Dynamic Combinatorial Control under Uncertainty

Adversarial and Stochastic Elements in Autonomous Systems Workshop

David Castañón, Boston University

Sponsored by AFOSR

Center for Information and Systems Engineering

Motivating Problems

- **Mission Optimization for teams of unmanned air vehicles…**
  - Determining tasks to perform, by which vehicles, in which manner
  - Assignment, routing, scheduling…discrete decisions
  - Combinatorial growth of states, actions with number of tasks
- **in uncertain environments…**
  - Inaccurate models
  - Imperfect information
  - Uncertain knowledge of adversary activities
- **with teams of distributed agents**
  - Limited communications
  - Distributed information

Uncertain Elements

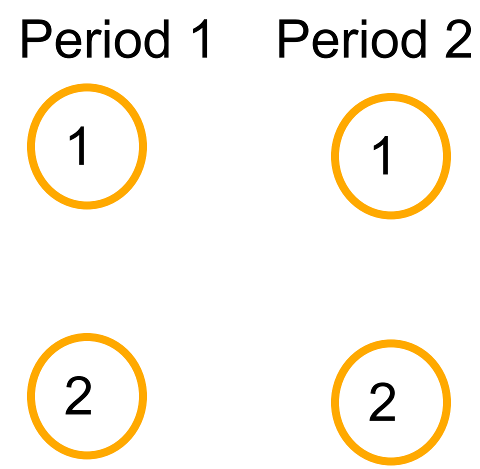
- **Unknown objectives**
  - Future tasks, constraints, resources…
- **Unknown environments**
  - Inaccurate information on adversarial resources and capabilities
  - Uncertain evolution in response to actions
  - Uncertain evolution of information
- **Multiple agents**
  - Potentially limited knowledge of team activities
  - Limited knowledge of adversary objectives and activities

Control Approaches

- **Heuristics**
  - Index-based scheduling, greedy assignment, others …
  - Adaptive indexing, easy to compute in real time
- **Open-loop plans with dynamic replanning**
  - Discrete optimization problems (assignment, scheduling, …)
  - Adapts through replanning
  - Harder computation in real time
- **Closed-loop plans**
  - Dynamic modeling of information, uncertainty
  - Hard to compute off-line, store for on-line (dynamic programming)
- **Real-time closed-loop planning**
  - Simulation-based learning (e.g. neuro-dynamic programming, Q-learning) == hard to generalize
  - Future value real-time approximations (rollout, bounds, etc) – generalizes but hard to compute

A Simple Replanning Example

- **Two tasks, two periods**
  - Can attempt one task per period
  - Attempts may fail independently
  - Prob. Success is period-dependent



- **Task 1: value 8; Task 2: value 4**
  - Ps = 0.75 period 1, 0.8 period 2
- **Objective: max expected value accomplished**
- **Open-loop: attempt 2, then 1 → 9.4 expected value**
  - Fails to account for value of new information
- **Feedback strategy: attempt 1, observe success, then either attempt 1 again or 2 → 9.85 expected value**

Generalization: Dynamic Assignment

- Motivation: Dynamic search, unreliable resource allocation, …
- N tasks, two periods
- M resource types

M_j : Number of resources of type j

p_{ij} : Probability that single resource of type j successfully completes task i

x_{ij} : Number of resources of type j assigned to task i

R_j : Cost of using resource of type j

- Independence of success outcomes

Two Stage Single Resource Type

- **Define a task completion state after each stage**

$$\omega_i(k) \in \{0,1\} \text{ denotes the completion state of task } i \text{ after stage } k$$
$$\overline{\omega}(k) = \{\omega_1(k),\ldots,\omega_N(k)\} \text{ is the overall task completion state after } k$$

  - Task completion state observed after each stage

- **Decisions are now feedback policies**

$$x_i(k,\overline{\omega}(k-1)) = \text{resources assigned to task } i \text{ in stage } k$$
$$\overline{x}(k,\overline{\omega}(k-1)) = \text{ vector of resource allocations at stage } k$$

- **Task completion state dynamics: Controlled Markov chain**
  - Independence of completion event outcomes decouples dynamics

$$P(\omega_i(k) = 1 \mid \omega_i(k-1) = 1, x_i(k,\overline{\omega}(k-1)) = n) = (1 - p_i(k))^n$$

