Dynamic Programming & Optimal Control

Lecture 1
Introduction to Dynamic Programming

Fall 2023

Prof. Raffaello D'Andrea

ETH Zurich

Learning Objectives

Topic: Introduction to Dynamic Programming

Objectives

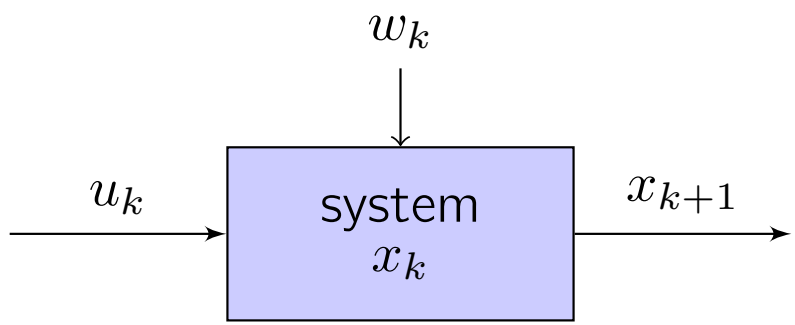
- You can explain the *problem of dynamic programming*.
- You know the *key ingredients* of a dynamic programming problem.
- You know the difference between *open loop (OL)* and *closed loop (CL)* control.
- You can express the dynamics of a *discrete, finite state system* in terms of *transition probabilities*.

Outline

Problem Statement

Given a model of how a dynamic system evolves and a direct measurement of its state, apply a control input to the system so that a given cost is minimized.

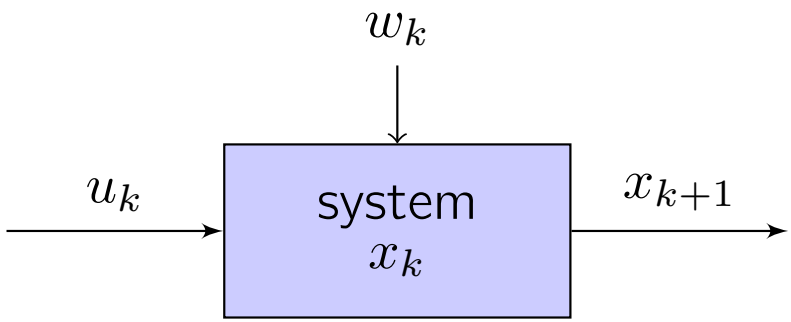
Dynamics



$$x_{k+1} = f_k\left(x_k, u_k, w_k\right), \quad k = 0, 1, ..., N-1$$

Cost Function

$$\underbrace{g_N(x_N)}_{\text{terminal cost}} + \underbrace{\sum_{k=0}^{N-1} \underbrace{g_k(x_k, u_k, w_k)}_{\text{stage cost}}}_{\text{accumulated cost}}$$



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

$$x_{k+1} = f_k\left(x_k, u_k, w_k\right),\ k = 0, 1, ..., N-1$$

- k: discrete time index
- N: time horizon
- $x_k \in \mathcal{S}_k$: system state vector at time k, can be measured at time k
- $u_k \in \mathcal{U}_k(x_k)$: control input vector at time k
- w_k: disturbance vector at time k. Assumption: w_k is conditionally independent with all prior variables $x_l,\ u_l,\ w_l$, $l < k$, given x_k and u_k
- $f_k\left(\cdot, \cdot, \cdot\right)$: function capturing the system evolution at time k

Example: Inventory Control

We are keeping an item stocked in a warehouse. If there is too little, we will run out of it and lose sales (not preferred). If there is too much, there will be more cost of storage and misuse of capital (not preferred).

We will model this scenario as a discrete time system:

- $x_k \in \mathcal{S}_k = \mathbb{R}$: stock available in the warehouse at the beginning of the k^{th} time period
- $u_k \in \mathcal{U}_k(x_k) = \mathbb{R}_{\geq 0}$: stock ordered and immediately delivered at the beginning of the k^{th} time period (supply)
- w_k: demand during the k^{th} time period, with some given probability distribution
- *Dynamics*: $x_{k+1} = f_k(x_k, u_k, w_k) = x_k + u_k - w_k$. We assume that excess demand is back-logged, which corresponds to negative x_k

Example: Inventory Control

We are keeping an item stocked in a warehouse. If there is too little, we will run out of it and lose sales (not preferred). If there is too much, there will be more cost of storage and misuse of capital (not preferred).

- *Cost function*:

$$R(x_N) + \sum_{k=0}^{N-1} r(x_k) + cu_k - pw_k$$

where

pw_k: revenue;

cu_k: cost of items;

$r(x_k)$: cost associated with too much stock or negative stock;

$R(x_N)$: terminal cost; cost associated with stock left at the end which we can't sell, or demand we can't meet;

Expected Cost (1/2)

Let

$$X_1 := (x_1, \ldots, x_N)$$
$$U_0 := (u_0, \ldots, u_{N-1})$$
$$W_0 := (w_0, \ldots, w_{N-1})$$

Given x_0, the variables X_1, U_0 and W_0 are all random variables due to the disturbances w_k and the dynamic coupling

$$x_{k+1} = f_k\left(x_k, u_k, w_k\right).$$

The control inputs can either be fixed and thus deterministic, or a function of the state and thus random variables.

