cuLocalMIP

Honorable Mention Award
2026 Land-Doig MIP Competition

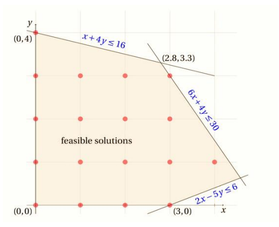
A GPU-Oriented Local Search Framework for Mixed Integer Programming

Peng Lin, Zhe Wang, Kewu Yang, and Shaowei Cai*
Institute of Software, Chinese Academy of Sciences, Beijing, China

Mixed Integer Programming

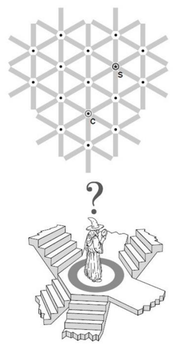
$$\min \left\{ \boldsymbol{c}^\top \boldsymbol{x} \mid \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b},\ \boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u},\ x_j \in \mathbb{Z}\ \forall j \in \mathcal{I} \right\}$$

- A foundational model for optimization
- Broad real-world applications
- Hard to solve due to discrete decisions
- Fast discovery of high-quality feasible solutions is crucial



Local Search

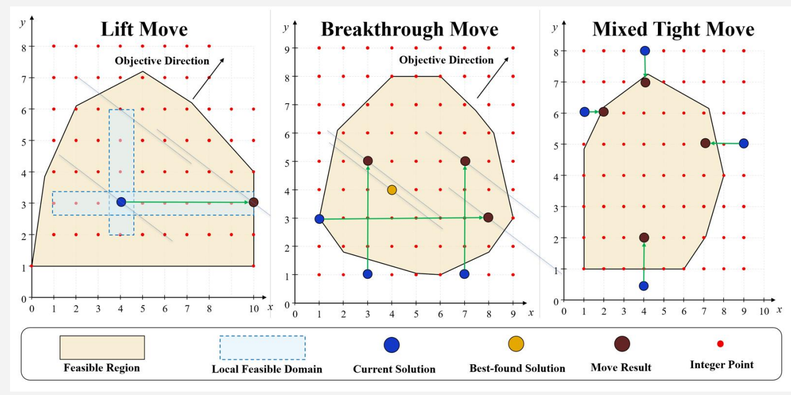
- Start from an initial solution
- Repeatedly explore neighboring solutions
- Operators define candidate moves
- A scoring function selects the next solution
- Effective for quickly finding high-quality feasible solutions for many optimization problems



Local-MIP: Local Search for General MIPs

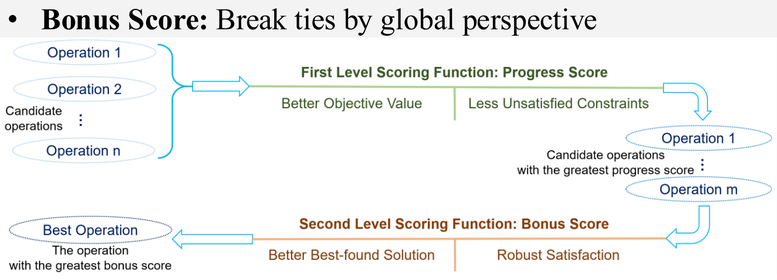
Three Move Operators

- **Lift Move**: Improves the objective within a local feasible domain while maintaining feasibility.
- **Breakthrough Move**: Moves toward an objective value better than the best-found solution.
- **Mixed Tight Move**: Moves a variable to make a constraint tight, reducing constraint violation with minimal perturbation.



Two-level Scoring Function

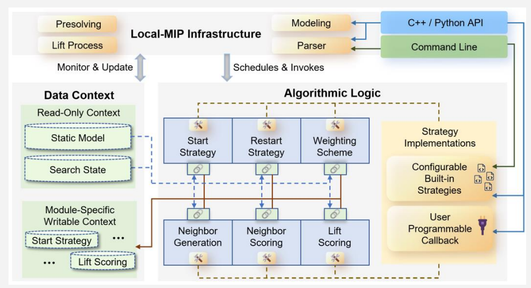
- **Progress Score**: Improve the current solution
- **Bonus Score:** Break ties by global perspective



Local-MIP 2.0

https://local-mip.github.io/

- Extensible Open-Source Library
- Customizable Callbacks
- C++/Python APIs



cuLocalMIP

1 Current Search State

- Current solution
- Objective value
- Constraint activities
- Satisfied & violated constraints

$$\min_{\mathbf{x}} \quad \mathbf{c}^\top \mathbf{x} \quad s.t. \quad \begin{aligned} &\mathbf{A}\mathbf{x} \le \mathbf{b}, \\ &\mathbf{l} \le \mathbf{x} \le \mathbf{u}, \\ &x_j \in \mathbb{Z} \quad \forall j \in I, \\ &\mathbf{x} \in \mathbb{R}^n. \end{aligned}$$

2 GPU PARALLEL Move Generation

- Generate many candidate moves in parallel from the current search state
- Operators act on different variables simultaneously to produce diverse moves.

Move 1 | Move 2 | ... | Move k

Move families (priority order)
mixed tight > breakthrough > lift

3 GPU PARALLEL Move Evaluation

- For each candidate move, the resulting changes in the objective and constraints can be computed independently,
- Move scoring is naturally data-parallel.

