Architecture evolutions and its impact on legacy codes

29 novembre 2019
Guillaume Colin de Verdière
Architecture evolutions
EVOLUTION DRIVER: TECHNOLOGICAL CONSTRAINTS

- Power wall
- Scaling wall
- Memory wall
- Towards accelerated architectures
POWER WALL

\[ P = cV^2F \]

- Reduce V
  - Reuse of embedded technologies
- Limit frequencies
- More cores
- More compute, less logic
  - SIMD larger
  - GPU like structure

Parallelism and vectorization are mandatory

- **Moore’s law comes to an end**
  - Probable limit around 5 nm
  - Need to design new structure for transistors

- **Limit of circuit size**
  - Yield decrease with the increase of surface
    - Chiplets will dominate
  - Data movement will be the most expensive operation
    - 1 DFMA = 20 pJ, SRAM access= 50 pJ, DRAM access= 1 nJ (source NVIDIA)

**Programmers will focus on data location / movement**
- Recompute instead of storing
- Stay in Caches

$1 \text{nJ} = 1000 \text{pJ}, \quad \phi_{\text{Si}} = 110 \text{pm}$
Better bandwidth with HBM
- DDR5 @ 5200 MT/s 8ch = 0.33 TB/s
- HBM2 @ 4 stacks = 1.64 TB/s

Latencies don’t improve
- More hyperthreads

Deeper memory hierarchy
- Caches + HBM + DDR (+ NVM)

Impact of non volatile memories?

Programming will be impacted
- Importance of spatial and temporal localities
  - Caches will be ever more important
- Hyperthreading is unavoidable
Exaflop with regular CPUs will be too power hungry

One or more accelerator per node

Accelerator type will depend on applications
  - GPU
    - Compute and IA
  - FPGA
  - TPU
  - DSP
  - Neuromorphic
  - Quantum accelerator

Applications will have to (be) adapt(able)
Focus on data movements
CONCLUSION

Fundamental invariants exist

- Limit / control data movements (caches)
  - Energy & performance

- Algorithms that can vectorize (CPU and/or GPU)
  - Energy & performance

- Multi level parallelism
  - Inter node (macro), intra node (meso), vectorization (micro)

- Importance of data structures and algorithms
Impact on legacy codes
Definition: a legacy code (LC) is a code

- That is older than 10 years
  - Has seen at least two system generations
    - Sometimes more than 6!

- That is vital to the entity

- That needs to be ported to newer systems
  - With a “reasonable” performance
LEGACY CODE CHARACTERISTICS (1/2)

- Original team (often) not available anymore
  - Need to “educate” new generations of developers

- Large source code
  - \( \gg 1M \) SLOC
  - Intimate internal knowledge is potentially lost

- Needs Funding
  - to maintain the code
  - to port to the next generations of machines
LEGACY CODE CHARACTERISTICS (2/2)

- Use older practices
  - Data structures ("à la CRAY")
  - Standards (F77)

- Parallelism is somewhat limited
  - MPI only, seldom GPU versions or OpenMP

- Relies on “old” third party libraries
Technology puts pressure on LC

<table>
<thead>
<tr>
<th>Technology</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy savings</td>
<td>Focus on data movements</td>
</tr>
<tr>
<td>More cores &amp; hyperthreading</td>
<td>More in node parallelism (threads, tasks)</td>
</tr>
<tr>
<td>Wider SIMD</td>
<td>Vectorize algorithms</td>
</tr>
<tr>
<td>Deeper memory hierarchy</td>
<td>Focus on data movements</td>
</tr>
<tr>
<td>Accelerators</td>
<td>Focus on data movements Vectorize algorithms</td>
</tr>
</tbody>
</table>

Fast languages evolutions
- C++/14 – 17 - ...
- Decrease of FORTRAN knowledge, increase of Python usage, ...

(not an option) stay put

Start **rewriting** to separate « what » from « how »
- Bottom-up approach: Frameworks (e.g. Arcane, kokkos, …)
- Top-down approach: DSLs or task-based models (e.g. Charm++)

Focus on performance portability for the long run

Evaluate long term support options
- Tools, libraries
- Community presence

---

B. Hendrickson, The Day After Tomorrow: The Looming Post-Exascale Crisis, IPDPS, 2018
• Establish a quantitative measure of LC potential of evolution
  o Assumption: Exascale+ systems will be accelerated

• Code Viscosity

\[ V = \frac{A \times L}{TC \times P} \]

  o A = age of code
  o L = SLOC in million
  o TC = Team confidence to develop accelerated codes [0.001, 1]
  o P = % of acceleration [0.001, 1]

• More to come in 2020

The EXDCI-2 project receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 800957.
CONCLUSION

- LC are here to stay yet they must evolve
  - A costly process – viscosity evaluation under way
    - 300 ESLOC/month = 10€/ESLOC (*)
    - ESLOC = effective source line of code, fully tested, validated, commented, documented

- Get prepared for a cultural shift
  - Team composition
  - Development methods

- Fortunately we are facing interesting times
  - Computer architectures
  - Code development
  - Algorithms and numerical methods

Journée CCRT 2019 – LE JEUDI 12 DÉCEMBRE AU TGCC

De la simulation numérique... aux premiers pas quantiques

Inscription: secretariat.dssi@cea.fr
Questions?