

E' il principio di programmazione più efficiente che ci sia

Business Analytics – 2020/21

The dynamic programming principle

Prof. Paolo Brandimarte

Dip. di Scienze Matematiche – Politecnico di Torino

e-mail: paolo.brandimarte@polito.it

URL: <https://staff.polito.it/paolo.brandimarte>

This version: April 3, 2023

NOTE: For internal teaching use within the Masters' Program in Mathematical Engineering. Do not post or distribute.

References

These slides are taken from my book:

P. Brandimarte. *From Shortest Paths to Reinforcement Learning: A MATLAB-Based Introduction to Dynamic Programming*. Springer, 2021.

MATLAB code may be downloaded from my web page:

<https://staff.polito.it/paolo.brandimarte/>

Other useful references:

- D.P. Bertsekas. *Dynamic Programming and Optimal Control Vol. 1* (4th ed). Athena Scientific, 2017.
- D.P. Bertsekas. *Dynamic Programming and Optimal Control Vol. 2* (2nd ed). Athena Scientific, 2012.
- W.B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality* (2nd ed). Wiley, 2011.

What is dynamic programming?

Dynamic programming (DP) is not an algorithm like the simplex algorithm for linear programming. Rather, it is a remarkably general and flexible principle that may be used to design a range of optimization algorithms.

è un principio per generare algoritmi di ottimizzazione

The core idea hinges on the decomposition of a multistage, dynamic decision problem into a sequence of simpler, single-stage problems.

le decisioni si prendono in base alle info che arrivano in + momenti, cioè adattativo

→ In questo principio, i problemi si decompongono a grado singolo, andando a fare tagli rispetto al tempo e non rispetto agli scenari

A multistage decision problem should not be confused with a multiperiod decision problem. Consider a finite-horizon problem where we have to select T decisions x_t that will be applied at a sequence of time instants $t = 0, 1, 2, \dots, T - 1$. When are those decisions made?

si contrappone ad adattativo

ci sono molti istanti di tempo, ma le decisioni sono state prese TUTTE al tempo o con le info del tempo o

In a static, multiperiod problem, they are all made at time $t = 0$, which is natural in a deterministic problem (open-loop approach). However, in a stochastic problem, we observe over time a sample path of the relevant risk factors and adapt decisions along the way.

What we need is a strategy, i.e., a recipe to make decisions after observing random realizations (closed-loop).

→ Nel caso stocastico, si vuole il closed-loop

DP is not a universal approach, since its application requires some specific structure in the system model, which must be Markovian. → Deve esserci il concetto di «stato»

Nevertheless, DP can be applied to a wide range of problems:

- Continuous and discrete problems, where the discrete/continuous feature may refer to:
 - state variables,
 - decision variables,
 - the representation of time. → Guarderemo solo tempi discreti
- Deterministic and stochastic problems.
- Finite- and infinite-horizon problems.

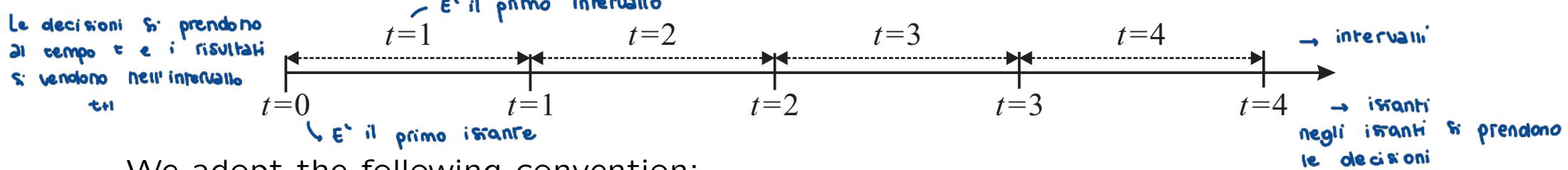
Within the resulting wide collection of problems, we may need to solve finite- as well as infinite-dimensional problems and, for some problem structures, DP is actually the only viable solution approach.

Dynamic decision problems

We will deal with discrete-time models, and we have to clarify how we deal with time instants and time intervals (also called time periods or time buckets).

→ Non ha senso a tempo continuo
Le decisioni si prendono prima di conoscere i fattori di rischio

We make decisions at time instants, but they might be implemented over a time interval, rather than instantaneously. Furthermore, we have to make decisions before observing the realization of risk factors, which may also take place over the subsequent time period.



We adopt the following convention:

- We consider time instants indexed by $t = 0, 1, 2, \dots$. At these time instants, we observe the system state and make a decision.
- By a time interval t , we mean the time elapsing between time instants $t-1$ and t . After applying the decision made at time instant $t-1$, the system evolves during the next time interval, and a new state is reached at time instant t . During this time interval, some external input, possibly a random "disturbance," will be realized, influencing the transition to the new state.

We may have a finite horizon problem, defined over time instants $t = 0, 1, \dots, T$, or an infinite horizon problem defined over time instants $t = 0, 1, \dots$. Note that the first time instant corresponds to $t = 0$, but the first time interval is indexed by $t = 1$. For a finite-horizon problem, we have T time periods and $T + 1$ time instants.

The system dynamics is represented by a state transition equation like

$$S_{t+1} = g_{t+1}(s_t, x_t, \xi_{t+1}), \quad \begin{array}{l} \text{Fattori di rischio} \\ \downarrow \\ \text{Variabile decisionale} \end{array} \quad \begin{array}{l} \rightarrow \text{Equazione di transizione di stato} \\ \text{Non e' detto che si conosca realmente} \\ \text{Stato all'istante } t+1, \text{ che e' fisico, ma puo' rappresentare anche informazioni} \\ \text{Assumiamo lo stato assolutamente osservabile (anche se nella realta' ogni tanto lo devi stimare)} \end{array} \quad (1)$$

where:

- g_{t+1} is the state transition function over time interval $t + 1$; the time subscript may be omitted if the dynamics do not change over time, which is the common assumption when dealing with infinite-horizon problems.
- s_t is a vector of state variables at time instant t . State variables summarize all the information we need from the past evolution of the system that we want to control. s_0 is the initial state, which is typically known, whereas s_T is the terminal state. \rightarrow Potrebbe essere un vettore, una matrice, un insieme (es. clienti da visitare o già visitati), altre strutture dati per sistemi di routing dinamico
- x_t is the vector of decision variables at time instant t , also called control variables. We will also use the term **action** when the set of possible decisions is finite; in such a case, we may use the notation a . In a finite horizon problem, control variables are defined for $t = 0, 1, \dots, T - 1$.
- The state at time instant $t + 1$ also depends on the realization of an external or exogenous factor during the time interval $t + 1$, which we denote as ξ_{t+1} . Exogenous factors may be associated with a time instant or a time interval.

