

Marco Casu

♪ Automazione ♪



SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Dipartimento di Informatica

Questo documento è distribuito sotto la licenza [GNU](#), è un resoconto degli appunti (eventualmente integrati con libri di testo) tratti dalle lezioni del corso di Automazione per la laurea triennale in Informatica. Se dovessi notare errori, ti prego di segnalarli.

Nota bene : Essendo questi appunti di un corso esterno alla facoltà di Informatica, è presente un capitolo "Complementi" che può risultare utile al lettore.



INDICE

1 L'Automazione Industriale	5
1.1 Introduzione	5
1.2 Processi Industriali	10
1.2.1 Sistema di Controllo	11
1.3 Analisi dei Sistemi di Produzione	14
1.3.1 Linee di Trasferta	15
1.3.2 Flow Shop	20
1.4 Sistema di Supporto	23
1.5 La Piramide CIM	24
1.5.1 Livelli della Piramide	24
1.5.2 Auto-diagnostica	27
1.5.3 Architetture per il Controllo	28
1.6 Industria 4.0	31
1.6.1 Tecnologie Abilitanti	32
1.6.2 Modelli di Business	33
1.6.3 Industria 5.0	36
2 Reti per l'Automazione	37
2.1 Sistemi di Comunicazione	37
2.2 Classificazione ed Architetture delle Reti	39
2.2.1 Tipologie di Reti	40
2.3 Livello Fisico e di Collegamento	42
2.3.1 Protocolli MAC	44
2.4 I Protocolli Fieldbus	45
3 Sistemi di Controllo Real Time	47
3.1 Parallelismo e Programmazione Concorrente	48
3.2 Il Problema dello Scheduling	50
3.2.1 Classificazione degli Algoritmi	51
3.2.2 Scheduling di Task Periodici	52
3.2.3 Utilizzo del Processore	54
3.3 Gli Algoritmi di Scheduling	54
3.3.1 Rate Monotoning Priority Ordering (RMPO)	54
3.3.2 Earliest Deadline First (EDF)	56
3.3.3 Deadline Monotoning Priority Ordering (DMPO)	58
3.3.4 Time Scheduling (TS)	58
3.4 Scheduling di Task Misti	60

3.5	Servizio in Background	60
3.6	Processo Server	61
3.6.1	Polling Server	62
3.6.2	Deferrable Server	63
3.7	Esercizi sullo Scheduling	63
4	Implementazione di Sistemi Real Time	65
4.1	Hardware Abstraction Layer	65
4.1.1	Sistemi di Automazione Real Time	67
4.2	Sistemi Operativi Real Time	67
4.3	Controllori Embedded	68
4.3.1	Arduino e Raspberry PI	69
4.4	PLC	70
4.5	Sequential Functional Chart	71
4.5.1	Elementi di Base	71
4.5.2	Regole di Evoluzione	72
4.5.3	Esecuzione Ciclica e Ambiguità	74
4.6	Sintassi del Linguaggio SFC	75
4.6.1	Qualificatori delle Azioni	75
4.6.2	Strutture di Collegamento	79
4.7	Esercitazione sul Linguaggio SFC	87
4.7.1	Esercizio 1 - Carrelli semoventi	87
4.7.2	Esercizio 2 - Cisterna di miscelazione	90
5	Attuazione e Controllo del Moto	91
5.0.1	Amplificazione PWM	92
5.1	Il Motore a Corrente Continua	93
5.1.1	Modello del Motore DC	95
5.1.2	FeedForward	99
5.2	Controllore del Motore DC	100
5.2.1	Esempio di FeedForward	100
5.2.2	Accenno ai Regolatori PID	102
5.2.3	Progetto del Controllore - Anelli a Cascata	103
5.3	Sincronizzazione dei Moti	106
5.3.1	Profili di Moto	109
6	Regolatori PID	113
6.1	Campionamento	113
6.1.1	Perdita di Informazioni	115
6.1.2	Ricostruzione ZOH	116
6.2	Progetto del Regolatore PID	117
6.2.1	Specifiche del Progetto	117
6.2.2	Caratteristiche del Regolatore	118
6.2.3	Funzione di Trasferimento	120
6.3	Realizzazione Digitale	121
6.3.1	Discretizzazione del Regolatore	122
6.3.2	Derivata Filtrata in Banda	124
6.3.3	Schemi Realizzativi	126
6.3.4	Simulazione e Sforzo di Controllo	127
6.4	Desaturazione e Tuning	130
6.4.1	Saturazione dell'Attuatore	131
6.4.2	Anti Wind-Up	134
6.4.3	PID Digitale con Anti Wind-Up	136
6.5	Tuning del PID	138
6.5.1	Primo Metodo di Ziegler-Nichols	138
6.5.2	Esempio Numerico 1	139
6.5.3	Esempio Numerico 2	143
6.5.4	Secondo Metodo di Ziegler-Nichols	146
6.5.5	Esempio Numerico 3	146

7 Sistemi a Eventi Discreti	148
7.1 Automi a Stati Finiti	148
7.1.1 Controllo dei DEDS	153
7.2 Reti di Petri	156
7.2.1 Scatto delle Transizioni	158
7.2.2 Dinamica	160
7.3 Proprietà delle Reti di Petri	161
7.3.1 Strutture Fondamentali	164
7.4 Classi Particolari di Reti di Petri	165
7.4.1 Esempi di Reti di Petri	167
7.5 Albero di Raggiungibilità	171
7.6 Analisi Matriciale	176
8 Complementi	178
8.1 La Trasformata di Laplace	178
8.1.1 Proprietà della Trasformata	179
8.1.2 Trasformata inversa	181
8.1.3 Trasformate note	182
8.1.4 Funzione di trasferimento	182
8.1.5 Zeri e Poli	184
8.1.6 Coefficiente di Smorzamento e Pulsaione Naturale	185
8.1.7 Espansione in Fratti Semplici	187
8.2 Elementi di Teoria dei Sistemi e Controlli Automatici	187
8.2.1 Equilibrio	189
8.2.2 Rappresentazione con Matrici	189
8.3 Stabilità di un Sistema	191
8.3.1 Studio di e^{At} per la Stabilità	194
8.4 Raggiungibilità e Osservabilità	198
8.5 Schemi a Blocchi	200
8.5.1 Risposta allo Scalino	203
8.6 Analisi in Frequenza	210
8.7 Banda Passante e Margine di Fase	213
8.7.1 Criterio di Nyquist	214
8.8 Analisi dei Sistemi Retro Azionati	216
8.8.1 Analisi di $F(s)$	217
8.8.2 Esempio di Progettazione	219

CAPITOLO

1

L'AUTOMAZIONE INDUSTRIALE

1.1 Introduzione

Con il termine *Automazione*, si intende la trasformazione di un processo pre-esistente, al fine di renderlo autonomo, riducendo o sostituendo del tutto l'intervento umano, verrà trattata l'automazione dei processi industriali e manifatturieri, e del loro controllo e supervisione. I sistemi presi in considerazione evolvono nel tempo e reagiscono ad eventi, che ne cambiano lo stato, e scaturiscono dei feedback, per un eventuale correzione dell'errore.

Con sistema *autonomo* si intende un sistema in cui viene ridotto l'intervento umano. L'*Automatica* si occupa di sfruttare gli strumenti dell'informatica per l'automazione, acquisendo informazioni dal mondo fisico tramite appositi sensori, per poi essere elaborate su un sistema di controllo (calcolatore), o una rete di calcolatori, che implementa dei protocolli standard per l'industria. Differentemente da altri contesti, come la trasmissione (ad esempio, di un video in streaming) nelle reti dell'automazione i ritardi risultano critici, e vanno ridotti al minimo.

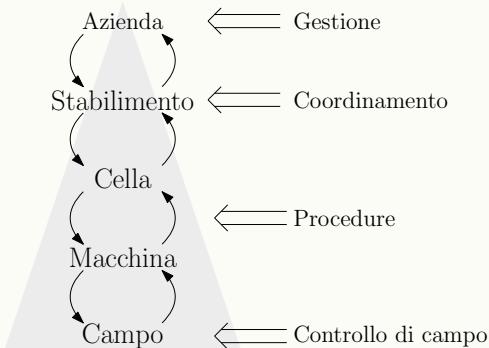


Figura 1.1: Piramide CIM

La *piramide CIM*, mostrata in figura 1.1 schematizza la gestione di un processo industriale e delle sue procedure, ogni strato comunica con quelli adiacenti scambiandosi informazioni, nei livelli più alti, le informazioni sono più *raffinate* ed astratte, nei livelli più bassi sono più *grezze*, ad esempio

- Al livello azienda viene decisa la produzione di un articolo (che coinvolgerà l'utilizzo di un braccio robotico)

- Al livello macchina, l'informazione che arriverà al braccio sarà semplicemente relativa ai gradi in cui i suoi giunti devono ruotare
- Al livello di campo, l'informazione comprenderà semplicemente il voltaggio da applicare alla macchina in questione per avere l'effetto desiderato.

Con **cella**, si intende un'unità composta da più macchine, in cui viene scambiato e lavorato del materiale per compiere delle azioni, il *controllo delle procedure* si occupa delle **macchine**, ed uno **stabilimento** è un complesso di celle/parti e catene di montaggio. Nel livello di campo, vengono utilizzati vari dispositivi, quali

- motori elettrici, servomotori, encoder
- azionatori di valvole, dinamo tachimetriche, sensori di temperatura

Tali sensori presenteranno un comportamento lineare, ad esempio, se una tensione x causa una rotazione di y giri per minuto, allora una tensione $2x$ causerà una rotazione di $2y$. Anche se tali dispositivi non si prestano ad un comportamento lineare, ne verrà causata una volontaria linearizzazione, correggendo il comportamento.

Argomento centrale saranno i regolatori *PID*, la cui definizione, come molte altre trattate in questo capitolo, sarà ripresa ed approfondita in seguito. Tali regolatori agiscono su delle grandezze di campo.



Figura 1.2: schematizzazione del regolatore PID

Si consideri il seguente esempio di regolatore, vi è una stufa che deve riscaldare una stanza, ed un sensore che ne misura la temperatura, il valore da raggiungere, detto *setpoint*, è di 20 gradi celsius. Si supponga che il sensore, una volta rilevata la temperatura, debba accendere e spegnere la stufa in modo che si raggiunga la temperatura adeguata.

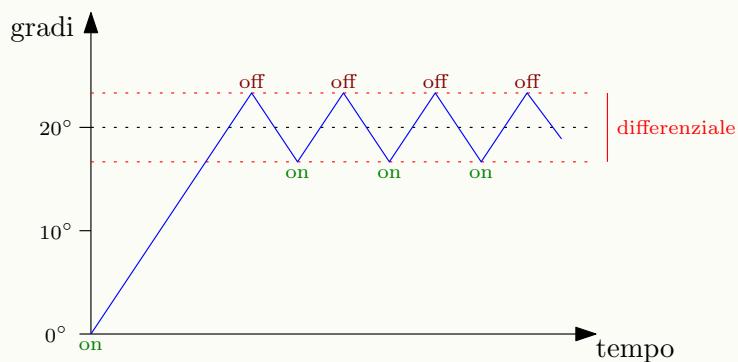


Figura 1.3: Azioni sulla stufa

In figura 1.3, il differenziale rappresenta un margine di differenza rispetto il setpoint, quando la temperatura è sotto il limite inferiore, la stufa viene accesa, quando è oltre il limite superiore, viene spenta (è chiaro che la velocità con la quale la temperatura cambia dipende dalle capacità della stufa e dalla dispersione del calore nella stanza).

Ridurre il valore del differenziale costringerebbe la temperatura ad assestarsi sempre di più sul valore desiderato, ma ciò, comporterebbe un'accensione/spegnimento della stufa più frequente, aumentando lo *sforzo di controllo*, è quindi, in questo caso, accettabile un differenziale di 2° .

Tale modello di controllo è il più semplice che ci sia, esistono ovviamente altri modi di regolare un segnale in modo che esso raggiunga il valore desiderato, ad esempio, calcolare l'errore e (ossia la differenza fra il valore desiderato ed il valore effettivo) e scalarlo ad una certa costante K_p per poi utilizzare tale valore nella regolazione del segnale.



Figura 1.4: Regolatore proporzionale

Anche se la variazione della temperatura è continua nel tempo, il suo superare una certa soglia è un evento, i PLC (controllori logici programmabili) agiscono sulle misure di campo, un noto linguaggio utilizzato per descriverne il funzionamento è noto come *Sequential Flow Chart (SFC)*.

Nei sistemi di automazione industriale vengono prediletti controllori e sensori distribuiti piuttosto che centralizzati, se ne vuole dare una dimostrazione pratica con il seguente esempio : Siano dati i due modi per trasportare un oggetto su un nastro trasportatore

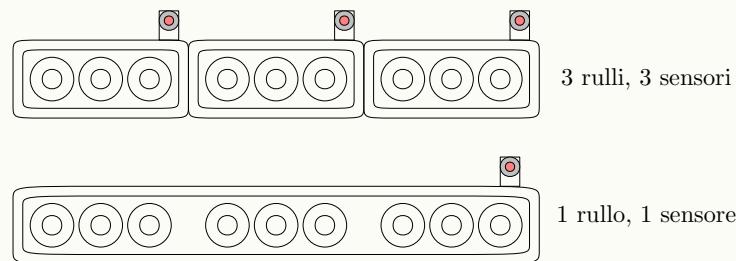


Figura 1.5: Rulli

Ogni nastro ha un sensore, se un oggetto è rilevato sopra il nastro, allora il motore si attiva. Risulta più efficiente la soluzione con 3 nastri in quanto sarà adoperata solamente la zona del nastro in cui è rilevato l'oggetto, piuttosto che l'intero nastro.

Per la modellizzazione di sistemi autonomi verranno adoperati automi a stati finiti, ampiamente trattati nel corso di **Automi, Calcolabilità e Complessità**, e *Reti di Petri*. Una rete di Petri, non è altro che un grafo bipartito, in cui ogni nodo appartiene ad un'insieme fra

- nodi *posto*
- nodi *transizione*

Inoltre, i nodi posto possono essere annotati con dei pallini neri, detti *token*, essi rappresentano lo stato del sistema in quanto indicano che delle risorse (in senso generale) sono disponibili in un posto, permettendo eventualmente una transizione. Ogni arco del grafo collega un nodo posto ad un nodo transizione.



Figura 1.6: Esempio di una rete di Petri

Le macchine per l'automazione possono essere di vari tipi, ad esempio, comprendere un unico attuatore, e più meccanismi di attuazione del moto che utilizzano una sola fonte. Un altro tipo di macchine sono quelle a *controllo numerico*, macchine programmate per fare compiti elementari periodicamente.

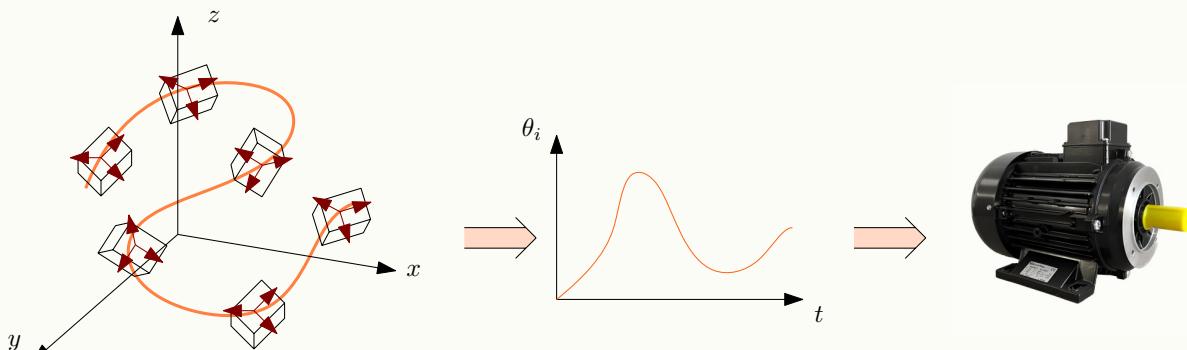
Quando in un processo produttivo il materiale viene trasformato in maniera continuativa (come nell'industria farmaceutica o alimentare) si parla di **produzione continua**. Nel caso in cui i materiali sono processati in quantità finite e determinate si parla di **produzione a lotti**, le pause fra la lavorazione di un lotto e l'altro sono dovute al fatto che è necessario trasformare solo una determinata quantità di materiale grezzo.

L'automazione industriale fa largo utilizzo dei *robot*, bracci meccanici che presentano diversi gradi di libertà, ossia giunti, che possono ruotare attorno un certo asse.



L'organo terminale, posto alla fine del braccio (in un certo senso, la sua "mano"), può assumere una certa configurazione (posizione e direzione) a seconda della rotazione di ogni giunto del braccio. Si può dire che la posizione finale p e la sua direzione sono in funzione degli angoli $\theta_1, \theta_2, \dots, \theta_n$ di rotazione di ogni giunto.

La procedura di comando da far eseguire al robot si traduce in una funzione nel tempo che descrive in che modo deve variare la rotazione di ogni singolo giunto, quest'ultima al livello di campo, si traduce nell'attuazione dei motori elettrici posti sui giunti.



Altri tipi di macchine per la movimentazione oltre i robot sono i rulli o i carrelli automatici, questi ultimi inizialmente potevano muoversi seguendo un percorso stabilito da magneti posti sul terreno, vengono dotati di sensori di prossimità per evitare collisioni. I carrelli moderni non sono limitati da percorsi prestabiliti, sono autonomi e possono fare percorsi arbitrari, che vengono calcolati da un elaboratore che ha la "visione" completa di essi.

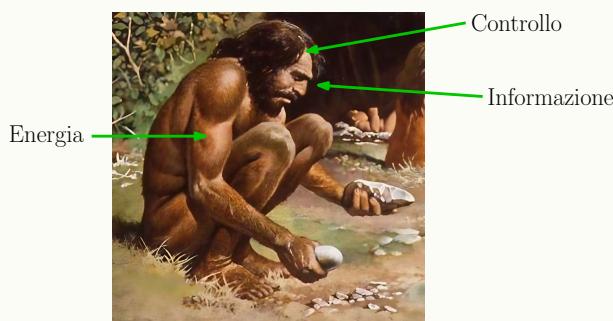
Sorge spontaneo chiedersi quale sia la differenza fra Automazione e Robotica.

- *Analogie* : Entrambe coinvolgono l'informatica ed i calcolatori, interfacciandosi con il mondo fisico, entrambe sfruttano conoscenze e tecnologie multi-disciplinari.
- *Differenze* : La robotica mostra la fattibilità di una soluzione, l'automazione si occupa di porsi delle domande riguardo tale soluzione, fra cui l'efficienza, l'ottimalità e l'affidabilità.

Formalmente, si definisce **processo** la *trasformazione* di *materiali* in *prodotti*. Tale trasformazione richiede

- Energia
- Informazione
- Controllo

Anche la costruzione di uno strumento per la caccia da parte di un uomo primitivo è un processo, in quel caso, l'energia è data dai muscoli del corpo, l'informazione viene dai sensi, quali vista e tatto, ed il controllo avviene da parte del cervello.



Lo scopo dell'automazione nel tempo è stato quello di sostituire o eliminare l'intervento dell'uomo nei processi, spesso è faticoso e pericoloso fornire energia, e l'uomo non ha le capacità sufficienti per gestire in maniera precisa l'informazione ed il controllo.

Il **primo passo** di industrializzazione è stato quello di sostituire l'energia fornita dall'uomo con l'energia naturale ed animale, durante la prima rivoluzione industriale, dove la produzione dipendeva da macchine azionate tramite potenza meccanica derivante da fonti energetiche come mulini.

Il **secondo passo** riguarda la sostituzione delle operazioni di controllo, un importante esempio fu il *regolatore di velocità* di Watt (1785), fu la prima applicazione di regolatore automatico, e sfruttava la forza centrifuga di due masse in rotazione per regolare la velocità di una macchina a vapore.

Il funzionamento è semplice, il vapore passante per la valvola fa aumentare la velocità di rotazione del regolatore, per la forza centrifuga, le masse poste sulla valvola a farfalla si allontanano, alzandosi, qui la gravità oppone resistenza facendo chiudere la valvola essendo che le masse tendono ad avvicinarsi al suolo, facendone diminuire la velocità di rotazione. Tali automatismi sono compresi dalla teoria dei controlli automatici, che definisce l'azione di comando più efficace per ottenere il comportamento desiderato a seguito di una certa misurazione fisica.

Il **terzo passo** riguarda la gestione delle informazioni mediante sistemi combinatori/sequenziali che al verificarsi di determinate condizioni reagiscono con operazioni di base. La prima generazione di controllori prevedeva circuiti elettronici composti da bobine e relé, essi erano ingombranti e lenti nell'acquisizione delle informazioni, inoltre la loro logica era prestabilita e ridefinirla scaturiva una modifica sostanziale del circuito.

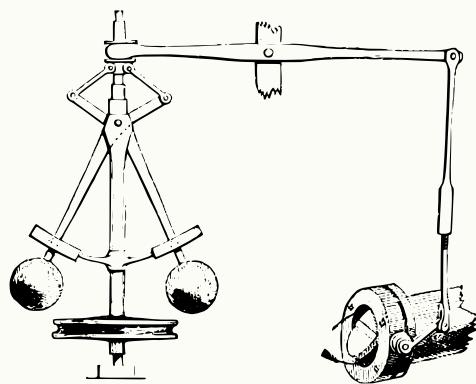


Figura 1.7: regolatore di Watt

Con l'avvento dei semiconduttori si è introdotta la seconda generazione di controllori, basati su schede elettroniche stampate, riducendo i costi ed aumentando l'efficienza, non risolvendo però il problema della bassa flessibilità, in quanto tali schede erano progettate per gestire una specifica logica.

La terza generazione di controllori vede i microprocessori protagonisti, grazie all'evoluzione dell'elettronica e dell'informatica sono ad oggi utilizzabili schede riprogrammabili (PLC) altamente flessibili, capaci di eseguire un generico algoritmo logico sequenziale.

Rivoluzione Industriale	Periodo Temporale	Tecnologie e Caratteristiche
prima	1785 – metà 19° secolo	utilizzo di macchine azionate da energia meccanica (vapore, acqua)
seconda	fine 19° secolo – 1970	azionamento elettrico delle macchine e produzione di massa basata sulla divisione del lavoro (catene di montaggio)
terza	1970 – oggi	utilizzo dell'elettronica e delle tecnologie dell'informazione (IT) per aumentare il livello di automazione di attività complesse (CNC, robot e computer)
quarta	oggi – futuro	sviluppo di macchine sensorizzate e intelligenti, interconnesse tra loro e con internet, con la raccolta, analisi e uso di grandi quantità di informazioni (Big data), per una specializzazione di massa del prodotto, l'integrazione della catena produttiva (supply and value chains) e una maggiore efficienza



1.2 Processi Industriali

I sistemi di produzione automatizzati sono composti da diverse componenti

- processo produttivo - movimentazioni meccaniche, attuazioni, trasformazioni fisiche e chimiche
- sistema di controllo - uno o più dispositivi messi in comunicazione con il processo produttivo, che possono agire su di esso riducendo l'intervento umano.
- impianto di produzione - macchinari, edifici, componenti

Si considerino i seguenti esempi di produzione industriale

- **produzione di energia elettrica**

- materie prime (input continuo) : combustibile fossile, ossigeno
- prodotto (output continuo) : energia elettrica misurata in Kilowatt/ore
- impianto necessario : tubature, caldaia, turbine, bruciatori, pompe, valvole, camini, edifici di sostegno e di contenimento, sensori

- **produzione di vernice**

- materie prime (input continuo) : resine, coloranti, acqua, additivi
- prodotto (output discreto) : barattoli di vernice
- impianto necessario : reattori (dove avvengono le reazioni principali), miscelatori, riscaldatori, tubature, pompe, valvole, edificio di sostegno e di contenimento, sensori

- **produzione di parti meccaniche di motori**

- materie prime (input discreto) : pezzo metallico grezzo
- prodotto (output discreto) : componente del motore
- impianto necessario : macchina con mandrini per la meccanica (fresatura, foratura, ...), sistema di controllo numerico (posizionamento corretto dell'utensile del mandrino), dispositivo di cambio utensile automatico, protezioni, sistemi di scarico trucioli

1.2.1 Sistema di Controllo

Il sistema di controllo interagisce con il processo attraverso *sensori* e *trasduttori*. Acquisiscono informazioni dal mondo fisico (pressione, temperatura) e le convertono in segnali facilmente analizzabili e controllabili (segnali elettrici).

Un esempio di sensore per misurare la forza, consiste in un circuito il cui resistore viene deformato quando rilevata una pressione sul sensore, variandone la resistenza.

Una volta raccolte le informazioni, è possibile cambiare le variabili di controllo del processo per ottenere il comportamento desiderato. Solitamente, il segnale di controllo è a bassa potenza, non sufficiente per correggere il comportamento di grossi attuatori, a tal proposito sono adoperati gli *amplificatori*. Le informazioni sono elaborate da un apposito calcolatore che può essere inglobato o esterno alla macchina in questione.

Nei sistemi di controllo è stata definita una normativa, nota come IEC61499 per normalizzare il funzionamento generale di tali dispositivi. In particolare, un sistema di controllo deve rifarsi ai seguenti punti

- è un sistema informatico che elabora informazioni ed esegue/applica algoritmi.
- è costituito da vari dispositivi che comunicano attraverso un'apposita rete.
- i dispositivi devono implementare delle funzionalità, denominate *applicazioni*.
- tali applicazioni possono essere distribuite fra i vari dispositivi.
- i dispositivi devono interfacciarsi con la rete e con il processo, inoltre le applicazioni costituiscono le loro *risorse*.
- in generale, una *risorsa* è costituita da
 - una o più applicazioni
 - funzioni che collegano dati ed eventi
 - funzioni di pianificazione delle attività (un sistema operativo)

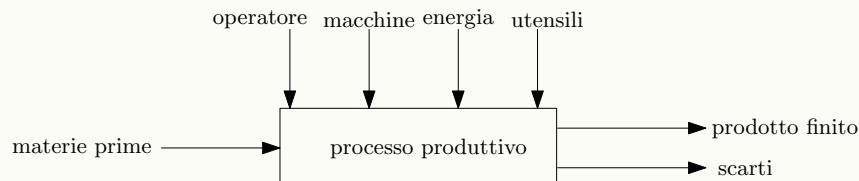
Si definisce **Manufacturing** l'insieme dei processi produttivi da applicare per ottenere un prodotto finale desiderato, a partire da materiali grezzi. Richiede



Figura 1.8: schematizzazione di un sistema di controllo

- energia
- macchine
- intervento umano
- informazioni

Da un punto di vista economico, è il processo che da valore aggiunto ai materiali utilizzati.



Spesso di questi processi produttivi ce ne sono molteplici messi a catena, in modo che l'output di un processo sia l'input di un altro. Si consideri il seguente esempio che schematizza un sistema di produzione del cemento. Il cemento viene prodotto a partire da calcari puri ed argilla, viene mandato nel frantocio per

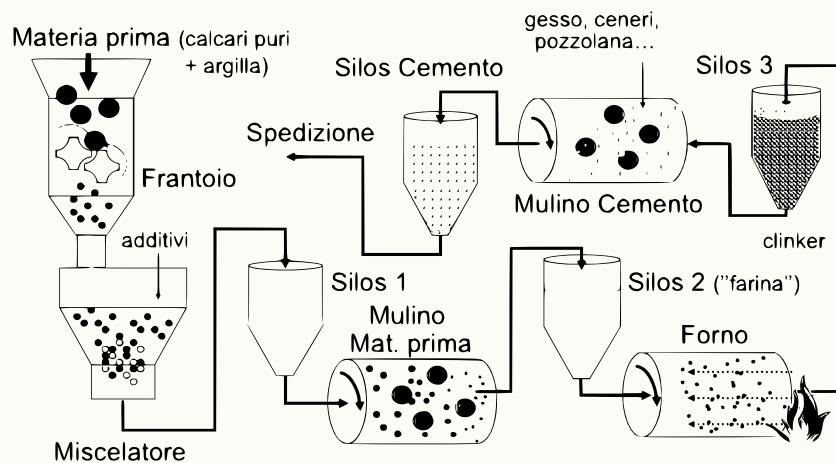


Figura 1.9: Cementificio : schema

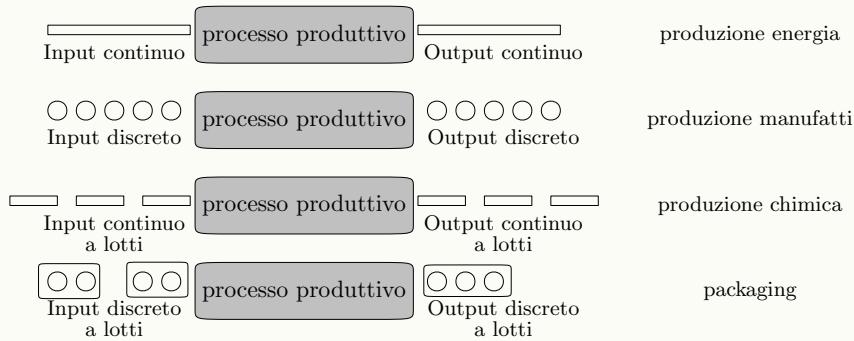
essere tritato, ciò che ne esce viene miscelato per poi venire accumulato in un silos. Quest'ultimo, funge da *buffer*, ossia un accumulo del prodotto non ancora terminato durante la sequenza di produzione, utile

nel sincronizzare la catena, qui viene riempito totalmente prima di passare allo step successivo. Il primo mulino esegue una fase di pre riscaldamento, il calore utilizzato deriva dall'energia termica scartata dal forno principale ad alte temperature.

Il processo consiste in una *sequenza di operazioni elementari* di varia natura

- lavorazione e assemblaggio
- trasporto e stoccaggio
- verifica, test, e coordinamento

Fin'ora è stata evidenziata la *numerabilità* delle materie prime o del prodotto finale di un certo processo, a tal proposito, è possibile classificare i processi produttivi nel seguente modo



- **processi continui** : Trattano la trasformazione continua nel tempo della materia, energia e quantità di moto. L'obiettivo è mantenere uniforme nel tempo la qualità del prodotto, gli scambi avvengono sulle variabili fisiche.
- **processi a lotti** : Possono essere sia continui che discreti, il lavoro finale necessita di una specifica, e finita, quantità di materia prima per essere prodotto. Segue una determinata sequenza di lavorazione detta *ricetta*, il processo si intorrempe fra un lotto ed un altro. Obiettivo dell'automazione è la definizione delle ricette, garantire un uso corretto delle risorse e realizzare tali sistemi in modo che siano ripetibili.
- **processi semi continui** : La produzione avviene in lotti o batch, con fasi di lavorazione continue intervallate da pause per il cambio di prodotto o la pulizia dell'impianto
- **processi discreti** : Si lavora su singoli prodotti, materiali numerabili, è una produzione tipica dell'industria manifatturiera.

I sistemi di controllo dei processi prevedono azioni di tipo *logico* oppure *diretto*

	Controllo logico	Controllo diretto
Processi continui	coordinamento complessivo avviamento e spegnimento guasti e emergenze	controlli primari (livelli, temperature, pressioni) controlli asserviti (portata pompe, posizione valvole)
Processi a lotti	controllo delle ricette supervisione impianto avviamento e spegnimento guasti e emergenze allocazione risorse impianto	controlli primari (livelli, temperature, pressioni) controlli asserviti (portata pompe, posizione valvole)
Processi discreti	controllo sequenze di lavoro delle singole macchine supervisione impianto avviamento e spegnimento guasti e emergenze	controlli asserviti (posizionamento, velocità motori elettrici)

- *regolazione* : Portare un segnale ad un determinato valore di riferimento

- **asservimento** : Far sì che il segnale segua una certa dinamica nel tempo

Il *controllo diretto* riguarda il rilevamento di grandezze fisiche analogiche da parte di sensori posti sul campo, tali valori continui nel tempo vengono convertiti in segnali digitali, confrontandoli poi con il valore di riferimento, dando l'azione di comando per correggere il comportamento delle variabili.

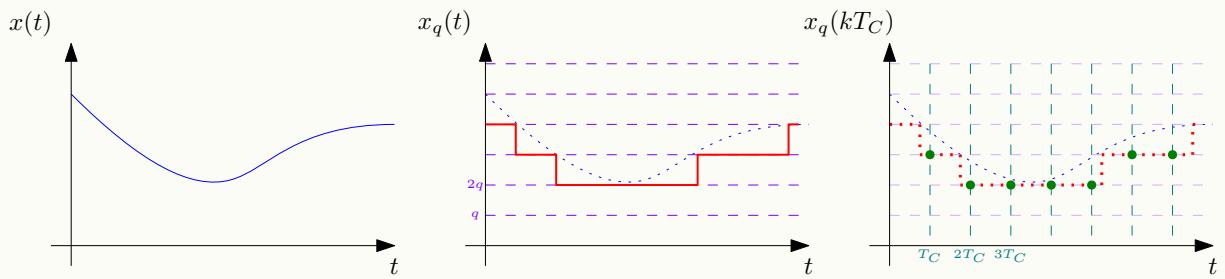
1. I sensori forniscono un segnale **analogico** $x(t)$ continuo nel tempo (temperatura, posizione)
2. Tale segnale viene **quantizzato**, denotato $x_q(t)$, costringendolo ad assumere un numero limitato di valori prestabiliti Δ , separati da un'ampiezza q , il numero di bit necessari a rappresentare i valori del segnale sarà $n = \log_2(\Delta/q)$

$$x_q(t) = \lfloor \frac{x(t)}{q} \rfloor \cdot q$$

3. Il segnale analogico quantizzato viene poi **campionato**, ossia, viene valutato esclusivamente in determinati intervalli di tempo separati da un tempo di campionamento T_C

$$x_q(t) \rightarrow x_q(kT_C) \quad t \in \mathbb{R} \quad k \in \mathbb{Z}$$

4. A tal punto si ha un **segnale digitale** codificato in numeri binari che può essere elaborato su un sistema di controllo digitale.



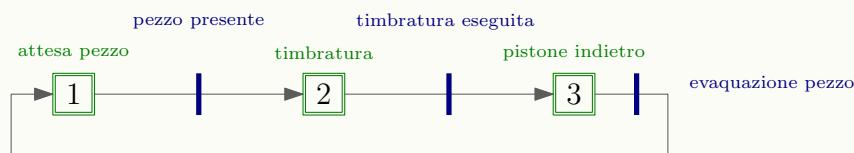
Le variabili logiche, ad esempio quelle booleane, assumono valori in un insieme numerabile, e su di esse è possibile eseguire operazioni logiche, si consideri il seguente esempio :

Il comando M è una variabile booleana che, se vera, aziona il moto di un motore elettrico. Il sensore di prossimità è descritto da una variabile booleana P che è vera se ci sono ostacoli nelle vicinanze, la variabile C descrive il consenso nel voler attivare il motore.

Risulta chiaro che

$$M = C \wedge \neg P$$

Il seguente SFC (Sequential Functional Chart) descrive il comportamento logico di una timbratrice automatica.



~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~

1.3 Analisi dei Sistemi di Produzione

Si considerino i sistemi di produzione manifatturiera, quindi, discreti. Essi possono seguire diverse strutture, fra le più importanti vi sono

- **linea di trasferta** : La produzione avviene secondo una specifica sequenza, rigida e non flessibile. Un esempio tipico è la classica catena di montaggio introdotta da Ford.

- **flow shop** : Vi sono diversi macchinari, e più flussi di produzione che si alternano i macchinari diversi.
- **job shop** : Ci sono differenti percorsi, intrecciati e complessi fra i diversi macchinari e reparti dell'impianto di produzione.
- **celle di produzione** : Le singole celle contengono più macchinari, ed ognuna di queste si occupa della produzione (dall'inizio alla fine) di un singolo prodotto, mantenendola concentrata nella cella, senza che essa sia dislocata nell'impianto.
- **FMS** : Diversi flussi fra le varie celle.

Vanno considerati vari aspetti nell'analisi dei sistemi di produzione, come la trattazione dei tempi di attesa, il *WIP - work in process*, è un termine utilizzato per indicare il numero di pezzi che vengono lavorati contemporaneamente.



Figura 1.10: sistemi di produzione

1.3.1 Linee di Trasferta

Verrà trattata in questa sezione la modellizzazione delle linee di trasferta, e verranno analizzati alcuni parametri che le caratterizzano. Una linea di trasferta consiste in una serie di macchine o stazioni (macchinari adatti a più compiti) connessi sequenzialmente da un sistema di trasporto. La sequenza di produzione è rigida ed i pezzi vengono processati continuamente nel tempo senza interruzioni. Le linee possono essere sincrone (i pezzi avanzano alla stessa velocità) oppure asincrone, tramite l'installazione di appositi buffer fra una macchina e l'altra.

- *pro* : La produzione avviene per un singolo prodotto, risulta molto efficiente, il WIP è ridotto ed il trasporto dei pezzi è semplice, dato che segue un'unica direzione. I tempi di avvio sono brevi.
- *contro* : La produzione è poco flessibile ed è a rischio obsolescenza, il malfunzionamento di una macchina può causare l'interruzione dell'intera linea (single point of failure).

Teorema (Legge di Little) : Data una linea di trasferta, ed i seguenti parametri

- p : tasso di produzione misurato in pezzi per unità di tempo
- T_a : tempo necessario per l'attraversamento di una linea
- WIP il work in process, il numero di pezzi che vengono lavorati contemporaneamente

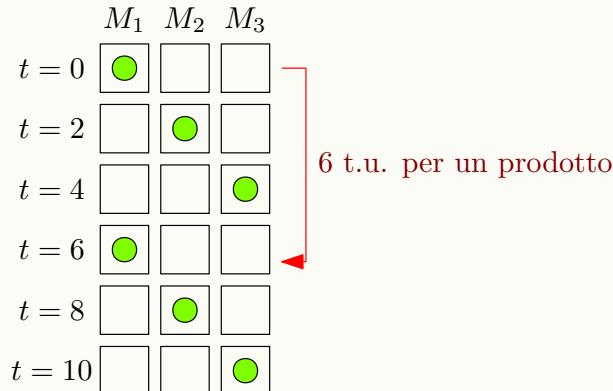
Vale la seguente relazione

$$WIP = p \cdot T_a$$

La legge si riferisce ad un sistema a regime permanente, quindi non considera l'avvio della linea, dato quindi un tasso p fisso, per ridurre il WIP è necessario ridurre il tempo necessario per l'attraversamento della linea.

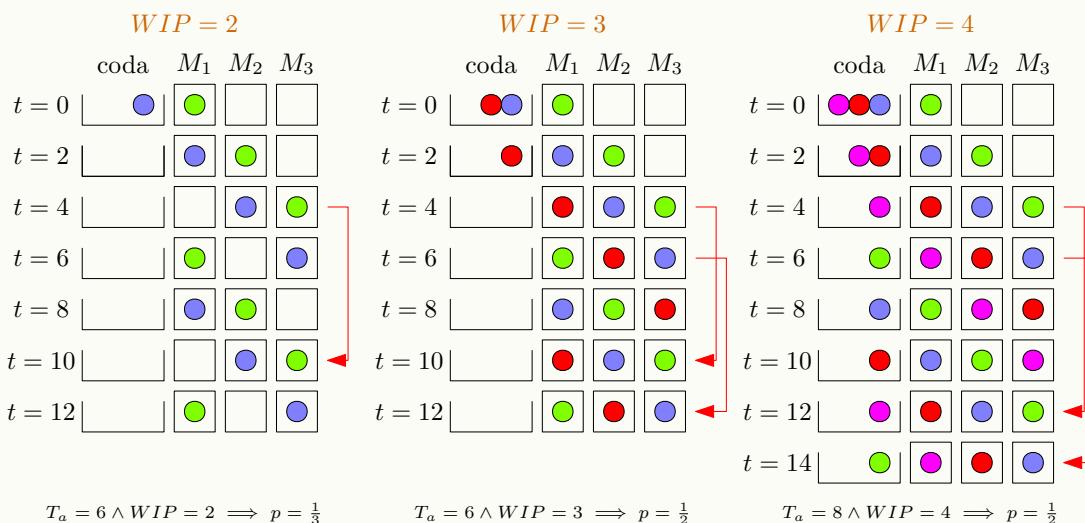
Esempio

Verrà considerata una linea di trasferta composta da 3 macchine M_1, M_2 e M_3 , i cui tempi di lavoro sono uguali e costanti $T_1 = T_2 = T_3 = 2$. Tale analisi è fatta a *WIP* costante.



In questo contesto, il WIP è 1, ciò significa che solamente un pezzo (indicato con una pallina verde) può occupare le 3 macchine contemporaneamente. Ogni 2 unità di tempo, il pezzo viene processato da una macchina all'altra, ci vogliono 6 unità di tempo per produrre un pezzo, $T_a = 6$.

$$T_a = 6 \wedge WIP = 1 \implies p = \frac{1}{6} \text{ un pezzo ogni 6 unità di tempo}$$



Si noti come nonostante il WIP aumenti, nel caso $WIP = 4$ il tasso di produzione rimane invariato, risulta quindi fondamentale trovare il valore di WIP ottimale che minimizzi T_a e massimizzi p .

L'esempio proposto in figura 1.11 è simile a quello precedente, ma in questo caso le macchine hanno tempi di lavoro differenti, precisamente, aumentano in ordine crescente.

Con **Dimensionamento** e **Bilanciamento** di una linea di trasferta, si intende la ricerca del compromesso ideale fra l'aumento della produttività (utilizzo di più macchine) e la riduzione dei costi, a parità di tempo totale di lavorazione. Le curve in figura 1.13, descrivono l'allocazione delle lavorazioni (distribuzione del carico) in un numero fissato di N stazioni/macchine.

- maggiore è il numero di stazioni utilizzate, minore sarà il carico medio delle stazioni e maggiore il costo unitario del singolo pezzo
- minore è il numero di stazioni utilizzate, maggiore sarà il carico medio delle stazioni e maggiore il costo del rischio di effettuare lavorazioni incomplete, a causa dell'elevata saturazione nell'impiego dei macchinari della stazione

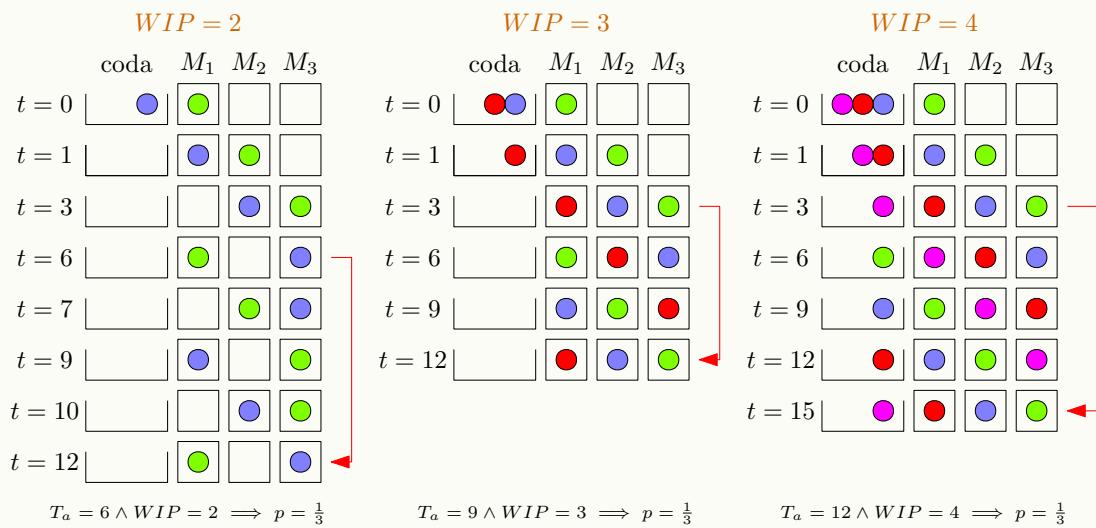
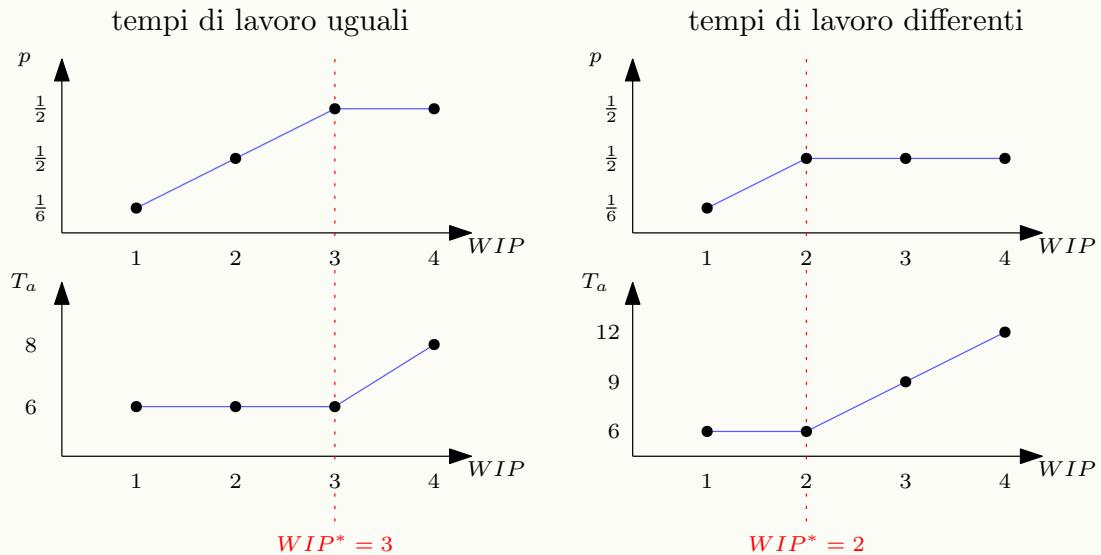
Figura 1.11: $T_1 = 1 \quad T_2 = 2 \quad T_3 = 3$ 

Figura 1.12: Analisi in funzione del WIP



Figura 1.13: Analisi in funzione del carico su una singola macchina

Il problema del bilanciamento di una linea di trasferta è NP-completo, per questo vengono utilizzate delle euristiche di soluzione, che garantiscono una soluzione prossima a quella ottimale.

Modello del Problema del Dimensionamento

Vi è una linea di trasferta con n macchine/stazioni, ogni macchina i -esima, ha un carico di lavoro C_i espresso in unità di tempo. La linea deve soddisfare un certo tasso di produzione p espresso in pezzi per unità di tempo. Il valore $CMT = \frac{1}{p}$ rappresenta il *carico massimo teorico*, è chiaro che se una qualsiasi stazione ha un carico $C_i > CMT$, allora il tasso di produzione p non è rispettato.

Un tasso $p = 7200 \frac{\text{pezzi}}{\text{mese}} = 10 \frac{\text{pezzi}}{\text{ora}}$ implica un carico di lavoro massimo pari a 6 minuti per pezzo.

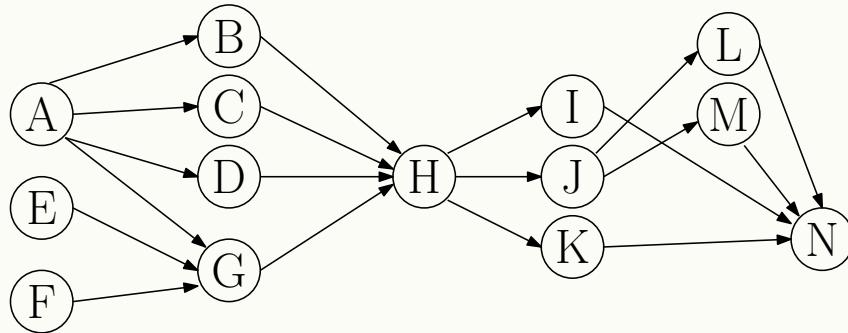
Sarebbe ideale inoltre ridurre il numero delle macchine n , dato che più macchine, equivale a dire un costo di manutenzione maggiore. Nella linea di trasferta, le varie lavorazioni da eseguire possono avere delle *dipendenze* (una lavorazione i può essere eseguita esclusivamente dopo una lavorazione j), si possono quindi rappresentare tali dipendenze attraverso un grafo orientato.

Dato quindi un insieme di lavorazioni, si vuole trovare

- una sequenza di produzione in cui il carico di lavoro per ogni macchina sia minore del carico di lavoro massimo teorico (ammissibilità)
- tale sequenza, deve rispettare le dipendenze (ammissibilità)
- che minimizzi il numero n di macchine/stazioni

Esempio

Lavorazione	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Tempo T_i in secondi	55	30	50	42	20	25	45	60	36	42	30	40	36	40
lavorazioni necessarie		A	A	A			A, E, F	B, C, D, G	H	H	H	J	J	I, K, L, M



La specifica del tasso di produzione è

$$p = 300 \frac{\text{pezzi}}{7h} = \frac{300}{7} \frac{\text{pezzi}}{h} = \frac{300}{3600 * 7} \frac{\text{pezzi}}{s} = \frac{1}{84} \frac{\text{pezzi}}{s}$$

Il carico massimo teorico è quindi

$$CMT = p^{-1} = 84 \text{ secondi a pezzo}$$

Essendo il tempo totale per la lavorazione di un pezzo uguale a 551 secondi, si ha che, al minimo, il numero di macchine presenti deve essere $n = \lceil T_{tot}/CMT \rceil = \lceil 551/84 \rceil = 7$. Per trovare una sequenza ammissibile ed ottimale, sarebbe necessaria una procedura non polinomiale, esistono quindi degli algoritmi che forniscono una soluzione ammissibile (quando possibile) tramite delle euristiche.

Ranked Positional Weight Technique

L'algoritmo consiste nell'assegnare ad ogni lavorazione i un peso PW_i , che consiste nel suo tempo necessario T_i sommato ai tempi di tutte le lavorazioni che dipendono da esso, direttamente ed indirettamente. In seguito, si ordinano le lavorazioni in base a tali pesi, in ordine decrescente, nell'esempio precedente si ha

Lavorazione	A	B	C	D	E	F	G	H	I	J	K	L	M	N
T_i	55	30	50	42	20	25	45	60	36	42	30	40	36	40
precedenti	B, C, D G..., N	H, I ..., N	H, I ..., N	H, I ..., N	G, H , I,... N	G, H , I,... N	H, I ...,N	I...,N	N	L, M, N	N	N	N	
PW_i	506	314	334	326	349	354	329	284	76	158	70	80	76	40

A tal punto si ordinano secondo PW_i :

$$A \rightarrow F \rightarrow E \rightarrow C \rightarrow G \rightarrow D \rightarrow B \rightarrow H \rightarrow J \rightarrow L \rightarrow I \rightarrow M \rightarrow K \rightarrow N$$

Si iniziano poi a considerare le lavorazioni in quest'ordine una dopo l'altra, sommandone i tempi necessari fino a ché sono minori di CMT , ad esempio

- Considero A, che ha tempo di lavorazione 55, lo assegno alla prima stazione.
◊ Carico della stazione 1 : $C_1 = 55 \leq CMT$
- Considero F, che ha tempo di lavorazione 25, lo assegno alla prima stazione.
◊ Carico della stazione 1 : $C_1 = 80 \leq CMT$
- Considero E, che ha tempo di lavorazione 20, non posso assegnarlo alla prima stazione, dato che il carico sarebbe maggiore di $CMT = 84$, quindi è necessaria una nuova stazione alla quale assegno E.
◊ Carico della stazione 2 : $C_2 = 20 \leq CMT$
- Considero C, che ha tempo di lavorazione 50, lo assegno alla seconda stazione.
◊ Carico della stazione 2 : $C_2 = 70 \leq CMT$
- ...e così via

Al termine, si avrà la seguente assegnazione

$CMT - C_i$	4	14	39	12	24	2	12	14
C_i	80	70	45	72	60	82	72	70
Lavorazioni	F	C	G	B	H	L	M	N
	A	E	D		J	I	K	
Stazione	1	2	3	4	5	6	7	8

Il termine $CMT - C_i$ è detto *sbilanciamento*, lo sbilanciamento medio equivale alla somma degli sbilanciamenti diviso il numero di stazioni, in questo caso $111/8 = 13.875$ secondi, si misura in percentuale rispetto il CMT , in questo caso, si ha uno sbilanciamento del 16.5% circa.

tempi lavorazioni	55	25	20	50	45	42	30	60	42	40	36	36	30	40		tempi lavorazioni sulle macchine morti	distanza da CMT = 84	
macchina																82 sec/pezzo		
1	A	F	ripete A, F su nuovi pezzi	80			A = 55	F = 25	2	4
2		E	C	ripete E, C su nuovi pezzi	70	a regime con		E = 20	C=50	12	14
3			G	45	avanzamento		G = 45	-	37	39
4				D	B	72	sincrono		D = 42	B = 30	10	12
5					H	60	della linea		H = 60	-	22	24
6						J	L	82	di trasferta		J = 42	L = 40	0	2
7							I	M	72	ogni 82 sec		I = 36	M = 36	10	12
8								K	N	70					K = 30	N = 40	12	14
tempo totale lavorazioni	551												tempo totale lavorazioni	8 x 82 = 656				
(effettivo sul primo pezzo)																		

1.3.2 Flow Shop

Un'altro tipo di struttura per un sistema industriale è il già citato flow shop, dove vi è la produzione di diversi prodotti, che devono seguire un determinato percorso fra le macchine presenti. Ogni prodotto deve vedere m lavorazioni, e differenti prodotti, possono passare per la stessa macchina, ma necessitando di tempi differenti, quindi, sarà assegnato un tempo di lavorazione ad ogni coppia (lavorazione, prodotto). Il tempo totale di completamento T_{max} , detto **makespan**, è il valore da minimizzare, possono essere presenti dei buffer nella struttura.

Il problema si complica rispetto la linea di trasferta, in questo caso però, date per assunte delle condizioni, vi è una regola che permette di stabilire una sequenza ottimale.

Teorema (Regola di Johnson) : Si considera l'assunzione per cui ci sono esclusivamente 2 macchine/stazioni, denotate M_1 e M_2 .

Sia $L = \{l_1, l_2, \dots, l_n\}$ un insieme di lavorazioni, i cui tempi per M_1 sono $\{t_{1_1}, t_{2_1}, \dots, t_{n_1}\}$, e i tempi per M_2 sono $\{t_{1_2}, t_{2_2}, \dots, t_{n_2}\}$. dove $\forall i, t_{i_1} > 0 \wedge t_{i_2} > 0$. Nella **soluzione ottimale**, la lavorazione l_i precede la lavorazione l_j se e solo se

$$\min(t_{i_1}, t_{j_1}) < \min(t_{j_1}, t_{i_2})$$

Tale teorema fornisce una procedura per trovare una soluzione ottimale, i passi sono i seguenti

1. si costruisce $S1 = \{l_i \in L \mid t_{i_1} < t_{i_2}\}$ (lavorazioni più rapide su M_1)
2. si costruisce $S2 = S1^C = \{l_i \in L \mid l_i \notin S1\}$ (lavorazioni più rapide su M_2)
3. Vengono schedulate le lavorazioni secondo la regola
 - prima si eseguono tutte le lavorazioni di $S1$ in ordine crescente secondo t_{i_1}
 - poi si eseguono tutte le lavorazioni di $S2$ in ordine decrescente secondo t_{i_2}

Esempio di Applicazione per 2 Macchine

Si hanno le seguenti lavorazioni

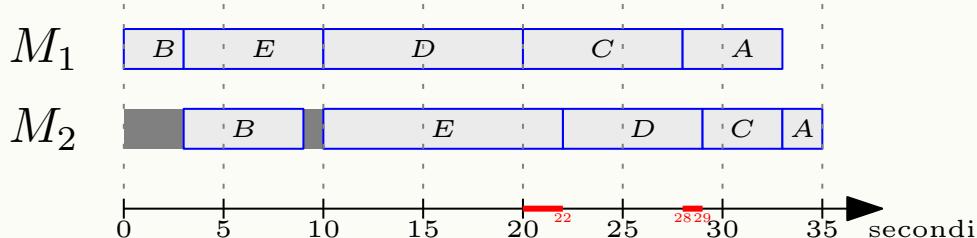
Lavorazione	A	B	C	D	E	T_{tot}
t_{i_1}	5	3	8	10	7	33
t_{i_2}	2	6	4	7	12	31

Si costruiscono i due insiemi

- $S1 = \{B, E\}$
- $S2 = \{A, C, D\}$

Si ordinano secondo i criteri della procedura ottenendo la sequenza

$$S^* = B \rightarrow E \rightarrow D \rightarrow C \rightarrow A$$



Gli intervalli rossi nella linea del tempo rappresentano il *tempo sprecato*, all'istante $t = 20$, la lavorazione D termina su M_1 , ed è pronta ad essere schedulata su M_2 , ma quest'ultima è ancora impegnata nella lavorazione di E, per questo il prodotto che ha appena terminato D dovrà attendere, sarà quindi necessaria la presenza di un buffer. Situazione analoga per la lavorazione C nell'istante 28.

Non ci sono attese sulla macchina 1, le attese sulla macchina 2 sono inevitabili, ma tale procedura le minimizza. La macchina 1 è costantemente carica, si dice che $T_{idle,1} = 0$ (non ci sono tempi di attesa). Differente è la situazione per la macchina 2 : $T_{idle,2} = 4$. Il tempo di lavorazione totale è $T_{max} = 35$.

Generalizzazione con 3 Macchine

Sotto ulteriori ipotesi, è possibile trovare una soluzione ottimale anche in un contesto con 3 macchine (o stazioni) M_1, M_2, M_3 , con relativi tempi di lavorazione $t_{i_1}, t_{i_2}, t_{i_3}$ con $i \in \{1, 2, \dots, n\}$ dove n = numero lavorazioni. Per poter trovare tale soluzione, è necessario che venga soddisfatta la seguente condizione

$$\max_{i \in \{1, 2, \dots, n\}} (t_{i_2}) \leq \min_{i \in \{1, 2, \dots, n\}} (t_{i_1}) \vee \max_{i \in \{1, 2, \dots, n\}} (t_{i_2}) \leq \min_{i \in \{1, 2, \dots, n\}} (t_{i_3})$$

Se tale condizioni è avverata, è possibile considerare una struttura flow shop a due macchine, precisamente M'_1, M'_2 , con le medesime lavorazioni, la cui soluzione è identica alla soluzione del sistema originale con 3 macchine.

Nel nuovo sistema, i nuovi tempi di lavorazione t'_{i_1}, t'_{i_2} vanno definiti come segue

- $t'_{i_1} = t_{i_1} + t_{i_2}$
- $t'_{i_2} = t_{i_2} + t_{i_3}$

Esempio di Applicazione per 3 Macchine

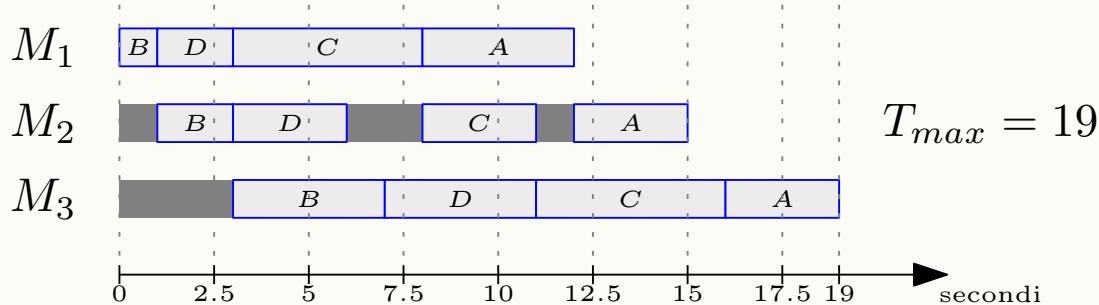
Si hanno le seguenti lavorazioni

Lavorazione	A	B	C	D
t_{i_1}	4	1	5	2
t_{i_2}	3	2	3	3
t_{i_3}	3	4	5	3

La condizione $\max_{i \in \{1, 2, \dots, n\}} (t_{i_2}) \leq \min_{i \in \{1, 2, \dots, n\}} (t_{i_3})$ è soddisfatta, quindi è possibile ridurre il sistema ad uno equivalente

Lavorazione	A	B	C	D
t'_{i_1}	7	3	8	5
t'_{i_2}	6	6	8	7

Si applica la regola di Johnson, trovando la sequenza $S^* = B \rightarrow D \rightarrow C \rightarrow A$, che verrà applicata al sistema originale con 3 macchine.



Un'altra struttura è la **produzione per reparti**, già accennata con il nome di **job shop**, in cui l'impianto è suddiviso in reparti contenenti stazioni, ed i vari prodotti devono essere passati fra i reparti attraverso sistemi di trasporto che seguono un percorso prefissato, è quindi necessario considerare il routing dei prodotti fra i reparti. Il modello job shop è caratterizzato da

- vaste categorie di prodotti, che subiscono lavorazioni con sequenze differenti
- operazioni non ripetitive ed alta flessibilità
- un maggiore work in process, i flussi lavorativi sono intricati
- elevati tempi di attraversamento, e qualità dei prodotti non sempre omogenea



Figura 1.14: job shop e celle di produzione

Quando è possibile individuare famiglie di prodotti simili con cicli di lavorazione omogenei, si identificano delle celle contenenti gruppi di macchine adatti a tale famiglia di prodotti, tale **produzione per celle** semplifica i flussi di produzione, il trasporto e la gestione, le varie celle sono indipendenti fra loro.

Quando si implementa il trasporto automatico ed un controllo tramite calcolatore nelle celle di produzione si parla di **Flexible Manufacturing/Assembly Systems (FMS/FAS)**, sono sistemi dotati di elevata flessibilità riguardo alle diverse sequenze delle lavorazioni e/o all'assegnamento di operazioni alle risorse. Sono caratterizzati da

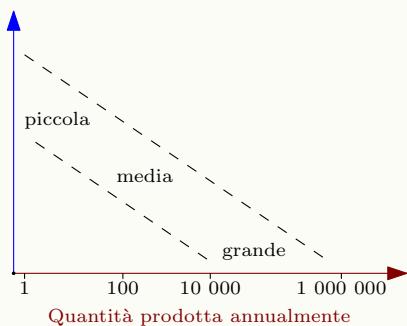
- diversi prodotti
- lavorazioni eseguite su più macchine
- assegnazione delle risorse tramite routing dei flussi
- problemi di sequenziamento locale dell'impiego delle risorse
- alto grado di automazione

È possibile individuare tre tipi di automazione industriale

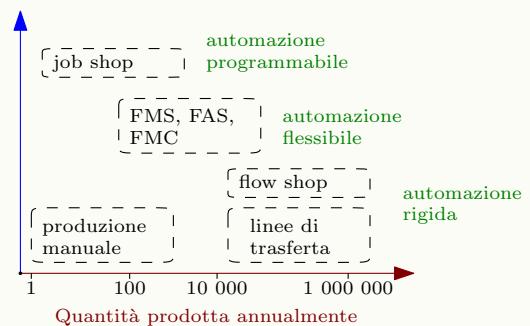
- *Automazione rigida* : La sequenza delle operazioni è fissata, sono sistemi poco flessibili (come le linee di trasferta) destinati a grandi produzioni con poca varietà di prodotto.
- *Automazione programmabile* : Si può cambiare la configurazione del flusso di produzione in modo da variare il prodotto, tipico delle industrie con produzione discreta a lotti. Vi è un tempo di attesa fra un lotto ed un altro per la riconfigurazione del flusso.
- *Automazione flessibile* : estensione dell'automazione programmabile in cui è possibile diversificare la produzione senza avere tempi morti di conversione dell'impianto, i macchinari utilizzati sono caratterizzati da un alta configurabilità.

In generale, per i sistemi di automazione, esiste una correlazione qualitativa fra la quantità di prodotto fabbricata, ed il numero di prodotti distinti.

Varietà dei prodotti



Varietà dei prodotti





1.4 Sistema di Supporto

Un sistema di supporto consiste in un insieme di attività legate all'impianto di produzione, in particolare

- attività di business, quali gestione degli ordini, marketing del prodotto e considerazioni economiche
- progettazione del prodotto sulla base delle esigenze del mercato
- pianificazione ("planning") della produzione, definizione della sequenza di lavorazione, politiche di stoccaggio e rifornimento
- controllo delle attività (gestione e supervisione)

Tali attività si re-iterano in un ciclo di produzione, cosiddetto "agile".



