Topics in Reinforcement Learning:
Lessons from AlphaZero for
(Sub)Optimal Control and Discrete Optimization

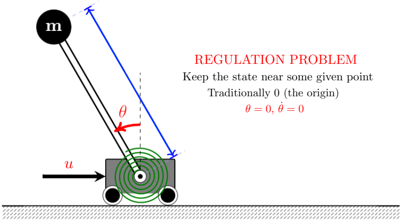
Arizona State University
Course CSE 691, Spring 2022

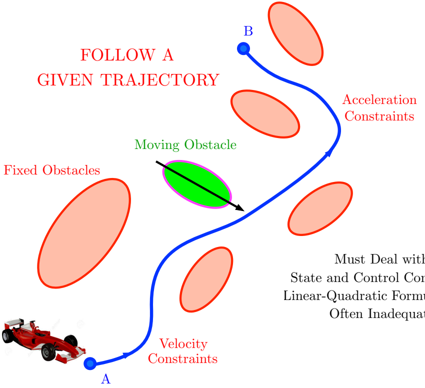
Links to Class Notes, Videolectures, and Slides at
http://web.mit.edu/dimitrib/www/RLbook.html

Dimitri P. Bertsekas
dbertsek@asu.edu

Lecture 6
Model Predictive Control, Multiagent Rollout

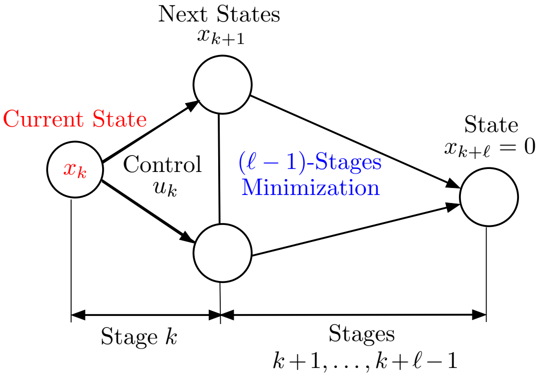
Outline





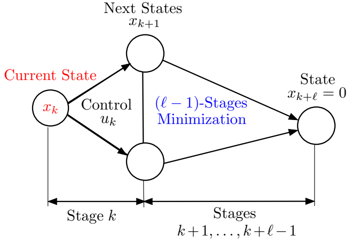
Must Deal with
State and Control Constraints
Linear-Quadratic Formulation is
Often Inadequate

The Original Form of MPC for Regulation to the Origin Problems



- System: $x_{k+1} = f(x_k, u_k)$; 0 is an absorbing (goal) state, $f(0, u) \equiv 0$.
- Cost per stage: $g(x_k, u_k) > 0$, except that 0 is cost-free, $g(0, u) \equiv 0$.
- Control constraints: $u_k \in U(x_k)$ for all k. Perfect state information.
- MPC: At x_k solve an ℓ-step lookahead version of the problem, requiring $x_{k+\ell} = 0$ (ℓ: fixed and sufficiently large to allow the transfer to 0).
- If $\{\tilde{u}_k, \dots, \tilde{u}_{k+\ell-1}\}$ is the control sequence so obtained, apply $\tilde{u}_k$, discard $\tilde{u}_{k+1}, \dots$

Relation to Rollout - Stability



- MPC is rollout w/ base heuristic the $(\ell-1)$-step min to 0 (and stay at 0).
- Let $H(x)$ denote the optimal cost of the $(\ell-1)$-step min, starting from x.
- This heuristic is sequentially improving (not sequentially consistent), i.e.,

$$\underbrace{\min_{u\in U(x)}\Big[g(x,u)+H\big(f(x,u)\big)\Big]}_{\substack{\text{opt cost from } x \text{ to 0 in } \ell \text{ steps} \\ \text{then stay at 0 for additional steps}}} \leq \underbrace{H(x)}_{\substack{\text{opt cost from } x \text{ to 0 in } (\ell-1) \text{ steps} \\ \text{then stay at 0 for additional steps}}}$$

  because (opt. cost to reach 0 in ℓ steps) $\leq$ (opt. cost to reach 0 in $\ell-1$ steps)
- Sequential improvement $\rightarrow$ "stability", i.e., that the MPC controller has a finite cost from every initial state x_0.
- Reason: By the cost improvement property, the cost of the MPC controller starting from x_0 is no greater than $H(x_0)<\infty$.