The cost function is also a random variable

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

Expected Cost (2/2)

A convenient metric for optimization is the expected value of the cost starting at an initial state x_0:

$$\underset{(X_1, U_0, W_0 | x_0)}{\mathrm{E}} \left[g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right]$$

subject to the dynamics

$$x_{k+1} = f_k\left(x_k, u_k, w_k\right).$$

Outline

Open Loop and Closed Loop Control

There are two different control methodologies:

- **open loop**: all the control inputs are determined at once at time 0.

- **closed loop**: the control inputs are determined in a "just-in-time fashion", depending on the measured state x_k at time k.

Open Loop Control

Given a set of control inputs $\bar{U}_0 := (\bar{u}_0, \ldots, \bar{u}_{N-1})$ that is fixed, and an initial state x_0, the cost becomes

$$g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \bar{u}_k, w_k)\,,$$

and is thus a function of the random variables x_k and w_k.

The open loop control problem is thus the following: at time $k = 0$, given x_0, find $\bar{U}_0$ that minimizes the **expected open loop cost**

$$\mathop{\mathrm{E}}_{(X_1, W_0 | x_0)} \left[g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \bar{u}_k, w_k) \right]$$

subject to the dynamics

$$x_{k+1} = f_k\left(x_k, \bar{u}_k, w_k\right).$$

This is called **open loop** control because measurements of the state are not used to calculate the control inputs.

Closed Loop Control (1/3)

Let $\mu_k(\cdot)$ map the state x_k to the control input u_k:

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

and define

$$\pi := (\mu_0(\cdot)\,, \mu_1(\cdot)\,, \ldots, \mu_{N-1}(\cdot))$$

where π is called an **admissible policy**.

Given an initial state x_0, the states $x_1, \ldots, x_N$, the control inputs $u_1, \ldots u_{N-1}$ and the disturbances $w_0, \ldots, w_{N-1}$ are random variables with probability density functions (PDF) defined through the system dynamics

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

and state feedback equations

$$u_k = \mu_k(x_k)\,.$$

Closed Loop Control (2/3)

The cost therefore becomes

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

a function of the random variables $x_1, \ldots, x_N$ and $w_0, \ldots, w_{N-1}$.

We define, for any $\mathrm{x} \in \mathcal{S}_0$, the **expected closed loop cost** associated with an admissible policy π to be

$$J_\pi(\mathrm{x}) := \mathop{\mathrm{E}}_{(X_1, W_0 | x_0 = \mathrm{x})} \left[g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k)\,, w_k) \right]$$

subject to the dynamics

$$x_{k+1} = f_k\left(x_k, \mu_k(x_k)\,, w_k\right).$$

Closed Loop Control (3/3)

Let Π denote the set of all admissible policies. Then π^* is called an **optimal policy** if

$$J_{\pi^*}(\mathrm{x}) \leq J_\pi(\mathrm{x}) \qquad \forall \pi \in \Pi, \forall \mathrm{x} \in \mathcal{S}_0.$$

The **optimal cost** is defined as $J^*(\mathrm{x}) := J_{\pi^*}(\mathrm{x})$.

$J^*(\cdot)$ is a function that maps initial states to optimal costs. The closed loop control problem aims at finding π^*.

Example: Inventory Control

We are keeping an item stocked in a warehouse. If there is too little, we will run out of it and lose sales (not preferred). If there is too much, there will be more cost of storage and misuse of capital (not preferred).

Recall:

- x_k : stock available in the warehouse at the beginning of the k^{th} time period
- u_k : stock ordered and immediately delivered at the beginning of the k^{th} time period (supply)

An intuitive example of an admissible policy is

$$\mu_k(x_k) = \begin{cases} s_k - x_k, & \text{if } x_k < s_k \\ 0, & \text{otherwise} \end{cases} \qquad k = 0, 1, \ldots, N-1$$

where s_k is some predefined, potentially time-varying threshold.

Performance

Open loop control can never give better performance than closed loop control since open loop control is a special case of closed loop control.

In the absence of disturbances w_k, the two give **theoretically** the same performance.

In practice, even without disturbances, closed loop control will give better performance than open loop control:

- x_0 is often not known precisely and may also be random.
- the dynamics $f_k(\cdot, \cdot, \cdot)$ are often not known precisely.

However, when a system is well-behaved and a good model for it exists, open loop control is a viable strategy for short time horizons.

Computation

It is typically much less demanding to find an open loop control strategy than a closed loop control one.

Consider a system with N_x distinct states and N_u distinct control inputs at every k^{th} time period:

- There are a total of N_u^N different open loop strategies.
- There are a total of $N_u(N_u^{N_x})^{N-1} = N_u^{N_x(N-1)+1}$ different closed loop strategies.

There are thus many more closed loop strategies than open loop ones.

As an example, let's take $N_u = 10$, $N_x = 10$, and $N = 4$. The number of open loop strategies is 10^4. The number of closed loop strategies is 10^{31}, which is almost 10 orders of magnitude larger than the number of stars in the observable universe!