Decisions x_t are made at time instant t , after observing the state s_t ; the next state s_{t+1} will depend on the realization of ξ_{t+1} .

We should also distinguish different kinds of state variables:

- **Physical** state variables are used to describe the state of literally physical, but also financial resources. They are directly influenced by our decisions. → *Se agisco in modo diretto su queste variabili*
 ↗ *«quanti soldi hai?»*
- **Informational** state variables are somewhat different. For instance, the price of a stock share is a relevant piece of information, but in liquid markets this is not influenced by the activity of a single investor. → *Su queste variabili non agisco io direttamente*
 ↗ *«Giò che io credo»*
Non è oggettivo
- **Belief** state variables are relevant in problems affected by uncertainty about uncertainty. In such a case, planning experiments to learn the parameters of the model is essential, and the model parameters themselves (or the parameters of a probability distribution), along with our uncertainty about them, become state variables.

We have a **data process** $(\xi_t)_{\{t \geq 1\}}$ or $(\xi_t)_{t \in \{1..T\}}$, a **decision process** $(x_t)_{\{t \geq 0\}}$ or $(x_t)_{t \in \{0..T-1\}}$, and a **state process** $(s_t)_{\{t \geq 0\}}$ or $(s_t)_{t \in \{0..T\}}$, which can be deterministic or stochastic.

When dealing with a dynamic decision problem under uncertainty, two crucial issues must be considered:

1. How to model information availability, i.e., what information can we actually rely on when making decisions. → Alcune cose le so misurare, altre no
2. How the elements in the data process are related to each other and, possibly, to the elements in the state and decision processes.

It is useful to introduce the following notation, which is used to represent the history observed so far at time instant t :

Magari voglio tenere traccia dello storico delle decisioni

$$\mathbf{x}_{[t]} \doteq (\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_t), \quad \boldsymbol{\xi}_{[t]} \doteq (\xi_1, \xi_2, \dots, \xi_t). \quad (2)$$

The decision \mathbf{x}_t may only depend on the sample path of risk factors up to time instant t (included), and the resulting random state trajectory. Foresight is ruled out, unless the problem is deterministic. → La politica deve essere non anticipativa perche' non vedi il futuro

More formally, an implementable decision policy is said to be non-anticipative.

Furthermore, the random vectors in the data process may feature different degrees of mutual dependence:

Questo dipende dal modello non dal sistema

- The easy case is when there is interstage independence, i.e., the elements of the data process are mutually independent. → indipendenza degli stati, quindi il processo marcativo non ha memoria
- The fairly easy case is when the process is Markovian, i.e., ξ_{t+1} depends only on the last realization ξ_t of the risk factors. → Modello Marcativo del primo ordine
- In the most general case the whole history up to time t is relevant, i.e., ξ_{t+1} depends on the full observed sample path $\xi_{[t]}$. → Caso opposto al primo, il processo dipende da tutta la storia passata

We should also specify whether the distribution of risk factors depends on time, i.e., whether the system is autonomous or not, and whether the distribution depends on states and decisions, i.e., whether the data process is purely exogenous, or at least partially endogenous. → influenzato parzialmente da me
non influenzato da me

The overall aim is to make decisions in such a way that an objective function is optimized over the planning horizon. DP lends itself to problems in which the objective function is additive over time, i.e., it is the sum of costs incurred (or rewards earned) at each time instant.

Finite horizon, discounted problems

→ E' scontata se e' presente il fattore γ :
 $0 < \gamma < 1$

A stochastic problem, with finite horizon T , might be stated as

*della funzione obiettivo
Decomposizione additiva risante per i risanze*

Problema ad orizzonte finito

L'obiettivo e' ottenere problemi
a studio singolo

$$\text{opt } E_0 \left[\sum_{t=0}^{T-1} \gamma^t f_t(s_t, x_t) + \gamma^T F_T(s_T) \right],$$

↑ Ultimo istante di tempo in cui si prende la decisione
 ↓ min, max t
 Valore atteso condizionato alle informazioni che ho ora
 ↗ Si prendono T decisioni
 ↘ reward finale che rappresenta lo stato finale

E' nascosta l'equazione di transizione di stato (3)

where "opt" may be either "min" or "max," depending on the specific application. The notation $E_0[\cdot]$ is used to point out that the expectation is taken at time $t = 0$; hence, it is an *unconditional expectation*, as we did not observe any realization of the risk factors yet (apart from those that led to the initial state s_0).

The objective function includes:

- The immediate contribution (cost or reward) $f_t(s_t, x_t)$, which we incur when we make decision x_t in state s_t , at time instants $t = 0, 1, \dots, T - 1$.
- The terminal cost/reward $F_T(s_T)$, which only depends on the terminal state s_T .

In discounted DP, we use a discount factor $\gamma \in (0, 1)$. For a finite-time problem, this is not strictly needed, since the objective is well-defined also when $\gamma = 1$ but it is common in financial applications.

→ A volte e' di natura oggettiva, altre volte e' di natura soggettiva
 → Se $\gamma = 1$, e' necessario scontare altrimenti la somma non converge
 → Se non scontiamo

Finding a good way to associate a value with the terminal state may require *ad hoc* reasoning.

It might be argued that the immediate contribution should involve the next realization of the risk factor, i.e., it should be of the form

$$h_t(s_t, x_t, \xi_{t+1}). \rightarrow \text{Il contributo immediato potrebbe anche essere (4) riscachico, cioè dipendente dal prossimo fattore di rischio}$$

Indeed, this may actually be the case, but in principle we may recover the form of Eq. (3) by taking a conditional expectation at time instant t :

M2. Non c'è detto che questo valore atteso sia calcolabile → 1. Computationalmente non semplice 2. Non hai la distribuzione di probabilità

$$f_t(s_t, x_t) = E_t[h_t(s_t, x_t, \xi_{t+1})]. \rightarrow \text{Valore atteso condizionato alle info che ho al tempo } t \rightarrow \text{Questo valore atteso elimina i fattori di rischio} \quad (5)$$

Here, the expectation is conditional, as the distribution of ξ_{t+1} may depend on the state and the selected decision.

In practice, things may not be that easy. The expectation may be tough to compute, or we may even lack a sensible model to compute it.