A Major Variant: MPC with Terminal Cost

- At state x_0, instead of requiring that $x_\ell = 0$, we solve

$$\min_{u_i,\, i=0,\ldots,\ell-1} \left[G(x_\ell) + \sum_{i=0}^{\ell-1} g(x_i, u_i) \right],$$

subject to $u_i \in U(x_i)$ and $x_{i+1} = f(x_i, u_i)$, where $G(x) > 0$ for $x \neq 0$, and $G(0) = 0$.

- This is ℓ-step lookahead minimization with terminal cost function G.
- Let us assume that $TG \leq G$, where T is the min-Bellman operator, i.e., for all x,

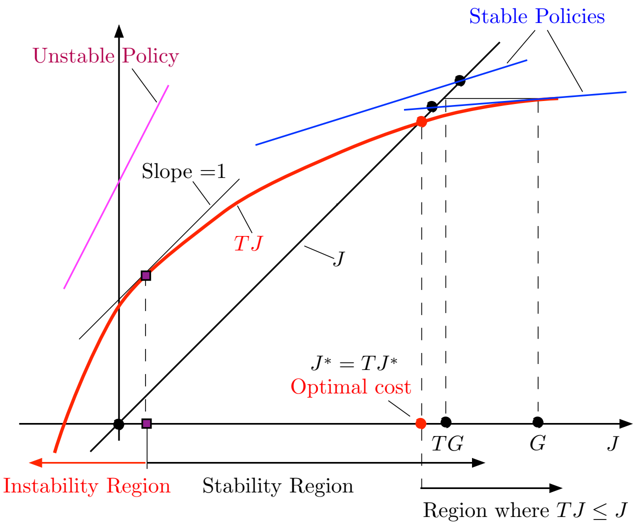
$$(TG)(x) = \min_{u \in U(x)} \big[g(x,u) + G\big(f(x,u)\big) \big] \leq G(x).$$

- We can show that this condition implies stability of the MPC controller. An analytical proof is possible (see the “Lessons ..." book, Section 3.2), but we give a graphical argument in this lecture.
- The argument is based on the concept of the region of stability: this is the set of all $\tilde{J}$ such that the policy $\tilde{\mu}$ obtained by one-step lookahead minimization,

$$T_{\tilde{\mu}} \tilde{J} = T \tilde{J},$$

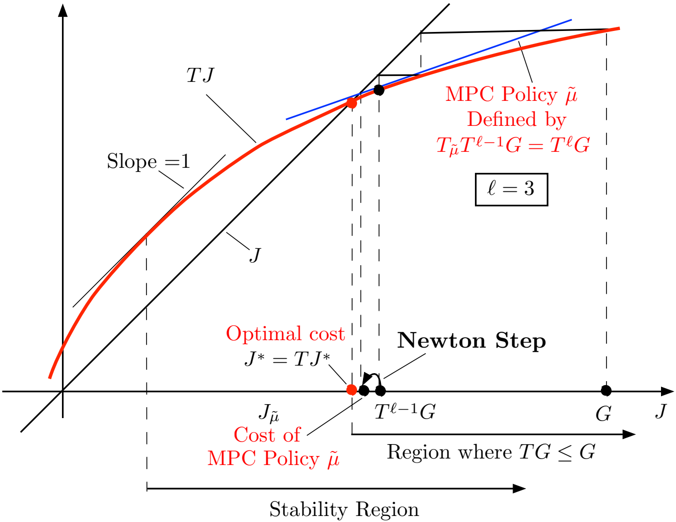
is stable.

