Bridging the Gap Between Centralised and Decentralised Multi-Agent Pathfinding

Problem Statement
Navigating multiple mobile units to their targets.

Avoiding collisions with:
- static obstacles, and
- other moving units in the shared environment.

Finding a solution typically requires searching for a sequence of actions that lead to the goal state. State: the positions of all units at a given time.

Real-life applications
- robotics - military operations
- logistics - disaster rescue - commercial games

Our current work assumes homogeneous agents on 2-D grid maps. A problem instance: a map with n traversable tiles and some obstacles, a collection of n units each having start and target positions.

For the sake of clarity, we allow only straight moves in the 4 compass directions.

Centralised vs. Decentralised Approach
Single-agent path planning can be tackled with A*.

In multi-agent problems, 2 traditional approaches are taken to extend A*:

Centralised
- Decomposes problem into a series of searches.
- Typically scales to larger problems.
- Significantly lower CPU + memory requirements, faster.
- Sub-optimal.
- Incomplete.
  - No formal criteria to distinguish between problems that can be solved, or where algorithm fails.
  - No guarantees w.r.t. running time, memory requirements, solution length.
  - e.g. Local Repair A*:
    - Flow Annotation Replanning (FAR, Wang and Botea 2008)

Decentralised
- Theoretically cost-optimal, complete.
- Scales up poorly in practice: combined state space has $O(n^3)$ states; maximum branching factor of 5
- e.g. Centralised A*

Decentralised
- Avoid replanning
- Prefer straighter paths
- Temporal reservations
- Waiting
- Otherwise, detect and break deadlocks by local plan repair

Steps:
1. Build a flow-annotated search graph:
   - control navigation flow
   - preserve local connectivity (Figure 1)
   - single-width tunnels
   - sinks
   - sources

2. Run one independent A* search per unit.

3. Execute the plan:
   - Avoid repositioning
   - Progression

Tractable Multi-Agent Pathfinding [Wang and Botea 2009]
- Aim: under what conditions are problems guaranteed to be solvable in polynomial time?
- A problem belongs to class \textit{Tractable} iff for each unit a path exists s.t.
  1. alternate connectivity (Figure 2)
  2. initial blank
  3. target isolation

- Blank travel operation:

MAPP (multi-agent path planning):
- $O(m^2 n^3 \log n)$ running time, $O(m^2 n^3)$ memory, $O(m^2 n^3)$ solution length, or linear in m when the maximal length of Q is constant.
- Incomplete in general case, but complete for \textit{Tractable}.
- Example:

Future Work
- Look at ways of obtaining different tractable classes, and using tractable classes to solve multi-agent pathfinding problems.
- Further study the trade-off between completeness and efficiency.
- Continuous to bridge the gap between centralised and decentralised approaches, aiming for both efficiency and formal guarantees.
- Investigate makespan-optimality.
- Extend to more general domains.
- Seek real-life applications, e.g. robots!

Publications