In supporto delle attività di business vi sono

- **Enterprise Resource Planning (ERP)** : Un insieme di software volti all'automazione dell'amministrazione, della logistica, della produzione e delle risorse umane.
- **Decision Support System (DSS)** : Sistemi che hanno lo scopo, dato un insieme di dati e dei vincoli da rispettare, di migliorare il processo decisionale nell'impianto.

Inoltre le macchine sul campo vengono supportate da sensori che hanno lo scopo di raccogliere dati ed informazioni in maniera massiva, in modo da poter essere utilizzati per i modelli di intelligenza artificiale e machine learning, sfruttando appunto, tali dati per effettuare previsioni con maggiore precisione.

In supporto alle attività di progettazione vi sono

- **Computer Aided Design (CAD)** : Un insieme di strumenti informatici utili nella progettazione.
- **Computer Aided Engineering (CAE)** : Un insieme di strumenti informatici utili nella verifica delle funzionalità tramite simulazioni e modelli matematici.
- **Computer Aided Manufacturing (CAM)** : Un insieme di software utili nell'automatizzazione ed organizzazione delle sequenze di operazioni nella produzione.
- **Computer Aided Process Planning (CAPP)** : software che permette di automatizzare/ottimizzare il planning della produzione

È possibile utilizzare un modello CAD per ottenere un programma da eseguire su una CNC (macchina a controllo numerico)

- Caricamento di un modello CAD
- Impostazione del sistema di coordinate
- Impostazione di parametri
- Generazione delle istruzioni per la macchina in questione
- Invio dei dati al controllo numerico



1.5 La Piramide CIM

Quello della piramide CIM è un modello teorico che descrive la struttura di un processo produttivo, integrata con i sistemi di automazione ed i sistemi informatici gestionali, tale modello propone diversi vantaggi

- miglioramento della qualità di produzione
- tempi e costi ridotti
- aumento della flessibilità
- diminuzione degli scarti e manutenzione predittiva
- fondamentale per conformarsi a leggi e regolamenti sulla sicurezza del processo produttivo, sulla qualità del prodotto e sulla riduzione dell'impatto ambientale

Il modello CIM è gerarchico e suddivide il processo di produzione su vari livelli, i cui elementi comunicano orizzontalmente con quelli adiacenti. Ogni livello assume che l'automazione nel livello inferiore sia ideale, nessun livello è privilegiato rispetto ad un altro.



Dai livelli superiori le informazioni sono più elaborate e di minore quantità, ed i comandi vengono dati con minore frequenza, nei livelli inferiori, ci sono più informazioni, ma sono semplici, ed i comandi vengono dati con maggiore frequenza. La comunicazione verticale è privilegiata, se più dispositivi di campo devono comunicare, devono farlo attraverso il sistema di controllo della macchina di cui fanno parte.

Al livello di macchina e campo il controllo può essere sia *logico* che *diretto* (real time), attuato da microprocessori, con il salire dei livelli nella piramide il controllo assume sempre più un carattere logico, dettato da eventi.

1.5.1 Livelli della Piramide

Livello di Campo

Tale livello contiene dispositivi quali sensori ed attuatori, in generale, i componenti hardware che eseguono le attività di produzione e controllo. Tali dispositivi interfacciano il livello superiore al livello fisico tramite segnali di ingresso ed uscita, possono inoltre comunicare informazioni sul proprio stato (auto diagnosi), oltre a ciò, sono generalmente semplici.

Sono raggruppati in semplici sistemi di controllo e i livelli superiori li assumono come dispositivi ideali, sono attrezzati di apposito hardware di controllo, quali sistemi digitali a microprocessori/sistemi embedded.



Livello di Macchina

Una macchina raggruppa più dispositivi di campo volti al fornire una determinata funzionalità. Ad esempio, diversi motori elettrici (dispositivi di campo) possono costituire un braccio robotico (macchina) volto allo spostamento di oggetti.

Tali componenti vengono controllati tramite dei controllori logici programmabili (PLC), tramite la regolazione di variabili analogiche e la realizzazione di sequenze di operazioni.

Esempio: a livello di campo si controllano le posizioni dei singoli giunti; a livello di macchina viene pianificato il movimento del robot nello spazio operativo e la sequenza delle azioni che deve effettuare.

Livello di Cella

Diverse macchine vengono raggruppate insieme formando delle celle di produzione, atte a produrre prodotti, anche diversi ma tecnologicamente affini, le macchine sono interconnesse fisicamente da un sistema locale di trasporto e stoccaggio. Nonostante il vincolo real time sia presente, il livello di controllo è quasi esclusivamente logico, e coordinano le macchine della cella.

Livello di Stabilimento

Racchiude più celle e linee produttive di un impianto industriale, riceve istruzioni dal livello gestionale e le attua sotto forma di piani operativi per la produzione. I sistemi di controllo a tale livello sono denominati *SCADA* (*Supervisory Control And Data Acquisition*), tipicamente implementati in workstation. Sono sistemi di controllo che dispongono di:

- Un interfaccia con l'operatore, assente nei microcontrollori e nei PLC
- Gestione di allarmi e ricette
- Programmazione dei lavori e basi di dati del processo produttivo
- Controllo statistico e supporto alla manutenzione, potendo fare previsioni e rapporti



Figura 1.15: sala di controllo di una centrale elettrica

Livello di Azienda

Si occupa dei processi gestionali, il sistema non è di controllo ma *decisionale*, costituito da un infrastruttura software connessa al mainframe aziendale, non esistono vincoli di tipo temporale.

Esiste uno standard (ANSI/ISA-S88.01-1995) che normalizza lo schema generale di controllo su 3 livelli gerarchici.

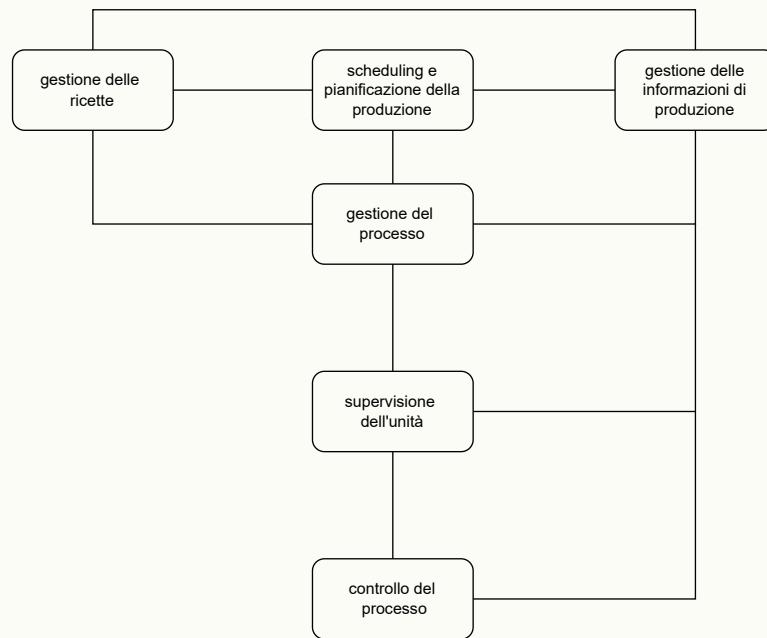
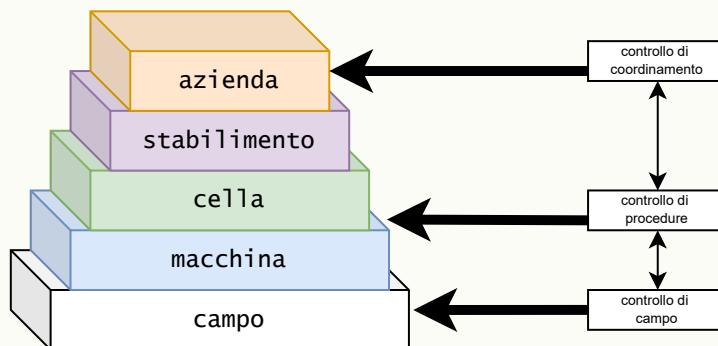


Figura 1.16: modello delle attività di controllo

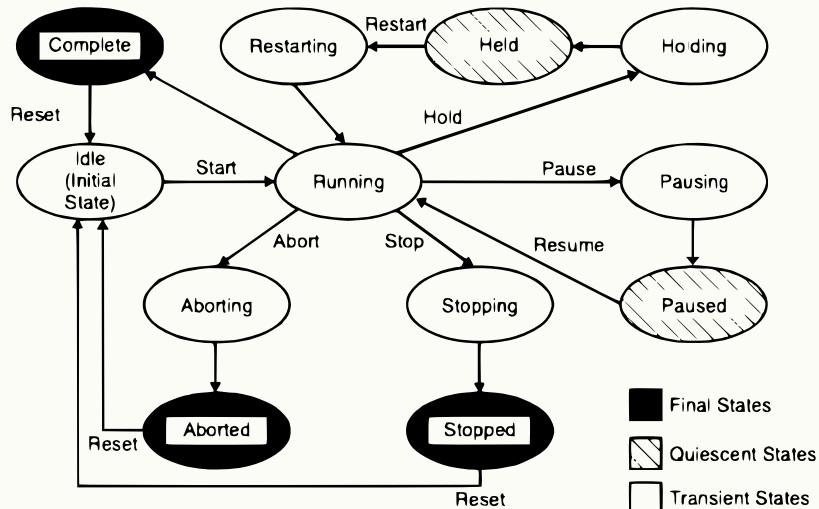
- **controllo di campo** : controllo che agisce sui dispositivi di campo, in particolare, su variabili continue, ed è implementato su dispositivi dedicati. Le informazioni sono semplici e trasmesse ad alta frequenza.
- **controllo di procedure** : riguarda il controllo dei livelli di macchina e cella della piramide, riguarda gruppi strutturati di componenti di campo, a tale livello vengono svolte funzioni di auto-diagnosica, è implementato su schede dedicate o anche su pc industriali. Prevede algoritmi più complessi di quelli del controllo di campo. Il controllo è
 - *diretto* : riguarda il controllo di gruppi di variabili continue o funzioni più avanzate
 - *logico* : coordinamento dei sistemi di campo sulla base della lista di operazioni sequenziali
- **controllo di coordinamento** : si pone al livello dello stabilimento ed opera su dati strutturati a bassa frequenza, riguarda il coordinamento e la gestione delle varie celle di produzione, vincoli temporali molto laschi.



Lo standard definisce anche un diagramma logico delle transizioni tra stati di un processo. Si noti come i due stati *Held* e *Paused* sono differenti:

- *Paused* : è scaturita da un operatore esterno (esempio : causa operazioni di verifica)

- *Held* : il processo è in attesa di un evento e viene bloccato



1.5.2 Auto-diagnostica

I sistemi di controllo devono avere la possibilità di generare segnali ausiliari che danno allarme di eventuali guasti nei dispositivi soggetti ad usura. Occorre individuare dei segnali dal dispositivo che possono essere valutati secondo appropriati modelli matematici.

Con *signature*, si definisce una firma del guasto, ossia una sua manifestazione sottoforma di segnale, un esempio può essere la misura tramite un accelerometro di vibrazioni in un sistema meccanico. Si sviluppano algoritmi atti all'analisi dei segnali (basati ad esempio sulla trasformata di Fourier).

Definiamo *fault* i guasti, e li denotiamo f , su di essi si formulano

- *fault detection* : rilevamento dei guasti sul sistema fisico o software
- *fault isolation* : classificazione del guasto, e distinzione di esso rispetto ad altri
- *fault identification* : determinazione del profilo temporale del guasto f
- *fault accommodation* : riconfigurazione a seguito del guasto, coinvolge modifiche hardware o software

Rilevazione ed Isolazione

Quando un sistema dinamico è soggetto ad un possibile guasto, si definisce un sistema ausiliario detto **generatore di residuo**, il cui segnale in uscita dipenderà dalla presenza di un particolare guasto f .

Un residuo r dipende da un guasto f

Nel caso in cui possono esserci più guasti, si definisce un vettore \bar{r} di residui, in cui r_i dipende da f_i . Tipicamente il residuo riproduce approssimativamente l'evoluzione temporale del guasto. Con FDI si intende l'operazione di "Fault Detection Isolation", queste ultime due possono avvenire in contemporanea, vi sono differenti modelli proposti

- *model-based* : si definisce un sistema dinamico che rappresenta il guasto (generatore di residuo)
- *signal-based* : si utilizzano esclusivamente i segnali provenienti da misure del sistema, la loro elaborazione può far luce su eventuali guasti
- *ibrido* : si combinano entrambe le tecniche

Esempio di Residuo

Vi è un processo (soggetto a guasto) modellato dal seguente sistema dinamico

$$\begin{cases} \dot{x} = ax + bu + ef \\ y = cx \end{cases}$$

Dove f è il generico fault. Il residuo r viene implementato come osservatore di un disturbo, o di un segnale non noto (che in questo caso è f). Rappresenta proprio la stima del disturbo al tempo t .

$$r(t) = \frac{k}{e} \left[\frac{y(t)}{c} - \int_0^t [ax(\tau) + bu(\tau) + er(\tau)d\tau] \right] \quad \text{con } \begin{cases} k > 0 \\ r(0) = 0 \end{cases}$$

L'integrale indica che la stima del disturbo viene calcolata tenendo conto di tutti i valori passati del disturbo stesso.

Il parametro k è un guadagno che determina quanto "velocemente" l'osservatore reagisce ai cambiamenti del disturbo. Un valore di k più alto significa una risposta più rapida, ma può anche introdurre più rumore nella stima. L'evoluzione nel tempo del residuo è data da

$$\dot{r} = \frac{k}{e} \left[\frac{\dot{y}}{c} - [ax + bu + er] \right] = \quad (1.1)$$

$$\frac{k}{e} \left[\frac{c[ax + bu + ef]}{c} - [ax + bu + er] \right] = \quad (1.2)$$

$$\frac{k}{e} e[f - r] = k[f - r] \quad (1.3)$$

Si ha quindi l'equazione lineare del primo ordine

$$kr + \dot{r} - kf = 0$$

Si porta nel dominio di Laplace

$$\begin{aligned} \mathcal{L}[kr + \dot{r} - kf](s) &= \\ k\mathcal{L}[r](s) + \mathcal{L}[\dot{r}](s) - k\mathcal{L}[f](s) &= \\ k\mathcal{L}[r](s) + s\mathcal{L}[r](s) - r(0) - k\mathcal{L}[f](s) &= \end{aligned}$$

Si ricordi come $r(0) = 0$. Denoto $\mathcal{L}[f] = F$ e $\mathcal{L}[r] = R$

$$\begin{aligned} kR(s) + sR(s) - kF(s) &= 0 \implies \\ R(s)[k + s] &= kF(s) \\ R(s) &= \frac{kF(s)}{k + s} \implies \\ \frac{R(s)}{F(s)} &= \frac{k}{k + s} = \frac{1}{1 + \frac{1}{k}s} \end{aligned}$$

Nel caso il guasto fosse costante $f(t) = f_0$, nel dominio del tempo si avrebbe

$$r(t) = f_0[1 - e^{-kt}]$$

1.5.3 Architetture per il Controllo

Il controllo è gestito da dispositivi elettronici ed informatici, è possibile individuare 3 tipi di sistemi

- **controllori embedded** - per il controllo di campo
- **architettura a bus** - per il controllo di procedure
- **PC-based**

Sistemi Embedded

Tali sistemi di controllo sono "fusi" alle funzionalità del dispositivo, e sono adoperati per un unico, specifico e predeterminato compito. Sono piattaforme hardware create "ad hoc", e la loro progettazione dipende dalla conoscenza dei compiti che dovranno svolgere. La realizzazione può avvenire in due modi

- singolo chip integrato (microcontrollori)
- singola scheda (DSP o FPGA)

L'hardware è semplice, vi è una CPU per gli algoritmi, una memoria per i dati ed i programmi e dei circuiti per gestire input ed output (analogici o digitali, con eventuali convertitori). Hanno un software integrato molto semplice ed è orientato all'automazione, vi è una gestione a basso livello delle risorse e delle comunicazioni.

Microcontrollori

I microcontrollori (a singolo chip) sono nati a seguito della miniaturizzazione dell'elettronica, sono utilizzati in una grandissima varietà di applicazioni e possono essere integrati con un sistema di sviluppo per la loro programmazione.

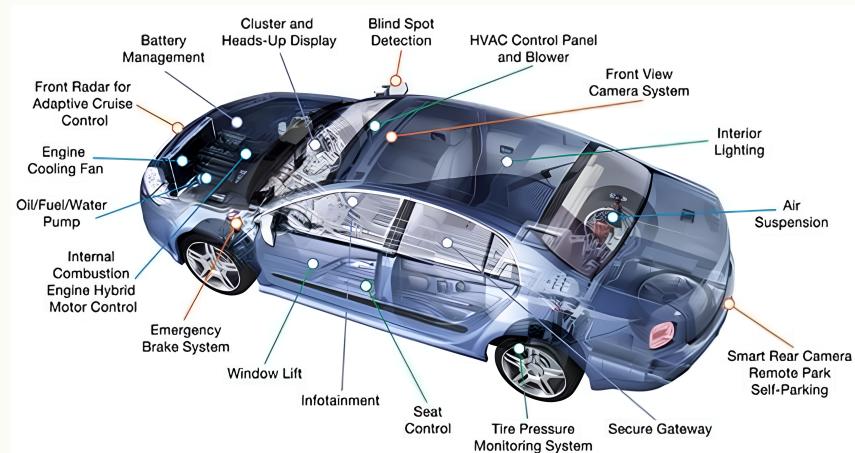


Figura 1.17: Sistemi di controllo embedded in automotive

Caratteristiche ed impieghi :

- applicazioni semplici
- basso consumo e poco spazio occupato
- numero limitato di segnali da gestire. Interfaccia utente assente
- scarsa integrazione con dispositivi simili, difficile espansione

Singola Scheda

Prevedono più componenti standard integrati su una singola scheda, fra questi vi sono

- DSP (Digital Signal Processor) : processori atti al trattamento dei segnali ed esecuzione di funzioni su numeri interi o reali
- FPGA (Field Programmable Gate Array) : circuiti integrati riconfigurabili dall'utente per realizzare funzioni logiche complesse tramite blocchi logici e elementi di memoria

Rispetto ai microcontrollori hanno una maggiore capacità di elaborazione e possono gestire un maggior numero di segnali.

Ricapitolando, i sistemi di controllo embedded prevedono i seguenti:

- **PRO**

- combinazione hardware/software specificamente studiata
- ottimizzazione spaziale e di complessità
- minori ingombri, basso consumo
- minori costi

- **CONTRO**

- interfaccia uomo-macchina poco evoluta
- gestione di un numero limitato di segnali in input/output
- costi di progettazione (hardware e software) non irrilevanti
- poca flessibilità: modifiche ai compiti da svolgere possono rendere necessaria la progettazione di un nuovo dispositivo

Utili quando i compiti di controllo sono noti a priori.

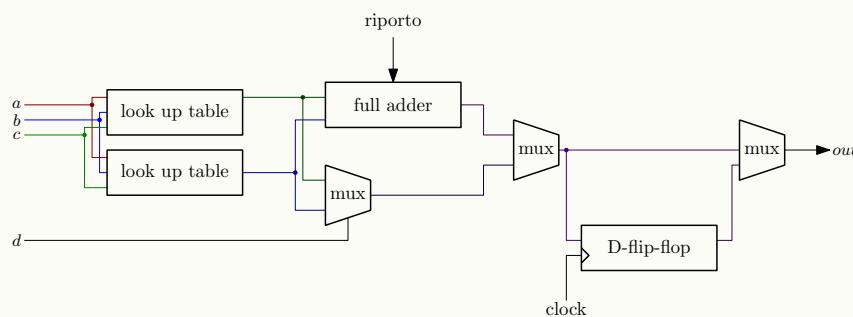


Figura 1.18: cella logica elementare in un FPGA

Architettura a Bus

Con *bus* si intende una linea elettrica che mette in comunicazione più dispositivi, vi è una scheda madre con un bus a cui si connettono più schede, in modo da rendere modulare il sistema. Un bus è composto da diverse linee

- linee dati
- linee indirizzi
- linee di alimentazione
- linee per la comunicazione

- **PRO**

- flessibilità di progettazione
- scelta dei moduli secondo le funzionalità da implementare

- **CONTRO**

- Sistema Operativo più complesso
- gestione dei moduli interconnessi e delle comunicazioni
- vincoli real-time



Figura 1.19: Modularità dell'architettura a bus

PC-Based

Ultimamente, anche i comuni computer (corazzati per resistere alle condizioni sfavorevoli degli impianti) sono adoperati come sistemi di controllo, sono sistemi informatici con architettura a bus ed implementano una semplice interfaccia uomo macchina, e sono semplici da connettere alle reti informatiche.

Per far sì che risultino efficaci come sistemi di controllo, esistono appositi sistemi operativi atti all'utilizzo real time (ad esempio, RTAI-Linux). Implementano moduli/schede per l'interconnessione con un elevato numero di segnali input/output, inoltre, esistono appositi protocolli per la comunicazione fra PC e PLC.



1.6 Industria 4.0

La seconda rivoluzione industriale, agli inizi del XX° secolo ha introdotto la produzione di massa e la catena di montaggio classica (Ford), nei primi anni 70 l'avvento dei computer e dei robot ha sondato la terza rivoluzione industriale, ad oggi vige l'industria 4.0 in cui le tecnologie ICT (Information and Communication Technology) pervadono i processi produttivi. Anche la presenza di dispositivi intelligenti (IOT) capaci di fornire dati sulla rete tramite sensori e comunicare in tempo reale ha modificato il modo di produrre.

Industria 4.0:

“the comprehensive transformation of the whole sphere of industrial production through the merging of digital technology and the internet with conventional industry”

(Angela Merkel - Organization for Economic Cooperation and Development, 19 Febbraio 2014)

L'avvento di tali tecnologie ha portato un cambiamento industriale ma anche sociale, e sono diventate necessarie diverse figure professionali che richiedono abilità trasversali per gestire le nuove tecnologie. Definiamo le seguenti, **tecnologie abilitanti**

- *Soluzioni manifatturiere avanzate* : utilizzo dei robot interconnessi e riprogrammabili nell'automazione di attività precedentemente non automatizzabili.
- *Additive manufacturing* : produzione per "addizione" di materiale, resa possibile dalle stampanti 3D connesse ai sistemi CAD.
- *Realtà aumentata* : l'uso della realtà aumentata e della realtà virtuale ha permesso la simulazione di processi fisici a supporto dei processi produttivi.
- *Simulazione* : Simulazione in tempo reale del sistema di produzione in grado di migliorare il processo decisionale e predittivo tramite l'uso dei digital twin.
- *Integrazione orizzontale/verticale* : Integrazione di informazioni lungo la catena del valore, dal consumatore al fornitore.

- *Internet industriale* : Comunicazione multidirezionale diffusa tra processi produttivi e prodotti.
- *Cloud* : Uso del cloud per la raccolta dei dati in maniera distribuita.
- *Sicurezza* : Metodologie di difesa alle minacce informatiche alla quale sono soggetti i sistemi aperti ed interconnessi.
- *Big data* : Uso dei modelli voltati all'analisi statistica, che sfruttano le grandi quantità di dati disponibili allo scopo di estrarne informazioni.

L'uso delle tecnologie abilitanti porta vari benefici

- Flessibilità nella produzione di piccoli lotti ai costi della grande scala
- Maggiore velocità della progettazione e produzione dei prodotti
- Maggiore produttività e riduzione dei tempi morti (tempi di setup, errori e fermi macchina)
- Migliore qualità del prodotto e riduzione degli scarti grazie a sensori che monitorano la produzione in tempo reale

I sistemi ICT hanno permesso la creazione di modelli di business favoriti e definiti da apposite linee guida di sviluppo.

1.6.1 Tecnologie Abilitanti

I **collaborative robots**, o più semplicemente *cobots*, permettono la collaborazione fisica fra robot ed operatori rimuovendo le barriere dell'area di lavoro, questi ultimi entrano in contatto diretto, sono sensibili all'ambiente circostante grazie a sensori laser e sistemi di visione, ed appositi sistemi di controllo programmati per il riconoscimento e rilevamento di possibili urti con gli operatori. Presentano spesso strutture leggere e sono dotati di cedevolezza nei giunti, in grado di assorbire energia da urti esterni. Lo spazio di lavoro condiviso apre le possibilità a nuove applicazioni.



Figura 1.20: Robot collaborativo

L'**additive manufacturing** permette la produzione di oggetti tridimensionali di varia forma a partire da un modello digitale CAD, la loro produzione richiede il materiale strettamente necessario per la stampa diminuendo i residui.

I **digital twin** definiscono la programmazione e previsione dei sistemi tramite simulazioni digitali, utili durante la progettazione della linea di produzione, inoltre, in fase di esecuzione di essa è possibile eseguire in contemporanea il modello digitale in modo da supervisionare il processo.

Le tecniche di **machine learning** permettono il processamento di informazioni al fine di migliorare la qualità del prodotto grazie alla manutenzione predittiva.

L'**IOT** (Internet Of Things) riguarda i dispositivi capaci di rilevare e processare dati dal mondo fisico



Figura 1.21: Uso della stampa 3D

(temperatura, illuminazione, umidità, ...) ha applicazioni di notevole impatto, ad esempio, in ambito medico, tali dispositivi condividono informazioni e comunicano con gli altri dispositivi e macchine connesse in rete.



Figura 1.22: Un'illustrazione di come questa rivoluzione nella medicina apparirà in un tipico ospedale che fa uso di IOT

Le diverse tecnologie vengono incanalate in determinate linee guida di sviluppo, i sistemi distribuiti che permettono la circolazione rapida di informazioni sono combinati ad una centralizzazione della gestione di essi, garantendo la possibilità di controllare e monitorare in modo efficiente dati e risorse da e in qualsiasi parte dell'azienda.

Nonostante le macchine moderne siano indipendenti da un intervento continuo, devono essere supervisionate da un operatore umano, quest'ultimo deve avere la possibilità di interagire con esse in maniera intuitiva, la branca che si occupa di rendere semplice tale operazione è denominata *interazione uomo-macchina*.

1.6.2 Modelli di Business

2 aspetti sono fondamentali nell'industria 4.0

- Qualità delle aziende migliorata grazie all'uso delle nuove tecnologie
- Nuovi schemi di competizione del mercato basati sui nuovi modelli di business

Tali modelli di business si sono sviluppati solo dopo l'avvento delle tecnologie abilitanti che li hanno resi realizzabili.

- **Centralità del cliente** : al centro della catena del valore dell'industria, c'è il consumatore, e le aziende puntano a soddisfare le richieste di quest'ultimo, talvolta anticipandone le necessità e le richieste di servizio.
- **Economia circolare** : risponde alla necessità di passare ad un modello circolare di produzione, in cui i prodotti ed i processi manifatturieri vanno improntati al riutilizzo.

progetta → usa → ricicla → riusa

Tale modello porta un risparmio dovuto alla riduzione degli scarti e ai minori costi di approvvigionamento, nonché una notevole riduzione dell'impatto ambientale.

- **Strategie basate su ICT** : emergono nuove strategie di mercato che hanno lo scopo di portare il prodotto più vicino al consumatore, le tecnologie alla base dell'industria 4.0 mettono a disposizione una grande mole di dati utili al miglioramento dell'efficienza dell'azienda.
 - negozi online
 - servizi ad abbonamento
 - affitto di infrastrutture informatiche
- **Economia della condivisione** : modelli di condivisione di beni e servizi tramite piattaforme digitali, come i servizi di car sharing o bike sharing, irrealizzabili senza un'infrastruttura informatica, che permette il monitoraggio dello stato dei veicoli.
- **Economia del fare** : le stampanti 3D, grazie alla prototipazione rapida, hanno permesso l'insorgere di un "artigianato digitale" autoprodotto a basso costo, con uso di materiali plasmabili e anche di robot programmati.

Cloud Automation

I robot di servizio nei centri di *Amazon* che si occupano dello stoccaggio dei prodotti funzionano grazie a degli algoritmi di instradamento che ne determinano i percorsi ottimali, tali algoritmi molto spesso risultano costosi da un punto di vista computazionale, de facto, i robot di questo tipo inviano i dati ad un server centralizzato che si occupa di eseguire gli algoritmi per poi restituire i risultati ai robot.

Un aspetto fondamentale dell'industria 4.0 è il processo di **trasformazione digitale** nelle aziende tramite l'inclusione delle tecnologie ICT, in particolare, sono identificati 6 passi fondamentali nella trasformazione, che incrementano il valore dell'azienda e del prodotto finale.

Ovviamente tale processo ha dei costi da sostenere, banalmente l'implementazione di hardware e software, inoltre, è soggetto ad errori che possono comportare rischi non di poco conto, i quali

- mancanza di una strategia chiara
- mancanza di consenso ed impegno nella leadership
- concentrazione sulle tecnologie piuttosto che sulle persone
- farsi guidare dalle tendenze senza avere una visione
- trascurare il contributo dei clienti
- voler organizzare il processo in autonomia
- sottovalutare le competenze interne
- non considerare la sicurezza dei dati

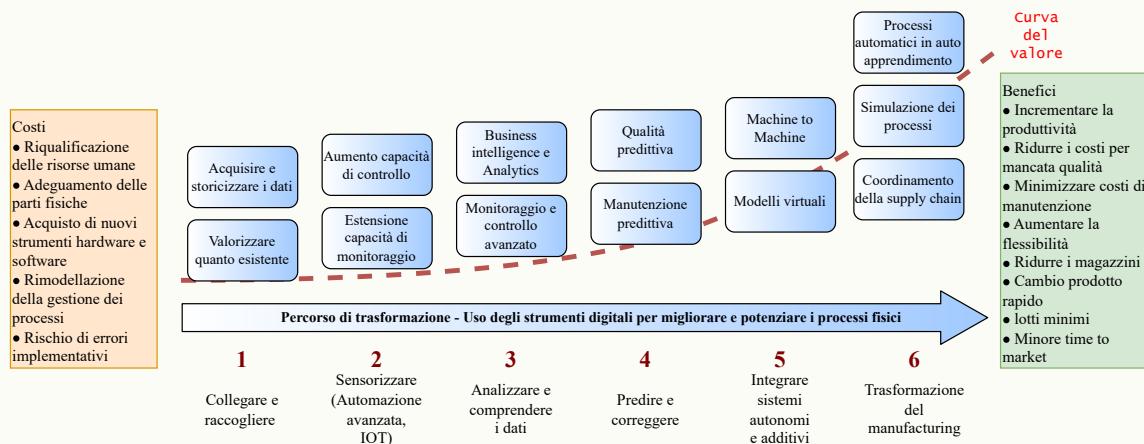


Figura 1.23: i 6 passi della trasformazione digitale

- mancanza di flessibilità
- carenza di comunicazione
- sottovalutazione delle complessità

La *McKinsey* nel 2017 ha rilasciato un rapporto riguardante le attività lavorative ed il possibile grado di automatizzazione di queste, in particolare, si hanno i seguenti risultati, su un analisi che ha coinvolto circa 2000 attività lavorative relative a circa 800 occupazioni

- Storicamente l'automazione ha aumentato la produttività del (circa) 1% all'anno
- Con le attuali tecnologie (in riferimento al 2017), è possibile automatizzare circa la metà delle attività lavorative
- Le attività che si prestano maggiormente ad essere automatizzate, sono quelle che caratterizzano lavori fisici, prevedibili e ripetitivi, che richiedono tipicamente basse capacità cognitive. Costituiscono circa il 51% delle attività negli USA
- Le occupazioni le cui attività possono essere completamente automatizzate sono circa il 5%
- Nel 60% delle occupazioni, circa il 30% delle attività possono essere automatizzate
- Il lavoro umano affiancato ai robot è necessario a garantire la crescita del PIL

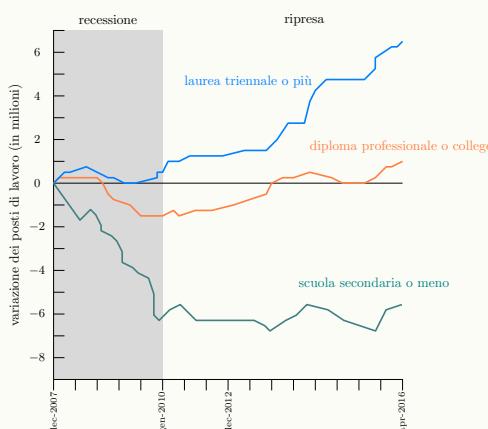


Figura 1.24: Andamento dei posti di lavoro in base al livello di educazione



1.6.3 Industria 5.0

La commissione Europea ha trattato un rapporto sull'industria 5.0, che mira a sviluppare il modello dell'industria 4.0 verso un'industria europea sostenibile, resiliente e centrata sulla persona. Il progetto di ricerca *Horizon Europe* punta ad investire sui progetti di ricerca che incorporano tali modelli, in Italia, il **PNNR** è un insieme di iniziative che puntano alla trasformazione dei sistemi di produzione, in particolare

1. digitalizzazione, innovazione, cultura
2. transazione al verde
3. infrastrutture per una mobilità sostenibile
4. istruzione e ricerca
5. coesione ed inclusione sociale
6. salute

Come già accennato, le parole chiave dell'industria 5.0 sono le seguenti

- **resilienza** : gestire i cambiamenti desiderati e non (ad esempio, epidemie) con una produzione industriale dotata di supporti per le infrastrutture critiche e resistente a "interruzioni".
- **centralità della persona** : mettere l'essere umano al primo posto e chiedersi cosa può fare la tecnologia per noi, e non cosa possiamo fare noi per la tecnologia, in particolare, adottarla per adattare i processi produttivi alle necessità dei lavoratori.
- **sostenibilità** : economia circolare, riduzione dei rifiuti e dell'impatto ambientale, assicurare i bisogni odierni senza mettere a repentaglio le future generazioni.

CAPITOLO

2

RETI PER L'AUTOMAZIONE

Molti dei concetti trattati in questo capitolo, sono considerati più approfonditamente negli appunti del corso di [Reti di Elaboratori](#).

2.1 Sistemi di Comunicazione

Le informazioni vengono condivise orizzontalmente e verticalmente fra i livelli della piramide CIM, in particolare

- si acquisiscono informazioni
- si elaborano strategie
- si attuano azioni correttive

I sistemi di comunicazione devono essere adeguati al garantire il flusso di informazioni fra i vari livelli della rete che differiscono nelle caratteristiche

- tipologie di dati differenti
- differenti vincoli di comunicazione (ad es. il livello di campo avrà dei vincoli real time)

È necessaria una standardizzazione dei protocolli di comunicazione digitale.



Gli elementi base di una rete sono

- Un mittente
- Un destinatario
- Un mezzo fisico sul quale viaggerà l'informazione

I dati sono sul mezzo delle informazioni fisiche (luce, tensioni elettriche), i segnali possono essere analogici o digitali, in base alla definizione del protocollo. Il tipo di trasmissione può variare a seconda di diversi fattori, quali la **direzione**

- *Simplex* : la trasmissione è unidirezionale
- *half duplex* : la trasmissione è bidirezionale ma alternata, non si può trasmettere informazione in entrambe le direzioni nello stesso momento
- *full duplex* : bidirezionale simultanea

il posizionamento dei bit

- *parallela* : i bit di un byte vengono trasmessi in parallelo su più canali, viene usata su distanze ridotte causa la facile interferenza alla quale son soggetti.
- *seriale* : il mezzo fisico è tipicamente suddiviso in "invio", "ricezione" e "massa", ed i bit di un byte sono trasmessi in sequenza uno dopo l'altro. Quest'ultima può a sua volta essere
 - sincrona : i dati sono trasmessi continuamente insieme ad un segnale di sincronizzazione
 - i dati sono trasmessi in modo irregolare a frequenza costante, con bit di sincronizzazioni incapsulati nei dati

Le reti industriali prevedono solitamente una trasmissione digitale half duplex, seriale ed asincrona.



Figura 2.1: tipi di reti e protocolli industriali

Le *Rete Enterprise* sono adottate per le informazioni gestionali e seguono il classico modello client-server, non hanno vincoli di real-time, più che la "robustezza" dell'informazione rispetto ai disturbi ambientali (che possono essere presenti in una cella), è importante la sicurezza e la riservatezza di esse. Tali reti seguono lo standard Ethernet, e sfruttano i protocolli di rete e di trasporto (ossia, IP e TCP) per garantire la ritrasmissione sicura dei dati in caso di perdite di pacchetti.

Le reti al livello di *Controllo* e *Campo* invece gestiscono le informazioni nelle celle e fra le varie macchine, i dispositivi comunicanti spesso non sono standard, ma sono PLC, controllori embedded, e dispositivi di campo, quindi è importante che i protocolli in tale livello siano flessibili ed adattabili all'eterogeneità dei

client.

Qui i dati trasmessi hanno piccole dimensioni, ma la frequenza di trasmissione è alta, e deve soddisfare i vincoli real time, per questo Ethernet non è appropriato e si utilizzano soluzioni apposite. Essendo l'ambiente industriale "ostile", i mezzi di trasmissione devono essere robusti, i ritardi nella trasmissione nella struttura ad anello dei controllori possono causare un degrado nelle prestazioni non trascurabile, è più che mai necessario *determinismo* nella trasmissione.



2.2 Classificazione ed Architetture delle Reti

Nel livello di campo, esistono due possibili architetture della rete da adottare che si differenziano nella topologia della rete ed altri fattori. In particolare

- **Architettura tradizionale** : Un architettura centralizzata in cui il controllore prevede un collegamento punto-punto con ogni altri dispositivo di campo.

Vantaggi

- sistema affidabile e collaudato
- disponibilità di tutte le tipologie di strumentazione sul mercato

Svantaggi

- elevato numero di collegamenti
- cablaggio costoso
- lavoro di stesura e protezione dei fili critico

- **Architettura a bus di campo (fieldbus)** : Un architettura che prevede la presenza di un bus centrale (la cui comunicazione è digitale) alla quale ogni dispositivo si connette direttamente.

Vantaggi

- installazione più economica
- modularità : è facile aggiungere e rimuovere dispositivi
- tolleranza ai guasti
- condivisione delle risorse

Svantaggi

- i dispositivi comunicano sullo stesso mezzo, è quindi necessario un protocollo di accesso al mezzo
- difficile da installare in aree pericolose



Figura 2.2: soluzioni a livello di campo

2.2.1 Tipologie di Reti

Una rete può essere

- **broadcast** : vi è un unico canale di comunicazione condiviso da ogni macchina sulla rete, i messaggi arrivano ad ogni nodo, i nodi scarteranno i messaggi di cui non sono destinatari. Queste possono essere a loro volta
 - *reti a bus* : canale condiviso, in ogni istante un solo nodo può trasmettere contemporaneamente, è necessario arbitraggio.
 - *reti ad anello* : i pacchetti circolano in serie su un anello, necessario arbitraggio per accessi simultanei



- **peer-to-peer** : per ogni coppia di nodi vi è una connessione dedicata, se due macchine non sono connesse fisicamente, è necessario definire un cammino fra di esse tramite appositi algoritmi di routing, se più pacchetti relativi allo stesso messaggio seguono cammini diversi, è necessario gestire anche la sequenza con cui essi vengono ricevuti. Vengono applicate a reti di dimensioni maggiori, normalmente più reti LAN sono connesse in tal modo.



- Nella maggioranza delle applicazioni più complesse, la soluzione più frequente ricade in **reti ibride**.

Esiste una scala di classificazione delle reti :

Scala	Tipo	Nome completo	Esempio
Distanza ravvicinata	PAN	Personal Area Network	Bluetooth
Edificio	LAN	Local Area Network	WiFi, Ethernet
Città	MAN	Metropolitan Area Network	Cablata, DSL
Paese	WAN	Wide Area Network	Grandi ISP
Pianeta	Internet	La rete di tutte le reti	L'Internet

La **LAN** è la rete locale, come una rete domestica, è una rete privata ed ogni terminale connesso ad essa è identificato da un indirizzo distinto dagli altri, può essere a *cavo condiviso* oppure a *commutazione* con uno switch. In tale modello di cavo condiviso il pacchetto inviato ad un dispositivo viene ricevuto da tutti, solo il destinatario lo elaborerà, tutti i restanti host lo ignoreranno.

Quest'ultimo a commutazione è il più utilizzato tutt'oggi, ogni dispositivo è direttamente collegato allo switch, ed esso è in grado di riconoscere gli host ed inviare i pacchetti esclusivamente al destinatario, riduce il traffico nella LAN.

Le reti **WAN** sono reti geografiche, vengono interconnessi dispositivi di comunicazione, necessari a città, regioni o perfino nazioni. I dispositivi in questione sono switch, router e modem, tale rete è gestita da

un grande operatore/ente di telecomunicazioni detto IPS (Internet Service Provider) che fornisce i servizi alle organizzazioni.

Una WAN può vedere i suoi dispositivi di comunicazione connessi punto-punto, oppure a commutazione, con più punti di terminazione (usata nelle dorsali di Internet), tutt'oggi è raro trovare LAN o WAN isolate, spesso sono connesse fra loro per formare una internetwork (internet), per mettere in comunicazione due LAN in città differenti tramite una WAN.

Dispositivi di Interconnessione

Per il collegamento fra tratti di una stessa rete, oppure fra reti diverse, sono coinvolti vari dispositivi

- **repeater (ripetitore)** : amplifica e ricostituisce il segnale originale su segmenti analoghi della stessa rete [repeater RS485]
- **hub (concentratore)** : estende una rete a stella, amplifica e ricostituisce lo stesso segnale su tutte le porte, non riduce le collisioni [Ethernet hub]
- **switch (interruttore)** : come un hub, ma su una singola porta per volta, può ridurre le collisioni [Ethernet switch]
- **transceiver (ricetrasmettitore)** : connette a una stessa rete segmenti di diversa tipologia [RS232/RS485 transceiver]
- **bridge (ponte)** : connette due reti che usano lo stesso protocollo ma che hanno layer differenti al livello inferiore [Modbus RS485 / Ethernet TCP-IP bridge]
- **router (instradatore)** : connette due reti dello stesso tipo [Ethernet TCP-IP router]
- **gateway (portale)** : connette due reti di tipo diverso [Ethernet / Modbus gateway]

Nelle reti broadcast, l'accesso al canale può venire per vie di allocazione

- **statica** : il tempo viene suddiviso in time slices (quanti) e ad ogni nodo viene dedicato un intervallo periodico in cui può comunicare, nel caso non abbia nulla da trasmettere, il canale resta inutilizzato (spreco di banda).
- **dinamica** : Si suddivide in
 - controllo centralizzato : un nodo *master* determina il prossimo dispositivo che potrà comunicare tramite strategie di polling.
 - controllo decentralizzato : ogni nodo decide autonomamente se trasmettere o no.
 - sistemi a collisione : i nodi possono trasmettere, se capita che due di essi trasmettano contemporaneamente, vi sarà una *collisione* che verrà rilevata ed eventualmente risolta.

Architettura Software

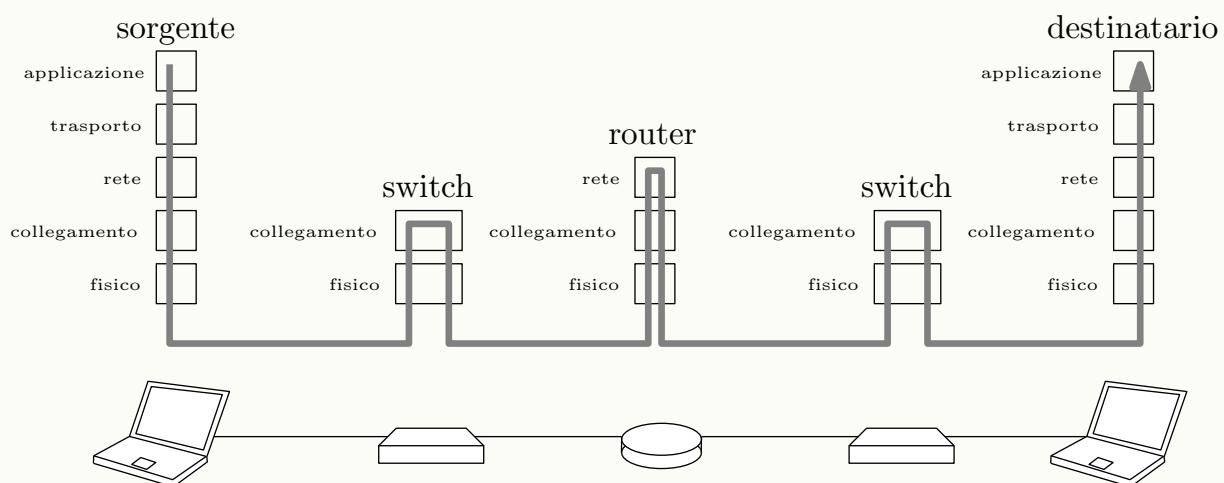
I 5 macro-livelli sulla quale si fonda la comunicazione sono i seguenti :

applicazione	supporto delle applicazioni di rete
trasporto	trasferimento dati fra processi
rete	instradamento dei pacchetti dalla sorgente alla destinazione
collegamento	trasferimento di dati tra elementi di rete vicini
fisico	bit sul canale fisico (cavo o wireless)

Sino ad ora i dati incapsulati che vengono comunicati sulla rete sono stati chiamati generalmente "pacchetti", vedremo che questi assumono una denominazione diversa per ogni livello. Ogni protocollo fa parte di un livello, ed anche se esistono più protocolli per un livello, ogni pacchetto che viene trasmesso usufruisce di un solo protocollo per livello. I nomi dei protocolli citati in seguito, verranno approfonditi e caratterizzati in seguito.

1. Il livello di **applicazione** è dove risiedono le applicazioni di rete che usufruiscono dei servizi di Internet, alcuni dei protocolli presenti in questo livello sono *HTTP, SMTP, FTP, DNS*, in questo livello, i pacchetti sono chiamati **messaggi**.
2. Il livello di **trasporto** si occupa del trasferimento dei messaggi dal livello di applicazione di un client al livello di applicazione del server, alcuni protocolli sono *TCP e UDP*, in questo livello, i pacchetti sono chiamati **segmenti**.
3. Il livello di **rete** riguarda l'instradamento dei segmenti dall'origine alla destinazione, un noto protocollo è l'*IP*, i pacchetti in questo livello sono detti **datagrammi**.
4. Il livello di **collegamento** si occupa della trasmissione dei datagrammi da un nodo della rete al nodo successivo sul percorso, alcuni protocolli sono *Ethernet, Wi-Fi e PPP*, lungo un percorso sorgente-destinazione, un datagramma può essere gestito anche da differenti protocolli, i pacchetti qui sono detti **frame**.
5. Il livello **fisico** riguarda il trasferimento dei singoli **bit** sul canale fisico, tramite elettricità nei cavi, oppure onde elettromagnetiche.

Durante la comunicazione, non tutti i sistemi intermedi richiedono che il messaggio venga processato su tutti i livelli, alcuni dispositivi richiedono solo alcuni layer, riducendo la complessità.



Lo strato di un livello ha una comunicazione logica/virtuale con lo stesso livello su un altro computer, ma i dati non sono trasferiti direttamente da uno strato all'altro, passano per tutti i livelli inferiori, un *protocollo* è quindi un insieme di regole che controllano il formato ed il significato dei pacchetti scambiati tra le entità *pari* all'interno di uno strato, un *servizio* invece è un insieme di primitive che uno strato offre a quello superiore, ossia :

- Quali operazioni fornisce (senza dire nulla sull'implementazione).
- Posto come interfaccia fra due strati, quello inferiore fornisce il servizio, quello superiore ne usufruisce.

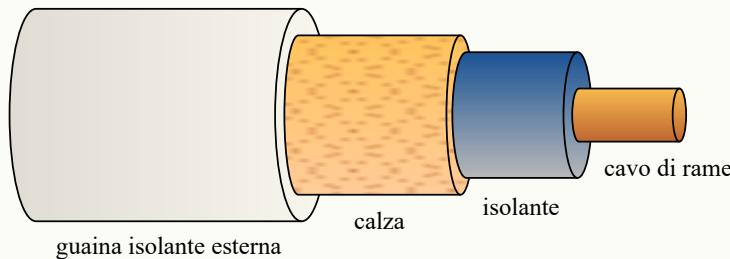
♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪

2.3 Livello Fisico e di Collegamento

Il livello fisico dello stack protocollare riguarda la trasmissione fisica dei bit sul canale e la definizione delle convenzioni elettriche. Esistono differenti mezzi di trasmissione dei dati

- *doppino telefonico* : È una coppia di cavi intrecciati, mezzo piuttosto antiquato e la trasmissione è nell'ordine dei Kbps (Kilobit per secondo).
- *cavo coassiale* : È un conduttore di rame circondato da materiale isolante, resiste bene ai disturbi e la banda è buona, la trasmissione è nell'ordine dei Mbps.

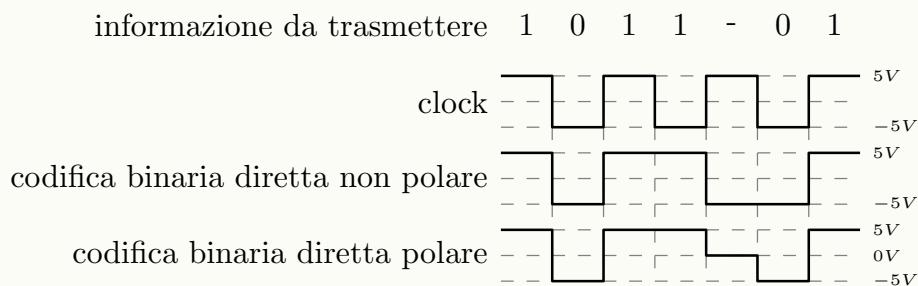
- *fibra ottica* : È un cavo in fibra di vetro in cui l'informazione passa sotto forma di impulsi luminosi. I segmenti possono essere lunghi fino a 2km e la trasmissione è nell'ordine dei Tbps.
- *wireless* : La trasmissione si propaga nel vuoto sotto forma di onde elettromagnetiche.



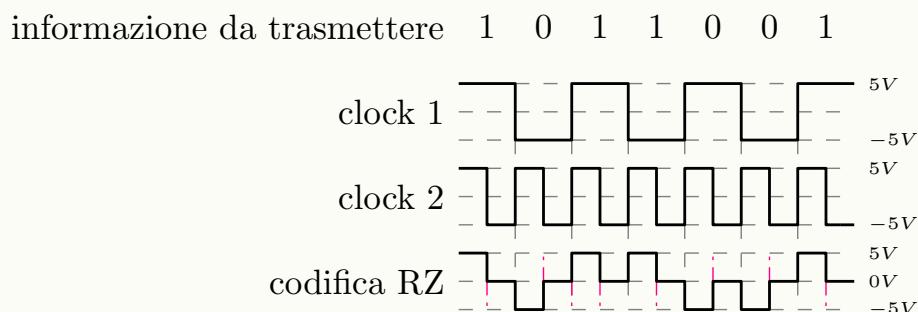
L'informazione viene codificata con dei bit che vengono trasmessi su diversi spettri di sequenza, con media del segnale nulla oppure no. Una codifica ideale ricerca

- mantenimento sincronizzazione sorgente/destinazione
- bassa probabilità di errore (a causa di rumore/interferenze)
- minima banda utilizzata, a parità di informazione trasmessa
- capacità di rivelazione di errori

La **codifica binaria diretta** più semplice, dove un livello alto di tensione rappresenta un 1 e un livello basso (o assenza di segnale) rappresenta uno 0, si dice non polare quando l'assenza di segnale rappresenta un bit, è polare invece quando l'assenza di segnale è associata alla non trasmissione.



La codifica **return to zero (RZ)** prevede il ritorno del segnale a zero volt almeno una volta per ogni bit trasmesso, permettendo di inserire un riferimento temporale per la sincronizzazione dei dati. Saranno quindi utilizzati due clock, uno legato alla frequenza dei bit (indica l'inizio e fine di ogni bit), ed un altro con frequenza doppia rispetto al primo clock, viene utilizzato per individuare il punto medio di ogni bit, ovvero il momento esatto in cui il segnale ritorna a zero.



Nella codifica **manchester**, simile a quella RZ, il clock viene utilizzato per metà, la prima parte viene utilizzata per l'informazione, la seconda per una "risalita" o "discesa" verso lo zero, cambiando bit.

La codifica **manchester differenziale** è simile alla codifica manchester ma il segnale viene paragonato con il semiperiodo precedente per far sì che ci sia sempre un cambio di segnale al "centro" del bit time. L'assenza di transizione al centro del bit time indica una violazione della codifica: viene usata per delimitare un frame di trasmissione.



Esistono differenti standard di connessione elettrica/meccanica:

- IEEE RS232 : (tra i più consolidati, punto-punto), comprende un connettore a 25 pin la trasmissione è limitata a 20Kbps e la lunghezza dei segmenti è al più di 15 metri.
- IEEE RS422 : ampiamente utilizzato in ambito industriale, il throughput è fino a 10Mbps ed i segmenti possono essere lunghi anche 1200 metri. In configurazione simplex, la comunicazione avviene in una sola direzione, può essere composto da due coppie di cavi più massa per il full duplex.

In una rete connessa in tal modo ci può essere al più un trasmettitore alla volta e al più 10 ricevitori.

Il segnale viene trasmesso in modo differenziale, ovvero come la differenza tra due segnali complementari (A e -A). Questo metodo permette di ridurre notevolmente il rumore e aumenta la distanza di trasmissione.

- IEEE RS485 : anch'esso ampiamente adoperato in ambito industriale, utilizza la trasmissione differenziale e offre una buona immunità ai disturbi. A differenza dell'RS422, l'RS485 è half-duplex, ovvero la comunicazione avviene in una sola direzione alla volta.

Risulta necessario controllare l'accesso, solo un dispositivo può trasmettere alla volta. Gli altri dispositivi hanno il circuito di ingresso in uno stato di alta impedenza, come se fossero scollegati. Questo meccanismo evita conflitti durante la trasmissione.

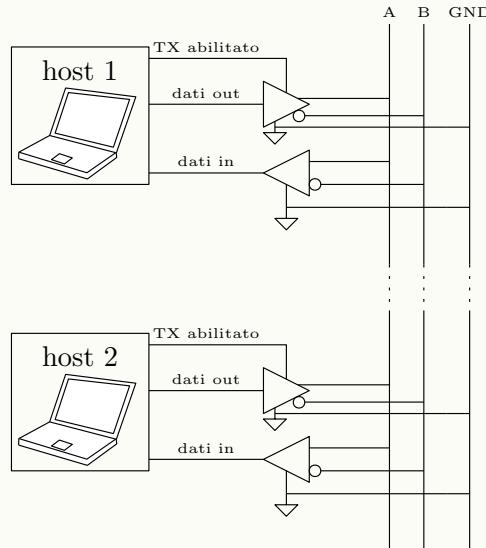


Figura 2.3: RS485

2.3.1 Protocolli MAC

Il livello di collegamento si occupa di ricevere i dati da inviare e di suddividerli in *frame*. Si occupa anche di inviare appositi ACK (messaggi di conferma di ricezione) per la comunicazione affidabile e di gestire l'accesso al mezzo fisico (MAC).

Il **CSMA/CD** (Carrier Sense Multiple Access / Collision Detection) è un protocollo per l'accesso al mezzo in cui i nodi sono costantemente in ascolto sul canale, quando un nodo trasmette, confronterà il segnale di ritorno con il segnale originale, se differiscono, vorrà dire che c'è stata una collisione. Ogni nodo a seguito di una collisione attenderà un lasso di tempo casuale prima di ritrasmettere.

Nel **CSMA/CR** (Carrier Sense Multiple Access / Collision Resolution) ogni nodo è sempre in ascolto sul canale, è il metodo MAC utilizzato nei protocolli *CAN* e *CANOpen*, vi è uno stato fisico del canale che si occupa dell'*arbitraggio logico* a livello dei bit. Ogni dispositivo ha un identificativo (ID) unico, codificato in una sequenza di bit. Quando un dispositivo vuole trasmettere, trasmette il suo ID bit per bit sul bus.

- Se più dispositivi trasmettono contemporaneamente, si verifica una collisione.
- Il bus calcola l'AND bit a bit degli ID dei dispositivi che stanno trasmettendo.
- Il dispositivo che ha trasmesso il bit dominante (solitamente 0) in quella posizione dell'ID "vince" la collisione e continua a trasmettere il bit successivo.
- Gli altri dispositivi che hanno perso la collisione interrompono la trasmissione e riprovano più tardi.



Il modello **Token Bus/Ring** vede i nodi della rete scambiarsi un "token", uno speciale pacchetto di dati che regola i turni di trasmissione. Al costo di aumentare la latenza di trasmissione, si eliminano le collisioni.

Precisamente, quando un nodo vuole trasmettere ed è in possesso del token

- lo modifica e rispedisce in circolo assieme al frame del dato (e con un host destinatario)
- " la presenza del token "modificato" non permette la trasmissione agli altri nodi
- il ritorno del token modificato al nodo trasmettente indica l'avvenuto successo della trasmissione
- il token originale viene rimesso in circolo e altri nodi potranno quindi trasmettere

L'utilizzo di differenti token permette un partizionamento dei messaggi per livelli di priorità.



2.4 I Protocolli Fieldbus

Nell'ambito dell'automazione lo stack protocollare viene ridotto considerando esclusivamente

- livello di applicazione (tipo di comunicazione)
- livello di collegamento (gestione dell'accesso)
- livello fisico

Le reti field bus vedono diversi nodi, detti *DLE* (Data Link Element), collegati ad una comune rete elettrica dove viaggia l'informazione. Vi è un elemento speciale, il **Link Active Scheduler (LAS)** che si occupa di gestire staticamente l'accesso al mezzo concedendolo ai vari DLE tramite un'apposita tabella di scheduling, l'esecuzione dei task periodici real time. Negli intervalli di tempo liberi non occupati dai task periodici, vengono gestiti i task periodici in modo dinamico.

Il LAS utilizza un token per la concessione ai DLE dell'utilizzo del mezzo, questi ultimi una volta finito, dovranno restituire un frame *return token* in modo che il LAS possa passare al prossimo DLE. Le reti fieldbus sono caratterizzate da

- mezzo fisico : doppino twisted schermato (2 fili o 4 fili)
- rete a bus con brevi tratti e resistenze di 120Ω come terminatori di linea.
- caratteristiche del mezzo
 - massima lunghezza: 1000 m
 - 9 velocità: da 10Kbps a 1 Mbps, in funzione della lunghezza del bus (1 Mbps @25 m)
 - massimo numero di dispositivi: 128 (1 master e 127 slaves)
 - vari tipi di connettori: RJ45, 9-pin SUB D DIN 41652, ecc...
- protocolli di accesso al mezzo fisico : CSMA/CR oppure CSMA/CA
 - ogni dispositivo può inviare dati appena il bus è libero
 - arbitrario delle collisioni non-distruttivo a livello di bit, con bit 0 dominante
 - messaggi a priorità maggiore hanno valore binario minore dell'identificatore
- il modello della comunicazione è *producer/consumer*
 - formato frame: 11 bit in testa indicano tipo dei dati utili trasmessi e nodo produttore
 - seguono 5 bit di controllo (RTR, start, lunghezza dati)
 - seguono i dati utili: massimo 8 bytes per frame
 - dispositivi e procedure a elevata sicurezza

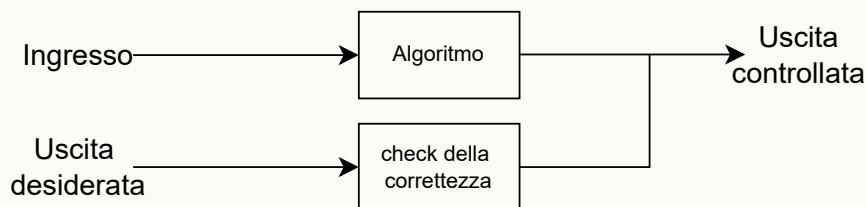
CAPITOLO

3

SISTEMI DI CONTROLLO REAL TIME

Nell'ambito dell'automazione, è fondamentale definire un sistema di controllo che agisca in tempo reale, che implementi un algoritmo con lo scopo di eseguire i generici task di un processo produttivo, lo scopo dell'algoritmo è quello di *sequenziare* le varie operazioni in modo che rispettino dei vincoli temporali.

In un processo vi è un insieme diffusi di singoli componenti (come sensori o attuatori) che comunicano con il sistema di controllo che gestisce i riferimenti dei segnali, i valori di riferimento sono determinati dall'*algoritmo* in questione, che viene implementato attraverso gli strumenti dell'informatica.



L'algoritmo si definisce **logicamente corretto** se, definiti i dati in ingresso ed i dati in uscita, fornisce i risultati attesi. Si dice **temporalmente corretto** se i risultati forniti rispettano i vincoli di tempo (deadline) imposti. Un sistema *real time* deve essere logicamente e temporalmente corretto.

Erroneamente, si può pensare che per far sì che un sistema sia real time (rispetti i vincoli di tempo), basti rendere il sistema più veloce aumentando la potenza di calcolo, ciò non è in realtà sufficiente, è necessario che l'unità di calcolo si dedichi esclusivamente all'algoritmo di controllo, è quindi importante che sia correttamente configurato.

Un sistema deputato al real time deve dare *massima priorità* all'algoritmo di controllo, per questo i sistemi real time sono sistemi dedicati realizzati ad hoc, e non vengono utilizzati i sistemi come windows o linux, in quanto essendo general purpose, prevedono molte funzionalità (ad esempio, la gestione dell'I/O) che possono ritardare l'esecuzione dell'algoritmo.

Un sistema real time deve quindi essere *prevedibile* e *deterministico*, possiamo suddividere i task da sequenziare in due categorie

- relativi alla sicurezza
- nice to have

I task responsabili (direttamente o indirettamente) della sicurezza fisica delle persone devono avere massima priorità e la loro esecuzione deve essere strettamente deterministica, un esempio di task di questo

tipo, è l'azionamento del freno di un ascensore. I task nice to have invece, "prenotano" l'esecuzione ma potrebbero dover dare la priorità ai task più importanti, un esempio di task di questo tipo può essere l'attivazione dello stereo in un'automobile (nice to have = godibile, ma non fondamentale).

Introduciamo a tal proposito due coefficienti, siano

- r_c : risultati corretti
- r_e : risultati elaborati
- r_t : risultati ottenuti rispettando i vincoli

Si hanno

- **coefficiente logico** : $C_l = \frac{r_c}{r_e}$
- **coefficiente temporale** : $C_t = \frac{r_t}{r_e}$

Un sistema si dice **hard real time** se entrambi i coefficienti sono uguali ad 1. I già accennati task che hanno a che fare con l'incolumità delle persone devono essere hard real time. I sistemi **soft real time** garantiscono la correttezza temporale solamente per una certa percentuale di task.

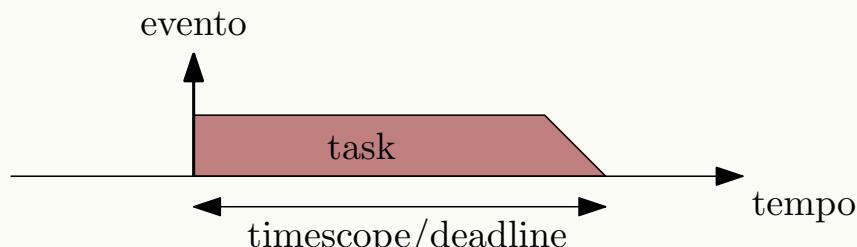
- hard real time : $\min(C_l, C_t) = 1$
- soft real time : $\min(C_l, C_t) < 1$

Per un generico task denotiamo T_i il tempo minimo necessario per eseguirlo, e denotiamo d_i la sua deadline (tempo massimo in cui va eseguito), il *vincolo real time* T_i/d_i da una misura sul margine di tempo rimanente fra il completamento di un task e la sua deadline, deve essere auspicabilmente minore di 1. Se è molto minore di 1, si dice che il vincolo è *largo*, se è prossimo all'unità, allora è *stretto* ed implica una programmazione sistematica.

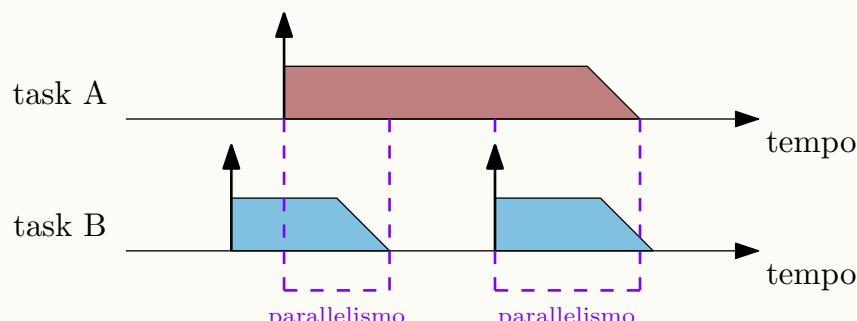
❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖

3.1 Parallelismo e Programmazione Concorrente

I task verranno rappresentati tramite un diagramma che prevede un asse temporale e dei blocchi rappresentanti la "vita" di tali task nel tempo.



