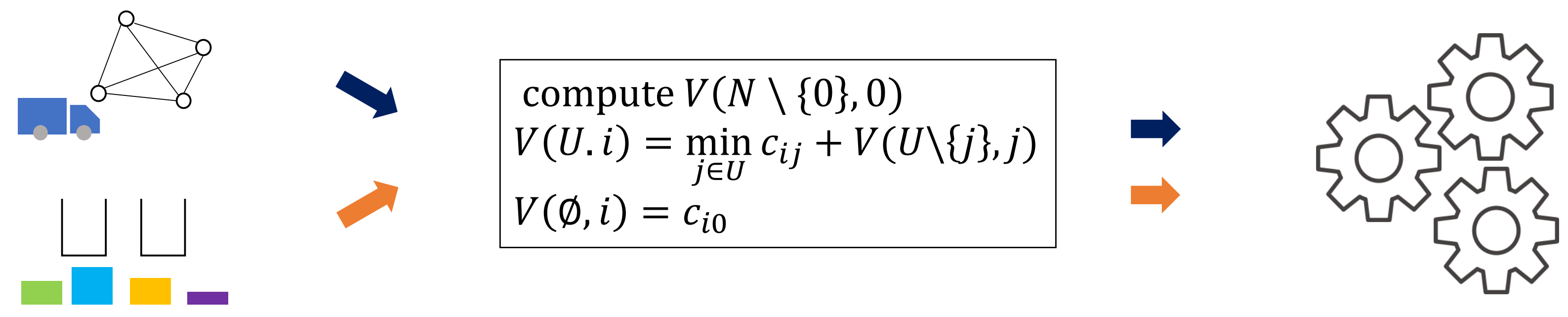
Parallel Beam Search Algorithms for Domain-Independent Dynamic Programming

Ryo Kuroiwa and J. Christopher Beck
Department of Mechanical and Industrial Engineering, University of Toronto

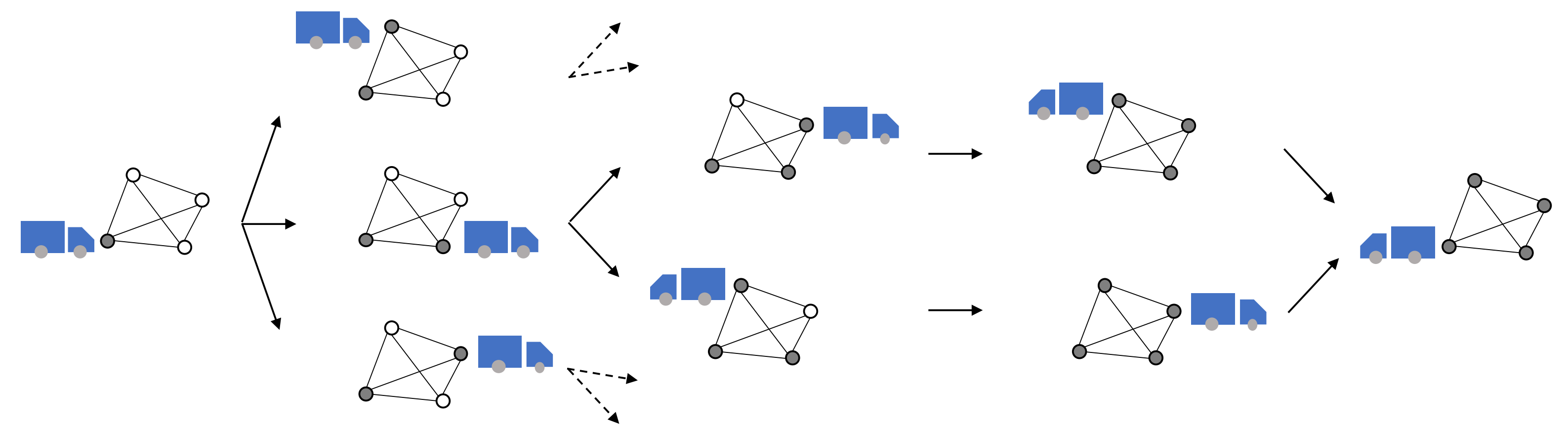
Domain-Independent Dynamic Programming (DIDP)

