Imperial College London

Towards Heterogeneous Solvers for Large-Scale Linear Systems

Stylianos I. Venieris, Grigorios Mingas, Christos-Savvas Bouganis
stylianos.venieris10@imperial.ac.uk

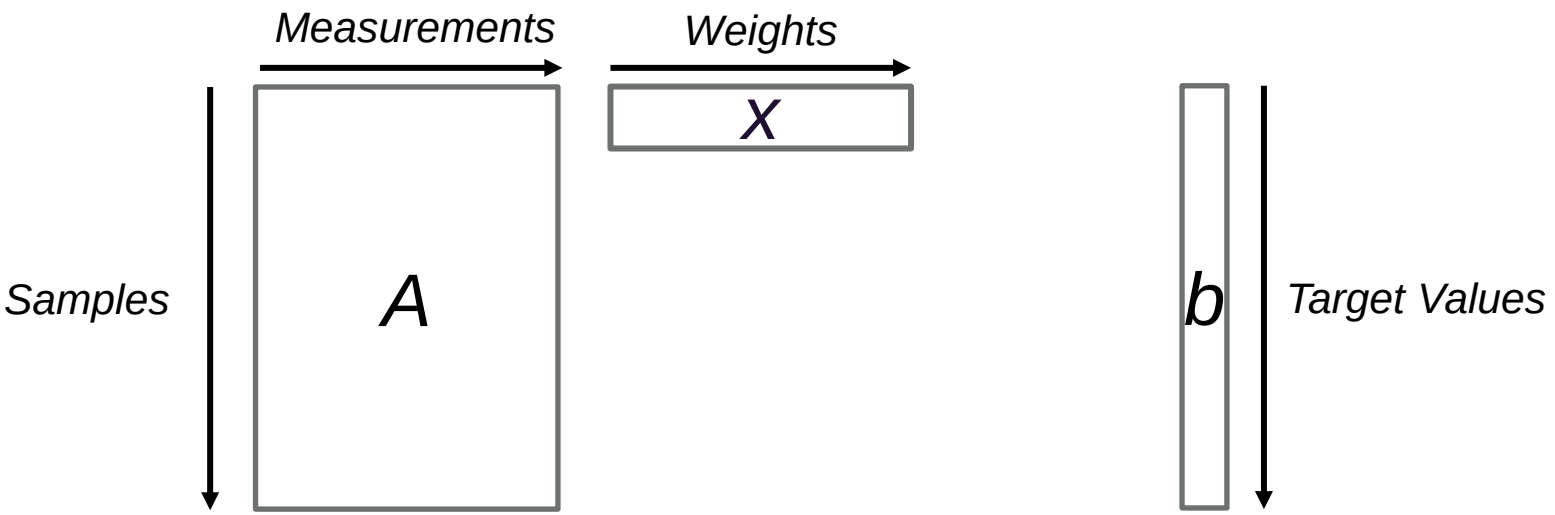
FPL 2015, London
2 Sept 2015

Introduction – Solving Linear Systems

- Given $\boldsymbol{A} \in R^{(m \times n)}$, $\boldsymbol{b} \in R^{m}$ and $m \geq n$:

$$\min_{\boldsymbol{x} \in R^{n}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2$$

- Find vector $\boldsymbol{x}$

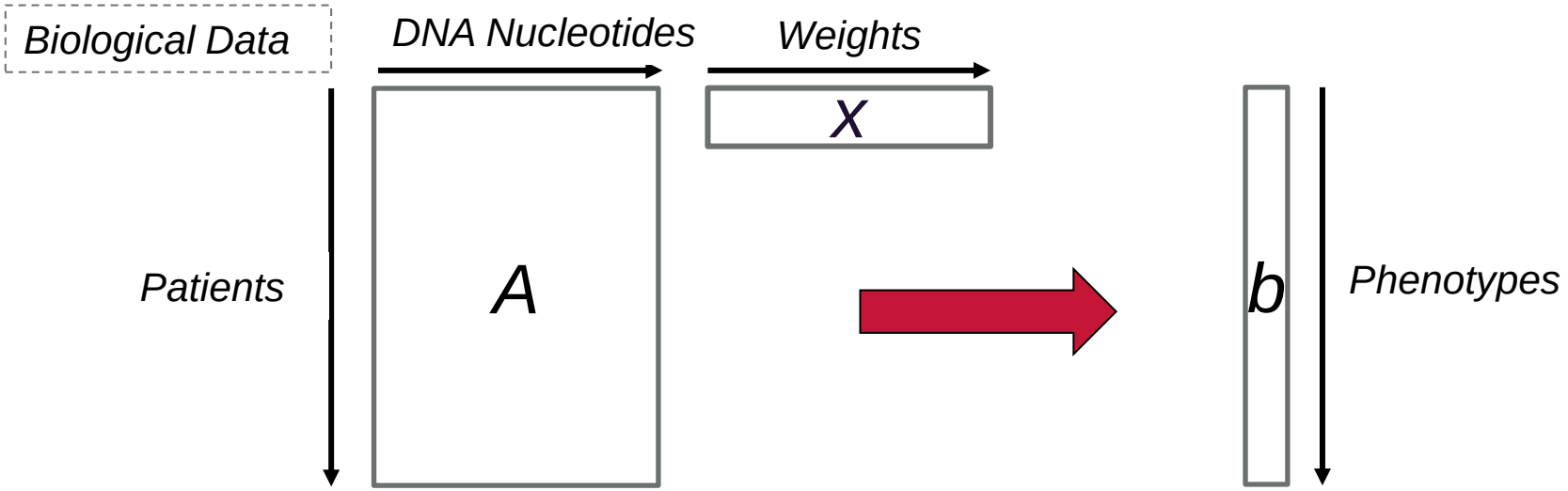


Introduction – Solving Linear Systems

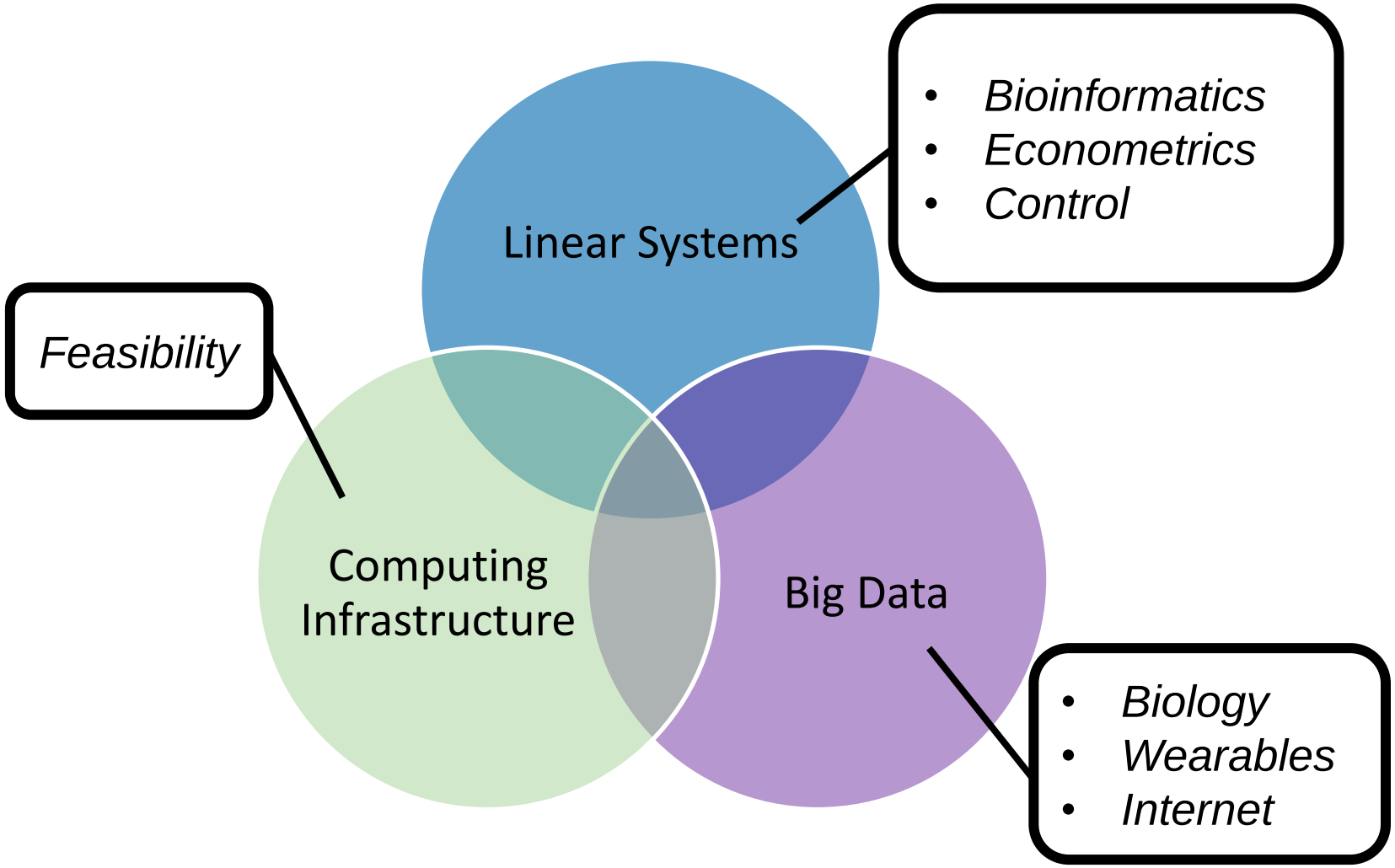
- Given $\boldsymbol{A} \in R^{(m \times n)}$, $\boldsymbol{b} \in R^{m}$ and $m \geq n$:

$$\min_{\boldsymbol{x} \in R^{n}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2$$

- Find vector $\boldsymbol{x}$

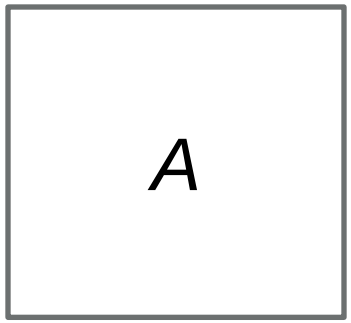


Introduction – Solving Linear Systems



Introduction – Data Systems with different structure

Square System



Tall-Skinny System

- *Many samples (rows)*
- *A few measurements (columns)*

$m = n$

$m \gg n$

Anything in between

Introduction – Linear Systems in Genetic Analysis

- Real-life Example:
  - Genetic Analysis [1]
  - Search for gene combinations by solving lots of linear systems
- Application Characteristics:
  - A lot of linear systems
  - Linear systems of varying size
  - Up to 100,000 rows and a few hundred columns

[1] L. Bottolo and S. Richardson, "Evolutionary Stochastic Search for Bayesian Model Exploration", Bayesian Analysis, vol. 5, no. 3, pp. 583–618, 09 2010.

Our focus

- Towards heterogeneity for high performance across matrix sizes
- Novel FPGA solver for tall-skinny linear systems
- Modelling framework
  - Performance and resource estimation (compile and runtime)
  - Optimal hardware configuration (compile-time)
- Up to 18x speed-up in GFLOPS across matrix sizes compared to existing works

Route Map

- Background
- Towards Heterogeneity for Performance
- An Enhanced FPGA Solver
- Evaluation Results
- Conclusions

Background – Solving Linear Systems

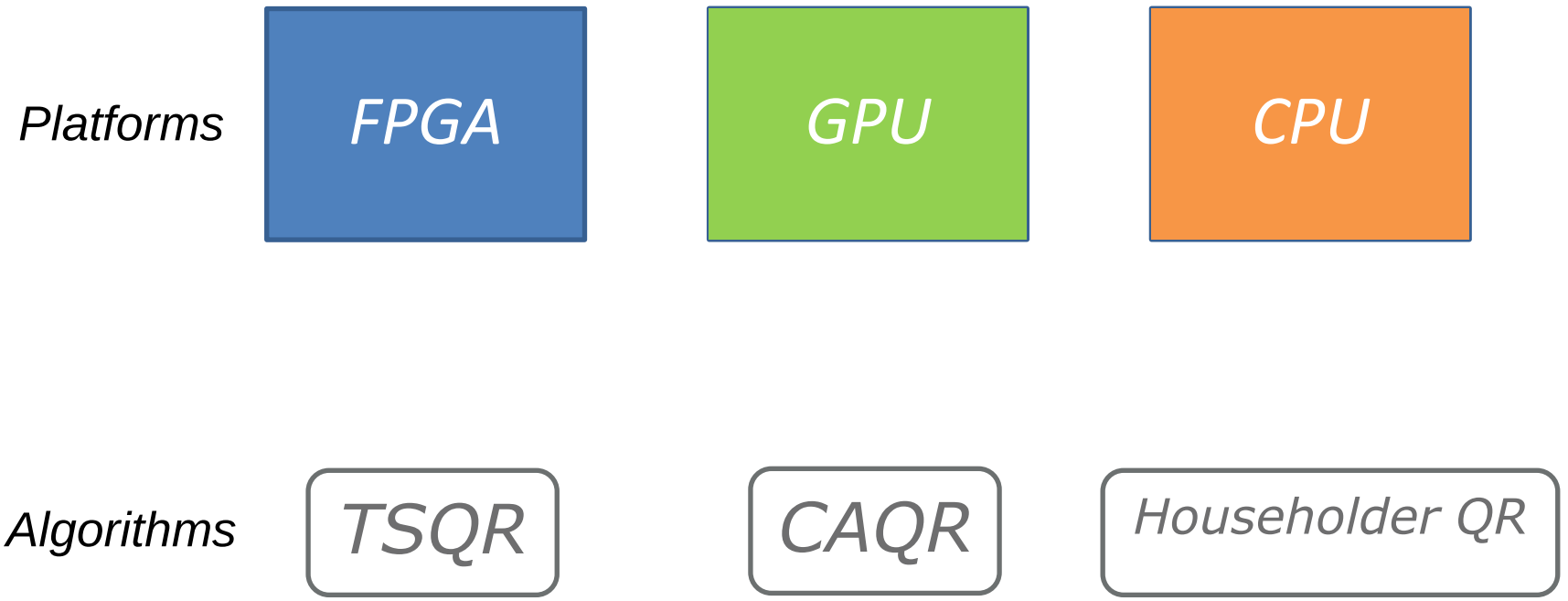
- The QR factorisation-based methods are dominant because of their properties
- $\boldsymbol{A = QR}$ – Orthogonal $\boldsymbol{Q}$, upper-triangular $\boldsymbol{R}$
- $\boldsymbol{Ax = b \Rightarrow QRx = b}$
- $\boldsymbol{Q^T b}$ – Matrix-vector product
- $\boldsymbol{x = R^{-1} Q^T b}$ ← *Solution using back-substitution*

