Dynamic Task Allocation in Heterogeneous Multi-Robot Teams

Dr. Lynne E. Parker

Professor and Associate Head

Fellow, IEEE

Dept. of Electrical Engineering and Computer Science

University of Tennessee, Knoxville

leparker@utk.edu

http://web.eecs.utk.edu/~parker

August 17, 2010

Outline

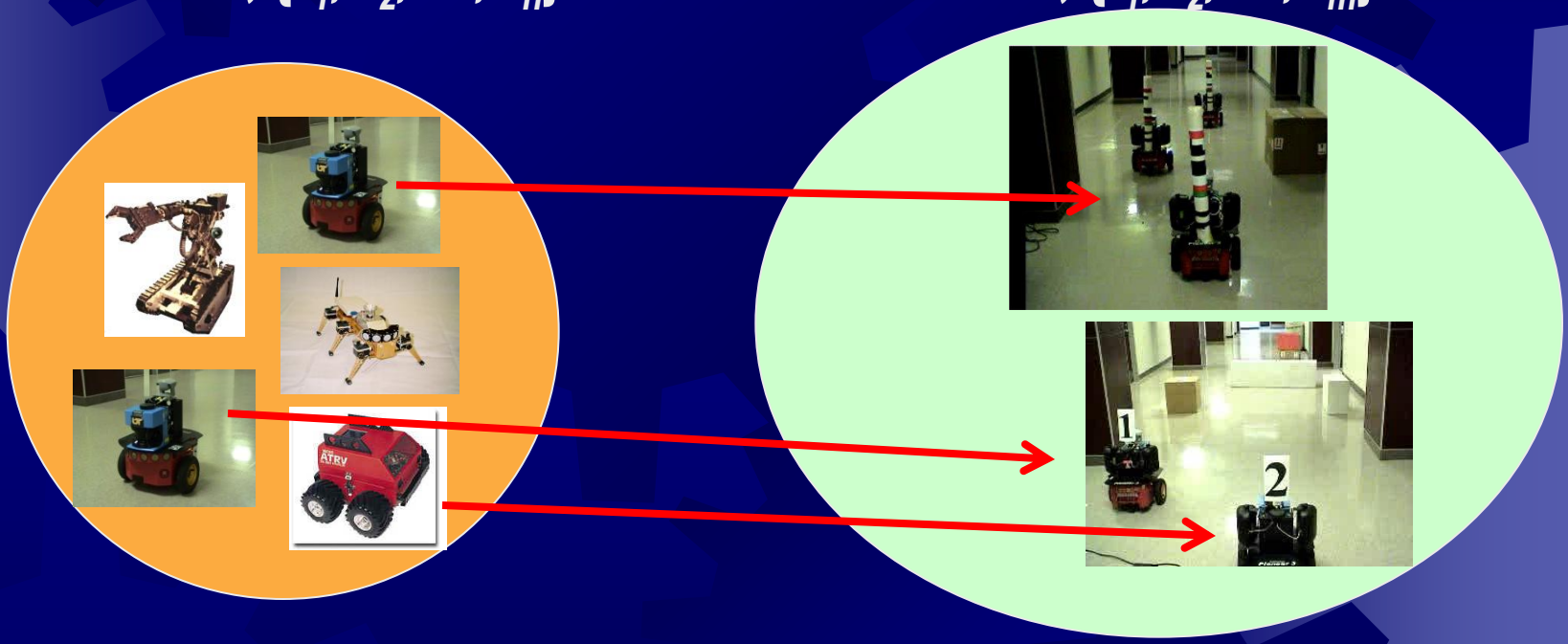
- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

Outline

- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

Defining Task Allocation

- **Task allocation** is the problem of determining which robot should perform which task(s)
- **Given:** *n* robots, $\{r_1, r_2, \ldots, r_n\}$ *m* tasks, $\{t_1, t_2, \ldots, t_m\}$



- **Objective:** find mapping of tasks to robots, optimizing the objective function (e.g., quickest task completion time, min energy, etc.)

The Challenge of Task Allocation

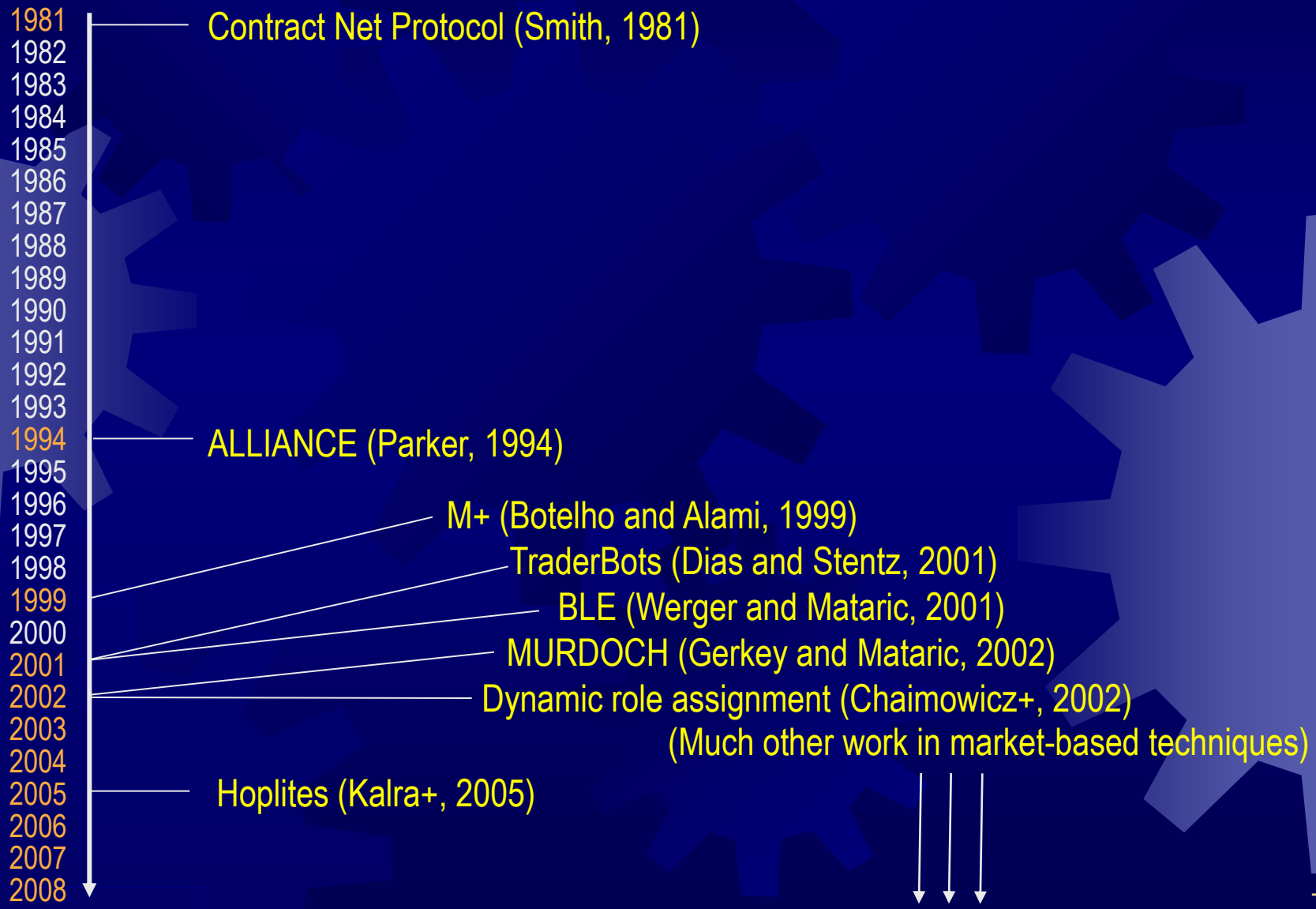
- The most practical variant of this problem (i.e, handling multiple tasks simultaneously, with heterogeneous robot capabilities) is easily shown to be NP-hard

- Thus, current techniques are approximate, not exact, solutions

Outline

- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

History of Multi-Robot Task Allocation



Outline

- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

Gerkey's Taxonomy for Task Allocation

- **Tasks:** single-robot (SR) or multi-robot (MR)
- **Robots:** single-task (ST) or multi-task (MT)
- **Assignments:** instantaneous (IA) or time-extended (TA)

Combine these 3 axes into a single descriptive, such as:

- **SR-ST-TA:** Single-robot tasks, single-task robots, with time-extended assignment
- **MR-ST-IA:** Multi-robot tasks, single-task robots, instantaneous assignment

Example Architectures within the Taxonomy

- **ST-SR-IA:** ALLIANCE, BLE, M+, MURDOCH
- **ST-SR-TA:** ALLIANCE, TraderBots
- **ST-MR-IA:** ASyMTRe
- **ST-MR-TA:** combinatorial auction work (e.g., by Dias, by Koenig, etc.)
- **MT-SR-IA / MT-SR-TA:** Uncommon (requires robots to each concurrently execute multiple tasks)
- **MT-MR-IA:** Vig & Adams

Most work: ST-SR-IA and ST-SR-TA

- Today, we'll examine 4 approaches:
  - ALLIANCE (Parker, 1994)
    [ST-SR-IA/TA, Behavior-based]
  - MURDOCH (Gerkey & Mataric, 2002)
    [ST-SR-IA, Publish/subscribe]
  - TraderBots (Dias and Stentz, 2002)
    [ST-SR-TA, Market-based]
  - ASyMTRe (Parker and Tang, 2006)
    [ST-MR-TA, Coalition formation]

Outline

- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

ALLIANCE [An ST-SR-IA/TA alg.]

- Developed by Parker (MIT) in 1994
- Key features of ALLIANCE:
  - Fully distributed
  - Behavior-based
  - Works with heterogeneous robots
  - Enables dynamic task-reallocation
  - Reduces communication overhead; no negotiations
  - Uses mathematical motivation models, *impatience* and *acquiescence*, towards adaptive action selection
  - Implemented on teams of physical robots

Assumptions in ALLIANCE

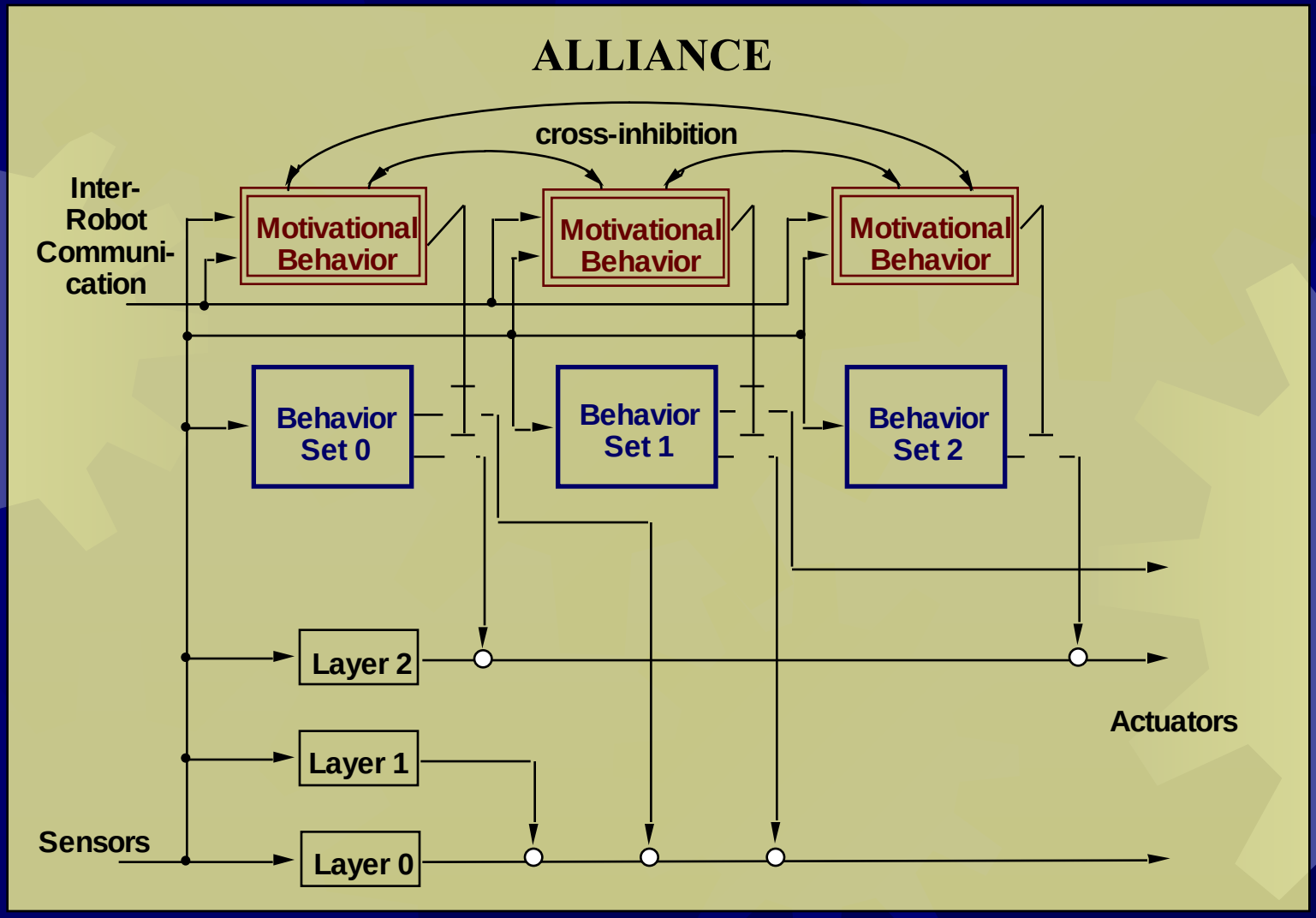
- Robots can detect the effects of their own actions.
- Robot r_i can detect the actions of other team members through explicit communication.
- Robots on the team are not intentionally adversarial.
- Tobots do not possess perfect sensors.
- Any of the robot subsystems can fail.
- The communication medium is not guaranteed to be available.
- Robot failure cannot necessarily be communicated to other robots.
- The robots do not have complete world knowledge.

