Implementing Modern Fast Multipole Methods

Srinath Kailasa

Department of Mathematics
University College London

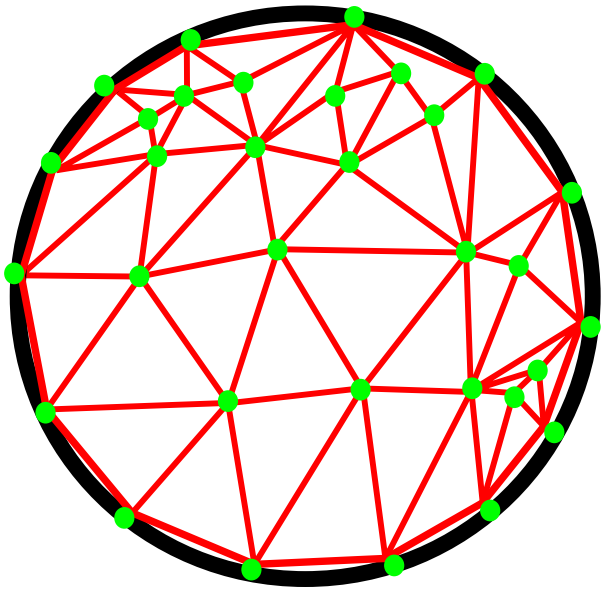
March 5, 2024

Table of Contents

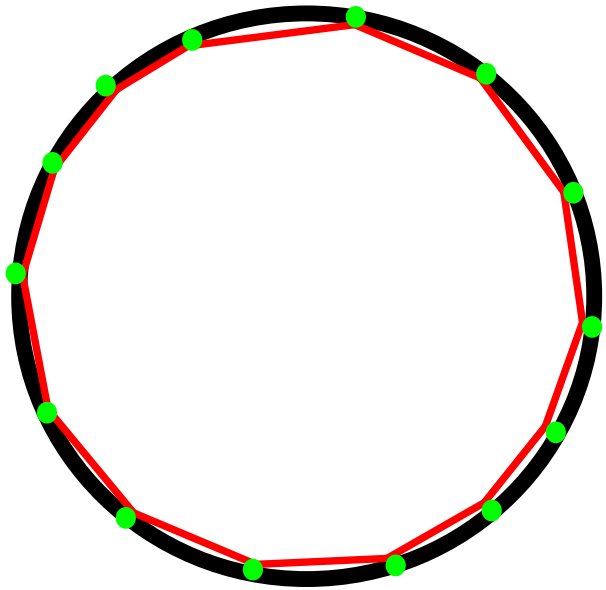
Table of Contents

Numerical Methods for PDEs

Consider 2D problem:



FEM



BEM

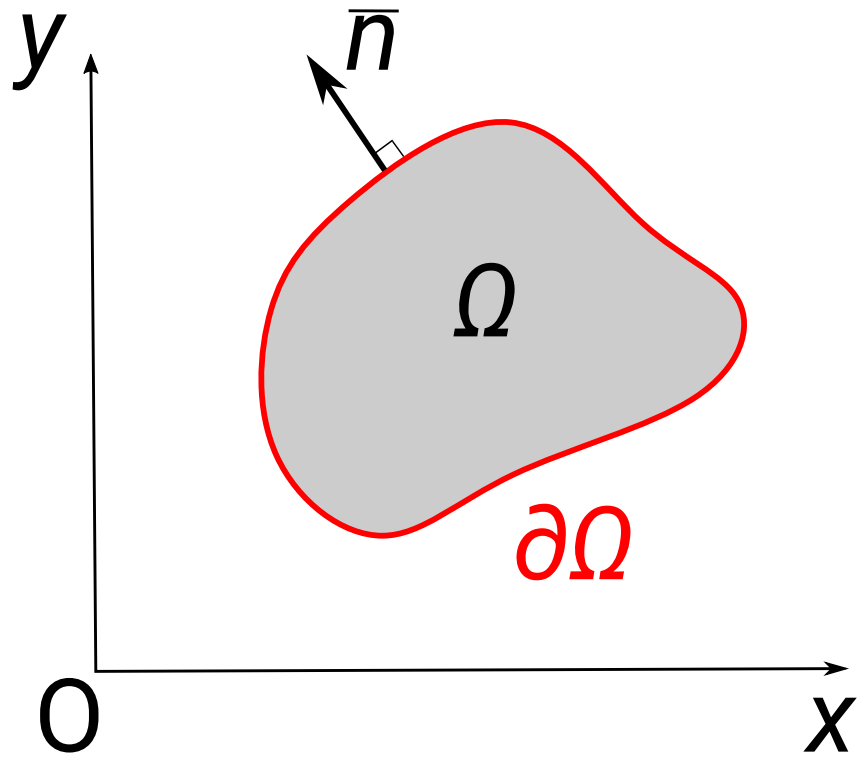
Laplace Interior Boundary Value Problem I

