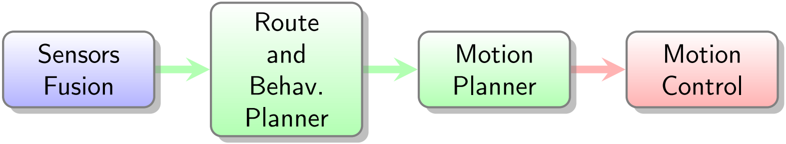
Decision Making under Uncertainty for Autonomous Driving

Philippe Weingertner

January 8, 2019

Decision Making under Uncertainty for Autonomous Driving



Decision Making under Uncertainty for Autonomous Driving

- Driving Strategy is a Sequential Decision Making problem:
  - Strategic level: every e.g. a few seconds decide which maneuver to do
  - Tactical level: every e.g. 100 ms decide how to change $a_{longitudinal}, a_{lateral}$
  - Trajectory can be seen as:
    - a set of spatio-temporal points $\{x_i, y_i, t_i\}_{1..N}$
    - or a starting point with a set of accelerations every time steps $\{\ddot{x}_i, \ddot{y}_i\}_{1..N}$
- Route planner: long-term decisions
- Behavioral planner: mid-tem decisions
- Motion planner: short-term decisions

Decision Making under Uncertainty for Autonomous Driving

- Multiple sources of Uncertainty:
  - sensors uncertainty
  - occlusions
  - other agents behaviors
- Conflicting objectives:
  - efficiency: Time To Goal
  - comfort: low jerk
  - safety: safety distances
- How to make good decisions dealing with multiple sources of uncertainty and satisfying conflicting objectives ?

Rationale Decision Making

- Rationale Decision Making is reasoning about uncertainty and objectives
  - Uncertainty: a Bayesian Network models uncertainties and dependencies. It is a joint distribution model.
  - Objectives: define a utility function (or value function or Q function) which corresponds to our preferences (efficiency, comfort, safety...)

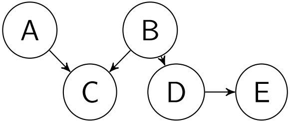
Uncertainty: Probabilities and Bayes Rule

- Definition of Conditional Probability: $P(A \mid B) = \frac{P(A,B)}{P(B)}$
- Law of Total Probability:
  - $P(A) = \sum_{B \in \mathcal{B}} P(A,B) = \sum_{B \in \mathcal{B}} P(A \mid B)P(B)$
  - $P(A \mid C) = \sum_{B \in \mathcal{B}} P(A,B \mid C) = \sum_{B \in \mathcal{B}} P(A \mid B,C)P(B \mid C)$
- Bayes Rule: $P(State \mid Obs) = \frac{P(O|S)P(S)}{P(O)} \propto P(O \mid S)P(S)$
- Bayesian Network is a compact representation of joint distribution
  - $\boxed{P(E \mid B,S) \text{ has } (n_E - 1) \times n_B \times n_S \text{ independant params}}$
  - BN chain rule: $P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid pa_{x_i})$

Uncertainty: Bayesian Network

- Compact representation of a joint distribution
- Inference: find a distribution over some unobserved variables given a set of observed variables. It might be used when the structure and parameters of the Bayesian network are known

Exact Inference



$P(a^1 \mid b^1, d^1) = \frac{P(a^1, b^1, d^1)}{P(b^1, d^1)}$ by definition of Cond Prob

$P(a^1, b^1, d^1) = \sum_c \sum_e P(a^1, b^1, c, d^1, e)$ by Law of Tot Prob

$P(a^1, b^1, c, d^1, e) = P(a^1)\mathbf{P(b^1)P(d^1 \mid b^1)}P(c \mid a^1, b^1)P(e \mid d)$ by BN

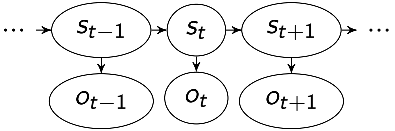
We sum over unobserved variables

4 summations for numerator

8 summations for denominator (but **simplification** here)

Uncertainty: Example of a Bayesian Network, Kalman Filter

Inference for Temporal Models: HMM, Kalman Filter ...



Filtering problem: $P(S_t \mid O_{0:t})$?