Objectives and Constraints

- **Objective: minimize expected uncompleted task value plus expected resource use costs**

$$\min_{\{\bar{x}(1),\bar{x}(2,\bar{\omega}(1))\}} E\left\{\sum_{i=1}^{N} V_i I\{\omega_i(2)=1\} + R_1\left[x_i(1) + x_i(2,\bar{\omega}(1))\right]\right\}$$

- **Constraints**

$$\sum_{i=1}^{N} x_i(1) + x_i(2,\bar{\omega}(1)) \le M_1 \quad \text{for all outcomes } \bar{\omega}(1)$$

$$x_i(1), x_i(2,\bar{\omega}(1)) \in \{0, 1, \ldots, M_1\}$$

- **Dynamic programming possible, but large number of states**

Approximate Dynamic Programming

- **Relax constraints to expand admissible strategies**
  - Generates lower bound to optimal value function
  - New constraint on average number of resources

$$\sum_{\{\overline{\omega}(1)\}} P(\overline{\omega}(1) \mid \overline{x}(1)) \left[\sum_{i=1}^{N} x_i(1) + x_i(2,\overline{\omega}(1)) \right] \leq M_1$$

$$x_i(1), x_i(2,\overline{\omega}(1)) \in \{0, 1, ..., M_1\}$$

- **Relaxes exponential number of constraints to a single constraint**
  - Simple result: All feasible strategies in original problem are feasible in current problem

Characterization of Optimal Strategies

- **Important concept: Mixed local strategies**
  - Local strategies: feedback strategies such that the actions on a given task depend only on the state of that task

$$x_i(2,\overline{\omega}(1)) \equiv x_i(2,\omega_i(1))$$

  - Mixed strategy: random combination of pure strategies
    - Mixed strategies may achieve better performance than pure strategies in relaxed problem
- **Theorem: In relaxed problem, for every pure strategy, there is a mixed local strategy which uses same resources and achieves same expected performance**
  - Proven by construction
  - Restricts search to local mixed strategies

Solution of Relaxed Problem

- Can solve independent subproblems parameterized by expected resource use
- Primal dual stochastic optimization algorithm

$$\sum_{i=1}^{N} \min_{x_i(1), x_i(2,\omega_i(1))} F_i(x_i(1), x_i(2,\omega_i(1))) + \lambda T_i(x_i(1), x_i(2,\omega_i(1)))$$

- Theorem: Optimal solution of relaxed problem with single resource type can be obtained in complexity $O((M_1+N)\log(N))$
- Scales to large numbers of objects
- Generalizations to multiple resource types, more complex problems

Castañón-Wohletz, TAC '09 (to appear)

Control Approach

- **Solution of relaxed problem not guaranteed to be feasible over entire horizon**
  - Feasible for first stage…
  - Use exact solution of approximate model to generate first period resource assignments
- **Optimal strategies are mixed strategies**
  - Randomize selection
- **Control: implement parts of approximate strategy, observe outcomes, then replan subsequent allocations**
  - Receding horizon approach with two-stage horizon

Results

- **Larger experiments**
  - Only Greedy and MPC algorithms
  - Same value and probability ranges as before
  - 100 random problems per data point
  - performance: percent of task value completed by Greedy algorithm

Tasks	Resources	MPC Ave.	MPC Worst
16	12	99.8%	99.2%
16	16	99.8%	99.3%
16	20	99.9%	99.7%
20	12	99.8%	99.5%
20	16	99.8%	99.5%
20	20	99.9%	99.4%

Computation requirements on Pentium 1.4 GHz, Linux:
Greedy: 13 minutes for 20 tasks
MPC: 0.04 seconds for 1000 tasks

Extension: Discrete Sequential Search

- **Allow for parallel tasks, changing focus of attention**
  - Multiple agents can look at cells in parallel
  - Can leave cell without making decision and return to it ()
  - Agents may overlap on tasks, collaborate on collecting decision/information

- **Goals: Find and classify objects by collecting information over time**

  Leads to partially-observed assignment

Information State

- **Conditional probability that cell i contains object of given type j given measurements and actions up to but not including time t**
  - ¼$_i$(t) = p(x_i |y(0), a(0), …, y(t-1), a(t-1))

- **Result: Under simple conditional independence assumptions, a sufficient statistic is Π (t) = {π_1(t), …, π_N(t)},**