Want to solve

$$\Delta u(x) = 0 \qquad (1)$$

For $x \in \Omega$, with boundary $\partial\Omega$.

Form an *integral equation* using representation formula + boundary data.



We seek $u \in C^2(\Omega) \cap C^2(\bar{\Omega})$ which satisfies $u = f$ on $\partial\Omega$.
Let's write down a solution using a 'double layer potential',

$$u(x) = K\phi = \int_{\partial\Omega} \phi(y) \frac{\partial \Phi(x,y)}{\partial n(y)} ds(y), \;\; x \in \Omega$$

$\phi(y)$ unknown surface density, $\Phi(x,y)$ fundamental solution. End up with *boundary* integral equation,

$$\phi(x) - 2\int_{\Gamma} \phi(y) \frac{\partial \Phi(x,y)}{\partial n(y)} ds(y) = -2f(x), \;\; x \in \Gamma$$

$$-\frac{1}{2}\phi + K\phi = f$$

of the form

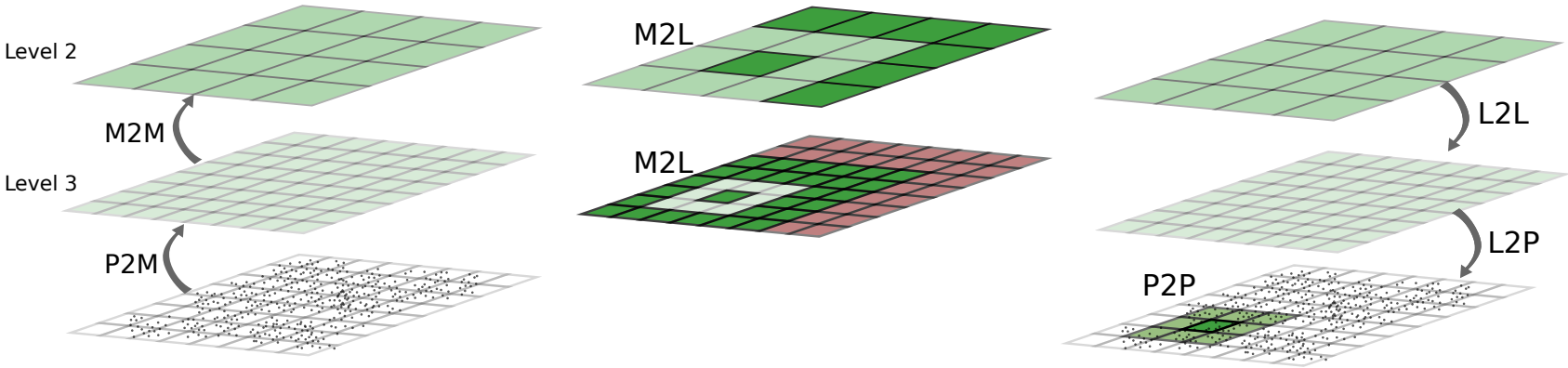
$$A\phi = f$$

with $A = K - \frac{1}{2}I$

Problem: A is *dense*. Slow $O(N^2)$ application cost, $O(N^3)$ inversion cost. CPUs of the 1980s can't handle this.
... But get to automatically incorporate boundary data *and* get a reduction in problem dimension.

Solution: Hierarchical Solvers (Fast Multipole Method). Can apply in $O(N)$ (sometimes), and invert using Krylov methods. Workaround for slow computers!

Fast Multipole Methods I



Consider N source and target particles,

- $O(N)$ boxes, so at least $O(N \log N)$ complexity.

FMM for Laplace is just $O(N)$, as number of boxes in interaction lists limited to constant number.