Note : The assumptions are made with respect to small to medium sized team of multi-robots.

Overview of ALLIANCE

- Overall mission is decomposed into a set of high level tasks.
- High level tasks are achieved by means of a number of *behavior sets* that an individual robot is capable of executing.
- Behavior sets are classified as *active*, if robot is executing that behavior set, or *hibernating*, if otherwise.
- Only one behavior set is active at any point in time.
- The selection of the behavior set is done by means of *motivational behaviors*, each of which controls the activation of one behavior set.

ALLIANCE Architecture



Motivational Behaviors

- ALLIANCE uses *motivation* for task monitoring and dynamic task reallocation.
- Each motivational behavior receives input from a number of sources including:
  - Sensory feedback
  - Inter-robot communication
  - Inhibitory feedback
  - Internal motivations.

  These inputs are used to generate the output at any point of time.
- The output defines the *activation level* of each behavior.
- Once the activation level exceeds the preset threshold for each behavior, the behavior is *activated*.
- ALLIANCE uses 2 types of internal motivation: *impatience* and *acquiescence*
  - *Impatience*: enables the robot to handle situations external to itself.
  - *Acquiescence*: enables the robot to handle internal situations.

Motivational Behaviors (con't.)

- A robot's motivation value to activate a behavior is initialized to 0.
- Over a period of time the robot's motivation level increases at a rate that depends on the activities of its teammates:
  - If no robot is accomplishing a behavior, then the motivation level increases at a fast rate of impatience.
  - If another robot is working on the behavior then the motivational level increases at a slower rate of impatience.
- At the same time the robot's willingness to give up a task increases over time as long as the sensory task indicates the task is not being accomplished.

ALLIANCE Formal Model

$R = \{r_1, r_2, ..., r_n\}$	n robots
$T = \{task_1, task_2, ..., task_m\}$	m independent subtasks
$A_i = \{a_{i1}, a_{i2}, ...\}$	Behavior sets of robot r_i
$h_i(a_{ik})$	Task in T that r_i is working on when a_{ik} is active
θ	Threshold of activation

$$sensory\_feedback_{ij}(t) = \begin{cases} 1 & \text{If sensory feedback of } r_i \text{ at time } t \text{ indicates that } a_{ij} \textit{ is applicable} \\ 0 & \text{Otherwise} \end{cases}$$

$$comm\_received(i,k,j,t_1,t_2) = \begin{cases} 1 & \text{If } r_i \text{ has received message from } r_k \text{ concerning task } h_i(a_{ij}) \text{ in } (t_1,t_2) \\ 0 & \text{Otherwise} \end{cases}$$

$$activity\_suppression_{ij}(t) = \begin{cases} 0 & \text{If } a_{ij} \text{ is active, } k \neq j, \text{ on robot } r_i \text{ at time } t \\ 1 & \text{Otherwise} \end{cases}$$

ALLIANCE Formal Model (con't.)

$$impatience_{ij}(t) = \begin{cases} \min_k(\delta\_slow_{ij}(k,t)), & if \quad (comm\_received(i,k,j,t-\tau_i,t)=1) \quad and \\ & (comm\_received(i,k,j,0,t-\phi_{ij}(k,t))=0) \\ \delta\_fast_{ij}(t), & otherwise \end{cases}$$

$$impatience\_reset_{ij}(t) = \begin{cases} 0, & if \quad \exists k.((comm\_received(i,k,j,t-\delta t,t)=1) \, and \\ & (comm\_received(i,k,j,0,t-\delta t)=0), \\ 1, & otherwise \end{cases}$$

$$acquiescence_{ij}(t) = \begin{cases} 0 & \begin{array}{l} if \quad ((a_{ij}(t) \quad active \quad more \quad than \, \varphi_{ij}(t)) \\ \quad and \, (\exists x.comm\_received(i,x,j,t-\tau_i,t)=1)) \\ or \, (a_{ij}(t) \, active \, more \, than \, \lambda_{ij}(t)) \end{array} \\ 1 & otherwise \end{cases}$$

ALLIANCE Formal Model (con't.)

$$m_{ij}(0) = 0$$

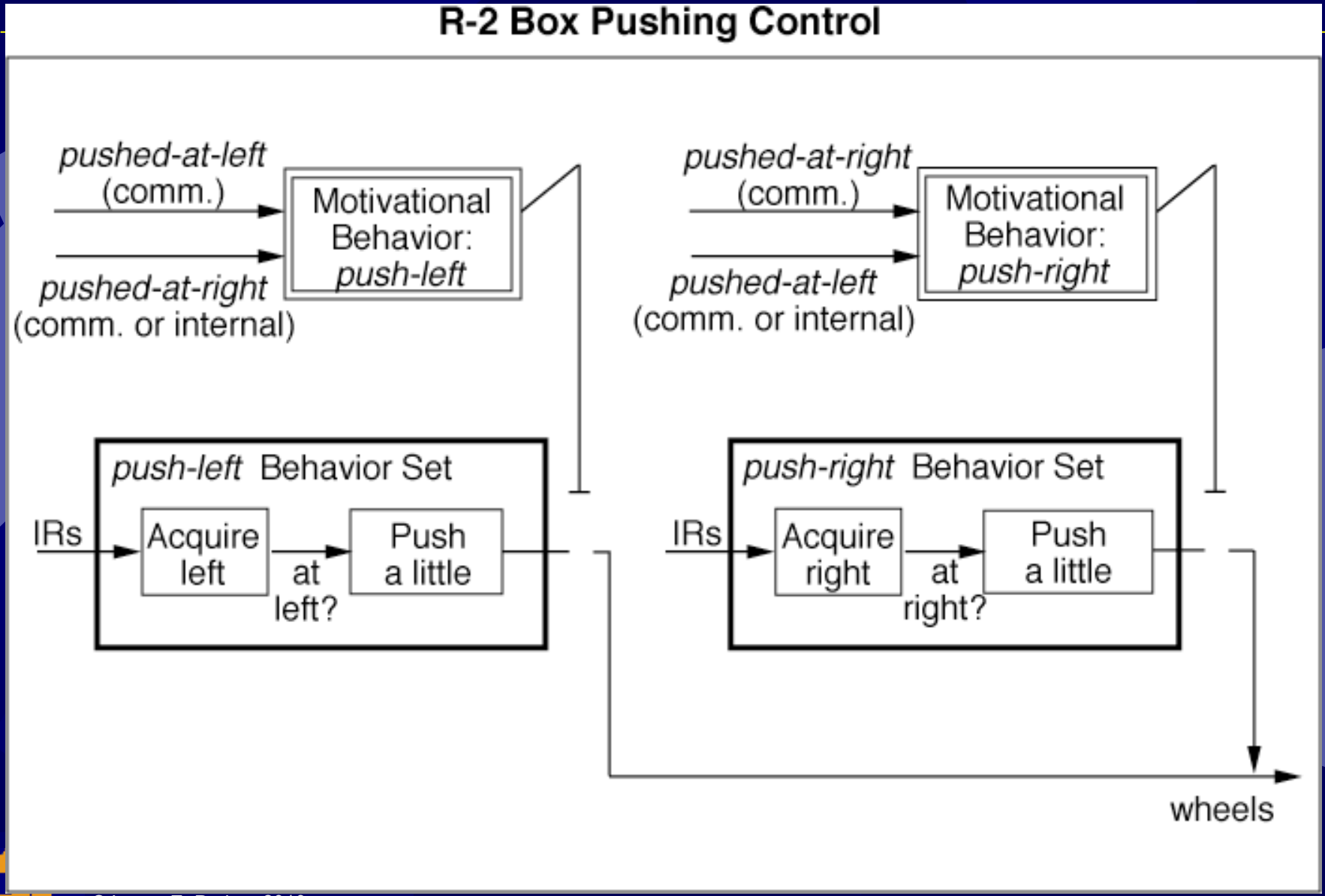
$$m_{ij}(t) = [m_{ij}(t-1) + impatience_{ij}(t)]$$
$$\times \quad sensory\_feedback_{ij}(t)$$
$$\times \quad activity\_spression_{ij}(t)$$
$$\times \quad impatience\_reset_{ij}(t)$$
$$\times \quad acquiescence_{ij}(t)$$

Whenever $m_{ij}(t) > \theta$, a_{ij} is activated.

