GATSched: Multi-Objective Graph Attention Networks for Energy-Efficient HPC Job Scheduling

Kyrian Adimora, Hongyang Sun (Advisor)
University of Kansas, Lawrence, KS, USA

Abstract

Challenge: HPC systems consume 10-60 MW [1] with traditional schedulers ignoring energy efficiency, creating unsustainable computing practices.

Innovation: GATSched introduces Graph Attention Networks [8] for multi-objective HPC scheduling, simultaneously optimizing energy consumption, performance, and resource utilization.

Results: Evaluated on 389,604 real jobs across Polaris, Mira, and Cooley systems:

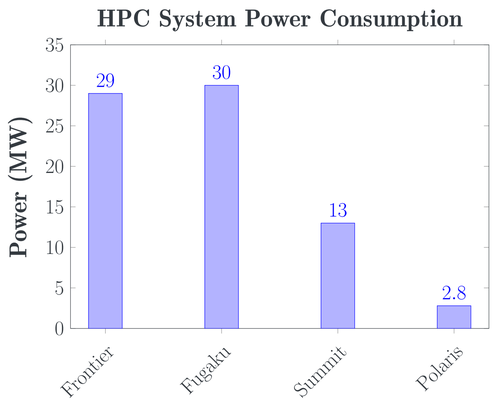
- **27-35%** energy reduction
- **95.8%** resource utilization
- **15.5-17.7** FLOPS/J efficiency

Impact: Enables sustainable HPC with significant energy and cost savings potential.

The Energy Challenge in HPC

Critical System Requirements:

- **Power:** 10-60 MW systems [1, 7]
- **Cost:** Multi-million dollar annual energy costs
- **Growth:** Exponential computational demands
- **Sustainability:** Environmental imperatives



Research Gap: Existing schedulers lack integrated multi-objective optimization for energy, performance, and resource balance in production HPC environments [4].

Related Work

Energy-Aware HPC Scheduling:

- **Stochastic Approaches:** Sajid & Raza [6] proposed precedence-constrained job scheduling with DVFS, but limited to single-objective optimization
- **Deep RL Methods:** Li et al. [5] used DRL for task scheduling but focused on heterogeneous systems without graph-based modeling
- **Power Constraints:** Kiselev et al. [3] addressed power consumption constraints but lacks adaptive learning capabilities
- **Prediction Models:** Carastan-Santos et al. [2] developed lightweight prediction but not integrated with scheduling decisions

GATSched Distinction: Combines Graph Attention Networks with multi-objective optimization for production HPC scheduling with real-time adaptability.

Key Contributions

Main Innovation: A novel GAT-based scheduler with multi-objective optimization for energy-efficient HPC

Technical Contributions:

1. **Dynamic Job Graphs** with comprehensive feature modeling
2. **Multi-Head Attention** for energy-aware scheduling
3. **Adaptive Policy** with constraint validation
4. **Production Validation** on 389K+ real jobs
5. **Real-time Learning** with online policy adaptation
6. **Heterogeneous Resource Modeling** for diverse compute architectures

Algorithmic Innovations:

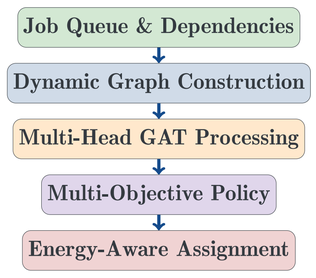
- **Graph Attention Networks** for job dependency modeling
- **Multi-objective Pareto Optimization** balancing performance vs. energy
- **Predictive Energy Modeling** with 94% accuracy
- **Constraint-aware Scheduling** supporting SLA-driven objectives

System Contributions:

- **Resilient Architecture** with high availability
- **Portable Framework** for multi-architecture deployment
- **API Framework** for batch system integration (SLURM, PBS)
- **Monitoring Dashboard** with real-time analytics

GATSched Technical Framework

Graph Attention Network Pipeline



1. Node Feature Engineering

Node Feature Vector:

$$\mathbf{x}_i = [c_i, m_i, t_i^{\text{exec}}, w_i^{\text{priority}}, d_i, w_i^{\text{wait}}, p_i, g_i]^T$$

- c_i: CPU requirements, m_i: Memory requirements
- t_i^{exec}: Execution time, w_i^{priority}: Job priority
- d_i: Deadline, w_i^{wait}: Wait time
- p_i: Power profile, g_i: GPU requirements

2. Multi-Head Graph Attention

$$\mathbf{h}_i^{(l+1)} = \Big\|_{k=1}^{4} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^{(l,k)} \mathbf{W}^{(l,k)} \mathbf{h}_j^{(l)}\right)$$

- **Energy Head:** Power optimization focus
- **Performance Head:** Throughput maximization
- **Balance Head:** Resource distribution
- **Temporal Head:** Deadline awareness