Un *evento* avviene in un certo istante di tempo e sancisce il momento in cui il task può cominciare, tale diagramma individua l'intervallo temporale in cui il task è pronto per essere eseguito fino al momento in cui deve essere terminato. Può accadere che i timescope di due task si intersechino necessitando di un'esecuzione *parallela*.



Un sistema di controllo con n processori permette il *parallelismo reale*, in cui nello stesso istante più task vengono eseguiti in parallelo su differenti unità di calcolo. Dato che in genere, il numero dei task è sempre maggiore del numero dei processori, è necessaria la gestione del **parallelismo logico**, in cui più task vanno alternati su un singolo processore, varrà quindi l'ipotesi che l'unità di calcolo dei sistemi di controllo trattati in questo corso sia unica.

L'obiettivo è quello di disegnare un **algoritmo di scheduling**, che si occupi di decidere in che modo i task vanno sequenziati nel tempo. Molti dei concetti presenti sono ampiamente trattati negli appunti del corso di [Sistemi Operativi 1](#).

Caratteristiche di un Task

Ogni task nell'ambiente del sistema di controllo, in un preciso istante può trovarsi in uno dei seguenti stati

- *attivo* : L'evento che lo ha scaturito è precedente all'istante attuale, e la sua deadline è futura all'istante attuale. Un task attivo inoltre può essere
 - *pronto* : attivo, ma in attesa di essere eseguito sul processore.
 - *in esecuzione* : in esecuzione sul processore.
- *Inattivo* : Il contrario di attivo.

Inoltre per ogni task i sono definiti i seguenti istanti che lo caratterizzano

- *activation time* a_i : L'istante in cui avviene l'evento che lo scaturisce.
- *start time* s_i : L'istante in cui viene eseguito sul processore per la prima volta.
- *end time* e_i : L'istante in cui l'esecuzione è terminata.
- *deadline* d_i : L'istante in cui la vita del processo non può protrarsi oltre. Inoltre, dati gli istanti presentati, si identificano 4 intervalli di tempo relativi ad un task i :
 - *computation time* $C_i = e_i - s_i$: il tempo effettivo impiegato sul processore.
 - *deadline relativa* $D_i = d_i - a_i$: la durata del suo timescope.
 - *response time* $R_i = e_i - a_i$: tempo impiegato per completare il task.
 - *lateness* $L_i = e_i - d_i$: il ritardo fra la deadline e la terminazione del processo, se tale valore è maggiore di 0 si parla di soft real time, deve auspicabilmente essere minore di 0.



Lo scheduler (unità logica che applica l'algoritmo di scheduling) deve attuare una strategia che

- Rispetti i vincoli delle risorse
- Rispetti le priorità dei task
- Massimizzi opportuni indici prestazionali

Quando un evento genera un task, questo viene messo nella *ready queue*, ossia la coda dei task pronti per essere eseguiti, l'algoritmo deve selezionare dalla coda il task successivo che verrà eseguito sull'unità di calcolo (dispatching), inoltre uno scheduler può essere **preemptive** :

Se vi è un processo *A* in esecuzione e ne arriva nella ready queue uno nuovo *B* di priorità maggiore, lo scheduler *preemptive* bloccherà l'esecuzione del processo *A* per eseguire il processo *B*, una volta terminato, *A* potrà tornare in esecuzione.

I task, possono richiedere l'accesso a delle risorse

Ad esempio, un task che si occupa della stampa di un foglio può richiedere accesso alla stampante

Inoltre queste risorse possono essere condivise fra più task, l'accesso a queste deve essere **mutualmente esclusivo**, quando un task sta utilizzando una risorsa, nessun task che la necessita può essere eseguito. I task pronti ad essere eseguiti, ma che richiedono una risorsa già in uso, vengono messi in una *blocked queue*, e saranno "bloccati" finché la risorsa di cui necessitano non tornerà disponibile.



Figura 3.1: modello generico di scheduling



3.2 Il Problema dello Scheduling

Lo scheduling viene applicato a livello di coordinamento per la sequenziazione dei task, in particolare, il problema consiste nel sequenziare n task, a cui sono associati i rispettivi vincoli temporali e di accesso alle risorse, ci si impone una percentuale π di task che rispettino la correttezza temporale, e che eventualmente rispettino i vincoli di mutua esclusione nell'accesso alle risorse.

Definizione : Dato un insieme di task, esso è *schedulabile* se esiste almeno un algoritmo di scheduling che lo *risolve* (ne trova una sequenziazione ammissibile, ossia che rispetti i vincoli).

s Si consideri il seguente esempio, vi sono due task (A_1, A_2), con i rispettivi vincoli temporali

$$\begin{array}{llll} a_1 = 35 & C_1 = 55 & D_1 = 80 & d_1 = 115 \\ a_2 = 10 & C_2 = 60 & D_2 = 145 & d_2 = 155 \end{array}$$

Il tempo è misurato in generiche unità di tempo *t.u..*. Per lo scheduling di questi task, si tenta un approccio **First In First Out (FIFO)**, ossia, si schedulano i task in ordine di arrivo.



Come si vede dalla traccia dello scheduling, l'algoritmo FIFO non risolve il problema in quanto fa sì che il task A_1 termini all'istante 125 nonostante la sua dead line assoluta fosse 115. Si prova allora uno scheduling FIFO ma di tipo preemptive, in cui si da priorità al task con tempo di completamento minore.

nonostante ci sia A_2 in esecuzione, all'attivazione di A_1 , lo scheduler gli dà la precedenza in quanto ha un tempo di completamento minore



una volta terminato A_1 , il task A_2 può riprendere controllo del processore

Questa serializzazione dei task è ammmissible dato che i vincoli temporali sono rispettati, l'algoritmo FIFO Preemptive risolve l'insieme (A_1, A_2).

Ci si pone la domanda : Esiste un algoritmo generale che, dato un arbitrario numero di task con rispettivi vincoli temporali, ne trovi una serializzazione ammmissible? Si è dimostrato che il problema dello scheduling è NP-completo, non è possibile trovare una soluzione in maniera efficiente. Nell'ambito dell'automazione industriale però, è possibile concedersi ipotesi largamente semplificate che permettono di adoperare degli algoritmi che risolvano gli insiemi di task trattati in tali contesti.

3.2.1 Classificazione degli Algoritmi

Gli algoritmi di scheduling si classificano secondo 5 differenti parametri. Un algoritmo può essere

- **best effort** : non è hard real time, $\pi < 1$
- **guaranteed** : è hard real time, $\pi = 1$, rispetta sempre i vincoli temporali, per garantire la correttezza si basa su alcuni "test" di garanzia, quando evento genera un task, vi è una fase di *admission control* in cui il task viene accettato nella ready queue solo se, la sua aggiunta all'insieme dei task lo rende ancora un insieme schedulabile.



Un altro criterio di classificazione è il seguente, uno algoritmo può essere

- **mono processore** (verranno trattati questi)
- **multi processore**

Può essere inoltre

- **preemptive** : se esso è in grado di interrompere l'esecuzione in una delle unità di calcolo di un task a minor priorità a favore dell'esecuzione di un task a maggior priorità.
- **non preemptive** : se esso non è in grado di interrompere l'esecuzione di un task quando esso è stato inviato ad una delle unità di calcolo.

Un importante distinzione è la seguente

- **offline** : Se le decisioni di scheduling sono prese prima dell'attivazione dei task, quindi la sequenza è nota a priori.

- **online** : Se le decisioni dipendono dall'ordine di arrivo dei task.

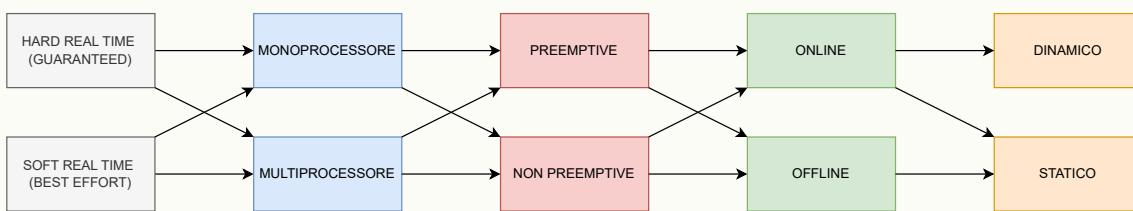
L'algoritmo FIFO è online perché le decisioni dipendono dagli activation time. L'ultima distinzione è la seguente

- **dinamico** : I parametri sulla quale si basa il dispatching (priorità) possono variare nel tempo.
- **statico** : I parametri sulla quale si basa il dispatching (priorità) sono costanti.

FIFO è un algoritmo statico che basa le priorità sull'activation time.

Proposizione : Un algoritmo offline è anche necessariamente statico.

Dimostrazione : Supponiamo per assurdo che un algoritmo offline sia dinamico, ad un certo istante t , un generico evento E comporta un cambio nei parametri su cui si basa lo scheduling, facendo cambiare l'ordine di serializzazione, tale ordine non era però noto a priori in quanto è dipeso dall'evento E , quindi è impossibile che l'algoritmo sia offline.



In generale, gli algoritmi guaranteed sono dinamici e online (hanno più gradi di libertà), ma sono anche computazionalmente più complessi.

3.2.2 Scheduling di Task Periodici

Nel controllo dell'automazione industriale, i task presentano delle proprietà comuni, è possibile identificare nel processo produttivo dei *cicli di lavoro* che si ripetono periodicamente. L'ipotesi di periodicità che a breve verrà definita, ha notevole impatto sulla risoluzione dei task. Come prima ipotesi, si assume che vi è un insieme finito di n task, che possono ripetersi nel tempo, ossia possono esistere più istanze dello stesso task in istanti diversi, il generico task A_i verrà ora denotato A_i^k , dove k rappresenta la sua k -esima apparizione. Anche i parametri già definiti dipenderanno da k

parametri	intervalli
$a_i(k)$	$C_i(k)$
$s_i(k)$	$D_i(k)$
$e_i(k)$	$R_i(k)$
$d_i(k)$	$L_i(k)$

Osservazione : Quando si tratta il computation time, si intende sempre il tempo di lavoro minimo possibile per completare un task. La riduzione del computation time di un task rappresenta nel mondo reale l'utilizzo di una strumentazione differente capace di eseguire lo stesso task in tempo minore. D'altro canto, l'aumento del computation time rappresenta l'acquisto di una strumentazione *più economica* che esegue lo stesso task in tempo maggiore.

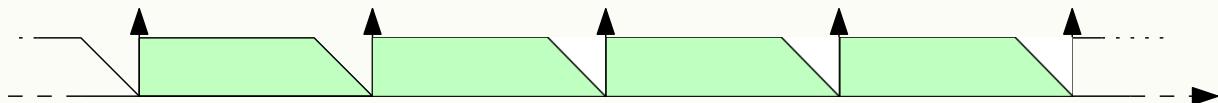
Definizione : Dato un task A_i di un insieme di task, si definisce **activation period** (oppure periodo di attivazione) il valore (in funzione della k -esima occorrenza) :

$$T_i(k) = a_i(k+1) - a_i(k)$$

rappresenta il periodo che intercorre fra la k -esima attivazione di un task e la sua $k+1$ -esima attivazione. Diremo che il task A_i è **periodico** se per ogni k , l'activation period è costante

$$\text{⌚⌚ } T_i(k) = T_i(k+1) \quad \forall k \in \mathbb{N} \quad \text{⌚⌚}$$

Vuol dire che il task si ripete periodicamente scandito da un certo periodo.



Inoltre, verranno formulate *ulteriori ipotesi* che da questo punto in poi devono considerarsi vere nell'ambito dei task periodici.

1. La deadline relativa di un task periodico sarà uguale al suo periodo di attivazione.
2. I task sono fra loro indipendenti (non ci sono blocking queue)
3. Il computation time C_i di un task è costante per tutte le sue istanze
4. Tutti i task condividono il primo activation time : $\forall i, j \in [1 \dots, n] \quad a_j(1) = a_i(1)$

Definiamo adesso i parametri in gioco quando si tratta dello scheduling di task periodici. Il generico problema di scheduling prevede

- Un insieme di n task periodici ($A_1, A_2, A_3 \dots, A_n$)
- I *Requisiti*, ossia i periodi di attivazione per ogni task ($T_1, T_2, T_3 \dots, T_n$)
- I *Vincoli*, ossia i computation time per ogni task ($C_1, C_2, C_3 \dots, C_n$)

Definizione : Dato un insieme di n task periodici con rispettivi requisiti e vincoli, si definisce **fattore di utilizzazione** il coefficiente

$$U = \sum_{i=1}^n \frac{C_i}{T_i}$$

Il denominatore di U , ossia il minimo comune multiplo dei periodi

$$\text{mcm}(T_1, T_2 \dots, T_n)$$

rappresenta il *periodo totale del sistema*, ossia il periodo con cui l'intero insieme di task si ripete ciclicamente. È quindi opportuno restringere lo scheduling solo per le prime $\text{mcm}(T_1, T_2 \dots, T_n)$ unità di tempo, in quanto per le successive sarà semplicemente una ripetizione di queste. Il numeratore di U invece rappresenta le unità di tempo in cui il processore è effettivamente utilizzato.

Dato un problema di scheduling, la **condizione necessaria** affinché sia risolvibile è che il fattore di utilizzazione sia minore o uguale ad 1.

$$\begin{cases} U \leq 1 & \text{possibilmente risolvibile} \\ U > 1 & \text{non risolvibile} \end{cases}$$

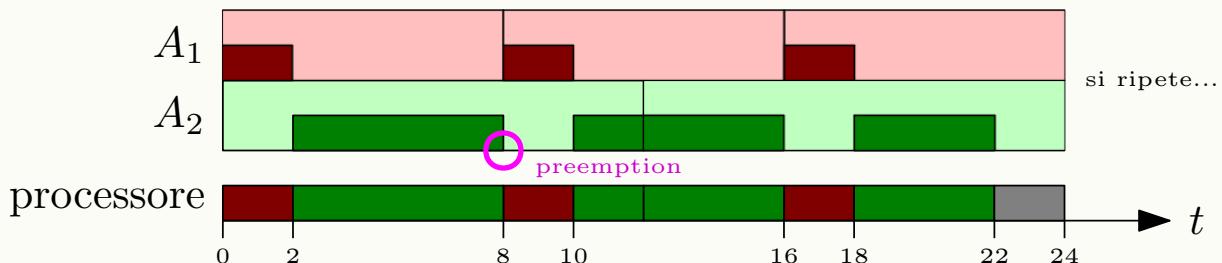
Si prenda in esempio la coppia di task

$$\begin{array}{lll} A_1 & T_1 = 8 & C_1 = 2 \\ A_2 & T_2 = 12 & C_2 = 8 \end{array}$$

Si ha

$$U = \frac{2}{8} + \frac{8}{12} = \frac{22}{24} \simeq 0.917 < 1$$

L'insieme potrebbe essere schedulabile. Verrà applicato un algoritmo preemptive che assegna maggiore priorità al task il cui activation period è minore. Verrà quindi sempre favorito il task A_1 .



Si noti come nella traccia dello scheduling il lavoro effettivo sia durato 22 t.u., proprio come il numeratore del fattore U , sono infatti rimaste 2 t.u. (evidenziate in grigio) in cui la ready queue era vuota e non c'era nulla da schedulare.

3.2.3 Utilizzo del Processore

Si è definito il fattore di utilizzazione U , esso dipende dai computation time C_i , tale valore U può essere eventualmente massimizzato. Cosa vuol dire massimizzare U ? Renderlo il più possibile vicino ad 1, finché è minore o uguale all'unità, lo scheduling è ammissibile, è quindi possibile cercare di aumentare i valori C_i , l'aumento di tali tempi equivale ad un utilizzo di una strumentazione più lenta (quindi economica) nello svolgimento dei task. È quindi di interesse del progettista cercare di "spremere" il più possibile uno scheduling cercando di massimizzare U .

Dato un algoritmo F , un insieme di n task, e rispettivi requisiti T_i , si definisce il **fattore di utilizzazione massimo**

$$U_{max}(n, T_i, F) = \max_{\substack{C_i \in \mathbb{R}^+ \\ F \text{ soluzione}}} (n, T_i) = \max_{\substack{C_i \in \mathbb{R}^+ \\ F \text{ soluzione}}} \sum_{i=1}^n \frac{C_i}{T_i} \quad \text{tale che } U_{max} \geq 1$$

Definizione : Dato un algoritmo di scheduling diremo che il processore è **completamente utilizzato** se, l'aumento di un qualsiasi C_i di un valore positivo qualsiasi ε rende lo scheduling inammissibile ($U > 1$).

Proposizione : Dato un algoritmo di scheduling, se il processore è completamente utilizzato allora il suo fattore di utilizzazione è massimale : $U = U_{max}$. (la dimostrazione è banale)

Nell'esempio con i due task (A_1, A_2) visto in precedenza, il processore è completamente utilizzato. infatti all'istante 12, il task A_2 termina esattamente in pari con la sua deadline, l'aumentare di un qualsiasi valore ε di un computation time di uno dei due task renderebbe lo scheduling inammissibile.



Esiste un altro coefficiente che caratterizza l'efficacia di un algoritmo di scheduling, ossia il **limite superiore minimo del fattore di utilizzazione**, dato uno scheduling F , esso è il minimo fra tutti i massimi fattori di utilizzazione calcolati al variare di un qualsiasi possibile insieme di task periodici.

$$U_{lsm}(F) = \min_{\substack{n \in \mathbb{N} \\ T_i \in \mathbb{R}^+}} \left(U_{max}(n, T_i, F) \right) = \min_{\substack{n \in \mathbb{N} \\ T_i \in \mathbb{R}^+}} \left(\max_{\substack{C_i \in \mathbb{R}^+ \\ F \text{ soluzione}}} \sum_{i=1}^n \frac{C_i}{T_i} \right)$$

Describe il carico computazionale massimo che può essere sicuramente gestito, infatti si ha la seguente proposizione.

Proposizione : Dati n task con rispettivi vincoli e requisiti, essi sono sicuramente schedulabili se esiste un algoritmo F tale che $U \leq U_{lsm}(F)$. Tale condizione è *sufficiente* per rendere l'insieme schedulabile.

~*~*~*~*~*~*~*~*~*~*~*~*~*~*~*~*~

3.3 Gli Algoritmi di Scheduling

In questa sezione verranno presentati diversi algoritmi per lo scheduling di task, differenti da un punto di vista della loro classificazione.

3.3.1 Rate Monotoning Priority Ordering (RMPO)

L'RMPO è un algoritmo che funziona esclusivamente se i task in questione sono tutti periodici, de facto, è un algoritmo preemptive che assegna la proporzionalità in modo inversamente proporzionale all'activation

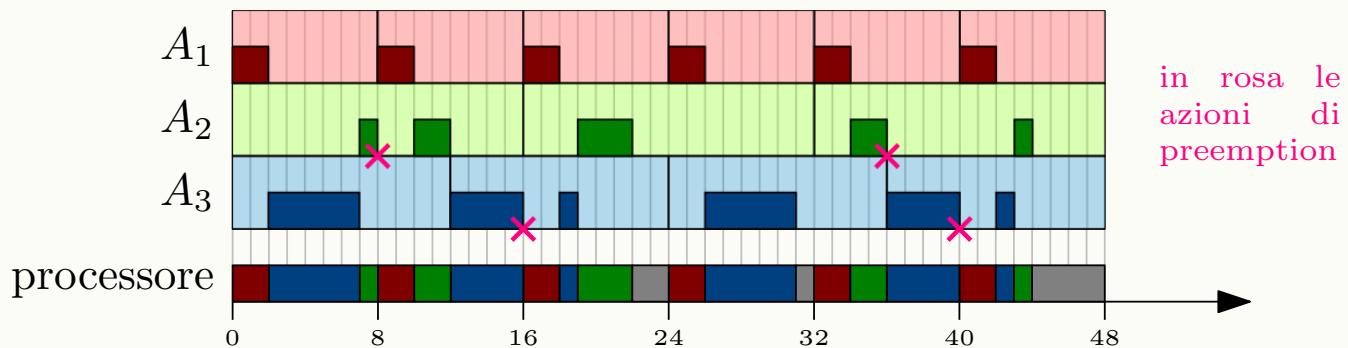
period dei task. Tale priorità non varia, lo rende quindi un algoritmo online statico. Vediamo subito un esempio di applicazione, si consideri il seguente insieme di task

$$\begin{array}{lll} A_1 & T_1 = 8 & C_1 = 2 \\ A_2 & T_2 = 16 & C_2 = 3 \\ A_3 & T_3 = 12 & C_3 = 5 \end{array}$$

Il fattore di utilizzazione è

$$U = \frac{2}{8} + \frac{3}{16} + \frac{5}{12} = \frac{41}{48} \simeq 8.8542 < 1$$

La condizione necessaria è soddisfatta. Il task A_1 avrà sempre priorità sugli altri, si procede nel disegnare la traccia dello scheduling.



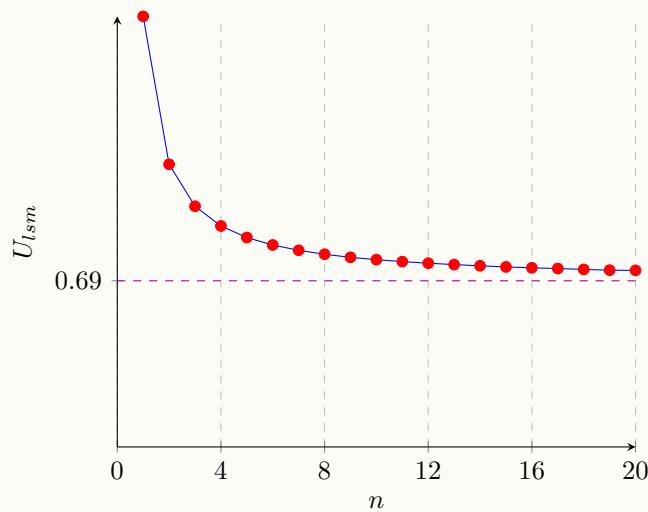
Proprietà dell'algoritmo

Proposizione 1 : l'algoritmo RMPO è il miglior algoritmo statico per lo scheduling di task periodici. Se un insieme di questo tipo non è schedulabile con RMPO, non esiste alcun altro algoritmo statico capace di risolverlo.

Proposizione 2 : per l'RMPO, dato un numero n di task, si ha che

$$U_{lsm}(\text{RMPO}) = n(2^{\frac{1}{n}} - 1)$$

Si può tracciare il grafico di tale fattore al variare di n .



Si calcola il limite per n che tende ad infinito

$$\lim_{n \rightarrow +\infty} n(2^{\frac{1}{n}} - 1) = \lim_{n \rightarrow +\infty} \frac{2^{1/n} - 1}{1/n} = \quad (3.1)$$

$$\text{de l'Hôpital} \implies \lim_{n \rightarrow +\infty} \frac{\frac{d}{dn}(2^{1/n} - 1)}{\frac{d}{dn}(1/n)} = \quad (3.2)$$

$$\lim_{n \rightarrow +\infty} \frac{2^{1/n} \ln(2)(-1/n^2)}{-1/n^2} = \quad (3.3)$$

$$\lim_{n \rightarrow +\infty} 2^{1/n} \ln(2) = \ln(2) \simeq 0.693 \quad (3.4)$$

Osservazione : Dato un qualsiasi insieme di task periodici, se il fattore di utilizzazione è minore di $\ln(2)$, allora l'insieme è sicuramente risolvibile con RMPO (condizione sufficiente).

Definizione (Relazioni Armoniche) : : Dato un insieme di n task, essi sono *legati da relazioni armoniche* se esiste un task i il cui periodo di attivazioni è multiplo di tutti gli altri periodi di attivazione.

$$\exists i \in [1, 2, \dots, n] \mid \forall j \in [1, 2, \dots, n] \ k_j T_j = T_i \text{ per qualche } k_j \in \mathbb{Z}$$

Proposizione 3 : Se un insieme di task è legato da relazioni armoniche, allora RMPO lo risolve (condizione sufficiente).

In conclusione, si è visto come RMPO sia il migliore algoritmo statico, nonostante ciò, per tutti gli insiemi di n task periodici che non sono legati da relazioni armoniche, e per cui il loro valore di utilizzazione è

$$n(2^{\frac{1}{n}} - 1) < U \leq 1$$

non è garantita la correttezza temporale (e va verificato con la traccia dello scheduling). Esiste un algoritmo le cui assunzioni per garantire la correttezza sono più larghe, ma non è un algoritmo statico bensì dinamico.

3.3.2 Earliest Deadline First (EDF)

Si tratta di un algoritmo dinamico in cui il task con maggiore priorità è quello la cui deadline è più prossima nel tempo, quindi inversamente proporzionale alla deadline assoluta. In caso di deadline assoluta identica fra 2 task, favorisce quello che è stato attivato prima. Si consideri il seguente esempio :

$$\begin{array}{lll} A_1 & T_1 = 8 & C_1 = 4 \\ A_2 & T_2 = 24 & C_2 = 6 \\ A_3 & T_3 = 12 & C_3 = 3 \end{array} \quad U = \frac{4}{8} + \frac{6}{24} + \frac{3}{12} = \frac{24}{24}$$



Proprietà dell'algoritmo

Proposizione : Se per un qualsiasi insieme di task vale che $U \leq 1$, allora è schedulabile tramite EDF (condizione sufficiente). Significa che EDF è il *miglior algoritmo* per lo scheduling di task periodici, se un insieme non è schedulabile con EDF, allora non è schedulabile da nessun altro algoritmo.

Riassumendo, dato un insieme di task, si ha che

- Se $U \leq \ln(2)$ sicuramente è schedulabile con RMPO
- Se i task sono n e $U \leq n(2^{\frac{1}{n}} - 1)$ sicuramente è schedulabile con RMPO
- Se i task sono legati da relazioni armoniche e $U \leq 1$, sicuramente è schedulabile con RMPO
- In ogni circostanza, se $U \leq 1$ sicuramente è schedulabile con EDF

EDF ha condizioni di schedulabilità più larghe e risulta più efficace di RMPO, è però computazionalmente più complesso, quindi sotto certe ipotesi può essere meglio RMPO. Inoltre

- RMPO funziona solamente con task periodici
- EDF funziona anche con task aperiodici, anche se per questi non valgono le condizioni sufficienti di schedulabilità.

Esercizio

Dato il seguente insieme di task

$$\begin{array}{lll} A_1 & T_1 = 8 & C_1 = 3 \\ A_2 & T_2 = 16 & C_2 = 3 \\ A_3 & T_3 = 12 & C_3 = 5 \end{array}$$

Mostrare che è schedulabile con RMPO, in caso contrario, mostrare lo scheduling con EDF.

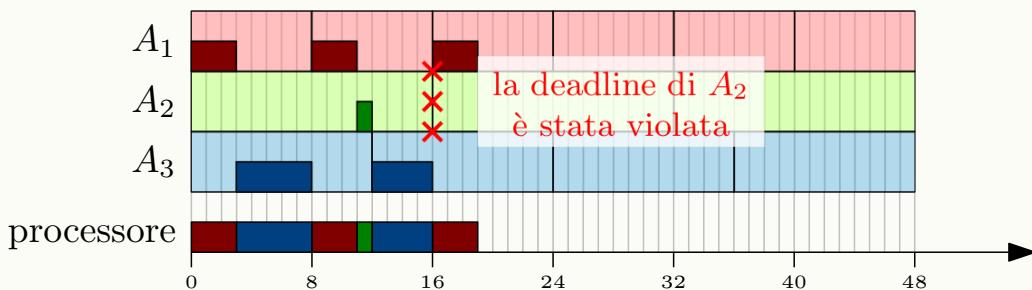
Come prima cosa, si calcola il fattore di utilizzazione

$$U = \frac{3}{8} + \frac{3}{16} + \frac{5}{12} = \frac{47}{48} < 1$$

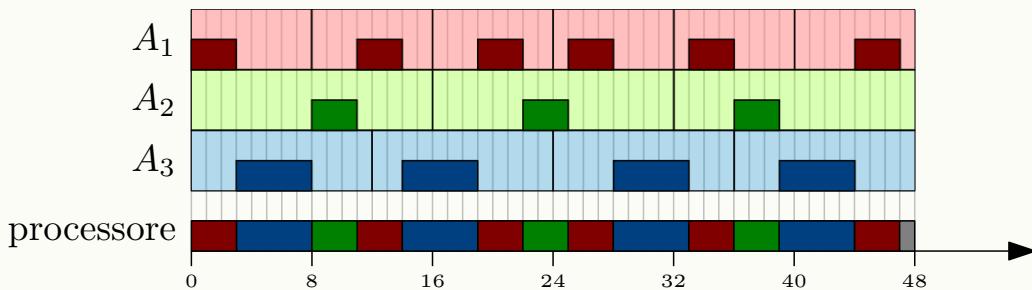
Si ha che

$$n(2^{\frac{1}{n}} - 1) = 3(2^{\frac{1}{3}} - 1) \simeq 0.779 < \frac{47}{48}$$

quindi le condizioni sufficienti non sono soddisfatte, è necessario disegnare la traccia dello scheduling.



RMPO non funziona, si tenta con EDF



Essendo il fattore di utilizzazione minore di 1 era certo che EDF sarebbe riuscito a schedulare l'insieme, lasciando l'ultima unità temporale libera.

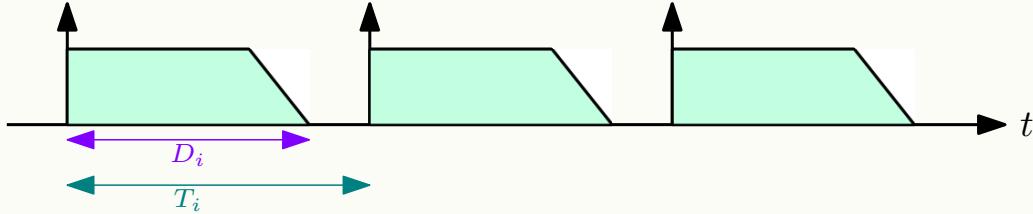
3.3.3 Deadline Monotonic Priority Ordering (DMPO)

Fin'ora l'ipotesi dei task periodici ha permesso la formulazione di algoritmi efficienti capaci di schedulare un qualsiasi insieme di task che rispetti determinati requisiti, più o meno laschi.

Si considera adesso un insieme di task **periodici generalizzati**, che differiscono dai task periodici classici per un vincolo : Non è necessariamente vero che la deadline relativa ed il activation period coincidano. Un insieme di task rispetta ora le seguenti ipotesi

1. La deadline relativa di un task periodico sarà **minore o uguale** al suo periodo di attivazione.
2. I task sono fra loro indipendenti (non ci sono blocking queue)
3. Il computation time C_i di un task è costante per tutte le sue istanze
4. Tutti i task condividono il primo activation time : $\forall i, j \in [1 \dots, n] \quad a_j(1) = a_i(1)$

$$T_i \geq D_i$$



Bisogna definire il *coefficiente di utilizzazione relativo* :

$$U_{rel} = \sum_{i=1}^n \frac{C_i}{D_i} \geq U = \sum_{i=1}^n \frac{C_i}{T_i}$$

L'algoritmo *DMPO* generalizza l'*RMPO*, opera nel campo dei task periodici generalizzati ed assegna un valore di priorità inversamente proporzionale alla deadline relativa di ogni processo. È un algoritmo preemptive e statico perché le deadline relative sono costanti.

Proposizione 1 : È il migliore algoritmo statico per i task periodici generalizzati.

Proposizione 2 : Se un algoritmo statico risolve un insieme di task periodici generalizzati, anche DMPO lo risolve.

Proposizione 3 : (Condizione sufficiente) un insieme di n task periodici generalizzati è *sicuramente schedulabile* se

$$U_{rel} \leq n(2^{1/n} - 1)$$

Osservazione : La condizione sufficiente di schedulabilità è più stretta rispetto a quella del RMPO dato che $U \leq U_{rel}$

$$\begin{cases} U \leq n(2^{1/n} - 1) & \text{condizione RMPO} \\ U_{rel} \leq n(2^{1/n} - 1) & \text{condizione DMPO} \end{cases} \quad U \leq n(2^{1/n} - 1) \not\Rightarrow U_{rel} \leq n(2^{1/n} - 1)$$

DMPO può fallire dove RMPO riesce.

3.3.4 Time Scheduling (TS)

L'algoritmo *TS* opera nel campo dei task periodici classici ($T_i = D_i$) ed è un algoritmo offline e non preemptive, è quindi estremamente semplice in termini di complessità computazionale e garantisce un alto grado di determinismo.

L'algoritmo prevede una *discretizzazione* dell'asse temporale in intervalli di tempo regolari chiamati **time slices**.



Si definiscono due coefficienti

- **minor cycle** : $\text{MCD}(T_i)$
- **major cycle** : $\text{mcm}(T_i)$

Nota : Ha senso parlare di MCD e mcm nel contesto degli activation period che sono intervalli temporali? Anche se nel mondo fisico $T_i \in \mathbb{R}^+$, è più che ammissibile permettersi un rilassamento ingegneristico e trattare i periodi temporali come numeri in \mathbb{N} , tutti i periodi saranno multipli interi di una determinata time unit.

Per applicare TS, è necessario che

$$\forall i \quad C_i \leq \text{MCD}(T_1, T_2, \dots, T_n)$$

L'asse temporale viene suddiviso in time slices grandi quanto il minor cycle, una volta fatto ciò, è possibile *schedulare in maniera arbitraria* i task sulle time slices in modo tale che

- ogni time slices ha almeno uno o più task
- un task viene interamente schedulato su una singola time slices, non può essere suddiviso

Lo scheduling dell'intero insieme sarà periodico, di periodo uguale al major cycle. **Esempio**

$$\begin{array}{lll} A_1 & T_1 = 8 & C_1 = 2 \\ A_2 & T_2 = 16 & C_2 = 4 \\ A_3 & T_3 = 32 & C_3 = 6 \end{array} \quad U = \frac{22}{32} < \ln(2) \text{ RMPO lo risolve}$$

Si vuole applicare TS, si ha che il minor cycle è $\text{MCD}(8, 16, 32) = 8$ ed il major cycle è $\text{mcm}(8, 16, 32) = 32$.



Esistono più scheduling equivalenti, quello corretto non è univoco.



L'algoritmo time scheduling è estremamente semplice ed immediato nella computazione, è però caratterizzato anche da **poca flessibilità** e **poca robustezza**, essendo lo scheduling deciso a priori un cambiamento nella trama dovuto a ritardi non prevedibili di task potrebbe causare la violazione del vincolo di correttezza temporale, è quindi un algoritmo da applicare in contesti *strettamente deterministicici* e gestisce esclusivamente task periodici.



3.4 Scheduling di Task Misti

Riguardo la piramide CIM : è vero che nel livello di campo i task sono relativi ad attuazioni e rilevazioni da parte di macchine e sensori, task strettamente rigidi, deterministici, immersi in un ciclo di produzione che li rende strettamente periodici.

Salendo sulla piramide, i tempi di reazione si allargano ed i task assumono una connotazione di controllo e gestione piuttosto che azionamento meccanico/elettrico, persistono sempre cicli periodici, ma potrebbero essere inclusi anche dei task **aperiodici**, il cui avvenimento può essere condizionale ed asincrono, quindi non predeterminato. Un esempio di task aperiodico può essere la gestione di un allarme oppure la gestione dell'input da parte di un utente.

Un insieme di task può quindi contenere task periodici e task aperiodici, si dice che l'insieme di task è *misto*, come prima cosa, in base alla natura del task, bisogna suddividere i task aperiodici in

- aperiodici hard real time (Es. gestione di allarmi riguardanti l'incolumità fisica degli operatori)
- aperiodici soft real time (Es. interazione con il display da parte dell'utente)

Idea : Si vogliono trasformare i task aperiodici in task periodici equivalenti

Ipotesi :

1. Dato un task aperiodico hard real time A_i , si assume di conoscere il *minimo intervallo di occorrenza* di esso denotato T_i^{min} , è impossibile che A_i entri due volte nella coda dei task pronti in due istanti che distano meno di T_i^{min} .
2. Si assume inoltre di essere a conoscenza del *massimo tempo di computazione* C_i^{max} , è impossibile che il task occupi la CPU per più di C_i^{max} t.u.

Dati dei task periodici

$$(n, T_1 \dots, T_n) \\ (C_1 \dots, C_n)$$

e dei task aperiodici trasformati in periodici

$$(m, T_1^{min} \dots, T_m^{min}) \\ (C_1^{max} \dots, C_m^{max})$$

Si considera l'insieme dei task che rappresenta un problema equivalentemente periodico

$$(n + m, T_1 \dots, T_n, T_1^{min} \dots, T_m^{min}) \\ (C_1 \dots, C_n, C_1^{max} \dots, C_m^{max})$$

i task aperiodici hard real time vengono resi task periodici equivalenti

È chiaro che la CPU dedicherà delle unità di tempo a task aperiodici come se fossero periodici, causando un inevitabile spreco di computazione nel caso questi non dovessero sussistere.

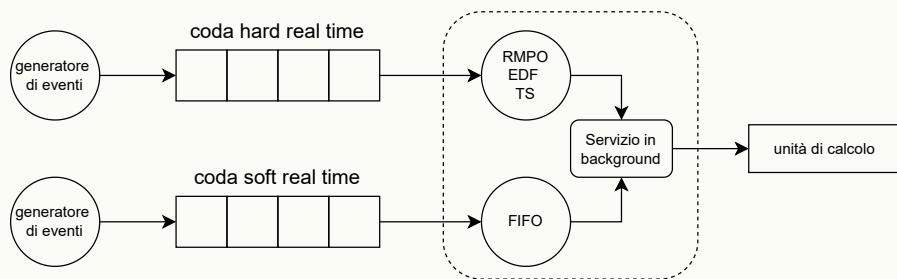
I task soft real time (che sono altrettanto diffusi) possono essere gestiti in maniera best effort.

~*~*~*~*~*~*~*~*~*~*~*~*~*~

3.5 Servizio in Background

Per i task periodici (e quelli aperiodici da gestire in maniera hard real time) sono state coperte tutte le casistiche. Questa sezione (e quella successiva) tratteranno le metodologie di scheduling dei task aperiodici da gestire in maniera best effort, in cui è possibile violare la deadline.

Dato un insieme di n task (periodici, oppure aperiodici ma resi equivalentemente periodici) si considerano m task aperiodici aggiuntivi, da gestire in maniera soft real time, l'idea del *servizio in background*, è quella di considerare un task aggiuntivo che occuperà la CPU ogni qual volta quest'ultima è libera dai task hard real time. Se non differentemente specificato, i task soft real time saranno gestiti con una politica FIFO. Lo schema dello scheduler si complica leggermente, vi saranno adesso due code di task pronti all'esecuzione.



Esempio di Scheduling

In un sistema di controllo real time è necessario schedulare i seguenti task periodici in maniera hard real time

$$\begin{array}{lll} A_1 & T_1 = 8 & C_1 = 2 \\ A_2 & T_2 = 16 & C_2 = 4 \end{array}$$

è inoltre presente un task periodico i cui dati misurati (tempo minimo di occorrenza e tempo massimo di completamento) sono

$$A_3 \quad T_3^{min} = 32 \quad C_3^{max} = 8$$

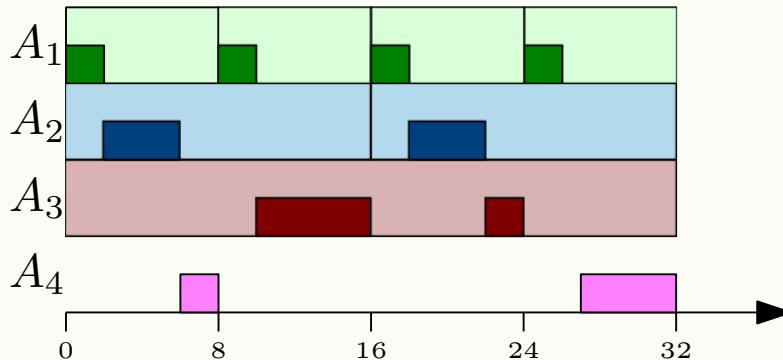
è inoltre presente un task A_4 da gestire in maniera soft real time tramite il servizio in background, i cui activation time, completion time e deadline assoluta sono

$$a_4 = 9 \quad c_4 = 9 \quad D_4 = 55$$

Per i task periodici verrà utilizzato l'algoritmo RMPO. Per i task soft real time, una politica FIFO. Si calcola il fattore di utilizzazione:

$$U = \frac{2}{8} + \frac{4}{16} + \frac{8}{32} = \frac{24}{32} = 0.75 < 1$$

I task sono legati da relazioni armoniche, sicuramente l'insieme è schedulabile. La trama dello scheduling è la seguente



Il task A_4 , che rappresenta il processo in background, nelle prime 32 unità di tempo viene eseguito per 7 unità di tempo, l'ultima unità di tempo la esaurira nell'istante 40, rispettando la deadline assoluta. Il servizio in background è molto semplice da applicare, ma esistono metodologie che riescono a massimizzare la responsività dei processi soft real time, rispettando comunque i vincoli di quelli hard real time.

~*~

3.6 Processo Server

Si considera un task periodico aggiuntivo detto *processo server*, di activation period e completion time

$$T_{server} \quad C_{server}$$

Viene aggiunto all'insieme di task hard real time, è un processo "virtuale", il tempo di esecuzione che occupa sulla CPU sarà dedicato ai task soft real time. I valori T_{server} , C_{server} vanno selezionati con cura in modo che il fattore di utilizzazione rimanga minore di uno, senza intaccare la schedulabilità hard real time dei restanti task.

3.6.1 Polling Server

L'algoritmo polling server considera ogni singola istanza del processo server, e ne determina *dinamicamente* il completation time, studiando lo stato della coda dei task soft real time.

Sia t_k l'istante di attivazione di un'istanza del processo server, e siano $\{C_1, C_2 \dots, C_n\}$ i tempi di completamento dei task nella coda di attesa soft real time nell'istante t_k . Il completation time calcolato dinamicamente sarà

$$\min(C_{server}, \sum_{i=1}^n C_i)$$

In tal maniera, il processo server utilizzerà solamente il tempo strettamente necessario per occupare la CPU, se non ci sono task soft real time in attesa, il completation time dinamico sarà nullo.

Esempio di Scheduling

L'insieme dei task hard real time, compreso il processo server (da eseguire con RMPO) è il seguente

$$\begin{array}{lll} A_1 & T_1 = 8 & C_1 = 4 \\ A_2 & T_2 = 16 & C_2 = 2 \\ A_{server} & T_{server} = 12 & C_{server} = 3 \end{array}$$

è presente un solo task A_3 soft real time di cui

$$a_3 = 14 \quad C_3 = 5 \quad D_3 = 47$$

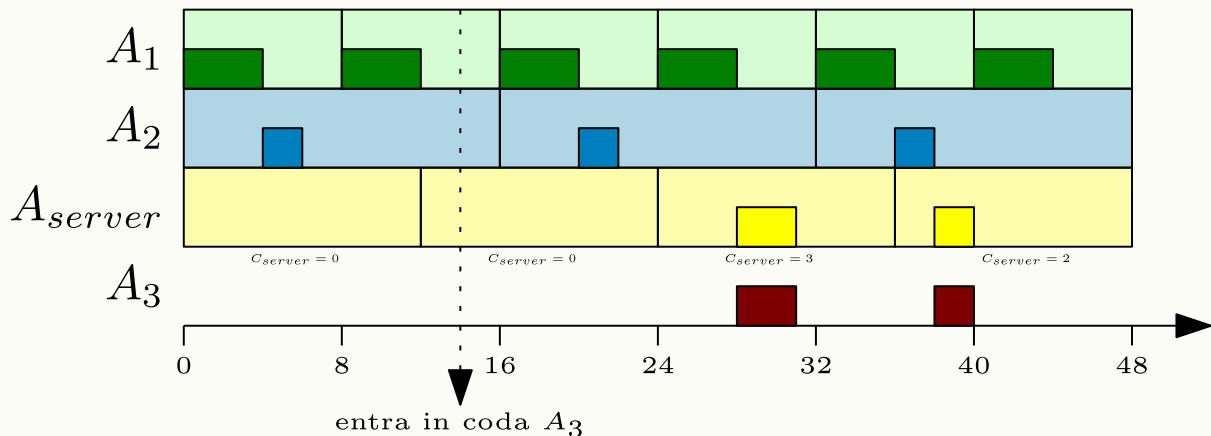
L'insieme dei task hard real time ha fattore di utilizzazione

$$U = \frac{4}{8} + \frac{2}{16} = \frac{10}{16} = 0.625 < \ln(2)$$

è quindi sicuramente schedulabile con RMPO. Considerando anche il processo server si ha

$$U' = \frac{10}{16} + \frac{3}{12} = 0.875 < 1$$

Le condizioni necessarie di schedulabilità non sono violate. Si disegna dinamicamente la trama dello scheduling.

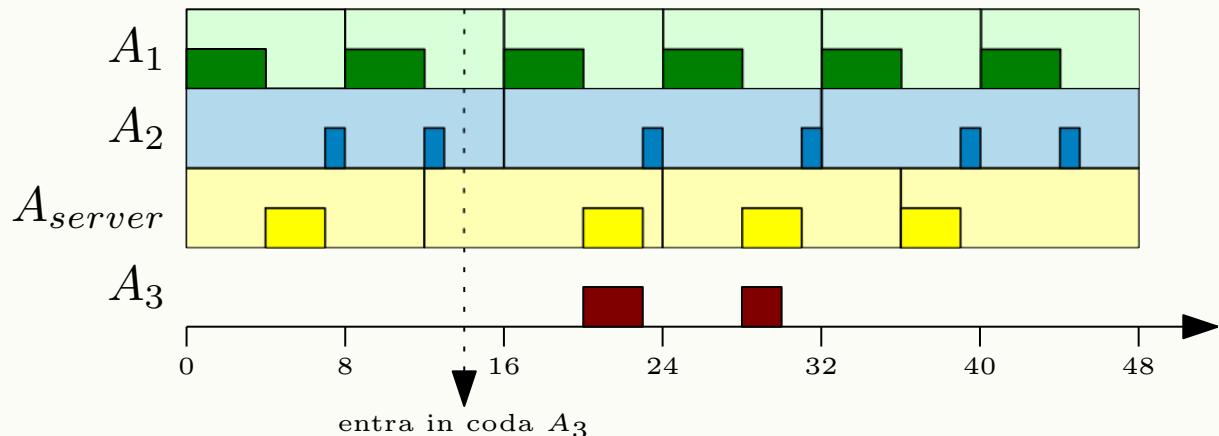


Si considerino separatamente i 4 periodi di attivazione del processo server

- $t = 0$: il processo server si attiva, non ci sono task nella coda FIFO, quindi il suo completation time per tutto lo slot di tempo sarà nullo e verrà ignorato.
- $t = 12$: il processo server si attiva, non ci sono task nella coda FIFO (dato che il task A_3 arriverà 2 istanti dopo), quindi il suo completation time per tutto lo slot di tempo sarà nullo e verrà ignorato.
- $t = 24$: in coda vi è un processo con completation time pari a 5. Il minimo fra 5 ed il valore C_{server} stabilito è proprio $C_{server} = 3$, quindi sarà concesso un tempo di esecuzione sulla CPU pari a 3 unità di tempo. Il tempo rimanente al task A_3 è di 2 unità di tempo
- $t = 36$ in coda vi è un processo con completation time pari a 2. Il minimo fra 2 ed il valore C_{server} stabilito è 2, quindi sarà concesso un tempo di esecuzione sulla CPU pari a 2 unità di tempo, ed il task A_3 sarà completato.

3.6.2 Deferrable Server

Il *deferrable server* è un'alternativa più statica rispetto il polling server, in pratica, è identico, ma il completion time del processo server è statico e rimane quello deciso a priori, cercando di dare a tale processo la *massima priorità possibile*, sempre rispettando il vincolo real time per il restante insieme di task. Si consideri l'esempio precedente in cui si è applicato il polling server, applicando il deferrable server la trama sarà la seguente:



Il processo server diventa in tutto e per tutto un task periodico. Nonostante aumenti la responsività dello scheduling dei task soft real time, il tempo sprecato dal processo server sulla CPU è maggiore.

»»»»»»»»»»

3.7 Esercizi sullo Scheduling

Esercizio 1

In un sistema di AUTOMAZIONE devono essere GARANTITI tre task:

1. A livello di campo è necessario garantire il processo di lettura dei dati da sensore che impiega 8 t.u. per essere elaborato e che deve essere ripetuto ogni 16 t.u.;
2. A livello di coordinamento è necessario garantire un task meccanico che deve essere ripetuto ogni 48 t.u. e la cui elaborazione necessita 12 t.u.;
3. A livello di campo è necessario garantire il processo di elaborazione della legge di controllo e di attuazione della relativa azione di intervento. Tale processo impiega 6 t.u. per essere elaborato e deve essere ripetuto ogni 24 t.u.;

Ipotizzando di volere un sistema di controllo HARD REAL TIME e di avere a disposizione di un sistema MONOPROCESSORE, calcolare il fattore di utilizzazione e tracciare il grafico dello scheduling dei task mediante algoritmo EDF.

Soluzione 1

Il fattore di utilizzazione è

$$U = \frac{8}{16} + \frac{12}{48} + \frac{6}{24} = \frac{24 + 12 + 12}{48} = 1$$

Le condizioni necessarie e sufficienti per la schedulabilità con EDF sono rispettate.

La trama dello scheduling è la seguente

**Esercizio 2**

Una ditta agroalimentare deve progettare un sistema di AUTOMAZIONE di un impianto di packaging il cui sistema di controllo è basato su un singolo PROCESSORE che deve GARANTIRE L'esecuzione di tre task a livello di coordinamento:

1. L'imballatrice viene attivata ogni 16 t.u. e impiega 3 t.u. a completare il task;
2. L'impacchettatrice viene attivata ogni 4 t.u. e impiega 1 t.u. a completare il task;
3. L'etichettatrice viene attivata ogni 8 t.u. e impiega 2 t.u. a completare il task.

Ipotizzando di avere implementato nel processore un algoritmo TIMELINE SCHEDULING, studiarne la schedulabilità.

Soluzione 2

Il fattore di utilizzazione è

$$U = \frac{3}{16} + \frac{1}{4} + \frac{2}{8} = \frac{3+4+4}{16} = \frac{11}{16}$$

Le condizioni necessarie e sufficienti per la schedulabilità con TS sono rispettate. Si hanno

$$\begin{aligned} \text{minor cycle} &= 4 \\ \text{major cycle} &= 16 \end{aligned}$$

La trama dello scheduling è la seguente



CAPITOLO

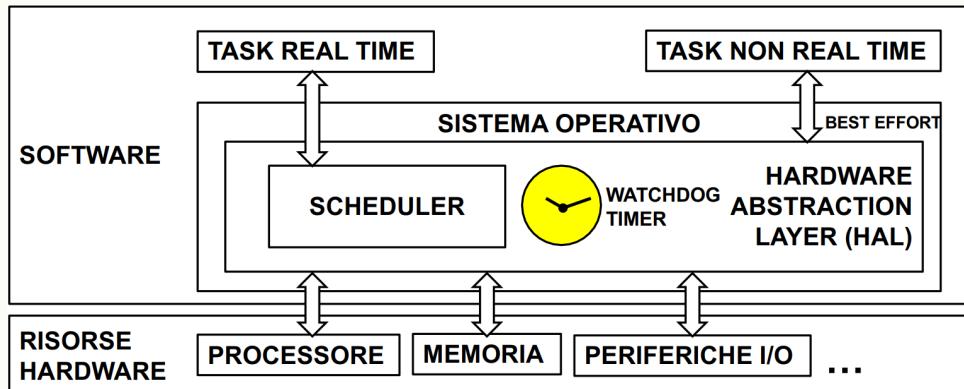
4

IMPLEMENTAZIONE DI SISTEMI REAL TIME

4.1 Hardware Abstraction Layer

I sistemi di controllo devono essere implementati via software su appositi dispositivi. Il sistema operativo, è la componente software che interfaccia la logica del sistema definita dall'utente, con l'hardware, funge da interfaccia. Definiamo **HAL** (Hardware Abstraction Layer) il componente che serve a disaccoppiare il sistema operativo dalle possibili risorse hardware, qui è contenuto lo *scheduler*, che dovrà gestire in maniera hard real time i task definiti dal sistema di controllo. L'HAL gestirà i task soft real time con una politica best effort.

È inoltre presente nell'HAL, anche il cosiddetto **watchdog timer**, un componente che si occupa di tenere traccia delle deadline dei task hard real time.



In passato i sistemi di controllo erano di tipo **event driven**, allo scaturire di determinate condizioni fisiche, veniva subito rilevato un *evento* che faceva agire immediatamente il sistema di controllo (implementato fisicamente). Ad esempio, un derivatore pneumatico alla rilevazione della variazione di una grandezza, scaturiva immediatamente un attuazione.

Uno scheduler event driven quindi deve poter schedulare un task non appena l'evento che lo scaturisce si verifica.

- Vantaggi

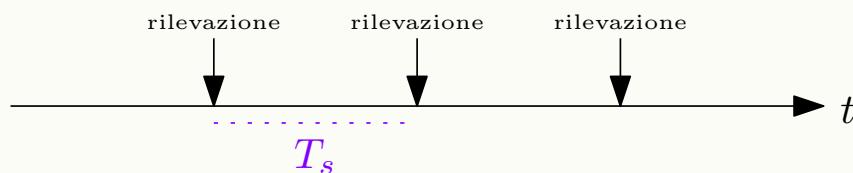
- Intuitivo: ogni task viene mandato allo scheduler non appena l'evento che lo attiva occorre;
- Semplice da usare: ad ogni task può essere associata una priorità e l'algoritmo di scheduling pensa a soddisfare anche vincoli hard real time.

- Svantaggi

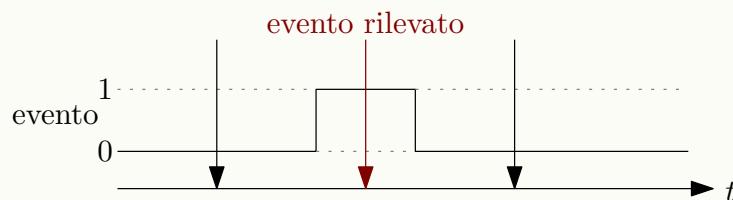
- Complesso da realizzare: definire un algoritmo di scheduling generale che risolva il problema della programmazione concorrente è difficile ed oneroso se non si conoscono a priori le caratteristiche dei task in ingresso

Se l'unità di elaborazione è digitale, l'approccio event driven è intrinsecamente irrealizzabile. Un sistema digitale è regolato da un clock, e non è in costante ascolto di eventi, bensì, avviene un controllo periodico ed il tempo è discretizzato.

Tale approccio è detto **time driven** ed è quello applicato nei sistemi di controllo odierni. Il periodo di tempo che intercorre tra due rilevazioni consecutive è detto *periodo di rilevazione* e si indica con T_s .

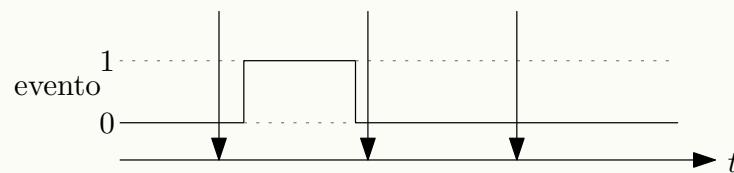


Gli eventi sono asincroni e vanno gestiti in maniera sincrona. Un evento viene associato ad un segnale booleano (che può valere 0 o 1), ed indica la sua presenza o meno.

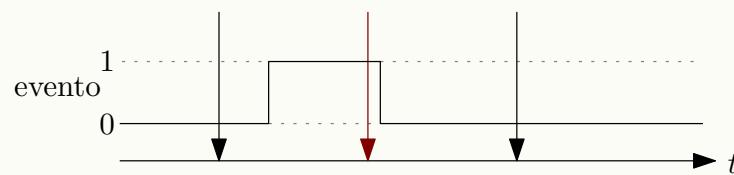


Ci sono 3 principali problemi relativi ai sistemi time driven

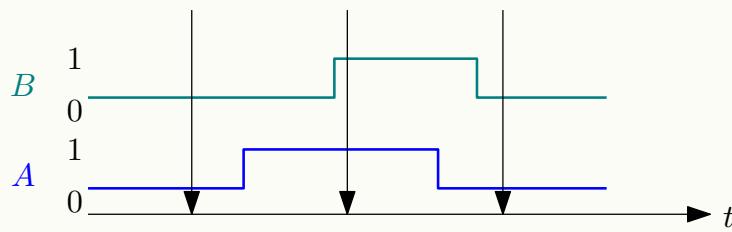
- **problema 1:** un evento in un sistema di questo tipo potrebbe avvenire e non essere osservato. Ciò è possibile se il tempo di attivazione di tale evento è minore al periodo di rilevazione.



- **problema 2:** i task sono soggetti ad un ritardo nella rilevazione, dato che questa può avvenire diversi istanti dopo l'avvenimento di un evento.



- **problema 3:** nonostante un task *A* potrebbe essere scaturito prima di un task *B*, la rilevazione potrebbe avvenire in seguito alla creazione di entrambi, vedendoli come se fossero stati generati nello stesso momento, causando un problema nell'ordine di occorrenza.



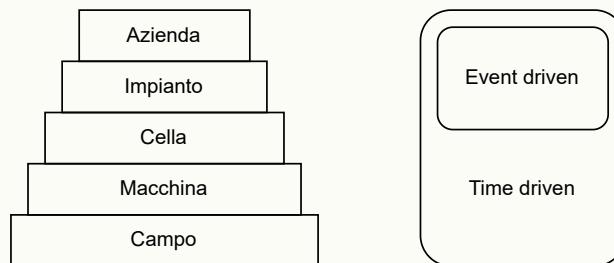
I sistemi time driven sono *semplici da implementare* in quanto è sufficiente abilitare un timer per il rilevamento periodico, inoltre, è necessario rendere il periodo di rilevazione molto piccolo rispetto il periodo di vita media di un generico task per aumentare la reattività. Nonostante questo, i problemi evidenziati riducono il campo di azione di un controllo di questo tipo.

4.1.1 Sistemi di Automazione Real Time

Nella piramide CIM, il controllo è differente in base al livello

- a livello di impianto ed azienda, il controllo avviene su sequenze logiche per la supervisione, i task possono essere di vario tipo, come task di log, di presentazione di informazioni e di gestione di allarmi. I task sono prevalentemente aperiodici.
- nel livello di coordinamento si controllano sequenze logiche di task che possono essere periodici ed aperiodici.
- nel livello di campo vi è il controllo digitale di variabili analogiche, i task sono prevalentemente periodici.

A livello di coordinamento sono presenti task misti, pertanto è intuitivo pensare di usare sistemi di controllo real time event driven. Ma da un punto di vista implementativo è molto più conveniente realizzare sistemi di controllo finalizzati all'Automazione completamente time driven.



❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖

4.2 Sistemi Operativi Real Time

I sistemi operativi più diffusi (Windows, Linux, Mac OS) non sono adatti per gestire sistemi di controllo real time. Il problema principale risiede nella impossibilità di determinare il massimo tempo di esecuzione di un task (processo o thread che sia). Un computer non industriale è equipaggiato con risorse che bloccano la CPU e rendono difficilissimo gestire il determinismo dello scheduler implementato nel Kernel:

- Periferiche di I/O (Universal Serial Bus - USB);
- DMA (Direct Memory Access) dell'Hard Disk;
- CACHE della CPU (quando si svuota la CPU non può lavorare);
- Memoria Virtuale (paging);
- IRQ (Interrupt Request) da parte di periferiche PCI;
- ACPI (Advanced Configuration and Power Interface) cambia la frequenza della CPU.

Per rendere un Sistema Operativo Real Time è necessario mettere mano al codice del suo scheduler e del suo HAL (Hardware Abstraction Layer). Ma questo è possibile esclusivamente se il Kernel del sistema operativo è Open Source.

Alcuni noti sistemi operativi per il controllo real time sono

Nome	Caratteristiche
ChibiOS	Open Source ; Sistemi Embedded
TinyOS	Open Source; Wireless Sensor Nodes
RTAI Linux	Open Source (italiano); Computer Industriali; Linux based
BeRTOS	Open Source (italiano); Sistemi Embedded; Arduino
free RTOS	Open Source; Cross platform
Ethernut	Open Source; Sistemi embedded dedicati
milOS	Open Source; Sistemi embedded

❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖

4.3 Controllori Embedded

Abbiamo visto che un sistema di controllo real time deve agire a livello di campo, di coordinamento e di conduzione per eseguire tutti i task necessari al corretto funzionamento del sistema complesso. In Automazione i sistemi di controllo Real Time, centralizzati e di piccola taglia possono essere realizzati tramite controllori embedded che eseguono opportuni programmi software sviluppati per il controllo logico sequenziale, questi sono largamente adottati nel livello di campo. Salendo al livello di macchina o cella, i sistemi di controllo sono più complessi, gestiscono un maggior numero di segnali e sono realizzati tramite *PLC* (Programmable Logic Controller). Nel livello di impianto i sistemi operanti sono detti SCADA (Supervisory Control and Data Acquisition).

Definizione : Un qualsiasi sistema realizzato tramite una singola scheda elettronica viene chiamato **sistema embedded**.

Definizione : Un sistema di controllo realizzato tramite un embedded system viene chiamato **controllore embedded**.

Un embedded controller contiene al suo interno tutto il necessario sia per connettere il controllore al sistema da controllare, sia per eseguire gli algoritmi di controllo definiti dall'utente, viene progettato o scelto in maniera tale che la configurazione hardware e software sia ad-hoc rispetto al problema di Automazione da risolvere. Ciò richiede la conoscenza a priori dei compiti da eseguire.



- vantaggi :** Noto a priori il problema di automazione, l'embedded controller permette di ridurre l'hardware, lo spazio, i consumi energetici, i tempi di realizzazione ed il costo necessari a realizzare il sistema di controllo.

- **svantaggi** : Di contro, l'embedded controller è caratterizzato da scarsa flessibilità, bassissima estendibilità e difficile intercambiabilità

L'uso degli embedded controller nell'ambito della Automazione si è diffuso moltissimo soprattutto a livello di campo, dove i task del sistema di controllo sono ripetuti e noti a priori. Un controllore embedded presenta le seguenti componenti:

- CENTRAL PROCESSING UNIT (CPU): È il processore che si occupa di eseguire sia il Basic I/O System (BIOS), ovvero il sistema operativo, che il programma realizzato dall'utente che implementa l'algoritmo di controllo. Può essere un microprocessore general purpose (numeri interi), oppure un digital signal processor (numeri interi e in virgola mobile), o un field programmable gate array (funzioni logiche)
- MEMORIA RAM: È la memoria volatile necessaria per mantenere lo stato del programma, ovvero tutti i dati temporanei necessari durante l'elaborazione degli algoritmi di controllo logico sequenziale.
- MEMORIA (EPP)ROM: È la memoria non volatile necessaria per memorizzare permanentemente il sistema operativo (ROM) e il programma utente ([EEP]ROM).
- GESTIONE TEMPORIZZAZIONI: Gestione dei timer utili per la sincronizzazione e la temporizzazione delle attività della CPU.
- GESTIONE RETE: Contiene i protocolli di base (di livello fisico e di accesso al mezzo trasmissivo) per gestire la comunicazione con altri dispositivi di controllo (embedded e non).
- GESTIONE USCITE: Contiene una serie di circuiti per la generazione di segnali analogici (DAC) e digitali.
- ACQUISIZIONE SEGNALI DIGITALI: Circuiti per l'accoppiamento con i segnali digitali in ingresso (ad es. TTL 0-5V, CMOS 0-3/15V, OPTO)
- ACQUISIZIONE SEGNALI ANALOGICI e ADC: Circuiti per l'accoppiamento con i segnali analogici in ingresso, l'eventuale multiplexer e il convertitore A/D (ADC) per il campionamento e la quantizzazione. È il componente più costoso del controllore.
- GESTIONE I/O: Circuiti per l'accoppiamento con bus dedicati o schede di espansione.
- GESTIONE INTERRUPT: Circuiti per la rilevazione degli eventi.

