Sampling-based Algorithms
planning for stochastic systems and
complex task specifications

Pratik Chaudhari*

Laboratory of Information and Decision Systems,
Massachusetts Institute of Technology

* Joint work with
Luis Reyes-Castro, Valerio Varricchio, Jana Tumova, Nok Wongpiromsarn,
Sertac Karaman, David Hsu, Emilio Frazzoli, Daniela Rus

April 2, 2014
Planning and control of autonomous systems

**Challenges:**
- continuous vs discrete
  - physical vs computational
- uncertainty
  - sensor noise, unknown dynamics
- logical specifications

**Emphasis:**
- provable guarantees
  - optimality, robustness, completeness
- real-time computation
- guarantee safety
An example: Robot motion-planning

Problem:

- **dynamics:**

  \[ \dot{x}(t) = f(x(t), u(t)), \quad \forall t \]

- **objective:**

  **find dynamically feasible trajectory s.t.**
  
  1. does not collide against obstacles

  \[ x(t) \notin X_{obs}, \quad \forall t \]

  2. optimizes a cost function

  *time, control effort, reward states etc.*
RRT* algorithm

1. draw random samples from $X_{\text{free}}$
2. connect nearby samples
   - use *locally optimal controllers*
   - rewire vertices
   - *can neighbors improve cost using me?*
3. repeat until goal

**Key idea**

Graph $(V_n, E_n)$ is a finite, deterministic and *optimal* abstraction

**Asymptotic optimality**

w.p. 1, the cost of best trajectory in the tree converges to the optimal cost $c^*$

$$\mathbb{P} \left( \lim_{n \to \infty} c_n = c^* \right) = 1$$

**Probabilistic completeness**

if there exists a feasible solution, the algorithm finds it

$$\lim_{n \to \infty} \mathbb{P} (V_n \cap X_{\text{goal}} = \emptyset) = 0$$
Proof techniques

1. RRT* is an example of a random geometric graph
2. rewire all neighbors within a distance $r^*_n = O\left(\frac{\log n}{n}\right)^{1/d}$
3. if $r_n = r^*_n$, graph connected w.p. 1
4. connectivity of the graph is crucial for optimality

# neighbors < $O\left(\log n\right)$  # neighbors = $O\left(\log n\right)$
In action …
In action ...
1. **Stochastic Estimation and Control**
   - Nonlinear filtering
   - POMDPs

2. **Urban Planning with Formal Specifications**
   - Linear Temporal Logic
   - Process Algebras
Problem:

- dynamics:
  \[ dx(t) = f(x(t), u(t)) \, dt + F \, dw(t) \]

- observations:
  \[ dy(t) = g(x(t)) \, dt + G \, dv(t) \]

- \( w(t), v(t) \) is Brownian motion

  same as Gaussian noise, just continuous
Problem:

- **dynamics:**
  \[ dx(t) = f(x(t), u(t)) \, dt + F \, dw(t) \]

- **observations:**
  \[ dy(t) = g(x(t)) \, dt + G \, dv(t) \]

- \( w(t), v(t) \) is Brownian motion

**Filtering:** find best estimate of \( x(t) \) using all observations upto time \( t \)

\[ \hat{x}(t) = \mathbb{E} [x(t) \mid Y_t] \]

Construct a HMM
Stochastic estimation and control

Problem:

- **dynamics:**
  \[ dx(t) = f(x(t), u(t)) \, dt + F \, dw(t) \]

- **observations:**
  \[ dy(t) = g(x(t)) \, dt + G \, dv(t) \]

- \( w(t), v(t) \) is Brownian motion

Filtering: find best estimate of \( x(t) \) using all observations upto time \( t \)

\[ \hat{x}(t) = \mathbb{E}[x(t) \mid Y_t] \]

Output feedback: find best control \( u(t) \) using all observations upto time \( t \)

\[ u^* = \arg \max_u \mathbb{E}_{w,v}[J(x, u) \mid Y_t] \]

Construct a HMM

Looks like a POMDP
Let us construct a finite Markov chain $M$

- propagate $m$ particles from state $x$ for time $\Delta t$

$$P_{pf}(x' \mid x) = \frac{\# \text{ particles in voronoi}(x')}{m}$$

- can show that this Markov chain converges to the continuous system (？)

$$\lim_{n,m \to \infty} M_{pf} \to dx(t) = f(x(t)) \, dt + F \, dw(t)$$

$p_1, p_2 >> p_3, p_4, p_5$
Let us construct a finite Markov chain $M$

- propagate $m$ particles from state $x$ for time $\Delta t$

$$P_{pf}(x' | x) = \frac{\# \text{ particles in voronoi}(x')}{m}$$

- can show that this Markov chain converges to the continuous system (?)

$$\lim_{n,m \to \infty} M_{pf} \to dx(t) = f(x(t)) \, dt + F \, dw(t)$$

Do we need to work this hard?
Some intuition

Let us construct a finite Markov chain $M$

- propagate $m$ particles from state $x$ for time $\Delta t$

$P_{pf}(x' | x) = \frac{\# \text{particles in voronoi}(x')}{m}$

- can show that this Markov chain converges to the continuous system(?)

$$\lim_{n,m \to \infty} M_{pf} \to dx(t) = f(x(t)) \, dt + F \, dw(t)$$

Do we need to work this hard? No . . .