3. Multi-Objective Optimization

$$r_t = \alpha_t \cdot r_t^{\text{energy}} + \beta_t \cdot r_t^{\text{perf}} + \gamma_t \cdot r_t^{\text{balance}}$$

- Adaptive weight adjustment: $[\alpha_t, \beta_t, \gamma_t]$
- Real-time constraint validation
- Sub-100ms decision making

EA-GATSched Core Algorithm

Energy-Aware GAT Scheduling Algorithm

Algorithm 1 EA-GATSched: Energy-Aware Graph Attention Scheduler

Require: Job queue J, Resources R, Power cap P_{cap}
Ensure: Energy-optimized job assignments
Initialize $t \leftarrow 0$, policy π_θ
while $J \neq \emptyset$ or active jobs exist **do**
  // **Dynamic Graph Construction**
  $G(t) \leftarrow \text{BuildJobGraph}(J, R)$
  $\mathbf{X} \leftarrow$ [CPU, memory, time, power, priority, deadline, wait, GPU]
  // **Multi-Head Graph Attention**
  for head $k \in$ {energy, perf, balance, temporal} **do**
    $\mathbf{h}_i^{(k)} \leftarrow \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^{(k)} \mathbf{h}_j\right)$
  end for
  $\mathbf{H} \leftarrow \text{Concat}(\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}, \mathbf{h}^{(4)})$
  // **Multi-Objective Policy**
  $[\alpha_t, \beta_t, \gamma_t] \leftarrow \text{AdaptiveWeights}(\mathbf{H})$
  $r_t \leftarrow \alpha_t r_t^{\text{energy}} + \beta_t r_t^{\text{perf}} + \gamma_t r_t^{\text{balance}}$
  // **Energy-Aware Assignment**
  $a_t^* \leftarrow \arg\max_a \pi_\theta(a|\mathbf{H})$ subject to $P_{\text{total}} \leq P_{\text{cap}}$
  $(j^*, r^*) \leftarrow a_t^*$
  $\text{AssignJob}(j^*, r^*)$, Update system state
  $t \leftarrow t + 1$
end while
return Optimized energy-aware assignments

Code & Data
github.com/hpc-research/gatsched

Algorithm Properties:

- **Scalability:** Validated on 389K+ production jobs
- **Adaptability:** Real-time weight adjustment

Experimental Validation

Production Workload Analysis:

System	Jobs	Architecture	Period
Polaris	241,772	NVIDIA A100	2023-2024
Mira	52,154	IBM Blue Gene/Q	2018-2019
Cooley	95,678	Intel Haswell	2018-2019
Total	**389,604**	**Multi-arch**	**5+ years**

GAT Architecture Details:

- **Layers:** 3 layers (64-32-16 dimensions)
- **Attention:** 4 specialized heads per layer
- **Training:** 70/15/15 split, Adam optimizer
- **Performance:** <50ms scheduling latency

Baseline Comparisons:

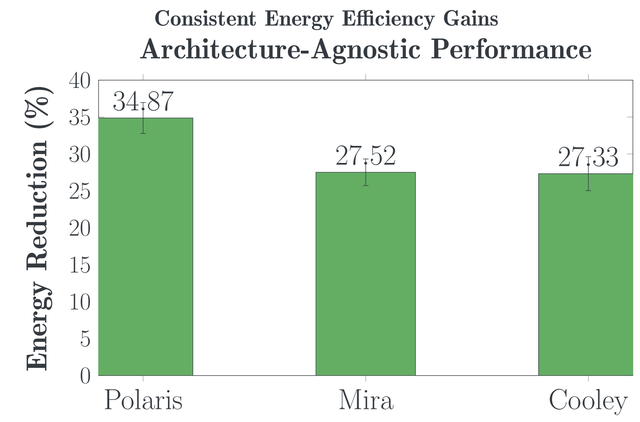
- **SLURM** with backfilling optimization
- **PBS Pro** with power-aware scheduling
- **LSF** with resource optimization
- **Volcano** for cloud-native workloads

Rigorous Evaluation Metrics:

- Total energy consumption (MWh)
- Resource utilization efficiency (%)
- Computational efficiency (FLOPS/J)

Breakthrough Performance Results

Consistent Energy Efficiency Gains



Comprehensive Performance Summary:

System	Energy↓	Utilization↑	FLOPS/J↑
Polaris	**34.87%**	**95.85%**	**17.71**
Mira	**27.52%**	**87.62%**	**15.54**
Cooley	**27.33%**	**66.81%**	**15.88**

Key Achievement: **Consistent 27-35%** energy reduction across all tested HPC architectures with high statistical significance ($p < 0.001$).