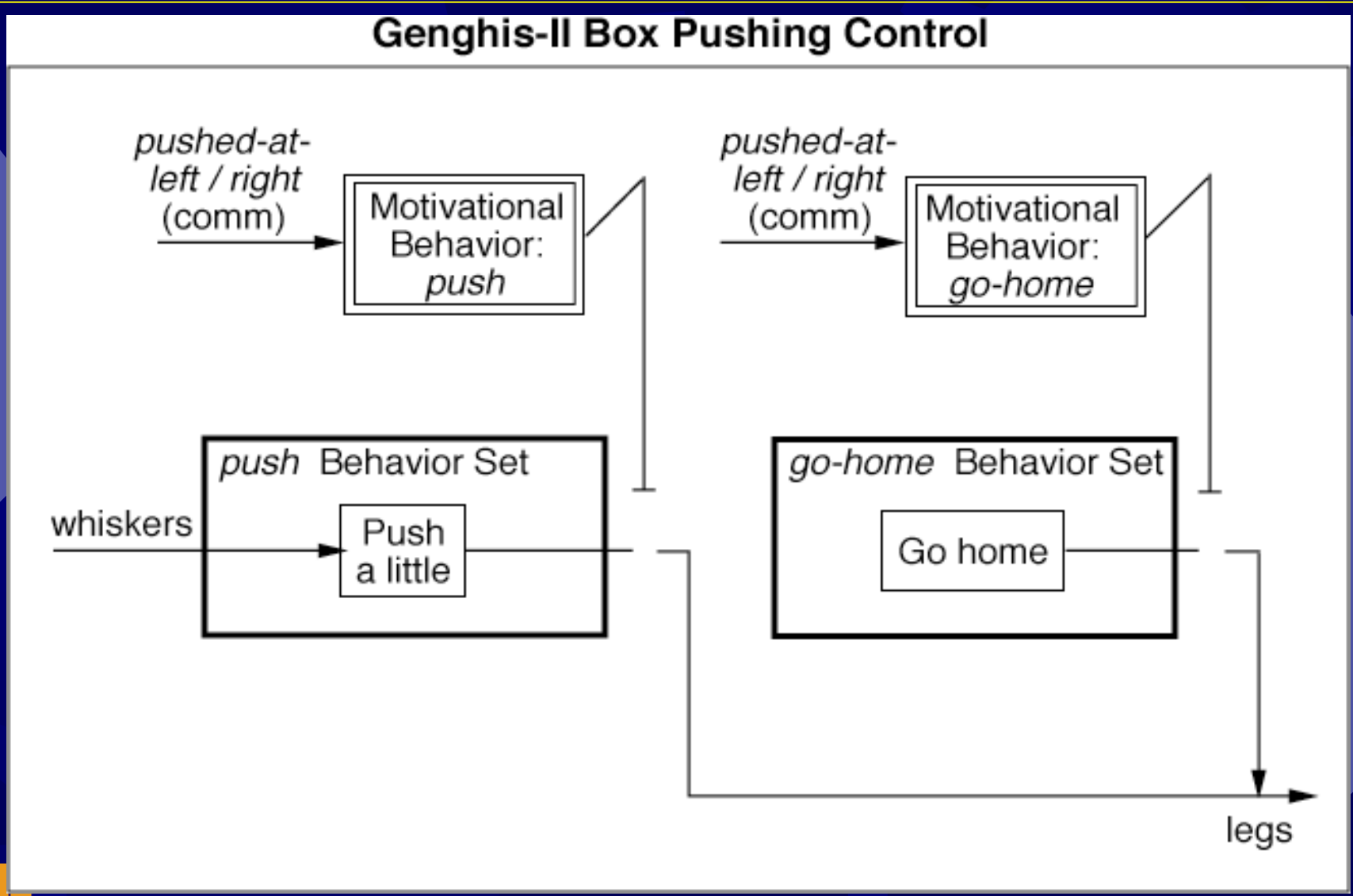
Example Adaptive Box Pushing

Parker, MIT, 1994

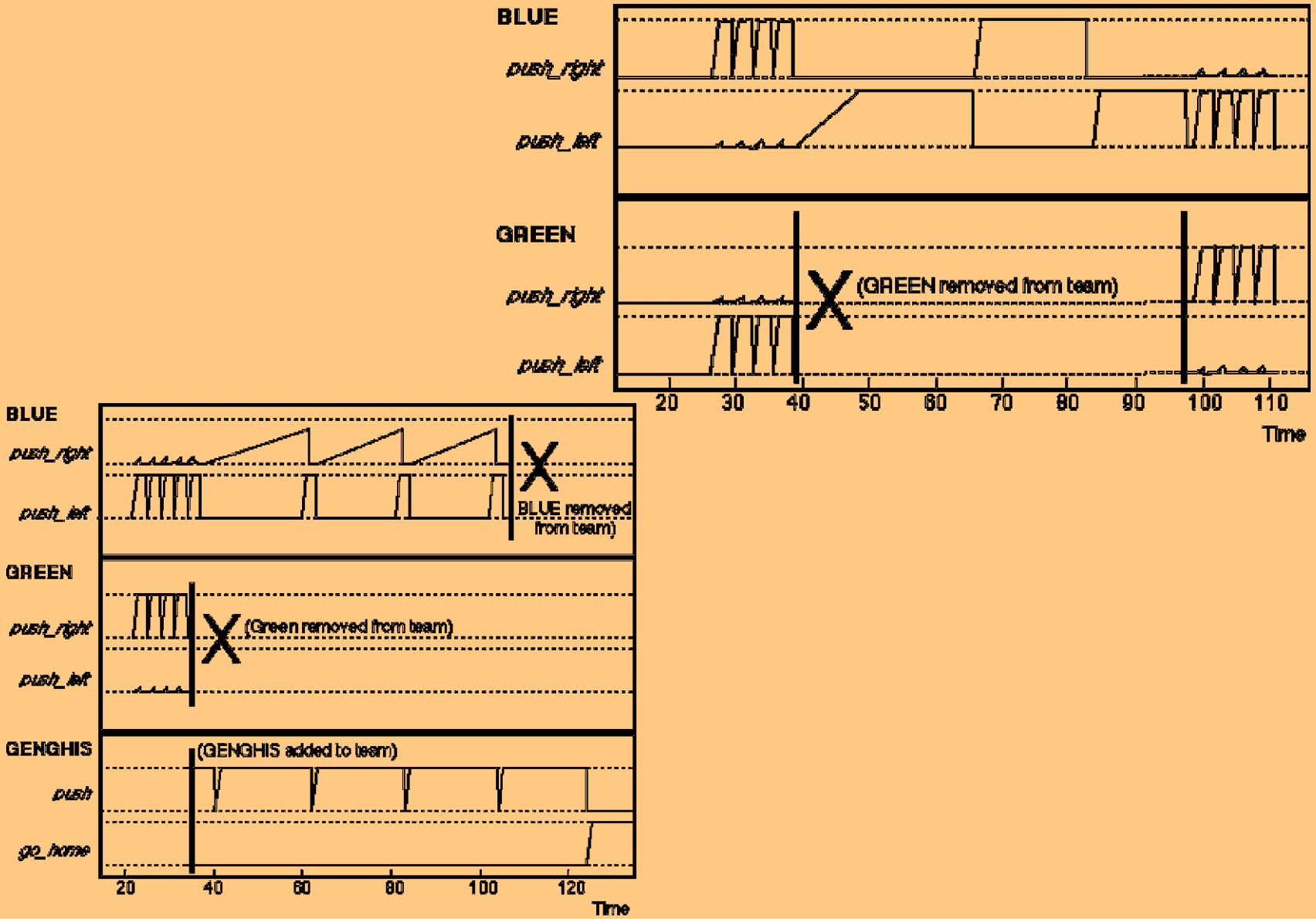
Robot Control in Box Pushing



Robot Control in Box Pushing (con't.)

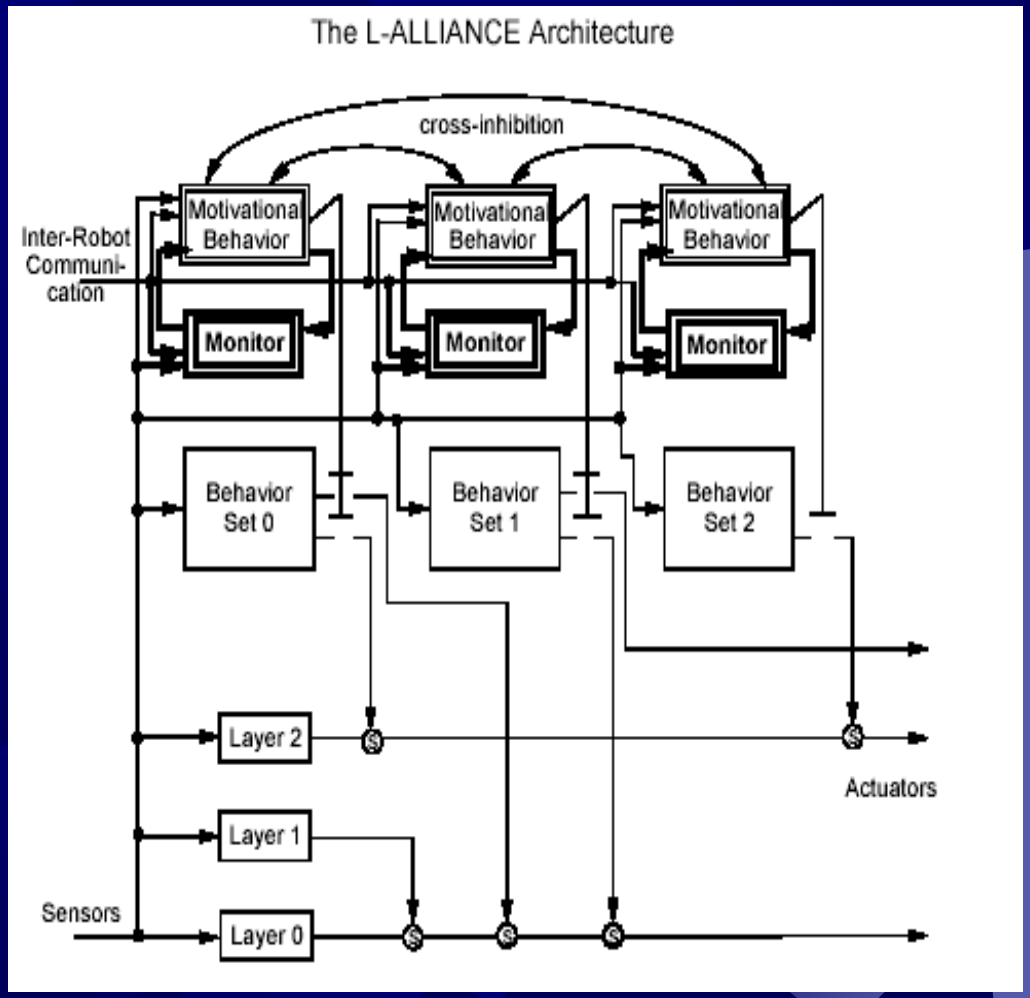


Typical Behavior Traces in ALLIANCE



L-ALLIANCE

- Dynamically updates the parameter settings based upon knowledge learned from previous experiences.
- Each robot 'observes', evaluates and cataloges the performance of any team member whenever it performs a task of interest to that robot.
- These 'learned' observations allow the robot to adapt their action selection over time.
- The underlying algorithm is distributed across the behavior sets of ALLIANCE.

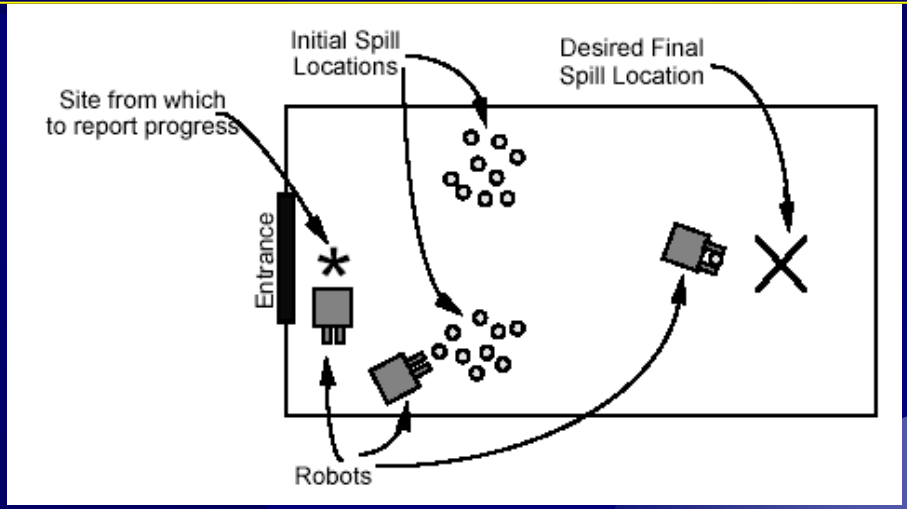


More Experiments

- Experiments conducted on physical robots: *teams of 3 R-2 robots were used in all experiments.*

- *Hazardous waste cleanup mission*
  - Mission requires two artificially 'hazardous' waste spills in an enclosed room to be cleaned up by a team of three robots.

  - The robot team must locate the two waste spills, move the spills to a goal location, while also periodically reporting the team progress to humans monitoring the system.



Videos of Cleanup Task using ALLIANCE

Parker, MIT, 1994

Summary of ALLIANCE Results

- The cooperative team under ALLIANCE was robust
- The team was able to respond autonomously to various types of unexpected events either in the environment or in the robot team without the need for external intervention.
- The cooperative team need not have a priori knowledge of the abilities of the other team members to efficiently complete the task.
- ALLAINCE allows the robot teams to accomplish their missions even when communication system breaks down.

Summary of ALLIANCE

- ALLIANCE is a fully distributed, behavior based approach for fault tolerant mobile-robot cooperation.
- ALLIANCE enhances team robustness through usage of motivational behavior mechanism.
- Physical redundancy can be used to enhance fault tolerance of the system.
- L-ALLIANCE enhances ALLIANCE architecture by using learning algorithm to fine tune the impatience and acquiescence parameters.
- The architecture has been implemented on a team of physical robots, thereby illustrating its feasibility.

Outline

- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

Another Task Allocation Approach: MURDOCH (Gerkey '02) [An ST-SR-IA algorithm]

- Anonymous communication via broadcast
  - saves bandwidth when sending messages to multiple recipients
  - allows robots to move in and out of range
- Hierarchical task structure
  - each task is a tree containing other tasks
  - flexible enough to handle a wide variety of tasks
- Auctions
  - scalable
  - cheap to broadcast and compute (only one round of bidding)
  - allows modularization
  - similar to CNP negotiation scheme, but without centralized broker

MURDOCH Approach

- Publish/Subscribe messaging
  - *subject-based addressing*: messages addressed by content rather than by destination
  - a data-producer tags a message with a particular subject
  - only data-consumers interested in the specified subject will receive the message
  - subjects represent a robot's resources
  - resources can be:
    - physical devices (e.g., `camera`, `gripper`)
    - high-level capabilities (e.g., `mobile`, `tracking`)
    - abstracted notions of current state (e.g., `idle`, `pushing-box`)
  - for example, to send a message to all robots capable of retrieving a red can, a subject such as {`mobile`, `camera`, `gripper`, `idle`} would be appropriate

MURDOCH (con't.)

- Auction Protocol
  - A task can be introduced to the system by a human, an automated task generator, a higher level task already in progress, or many other ways
  - Each new task triggers a 5-step auction:
    1) task announcement - an agent acts as "auctioneer", publishing an `announcement` containing the details of the task and with an appropriate subject
    2) metric evaluation - the `announcement` contains metric(s) to determine task fitness
    3) bid submission - each candidate robot calculates and publishes its "score" as a `bid`
    4) close of auction - the auctioneer processes the `bids` and sends a `close` message, the winner receives a time-limited contract to execute the task
    5) progress monitoring/contract renewal - the auctioneer monitors task progress and continues to send `renewal` messages to the winner as long as progress is satisfactory, the winner replies to each `renewal` with an `acknowledge` message

MURDOCH (con't.)

- Time-limited contracts provide fault tolerance
- Tasks are always assigned to the most capable robot, thus MURDOCH is an instantaneous greedy task scheduler
- Compared to a centralized task allocation system:
  - PRO: tasks may be randomly input at anytime
  - CON: use of resources can not be optimized by analyzing concurrent tasks

Testing of Murdoch

- Closed indoor environment
- Pioneer 2-DX mobile robots
  - Sensors (each robot has one or two): camera, laser range finder, tactile bumper
- Wireless ethernet with shared bandwidth of ~1.9 Mb/s, allowing robots to communicate freely with one another at all times

Murdoch Testing (con't.)