Definizione : Un embedded controller realizzato tramite un singolo circuito integrato viene chiamato **microcontrollore**. Ha senso prendere in considerazione l'utilizzo di microcontrollori esclusivamente in quei sistemi di Automazione che richiedano:

- notevole riduzione degli ingombri (cellulari, elettrodomestici, servizi di rete, centraline elettroniche, wearable devices, etc.);
- numero limitato di segnali di ingresso / uscita (digitali o analogici);
- basso consumo energetico (alimentazione a batteria);
- HMI minimale (ad es. touchscreen, tastiera, display LCD, LED) o nulla;
- interoperabilità e integrazione con altri dispositivi limitata o nulla.

4.3.1 Arduino e Raspberry PI

Arduino è una piattaforma open-source per la prototipazione elettronica, basata su hardware e software flessibili e facili da usare. È pensata per artisti, designer, hobbisti e chiunque sia interessato a creare oggetti o ambienti interattivi. Arduino è una iniziativa finalizzata alla definizione di requisiti hw/sw per la costruzione open-source di controllori embedded operanti a livello di campo. Il successo dell'iniziativa si basa su

- Possibilità di scaricare gratuitamente gli schemi hardware dal sito apposito, relativi a tutti i modelli delle BOARD Arduino;

- Possibilità di costruire gratuitamente (no royalties) delle SHIELD Arduino da collegare meccanicamente con le BOARD al fine di estendere le capacità e le funzionalità della stessa;
- Possibilità di scaricare gratuitamente l'SDK per iniziare da subito a programmare le BOARD Arduino e le sue SHIELD ufficiali.

Il Raspberry PI, è un computer compatto ed economico, delle dimensioni di una carta di credito, che può essere collegato a un monitor o a un televisore e utilizza una tastiera e un mouse standard. Raspberry Pi è un mini-PC perfetto per sviluppare sistemi di controllo a livello di coordinamento o di supervisione. Il successo dell'iniziativa si basa sulla possibilità di costruire sistemi di automazione low-cost (ad es. domotica).

..*..*..*..*..*..*..*..*..*..*..*..*..*..

4.4 PLC

Quando si deve realizzare un sistema di controllo logico sequenziale caratterizzato da:

- complessità di calcolo degli algoritmi molto elevata;
- elevato numero di ingressi e/o uscite analogiche/digitali;
- HMI complesse ed articolate;
- interoperabilità con altri sistemi di controllo ed informativi

Si preferisce sostituire i controllori embedded con i *controllori a bus*.

Definizione : Un *bus* è un insieme di linee di trasporto di energia e di informazione (in generale di tipo elettrico) che permettono la comunicazione tra più dispositivi.

Per identificare un BUS è pertanto necessario fissare il numero di linee, definire le funzionalità offerte da ciascuna linea, i protocolli di comunicazione usati dai dispositivi interconnessi e le interfacce meccaniche. In una architettura a bus, ad un modulo principale ospitante il processore, vengono connessi tutti gli altri moduli necessari a comporre il controllore (memoria, moduli I/O, periferiche di HMI, interfacce di rete, schede dedicate, etc.). È composto da diverse linee

- linee dati
- linee indirizzi
- linee di alimentazione
- linee per la comunicazione



I **PLC** sono controllori basati su architettura a bus e hanno riscosso ampio successo in ambito della automazione industriale. La programmazione dei PLC è stata standardizzata secondo le norme IEC61131-3. Essi possono essere programmati in diversi modi

- *Ladder Diagram*: è un linguaggio grafico, ottenuto come trasposizione informatica dei quadri a relè.
- *Functional Block Diagram*: è un linguaggio grafico, ottenuto come trasposizione dei diagrammi circolari in cui le interconnessioni rappresentano i percorsi dei segnali che collegano i vari componenti. I blocchi rappresentano le singole operazioni logiche.
- *Sequential Functional Chart*: è un linguaggio grafico, ottenuto applicando un formalismo grafico per la descrizione di operazioni logiche sequenziali e formalismi grafici propri di altri linguaggi di programmazione. utilizzato per descrivere in maniera orientata alla progettazione sistemi complessi di automazione.
- *Instruction list*: è un linguaggio di programmazione di basso livello molto simile all'assembler. Le istruzioni sono costituite da un operatore e da un solo operando e fanno riferimento ad un registro di memoria. I formalismi adottati possono essere molto differenti in quanto fissati dal produttore dell'hardware per il PLC.
- *Structured text*: è un linguaggio di programmazione strutturato ad alto livello con un formalismo che si ispira al Basic e al Pascal. È adatto alla rappresentazione di procedure complesse che non potrebbero essere descritte con i linguaggi grafici.

~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~

4.5 Sequential Functional Chart

L'SFC, è un linguaggio *grafico* ed *autoesplicativo* adottato nella descrizione del comportamento dei PLC, è nato in Francia nel 1975 come strumento formale per la progettazione di tipo logico-sequenziale, è de facto uno strumento utile sia per la programmazione dei PLC, sia per la progettazione del loro funzionamento. Ad oggi SFC è stato incluso nell'insieme dei diagrammi UML, denominato diagramma degli stati.

4.5.1 Elementi di Base

Nel livello di campo vengono controllati i singoli elementi tramite appositi sistemi di controllo modellizzati come sistemi continui nel tempo. Salendo per i livelli della piramide CIM, il sistema diventa più complesso, i vincoli temporali si fanno laschi ed il tempo è considerato discreto, al livello di coordinamento è addirittura trascurabile rendendo le sequenze operative esclusivamente logiche.

Definizione : definiamo **stato** una *condizione operativa* del sistema complesso da modellare, è invariante e può cambiare esclusivamente in seguito ad un evento. Ogni stato del sistema può trovarsi in una delle due possibili condizioni

- attivo
- inattivo

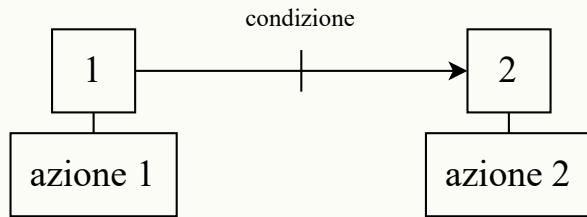
L'SFC si occupa di modellare le successioni di condizioni operative, un sistema può avere anche infiniti stati, ma questi devono essere numerabili.

Definizione : Un'operazione che viene eseguita quando uno stato è attivo è detta **azione** dello stato. Ogni stato è caratterizzato da un determinato insieme di azioni da svolgere quando questo è attivo.

Definizione : Definiamo **transizione** il passaggio da uno stato ad un altro in seguito ad un evento scatenante (a patto che lo stato sia attivo).

Definizione : L'evento scatenante che permette il passaggio di stato è detto **condizione** ed è modellato da una funzione booleana. Se uno stato è attivo e la condizione è verificata (la funzione booleana è vera), allora avviene il passaggio di stato.

Nel diagramma, gli stati vengono rappresentati con dei rettangoli denotati con una stringa, questa è convenzionalmente un numero intero univoco che identifica lo stato (è opportuno specificare il significato di ogni stato con una breve descrizione), le azioni sono dei rettangoli collegati da una linea agli stati, ed una transizione è un arco orientato, annotato con la funzione booleana che ne descrive la condizione.



L'azione si occupa di eseguire un attuazione sul sistema, la condizione verifica un certo valore di riferimento. Se un arco di transizione non è annotato da alcuna condizione, si assume che questa sia sempre vera. Le condizioni si verificano con le canoniche strutture di controllo

- if
- for
- while

e le variabili controllate sono, generalmente, valori che provengono dai sensori.

4.5.2 Regole di Evoluzione

Uno o più stati nell'SFC sono considerati *iniziali* e sono denotati con una doppia bordatura.



All'avvio di un PLC, il sistema logico avrà la seguente configurazione

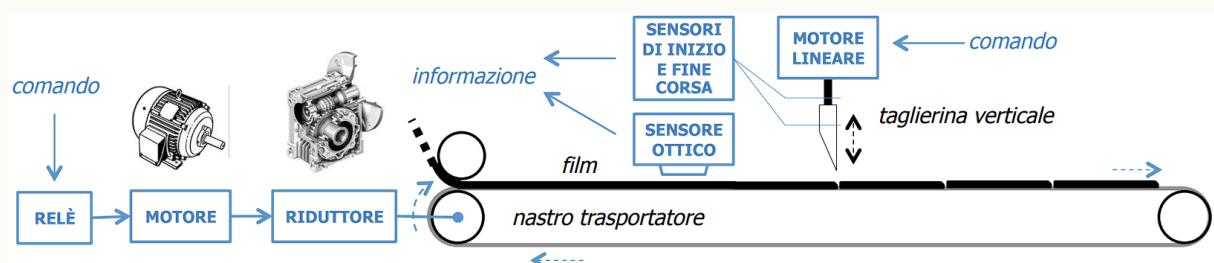
- tutti gli stati iniziali saranno attivi
- tutti gli stati non iniziali saranno inattivi

Una transizione, si verifica se e solo se

- lo stato che ha l'arco uscente è attivo
- la condizione è verificata

Esempio di Modellizzazione

Si consideri il seguente dispositivo di automazione industriale



è composto da un nastro trasportatore su cui è disposto un film (pellicola) omogeneo che deve essere tagliato da una taglierina verticale. Il nastro trasportatore è attuato da un motore/riduttore attivato da un relè. La taglierina è attuata da un motore lineare. Un sensore ottico rileva un segno presente sul film indicante quando è necessario tagliare, mentre due sensori di inizio e fine corsa indicano la posizione della taglierina.

I requisiti di funzionamento del sistema, prevedono le seguenti specifiche

- Quando il sensore ottico rileva il segno sul film, deve essere effettuato il taglio
- Quando la taglierina esegue il taglio, il nastro trasportatore deve essere fermo
- Quando la taglierina viene riportata in condizioni di riposo, il nastro trasportatore può essere riattivato

Si vuole progettare un diagramma SFC che descriva il sistema. Si assume che ad avviamento della macchina, il nastro trasportatore sia attivo, e la taglierina in una situazione di riposo. La taglierina può trovarsi in una delle 3 seguenti possibili condizioni

- riposo
- scende per tagliare
- risale

Si identificano 3 stati per il sistema

1. il nastro trasportatore è acceso, la taglierina è a riposo (stato iniziale)
2. il nastro trasportatore è fermo, la taglierina scende ed effettua il taglio
3. il nastro trasportatore è fermo, la taglierina risale

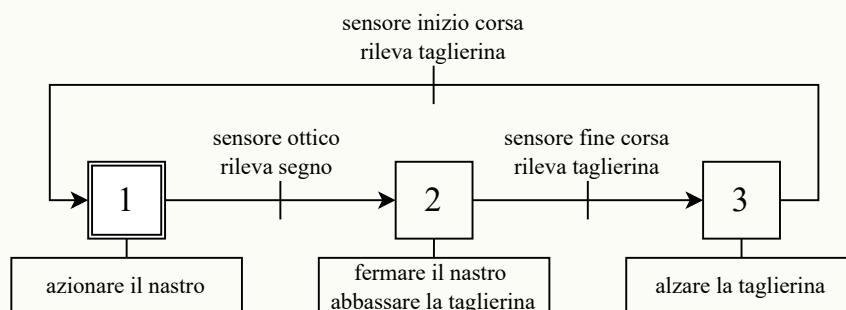
Ai 3 stati sono associate le seguenti azioni

- Stato 1 : Si avvia l'attuazione del motore che muove il nastro
- Stato 2 : Si ferma il nastro, e si abbassa la taglierina per eseguire il taglio
- Stato 3 : Si riporta la taglierina nella situazione di riposo



A tal punto vanno specificate le condizioni di transizione

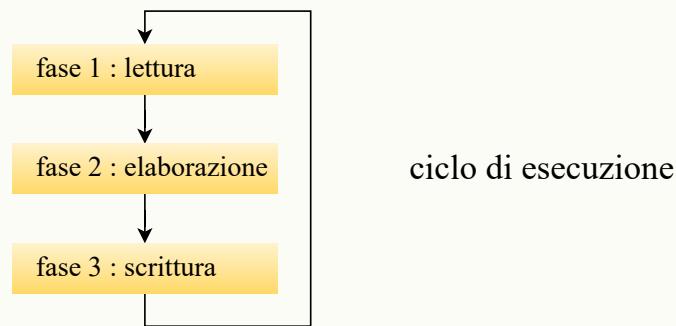
- Il sistema passa dallo stato 1 allo stato 2 quando viene rilevato il segno sul film dal sensore ottico
- Il sistema passa dallo stato 2 allo stato 3 una volta che il sensore di fine corsa rileva la taglierina (il taglio è stato effettuato)
- Il sistema passa dallo stato 3 allo stato 1 quando il sensore di inizio corsa rileva la taglierina (questa è tornata nella posizione di riposo)



4.5.3 Esecuzione Ciclica e Ambiguità

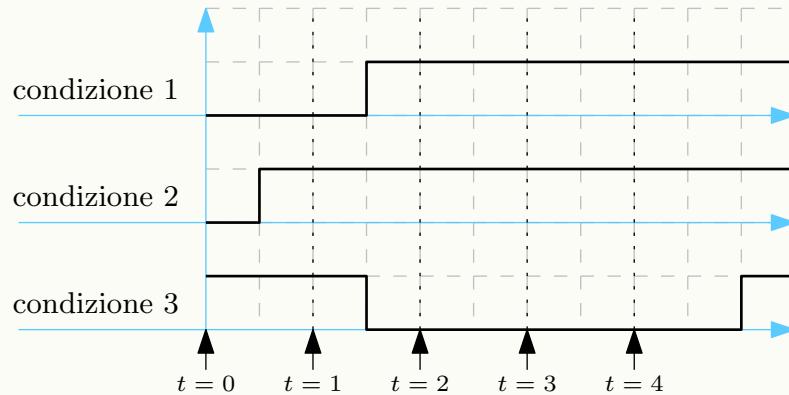
L'SFC descrive un comportamento ciclico del PLC. Il processore, esegue principalmente 3 macrofasi in loop

- **macrofase 1** : Vengono lette le informazioni provenienti dai sensori, vengono opportunamente filtrate e scalate per poter essere elaborate, ed in base a queste e allo stato complessivo del sistema, si verificano le transizioni, eventualmente identificando i nuovi stati.
- **macrofase 2** : Vengono definite le azioni associate a tutti gli stati attivi, procedendo all'attuazione. Questa fase è "costantemente" in esecuzione.
- **macrofase 3** : Vengono considerate le uscite, verificandone la coerenza, eventualmente scalandole per poi venire inviate agli attuatori del mondo fisico.



I PLC sono soggetti ad alcune ambiguità che ne possono compromettere il funzionamento. Essendo sistemi digitali, il controllo delle condizioni non è continuo ma viene effettuato in maniera discreta ad intervalli equidistanti. Si definisce *ciclo di esecuzione* il tempo che intercorre fra l'avvio della fase 1, ed il suo avvio successivo in seguito del termine della fase 3.

Si consideri l'esempio della taglierina, o un qualsiasi sistema di 3 stati topologicamente equivalente. Le condizioni sono variabili booleane continue nel tempo che vengono lette ogni ciclo di esecuzione.



- Supponendo che lo stato 3 sia quello attivo all'istante iniziale $t = 0$, essendo la condizione 3 avverata, avverrà un cambio di stato, rendendo inattivo il terzo, e rendendo attivo il primo.
- all'istante $t = 1$, lo stato attivo è il primo, la condizione 1 non è verificata quindi non avverà alcun cambio di stato.
- all'istante $t = 2$, lo stato attivo è il primo e la condizione 1 è verificata, quindi avverrà un cambio di stato, e lo stato 2 diverrà quello attivo, essendo però anche la condizione 2 attiva, avverrà una transizione istantanea allo stato 3.

l'azione dello stato 2, seppur doveva essere eseguita, è stata ignorata

Tale istante in cui si generano ambiguità come questa è detto **istante critico**, la risoluzione è semplice, si *impone* al PLC un'ulteriore regola

Le azioni associate ad uno stato appena reso attivo devono sempre essere eseguite almeno per un ciclo di funzionamento.

Pertanto le azioni associate ad un qualsiasi stato la cui transizione in uscita è caratterizzata da una condizione già verificata nel momento della sua attivazione, vengono eseguire almeno per un ciclo di funzionamento.

~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~ ~*~

4.6 Sintassi del Linguaggio SFC

Il microprocessore definisce, per ogni stato del diagramma, due variabili

- **marker** è una variabile booleana, se q è uno stato, si indica con $q.X$, ed è uguale a 1 (**True**) se e solo se lo stato è attivo.
- **timer** è una variabile intera positiva, indicata con $q.T$, ed indica il lasso di tempo trascorso dall'attivazione dello stato, memorizzando la durata di attivazione.

Riguardo il timer:

- se lo stato è attivo, il timer si aggiorna incrementando il tempo trascorso
- se lo stato è inattivo, il timer è fermo, lasciando invariato il valore
- se lo stato passa da inattivo ad attivo, il timer si resetta al valore 0

4.6.1 Qualificatori delle Azioni

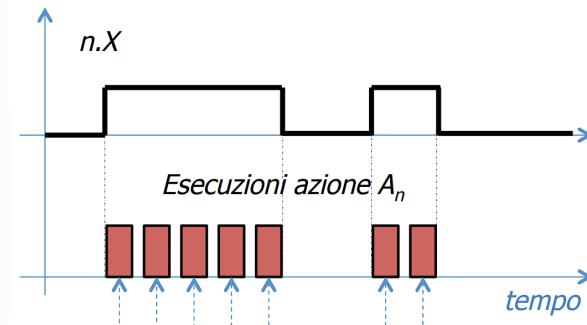
Ad ogni stato m è associata al più un'azione, questa è composta da 3 elementi

- **identificatore** A_m : è una stringa univoca che identifica l'azione
- **qualificatore** Q_m : definisce il metodo di esecuzione delle azioni definite
- **variabile** V_m è una variabile booleana, è 1 se le azioni definite sono state eseguite e portate al termine almeno una volta.

In questa sezione descriveremo il comportamento dei vari qualificatori, e di come questi possono influenzare il flusso di esecuzione.

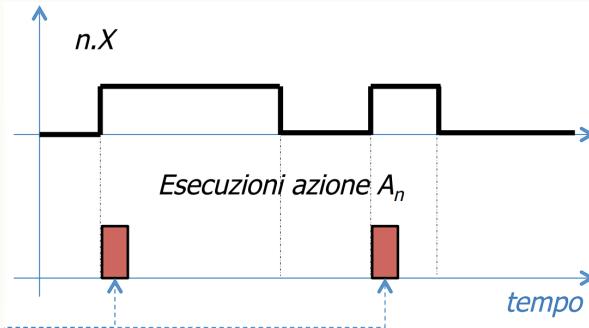
Normal

Il qualificatore N , è quello assegnato di default alle azioni, si assumerà quindi che, se un'azione non specifica alcun qualificatore, il comportamento sarà di questo tipo. Semplicemente, l'azione associata ad uno stato n verrà eseguita ciclicamente finché questo è attivo, ossia $n.X = 1$



Pulse

Dato uno stato n , se il suo qualificatore è di tipo Pulse $Q_n = P$, allora, ogni volta che lo stato passerà da inattivo ad attivo, l'azione associata verrà eseguita una volta.

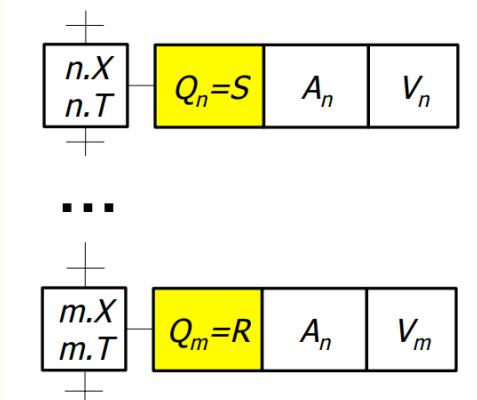


Set e Reset

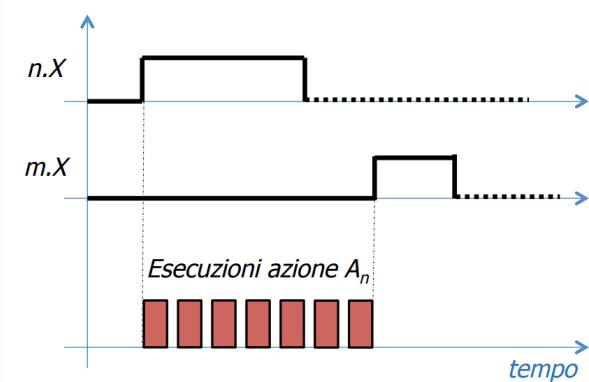
I seguenti qualificatori vengono sempre considerati in coppia, questi sono *Set* e *Reset*, si considerano due stati (distinti) n ed m , alla quale sono associati questi qualificatori

$$Q_n = S \quad Q_m = R$$

è importante che, i due differenti stati che condividono la coppia di qualificatori, abbiano lo stesso identico identificatore A_n



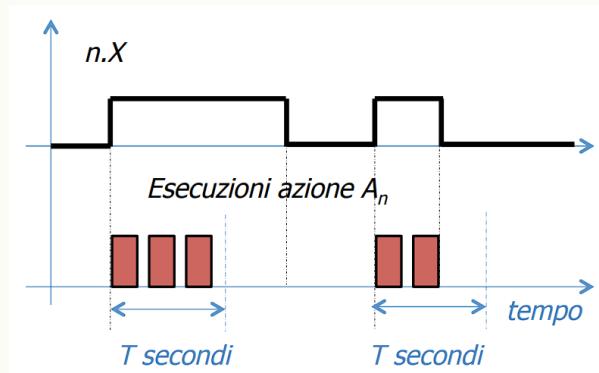
Quando uno stato con qualificatore S diviene attivo, allora la sua azione verrà eseguita ciclicamente, e continuerà ad essere eseguita anche quando lo stato diverrà inattivo, solo quando diverrà attivo uno stato (con lo stesso identificatore di azione) che ha qualificatore R , allora l'azione terminerà di essere eseguita.



Se diviene attivo uno stato con qualificatore R , ma l'azione in questione non era già in esecuzione, non succede nulla.

Time Limited

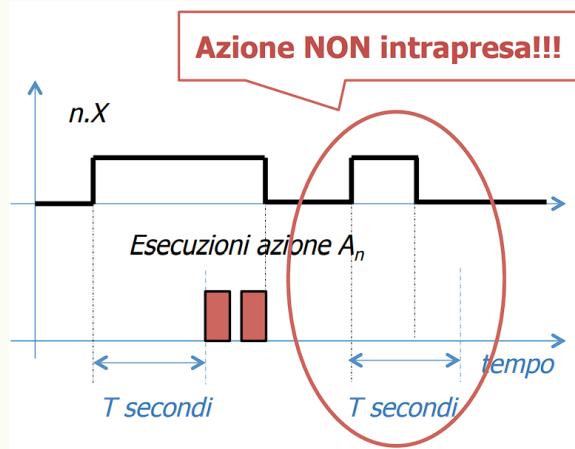
Il qualificatore L ha sempre associato un lasso di tempo T (che può essere misurato in secondi, minuti, o una qualsivoglia unità di tempo). Quando uno stato con qualificatore L (con intervallo T) diventa attivo, allora l'azione ad essa associata verrà eseguita per T unità di tempo.



Se lo stato diventa inattivo prima che i T secondi siano passati, l'azione si interrompe comunque.

Time Delayed

Il qualificatore D è l'opposto del qualificatore L . Anche questo ha associato un tempo T , quando uno stato diviene attivo, l'azione ad essa associata verrà eseguita solamente dopo che sono trascorse T unità di tempo, una volta avviata l'esecuzione, questa sarà ciclica finché lo stato è attivo.



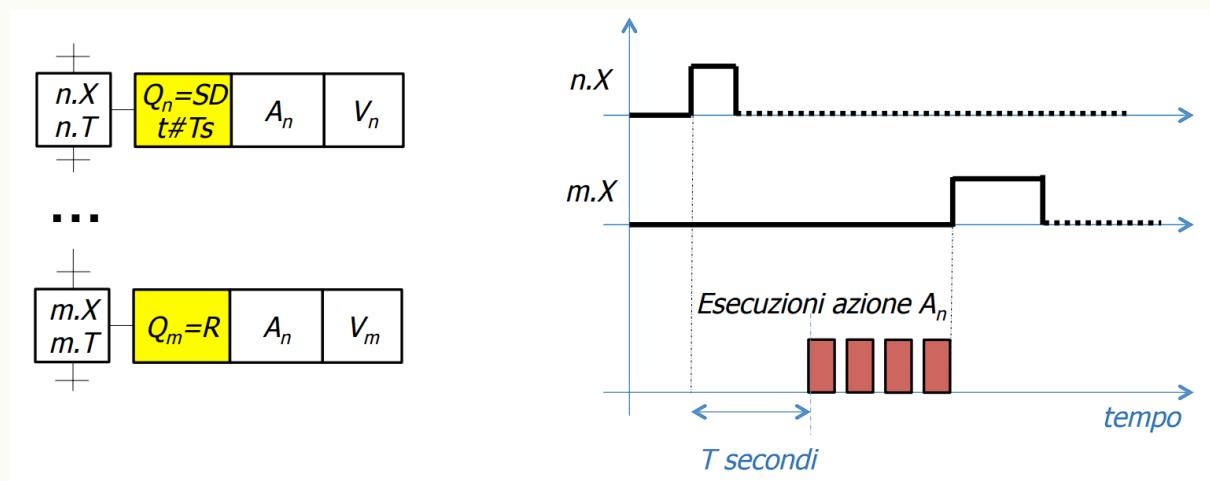
Se l'intervallo T è maggiore del tempo totale di attività dello stato, l'azione non verrà eseguita.

È possibile considerare dei qualificatori che sono *combinazioni* di qualificatori già esistenti, seguono alcuni esempi.

Qualificatore Combinato SD

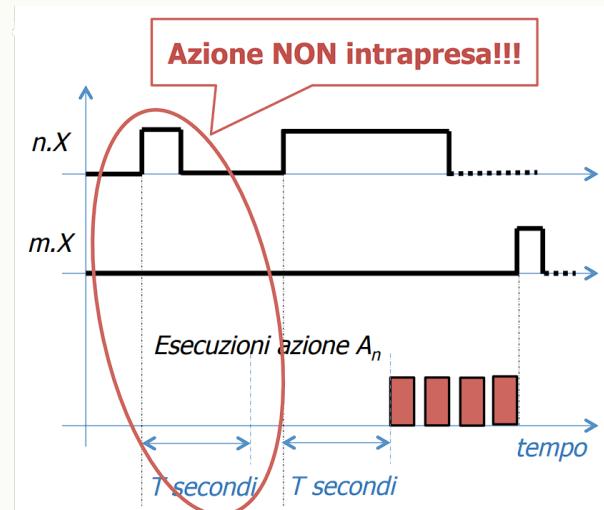
Il qualificatore SD combina l'azione Set/Reset all'azione time delayed, ha quindi associato un intervallo di tempo T e coinvolge sempre due stati, uno con qualificatore SD ed uno con qualificatore R (ed identico identificatore).

Quando uno stato SD diviene attivo, dopo T unità di tempo verrà avviata l'esecuzione ciclica dell'azione associata, che si fermerà solamente quando diverrà attivo lo stato con qualificatore R .



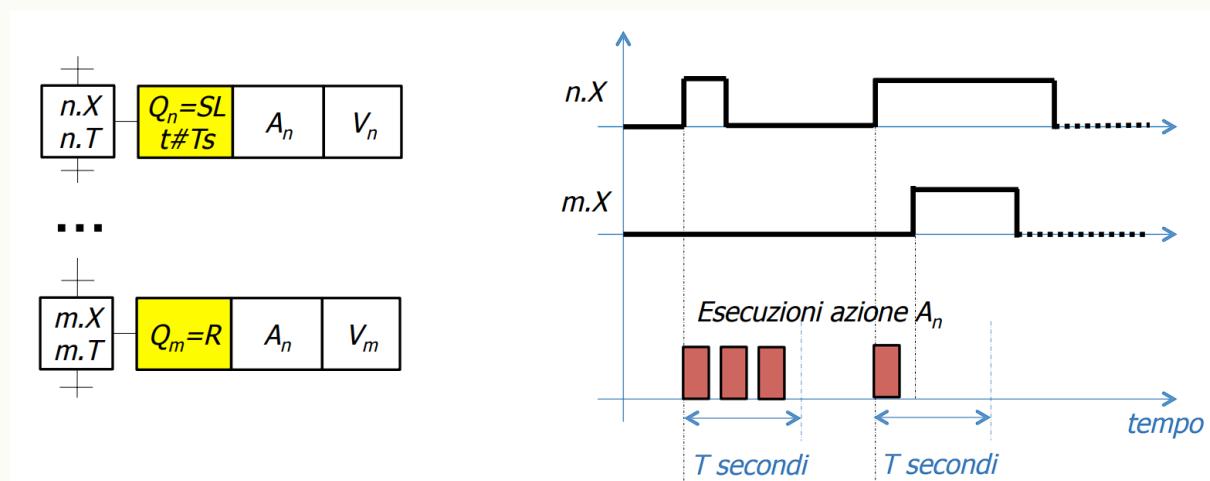
Qualificatore Combinato *DS*

È come il qualificatore *SD* ma con sequenza delle operazioni invertita, quando lo stato con tale qualificatore diviene attivo, attende T unità di tempo, se al termine di queste lo stato è ancora attivo, viene iniziata l'esecuzione ciclica, altrimenti, non verrà eseguita alcuna azione.



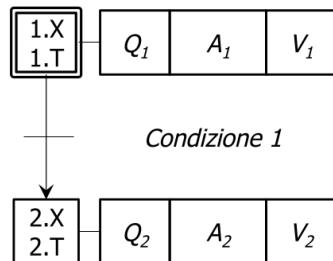
Qualificatore Combinato *SL*

Come ci si può aspettare, il comportamento è quello del Set/Reset, solo che l'esecuzione ciclica si fermerà dopo T unità di tempo associate al qualificatore.



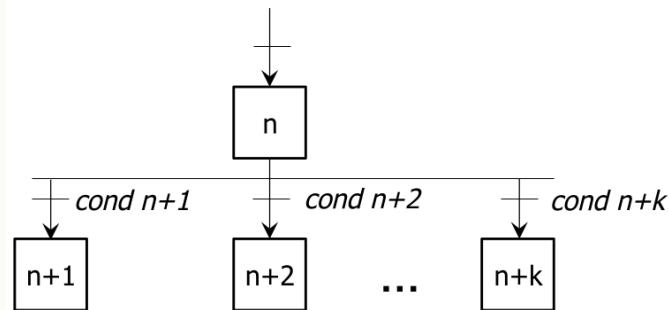
4.6.2 Strutture di Collegamento

Per rendere più ampio il campo in cui può operare SFC si considerano diverse strutture di collegamento che vanno oltre alla semplice transizione con condizione, seppur queste siano basilari, una corretta modellazione dei collegamenti fra gli stati è ciò che richiede più impegno da parte del progettista. Abbiamo già visto la struttura di base, che permette la transizione quando lo stato coinvolto è attivo, e la condizione indicata è verificata.



Scelta o Divergenza

La seguente struttura di collegamento modella la scelta di una via piuttosto che di un'altra, in pratica riproduce i blocchi `if/else`.

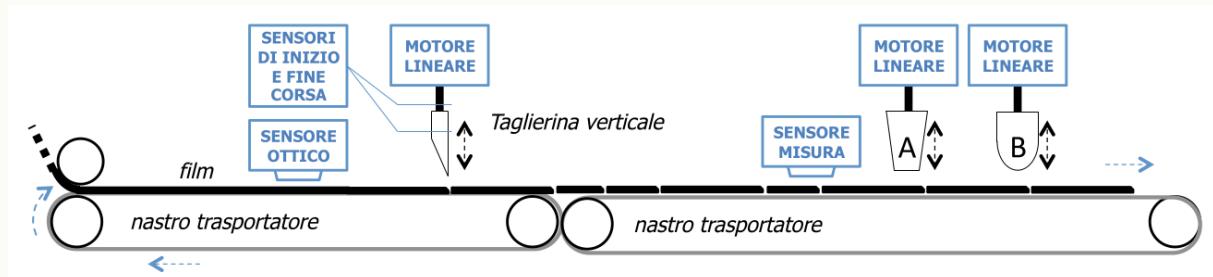


Quando è attivo lo stato n , solo uno dei differenti flussi verrà percorso, è quindi fondamentale che le diverse condizioni sui rami siano mutualmente esclusive. Per garantire ciò, si può permettere che più di una condizione sia vera contemporaneamente, stabilendo però una gerarchia che stabilisce quale delle vie dovrà essere intrapresa, per ogni condizione $cond\ n + i$, su una diramazione di k possibili scelte, viene valutata piuttosto che questa la condizione

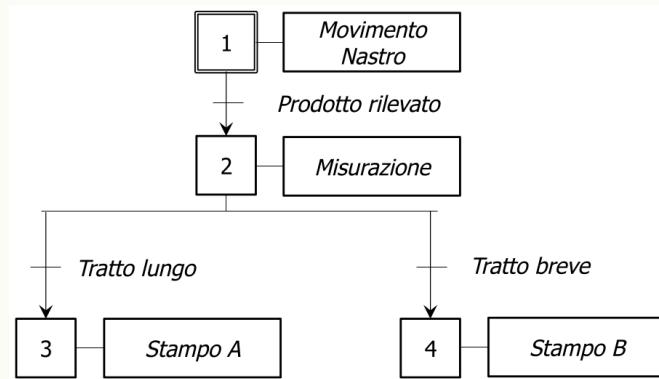
$$cond'\ n + i = cond\ n + i \wedge \left(\bigwedge_{j=1}^{k-i} \overline{cond\ n + i + j} \right)$$

In tal modo, una condizione $n + i$ avrà minore priorità di $n + i - 1$, dato che quest'ultima deve essere per forza falsa per garantire la verità della prima.

Si consideri l'esempio della taglierina visto nella sezione 4.5.2, si aggiunge un nuovo nastro che in base alla lunghezza del pezzo tagliato, deve eseguire uno dei due stampi differenti A o B .



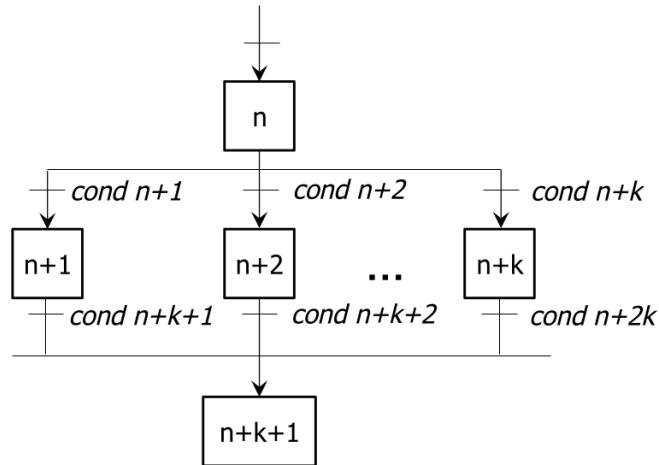
L'SFC dovrà modellare la mutua esclusività dei due stampi, si dovrà introdurre un collegamento di divergenza che verifichi le condizioni (lunghezza del pezzo).



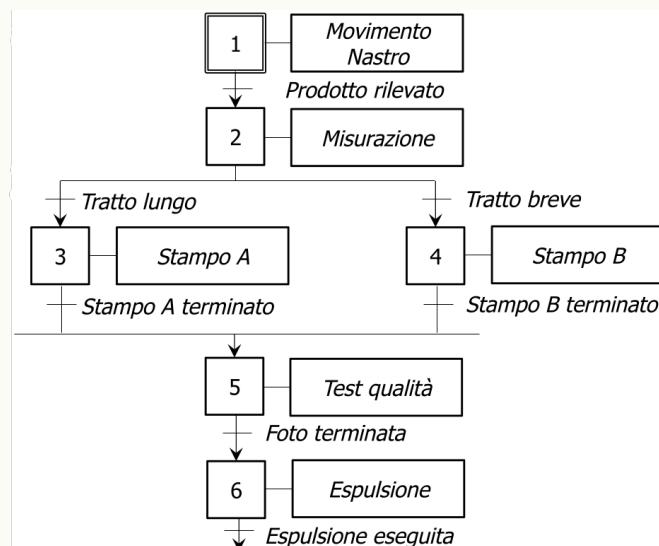
Attenzione: La sintassi del collegamento di divergenza richiede sempre che

1. a monte della divergenza ci sia sempre uno stato
2. per ogni possibile scelta, c'è una condizione
3. in seguito ad ogni scelta, c'è uno stato

Come si può intuire, è necessaria una struttura di **convergenza** che congiunge ad uno stato differenti flussi prima creatisi da una divergenza/scelta.

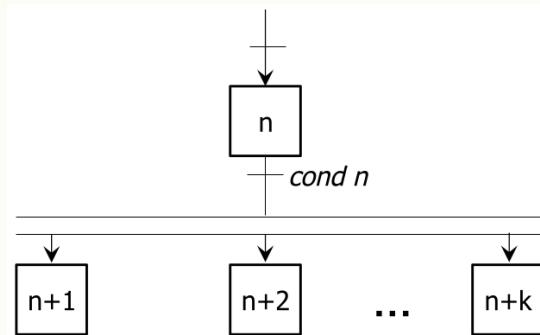


Considerando l'esempio della taglierina, supponiamo che a valle del secondo nastro ci sia un dispositivo che deve fotografare il pezzo con la stampa per valutarne la qualità, a prescindere dal fatto che questo sia stato stampato con lo stampo A o B. I due flussi dovranno ricongiungersi, come rappresentato:

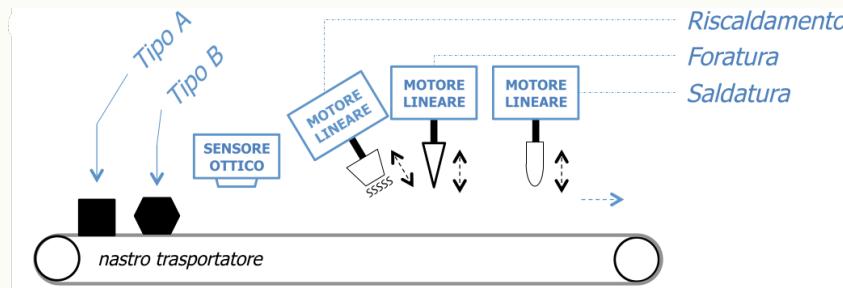


Parallelismo

È necessaria una struttura di collegamento che descriva l'esecuzione parallela di più flusso di lavoro. Nella sintassi della struttura di parallelismo, vi è un singolo stato a monte, con una condizione, se verificata, verranno attivati tutti gli stati a valle compresi nel collegamento. Il fork si rappresenta con una doppia linea.

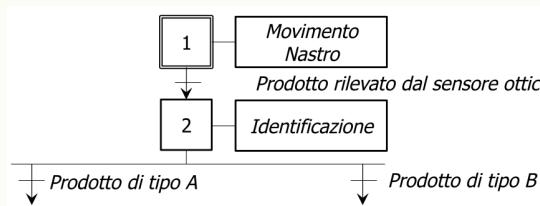


Si consideri il seguente *Esempio*, vi è un sistema di lavorazione per due elementi metallici di tipo *A* e di tipo *B*, posti su un nastro trasportatore, un sensore ottico rileva la tipologia del pezzo, e ci sono poi 3 azionamenti dedicati al riscaldamento, foratura e saldatura.

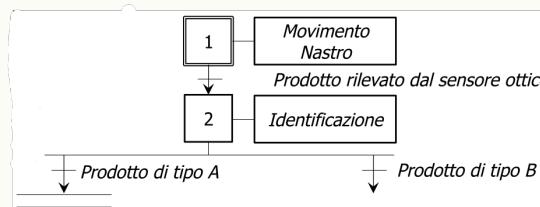


- il pezzo di tipo *A* dovrà essere (in parallelo) riscaldato e forato.
- il pezzo di tipo *B* dovrà essere (in parallelo) riscaldato e (queste due sequenzialmente) forato e saldato.

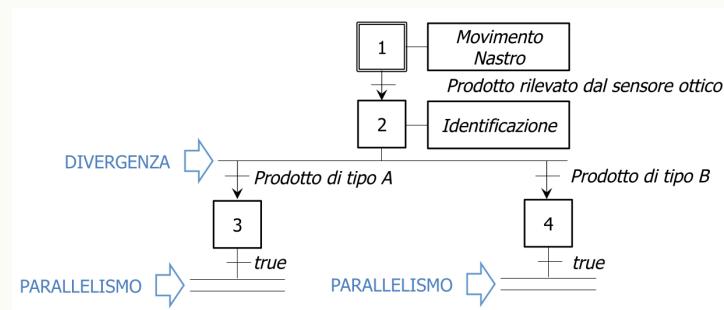
A monte ci dovrà essere una divergenza che tramite il sensore ottico rilevala tipologia del pezzo.



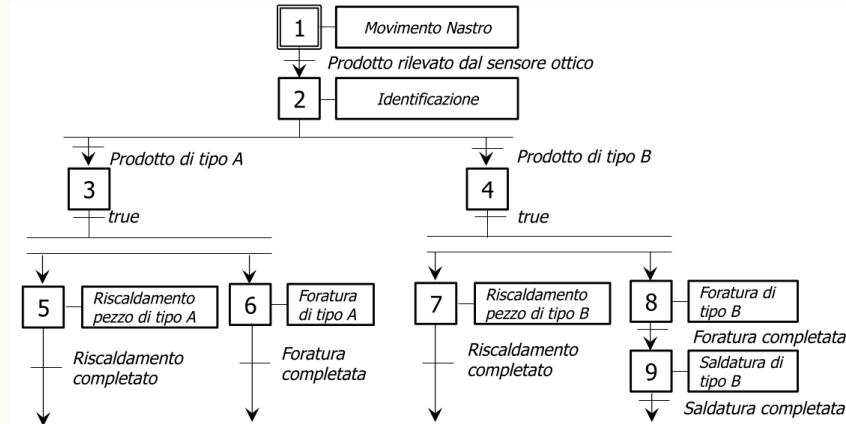
Se il pezzo è di tipo *A*, dovrà essere contemporaneamente riscaldato e forato, verrà quindi considerata una struttura di parallelizzazione.



Questa struttura però viola la sintassi, in quanto è necessario che a valle di un fork ci sia uno stato, si inserisce quindi uno stato "fantoccio" privo di azioni, che serve solo a mantenere la coerenza sintattica, che non necessita di condizioni per poter avviare il fork. Si considera anche per il flusso di *B*, dato che anche tale pezzo dovrà ricevere delle lavorazioni in parallelo.

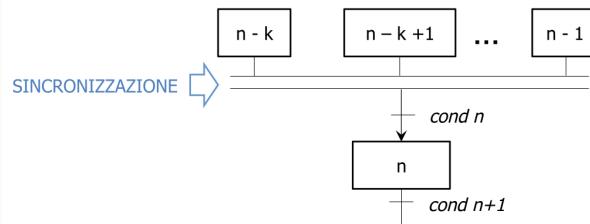


Si ricordi che in una transizione, la condizione **true** può essere omessa. Una volta eseguito il fork per entrambe le tipologie di pezzi, sarà possibile eseguire le lavorazioni in questione.



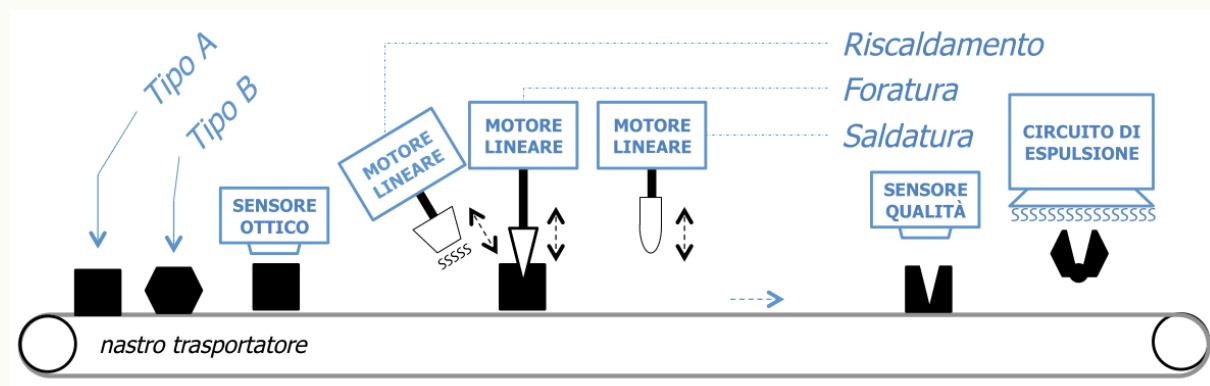
Sincronizzazione

A seguito di una parallelizzazione dei flussi, può essere necessario un join di questi, per poter ritornare ad eseguire un unico flusso, la sintassi prevede k stati a monte, ed un unico stato a valle, la transizione presenta un'unica condizione.

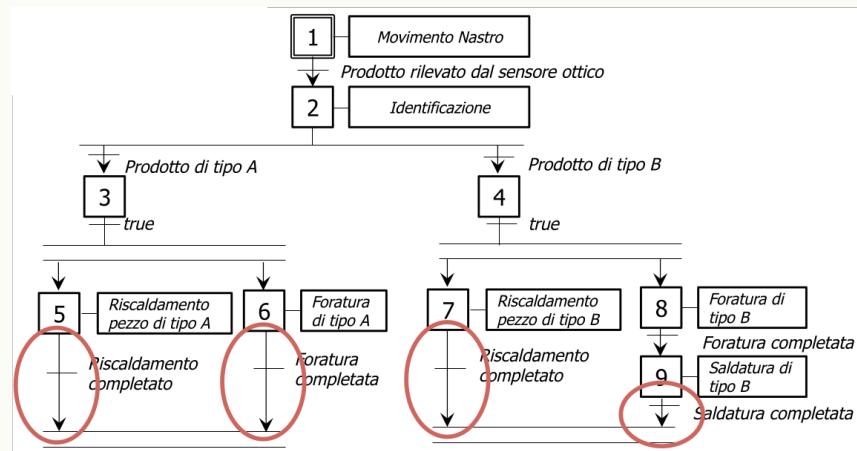


Quando ogni stato a valle è attivo, e la condizione del collegamento è verificata, allora avverrà la transizione.

Si consideri nuovamente l'esempio dei pezzi metallici, supponiamo che alla fine del nastro ci sia un sensore di qualità che deve valutare il pezzo, qualsivoglia sia il tipo, per poi mandarlo al circuito di espulsione.

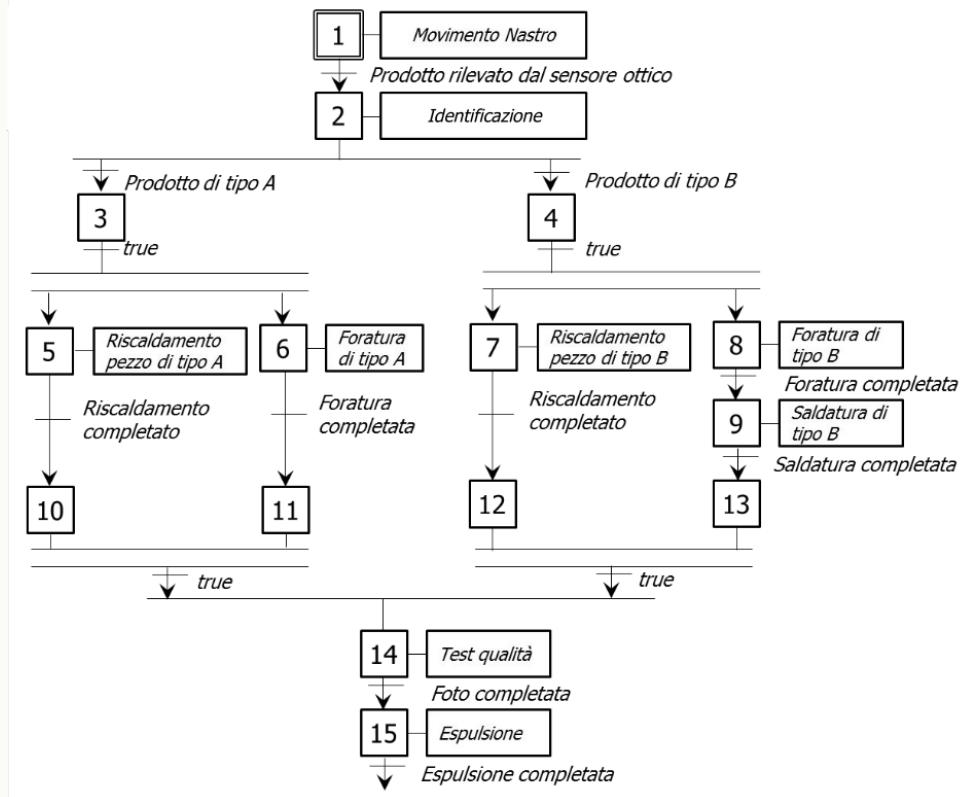


Va eseguita la sincronizzazione dei flussi paralleli (per la tipologia A e la tipologia B)

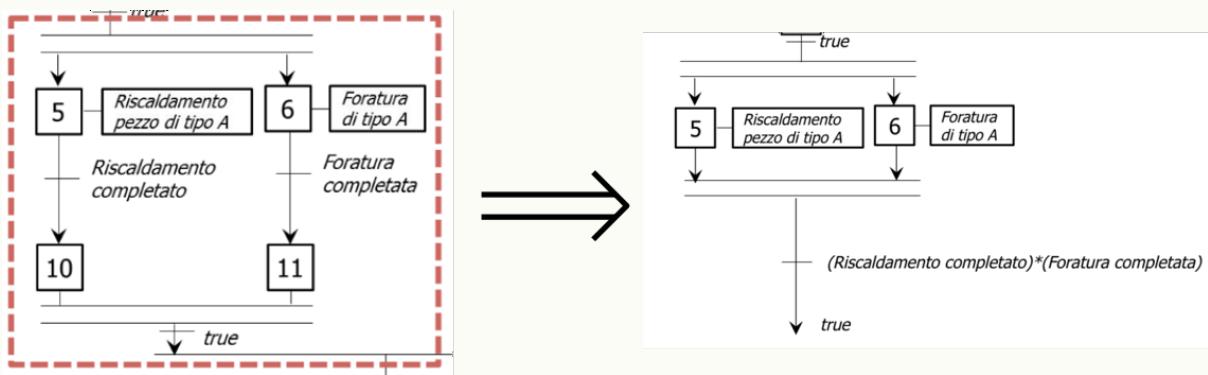


Attenzione, il diagramma sopra riportato è irrealizzabile, la sintassi richiede che a monte di ogni join ci siano degli stati, si considerano quindi degli stati fantoccio (senza azioni ne condizioni) detti *stati di attesa*.

In seguito alla sincronizzazione dei flussi sia per A che per B, sarà necessaria una convergenza, in quanto il sensore di qualità dovrà valutare il pezzo indipendentemente dal suo tipo. Il diagramma SFC completo è il seguente:



Si consideri il sistema degli stati 5,6,10 e 11, si noti come lo stato 5 può essere collassato con lo stato 10, e lo stato 6 può essere collassato con lo stato 11, senza dover aggiungere due stati fantoccio, e mettendo le due condizioni *Riscaldamento completato* e *Foratura completata* in **and** nel blocco di sincronizzazione.



Questa situazione **non è ammissibile** per un semplice motivo: Si introduce una relazione di dipendenza fra i due stati

1. parallelamente lo stato 5 e lo stato 6 verranno eseguiti
2. se lo stato 5 termina, prima di poter ricongiungersi nella struttura di join, deve attendere la terminazione dello stato 6
3. in tal modo, l'azione dello stato 5 rimarrà attiva senza poter essere interrotta
4. l'aggiunta degli stati fantoccio permette l'indipendenza fra le azioni degli stati 5 e 6

Mutua Esclusione

Vogliamo modellare l'accesso esclusivo ad una risorsa, supponiamo che due stati distinti su due flussi paralleli differenti, vogliano eseguire un'azione che richiede lo stesso attuatore (ad esempio, un manipolatore robotico). È chiaro che l'esecuzione delle azioni in parallelo porta ad un comportamento anomalo, e potenzialmente pericoloso dell'attuatore, è quindi necessario che gli stati in questione non siano mai attivi contemporaneamente.

Si consideri il seguente diagramma:



I due flussi sono paralleli. Le azioni degli stati n ed $n + 1$ richiedono lo stesso attuatore delle azioni degli stati m ed $m + 1$, quindi, quando è attivo uno degli stati n , $n + 1$ (o analogamente uno fra m , $m + 1$) bisogna accertarsi, prima di eseguire la transizione, che la risorsa sia libera, altrimenti si violerebbe la mutua esclusione.

Si definisce uno *stato ausiliario S* detto **semaforo**, questo, tramite il suo marker $S.X$, descriverà la disponibilità della risorsa, memorizzandone lo stato d'uso.

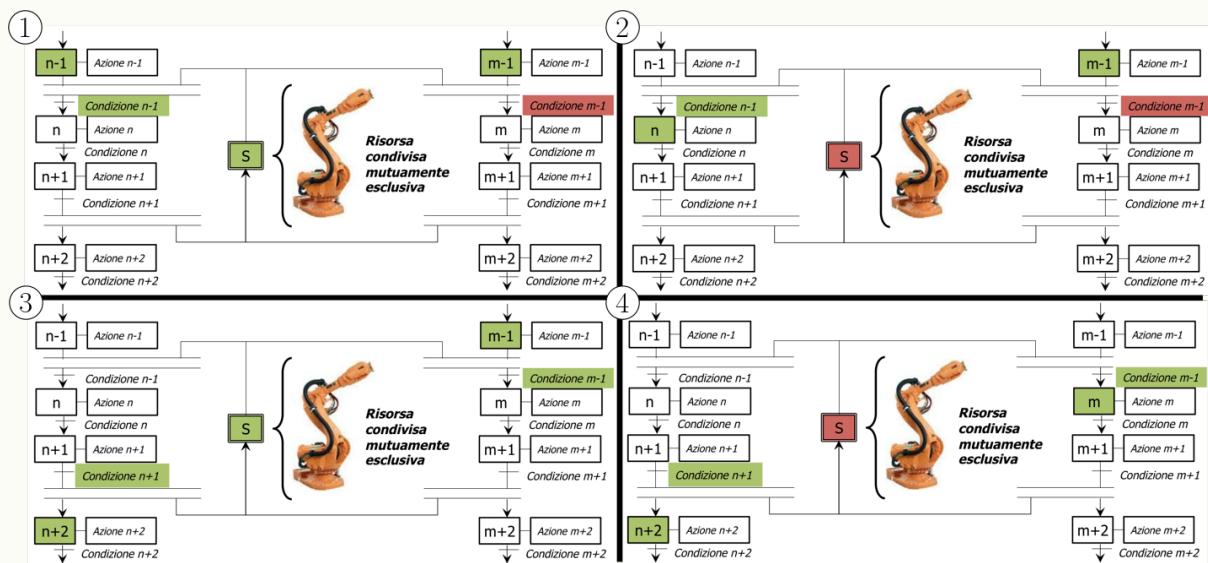
- $S.X = 1$, lo stato è attivo, la risorsa è disponibile
- $S.X = 0$, lo stato è inattivo, la risorsa non è disponibile, non può avvenire la transizione

Le variabili semaforo sono stati iniziali in quanto, si assume che ad inizio processo la risorsa sia disponibile. La struttura di collegamento che permette la mutua esclusione è la seguente:



Il funzionamento verrà descritto per il flusso n , ma è analogo per il flusso m :

- Supponiamo sia attivo lo stato $n - 1$
- Data la struttura di sincronizzazione, per passare allo stato n è necessario che, sia n che S (l'altro stato a monte della sincronizzazione) siano attivi. Quindi, lo stato n può essere attivato solo se $S.X = 1$, ossia, la risorsa è libera.
- quando avviene la transizione da $n - 1$ ad n , lo stato S viene reso inattivo, quindi l'eventuale transizione da $m - 1$ ad m non potrà avvenire.
- Quando avviene la transizione da $n + 1$ ad $n + 2$ (si termina l'uso della risorsa esclusiva), viene eseguito un fork, e lo stato S viene reso attivo, quindi $S.X = 1 \implies$ la risorsa sarà di nuovo disponibile, e la transizione da $m - 1$ ad m potrà accadere.



Attenzione, è importante capire che in questa struttura può verificarsi una condizione di ambiguità, in particolare, se gli stati $n - 1$ ed $m - 1$ dovessero essere attivi, e le condizioni $n - 1$ ed $m - 1$ dovessero diventare vere nello stesso momento, le transizioni avverrebbero su entrambi i flussi, e diverrebbero attivi sia n che m , accedendo alla stessa risorsa condivisa violando la mutua esclusività.

È quindi importante che le condizioni $n - 1$ ed $m - 1$ siano rese mutualmente esclusive, tramite una relazione gerarchica, come nel caso delle condizioni di un collegamento di divergenza/scelta, in questo

caso si può dare priorità alla transizione da $n - 1$ ad n imponendo alla condizione $m - 1$ di verificare che la condizione $n - 1$ sia falsa:

$$\text{cond } m - 1 = \text{cond } m - 1 \wedge \overline{\text{cond } n - 1}$$

equivalentemente si può verificare che lo stato $n - 1$ non sia attivo

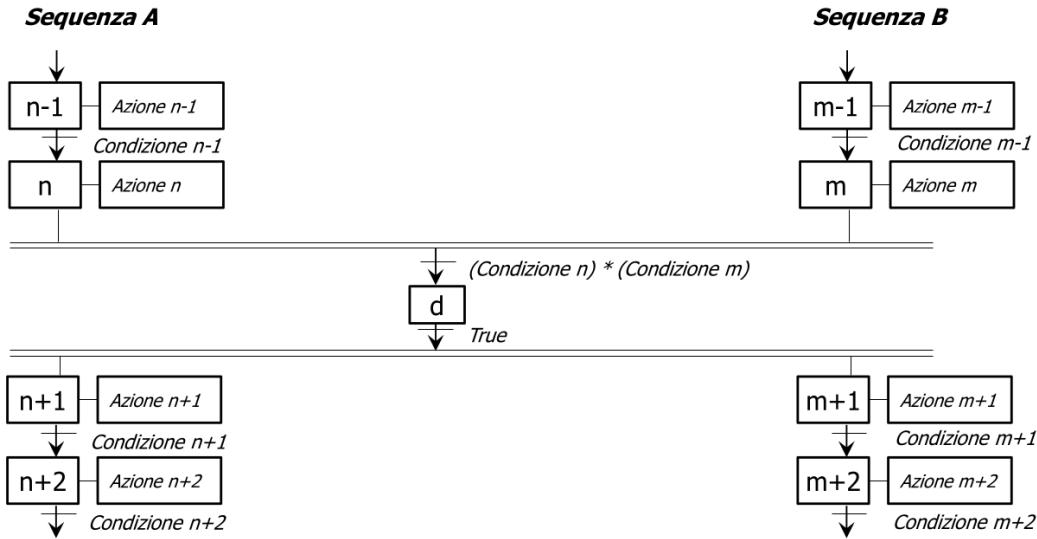
$$\text{cond } m - 1 = \text{cond } m - 1 \wedge (n - 1).X == 0$$

Sincronizzare due Sequenze

Molte volte, può essere necessario mettere in relazione due flussi di esecuzione paralleli, costringendo uno stato, a dover attendere che un'azione (di uno stato di un flusso differente) termini, prima di poter eseguire la transizione. Si consideri il seguente schema



Si vuole far sì che, la sequenza B possa procedere oltre lo stato m se e solo se, nella sequenza A (che avviene in parallelo) le azioni dello stato n siano state completate. Un'idea di soluzione potrebbe essere la seguente:



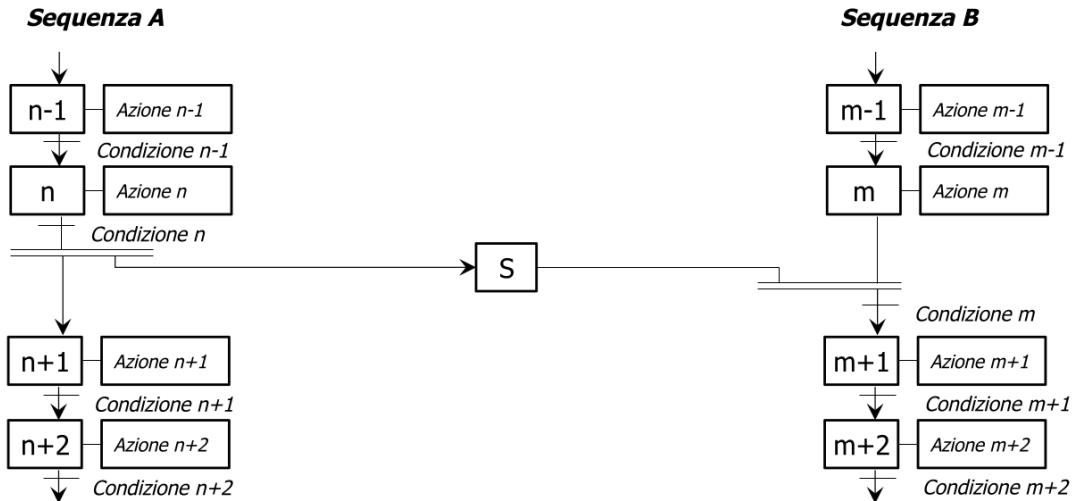
Questa soluzione è *errata*, seppur riesca a modellare la dipendenza di B da A , introduce anche la dipendenza di A da B , infatti non si potrà passare dallo stato n allo stato $n + 1$ finché la sequenza B non conclude le azioni dello stato m . Vogliamo che A sia indipendente.

Un'altra idea può essere quella di modificare la condizione della transizione dallo stato m allo stato $m + 1$, introducendo la condizione n

$$\text{cond } m = \text{cond } m \wedge \text{cond } n$$

ma anche questa soluzione è errata, in quanto la condizione n , può essere vera indipendentemente dal fatto che le azioni dello stato n siano state terminate o meno.

La corretta modellazione avviene introducendo un semaforo tramite uno stato ausiliario S , per cui deve essere vero che $S.X = 1$ per permettere alla sequenza di B di andare oltre lo stato m , e si da alla sequenza A la possibilità di impostare l'attività di S tramite un fork fra l'azione n e l'azione $n + 1$.



Si noti come

- la sequenza A è indipendente e può essere eseguita ciclicamente un numero indefinito di volte, semplicemente, al passaggio fra lo stato n ed $n + 1$ viene reso attivo lo stato S .
- dallo stato m , per proseguire è necessario che S sia attivo
- la transizione da m ad $m + 1$ rende inattivo S , sarà necessario che nel flusso A avvenga nuovamente la transizione $n \rightarrow n + 1$.

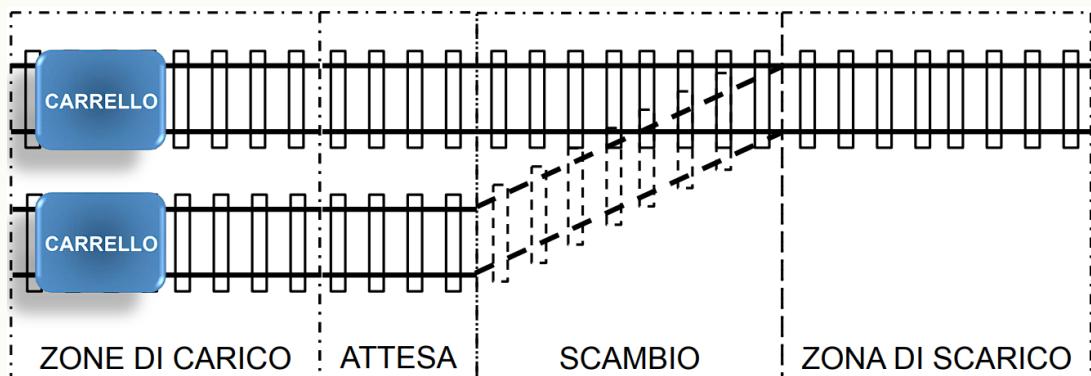
❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖

4.7 Esercitazione sul Linguaggio SFC

In questa sezione vediamo alcuni esercizi che prevedono la modellazione del comportamento logico di un sistema di automazione tramite il linguaggio SFC.