No, just ensure local consistency

for all $x \in M$, as $n \to \infty$,

$$\Delta t_n(x) \to 0$$

$$\frac{\mathbb{E}[\Delta x \mid x]}{\Delta t_n(x)} \to f(x) \quad \text{and} \quad \frac{\text{Cov}[\Delta x \mid x]}{\Delta t_n(x)} \to F F'$$

Theorem [Kushner ‘01]

Trajectories of Markov chain converge in distribution to those of the continuous system

$$\psi_n \overset{D}{\to} x$$
An example

\[
\begin{align*}
\dot{x}_1 &= -\frac{x_1}{2} + \sigma \tilde{w}_1, \\
\dot{x}_2 &= -x_2 + \sigma \tilde{w}_2,
\end{align*}
\]

\[
x_1(0) \sim 0.8 \
x_2(0) \sim 0.8
\]

1000 states

40,000 states
Discrete POMDPs

- Tuple of $(S, U, O, P, Q, b_0)$
  - $S$: set of states
  - $U$: set of controls
  - $O$: set of observations
  - $P$: transition probabilities between states
  - $Q$: observation probabilities at every state
  - $b_0$: initial belief

- Find $\pi : B \rightarrow U$ to minimize

$$\mathbb{E} \left[ \sum_{k=1}^{T} l(s, \pi(b), k) + L(s(T)) \mid s(0) \sim b_0 \right]$$

- Exact solution only for a few cases, e.g., LQG
- Approximate cost function using Bellman backup on $\alpha$-vectors
Approach

**Solution techniques**

- Belief space is huge: $|S|$-dimensional
- Reachable belief space $\mathcal{R}(b_0)$ is much smaller
- **Optimally** reachable belief space $\mathcal{R}^*(b_0)$ is even smaller

**SARSOP [Kurniawati, Hsu ’08]**

- Sample a finite set of beliefs, perform Bellman updates
- Can solve large problems with 1000’s of states

**Key idea**

- Discrete approximations of continuous POMDPs
- Solve *incrementally*, i.e., reuse previous policy
Approach

Solution techniques

- Belief space is huge: $|S|$-dimensional
- Reachable belief space $\mathcal{R}(b_0)$ is much smaller
- **Optimally** reachable belief space $\mathcal{R}^*(b_0)$ is even smaller

SARSOP [Kurniawati, Hsu ’08]

- Sample a finite set of beliefs, perform Bellman updates
- Can solve large problems with 1000’s of states

Key idea

- Discrete approximations of continuous POMDPs
- Solve **incrementally**, i.e., reuse previous policy

Optimally reachable belief space
Example – Light-dark domain

- Inaccurate observations in “dark” region with accurate observations near beacons,

\[
dx = u \, dt + F \, dw \\
dy = x \, dt + G(x) \, dv
\]

\[
G(x) = \begin{cases} 
\epsilon & : |x - b_1| < e_1 \\
1/\epsilon & : \text{otherwise}
\end{cases}
\]

- Terminal action to claim reward
- Reward of 1000 if inside goal region, else penalty of -1000
- Cost function quadratic in control
- Optimal policy shows information-gathering behavior

Blue: discrete belief, Grey: dark region, Red: goal region, Green: light region
Example – Light dark domain
1. Stochastic Estimation and Control
   - Nonlinear filtering
   - POMDPs

2. Urban Planning with Formal Specifications
   - Linear Temporal Logic
   - Process Algebras
Some examples

DARPA Urban Challenge:

- Travel an urban landscape with traffic
- Obey rules of the road, e.g., intersections, passing, merging
- Execute parking maneuvers, U-turns
Some examples

DARPA Urban Challenge:

- Travel an urban landscape with traffic
- Obey rules of the road, e.g., intersections, passing, merging
- Execute parking maneuvers, U-turns

Meet Talos:

- LR3 Landrover with state of the art sensors
- 5 cameras, 12 LIDARs, 16 radars and a 3D laser
- Several megabytes of data / sec
- Computer with 40 CPUs
Let us elaborate on the rules …

**Decoupled approach**

- **Intuition:** Driving rules / safety constraints translate into obstacles
- **If goal infeasible,** relax constraints one by one until car finds a plan