Fast Multipole Methods II

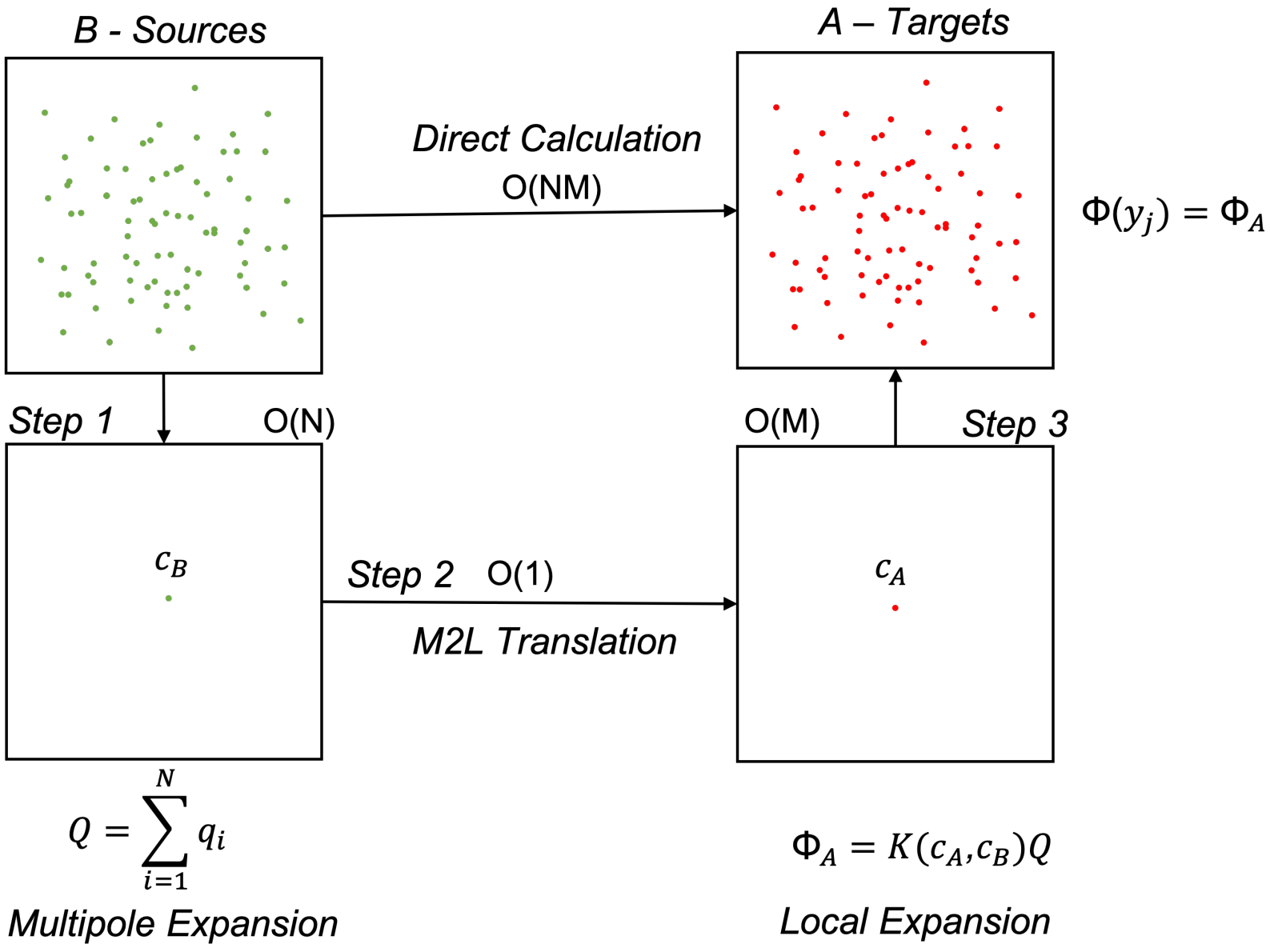
