SC14 Poster @ New Orleans, LA, USA 16-21, Nov. 2014

New Parallelization Model of Sequential Monte Carlo Analysis with Prediction-Correction Computing

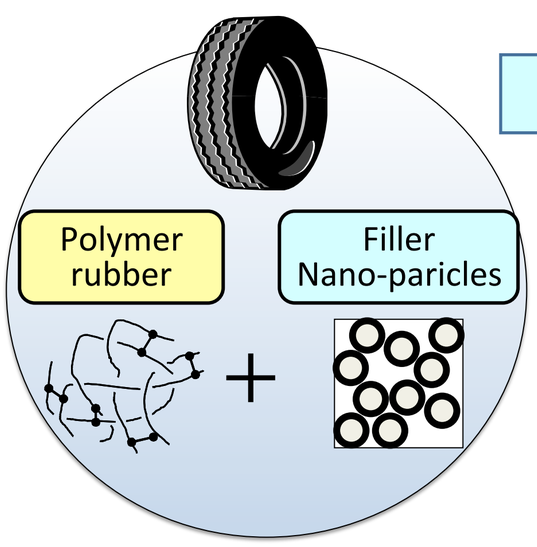
Eiji Tomiyama*, Hiroshi Koyama*, and Katsumi Hagita†

**Research Organization for Information Science and Technology (RIST)*

†National Defense Academy of JAPAN

Inverse solving by sequential Monte Carlo analysis

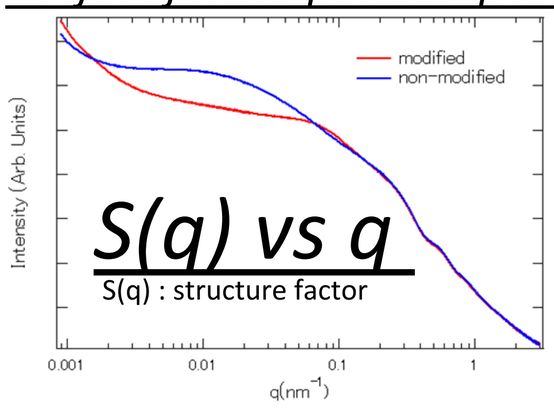
Building 3d model (particle positions) from X-ray scattering data.



SPring-8 @ JAPAN

courtesy of RIKEN/JASRI

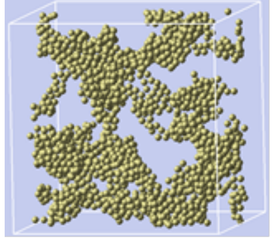
X-ray scattering data (S(q))
= info. of nano-partiles' positions

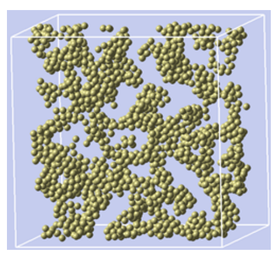


MC Search

$$g(r) = \frac{h(r)}{4\pi r^2 \rho}$$

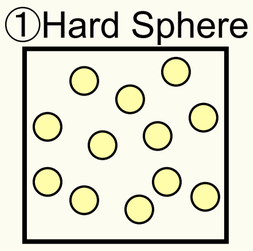
$$S(q) = 4\pi\rho \int_0^\infty dr r^2 (g(r) - 1) \frac{\sin(qr)}{qr} + 1$$



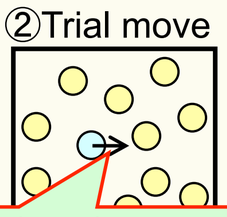


Algorithm of Conventional Reverse Monte Carlo method

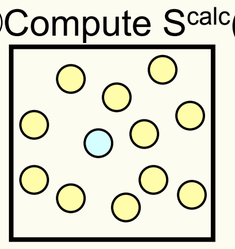
①Hard Sphere



②Trial move



③Compute $S^{calc}(q)$



④Compute error sum of square$\Delta\chi^2$

$$\chi^2 = \frac{\sum_q \left(S^{exp}(q) - S^{calc}(q)\right)^2}{\sigma^2}$$

Repeating it, χ^2 can be minimized.

→ Best estimated positions

$\Delta\chi^2 < 0$ → Accept

$\Delta\chi^2 > 0$ → Accept with the probability $\exp(-\Delta\chi^2/2)$

① Give initial configuration

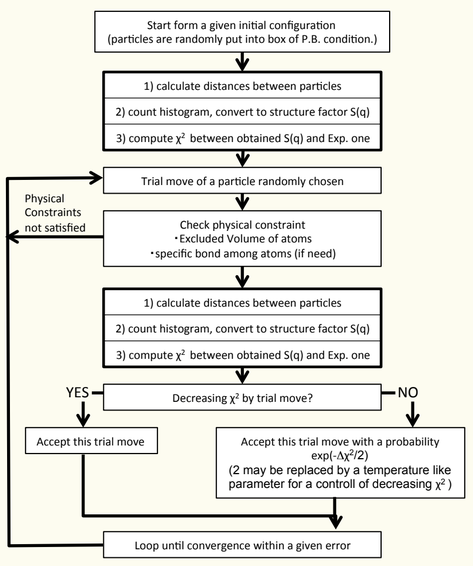
② Calculate S(q) and the change of χ^2 by trial move chosen satisfying with physical constrains