Outline

Discrete State and Finite State Problems (1/2)

When the state x_k takes on discrete values, or it is finite in size, it may be more convenient to express the dynamics in terms of *transition probabilities*

$$\begin{aligned} P_{ij}(u,k) &:= \Pr\left(x_{k+1} = j \mid x_k = i, u_k = u\right) \\ &= p_{x_{k+1}|x_k,u_k}(j|i,u), \end{aligned}$$

where $p_{x_{k+1}|x_k,u_k}(\cdot \,|\, \cdot, \cdot)$ denotes the PDF of x_{k+1} given x_k and u_k.

Given the transition probabilities, a system can be described equivalently with the dynamics

$$x_{k+1} = w_k$$

where w_k has the following probability distribution:

$$p_{w_k|x_k,u_k}(j|i,u) = P_{ij}(u,k).$$

Discrete State and Finite State Problems (2/2)

Conversely, given a system with the dynamics $x_{k+1} = f_k(x_k, u_k, w_k)$ and $p_{w_k|x_k,u_k}(\cdot \,|\, \cdot, \cdot)$, we can provide an equivalent transition probability description, where the transition probabilities are

$$P_{ij}(u,k) = \sum_{\{\bar{w}_k \,|\, f_k(i,u,\bar{w}_k)=j\}} p_{w_k|x_k,u_k}(\bar{w}_k|i,u)$$

that is, $P_{ij}(u,k)$ is equal to the sum over the probabilities of all possible disturbances $\bar{w}_k$ that get us to state j from state i using control u at time k.

Example: Optimizing chess match strategy

Consider a two-game chess match with an opponent. Our objective is to come up with a strategy that maximizes the chance of winning the match.

Each game can have one of two outcomes:

1. Win/Lose: 1 point for the winner, 0 for the loser
2. Tie: 0.5 points for each player

If at the end of two games the score is equal, the players keep on playing new games until one wins, and thereby wins the match (sudden death).

There are two possible playing styles for our player: timid and bold.

- Timid: our player ties with probability p_d and loses with probability $(1 - p_d)$
- Bold: our player wins with probability p_w and loses with probability $(1 - p_w)$

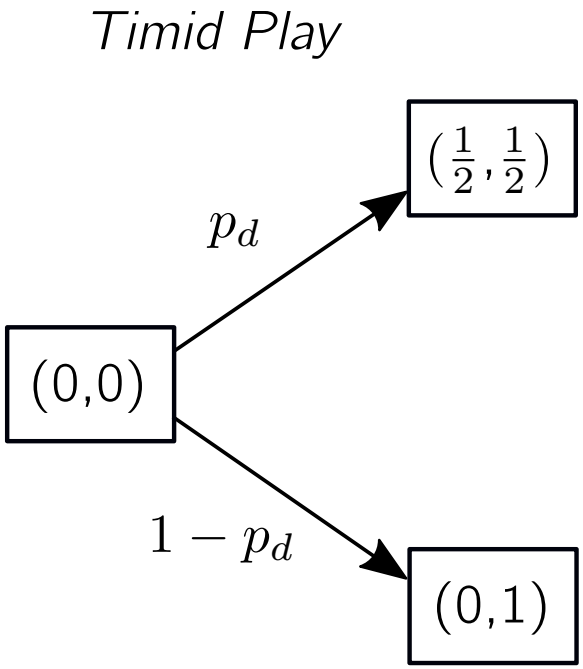
Assume that $p_d > p_w$.

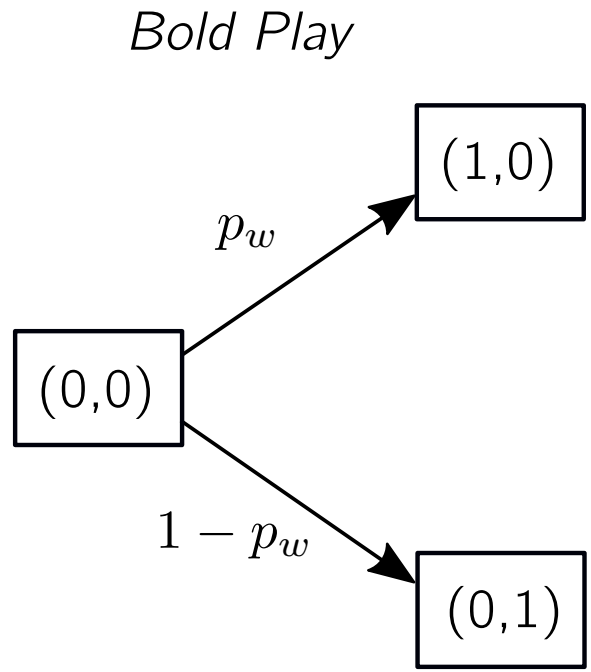
We will model this as a finite state problem:

- *state* $x_k =$ (score of our player, score of opponent)
- *control input* $u_k =$ timid or bold
- *dynamics:* we can construct a *Transition Probability Graph*, which can then be used to deduce $P_{ij}(u, k)$ to express the dynamics
- *objective:* maximize the probability of winning the match, P_{win} (this is equivalent to minimizing the cost $-P_{\text{win}}$)

