“There are two ways of constructing a software design. One way is to make it so simple that there are obviously no deficiencies. And the other way is to make it so complicated that there are no obvious deficiencies.”

-- C.A.R. Hoare
Announcements

• Final Project Proposals:
  – Everyone should have received individual email feedback Monday evening

• Final Project:
  – NEW DUE DATE: Thursday, November 26 (one extra class period)

• Assignment #5:
  – Questions?
  – NEW DUE DATE: Tuesday, November 12 (one extra class period)

• Next few weeks: we’ll study Chapter 10 first, then Chapter 9
Objectives

• Understand techniques for evaluating robot control code:
  – Theoretical / Analytical
  – Empirical

• Understand case studies in terms of their evaluation techniques
Techniques for Evaluating Robot Control Code

• **Theoretical / Analytical:**
  – Proofs of correctness

• **Empirical:**
  – Multiple experimental runs
  – Experiments to eliminate effect of specific starting conditions, environments, parameter settings, etc.
  – Average experiments over many trials
  – Maintain data on averages and standard deviations
  – Number of experiments needed: enough to ensure statistical significance
Case Study:
Theoretical / Analytical Proof of Robot Control Correctness

• Case Study: ALLIANCE architecture for multi-robot control (by Parker)

• ALLIANCE: Software control architecture for distributed multi-robot action selection that results in high degree of fault tolerance in cooperative control

• Developed: Theoretical Proof of Mission Completion using ALLIANCE

ALLIANCE: An Architecture for Heterogeneous Cooperative Control

- **Layer 0**: Inter-Robot Communication
- **Layer 1**: Motivational Behavior
- **Layer 2**: Behavior Set 0, 1, 2

Cross-inhibition connections between Motivational Behaviors.
ALLIANCE Formal Model

\[ R = \{ r_1, r_2, \ldots, r_n \} \]
\[ T = \{ \text{task}_1, \text{task}_2, \ldots, \text{task}_m \} \]
\[ A_i = \{ a_{i1}, a_{i2}, \ldots \} \]
\[ H: A_i \rightarrow T, \quad H = \{ h_1(a_{1k}), h_2(a_{2k}), \ldots, h_n(a_{nk}) \} \]
\[ \theta \]

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

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

\[ \text{activity\_suppression}_{ij}(t) = \begin{cases} 0 & \text{If } a_{ik} \text{ is active, } k \neq j, \text{ on robot } r_i \text{ at time } t \\ 1 & \text{Otherwise} \end{cases} \]
\[ \phi_{ij}(i, t) = \text{Time during which } r_i \text{ is willing to allow } r_k' \text{'s communication to affect motivation of } a_{ij}. \]

\[ \text{impatience}_{ij}(t) = \begin{cases} 
\min_k (\delta_{\text{slow}}_{ij}(k, t)), & \text{if } (\text{comm}_\text{received}(i, k, j, t - \tau_i, t) = 1) \text{ and } \\
& (\text{comm}_\text{received}(i, k, j, 0, t - \phi_{ij}(k, t)) = 0) \\
\delta_{\text{fast}}_{ij}(t), & \text{otherwise} 
\end{cases} \]

\[ \text{impatience}_\text{reset}_{ij}(t) = \begin{cases} 
0, & \text{if } \exists k.((\text{comm}_\text{received}(i, k, j, t - \delta t, t) = 1) \text{ and } \\
& (\text{comm}_\text{received}(i, k, j, 0, t - \delta t) = 0), \\
1, & \text{otherwise} 
\end{cases} \]

\[ \text{acquiescence}_{ij}(t) = \begin{cases} 
0, & \text{if } ((a_{ij}(t) \text{ active more than } \varphi_{ij}(t)) \\
& \text{and } (\exists x.\text{comm}_\text{received}(i, x, j, t - \tau_i, t) = 1)) \\
1, & \text{otherwise} 
\end{cases} \]
Motivation to activate a given behavior set:

\[ m_{ij}(0) = 0 \]

\[ m_{ij}(t) = [m_{ij}(t - 1) + \text{impatience}_{ij}(t)] \times \text{sensory\_feedback}_{ij}(t) \times \text{activity\_suppression}_{ij}(t) \times \text{impatience\_reset}_{ij}(t) \times \text{acquiescence}_{ij}(t) \]