Combinatorial Optimization Problem **DP Model** **General-Purpose DP Solver**

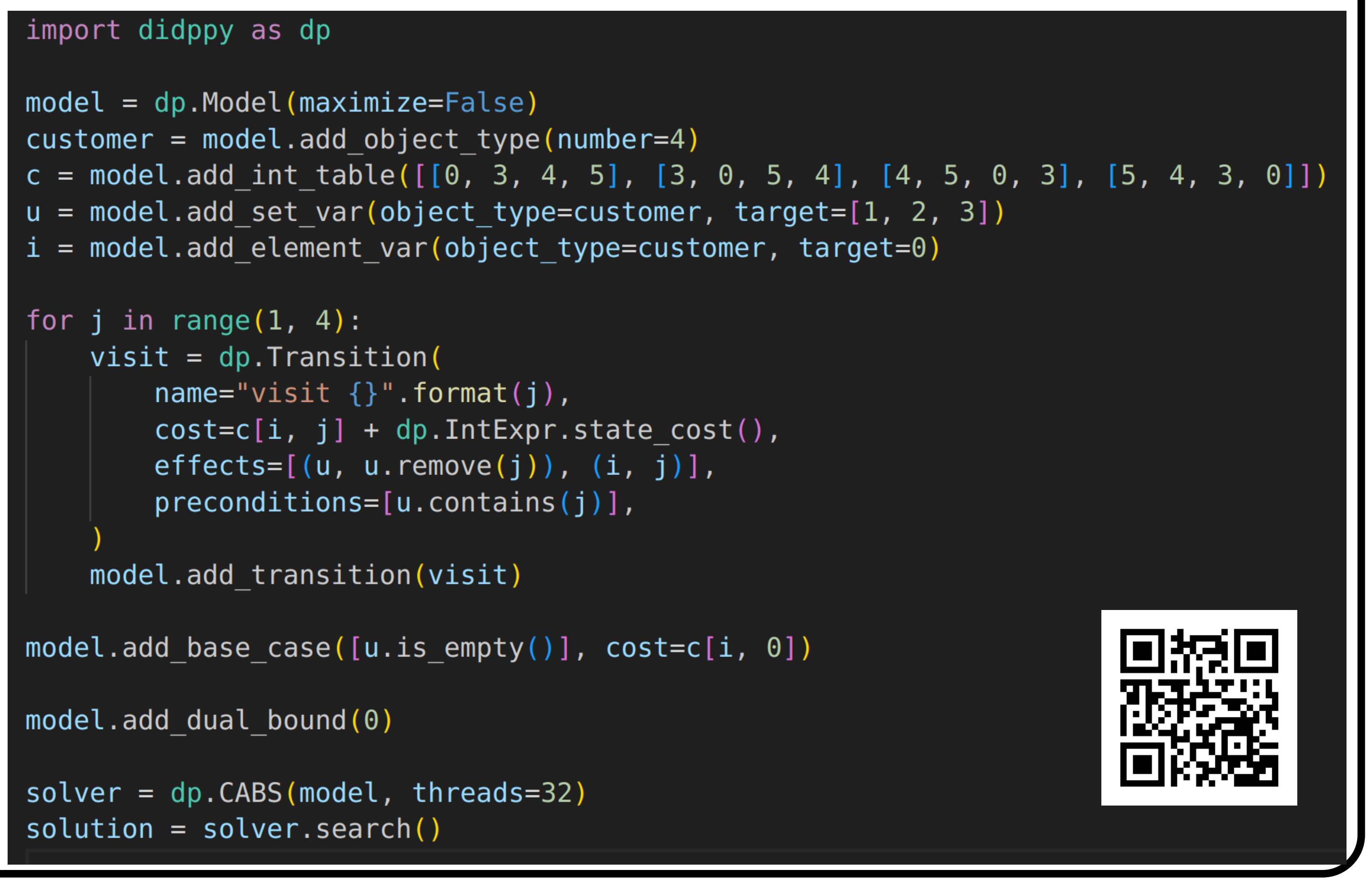


$$\text{compute } V(N \setminus \{0\}, 0)$$
$$V(U.i) = \min_{j \in U} c_{ij} + V(U \setminus \{j\}, j)$$
$$V(\emptyset, i) = c_{i0}$$