4.7.1 Esercizio 1 - Carrelli semoventi

Descrizione del problema : Si vuole modellare un sistema di carico/scarico di materiali tramite dei carrelli che si muovono su dei binari, si osservi la seguente figura:



Ci sono due carrelli, che per semplicità denotineremo A e B , entrambi si trovano inizialmente nella zona di carico, il processo viene avviato quando un operatore preme start su un telecomando. Sono presenti sui binari i seguenti sensori

- *poscarA*: sensore booleano, True quando il carrello A è nella zona di carico.
- *poscarB*: sensore booleano, True quando il carrello B è nella zona di carico.
- *posattA*: sensore booleano, True quando il carrello A è nella zona di attesa.
- *posattB*: sensore booleano, True quando il carrello B è nella zona di attesa.
- *posscar*: sensore booleano, True quando uno dei due carrelli è nella zona di scarico.

Inoltre i carrelli, presentano dedicati attuatori per la movimentazione sui binari, questi possono muoversi davanti e indietro, gli attuatori sono

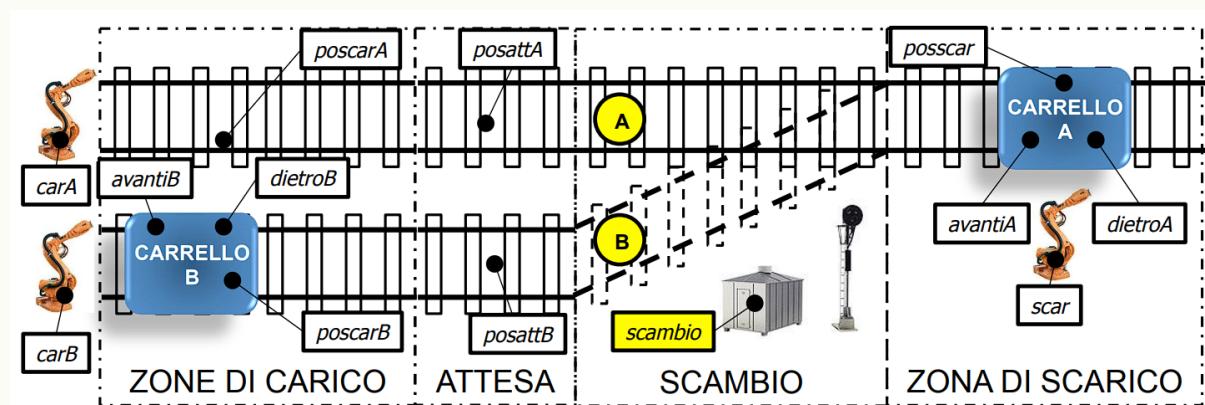
- *avantiA*: se True, il carrello A si muove in avanti (verso la zona di scarico)
- *avantiB*: se True, il carrello B si muove in avanti (verso la zona di scarico)
- *dietroA*: se True, il carrello A si muove indietro (verso la zona di carico)
- *dietroB*: se True, il carrello B si muove indietro (verso la zona di carico)

Sono poi presenti tre robot manipolatori, in particolare, due robot nella zona di carico, per caricare i due carrelli (tempo necessario per caricarli : 10 secondi), ed un robot nella zona di scarico, che deve scaricare il materiale, ed impiega 2 secondi.

- *carA*: se True, carica i materiali sul carrello A
- *carB*: se True, carica i materiali sul carrello B
- *scar*: se True, scarica i materiali dal carrello nella zona di scarico

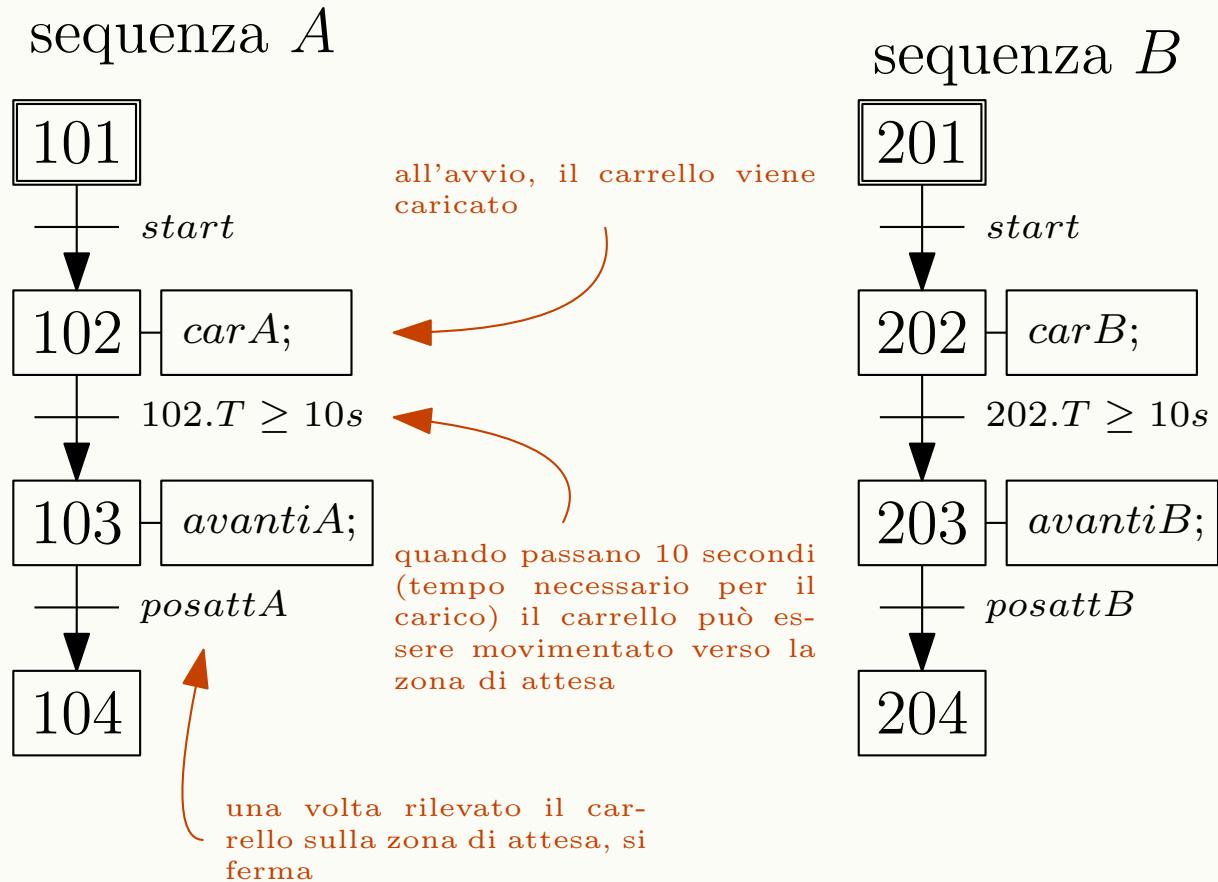
è infine presente un'attuatore che permette di attivare la zona di scambio dei binari per uno dei due carrelli:

- *scambio*: se True, potrà transitare sulla zona di scambio il carrello A, se False, potrà transitare sulla zona di scambio il carrello B.

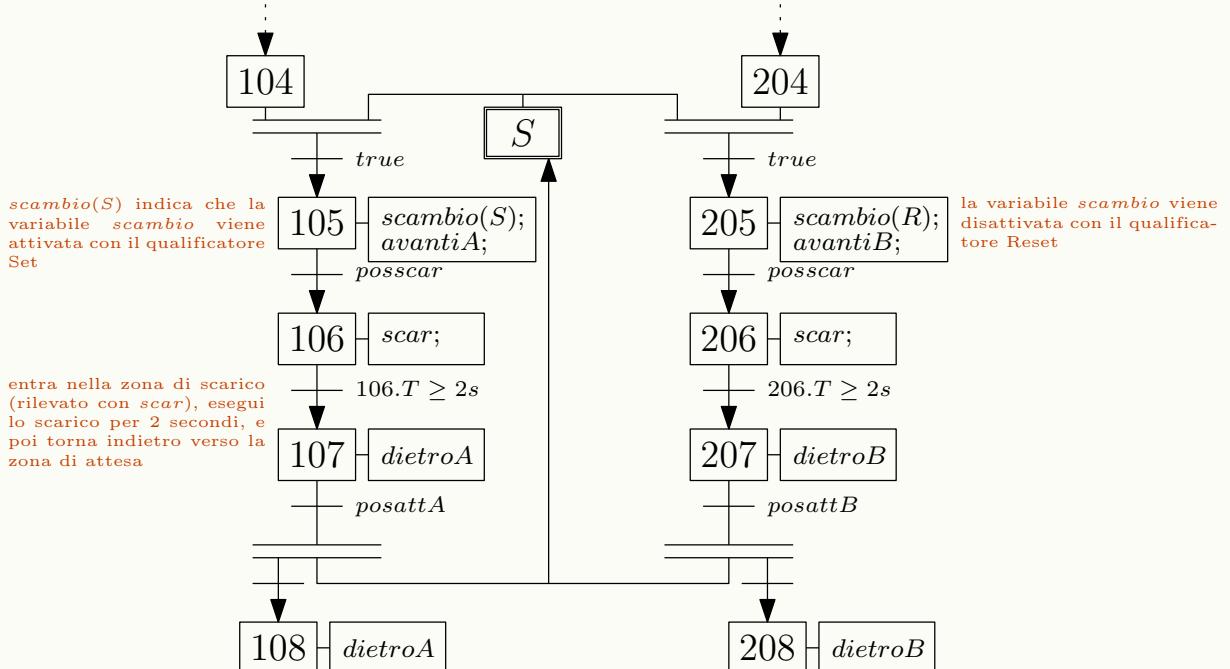


I due carrelli non possono transitare contemporaneamente nella zona di scambio, l'accesso a questa deve essere mutualmente esclusivo.

Risoluzione : Definiremo indipendentemente due flussi del diagramma, inerenti al carrello A e al carrello B, questi due, partono entrambi dalla zona di carico, ed il loro comportamento è simmetrico, quando viene premuto il tasto start, un carrello nella zona di carico, dovrà caricare il materiale con il braccio manipolatore, per poi dirigersi alla zona di attesa.

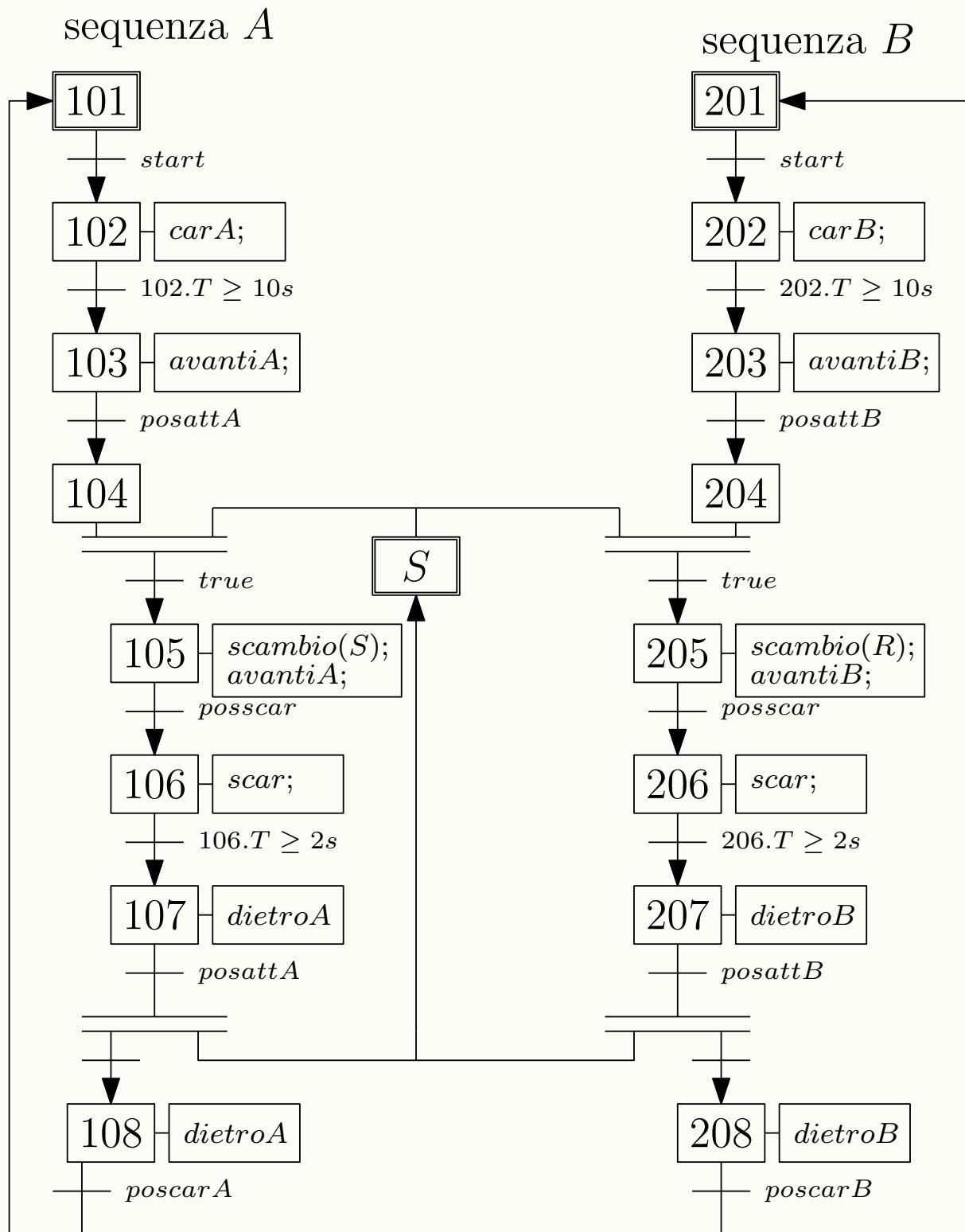


L'accesso allo scarico va gestito con una mutua esclusione, si utilizzerà la struttura vista nelle sezioni precedenti, usando lo stato ausiliario S . Quando il carrello A deve passare, va attivato l'attuatore $scambio$, quando deve passare B , va disattivato.



Una volta usciti dalla zona di scarico, i carrelli potranno tornare nella zona di carico, in attesa che un operatore prema di nuovo start.

Il diagramma SFC completo è il seguente.



4.7.2 Esercizio 2 - Cisterna di miscelazione

Descrizione del problema : Si consideri una cisterna di miscelazione in cui vanno inseriti 3 liquidi diversi, uno di tipo alcanino, e altri due che non incidono sull'alaninità o sull'acidità della soluzione. Sono presenti 3 elettrovalvole (attuatori) che attivano il flusso in ingresso degli attuatori.

CAPITOLO

5

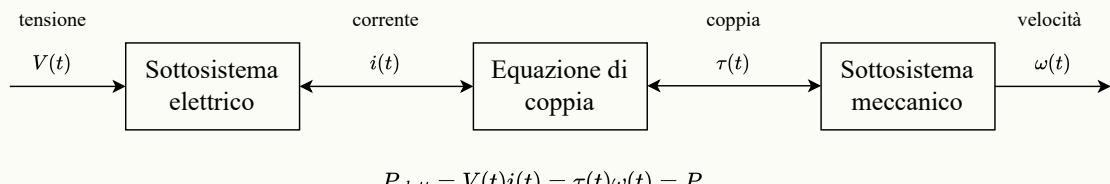
ATTUAZIONE E CONTROLLO DEL MOTO

In questo capitolo verranno affrontati gli azionatori elettromeccanici, definiamo **trasduttore** un dispositivo che converte energia da una forma ad un'altra (ad esempio, da elettrica a meccanica), i *sensori* eseguono delle misurazioni convertendo grandezze fisiche (come la temperatura) in segnali elettrici, gli *attuatori* trasformano segnali di comando in moto meccanico.

Vedremo in particolare i **trasduttori elettromeccanici**

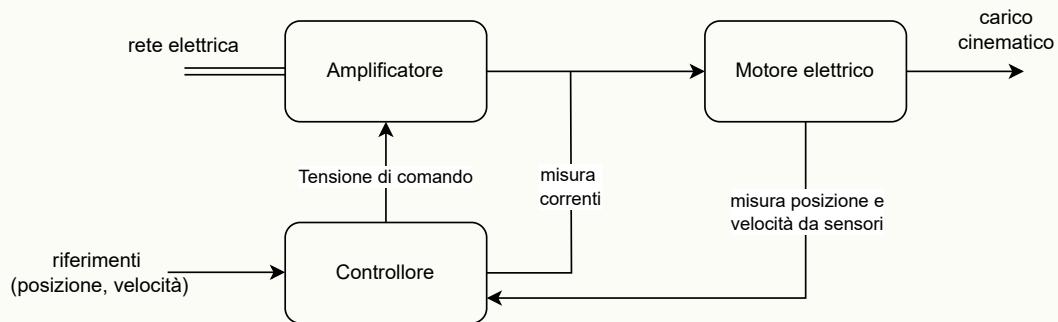
- *generatore* : trasforma energia meccanica in energia elettrica
- *motore* : trasforma energia elettrica in energia meccanica

Nella trasformazione di energia da un dominio all'altro, varrà che la *potenza rimarrà costante*, quest'ultima può essere elettrica o meccanica, l'equazione di coppia descrive tale identità.



Lo scopo degli *azionamenti elettrici* è la conversione controllata di energia elettrica in energia meccanica, si vuole *imporre* del moto ad un carico meccanico. Tali azionamenti sono composti da 3 elementi fondamentali

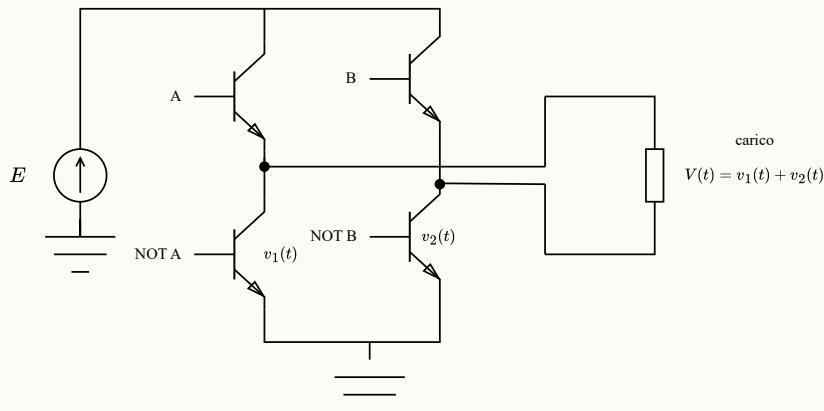
- **Amplificatori** : Essendo che il segnale di controllo non è necessariamente forte per azionare un carico, è necessaria un amplificazione tramite semplici amplificatori analogici lineari. Nel caso il motore necessiti di grande potenza, sarà necessario un circuito di tipo switching con apposite tecniche di modulazione (PWM)
- **motore elettrico** : Ce ne sono di diverse tipologie, alimentato in corrente continua (DC) o alternata (AC), brushless, passo-passo (stepper), asincrono (a induzione), sincrono ed altre.
- **controllore** : L'unità che si occupa della regolazione tramite leggi a feedback e generazione di riferimenti/comandi.



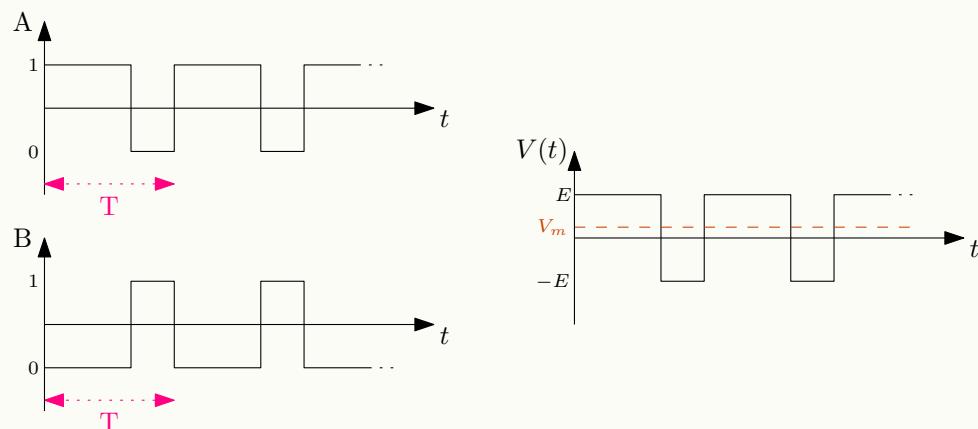
Il motore dovrà spostare un carico cinematico, il cui peso si opporrà al moto, costituendo una *coppia in opposizione* che nel modello matematico potrà essere trattata come un *disturbo*. I motori sono provvisti di un encoder per la misurazione della posizione angolare. Il controllore è tipicamente digitale e si occupa della regolazione e dell'asservimento di riferimenti, costanti o variabili.

5.0.1 Amplificazione PWM

Si vuole dedicare questa breve sezione alla **Pulse Width Modulation (PWM)**, ossia una tecnica per l'amplificazione e conversione di potenza con componenti elettronici di tipo switching. È un sistema elettronico di potenza che, grazie all'energia della rete elettrica permette di attuare la tensione richiesta dall'unità di controllo.

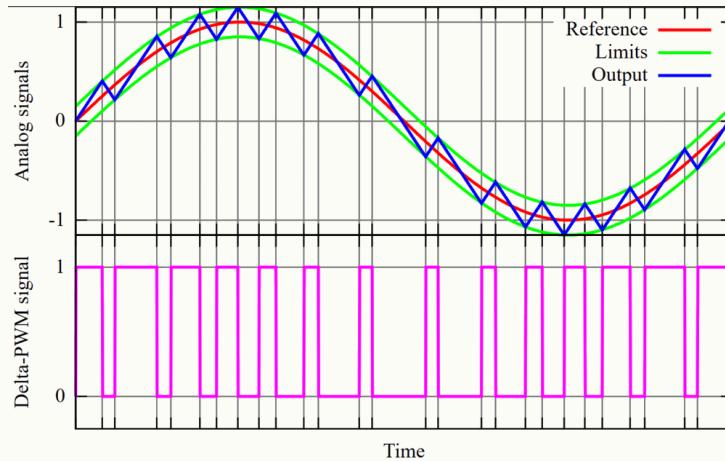


Il rendimento degli amplificatori di tipo switching è superiore a quello degli amplificatori analogici. Nell'immagine sovrastante, il motore elettrico (carico) è connesso alla corrente E e controllato da quattro switch pilotati da due segnali A e B, che sono onde quadre di periodo T fra loro in controfase.



La tensione media V_m è pari a $k \cdot E$ dove k è un certo valore che dipende dal rapporto tra il periodo in cui il segnale A è uguale ad 1 e T , tale rapporto è detto **duty cycle**. Ricapitolando : Facendo variare

il segnale A è possibile far assumere alla tensione il valore desiderato. Ci sono varie implementazioni di schemi PWM, ad esempio, il **metodo delta**, dato un *offset* (ossia un valore limite di differenza dal segnale di riferimento), commuta l'uscita PWM fra ON ed OFF ogni volta che si raggiunge un limite. Il segnale riprodotto si ottiene integrando tale uscita.

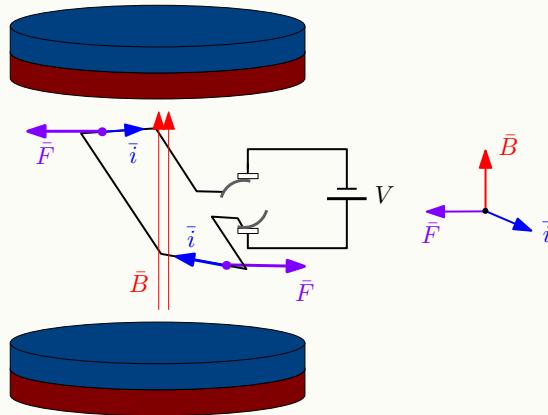


5.1 Il Motore a Corrente Continua

Quando si parla di motore elettrico, esistono due schemi realizzativi

- *con spazzole alimentato da corrente continua*
- *brushless pilotato con elettronica switching*

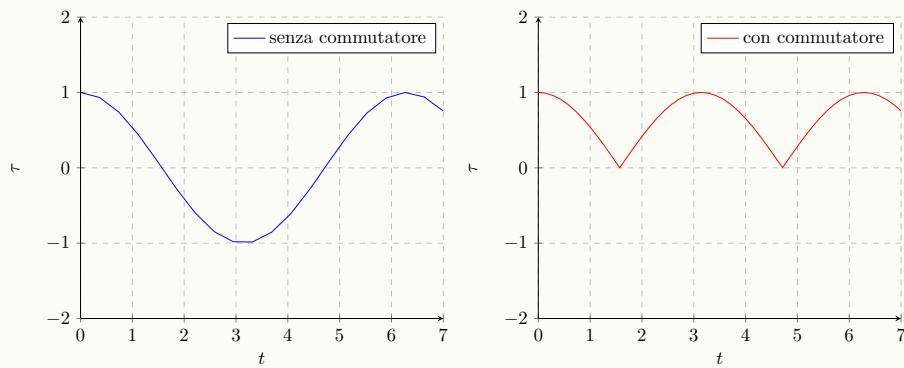
Il motore a corrente continua ha una costruzione complessa, ma il modello matematico che lo descrive è semplice. Il rotore (parte che rotea) comprende degli avvolgimenti (spire) di rame su cui circola la corrente in una determinata direzione, l'alimentazione avviene grazie al contatto delle spazzole che struscano sul motore. Il tutto è immerso in un campo magnetico fornito da magneti permanenti.



Gli elettroni si muoveranno in direzione della corrente \bar{i} , essendo presente un campo magnetico \bar{B} , saranno soggetti alla **forza di Lorentz**

$$\bar{F} = (\bar{i} \times \bar{B})$$

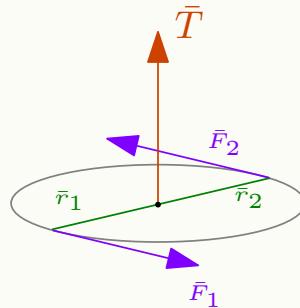
In questo modo verrà generata una coppia facendo ruotare il circuito. Le spazzole, colorate in giallo nella figura, con il contatto fanno sì che la corrente circoli, si noti come l'anello collettore è diviso in due parti, funge da *commutatore*, tale costruzione permette alla corrente di alternare il verso ad ogni mezzo giro, se così non fosse, la forza di Lorentz si alternerebbe facendo oscillare il rotore fino a fermarsi senza farlo roteare.

Figura 5.1: andamento della coppia τ

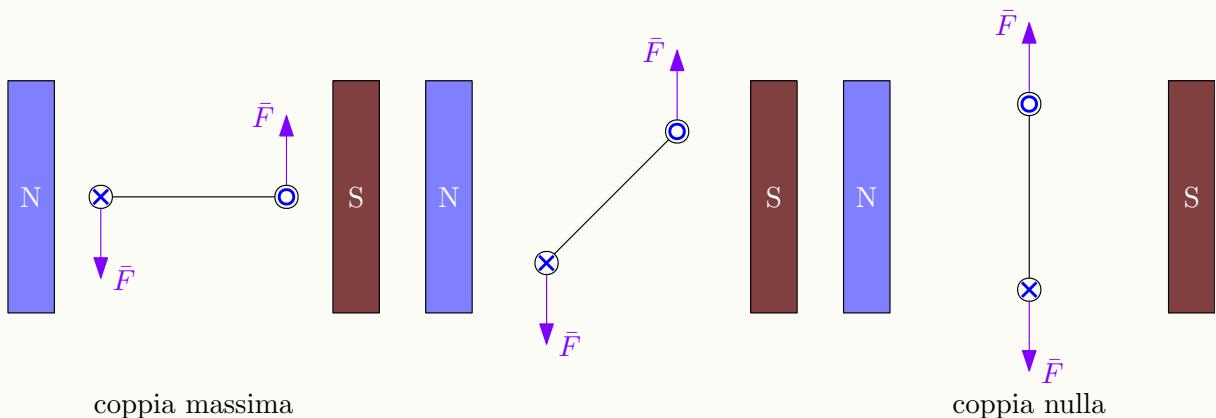
Il **momento torcente (coppia)** vale

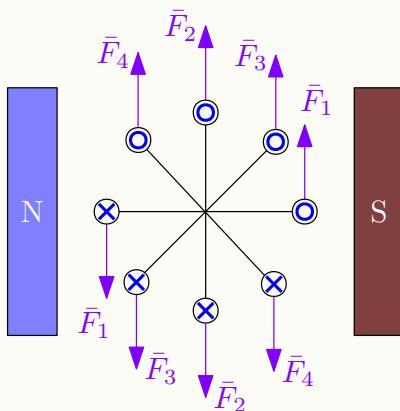
$$\bar{T} = (\bar{r}_1 \times \bar{F}_1 + \bar{r}_2 \times \bar{F}_2)$$

Verrà utilizzato il simbolo $\tau = \pm |\bar{T}|$ per denotare la *coppia scalare*.



La forza \bar{F} dipende dall'angolo θ di rotazione del motore, in quanto lo spostamento della direzione della corrente \bar{i} può far attenuare la forza. La coppia si azzera ogni volta che l'angolo fra la forza generata ed il braccio di rotazione è di $k\pi$ con $k \in \mathbb{N}$.





Per via di questo, la coppia ha un andamento oscillatorio e non è costante, tale fenomeno è denominato **ripple**, per limitare ciò (ed aumentare la coppia) si possono aggiungere più spire in modo tale che, nell'istante in cui una spira produce coppia nulla, un'altra spira produce coppia massima.

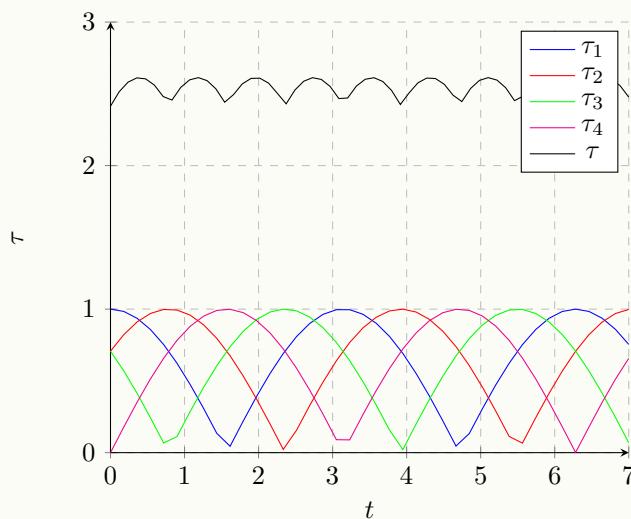
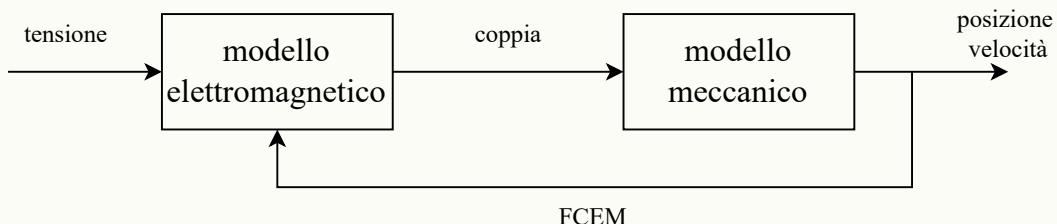


Figura 5.2: Coppia generata da più spire

Più è il numero di spire incluse, più la coppia si "spiana" tendendo ad un andamento costante.

5.1.1 Modello del Motore DC

Il modello del motore DC ha una componente elettromagnetica (la cui dinamica è dettata dalle leggi dell'elettronica sulle tensioni) ed una componente meccanica (la cui dinamica è dettata dall'equazione di bilancio delle forze).



Il motore è sempre soggetto ad una *tensione contro elettro motrice (FCEM)*, è una tensione contraria a quella applicata (si oppone all'ingresso) e dipende dalla velocità di rotazione del motore. Questa FCEM è un "feedback" intrinseco nella fisica del sistema.

Il modello del motore sarà lineare, gli azionamenti elettrici sono ovviamente sottoposti a dei limiti fisici

- tensione massima
- corrente massima

Che per costruzione implicano dei limiti per il modello meccanico

- velocità massima del motore
- coppia di picco erogabile massima

Il modello matematico del motore DC mostrato in figura 5.3 è caratterizzato da 5 parametri

- R_a resistenza di armatura
- L_a induttanza di armatura
- k_m coefficiente di coppia/FCEM
- j inerzia del rotore
- b coefficiente di attrito viscoso

e da 6 variabili

- i_a corrente di armatura
- ω velocità angolare del rotore
- θ posizione angolare del rotore
- V_a tensione di ingresso
- τ coppia prodotta dal motore
- τ_r coppia di carico (disturbo)

La *coppia di carico* τ_r è una forza che si oppone alla coppia data e dipende dal carico (massa) che l'attuatore cerca di muovere. Nonostante si possa modellizzare, verrà trattata in questo contesto come un disturbo.

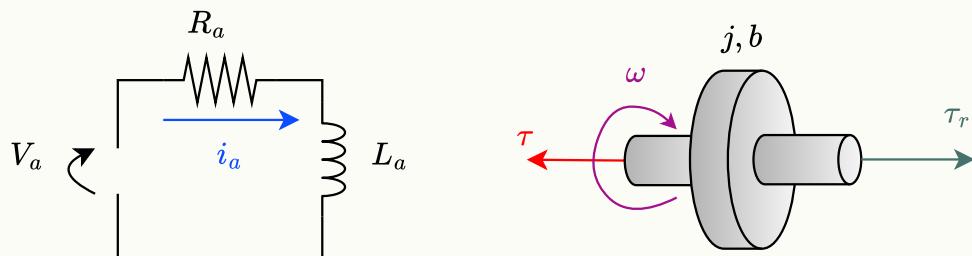


Figura 5.3: modello del motore a corrente continua

Il coefficiente k_m mette in relazione i due modelli. Infatti, descrive la relazione fra la corrente i_a e la coppia τ . La stessa identica costante descrive anche la relazione fra la corrente V_a e la velocità di rotazione ω (modello della FCEM). Ciò è vero dato che la potenza dei due modelli deve essere identica. Sia V_{cem} la FCEM:

$$\begin{aligned} P_{elec} &= V_{cem} \cdot i_a = \tau \omega = P_{mecc} \\ \tau &= k_m i_a \quad V_{cem} = k_m \omega \end{aligned}$$

Il modello elettrico è descritto dalla legge di Kirchoff per le maglie chiuse

$$L_a \frac{di_a}{dt} = V_a - R_a i_a - k_m \omega$$

Il modello meccanico

$$j \frac{d\omega}{dt} = k_m i_a - b\omega - \tau_r$$

La posizione angolare si ricava integrando la velocità: $\frac{d\theta}{dt} = \omega$. Mettendo insieme le equazioni, il motore elettrico è descritto dal seguente sistema dinamico lineare tempo invariante

$$\begin{cases} L_a \frac{di_a}{dt} = V_a - R_a i_a - k_m \omega \\ j \frac{d\omega}{dt} = k_m i_a - b\omega - \tau_r \\ \frac{d\theta}{dt} = \omega \end{cases}$$

L'ingresso del sistema sarà la tensione

$$u = V_a$$

L'ordine del sistema è 3, il vettore di stato sarà

$$\bar{x} = \begin{bmatrix} \frac{di_a}{dt} \\ \frac{d\omega}{dt} \\ \frac{d\theta}{dt} \end{bmatrix}$$

Supponiamo che la tensione di ingresso V_a sia costante, essendo il sistema *asintoticamente stabile*, a regime si avrà una corrente costante ed una velocità angolare costante. Essendo la tensione costante, la derivata della corrente è nulla.

$$\frac{di_a}{dt} = 0 \implies V_a - R_a i_a - k_m \omega = 0$$

Quindi

$$(V_a - R_a i_a) \frac{1}{k_m} = \omega$$

Ma allora essendo ω costante, la sua derivata è nulla

$$\frac{d\omega}{dt} = 0 \implies k_m i_a - b\omega = 0$$

Nell'ipotesi che la tensione in ingresso sia costante il sistema si riduce ad un semplice sistema algebrico.

$$\begin{cases} V_a - R_a i_a - k_m \omega = 0 \\ k_m i_a - b\omega = 0 \end{cases}$$

Si possono quindi ricavare i valori di velocità angolare, coppia di disturbo e corrente di armatura.

- $\omega = \left(b + \frac{k_m^2}{R_a} \right)^{-1} \cdot \left(\frac{k_m}{R_a} V_a - \tau_r \right)$
- $\tau_r = \frac{k_m}{R_a} V_a - \left(b + \frac{k_m^2}{R_a} \right) \omega$
- $i_a = \frac{1}{R_a} \left(b + \frac{k_m^2}{R_a} \right)^{-1} \cdot (bV_a + k_m \tau_r)$

oppure

- $i_a = \frac{1}{R_a} (V_a - k_m \omega)$

Caratteristiche Statiche

Definiamo **caratteristiche statiche** del motore, i parametri che influiscono sul comportamento a regime. L'aumentare della massa del carico, fa aumentare la coppia di disturbo τ_r , facendo rallentare la rotazione ω . Quando $\omega = 0$, il carico è *massimo* e si definiscono la coppia e la corrente *di stallo*

$$\tau_s = \tau_r \Big|_{\omega=0} = \frac{k_m}{R_a} V_a$$

$$i_s = i_a \Big|_{\omega=0} = \frac{V_a}{R_a}$$

Quando non è presente nessun carico $\tau_r = 0$, si definiscono la velocità e la corrente *a carico nullo*

$$\omega_n = \omega \Big|_{\tau_r=0} = \left(b + \frac{k_m^2}{R_a} \right)^{-1} \tau_s$$

$$i_n = i_a \Big|_{\tau_r=0} = \frac{b}{k_m} \omega_n$$

A regime, la *potenza in uscita* è data da

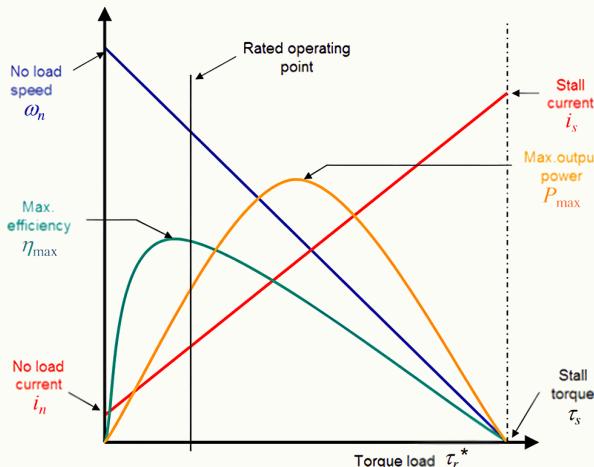
$$P = \left(b + \frac{k_m^2}{R_a} \right)^2 \cdot (\tau_s \tau_r - \tau_r^2)$$

La *massima efficienza* si ha quando la derivata della potenza è nulla

$$\frac{dP}{d\tau_r} = 0 \implies P_{max} = \frac{\tau_s \omega_n}{4}$$

L'*efficienza* è il rapporto fra la potenza in uscita e quella in entrata

$$\eta = \frac{\tau_r \omega}{V_a i_a}$$



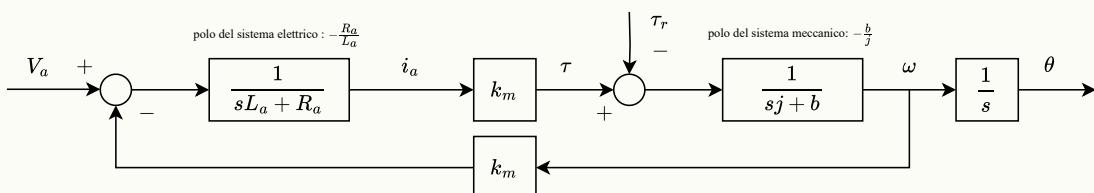
Trasformando le equazioni nel dominio di Laplace (s variabile complessa)

$$L_a \frac{di_a}{dt} = V_a - R_a i_a - k_m \omega \implies \frac{1}{sL_a + R_a}$$

$$j \frac{d\omega}{dt} = k_m i_a - b\omega - \tau_r \implies \frac{1}{sj + b}$$

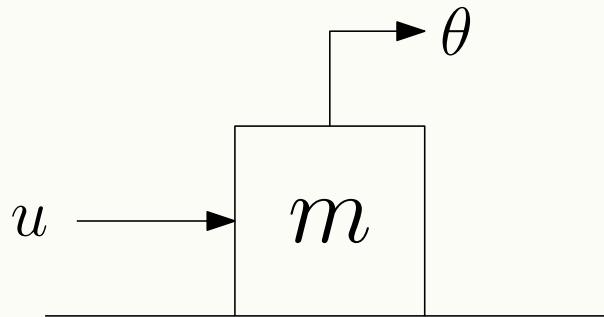
$$\frac{d\theta}{dt} = \omega \implies \frac{1}{s}$$

Lo schema a blocchi del sistema è il seguente



5.1.2 FeedForward

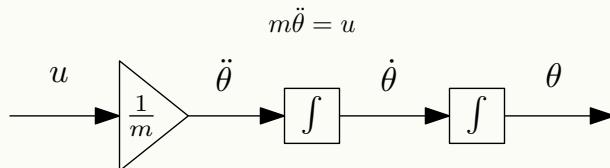
Si consideri il seguente sistema meccanico



Si vuole applicare una forza (ingresso) u da applicare per far sì che la posizione θ del blocco segua la dinamica

$$\theta(t) = A \sin(\omega t)$$

Il sistema (in assenza di attrito) è descritto dall'equazione



Se θ_d è la traiettoria desiderata, allora l'ingresso necessario sarà $u_d = m\ddot{\theta}_d$. Integrando

$$\dot{\theta}_d = A\omega \cos(\omega t) \quad (5.1)$$

$$\ddot{\theta}_d = -A\omega^2 \sin(\omega t) \quad (5.2)$$

Invertendo il sistema si è risaliti all'ingresso necessario. Va considerato però, che gli integratori hanno uno *stato interno*, definiti $\theta_d(0)$ e $\dot{\theta}_d(0)$ deve essere che la posizione e la velocità iniziale combacino con quelle desiderate per ottenere il comportamento desiderato. Se $t = 0$

$$\theta_d(0) = \theta(0) = 0$$

Se $t = 0$, allora

$$\dot{\theta}_d(0) = A\omega \cos(0) = A\omega$$

deve essere che

$$\dot{\theta}(0) = A\omega$$

Il sistema deve quindi essere già in movimento per far sì che si possa ottenere il comportamento desiderato.

Consideriamo adesso una traiettoria θ_d desiderata che abbia velocità e posizione iniziale uguali a zero

$$\theta_d(t) = A(1 - \cos(\omega t))$$

$$\dot{\theta}_d(t) = A\omega \sin(\omega t)$$

allora

$$\ddot{\theta}_d(t) = A\omega^2 \cos(\omega t) \implies$$

$$u_d = mA\omega^2 \cos(\omega t)$$

Questo comando u_d riproduce la traiettoria desiderata date delle condizioni iniziali favorevoli, se così non fosse, sarebbe *impossibile* con la sola azione di feedforward, e sarebbe necessario chiudere l'anello con un'azione di feedback. Quest'ultima azione, è anche necessaria per risolvere errori dovuti a disturbi, che il feedforward non può correggere.

- **feed forward** : ottime prestazioni in condizioni nominali/ideali. La traiettoria desiderata deve però essere *liscia* (sufficientemente differenziabile).



5.2 Controllore del Motore DC

In questa sezione verrà analizzato e definito il progetto per il sistema di controllo del motore a corrente continua, gli obiettivi prefissati sono

- *Stabilità* (idealmente, robusta)
- *Asservimento* di una posizione angolare di riferimento $\theta_{rif}(t)$
- *Reiezione* dei disturbi non noti dati dalla coppia di disturbo $\tau_r(t)$

Il progetto coinvolgerà ben tre anelli di feedback posti "a cascata", verranno considerati "separatamente" il progetto elettrico e quello meccanico, quest'ultimo ha una velocità di risposta molto più lenta rispetto al primo.

- *Banda passante del controllo di posizione*: Indica l'intervallo di frequenze entro cui il sistema è in grado di seguire variazioni nella posizione desiderata. Una banda passante più ampia significa che il sistema può rispondere più rapidamente a cambiamenti nella posizione, ma potrebbe essere più sensibile a disturbi ad alta frequenza.
- *Banda passante del controllo di corrente*: Indica l'intervallo di frequenze entro cui il sistema è in grado di controllare la corrente nei motori o negli attuatori. Una banda passante più ampia del controllo di corrente garantisce una risposta più rapida del motore, ma potrebbe introdurre più rumore nel sistema.

la banda passante del controllo di posizione è circa 1/100 o 1/25 della banda passante del controllo di corrente. Questo significa che il controllo di corrente ha una risposta molto più rapida rispetto al controllo di posizione. Questa scelta è tipica nei sistemi di controllo, poiché il controllo della corrente è solitamente più veloce e permette di ottenere una risposta più precisa del motore.

Ovviamente il progetto è sottoposto ad alcune limitazioni fisiche

- rumori
- saturazioni "fisiche"
- il controllore implementato è digitale, il tempo di campionamento non può essere infinitamente piccolo
- la banda passante dell'anello di feedback relativo al controllo della corrente è tipicamente di $1/2 \cdot 10^4 \frac{\text{rad}}{\text{sec}}$
- la banda passante dell'anello di feedback relativo al controllo della posizione è tipicamente di $400 \frac{\text{rad}}{\text{sec}}$

Come accennato nella sezione 5.1.2, può essere utilizzato in condizioni nominali, e può compensare esclusivamente disturbi misurabili (la cui dinamica nel tempo è nota), ed il comando è basato sull'inversione ingresso-uscita.

per la riproduzione esatta (da $t = 0$ in poi) di una traiettoria di riferimento in uscita, è necessario che il sistema (processo e eventuale controllore) si trovi in un ben definito stato iniziale, funzione della traiettoria desiderata e delle sue derivate all'istante $t = 0$

5.2.1 Esempio di FeedForward

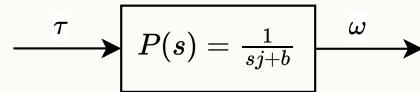
Il feedforward viene **combinato** con il feedback, chiudendo l'anello di controllo. Il solo feedforward non sarebbe necessario soprattutto in caso di paraemtri incerti di attrito ed inerzia ed i condizioni iniziali non favorevoli.

L'azione di feedback garantisce la stabilità asintotica e la reiezione dei disturbi o errori dovuti a condizioni iniziali, il feedforward invece migliora la riproduzione del riferimento che ha un andamento temporale complicato.

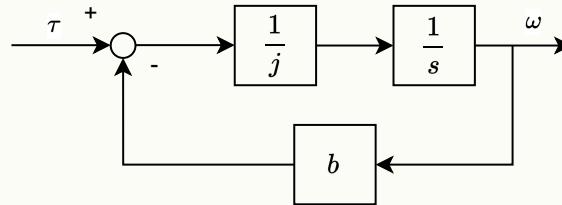
Per illustrare i concetti di feedForward, si consideri il sotto-caso del controllo della velocità angolare ω

del motore, in cui l'ingresso è la coppia τ . Sia j l'inerzia e b l'attrito viscoso. Il modello è descritto dall'equazione

$$j\dot{\omega} + b\omega = \tau$$



Il valore di τ si può ottenere per inversione dato un riferimento da seguire ω_d . Nel dominio del tempo l'ingresso $\tau_d(t)$ dovrà essere $\tau_d(t) = j\dot{\omega}_d(t) + b\omega_d(t)$. Si può riarrangiare lo schema a blocchi come segue:



Nel dominio di Laplace, essendo

$$\omega_d(s) = P(s)\tau_d(s)$$

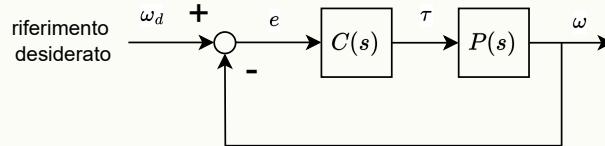
Si ha

$$\tau_d(s) = \frac{\omega_d(s)}{P(s)} = \omega_d(s) \cdot (js + b)$$

Come già trattato, per far sì che il comando τ_d riproduca l'uscita desiderata è necessario che le condizioni iniziali combacino, ossia che

$$\omega(0) = \omega_d(0)$$

Si utilizza un controllore PI per un azione feedback, che introduce due azioni, proporzionale (all'errore corrente) ed integrale (proporzionale all'errore passato) in modo da avere errore nullo a regime dato un riferimento ω_d costante.



Il controllore PI $C(s)$ ha funzione di trasferimento

$$C(s) = K_p + K_i \frac{1}{s} = \frac{K_p s + K_i}{s}$$

Si ha il guadagno in anello aperto

$$F(s) = C(s)P(s)$$

La funzione di trasferimento totale sarà quindi

$$W(s) = \frac{F(s)}{1 + F(s)} = \frac{K_p s + K_i}{js^2 + (b + K_p)s + K_i}$$

Vogliamo che l'uscita del sistema segua perfettamente il riferimento, quindi che $W(0) = 1$. Si ricordi che $W(0)$ è la funzione di trasferimento del sistema a regime permanente.

$$\omega = \omega_d \cdot W(0)$$

A questo punto, si **combina** all'anello l'azione di feedforward tramite un generatore statico di riferimento, che genera $\omega_d(t)$ e le sue derivate analitiche necessarie per il calcolo dell'azione τ da applicare al processo.

$$\begin{aligned} & \omega_d(t) \text{ riferimento da seguire} \\ & \tau_{ffw}(t) = j\dot{\omega}_d(t) + b\omega_d(t) \text{ coppia per azione feedforward} \end{aligned}$$

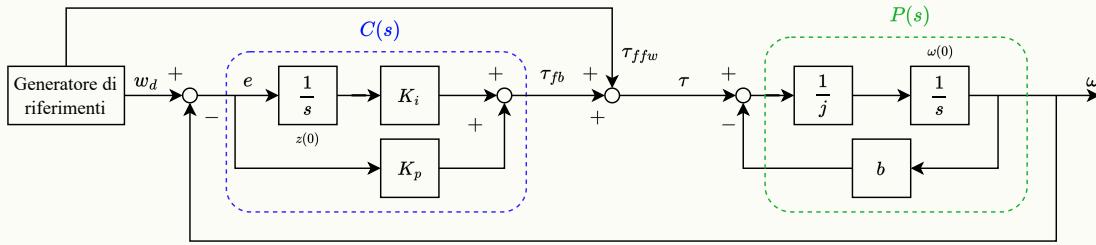
Inoltre, ci sarà anche la componente di feedback che calcola il valore della coppia τ_{fb} da dare in ingresso al sistema in base all'errore corrente

$$\tau_{fb}(t) = K_p(\omega_d(t) - \omega(t)) + K_i \int_0^t (\omega_d(\sigma) - \omega(\sigma)) d\sigma \quad \text{azione PI}$$

Inoltre le condizioni iniziali degli integratori devono "matchare"

$$\omega(0) = \omega_d(0)$$

$$\text{integratore PI : } z(0) = 0$$



5.2.2 Accenno ai Regolatori PID

Nonostante i regolatori PID godano di un capitolo a loro dedicato: 6, sarà necessario introurli al fine di descrivere il progetto del controllore del motore DC.

Definizione : un regolatore **PID** (Proporzionale-Integrale-Derivativo) è un regolatore che produce tre azioni di controllo proporzionali a

- L'errore istantaneo
- L'errore accumulato (il suo integrale)
- La stima dell'errore negli istanti successivi (derivata)

È un regolatore standard con procedure automatiche ed è semplice nella taratura in quanto ha solamente tre parametri (relativi alle tre azioni).

Sia e l'errore, definito come differenza tra l'uscita desiderata y_{rif} e l'uscita effettiva y del sistema

$$e(t) = y_{rif} - y(t)$$

Si hanno le tre azioni di comando

$$u(t) = K_p e(p) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

Dove

- K_p è il parametro proporzionale
- K_i è il parametro integrale
- K_d è il parametro derivativo

Ponendo

- $T_i = \text{Tempo di integrazione, con } K_i = \frac{K_p}{T_i}$

- $T_d = \text{Tempo di derivazione, con } K_d = K_p T_d$

Si può riscrivere

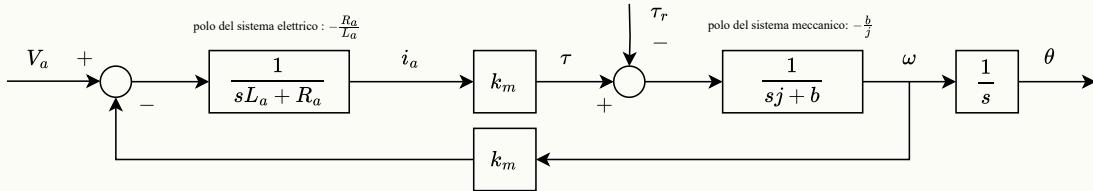
$$u(t) = K_p \left(e(p) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right)$$

- Azione Proporzionale** : Accelera il comportamento transitorio del sistema e riduce l'errore a regime. Un alto valore di K_p tende a destabilizzare il sistema.
- Azione Integrale** : Agisce anche quando l'errore è nullo ed annulla l'errore a regime. Tende a destabilizzare il sistema al diminuire del tempo di integrazione T_i .
- Azione Derivativa** : Anticipa l'errore evitando che la risposta del sistema si allontani troppo dal riferimento, rallenta la risposta del sistema ma tende a stabilizzarlo. La derivata effettiva dell'errore di può solamente approssimare (non si può guardare nel futuro).

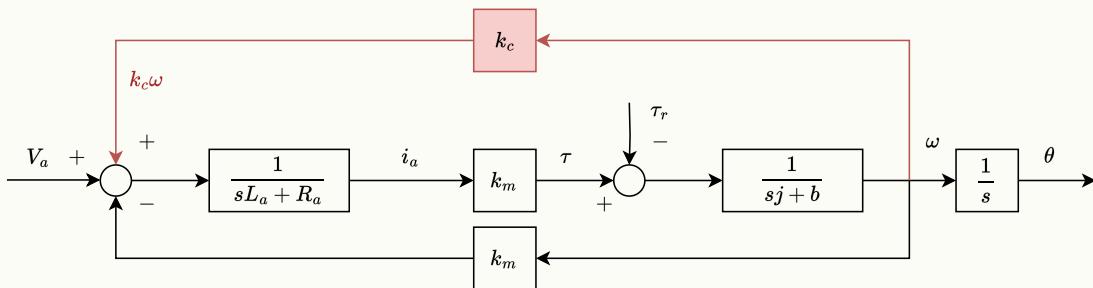
Aumentando l'azione	prontezza di risposta	margine di stabilità	errore a regime
P (aumento di K_p)	aumenta	diminuisce	diminuisce
I (diminuzione di T_i)	diminuisce	diminuisce	azzerata
D (aumento di T_d)	diminuisce	migliora	influente

5.2.3 Progetto del Controllore - Anelli a Cascata

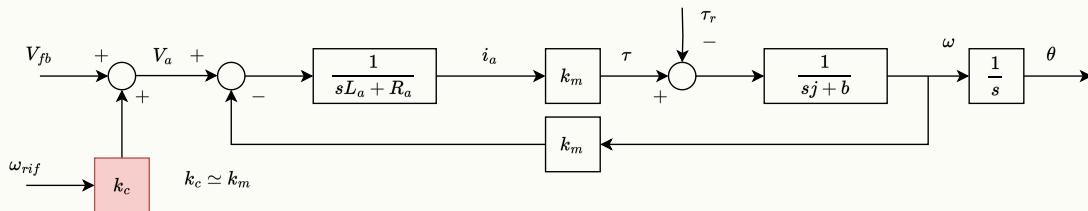
Nella sezione precedente si è data una dimostrazione della combinazione fra feedback e feedforward. In questa sezione verrà tratta la progettazione del controllore vero e proprio per il motore DC, il cui schema si ricordi essere



Si può eliminare la retroazione data dalla forza FCEM considerando una cancellazione positiva tramite un coefficiente k_c



In questo caso, possono insorgere problemi di instabilità nel caso $k_c > k_m$. Se il parametro k_m è perfettamente noto, si può compensare la FCEM (in condizioni nominali) tramite un'azione di feed-forward.



Si ricordi che il controllo del motore deve avvenire su 3 livelli

- controllo della corrente
- controllo della velocità angolare
- controllo della posizione

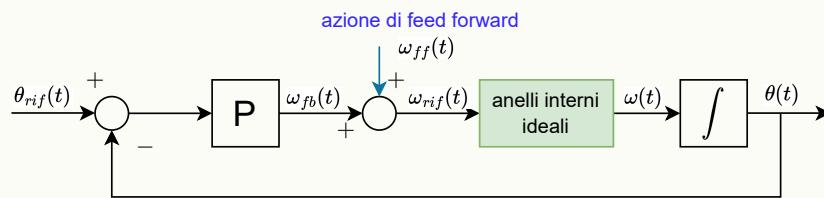
Una volta compensata la FCEM, si separa il progetto, dividendo il controllo *in tre anelli* di retroazione, considerando separate le 3 dinamiche.

L'idea alla base di questo approccio è quella di scomporre il sistema complessivo in sotto-sistemi più semplici, ognuno dei quali controlla una specifica variabile (posizione, velocità, corrente) e può essere progettato in modo indipendente. Questa semplificazione facilita la progettazione e l'analisi del sistema.

Il controllore della posizione, vedrà gli anelli interni come **attuatori ideali**, così come il controllore della velocità vedrà l'anello interno (corrente) come ideale.

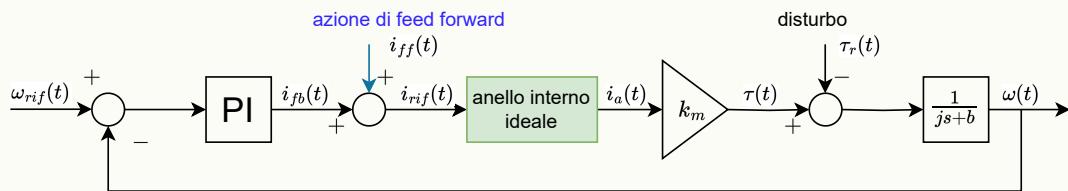
Anello Posizione

Il controllore della posizione genererà una posizione di riferimento, per l'errore calcolato verrà utilizzata esclusivamente un'azione proporzionale P, gli anelli interni sono visti come ideali. Tale anello è semplice in quanto la posizione θ si ottiene semplicemente integrando la velocità angolare, il processo è un integratore.



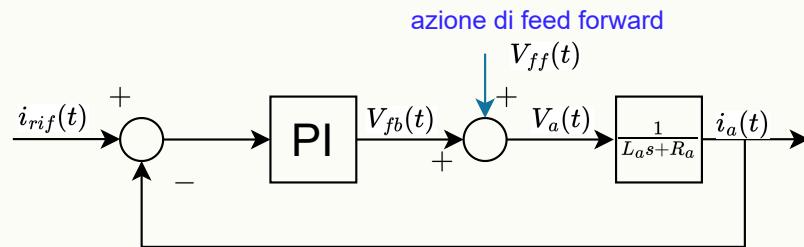
Anello Velocità

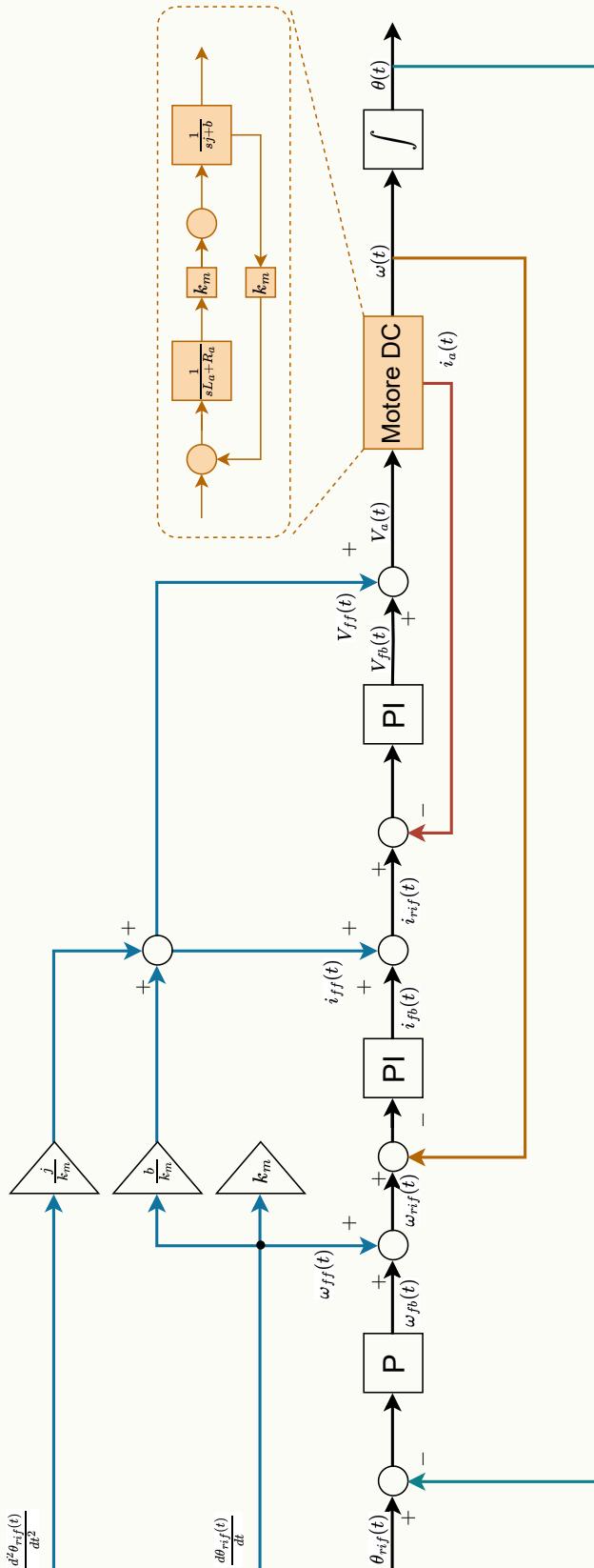
Il secondo anello (assunto ideale da quello più esterno) assumerà che l'anello interno (della corrente) sia ideale. Il processo da controllare sarà meccanico, e deve controllare la velocità angolare ω . Verrà regolato con un regolatore proporzionale ed integrale (PI) (l'azione integrale garantisce l'errore nullo a regime permanente), utile per la reiezione del disturbo di coppia τ_r .



Anello Corrente

L'ultimo anello deve regolare la corrente i_a . Viene adoperato un regolatore proporzionale integrale per garantire robustezza date le incertezze dei parametri elettrici. Si noti come su ogni singolo anello viene implementata un'azione di feed forward per compensare.

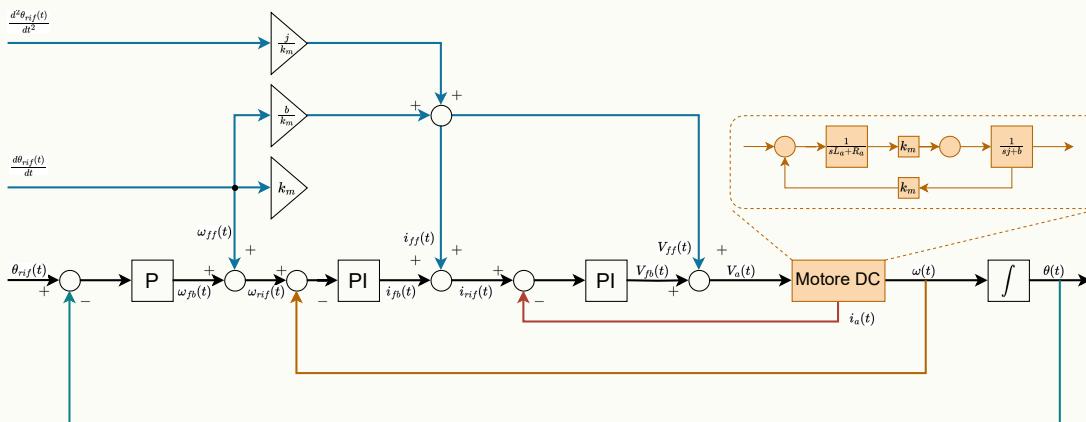




Si ricordi che la funzione di trasferimento di un regolatore PI è

$$K_p + \frac{K_i}{s}$$

Lo schema completo del regolatore del motore DC segue qui sotto. Si noti come vengono utilizzate le azioni di feedforward semplicemente valutando le derivate analitiche del segnale di posizione desiderato. L'immagine dello schema del motore DC è riportata in due versioni in modo che sia facilmente visualizzabile anche da un lettore che utilizza il PDF e non può ruotare il foglio.

