A Distributed Sampling-based Motion Planner

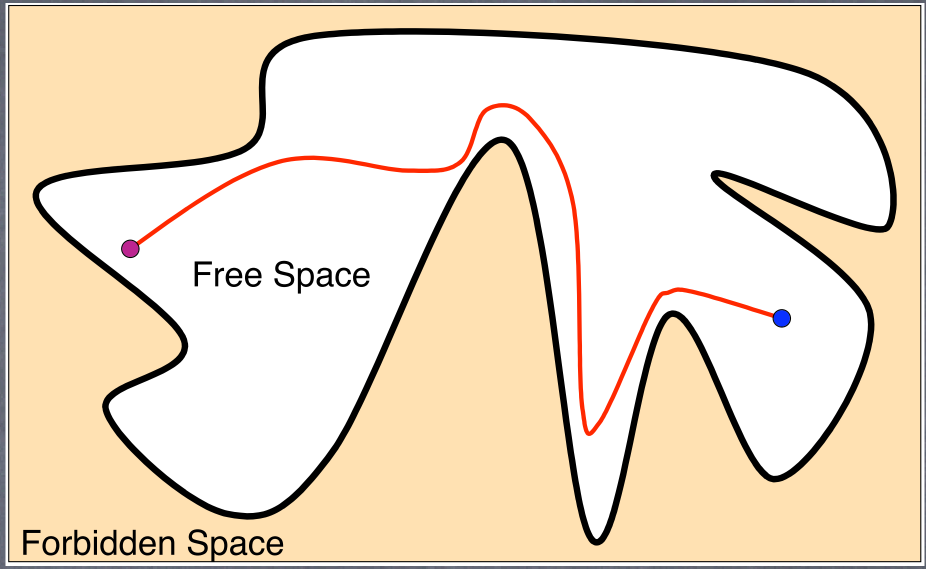
Presented by Jing Yang
Oct 11, 2007

Reference: Erion Plaku and Lydia E. Kavraki, "Distributed Sampling-based Roadmap of Trees for Large-Scale Motion Planning." IEEE International Conference on Robotics and Automation, Barcelona, Spain, 2005, pp. 3879-3884.

Agenda

- Introduction
  - Basics of Robot Motion Planning
  - Sampling-based Roadmap of Trees (SRT)
- Distributed SRT
  - Client-master architecture
  - Three stages of the algorithm
- Experimental Results and Discussions

