DPOC Summary

Jorit Geurts, jgeurts@ethz.ch

Version: 25. Januar 2024

1 Mathematics

1.1 Linear Algebra

(Semi) Posivite Definite Matrix iff all eigenvalues $(\geq 0) > 0$

Matrix Inverse 2x2: $A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} A_{22} & -A_{12} \\ -A_{21} & A_{11} \end{bmatrix}$

1.2 Calculus

Del-Operator (Gradient): $\nabla_x f(x) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(x) & \cdots & \frac{\partial}{\partial x_n} f(x) \end{bmatrix}^\top$

Hessian

$$\frac{\partial^2 f}{\partial x^2} = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Nonhomogeneous DGL Solution:

$$\dot{x}(t) + cx(t) = f(t) \rightarrow x(t) = x_h(t) + x_p(t)$$

$$x_h(t) \rightarrow f(t) = 0, \quad x_p(t) \text{ using Ansatz solution}$$

1. order inhomogeneous DE

$$\dot{p}(t) + ap(t) = be^{ct} + d$$

$$p(t) = \frac{b}{a-c} e^{ct} + \frac{d}{a} + Ke^{-at}$$

Vareous

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \quad \sum_0^\infty q^k = \frac{1}{1-q}$$

2 Probability Theory

2.1 Random Variables

2.1.1 Discrete Random Variables

Random Variable defined by $p_x, \mathcal{X}$

- $\mathcal{X} \subset \mathbb{Z}$ of all possible outcomes
- $p_x(\bar{x}) \geq 0$ and $\sum_{\bar{x} \in \mathcal{X}} p_x(\bar{x}) = 1$ bzw. $\sum_{\bar{x} \in \mathcal{X}} p_{x|y}(\bar{x}|\bar{y}) = 1$

Margin. / Sum Rule	$p(\bar{x}) = \sum_{\bar{y} \in \mathcal{Y}} p_{xy}(\bar{x}, \bar{y})$
Cond. / Product Rule	$p_{x\|y}(\bar{x}\|\bar{y}) := \frac{p_{xy}(\bar{x},\bar{y})}{p_y(\bar{y})}$
Total Prob. Theorem	$p_x(\bar{x}) := \sum_{\bar{y} \in \mathcal{Y}} p_{x\|y}(\bar{x}\|\bar{y}) p_y(\bar{y})$

2.1.2 Conditional PDF

Conditional PDF

Margin. / Sum Rule	$p(\bar{x}\|\bar{z}) = \sum_{\bar{y} \in \mathcal{Y}} p_{xy\|z}(\bar{x}, \bar{y}\|\bar{z})$
Cond. / Product Rule	$p_{x\|yz}(\bar{x}\|\bar{y}, \bar{z}) := \frac{p_{xy\|z}(\bar{x},\bar{y}\|\bar{z})}{p_{y\|z}(\bar{y}\|\bar{z})}$

Independence

$$p(x|y) = p(x) \quad \Leftrightarrow \quad p(y|x) = p(y) \quad \Leftrightarrow \quad p(x,y) = p(x)p(y)$$

$p(x,y,z) = p(x,y|z)p(z) \rightarrow$ x,y indp. $p(x,y|z) = p(z|x,y)$

Conditional Independence

The knowledge of z makes x and y independent:

$$p(x|y,z) = p(x|z) \quad \Leftrightarrow \quad p(x,y|z) = p(x|z)p(y|z)$$

Caution!!! in general we still have: $p(x,y) \neq p(x)p(y)$

Caution!!! Independence $\not\Rightarrow$ Conditional Independence

2.2 Expectation and Variance

2.2.1 Expectation

Definition: Integral for CRV!

$$\mathbb{E}_x[x] = \sum_{\bar{x} \in \mathcal{X}} \bar{x} p_x(\bar{x})$$

Linearity	$\mathbb{E}_{xy}[a + bx + cy] = a + b\,\mathbb{E}_x[x] + c\,\mathbb{E}_y[y]$
Multi Variable	$\mathbb{E}_{xy}[g(x,y)] = \sum_{\bar{y}} \sum_{\bar{x}} g(\bar{x},\bar{y}) p_{xy}(\bar{x},\bar{y})$
Independence	$\mathbb{E}_{xy}[xy] = \mathbb{E}_x[x]\,\mathbb{E}_y[y]$

Law of Unconcious Statistician for $y = g(x)$

$$\mathbb{E}_y[y] = \sum_{\bar{y} \in \mathcal{Y}} \bar{y} p_y(\bar{y}) = \sum_{\bar{x} \in \mathcal{X}} g(\bar{x}) p_x(\bar{x})$$

