Data Processing on Modern Hardware

Viktor Leis
What This Course is About

• make programs faster on modern multi-core CPUs using
  ▶ data-parallel instructions (SIMD)
  ▶ efficient synchronization of data structures
  ▶ parallelization on multi-core CPUs

• not part of this course
  ▶ number crunching with floating point numbers
  ▶ distributed programming
  ▶ GPU
  ▶ FPGA
  ▶ ...

Programming Assignments

- this course is about skills, not about abstract knowledge
- unless you do the assignments, this course is fairly pointless
- you will get a grade for the assignments $h$, which improves the grade of the exam $e$: $\min(e, 0.6e + 0.4h)$
- (more information later)
Hardware Trends

• in the past decades, single-threaded performance doubled every 18-22 months
• now single-threaded performance is stagnating (yearly single digit percentage increases)
• number of transistors is still growing quickly (“Moore’s law”)
• as a result, chips offer more and more parallelism (in particular on servers)
• to benefit from this parallelism, the software must generally be rewritten (“the free lunch is over”)
• software is becoming “a producer of performance” instead of a “consumer of performance” (Mark Hill)
Number of Cores
Single Instruction, Multiple Data (SIMD) Width

- 1997: MMX 64 bit (Pentium 1)
- 1999: SSE 128 bit (Pentium 3)
- 2011: AVX 256 bit float (Sandy Bridge)
- 2013: AVX2 256 bit int (Haswell)
- 2017: AVX-512 512 bit (Skylake Server)
Memory Bandwidth Per Core

- NetBurst (Foster)
- NetBurst (Paxville)
- Core (Kentsfield)
- Core (Lynnfield)
- Nehalem (Beckton)
- Nehalem (Westmere EX)
- Sandy Bridge EP
- Ivy Bridge EP
- Ivy Bridge EX
- Haswell EP
- Skylake SP
- Broadwell EX

DRAM bandwidth per core [GB/s]

year


0 5 10
Outlook: Dark Silicon

- heat dissipation is becoming a major problem
- soon it will be impossible to power all available transistors at the same time
- SIMD code already runs at lower frequencies than sequential code
- one possible solution: many cores at low frequency (e.g., Intel’s Many Integrated Core architecture: Xeon Phi)
- another possible solution: many specialized, heterogeneous cores and function units
Hardware Specialization

- special purpose instructions (e.g., AES encryption, video decoding)
- Graphics Processing Unit (GPU)
- Accelerated Processing Unit (APU, by AMD)
- Oracle Sparc T7, Data Analytics Accelerators (DAX): decompression
- Oracle RAPID (research project)
- Google Tensor Processing Unit (TPU)
- Field-Programmable Gate Array (FPGA): “software-defined hardware”
- Application-Specific Integrated Circuit (ASIC), e.g., for bitcoin mining
- historical: Gamma database machine, Lisp machine
Target Platform: Skylake Server

- server variant of Intel Skylake microarchitecture, up to 28 cores
- available since August 2017
- our model: Intel Core i9-7900X (10 cores, 20 hyperthreads)
Skylake Server Pipeline
## Skylake Server Caches and Memory

<table>
<thead>
<tr>
<th></th>
<th>L1 per core</th>
<th>L2 per core</th>
<th>L3 shared</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>size per core [KB]</td>
<td>32</td>
<td>1024</td>
<td>1408</td>
</tr>
<tr>
<td>latency [cycles]</td>
<td>4-6</td>
<td>14</td>
<td>50-70</td>
</tr>
<tr>
<td>max bandwidth [bytes/cycle]</td>
<td>192</td>
<td>64</td>
<td>16</td>
</tr>
<tr>
<td>sustained bandwidth [bytes/cycle]</td>
<td>133</td>
<td>52</td>
<td>15</td>
</tr>
<tr>
<td>associativity</td>
<td>8</td>
<td>16</td>
<td>-</td>
</tr>
</tbody>
</table>

- cache line size: 64 byte
- Skylake SP: 6 channels DDR4 2400 (nominal bandwidth: $6 \times 2400 \text{ MHz} \times 8 \text{ byte} = 112 \text{ GB/s}$)
- Skylake X: 4 channels DDR4 2100 channels (nominal bandwidth: $4 \times 2100 \text{ MHz} \times 8 \text{ byte} = 66 \text{ GB/s}$)
Simultaneous Multithreading (SMT) aka Hyperthreading

- goal: improve utilization of execution units
- each core is exposed as 2, 4, or 8 hardware threads
- threads running on the same core share most resources (e.g., L1 cache, execution units)
- fully transparent to software and OS (looks like “real” cores)
- allows hiding latencies (from cache misses, expensive instructions, etc.)
- one cannot expect linear speed up from this, but often gives moderate performance boost at very little hardware cost
- Intel: 2-way Hyperthreading