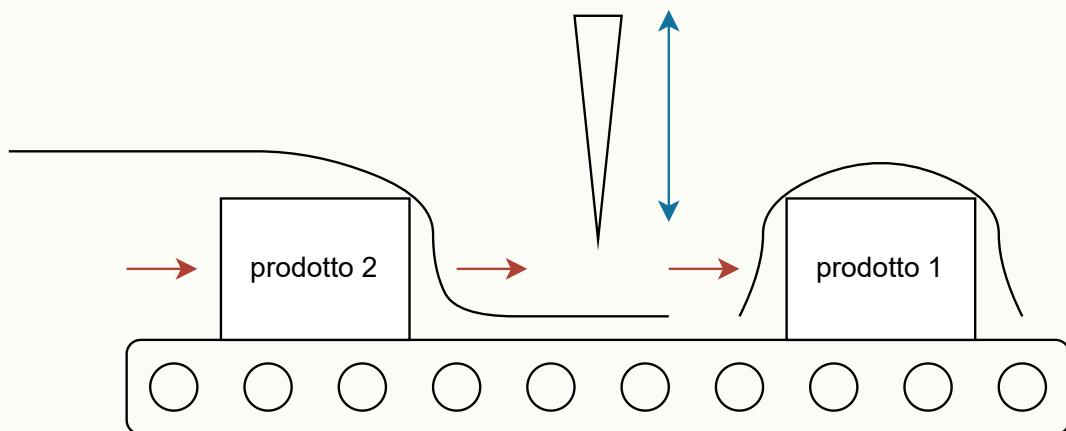


5.3 Sincronizzazione dei Moti

Quando un sistema ha più gradi di libertà (assi su cui ruotare), si può procedere in due differenti modi

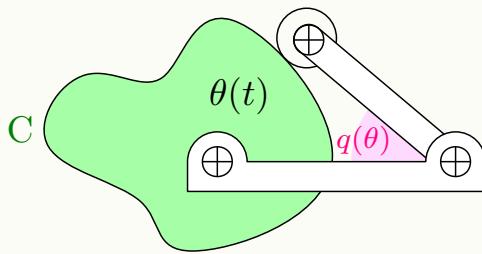
- tramite un sistema meccanico è possibile azionare più moti tramite un solo attuatore, in modo che l'azionamento sia meccanicamente coordinato
 - si considerano più attuatori coordinati fra loro

Si consideri il seguente esempio di movimentazione sincronizzata



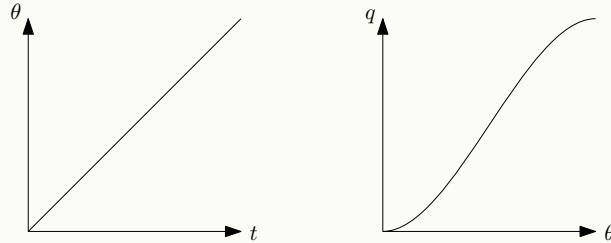
Vi è un rullo che sposta dei prodotti avvolti da una pellicola che un apposita taglierina deve tagliare, eseguendo l'azione periodicamente. È chiaro che il movimento del nastro deve essere opportunamente sincronizzato con lo spostamento dei prodotto sul rullo. Anche se le singole parti hanno velocità di azionamento diverse è possibile considerare un unica *sorgente di moto*, la cui velocità condiziona ogni azionamento, e quindi il tasso di produzione complessivo.

La distribuzione del moto può avvenire tramite delle **camme** o **catene cinematiche**.



Definizione : Una *camma* è un componente meccanico con una forma particolare, spesso a profilo curvilineo. Il suo scopo principale è quello di trasformare un moto rotatorio (come quello di un albero a camme) in un moto alternativo (come quello di una valvola). In altre parole, la camma serve a comandare il movimento di un altro elemento meccanico secondo una legge precisa.

La rotazione della camma C (indicata in verde) dipende dall'angolo θ , questo è detto *master*, in base alla forma della camma, varierà di conseguenza la variabile q , detta *slave*.



- pro : sistema robusto ed affidabile
- contro : molto rigido, poco flessibile

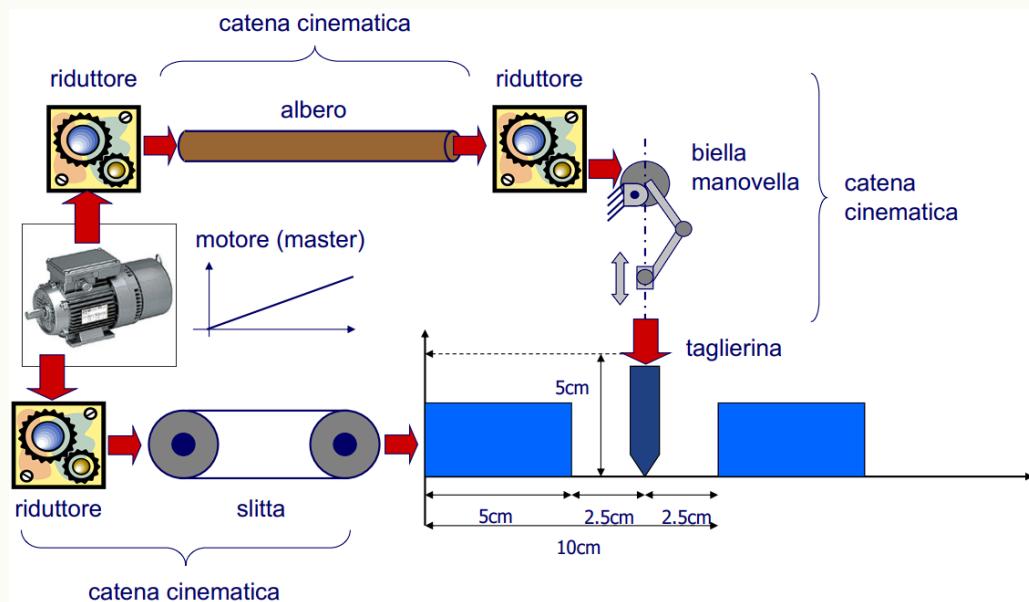


Figura 5.4: modello dettagliato dei prodotti sul rullo con taglierina

Con l'avvento dei calcolatori si sono introdotte le cosiddette *camme elettroniche*, il nome è fuorviante, non ci sono sistemi meccanici di trasmissione del moto, bensì, per ogni grado di libertà è presente un attuatore indipendente che viene controllato e coordinato da un software, che ne determina le traiettorie. La sincronizzazione avviene completamente a livello software, vengono separate le traiettorie spaziali da quelle temporali, si definisce un profilo master $\theta(t)$ che dipende dal tempo, e più profili slave $q_i(\theta(t))$ definiti in modo parametrico, che dipendono dal profilo master.

$$\theta(t) \Rightarrow \begin{cases} q_1(\theta) \\ q_2(\theta) \\ \vdots \\ q_n(\theta) \end{cases}$$

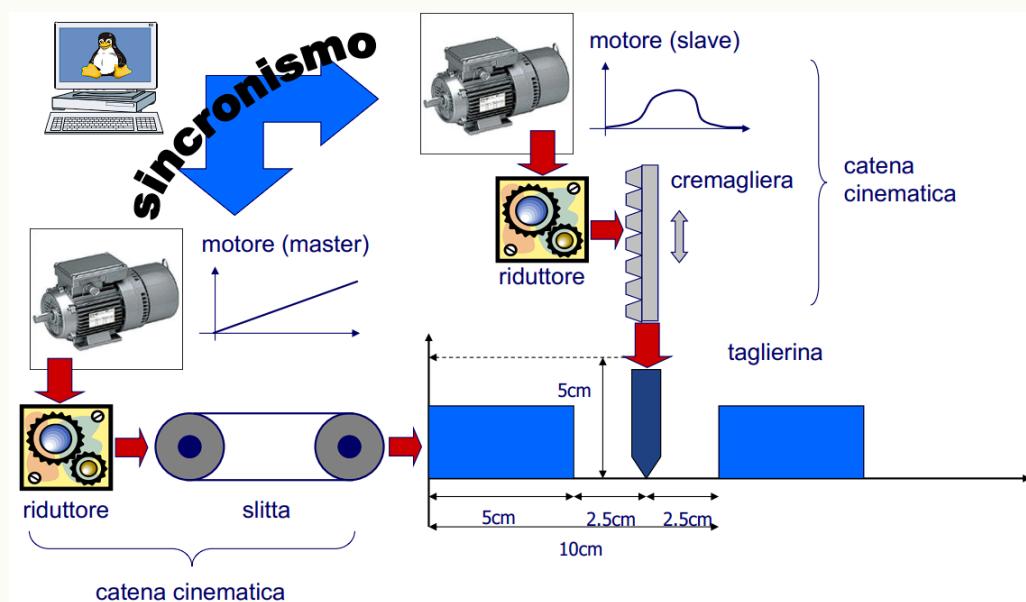
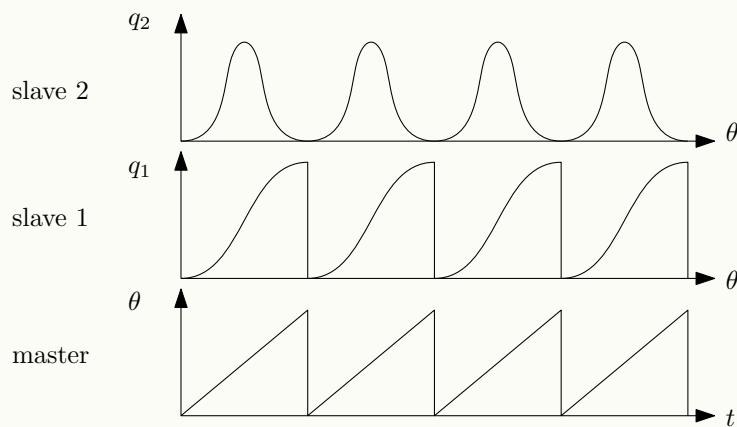


Figura 5.5: modello con più attuatori



Il motore master si muove (solitamente) ad una velocità costante v , i vari attuatori slave hanno dei limiti fisici di saturazione, riguardanti velocità massima ed accelerazione massima, essendo che questi dipendono da θ , la velocità v deve essere conservativa e rispettare tutti i limiti di velocità ed accelerazione. La velocità massima v_m che può assumere il profilo master è quindi

$$v_m = \min \left\{ \frac{v_{\max,i}}{|q'_i(\theta)|_{\max}}, \sqrt{\frac{a_{\max,i}}{|q''_i(\theta)|_{\max}}} \right\}$$

dove

- $v_{\max,i}$ è la velocità massima che può assumere l'attuatore controllato dal profilo q_i
- $a_{\max,i}$ è l'accelerazione massima che può assumere l'attuatore controllato dal profilo q_i
- $|q'_i(\theta)|_{\max}$ è il massimo della derivata prima di q_i
- $|q''_i(\theta)|_{\max}$ è il massimo della derivata seconda di q_i

$$\begin{aligned} q'(\theta) &= \frac{dq}{d\theta} \frac{d\theta}{dt} \\ q''(\theta) &= \frac{dq}{d\theta} \frac{d^2\theta}{dt^2} + \frac{d^2q}{d\theta^2} \left(\frac{d\theta}{dt} \right)^2 \end{aligned}$$

5.3.1 Profili di Moto

Il master θ ha una legge oraria rispetto al tempo, per le leggi parametriche degli slave q , si scelgono tipicamente delle leggi polinomiali

$$q = f(\theta) = a_0 + a_1\theta + a_2\theta^2 + \cdots + a_n\theta^n$$

È prefribile considerare una *doppia normalizzazione*, facendo variare θ fra 0 ed 1, se θ_{in} e θ_{fin} sono il massimo ed il minimo di θ , si considera la legge normalizzata

$$\theta_N = \frac{\theta - \theta_{in}}{\theta_{fin} - \theta_{in}} \in [0, 1]$$

Se q varia in $[q_{in}, q_{fin}]$, si considera anche la normalizzazione di questa

$$q_N = q_{in} + (q_{fin} - q_{in})f_N(\theta_N)$$

si ha

$$\begin{aligned} f_N(0) &= 0 \\ f_N(1) &= 1 \end{aligned}$$

Qualsivoglia siano le funzioni scelte per le leggi spaziali, è importante analizzarne il contenuto in frequenza, se i moti hanno contenuti di frequenze nella banda passante del sistema di controllo, potrebbero venire eccitate alcune frequenze.

La scelta del polinomio f_N (e del suo grado), dipende da quali sono le condizioni ricercate al contorno, si consideri un polinomio di grado 3

$$f_N(\theta_N) = a_0 + a_1\theta + a_2\theta^2 + a_3\theta^3$$

Si vogliono imporre le seguenti condizioni

- la funzione deve essere normalizzata $f_N(0) = 0 \wedge f_N(1) = 1$
- la derivata prima deve essere nulla all'inizio e alla fine

Quindi

$$\begin{aligned} f_N(0) &= a_0 = 0 \implies a_0 = 0 \\ f_N(1) &= a_0 + a_1 + a_2 + a_3 = 1 \\ \frac{df_N}{d\theta_N}(0) &= a_1 = 0 \\ \frac{df_N}{d\theta_N}(1) &= a_1 + 2a_2 + 3a_3 = 0 \end{aligned}$$

Si ha quindi il seguente sistema lineare

$$\begin{cases} a_0 = 0 \\ a_0 + a_1 + a_2 + a_3 = 1 \\ a_1 = 0 \\ a_1 + 2a_2 + 3a_3 = 0 \end{cases}$$

Si può risolvere semplicemente

$$\begin{cases} a_0 = 0 \\ 0 + 0 + a_2 + a_3 = 1 \\ a_1 = 0 \\ 0 + 2a_2 + 3a_3 = 0 \end{cases} \quad (5.3)$$

$$\begin{cases} a_0 = 0 \\ a_2 = 1 - a_3 \\ a_1 = 0 \\ 2a_2 = -3a_3 \end{cases} \quad (5.4)$$

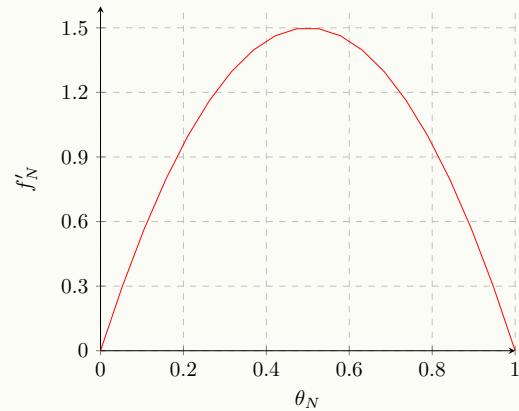
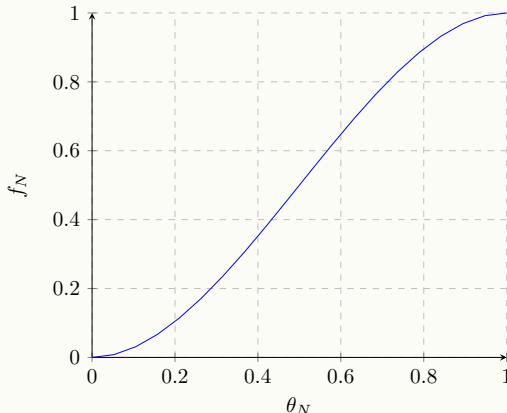
$$\begin{cases} a_0 = 0 \\ a_2 = 1 - a_3 \\ a_1 = 0 \\ 2(1 - a_3) = -3a_3 \end{cases} \quad (5.5)$$

$$\begin{cases} a_0 = 0 \\ a_2 = 1 - a_3 \\ a_1 = 0 \\ a_3 = -2 \end{cases} \quad (5.6)$$

$$\begin{cases} a_0 = 0 \\ a_2 = 3 \\ a_1 = 0 \\ a_3 = -2 \end{cases} \quad (5.7)$$

f_N sarà quindi definita come segue

$$f_N(\theta_N) = 3\theta^2 - 2\theta^3$$



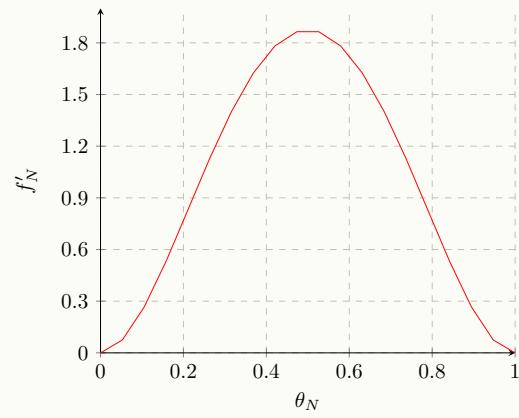
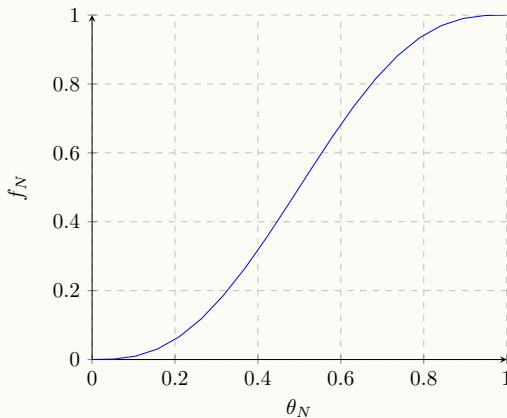
Se si volessero introdurre ulteriori condizioni (ad esempio, valori di contorno per la derivata seconda), un polinomio di grado 3 non sarebbe sufficiente a soddisfarle. Si consideri ad esempio che la traiettoria che si vuole seguire ha le seguenti condizioni

- la funzione deve essere normalizzata $f_N(0) = 0 \wedge f_N(1) = 1$
- la derivata prima deve essere nulla all'inizio e alla fine
- la derivata seconda deve essere nulla all'inizio e alla fine

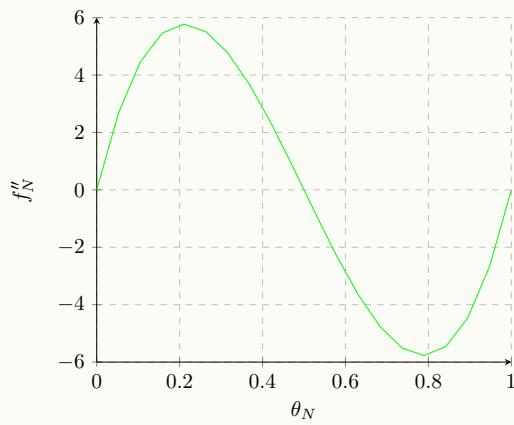
Sono necessari 6 coefficienti, e quindi un polinomio di grado 5. Senza mostrare la risoluzione del sistema lineare, il polinomio in questione sarà

$$f_N(\theta_N) = 10\theta_N^3 - 15\theta_N^4 + 6\theta_N^5$$

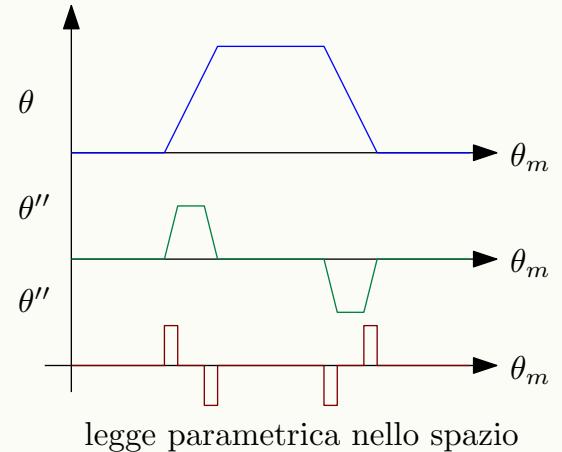
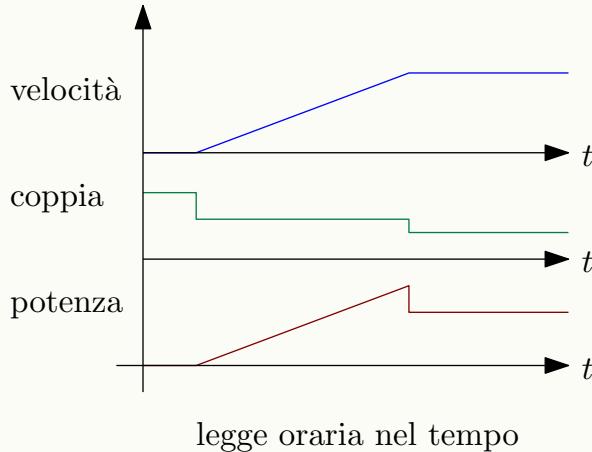
l'andamento di f e della sua derivata prima è il seguente



La derivata seconda ha il seguente andamento



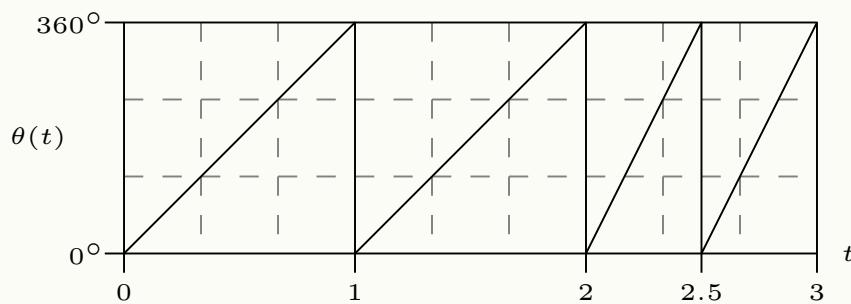
I seguenti grafici riportano alcuni esempi di profili temporali e di posizione, uno per un moto uniforme (passare da fermo a velocità costante) ed uno per un moto ciclico (in cui si ritorna alla posizione iniziale).



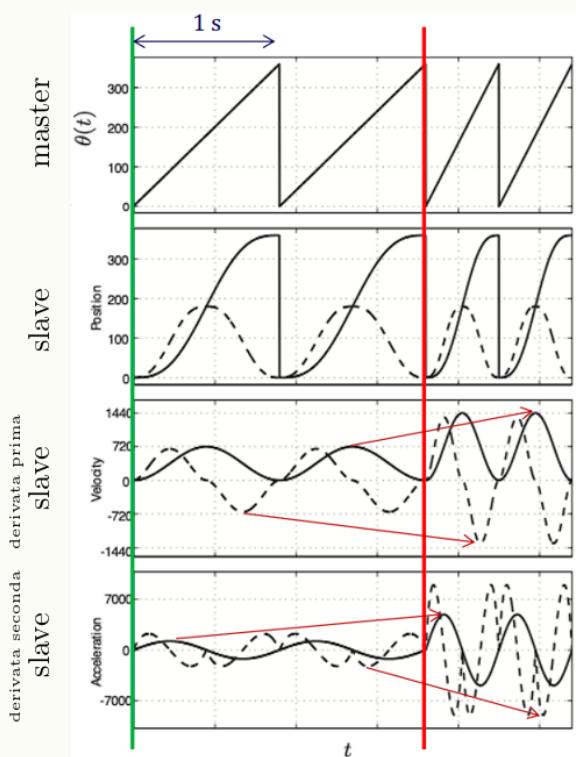
Cambio di Velocità

È importante inoltre, considerare che effetti può avere una variazione della velocità del profilo master θ sul moto dei profili slave. Si considera un profilo master la cui velocità cambia in due differenti fasi

- esegue prima due cicli lenti, ruotando ad una velocità di $360^\circ = 2\pi$ al secondo.
- esegue poi due cicli veloci, raddoppiando la velocità, e ruotando di $720^\circ = 4\pi$ al secondo.



Si osservi il seguente grafico che riporta l'andamento di due profili slave, uno rappresentato con una linea continua, e l'altro con una linea tratteggiata.



Sono cruciali le seguenti osservazioni

- se il master raddoppia la sua frequenza, anche i profili slave seguiranno il riferimento al doppio della frequenza
- Si osservi il profilo temporale della derivata prima degli slave (indicato con "velocity"), se nei cicli lenti la velocità varia fra $[-720, 720]$, nei cicli veloci, varia fra $[-1440, 1440]$, quindi la velocità raddoppia.
- Si osservi il profilo temporale dell'accelerazione, qui, al raddoppiare della velocità del master, i valori fra cui varia si quadruplicano.

CAPITOLO

6

REGOLATORI PID

In passato i regolatori PID venivano implementati in modo completamente analogico (ad esempio, tramite un sistema pneumatico), ad oggi sono implementati in maniera completamente digitale, lavorano su segnali analogici che vengono opportunamente campionati.



Figura 6.1: regolatore PID pneumatico

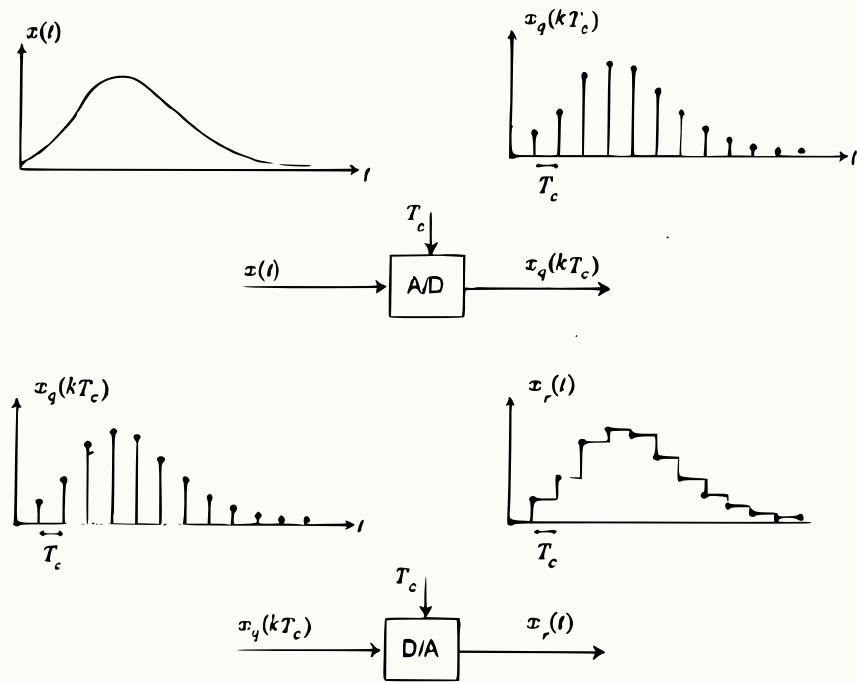
6.1 Campionamento

Essendo i PID (che considereremo) digitali, ed essendo i segnali su cui lavorare analogici, sarà necessario un opportuno campionamento, convertendo il segnale da logico a digitale, per poterci lavorare sopra, per poi applicare in uscita un segnale analogico ricostruito a partire da quello digitale sulla quale si è svolta l'operazione di controllo.

- $x(t)$ segnale analogico (continuo nel tempo $t \in \mathbb{R}$)
- $x_k = x(kT_C)$ segnale digitale discreto, $k \in \mathbb{Z}$, T_C è l'intervallo di campionamento, più questo è piccolo, più il segnale campionato sarà preciso.

Nella ricostruzione digitale-analogica, verrà considerato un organo di tenuta di ordine zero (*ZOH*), ossia, nei punti in cui il segnale digitale non è definito, il valore del segnale ricostruito x_r sarà identico (e costante) all'ultimo campionamento.

$$x_r(t) = x(kT_C), \quad t \in [kT_C, (k+1)T_C]$$



Per ottenere un segnale campionato, si moltiplica il segnale continuo $x(t)$ per un treno di impulsi unitari equidistanziati (di distanza T_C), tale segnale equivale a

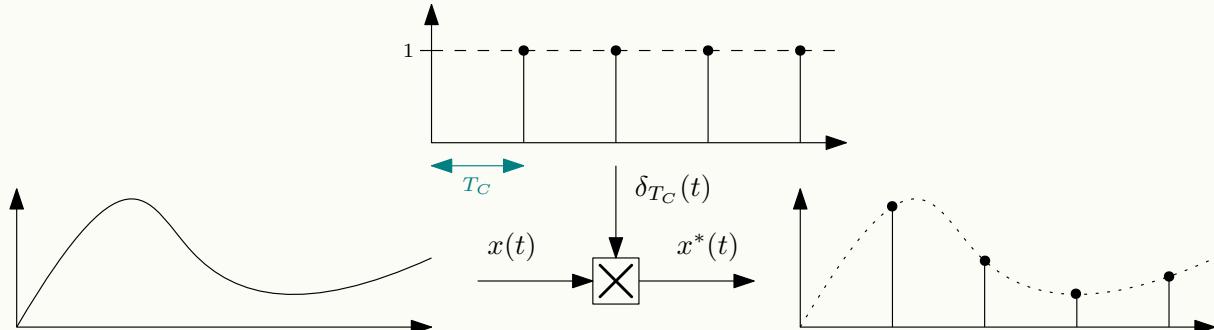
$$\delta_{T_C}(t) = \sum_{k=0}^{\infty} \delta(t - kT_C)$$

dove

$$\delta(t) = \begin{cases} 1 & \text{se } t = 0 \\ 0 & \text{se } t \neq 0 \end{cases}$$

il segnale campionato sarà quindi

$$x^*(t) = x(t)\delta_{T_C}(t)$$



Dato il segnale continuo $x(t)$ e la sua risposta in frequenza $X(j\omega)$, si può considerare lo spettro delle frequenze del segnale campionato

$$X^*(j\omega) = \frac{1}{T_C} \sum_{n=-\infty}^{+\infty} X(j\omega - jn\frac{2\pi}{T_C})$$

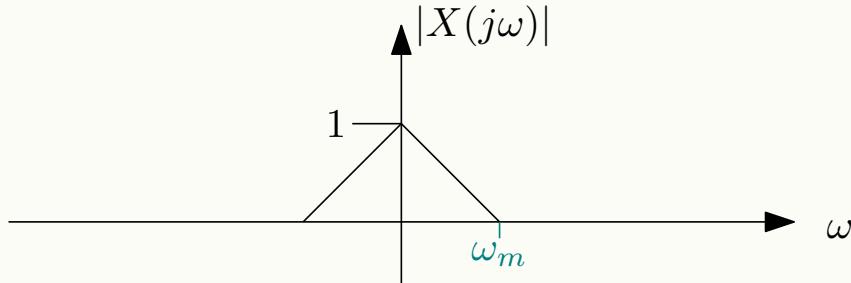
Si può riscrivere considerando la pulsazione di campionamento $\omega_C = \frac{2\pi}{T_C}$

$$X^*(j\omega) = \frac{1}{T_C} \sum_{n=-\infty}^{+\infty} X(j\omega - jn\omega_C)$$

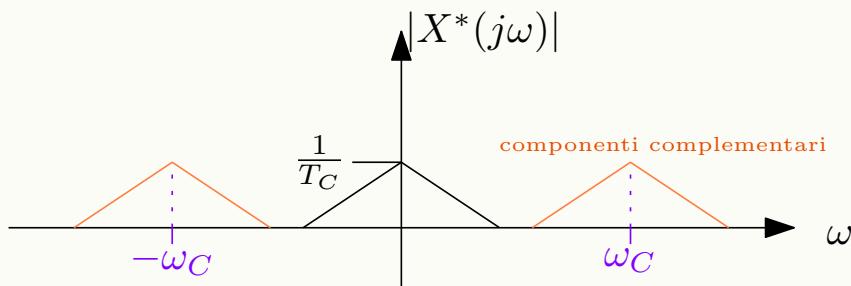
6.1.1 Perdita di Informazioni

Quando si campiona un segnale, necessariamente i valori di questo nei punti non coperti dall'intervallo di campionamento sono persi, è chiaro che un tempo di campionamento T_C più piccolo equivale ad una maggiore precisione. Si vuole inoltre ricostruire il segnale da digitale ad analogico in modo che sia il più possibilmente fedele al segnale desiderato.

Si consideri lo spettro di un segnale (continuo) $X(j\omega)$ limitato in banda



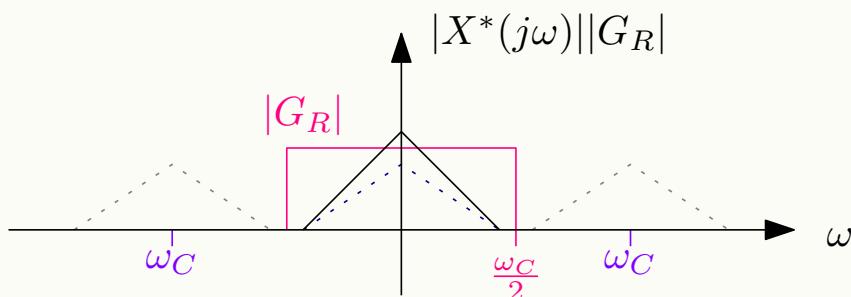
Quando questo viene campionato, il modulo della risposta in frequenza campionata $|X^*(j\omega)|$ sarà scalato di un fattore $\frac{1}{T_C}$. Inoltre, il campionamento di un segnale continuo nel tempo equivale, nel dominio della frequenza, a moltiplicare lo spettro originale del segnale per un treno di impulsi. Questa moltiplicazione ha l'effetto di replicare periodicamente lo spettro originale, creando così delle *componenti complementari*.



Il segnale originale $|X(j\omega)|$ si può ricostruire con un filtro ricostruttore G_R definito come segue

$$G_R(j\omega) = \begin{cases} T_C & \text{se } |\omega| \leq \frac{\omega_C}{2} \\ 0 & \text{altrimenti} \end{cases}$$

è chiaro che $|X(j\omega)| = |G_R(j\omega)| \cdot |X^*(j\omega)|$



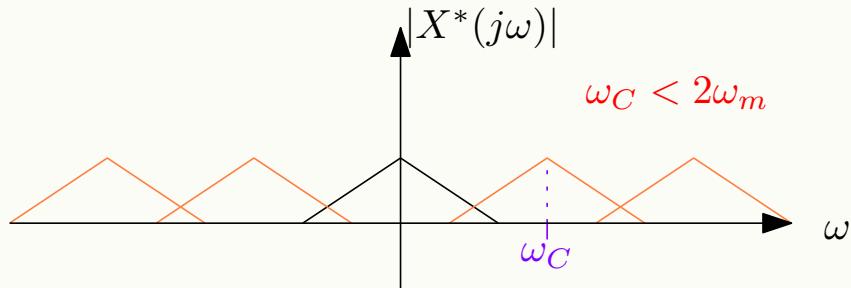
Tale esempio rende chiaro il seguente risultato.

Teorema di Shannon : dato il segnale $X(j\omega)$ sia $\omega_m = \max\{\omega_i \mid X(j\omega_i) > 0\}$, ossia la frequenza massima ammessa dal segnale, per far sì che il segnale si possa ricostruire senza errori a partire da un campionamento $X^*(j\omega)$, è necessario che la frequenza di campionamento ω_C sia almeno il doppio di ω_m

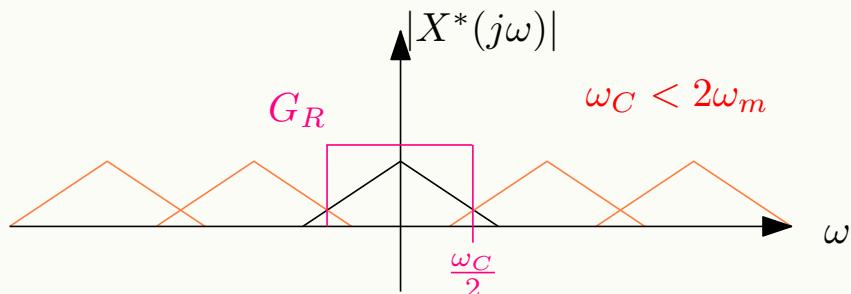
$$\omega_C \geq 2\omega_m$$

Se così non fosse, le bande complementari si sovrapporrebbero alla banda originale, causando il cosiddetto fenomeno di *aliasing*, che comporta una costruzione del segnale *corrotta*, a prescindere dal filtro utilizzato.

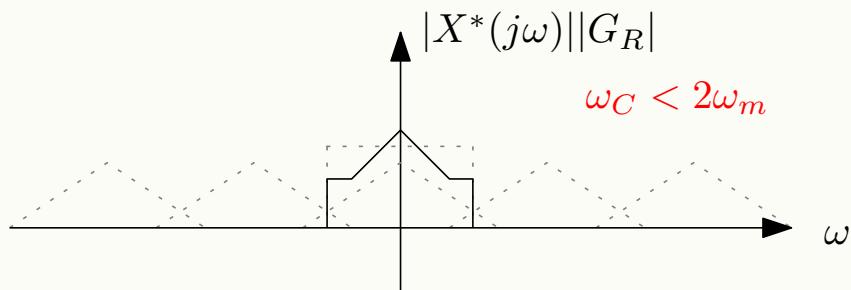
Per il segnale preso in considerazione nell'esempio precedente, si considera una frequenza di campionamento ω_C minore di $2\omega_m$



Si applica il filtro



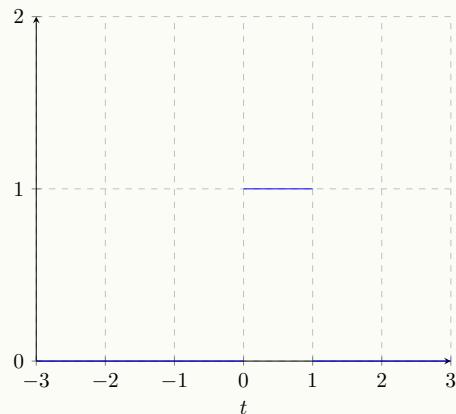
il segnale ricostruito è corrotto



I segnali reali, hanno componenti a frequenze sufficientemente alte da non rendere possibile una frequenza di campionamento sufficientemente alta, è quindi inevitabile che ci sia aliasing, per questo vengono utilizzati dei filtri analogici a monte prima del campionamento.

6.1.2 Ricostruzione ZOH

Per la ricostruzione si usa un filtro ZOH, la cui risposta al gradino unitario è la seguente (nell'esempio, $T_C = 1$)



Se $\delta_{-1}(t) = \begin{cases} 1 & \text{se } t \geq 0 \\ 0 & \text{se } t < 0 \end{cases}$ è il gradino unitario, si ottiene la risposta dello ZOH come la differenza di due gradini

$$h_0(t) = \delta_{-1}(t) - \delta_{-1}(t - T_C)$$

Nel dominio della frequenza si ha

$$H_0(j\omega) = \frac{1}{j\omega} - \frac{1}{j\omega} e^{-j\omega T_C} = \frac{1 - e^{-j\omega T_C}}{j\omega}$$

Mettendo in evidenza il termine $e^{-j\omega T_C/2}$ si ha

$$H_0(j\omega) = \frac{e^{j\omega \frac{T_C}{2}} - e^{-j\omega \frac{T_C}{2}}}{1} e^{-j\omega \frac{T_C}{2}}$$

sviluppando la formula di Eulero $e^{jx} = \cos x + j \sin x$ si ottiene

$$H_0(j\omega) = T_C \frac{\sin(\omega \frac{T_C}{2})}{\omega \frac{T_C}{2}} e^{-j\omega \frac{T_C}{2}}$$

si ha che

$$|H_0(j\omega)| = T_C \frac{\sin(\omega \frac{T_C}{2})}{\omega \frac{T_C}{2}} \implies |H_0(j\omega)| \simeq T_C \iff \omega T_C \ll 1$$

Riguardo il passo di campionamento

- T_C deve essere sufficientemente piccolo per evitare perdita di informazioni
- T_C non può essere troppo piccolo perché cresce troppo il costo computazionale

❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖ ❖

6.2 Progetto del Regolatore PID

6.2.1 Specifiche del Progetto

Le specifiche che idealmente un regolatore PID deve soddisfare sono

- stabilità asintotica
- errore sotto una certa soglia a regime permanente
- prestazioni dinamiche sul transitorio (ad esempio, velocità di risposta)
- robustezza ai disturbi
- specifiche su ingresso-uscita
- garantire uno sforzo di controllo non troppo elevato

Vanno considerati

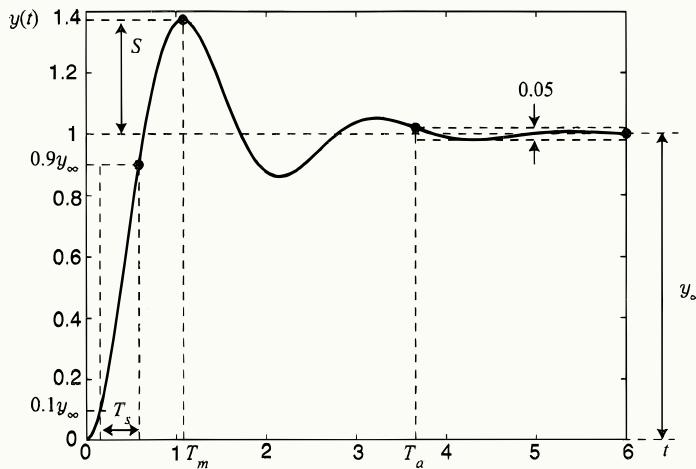
- limiti fisici degli attuatori
- limiti dovuti al fatto che il PID è implementato digitalmente (passo di campionamento)

I seguenti parametri si valutano sulla risposta del sistema al gradino unitario. Definiamo il **tempo di salita** T_s come il tempo che impiega la risposta a passare dal 10% al 90% del valore di riferimento.

Il **tempo di assestamento** T_a è il tempo che impiega l'uscita a raggiungere un margine di errore dal valore di riferimento minore o uguale del 3% – 5%.

T_m è l'istante di **massima sovraelognazione** S , questa è

$$S = \frac{y(T_m) - y(\infty)}{y(\infty)}$$



Tali specifiche devono essere soddisfatte al meglio delle possibilità, variando i 3 disponibili parametri di configurazione dei PID.

6.2.2 Caratteristiche del Regolatore

Il 95% dei regolatori digitali (industriali) implementa una legge di controllo PID, data la semplicità nella taratura dei parametri e la facile interpretazione fisica in termini di controllo. Dato un errore $e(t) = y_{rif}(t) - y(t)$, l'azione PID risulta essere

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right)$$

che si può riscrivere anche come

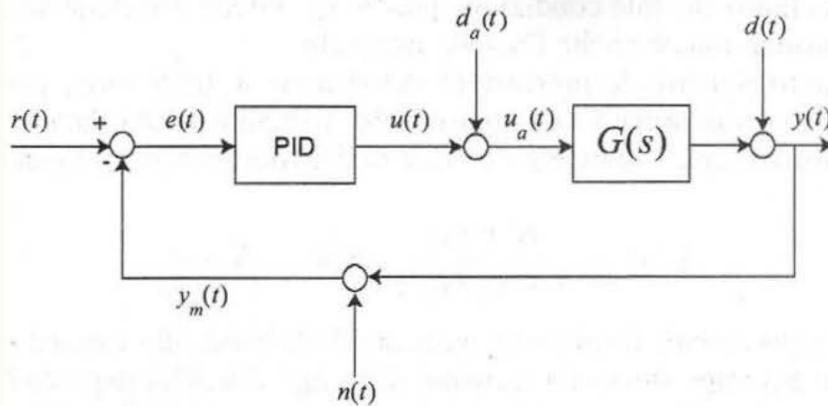
$$K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

con le ovvie relazioni

- $T_i = \frac{K_p}{K_i}$
- $T_d = \frac{K_d}{K_p}$

tali termini sono rispettivamente il *tempo di integrazione* ed il *tempo di derivazione*.

Lo schema di controllo in retroazione mediante il PID è il seguente



Dove $G(s)$ è la funzione di trasferimento del processo che si vuole controllare.

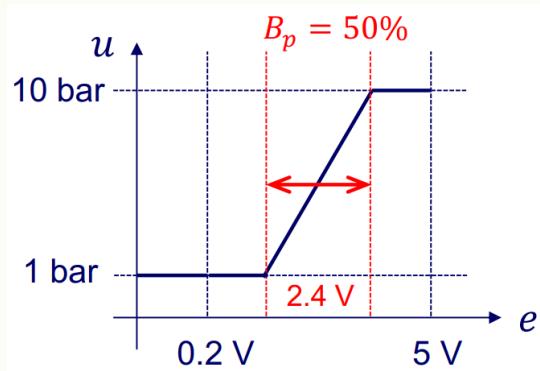
Piuttosto che K_p (chiamato guadagno proporzionale) nella pratica industriale si considera un altro parametro denotato B_p , e chiamato **banda proporzionale**. Tale valore riguarda la variazione dell'errore

(dato come ingresso al PID) e , de facto, B_p è la *minima variazione* necessaria che porta l'uscita del regolatore u dal minimo al massimo dei valori del suo range.

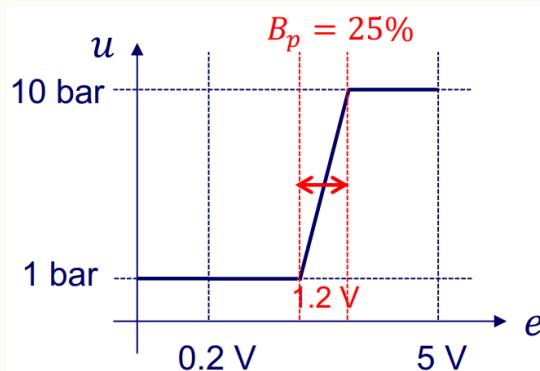
La definizione risulta più chiara con un esempio, sia u limitata fra 1 bar e 10 bar, ed e limitata fra 0.2V e 5V

$$e \in [0.2, 5] \quad u \in [1, 10]$$

Il valore di u è in funzione dell'errore e . Si osservi il seguente grafico



Una variazione di 2.4V fa sì che u passi dal minimo (1 bar) al massimo (10 bar), spaziando fra tutti i valori del suo intervallo. La banda proporzionale B_p è tale variazione espressa in percentuale rispetto il range di valori che può assumere e . Essendo $e \in [0.2, 5]$, si ha che $5 - 0.2 = 4.8$ è il range e $2.4 = B_p \cdot 4.8 \Rightarrow B_p = 0.5 = 50\%$. La banda proporzionale è quindi del 50%.



Se per far passare u dal minimo al massimo fosse necessaria una variazione di 2.4V allora la banda proporzionale sarebbe del 25%.

La banda proporzionale è relazionata al guadagno proporzionale, si ha che

$$K_p = \frac{u(t)}{e(t)}$$

ma questo solo nel caso in cui i minimi di u ed e siano 0 (senza offset), è più corretto considerare le variazioni di u ed e

$$K_p = \frac{\Delta u(t)}{\Delta e(t)}$$

Il guadagno proporzionale normalizzato si ottiene dal rapporto dei range di u ed e

$$K_{norm} = \frac{u_{range}}{e_{range}}$$

Nell'esempio precedente si ha

$$K_{norm} = \frac{9}{4.8} = 1.875$$

La banda proporzionale è legata alla *normalizzazione* del guadagno proporzionale, infatti

$$\frac{K_p}{K_{norm}} = \frac{1}{B_p} \Rightarrow B_p = \frac{K_{norm}}{K_p}$$

Nell'esempio precedente, con una banda proporzionale del 50% si può ricavare il guadagno proporzionale

$$\frac{K_p}{1.875} = \frac{1}{0.5} \implies K_p = 1.875 \cdot 2 = 3.75$$

6.2.3 Funzione di Trasferimento

Fisicamente è impossibile realizzare un'azione puramente derivativa, in quanto non si può "scorgere" il futuro, analizziamo un PID ideale ignorando momentaneamente questa irrealizzabilità. Sapendo l'azione nel tempo del PID è

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right)$$

la funzione di trasferimento nel dominio di Laplace è

$$PID_{ideale}(s) = \frac{u(s)}{e(s)} = K_p \left(1 + \frac{1}{T_i s} + T_d s \right)$$

Scrivendola in forma razionale si ha

$$PID_{ideale}(s) = \frac{K_p}{T_i} \frac{T_i T_d s^2 + T_i s + 1}{s}$$

La funzione di trasferimento mostra chiaramente l'irrealizzabilità dovuta al termine derivativo, infatti il sistema è improprio (il numeratore ha grado maggiore del denominatore) se e solo se è presente l'azione derivativa, ossia $T_d \neq 0$.

Vi è un polo nell'origine, e due zeri a parte reale negativa. Gli zeri sono reali se e solo se $T_i \geq 4T_d$.

In un progetto analitico, si possono accuratamente scegliere i valori T_d e T_i per far sì che tali zeri cancellano poli stabili del processo da controllare, se questi sono troppo lenti.

Se ne vuole dare un esempio, si consideri il seguente processo da controllare

$$G(s) = \alpha \frac{1}{(s+1)(s+0.1)}$$

ci sono due poli in $s = -1$ e $s = -0.1$, quest'ultimo è particolarmente lento e potrebbe causare una risposta lenta del sistema. Si può scegliere quindi un controllore PID con il termine $T_i = 10$ per cancellarlo.

Come già accennato, il PID reale deve avere una funzione di trasferimento propria. Si può approssimare l'azione derivativa aggiungendo un polo in alta frequenza su tale azione.

$$PID(s) = \frac{u(s)}{e(s)} = K_p \left(1 + \frac{1}{T_i s} + \frac{T_d s}{1 + \frac{T_d}{N} s} \right)$$

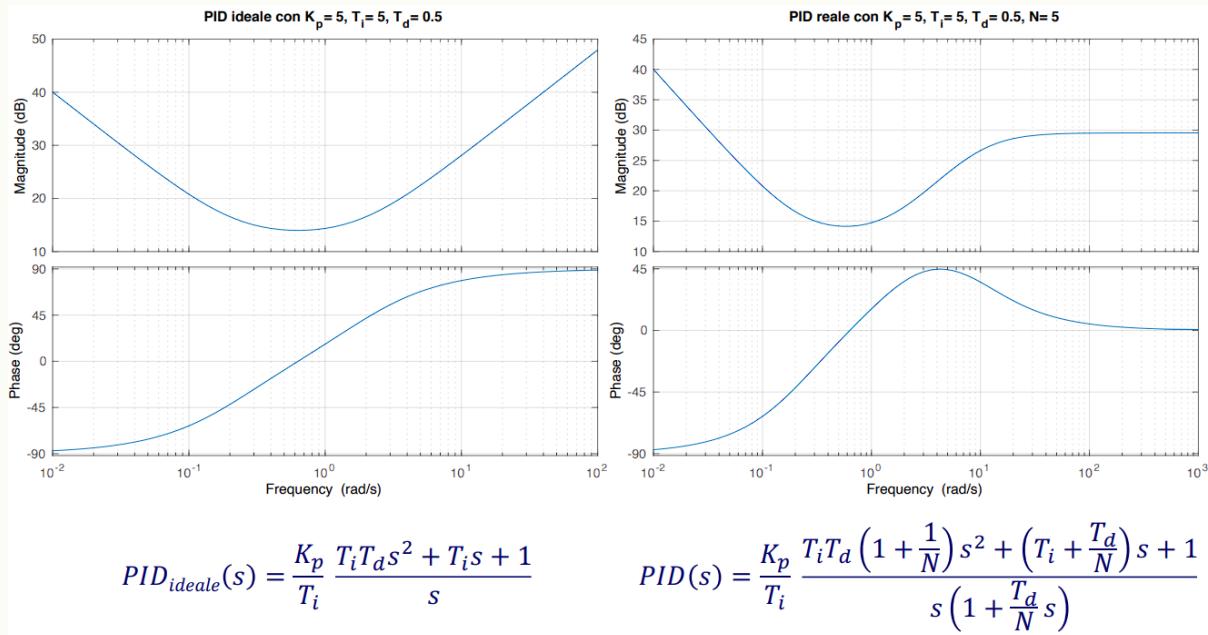
Questo polo limita la banda passante del derivatore, evitando che amplifichi eccessivamente il rumore ad alta frequenza presente nel segnale di errore. In pratica, il derivatore diventa meno sensibile alle variazioni rapide del segnale di errore. Ciò ne implica la realizzabilità fisica, un derivatore reale non può essere sensibile ad una variazione infinitesima.

La funzione del PID in forma razionale è

$$PID(s) = \frac{K_p}{T_i} \frac{T_i T_d (1 + \frac{1}{N}) s^2 + (T_i + \frac{T_d}{N}) s + 1}{s (1 + \frac{T_d}{N} s)}$$

Si è aggiunto un secondo polo in $s = -\frac{N}{T_d}$, rendendo la funzione di trasferimento propria, quindi realizzabile. Inoltre

$$\lim_{N \rightarrow +\infty} PID(s) = \frac{K_p}{T_i} \frac{T_i T_d (1 + 0) s^2 + (T_i + 0) s + 1}{s (1 + 0 s)} = PID_{ideale}(s)$$



~*~*~*~*~*~*~*~*~*~*~

6.3 Realizzazione Digitale

Il regolatore PID implementato su un sistema digitale presenta l'azione integrale, questa deve essere approssimata nel dominio discreto dei sistemi digitali. I segnali analogici come e vanno campionati, si denota con T_C il passo di campionamento. Si ricordi che un segnale campionato x_k è definito come

$$x(kT_C), \quad k \in \mathbb{Z}^+$$

Definizione : Dato un segnale campionato x_k , la trasformata \mathcal{Z} del segnale è la funzione di variabile complessa $z \in \mathbb{C}$ definita come

$$X(z) = \mathcal{Z}[x_k](z) = \sum_{k=0}^{\infty} x_k \cdot z^{-k}$$

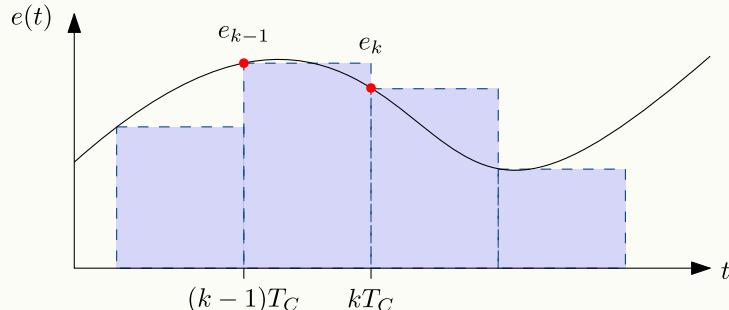
Vediamo adesso diversi modi di implementare nel dominio discreto l'integrazione.

Integrazione rettangolare

Sia u_k il segnale che approssima l'integrale dell'errore e fino all'istante $t = kT_C$, ossia

$$\int_0^{kT_C} e(\tau) d\tau$$

Ad ogni istante k , si può sommare al valore corrente il valore dell'istante precedente, aggiungendo il contributo dell'errore e pesato per il passo di campionamento, è chiaro che più T_C è piccolo, più l'integrale sarà preciso.



L'integrale è descritto dalla formula ricorsiva

$$u_k = u_{k-1} + T_C e_{k-1}$$

Portandolo nel dominio della trasformata \mathcal{Z} si ha

$$U(z) = \mathcal{Z}[u_k](z) = U(z)z^{-1} + T_C E(z)z^{-1}$$

Dove $E(z)$ è la trasformata dell'errore e . Si ricordi che un passo all'indietro nel dominio \mathcal{Z} si esprime moltiplicando la funzione per z^{-1} .

La funzione di trasferimento dell'integratore, che lega l'ingresso e all'uscita u è data da

$$\frac{U(z)}{E(z)} = \frac{T_C z^{-1}}{1 - z^{-1}} = \frac{T_C}{z - 1}$$

Il termine $\frac{T_C}{z-1}$ è simile al termine $\frac{1}{s}$ che rappresenta l'integratore \int nel dominio di Laplace. Il polo $s = 0$ nel dominio di Laplace equivale al polo $z = 1$ nel dominio della trasformata \mathcal{Z} .

L'integrazione rettangolare può anche essere fatta "in avanti" considerando l'errore corrente

$$u_k = u_{k-1} + T_C e_k$$

In questo caso si avrà

$$U(z) = \mathcal{Z}[u_k](z) = z^{-1}U(z) + T_C E(z)$$

$$\frac{U(z)}{E(z)} = \frac{T_C}{1 - z^{-1}} = \frac{T_C z}{z - 1}$$

Esistono anche altri tipi di integrazione, ad esempio quella *trapezoidale* che ha funzione di trasferimento

$$\frac{U(z)}{E(z)} = \frac{T_C}{2} \frac{z + 1}{z - 1}$$

6.3.1 Discretizzazione del Regolatore

Si vogliono rappresentare le 3 azioni del PID in forma discreta, sull'errore $e(t) = y_{rif}(t) - y(t)$, per poter essere implementate in un sistema digitale che ha un clock, o meglio, passo di campionamento T_C .

- **azione proporzionale**

$$K_p e(t) \quad \text{diventa} \quad K_p e(kT_C) = K_p e_k$$

- **azione integrale**

$$\frac{K_p}{T_i} \int_0^t e(\tau) d\tau \quad \text{diventa} \quad \frac{K_p}{T_i} \sum_{j=1}^k T_C e_j$$

- **azione derivativa**

$$K_p T_d \frac{de(t)}{dt} \quad \text{diventa} \quad K_p T_d \frac{e_k - e_{k-1}}{T_C}$$

L'azione totale del PID in forma digitale è quindi

$$u_k = K_p e_k + \frac{K_p}{T_i} \sum_{j=1}^k T_C e_j + K_p T_d \frac{e_k - e_{k-1}}{T_C}$$

Il PID espresso in tal modo è detto in **forma di posizione**. Ovviamente quando si esegue un algoritmo che implementa il PID, ad ogni passo, per l'integrazione, non viene ri-eseguita l'intera somma dell'errore dall'inizio fino all'istante corrente, bensì si implementa in maniera ricorsiva considerando una variabile ausiliaria che tiene conto del valore integrale dell'istante precedente

$$u_{i,k} = u_{i,k-1} + \frac{K_p}{T_i} T_C e_k \quad \text{azione integrale ricorsiva}$$

si esprime quindi il regolatore in tal senso (**Proporzionale Integrale Derivativa**)

$$u_k = K_p e_k + u_{i,k-1} + \frac{K_p}{T_i} T_C e_k + K_p T_d \frac{e_k - e_{k-1}}{T_C}$$

Il PID digitale può essere espresso anche diversamente, nella cosiddetta **forma di velocità**. Si consideri la differenza fra l'azione in due istanti consecutivi

$$\begin{aligned} u_k &= K_p e_k + \frac{K_p}{T_i} \sum_{j=1}^k T_C e_j + K_p T_d \frac{e_k - e_{k-1}}{T_C} \\ u_{k-1} &= K_p e_{k-1} + \frac{K_p}{T_i} \sum_{j=1}^{k-1} T_C e_j + K_p T_d \frac{e_{k-1} - e_{k-2}}{T_C} \\ \Delta u_k &= u_k - u_{k-1} \end{aligned}$$

Sviluppando la differenza si trova che

$$\Delta u_k = K_p(e_k - e_{k-1}) + \frac{K_p T_C}{T_i} e_k + \frac{K_p T_d}{T_C} (e_k - 2e_{k-1} + e_{k-2})$$

L'espressione in forma di velocità del PID è la seguente

$$u_k = u_{k-1} + \Delta u_k$$

Nel dominio della trasformata \mathcal{Z} , si considera l'operatore del ritardi di campionamento z^{-1} , se U_k è la trasformata di u_k , allora

$$U_{k-1} = U_k \cdot z^{-1} \quad \text{è la trasformata di } u_{k-1}$$

Ne consegue che se

$$\Delta u_k = u_k - u_{k-1}$$

allora nel dominio \mathcal{Z}

$$\Delta U_k = U_k - U_k z^{-1} = (1 - z^{-1}) U_k$$

precisamente

$$\Delta U_k = (1 - z^{-1}) U_k = K_p(1 - z^{-1}) E_k + \frac{K_p T_C}{T_i} E_k + \frac{K_p T_d}{T_C} (1 - 2z^{-1} + z^{-2}) E_k$$

Dove E_k è la trasformata dell'errore e_k . Sapendo che $(1 - 2z^{-1} + z^{-2}) = (1 - z^{-1})^2$ si può riscrivere

$$\Delta U_k = (1 - z^{-1}) U_k = K_p(1 - z^{-1}) E_k + \frac{K_p T_C}{T_i} E_k + \frac{K_p T_d}{T_C} (1 - z^{-1})^2 E_k$$

Isolando U_k si trova un'espressione in z^{-1} che sarà utile per la realizzazione

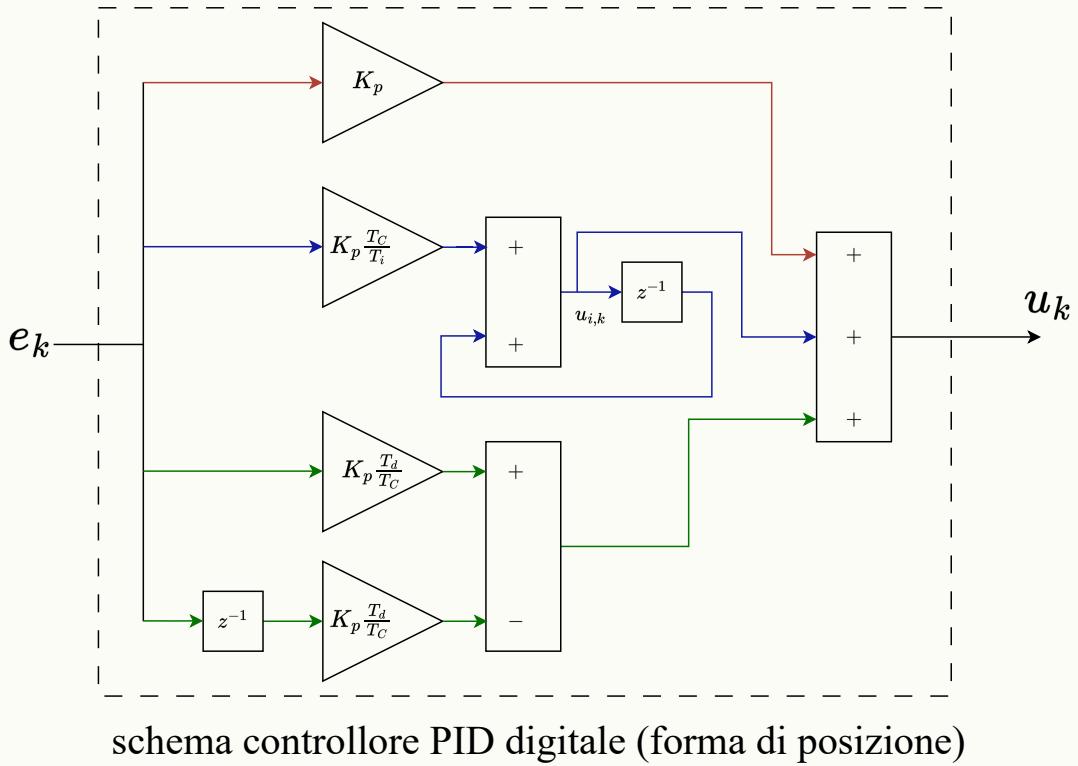
$$U_k = \left[K_p + \frac{K_p}{T_i} \frac{T_C}{1 - z^{-1}} + \frac{K_p}{T_C} T_d (1 - z^{-1}) \right] E_k$$

Il termine fra le parentesi quadre è la trasformata del PID digitale

$$PID(z) = \left[K_p + \frac{K_p}{T_i} \frac{T_C}{1 - z^{-1}} + \frac{K_p}{T_C} T_d (1 - z^{-1}) \right]$$

$$U_k = PID(z) E_k$$

In seguito è riportata una possibile implementazione digitale del PID in forma di posizione. I colori servono esclusivamente ad evidenziare le 3 diverse azioni.