The Challenge

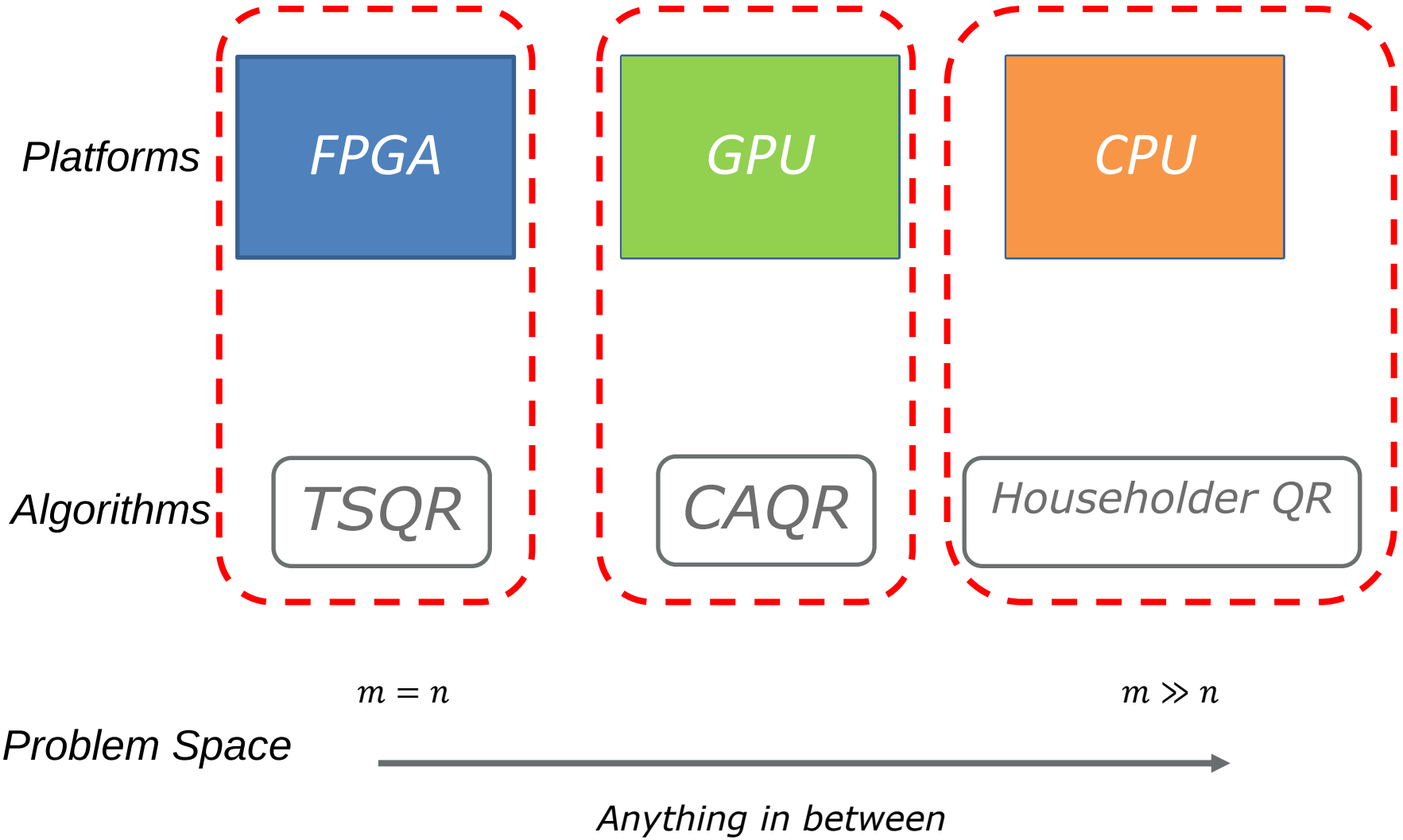
- Existing solvers
  - Target CPUs, GPUs and FPGAs
  - Employ different algorithms
  - Tailored to specific matrix sizes
- Key Challenge:

  Sustain high performance across matrix sizes

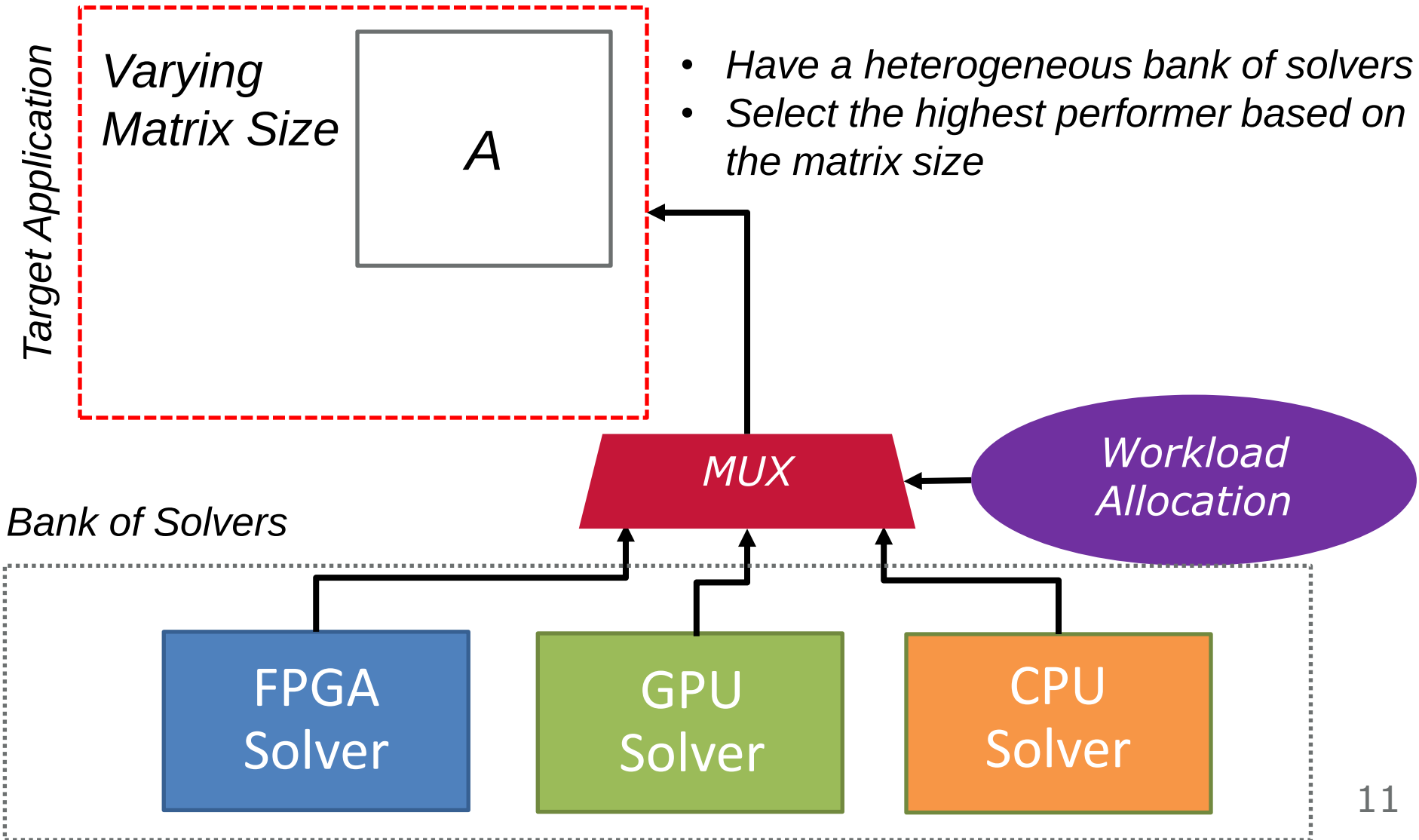
Heterogeneity for Performance – Different Solvers



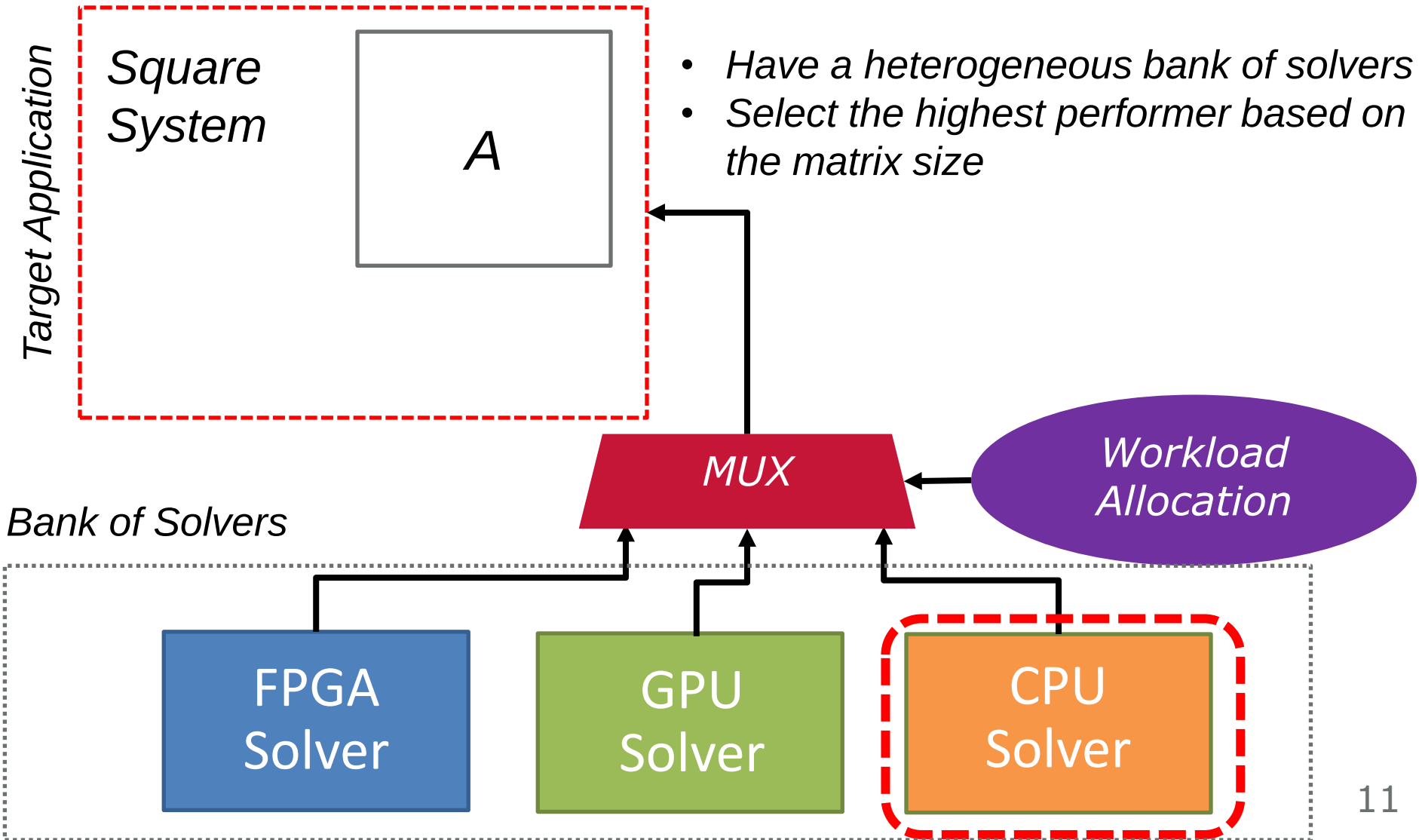
Heterogeneity for Performance – Different Solvers



