EFFICIENT TOMOGRAPHIC RECONSTRUCTION FOR COMMODITY PROCESSORS WITH LIMITED MEMORY BANDWIDTH

Hiroshi Inoue (inouehrs@jp.ibm.com) IBM Research – Tokyo (University of Tokyo)

Summary

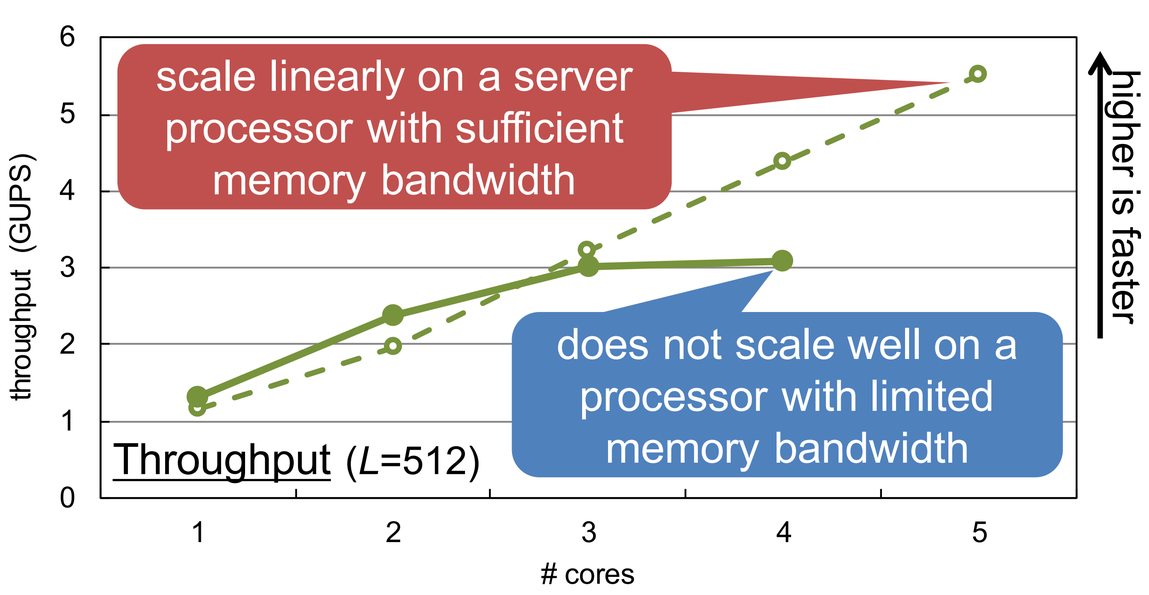
- Goal: to use commodity processors (e.g. for PC) for CT reconstruction without costly accelerators
- Challenge: commodity processors typically have limited system memory bandwidth

➔We developed a technique to reduce memory bandwidth requirement; we achieved up to 80% speedup in RabbitCT benchmark when memory bandwidth was not sufficient

1. Introduction

- Today's commodity processors are becoming more and more powerful in computation power with multiple cores and vector instructions
  - Even smartphones or tablets use quad- or octa-core processors
- However, memory systems are relatively weak in such commodity processors
- Question: Can we use (low-cost and low-power) the commodity processors in CT systems?

2. Workload Analysis



✓ Computation power cannot be fully utilized if memory bandwidth is not sufficient

Overview of FDK CT reconstruction algorithm

```
for each 2D projection image {
   for x = 0 to L-1 {
     for y = 0 to L-1 {
       for z = 0 to L-1 {
           project voxel (x,y,z) onto 2D projection image
           read value from 2D image at projected point
           update density value of voxel (x,y,z)
   }}}
}
```

- Each iteration of outer-loop accesses
  - ✓ read from **2D projection image** (< 10 MB)
  - ✓ read and write to **density values of voxels in 3D structure** (> 300 MB)

We need to reduce accesses to 3D volume data!