- Loosely coupled task allocation
  - long-term autonomy
  - randomly generated sequence of tasks (box pushing and target tracking)
  - overhead-camera attached to desktop PC used as the sole auctioneer

- Results
  - system ran successfully over a period of about 3 hours
  - resources allocated efficiently, i.e., the most capable robot available was always assigned the new task
  - bandwidth usage was very small, implying good scalability

Murdoch Testing (con't.)

- Box pushing
  - requires tightly coupled cooperation
  - system composed of "watchers" and "pushers"
  - the box is moved via the `pusher-watcher` approach:
    - watcher stays in front of box and measures angular error between box and goal
    - pushers move forward in such a way that angular error is reduced

  - a watcher task is auctioned by the system whenever a box must be moved
  - this watcher auctions two pusher tasks based on angular error
  - the side of the box lagging behind gets the first pusher, thus if only one pusher is available it will continually switch sides as one falls behind the other

Murdoch Testing (con't.)

- Results
  - Out of 40 trial runs, task completed successfully 90% of the time
  - Partial pusher failure much more time-consuming than total pusher failure because the watcher had to recognize lack of progress, rather than simply lack of acknowledgement or resources
  - Allowing a failed pusher to rejoin proved time-saving
  - The box was kept along a near-ideal trajectory

Summary of MURDOCH

- Fully distributed method of task allocation
- Anonymous, resource-centric communication
- Hierarchical task structure
- Each new task auctioned to the most capable agent available
- Reactive to environmental changes such as robot failure and randomly introduced new tasks
- Empirically demonstrated on physical robots in situations involving both tightly coupled cooperation and long-term loosely coupled cooperation

Outline

- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

TraderBots [An ST-SR-TA alg.]

- Developed by Dias and Stentz (CMU, 2002)
- Key Characteristics:
  - Market-based approach
  - Revenue and cost functions defined across possible robot plans
  - Task divided into sub-tasks
  - Robots bid and negotiate to carry out sub-tasks
  - Robots aim to maximize their personal profits

TraderBots -- Free Market System

- Inspired by (human-based) free market economies
  - Agents coordinate to produce goods
  - Individuals are free to exchange goods and services, and enter into contracts as they see fit
  - While individuals are self-interested, aggregate effect is highly productive society
  - Individuals are in best position to understand their needs and the means to satisfy them
    - Individuals reap direct benefits of own good decisions, and suffer direct consequences of bad decisions
    - Can cooperate with others to achieve outcome greater than individual alone

Implementing Free Market Approach in Robots

- Revenues and Costs:
  - trev() : function that maps possible task outcomes onto revenue values
  - tcost() : function mapping possible schemes for performing task onto cost values
  - Teams' goal is to execute plan P, such that profit

    profit: trev(P) - tcost(P)

    is maximized

- An individual's contribution to the team's profit are measured using it's own revenue and cost functions:
  - rrev() : function that maps individual's possible task outcomes onto revenue values
  - rcost() : function mapping individual's possible schemes for performing task onto cost values

Role of Price and the Bidding Process

- Robots can help other robots by providing goods or services
- Robots have incentive to collaborate if they can produce more aggregate profit together than apart
- Example:
  - Assume robot A can make additional profit of value X if robot B performs a service for A.
  - Assume it costs robot B an amount of Y to perform that service.
  - If X > Y, then both parties have incentive to work together
  - But, how to distribute composite profit (X-Y)?
    - "Fair" value may be hidden or complex
    - Thus, allow robots to *bid* for a good or service until a mutually acceptable price is found
- Robots can negotiate several potential deals simultaneously
- Price is a low-bandwidth encoding of communicating aggregate information about costs

Example implementation: Exploration

- Free market architecture used as coordination mechanism for multi-robot exploration
- Robots negotiate and exchange tasks for revenue in order to obtain profitable tours (i.e., through the environment)
- By using the negotiation process to continually improve their tours, the tendency is for the robots to cover the environment quickly while remaining far enough apart that there is little repeated coverage.
- Using multiple robots gives a faster (i.e., robots can act in parallel), more robust (i.e., no single point of failure) solution.

Exploration: Details of Market Approach

- Objective: build grid-based map of terrain
- Cost Model:
  - Minimize travel distance
- Revenue Model:
  - Expected amount of new information to be obtained by visiting a goal point. The information gained is from the discovery of new occupancy grid values (new terrain).
- Distributed negotiations:
  - Robots exchange tasks and revenue with one another without any reliance on a central agent to coordinate the task.
- Robustness to communication/robot failure:
  - Communications are always sent to all robots that are reachable. If a robot drops off the network due to moving out of communication range or robot failure, the other robots will simply cease negotiating with that robot until it is redetected.

Negotiation Protocol for Exploration Task

- Negotiation protocol:
  1. Robot finds a list of "interesting" goal points (goal points in regions of unknown terrain)
  2. Robot greedily arranges all of the goal points into a tour
  3. Robot announces an auction for each goal point in its tour; other robots will bid on these tasks by calculating their expected profits from adding this goal to their own tours
  4. If another robot bids more money than the auctioneer expected to earn by performing this task, the highest bidder is awarded the task in exchange for the price of the bid
  5. Once all tasks have been offered, the robot will begin its tour by visiting its first goal point
  6. Upon arrival at a each goal point, the robot will go to step 1
- Note: negotiations are completely asynchronous -- a robot may receive calls for bids or other messages in any state, and thus tours are continuously being updated/improved

Goal point selection strategies

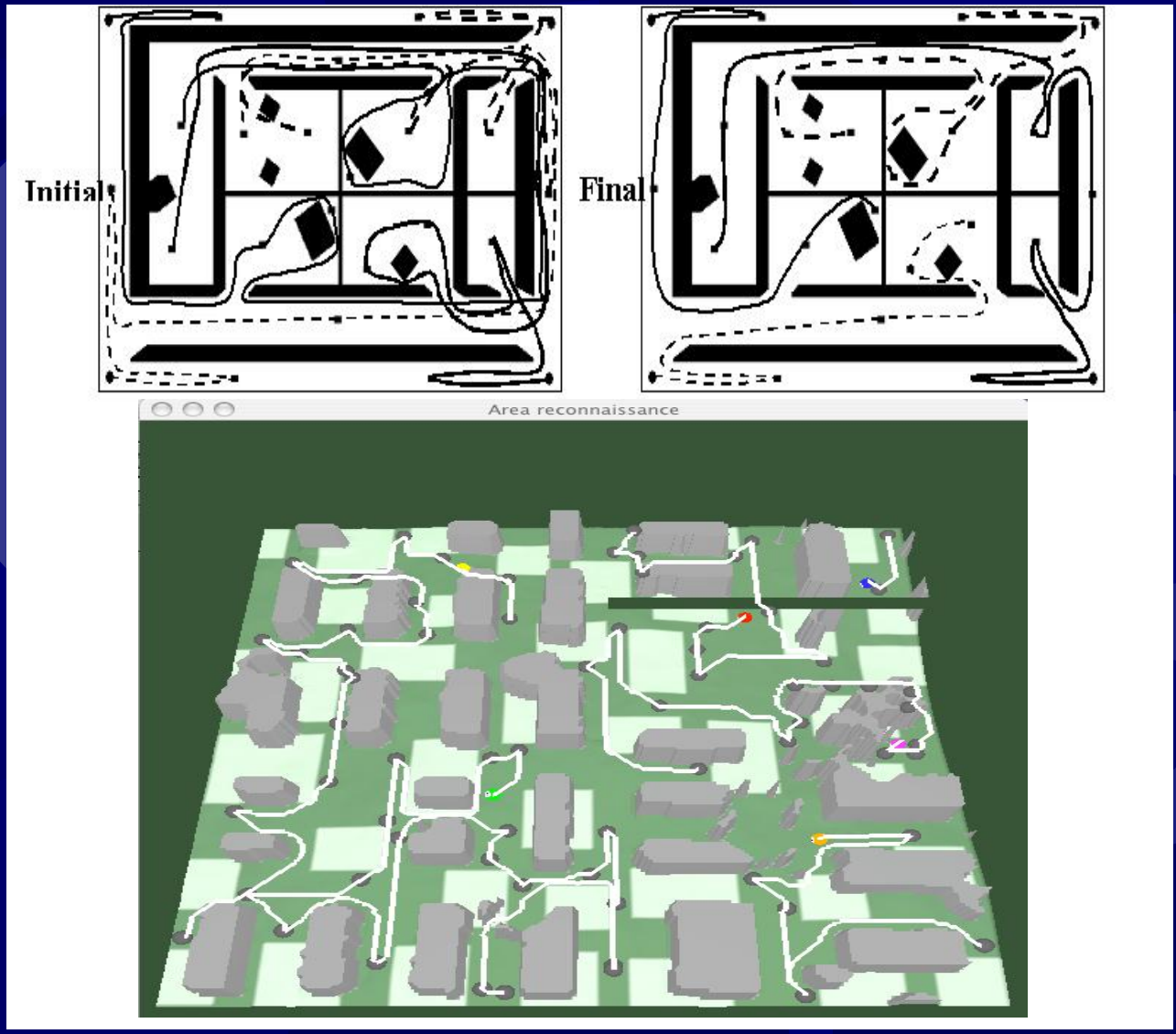
- **Random:** The goal points are generated at random places.
- **Greedy:** The goal generated is the closest unexplored region to the robot.
- **Uniform grid coverage:** Goals are arranged in a grid, equally spaced. The goals can be placed at approximately one sensor radius apart to ensure complete coverage of the environment.
- **Quadtree space decomposition:** The goals are generated at the centers of quadtree leaves, which are further subdivided if the fraction of cells within a leaf are that are known/unknown fall below a certain threshold.

Note: in the Exploration experiments reported shortly, Random and Quadtree performed best, followed (distantly) by Greedy

Methods of Information Sharing

- Through negotiation:
  - Bidder can inform auctioneer if an area being offered is already mapped, and the auctioneer will in turn cancel the auction
  - Robots tend to be spaced apart. Thus, if a goal falls into a region being covered by a different robot, that robot should have the lowest cost to visit that goal, and thus should win an auction for that goal
- Explicit:
  - Robots periodically share sections of their maps

TraderBots Results for Exploration Task



Summary of TraderBots

- Free market architecture for distributed control of multi-robot teams performing decomposable tasks
- Revenue function rewards robots for performing subtasks
- Robots negotiate to minimize costs and maximize profits
- Tested in applications such as exploration/mapping, traveling salesman, etc.

Outline

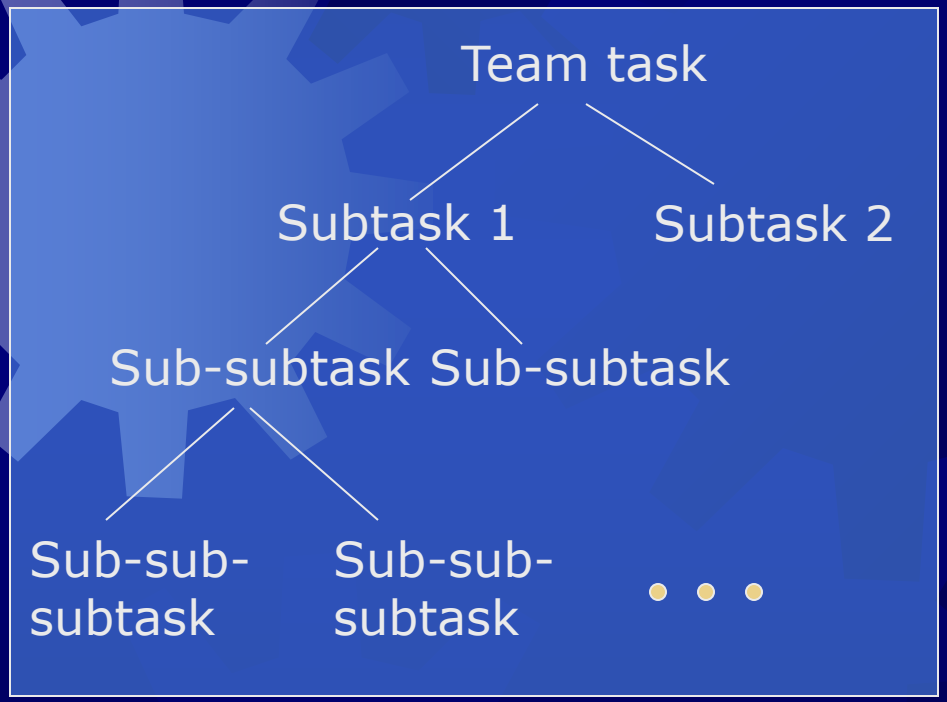
- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

ASyMTRe [An ST-MR-TA alg.]