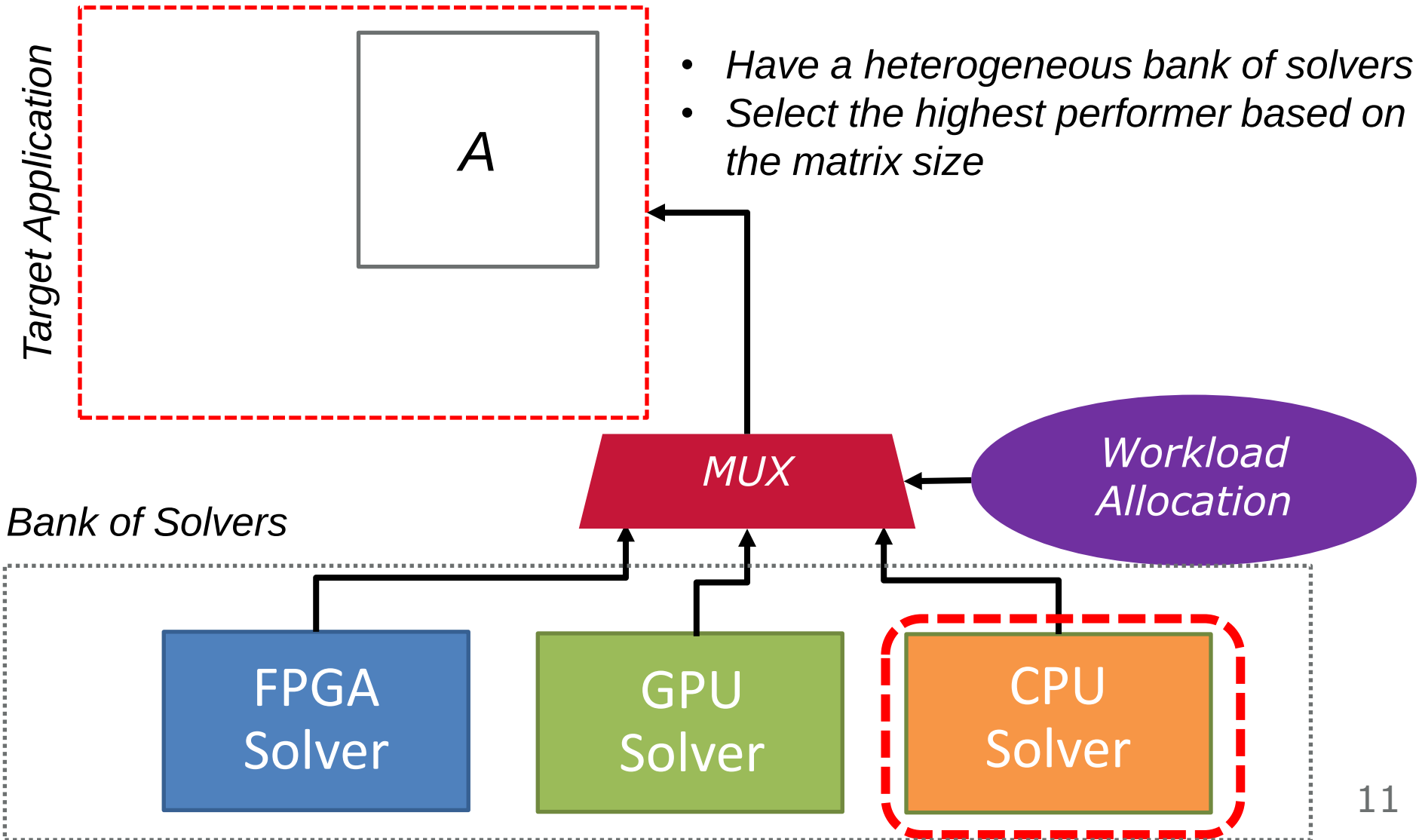
Heterogeneity for Performance - A Heterogeneous Solver



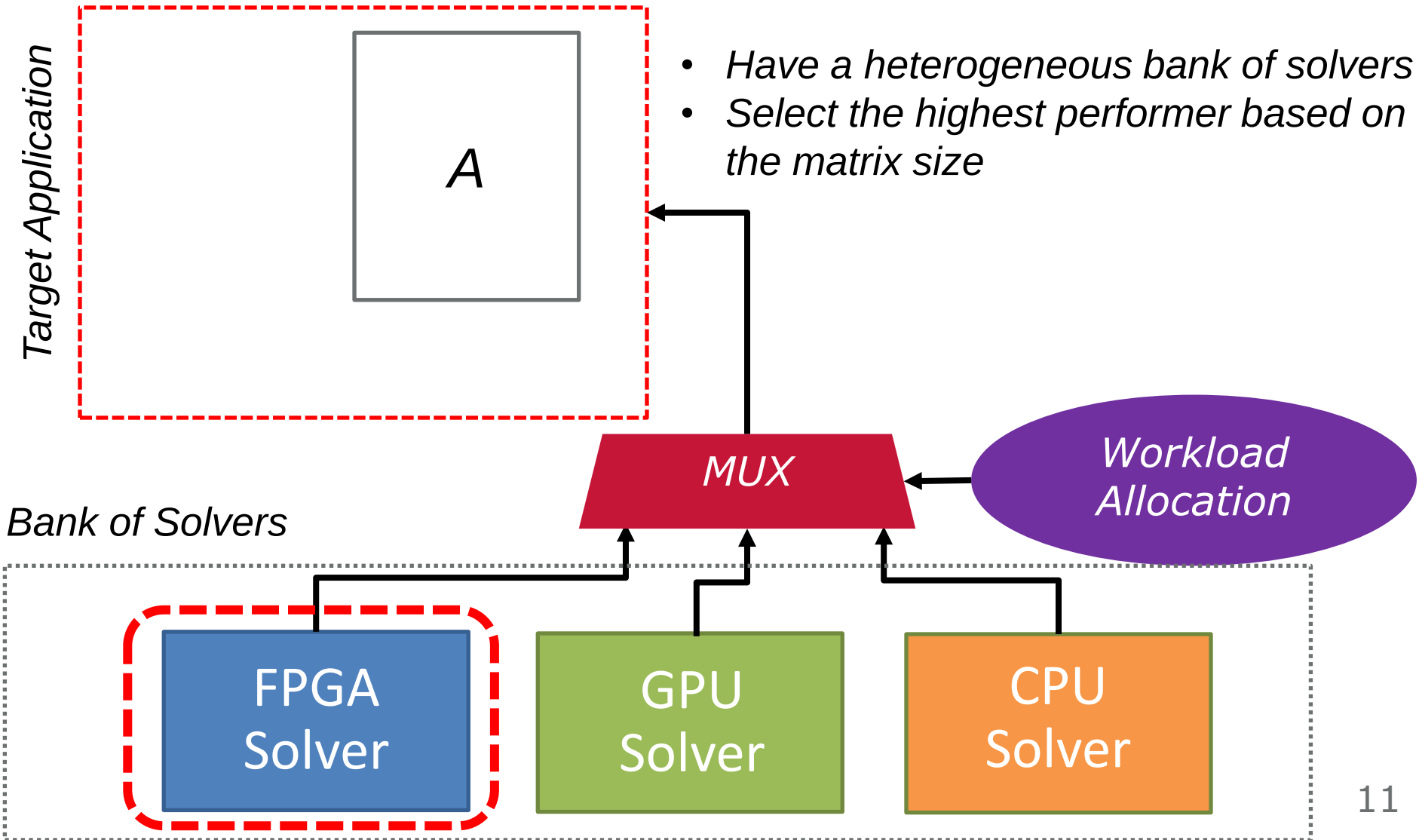
Heterogeneity for Performance - A Heterogeneous Solver



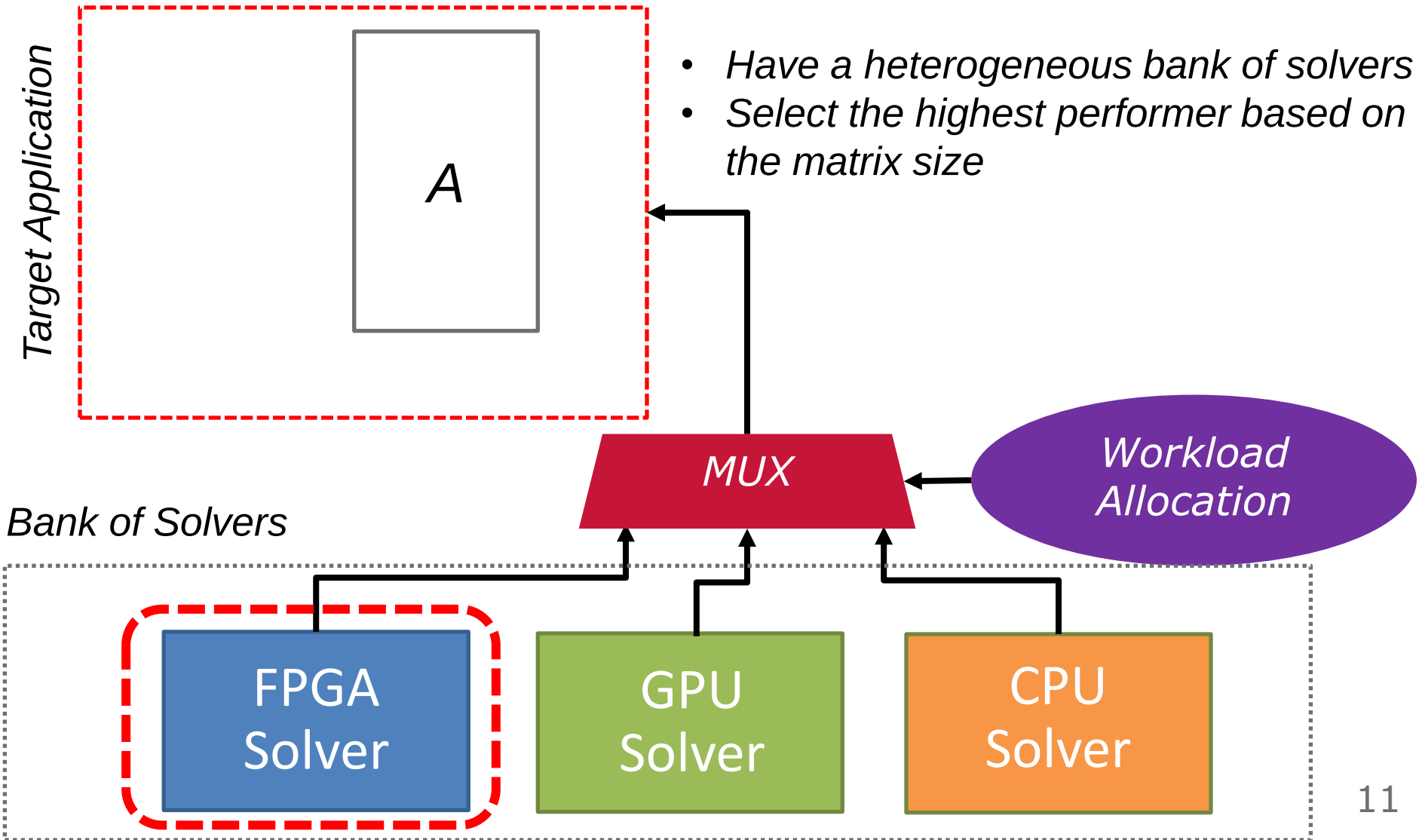
Heterogeneity for Performance - A Heterogeneous Solver



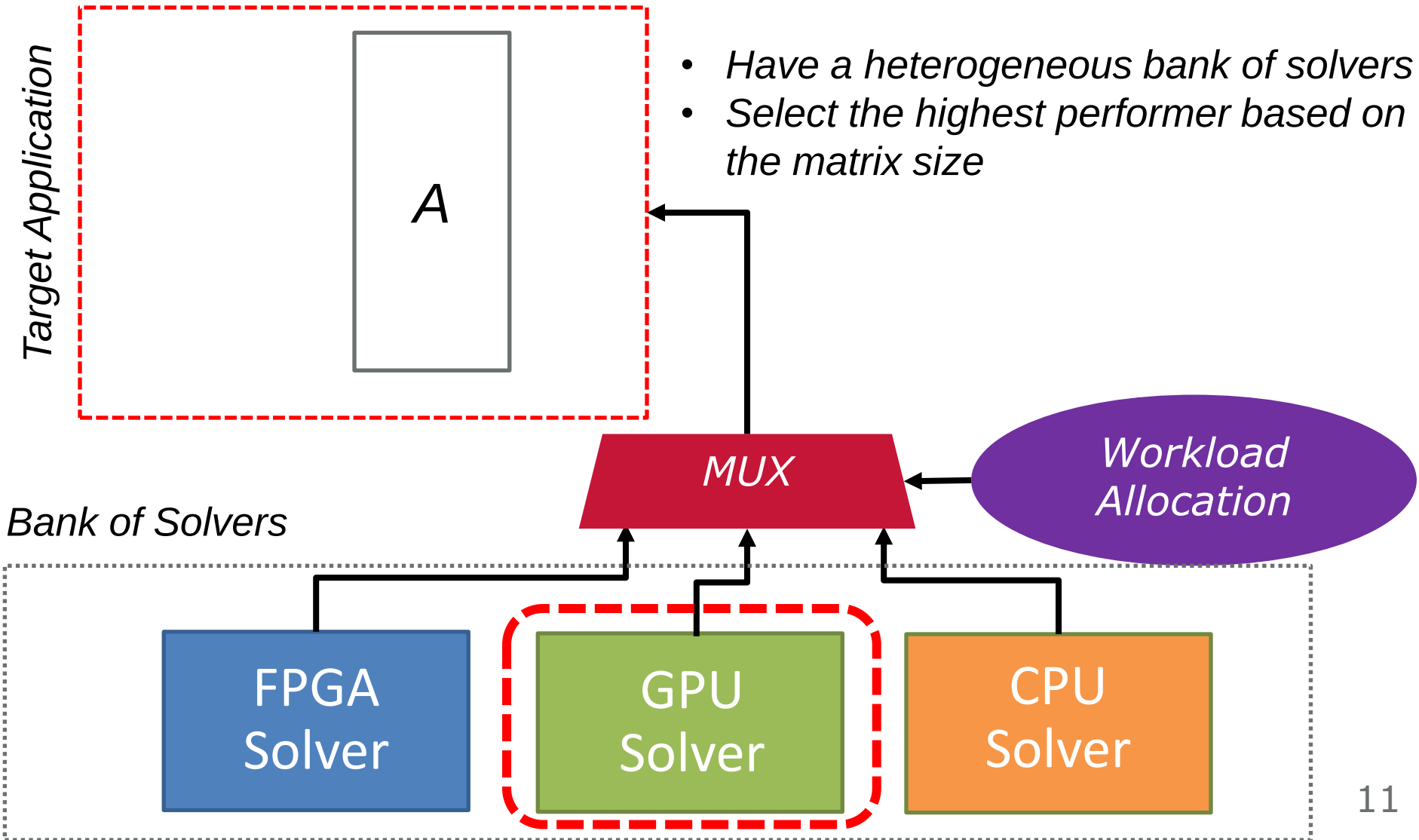
Heterogeneity for Performance - A Heterogeneous Solver