**Examples**

- **Lane constraints** - if no progress in 15 seconds, relax the constraint *do not enter left lane*
  - Ensures that nominally, car stays in right lane, goes to left lane only to pass a stopped car

- **Speed constraints** - stay below certain speed, easy to enforce
  - Have to follow a slow moving car, i.e., *conflicts* with lane constraint

- **Dynamical constraints** - How to check a *continuous* trajectory for *logical* rules?
Lane constraint in action
Lessons to be learnt

Drawbacks

- Debugging is tedious
- Parameters have to be tuned and tested extensively
- Difficult to verify interaction of different rules
  - Cornell’s car backs up in the middle of the road
Lessons to be learnt

Drawbacks

- Debugging is tedious
- Parameters have to be tuned and tested extensively
- Difficult to verify interaction of different rules
  - Cornell’s car backs up in the middle of the road
The Cornell-MIT saga …
Lessons to be learnt

Key points

- Integrate *high-level* planning and *low-level* execution tightly
  - Bridge gap between *discrete decision making* and *continuous control* trajectories

- Need a formal way to analyse motion planning with safety rules
Problem setup

- A typical problem can be formulated as,

<table>
<thead>
<tr>
<th>Problem statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Given,</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>- a dynamical system ( \dot{x} = f(x, u), \quad x(0) = x_0, )</td>
</tr>
<tr>
<td>- a temporal logic formula ( \phi ) on ( \Pi ),</td>
</tr>
<tr>
<td>- a labeling function ( \mathcal{L} : X \rightarrow 2^\Pi ),</td>
</tr>
</tbody>
</table>

find a control law \( u(x) \) such that the system satisfies \( \phi \).

- Rules of the road are modeled using (Finite) Linear Temporal Logic, e.g.,

\[
\psi_1 = G(\text{right lane}), \quad \psi_2 = G(\neg \text{wrong direction}), \ldots
\]

- **Focus:** What if \( \phi \) cannot be satisfied?

  - *is there a subset of specifications that can be broken to satisfy the task?*
Safety rules

- **Examples:**
  1. Never hit pedestrians / obstacles
  2. Always travel in the correct direction:
     \[ \psi_2 = G \left( \bigvee_{* \in 2^\Pi} (\ast, \text{dir}) \right) \]
  3. Always travel in right lane:
     \[ \psi_3 = G \neg \left( (rl, ll) \lor (ll, rl) \right) \]
Safety rules

**Examples:**

1. Never hit pedestrians / obstacles
2. Always travel in the correct direction:
   \[
   \psi_2 = G \left( \bigvee_{\ast \in 2^\Pi} (\ast, \text{dir}) \right)
   \]
3. Always travel in right lane:
   \[
   \psi_3 = G \neg \left( (\text{rl, ll}) \lor (\text{ll, rl}) \right)
   \]

**Level of unsafety**

- For a word \( w \) constructed from a continuous trajectory \( x \),
  \[
  \lambda(w, \psi) = \min_{w' \models \psi} \langle w \rangle - \langle w' \rangle
  \]
  i.e., find the largest sub-sequence \( w' \) that satisfies the safety rule \( \psi \)

- Lexicographical ordering for prioritized specifications,
  \[
  \lambda(w) = (\lambda(w, \psi_1), \ldots, \lambda(w, \psi_n))
  \]
Solution

- Iteratively refine Kripke structure, e.g., using RRT*
- Construct weighted automaton that combines,
  1. durational Kripke structure
  2. product automaton of all safety rules that computes the level of unsafety of a word
- Find the trajectory that minimizes the lexicographical weight
  - translates to the minimum-violation trajectory of the dynamical system
Iteratively refine Kripke structure, e.g., using RRT*

Construct weighted automaton that combines,

1. durational Kripke structure
2. product automaton of all safety rules that computes the level of unsafety of a word

Find the trajectory that minimizes the lexicographical weight

- translates to the minimum-violation trajectory of the dynamical system

**Theorem**

- If $x_n$ is the trajectory returned by the algorithm after $n$ iterations,

$$\Pr \left( \lim_{n \to \infty} \|x_n - x^*\|_{BV} = 0 \right) = 1$$

where $x^*$ is the optimal trajectory

- $\Theta(m^2 \log n)$ work per iteration for a product automaton of size $m$
• do not go on sidewalk, do not travel in wrong direction

• Add do not change lanes frequently
Examples - 2
Autonomous golfcart
Planning with process algebras
Powerful methods to construct *concretizations* of dynamical systems

*Tightly integrate* estimation, control and even verification with motion planning
  
  *translates to real-time algorithms very naturally*

**Possible future directions—**

- Multi-agent systems
  
  *compositional approaches, conflict resolution, fixed-point logics*

- A *robot may not injure ... or allow a human being to come to harm*
  
  *can an autonomous car detect a tired, intoxicated driver and take control?*