3. Optimization

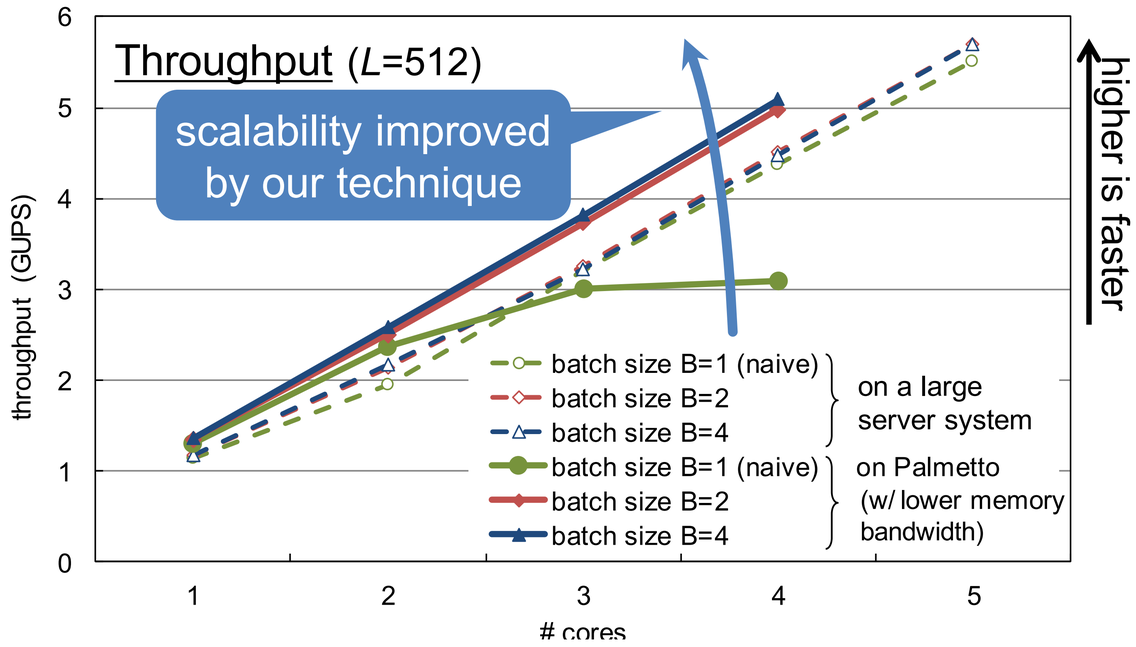
- Idea: to process multiple (***B***) projection images in each iteration → we need to read and write 3D volume data only once per ***B*** images and hence bandwidth requirement becomes **1/*B***

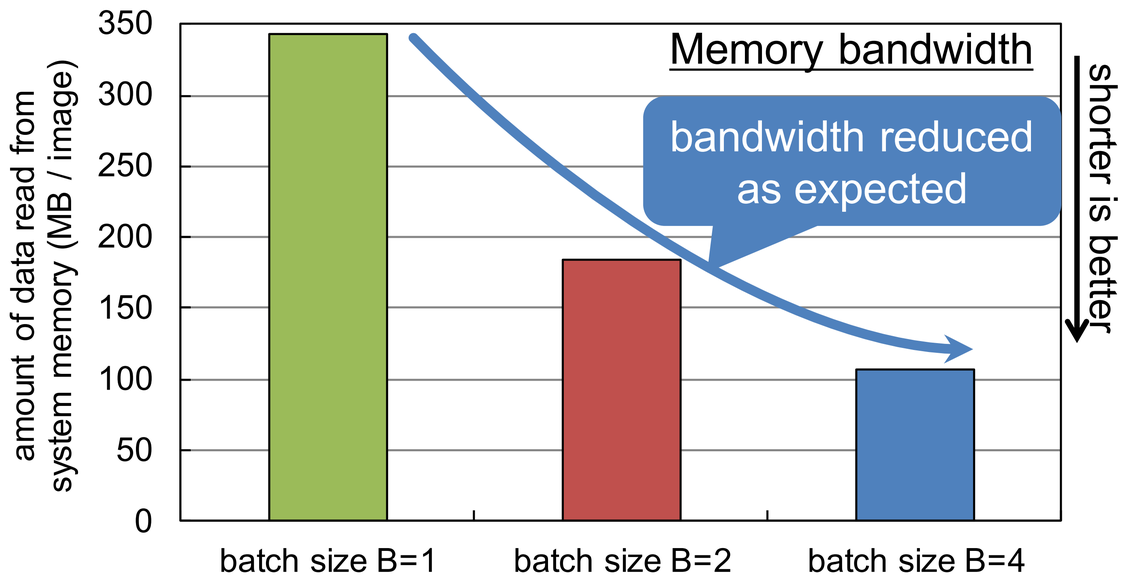
Overview of reconstruction with our technique

```
for each batch of B projection image {
   for x = 0 to L-1 {
     for y = 0 to L-1 {
       for z = 0 to L-1 {
         for k = 0 to B-1 {
           project voxel (x,y,z) onto k-th image of batch
           read value from k-th image at projected point
           update density value of voxel (x,y,z)
         }
   }}}
} // number of iteration becomes 1/B
```

4. Performance results

- Evaluated our technique on IBM POWER8 using RabbitCT benchmark





Performance comparisons with previous RabbitCT scores

Category	Processor	# Core / # Boards	GUPS
Low mem. bandwidth	**POWER8 4.32 GHz**	4 cores (1 socket)	5.1
Server-grade processor	**POWER8 3.69 GHz**	20 cores (2 sockets)	21.4
	POWER8 3.69 GHz	10 cores (1 socket)	10.8
	IvyBridge-EP 2.2 GHz	20 cores (2 sockets)	about 7.0
	Westmere-EX 2.4 GHz	40 cores (4 sockets)	8.3
Accelerator	Xeon Phi 5110P	1 board	about 8.5
	nVidia GTX 670	2 boards	152.9