  → Joint conditional probability is product of marginals

- **Information Dynamics (discrete event system): Bayes' Rule**
  - Act locally on cells: only measured sites change information state
  - Similar structure to multi-armed bandit problem

Result: Lower Bound POMDP

- **Minimize** $$J = \sum_{i=1}^{N} E\{\min_{v_i} c(x_i(T), v_i(T))\}$$

- **Subject to constraints**

$$\sum_{\tau=0}^{T} \sum_{i=1}^{N} \sum_{m=1}^{M} E\{R_{ikm} u_{ikm}(\tau)\} \leq C_k$$

$$\sum_{m=1}^{M} u_{ikm}(\tau) \leq 1$$

$$\pi_i^s(\tau+1) = \frac{\pi_i^s(\tau) \prod_m P(y_{ikm} | x_i = s, u_{ikm}(\tau))}{\sum_{s'} \pi_i^{s'}(\tau) \prod_m P(y_{ikm} | x_i = s', u_{ikm}(\tau))}$$

$$u_{ikm}(\tau) : [\pi_1(\tau) \ldots \pi_N(\tau)] \rightarrow \{0, 1, \ldots, M\}$$

Weak Duality

- **Use Lagrange multipliers to incorporate relaxed resource constraints into objective: Lagrangian, for λ >= 0:**

$$J(\lambda,\gamma) = E_\gamma\{\sum_{i=1}^{N}[c(v_i,x_i)+\sum_k \lambda_k \sum_{\tau=0}^{T-1}\sum_{m=1}^{M} R_{ikm}u_{ikm}(\tau))\}-\sum_k \lambda_k C_k$$

- **Lower bounds given by weak duality**

$$\min_\gamma J(\lambda,\gamma) \le \max_{\lambda\ge 0}\min_\gamma J(\lambda,\gamma) \le \min_\gamma J(\gamma)$$

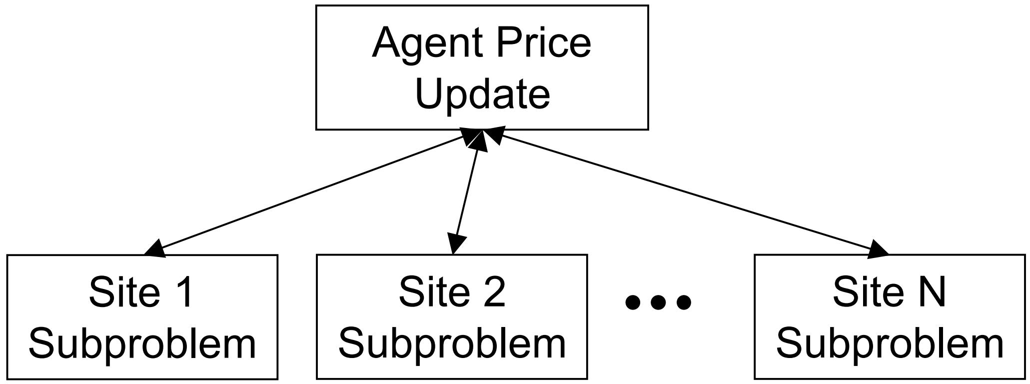
- **Lagrangian problem is almost separable over objects**
  - Coupled only by feedback strategies!

Enabling Result

- **Under mild independence assumptions, optimal solution of relaxed problem can be obtained using local adaptive strategies**
  - Adapt strategies for each location based only on information collected for that location
  - For every global adaptive strategy, there is an equivalent random local strategy that achieves the same performance

- **Leads to scalable mission control algorithms**
  - Solved by optimizing Lagrangian dual in hierarchical fashion

