

# **Task Allocation – Negotiation and the Contract Net Protocol**

March 25, 2003

Class Meeting 19

# Outline

---

- Introduction to Task Allocation
- The Contract Net Protocol

# Multi-Robot Task Allocation

---

- What is task allocation?
  - Determining which robot performs which action (or task)
- Same concept: *action selection*
- Decision can be made either *centrally* or in a *distributed* fashion
- Usually: *task allocation* is distributed

# Formal Definition of Task Allocation Problem

- Given:

$$\begin{array}{ll} R = \{r_1, r_2, \dots, r_n\} & n \text{ robots} \\ T = \{task_1, task_2, \dots, task_m\} & m \text{ independent subtasks} \\ c_{ij} & \text{Cost for robot } r_i \text{ to perform } task_j \end{array}$$

– Find the set of tasks  $U_i$  for all  $r_i$ , such that:

$$\forall_k . \exists_i . task_k \in U_i$$

That is, for all tasks  $k$ , there exists a robot  $r_i$  such that  $task_k$  is assigned to robot  $r_i$

and  $\sum_{i=1}^n \sum_{task_k \in U_i} c_{ik}$  is minimized.

That is, the sum of the costs of the tasks assigned to each robot, summed over all robots, is minimized

**IMPORTANT NOTE:** This task assignment problem is NP-hard. [Parker, 1994]

# Two primary approaches to task allocation

---

- Negotiation / market-based

- Robots negotiate with each other and make “bids” on “contracts” for executing tasks
- Example: MURDOCH (to be studied next time)

- Modeling robots

- Robots use models embedding information of other robot capabilities to determine their own “motivations” for performing tasks
- Example: ALLIANCE (to be studied next week)

# Today: Contract Net Protocol

---

- **Contract Net Protocol** is the basis for most approaches to multi-robot task allocation using negotiation
- “The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver”, by R. G. Smith, *IEEE Transactions on Computers*, C-29 (12), December 1980.
- **Primary purpose:** defining a negotiation scheme that enables distributed problem solving among agents.
- In this discussion, **agent = node = robot**

# Distributed Problem Solving

---

- Distributed problem solving:
  - **Cooperative solution:** no Knowledge Source (KS) has sufficient information to solve entire problem
  - **Decentralized collection of KSs:** both control and data are logically and often geographically distributed; no global control or data storage
  - **Loosely coupled collection of KSs:** individual KSs spend most of their time in computation rather than communication
- Advantages:
  - Speed, reliability, extensibility, potential for increased tolerance to uncertain data and knowledge, ability to handle applications with natural spatial distribution
- Often used in Artificial Intelligence; Applications include:
  - Traffic-light control
  - Distributed sensing
  - Heuristic search
  - Robotics

# Character of Problems Solved

---

- Involves:

- Tasks that do not have well-defined algorithms for their solutions (e.g., most AI-type problems)
- Generation of many tasks during search for solutions
- Generation of many tasks whose execution often will not lead to a solution
- Unknown *a priori* mapping of “best” knowledge sources for each task

- Fundamental challenge:

- Finding appropriate tasks for idle agents
- Finding appropriate idle agents for tasks

# Components of Contract Net Negotiation

---

1. Local process that does not involve centralized control
2. Two-way exchange of information
3. Each party to the negotiation evaluates the information from its own perspective
4. Final agreement is reached by mutual selection

# The Contract Net

---

- Contract Net = collection of nodes
- Execution of task: dealt with as a contract between nodes
- Two roles for nodes:
  - Manager: responsible for monitoring execution of task and processing results of execution
  - Contractor: responsible for actual execution of task
- Any node can take on either role dynamically during problem solving

# The Contract Process: High Level

---

- **Contract:** established by local mutual selection based on 2-way transfer of information
- **Contractors:**
  - Evaluate task announcements made by managers
  - Submit bids on those for which they are suited
  - When receive awards, may subcontract through recursive negotiation
- **Managers:**
  - Evaluate bids and award contracts to nodes they determine to be most appropriate
- Can involve **multiple rounds** of negotiation

# Task Announcements

---

- Task Announcement:
  - Used by a node to initiate contract negotiation
  - Advertises that a task is available
  - Can be addressed to:
    - All nodes (general broadcast)
    - Subset of nodes (limited broadcast)
    - Single node (point-to-point)

**<task-announcement> →**

**TASK-ANNOUNCEMENT [name] {task-abstraction}**

**{eligibility-specification} {bid-specification} {expiration-time}**

# Composition of Task Announcement: Example

---

- To: \* (i.e., broadcast message)
- From: 25
- Type: TASK ANNOUNCEMENT
- Contract: 22-3-1
- Task Abstraction:
  - TASK TYPE SIGNAL
  - POSITION LAT 47N LONG 17E
- Eligibility Specifications:
  - MUST-HAVE SENSOR
  - MUST-HAVE POSITION AREA A
- Bid Specification:
  - POSITION LAT LONG
  - EVERY SENSOR NAME TYPE
- Expiration Time
  - 28 1730Z FEB 1979