Then, all we may have is the possibility of observing a sample of random contributions, either by Monte Carlo simulation or online experimentation (as is typical in reinforcement learning). → Approssimazione del valore atteso

- Caso a orizzonte infinito →
1. Ovella con lo sconto
 2. Ragioniamo sulla media nel tempo
 3. Quando f ha uno stato assorbente da cui non ci spostiamo più e ciò non ci costa più nulla, cioè $f(s_t, x_t) \rightarrow 0$ in un tempo finito → STOCHASTIC SHORTEST PATH

Infinite horizon problems

1. An infinite horizon, discounted DP problem has the form:

La sommatoria ha ∞ termini e per far convergere la serie metto $\gamma < 1$ e f non dipende dal tempo

E' la situazione più facile perché lo sconto fa sì che

1. Algoritmica mente è più facile
2. Ci sono teoremi a riguardo

$$\text{opt } E_0 \left[\sum_{t=0}^{\infty} \gamma^t f(s_t, x_t) \right]$$

Se $\gamma < 1$ e $T \rightarrow \infty$ questo va velocemente a 0 e il sistema non raggiunge mai la stazionalità
→ Opzione facile: scontiamo!
Fare così

dinamica del sistema che lo pone inizialmente, cioè il modo di comportarsi del sistema figura sempre essere lo stesso

(6)

where the sum makes sense if all of the contributions are positive and bounded and we use a proper discount factor $\gamma < 1$.

Discounted DP is quite natural in business and economics applications. Sometimes, the discount factor has no real justification, and it is only used as a device to make an infinite-horizon problem easier to deal with.

2. In some engineering applications we might prefer an average over time:

In questo caso si somma, sente scontare e poi si prende la media.

In questo caso diamo per scontato che esiste una soluzione, ma non è sempre detto

$$\text{opt } \lim_{T \rightarrow \infty} E_0 \left[\frac{1}{T} \sum_{t=0}^{T-1} f(s_t, x_t) \right]$$

Computazionalmente è più
ragionevole trattare da un
punto di vista algoritmico
→ opzione più difficile

(7)

This leads to DP formulations with average contribution per stage.

3. We should also note that there are problems with an undefined horizon, where the decision process stops when we reach a target state.

Quando non sai in quanto tempo raggiungi lo stato assorbente

Domanda. Come deve essere fatta la soluzione? Voglio una politica decisionale, quindi l'output è una funzione o una sequenza di funzioni

Decision policies

At each time instant, we have to select a x_t at time t . We may introduce a rather abstract constraint like

$$x_t \in \mathcal{X}(s_t), \quad (8)$$

stating that the decision at time t must belong to a feasible set \mathcal{X} , possibly depending on the current state s_t .

This would be all we need in a deterministic setting, where at time $t = 0$ we must find a sequence of decisions $(x_t)_{\{t \geq 0\}}$ that will be implemented over time.

However, a fundamental constraint is relevant in the stochastic case. Decisions must be non-anticipative, i.e., they cannot rely on future information that has not been revealed yet.

A closed-loop, non-anticipative decision policy is naturally stated in the following feedback form:

$$x_t = \mu_t(s_t) \in \mathcal{X}(s_t),$$

↗ Mappa lo stato nella
decisione
 ↗ Diamo per scontato che la politica sia deterministica:
 e' deterministica la mappa tra lo stato e l'azione che fara'
 ↗ Politica decisionale non anticipativa

(9)

where $\mu_t(\cdot)$ is a function mapping the state at time t into a feasible decision.

We may think of a policy as a sequence of functions,

Esempio di politica decisionale che ha senso a orizzonte finito
 $s_t \leq I_t$ Il livello di magazzino soglia
 $I_t = s$

$$\mu \equiv (\mu_0(\cdot), \mu_1(\cdot), \dots, \mu_{T-1}(\cdot)),$$

↗ Funzioni che dicono come mappo le decisioni in base allo sviluppo delle cose
 → Supponiamo un orizzonte finito

(10)

one for each time instant at which we have to map the state s_t into a decision x_t .

A tempo ∞ vorremmo ottenere un schema stazionario e quindi una unica funzione = politica

In the case of an infinite horizon problem, we will look for a stationary policy, which is characterized by the same function $\mu(\cdot)$ for each stage.

Hence, a more precise statement of problem (3) could be

$$\underset{\mu \in \mathcal{M}}{\text{opt}} \mathbb{E}_0 \left[\sum_{t=0}^{T-1} \gamma^t f_t(s_t, \mu_t(s_t)) + \gamma^T F_T(s_T) \right], \quad (11)$$

where \mathcal{M} is the set of feasible policies, i.e., such that $x_t = \mu_t(s_t) \in \mathcal{X}(s_t)$ at every time instant.
La mia decisione è data dalla politica: le decisioni vengono prese da un set di politiche ammissibili - Anello chiuso

We have introduced deterministic policies. There are cases in which randomized policies may be necessary or helpful. utili se In delle situazioni potrei avere vincoli probabilistici del tipo (...) . Quando si sta facendo reinforcement learning e quindi si sta apprendendo

Randomized policies may be needed in order to cope with chance constraints on states.

$$\textcircled{a} \quad P\{s_t \in \mathcal{G}\} \geq 1 - \alpha,$$

where α is a suitably small number, the acceptable probability of violating the constraint.

Randomized policies will prove also necessary later, when dealing with the exploitation vs. exploration tradeoff in reinforcement learning.

si prova a cambiare

si guarda sempre nello stesso posto

An example: dynamic single-item lot-sizing

Let us consider the uncapacitated single-item lot-sizing problem;

Bisognerebbe modellare i ritardi della consegna, cioè se la consegna non è o quello che arriva in magazzino ora è quello che ho ordinato del tempo fa

$$\min \sum_{t=0}^{T-1} [cx_t + \phi \cdot \delta(x_t)] + \sum_{t=1}^T hI_t \quad \begin{array}{l} \text{costo variabile} \\ \text{costo fisso} \\ \text{contributo immediato} \end{array} \quad \begin{array}{l} \text{Livello di magazzino a } T \text{ che in generale} \\ \text{all'ottimo sarà o} \\ \text{Bisognerebbe avere vincoli sullo stato finale} \end{array} \quad (12)$$

$$\text{s.t. } I_{t+1} = I_t + x_t - d_{t+1}, \quad t = 0, 1, \dots, T-1 \quad (13)$$

$$x_t, I_t \geq 0,$$

where:

- x_t is the amount ordered and immediately delivered at time instants $t = 0, 1, \dots, T-1$ (under the assumption of zero delivery lead time);
- I_t is the on-hand inventory at time instants $t = 0, 1, \dots, T$; → Variabili di stato
- d_t is the demand during time interval $t = 1, \dots, T$.