Whenever \( m_{ij}(t) > \theta \), \( a_{ij} \) is activated, and robot has selected an action.
Proving Guarantee of Mission Completion in ALLIANCE

• Key issue: Does ALLIANCE allow robot team to complete its mission, even in presence of robot difficulties and failure?

• We’ll show: With certain restrictions on parameter settings, ALLIANCE is guaranteed to allow team to complete mission for broad range of applications.

• Definitions:

\[ R = \{r_1, r_2, \ldots, r_n\} \]  \hspace{2cm} n \text{ robots}

\[ T = \{\text{task}_1, \text{task}_2, \ldots, \text{task}_m\} \]  \hspace{2cm} m \text{ independent subtasks}

\[ A_i = \{a_{i1}, a_{i2}, \ldots\} \]  \hspace{2cm} \text{Behavior sets of robot } r_i

\[ H: A_i \rightarrow T, \quad H = \{h_1(a_{1k}), h_2(a_{2k}), \ldots, h_n(a_{nk})\} \]  \hspace{2cm} \text{Task in } T \text{ that } r_i \text{ is working on when } a_{ik} \text{ is active}

\[ \theta \]  \hspace{2cm} \text{Threshold of activation}

Goal-relevant capabilities of robot \( r_i \):

\[ GRC_i = \{a_{ij} \mid h_i(a_{ij}) \in T\} \]
• Definitions (con’t.):
  – *Task coverage*:
    \[
    task\_coverage(task_k) = \sum_{i=1}^{n} \sum_{j} \begin{cases} 
      1, & \text{if } h_i(a_{ij}) = task_k \\
      0, & \text{otherwise}
    \end{cases}
    \]

  – *Condition 1: Sufficient task coverage*:
    \[
    \forall (task_k \in T). (task\_coverage(task_k)) \geq 1
    \]

  – *Limitedly-reliable* robot: robot whose probability of success < 1

  – *Active* robot team:
    \[
    R \text{ such that } \forall (r_i \in R). \forall (a_{ij} \in GRC_i). \forall (r_k \in R). \forall t. \\
    [(\delta_{\_slow_{ij}}(k,t) > 0) \land (\delta_{\_fast_{ij}}(t) > 0) \land (\theta \text{ is finite})]
    \]
More Proof Definitions (con’t.)

– **Condition 2: Progress when Working:**
  
  - Let $z =$ finite amount of work remaining to complete $task_w$.
  - Then, when robot $r_i$ activates behavior set corresponding to $task_w$, either:
    1) $r_i$ remains active for a sufficient, finite length of time $\varepsilon$ s.t. $z$ is reduced by a finite amount which is at least some constant $\delta > 0$;
    2) or, $r_i$ experiences a failure w.r.t. $task_w$
  - Additionally, if $z$ ever increases, the increase is due to an influence external to the robot team.
• Theorem:
  – Let $R$ be a *limitedly-reliable, active* robot team, and let $M$ be the mission to be solved by $R$, such that Conditions 1 and 2 hold (i.e., sufficient task coverage and “progress when working”). Then,
  – Part I: Either:
    1) ALLIANCE enables $R$ to accomplish $M$
    2) or, a robot failure occurs.
  – Part II: Further, if robot $r_f$ fails, then the only tasks of $M$ that are not completed are some subset of:
    a) set of tasks only $r_f$ was designed to accomplished, unioned with:
    b) set of tasks dependent upon the capabilities of only $r_f$
Proof of Mission Completion Guarantee

Proof:

Step 1: Show that motivational behavior calculation guarantees each robot will eventually activate behavior set corresponding to an incomplete task:

