CS528
High Performance Computing

Roofline Model

A Sahu
Dept of CSE, IIT Guwahati

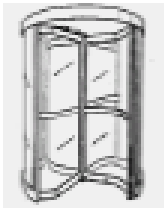
Outline

- Data Access Optimization
  - Roofline Model
  - Caching optimization
  - App classification based DA: N/N, N^2/N^2, N^3/N^2
- ***[Ref: Hager Book, PDF uploaded to MS Team]***

Data Access Optimization

Performance of System: Modeling Customer Dispatch in a Bank

Resolving door Throughput:
b_s[customer/sec]



Processing Capabilty:
P_{peak} [task/sec]

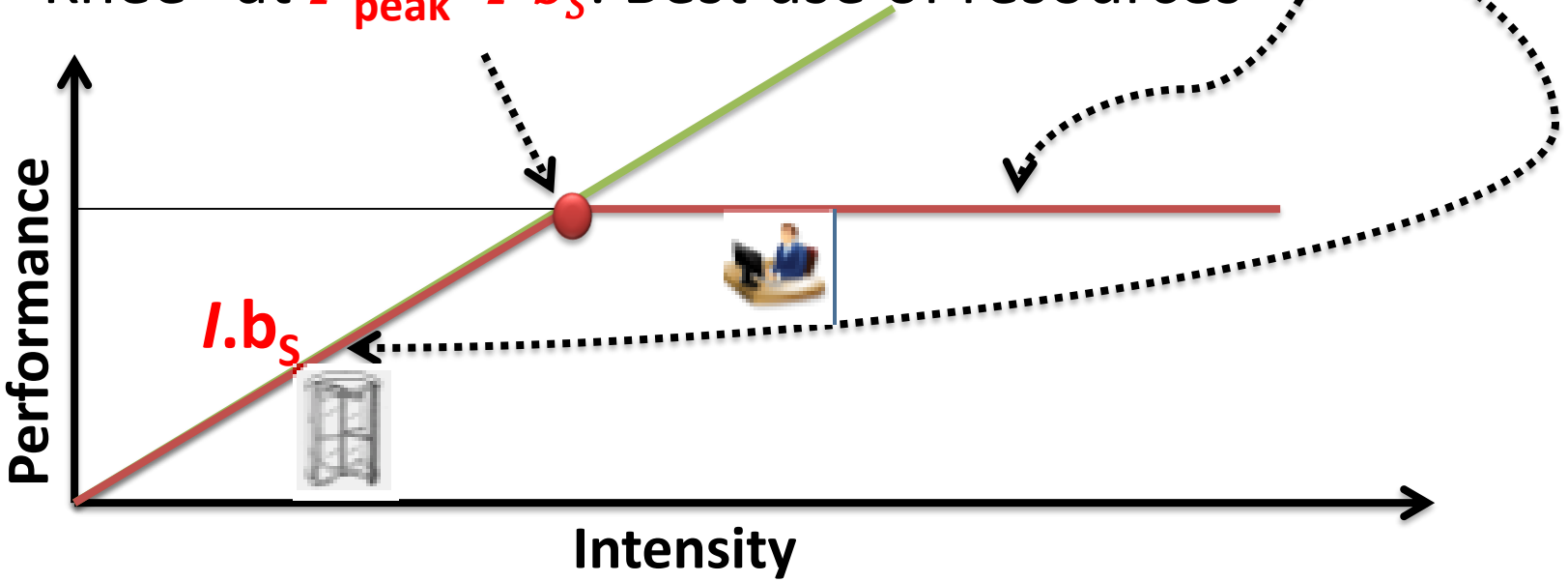
Intensity:
I [task/customer]

Modeling Customer Dispatch in a Bank

- How fast can tasks be processed? P[tasks/sec]
- The bottleneck is either
  - The service desks (peak. tasks/sec): P_{peak}
  - The revolving door (max. customers/sec): $I \cdot b_S$
- Performance $P=\min(P_{peak}, I \cdot b_S)$
- **This is the "Roofline Model"**
  - High intensity: P limited by "execution"
  - Low intensity: P limited by "bottleneck"