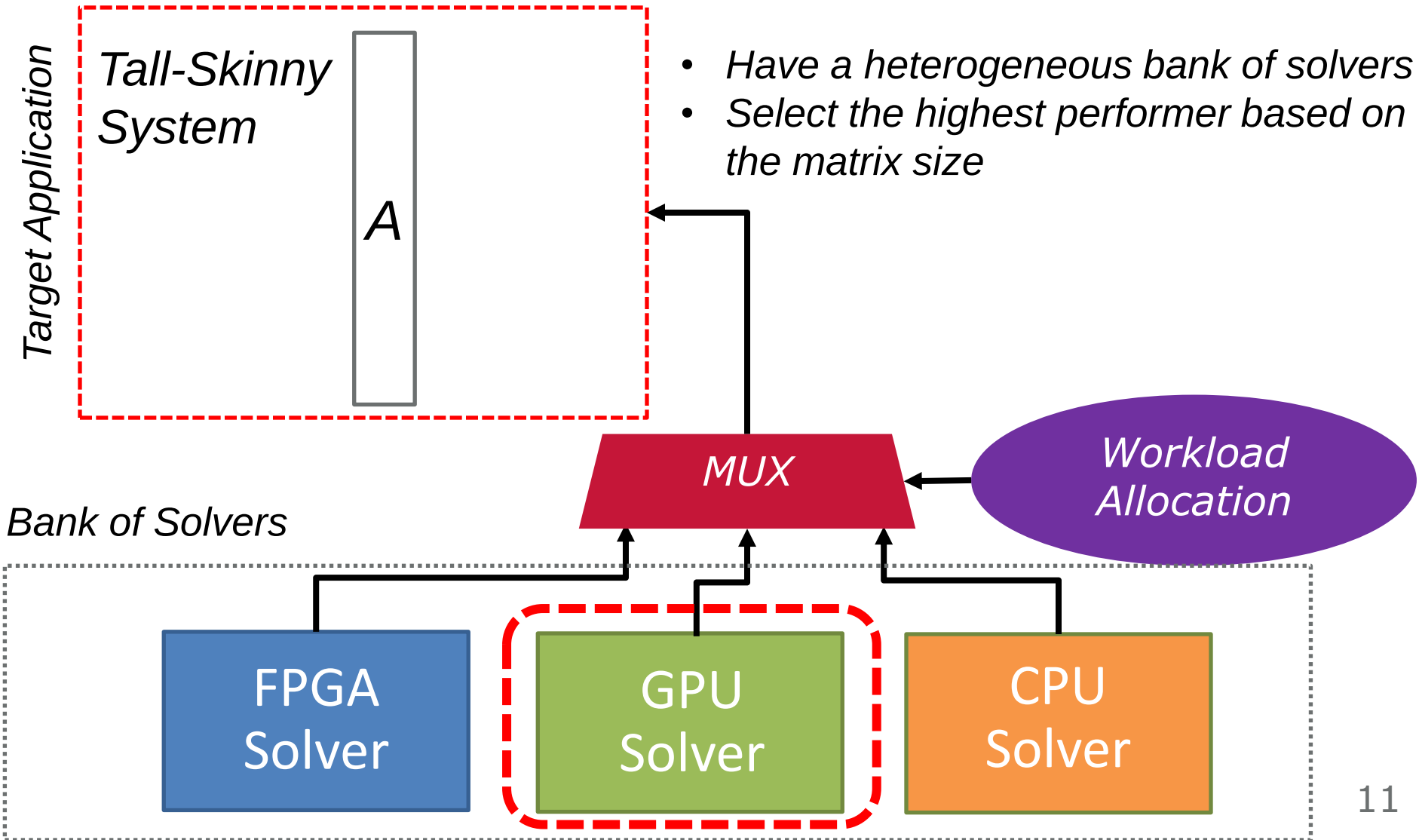
Heterogeneity for Performance - A Heterogeneous Solver



Heterogeneity for Performance - A Heterogeneous Solver



Heterogeneity for Performance - A Heterogeneous Solver



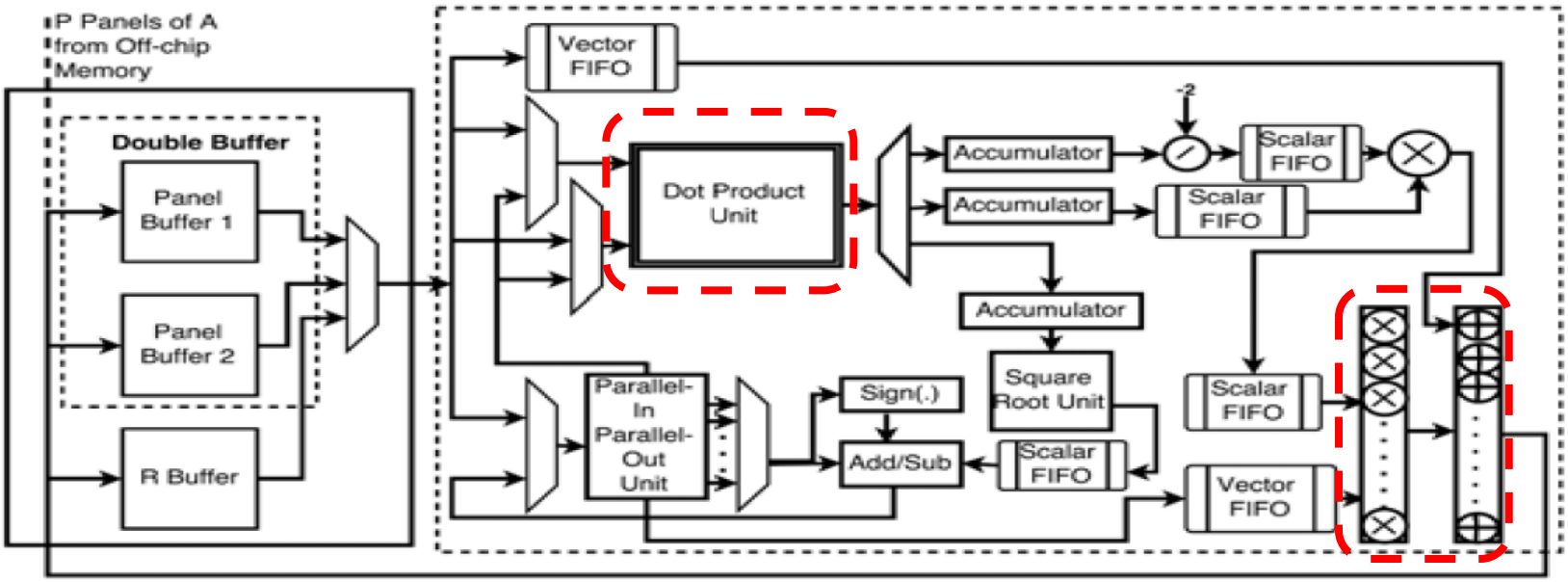
Compute Engines

Compute Engines: FPGA Solver

- *Based on existing architecture for tall-skinny QR factorisations [2]*
- ***Functionality Extension***
  - *From QR to Linear Systems workloads*
  - *Exploited an algorithmic property for acceleration*
    - *"Concurrent Solution and Factorisation"*
- *Any no. of rows*
- *Up to a max no. of columns*
  - *5x the max no. of columns of existing FPGA work for the same device*

[2] A. Rafique et al., FPL 2012.

Compute Engines: FPGA Solver



- *Configurable parameters*
  - *Size of arithmetic units*
  - *No. of blocks to be active in parallel in the architecture*

Concurrent Solution and Factorisation

1. $A = QR$

- *Numerically stable algorithms do not return Q explicitly*
- *Additional computations for the reconstruction of Q and for $Q^T b$*

2. $Ax = b \Rightarrow QRx = b$

3. $Q^T b$ – Matrix-vector product

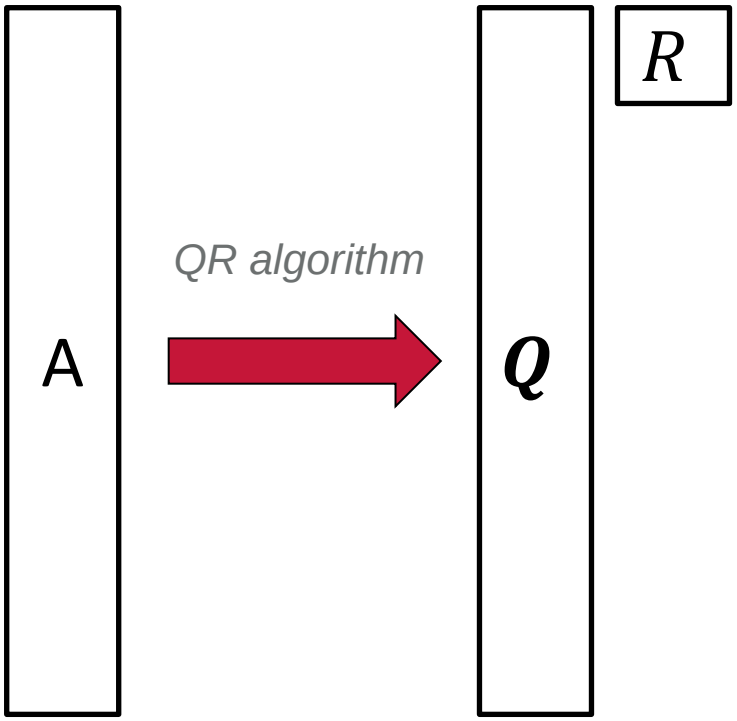
4. $x = R^{-1}Q^T b$

Solution

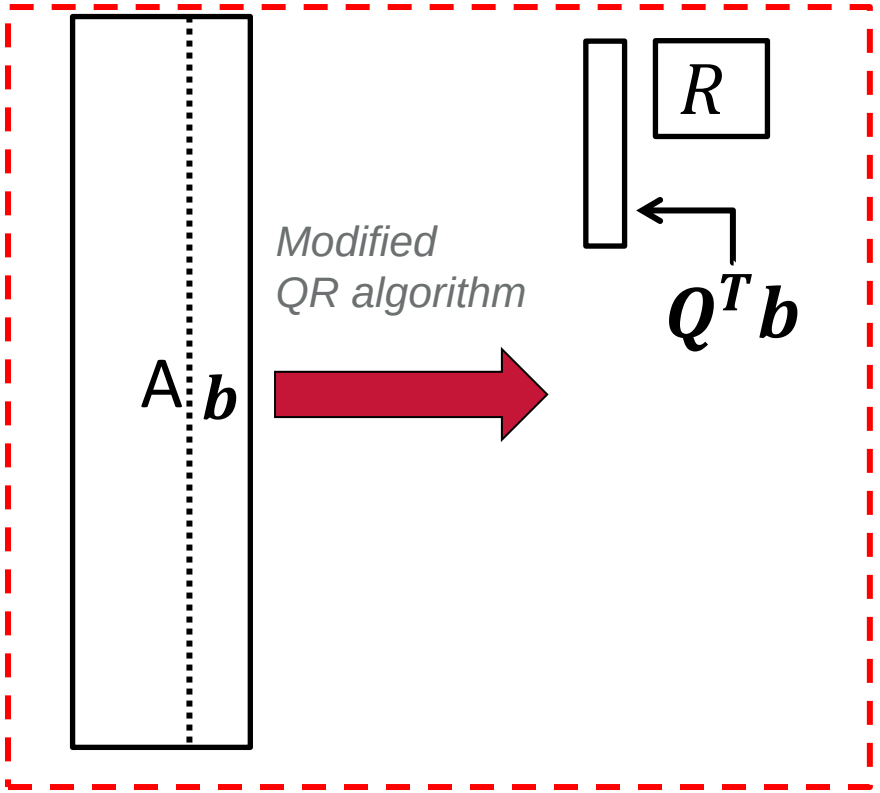
Concurrent Solution and Factorisation

- *Compute $\boldsymbol{Q}^T\boldsymbol{b}$ without forming $\boldsymbol{Q}$ explicitly*

Previous works



This work



Q reconstructed from partial results

Compute Engines: GPU and CPU Solvers

- *GPU Solver*
  - *Based on the state-of-the-art work on tall-skinny QR factorisations [3]*
  - *Extended its functionality to Linear Systems by means of the Concurrent Solution and Factorisation*
- *CPU Solver*
  - *Optimised multithreaded linear algebra library (OpenBLAS)*

[3] M. Anderson, C. Ballard, J. Demmel, and K. Keutzer, "Communication-Avoiding QR Decomposition for GPUs", in Parallel Distributed Processing Symposium (IPDPS), 2011 IEEE International, May 2011, pp.48-58.