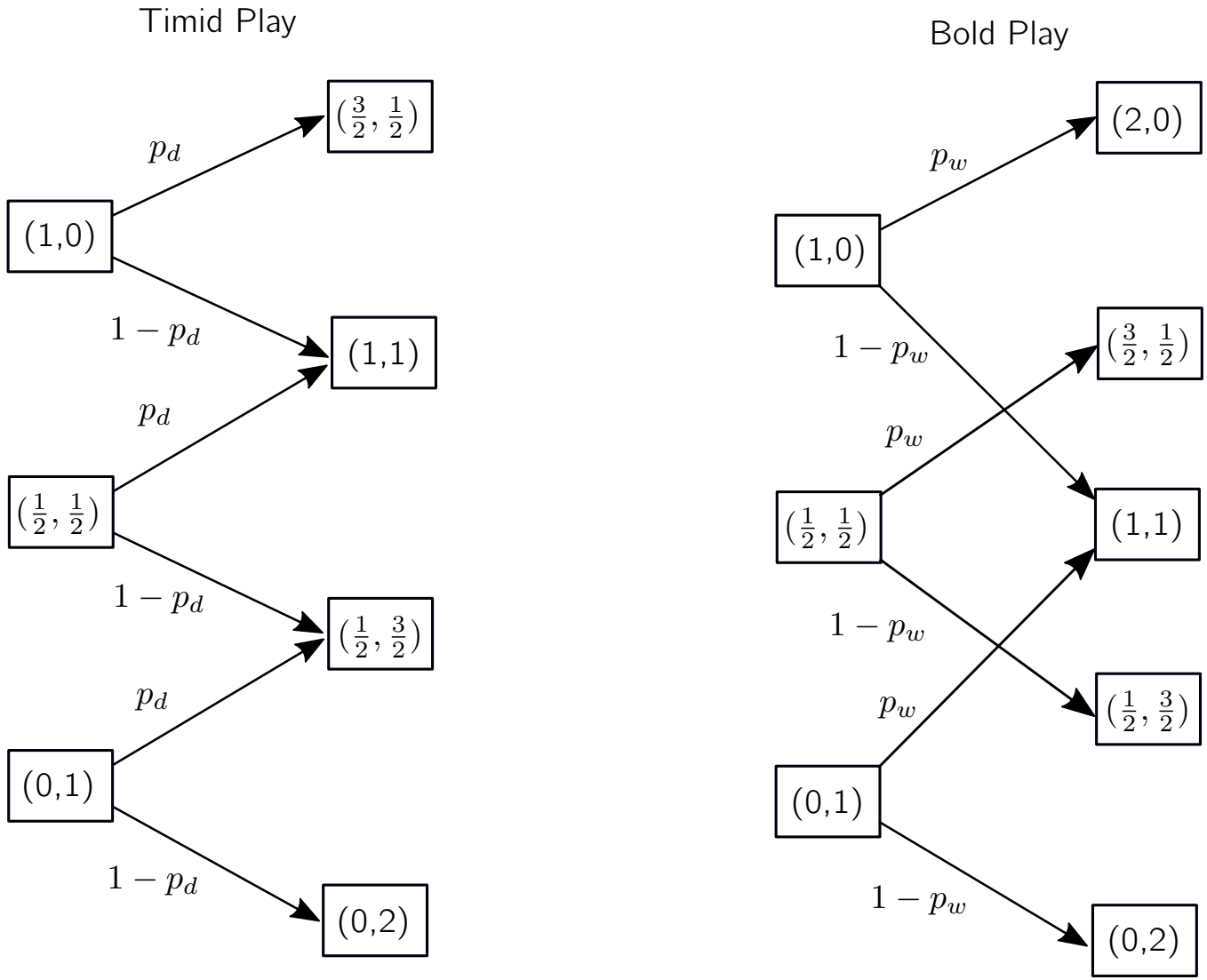
Since it doesn't make sense to play timid if the game goes into sudden death (probability of winning is zero), the problem can be modelled as a two-stage finite state problem.