State Transition Graph for a DP Model



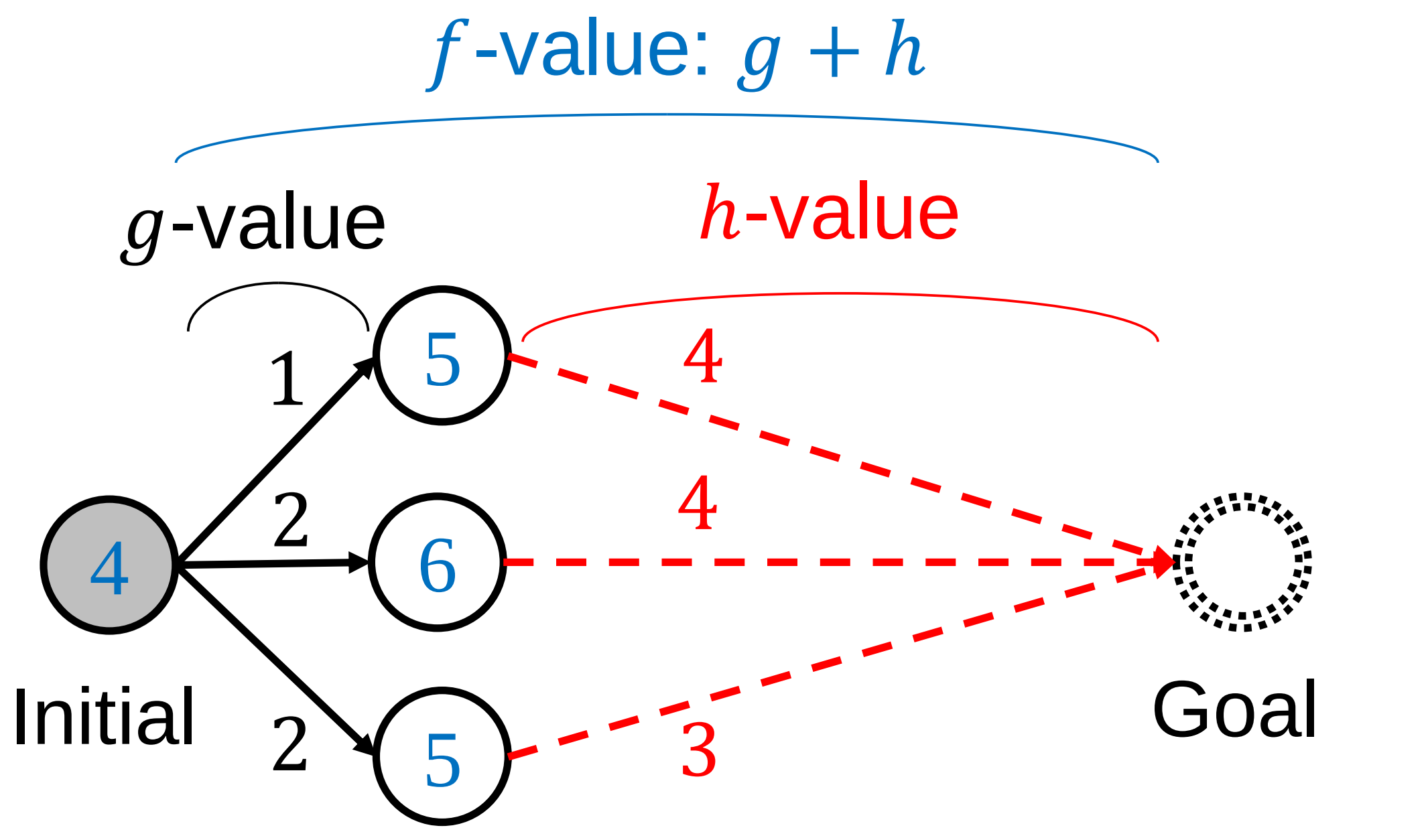
DIDPPy: Python Modeling Interface for DIDP



```python
import didppy as dp

model = dp.Model(maximize=False)
customer = model.add_object_type(number=4)
c = model.add_int_table([[0, 3, 4, 5], [3, 0, 5, 4], [4, 5, 0, 3], [5, 4, 3, 0]])
u = model.add_set_var(object_type=customer, target=[1, 2, 3])
i = model.add_element_var(object_type=customer, target=0)

for j in range(1, 4):
    visit = dp.Transition(
        name="visit {}".format(j),
        cost=c[i, j] + dp.IntExpr.state_cost(),
        effects=[(u, u.remove(j)), (i, j)],
        preconditions=[u.contains(j)],
    )
    model.add_transition(visit)

model.add_base_case([u.is_empty()], cost=c[i, 0])

model.add_dual_bound(0)

solver = dp.CABS(model, threads=32)
solution = solver.search()
```