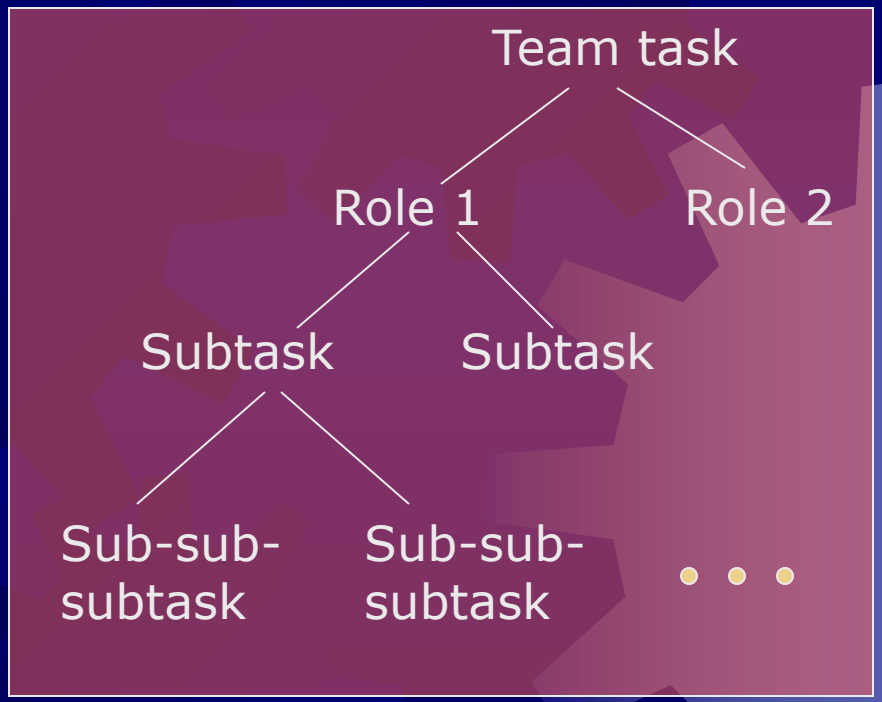
- Developed by Parker and Tang (Univ. Tenn., 2006)
- "ASyMTRe": Automated Synthesis of Multi-robot Tasks through software Reconfiguration
  - Pronounced "Asymmetry"
- Key characteristics:
  - Forms *coalitions* of robots to cooperatively solve tasks
  - Allows automatic sharing of sensors across multiple robots
  - Automatically reconfigures low-level building blocks (schemas) to solve tasks

Recall: Typically, Multi-Robot Task Decomposition or Roles Defined in Advance

Task Decomposition Tree



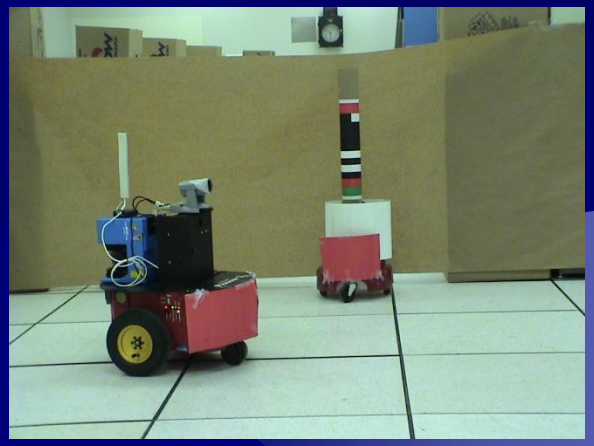
Role Breakdown

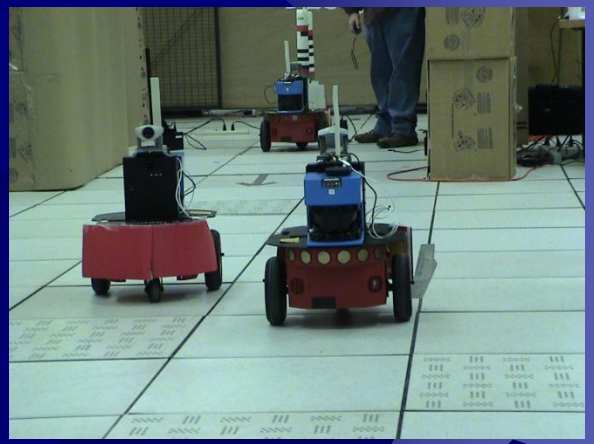


Dynamic action selection typically determines which robot performs which (sub)task or role

But, dynamic team composition can make pre-definition difficult

- Robot team goal (i.e., *what to do*) is known
- Robot task solution strategy (i.e., *how to do it*) is dependent on team capabilities
- Team capabilities evolve during the mission
- Robots are heterogeneous, not interchangeable, and may not be self-sufficient regarding achievement of team goals





Parker, "The Effect of Heterogeneity in Teams of 100+ Robots", *Proc. of 2nd NRL International Workshop on Multi-Robot Systems,* 2003.

Also Related to Research Issue: How to enable robots to share heterogeneous resources?

- Fixed roles, e.g. for coop. construction:

Autonomous Assembly (Singh, et al., CMU, 2000+)

- Publish/Subscribe approaches, e.g.: MURDOCH
  - Tasks require certain sensor requirements
  - Sharing occurs by publishing requirements and subscribing if possess relevant capabilities

MURDOCH (Gerkey, et al., USC, 2002)

Sharing heterogeneous resources (con't.)

- Dynamic sensor sharing, e.g.:

"I need my global pos."

"I'll compute it for you using my vision-based relative localization of you, plus my own global pos."

ASyMTRe, for need-based, dynamic sensor sharing (Parker, et al., UT, 2006+)

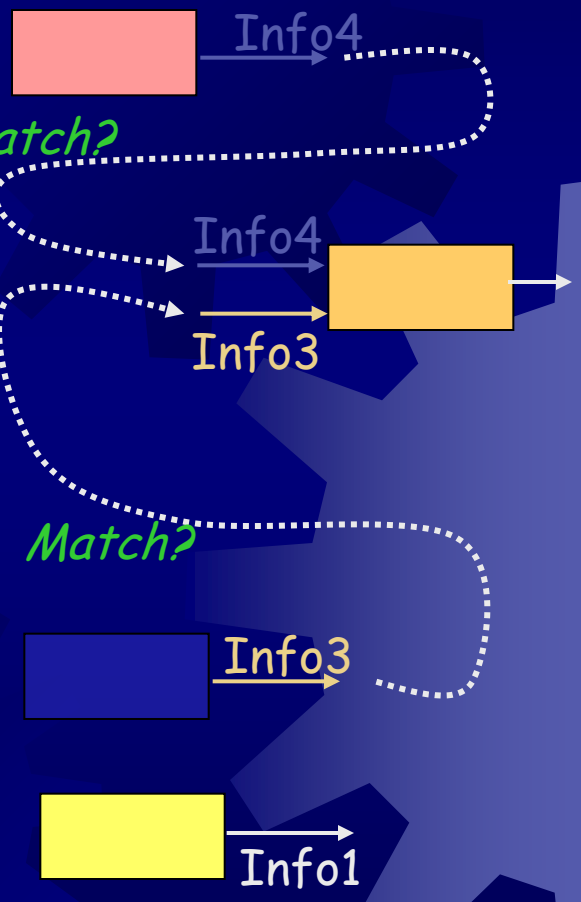
Our ASyMTRe Approach: Automated Task Solution Synthesis

- Develop autonomously reconfigurable "building blocks" (schemas)
  - Distributed across networked robots
- Different interconnections → different task solutions
- Heterogeneous robots can share capabilities
- Autonomous reconfigurability → robots can determine best task solution given team composition

Parker and Tang, "Building Multi-Robot Coalitions through Automated Task Solution Synthesis", *Proceedings of the IEEE, 2006.*

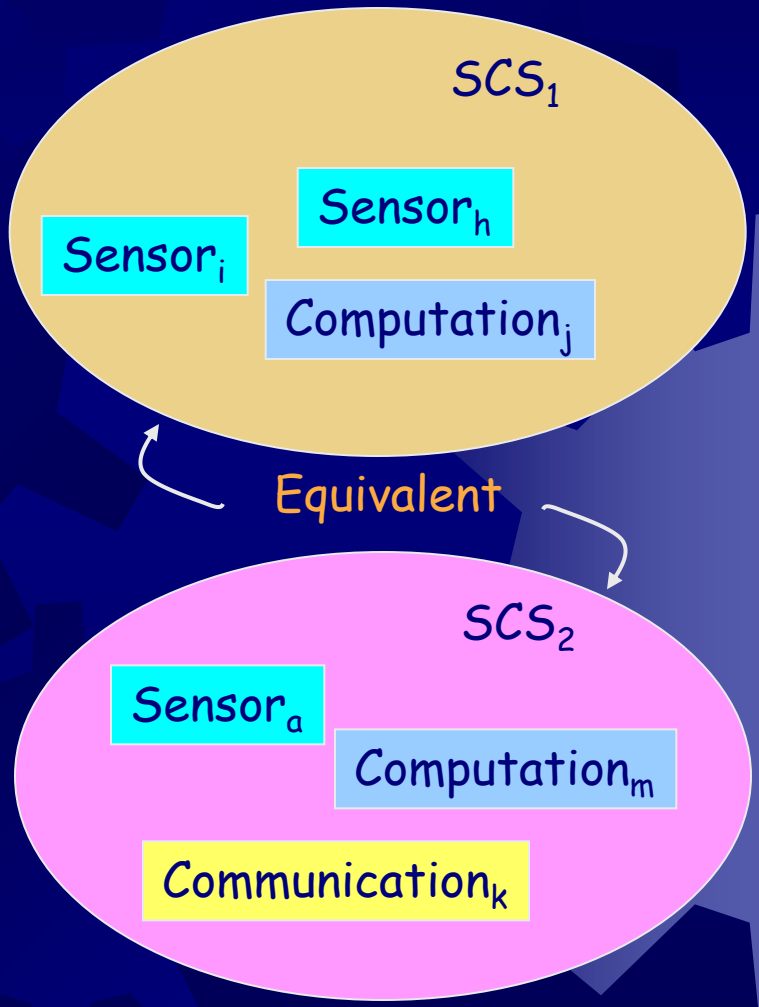
ASyMTRe General Idea: Reconfigurable Building Blocks

- Provide each robot with low-level "building blocks" that process information.
- Inputs and outputs of each building block have corresponding "information labels".
- Information labels define the type of information used by (input), or generated by (output), each building block.
- The source of the information is not important – can come from any process that generates that information.
- Robots can autonomously connect inputs and outputs of building blocks (whether on a single robot or on multiple robots) based on information content.



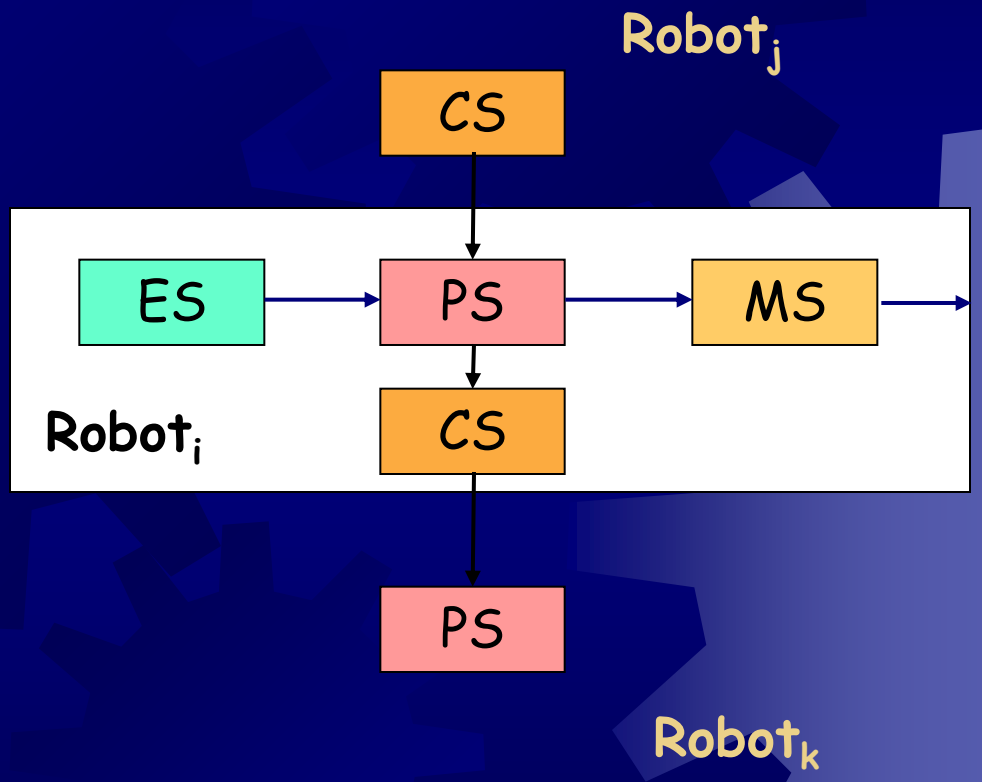
Our Inspiration: Information Invariants (Donald, *et al.*, '93)

- Showed equivalences between different combinations of sensing, effectors, and computation, and communication:
  - Robot teams can achieve same goal, but with very different ways of accomplishing that goal
  - Sensori-Computational System (SCS): module that computes a function of its inputs and current pose or position; composed of specific sensor and computational unit.