2.2.2 Variance (generally a matrix)

$$\underset{x}{\mathrm{Var}}[x] = \underset{x}{\mathbb{E}}\left[(x - \mathbb{E}_x[x])(x - \mathbb{E}_x[x])^\top\right] = \underset{x}{\mathbb{E}}\left[x^2\right] - \underset{x}{\mathbb{E}}[x]^2 = \sigma^2$$

Linearity

$$\mathrm{Var}_x[a + bX + cY] = b^2\,\mathrm{Var}_X[x] + c^2\,\mathrm{Var}_Y[y] + 2bc\mathrm{Cov}[X,Y]$$

Covariance: $\mathrm{Cov}(X,Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] \overset{ind.}{=} 0$

3 Dynamic Programming

- Open Loop can never give better performance than Closed Loop
- Open loop is a special case of closed loop
- Without disturbances, theoretically the same
- Open Loop: N_u^N
- Closed Loop: $N_u (N_u^{N_x})^{N-1} = N_u^{N_x(N-1)+1}$

3.1 DP-Setup

Stage: k, with $k = 0, 1, \ldots, N-1$

Dynamics

DP uses a Markov Chain, i.e. next state is fully determined by current state and action, i.e. states are conditionally independent from previous states.

w_k: disturbance vector, independent of the previous states and actions.

$$x_{k+1} = f_k(x_k, u_k, w_k)$$

$$\text{with} \quad x_k \in \mathcal{S}_k, u_k \in \mathcal{U}_k, w_k \sim p_{w_k|x_k,u_k}$$

$$p_{w_k|x_k,u_k,*} = p_{w_k|x_k,u_k} \quad \forall * \in \{x_l, u_l, w_l | l < k\}$$

Policy

controll inputs u_k are generated by an admissible policy $\pi \in \Pi$ such that.

$$u_k = \mu_k(x_k) \quad \forall k \in \{0, 1, \ldots, N-1\}$$

Cost

$$J_N(x_N) = \underset{X_1, W_0 | x_0 = x}{\mathbb{E}} \left[g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right]$$

$$\text{with } g_k(x_k, u_k, w_k) \text{ stage cost,} \quad g_N(x_N) \text{ terminal cost}$$

$$\text{and } X_1 := \{x_1, \ldots, x_N\}, \; W_0 := \{w_0, \ldots, w_{N-1}\}$$

Objective

$$\pi^* = \arg \min_{\pi \in \Pi} J_\pi(x_N)$$

3.2 Principle of Optimality

Let π^* be an optimal policy. For the truncated problem starting at x_i:

$$\underset{X_i+1, W_i | x_i = x}{\mathbb{E}} \left[\sum_{k=i}^{N-1} g_k(x_k, u_k, w_k) + g_N(x_N) \right]$$

the policy $\pi^* = (\mu_i^*(\cdot), \mu_{i+1}^*(\cdot), \ldots, \mu_{N-1}^*(\cdot))$ is also optimal.

3.3 DP-Algorithm

Initialization

$$J_N(x) := g_N(x) \quad \forall x \in \mathcal{S}_N$$

Recursion

$$J_k(x) = \min_{u_k \in \mathcal{U}_k(x)} \underset{*}{\mathbb{E}} \left[g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \right]$$

$$* = w_k | x_k = x, u_k = u$$

for maximization problems, replace $\min$ by $\max$.

There occurs a minimization in u for each each state at each time step.

This results in: $N_u + N_u N_x (N-1)$ minimizations.

Different Cost Functions

Exponential:

$$J_N(x) = \underset{w_k}{\mathbb{E}} \exp\left(g_N(x_N) + \textstyle\sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right)$$

DP-Algorithm:

$$J_N(x_N) = \exp(g_N(x_N))$$

$$J_k(x_k) = \min_{u_k \in \mathcal{U}} \underset{w}{\mathbb{E}} \left[J_{k+1}(f_k(x_k, u_k, w_k)) \exp(g_k(x_k, u_k, w_k)) \right]$$

3.3.1 Converting to Standard from

Time Lags

Dynamics with a time delay i.e.:

$$x_{k+1} = f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k)$$

Define $y = x_{k-1}$ and $s = u_{k-1} \Rightarrow \tilde{x}_k = (x_k, y_k, s_k)$

New Dynamics in standard form:

$$\tilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ y_k \\ s_k \end{bmatrix} = \begin{bmatrix} f_k(x_k, y_k, s_k, u_k, w_k) \\ x_k \\ u_k \end{bmatrix} := \tilde{f}_k(\tilde{x}_k, u_k, w_k)$$

can be done repeatedly for multiple time lags.

