Priority Inheritance with Backtracking for Iterative Multi-agent Path Finding

Keisuke Okumura[1], Manao Machida[2], Xaxier Défago[1], Yasumasa Tamura[1]

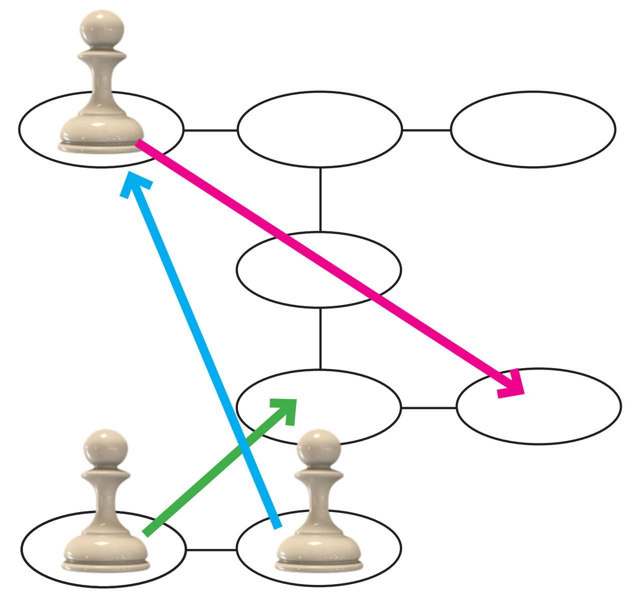
1. Tokyo Institute of Technology, 2. NEC Corporation

SHORT DESCRIPTION

Propose an algorithm to plan paths for multiple moving agents iteratively/flevibly/scalably, called **PIBT**.

In biconnected-like graphs, PIBT ensures reachability, i.e., all agents reach their goals in finite time after being given.

MULTI-AGENT PATH FINDING (MAPF)



Plan efficient paths while avoiding collisions

Computationally-hard to obtain optima [Yu and LaValle, AAAI13]

Applications

YouTube/Mind Blowing Videos

- traffic control
- automated warehouse

PRINCIPLES OF PIBT

In reality, MAPF must be solved **iteratively**

requests to ⇓ MAPF solvers

1. **Speed** over Optimility → Online, Real-time
2. **Decentralized** over Centralized → Robustness, Scalability
3. **Local** over Global Interaction → Concurrency, Scalability
4. Adaptivity for **Iterative** Use → Practicality