"Building Blocks" Based on Motor Schemas

- Fundamental robot capabilities defined using Arkin's Motor Schema approach:
  - Environmental Sensors (ES): sensors available on robot
  - Perceptual Schemas (PS): interpret sensor data
  - Motor Schemas (MS): control effectors to achieve desired behaviors
- Our new addition:
  - Communication Schemas (CS): communicate PS data from one robot to another

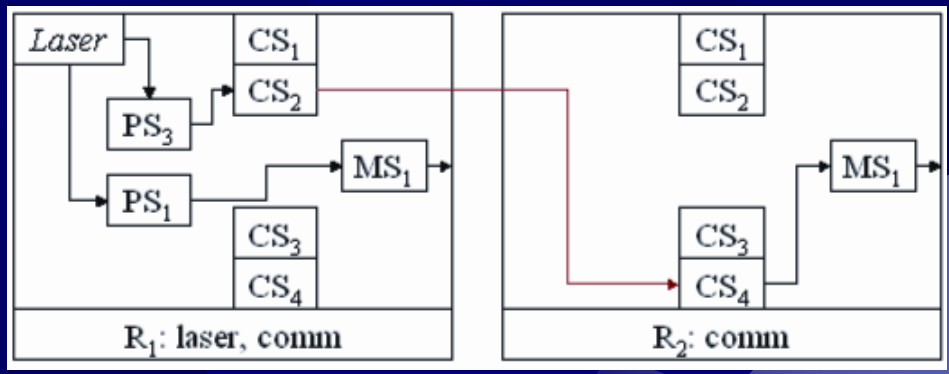


ASyMTRe: Our Reasoner for Generating Autonomous Inter-Connections

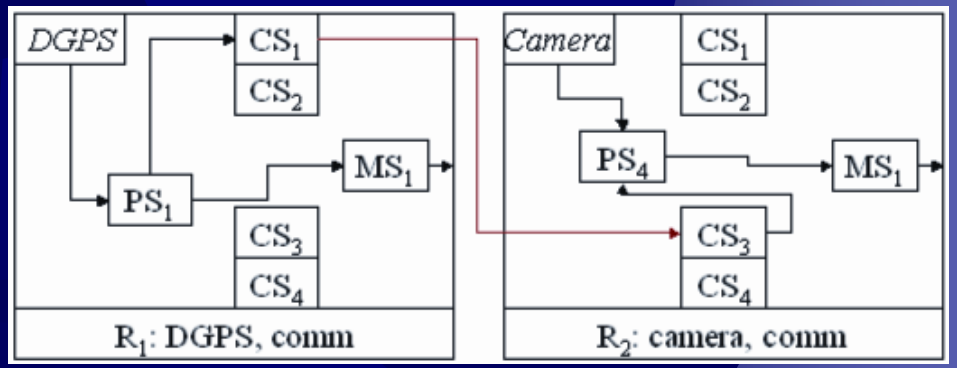
- Implemented in 2 versions:
  - Centralized
  - Distributed
- Team objective/goal stated in terms of motor schemas *across team* that should be instantiated.
- ASyMTRe designed to be online, real-time decision maker, enabling reconfigurations when needed

Example of Reconfiguring Interconnections of Schemas; Team Objective: Go to goal

Case 1: R_1: laser localization; R_2: no environmental sensors
Solution: R_1 uses laser to calc. its own global position and to find relative position of R_2. R_1 communicates global position of R_2 to R_2. R_1 goes to goal on its own. R_2 goes to goal based on assistance from R_1.

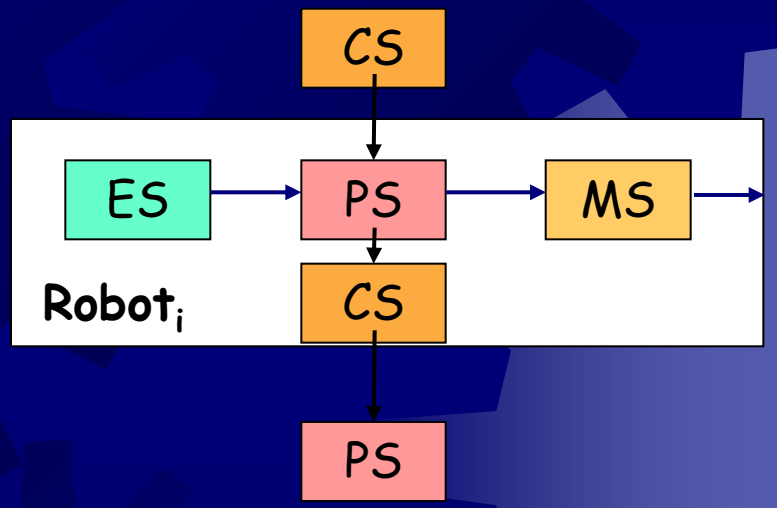


Case 2: R_1 :DGPS; R2: camera
Solution: R_1 uses DGPS to localize, communicates its own position to R_2; R_2 uses camera to determine its location relative to R_1, then to calc. its own global position. R_1 goes to goal on its own. R_2 goes to goal based on assistance from R_1, plus its own relative positioning calc.



Formal Problem Definition

- Set of *n* robots, $R = \{R_1, R_2, \ldots, R_n\}$
- Class of Information, $F = \{F_1, F_2, \ldots\}$
- Environmental Sensors, $ES = \{ES_1, ES_2, \ldots\}$
  - Input: physical sensor signal
  - Output: $O^{ESi} \in F$
- Perceptual Schemas, $PS = \{PS_1, PS_2, \ldots\}$
  - Input: $I_k^{PSi} \in F$
  - Output: $O^{PSi} \in F$
- Communication Schemas, $CS = \{CS_1, CS_2, \ldots\}$
  - Input: $I_k^{CSi} \in F$
  - Output: $O^{CSi} \in F$
- Motor Schemas, $MS = \{MS_1, MS_2, \ldots\}$
  - Input: $I_k^{MSi} \in F$
  - Output: $O^{MSi} \in F$



Formal Problem Definition (con't.)

- **Connection Rules:** $\forall k.\exists i.\text{Connect}(O^{S_i}, I_k^{S_j}) \Leftrightarrow O^{S_i} = I_k^{S_j}$
  - "For all inputs of S_j (i.e., some type of schema), there exists some S_i whose output is connected to one of the needed inputs of S_j."

- **Utility:**
  - *Sensori-Computational System*, SCS = {SCS_1, SCS_2, …} – a sensor plus its computational unit
  - *SCS Success Probabilities*, P = {P_1, P_2, …}, where P_k = *probability*(SCS_k)
  - *Sensing Costs*, C = {C_1, C_2, …}, where C_k = *cost*(ES_k)
  - Weight, *w*, balances combination of sensing costs and probability of success
  - Fitness of solution on robot R_j = $\mu(j) = \Sigma_i \left(w \cdot \frac{1}{C_i} + (1-w) \cdot P_i \right)$

Formal Problem Definition (con't.)

- **Task**, T = {MS_a, MS_b, …}
  - The set of motor schemas across the team that should be instantiated.

- **Objective**: Find solution, (*R, T, U)* satisfying following constraints:

$\forall_{MS_i \in T} \forall_{j,k} \exists \text{Connect}(O^{S_j}, I_k^{MS_i})$. The inputs of MS_i are satisfied.

$\forall_{S_j} \forall_{m,k} \exists \text{Connect}(O^{S_m}, I_k^{S_j})$. The inputs of S_j are satisfied.

$\forall_{S_q} \forall_{p,k} \exists \text{Connect}(O^{S_p}, I_k^{S_q})$. The inputs of S_q are satisfied.

$\forall_{S_p} \forall_{n,k} \exists \text{Connect}(O^{ES_n}, I_k^{Sp})$. The inputs of S_p are satisfied by some ES or $ES's$.

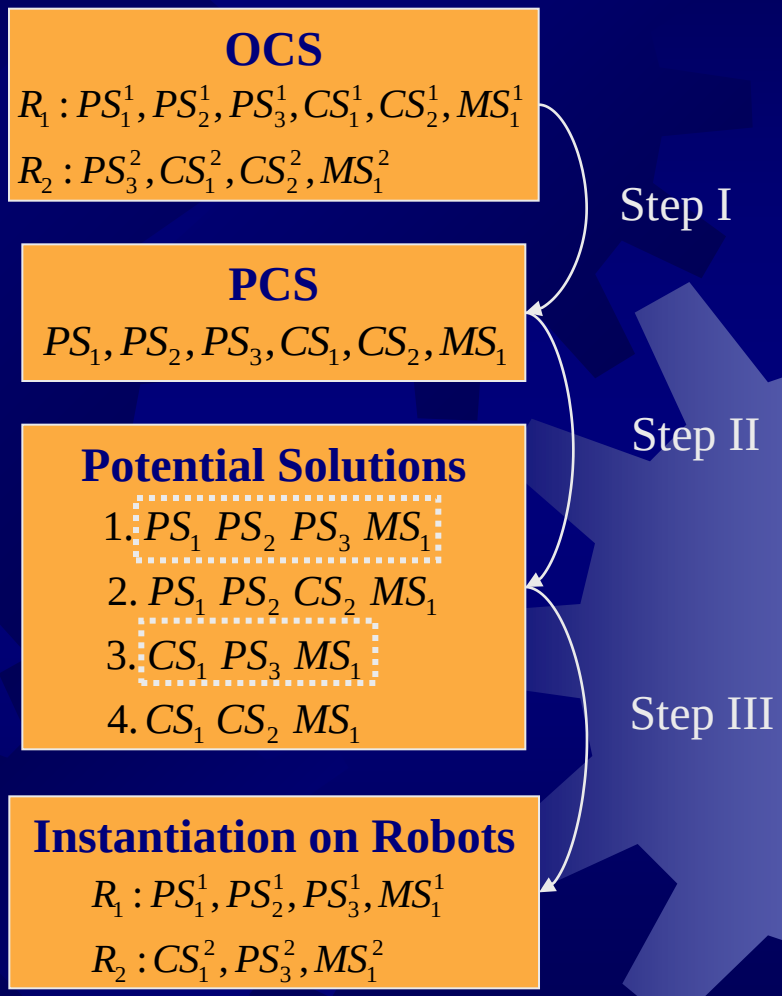
where $S_j, S_m, S_q \in PS \bigcup CS$, $S_p \in PS$, $ES_n \in ES$,

the utility $\sum_j \mu(j)$ for every R_j is maximized,

and the number of subteams is maximized.

How to Search for Solution?

- **Step I:** Reduce Original Configuration Space (OCS) to the Potential Configuration Space (PCS)
  - Approach: Include only one entry in PCS for each equivalence class of schema
- **Step II:** Find potential solutions in PCS
- **Step III:** Instantiate solution on specific robots



Speed Search Process: PCS + Heuristics

- **Challenge:** Globally optimal solution is NP-hard problem.
  - Searching Original Configuration Space (OCS) is $O(2^n)$.
- **Our approach** to speeding search for solution:
  - Reduce Search Space (OCS) via Potential Configuration Space (PCS)
  - Define greedy search heuristics that work well in practice, based on ordering robots according to:
    - Increasing sensing capabilities,
    - Increasing numbers of robots that can be assisted.

L. E. Parker and F. Tang, "Building Multi-Robot Coalitions through Automated Task Solution Synthesis", *Proceedings of the IEEE*, special issue on Multi-Robot Systems, vol. 94, no. 7, 2006.

Example of Reconfiguring Interconnections of Schemas

Team Objective: Go to goal

- *Case 1: Fully Capable Robots*
  Robot 1 -> Laser Scanner + Map
  Robot 2 -> Laser Scanner + Map

- *Case 2: Communicate Own Current Position*
  Robot 1 (Helper) -> Laser Scanner + Map
  Robot 2 (Needy) -> Camera

- *Case 3: Communicate Other's Current Position*
  Robot 1 (Helper) -> Laser Scanner + Map and Camera
  Robot 2 (Needy) -> nil

L. E. Parker, Chandra, and Tang, "Enabling Autonomous Sensor-Sharing for Tightly-Coupled Cooperative Tasks", *3rd NRL International Workshop on Multi-Robot Systems*, March 2005.