Correlated Disturbances

w_k correlated over time (colored noise) can be modeled as:

$$w_k = C_k y_{k+1}, \quad y_{k+1} = A_k y_k + \xi_k$$

A_k and C_k are given and ξ_k are independent random Variables

Augmented state is $\tilde{x}_k = (x_k, y_k)$, y_k must be observed/estimated.

Dynamics of augmented state:

$$\tilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} f_k(x_k, y_k, u_k, w_k) \\ A_k y_k + \xi_k \end{bmatrix} := \tilde{f}_k(\tilde{x}_k, u_k, w_k)$$

Forecasts

If we get a forcast which reveals the probability distribution of w_k (it is assumed that w_k is independent of x_k and u_k).

We get the forecast y_k that w_k will attain a probability distribution out of a finite collection $\{p_{w_k|y_k}(\cdot|1), \ldots, p_{w_k|y_k}(\cdot|m)\}$. In particular we get the forecast $y_k = i$ and thus $w_k \sim p_{w_k|y_k}(\cdot|i)$.

y_k is also distributed according to $y_{k+1} = \xi_k$ where ξ_k are independent variables with values $i \in \{1, \ldots, m\}$ and probabilities $p_{\xi_k}(i)$.

We augment the state with y_k and get $\tilde{x}_k = (x_k, y_k)$.

The new disturbance is now:

$$p(\tilde{w}_k | \tilde{x}_k, u_k) = p(w_k \xi_k | x_k, y_k, u_k) = p(\tilde{w}_k | y_k) p_(\xi_k)$$

New dynamics:

$$\tilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} f_k(x_k, y_k, u_k, w_k) \\ \xi_k \end{bmatrix} := \tilde{f}_k(\tilde{x}_k, u_k, w_k)$$

New DP-Algorithm:

Initialization:

$$J_N(\tilde{x}) = J(x, y) = g_N(x) \quad \forall x \in \mathcal{S}_N, y \in \{1, \ldots, m\}$$

Update: Have to do case distinction for each y resp. i.

$J_k(\tilde{x}) = J_k(x, y)$

$$= \min_{u_k \in \mathcal{U}_k} \underset{*}{\mathbb{E}} \left[g_k(x_k, u_k, w_k) + \sum_{i=1}^m p_{\xi_k}(i) J_{k+1}(f_k(x_k, u_k, w_k), i) \right]$$

$* = (w_k | y_k = y)$

J_{k+1} is not always a function of $\xi_k \rightarrow$ thus just deterministic/the mean is equal to the actual cost.

4 Infinite Horizon Problems

4.1 Setup

Standard DP setup, with $N \rightarrow \infty$.

Dynamics (becomes time invariant)

$$x_{k+1} = f(x_k, u_k, w_k), \quad x_k \in \mathcal{S}, \quad u_k \in \mathcal{U}, \quad w_k \sim p_{w|x,u}$$

Policy

Controll inputs u_k are generated by policy $\pi \in \Pi$ (timeinvariant)

$$u_k = \mu(x_k) \quad \forall k \geq 0$$

Cost (no terminal cost and stationary)

$$J(x) = \underset{X_1, W_0 | x_0 = x}{\mathbb{E}} \left[\sum_{k=0}^\infty g(x_k, u_k, w_k) \right]$$

4.2 Bellman Equation

$$J(x) = \min_{u \in \mathcal{U}} \underset{w}{\mathbb{E}} \left[g(x, u, w) + J(f(x, u, w)) \right]$$

Solving the BE is hard. (Has to be done for all $x \in \mathcal{X}$ simultaneously)

4.3 Stochastic Shortest Path Problem

Dynamics

$$x_{k+1} = w_k, \quad x_k \in \mathcal{S}$$

$$\Pr(w_k = j | x_k = i, u_k = u) = P_{ij}(u), \quad u \in \mathcal{U}$$

Assumption 4.1:

There exists a cost-free termination state, which we designate as state 0. In particular there are $n+1$ states with $\mathcal{S} = \{0, 1, \ldots, n\}$ where.

$$P_{00}(u) = 1 \text{ and } g(0, u, 0) = 0 \quad \forall u \in \mathcal{U}(0)$$

Assumption 4.2:

There exists at least one proper policy $\pi \in \Pi$. Furthermore for every improper policy μ' and at least one state $i \in \mathcal{S}$, the corresponding cost funtion is $J_{\mu'} = +\infty$.

Proper Policy:

A policy is proper if there exists a integer m such that

$$\Pr(x_m = 0 | x_0 = i) > 0 \quad \forall i \in \mathcal{S}$$

There is a path to the goal for every state.