Modelling Framework

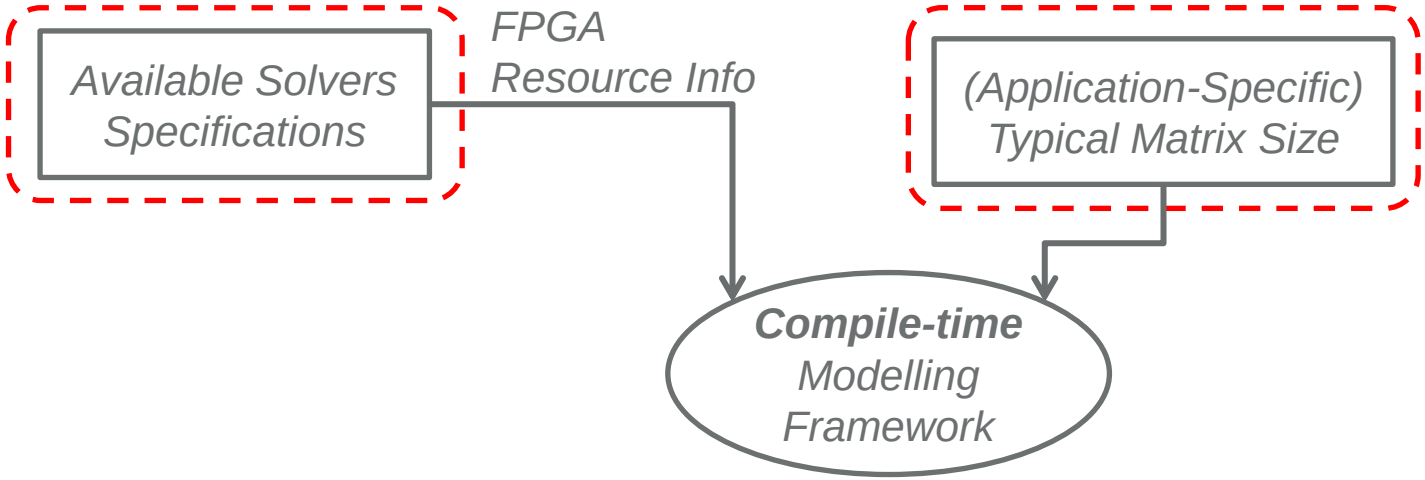
- *Compile-time Framework*
  - *Performance in GFLOPS and resource estimation models for the FPGA solver*
  - *Optimal hardware configuration of the FPGA solver*
- *Runtime Framework*
  - *Workload allocation among the available solvers*

Heterogeneity for Performance - Proposed Design Flow

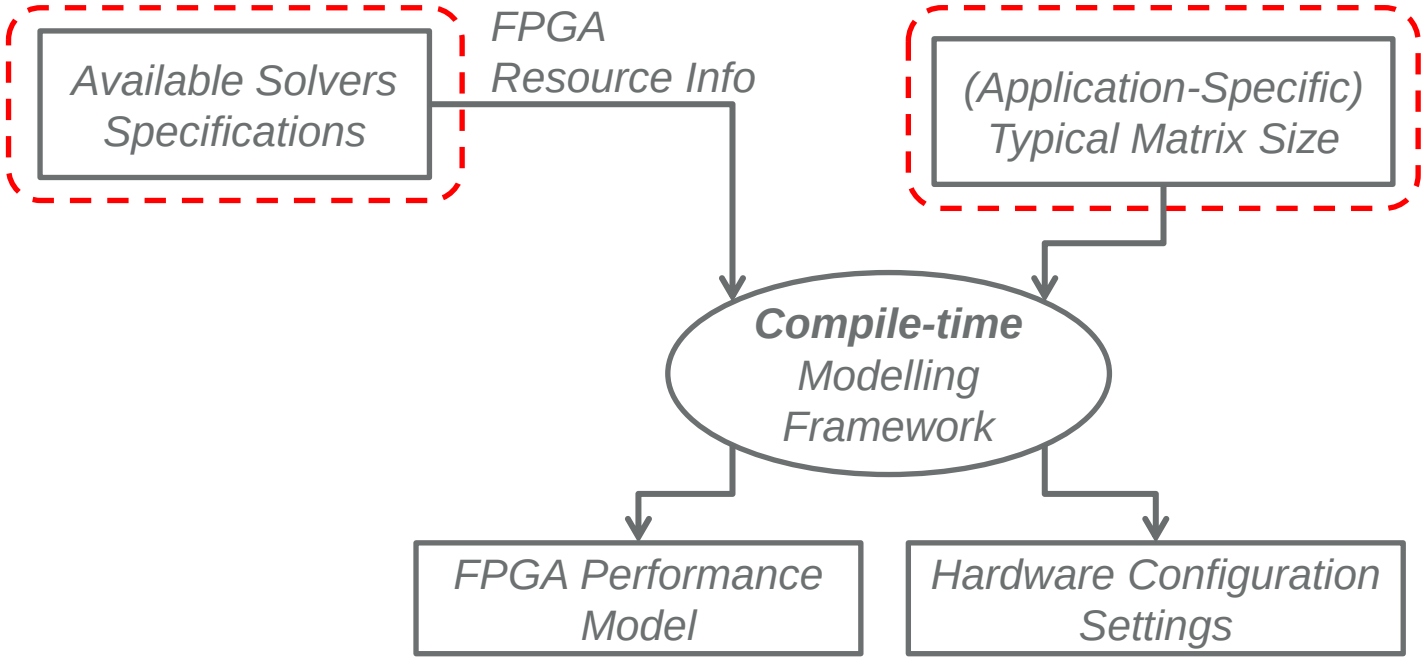
Available Solvers Specifications

Supplied by Application Expert

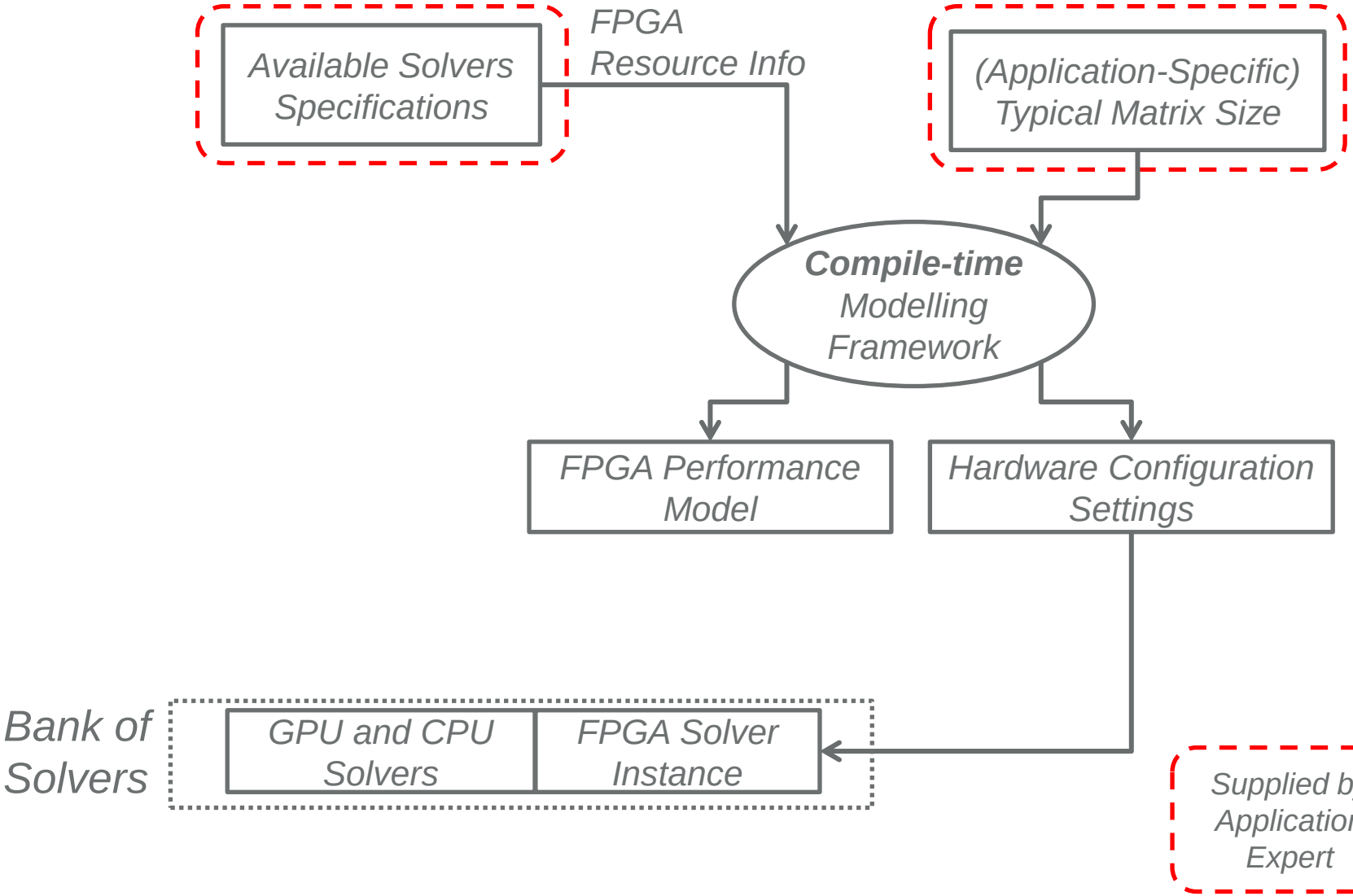
Heterogeneity for Performance - Proposed Design Flow



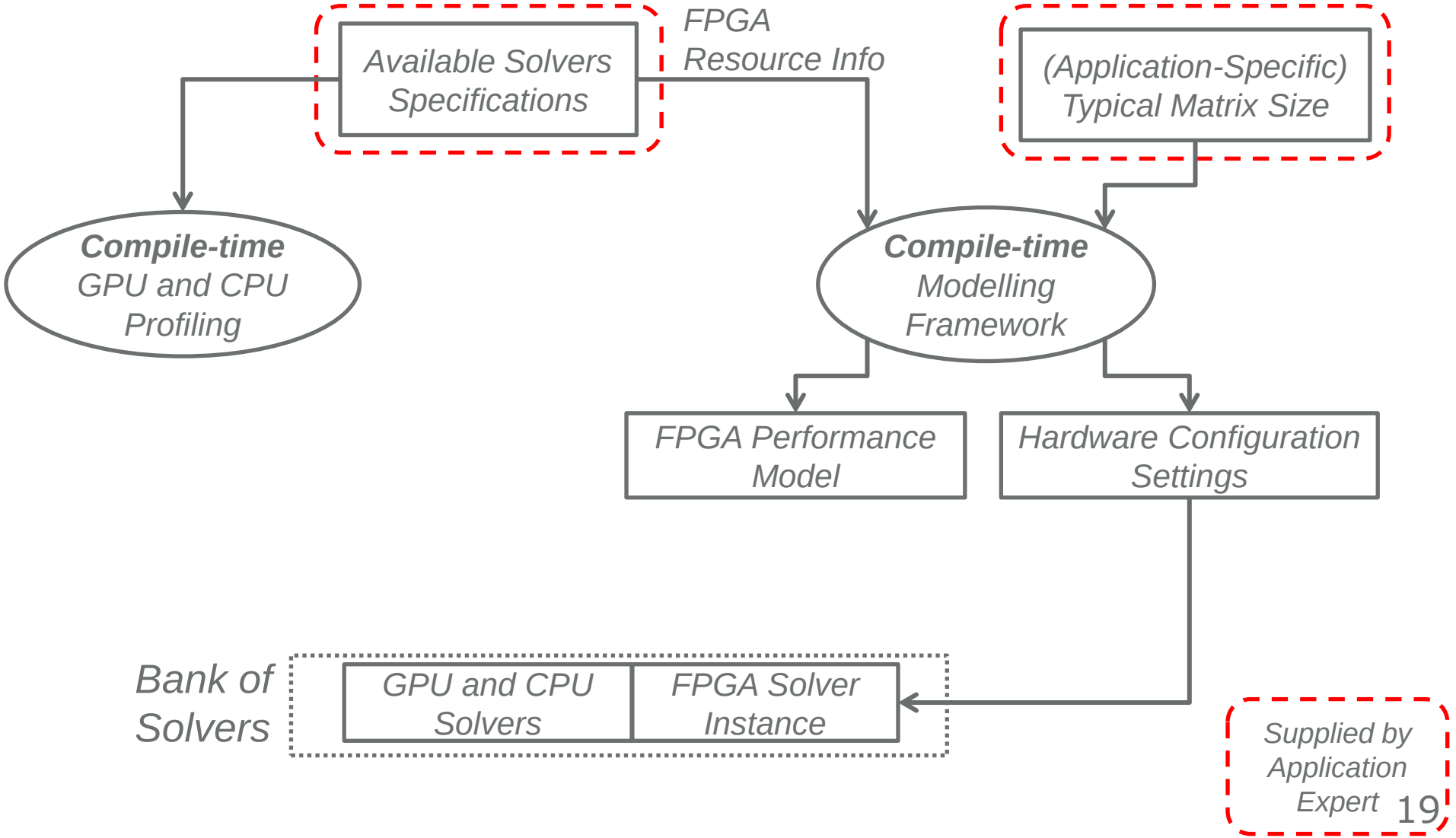
Heterogeneity for Performance - Proposed Design Flow