# Common Internode Language

---

- **Common Internode Language:** single high-level language understandable to all nodes
- **This language + high-level programming language** (for transfer of procedures between nodes) **forms a common basis** for communicating slot information among the nodes.
- **Contract Net Protocol:**
  - Provides *type* of information that is needed
  - Leaves *content* of messages up to user

# Task Announcement Processing

---

- All tasks are **typed**
- For each task type, node maintains a **rank-ordered list** of announcements that have been received and have not yet expired
- Each node checks **eligibility specifications** of all task announcements it receives
- If node is eligible, then it **ranks the task** relative to others under consideration

# Bidding

---

- Node can bid when it is not performing a task (i.e., when it is idle)
- Idle node decides:
  - Whether to make a bid
  - Whether to wait on further task announcements
- “Node abstraction” slot of bid:
  - Filled with specification of capabilities of the node that are relevant to the announced task.
  - Can include REQUIRE statement, indicating that bidder needs more information

**<bid> → BID [name] {node-abstraction}**

# Bid Processing

---

- Manager:
  - Queues contracts locally until they can be awarded
  - Maintains rank-ordered list of bids received
  - Determines if any of the received bids are satisfactory
    - If so, contract is awarded to the bidder
    - Otherwise, manager waits for further bids
- If expiration time reached and no contract awarded, can do one of the following:
  - Award contract to most acceptable bidder(s)
  - Transmit another task announcement (if no bids have been received)
  - Wait for a time interval before transmitting another task announcement (if no acceptable bids have been received)

# Contract Awards

---

<announced-award> → ANNOUNCED-AWARD [name] {task-specification}

# Directed Awards

---

- Directed award:

- Manager knows exactly which node is appropriate
- No task announcements made
- No bids submitted

<directed-award> → DIRECTED-AWARD [name] {task-abstraction}  
{eligibility-specification} {task-specification}

- Acknowledgment: allows refusal

<acknowledgment> → ACCEPTANCE [name]  
→ REFUSAL [name] {refusal-justification}

# Contract Processing, Reporting Results, Termination

---

- Information message: used for general communication between manager and contractor

<information-message> → INFORMATION [name]  
{eligibility-specification} {information-specification}

- Request: used to request information

<request-message> → REQUEST [name]  
{eligibility-specification} {request-specification}

- Report: used by contractor to inform manager (and perhaps other report recipients) that a task has been partially or completely executed

<report> → INTERIM-REPORT [name] {result-description}  
FINAL-REPORT [name] {result-description}

- Termination message: allows manager to terminate contracts

<termination> → TERMINATION [name]

# Negotiation Tradeoffs

---

- Contract Net Protocol:
  - Allows a node to submit at most one bid at each opportunity
  - Only idle nodes can submit bids
    - Reduces message traffic and delay in allocating tasks

# Immediate Response Bids

---

- Problems with only allowing idle nodes to bid:
  - Node that issues task announcement might not receive any bids, because:
    1. There are no idle nodes
    2. Some node is both idle and eligible, but ranks the task too low
    3. No node is capable of working on the task, even if it were idle
- Cases 1-2: Can re-issue task announcements until a bid is obtained
- Case 3: Re-issuing announcements is pointless
  - Therefore, “immediate response” bids
  - Enable node to indicate BUSY, INELIGIBLE, or LOW RANKING

# Node Available Messages

---

- When processing load is high, most task announcements will not be answered because all nodes will be busy
- Therefore, use *node available* message

**<node-available-message> →**

**NODE-AVAILABLE {eligibility-specification} {node-abstraction}  
[expiration-time]**

# Acquiring Contracts

---

1. A node can wait for a suitable task announcement and submit a bid
2. A node can transmit a node available message and wait for a directed award

If net is not heavily loaded, use of task announcement is always warranted.

If net is heavily loaded, node available messages are preferred.

# Task Processing Procedures

---

- Announcement Procedure
- Announcement Ranking Procedure
- Bid Procedure
- Bid Ranking Procedure
- Award Procedure
- Acknowledgment Procedure
- Refusal Processing Procedure
- Report Acceptance Procedure
- Termination Procedure
- Information Acceptance Procedure
- Execution Procedure

# Contract Net Protocol Contributions

---

- Main contribution:
  - Mechanism it offers for structuring high-level interactions between nodes for cooperative task execution
  - Stresses utility of negotiation as an interaction mechanism

# Preview of Next Class

---

- Task Allocation – Market Methods