The aim is to satisfy demand at minimum cost, which comprises an inventory holding charge, related with a holding cost h per item and per unit time, and an ordering cost, which includes a variable cost with rate c and a fixed cost ϕ , which is incurred only when ordering a positive amount.

To represent the fixed charge, we introduce a function $\delta(x)$ defined as

$$\delta(x) = \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{if } x = 0. \end{cases}$$

Also notice that we include cost terms cx_t accounting for linear variable costs but, in the deterministic case, this is not really necessary.

On-hand inventory level I_t is a state variable, the ordered amount x_t is a control variable, and demand d_t may be interpreted as an exogenous input if the problem is deterministic, or as a risk factor if it is stochastic.

Equation (13) is the state transition equation; I_0 is the initial state (before serving d_1), and I_T is the terminal state at the end of the planning horizon.

Se considero il problema stocastico e la domanda e' incerta il vincolo (13) non e' sempre soddisfacibile
How should we adapt the model, if we consider demand uncertainty? There are two basic models, depending on specific assumptions about customers' behavior:

*Se non soddisfo la domanda? 1. Vendite perse $d_t - I_t$ 2. Il cliente e' paziente, quindi aspetta che la merce arrivi 3. Magazzino on hand + Magazzino Backlog
 4. Il magazzino puo' andare in negativo*

- Under the lost sales assumption, the state transition equation becomes

$$I_{t+1} = \max \{0, I_t + x_t - d_{t+1}\}.$$

We should also introduce a penalty q for each unit of unsatisfied demand and include an additional term into the objective function:

$$\sum_{t=1}^T q \max \{0, d_t - (I_{t-1} + x_{t-1})\}.$$

- Penalita' per le vendite perse

2. If customers are patient, we may backlog demand at a penalty b (clearly, $b > h$).

If backlog is allowed, we have two modeling choices. 1) Keep a single state variable I_t and relax its non-negativity condition, so that I_t is interpreted as a backlog when it takes a negative value. 2) Introduce a pair of state variables: on-hand inventory $O_t \geq 0$ and backlog $B_t \geq 0$. Essentially, we are splitting I_t into its positive and negative parts, and the inventory related cost becomes

Visualizzazione più naturale

$$\sum_{t=1}^T (hO_t + bB_t).$$

Furthermore, we also split state transition equations in two parts:

Ci si aspetta che O_{t+1} o B_{t+1} sia uguale a se ci sono vendite parziali

$$O_{t+1} = \max \{0, O_t - B_t + x_t - d_{t+1}\}, \rightarrow \text{Ciò che fisicamente hai in magazzino}$$

$$B_{t+1} = \max \{0, -O_t + B_t - x_t + d_{t+1}\}. \rightarrow \text{Backlog (cioè che il cliente ti ordina mentre non hai disponibilità)}$$

There are further issues that we may have to deal with in real life:

- The value of the terminal state.
- Partially observable states. \rightarrow Problemi di qualità o di usura
- Censored demand. \rightarrow Se ho clienti impazienti non è detto che sappiamo la domanda si osservano le vendite, ma non la domanda in generale
- Delays due to delivery lead time. \rightarrow Non è detto che le cose vengano consegnate subito
- State- and decision-dependent uncertainty.

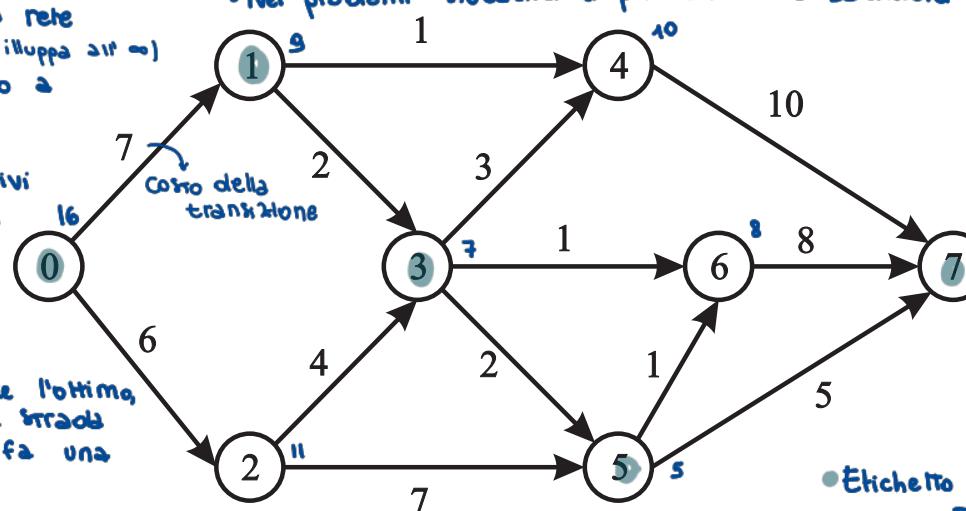
A glimpse of the DP Principle: the shortest path problem

To get a clue about the nature of DP and its optimality principle, let us consider a simple deterministic problem: the shortest path problem on a directed and acyclic network.

Problema deterministico su una rete orientata e aciclica (\rightarrow non ci si illuppa \Rightarrow si).

L'obiettivo è trovare il percorso a distanza minima tra 0 - 7

« Se gli archi hanno costi negativi e il grafo fosse ciclico il costo minimo sarebbe $-\infty$ »



Il sistema è Markoviano perché quando siamo in uno stato non è necessario sapere come ci siamo arrivati e la funzione obiettivo è additiva

La regola più semplice, che non garantisce l'ottima, per risolvere il problema è prendere la strada con peso minore tutte le volte che si fa una scelta \Rightarrow soluzione greedy myope

A **directed network** consists of a set of **nodes**, denoted by $\mathcal{N} = \{0, 1, 2, \dots, N\}$ and set of **arcs** joining pairs of nodes, denoted by \mathcal{A} .

In a directed network, an arc is an ordered pair of nodes, denoted by $(i, j) \in \mathcal{A}$, where $i, j \in \mathcal{N}$. Each arc is associated with a number $c_{ij} > 0$, which can be interpreted as the arc length (or the cost of moving along the arc).

We assume that the network is acyclic.