4.3.1 Solution to the SSP

If Assumption 4.1 and 4.2 hold, then:

1. Given a initial condition $V_0(1), \ldots, V_0(n)$, the sequence $V_l(i)$ generated by the iteration

$$V_{l+i}(i) = \min_{u \in \mathcal{U}} \left[g(i, u) + \sum_{j=1}^n P_{ij}(u) V_l(j) \right] \quad \forall i \in \mathcal{S}^+$$

where $\mathcal{S}^+ := \mathcal{S} \setminus \{0\}$ and $q(i, u) = \underset{w}{\mathbb{E}}[g(x, u, w)]$

converges to the optimal cost $J^*(i)$ for all $i \in \mathcal{S}^+$.

2. The optimal cost satisfies the Bellman Equation:

$$J^*(i) = \min_{u \in \mathcal{U}} \left(q(i, u) + \sum_{j=1}^n P_{ij}(u) J^*(j) \right) \quad \forall i \in \mathcal{S}^+$$

3. The solution is unique
4. The minimizing u for each $i \in \mathcal{S}^+$ of the BE gives an optimal policy which is proper

4.4 Solving the Bellman Equation

4.4.1 Value Iteration

Simply iterate the BE until convergence (arbitrary initialization).

$$V_{l+i}(i) = \min_{u \in \mathcal{U}} \left[g(i, u) + \sum_{j=1}^n P_{ij}(u) V_l(j) \right] \quad \forall i \in \mathcal{S}^+$$

Need infinite iterations in theory.

4.4.2 Policy Iteration

Iterate over policies until convergence (terminal state must be exclueded):

Initialization Initialize with a proper policy $\mu^0 \in \Pi$

Stage 1: Policy Evaluation Given a policy μ^h solve for J_μ^h:

$$J_{\mu^h}(i) = q(i, \mu^h(i)) + \sum_{j=1}^n P_{ij}(\mu^h(i)) J_{\mu^h}(j) \quad \forall i \in \mathcal{S}^+$$

Can be represented as solving a linear system $J = G + PJ$. A solution exists if and only if $(I - P)$ is invertible (is the case for proper policies).

Stage 2: Policy Improvement Obtain a new policy μ^{h+1} by:

$$\mu^{h+1}(i) = \arg \min_{u \in \mathcal{U}} \left[q(i, u) + \sum_{j=1}^n P_{ij}(u) J_{\mu^h}(j) \right] \quad \forall i \in \mathcal{S}^+$$

Stage 3: Termination Iterate until $J_{\mu^{h+1}}(i) = J_{\mu^h}(i) \quad \forall i \in \mathcal{S}^+$

- Under Assumptions 4.1 and 4.2, policy iterations always converges to an optimal policy in finite time.
- The policy evaluation (Step 1) always has a unique solution.
- Since PI is initialized with a proper policy, the policy improvement (Step 2) always results in a proper policy.
- Policy Improvement always means: $J_{\mu^{h+1}}(i) \leq J_{\mu^h}(i)$ (not the case for VI)

4.5 VI vs. PI

- Stage 1 of PI is the same as running VI infinitely many times
- PI has time complexity of $\mathcal{O}(n^2(n+p))$ per iteration
- VI has time complexity of $\mathcal{O}(n^2 p)$ per iteration
- VI faster per iteration but in theory needs infinite iterations where as PI terminates in the worst case after p^n iterations (in practice much faster).
- The cost of PI and VI is always the same.
- The policy can be different.

4.5.1 Variant of PI and VI

Gauss-Seidel Update

In practice VI updates V for all states (Calculate $\bar{V}(i)$ for all i with the old values of $V(i)$ and store the $\bar{V}(i)$, then update all $V(i)$ with $\bar{V}(i)$). We can do it iteratively in place:

$$V(i) \leftarrow \min_{u \in \mathcal{U}} \left[q(i, u) + \sum_{j=1}^n P_{ij}(u) V(j) \right]$$

Asynchronous PI

Under mild conditions all combinations of the following will converge:

- Any number of value updates between policy updates
- Any number of states updated at each value update
- Any number of states updated at each policy update

4.5.2 Linear Programming

It can be shown that $V_{l+1}(i) \geq V_l(i) \quad \forall i \in \mathcal{S}^+, \forall l$.

We can thus formulate the BE as a optimization problem:

$$\max_V \sum_{i \in \mathcal{S}^+} V(i)$$

$$\text{s.t.} \quad V(i) \leq q(i, u) + \sum_{j=1}^n P_{ij}(u) V(j) \quad \forall i \in \mathcal{S}^+, \forall u \in \mathcal{U}$$

This is a Linear Program and can be solved using commercial solvers.