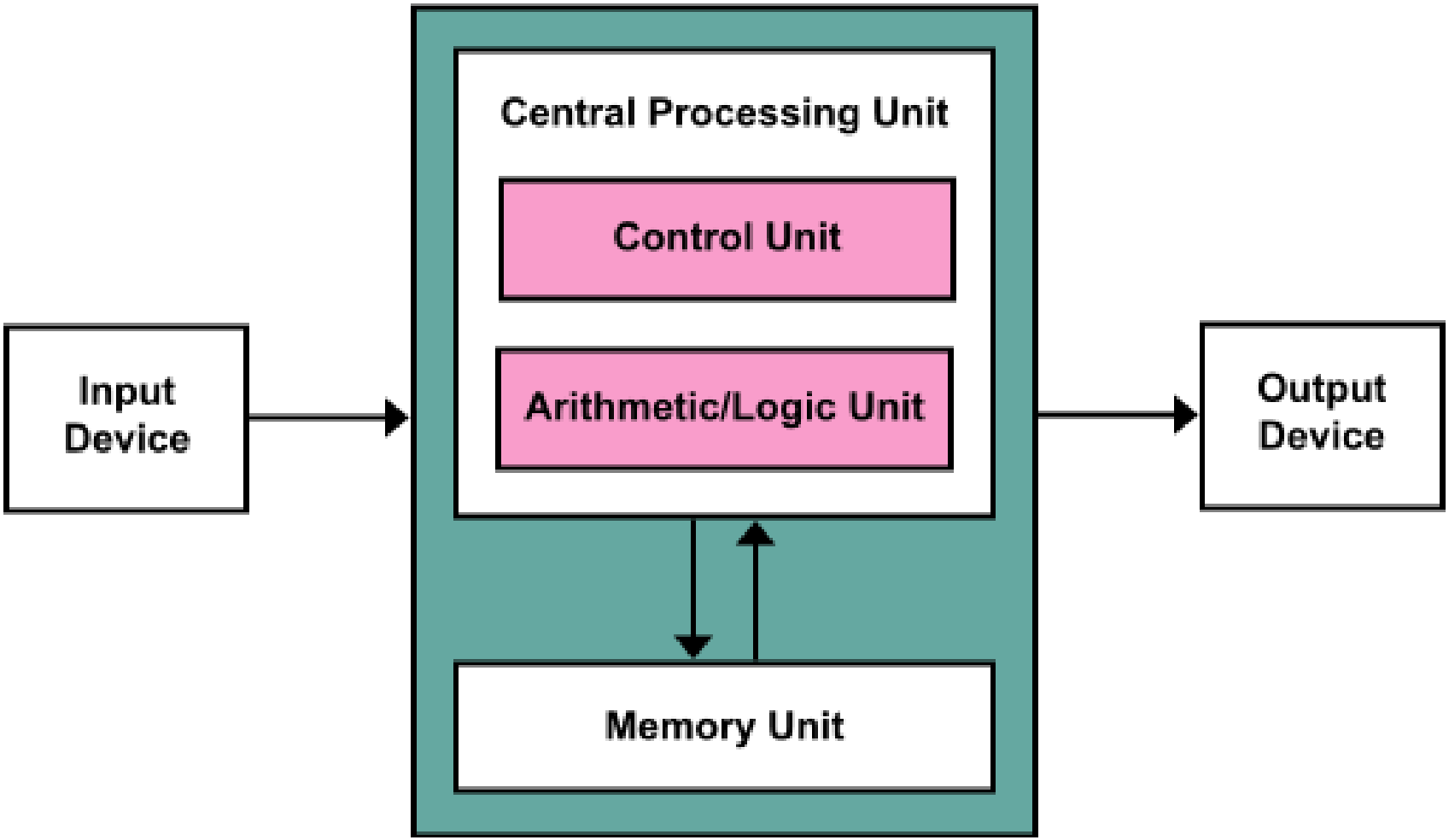


