Searching with Consistent Prioritiz^n for MAPF

We explore prioritized planning of Multi-Agent Path Finding (MAPF) algo. w.r.t. optimality & completeness.

1. Introduction

- For a MAPF we analyze quality by :- flowtime (sum of arrival time at targets) or makespan (max^m of arrival time of agents at targets).
- Prioritized MAPF are most efficient for MAPF which follows the algo., each agent gets unique priority and computes min^m-cost path from start to target without collision with planned path of higher priority ones.
- Predefined total priority ordering - a priori
- we look into a framework, discussing the limits of prioritized planning.
- Two prioritized MAPF :-

(i) CBS w/P
→ Conflict based search with Priorities
→ A best-first search.
→ First lazy prioritizing for all agents using BFS & introduce order pairs when collision.

(ii) PBS
→ Priority based search
→ It explores total space of priority orderings lazily. ~~using system~~ a.
→ uses Depth-first search.
→ It can take partial user-specified priorities as inputs and dynamically add new ordered pairs keeping in mind the consistency with partial priorities.

Note → Standard prioritized MAPF is special case of PBS

Optimality: PBS > CBS w/P > (State-of-the-art CBS)

2. Problem Definition

- undirected Graph $G = (V, E)$, M agents $\{a_i\}, i \in [M]$
- Each a_i has $s_i, t_i \in V$, start & target, with timesteps $t = 0, 1, \ldots, \infty$
- $\pi_i(t)$ is vertex occupied by a_i at 't'. $\pi_i(0) = s_i$, $\pi_i(T) = t_i$
- Path $\pi_i = \langle \pi_i(0), \ldots, \pi_i(T_i), \pi_i(T_{i+1}), \ldots \rangle$ for a_i
- Vertex collision $\langle a_i, a_j, v, t \rangle$
- Edge collision $\langle a_i, a_j, u, v, t \rangle$, edge (u, v).
- Quality by $\boxed{\text{flowtime } \sum_{i \in [M]} T_i}$

---

3. Prioritized Planning

- Basically plans high priority ones first (with only fixed obstacles) → then, low priority (checking fixed obstacles + high priority collision).
- high priority obstacles ⟹ <u>Dynamic obstacle</u>

<u>Definition-1</u> Priority ordering $\prec$ is strict partial order on $[M]$. a_i has high prior. than a_j iff $i \prec j$

---

4. Theoretical Results

<u>Theorem-1</u> Arbitrary prioritized planning $\prec$ is incomplete for MAPF usually.

- We define P-solvable, as the class of MAPF instances which has sol^n for <u>prioritized planning</u>.

<u>Definition-2</u> Solution $L = \{\pi_i \mid i \in [M]\}$ is consistent with $\prec$ if, for all pair of agents $a_i \prec a_j$, we can improve arrival time of a_i at t_i by removing a_j.

<u>Definition-3</u> MAPF instance is P-solvable iff there's a sol^n $L = \{\pi_i \mid i \in [M]\}$ which is consistent with $\prec$.

Theorem - 2 Prioritized planning with any $\prec$ is incomplete for class of P-solvable MAPF instance.

Theorem - 3 Prioritized planning with any $\prec$ is complete for well-formed MAPF instances

Theorem - 4 Prioritized planning is sub-optimal in general for flowtime objective for class of P-solvable.

Corollary - 5 Theorem 4 holds for makespan objective also.

Definition - 4 MAPF instance is OP - solvable if:
(i) admits sol^n L^*, consistent with some $\prec^*$, and
(ii) L^* is optimal among all consistent or inconsistent sol^n

---

Theorem - 6 Prioritized planning with fixed total $\prec$ is incomplete usually for OP-solvable.

---

5. CBS w/P

- It performs BFS on high priority ones, building a constraint Tree (CT)
- Each node N with N.constraint, N.plan, N.cost
- It expands CT node with smallest cost, and also stores $\prec_N$ in each CT node N, and extend CT node N ~~to~~ N' whose $\prec_{N'}$ extend $\prec_N$.

Definition - 5 A $\prec_A$ extends $\prec_B$ if $\forall\, i,j \in [M]$: $i \prec_B j \Rightarrow i \prec_A j$, i.e. $\prec_A$ has priority info. of $\prec_B$ also.

Note :- CBS w/P uses space - time A* algo. to find an individually optimal path for a_i with constraint N_i. constraints.

Properties : CBS w/P does new partial ordering of child pair nodes, whenever it splits, hence : $O(M^2)$.

6. P.B.S

- 2 level prioritized
- DFS on high level to dynamic partial ordering & builds Priority Tree (PT).
- PBS greedily chooses to give high priority and backtracks only when no solⁿ in current branch
- Hence, constructs incremental single partial priority ordering until no collisions.
- PT splits node like CBS w/P, if collision but doesn't store constraints maintains that there's no collisions when $a_i < a_j$ ($i <_N j$) for PT node N.

Note :- PBS splits like CBS w/P so $O(M^2)$ and also its depth is $O(M^2)$.

---

7. Experiments

- CBS
- FIX (PBS with fixed total priority ordering)
- LH (Priority high for long individual optimal path)
- SH (" for short ")
- RND (runs PBS 10 times with random total priority ordering & picks solⁿ with smallest cost).

- Acc. to 0% & 10% obstacles

PBS >> RND >> SH >> CBS w/P >> CBS

- for runtime 0% & 10% obstacles

(CBS w/P) >> CBS & (RND << PBS variants)

- flowtime / optimal flowtime

[CBS w/P always optimal] & [PBS close to optimal]

- By game maps : CBS w/P outperforms CBS & [PBS outperforms FIX, CBS, CBS w/P] in terms of success rat
- PBS nearly optimal solⁿ in game maps.

8. Conclusion
- Conceptually we discussed limits of prioritized planning.
- Practically developed CBS w/P & PBS which give 'good' solⁿ.
- CBS w/P used BFS & PBS used DFS

[Darpan Singh - IMT2020133]