Motion Planning Basics

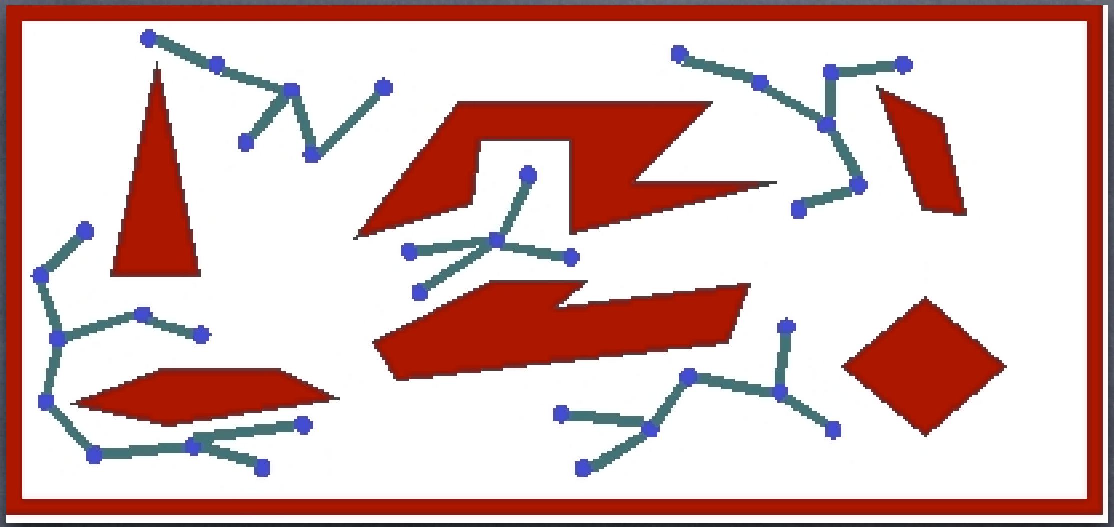


- Configuration space (C-Space)-- The space of all the configurations of the robot.
- Free C-Space -- The set of configurations at which the robot does not collide with any obstacles.
- Motion Planning -- Given two configurations of a robot, find a free path in the free C-Space that connects them.

Complexity of Motion Planning

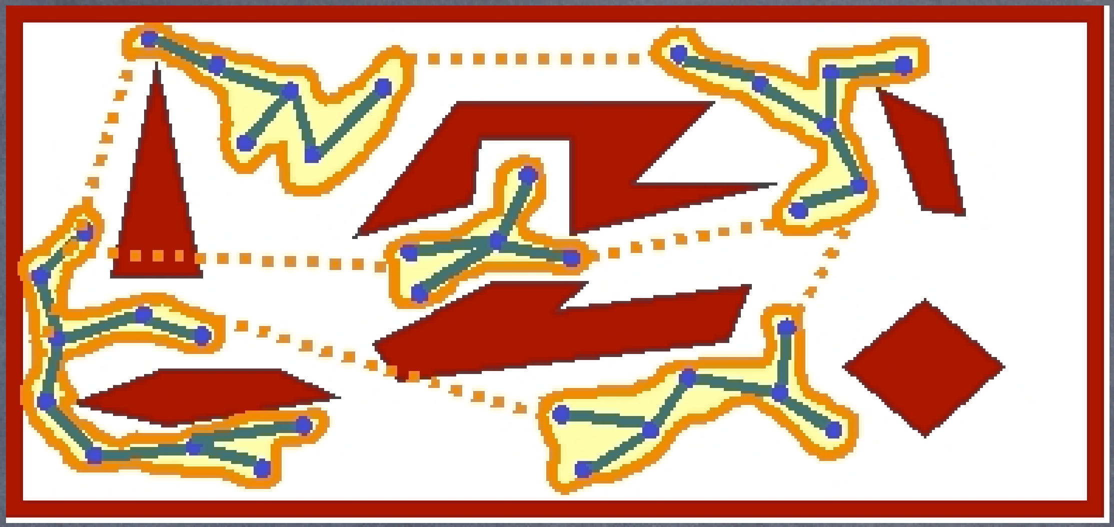
- Difficulty
  - Dimension of the configuration space = # of degree of freedom of the robot
  - Geometric complexity
- Complete motion planners takes exponential time in the # of degrees of freedom
- Randomized algorithms (sampling-based) are the trend

SRT (1)



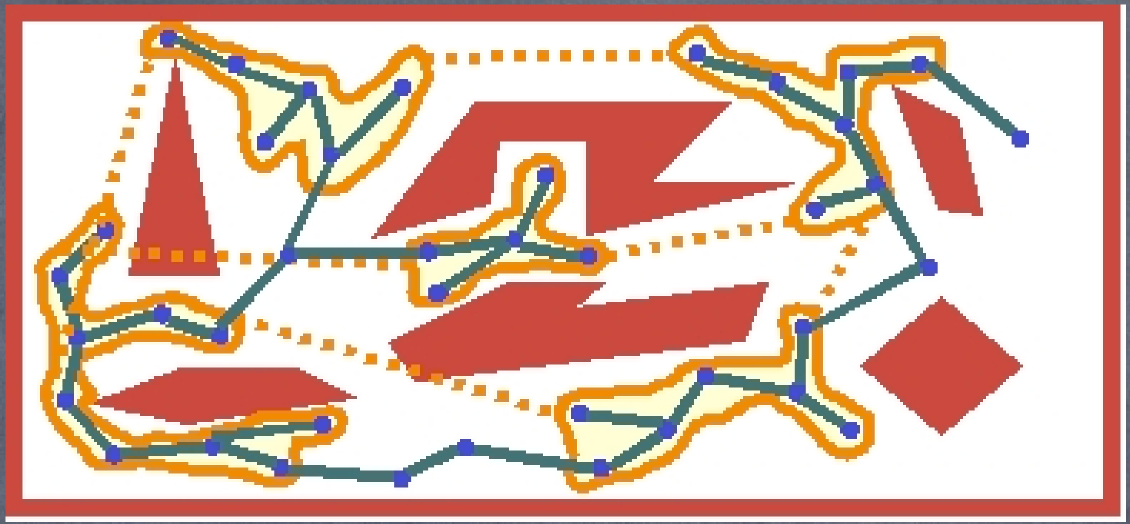
- Milestone Computations
- Candidate Edge Computations
- Edge Computations