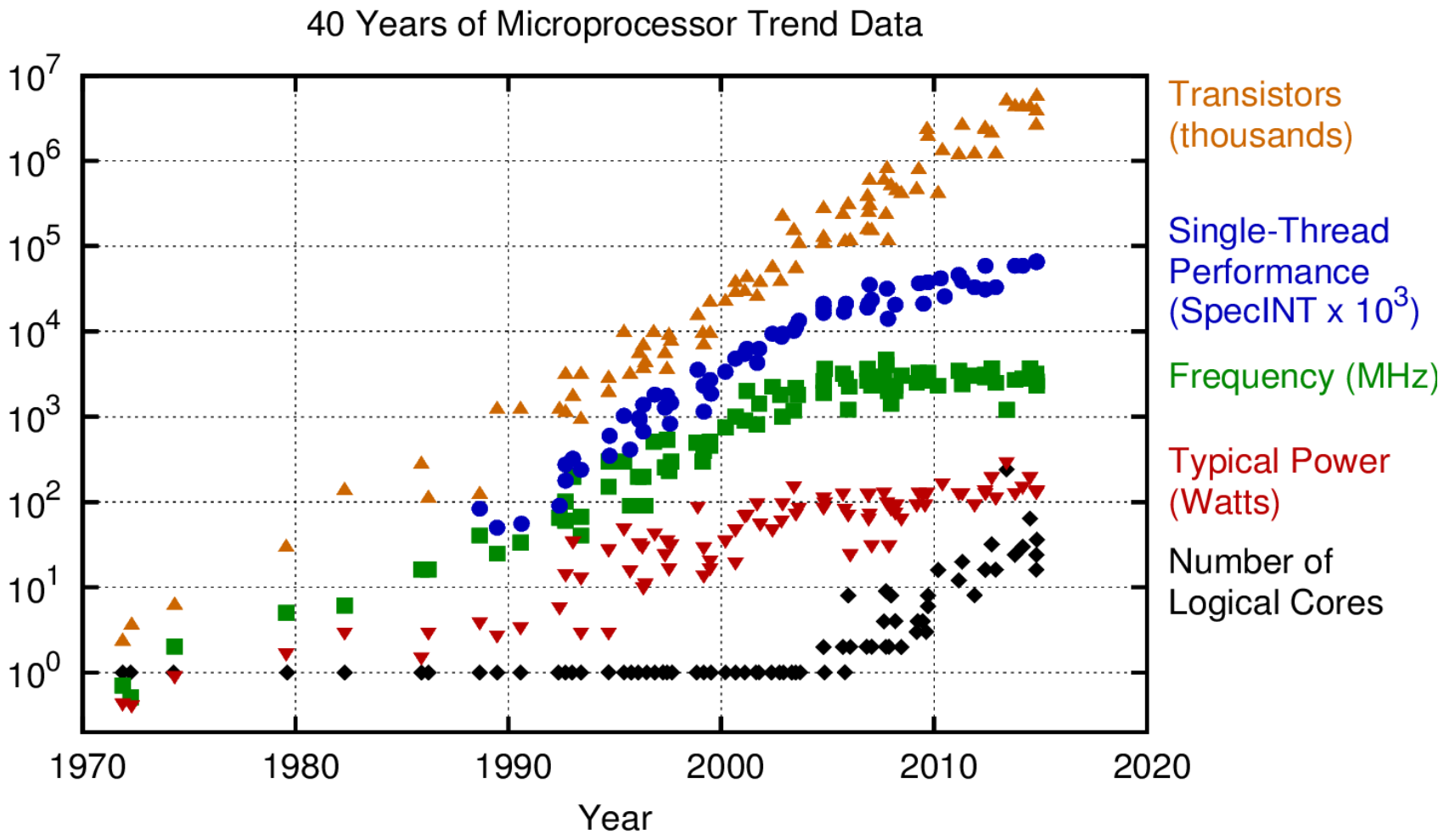
Table of Contents

Computer History I



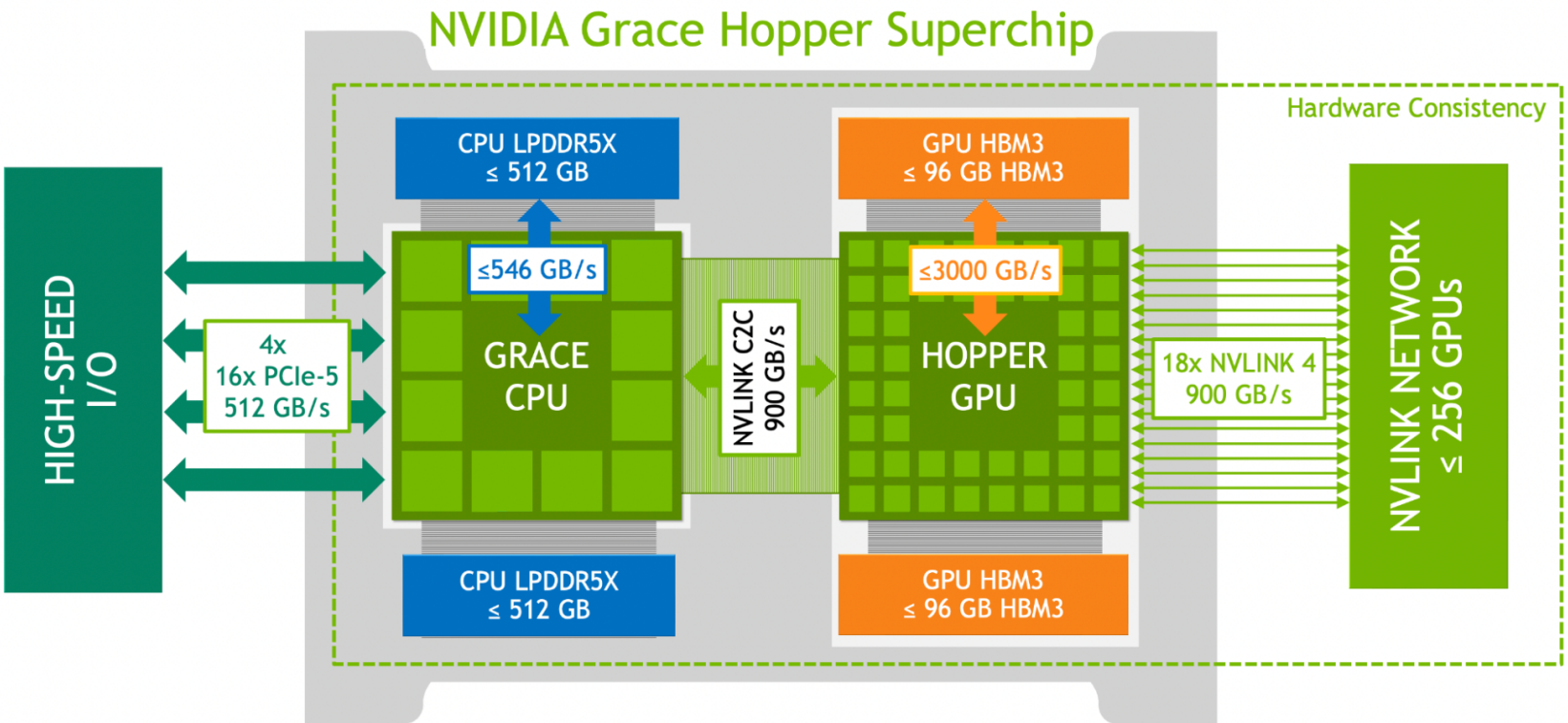
source: Wikipedia: Von Neumann Architectures

Computer History II



source: Conte, T, IEEE (2015)