For a given node $i \in \mathcal{N}$, we shall denote by \mathcal{S}_i the set of (immediate) successor nodes,

$$\mathcal{S}_i = \{j \in \mathcal{N} \mid (i, j) \in \mathcal{A}\}, \rightarrow \text{Insieme dei successori immediati}$$

and by \mathcal{B}_i the set of (immediate) predecessor nodes,

$$\mathcal{B}_i = \{j \in \mathcal{N} \mid (j, i) \in \mathcal{A}\}. \rightarrow \text{Insieme dei predecessori immediati}$$

For instance,

$$\mathcal{S}_3 = \{4, 5, 6\}, \quad \mathcal{B}_3 = \{1, 2\}.$$

These sets may be empty for some nodes:

$$\mathcal{S}_7 = \emptyset, \quad \mathcal{B}_0 = \emptyset.$$

A directed path from node i to node j is a sequence of nodes $\mathcal{P} = (k_1, k_2, \dots, k_m)$, where $k_1 \equiv i$ and $k_m \equiv j$, and every ordered pair $(k_1, k_2), (k_2, k_3), \dots, (k_{m-1}, k_m)$ is an arc in \mathcal{A} . The total length of the path, denoted by $L(\mathcal{P})$, is just the sum of the arc lengths on the path.

We may consider each node as a state and each arc as a possible transition. At each state, we must choose a transition to a new state, in such a way that we move from the initial to the terminal state at minimum total cost.

A simple decision rule would be, when at a node i , moving to the nearest node:

La regola ci permette di ridefinire il problema come
a gradino singolo. «Ti preoccupi solo di oggi e non nel
domani»

$$\min_{j \in \mathcal{S}_i} c_{ij}.$$

Allo stato i scegli la transizione
verso lo stato j che ha costo minore (14)

Clearly, such a greedy decision approach need not be the optimal one (try it!).

Per non essere miopi, si assegna un valore allo stato successivo introducendo il termine V_j

However, the idea of solving a sequence of simple single-stage problems does have some nice appeal. It might be useful to refine the objective by introducing some measure of how good the next state is.

If we could associate the successor node j with a measure V_j of its value, in state i we could solve a problem like

Risolvo sempre una serie di problemi a grado singolo, ma almeno guardo la qualità oltre al costo del nuovo nodo che vado a visitare

$$\min_{j \in S_i} (c_{ij} + V_j).$$

breve termine lungotermine ↓
 Lo stato in cui vado a finire è buono o no?

How can we come up with such an additional term?

Poniamola come la lunghezza del cammino minimo da i a N destinazione finale

The starting point is to find a characterization of the optimal solution. Let V_i be the length of the shortest path from node $i \in \mathcal{N}$ to the terminal node N , denoted as $i \xrightarrow{*} N$:

$$V_i \doteq L(i \xrightarrow{*} N).$$

Let us further assume that, given an optimal path from i to N , a node j lies on this path.

Then, it is easy to realize that the following property must hold:

$$j \xrightarrow{*} N \text{ is a subpath of } i \xrightarrow{*} N.$$

→ Se nel cammino ottimo passo per j , allora i cammini ottimi da $i \rightarrow j$ e da $j \rightarrow N$ sono sottocammini di quello ottimo

To understand why, consider the decomposition of $i \xrightarrow{*} N$ into two subpaths: $\mathcal{P}_{i \rightarrow j}$ from node i to node j , and $\mathcal{P}_{j \rightarrow N}$ from node j to node N . The length of $i \xrightarrow{*} N$ is the sum of the lengths of the two subpaths:

$$V_i = L(i \xrightarrow{*} N) = L(\mathcal{P}_{i \rightarrow j}) + L(\mathcal{P}_{j \rightarrow N}). \quad (15)$$

Note that the second subpath is not affected by *how* we go from i to j . This system is Markovian, in the sense that how we get to node j has no influence on the length of any “future” path starting from node j .

Our claim amounts to saying that the subpath $\mathcal{P}_{j \rightarrow N}$ in the decomposition of Eq. (15) is optimal, in the sense that $L(\mathcal{P}_{j \rightarrow N}) = L(j \xrightarrow{*} N)$. To see this, assume that $\mathcal{P}_{j \rightarrow N}$ is *not* an optimal path from j to N , i.e., $L(\mathcal{P}_{j \rightarrow N}) > L(j \xrightarrow{*} N)$.

Then we could improve the right-hand side of Eq. (15) by considering the path consisting of the same initial path $\mathcal{P}_{i \rightarrow j}$, followed by an optimal path $j \xrightarrow{*} N$:

$$L(\mathcal{P}_{i \rightarrow j}) + L(j \xrightarrow{*} N) < L(\mathcal{P}_{i \rightarrow j}) + L(\mathcal{P}_{j \rightarrow N}) = L(i \xrightarrow{*} N) = V_i,$$

which is a contradiction, since we have assumed that V_i is the length of a shortest path.

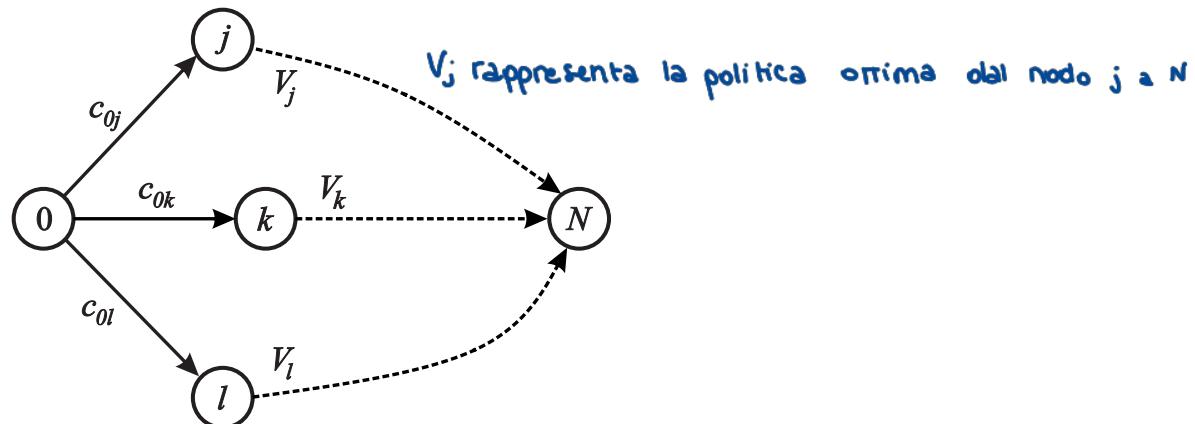
I percorsi di lunghezza minima godono di proprietà di amidamento
Shortest paths enjoy a sort of nesting property, and this finding suggests a recursive
decomposition of the overall task of finding the shortest path $0 \xrightarrow{*} N$.