SRT (2)



- Milestone Computations
- Candidate Edge Computations
- Edge Computations

SRT (3)



- Milestone Computations
- Candidate Edge Computations
- Edge Computations

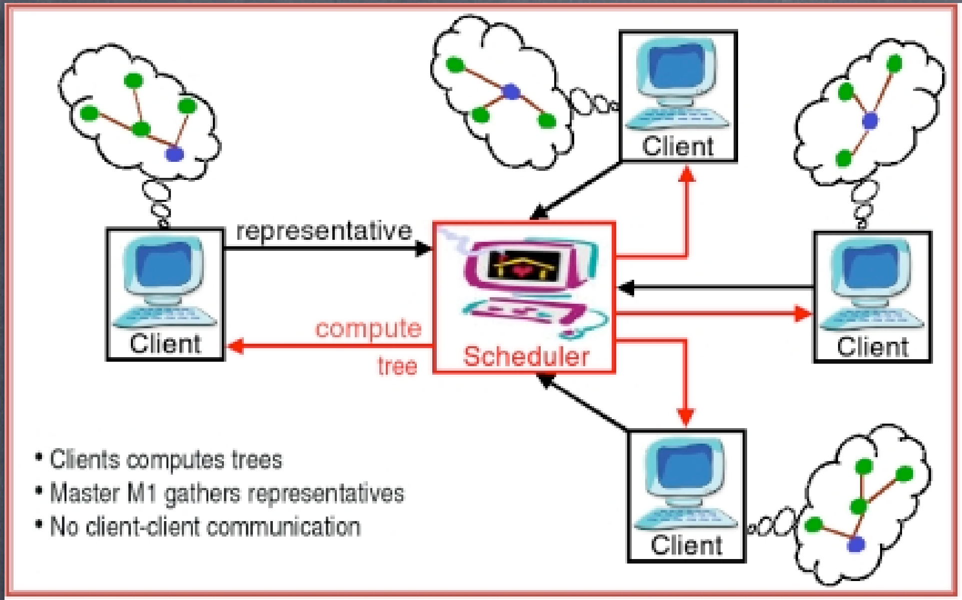
Challenges for SRT

- Complex robotic systems have configuration space with thousands of dimensions
- Require development of distributed planners that take full advantage of all the available resources

