Oak Ridge National Laboratory

International Workshop on Parallel and Distributed Algorithms for Decision Sciences (PDADS) 2021

Design Considerations for GPU-based Mixed Integer Programming on Parallel Computing Platforms

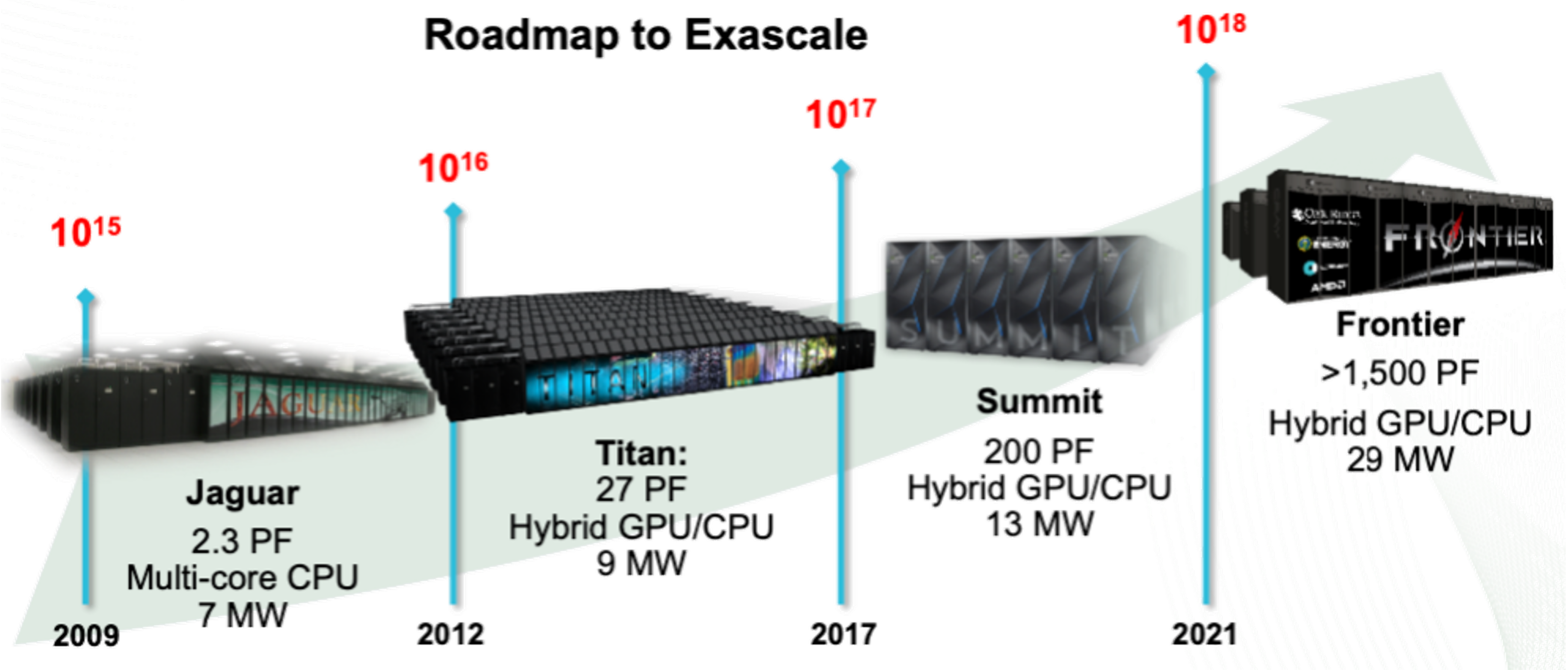
Kalyan Perumalla

Maksudul Alam

U.S. DEPARTMENT OF ENERGY

Int'l Conference on Parallel Processing (ICPP) 2021
August 9-12, 2021

GPUs enable much of the recent supercomputing power

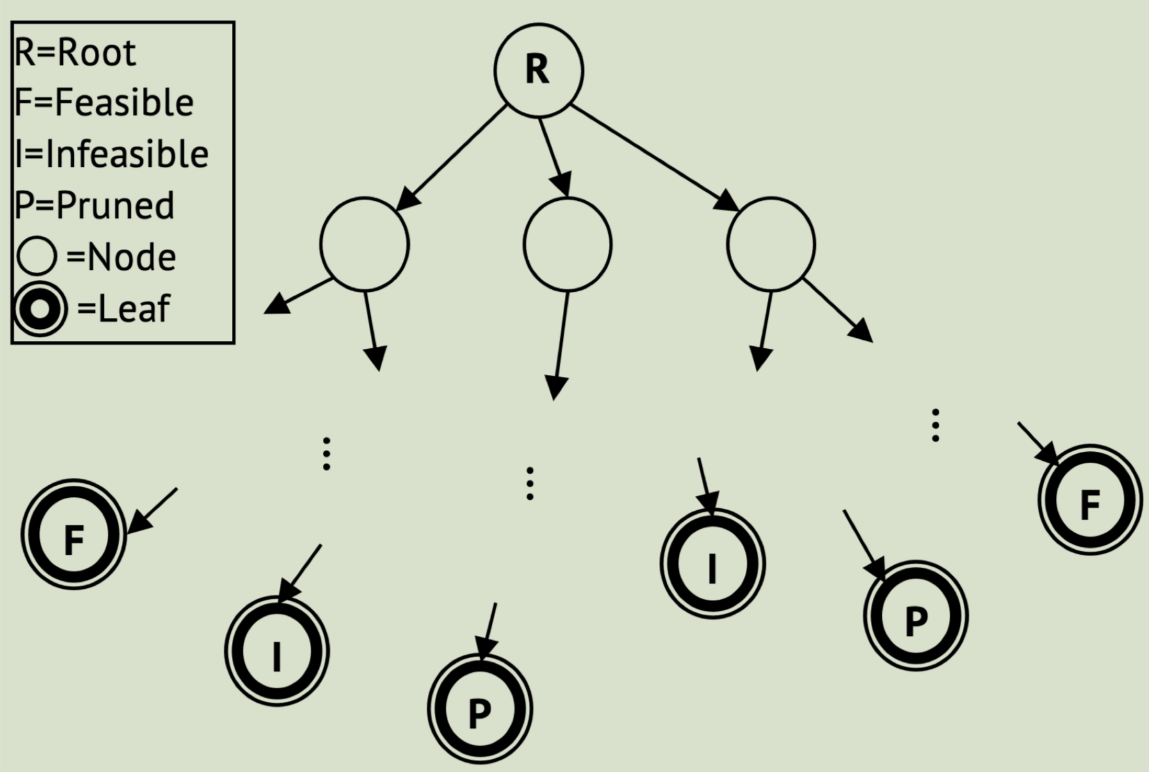


Mixed Integer Programming (MIP)

General Problem Formulation

$$\text{Maximize } c^T x$$
$$\text{such that } Ax \leq b,$$
$$\text{where } x = \{x_r, x_z\},$$
$$x_r \in \mathcal{R} \text{ (reals), and}$$
$$x_z \in \mathcal{Z} \text{ (integers).}$$

Basic Solution Approach



CPU vs. GPU-based Parallel MIP Solvers

CPU-based

- Fairly mature technology
- Many open-source implementations available
- Many commercial packages available
- Extremely fast solvers based on advanced
  - Linear algebra
  - Branch-and-Cut/Price
  - Heuristics
- References are provided in our paper

GPU-based

- Very few available to exploit the power of latest parallel processing
- Technical solution approaches are yet to be fully unraveled
- Need to guide the field with design considerations
  - To evaluate different choices
  - To determine most promising approach(es)

OAK RIDGE National Laboratory

Our Identification of GPU-based Parallel Execution Strategies

Entirely GPU-based

- Entire solution tree stored in GPU memory
- Tree updated by GPU only
- Branch-and-cut algorithm performed on GPU
- Linear algebra steps performed on GPU

CPU-driven GPU

- Entire solution tree stored in main memory
- Tree updated by CPU only
- Branch-and-cut algorithm performed by CPU
- Linear algebra steps delegated to GPU

Hybrid CPU-GPU