Parallel Beam Search for DIDP

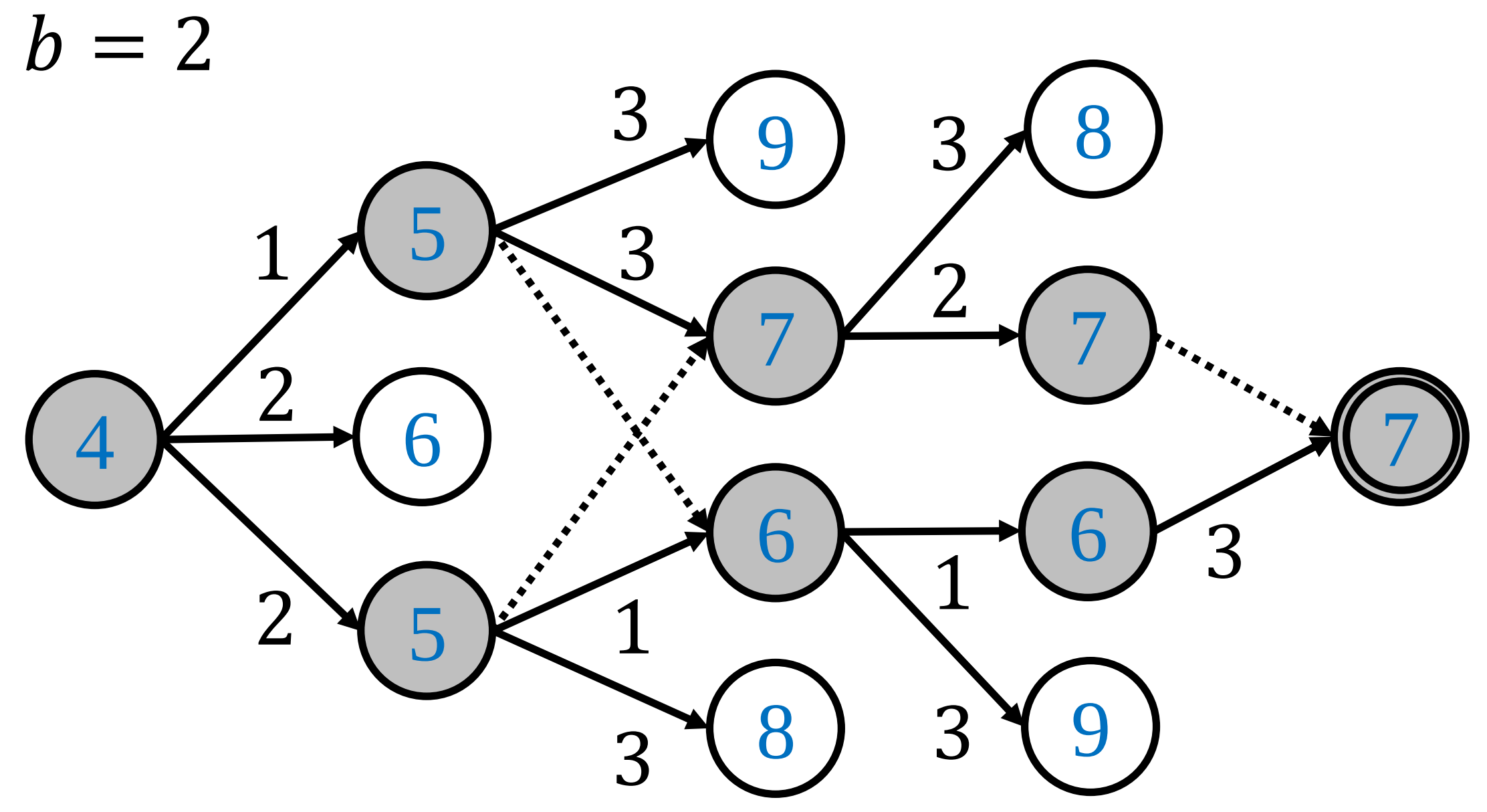
Heuristic Search for DIDP



- Find a path in a state transition graph
- g: actual path cost
- h: estimation by a heuristic function (given with a model in current DIDP)
- f: priority to expand