Modeling Customer Dispatch in a Bank

- Performance P=min(P_{peak}, $I \cdot b_S$)
- **This is the "Roofline Model"**
  - High intensity: P limited by "execution"
  - Low intensity: P limited by "bottleneck"
  - "Knee" at P_{peak}=$I \cdot b_S$: Best use of resources



- Roofline is an "optimistic" model

The Roofline Model

- P_{max}= Peak performance of the machine
- I= Computational intensity ("work" per byte transferred) over the slowest data path utilized ("the bottleneck")
- b_S= Applicable peak bandwidth of the slowest data path utilized

Expected performance:

$$P = \min(P_{peak}, I \cdot b_S)$$

[F/B] [B/s]

Apply Roof line to Machine and Code

- Machine Parameter 1 : P_{peak} [F/s]=4 G F/S
- Machine Parameter 2 : b_S [B/s] =10 G B/s
- Application Properties: I [F/B] = 2F/8B=0.25F/B

```
for(i=0;i<N;i++) s=s+a[i]*a[i]; // double s, a[]
```

- Performance = P = min(P_{peak}, I*b_s)

  =min(4 GF/s, 0.25 F/B *10 G.B/s)

  =min(4 GF/s, 2.5 GF/s)

  =2.5 G F/s

The Refine Roofline Model

- P_{max}= Peak performance of a loop assuming that data comes from L1 cache **(is not necessarily P_{peak})**
- I= Computational intensity ("work" per byte transferred) over the slowest data path utilized ("the bottleneck"),
- **Code Balance $B_c = I^{-1}$= in Byte per Flop**
- b_S= Applicable peak bandwidth of the slowest data path utilized

Expected performance:

[F/B] [B/s]