Transition Probability Graph: First Game





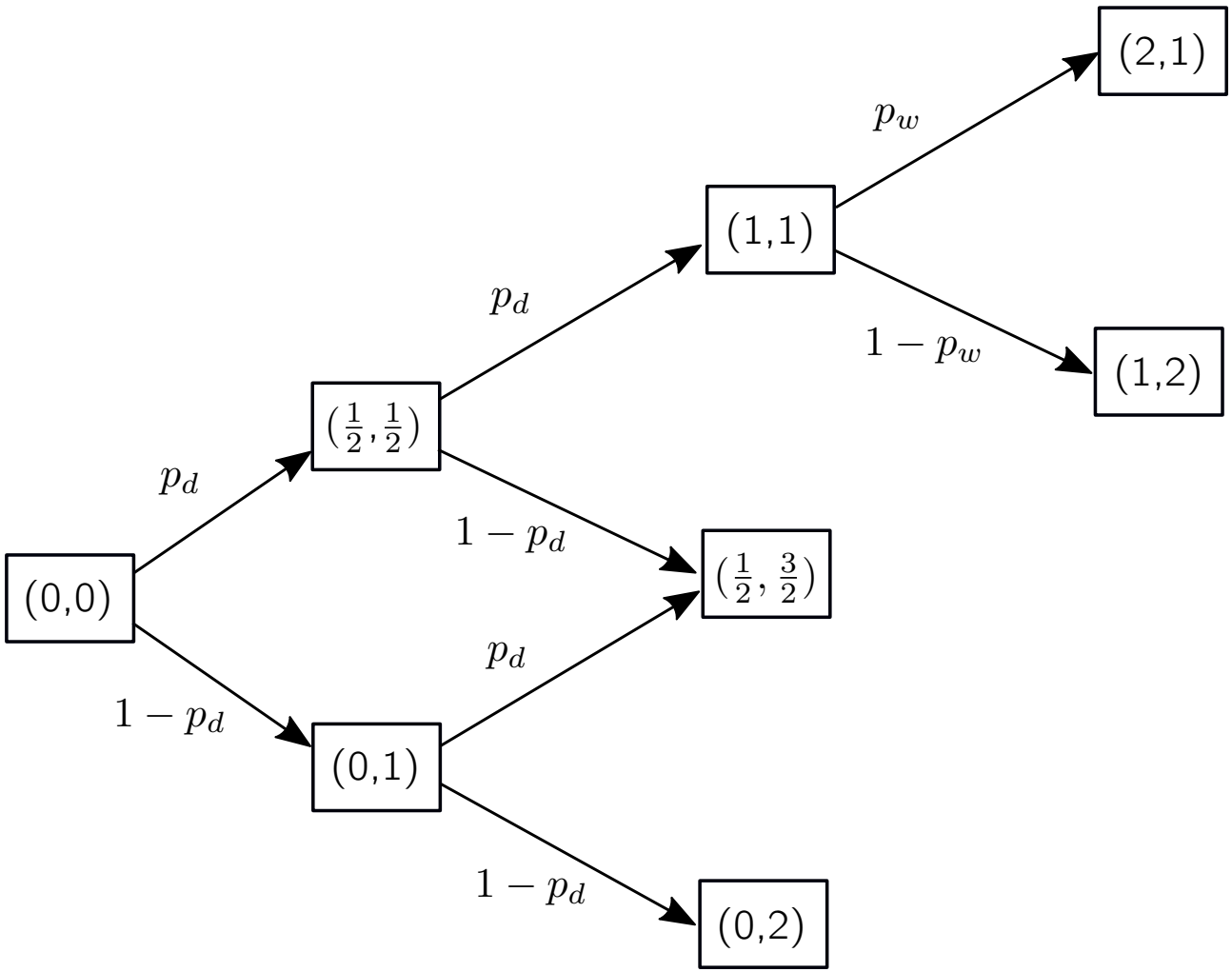
Transition Probability Graph: Second Game



Example: Open Loop Strategy (1/6)

There are 4 possibilities:

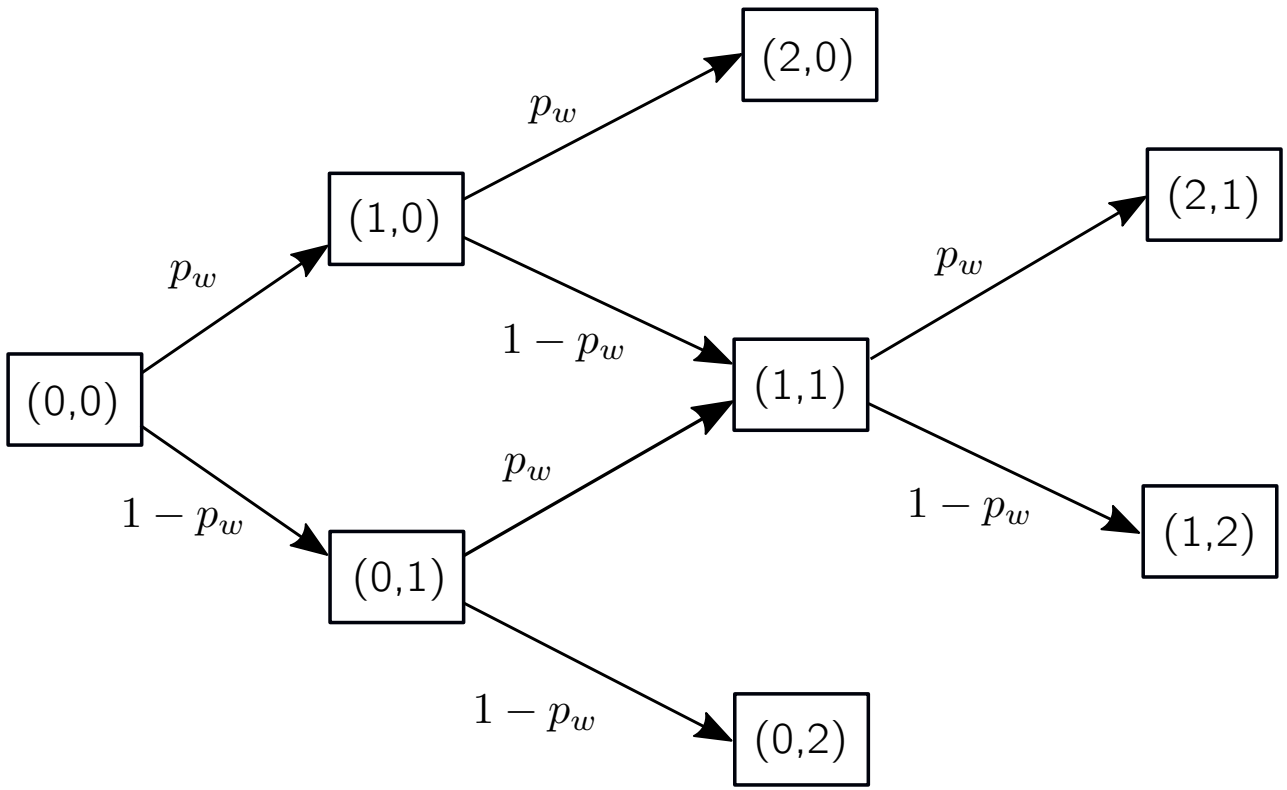
1) Play timid in both games: $P_{\text{win}} = p_d^2 p_w$