By Bayes rule, d-sep, Law of Total Probability

$P(s_t \mid o_{0:t}) = P(s_t \mid o_t, o_{0:t-1}) \propto P(o_t \mid s_t, o_{0:t-1}) P(s_t \mid o_{0:t-1})$

$\mathbf{P(s_t \mid o_{0:t}) \propto P(o_t \mid s_t) \sum_{s_{t-1}} P(s_t \mid s_{t-1}) P(s_{t-1} \mid o_{0:t-1})}$

With continuous variables replace $\sum$ with $\int$

Our known model is:

- Observation model: $P(o_t \mid s_t)$
- State transition model: $P(s_t \mid s_{t-1})$

So we get a **recursive formula** about our belief $b_t(s)$ based on all possible previous states belief $b_{t-1}(s')$

Uncertainty: Recursive Bayesian Estimation

Recursive Bayesian Estimation

function RecursiveBayesianEstimation
    $b_0(s) \leftarrow P(o_0 \mid s)P(s_0)$ for all s
    Normalize b_0
    for $t \leftarrow 1$ to ∞ **do**
        $b_t(s) \leftarrow P(o_t \mid s)\sum_{s'} P(s \mid s')b_{t-1}(s')$ for all s
        Normalize b_t
    end for
end function

Objectives: Utility or Value function

- We define a utility related to preferences (or objectives or costs) so that it follows 4 axioms

Constraints on Rational Preferences

Completeness: we can compare them $A \succ B, A \prec B, A \sim B$

Transitivity: $A \succeq B, B \succeq C \Rightarrow A \succeq C$

Continuity: $A \succeq C \succeq B \Rightarrow \exists p$ s.t. $[A : p; B : 1-p] \sim C$

translated to $\boxed{pU(A) + (1-p)U(B) = U(C)}$

Independence: $A \succ B \Rightarrow \forall(C, p), [A : p; C : 1-p] \succ [B : p; C : 1-p]$

Objectives: Maximum Expected Utility principle

- Bayesian Network + Utility $\implies$ Decision Network

Maximum Expected Utility Principle

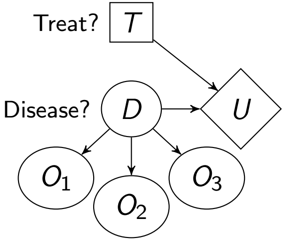
$E[U(a \mid o)] = \sum_{s'} P(s' \mid a, o) U(s')$

A rational agent chooses $a^* = \underset{a}{argmax}\ E[U(a \mid o)]$

Decision Network: example

Evaluating Decision Networks

T	D	U(T,D)
0	0	0
0	1	-10
1	0	-1
1	1	-1



$E[U(a \mid o)] = \sum_{s'} P(s' \mid a, o) U(s')$

s' represents an instanciation of the nodes in the decision network

$E[U(t^1 \mid o_1^1)] = \sum_d P(d \mid t^1, o_1^1) U(t^1, d)$

Then any inference method can be used to evaluate $P(d \mid t^1, o_1^1)$

$\Longrightarrow$**Inference pb: what is the distribution of the variables that are parents to the utility node ?**

To decide whether to apply a treatment compare $E[U(t^1 \mid o_1^1)]$ vs $E[U(t^0 \mid o_1^1)]$

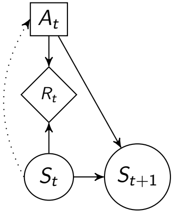
MDP: Markov Decision Process

MDP $< \mathcal{S}, \mathcal{A}, T, R >$ stationary representation

Markov assumption: current state only depends on your previous state and the action you took to get there

Stationary: T, R do not change with time

Decision Network but with $P(S_{t+1} \mid S_t, A_t)$ and $P(R_t \mid A_t, S_t)$, not stationary, in the general case



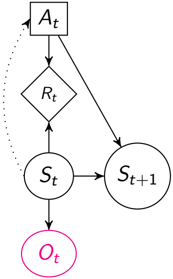
$R(s, a)$: expected reward received when executing action a from state s. Here we assume it is a deterministic function (but not required).