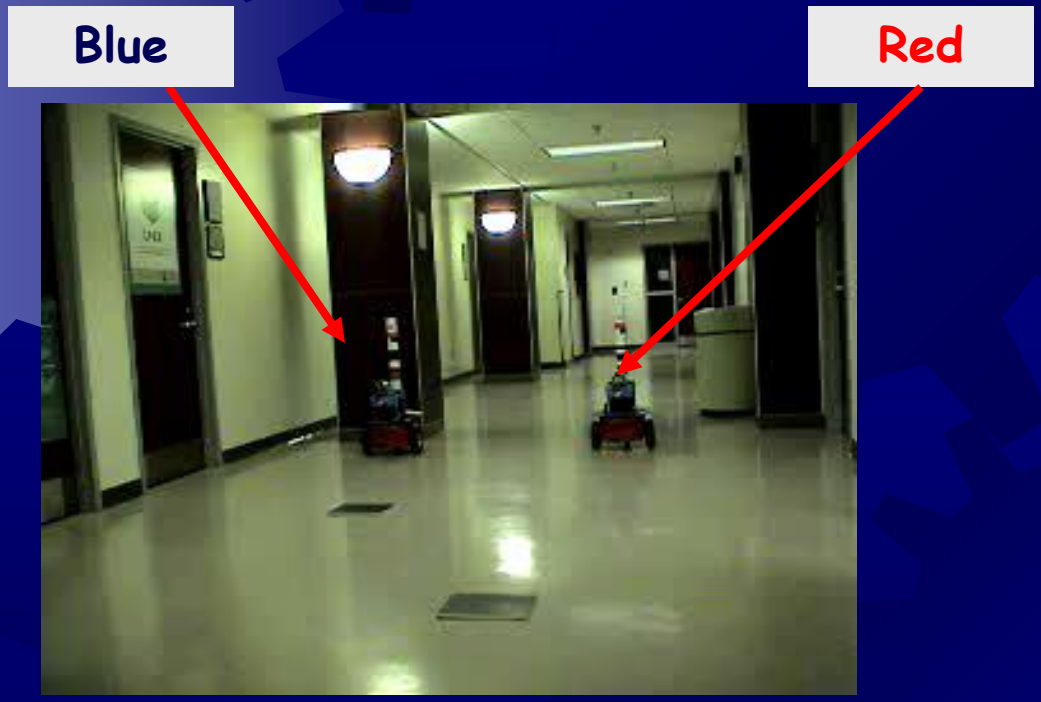
Chandra, "Software Reconfigurability for Heterogeneous Robot Cooperation", UTK M.S. thesis, Spring 2004.

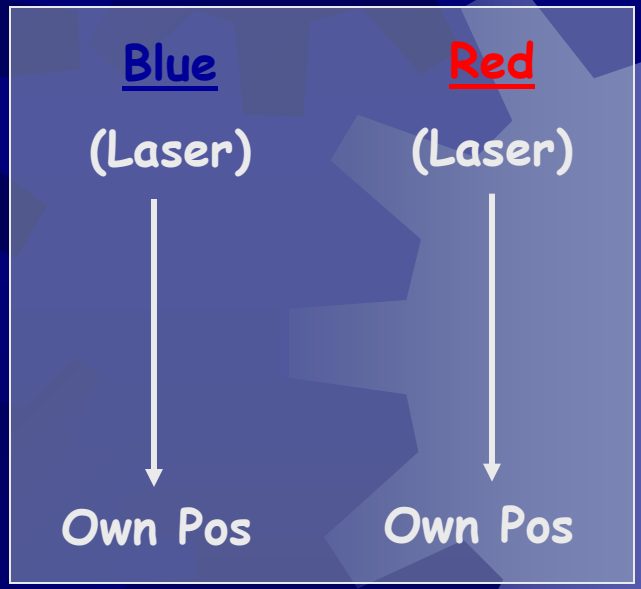
Case 1: Fully Capable Robots

Fully Capable Robots:

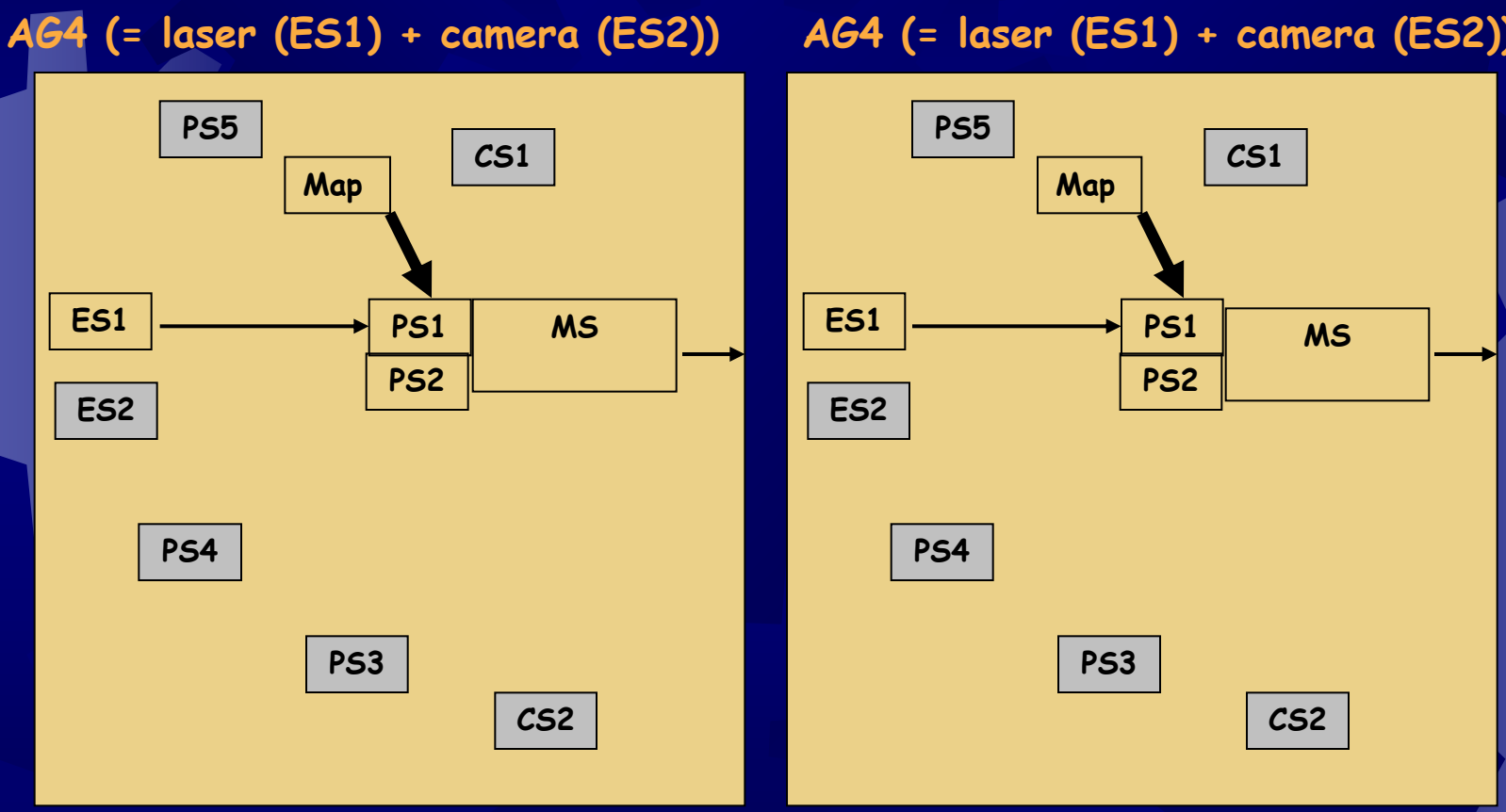
Robot 1 -> Laser Scanner + Map

Robot 2 -> Laser Scanner + Map





ASyMTRe-Derived Schema Configurations for Case 1

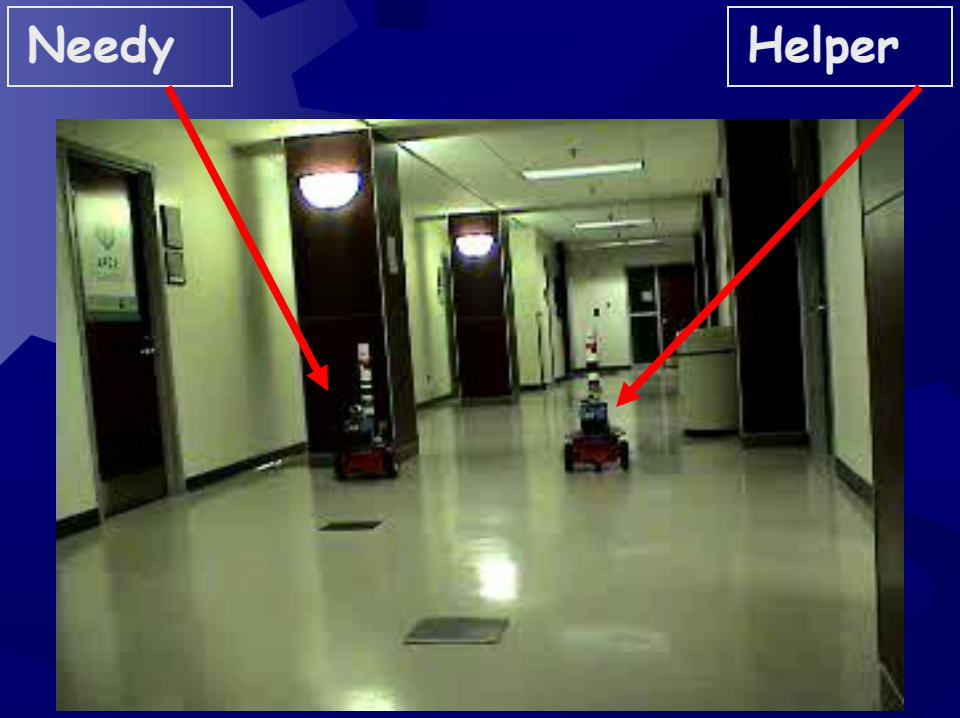


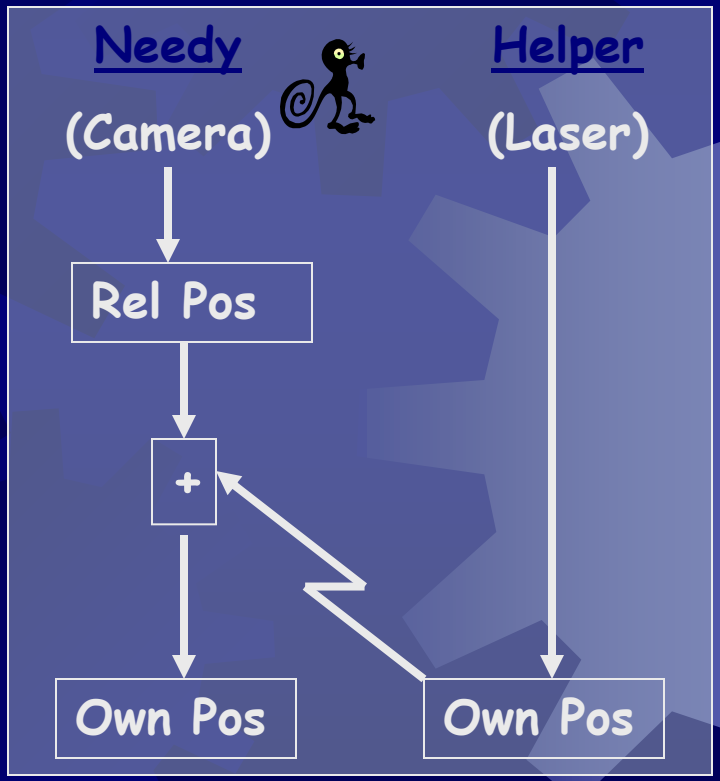
Case 2 – Helper robot communicates own global position