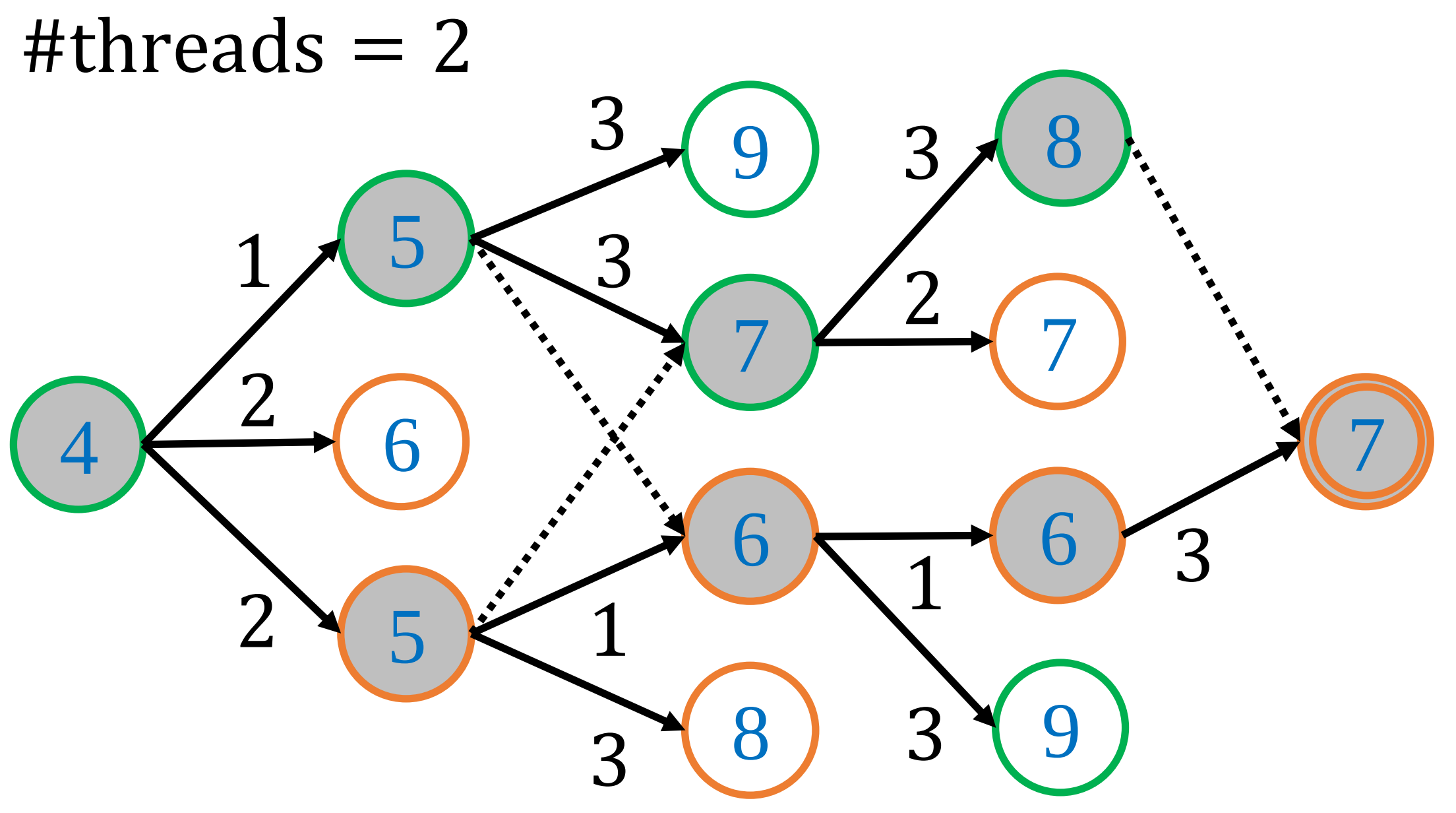
SOTA DIDP Solver: CABS

[Kuroiwa and Beck 2023c]