Distributed SRT - Overview

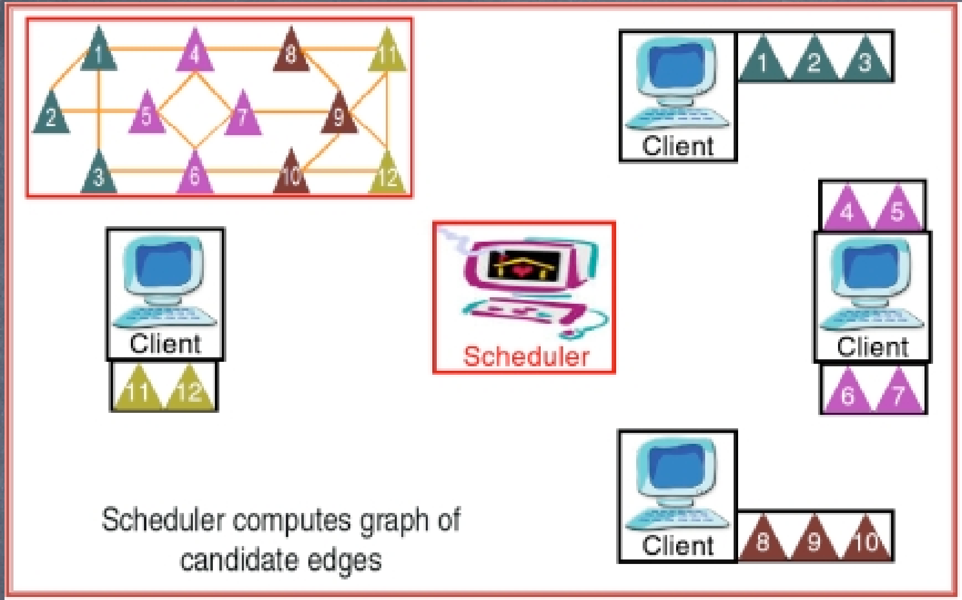
- A distributed version of sequential SRT using a master-client architecture
  - Clients {C1,...,Cc} (useful computations): milestone and edge computations
  - Masters {M1,...,Mm} (schedulers): ensure the task load is distributed evenly among the clients

Milestone Computations



- No dependencies between milestones
- M1 is the main scheduler (counting # of milestones created)
- Each milestone is processed in parallel by all the clients and masters except M1