Communicate Own Current Global Position

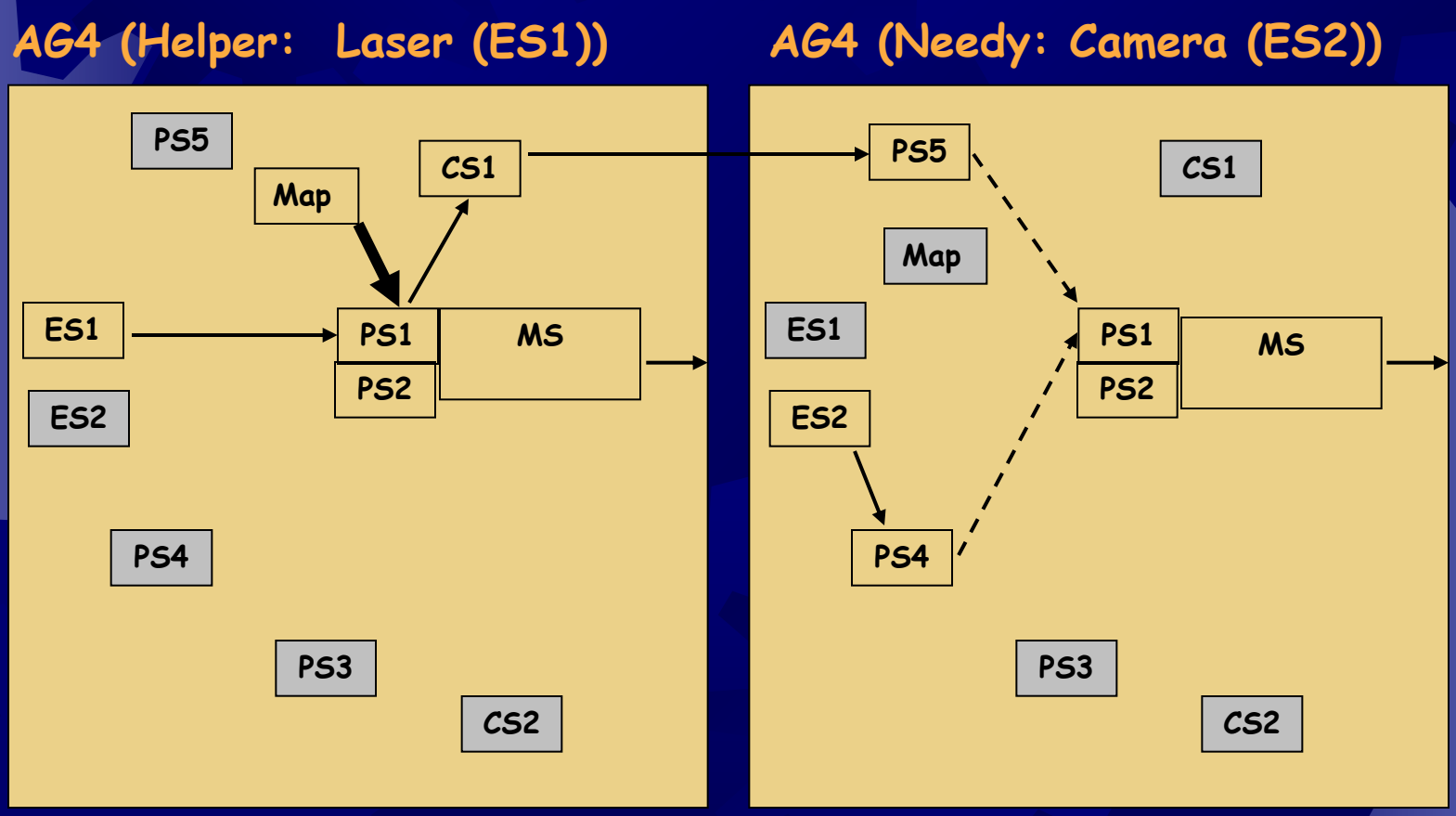
Robot 1 (Helper) -> Laser Scanner + Map

Robot 2 (Needy) -> Camera





ASyMTRe-Derived Schema Configurations for Case 2

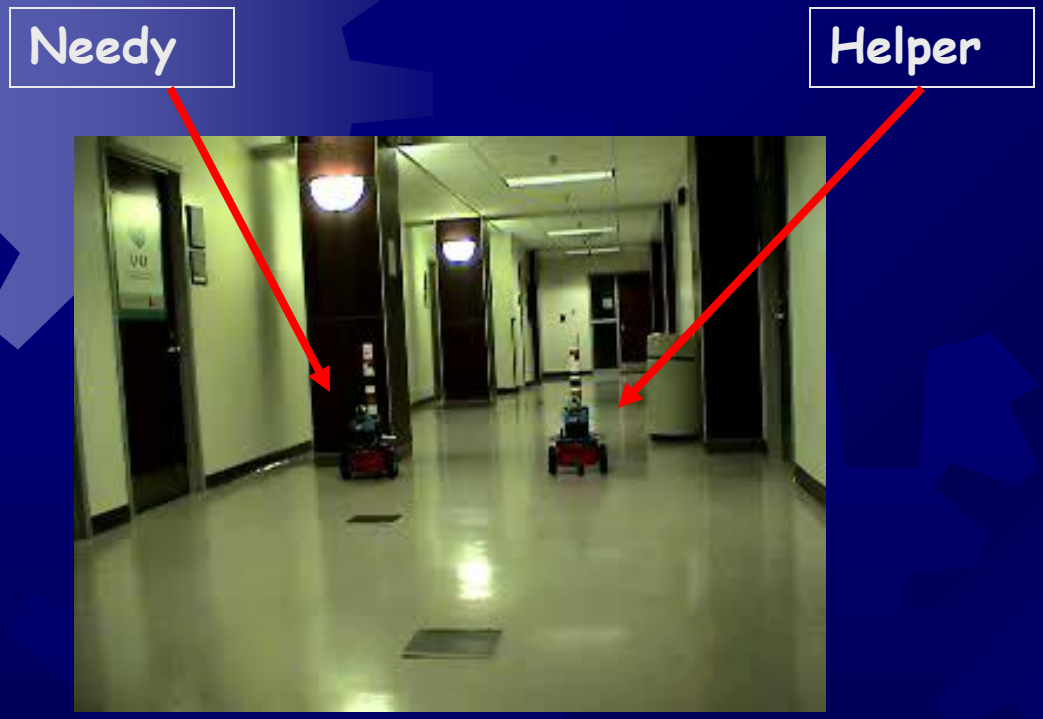


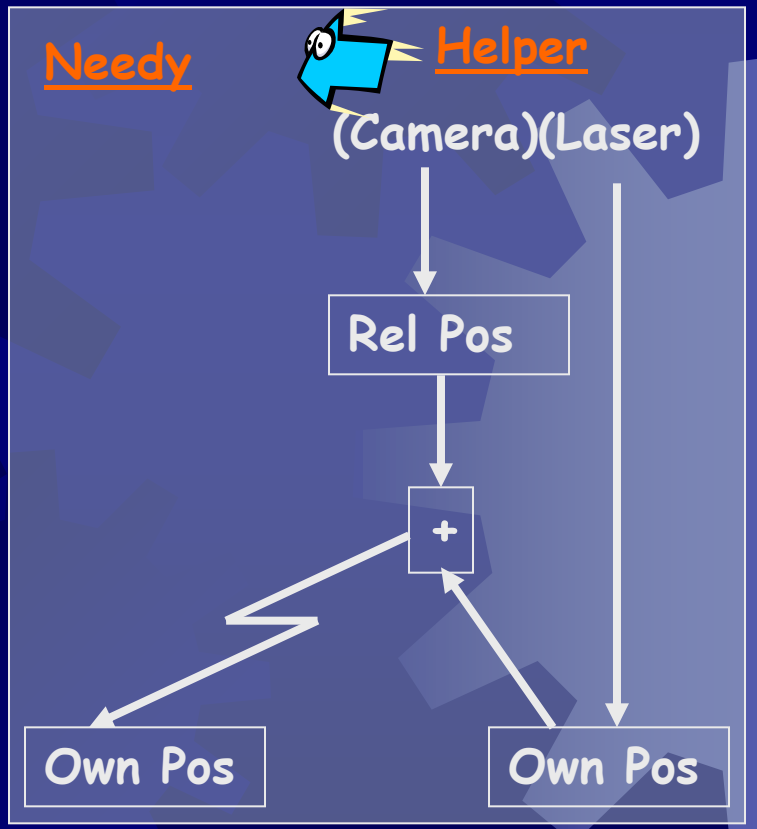
Case 3 – Helper Robot Communicates other Robot's Global Position

Communicate Other's Current Global Position

Robot 1 (Helper) -> Laser Scanner + Map and Camera

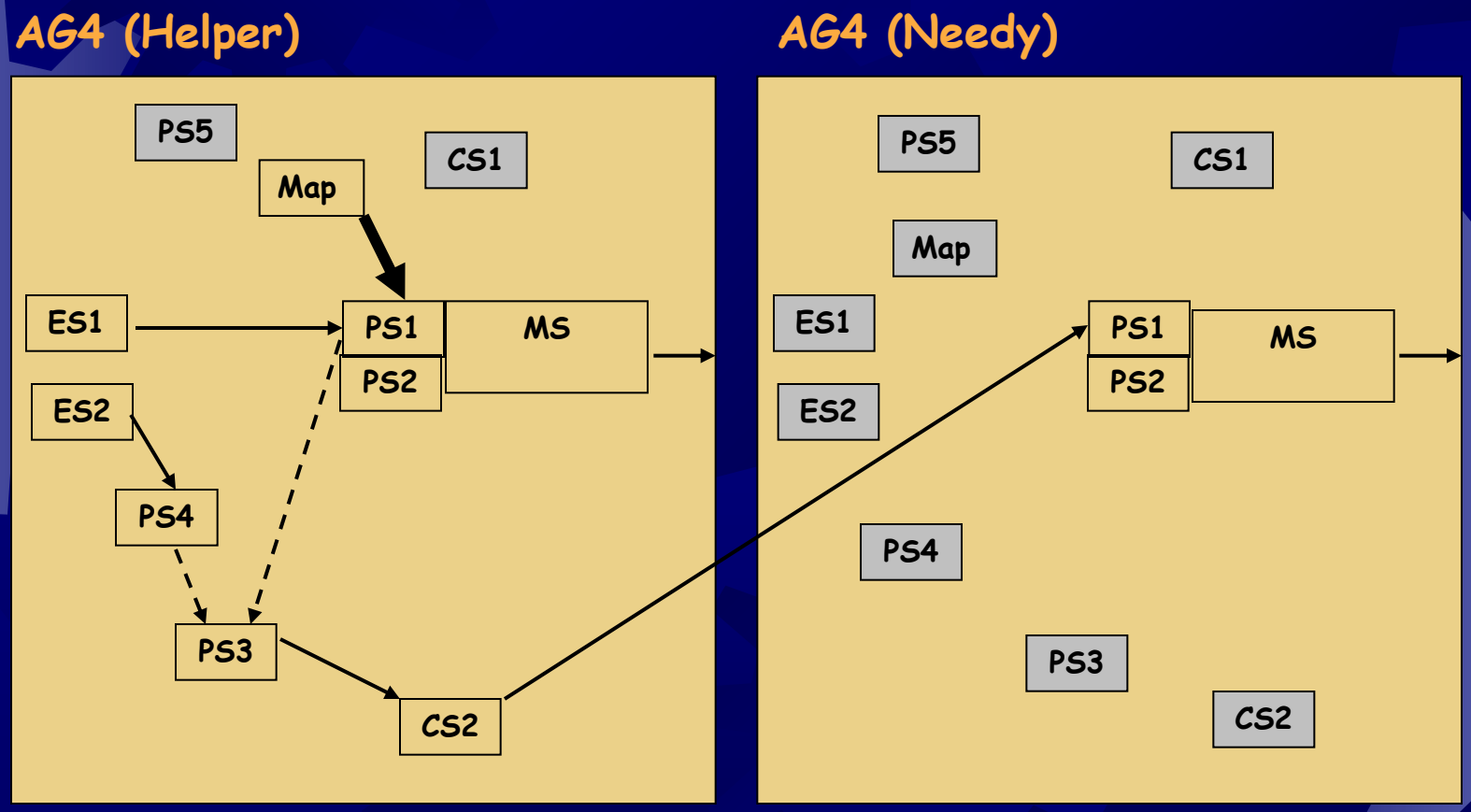
Robot 2 (Needy) -> nil





ASyMTRe

ASyMTRe-Derived Schema Configurations for Case 3



Movies of ASyMTRe

Ongoing and Future Work with ASyMTRe

- Prove formal properties of system
- Extend approach:
  - Additional sensor modalities (including communications as sensing)
  - More precise sensor models
  - Sensor fusion
  - More explicit representation of goals
  - Incorporate motion prediction for constrainer
  - More formal manner of handling distributed information sharing for indirect constraint satisfaction
  - Handling clique constraints, rather than just pair-wise constraints
- Address broader variety of applications:
  - Cooperative box pushing task (with obstacles)
  - Cooperative tasks with manipulation
  - Cooperative formations
- Address implementation efficiencies
- Extend ASyMTRe to incorporate these concepts

For more information:

Zhang and Parker, "A general information quality approach for satisfying sensor constraints in multi-robot tasks", ICRA 2010.

Outline

- Definition of Task Allocation
- History of Task Allocation
- Taxonomy of Task Allocation
- Example Approaches
  - ALLIANCE
  - MURDOCH
  - TraderBots
  - ASyMTRe
- Comparisons of Alternative Approaches
- Summary/Conclusions

Comparisons of Approaches

[Gerkey and Mataric, *Int'l. J. of Robotics Research*, 2004]

(m robots, n tasks)

Name	Computation per Iteration	Communication per Iteration	Solution Quality
ALLIANCE (Parker, '98)	O(mn)	O(m)	2-competitive or better
BLE (Werger and Mataric ,'01)	O(mn)	O(mn)	2-competitive
M+ (Botelho and Alami, '99)	O(mn)	O(mn)	2-competitive
MURDOCH (Gerkey and Mataric, '02)	O(1) / bidder O(n) / auctioneer	O(n)	3-competitive
TraderBots (Dias and Stentz, '01)	O(1) / bidder O(n) / auctioneer	O(n)	3-competitive or better
Dyn. Role Assgmnt. (Chaimowicz, Campos, and Kumar, '02)	O(1) / bidder O(n) / auctioneer	O(n)	3-competitive or better

Definition of α-competitive (for minimization problem): An algorithm is α-competitive if, for any input, it finds a solution whose total utility is never greater than α times the optimal utility.

Summary of Task Allocation

- Many approaches
- Each approach is relevant to some subset of task allocation taxonomy
  - SR-ST-IA and SR-ST-TA are most common
- General problem is NP-Hard
  - → Develop approximate, distributed approaches that work well in practice
- Formal comparisons of task allocation have been made