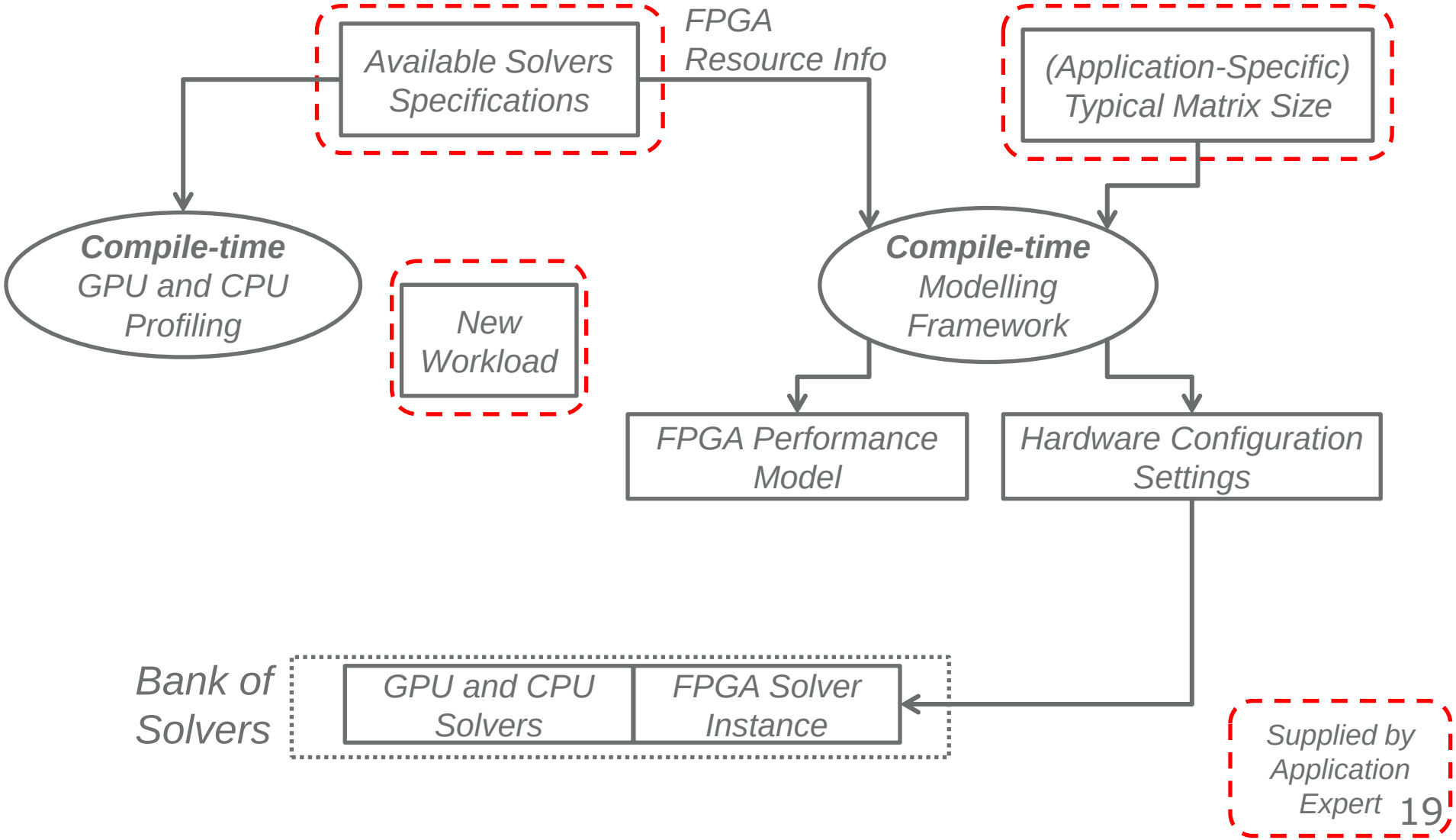
Heterogeneity for Performance - Proposed Design Flow



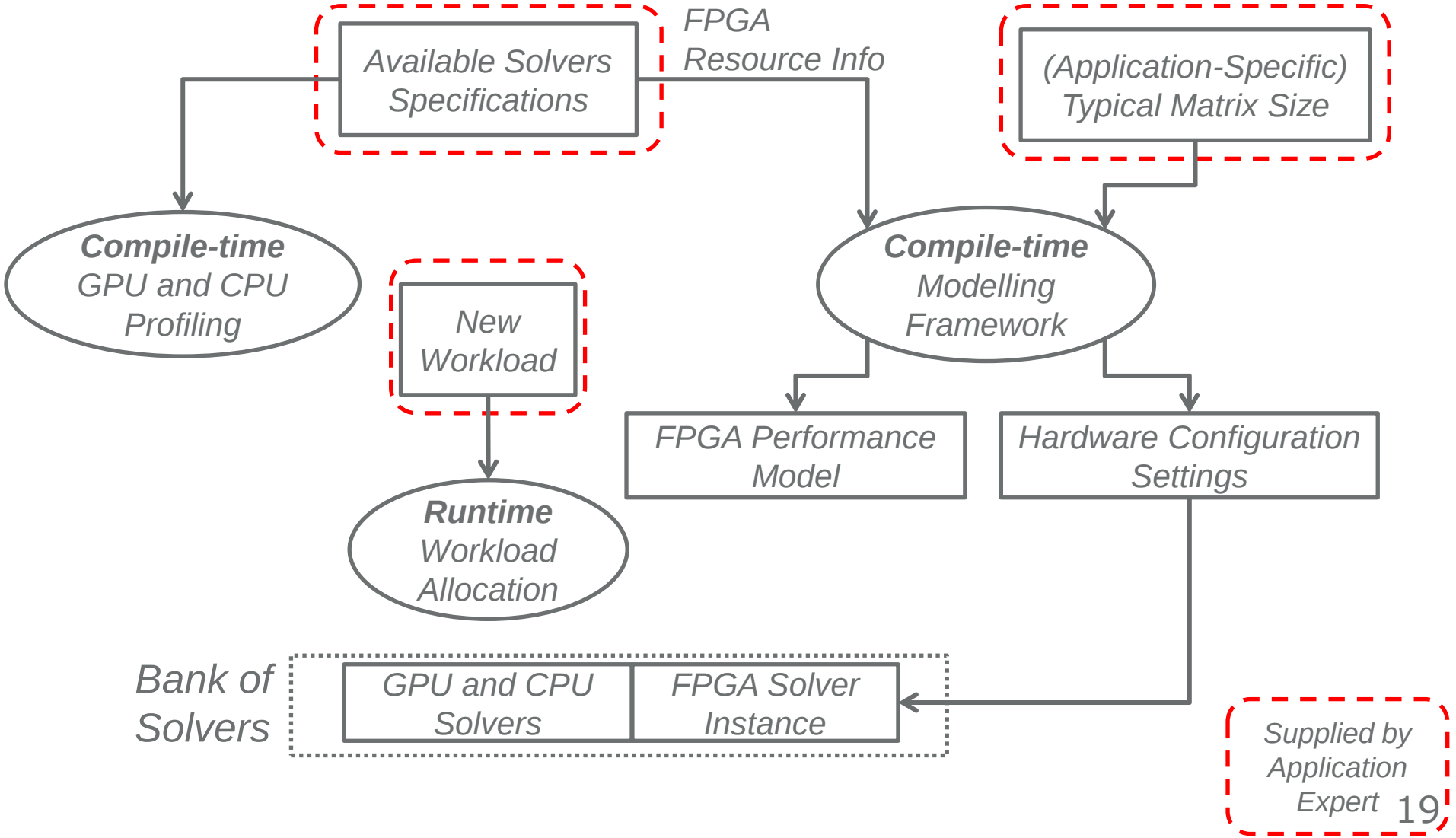
Heterogeneity for Performance - Proposed Design Flow



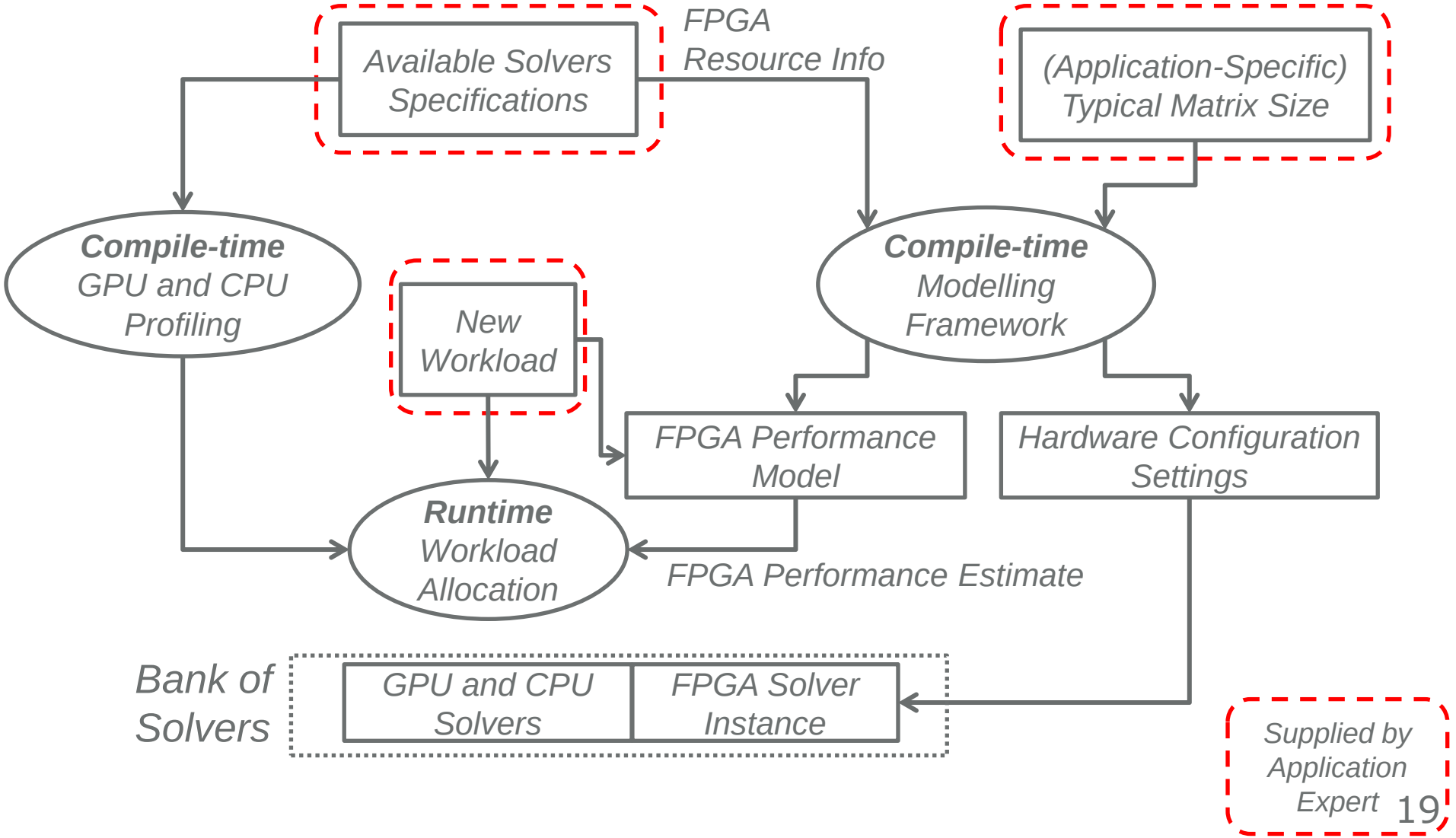
Heterogeneity for Performance - Proposed Design Flow



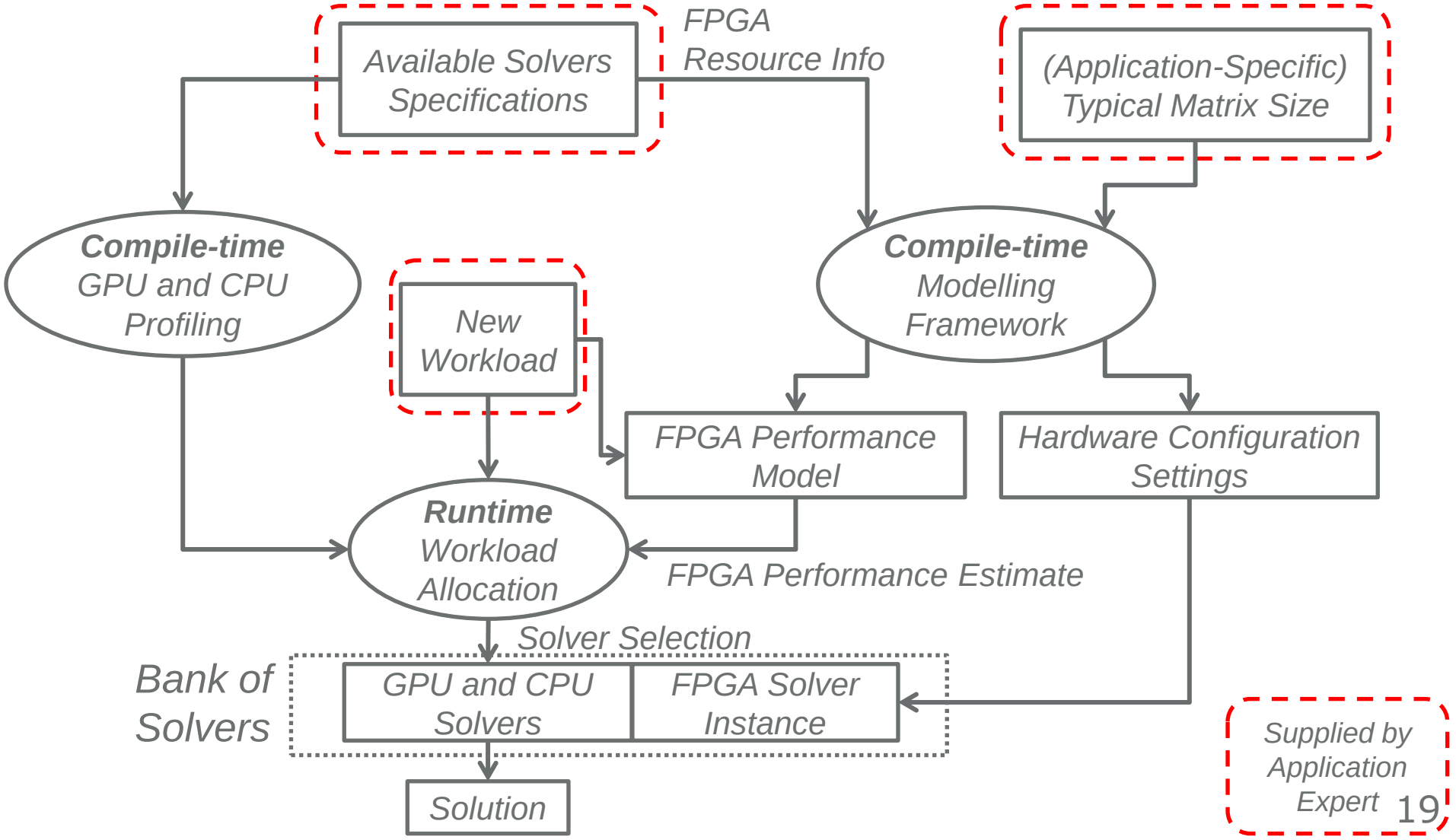
Heterogeneity for Performance - Proposed Design Flow