Utility function decomposed into rewards $R_{0:t}$

POMDP: Partially Observable Markov Decision Process

POMDP: $< \mathcal{S}, \mathcal{A}, \mathcal{O}, T, R, O >$

MDP + set of observations $\mathcal{O}$ + observation model O



The proba of observing o given state s is written $\mathbf{O(o \mid s)}$

The decision in a POMDP at time t can only be based on the history of observations $o_{1:t}$

Instead of keeping track of arbitrarily long histories, we keep track of the belief state

$b(s)$ is the probability assigned to being in state s

Policy

- Determines action given past history of states and actions $a = \pi_t(h_t) = \pi_t(s_{0:t}, a_{0:t-1})$
- But with MDP we just care about current state as s_t d-separates past from future
- $\Longrightarrow \pi_t(s_t)$ or $\pi(s_t)$ if the policy is stationary
- If states and actions are discrete: it is just a matrix specifying what action to do in a specific state. A Policy can be deterministic or stochastic
- An optimal policy π^* is a policy that maximizes expected utility: $\pi^*(s) = \underset{\pi}{argmax}\ U^\pi(s)$ for all states s

Bellman Equations

Bellman Equations

$$U_k^*(s) = \max_a \left[R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) U_{k-1}^*(s')\right]$$

$$\pi^*(s) = \operatorname*{argmax}_a \left[R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) U^*(s')\right]$$

$$U^*(s) = \max_a \left[R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) U^*(s')\right]$$

How to find a Policy ?

- Dynamic Programming: small states and actions spaces, known Transition and Reward models.
  - Policy Iteration or Value Iteration algorithms can compute the **optimal** policy (it is proven)
- Approximate Dynamic Programming: when states and/or actions spaces are big or continuous
  - Local Approximation
  - Global Approximation
- Reinforcement Learning:
  - Model-Based: T, R are estimated first (learned from data, Maximum Likelihood for example or derived from Expert knowledge) and then we derive a policy
  - Model-Free: T, R unknown. We try to derive directly the policy by interacting and observing rewards. It usually requires a simulator to avoid real word experimentation.
- Online Methods: do not compute a policy for the entire state space offline. Restrict computation to states reachable from current state

Reinforcement Learning Model-Based

- Estimate T, R, O models. From data or by expert knowledge or a combination ...
- Offline methods:
  - Policy Iteration or Value Iteration based
- Online methods:
  - Forward Search, Branch and Bound Search
  - Sparse Sampling, MCTS (Monte Carlo Tree Search)

Reinforcement Learning Model-Free

Incremental Estimation

X a random variable, try to estimate the mean: $\mu = \mathbb{E}[X]$

With samples $x_1, \dots, x_n$

$\hat{x}_n = \frac{1}{n} \sum_{i=1}^{n} x_i$

$\iff \hat{x}_n = \hat{x}_{n-1} + \frac{1}{n}(x_n - \hat{x}_{n-1})$

$\hat{x} \leftarrow \hat{x} + \alpha(n)(x - \hat{x})$

Key equation in Temporal Difference Learning

$\hat{x} \leftarrow \hat{x} + \alpha(x - \hat{x})$

With x new meas and $\hat{x}$ current estimate

Temporal difference error: $x - \hat{x}$ is the difference between a sample and our previous estimate

Constant LR $\Rightarrow$ decays the influence of past samples exponentially

And as we collect experience, more recent examples based on better Q , are better

A larger LR means new samples have a greater effect on the current estimate

Reinforcement Learning Model-Free

Q-learning

$Q(s,a) = R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) U(s')$

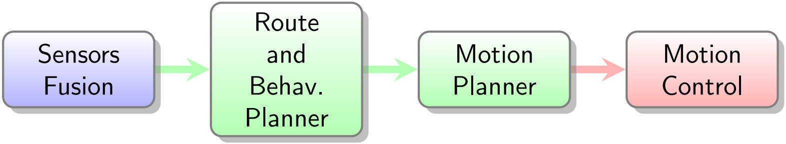
With $U(s') = \max_{a'} Q(s',a')$

How to update Q directly after we observe r and s' ?