– At time $t$, robot $r_i$’s motivation $m_{ij}(t)$ to perform behavior set $a_{ij}$ either:
  1) Goes to 0
    4 cases:
    » If sensory feedback indicates behavior set no longer applicable;
    » If another behavior set becomes active;
    » If some other robot has taken over task $h_i(a_{ij})$;
    » If robot has acquiesced its task.
  2) Changes from $m_{ij}(t-1)$ by the amount $impatience_{ij}(t)$
    Since $r_i$ is active, then
    Idle, yet active robot has strictly increasing motivation to perform incomplete task $\implies$ eventually threshold $\theta$ is reached.
    \[
    impatience_{ij}(t) \geq \min_k(\delta_{\_slow_{ij}}(k,t)) > 0
    \]
Proof of Mission Completion Guarantee (con’t.)

Step 2: Show that either ALLIANCE succeeds or a robot fails:
   – Assume no robot fails. Then after $r_i$ has performed task $w$ for $t > \varepsilon$;
      5 cases:
         » $r_j$ takes over task $w$, leading $r_i$ to acquiesce;
         » $r_i$ gives up on itself and acquiesces $w$;
         » $r_j$ takes over task $w$, but $r_i$ does not acquiesce;
         » $r_i$ continues $w$; and
         » $r_i$ completes $w$.

   – Since Condition 2 holds, 1st four cases reduce amount of work left to complete $w$ by at least a positive, constant amount $\delta$. Finite work $\implies$ finite time.

   – In the fifth case, since $w$ completed, robots go on to other tasks.

   – Thus, either some set of robots eventually attempts task enough times so that it becomes complete; or all robots designed to accomplish task have failed.
Step 3: Show that incomplete tasks are dependent upon a failed robot’s capabilities:

- Let $F$ be set of robots that fail during mission, and $A_F$ be union of:
  - Tasks that only robots in $F$ were designed to accomplish
  - Tasks dependent upon a task that only robots in $F$ were designed to accomplish

- For task $w \notin A_F$:
  - Since Condition 1 holds and robot team active, must be some $r_i$ that can successfully accomplish $w$;
  - Since Condition 2 holds, task will eventually be completed in finite time.

- For $w$ not completed, can show by way of contradiction that $w$ must be in $A_F$.

- Thus, tasks not dependent upon capabilities of failed robot are successfully completed.
Another Case Study for Analytical Evaluation: 
L-ALLIANCE: Towards Efficiency in Robot Action Selection

- While ALLIANCE can guarantee mission completion, no guarantees (to this point) on efficiency of task execution.

- Need: **L-ALLIANCE (Learning ALLIANCE)** => Automatic parameter update procedure to ensure efficiency of action selection, enabling:
  - Robots best suited to particular tasks to select to accomplish those tasks
  - Robots to dynamically adapt to changes in environment, robot capabilities, or team composition
  - Minimal need for significant pre-mission tuning of parameters by human operators
  - Easy use of custom-designed teams
  - Robots to improve their performance over time when working with “familiar” robot team members
How do we accomplish this? First: Recognize that optimal action selection is compute-intensive

• Definition of Heterogeneous Robot Action Selection Problem (HRASP):

  – Given $R$, $T$, $A$, and $H$,
  – Find the set of actions $U_i$ for all $r_i$, for a given performance metric $q(a_{ik})$ such that:

    $\forall (r_i \in R). U_i \subseteq A_i$, and
    $\forall (task_j \in T). \exists i. \exists k. ((task_j = h_i(a_{ik})) \land (a_{ik} \in U_i))$, and
    $\max_i (\sum_{a_{ik} \in U_i} q(a_{ik}))$ is minimized.
Proof that HRASP is NP-Hard

