Problem Statement
• Navigating multiple mobile units towards their targets.
• *Static* and dynamic obstacles in a shared environment.
• PSPACE-hard [Hopcroft et al. 1984].
• Often, needs to be solved in real time.
• Applications: robotics, air traffic control, vehicle routing, disaster rescue, military operation planning, computer games...
• This work assumes homogenous agents on grid maps.

The FAR Method
FAR (Flow Annotation Replanning)
• Decentralised, with online conflict resolution.
• Inspired by unidirectional lanes on a two-way road.
• Aim: cheap computations + low memory usage.
• Steps:
  1. Build a flow-annotated search graph:
     • control navigation flow
     • preserve map connectivity
     • single-width tunnels
     • sinks
     • sources
  2. Run one independent A* search per unit.
  3. Execute the plan:
     • Avoid replanning
       • favour straighter paths on equal $f$-values
       • temporal reservations, $k$ steps ahead
       • waiting
       • traffic lights: temporal flow regulation
     • Otherwise,
       • local plan repair
       • detect and break deadlocks

Experiments
• 2.8GHz Intel Core 2 Duo Mac with 2GB of RAM.
• 10 largest maps from *Baldur's Gate*, a standard data set.
• For each map, increase $N$, the number of mobile units, by 100 at a time. Generate 10 problem instances for each $N$.
• Time limit set to 10 minutes per problem. $k = 3$.
• Compared with WHCA*(8,1), with and without diagonals [Silver 2005; Sturtevant & Buro 2006].
• Run on the Hierarchical Open Graph framework (HOG) [Sturtevant 2007].

Future Work
• Investigate new heuristics, better waiting strategies, dynamic flows.
• Analytical studies.
• Incorporate FAR into a real game, or enter RoboCup Rescue.
• Extend FAR for: planning under uncertainty; initially unknown maps; dynamic environments; moving targets.