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(\mathbf{r} + \gamma \max_{\mathbf{a}'} \mathbf{Q}(\mathbf{s}',\mathbf{a}') - Q(s,a) \right)$$

This update + good exploration strategy $\Rightarrow Q(s,a) \rightarrow Q^*(s,a)$

Decision Making under Uncertainty for Urban Driving



Decision Making under Uncertainty: Volvo Trucks Thesis

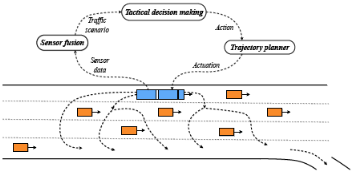


Figure 3.1: *A rough sketch of the architecture in a fully automated truck. Sensor fusion is where the sensor data is interpret, Tactical decision-making is where a high level decision about how to maneuver is taken, and in Trajectory planner the maneuver request is translated to actuations.*

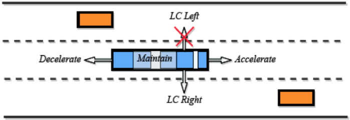
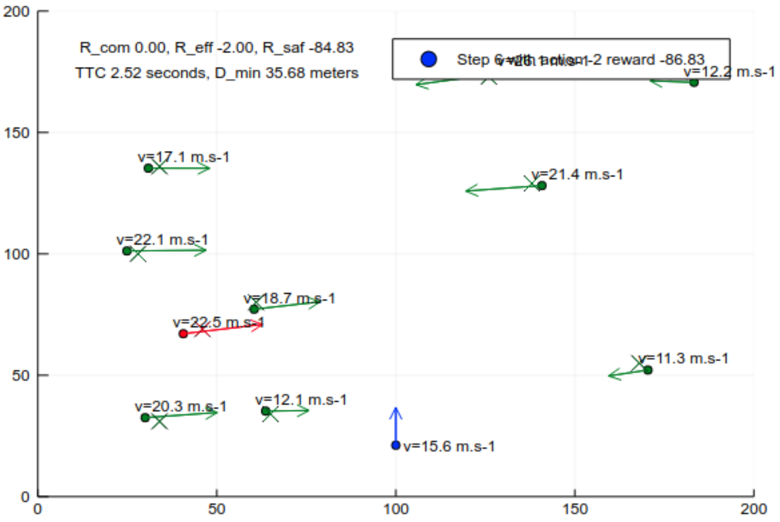


Figure 3.4: *A depiction of the five actions in the action space. The option to change lane to the left is in this instant pruned, since there is a vehicle currently inhabiting this neighbouring lane. The action to accelerate means to increase the value of the ACC state, and respectively to decelerate means to decrease it.*

Decision Making under Uncertainty for Urban Driving



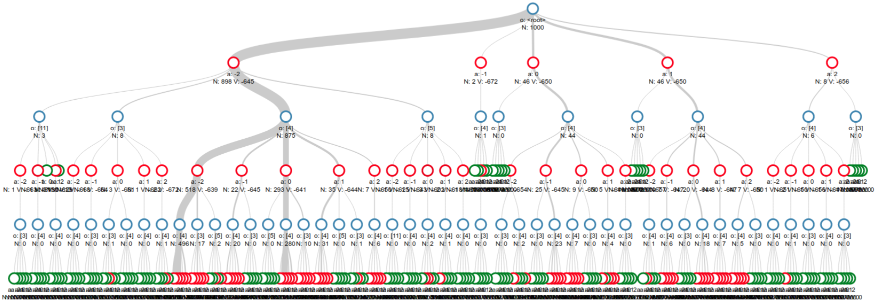
Decision Making under Uncertainty for Urban Driving

- **States**: $\{(x, y, v_x, v_y)_{ego}, (x, y, v_x, v_y)_{obj_{1..n}}\}$
- **Actions**: $a_{longitudinal} \in [-2, -1, 0, 1, 2]\ m.s^{-2}$
  - Follow a lane, think in Frenet coordinates and control $a_{longitudinal}$
  - Define a path for a lane change and control acceleration along that path
- **Observations**: States observed via a sensor
- **Transition model**: Linear Gaussian Dynamics, Kalman filter type with $T(s' \mid s, a) = \mathcal{N}(s' \mid T_s s + T_a a, \Sigma_s)$
- **Observation model**: Linear Gaussian Observation sensor model $O(o \mid s') = \mathcal{N}(o \mid O_s s', \Sigma_o)$
- **Reward model**: accounts for efficiency (Time To Goal), comfort (Hard Braking) and safety (Time To Collision)