Region of Stability - A Terminal Cost Function G Satisfying $TG \leq G$



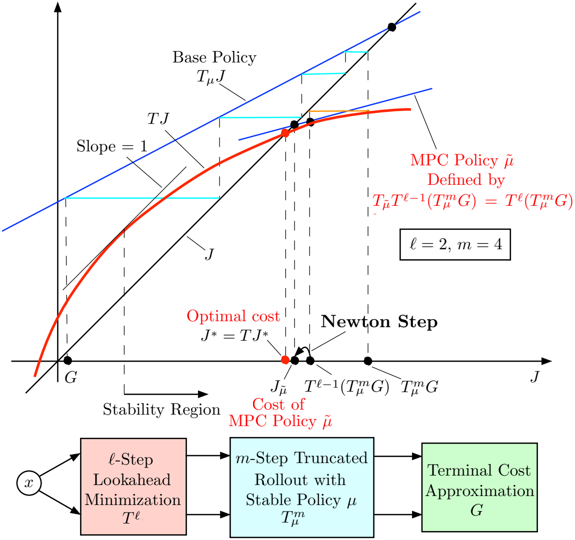
$TG \leq G$ implies that $T^{\ell}G$ lies within the region of stability for all ℓ

Abstract Visualization of MPC with ℓ-Step Lookahead Minimization and Terminal Cost Function G Satisfying $TG \leq G$



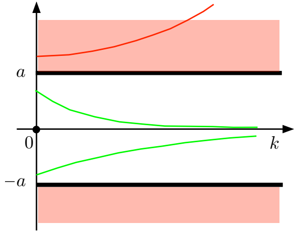
$TG \leq G$ implies that G lies in the region of stability, and so does $T^{\ell}G$ for any $\ell \geq 1$

MPC with ℓ-Step Lookahead Minimization, m-Step Truncated Rollout, and Terminal Cost Function G: The AlphaZero Architecture!



Larger values of m and ℓ help make the MPC policy stable

Other Variants of MPC



MPC with state/safety/tube constraints: $x_k \in X$ for all k

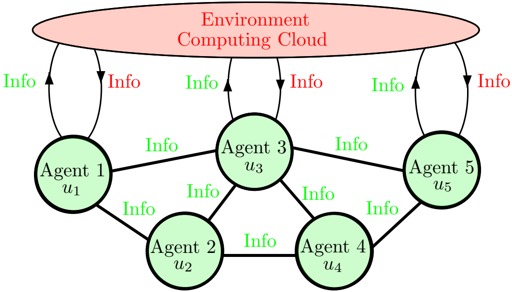
- Special difficulty: The tube constraint may be impossible to satisfy for some $x_0 \in X$
- Need to construct (off-line) an inner tube from within which the state constraints can be met
- Leads to the methods of reachability of target tubes (my 1971 PhD thesis, on-line)

Combinations with off-line training methods

Training of terminal cost function approximation, a base policy for truncated rollout, etc

MPC for stochastic problems: Must solve an ℓ-step stochastic DP problem on-line. Can be dealt with certainty equivalence, except for the first stage

Multiagent Problems - A Very Old (1960s) and Well-Researched Field

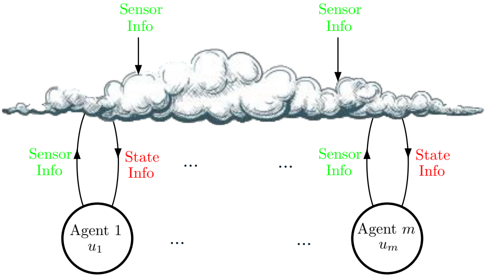


- Multiple agents collecting and sharing information selectively with each other and with an environment/computing cloud
- Agent i applies decision u_i sequentially in discrete time based on info received

The major mathematical distinction between structures

- The classical information pattern: Agents are fully cooperative, fully sharing and never forgetting information. Can be treated by Dynamic Programming (DP)
- The nonclassical information pattern: Agents are partially sharing information, and may be antagonistic. HARD because it cannot be treated by DP

Our Starting Point: A Classical Information Pattern ... but we will Generalize



The agents have exact state info, and choose their controls as functions of the state