$$P = \min(P_{max}, I \cdot b_S)$$

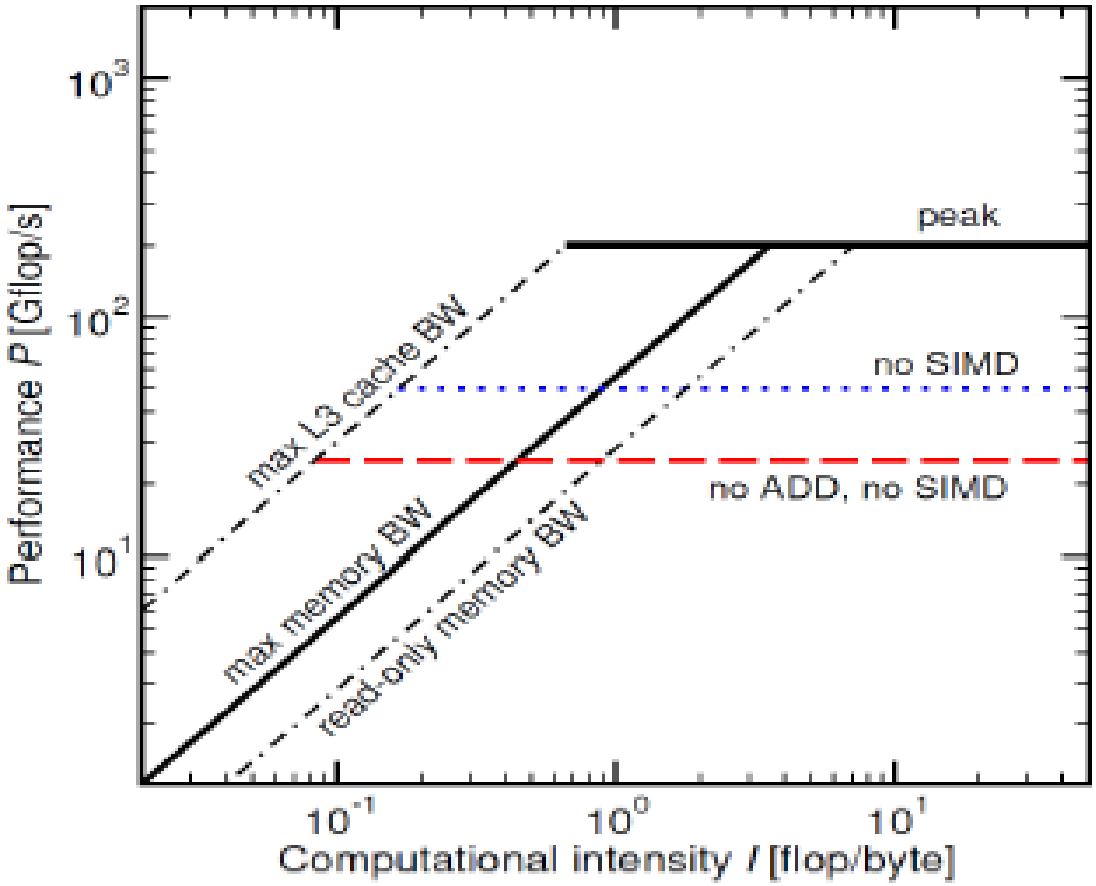
$$P = \min(P_{max}, b_S/B_c)$$

Factors to consider in Roofline Model

- BW Bound : may be simple
  - Accurate traffic calculation (write allocate, stride,..)
  - Practical not equal to theoretical BW limits
  - Saturation effects -> consider full socket only
- Core Bound : may be complex
  - Multiple bottlenecks: LD/ST, arithmetic, pipeline, SIMD, execution port
  - Limit is linear in # of cores

Refine RFL model: Graphical Representation

- Multiple ceiling may apply
  - Diff BW/Data paths -> Diff inclined ceilings
  - Different P_{max}-> Diff flat ceilings



P_{max} comes from code analysis : with/without SIMD, add other FUs

Apply Roof line to Haswell Core to Triad Code

- Achievable Max Performance 1: P_{max} [F/s]=12.27 G F/S
- Machine Parameter 2 : b_S [B/s] =50 G B/s
- Application Properties: I [F/B] = 2F/40B=0.05F/B

for(i=0;i<N;i++) a[i]=b[i]+c[i]*d[i]; // double a,b,c,d

- Performance = P = min(P_{max}, I*b_s)

  =min(12.27 GF/s, 0.05 F/B *50 G.B/s)

  =min(12.27 GF/s, 2.5 GF/s)

  =2.5 G F/s

Code Balance/Intensity Examples

```
for(i=0;i<N;i++)//Copy
    a[i]=b[i];
```

B_c=24B/0F=NA

```
for(i=0;i<N;i++)//Scale
     a[i]=s*b[i];
```

B_c=24B/1F=24 B/F

```
for(i=0;i<N;i++) //Add
     a[i]= b[i]*c[i];
```

B_c=32B/1F=32 B/F

A[i]=B[i];//Require reading of A[i], B[i] and writing to A[i] incase of L1 cache access; so 2 read and one write

Code Balance/Intensity Examples

```
double a[N],b[N],C[N], d[N];
for(i=0;i<N;i++)
   a[i]=a[i]+b[i];
```

B_c=24B/1F=24 B/F
I=0.042F/B

```
for(i=0;i<N;i++) //Triad
     a[i]=a[i]+s*b[i];
```

B_c=24B/2F=12 B/F
I=0.083 F/B

```
for(i=0;i<N;i++) //float a[]
     s= s+a[i]*a[i];
```

B_c=4B/2F=2 B/F
I=0.5 F/B

```
for(i=0;i<N;i++)//float a[],b[]
     s= s+a[i]*b[i];
```

B_c=8B/2F=4 B/F
I=0.25 F/B

Memory Analysis of Simple Triad Code on Intel i7-5960X

i7-5960x Spec (8C-16T, 22nm,3.5Ghz, 20MB Smart Cache)