• Proof:
  – By restriction to NP-complete problem PARTITION [Garey and Johnson, 1979]:
  – PARTITION problem:
    Given finite set $W$ and a "size" $s(w) \in \mathbb{Z}^+$ for each $w \in W$,
    determine whether there is a subset $W' \subseteq W$ such that:
    $$\sum_{w \in W'} s(w) = \sum_{w \in W-W'} s(w).$$
    – Then we have restriction of HRASP to PARTITION by allowing only instances of HRASP
    where:
    - $n = 2$ (i.e., $R = (r_1, r_2)$)
    - $A_1 = A_2 = W$
    - $T = \bigcup_{r_i \in R, a_{ij} \in A_i} (h_i(a_{ij}))$
    - $\forall (r_i \in R). \forall (a_{ij} \in A_i). (h_i(a_{ij}) = task_j)$
    - $\forall (a_{ij} \in A_i). (q(a_{1j}) = q(a_{2j}) = s(w_j)), \text{ for } w_j \in W.$
  – Then, since PARTITION is a special case of HRASP, HRASP must be NP-hard.
Key Point of Theoretical / Analytical Evaluation

• The ALLIANCE and L-ALLIANCE proofs are examples of theoretical / analytical techniques for validating robot control code.

• However, in general, it is difficult to generate such proofs.

• Many assumptions must be made; some assumptions may not be true in robot’s physical world.

• Therefore, it’s more common to validate robot control code empirically (i.e., experimentally)
Empirical Evaluation and Validation of Robot Control Code

• **Main point:** since it is difficult to sufficiently model robot’s environment and interactions, it’s better to validate robot control code empirically

• However, in empirical evaluations, must ensure that performance evaluation is not skewed through limited test case selection

• **Must generalize over:**
  – Range of parameter settings
  – Range of robot environments (or, at least across a certain class)
  – (Perhaps) numbers of robots
Typical Experimental Evaluation Approach

• Define **important variables/parameters** of the system
• Determine a **range of values** for variables/parameters to be tested
• Define **metrics** to be used for evaluating system
• Determine **characteristics of environment** which algorithm is designed to handle
• Generate a **variety of environments** within this class to test
  – Best if environments are randomly selected
  – For example, generate obstacle size and location based upon a particular distribution
• Determine how many experimental runs are needed to generate **statistical significance**
• Determine a **baseline performance** to which the new algorithm will be compared
• Run multiple experiments, varying starting position of robot randomly
• Collect data over multiple experimental runs, and **average data**, also maintaining **standard deviation**
## Example: Metrics Used in ALLIANCE Implementations

<table>
<thead>
<tr>
<th>Application domain</th>
<th># Robots</th>
<th>Metric description</th>
<th>Metric definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. “Mock” hazardous waste cleanup</td>
<td>2-5 (P)</td>
<td>a. Time of task completion</td>
<td>$t_{max}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. Total energy used</td>
<td>$\sum_{i=1}^{m} \sum_{t=1}^{t_{max}} e_i(t)$, where $e_i(t)$ is energy used by robot $i$ through time $t$ (m robots)</td>
</tr>
<tr>
<td>2. Box pushing</td>
<td>1-2 (P)</td>
<td>Perpendicular dist. pushed per unit time</td>
<td>$d_{\perp}(t)/t$, where $d_{\perp}(t)$ is $\perp$ distance moved through time $t$</td>
</tr>
<tr>
<td>3. Janitorial service</td>
<td>3-5 (S)</td>
<td>a. Time of task completion</td>
<td>$t_{max}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. Total energy used</td>
<td>$\sum_{i=1}^{m} \sum_{t=1}^{t_{max}} e_i(t)$, where $e_i(t)$ is energy used by robot $i$ through time $t$ (m robots)</td>
</tr>
<tr>
<td>4. Bounding overwatch</td>
<td>4-20 (S)</td>
<td>Distance moved per unit time</td>
<td>$d(t)/t$, where $d(t)$ is distance moved through time $t$</td>
</tr>
<tr>
<td>5. Formation-keeping</td>
<td>4 (P &amp; S)</td>
<td>Cumulative formation error</td>
<td>$\sum_{t=0}^{t_{max}} \sum_{i=leader} d_i(t)$, where $d_i$ = distance robot $i$ is misaligned at $t$</td>
</tr>
<tr>
<td>6. Simple multi-robot manipulation</td>
<td>2-4 (P)</td>
<td>Number of objects moved per unit time</td>
<td>$j(t)/t$, where $j(t)$ is number of objects at goal at time $t$</td>
</tr>
<tr>
<td>7. Cooperative tracking</td>
<td>2-4 (P)</td>
<td>Avg. number of targets observed (collectively)</td>
<td>$A = \sum_{t=1}^{t_{max}} \sum_{j=1}^{n} g[H(t),j]$, where $H(t) = [b_{ij}(t)]<em>{m \times n}$, (m robots, n targets) \ $b</em>{ij}(t) = 1 \iff$ robot $i$ observing target $j$ at $t$, \ $g(H(t),j) = \begin{cases} 1 \text{ if exists } i \text{ s.t. } b_{ij}(t) = 1 \ 0 \text{ otherwise} \end{cases}$</td>
</tr>
<tr>
<td></td>
<td>2-20 (S)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8. Multi-vehicle production dozing</td>
<td>2-4 (S)</td>
<td>Quantity of earth moved per unit time</td>
<td>$q(t)/t$, where $q(t)$ is quantity of earth moved through $t$</td>
</tr>
</tbody>
</table>
Example Case Studies of Empirical Evaluation