Model: Stochastic DP (finite or infinite horizon) with state x and control u

- Decision/control has m components $u = (u_1, \ldots, u_m)$ corresponding to m "agents"
- "Agents" is just a metaphor - the important math structure is $u = (u_1, \ldots, u_m)$
- We apply approximate DP/rollout ideas, aiming at faster computation in order to:
  - Deal with the exponential size of the search/control space
  - Be able to compute the agent controls in parallel (in the process we will deal in part with nonclassical info pattern issues)

Multiagent Rollout/Policy Improvement When $u = (u_1, \ldots, u_m)$

To simplify notation, consider infinite horizon setting. The standard rollout operation is

$$\big(\tilde\mu_1(x), \ldots, \tilde\mu_m(x)\big) \in \arg \min_{(u_1,\ldots,u_m)} E_w\Big\{g(x, u_1, \ldots, u_m, w) + \alpha J_\mu\big(f(x, u_1, \ldots, u_m, w)\big)\Big\};$$

the search space is exponential in m (μ is the base policy, seq. consistency holds)

Multiagent rollout (a form of simplified rollout; implies cost improvement)

Perform a sequence of m successive minimizations, one-agent-at-a-time

$$\tilde\mu_1(x) \in \arg\min_{u_1} E_w\Big\{g(x, u_1, \mu_2(x), \ldots, \mu_m(x), w) + \alpha J_\mu\big(f(x, u_1, \mu_2(x), \ldots, \mu_m(x), w)\big)\Big\}$$

$$\tilde\mu_2(x) \in \arg\min_{u_2} E_w\Big\{g(x, \tilde\mu_1(x), u_2, \mu_3(x) \ldots, \mu_m(x), w) + \alpha J_\mu\big(f(x, \tilde\mu_1(x), u_2, \mu_3(x), \ldots, \mu_m(x), w)\big)$$

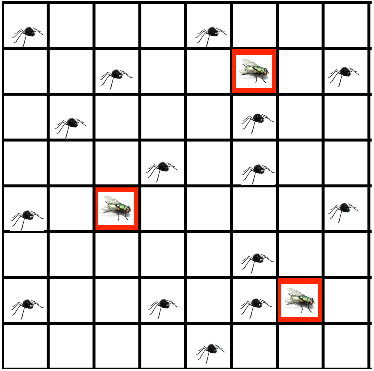
$$\cdots \qquad \cdots \qquad \cdots \qquad \cdots$$

$$\tilde\mu_m(x) \in \arg\min_{u_m} E_w\Big\{g(x, \tilde\mu_1(x), \tilde\mu_2(x), \ldots, \tilde\mu_{m-1}(x), u_m, w) + \alpha J_\mu\big(f(x, \tilde\mu_1(x), \tilde\mu_2(x), \ldots, \tilde\mu_{m-1}(x),$$

- Has a search space with size that is linear in m; ENORMOUS SPEEDUP!

Survey reference: Bertsekas, D., "Multiagent Reinforcement Learning: Rollout and Policy Iteration," IEEE/CAA J. of Aut. Sinica, 2021 (and earlier papers quoted there).

Spiders-and-Flies Example
(e.g., Delivery, Maintenance, Search-and-Rescue, Firefighting)



15 spiders move in 4 directions with perfect vision

3 blind flies move randomly

- Objective is to catch the flies in minimum time
- At each time we must select one out of $\approx 5^{15}$ joint move choices
- Multiagent rollout reduces this to $5 \cdot 15 = 75$ choices (while maintaining cost improvement); applies a sequence of one-spider-at-a-time moves
- Later, we will introduce “precomputed signaling/coordination" between the spiders, so the 15 spiders will choose moves in parallel (extra speedup factor of up to 15)

Four Spiders and Two Flies: Illustration of Various Forms of Rollout

Video: Base Policy

Video: Standard Rollout

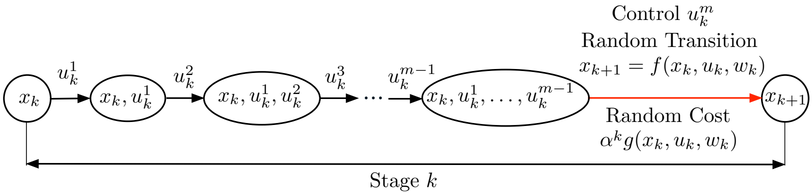
Video: Mutiagent Rollout

Base policy: Move along the shortest path to the closest surviving fly (in the Manhattan distance metric). No coordination.