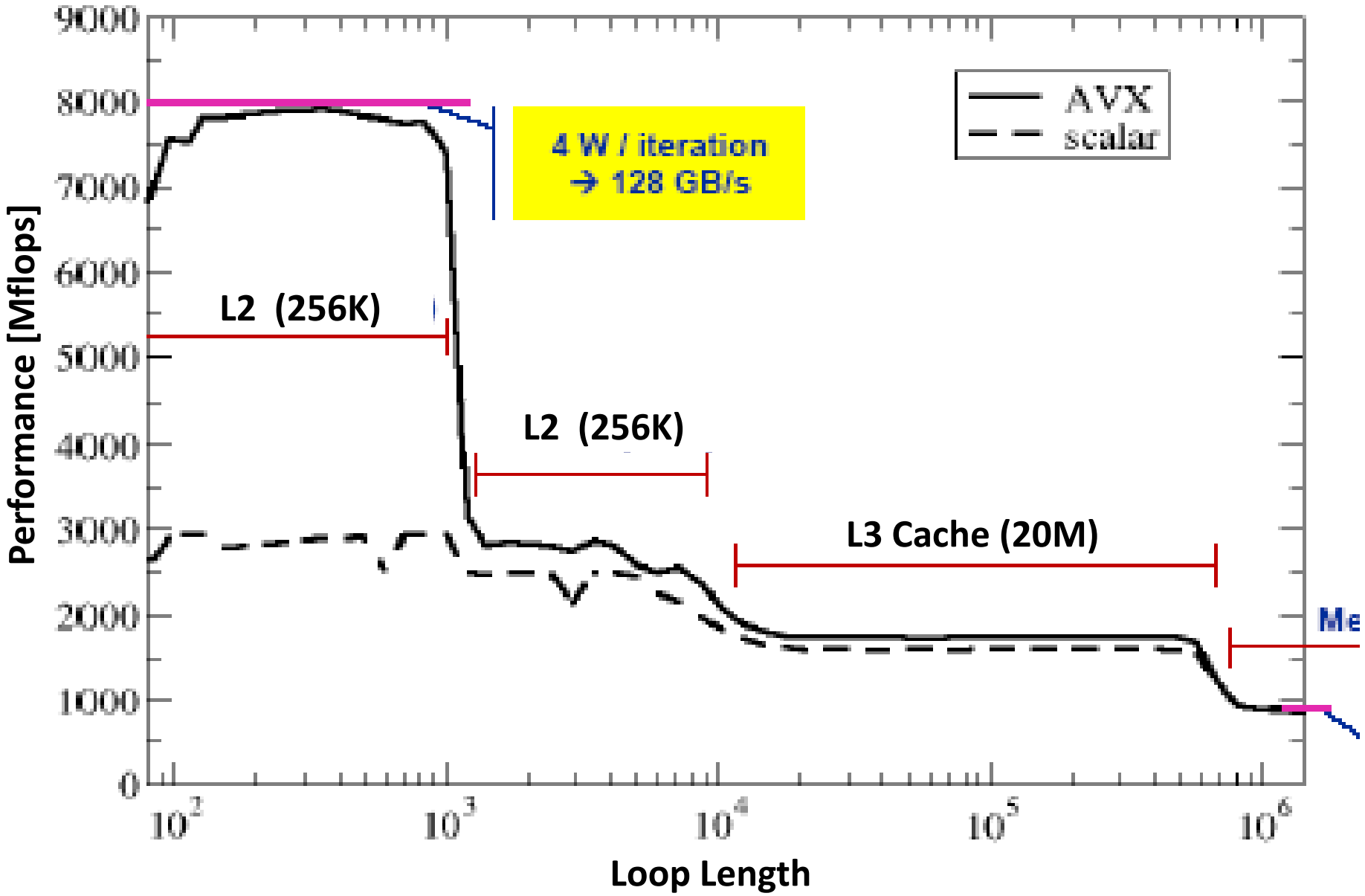
Memory Analysis of Simple Code

- Simple Streaming Benchmark
- *A "swiss army knife" for micro-benchmarking*
  - Report performance for different N
  - Choose NITER : Accurate time measurement is possible
- **This kernel is limited by data transfer performance for all memory levels on all current architectures!**

```
float   A[N],B[N],C[N],D[N];
for( j=0; j<NITER; j++) {
  for(i=0;i<N;i++)
        A[i]= B[i]+C[i]*D[i];
   if(i>N)   dummy(A,B,C,D);
 }
```