Computer History II

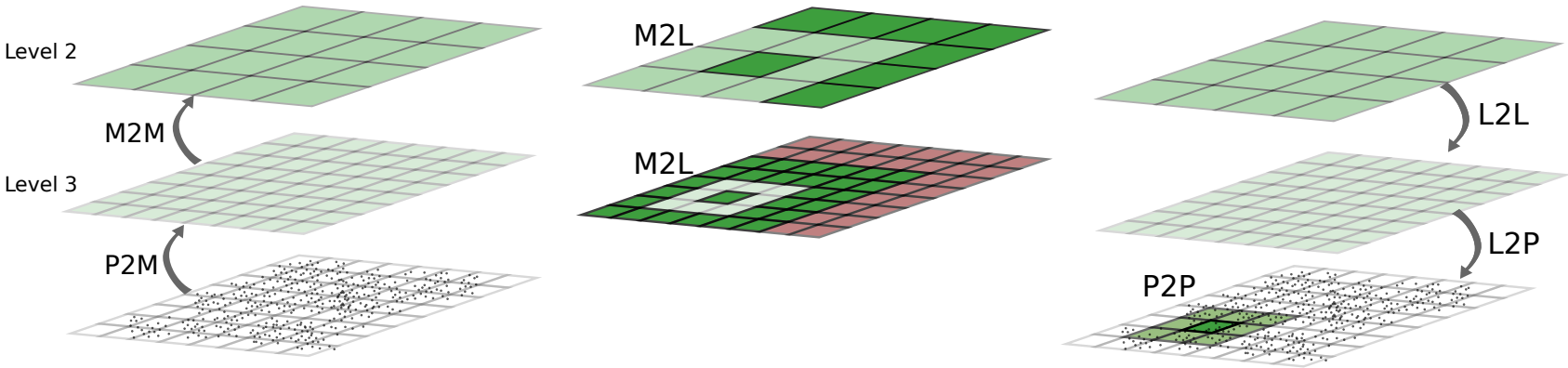


source: NVidia Corporation

Computational Bottlenecks for the FMM

- **P2P** - Direct Kernel evaluations - $O(N^2)$ flops $O(N)$ memory accesses.
- **M2L** - Translation of the Multipole into a Local Expansion (Formulation Dependant).

Computational Bottlenecks for the FMM II



The Kernel Independent FMM I

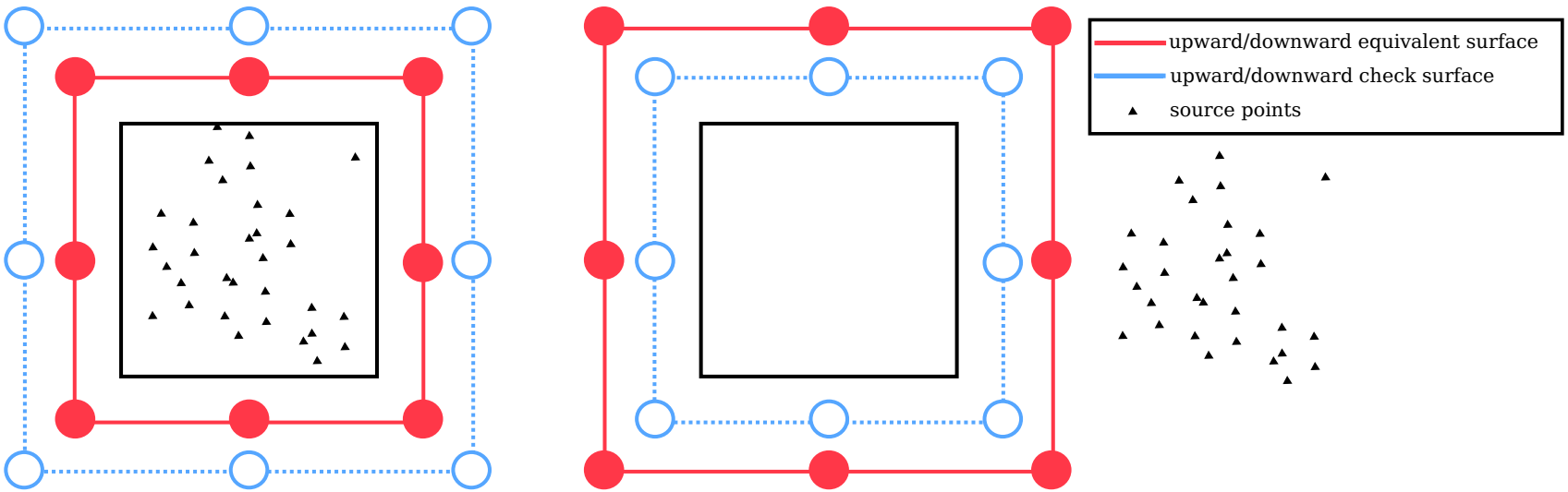
Consider simple test case, want to evaluate Laplace Green's function between N targets $\{x_i\}_{i=1}^N$ and N sources $\{y_i\}_{i=1}^N$.

$$\phi(x_i) = \sum_{j=1}^{N} K(x_i, y_j) q(y_j) \tag{2}$$

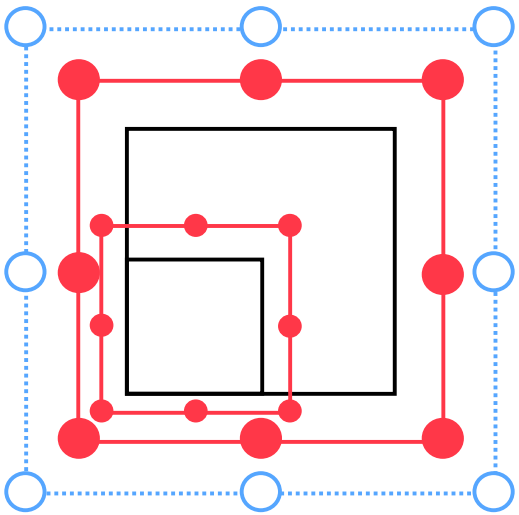
$$K(x, y) = \frac{1}{4\pi \| |x - y\|}$$