- Beam search expands the best b states minimizing the f-value in each layer
- CABS [Zhang 1998] repeats beam search with increasing b until proving optimality

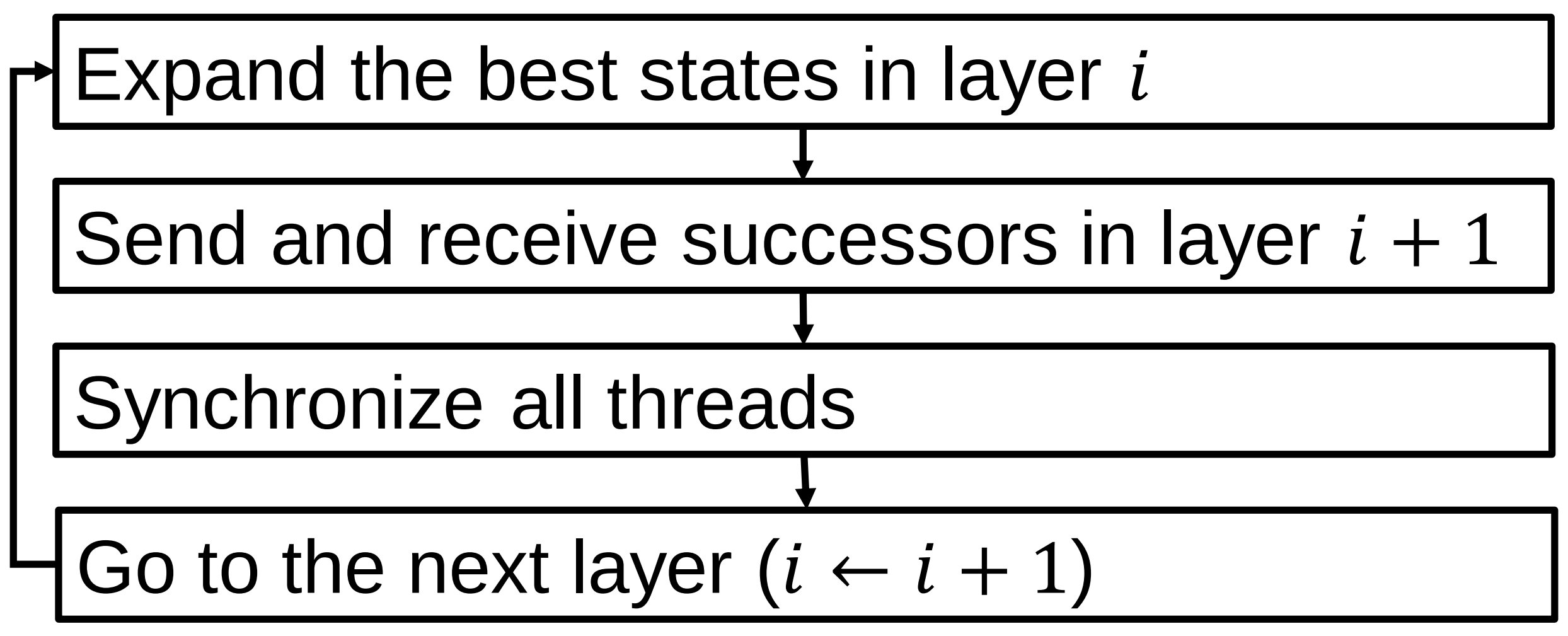
Hash Distributed Beam Search (HDBS)



- Adaptation of HDA* [Kishimoto+ 2013]
- Send a state to a thread assinged by its hash value with message passing
- Each thread expands $\frac{b}{\#\text{threads}}$ states