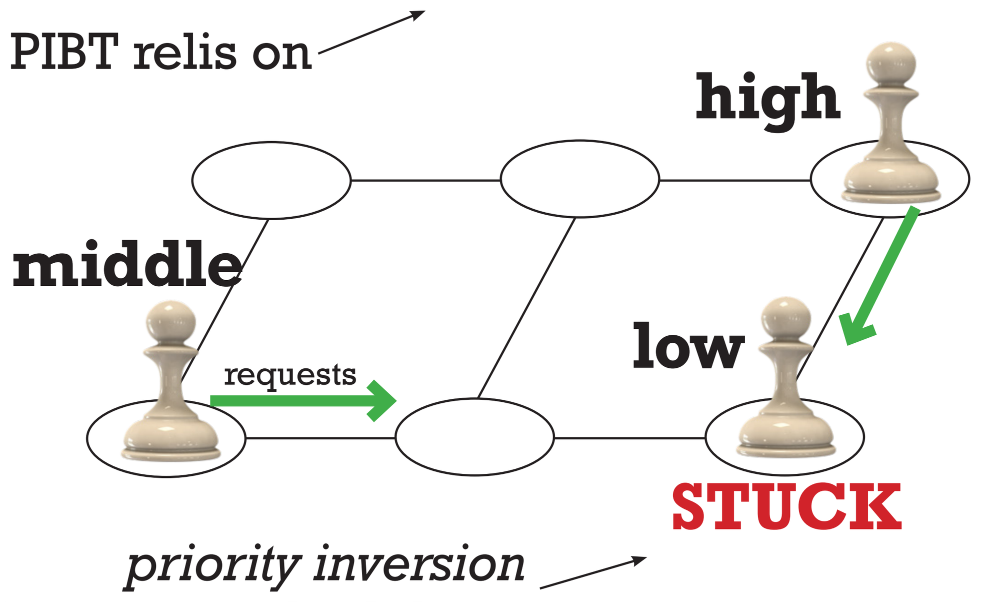
ALGORITHM

Prioritized Planning

pros: computationally-cheap
cons: **incomplete**

planning where time-window is just one

PIBT relis on

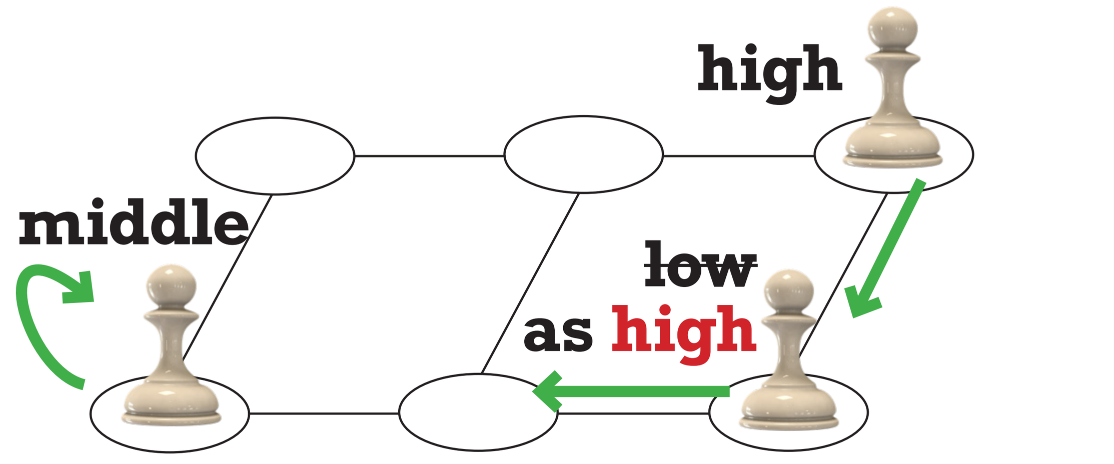


priority inversion (resource scheduling)

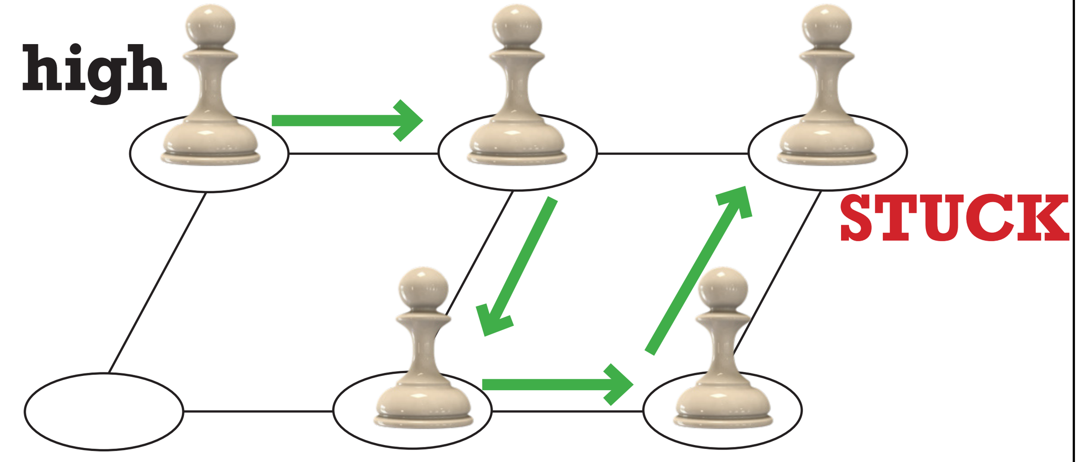
countermeasure: *priority inheritance* [Sha et al, 1990]

Priority Inheritance

The low-priority agent **temporarily inherits** the higher-priority.