• CMOMMT: Cooperative Target Tracking

• L-ALLIANCE for autonomous parameter update in multi-robot action selection
Case Study: CMOMMT

- **CMOMMT**: Cooperative Multi-robot Observation of Multiple Moving Targets
- **Definition**:

  **Given**: $S$: 2-D bounded, enclosed spatial region  
  $V$: team of $m$ robot vehicles, $v_i$, $i = 1, 2, \ldots, m$, with 360° FOV sensors  
  $O(t)$: set of $n$ targets, $o_j(t), j = 1, 2, \ldots, n$, such that target $o_j(t)$ is in $S$ at $t$

  Define $m \times n$ matrix $B(t)$:  
  $$B(t) = [b_{ij}(t)]_{mxn} \text{ such that } b_{ij}(t) = \begin{cases} 1 & \text{if robot } v_i \text{ is observing target } o_j(t) \text{ in } S \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

  **Goal**:
  
  Maximize:  
  $$A = \sum_{t=1}^{T} \sum_{j=1}^{n} \frac{g(B(t),j)}{T}$$

  where $g(B(t),j) = \begin{cases} 1 & \text{if there exists an } i \text{ such that } b_{ij}(t) = 1 \\ 0 & \text{otherwise} \end{cases}$
Hand-Generated Solution to CMOMMT Completed

**Overall Approach:**  **Weighted Local Force Vectors:**

- Each team member makes own action selection
- Summation of weighted local force vectors:
  - Attractive forces for targets, repulsive for robots
  - Weighted by locations of nearby robots
- At each time step, robot moves in direction of summed vector
Example of Weighted Force Vectors

- Communication range of $r_i$
- Predictive tracking range of $r_i$
- Sensing range of $r_i$

resultant direction of movement
Series of Experiments for Evaluating Hand-Generated Solution

- Number of robots, $m$ 1-10
- Number of targets, $n$ 1-20
- Target behavior: random vs. evasive
- Radius of robot sensing range, $r$ 2600
- Radius of arena ($S$), $R$ 1,000 - 50,000
- Parameter settings:
  - $do_1 = 400$
  - $do_2 = 800$
  - $do_3 = 2600$
  - *predictive tracking range* = 3000
  - $dr_1 = 1250$
  - $dr_2 = 2000$
- Measured $A$ every $\Delta t = 2$ seconds; 2 minute runs
- 250 runs per instantiation
- Compared runs by normalizing $A$: $A/n$
Evaluating Hand-Generated Solution to CMOMMT