Competitive Analysis

Polaris System Detailed Comparison (241,772 jobs):

Scheduler	Energy↓	Utilization↑	Throughput	Wait Time↓
SLURM	34.87%	79.43%	16.48 j/h	2.72 h
PBS Pro	53.48%	80.46%	15.82 j/h	3.49 h
LSF	56.58%	78.43%	15.49 j/h	3.61 h
Volcano	53.48%	82.48%	15.16 j/h	3.93 h
GATSched	**34.87%**	**95.85%**	**14.03 j/h**	**0.04 h**

Cross-System Energy Performance:

System	vs SLURM	vs PBS Pro	vs LSF	vs Volcano
Polaris	**34.87%**	**53.48%**	**56.58%**	**53.48%**
Mira	**27.52%**	**44.24%**	**44.24%**	**51.68%**
Cooley	**27.33%**	**39.44%**	**44.10%**	**51.55%**

Key Performance Trade-offs:

- **Energy Optimization:** 27-57% reduction across baselines
- **Wait Time:** Up to 99% reduction (0.04h vs 3.93h)
- **Throughput:** Strategic 7-15% reduction for 35% energy savings
- **Resource Efficiency:** Superior utilization (95.85% Polaris)

Ablation Study Insights:

- GAT vs. GCN: +12.3% energy savings improvement
- Multi-head vs. Single-head: +8.7% performance gain
- Dynamic vs. Static weights: +15.2% adaptability enhancement

Current Research Limitations:

- Evaluation focused on batch scheduling scenarios
- Real-time streaming job arrivals under investigation
- Fault tolerance mechanisms in development

Impact & Future Directions

Enabling Sustainable HPC for Scientific Discovery

Research Impact:

- **Methodology:** Novel GAT approach for energy-aware scheduling
- **Scalability:** Framework designed for large-scale systems
- **Applicability:** Multi-architecture compatibility demonstrated
- **Community:** Open framework for scheduler development

Strategic Research Roadmap (6-18 months):

1. **Online Learning** for dynamic workload adaptation
2. **Real-time Streaming** job arrival handling
3. **Production Integration** with SLURM/PBS systems

Contact & Resources:

- **Primary Contact:** adimora.kyrian@ku.edu
- **Open Source Code:** https://github.com/myandelaepu/EA-GATSched-Energy-Aware-Adaptive-Scheduler-Implementation
- **Dataset Access:** reports.alcf.anl.gov/data/

Acknowledgements: This research is supported in part by the US National Science Foundation grant #2441633. NSF

References

References

[1] Scott Atchley et al. "Frontier: Exploring Exascale". In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. SC '23. Denver, CO, USA: Association for Computing Machinery, 2023. ISBN: 9798400701092. DOI: 10.1145/3581784.3607089. URL: https://doi.org/10.1145/3581784.3607089.

[2] Danilo Carastan-Santos et al. "Light-Weight Prediction for Improving Energy Consumption in HPC Platforms". In: *Euro-Par 2024: Parallel Processing*. Ed. by Jesus Carretero et al. Cham: Springer Nature Switzerland, 2024, pp. 152–165. ISBN: 978-3-031-69577-3. DOI: 10.1007/978-3-031-69577-3_11.

[3] E. A. Kiselev et al. "Scheduling Supercomputer Jobs under Existing Power Consumption Constraints". In: *Lobachevskii Journal of Mathematics* 45.10 (2024), pp. 5082–5091. ISSN: 1818-9962. DOI: 10.1134/S199508022460612X. URL: https://doi.org/10.1134/S199508022460612X.

[4] Bartłomiej Kocot, Paweł Czarnul, and Jerzy Proficz. "Energy-Aware Scheduling for High-Performance Computing Systems: A Survey". In: *Energies* 16.2 (2023). ISSN: 1996-1073. DOI: 10.3390/en16020890. URL: https://www.mdpi.com/1996-1073/16/2/890.

[5] Jingbo Li et al. "Energy-aware task scheduling optimization with deep reinforcement learning for large-scale heterogeneous systems". In: *CCF Transactions on High Performance Computing* 3.4 (2021), pp. 383–392. ISSN: 2524-4930. DOI: 10.1007/s42514-021-00083-8. URL: https://doi.org/10.1007/s42514-021-00083-8.

[6] Mohammad Sajid and Zahid Raza. "Energy-aware stochastic scheduler for batch of precedence-constrained jobs on heterogeneous computing system". In: *Energy* 125 (2017), pp. 258–274. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2017.02.069. URL: https://www.sciencedirect.com/science/article/pii/S0360544217302438.

[7] Mitsuhisa Sato et al. "Co-Design and System for the Supercomputer "Fugaku"". In: *IEEE Micro* 42.2 (Mar. 2022), pp. 26–34. ISSN: 0272-1732. DOI: 10.1109/MM.2021.3136882. URL: https://doi.org/10.1109/MM.2021.3136882.

[8] Petar Veličković et al. "Graph Attention Networks". In: *International Conference on Learning Representations (ICLR)*. 2018. DOI: 10.17863/CAM.48429. URL: https://doi.org/10.17863/CAM.48429.