Candidate Edge Computations

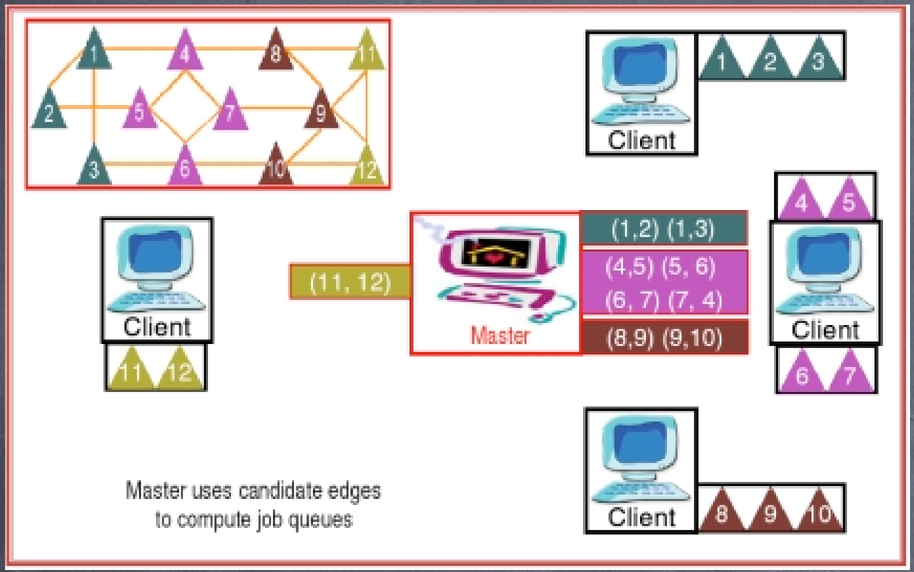


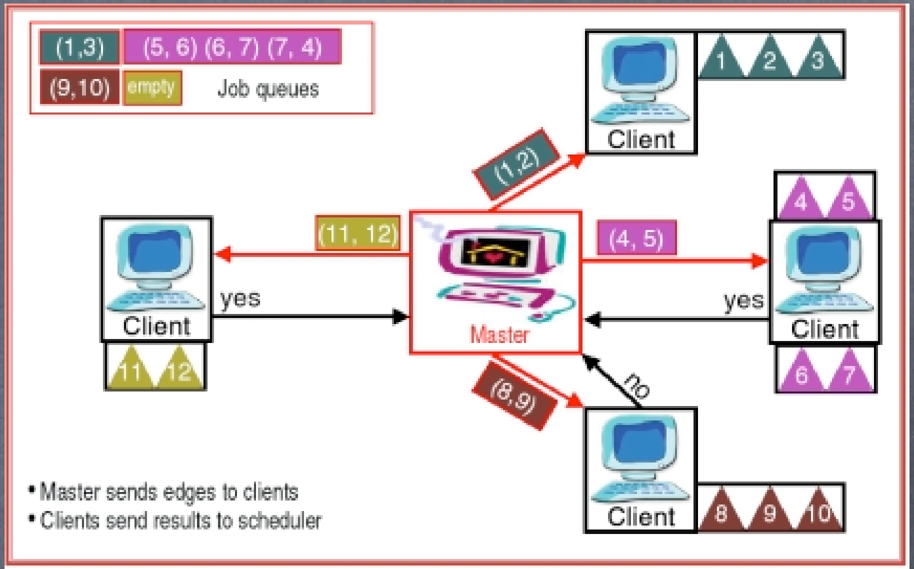
- Selection of candidate edges depends on nearest-neighbor milestones (Distributed K-Nearest-Neighbors method?)
- M1 computes the candidate edges among the representatives stored in its local memory, and send them to all masters.

Edge Computations

- A master chooses one of its clients that is currently available for edge computation.
- Both milestones of the edge must be stored in the local memory of the chosen client
- Two cases:
  - Both milestones are currently owned by the client (simple)
  - One or neither is owned by the client (complex)

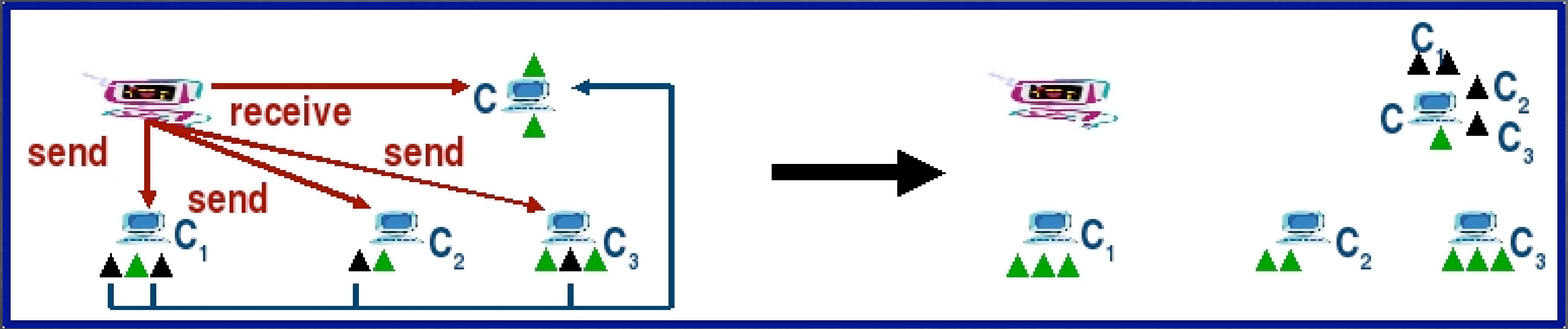
Edge Computations (case 1)