If we knew the values V_j for each node $j \in \mathcal{S}_0$, we could make the first decision by solving the single-stage problem

$$V_0 = \min_{j \in \mathcal{S}_0} (c_{0j} + V_j). \quad (16)$$

Non e' influenzato da come arrivo a j

Non e' l'algoritmo migliore e quindi neanche quello che e' presente nel navigatore
 Altri algoritmi, nel distribuito o a grandi dimensioni, etichettano i nodi ricorsivamente piu' volte



→ Ci costruiamo le informazioni che ci servono all'indietro, cioè partendo dalla destinazione

Equation (16) can be applied to any node, not just the initial one. This gives a **backward dynamic programming**, starting from the the terminal condition $V_N = 0$ and solving a **functional recursive equation**:

Così che i successori del nodo che sto etichettando siano già tutti etichettati

Per evitare il circolo vizioso bisogna seguire l'ordinamento topologico dei nodi quindi etichetti un nodo solo quando tutti i suoi successori sono già stati etichettati

$$V_i = \min_{j \in S_i \rightarrow \text{Successori di } i} \{c_{ij} + V_j\}, \quad \forall i \in \mathcal{N}.$$

Funzione valore
Risoluiamo il problema per enumerazione completa

Equazione ricorsiva da risolvere per ogni nodo (17)

The only issue is that we may label node i with its value V_i only after all of its successor nodes $j \in S_i$ have been labeled with values V_j . This may be accomplished by **topological ordering** of nodes, which can always be achieved for an acyclic network.

Equation (17) is a functional equation, since it provides us with a function, $V(\cdot) : \mathcal{N} \rightarrow \mathbb{R}$, mapping states to their values. In this discrete-state case, this amounts to finding a vector of state values. i.e., a numeric label for each node.

Una volta che hai la V_i si definisce e si ottiene anche la politica
Si costituisce la politica piano piano e se lo spazio è finito e a dimensione limitata la racchiudi in un vettore

Numerical example

Sei arrivato a definizione

We have the terminal condition $V_7 = 0$ for the terminal node, and we consider its immediate predecessors 4 and 6 (we cannot label node 5 yet; note that we did *not* number nodes using topological ordering).

We find

$$V_4 = c_{47} + V_7 = 10 + 0 = 10,$$
$$V_6 = c_{67} + V_7 = 8 + 0 = 8.$$

If we denote the selected successor node, when at node $i \in \mathcal{N}$, by a_i^* (the optimal action at state i), we trivially have

$$a_4^* = 7, \quad a_6^* = 7.$$

Now we may label node 5:

$$V_5 = \min \left\{ \begin{array}{l} c_{56} + V_6 \\ c_{57} + V_7 \end{array} \right\} = \min \left\{ \begin{array}{l} 1 + 8 \\ 5 + 0 \end{array} \right\} = 5 \quad \Rightarrow \quad a_5^* = 7.$$

Then we consider node 3, whose immediate successors 4, 5, and 6 have already been labeled:

$$V_3 = \min \left\{ \begin{array}{l} c_{34} + V_4 \\ c_{35} + V_5 \\ c_{36} + V_6 \end{array} \right\} = \min \left\{ \begin{array}{l} 3 + 10 \\ 2 + 5 \\ 1 + 8 \end{array} \right\} = 7 \quad \Rightarrow \quad a_3^* = 5.$$

By the same token, we have

$$V_1 = \min \left\{ \begin{array}{l} c_{13} + V_3 \\ c_{14} + V_4 \end{array} \right\} = \min \left\{ \begin{array}{l} 2 + 7 \\ 1 + 10 \end{array} \right\} = 9 \Rightarrow a_1^* = 3,$$

$$V_2 = \min \left\{ \begin{array}{l} c_{23} + V_3 \\ c_{25} + V_5 \end{array} \right\} = \min \left\{ \begin{array}{l} 4 + 7 \\ 7 + 5 \end{array} \right\} = 11 \Rightarrow a_2^* = 3,$$

$$V_0 = \min \left\{ \begin{array}{l} c_{01} + V_1 \\ c_{02} + V_2 \end{array} \right\} = \min \left\{ \begin{array}{l} 7 + 9 \\ 6 + 11 \end{array} \right\} = 16 \Rightarrow a_0^* = 1.$$

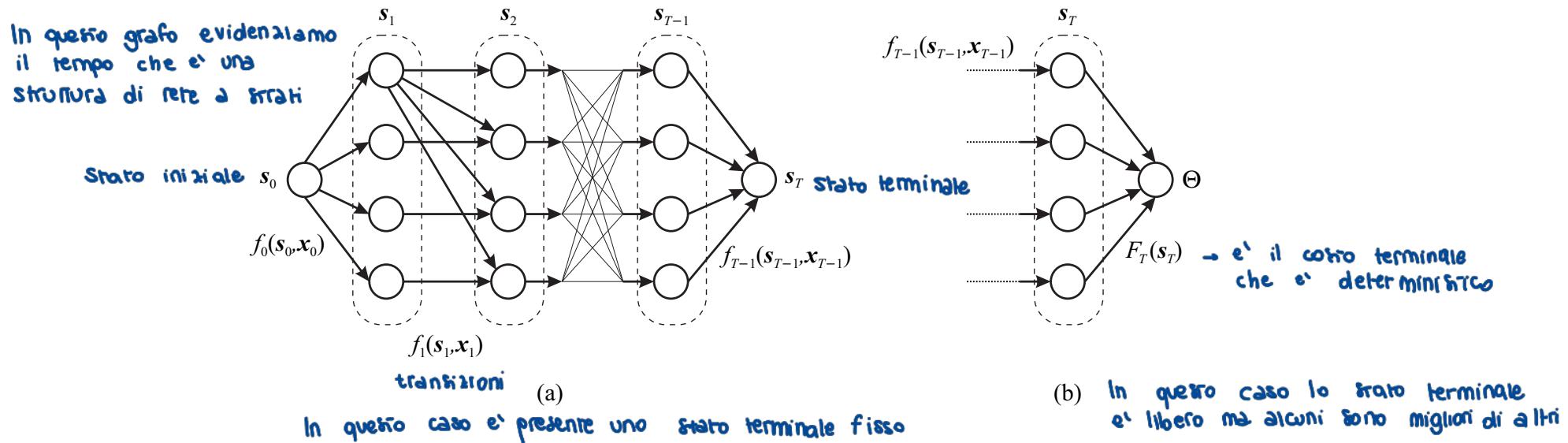
The optimal actions at each state can be stored into a table, and we may easily find the shortest path by applying the decision policy: (0, 1, 3, 5, 7), with total length 16.