- χ^2 is a square sum of differences between S(q) obtained from configuration and Experimental data.

$$\Delta\chi^2 = \sum_q \frac{-2\left(S^{exp}(q) - S^{prev}(q)\right)\Delta S(q) + \left(\Delta S(q)\right)^2}{\sigma(q)^2}$$

③ Accept with the probability exp(-$\Delta\chi^2/2$)

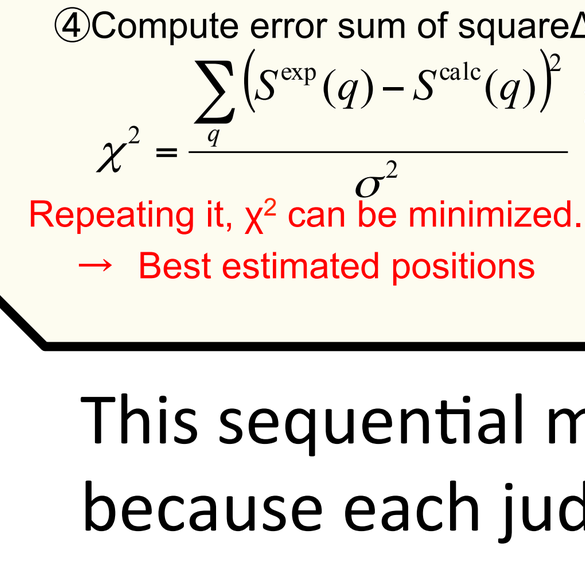
④ Until convergence

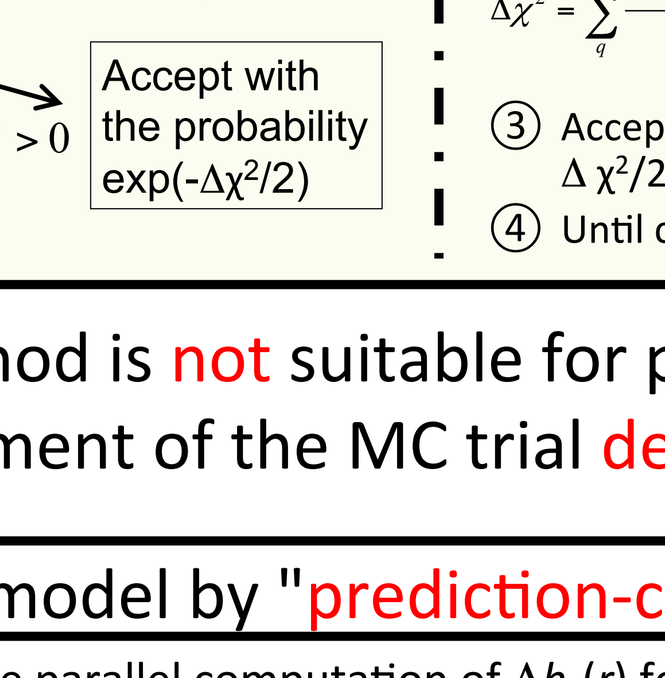


This sequential method is not suitable for parallelization because each judgement of the MC trial depends on its previous state.

New Parallelization model by "prediction-correction" method

Algorithm is changed to separate parallel computation of $\Delta h_k(r)$ for k = 1 to M and unparalleled correction part.
Communication overhead can be reduced to 1/M of that of "sequential" method.





"prediction" part calculates $\Delta h_k(r)$ M times simultaneously, where each $\Delta h_k(r)$ consists of an independent partcle move.

$\Delta h_1(r)$
$\Delta h_2(r)$ except for i_1-i_2
$\Delta h_3(r)$ ※i_1-i_2, i_1-i_3, i_2-i_3

Here, i_k is particle id of sampled as k-th trial.

"correction" part fixes $\Delta h_k(r)$, which were tentatively calculated for pairs of M trial particles in the prediction part.

$\Delta h_2(r)$ for i_1-i_2
$\Delta h_3(r)$ for i_1-i_2, i_1-i_3, i_2-i_3

For the efficiency, tentative $\Delta h_2(r)$ for i_1-i_2 is calculated in the prediction part. The tentative value is subtracted in the correction part.