Decision Making under Uncertainty for Urban Driving

- Transition model: Linear Gaussian Dynamics, Kalman filter type with $T(s' \mid s, a) = \mathcal{N}(s' \mid T_s s + T_a a, \Sigma_s)$
  - $s' = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} s + \begin{bmatrix} \frac{dt^2}{2} & 0 \\ 0 & \frac{dt^2}{2} \\ dt & 0 \\ 0 & dt \end{bmatrix} a = T_s s + T_a a$
  - $s' = \begin{bmatrix} x_{k+1} \\ y_{k+1} \\ x'_{k+1} \\ y'_{k+1} \end{bmatrix}$ and $s = \begin{bmatrix} x_k \\ y_k \\ x'_k \\ y'_k \end{bmatrix}$
  - $\begin{cases} x_{k+1} = x_k + x'_{k+1} dt + x''_{k+1} \frac{dt^2}{2} & + \mathcal{N}(0, \sigma_x) \\ y_{k+1} = y_k + y'_{k+1} dt + y''_{k+1} \frac{dt^2}{2} & + \mathcal{N}(0, \sigma_y) \\ x'_{k+1} = x'_k + x''_k dt & + \mathcal{N}(0, \sigma_{x'}) \\ y'_{k+1} = y'_k + y''_k dt & + \mathcal{N}(0, \sigma_{y'}) \end{cases}$

POMCP: MCTS with POMDP



POMCP: MCTS with POMDP

function SelectAction(b, d)
  $h \leftarrow \emptyset$
  loop
    $s \sim b$
    Simulate(s, h, d)
  end loop
  return $\arg\max_a Q(h, a)$
end function

function Simulate(s, h, d)
  if $d = 0$ **then**
    return 0
  end if
  $a \leftarrow \arg\max_a Q(h, a) + c\sqrt{\frac{\log N(h)}{N(h,a)}}$
  $(s', o, r) \sim G(s, a)$
  $o_{class}, ttc \leftarrow$ ClusterizeObs(s',o)
  if $hao_{class} \notin T$ **then**
    for $a \in A(s)$ **do**
      $(N(h, a), Q(h, a)) \leftarrow (N_0(h, a), Q_0(h, a))$
    end for
    $o^{id}_{class} = o_{class}, o^{ttc}_{class} = ttc, o^{raw}_{class} = o$
    $T = T \cup \{hao_{class}\}$
    return Rollout(s, d, π_0)
  else if ttc $< o^{ttc}_{class}$ **then**
    $o^{ttc}_{class} = ttc, o^{raw}_{class} = o$
  end if
  $q \leftarrow r + \lambda$ Simulate$(s', hao_{class}, d - 1)$
  $N(h, a) \leftarrow N(h, a) + 1$
  $Q(h, a) \leftarrow Q(h, a) + \frac{q - Q(h,a)}{N(h,a)}$
  return q
end function

function ClusterizeObs(s', o)
  $ttc \leftarrow$ SmallestTimeToCollision(s',o)
  $o_{class} = floor(min(ttc, 11))$
  return o_{class}, ttc
end function

function Rollout(b, d, π_0)
  if $d = 0$ **then**
    return 0
  end if
  $a \sim \pi_0(b)$
  $s \sim b$
  $(s', o, r) \sim G(s, a)$
  $b' \leftarrow$ UpdateBelief(b,a,o)
  return $r + \lambda$ Rollout$(b', d - 1, \pi_0)$
end function

Decision Making under Uncertainty for Urban Driving

- Paper: AA228 Final Project
- Demo: online demo

Table : Benchmark results

	% Collisions	Time To Goal	Hard Breaking
policy_v0	80%	11 s	8
Policy TTC	60%	11.8 s	10
policy_v2	60%	11.8 s	10
policy POMDP	0%	12.8 s	10

References

- Mykel J. Kochenderfer. *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2015.
- Yuanfu Luo et al. "Autonomous Driving among Many Pedestrians: Models and Algorithms". In: *CoRR* abs/1805.11833 (2018). arXiv: 1805.11833. url: http://arxiv.org/abs/1805.11833.
- Anders Nordmark and Oliver Sundell. "Tactical Decision-Making for Highway Driving". In: *Volvo Trucks Master thesis*. 2018. url: http://publications.lib.chalmers.se/records/fulltext/256127/256127.pdf.
- Zachary N. Sunberg and Mykel J. Kochenderfer. "Safety and efficiency in autonomous vehicles through planning with uncertainty". In: *PhD thesis*. 2018. url: https://zachary.sunberg.net/thesis.pdf.