- Shortest path algoritmo polinomiale
- Il problema di lot sizing non e' invece polinomiale

↳ Nel caso deterministico, ottenendo questo
otteniamo anche la politica

Shortest paths on structured networks

The shortest path problem that we have considered in the previous section features an unstructured network. A more structured case occurs when the nodes in the network correspond to states of a dynamic system evolving over time:



This kind of graph corresponds to a deterministic sequential decision problem with finite states, possibly resulting from discretization of a continuous state space.

If the terminal state s_T is fixed, there is no point in associating a terminal value with it [Fig. (a)]. If the terminal state is free, it may make more sense to associate a terminal contribution $F_T(s_T)$ with it. In Fig. (b) we add a time layer to the graph and introduce a dummy terminal node Θ .

The DP decomposition principle

The shortest path problem suggests that we may decompose a multistage decision problem into a sequence of simpler single-stage problems.

In a deterministic problem, we look for a sequence of vectors $(x_0, x_1, \dots, x_{T-1})$, to optimize e performance measure,

$$\text{opt } H(x_0, x_1, \dots, x_{T-1}; s_0, s_1, \dots, s_T),$$

subject to a set of constraints on states and decisions.

In the stochastic case, the problem is

$$\text{opt } E\left[H(x_0, x_1, \dots, x_{T-1}; s_0, s_1, \dots, s_T)\right],$$

↗ Nel caso deterministico, le decisioni sono tutte prese all'istante 0 e poi applicate ai vari istanti di tempo successivi
 ↘ Vettore di un numero di elementi finito

where expectation is taken with respect to a sequence of random variables (ξ_1, \dots, ξ_T) .

This notation hides the true multistage nature of the problem, as now we should find a sequence of functions $(x_0, x_1, \dots, x_{T-1})$.

In fact, apart from the initial decision x_0 to be made here and now, the decision vectors may actually be functions of past decisions and the observed realizations of the risk factors,

$$x_t = \mu_t(x_{[t-1]}, \xi_{[t]}), \quad t = 1, \dots, T-1. \tag{18}$$

↗ decisione presa all'istante di tempo t
 ↘ Funzione che vive in uno spazio a dimensione infinita

If we find a suitable set of state variables, we may simplify Eq. (18) and look for simpler functions mapping the current state into the optimal decision, $x_t = \mu_t(s_t)$.

Apart from this Markov structure in system dynamics, a nice form of decomposition is feasible if we assume that the objective function takes the additive form

$$E_0 \left[\sum_{t=0}^{T-1} \gamma^t f_t(s_t, x_t) + \gamma^T F_T(s_T) \right].$$

c'e' un contributo immediato per ogni istante di tempo e questo ci permette di decomporne

Bisogna chiedersi se T lo scelgo io oppure e' imposto dal problema

A quick and dirty decision rule is: when we are at state s_t , solve the myopic problem

↳ regola di decisione molto brutale che fa il meglio solo per il momento
questa regola non garantisce di ottenere l'ottimo

$$\underset{x_t \in \mathcal{X}(s_t)}{\text{opt}} f_t(s_t, x_t),$$

where $\mathcal{X}(s_t)$ is the set of feasible decisions at state s_t .

Vogliamo trovare un compromesso tra breve e lungo termine

We already know that such a greedy approach is not expected to perform well in general, and we must balance short- and long-term objectives, by introducing a suitable state value function $V_t(\cdot)$.

The fundamental idea of dynamic programming is that we may actually find a value function such that we can achieve the optimal performance.

→ la programmazione dinamica in generale gira attorno al valore di $V_t(s)$

The value $V_t(s)$ should be the expected reward/cost obtained when we apply an optimal policy from time t onwards, starting from state s .

Calcoliamo le V-function andando indietro nel tempo
Tipicamente si ottiene un'approssimazione di $V_t(s_t)$ che implicitamente definisce una politica

Formally, when in state s_t at time t , we must solve the problem:

Valore atteso condizionato alla decisione che prendo x_t , e' riferito a s_{t+1}

Definendo in questo modo V_t
e' possibile de componere

$$V_t(s_t) = \underset{x_t \in \mathcal{X}(s_t)}{\text{opt}} \left\{ f_t(s_t, x_t) + \gamma \mathbb{E} \left[V_{t+1}(g_{t+1}(s_{t+1}, x_t, \xi_{t+1})) \mid s_t, x_t \right] \right\}. \quad \text{stato futuro} \quad (19)$$

Valore dello stato s_t al tempo t
E' cio' che permette di rendere l'approccio non miopic: valore atteso dello stato futuro
fattore di scontro $= s_{t+1}$

↳ ottimizzo solo rispetto alla decisione che devo prendere ora → problema a grado singolo → Decomposition
This recursive functional equation is referred to as **Bellman's equation**.
↳ L'ha formalizzata nel 1957

The optimal decision x_t^* is obtained by solving an optimization problem parameterized by the current state s_t , based on the knowledge of the value function $V_{t+1}(\cdot)$.

In the case of a small finite state space, we may represent the optimal decision policy in a tabular form: we just have to store, for each time instant t and state s_t , the corresponding optimal decision. In general, the decision policy is not explicit, but implicit in the sequence of value functions.

In both cases, the optimal decision depends on the state and, conceptually, we find an optimal decision policy in the feedback form of Eq. (9), which we repeat here:

$$x_t^* = \mu_t^*(s_t) \in \mathcal{X}(s_t). \quad (20)$$

By putting all of the functions $\mu_t^*(\cdot)$ together, we find the overall optimal policy in the form of Eq. (10):

$$\mu^* \equiv (\mu_0^*(\cdot), \mu_1^*(\cdot), \dots, \mu_{T-1}^*(\cdot)). \rightarrow \text{Fa riferimento a tempo finito} \quad (21)$$

When we consider a single function $\mu^*(\cdot)$ that does not depend on time, we talk of a stationary policy. This is suitable for infinite horizon problems, as we shall see. Sometimes we may settle for a suboptimal policy, possibly resulting from an approximation of the optimal value function.