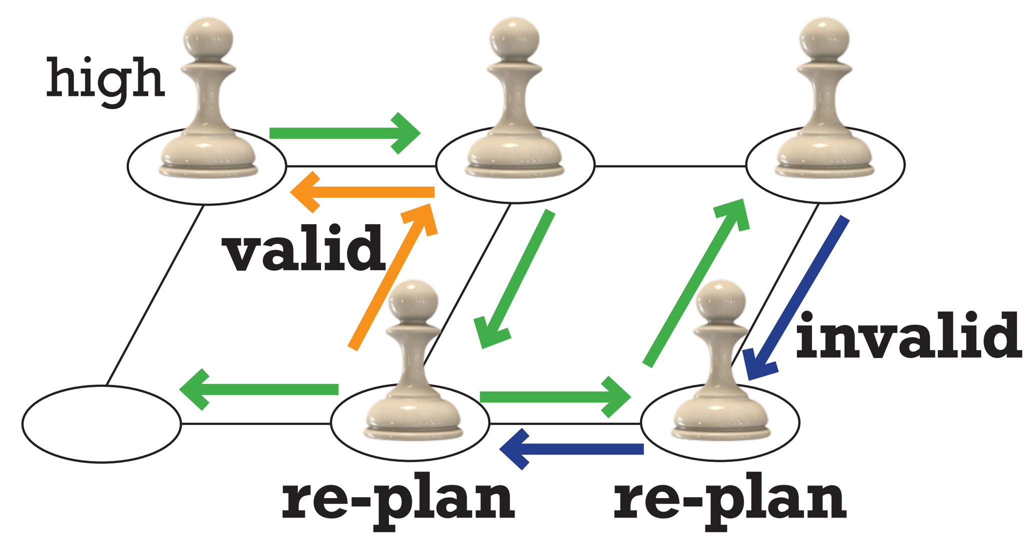
Priority inheritance itself is **insufficient**.



Backtracking

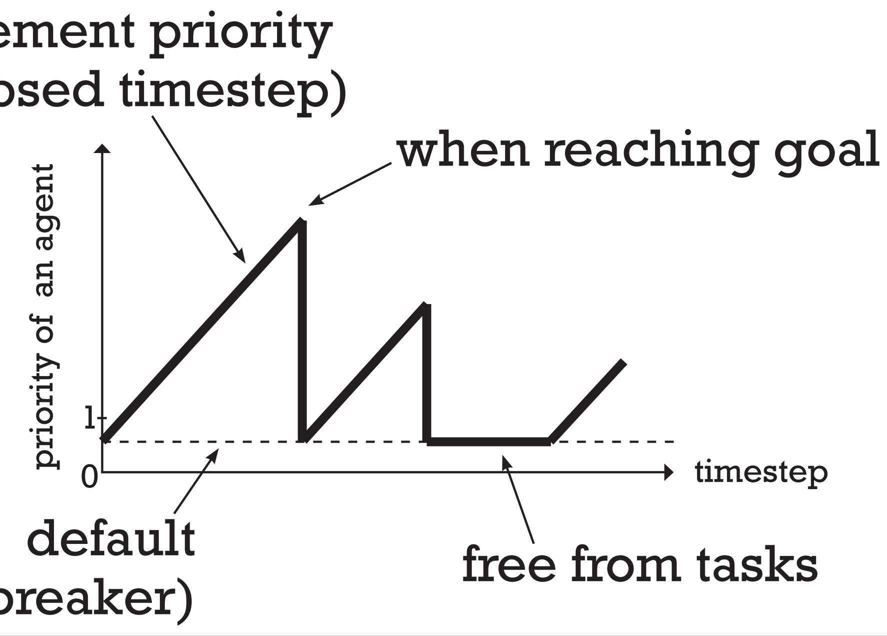
Agents with priority inheritance have to wait for *backtracking*.

→ valid — You can move
→ invalid — You must re-plan, I will stay



Dynamic Priorities

Once an agent reaches its goal, it drops its priority to ensure **fairness**.