Example: Open Loop Strategy (2/6)

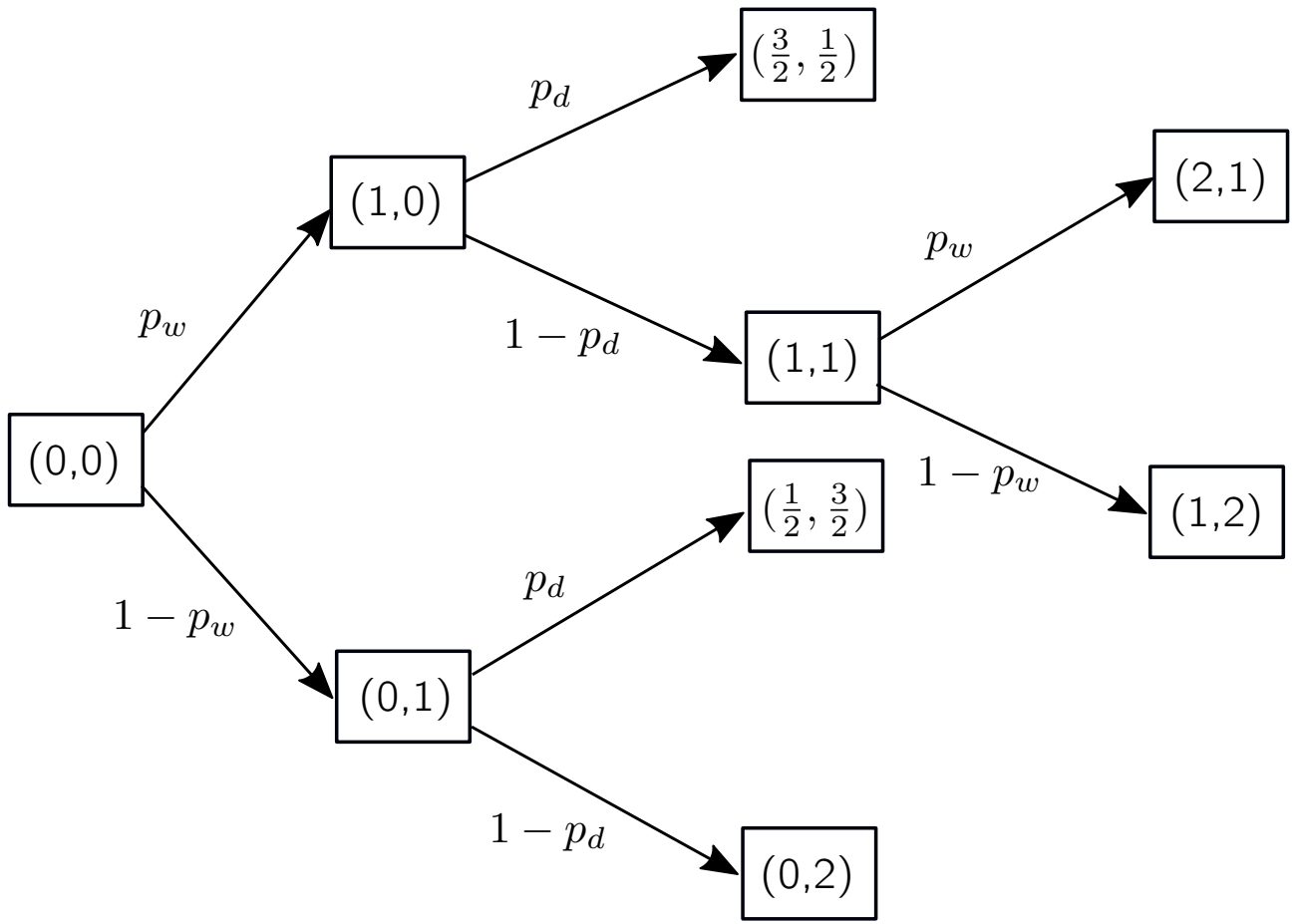
2) Play bold in both games:

$$P_{\text{win}} = p_w^2 + p_w(1-p_w)p_w + (1-p_w)p_w p_w = p_w^2(3-2p_w)$$



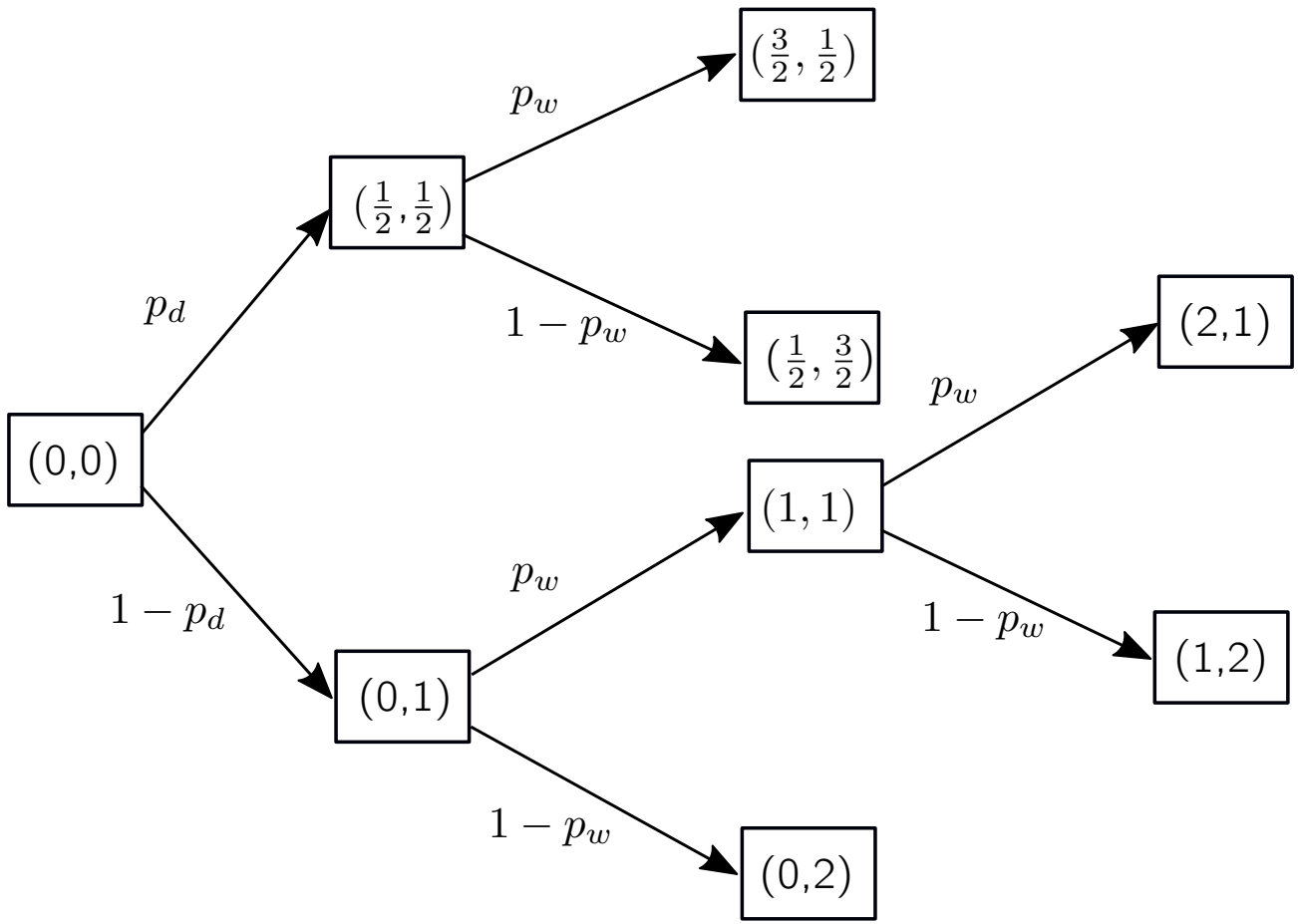
Example: Open Loop Strategy (3/6)

3) Play bold first, and timid in the second game: $P_{\text{win}} = p_w p_d + p_w(1-p_d)p_w$



Example: Open Loop Strategy (4/6)

4) Play timid first, and bold in the second game: $P_{\text{win}} = p_d p_w + (1 - p_d) p_w^2$



Example: Open Loop Strategy (5/6)

There are 4 possibilities:

1) Play timid in both games: $P_{\text{win}} = p_d^2 p_w$
2) Play bold in both games: $P_{\text{win}} = p_w^2(3 - 2p_w)$
3) Play bold first, and timid in the second game: $P_{\text{win}} = p_w p_d + p_w(1 - p_d)p_w$
4) Play timid first, and bold in the second game: $P_{\text{win}} = p_d p_w + (1 - p_d)p_w^2$

Since $p_d^2 p_w \leq p_d p_w \leq p_d p_w + (1 - p_d)p_w^2$, 1) is not the optimal open loop strategy. The best achievable winning probability P^*_{win} is:

$$\begin{aligned} P^*_{\text{win}} &= \max\{\overbrace{p_w^2(3 - 2p_w)}^{\text{2)}}, \overbrace{p_d p_w + (1 - p_d)p_w^2}^{\text{3) or 4)}}\} \\ &= p_w^2 + \max\{2(p_w^2 - p_w^3),\ p_d p_w - p_d p_w^2\} \\ &= p_w^2 + \max\{2p_w(1 - p_w)p_w,\ p_w(1 - p_w)p_d\} \\ &= p_w^2 + p_w(1 - p_w)\max\{2p_w,\ p_d\} \end{aligned}$$

Example: Open Loop Strategy (6/6)

$$P^*_{\text{win}} = p_w^2 + p_w(1-p_w)\max\{\overset{2)}{2p_w}, \overset{3) \text{ or } 4)}{p_d}\}$$

If $p_d > 2p_w$, then 3) and 4) are the best open loop strategies, otherwise 2) is the best open loop strategy.