E' come se stessimo mettendo in un teorema l'intuizione
 To ... M1-Ve ... il meglio To-Ve
 ... non puo' essere il meglio per M1-Ve

Theorem: the DP principle of optimality

Consider an optimal policy

✓ Immaginiamo di avere la politica e quindi la soluzione ottima

$$\left(\mu_0^*(\cdot), \mu_1^*(\cdot), \dots, \mu_{T-1}^*(\cdot) \right) \rightarrow \text{Politica ottima}$$

for the multistage problem

$$\text{opt } E_0 \left[\sum_{t=0}^{T-1} \gamma^t f_t(s_t, x_t) + \gamma^T F_T(s_T) \right].$$

✓ Siamo in uno stato e ora vogliamo fare il meglio che riusciamo per lo stato in cui siamo, non dall'origine
 Assume that at time τ we are in state s_τ , and consider the tail problem

$$\text{opt } E_\tau \left[\sum_{t=\tau}^{T-1} \gamma^{t-\tau} f_t(s_t, x_t) + \gamma^{T-\tau} F_T(s_T) \right].$$

Then, the truncated policy $(\mu_\tau^*(\cdot), \mu_{\tau+1}^*(\cdot), \dots, \mu_{T-1}^*(\cdot))$ is optimal for the tail problem.

✓ Per il problema di coda, la politica ottima rimane un troncamento della politica che abbiamo assunto all'inizio

Proofs rely on mathematical induction and may be rather complicated when subtle mathematical issues are accounted for.

The functional equation (19) is an optimality equation and requires finding the value function for each time instant.

The natural solution process goes backward in time, starting from the terminal condition

Es. La variabile di stato per la vendita di biglietti aerei e' quanti posti ancora hai disponibili in questo caso $V_T(s_T) = 0$ perché il problema e' a ottone finito

$$V_T(s_T) = F_T(s_T) \quad \forall s_T, \quad \begin{array}{l} \hookrightarrow \text{risolvendo iterativamente il problema} \\ \hookrightarrow \text{specificato dalle condizioni del problema} \\ \hookrightarrow \text{Condizione di innesco delle iterazioni all'indietro} \\ \text{stato terminale} \end{array}$$

i.e., the value of the terminal state is given by function $F_T(\cdot)$.

Then, at the last decision time instant, $t = T - 1$, we should solve the single-stage problem

*Problema statico, ma
Non myope*

$$V_{T-1}(s_{T-1}) = \underset{x_{T-1} \in \mathcal{X}(s_{T-1})}{\text{opt}} \left\{ f_{T-1}(s_{T-1}, x_{T-1}) + \right. \\ \left. \gamma \mathbb{E}[V_T(g_T(s_{T-1}, x_{T-1}, \xi_T)) | s_{T-1}, x_{T-1}] \right\}, \quad (22)$$

*Risolviamo per ogni
valore possibile dello
stato s_{T-1}*

In termini di decisioni, l'istante $T-1$ rappresenta l'ultima decisione presa

grazie a questo guardo anche il futuro e il problema non e' più myope

for every possible state s_{T-1} .

This is a static, but not myopic problem, since the terminal value function $V_T(\cdot)$ also accounts for the effect of the last decision x_{T-1} on the terminal state.

By solving it for every state s_{T-1} , we build the value function $V_{T-1}(\cdot)$. Then, unfolding the recursion backward in time, we find the value function $V_{T-2}(s_{T-2})$, and so on, down to $V_1(s_1)$.

Finally, given the initial state s_0 , we find the first optimal decision by solving the single-stage problem

Se sei in uno stato s_0 ben definito,
V_t non ha senso di calcolarla in quanto
non ha senso risolvere il problema
per tutti i valori iniziali quando sai
già qual è il tuo primo valore

$$V_0(s_0) = \underset{x_0 \in \mathcal{X}(s_0)}{\text{opt}} \left\{ f_0(s_0, x_0) + \gamma E[V_1(g_1(s_0, x_0, \xi_1)) | s_0, x_0] \right\}. \quad (23)$$

Si itera all'indietro fino alla V-funzione all'orizzonte iniziale o

How should we exploit the knowledge of the value functions $V_t(\cdot)$? →

Nel caso di orizzonte finito

Gi basta un'approssimazione
senz'aria che indica una politica
che funziona bene
- confronti con un'eutistica.
- la provi e vedi se migliori.

- In a deterministic setting, we may find the sequence of optimal decisions x_t^* by solving a sequence of single-stage problems, where we update the state variables according to the applied decisions. → costruisci la sequenza di decisioni e hai finito
- In a stochastic we may run a Monte Carlo simulation as follows:
 - Given the initial state s_0 and the value function $V_1(\cdot)$, solve the first-stage problem and find x_0^* . → Trovare x_0^* la applico per poi trarre lo stato successivo s_1 .
 - Sample the random risk factors ξ_1 and use the transition function to generate the next state, $s_1 = g_1(s_0, x_0^*, \xi_1)$.
 - Given the state s_1 and the value function $V_2(\cdot)$, solve the second-stage problem and find x_1^* .
 - Repeat the process until we generate the last decision x_{T-1}^* and the terminal state s_T .

Passiamo a orizzonte infinito → 1. Programmazione dinamica scontata → possiamo ottenere un operatore di contrazione che ha un unico punto fisso da cui possiamo dimostrare che l'algoritmo converge tanto velocemente in base a γ
 ↓
 2. Valore medio
 Cosa possiamo fare?
 → Applichiamo questi approcci a tempo discreto, nel continuo è più difficile

Stochastic DP for infinite time horizons

The recursive form of Eq. (19) needs to be adjusted when coping with an infinite-horizon problem like

$$\text{opt } E \left[\sum_{t=0}^{+\infty} \gamma^t f(s_t, x_t) \right], \quad (24)$$

Consideriamo solo il caso scontato

where we assume that immediate costs are bounded and $\gamma < 1$, so that the series converges to a finite value, and we drop the subscript t from the immediate contribution, as well as from the state-transition function $s_{t+1} = g(s_t, x_t, \xi_{t+1})$.

The functional equation boils down to

$$V(s) = \text{opt}_{x \in \mathcal{X}(s)} \left\{ f(s, x) + \gamma E[V(g(s, x, \xi))] \right\},$$

L'intuizione aiuta, ma non è effettivamente facile dimostrare che tutto sia ben definito
 Definiamo come equazione del punto fisso
 Ottieniamo una sola V-function
 Bisogna capire come risolvere questa equazione
 tramite un algoritmo numerico iterativo

where $\mathcal{X}(s)$ is the set of feasible decisions when we are in state s .

Now we have a value function defined as the fixed point of a possibly complicated operator, as we have $V(s)$ on both sides of the equation. An iterative method is needed to solve Eq. (25).