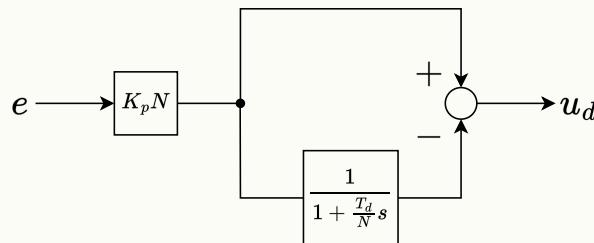


6.3.2 Derivata Filtrata in Banda

Abbiamo visto come, nel caso continuo, per poter far sì che il PID sia realizzabile si è considerato un polo in alta frequenza da aggiungere all'azione derivativa u_d del PID.

- ideale : $u_d = K_p T_d s$
- reale : $u_d = K_p T_d \frac{s}{1 + (\tau_d/N)s}$

Questo perché lo schema a blocchi di un sistema reale deve includere esclusivamente blocchi "causali", non possono essere presenti blocchi con delle funzioni improprie.



Nel dominio di Laplace l'azione derivativa vale

$$U_d(s) = K_p T_d \frac{s}{1 + \frac{T_d}{N}s} E(s)$$

Moltiplicando per $1 + \frac{T_d}{N}s$ si ha

$$\left(1 + \frac{T_d}{N}s\right) U_d(s) = K_p T_d s E(s)$$

Si vuole rendere realizzabile il PID digitale. Si ricordi che l'azione derivativa al passo k è

$$u_{d,k} = \frac{K_p T_d}{T_C} (e_k - e_{k-1})$$

Nel dominio \mathcal{Z} , facendo uso dell'operatore di ritardo z^{-1} si ha

$$\mathcal{Z}[u_{d,k}] = U_{d,k} = \frac{K_p T_d}{T_C} (1 - z^{-1}) E_k$$

considerando il termine derivativo approssimato mediante uso del filtraggio della derivata si ottiene

$$u_{d,k} = \frac{1}{\frac{T_d}{NT_C} + 1} \left[\frac{T_d}{NT_C} u_{d,k-1} + \frac{K_p T_d}{T_C} e_k + \frac{T_d}{T_C} e_{k-1} \right]$$

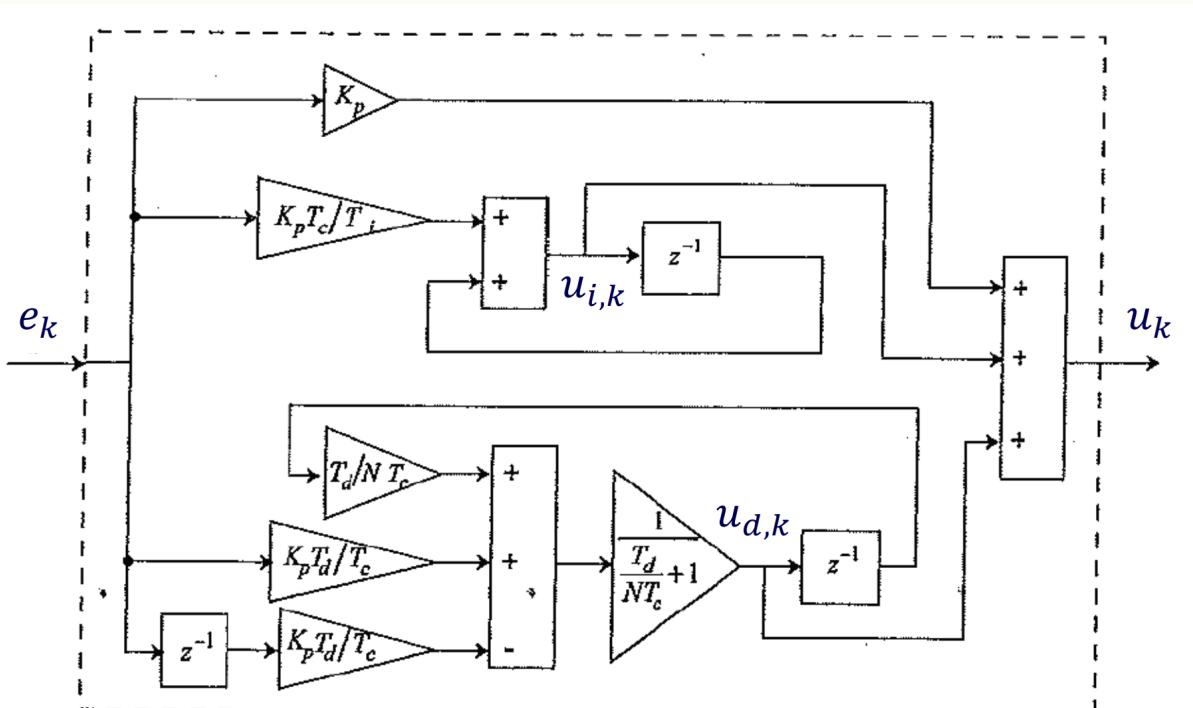
Nel dominio \mathcal{Z} con l'operatore z^{-1} si ha

$$U_{d,k} = \frac{K_p T_d}{T_C} (1 - z^{-1}) \frac{1}{1 + \frac{T_d}{NT_C} - \frac{T_d}{NT_C} z^{-1}} E_k$$

Infine quindi, il PID reale implementato digitalmente con la derivata filtrata in banda assume la forma definitiva

$$PID^*(z) = K_p + \frac{K_p}{T_i} \frac{T_C}{1 - z^{-1}} + \frac{K_p T_d}{T_C} \frac{1 - z^{-1}}{1 + \frac{T_d}{NT_C} - \frac{T_d}{NT_C} z^{-1}}$$

$$U_k = PID^*(z) E_k$$



controllore *PID* digitale con derivata filtrata in banda = PID^*

Si può moltiplicare per $\frac{z}{z}$ e riorganizzare l'espressione di $PID^*(z)$ per ottenere la versione fratta

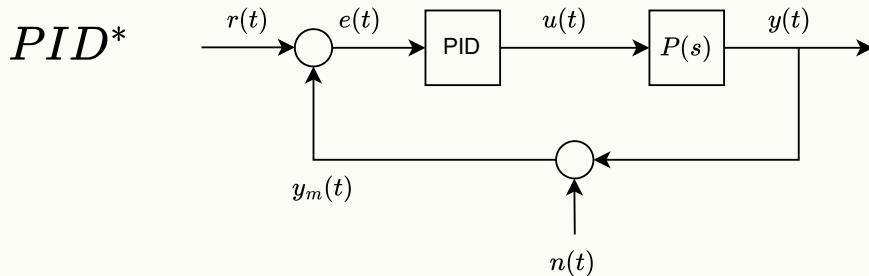
$$PID^*(z) = K_p \frac{a_2 z^2 + a_1 z + a_0}{(z - \frac{T_d}{T_d + NT_C})(z - 1)}$$

Dove a_0 , a_1 e a_2 sono i 3 coefficienti che dipendono dai parametri del PID, dal tempo di campionamento T_C e dalla scelta di N .

$$a_2 = 1 + \frac{T_c}{T_i} + \frac{NT_d}{T_d + NT_C}, \quad a_1 = -\frac{2N(T_d + T_c) + 2T_d + T_c T_d / T_i}{T_d + NT_C}, \quad a_0 = \frac{T_d}{T_d + NT_C} \left(N + 1 - \frac{T_c}{T_i} \right)$$

6.3.3 Schemi Realizzativi

Consideriamo adesso possibili schemi di controllo realizzativi che utilizzano una regola di controllo PID, sia $P(s)$ il processo da controllare. Sia $n(t)$ un disturbo sul sensore dell'uscita, e y_m l'uscita misurata affetta da tale errore.



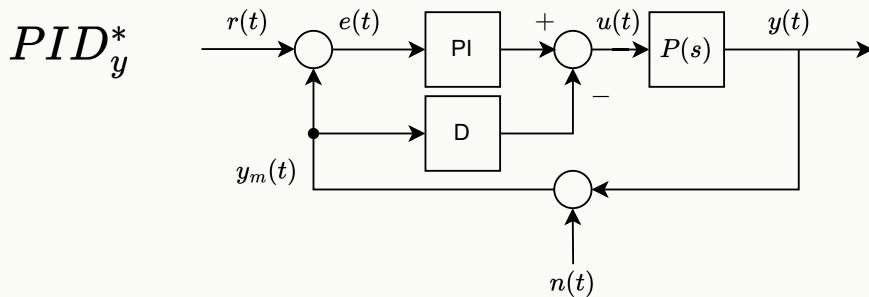
In questo caso, quando l'errore effettivo tende a zero $r(t) - y(t) \simeq 0$, l'errore dovuto al disturbo sulla misurazione diventa comparabile o addirittura maggiore all'errore effettivo.

Il segnale "utile" dell'errore che dovrebbe essere usato per la regolazione, risulta piccolo e viene "oscurato" dal disturbo, rendendo difficile per il regolatore prendere decisioni accurate sulla base delle misure.

Bisogna considerare che, l'errore può avere delle discontinuità molto marcate nei casi in cui il riferimento $r(t)$ passi da un valore costante ad un altro in un istante. Tali istanti di discontinuità rendono poco stabile il segnale derivativo che in quei punti avrà dei picchi dovuti alla variazione, che possono sollecitare gli attuatori in maniera innaturale. Inoltre solitamente, il segnale di riferimento è costante e la sua derivata è nulla.

$$r(t) = \text{costante} \implies \frac{de(t)}{dt} = \frac{dr(t)}{dt} - \frac{dy(t)}{dt} = -\frac{dy(t)}{dt}$$

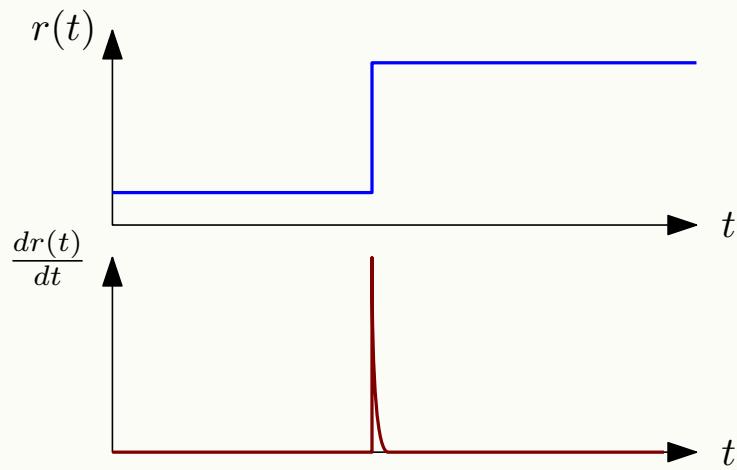
Risulta quindi utile calcolare l'azione derivativa *esclusivamente* sull'uscita del processo, come rappresentato in figura:



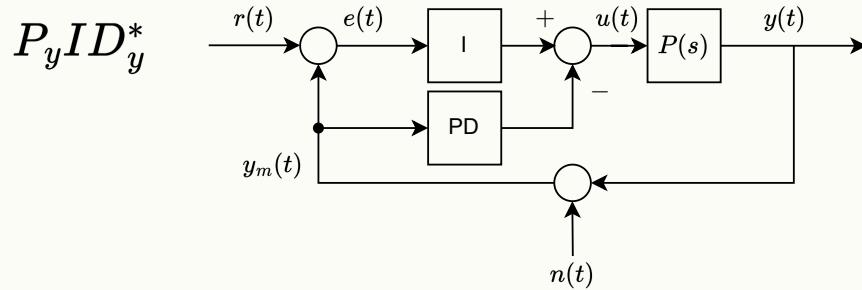
La y come pedice in PID_y^* indica che l'azione derivativa viene eseguita esclusivamente sul segnale di misura $y_m(t)$, l'asterisco come apice indica che è realizzata la derivata in banda.

Riassumendo ciò che è stato detto sopra

- non è utile derivare il riferimento $r(t)$ in quanto è costante
- variazioni istantanee del riferimento (gradino da una costante ad un'altra) fanno assumere all'azione derivativa dei picchi indesiderati



Esiste anche una terza implementazione in cui anche il termine proporzionale viene calcolato solamente sul segnale dell'uscita misurata $y_m(t)$, lasciando il segnale di errore $e(t)$ esclusivamente come input dell'azione integrale.



L'azione integrale conferirà ancora al regolatore la possibilità di annullare l'errore a regime permanente.

Il motivo per cui si esonera l'errore dall'azione proporzionale è analogo al caso precedente: Nell'istante in cui il riferimento cambia a gradino, da un valore costante ad un altro, c'è una grande variazione dell'errore che può mandare in saturazione gli attuatori.

6.3.4 Simulazione e Sforzo di Controllo

Si vuole mostrare la differenza fra un classico PID^* ed un PID_y^* , il processo da controllare è il seguente

$$P(s) = \frac{1}{(s+1)^3}$$

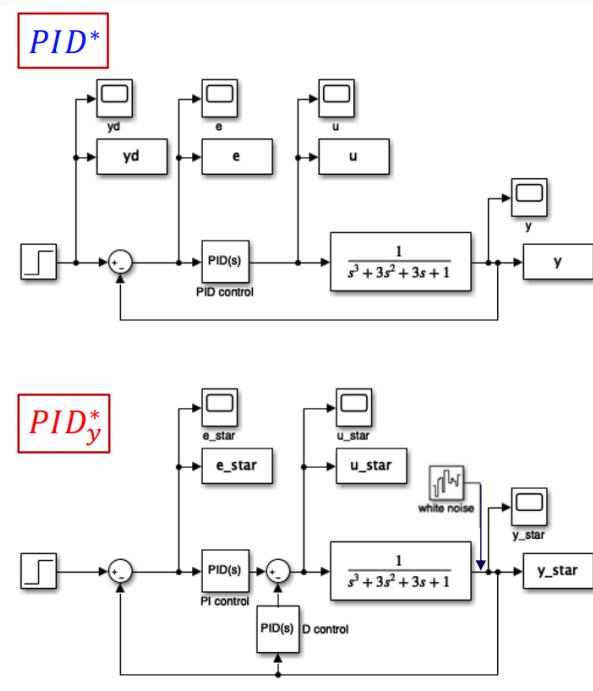
I parametri del PID ideale scelti sono i seguenti

$$K_p = 2, \quad K_i = 1, \quad K_d = 1$$

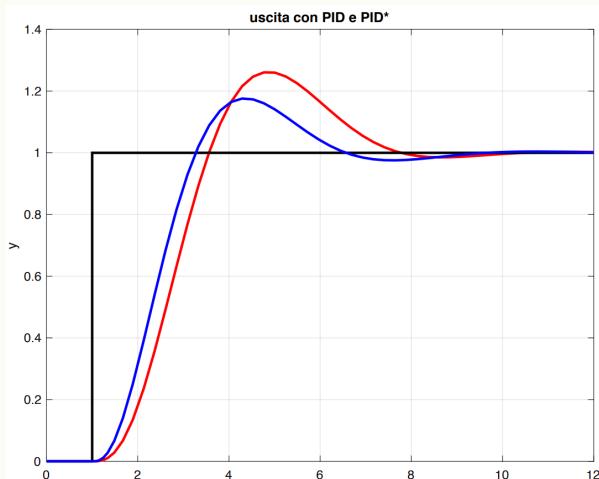
appositamente scelti per cancellare due poli stabili del processo, dato che la funzione del PID ideale è

$$PID(s) = \frac{(s+1)^2}{s}$$

$$PID(s)P(s) = \frac{(s+1)^2}{s} \cdot \frac{1}{(s+1)^3} = \frac{1}{s(s+1)}$$

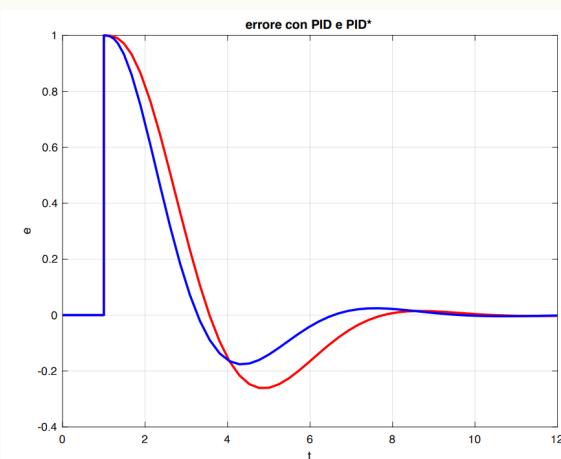


I seguenti grafici riportano la risposta y del sistema ad un ingresso a gradino (che parte nell'istante $t = 1$)

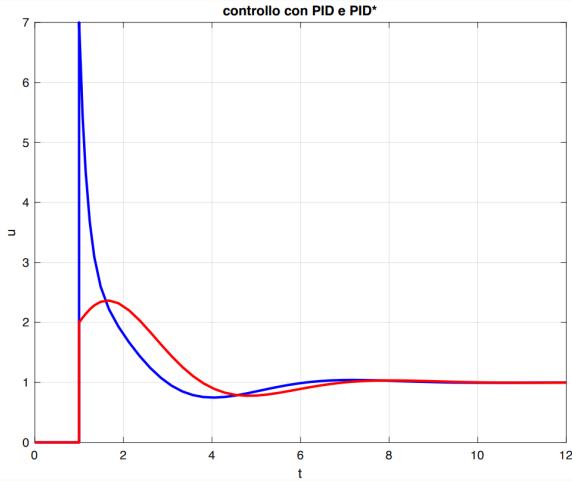


in rosso è riportata la risposta del **PID_y**, in blu la risposta del **PID***

Si può notare come non ci siano grandi differenze, bene o male entrambi i regolatori adempiono al loro compito, e possono essere considerati entrambi validi, è qui riportato anche l'andamento dell'errore $e(t)$



Nell'istante $t = 1$ il riferimento ha una variazione istantanea, ci si aspetta quindi che il PID^* abbia un picco in corrispondenza della discontinuità. Il seguente grafico mostra l'andamento del segnale di controllo u che viene dato in input al processo.

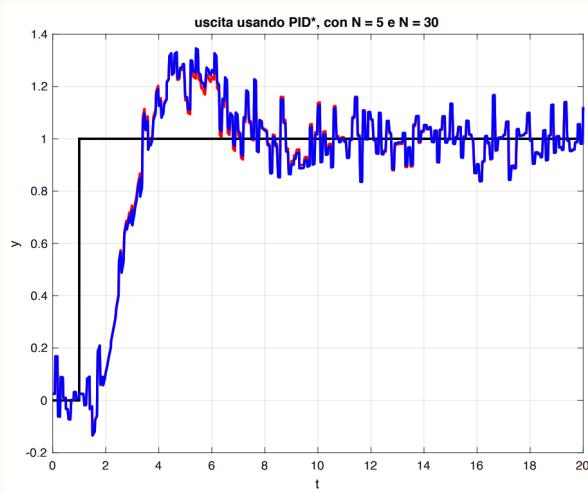


Come previsto, il PID_y^* mantiene un segnale di controllo stabile, invece il PID^* ha uno sforzo di controllo notevole, con un picco che può saturare l'attuatore, rendendo quindi il PID con azione derivativa solo sull'uscita misurata PID_y^* la scelta più adatta.

Vogliamo adesso misurare lo sforzo di controllo al variare del parametro N della derivata filtrata in banda, si ricordi che la funzione del PID con azione derivativa realizzabile è la seguente

$$\text{PID}(s) = K_p \left(1 + \frac{1}{T_i s} + \frac{T_d s}{1 + \frac{T_d}{N} s} \right)$$

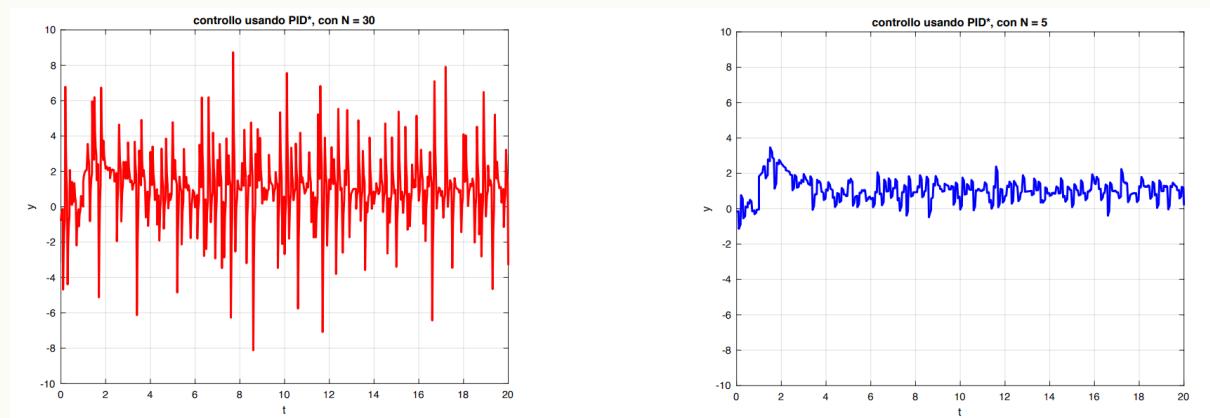
Si consideri esclusivamente il PID con derivata sull'uscita misurata PID_y^* , in due versioni, una con valore $N = 5$, ed un'altra con valore $N = 30$ (derivazione in banda con una frequenza maggiore). Si introduce la presenza di un disturbo sull'uscita in forma di rumore bianco a spettro uniforme. Gli andamenti dell'uscita del processo sono i seguenti



I segnali sono praticamente sovrapposti, e non vi è una grande differenza, si può quindi pensare che sia lecito aumentare il valore N per tener conto di variazioni in alta frequenza.

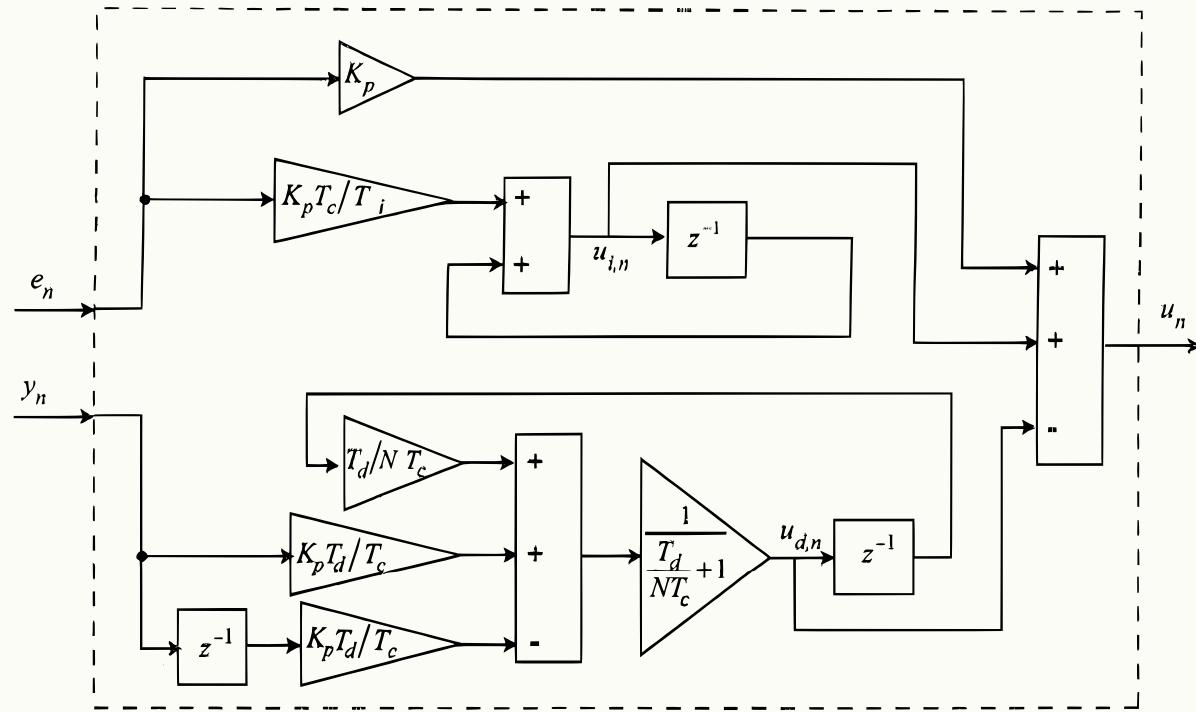
- si ricordi che N caratterizza "di quanto in alta frequenza" si sta valutando l'azione derivativa.

L'andamento del segnale di controllo del PID u , per i due valori di N , è il seguente



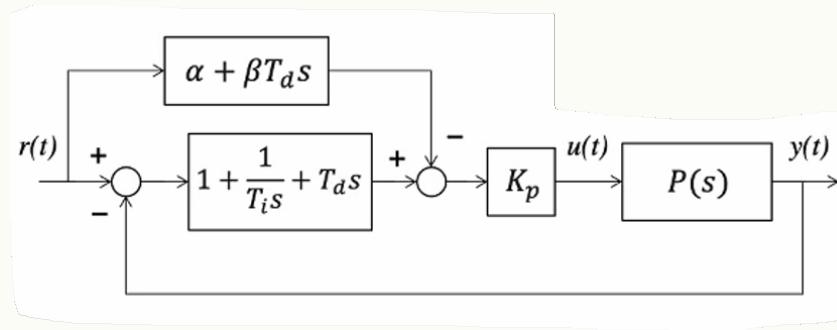
Il polo in alta frequenza $N = 30$ implica il PID ha continuato a derivare i segnali per una banda notevole. Nel caso $N = 5$ il controllo è nervoso (dato il disturbo) ma si mantiene limitato in (circa) $[-2, 2]$, mentre nel caso $N = 30$ il controllo è pieno di picchi che arrivano fino ad $u = 8$ o $u = 9$.

Lo schema del PID digitale PID_y^* con derivata in banda ed azione derivativa solo sull'uscita di misura è il seguente



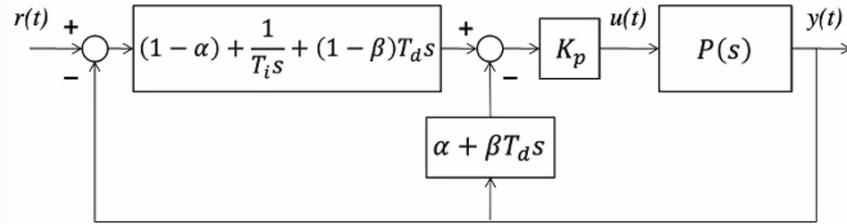
6.4 Desaturazione e Tuning

Riprendiamo in mano lo schema del PID (ideale e continuo), aggiungendo un azione di feedforward tramite un blocco aggiuntivo nel diagramma.



Si somma all'azione in retroazione un'azione che viene dal riferimento r , questo schema è stato standardizzato chiamato "a due gradi di libertà", dato che c'è un blocco che processa il riferimento ed uno che processa l'errore.

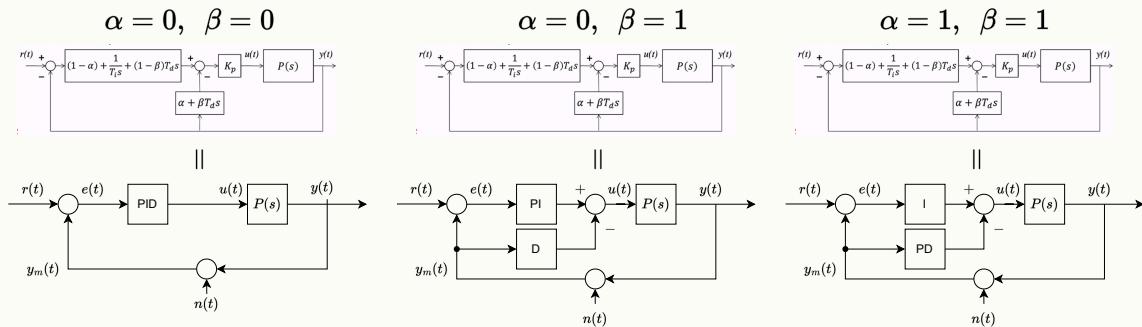
Quest'azione include il tempo di derivazione T_d (se presente l'azione derivativa) e due parametri α e β , uno schema equivalente a quello mostrato, che non include azione di feed forward ma porta i parametri dentro l'anello di controllo, è il seguente



tale azione entra con un segno negativo nel sommatore prima del blocco K_p . Questo schema è interessante in quanto si ha un termine che agisce direttamente sull'uscita di misura, ed un termine che agisce sull'errore, in maniera simile alle implementazioni del PID

- PID_y^*
- $P_y ID_y^*$

viste nella sezione precedente, questo schema non è altro che una generalizzazione dei possibili modi di implementare il PID già visti al variare di α, β .



E se α, β fossero compresi in $[0, 1]$? Si otterrebbero delle versioni "ibride" in cui una parte dell'azione derivativa (o proporzionale) è fatta sull'errore, ed un'altra parte è fatta sulla misura.

6.4.1 Saturazione dell'Attuatore

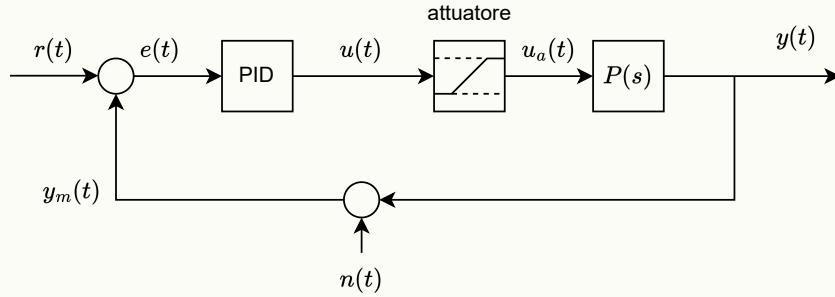
Ci sono dei fenomeni che si innescano in seguito ad una saturazione dell'attuatore legata al fatto che le variazioni dei segnali sono state "eccessive" e che l'azione integrale continua ad accumulare l'errore, nonostante il comando non possa realizzare il valore desiderato in presenza di saturazione. In questa sezione verranno analizzate le situazioni in cui l'uscita del regolatore PID non riesce a realizzarsi completamente, causando un fault dell'attuatore, o saturandolo, spingendo i guadagni oltre i limiti, ignorando lo sforzo

di controllo.

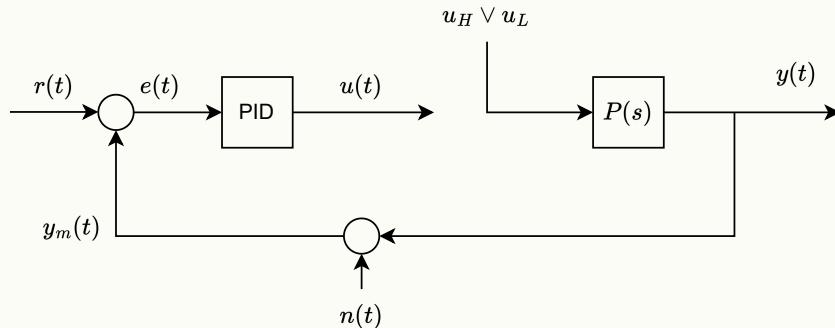
Definiamo la saturazione fisica dell'attuatore come dei limiti u_H, u_L imposti sul suo valore di ingresso nello schema a blocchi, "clampando" il valore di u

$$u_a(t) = \begin{cases} u_H & \text{se } u_H \leq u(t) \\ u(t) & \text{se } u_L \leq u(t) \leq u_H \\ u_L & \text{se } u_L \geq u(t) \end{cases}$$

soltamente (ma non necessariamente) $u_L = -u_H$.



In questa situazione di saturazione ($u_H \leq u(t) \vee u_L \geq u(t)$), il comando da dare al processo non dipende più dall'uscita controllata del PID, come se il comando fosse in anello aperto.



La presenza dell'azione integrale è **critica** in presenza di saturazione, questa ha una memoria di ciò che è successo, quindi l'azione integrale continua ad accumulare errore anche in una situazione temporanea di saturazione, quando questa finisce, avremmo un'accumulo di errore enorme a cui non è stata corrisposta un'azione che cercasse di ridurne l'accumulo.

Simulazione di saturazione da azione integrale

Vediamo un esempio numerico dell'evoluzione del sistema quando avviene la saturazione. Il processo da controllare è

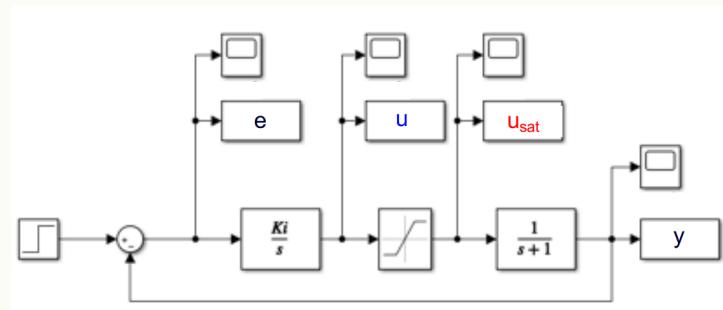
$$P(s) = \frac{1}{s + 1}$$

in seguito ad un ingresso a gradino a partire dall'istante $t = 1$. Si considera un controllore PID con sola azione integrale, quindi con parametri

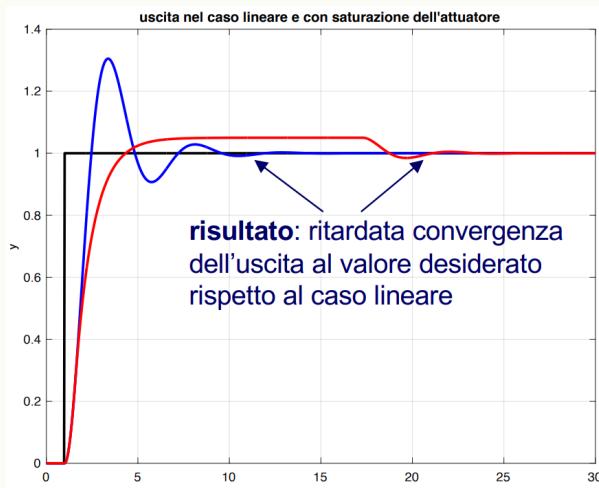
$$K_p = 0, \quad K_i = 2, \quad K_d = 0$$

L'attuatore del processo va in saturazione quando l'ingresso u supera i limiti, precisamente

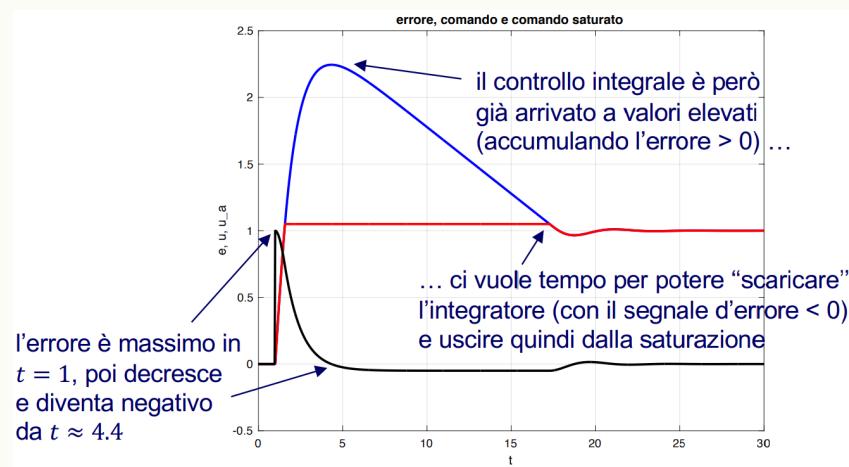
$$\begin{aligned} u_H &= 1.05 \\ u_L &= -1.05 \end{aligned}$$



Il seguente grafico ripora l'andamento dell'uscita y in presenza di saturazione (curva rossa) e senza saturazione (curva blu)

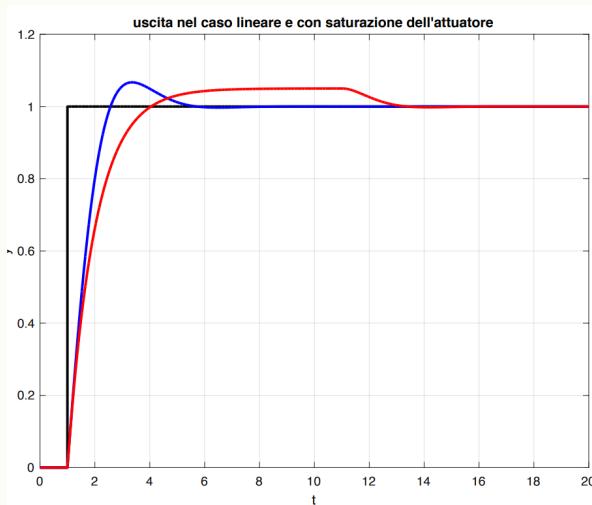


Nel caso lineare, in circa 10 secondi il sistema va a valore di regime. Se è presente saturazione, il comando da applicare si satura, e ciò è equivalente ad un abbassamento del guadagno integrale, rallentando la risposta. Nel seguente grafico sono riportati gli andamenti dei segnali di controllo, in entrambi i casi, e l'andamento dell'errore (curva in nero)



Solo quando l'errore diventa negativo si può "scaricare" l'accumulo dell'errore.

Considerando un PID con azione proporzionale $K_p = 1$ in aggiunta a quella integrale, gli andamenti dell'errore (in assenza e presenza di saturazione) sono i seguenti

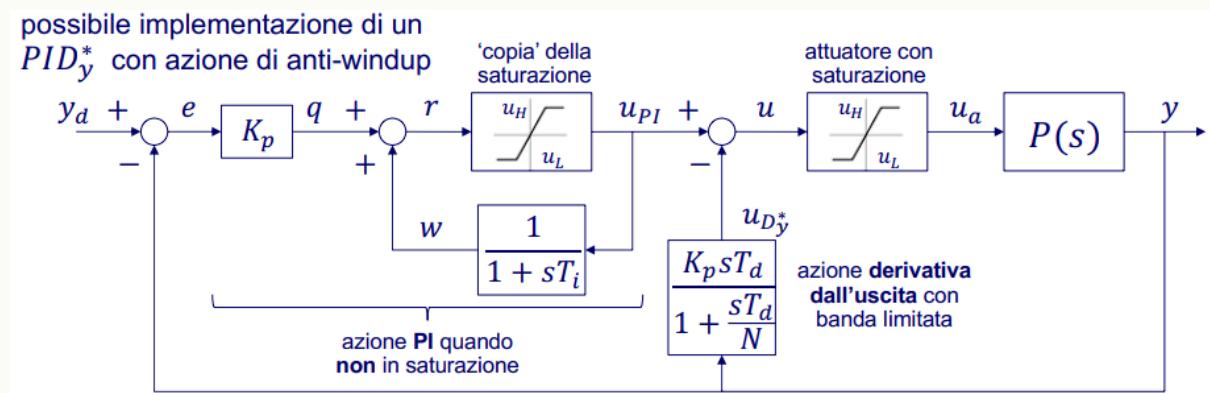


In assenza di saturazione il controllore è perfetto, in presenza di saturazione si causa lo stesso errore del caso precedente con sola azione integrale, in quanto è quest'ultima che causa il problema.



6.4.2 Anti Wind-Up

Abbiamo visto quali sono i problemi legati alla saturazione degli attuatori, una soluzione piuttosto sofisticata è l'azione di *anti wind-up*, con wind up si denota questo fenomeno di accumulo dell'errore integrale che fa degenerare l'azione di controllo. L'anti wind-up è relativo esclusivamente all'azione integrale.



Si osservi lo schema sopra riportato, e si noti come

- l'azione proporzionale del PID è rimasta invariata
- l'azione derivativa viene realizzata filtrata in banda ed esclusivamente sull'uscita misurata

Tali schemi di PID sono stati già presentati, la differenza in questo avviene sull'azione integrale. Si utilizza un *modello algebrico dell'attuatore*, che replica in tutto e per tutto la saturazione, nell'immagine indicato con entrata r ed uscita u_{PI} .

La saturazione è modellata nel termine integrale.

In una situazione di *non saturazione*, il segnale r entrante nel blocco replica non viene modificato, e si ha $u_{PI} = r$, nel *dominio di linearità* l'azione u_{PI} scaturita dal nuovo anello comporta dal segnale copia e dal blocco $\frac{1}{1+sT_i}$ è identica all'azione integrale proporzionale del PID classico

$$u_{PI} = \frac{1}{1 - \frac{1}{1+sT_i}} q(s) = \frac{1 + sT_i}{sT_i} q(s)$$

essendo $q(s) = K_p e(s)$

$$u_{PI} = \frac{1 + sT_i}{sT_i} q(s) = K_p \frac{1 + sT_i}{sT_i} e(s) \quad \text{azione PI classica}$$

Il segnale $w(s)$ rappresenta proprio l'azione integrale

$$w(s) = \frac{1}{1 + sT_i} u_{PI}(s) = \frac{K_p}{sT_i} e(s)$$

questo modello, fin tanto ché non si va in saturazione, è identico al modello classico PID, è quindi una sua diversa realizzazione

Nel caso si dovesse andare in *saturazione*, con

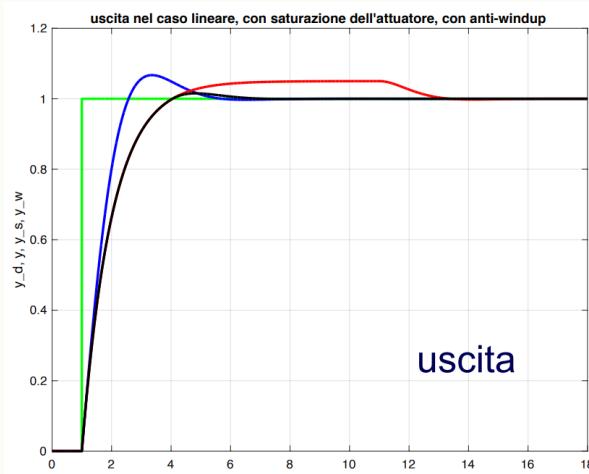
$$u_{PI} = u_H$$

o analogamente

$$u_{PI} = u_L$$

si avrebbe che il segnale r di ingresso è identico al segnale di uscita $u_{PI} = u_H$, stessa cosa per il segnale $w(t) = u_H$, e si smette di integrare l'errore. Il fatto che l'integrazione dell'errore si arresti può non risultare immediato, l'analisi più approfondita verrà trattata in seguito per l'implementazione digitale.

Si osservi il seguente grafico



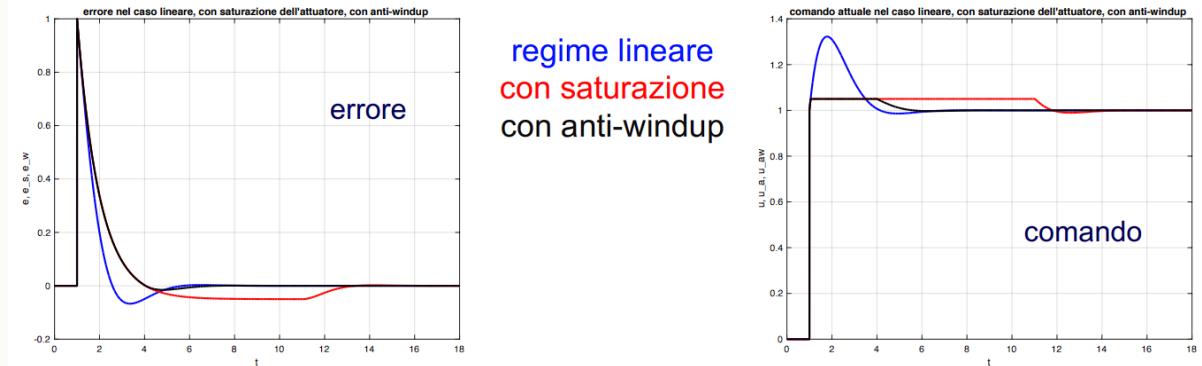
Si tratta dell'uscita del sistema precedente con processo $P(s) = \frac{1}{s+1}$, in risposta al gradino che parte in $t = 1$, i parametri del PID sono identici a quelli precedenti

$$K_p = 1, \quad K_i = 2$$

le curve colorate rappresentano

- in blu : il sistema in regime lineare senza saturazione (visto anche prima)
- in rosso : il sistema in presenza di saturazione (visto anche prima)
- in nero : il sistema in saturazione con azione anti wind-up

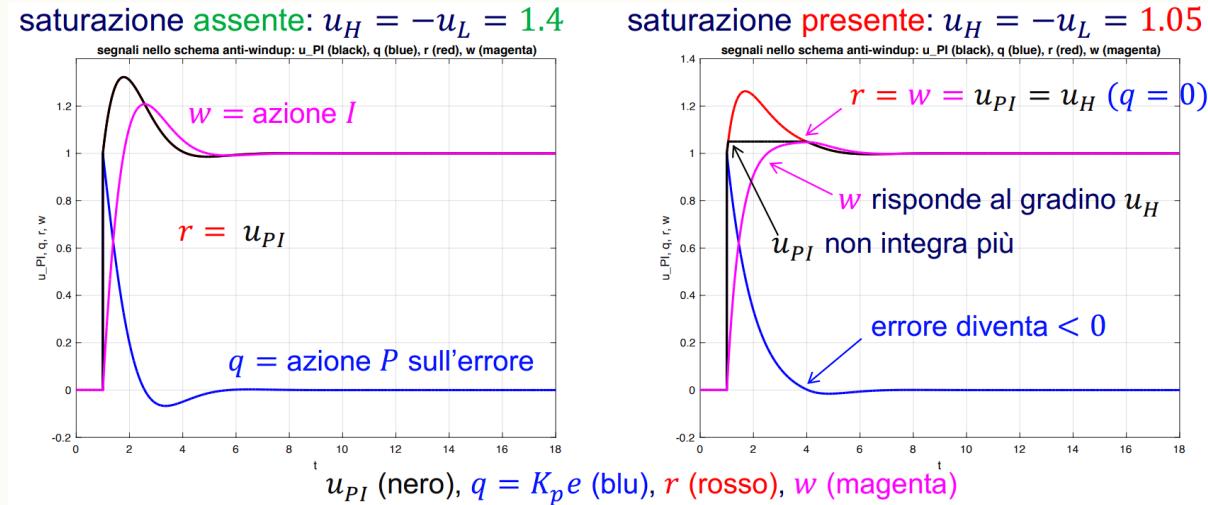
In seguito riportati gli andamenti degli errori e del comando.



il comando è molto simile al caso di saturazione, questo perché:

Il comando anti wind-up non evita la saturazione, ma nel momento in cui questa avviene, non viene più accumulato errore nell'azione integrale. Il comando va in saturazione per un tempo molto più breve.

i seguenti grafici riportano l'andamento dei segnali nello schema anti wind-up, ossia r, q e w , sia in saturazione che non.



6.4.3 PID Digitale con Anti Wind-Up

Consideriamo una trattazione più dettagliata dello schema con anti wind-up nel caso del PID digitale. Lo schema è riportato in figura 6.2. La prima parte fa un'azione PD che non considereremo, ci si sofferma sull'azione integrale, il segnale $w_{1,k}$ rappresenta la parte separata che implementa l'azione proporzionale derivativa, questa può essere una fra

$$PD \quad PD^* \quad PD_y^*$$

sono rappresentate due saturazioni, una sull'azione PD ed una sul comando finale, quest'ultima modella la saturazione dell'attuatore. In questo schema l'integrazione dell'errore è automaticamente bloccata in presenza di saturazioni. Evita l'accumulo, che dovrà poi essere scaricato quando l'errore cambia di segno.

Valutiamo il diagramma in assenza di saturazione, si ha che l'uscita u_k è

$$u_k = w_k = u_{1,k} + \frac{K_p T_C}{T_i} e_k + (u_{k-1} - u_{1,k-1}) = (1 - z^{-1})U_{1,k} + \frac{K_p T_C}{T_i} E_k + z^{-1} U_k$$

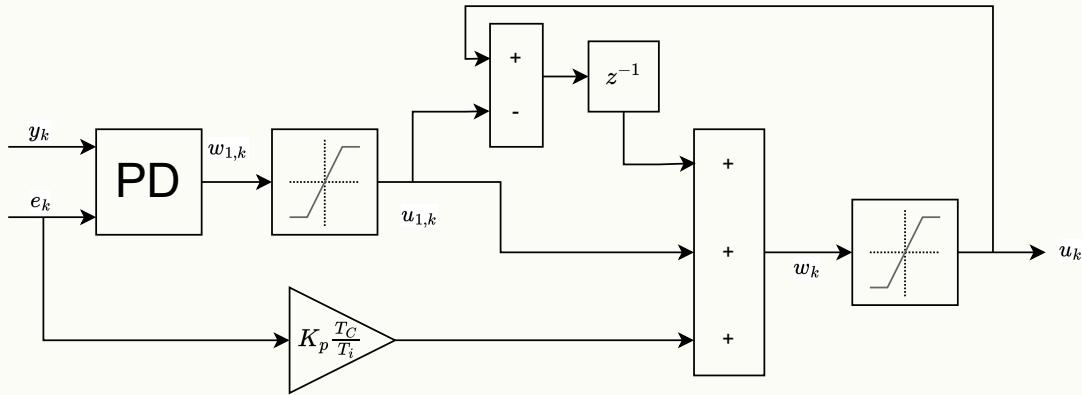


Figura 6.2: Schema digitale

Riproduce l'azione PID classica, l'uscita dell'intero controllo u_k ha i tre contributi. Si riorganizza l'equazione di u_k e si ottiene

$$U_k = U_{1,k} + \frac{K_p T_C}{T_i} \frac{E_k}{1 - z^{-1}}$$

$U_{1,k}$ include l'azione proporzionale e derivativa, e dipendono dal blocco PD, è poi presente l'azione integrale classica. Si ricordi che le lettere maiuscole (U_k piuttosto che u_k) sono usate quando si passa nel dominio \mathcal{Z} .

Consideriamo adesso il caso in cui vi è saturazione, da almeno due campioni (negli istanti k e $k-1$)

$$u_{k-1} = u_H \quad u_{1,k} = u_{1,k-1} = u_H$$

Calcoliamo il valore di w_k in questa situazione

$$w_k = u_H + \frac{K_p T_C}{T_i} e_k + (u_H - u_H) = u_H + \frac{K_p T_C}{T_i} e_k$$

è la somma di 3 contributi

- u_H viene dal campione di controllo al passo precedente
- $\frac{K_p T_C}{T_i} e_k$ azione proporzionale all'errore, rimane invariata
- $(u_H - u_H)$ differenza dal controllo che veniva dal PD

La variabile w_k uscente (che verrà poi saturata dal saturatore posto alla fine dello schema) è uguale al valore saturato più un valore proporzionale all'errore

$$w_k = u_H + \frac{K_p T_C}{T_i} e_k$$

Se $e_k > 0 \implies w_k > u_H \implies u_k = u_H$. Se invece il termine dovesse essere negativo, il termine w_k diminuisce grazie all'errore facendo uscire il sistema dalla saturazione.

$$e_k < 0 \implies w_k < u_H$$

La cosa più importante è che l'uscita di questo termine non accumula più l'errore.



6.5 Tuning del PID

Questa sezione tratterà la sintonizzazione dei guadagni del PID, in particolare, la selezione semi-automatica dei parametri del PID. In un processo da controllare non è detto che sia sempre necessario utilizzare tutte e 3 le azioni del regolatore. Si ricordi che ogni azione del PID ha un determinato effetto sul processo, in particolare

Aumentando l'azione	prontezza di risposta	margine di stabilità	errore a regime
P (aumento di K_p)	aumenta	diminuisce	diminuisce
I (diminuzione di T_i)	diminuisce	diminuisce	azzera
D (aumento di T_d)	diminuisce	migliora	influente

Il termine derivativo aumenta lo smorzamento per ridurre le sovraelognazioni. Esistono due approcci al problema della scelta dei guadagni

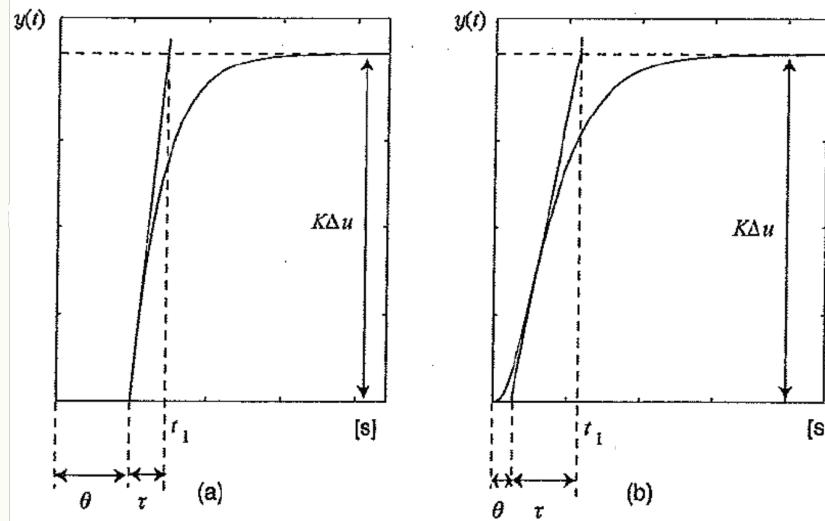
- è noto il modello del processo da controllare, descritto eventualmente da una funzione di trasferimento, le scelte dei guadagni possono essere fatte per assegnazione o cancellazione di poli stabili ma lenti.
- non è noto il modello del processo da controllare, si può sollecitare il processo con degli ingressi standard (a gradino) ed osservare sperimentalmente la risposta del sistema, nell'ipotesi che il sistema di controllo sia stabile asintoticamente. Se il sistema è instabile si può cominciare a chiudere un primo anello proporzionale, valutando la risposta in presenza di azione proporzionale.

6.5.1 Primo Metodo di Ziegler-Nichols

Il processo

$$P_m(s) = \frac{Ke^{-\theta s}}{1 + \tau s}$$

è un modello che verrà utilizzato per il progetto dei guadagni, ricavato da un sistema che ha un polo con una cerca costante di tempo τ , un guadagno K , ed un ritardo di tempo finito rappresentato da $e^{-\theta s}$ nel dominio di Laplace. Il seguente grafico evidenzia tali *parametri caratteristici* del processo nella risposta ad un gradino Δu



La risposta per le prime θ unità di tempo è assente, ciò avviene per il ritardo finito $e^{-\theta}s$, dopo di che risponde con costante di tempo τ , più questa è piccola, più è rapida la risposta verso il valore di regime.

Il valore t_1 rappresenta l'istante in cui l'uscita $y(t)$ raggiunge il 63% del suo valore di regime.

Il modello Z-N prevede la selezione dei guadagni su delle considerazioni fatte su un processo $P_m(s)$ di questo genere. Ziegler e Nichols hanno fatto degli studi su questa classe di processi per cercare di trovare i valori migliori per i parametri del PID, minimizzando lo sforzo di controllo.

Una volta valutati questi 3 parametri (θ, τ, t_1) facendo esperimenti ed osservazioni sul sistema (che era stabile asintoticamente, oppure instabile ma reso stabile), è possibile assegnare i valori dei coefficienti valutando la seguente tabella, che presenta il metodo Z-N, ed altre varianti sviluppate nel corso del tempo.

1° metodo di	Ziegler-Nichols	Cohen-Coon	3C
P	$K K_p = (\theta/\tau)^{-1}$	$K K_p = (\theta/\tau)^{-1} + 0.333$	$K K_p = 1.208(\theta/\tau)^{-0.956}$
PI	$K K_p = 0.9(\theta/\tau)^{-1}$	$K K_p = 0.9(\theta/\tau)^{-1} + 0.082$	$K K_p = 0.928(\theta/\tau)^{-0.946}$
	$T_i/\tau = 3.33(\theta/\tau)$	$T_i/\tau = \frac{3.33(\theta/\tau)[1+(\theta/\tau)/11]}{1+2.2(\theta/\tau)}$	$T_i/\tau = 0.928(\theta/\tau)^{0.583}$
PID	$K K_p = 1.2(\theta/\tau)^{-1}$	$K K_p = 1.35(\theta/\tau)^{-1} + 0.27$	$K K_p = 1.37(\theta/\tau)^{-0.95}$
	$T_i/\tau = 2(\theta/\tau)$	$T_i/\tau = \frac{2.5(\theta/\tau)[1+(\theta/\tau)/5]}{1+0.6(\theta/\tau)}$	$T_i/\tau = 0.74(\theta/\tau)^{0.738}$
	$T_d/\tau = 0.5(\theta/\tau)$	$T_d/\tau = \frac{0.37(\theta/\tau)}{1+0.2(\theta/\tau)}$	$T_d/\tau = 0.365(\theta/\tau)^{0.95}$

Questa tabella presenta un modo univoco per la scelta dei guadagni. La tabella è relativa ad un PID analogico, e alle sue varianti più semplici in assenza di azione derivativa e/o integrale.

Questi valori possono essere utilizzati anche per un PID digitale, dato il tempo di campionamento T_C , nel metodo Z-N basta sostituire la variabile θ con

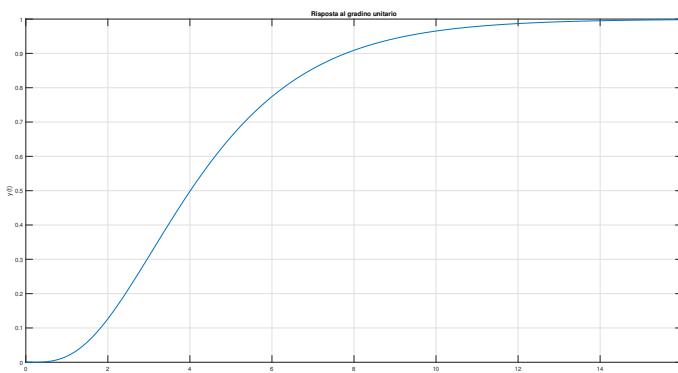
$$\theta^D = \theta + \frac{T_C}{2}$$

6.5.2 Esempio Numerico 1

Si consideri il seguente processo da controllare (attenzione, non è detto che il modello matematico sia perfettamente noto).

$$P(s) = \frac{1}{(1+0.5s)(1+s)^2(1+2s)}$$

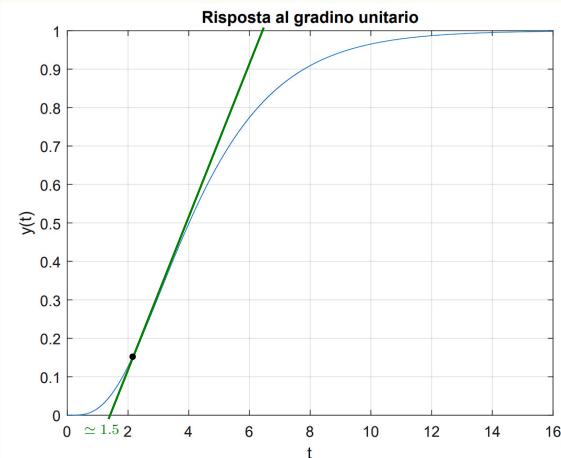
Supponiamo che la funzione $P(s)$ non sia nota al progettista, è possibile solamente valutare la risposta al gradino (unitario), che è la seguente:



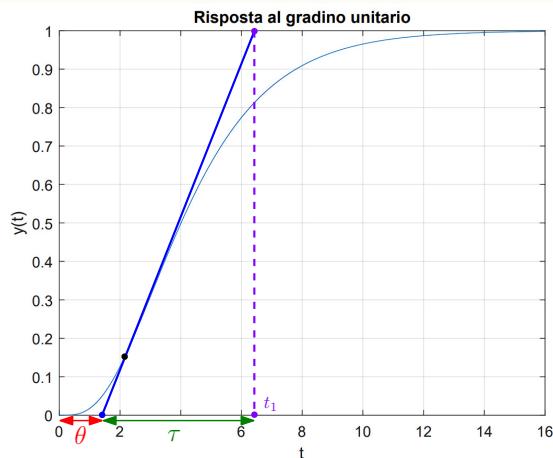
Come già detto, supponendo di non conoscere $P(s)$, si ricavano i 3 parametri caratteristici del sistema, essendo l'ingresso unitario, e l'uscita pure, è chiaro che il guadagno sia a sua volta unitario

$$K = 1$$

Si considera una retta tangente nel punto in cui la crescita è massima, tale retta interseca sia l'asse delle ascisse, che l'asse dato dal valore a regime (in questo caso 1).



Si noti come la tangente interseca l'asse delle ascisse in $t \simeq 1.5$, per $t < 1.5$ il valore dell'uscita è inferiore al 5% del valore di regime. Il punto in cui la tangente interseca l'asse del valore di regime determina l'istante t_1 in cui il valore è circa il 63% del valore a regime. Si trovano i valori τ e θ :



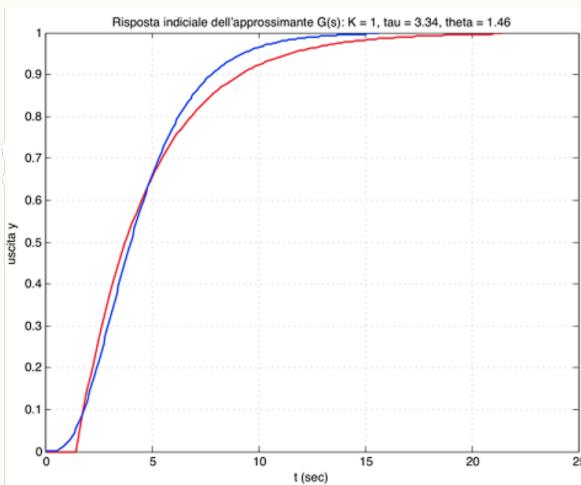
C'è una certa approssimazione in questa scelta. In particolare

$$\theta \simeq 1.46 \text{ secondi} \quad \tau \simeq 3.34 \text{ secondi}$$

Il processo modellato è

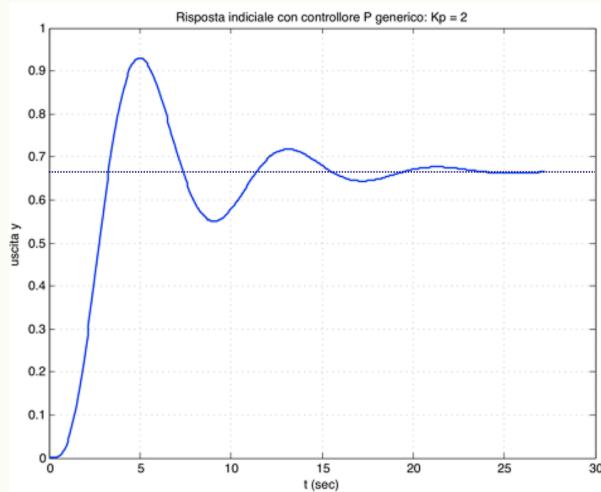
$$P_m(s) = \frac{e^{-1.46s}}{1 + 3.34s}$$

La risposta indiciale di $P_m(s)$ si può visualizzare (curva in rosso) sovrapposta alla risposta del processo (non noto) $P(s)$ (curva in blu):



I tre parametri per l'utilizzo del metodo Z-N sono pronti, *prima di ciò* però, proviamo a realizzare un controllore PID *per tentativi*, ignorando il metodo Z-N che verrà applicato in seguito.