Be careful for maximization the equality sign has to be flipped and the optimization becomes a minimization. For discounted problem insert α in front of the sum.

4.6 Discounted Problems

Dynamics: Same as for infinite horizon problems

$$x_{k+1} = w_k, \quad x_k \in \mathcal{S}$$

$$\Pr(w_k = j | x_k = i, u_k = u) = P_{ij}(u), \quad u \in \mathcal{U}(x_k)$$

but *without* a termination state.

Policy

Controll inputs u_k are generated by policy $\pi \in \Pi$

$$u_k = \mu_k(x_k) \quad \forall\, k \geq 0$$

Cost (no terminal cost)

$$J(x) = \mathop{\mathbb{E}}_{X_1, W_0 | x_0 = x} \left[\sum_{k=0}^{\infty} \alpha^k \tilde{g}_k(x_k, u_k, w_k) \right]$$

$\alpha \in (0,1)$ is the discount factor. For initialization only an admissible policy is needed and not a proper one since the discount factor makes sure the problem stays finite.

4.6.1 Conversion to SSP

Introduce a virtual termination state 0 with command $\mathcal{U}(0) = \{stay\}$.
The transition probabilities are ($\tilde{P}_{ij}(u)$ is the original probability):

$$p_{w|x,u}(j|i,u) = P_{ij}(u) = \alpha \tilde{P}_{ij}(u) \quad \forall\, u \in \mathcal{U}(i), \forall\, i,j \in \mathcal{S}^+$$

$$p_{0|x,u}(0|i,u) = P_{i0}(u) = 1 - \alpha \quad \forall\, u \in \mathcal{U}(i), \forall\, i \in \mathcal{S}^+$$

$$p_{w|x,u}(j|0,u) = P_{0j}(u) = 0 \quad \forall\, u = stay, \forall\, j \in \mathcal{S}^+$$

$$p_{w|x,u}(0|0,u) = P_{00}(u) = 1 \quad \forall\, u = stay$$

$$g(x_k, u_k, w_k) = \alpha^{-1} \tilde{g}(x_k, u_k, w_k)$$

Following the derivation the BE looks as follows:

$$J^*(i) = \min_{u \in \mathcal{U}} \left(q(i,u) + \alpha \sum_{j=1}^{n} \tilde{P}_{ij}(u) J^*(j) \right) \quad \forall\, i \in \mathcal{S}^+$$

$$q(i,u) = \sum_{j=1}^{n} P_{ij}(u) g(i,u,j) = \sum_{j=1}^{n} \tilde{P}_{ij}(u) \tilde{g}(i,u,j)$$

As above the system can be written as $J = G + \alpha \tilde{P} J$. If $(I - \alpha \tilde{P})$ is invertible, then the solution exists. It can be shown that this is the case. There is a mapping between the original discounted problem and the auxiliary problem.
PI and VI work the same but with the α in front of the sum.

5 Deterministic Systems

5.1 Deterministic Finite State (DFS) Problem

Dynamics

Same setup as normal DP system but without a disturbance:

$$x_{k+1} = f_k(x_k, u_k)$$

$$\text{with} \quad x_k \in \mathcal{S}_k, u_k \in \mathcal{U}_k$$

Policy

controll inputs u_k are generated by an admissible policy $\pi \in \Pi$ such that.

$$u_k = \mu_k(x_k) \quad \forall\, k \in \{0, 1, \dots, N-1\}$$

Cost

$$J_N(x_N) = g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$$

with $g_k(x_k, u_k)$ stage cost, $g_N(x_N)$ terminal cost

and $X_1 := \{x_1, \dots, x_N\}$

Objective (since deterministic no feedback needed)

$$\pi^* = \arg\min_{\pi \in \Pi} J_\pi(x_N)$$

5.2 Shortest Path Problem (SP)

Graph is defined by a finite vertex space $\mathcal{V}$ (all vertices including start and end) and a weighted edge space:

$$\mathcal{C} := \{(i, j, c_{i,j}) \in \mathcal{V} \times \mathcal{V} \times \mathbb{R} \cup \{\infty\} | i, j \in \mathcal{V}\}$$

$c_{i,j}$ is the cost of the edge from i to j, if no connection exists $c_{i,j} = \infty$.
Path is a ordered list of nodes $Q := (i_1, i_2, \dots, i_q)$. The set of all paths from some node $s \in \mathcal{V}$ to some node $t \in \mathcal{V}$ is denoted by $\mathbb{Q}_{s,t}$.
Path Length is the sum of the arc lenghts/costs:

$$J_Q := \sum_{k=1}^{q-1} c_{i_k, i_{k+1}}$$

Objective find $Q^* \in \mathbb{Q}_{s,t}$ that has the smallest length:

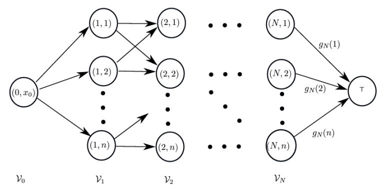
$$Q^* = \arg\min_{Q \in \mathbb{Q}_{s,t}} J_Q$$

Assumption 7.1:

For the Problem to make sense it must hold:
For all $i \in \mathcal{V}$ and for all $Q \in \mathbb{Q}_{i,i}$, $J_Q \geq 0$.