Heterogeneity for Performance - Proposed Design Flow



Heterogeneity for Performance - Proposed Design Flow

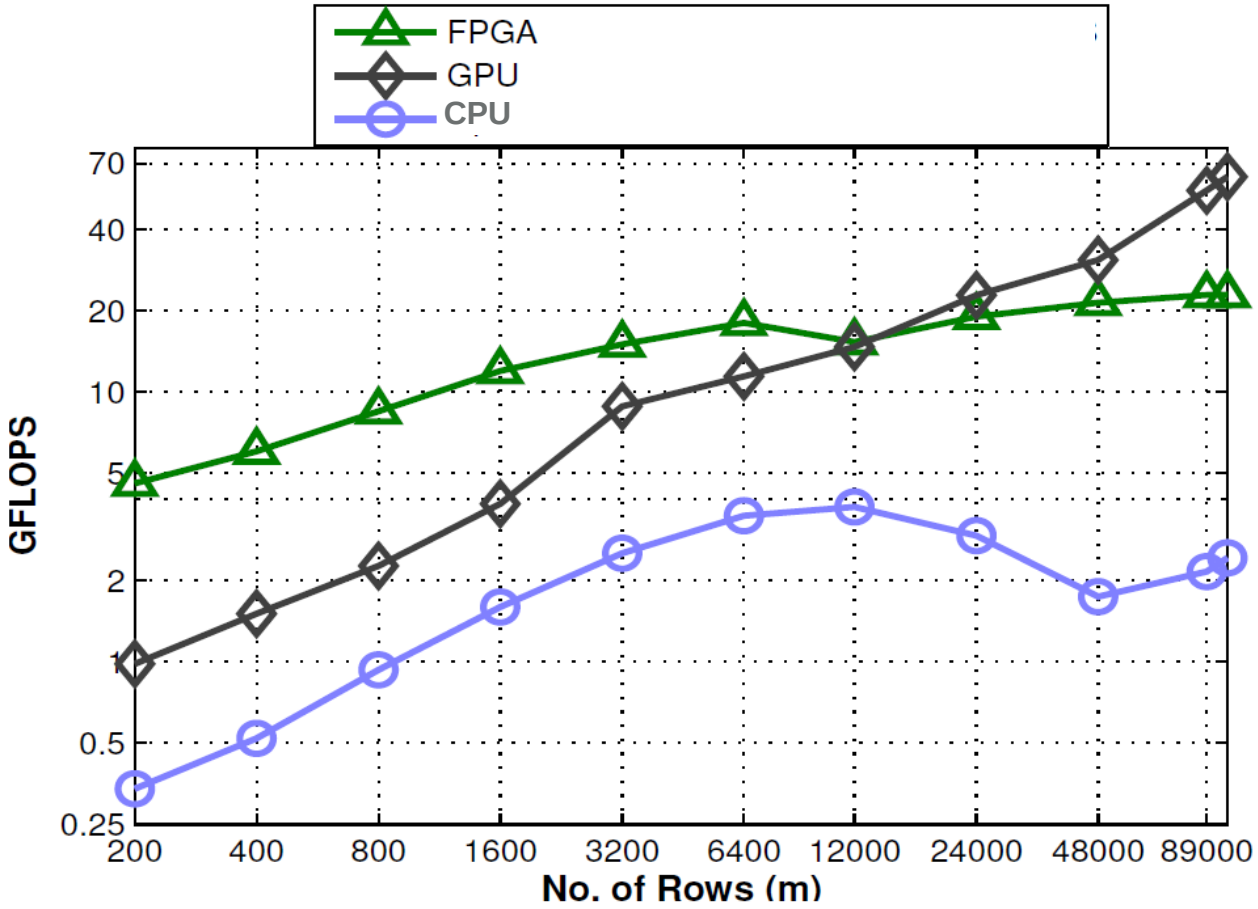


Evaluation

Experimental Setup

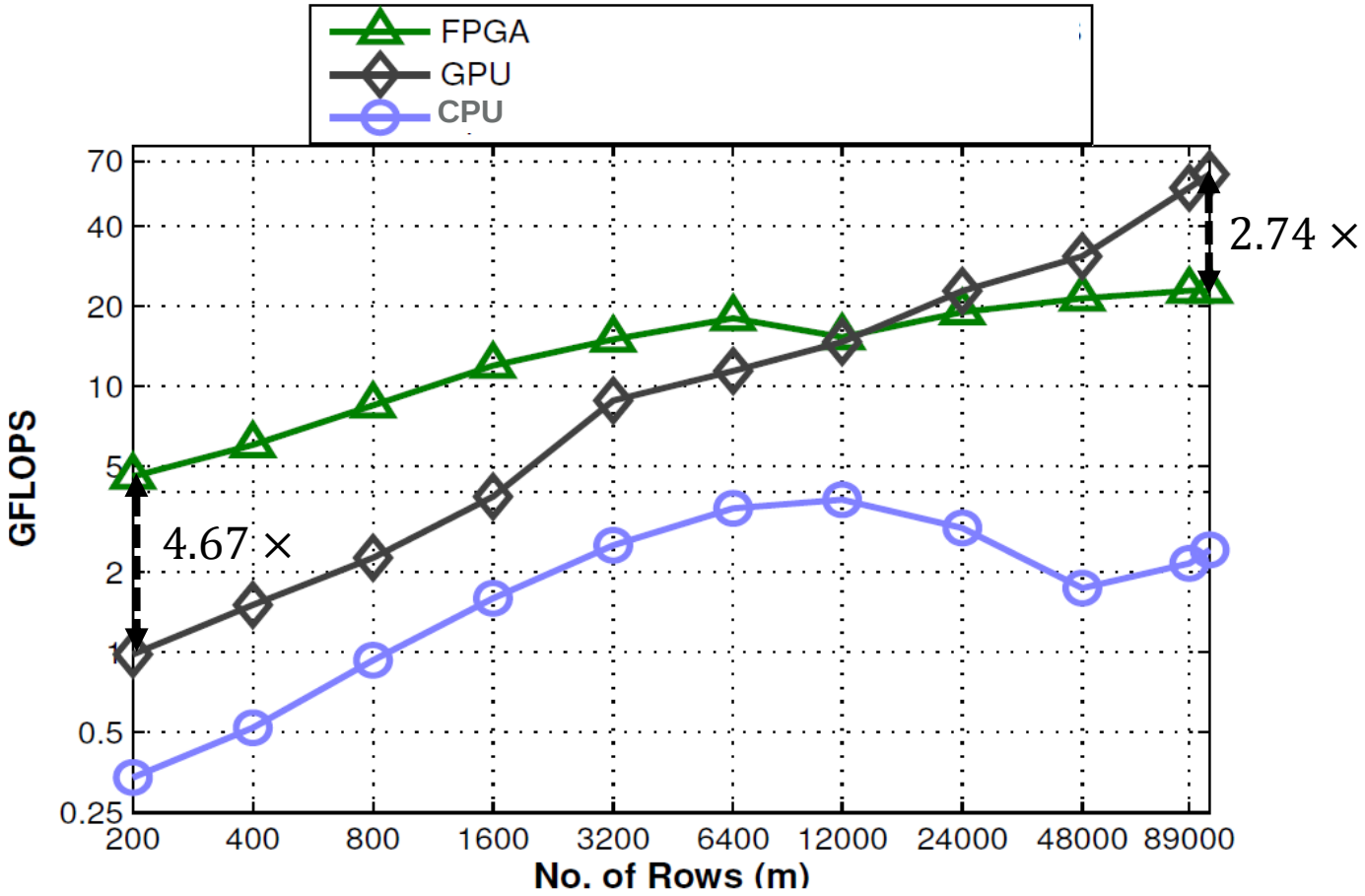
- *FPGA*
  - *Xilinx Virtex-6 SX475 at 200 MHz with 2016 DSPs*
  - *Double-precision floating-point*
  - *Up to 275 columns ($n = 275$) – 5x the max no. of columns of existing FPGA work*
  - *Any number of rows*
  - *84.23% BRAM utilization – post place-and-route*
  - *99.45% DSP utilization – post place-and-route*
- *GPU*
  - *NVIDIA Tesla K20*
  - *2496 cores at 706 MHz*
- *CPU*
  - *Intel i7-4770 at 3.40 GHz, 16 GB RAM, 8 MB cache*
  - *4 cores, 8 threads*