Obj score + Constraint 1 score ... Constraint m score

4 GPU PARALLEL Update Search State

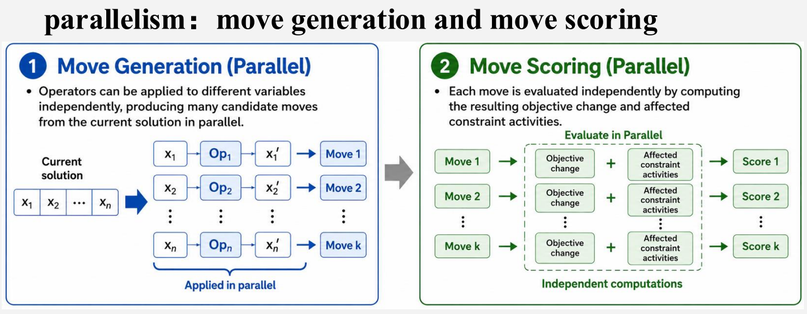
- Select the best-scoring move to apply
- Update the affected search state data in parallel.

New current solution → Constr 1 activities ... Constr m activities, Objective value

Repeat until time limit

Why GPU?

- Local-MIP-style search exposes two levels of **data parallelism: move generation and move scoring**

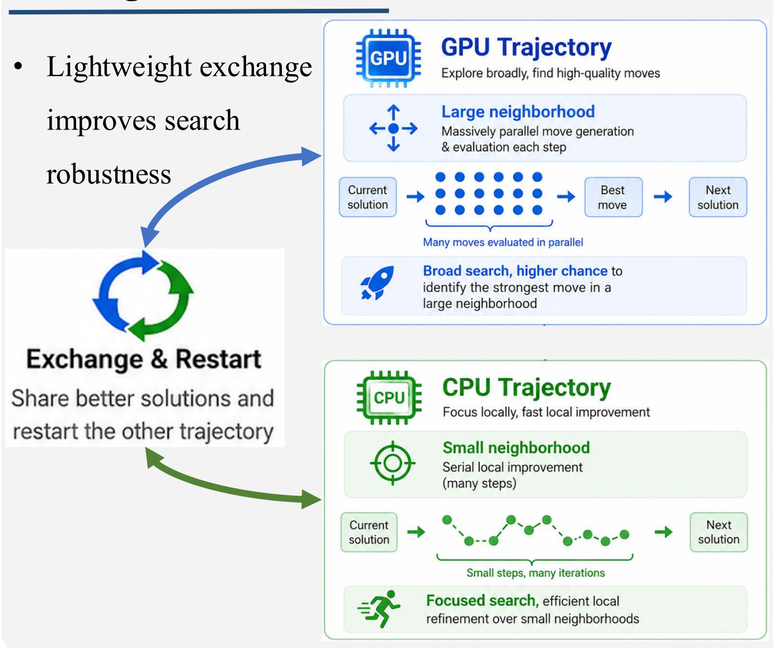


- GPUs are becoming a promising platform for mathematical optimization. Local-MIP's batched computation is a natural fit for GPUs.

Local-MIP → GPU PARALLEL

Heterogeneous Search

- Lightweight exchange improves search robustness



Conditional Node

- High-frequency search loop with small kernels.
- Conditional Nodes (While Node/ If Node) embed dynamic control flow into CUDA Graphs
- Reduces synchronization and redundant kernel execution

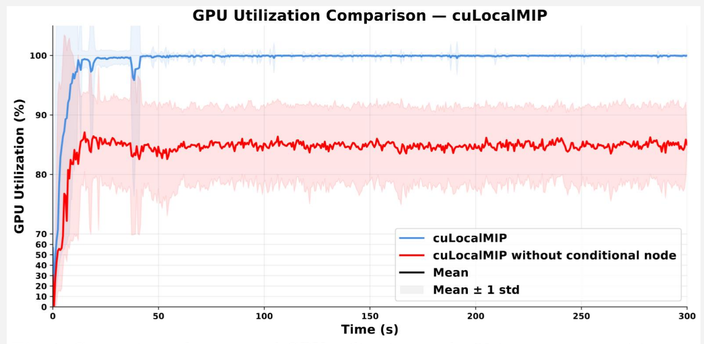


Fig. 1. Comparison of aggregated GPU utilization on the 50-instance competition test set, measured from the `utilization.gpu` field of `nvidia-smi`. The shaded regions show mean ± one standard deviation.

Computational Results

Public Benchmark Results

- Evaluated with a 5-minute time limit on the competition test set and MIPLIB 2017 using an NVIDIA V100 GPU, with CPU Local-MIP and multiple cuOpt variants as baselines.
- Experimental results show that cuLocalMIP substantially improves over CPU Local-MIP and achieves performance competitive with cuOpt full, cuOpt heur, and cuOpt FJ.

Table 1. Results on the MIPcc26 competition test set.

Metric	CPU Local-MIP	Gurobi-5m	$\text{cuOpt}_{\text{full}}$	$\text{cuOpt}_{\text{heur}}$	cuOpt_{FJ}	cuLocalMIP
#Feas	40.00	44.00	40.67	43.33	39.67	44.00
Avg Gap	0.5223	0.256	0.3667	0.3197	0.6916	0.3414
Avg PT	0.5605	–	0.5001	0.4639	0.7351	0.4132

Table 2. Results on MIPLIB 2017.

Metric	CPU Local-MIP	$\text{cuOpt}_{\text{full}}$	$\text{cuOpt}_{\text{heur}}$	cuOpt_{FJ}	cuLocalMIP
#Feas	195.00	216.00	214.00	195.00	212.00
Avg Gap	0.4738	0.1896	0.2270	0.5349	0.3160
Avg PT	0.4967	0.2827	0.3048	0.5561	0.3413

* Shaowei Cai is the corresponding author.

Mixed Integer Programming Workshop 2026

Contact: linpeng@ios.ac.cn, caisw@ios.ac.cn