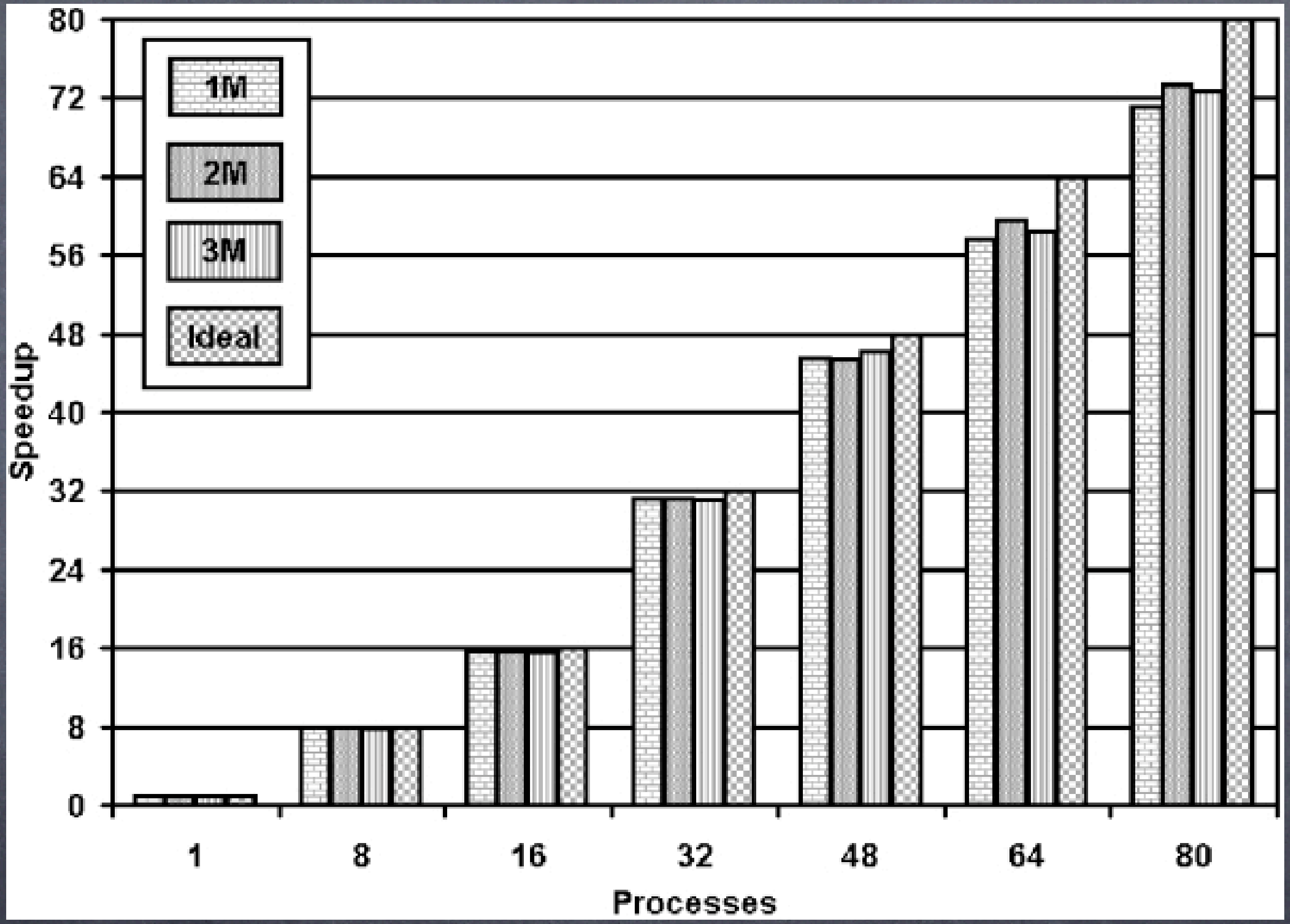
- Two milestones of the candidate edge are stored in the local memory of the client
- Master-client communication only; no client-client communication

Edge Computations (case 2)



- A client C needs to wait for other clients (C1,C2,C3) to send it copies of their milestones
- C computes the edge once the two milestones are stored in its local memory
- Update is needed
- Master-client and client-client communications

Experimental Result



- Nearly linear speedup with 80 processors

Discussions

- Power of distributed system in Robotics
- How to choose the number of masters?
- Message passing or shared memory?
- Any questions from you?

Thank you!