- Solution tree split across main memory and GPU memory
- Tree updated by GPU and CPU
- Both CPU and GPU participate as peers in branch-and-cut
- CPU & GPU perform linear algebra steps

Distributed Big-MIP

- Solution tree in small number of main node memories
- Tree updated by CPU only
- Branch-and-cut algorithm by lead-CPU orchestration
- Each linear algebra step on many GPUs

Optimal MIP Problem Size

Most effective GPU execution = Each tree node occupies one GPU memory

Small MIP Problem Size

Specialized GPU execution = Multiple tree nodes fit simultaneously in each GPU

Huge MIP Problem Size

Every tree node spans many GPUs

Linear Algebra Support

Software Considerations

- Matrix packages on GPU
  - **Dense matrices**: fairly mature on NVIDIA platforms
  - **Sparse matrices**: not as mature on any platform
  - Not easy to choose between dense and sparse, statically or dynamically
- GPUs are extremely efficient in dense linear algebra
  - Interior point methods for LP relaxation in branch-and-cut tree may work better

Algorithmic Considerations

- Sharing/reusing solutions across tree nodes
  - E.g., initial vector in iterative solvers
- Incorporation of generated cuts into GPU matrix structure
- Concurrent solution of small problems
- Solving multiple nodes simultaneously on same GPU

References are provided in the paper

Summary and Future Work

- GPUs dominate the current and future parallel processing
- Mixed integer programming (MIP) forms the core of several important applications
- Parallel MIP on GPU-based parallel platforms is not adequately understood
- Here, we unraveled key design considerations towards efficient execution of MIP on GPU-based parallel platforms
- Implementation of the most promising designs in actual software is needed as next step

Thank you for your attention!

- Q&A