Time to catch the flies

- Base policy (each spider follows the shortest path): Capture time = 85
- Standard rollout (all spiders move at once, $5^4 = 625$ move choices): Capture time = 34
- Agent-by-agent rollout (spiders move one at a time, $4 \cdot 5 = 20$ move choices): Capture time = 34

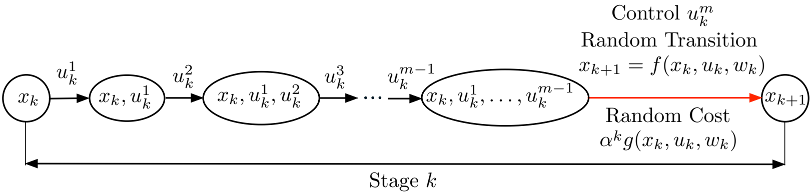
Let's Take a Working Break



Think about an equivalent problem reformulation for multiagent rollout

- "Unfold" the control action
- Consider standard (not multiagent) rollout for the reformulated problem
- What about cost improvement?

Justification of Cost Improvement through Reformulation: Trading off Control and State Complexity (NDP Book, 1996)



An equivalent reformulation - “Unfolding" the control action

- The control space is simplified at the expense of $m-1$ additional layers of states, and corresponding $m-1$ cost functions

$$J^1(x, u_1), J^2(x, u_1, u_2), \ldots, J^{m-1}(x, u_1, \ldots, u_{m-1})$$

- Multiagent rollout is just standard rollout for the reformulated problem
- The increase in size of the state space does not adversely affect rollout (only one state per stage is looked at during on-line play)
- Complexity reduction: The one-step lookahead branching factor is reduced from n^m to $n \cdot m$, where n is the number of possible choices for each component u_i

Multiagent MPC (A Form of Simplified MPC)

Consider MPC where u_k consists of both discrete and continuous components

$$u_k = (y_k^1, \dots, y_k^m, v_k),$$

where $y_k^1, \dots, y_k^m$ are discrete, and v_k is continuous.

- For example $y_k^1, ..., y_k^m$ may be system configuration variables, and v_k may be a multidimensional vector with real components (e.g., as in linear quadratic control).
- The base policy may consist of a "nominal configuration" $\bar{y}_k^1, ..., \bar{y}_k^m$ (that depends on the state x_k), and a continuous control policy that drives the state to 0 in $(\ell - 1)$ steps with minimum cost.
- In a component-by-component version of MPC, at state x_k:
  - $y_k^1, \dots, y_k^m$ are first chosen one-at-a-time, and with all future components fixed at the values determined by the nominal configuration/base policy.
  - Then the continuous component v_k is chosen to drive the state to 0 in ℓ steps at minimum cost with the discrete components fixed.
- This simplifies lookahead minimization by:
  - Separating the "difficult" minimization over $y_k^1, \dots, y_k^m$ from the continuous minimization over v_k
  - Optimizing over $y_k^1, \dots, y_k^m$ one-at-a-time (simpler integer programming problem).
- Maintains the cost improvement/stability property of MPC.

Parallelization of Agent Actions in Multiagent Rollout: Allowing for Agent Autonomy

Multiagent rollout/policy improvement is an inherently serial computation. How can we parallelize it, to get extra speedup, and also deal with agent autonomy?