Layer Synchronization

HDBS1: Immediate Layer Synchronization



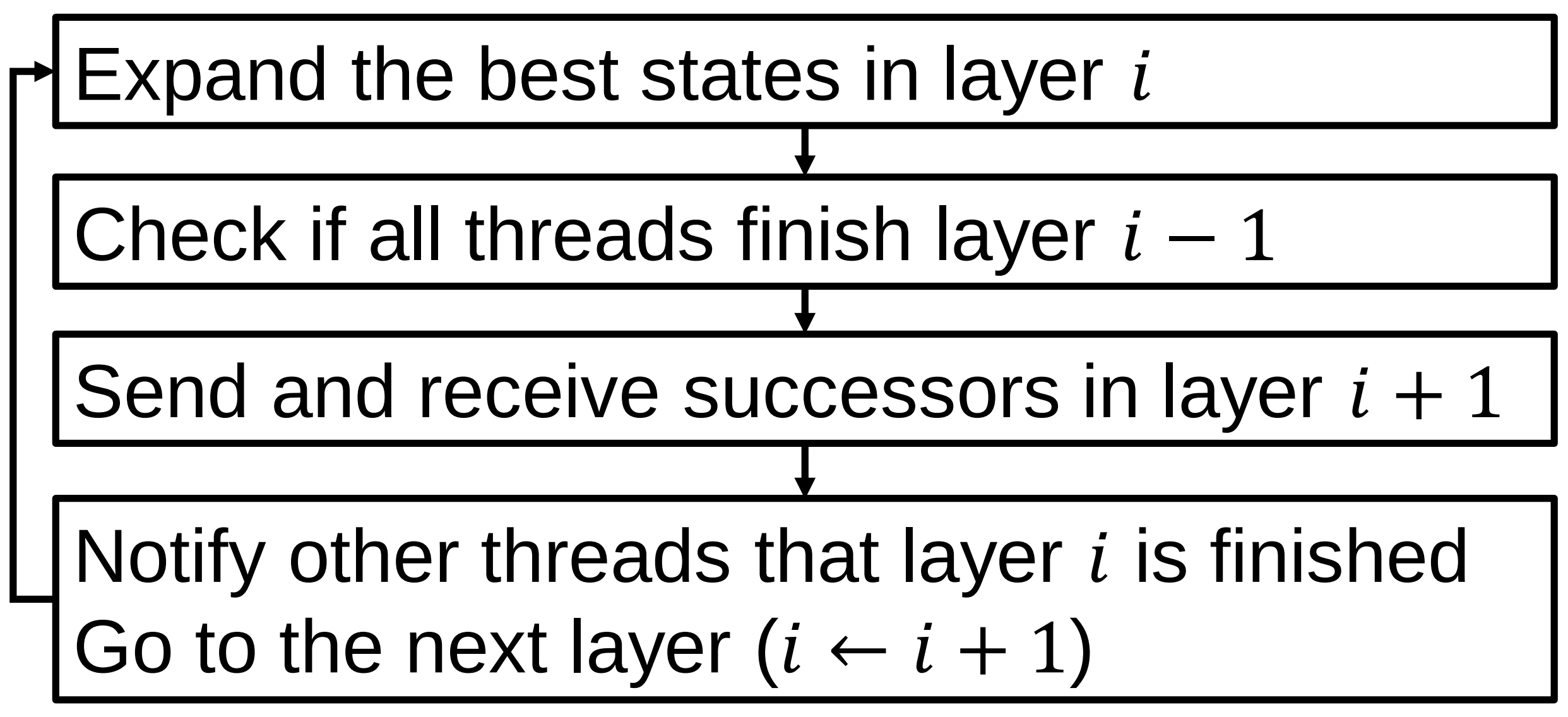
Expand the best states in layer i

Send and receive successors in layer $i + 1$

Synchronize all threads

Go to the next layer ($i \leftarrow i + 1$)