5.3 Equivalence of SP and DFS

5.3.1 DFS to SP



We have:

$$\mathcal{V} := \left(\bigcup_{k=1}^{N} \mathcal{V}_k \right) \cup \{\top\}, \quad \mathcal{V}_k := \{(k,i) | i \in \mathcal{S}_k\}$$

the cost is given by

$$c_{(k,i),(k+1,j)} = \min_{u_k} g_k(i, u_k, j), \quad \forall\, i \in \mathcal{S}_k, \forall\, j \in \mathcal{S}_{k+1}$$

5.3.2 SP to DFS

The optimal path needs at most $|\mathcal{V}|$ steps, thus we can formulate the SP as a DFS of length $N := |\mathcal{V}| - 1$:
- State space: $\mathcal{S}_k = \mathcal{V} \setminus \{t\}$ for $k = 1, \dots, N-1$ and $\mathcal{S}_N = \{t\}$ and $\mathcal{S}_0 = \{s\}$
- Control space: $\mathcal{U}_k = \mathcal{V} \setminus \{t\}$ for $k = 0, \dots, N-2$ and $\mathcal{U}_{N-1} = \{t\}$
- Dynamics: $x_{k+1} = u_k, \quad u_k \in \mathcal{U}_k, \quad k = 0, \dots, N-1$
- Stage Cost: $g_k(x_k, u_k) = c_{x_k, u_k}, \quad k = 0, \dots, N-1, \; g_N(t) = 0$

The solution can be found using DPA:

$$J_N(t) = 0$$

$$J_{N-1}(i) = c_{i,t}, \quad \forall\, i \in \mathcal{V} \setminus \{t\}$$

$$J_k(i) = \min_{j \in \mathcal{V} \setminus \{t\}} \left[c_{i,j} + J_{k+1}(j) \right] \quad k = N-2, \dots, 0$$

We can terminate if $J_k(i) = J_{k+1}(i)$ for all $i \in \mathcal{V} \setminus \{t\}$.

Forward DP Algorithm

The SP is symmetric, so we can reverse the problem and J can be interpreted as the cost to go. This also means the path can be build sequentially.

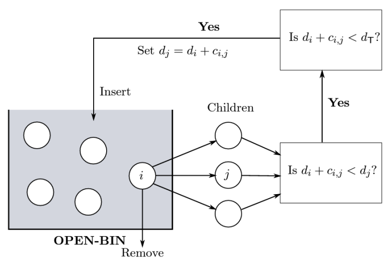
5.4 Shortest Path Algorithms

5.4.1 Lable Correcting Algorithm

Can only be used if assume $c_{i,j} \geq 0$

The steps of a general LCA are:

0. Place node s in OPEN, set $d_s = 0$ and $d_j = \infty, \forall\, j \in \mathcal{V} \setminus \{s\}$
1. Remove a node i from open and execute step 2 for all children j of i (for all nodes $j \in \mathcal{V}$ with $c_{i,j} < \infty$)
2. If $(d_i + c_{i,j}) < d_j$ and $(d_i + c_{i,j}) < d_t$ then set $d_j = d_i + c_{i,j}$ and set i as parent of j. If $j \neq t$ then place j in OPEN.
3. If OPEN is empty then stop, otherwise go to step 1.



Theorem 8.1

If at least one finite cost path from s to t exists, then the LCA terminates with $d_t = J_{Q^*}$. Otherwise the LCA terminates with $d_t = \infty$.

Different Methods

The LCA Algorithms only differ from how to remove the nodes from OPEN:
- **Depth-First Search:** "last in, first out", a node is removed from the top of OPEN and new nodes are placed on top.
- **Breadth-First Search:** "first in, first out" a node is removed from the bottom of OPEN and new nodes are placed on top.
- **Best-First Search (Dijkstra's Algorithm):** "priority queue" a node is removed from OPEN with the smallest d_i and new nodes are placed in OPEN according to their d_j.