Hierarchical Pricing of Agent Time



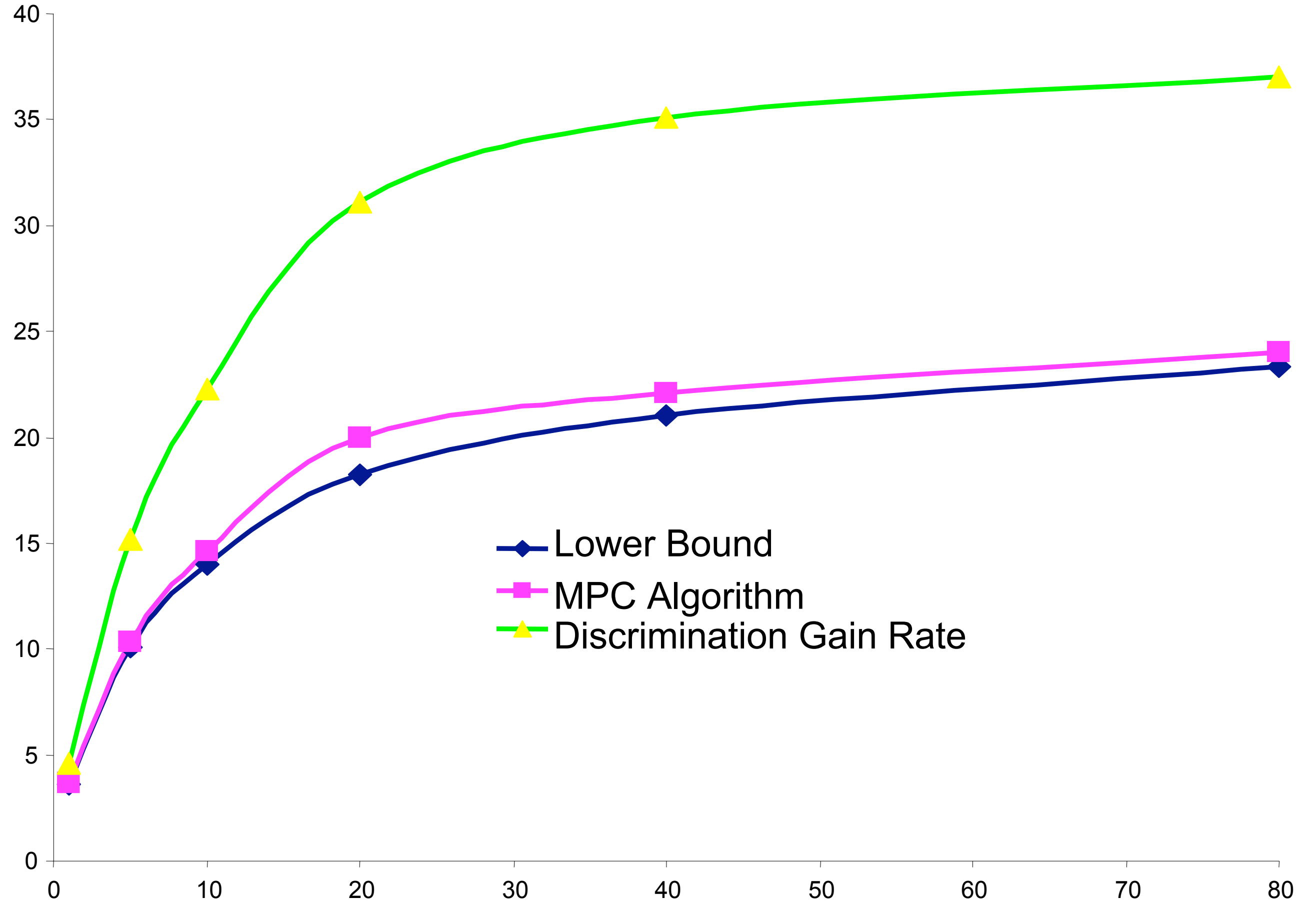
$$\min_p L(p,\lambda) = \sum_i \min_{p_i} p_i(\gamma_i)(J_i^{\gamma_i} - \sum_j \lambda_j R_{ij}^{\gamma_i}) + \sum_j C_j \lambda_j$$

Note: minimum is achieved in pure strategies for each price vector λ

- **Agent prices: dual variables for consuming sensor time for different sensors**
  - Subproblems solved optimally using small POMDP single object algorithms
  - NS-dimensional POMDP reduced to N single object S-dimensional POMDPs + dual

Two Agents, each with one mode

- 250 seconds of observations per agent
- Loss of performance over optimal partitioning of time among modes



Conclusions

- **Discussed approaches for real-time computation of controls for stochastic dynamic assignment problems with combinatorial action and decision spaces**
  - Embedding into nearly separable problems
  - Averaging over constraints
  - Model predictive receding horizon implementations
- **Generalization to other classes of problems needed**
  - Routing and scheduling – control of motion as well as task
  - Collaborative non-independent performance