Results studied in over 1,000,000 simulation test runs

Over 800 experimental runs on physical robots (either with or without obstacles)
Qualitative Comparison: 3 Robots, 3 Targets; Random/linear target movements
Qualitative Comparison: 5 Robots, 10 Targets; Random/linear target movements
Qualitative Comparison: 3 Robots, 3 Targets; Evasive target movements
Qualitative Comparison: 5 Robots, 10 Targets; Evasive target movements
Results for Random/Linear Target Movements

- $n/m = 1/5$
- $n/m = 1/2$
- $n/m = 1$
- $n/m = 4$
- $n/m = 10$
Results for Evasive Target Movements
A-CMOMMT Outperforms Local for $n/m > 0.5$

Random/linear target movements

Evasive target movements

<table>
<thead>
<tr>
<th>$n/m$</th>
<th>Random/linear</th>
<th>Evasive</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>-8%</td>
<td>-11%</td>
</tr>
<tr>
<td>0.5</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>1</td>
<td>14%</td>
<td>7%</td>
</tr>
<tr>
<td>4</td>
<td>27%</td>
<td>20%</td>
</tr>
<tr>
<td>10</td>
<td>31%</td>
<td>25%</td>
</tr>
</tbody>
</table>
Statistical Significance in CMOMMT

• Use Student’s t distribution, comparing policies 2 at a time for all 6 possible pairings
• In these computations, use null hypothesis:
  – $H_0$: $\mu_1 = \mu_2$, and there is essentially no different between the two policies.

• Under hypothesis $H_0$:

$$T = \frac{\bar{X}_1 - \bar{X}_2}{\frac{1}{\sigma} \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

where:

$$\sigma = \sqrt{\frac{n_1 S_1^2 + n_2 S_2^2}{n_1 + n_2 - 2}}$$
• Then, on the basis of a two-tailed test at a 0.01 level of significance, we would reject $H_0$ if $T$ were outside the range $-t_{.995}$ to $t_{.995}$.

• For $n_1 + n_2 - 2 = 250 + 250 - 2 = 498$ degrees of freedom, $-t_{.995}$ to $t_{.995}$ is in the range $-2.58$ to $2.58$.

• For the data shown in the previous figures, we can reject $H_0$ at a 0.01 level of significance for all pairing of policies.

• Therefore, we can conclude that the variation in performance of policies shown in the earlier diagrams is statistically significant.
Evaluation of Physical Experiments with/without obstacles

Each data point is average of 10 runs of 10 minutes each
Key Points of CMOMMT Case Study

• Experiments varied all parameter settings over “interesting” range

• Baseline approaches used as point of comparison

• Experimental settings (e.g., starting robot and target positions) varied randomly

• Number of experimental runs should be sufficient to ensure statistical significance

• Data to be evaluated is averaged over multiple runs

• Data is plotted with standard deviations over multiple runs shown

• When adding obstacles, positioning of obstacles generated automatically according to uniform distribution
Another Case Study of Empirical Evaluation: L-ALLIANCE: Towards Efficiency in Robot Action Selection

• While ALLIANCE can guarantee mission completion, no guarantees (to this point) on efficiency of task execution.

• Need: L-ALLIANCE (*Learning ALLIANCE*) ==> Automatic parameter update procedure to ensure efficiency of action selection, enabling:
  – Robots best suited to particular tasks to select to accomplish those tasks
  – Robots to dynamically adapt to changes in environment, robot capabilities, or team composition
  – Minimal need for significant pre-mission tuning of parameters by human operators
  – Easy use of custom-designed teams
  – Robots to improve their performance over time when working with “familiar” robot team members
L-ALLIANCE Parameter Update Strategy

- Robots monitor performance of teammates performing certain tasks, based upon pre-defined quality metric (*time*, in our case)

- Two phases:
  - Phase I: Active learning phase
  - Phase II: Adaptive learning phase

- Heuristic parameter update mechanism derived from analysis of statistical data obtained through empirical simulation studies


L-ALLIANCE: Automatic Parameter Updates via Performance Monitoring

Robots monitor quality metric for other robots performing same corresponding task
How are quality measures used to update parameters?