A* Algorithm

Adding a heuristic function $h(i)$ which is an estimate/lower bound of the cost from node i to t. The new cost function thus becomes:

$$d_j = d_i + c_{i,j} + h(j) < d_t$$

Only used to check if added to the OPEN list, not used for the actual cost.

Nice to know

Could also work with negative costs if there are no negative cycles (Assumption 7.1). But then the terminal cost check must be omitted.

5.5 Hidden Markov Models

Dynamics

$$x_{k+1} = w_k, \quad x_k \in \mathcal{S}, \quad P_{ij} := p_{w_k = j | x_k = i}(j|i), \quad i, j \in \mathcal{S}$$

x_0 is not known but its distribution p_{x_0} is known.
Measurement Model When the state transition occur the states before and after are unkwnon to us, but we obtain an observation that relates the two states.

$$M_{ij}(z) = p_{z|w,w}(z|i,j), \quad \forall\, z \in \mathcal{Z}$$

$p_{z|w,w}(z|i,j)$ is time-invariant and known to us (likelihood function).
Objective Given a measurement sequence $Z_1 = (z_1, \dots, z_N)$, find the most likely sequence of states $X_0 = (x_0, \dots, x_N)$ that generated the measurements:

$$\hat{X}_0 = \arg\max_{X_0} p_{X|Z}(X_0 | Z_1)$$

Using the conditioning rule and the fact that $p(Z_1)$ is fixed and non-negative, maximization of $p_{X|Z}(X_0|Z_1)$ is equivalent to maximization of $p(X_0, Z_1)$. This can be rewritten as:

$$p(X_0, Z_1) = p(x_0) \Pi_{k=1}^{N} P_{x_{k-1} x_k} M_{x_{k-1} x_k}(z_k)$$

This results in the minimazation of the negative log-likelihood:

$$\min_{X_0} \left(c_{s,(0,x_0)} + \sum_{k=1}^{N} c_{(k_1, x_{k-1}),(k, x_k)} \right)$$

where:

$$c_{\mathsf{S},(0,x_0)} = \begin{cases} -\ln(p(x_0)) & \text{if } p(x_0) > 0 \\ \infty & \text{if } p(x_0) = 0 \end{cases}$$

$$c_{(k-1,x_{k-1}),(k,x_k)} = \begin{cases} -\ln(P_{x_{k-1}x_k} M_{x_{k-1}x_k}(z_k)) & \text{if } P_{x_{k-1}x_k} M_{x_{k-1}x_k}(z_k) > 0 \\ \infty & \text{if } P_{x_{k-1}x_k} M_{x_{k-1}x_k}(z_k) = 0 \end{cases}$$

This is now a SP problem which can be solved using DPA.

6 Deterministic Continuous Optimal Control

Dynamics

$$\dot{x}(t) = f(x(t), u(t)), \; x(t) \in \mathcal{S} := \mathbb{R}^n, \; u(t) \in \mathcal{U} \subseteq \mathbb{R}^m, \; t \in [0, T]$$

Feedback Control Law

$$u(t) = \mu(t, x(t)), \quad \mu(x,t) \in \mathcal{U}, \quad \forall\, t \in [0,T], \forall\, x \in \mathcal{S}$$

Cost

$$J_\mu(t, x) := h(x(T)) + \int_t^T g(x(\tau), u(\tau)) d\tau$$

Objective Construct an optimal feedback control law $\mu^*(t,x)$ such that

$$J_{\mu^*}(0, x) \leq J_\mu(0, x), \quad \forall\, \mu \in \Pi, \; \forall\, x \in \mathcal{S}$$

Assumption 9.1

For any admissible control law μ, initial time $t \in [0,T]$ and intial condition $x(t) \in \mathcal{S}$, there exists a unique trajectory $x(\tau)$ that safitsfies:

$$\dot{x}(\tau) = f(x(\tau), \mu(\tau)), \quad t \leq \tau \leq T$$

Assumption 9.1 is needed for the problem to be well defined.

6.1 Hamilton-Jacobi-Bellman (HJB) Equation

HJB is a **sufficient condition** for optimality, i.e. if a trajectory satisfies HJB it is optimal. If candiadte is differentiable $\rightarrow$ must satisfy HJB for optimality
As a result we get: $(\forall\, t \in [0,T], \; \forall\, x \in \mathcal{S})$

$$0 = \min_{u \in \mathcal{U}} \left\{ g(x,u) + \frac{\partial J^*(t,x)}{\partial t} + \frac{\partial J^*(t,x)}{\partial x} f(x,u) \right\}$$

Theorem 9.1

Suppose $V(t,x)$ is continuously differentiable in t and x and solves the HJB:

$$0 = \min_{u \in \mathcal{U}} \left\{ g(x,u) + \frac{\partial V(t,x)}{\partial t} + \frac{\partial V(t,x)}{\partial x} f(x,u) \right\}$$

$$\text{s.t.} V(T,x) = h(x), \quad \forall\, x \in \mathcal{S}$$

If Assumption 9.1 holds, then $V(t,x)$ is equal to the optimal cost-to-go function:

$$V(t,x) = J^*(t,x), \quad \forall\, t \in [0,T], \; \forall\, x \in \mathcal{S}$$

The mapping $\mu^*(t,x)$ minimizing the HJB is an optimal feedback law.