Acknowledgement

Authors thank Drs. T. Tominaga and T. Sone of JSR Corporation. (Data obtained at SPring-8 and HPCI project hp140239.) We also thank Dr. T. Honda and Mr. A. Takayanagi of ZEON Corporation. (HPCI project hp140238). KH is supported by "Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures" and “High performance Computing Infrastructure” in Japan. (Project ID: jh140004-NA02, hp130062, hp140191)

Performance analysis and optimization

Optimization for the histogram calculation kernel

```
do i=1+nid,n,nproc
  xtmp(:)=pos(:,i) + 3.0d0
!***     Loop1    ***
  do j=1,i-1
    dx(:)=(xtmp(:) - pos(:,j))*0.5d0
    dx(:)=2.0d0*(dx(:) - aint(dx(:)))-1.0d0
    r=ma*dx(1)*dx(1)+mb*dx(2)*dx(2)+ mc*dx(3)*dx(3)
     ilist(j)=sqrt(r)*drinv + 0.5d0
   enddo
!*** End of Loop1  ***
!***     Loop2   ***
  do ip=1,nthr
       ip_st = int((i-1)*dble(ip-1)/dble(nthr)) + 1
       ip_en = int((i-1)*dble(ip)/dble(nthr))
     do j=ip_st,ip_en
        phist(ilist(j),ip)=phist(ilist(j),ip) + 1.d0
      enddo
   enddo
!*** End of Loop2 ***
enddo

do ip=1,nthr
  do ir=1,nr
    mpihist(ir)=mpihist(ir)+phist(ir,ip)
  enddo
enddo
```

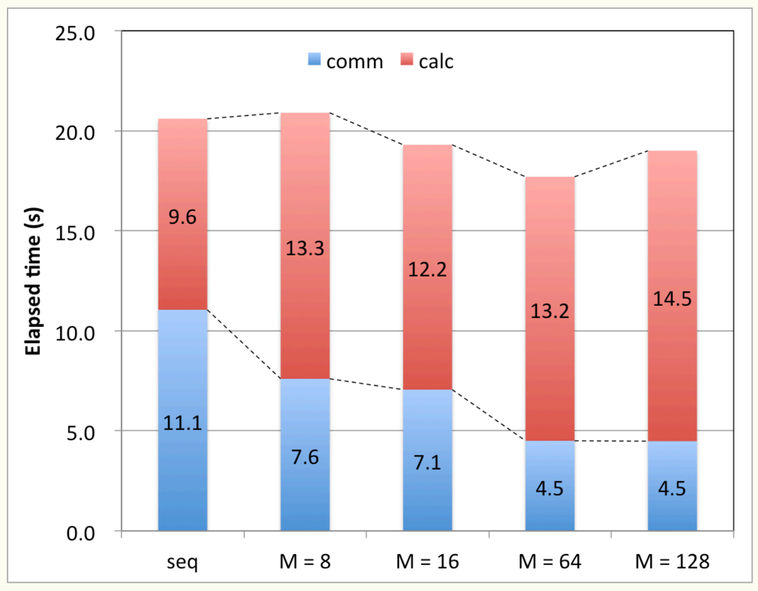
- We modified the original kernel loop for the single instruction multiple data (SIMD) hardware with fused multiply-add (FMA) support in terms of instruction-level parallelization.
- To prevent indirect memory access, we divided the kernel loop into two loops.
  (Loop1) distance computation for all particle pairs
  (Loop2) histogram reduction on each thread
- For Loop1, we achieved 51 % of theoretical peak.
- Total performance in both loops was 36.5 % where 75 % of the instructions were SIMD vectorized.

Estimation of Theoretical Limit of the Performance of Loop1.
Demand for memory: 16 bytes / Loop
Flop/Loop = 29 (including sqrt and aint functions.)
Byte/Flop = 0.55
Due to Limit (B/F=0.36) of K computer, 65 % is theoretical limit of Loop1.
Check of this limit with computation removing sqrt and aint.
Peak-ratio = 44.2 %. Flop/Loop = 26.01.
Validity of the estimated theoretical Limit is confirmed.

Performance of the "prediction-correction" computing

- comm time is reduced from 11.1 s to 4.5 s. It includes MPI_allreduce and MPI_bcast.
- In "prediction-correction" method, number of part of the MPI calls is reduced to 1/*M*.

- In 4,194,304-particle system, 24 nodes and *M* = 64 is sweet spot.The calculation time of $\Delta h(r)$ for *M* = 64 was 2.9 s (14 %) shorter than that of the "sequential" method.
- Our “prediction–correction” method is very useful for large-particle systems as well as large parallel computers.



Elapsed time of the $\Delta h(r)$ calculation on 24 nodes

Sequential	M=8	M=16	M=64	M=128
20.6 s	20.9 s	19.3 s	17.7 s	19.0 s

count of collective communication

MPI functions	Sequential	M=64
Allreduce	39,927	13,921
Bcast	153,236	14,337

We also benchmarked a 33,554,432-particle system on 256 nodes of the K computer; the elapsed time with *M* = 32 was 22.4 s, which is faster than that of the *M* = 64 run (24.7 s).

- For speed up of RMC computations, physical conditions such as excluded volume (EV) are checked before the $\Delta h(r)$ calculation.
- In this case, approximately 13 % of the 100,000 trails satisfied the EV conditions. Next, approximately 60% of the satisfied trials were accepted in terms of the χ^2 judgments.

- We achieved **36.5%** of peak performance on **“K computer”「京」** for the histogram calculation kernel.
  - 5.98 Tflops was achieved on 128 nodes with 4,194,304 particles.
  - 339 Tflops was achieved on 8,192 nodes with 33,554,432 particles.
- New algorithm "prediction-correction" is efficient.
  - The achieved speed up was 14% of the elapsed time.
  - Reduction of the MPI communications leads to reducing the elapsed time.