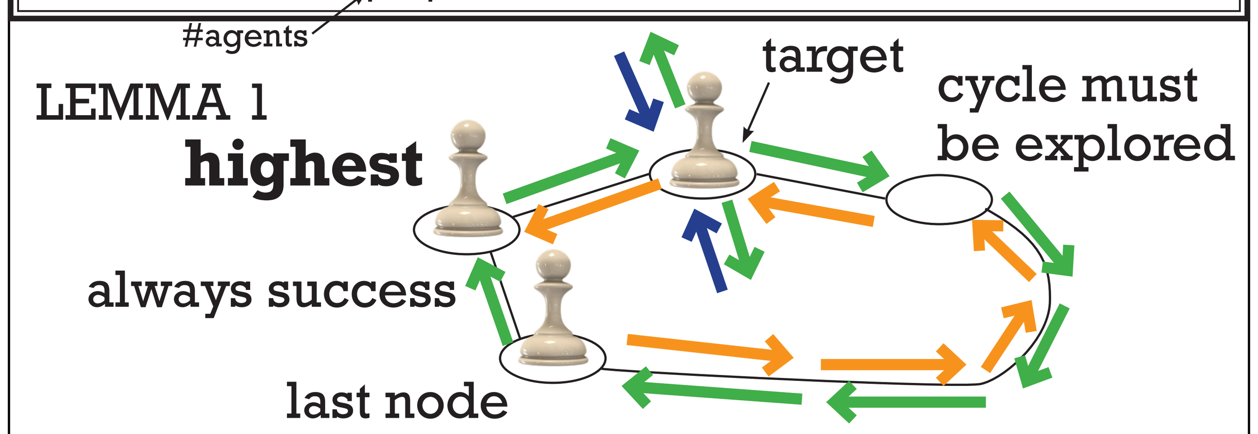


Reachability

THEOREM 2 — e.g.) biconnected graph

> If a garph G has a simple cycle for all paris of adjacent nodes, with PIBT, all agents reach their destination within diam(G)|A| timesteps after being given.

#agents

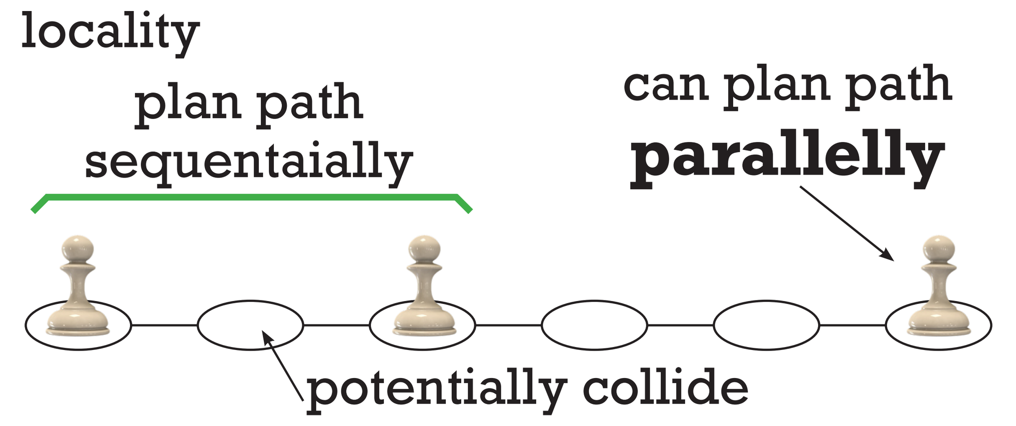


Property

- computational complexity per timestep

$O(|A| \cdot \Delta(G) \cdot F)$ — maximum time required to choose the next target node