6.2 Pontryagin's Minimum Principle

This is a **nessecary condition** for optimality, i.e. if a trajectory is optimal it satisfies the PMH.

Setup

Dynamics, Control Law and Cost are the same as for DCOC

Objective

Given an initial condition $x(0) = x \in \mathcal{S}$, find an optimal control trajectory $u^*(t)$ such that the Cost is minimized.

Theorem 10.1

For a given initial condition $x(0) = x \in \mathcal{S}$, let $u(t)$ be an optimal control trajectory with associated state trajectory $x(t)$ for the system. Then there exists a trajectory p(t) such that:

$$\dot{p}(t) = -\left. \frac{\partial H(x,u,p)}{\partial x} \right|_{\substack{x(t) \\ u(t) \\ p(t)}}^{\top}, \quad p(T) = \left. \frac{\partial h(x)}{\partial x} \right|_{x(T)}^{\top}$$

$$u(t) = \arg\min_{u \in \mathcal{U}} H(x(t), u, p(t))$$

$$H(x(t), u(t), p(t)) = constant \quad \forall\, t \in [0,T]$$

where $H(x,u,p) := g(x,u) + p^\top f(x,u)$ is the Hamiltonian.

6.2.1 Fixed Terminal State

Remove boundary condition on $p(T)$.

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0, \quad x(T) = x_T$$

If only a subset of the states is fixed i.e. $x_i(T) = x_{T,i}, \; \forall\, i \in \mathcal{I}$ we get the partial boundary conditions:

$$p_j(T) = \left. \frac{\partial h(x)}{\partial x_j} \right|_{x(T)}^{\top}, \quad \forall\, j \notin \mathcal{I}$$

6.2.2 Free Initial State

If the initial state is also free and we add a cost term $l(x(0))$ we get:

$$p(T) = \left. \frac{\partial h(x)}{\partial x} \right|_{x(T)}^{\top}, \quad p(0) = -\left. \frac{\partial l(x)}{\partial x} \right|_{x(0)}^{\top}$$

If only some parts of the initial state are free we can proceed similar to the fixed terminal state case.

6.2.3 Free Terminal Time

If the terminal time T is also subject to optimization we get:

$$H(x(t), u(t), p(t)) = 0, \quad \forall\, t \in [0,T]$$

6.2.4 Time Varying Systems

Dynamics: $\dot{x}(t) = f(x(t), u(t), t)$
Cost: $J(u) = h(x(T)) + \int_0^T g(x(\tau), u(\tau), \tau) d\tau$
Convert the system to a time invariant system by introducing a new state $y(t)$ representing time:

$$\dot{y}(t) = 1, \quad y(0) = 0 \Rightarrow y(t) = t$$

The augmented system $z(t) = (x(t), y(t))$ is now time invariant. When applying the conditions with an augmented $\bar{H}(z, u, \bar{p}) = H(x,u,p,y) + q$ we get:

$$\dot{p}(t) = -\left. \frac{\partial H(x,u,p,t)}{\partial x} \right|_{\substack{x(t) \\ u(t) \\ p(t)}}^{\top}, \quad p(T) = \left. \frac{\partial h(x)}{\partial x} \right|_{x(T)}^{\top}$$

$$u(t) = \arg\min_{u \in \mathcal{U}} H(x(t), u, p(t), t)$$

i.e. the Hamiltonian must not be constant along a trajectory.

6.2.5 Singular Problems

- If the Hamiltonian is linear in u the optimal control is bang-bang.

Sometimes the condition $u(t) = \arg\min_{u \in \mathcal{U}} H(x(t), u, p(t))$ is insufficient to determine $u(t)$, if the values of $x(t)$ and $p(t)$ are such that $H(x(t), u, p(t))$ is independent of u over a nontrivial interval of time. This results in a *singular* problem, where the solution consists over *regular arcs* where u can be determined using the Hamiltonian and *singular arcs* which can be determined from the condition that the Hamiltonian is independent of u.

7 Usefull stuff

$$(a + b + c)^2 = a^2 + b^2 + c^2 + 2ab + 2ac + 2bc$$