HDBS2: Delayed Layer Synchronization



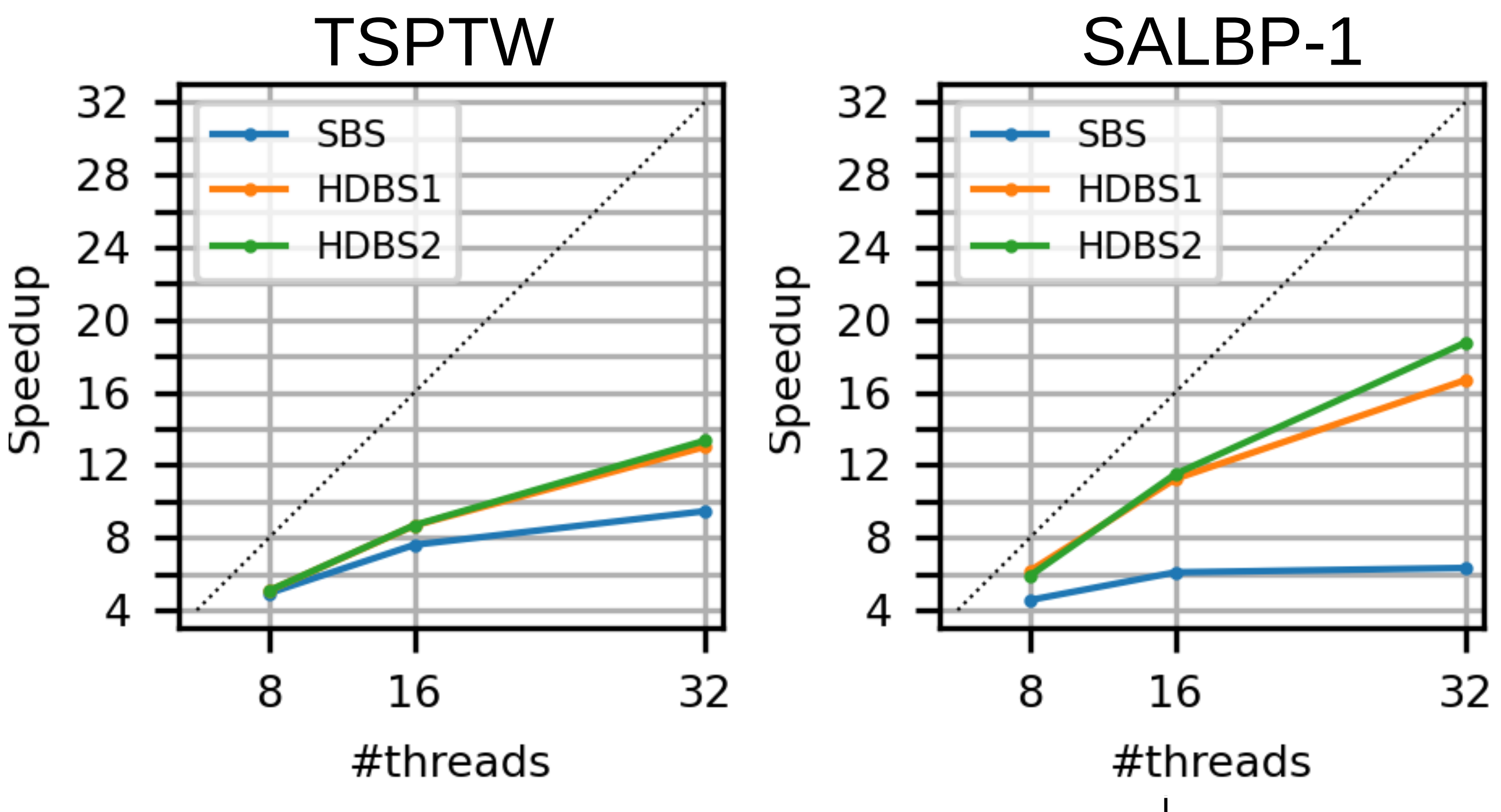
Expand the best states in layer i

Check if all threads finish layer $i - 1$

Send and receive successors in layer $i + 1$

Notify other threads that layer i is finished
Go to the next layer ($i \leftarrow i + 1$)

Experimental Evaluation



- 5-min, 188GB
- SBS: parallel beam search for DIDP with a concurrent hash table
- Gurobi: mixed-integer programming (MIP)
- CPO: CP Optimizer, constraint programming

Problem	Description	Gurobi	CPO	DIDP (HDBS2)
TSPTW (340)	TSP with time	239/4.2	27/0.1	**262**/13.3
CVRP (207)	vehicle routing	**29**/5.3	0/ -	8/ 9.3
SALBP-1 (2100)	line balancing	1351/1.3	1581/1.4	**1826**/18.8
Bin Packing (1615)	bin packing	1192/6.4	**1251**/9.2	1239/39.6
MOSP (570)	manufacturing	238/3.1	397/0.3	**531**/ 9.0
Graph-Clear (135)	building security	16/2.0	4/3.2	**113**/10.3

#optimally solved / mean speedup (32 threads)