- locality



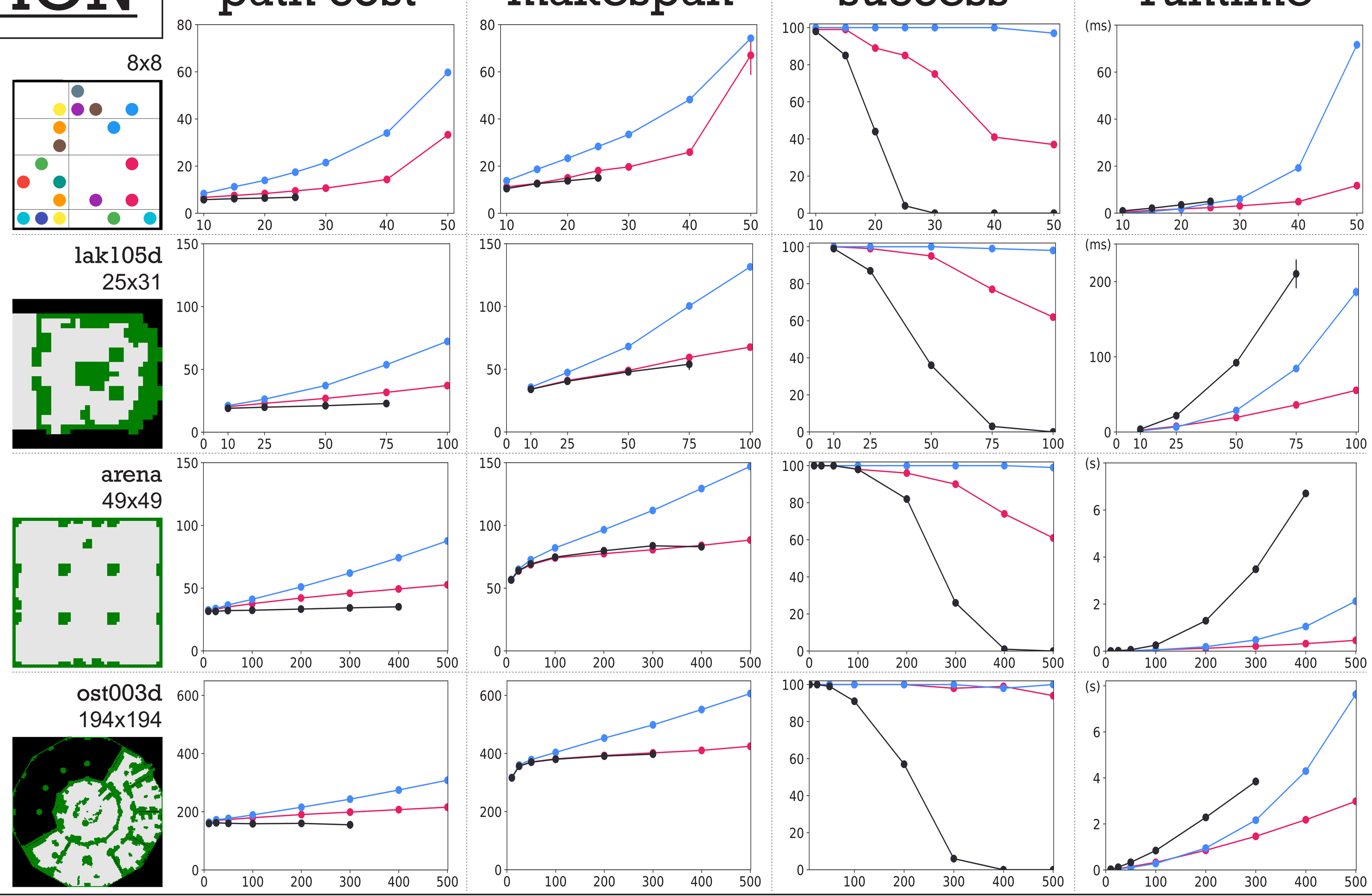
EVALUATION

Multi-agent Path Finding

PIBT

Parallel Push& Swap [Sajid et al., SoCS12]

WHCA* [Silver, AIIDE05] window: 5(8x8),10(otherwise)



- *path cost*: timestep when *each* agent reaches its goal and never move from then
- *makespan*: timestep when all agents reach their goals
- average over only success cases within the solver (in 100 instances) are shown

Multi-agent Pickup & Delivery

the problem for conveying packages in an automated warehouse [Ma et al., AAMAS17]

PIBT + task allocation

> for each timestep, each free agent moves to the nearest pickup location of the non-assigned task

All tasks are completed in finite time if graph condition is satisfied.

50 agents, 500 tasks — PIBT — TP [Ma et al., AAMAS17]