• Primary ALLIANCE parameters affecting action selection:

\[ \phi_{ij}(i,t) = \text{Time during which } r_i \text{ is willing to allow } r_k \text{'s communication to affect motivation of } a_{ij}. \]

\[ \delta_{\text{slow}}_{ij}(k,t) = \text{Rate of impatience of } r_i \text{ concerning behavior set } a_{ij} \text{ when } r_k \text{ performing task } h_i(a_{ij}). \]

\[ \delta_{\text{fast}}_{ij}(t) = \text{Rate of impatience of } r_i \text{ concerning behavior set } a_{ij} \text{ when no other robot performing task } h_i(a_{ij}). \]

• Statistical data collected and analyzed based upon simulations varying update strategies:
  – Impatience and acquiescence rates as a function of:
    • Robot’s own quality (here, \textit{time})
    • Quality (time) of another team member
    • Quality (time) of best team member
  – Task orderings based upon:
    • Greedy choice of:
      – Longest task remaining
      – Shortest expected execution time of remaining tasks
    • Random task selection
What Experiments Were Run to Empirically Evaluate Approaches?

• Number of robots, n: 2-20
• Number of tasks, m: 1-40
• Task_coverage: 1-10
• Degree of heterogeneity: 0-3200 percent
• Progress-When-Working (PWW): true or false

• 200 randomly generated test runs per each 5-tuple (scenario) of:
  (n, m, task_coverage, heterogeneity, PWW)
• Average over the 200 test runs was considered to be characteristic performance of that scenario
• Parameter update strategy based upon the strategy that works the best in most scenarios
Resulting L-ALLIANCE Action Selection Methodology

Conceptually, robot $r_i$ repeats the following until sensory feedback indicates no more tasks are remaining:

- **Divide the remaining tasks into two categories:**
  1) Tasks meeting the intersection of the following conditions:
     - Tasks $r_i$ expects to be able to perform better than any other team member
     - Tasks that no other robot is currently performing
  2) All other tasks $r_i$ can perform

- **Select tasks from the first category according to longest task first approach, unless no more tasks remain in the first category.**
  - If no more tasks remain in first category, select tasks from second category according to the shortest task first approach.
Key Points of L-ALLIANCE Case Study

• Parameter update strategy developed based upon outcome of experimental analysis

• Experiments varied all parameter settings over “interesting” range

• Experimental settings (e.g., robot capabilities) varied randomly

• Data to be evaluated is averaged over multiple runs
Remember: Key Points of Empirical Robot Control Evaluation

• Define important variables/parameters of the system
• Determine a range of values for variables/parameters to be tested
• Define metrics to be used for evaluating system
• Determine characteristics of environment which algorithm is designed to handle
• Generate a variety of environments within this class to test
  – Best if environments are randomly selected
  – For example, generate obstacle size and location based upon a particular distribution
• Determine how many experimental runs are needed to generate statistical significance
• Determine a baseline performance to which the new algorithm will be compared
• Run multiple experiments, varying starting position of robot randomly
• Collect data over multiple experimental runs, and average data, also maintaining standard deviation
Summary of Robot Control Code Evaluation

- Two primary approaches to evaluating control code:
  - Theoretical / analytical
  - Empirical
- Theoretical / analytical based upon proofs of correctness or fundamental limitations of approaches
- However, theoretical / analytical techniques can be difficult to develop due to need for unrealistic assumptions, difficulty in modeling, etc.
- Empirical approaches more common in robot control evaluation
- Must ensure that empirical data is relevant through:
  - Multiple runs varying parameters of interest
  - Randomly generating starting conditions
  - Statistically significant data
Preview of Next Class

• Begin study of Navigation, Localization, Path Planning, and Mapping:
  – Robot questions:
    • Where am I?
    • Where am I going?
    • What’s the best way to get there?
    • Where have I been?

• Remember: we’ll study Chapter 10 first, then Chapter 9