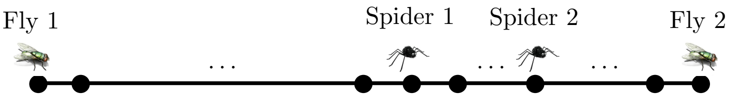
Precomputed signaling

- Obstacle to parallelization: To compute the agent ℓ rollout control we need the rollout controls of the preceding agents $i < \ell$
- Signaling remedy: Use precomputed substitute "guesses" $\widehat{\mu}_i(x)$ in place of the preceding rollout controls $\tilde{\mu}_i(x)$

Signaling possibilities

- Use the base policy controls for signaling $\widehat{\mu}_i(x) = \mu_i(x)$, $i = 1, \ldots, \ell - 1$ (this may work poorly)
- Use a neural net representation of the rollout policy controls for signaling $\widehat{\mu}_i(x) \approx \tilde{\mu}_i(x)$, $i = 1, \ldots, \ell - 1$ (this requires training/off-line computation)
- Other, problem-specific possibilities

The Pitfall of Using the Base Policy for Signaling



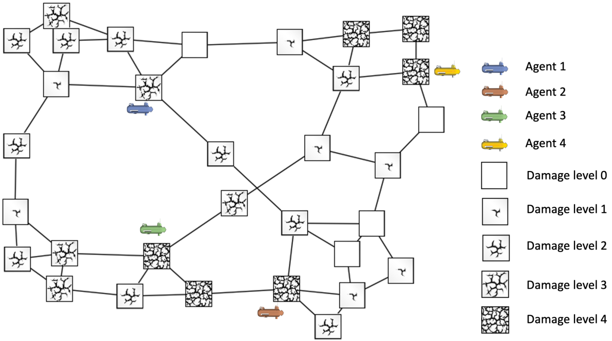
Two spiders trying to catch two stationary flies in minimum time

- The spiders have perfect vision/perfect information. The flies do not move.
- Base policy for eachspider: Move one step towards the closest surviving fly

Performance of various algorithms

- Optimal policy: Split the spiders towards their closest flies
- Standard rollout is optimal for all initial states (it can be verified)
- Agent-by-agent rollout is also optimal for all initial states (it can be verified)
- Agent-by-agent rollout with base policy signaling is optimal for “most" initial states, with A SIGNIFICANT EXCEPTION
- When the spiders start at the same location, the spiders oscillate and never catch the flies

Multirobot Repair of a Network of Damaged Sites (2020 Paper by Bhatacharya, Kailas, Badyal, Gil, DPB, from my Website)



- Damage level of each site is unknown, except when inspected. It deteriorates according to a known Markov chain unless the site is repaired (this is a POMDP)
- Control choice of each robot: Inspect and repair (which takes one unit time), or inspect and move to a neighboring site
- State of the system: The set of robot locations, plus the belief state of the site damages
- Stage cost at each unrepaired site: Depends on the level of its damage

Videos: Multirobot Repair in a Network of Damaged Sites (Agents Start from the Same Location)

Video: Base Policy (Shortest Path/No Coordination)

Video: Multiagent Rollout

Video: Multiagent with Base Policy Signaling

Video: Multiagent with Policy Network Signaling

Cost comparisons

- Base policy cost: 5294 (30 steps)
- Multiagent rollout : 1124 (9 steps)
- Multiagent Rollout with base policy signaling: 31109 (Never stops)
- Multiagent Rollout with neural network policy signaling: 2763 (15 steps)

We will return to this problem in the future (in the context of infinite horizon policy iteration)

About the Next Lecture

We will cover:

- Rollout algorithms for constrained deterministic problems
- Applications in combinatorial and discrete optimization

Note on today's and next lectures:

- The material on rollout and MPC are minimally covered in the class notes. The book "Lessons from AlphaZero ..." has more material.
- Multiagent rollout is covered extensively in the survey paper D. P. Bertsekas, “Multiagent Reinforcement Learning: Rollout and Policy Iteration," IEEE/CAA Journal of Automatica Sinica, Vol. 8, 2021, pp. 249-271; see also the corresponding Video at http://web.mit.edu/dimitrib/www/RLbook.html.

Homework: Exercise 1.3 of latest version of class notes; due Sunday, Feb. 27

About questions on your project

- Send me email (dbertsek@asu.edu)
- Make appointment to talk by zoom (there are no fixed office hours in this course)