- For $p_w = 0.45$ and $p_d = 0.9$, $P^*_{\text{win}} = 0.43$.
- For $p_w = 0.5$ and $p_d = 1.0$, $P^*_{\text{win}} = 0.5$.

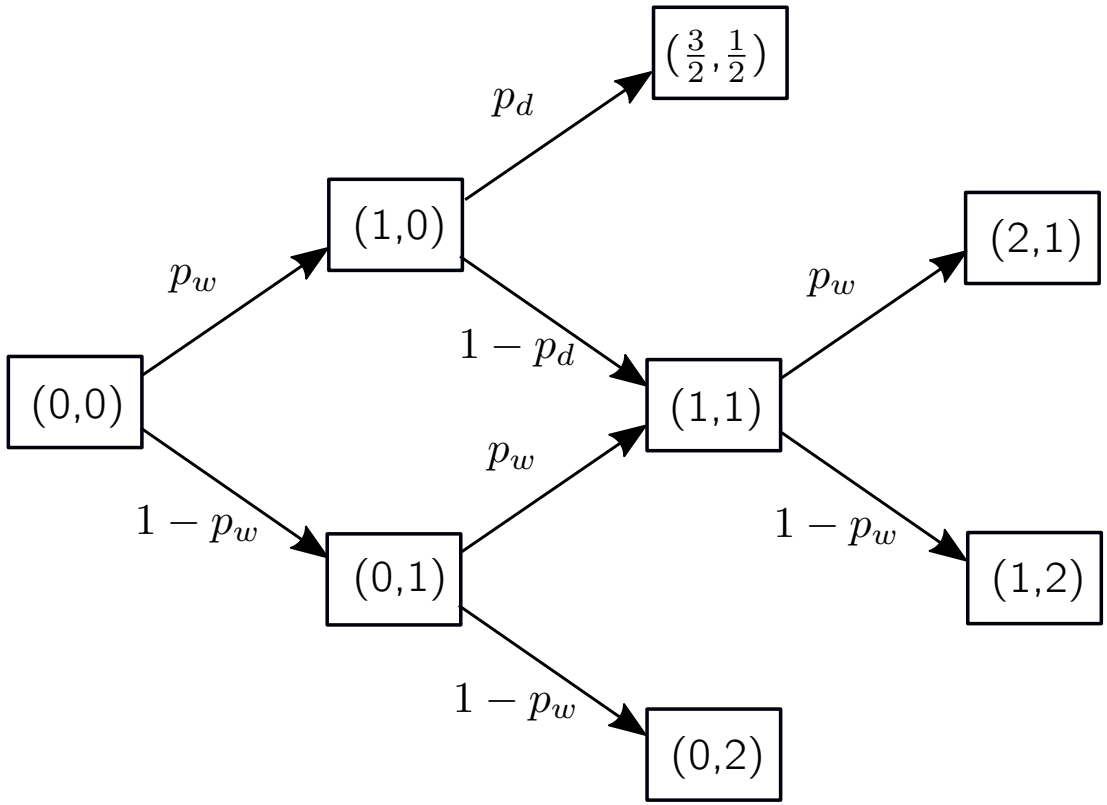
It can also be shown that, in the open loop case, if $p_w \leq 0.5$ then $P^*_{\text{win}} \leq 0.5$.

Example: Closed loop strategy (1/3)

There are 8 admissible policies.

Let's consider one possible policy: play timid if and only if the player is winning (in Lecture 3 we will show that this strategy is indeed the optimal policy).

Example: Closed loop strategy (2/3)



The associated probability of winning P_{win} is

$$p_d p_w + p_w((1-p_d)p_w + p_w(1-p_w))$$

Example: Closed loop strategy (3/3)

The associated probability of winning P_{win} is

$$p_d p_w + p_w((1-p_d)p_w + p_w(1-p_w))$$

- For $p_w = 0.45$ and $p_d = 0.9$, $P_{\text{win}} = 0.54$

- For $p_w = 0.5$ and $p_d = 1.0$, $P_{\text{win}} = 0.625$

Note that in the closed loop case we can achieve a winning probability larger than 0.5 even when p_w is less than 0.5.

Additional reading material

Dynamic Programming & Optimal Control plays an important role in practice, sometimes in unexpected ways:

- The Viterbi algorithm, with applications in communications, speech recognition, bioinformatics, and many more:
  `https://spectrum.ieee.org/2010-medal-of-honor-winner-andrew-j-viterbi`

- Sequence alignment, used to study functional and structural properties of biological sequences (DNA, RNA, and proteins):
  `https://en.wikipedia.org/wiki/Sequence_alignment`