Want to accelerate with an FMM, to compute in $O(N)$

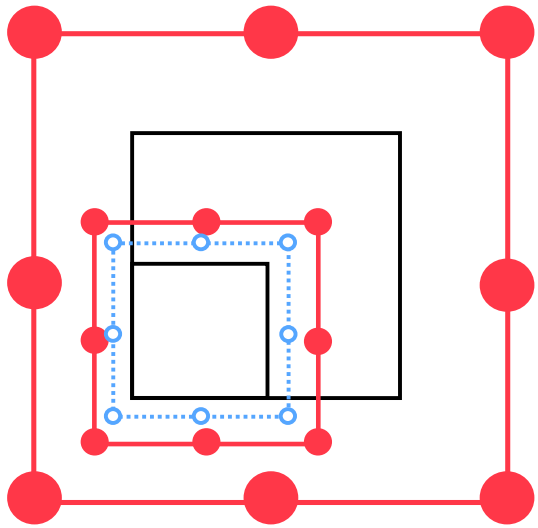
The Kernel Independent FMM II



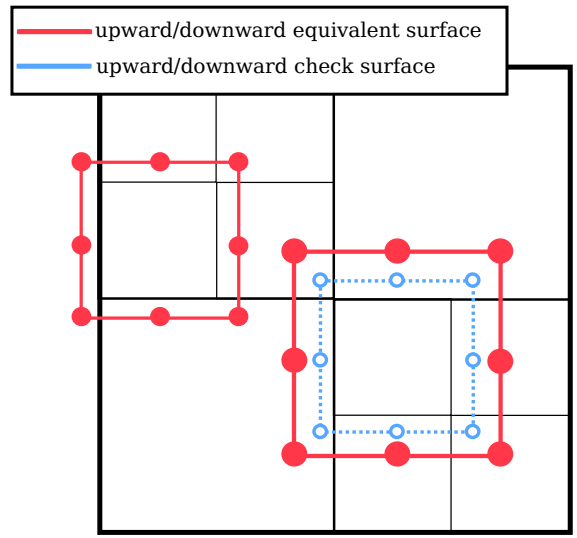
The Kernel Independent FMM III



M2M



L2L



M2L

Can view M2L as either a convolution (FFT) or as a dense matrix vector product (BLAS)

$$\phi^{B,d}(x) = (K * q)(x) = \int_{y^{B,d}} K(x-y)q^{A,u}(y)dy \tag{3}$$

FFT memory throttled (notice the element-wise product), but is of only $O(N \log N)$ flops.

$$\phi(x) = \mathcal{F}^{-1}\{\mathcal{F}\{K\}(k) \cdot \mathcal{F}\{q\}(k)\}(x) \tag{4}$$

BLAS is compute bound, but of $O(p^2)$ flops ...

Modern Implementations I

Modern implementations of algorithms have to reflect on advances in hardware, where we are memory bound and flops are *cheap*.

DESIGNING KERNEL INDEPENDENT FAST MULTIPOLE METHODS FOR MODERN ARCHITECUTURES *

SRINATH KAILASA† AND TIMO BETCKE‡

Abstract. Kernel independent Fast Multipole Methods (kiFMM), with their broad applicability to a wide variety of elliptic partial differential equations with asymptotically smooth kernels, have emerged as a preferred method for computer implementations of recent benchmark softwares due to

Modern Implementations II

- Counterintuitively prefer a dense matrix vector product to an FFT, because of greater arithmetic intensity.
- Can stack together vectors which share a matrix, i.e. matrix multiplication, and get even greater data re-use.
- Can use numerical compression (SVD etc) to reduce size of (low-rank) matrices.
- Use specialised hardware and unified memory to keep the cost of GPU-CPU data exchange minimal.

Table of Contents

Conclusion

Key Takeaway: Hardware, software and algorithms advance in lockstep. Who knows what the computing systems of the future will enable! (Or the algorithms that will be devised to circumvent their failings).

source: The Next Platform (2023)