Internal Comparisons



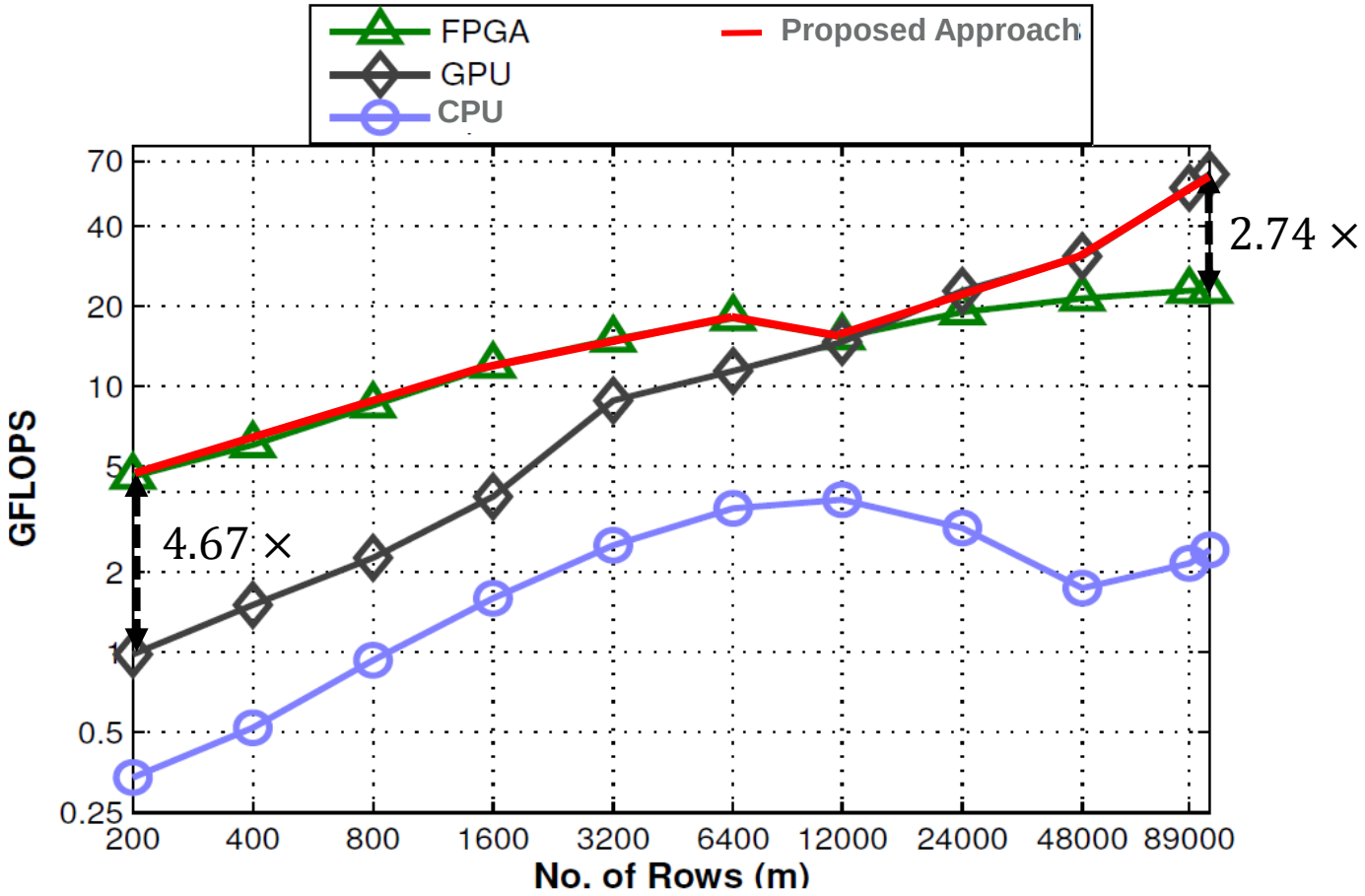
No. of Columns $(n) = 51$

Internal Comparisons



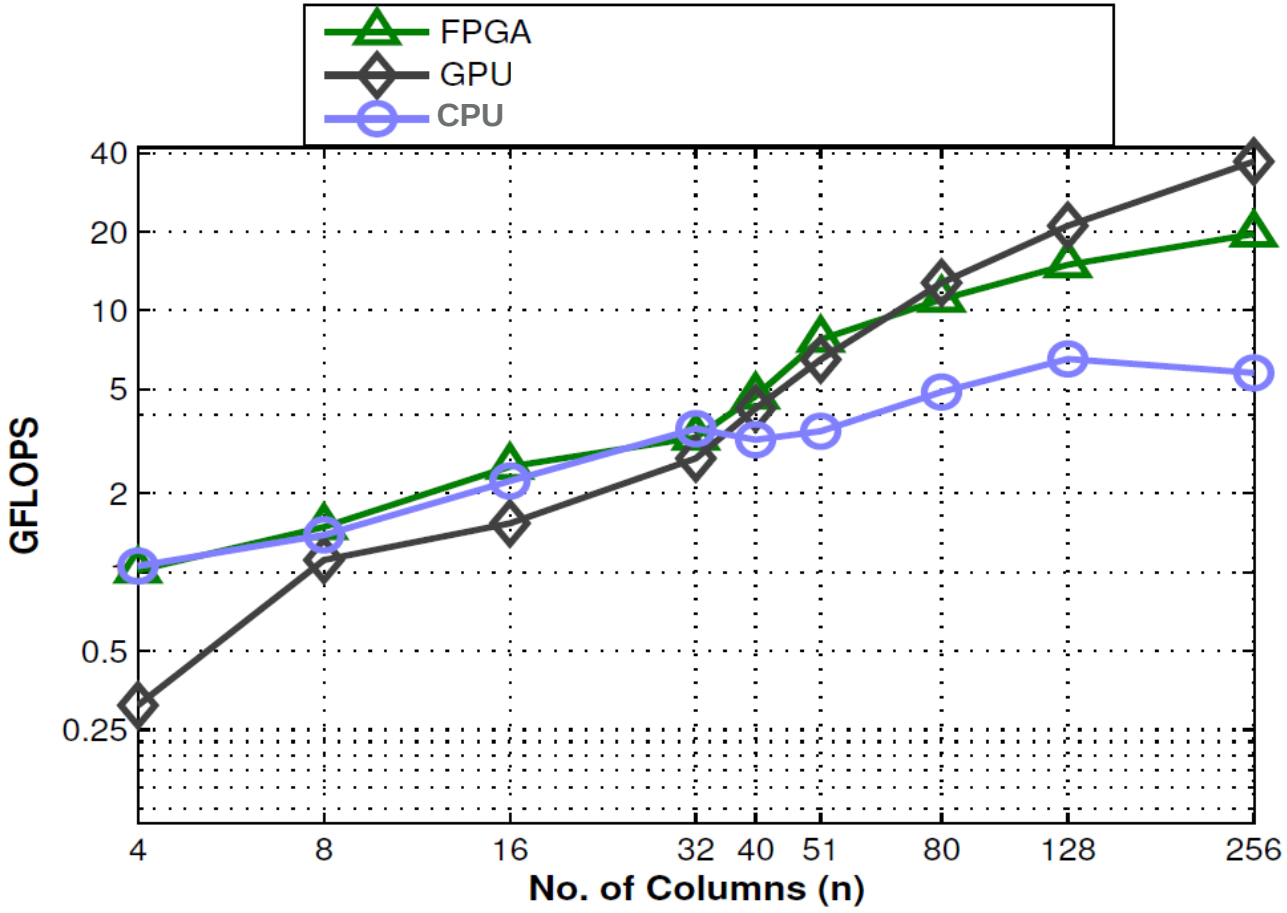
No. of Columns $(n) = 51$

Internal Comparisons



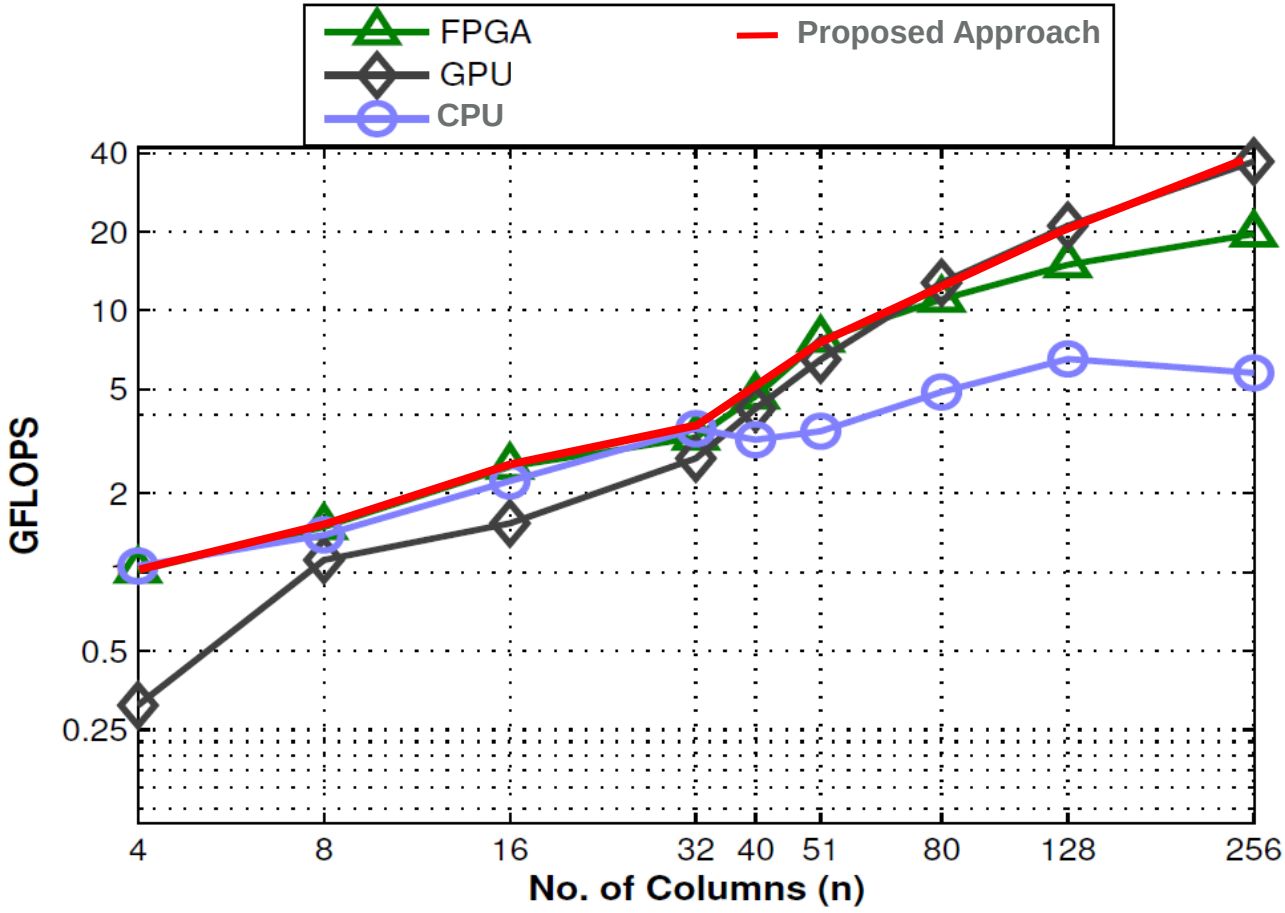
No. of Columns $(n) = 51$

Internal Comparisons



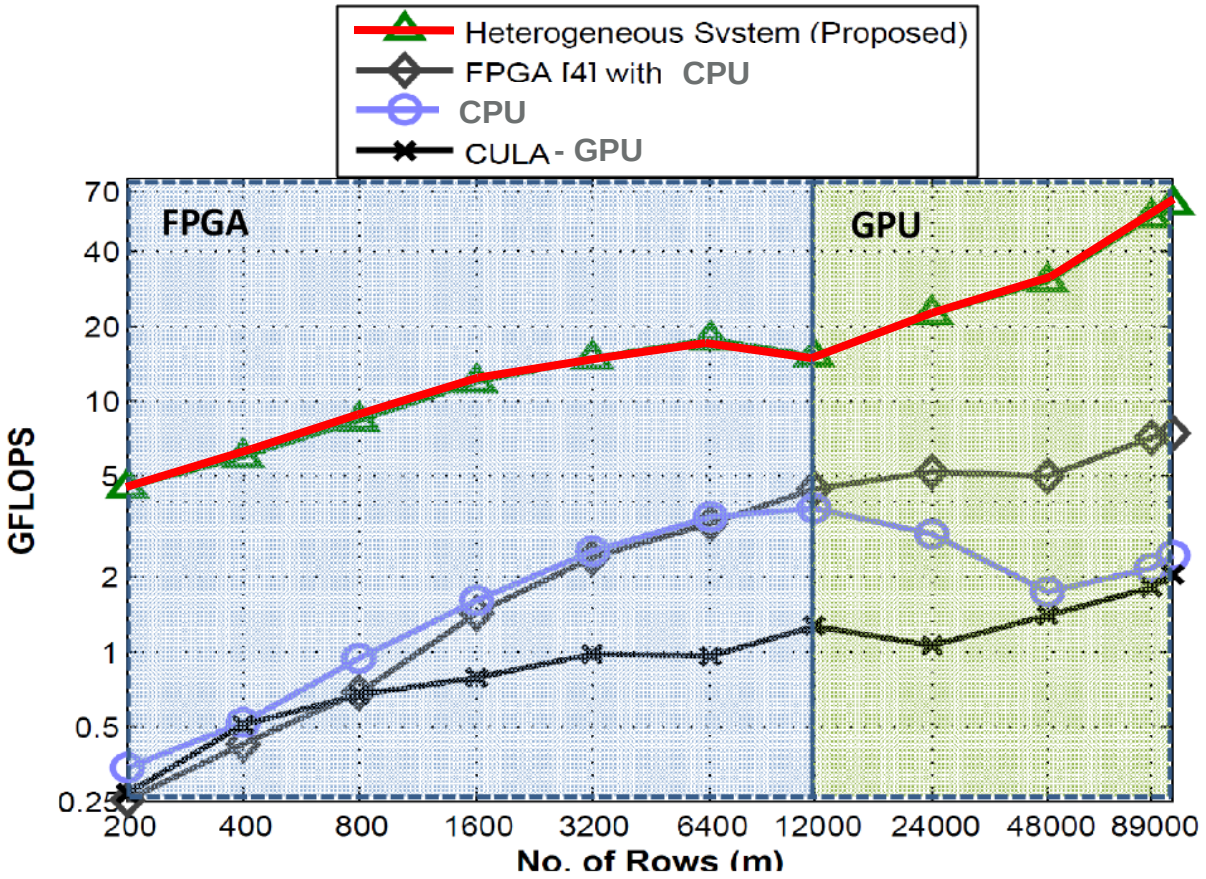
No. of Rows $(m) = 6400$

Internal Comparisons



No. of Rows $(m) = 6400$

External Comparisons



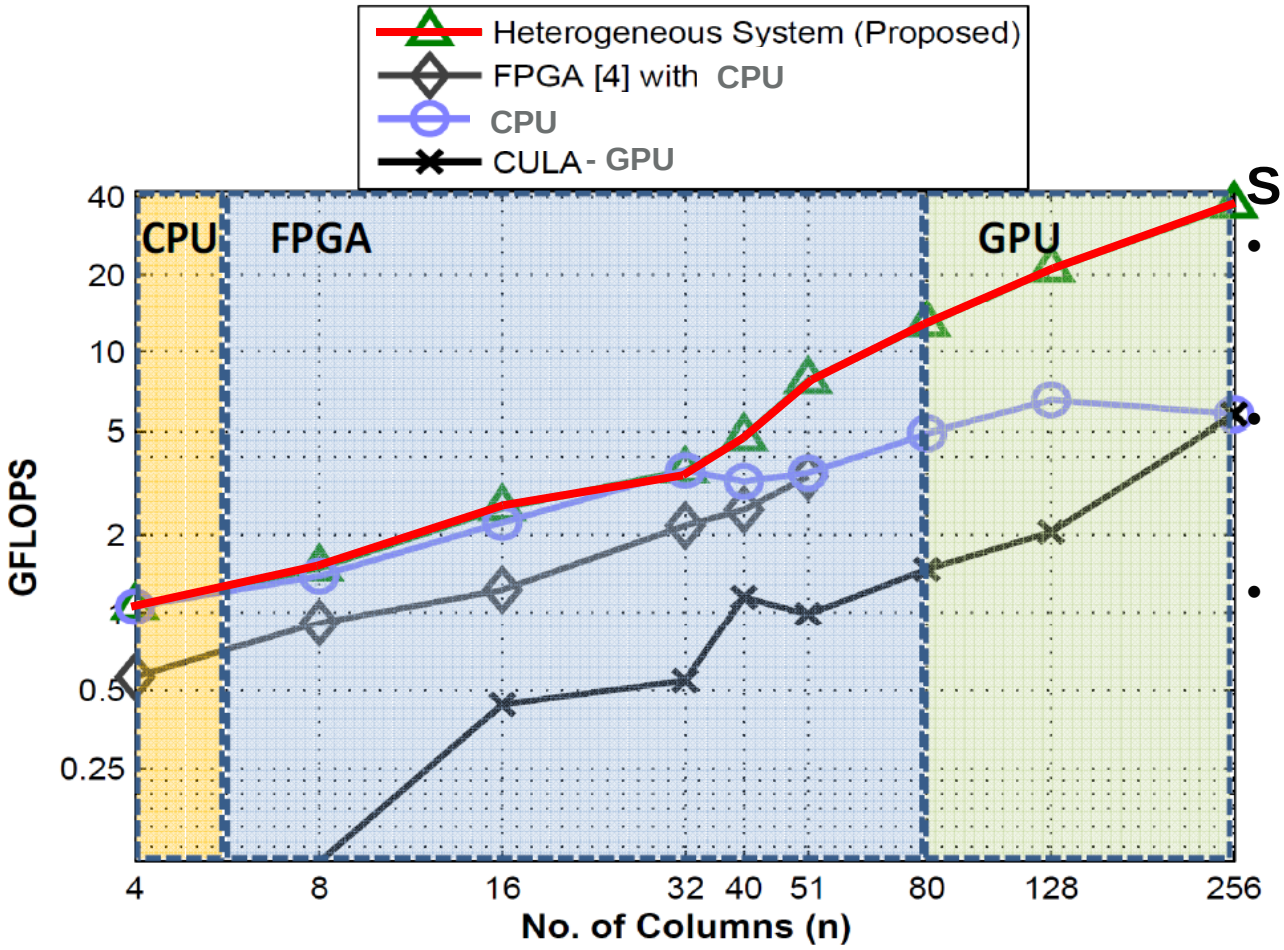
Speedup

- Up to 25.84x against software
- Up to 32.67x against CULA
- Up to18.07x against FPGA

No. of Columns $(n) = 51$

[4] A. Rafique et al., FPL 2012.

External Comparisons



Speedup

- Up to 25.84x against software
- Up to 32.67x against CULA
- Up to18.07x against FPGA

No. of Rows $(m) = 6400$

[4] A. Rafique et al., FPL 2012.

Conclusions

- Different solvers perform better on different matrix sizes
- Using heterogeneous solvers in a complementary way enable the high-performance solution of complex problems in fields such as genetic analysis

Thank You & Questions ?