Prevents smarty-pants compilers from doing "clever" stuff

On Sandy Bridge: Core i7 5960 Xtreme

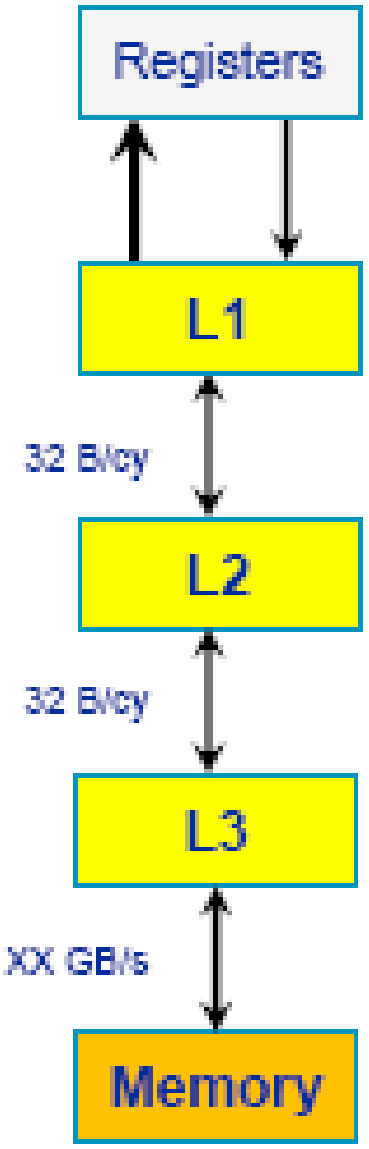


Memory Hierarchy

- Are the performance levels plausible?
- What about multiple cores?
- Do the bandwidths scale?

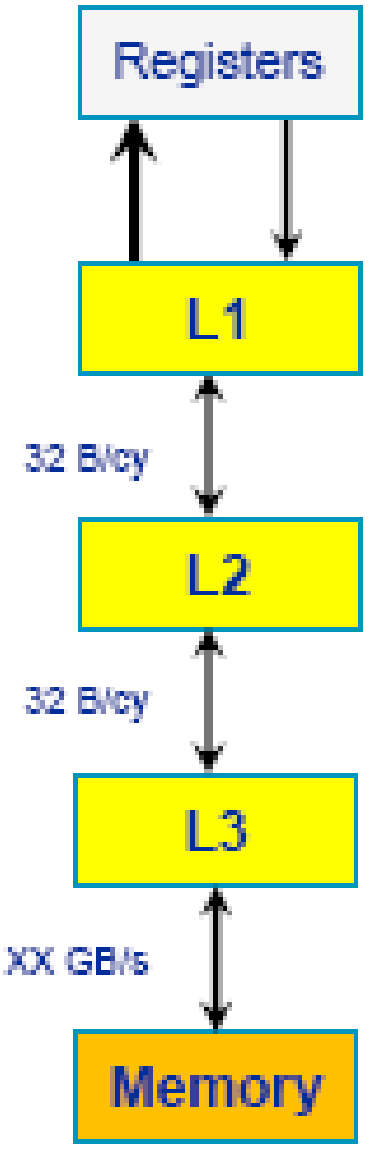
Throughput capabilities: i7 5960 Xtreme

- **Per cycle with AVX**
  - **1 load instruction (256 bits) AND ½ store instruction (128 bits)**
  - **1 AVX MULT and 1 AVX ADD instruction (4 DP / 8 SP flops each)**
  - **Overall maximum of 4 micro-ops**



Throughput capabilities: i7 5960 Xtreme

- Per cycle with SSE or scalar
  - 2 load instruction OR 1 load and 1 store instruction
  - 1 MULT and 1 ADD instruction
  - Overall maximum of 4 micro-ops
- Data transfer between cache levels
  - (L3 ↔ L2, L2 ↔ L1)
  - 256 bits per cycle, half-duplex (i.e., ful CL transfer == 2 cy)



On Sandy Bridge: Core i7 5960 Xtreme