_{B\beta}} + \sum_{ija'} \frac{(ja'|iq)[2(a'j|pi)-(a'i|pj)]}{\epsilon_{a'}-\epsilon_j+\epsilon_p-\epsilon_i+\Delta E_{B\beta}} \\
& - \sum_{ia'b'} \frac{(ia'|pb')[2(a'i|b'q)-(a'q|b'i)]}{\epsilon_{a'}-\epsilon_i+\epsilon_{b'}-\epsilon_q+\Delta E_{B\beta}}\Bigg) + \sum_{pqrs}\langle A|E_{pq,rs}|B\rangle\Bigg(\sum_i \frac{u_{iq}(pi|rs)}{\epsilon_p-\epsilon_i+\epsilon_r-\epsilon_s+\Delta E_{B\beta}} \\
& - \sum_e \frac{u_{pe}(eq|rs)}{\epsilon_e-\epsilon_q+\epsilon_r-\epsilon_s+\Delta E_{B\beta}} + \sum_i \frac{(iq|rs)u_{pi}}{\epsilon_p-\epsilon_i+\Delta E_{B\beta}} - \sum_e \frac{(pe|rs)u_{eq}}{\epsilon_e-\epsilon_q+\Delta E_{B\beta}} \\
& - \frac{1}{2}\sum_{ij} \frac{(iq|js)(pi|rj)}{\epsilon_p-\epsilon_i+\epsilon_r-\epsilon_j+\Delta E_{B\beta}} - \frac{1}{2}\sum_{a'e} \frac{(pa'|re)(a'q|es)}{\epsilon_{a'}-\epsilon_q+\epsilon_e-\epsilon_s+\Delta E_{B\beta}} - \frac{1}{2}\sum_{ae} \frac{(pe|ra)(eq|as)}{\epsilon_e-\epsilon_q+\epsilon_a-\epsilon_s+\Delta E_{B\beta}} \\
& + \sum_{ia'} \frac{(pa'|iq)(a'i|rs)}{\epsilon_{a'}-\epsilon_i+\epsilon_r-\epsilon_s+\Delta E_{B\beta}} + \sum_{ia'} \frac{(pa'|is)(a'q|ri)}{\epsilon_{a'}-\epsilon_q+\epsilon_r-\epsilon_i+\Delta E_{B\beta}}
\end{aligned}
$$

Why the costs of PT summation are important?

- Straightforward implementation of the summation of the PT series is very inefficient on modern computer architectures because:
  - *At least one (or more) slow and typically not pipelined divide operation* is required to calculate each individual term of the PT series
  - Summation runs over large amount of data involving some combinations of transformed two-electron integrals, so that it is typically *not processor cache-friendly*

Our goals

- Address both these problems:
  - **1.** Reformulate the rules of the summation of the PT series to completely eliminate the slow divide operations by
    - **A.** Removing redundant divides by replacing most of the work to be done by the fast matrix multiplications of some intermediate quantities
    - **B.** Removing non-redundant divides by replacing them by few fast addition and multiplication operations
  - **2.** At the same time, develop efficient families of *cache-friendly* algorithms by introducing the appropriate intermediates and restructuring the order of loops used for summation of the PT series

The source of divides

- Separate calculation of the contribution due to each separate term of PT
  - Myriad of terms - myriad of divides
- Normally, we do not need to know the value of each separate term, only their sum of some kind
  - Way to reduce the number of divide operations

Redundant divides

- Number of different numerators is greater than number of different denominators
- A simple example:

$$S = \sum_{ij} \frac{a_{ij}}{b_i} = \sum_i \frac{\sum_j a_{ij}}{b_i}$$

- More realistic example (MCQDPT2):

$$\sum_{Bpq} < A | E_{pq} | B > \sum_i \frac{u_{iq} u_{pi}}{\varepsilon_p - \varepsilon_i + \Delta E_{B\beta}}$$

1A. Redundant divides removal

$$\sum_{Bpq} < A | E_{pq} | B > \sum_i \frac{u_{iq} u_{pi}}{\varepsilon_p - \varepsilon_i + \Delta E_{B\beta}} =$$

$$\sum_{Bip} \sum_q \frac{u_{iq} u_{pi} < A | E_{pq} | B >}{\varepsilon_p - \varepsilon_i + \Delta E_{B\beta}} =$$

$$\sum_{Bip} \frac{u_{pi} \sum_q u_{iq} < A | E_{pq} | B >}{\varepsilon_p - \varepsilon_i + \Delta E_{B\beta}} =$$

$$\sum_{Bip} \frac{u_{pi} v_{ABpi}}{\varepsilon_p - \varepsilon_i + \Delta E_{B\beta}} ; v_{ABpi} = \sum_q u_{iq} < A | E_{pq} B >$$

1B. Non-redundant divides removal

$$\sum_{i}^{n} \frac{a_i}{b_i} = \frac{a_1}{b_1} + \frac{a_2}{b_2} + \ldots + \frac{a_n}{b_n}$$

$$\frac{a_1}{b_1} + \frac{a_2}{b_2} = \frac{a_1 b_2 + a_2 b_1}{b_1 b_2} \qquad \frac{a_{i-1}}{b_{i-1}} + \frac{a_i}{b_i} = \frac{a_{i-1} b_i + a_i b_{i-1}}{b_{i-1} b_i}$$

Let us define:

$$A_0 = 0;\ B_0 = 1;\ A_i = A_{i-1} b_i + B_{i-1} a_i; \quad B_i = B_{i-1} b_i$$

Then:

$$\sum_{i}^{n} \frac{a_i}{b_i} = \frac{A_n}{B_n}$$

3 multiplications for 1 divide

2. Cache-friendly approach and loops restructuring

Non cache-friendly code sample

$$S = \sum_{B} C_{B\alpha} C_{B\beta} \sum_{ijab} \frac{(ia \mid jb)[2(ia \mid jb) - (ja \mid ib)]}{\varepsilon_a + \varepsilon_b - \varepsilon_i - \varepsilon_j + \Delta E_{B\beta}}$$

- Loop over B
  - Loop over i
    - Loop over j
      - Loop over a
        - Sum over b: $$T = T + \sum_{b} \frac{(ia \mid jb)[2(ia \mid jb) - (ja \mid ib)]}{\varepsilon_a + \varepsilon_b - \varepsilon_i - \varepsilon_j + \Delta E_{B\beta}}$$
      - End loop over a
    - End loop over j
  - End loop over i
  - Accumulate S
- End loop over B

Code sample analysis

- Why this code sample is bad?
  - There is no data reuse in the inner loops
- Our data:
  - huge number of 2-e integrals;
  - <u>very small number of orbital energies</u>
- How to improve the code
  - Restructuring the code in order to allow data reuse

Cache-friendly code version

$$S = \sum_B C_{B\alpha} C_{B\beta} \sum_{ijab} \frac{(ia \mid jb)[2(ia \mid jb) - (ja \mid ib)]}{\varepsilon_a + \varepsilon_b - \varepsilon_i - \varepsilon_j + \Delta E_{B\beta}}$$

- Loop over i
  - Loop over j
    - Loop over a
      - Calculate $t_b = (ia \mid jb)[2(ia \mid jb) - (ja \mid ib)]$
      - Loop over B
        - Calculate $\Delta = \varepsilon_a - \varepsilon_i - \varepsilon_j + \Delta E_{B\beta}$
        - Sum over b: $W = W + \sum_b \frac{t_b}{\varepsilon_b + \Delta}$
        - Accumulate
      - End loop over B
    - End loop over a
  - End loop over j
- End loop over i

Main results

- Up to 50-100 times faster MCQDPT2