Proviamo con un controllore solo proporzionale con parametro $K_p = 2$, la risposta al gradino unitario è la seguente



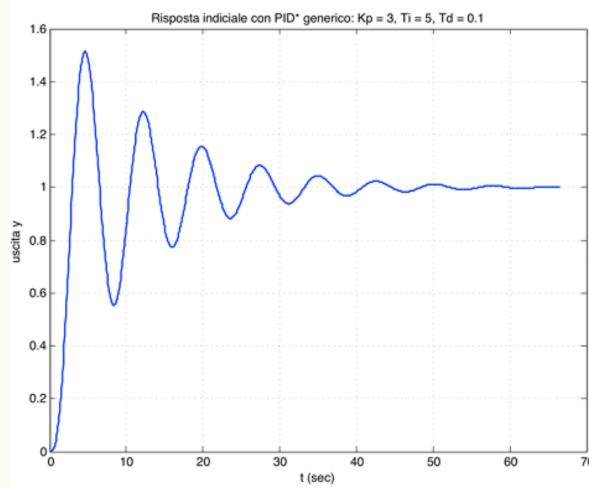
ci sono svariati problemi

- l'uscita non è quella desiderata, errore a regime del circa 33%
- tempo di salita più rapido (circa 3 secondi)
- sovraelognazione evidente, circa il 35%
- se proviamo ad aumentare il guadagno, le oscillazioni crescono

Un anello chiuso con controllore solo proporzionale quindi non va bene

- per ridurre l'errore a regime si aggiunge un azione integrale
- per ridurre l'eccessiva sovraelognazione si aggiunge un azione derivativa (filtrata in banda, e valutata solo sull'uscita misurata)

La risposta con un PID_y^* , con parametri $K_p = 3$, $T_i = 5$, $T_d = 0.1$ è la seguente



Con tale azione l'errore a regime è nullo, ma ci sono dei problemi

- spesso instabile quando si tentano guadagni differenti
- sovraelognazione eccessiva, oscillazioni notevoli (poco smorzamento)

- alto tempo di assestamento, circa 40/50 secondi

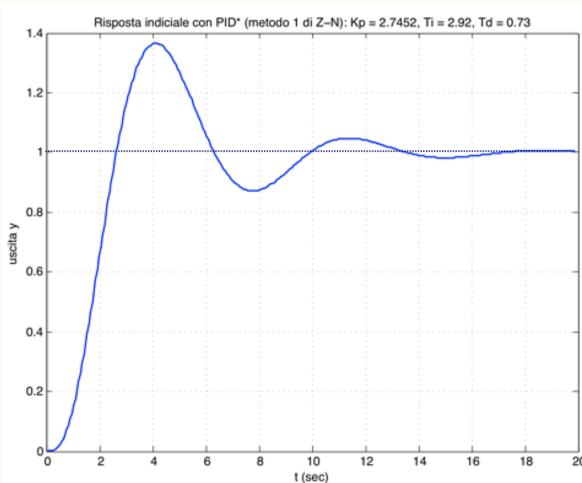
A questo punto, si prova a considerare un PID regolato dal metodo Z-N, si usa un PID^* con derivata filtrata in banda

$$\begin{aligned}\theta &= 1.46 & K_p &= 2.75 \\ \tau &= 3.34 \implies T_i &= 2.92 \\ K &= 1 & T_d &= 0.73\end{aligned}$$

La derivata è filtrata in banda con un valore $N = 5$, ne consegue che la funzione di trasferimento del PID è

$$PID^*(s) = K_p \left(1 + \frac{1}{T_i s} + \frac{T_d s}{1 + \frac{T_d}{N} s} \right) = 2.75 \left(1 + \frac{1}{2.92 s} + \frac{0.73 s}{1 + \frac{0.73}{5} s} \right)$$

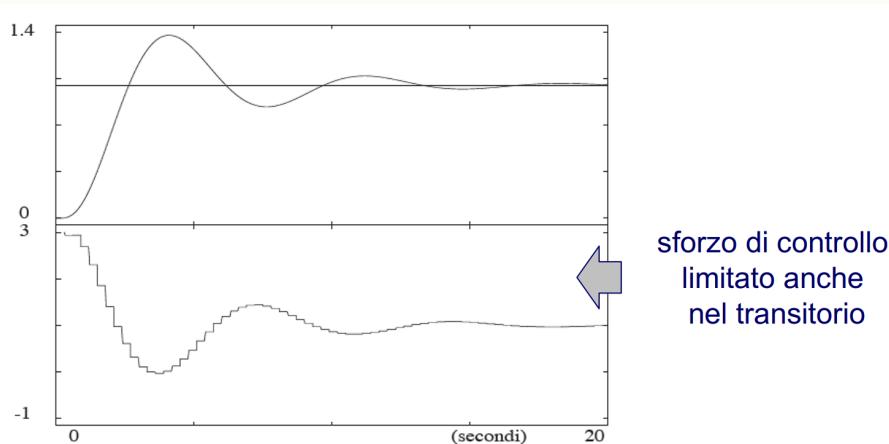
il risultato è il seguente



Il risultato è certamente migliore dei tentativi precedenti

- c'è un rapido tempo di salita, circa 2.5 secondi
- ovviamente l'errore a regime è nullo per la presenza del termine integrale
- le oscillazioni sono poche
- il tempo di assestamento è di circa 16 secondi
- la sovraeolognazione è comunque elevata, circa il 35% del valore di regime, si può ridurre aumentando l'azione derivativa

Consideriamo una versione digitale dello stesso regolatore, con tempo di campionamento $T_C = 0.3$ secondi, si deve valutare la tabella Z-N variando θ , considerando $\theta^D = \theta + \frac{T_C}{2} = 1.61 \implies \frac{\theta^D}{\tau} = 0.482$



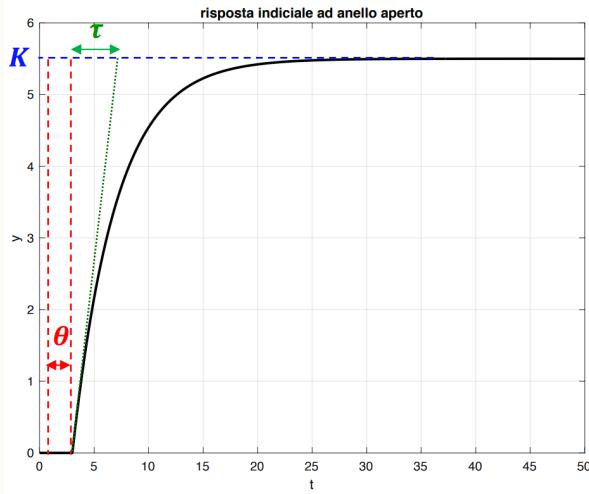
I parametri del PID sono leggermente diversi

$$K_p = 2.4894, \quad T_i = 3.22, \quad T_d = 0.805$$

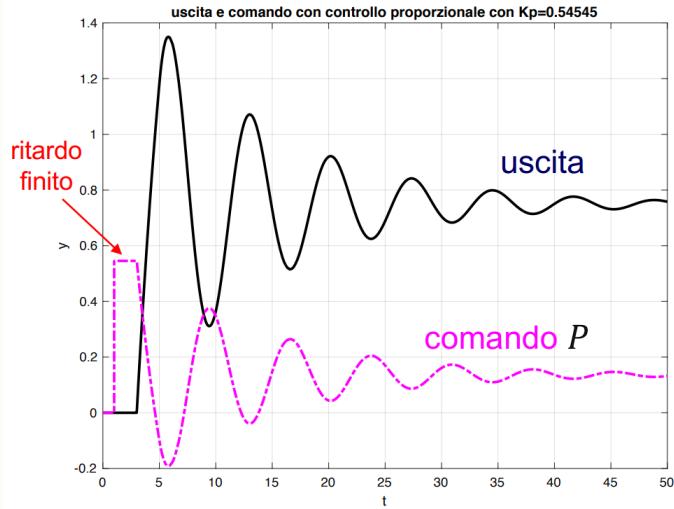
La risposta indiciale è praticamente identica al caso del controllore analogico, l'uscita del controllore ha un organo di tenuta zero, si noti come lo sforzo di controllo è ridotto, ha un picco nell'istante iniziale circa pari a (3 volte il valore di regime).

6.5.3 Esempio Numerico 2

Consideriamo un altro esempio in cui il processo ha effettivamente un ritardo finito. Senza conoscere il modello del processo, sappiamo che la risposta al gradino unitario, che parte all'istante $t = 1$, è la seguente



Proviamo con un tentativo, applicando un'azione proporzionale con costante $K_p = 0.54$



C'è un ritardo intrinseco nell'uscita, il valore a regime è circa 0.75, meglio del caso senza controllore in cui era circa 5.5, ma comunque distante dal valore desiderato 1. Utilizziamo il metodo Z-N, i valori caratteristici che si trovano analizzando il grafico della risposta indiciale sono i seguenti

$$K = 5.5, \quad \theta = 2, \quad \tau = 4$$

il modello è

$$P_m(s) = \frac{5.5e^{-2s}}{1 + 4s}$$

Si applica la derivata filtrata in banda con valore $N = 20$. Si considerano 4 diversi regolatori, i cui parametri si ottengono valutando la tabella di Ziegler-Nichols

- regolatore solo proporzionale P

$$K_p = 0.3636$$

- regolatore proporzionale integrale PI

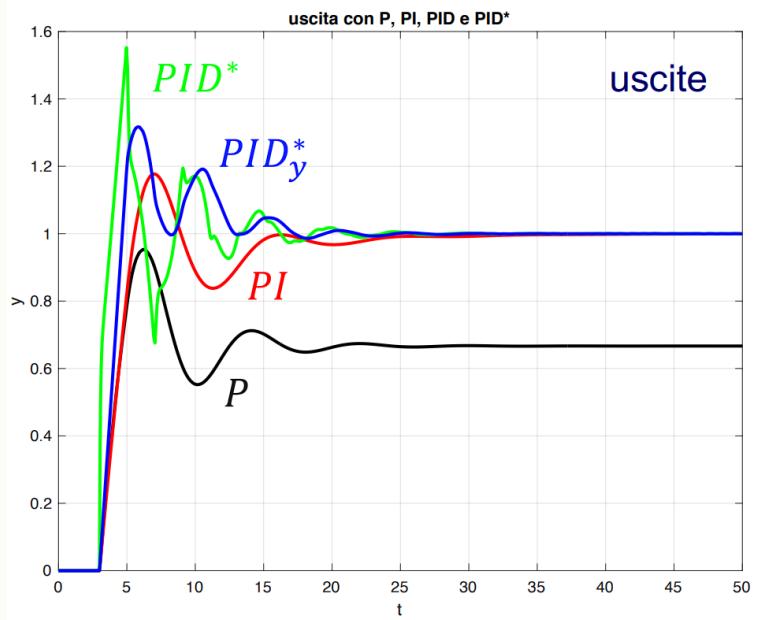
$$K_p = 0.3273, \quad T_i = 6.66$$

- regolatore PID^*

$$K_p = 0.4364, \quad T_i = 4, \quad T_d = 1$$

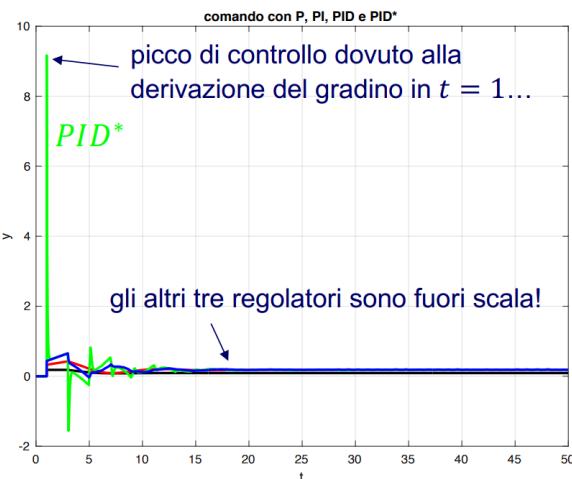
- regolatore PID_y^* con azione derivativa valutata solo sull'uscita misurata (parametri identici al caso precedente)

$$K_p = 0.4364, \quad T_i = 4, \quad T_d = 1$$

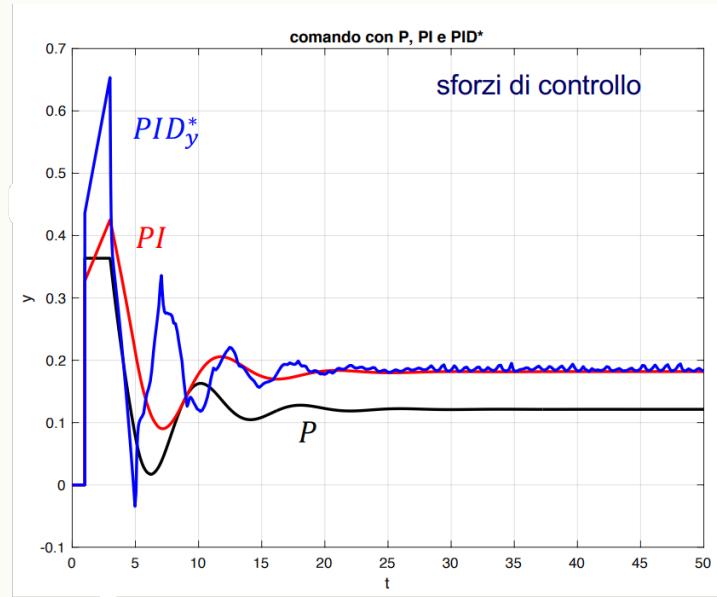


Il termine integrale è necessario per annullare l'errore a regime. Le risposte sono leggermente diverse fra i vari regolatori. Il regolatore PID^* ha un picco dovuto alla variazione istantanea dell'errore, in quanto si applica la derivazione anche su di esso, infatti il regolatore PID_y^* che applica la derivata solo sull'uscita misurata è esente da questo picco.

Valutiamo adesso lo sforzo di controllo, gli andamenti del comando u in uscita dal regolatore sono i seguenti



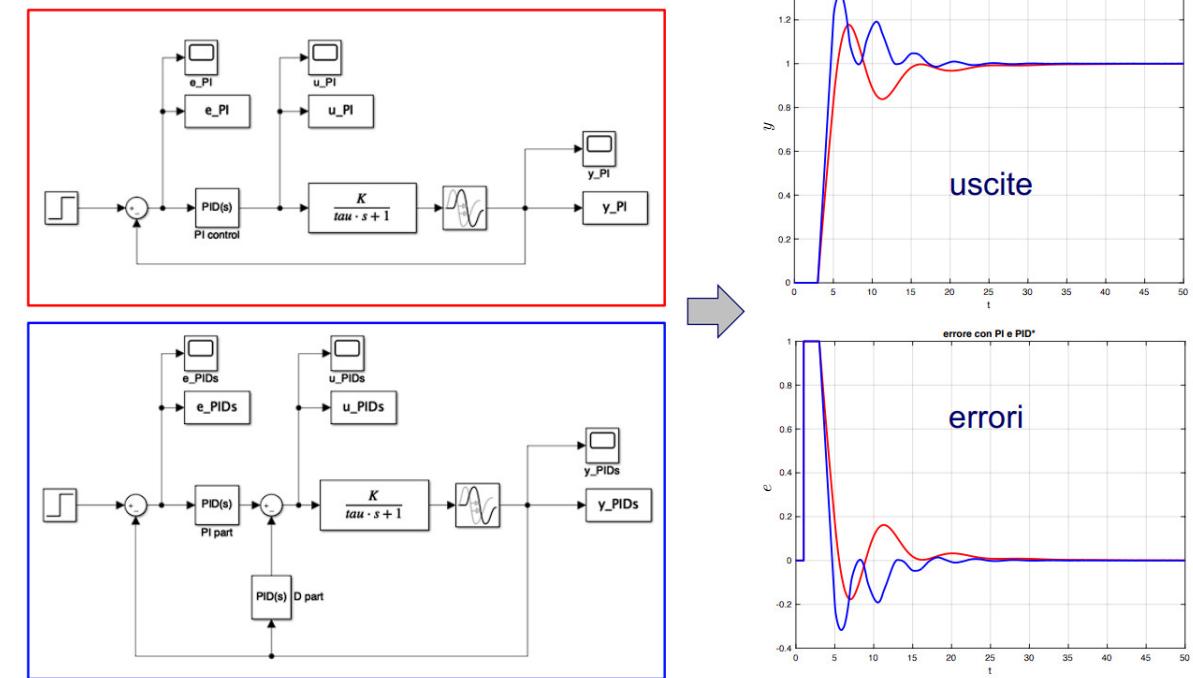
Il PID^* ha uno sforzo di controllo notevole causato dalla derivazione dell'errore, uno sforzo di controllo di questo tipo è inaccettabile e non è possibile adoperare un regolatore di questo tipo, che potrebbe rovinare gli attuatori con un comando eccessivamente alto. Consideriamo quindi lo sforzo di controllo degli altri regolatori escludendo quello del PID^* , che non permette di visualizzare correttamente gli altri andamenti nel grafico.



Il termine proporzionale ha uno sforzo di controllo ridotto. Gli altri regolatori hanno degli sforzi analoghi, data la presenza di $N = 20$ elevato, a regime il controllore PID_y^* con termine derivativo mantiene una sollecitazione permaente, diversamente dal PI che è costante.

Nell'intervallo in cui il controllore proporzionale rimane costante, i due controllori con azione integrale aumentano il comando in maniera proporzionale al guadagno (stanno integrando). In ultima analisi, i migliori regolatori sono PI e PID_y^* .

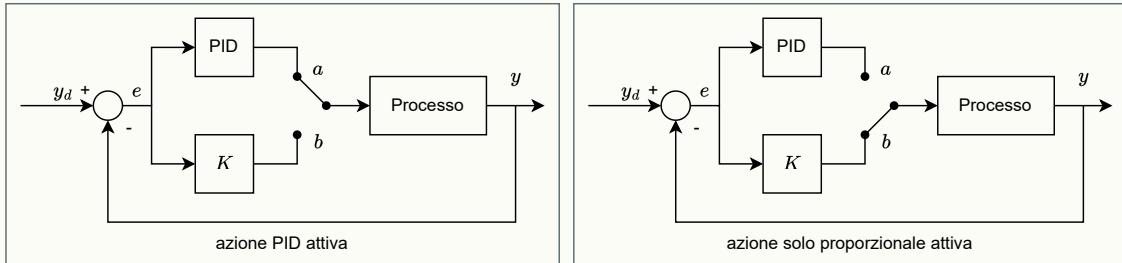
Schemi Simulink dei due migliori regolatori: PI e PID_y^*



Il blocco simulink a valle con disegnata una sinusode rappresenta l'intervalllo finito.

6.5.4 Secondo Metodo di Ziegler-Nichols

Esiste un'altra metodologia per il tuning, è un metodo che lavora ad anello chiuso. Si considera un anello di controllo chiuso con un processo da controllare, un PID (da progettare), ed un azione proporzionale che aumenteremo progressivamente, queste due sono mutualmente esclusive, regolate da uno switch che può chiudere il circuito su un nodo *a* o su un nodo *b*, come mostrato nell'immagine seguente:



Il procedimento è il seguente

1. si chiude l'anello sul nodo *b*, quindi con il solo guadagno proporzionale
2. si parte da un valore minimo K (molto basso) e si aumenta gradualmente fino ad un *valore critico* K_c che prota il sistema in oscillazione
3. Si considera il periodo delle oscillazioni P_c

Una volta ottenuti K_c e P_c si scelgono i parametri del PID secondo la seguente tabella

Tipo	K_p	T_i	T_d
P	$0.5K_c$		
PI	$0.45K_c$	$1.2^{-1}P_c$	
PID	$0.6K_c$	$0.5P_c$	$0.125P_c$
PD	$0.85K_c$		$0.125P_c$

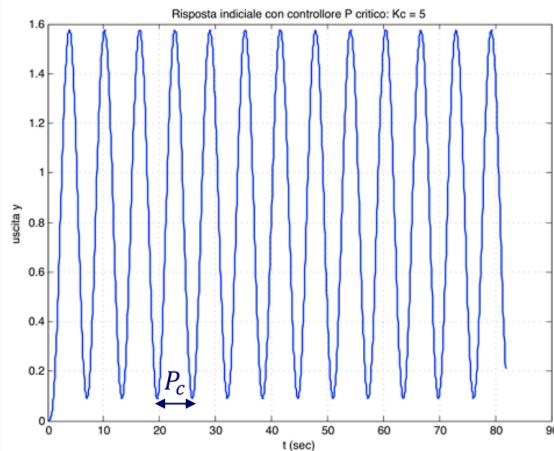
Una volta scelti i guadagni del PID, si chiude il circuito sul nodo *a*, applicando l'azione del PID.

6.5.5 Esempio Numerico 3

Vediamo un esempio di applicazione del secondo metodo, il processo in questione (la cui funzione non è nota al progettista) è lo stesso degli esempi precedenti

$$P(s) = \frac{1}{(1 + 0.5s)(1 + s)^2(1 + 2s)}$$

Si aumenta il guadagno proporzionale fino ad ottenere il guadagno critico $K_c = 5$



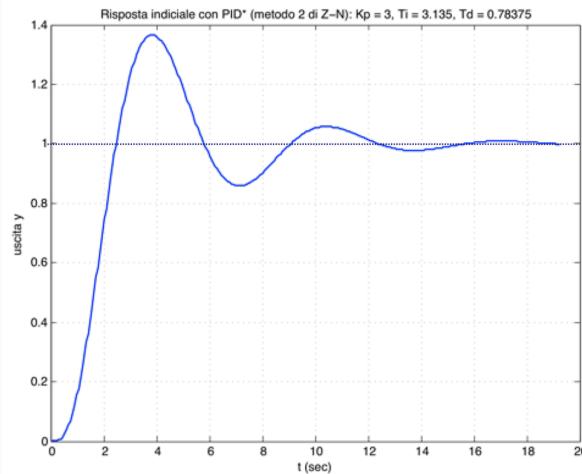
L'oscillazione è permanentemente e si è ai limiti della stabilità. Il periodo è $P_c = 6.27$ secondi. Valutando la tabella del secondo metodo, i parametri del PID sono

$$K_p = 3, \quad T_i = 3.13, \quad T_d = 0.78$$

Per la derivata filtrata in banda si sceglie $N = 5$. Il controllore è

$$PID^*(s) = K_p \left(1 + \frac{1}{T_i s} + \frac{T_d s}{1 + \frac{T_d}{N} s} \right) = 3 \left(1 + \frac{1}{3.13 s} + \frac{0.78 s}{1 + \frac{0.78}{5} s} \right)$$

La risposta del sistema ad anello chiuso con il regolatore PID^* è la seguente



Il risultato è molto simile al caso precedente della sezione 6.5.2 in cui si è applicato il primo metodo.

CAPITOLO

7

SISTEMI A EVENTI DISCRETI

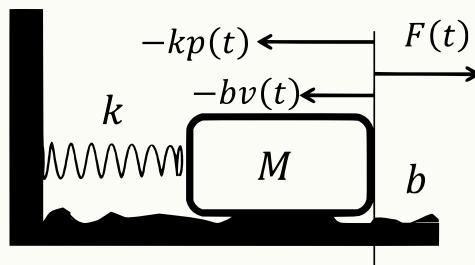
7.1 Automi a Stati Finiti

Definizione : Un **sistema** è un insieme di componenti cooperanti ed interagenti, fisiche o funzionali, che realizzano una funzionalità al fine di raggiungere un determinato obiettivo. Possono essere sistemi fisici oppure applicabili ad altri ambiti (economia, comportamento umano).

Tramite modelli matematici è possibile descrivere e definire sistemi complessi, sulla quale fare previsioni e simulazioni. Si possono anche capire alcune proprietà intrinseche del sistema studiandone il modello. Si identificano due classi di sistemi

- sistemi che evolvono nel tempo, modellati da equazioni differenziali o equazioni alle differenze
- sistemi che evolvono in corrispondenza di determinati eventi, in maniera asincrona

Nei sistemi guidati dal tempo, si definiscono relazioni dinamiche/algebriche fra le variabili in ingresso, quelle di stato, e quelle di uscita, si consideri il seguente esempio:



vi è una massa M , attaccata ad una molla con costante elastica k , soggetta all'attrito con la superficie scabra, dove b è la costante di attrito. F è la forza esterna (ingresso del sistema) applicato alla massa. Siano v e p la velocità e la posizione del corpo, il sistema è descritto dall'equazione differenziale

$$F(t) - kp(t) - bv(t) = Ma(t)$$

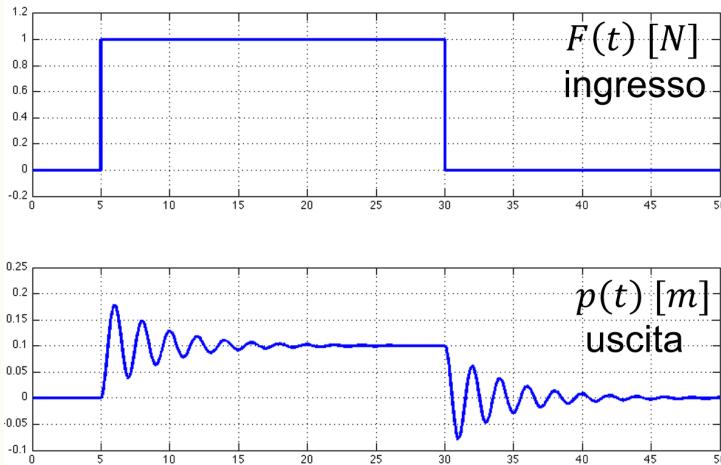
il vettore di stato è identificato dalla posizione e dalla velocità del corpo, l'uscita è solamente la posizione.

$$\begin{aligned} u(t) &= F(t) \\ x(t) &= [p(t) \ v(t)]^T \\ y(t) &= p(t) \end{aligned}$$

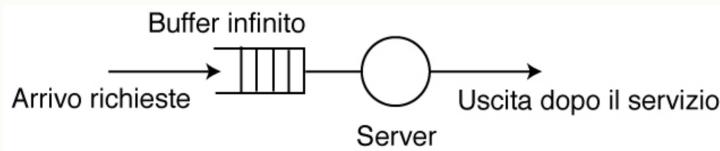
Il processo è modellato dal seguente sistema dinamico

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -kM & -\frac{b}{M} \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = [1 \ 0] x(t)$$



Un esempio di modello discreto invece, riguarda un sistema client/server in cui vi è una coda in cui arrivano delle richieste da servire



Molti sistemi ingegneristici, come

- sistemi manifatturieri
- sistemi di comunicazione
- sistemi di controllo del traffico
- ...

non possono essere descritti in maniera adeguata da modelli temporali, si considerano allora i **Discrete Event Dynamic Systems (DEDS)**, in cui

- le grandezze di interesse assumono valori in un insieme finito
- lo stato viene modificato in maniera asincrona, al verificarsi di opportuni eventi

Si consideri l'esempio delle richieste in coda, le grandezze di interesse sono

- il numero (naturale) di richieste nel buffer
- lo stato del server (I : in attesa, B : occupato)

Lo stato complessivo del sistema sarà quindi un valore in

$$X = \{I, B\} \times \mathbb{N}$$

Gli eventi che modificano lo stato del sistema sono

- arrivo di una richiesta, evento a
- presa in carico di una richiesta nel server, evento b
- completamento di una richiesta, evento c

Altri esempi di sistemi a eventi discreti possono essere

Magazzino di prodotti

Lo stato X è descritto dal numero (naturale) di prodotti nel magazzino

$$X = \mathbb{N}$$

Gli eventi sono

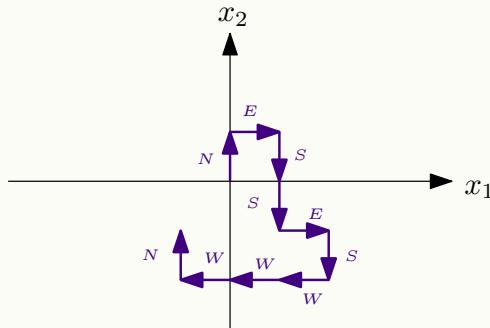
- arriva un prodotto in magazzino
- ritirano un prodotto dal magazzino

Passeggiata sul piano

Una passeggiata (insieme di passi unitari) su un piano discreto ha come stato X la posizione corrente del sistema (x_1, x_2)

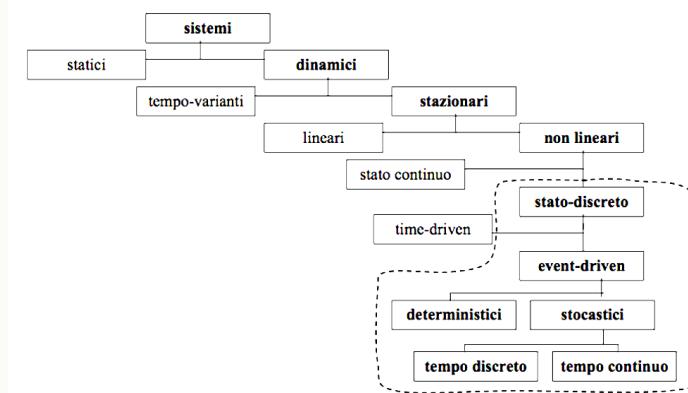
$$X = \mathbb{Z} \times \mathbb{Z}$$

Gli eventi sono



In questo esempio, gli eventi commutano, ciò non è sempre verificato. Il tempo può anche essere considerato nel modello, ma non deve essere rilevante per l'evoluzione del sistema, spesso si è interessati ad informazioni temporali per valutare le prestazioni del sistema.

I sistemi possono essere suddivisi nella seguente tassonomia (quelli tratteggiati, sono DEDS)



Dato un DEDS ed una **sequenza di eventi**, si identifica un'evoluzione univoca dello stato del sistema.

Definizione (Automa) : : Un'automa è una 5-tupla, (X, E, f, x_0, X_m) di cui

- X è l'insieme degli stati possibili
- E è l'alfabeto che compone le stringhe in input
- f è una mappa $X \times E \rightarrow X$ detta *funzione di transizione*.
- $x_0 \in X$ è lo stato iniziale.

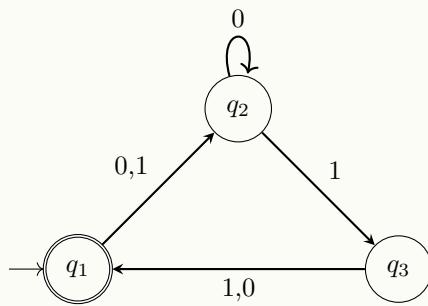


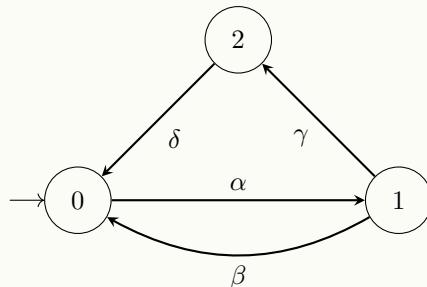
Figura 7.1: semplice automa

- $X_m \subseteq X$ è l'insieme degli stati di accettazione.

Un automa può avere **ingressi e uscite**, in particolare, la definizione è simile, ma si aggiunge una *funzione di uscita*. Tale automa è una 6-tupla (U, X, Y, f, h, x_0) dove

- U è l'insieme degli eventi in ingresso del sistema
- X è l'insieme degli stati possibili
- Y è l'insieme degli eventi in uscita del sistema
- f è una mappa $X \times U \rightarrow X$ detta *funzione di transizione*.
- $x_0 \in X$ è lo stato iniziale.
- h è una mappa detta funzione di uscita, e può essere definita in due modi
 - $h : X \times U \rightarrow Y$ **modello di Melay**
 - $h : X \rightarrow Y$ **modello di Moore**

Si consideri il seguente esempio, in cui un automa modella una macchina soggetta a dei guasti:



Gli stati sono

- 0 : la macchina è libera
- 1 : la macchina è occupata
- 2 : la macchina è guasta

gli eventi sono

- α : avvio lavorazione
- β : fine lavorazione
- γ : la macchina si guasta
- δ : la macchina viene riparata

Questo automa fa delle assunzioni implicite

- la macchina si può guastare solamente durante la lavorazione
- quando una macchina si guasta il prodotto in lavorazione viene scartato

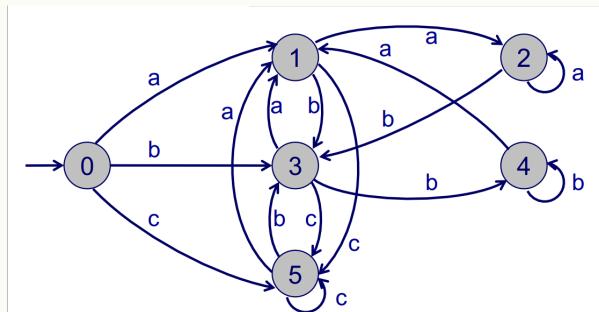
Si consideri ora il seguente esempio, vi è un processo in cui devono essere eseguite 3 operazioni (eventi), a, b e c . L'operazione c presenta un limite tecnologico : se l'ultima esecuzione è identica alla penultima, la c non potrà essere eseguita

- $a - b - c - a - b$ sequenza ammissibile
- $a - b - b - a - a - c$ sequenza non ammissibile
- $a - a - b - c - c - b - b - a$ sequenza ammissibile

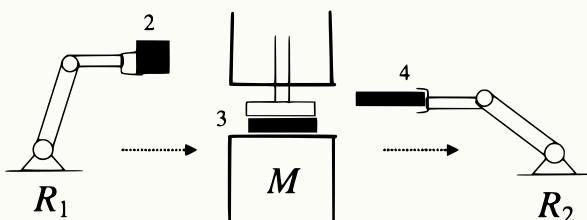
Si identificano i seguenti stati:

- 0 - nessuna operazione eseguita
- 1 - l'ultima eseguita è a , e la penultima non è a
- 2 - le ultime due operazioni sono di tipo a
- 3 - l'ultima eseguita è b , e la penultima non è b
- 4 - le ultime due operazioni sono di tipo b
- 5 - l'ultima operazione è di tipo c

Il sistema è descritto dal seguente automa:



Si consideri un'ultimo esempio, vi è un sistema composto da 2 robot R_1, R_2 ed una pressa M . Il robot R_1 può caricare dei pezzi sulla pressa M , questa, una volta che un pezzo è stato pressato, il robot R_2 può prelevarlo da M .



Gli stati sono i seguenti

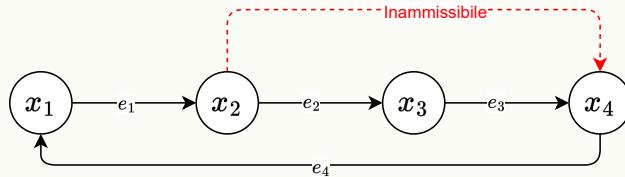
- x_1 la macchina M è in attesa e disponibile
- x_2 il pezzo viene caricato su M da parte di R_1
- x_3 la macchina esegue la pressa
- x_4 il pezzo viene scaricato da M da parte di R_2

Gli eventi identificati sono i seguenti

- e_1 carico del pezzo
- e_2 inizio lavorazione

- e_3 inizio scarico
- e_4 fine lavorazione, ritorno in attesa

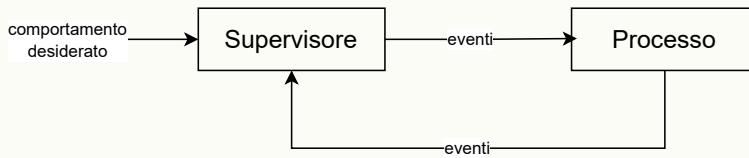
è chiaro che si vogliono evitare alcune sequenze di eventi, ad esempio, si vuole impedire che dallo stato x_2 si passi allo stato x_4 (il pezzo viene messo e poi tolto senza alcuna lavorazione).



7.1.1 Controllo dei DEDS

Come i sistemi lineari guidati dal tempo possono avere un regolatore (ad es. PID) anche i sistemi ad eventi discreti possono essere controllati, in modo da ottenere il comportamento desiderato, è possibile procedere in due modi

- si implementa direttamente il comportamento desiderato nel modello
- si modella in maniera modulare il processo da controllare, per poi definire un controllore (a sua volta un DEDS) che definisce gli eventi da scaturire nel sistema da controllare per avere il comportamento desiderato



sono necessari

- il processo da controllare (DEDS)
- le specifiche sul comportamento desiderato, espresse a loro volta tramite un DEDS, che evolve in modo da non violare i vincoli
- un supervisore guidato da eventi in ingresso, che restituisce in uscita degli eventi/comandi per il processo
- strumenti di progetto che permettano di combinare/integrare con relativa facilità diversi sottosistemi DEDS e di verificarne il comportamento

Interazione Supervisore-Calcolatore

Il supervisore è un sistema a eventi discreti (DEDS) che evolve in modo asincrono, ovvero non segue un ritmo costante. Interagisce con il processo da controllare tramite eventi (azioni che avvengono in momenti imprevedibili).

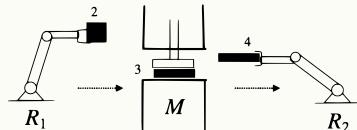
Il calcolatore è un sistema sincrono, ovvero segue un ritmo costante (clock). Interagisce con il processo tramite segnali (valori che cambiano in modo periodico). Il problema principale è che il supervisore e il calcolatore hanno modi diversi di funzionare. Per farli comunicare è necessario:

- Traduzione: trasformare gli eventi in segnali e viceversa.
- Sincronizzazione: coordinare l'evoluzione asincrona del supervisore con il ritmo sincrono del calcolatore.

Per risolvere questo problema, si usa un approccio "time-driven", ovvero si forza il supervisore a funzionare in modo più simile al calcolatore. Questo approccio presenta però delle sfide:

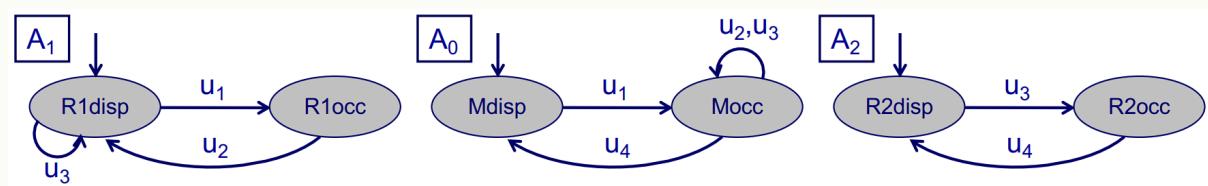
- Perdita di eventi: se un evento accade troppo vicino ad un altro, potrebbe non essere rilevato.
- Ritardi: le azioni di controllo potrebbero non essere eseguite immediatamente, causando ritardi nel sistema.
- Eventi simultanei: se più eventi accadono nello stesso momento, il loro ordine potrebbe non essere rilevante per la decisione da prendere.

Vediamo un esempio di automa con controllore, consideriamo il sistema di due robot ed una pressa appena visto, si cerca di descrivere il processo nel modo più "vanilla" possibile, senza imporre alcun comportamento logico, in particolare, si definiscono 3 automi A_1, A_0, A_2 , ognuno per ogni macchina presente (R_1, M, R_2).



Questi automi descrivono il comportamento fisico delle macchine, ogni componente può essere *occupata* o *disponibile*, ognuna di queste avrà quindi due stati, il processo totale definito da tutte le componenti avrà quindi $2^3 = 8$ possibili stati (combinazioni dei singoli stati), anche se alcune di queste non sono ammissibili. Gli eventi sono

- u_1 inizio ciclo di lavorazione
- u_2 fine del carico sulla macchina
- u_3 fine della lavorazione
- u_4 fine dello scarico della macchina



Definiamo adesso il controllore, questo avrà gli stati codificati con una stringa composta da 3 caratteri, ad esempio

ddo

dove

- il primo carattere identifica lo stato di R_1
- il secondo carattere identifica lo stato di M
- il terzo carattere identifica lo stato di R_2

Ogni carattere può essere uno fra

- *d* : disponibile
- *o* : occupato

Ad esempio, lo stato *ddo* indica

- R_1 disponibile
- M disponibile
- R_2 occupato (sta scaricando il pezzo)

chiaramente degli 8 possibili stati, alcuni non devono essere contemplati dal sistema, ad esempio

- *ooo* : tutte e 3 le macchine occupate, non può succedere che sia R_1 che R_2 siano occupate
- *odo* : inammisibile per lo stesso motivo della precedente

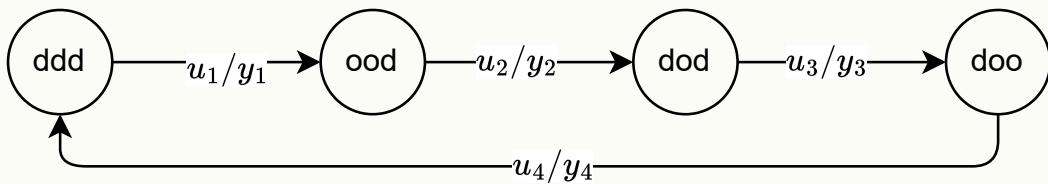
Inoltre, alcuni stati sono "transitori" e non interessa che questi siano inclusi nel supervisore, ad esempio

- *odd* : la prima macchina è occupata (sta caricando il pezzo), senza che nessun evento possa fare nulla, lo stato successivo sarà necessariamente *ood* (la pressa inizia a lavorare una volta caricata)
- *ddo* solamente il robot di scarico è occupato, chiaramente senza che nessun evento possa modificare l'evoluzione, a fine dello scarico anche R_2 tornerà disponibile, e l'evento successivo sarà *ddd*

Il supervisore quindi modellerà il comportamento logico e sarà un automa di Melay, a cui, per ogni evento, corrisponderà un'uscita, che sarà l'evento di ingresso del processo da controllare. Le possibili uscite, quindi ingressi per gli automi A_1, A_0, A_2 sono le seguenti

- y_1 : inizio del carico
- y_2 : inizio lavorazione (pressa)
- y_3 : inizio dello scarico
- y_4 fine del ciclo e ritorno nella fase di attesa

L'automa è il seguente



Si vuole complicare il modello per renderlo più efficiente, in particolare, aggiungendo due ulteriori stati per ogni dispositivo

- pressa M avrà
 - $Mdisp$ la macchina è disponibile
 - $Mcar$ la macchina sta essendo caricata
 - $Mlav$ la macchina sta eseguendo la pressa
 - $Mscar$ si sta scaricando la macchina
- robot R_1 avrà
 - $R1disp$ il robot è disponibile
 - $R1prel$ il robot sta prelevando il pezzo da caricare
 - $R1car$ il robot sta caricando il pezzo
 - $R1rit$ il robot ritorna nella posizione di attesa

lo stato $R1car$ deve essere sincronizzato con lo stato di carico della macchina M

- robot R_2 avrà
 - $R2disp$ il robot è disponibile
 - $R1scar$ il robot sta scaricando il pezzo
 - $R1rit$ il robot ritorna nella posizione di attesa

lo stato $R2scar$ deve essere sincronizzato con lo stato di scarico della macchina M

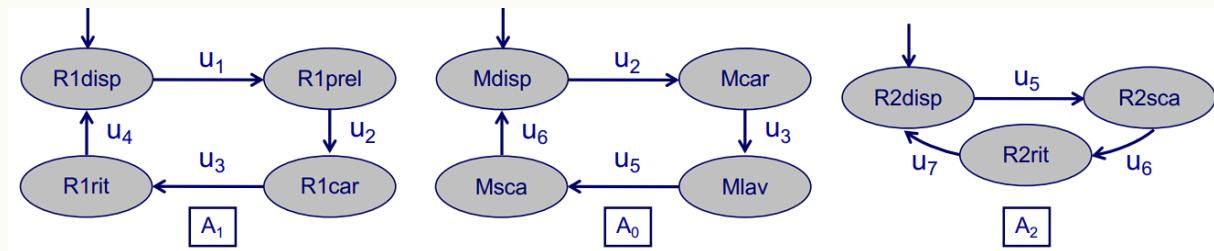
gli eventi e le azioni di controllo del supervisore crescono di conseguenza, gli ingressi sono

- u_1 : pezzo disponibile al prelievo
- u_2 : fine prelievo pezzo con R_1
- u_3 : fine carico di M con R_1
- u_4 : fine ritorno di R_1
- u_5 : fine lavorazione di M
- u_6 : fine scarico di M con R_2
- u_7 : fine ritorno di R_2

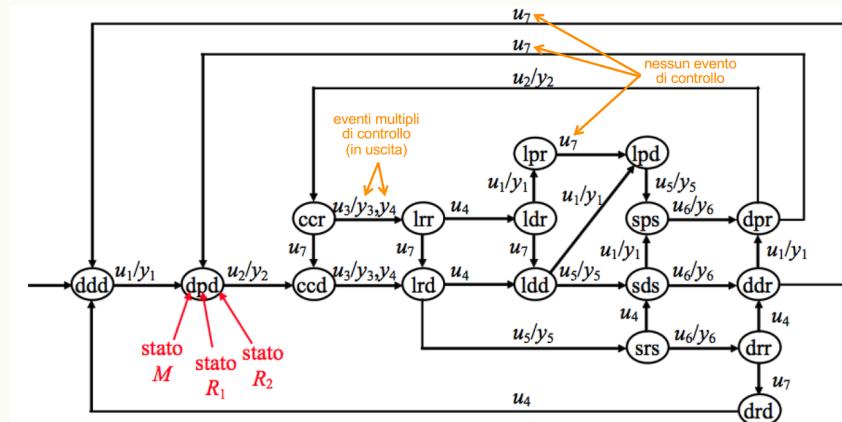
le uscite (eventi di controllo) sono

- y_1 : inizio prelievo pezzo con R_1
- y_2 : inizio carico di M con R_1
- y_3 : inizio ritorno di R_1
- y_4 : inizio lavorazione di M
- y_5 : inizio scarico di M con R_2
- y_6 : inizio ritorno di R_2

i 3 automi che descrivono le singole componenti sono i seguenti



Il modello del supervisore si complica notevolmente, le possibili combinazioni di stati delle componenti ora sono $4 \cdot 4 \cdot 3 = 48$, inoltre non c'è un legame con il modello precedente, ed è necessario riprogettare il supervisore da zero.



Dei 48 possibili stati, solo 17 sono validi da considerare nel supervisore. Seppur semplici gli automi a stati finiti, sono poco flessibili, esiste un modello più sofisticato.



7.2 Reti di Petri

Una rete di Petri, non è altro che un grafo bipartito, in cui ogni nodo appartiene ad un'insieme fra

- nodi *posto*
- nodi *transizione*

Inoltre, i nodi posto possono essere annotati con dei pallini neri, detti *token*, essi rappresentano lo stato del sistema in quanto indicano che delle risorse (in senso generale) sono disponibili in un posto, permettendo eventualmente una transizione. Ogni arco del grafo collega un nodo posto ad un nodo transizione.

Sono un formalismo grafico/matematico per la modellizzazione dei DEDS, e le transazione da uno stato ad un altro richiedono che determinate *condizioni esplicite siano* siano soddisfatte. Lo stato del sistema è un'informazione distribuita fra le entità posto, in particolare, è descritto dalla presenza dei token sui vari nodi.

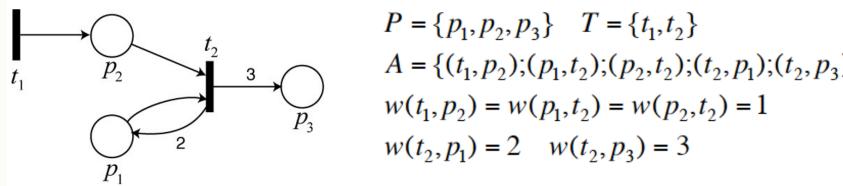
Le reti di Petri rispetto gli automi a stati finiti sono vantaggiose perché

- sono facilmente modificabili
- sono modulari
- sono facilmente interpretabili
- possono rappresentare sistemi ad infiniti stati con un numero finito di nodi

Una rete di Petri si definisce a partire da un **grafo di Petri**, come già detto, bipartito e suddiviso in nodi posto e nodi transizione.

$$PG = (P, T, A, w)$$

- $P = p_1, p_2 \dots p_{|P|}$ nodi posto
- $T = t_1, t_2 \dots t_{|T|}$ nodi transizione
- $A \subseteq P \times T \cup T \times P$ archi orientati che descrivono il flusso del grafo
- $w : A \rightarrow \mathbb{N}$ funzione che associa un peso (naturale) agli archi



Inoltre, per ogni transizione t_j si definiscono due insiemi I e O

- **posti in ingresso** : $I(t_j) = \{p_i \in P \mid (p_i, t_j) \in A\}$
- **posti in uscita** : $O(t_j) = \{p_i \in P \mid (t_j, p_i) \in A\}$

Nell'esempio sopra mostrato

$$\begin{aligned} I(t_1) &= \emptyset & O(t_1) &= \{p_2\} \\ I(t_2) &= \{p_1, p_2\} & O(t_2) &= \{p_1, p_3\} \end{aligned}$$

L'intera topologia del grafo di Petri può essere rappresentata dalle **matrici di ingresso ed uscita**, ossia **I** e **O**, entrambe matrici $|P| \times |T|$ definite come segue

$$\mathbf{I} = \begin{bmatrix} w(p_1, t_1) & w(p_1, t_2) & \dots & w(p_1, t_{|T|}) \\ w(p_2, t_1) & w(p_2, t_2) & \dots & w(p_2, t_{|T|}) \\ \vdots & \vdots & \ddots & \vdots \\ w(p_{|P|}, t_1) & w(p_{|P|}, t_2) & \dots & w(p_{|P|}, t_{|T|}) \end{bmatrix}$$

$$\mathbf{O} = \begin{bmatrix} w(t_1, p_1) & w(t_2, p_1) & \dots & w(t_{|T|}, p_1) \\ w(t_1, p_2) & w(t_2, p_2) & \dots & w(t_{|T|}, p_2) \\ \vdots & \vdots & \ddots & \vdots \\ w(t_1, p_{|P|}) & w(t_2, p_{|P|}) & \dots & w(t_{|T|}, p_{|P|}) \end{bmatrix}$$

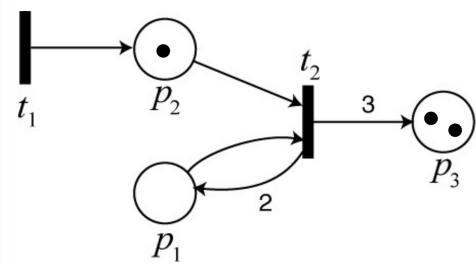
Quando un arco non è presente fra una coppia nodo-transizione, il suo peso è zero. Nell'esempio precedente si hanno le matrici

$$\mathbf{I} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \quad \mathbf{O} = \begin{bmatrix} 0 & 2 \\ 1 & 0 \\ 0 & 3 \end{bmatrix}$$

Ad un grafo di Petri si aggiunge una **funzione di marcatura** x per renderlo una rete di Petri, questa ne descrive lo stato, ed associa ad ogni stato un numero naturale di token.

$$x : P \rightarrow \mathbb{N}$$

ad esempio



$$x(p_1) = 0 \quad x(p_2) = 1 \quad x(p_3) = 2$$

lo stato del sistema è descritto da un **vettore di stato** (vettore colonna) il cui i -esimo elemento è la funzione di marcatura valutata su p_i

$$\mathbf{x} = \begin{bmatrix} x(p_1) \\ x(p_2) \\ \vdots \\ x(p_{|P|}) \end{bmatrix} \in \mathbb{N}^{|P|}$$

Una rete di petri PN è quindi un grafo di petri alla quale si aggiunge il vettore di stato definito dalla funzione di marcatura

$$PN = (P, T, A, w, \mathbf{x})$$

Una rete di Petri si dice *ordinaria* se ogni arco di peso non nullo ha peso 1.

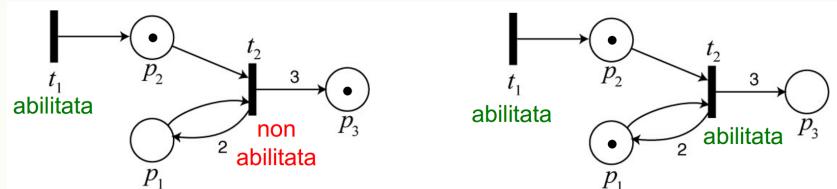
7.2.1 Scatto delle Transizioni

L'evoluzione della rete di Petri è dettata da eventi, un evento accade quando una transizione diviene **abilitata**, a quel punto, si dice che la rete *scatta* ed avviene un cambio di marcatura (il vettore \mathbf{x}) cambia.

Una transizione t_j risulta **abilitata** se

$$\forall p_i \in I(t_j) \text{ si verifica che } x(p_i) \geq w(p_i, t_j)$$

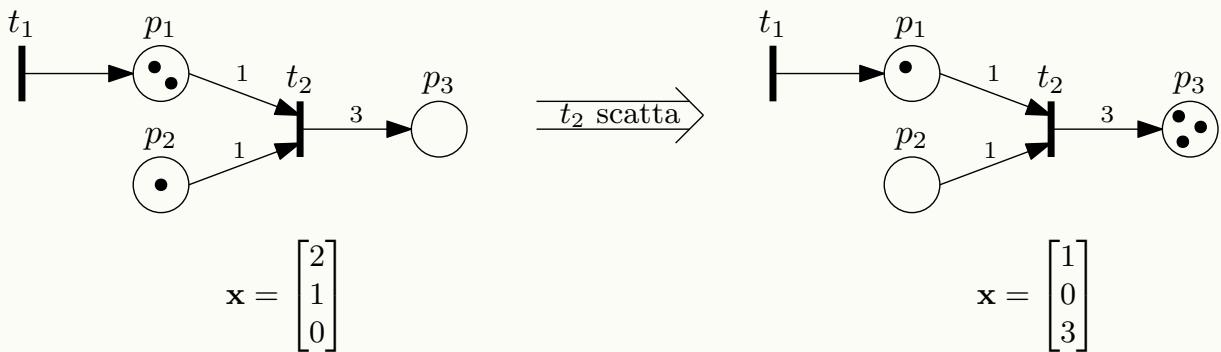
Ossia, tutti i posti entranti nella transazione hanno un numero di token maggiore o uguale al peso dell'arco che li lega a tale transazione.



Bisogna adesso definire come cambiano le marcature sui posti, quando una transizione scatta

- ogni posto perde un numero di token pari al peso dell'arco che lo collega (in entrata) alla transizione scattata

- ogni posto che ha un arco entrante collegato alla transazione scattata, guadagna un numero di token pari al peso dell'arco

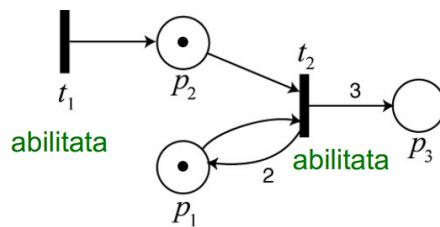


L'aggiornamento del vettore di stato si può definire con una funzione $f : N^{|P|} \times T \rightarrow N^{|P|}$, che ad ogni stato, associa lo stato successivo al verificarsi di una determinata transazione. Sia t_j la transazione che scatta, allora

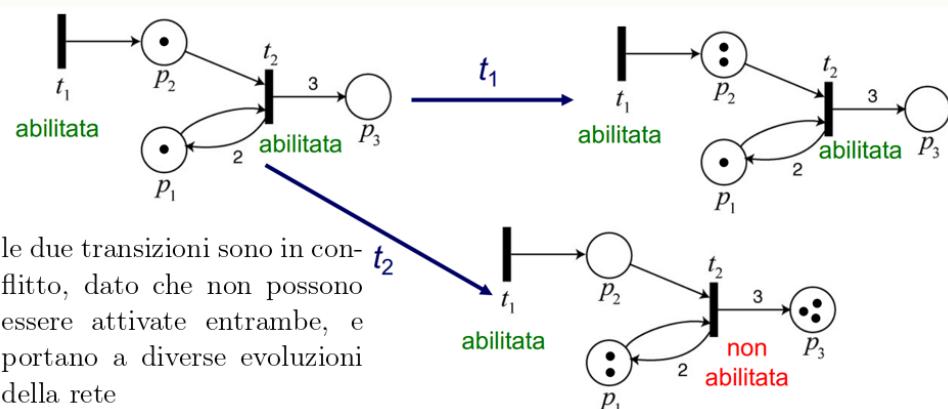
$$f(\mathbf{x}, t_j) = \begin{bmatrix} x(p_1) - w(p_1, t_j) + w(t_j, p_1) \\ x(p_2) - w(p_2, t_j) + w(t_j, p_2) \\ \vdots \\ x(p_{|P|}) - w(p_{|P|}, t_j) + w(t_j, p_{|P|}) \end{bmatrix}$$

è chiaro che con questo meccanismo, un scatto può far sia incrementare che decrementare il numero totale di token nella rete.

Cosa succede se due o più transazioni sono abilitate nello stesso momento?



due transizioni sono in **conflitto** se, entrambe possono scattare singolarmente, ma non ci sono abbastanza risorse per far sì che possano scattare entrambe



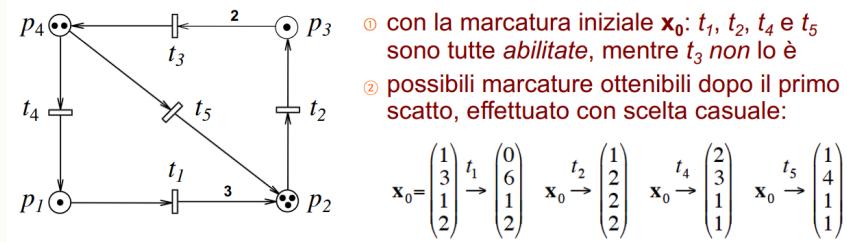
Se due transizioni sono abilitate e non vanno in conflitto (ci sono abbastanza risorse per farne scattare una e poi l'altra), allora si fanno scattare entrambe, se invece c'è un conflitto, come nel caso appena mostrato, si può

- scegliere casualmente

- dare priorità lessicografica
- dare priorità con memoria (si fa scattare quella che non scatta da più tempo)
- si definisce un'altra qualsiasi relazione d'ordine per la priorità

In termini matriciali, una transizione t_j è abilitata se il vettore \mathbf{x} è maggiore o uguale della j -esima colonna della matrice degli ingressi \mathbf{I} .

$$\mathbf{x} \geq \mathbf{I}_j$$



7.2.2 Dinamica

L'evoluzione di una rete di Petri può essere descritta da una sequenza S di transizioni. Per la rete appena mostrata, una sequenza ammissibile è

$$S = t_1 t_4 t_1 t_2 t_2 t_2 t_2 t_2 t_3 t_3$$

Proposizione : se una sequenza di transizioni è ammissibile, allora queste possono essere eseguite in un qualsiasi ordine, e porteranno sempre il sistema nello stesso stato \mathbf{x} finale.

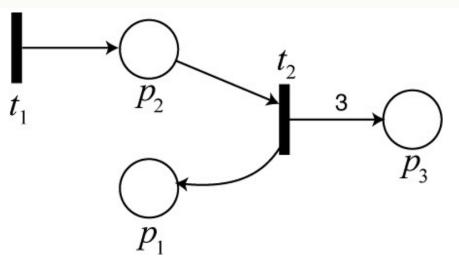
In termini matriciali, la funzione di transizione f è definita come segue

$$f(\mathbf{x}, t_j) = \mathbf{x} - \mathbf{I}_j + \mathbf{O}_j \quad \text{se } \mathbf{x} \geq \mathbf{I}_j$$

Per descrivere la dinamica si definisce quindi la **matrice di incidenza** $\mathbf{C} = \mathbf{O} - \mathbf{I}$ quindi

$$f(\mathbf{x}, t_j) = \mathbf{x} + \mathbf{C}_i \quad \text{se } \mathbf{x} \geq \mathbf{I}_j$$

La rete di petri in figura



ha matrice di incidenza

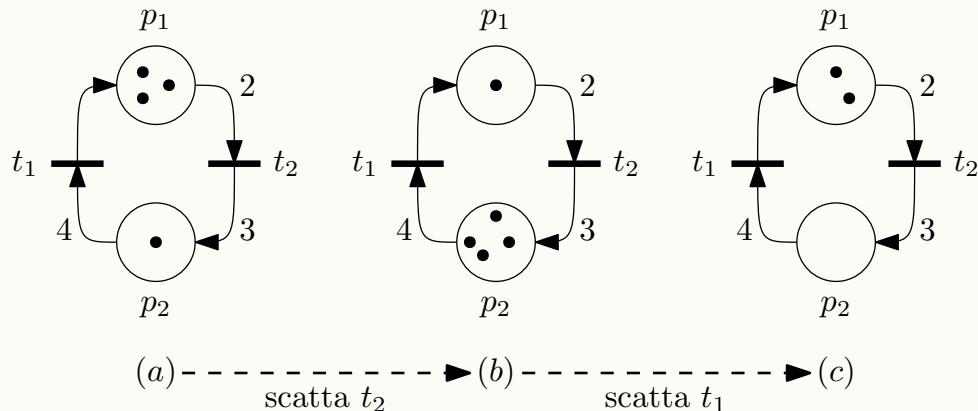
$$\mathbf{C} = \mathbf{O} - \mathbf{I} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 3 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -1 \\ 0 & 3 \end{bmatrix}$$

Osservazione : Le matrici \mathbf{I} e \mathbf{O} caratterizzano totalmente la topologia della rete, la matrice \mathbf{C} no.

Si è parlato di sequenza *ammissibile*, una sequenza di scatti S composta da n transizioni, per essere ammissibile deve verificare che

- la prima transizione della sequenza è abilitata nella marcatura corrente
- lo scatto di ogni transizione porta in una marcatura in cui è abilitata la transizione successiva nella sequenza

ad esempio :



La sequenza è $t_1 t_2$

la matrice di incidenza è

$$\mathbf{C} = \mathbf{O} - \mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} - \begin{bmatrix} 0 & 2 \\ 4 & 0 \end{bmatrix} = \begin{bmatrix} 1 & -2 \\ -4 & 3 \end{bmatrix}$$

passo 0

$$\mathbf{x} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

passo 1

$$\mathbf{x} = \begin{bmatrix} 3 \\ 1 \end{bmatrix} + \begin{bmatrix} -2 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \end{bmatrix}$$

passo 2

$$\mathbf{x} = \begin{bmatrix} 1 \\ 4 \end{bmatrix} + \begin{bmatrix} 1 \\ -4 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

Essendo che le transazioni di una sequenza ammisible commutano, sarebbe stato possibile eseguire $S' = t_1 t_2$ ed il vettore di stato finale al passo 2 sarebbe stato lo stesso. A tal proposito, è possibile data una sequenza S , definire un vettore \mathbf{s} di $|T|$ componenti, detto **vettore delle occorrenze**, in cui l' i -esimo elemento è uguale al numero di occorrenze di t_i nella sequenza, ad esempio

$$S' = t_1 t_2 \quad S = t_2 t_1$$

condividono il vettore delle occorrenze

$$\mathbf{s} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

A tal proposito, lo stato finale dell'evoluzione di una rete di Petri, la cui marcatura iniziale è \mathbf{x} , data una sequenza S con associato un vettore \mathbf{s} , sarà

$$\mathbf{x} + \mathbf{Cs}$$

Nell'esempio precedente

$$\begin{bmatrix} 3 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 & -2 \\ -4 & 3 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 3 \\ 1 \end{bmatrix} + \begin{bmatrix} -3 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪

7.3 Proprietà delle Reti di Petri

Sia $PN = (P, T, A, w, \mathbf{x}_0)$ una generica rete di Petri, con \mathbf{x}_0 la marcatura iniziale.

Definizione : una generica marcatura \mathbf{x}_i si dice **raggiungibile** se un'altra marcatura \mathbf{x}_j se esiste una sequenza di scatti che, porta dalla marcatura \mathbf{x}_i a quella \mathbf{x}_j .

Si definisce l'*insieme di marcature raggiungibili* della rete di Petri PN , l'insieme

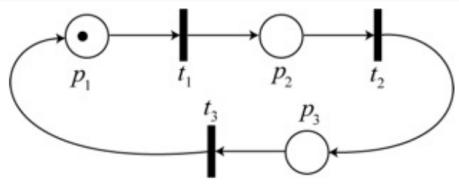
$$R(PN) = \{\mathbf{x}_j \mid \mathbf{x}_j \text{ è raggiungibile da } \mathbf{x}_0\}$$

ossia, l'insieme di tutte le marcature raggiungibili da quella iniziale, rappresenta tutti i possibili stati in cui può ricadere il sistema, considerando ogni possibile sequenza di transizioni.

Una marcatura \mathbf{x}^* si dice *marcatura di base* se essa è raggiungibile da ogni possibile marcatura $\mathbf{x} \in R(PN)$.

Definizione : La rete di Petri si dice **reversibile** se la marcatura iniziale \mathbf{x}_0 è una marcatura di base, ciò indica che da ogni possibile stato, è possibile tramite una sequenza di scatti, tornare allo stato iniziale.

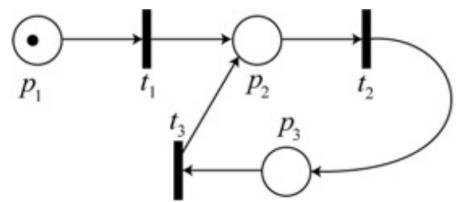
La rete mostrata in figura



è una rete reversibile, e l'insieme dei nodi raggiungibili è

$$R(PN) = \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right\}$$

La seguente rete invece



nonostante abbia lo stesso insieme di marcature raggiungibili $R(PN)$, non è reversibile, in quanto in nessun modo è possibile tornare allo stato in cui vi è un'unica marcatura su p_1 .

Definizione : Una transizione t di PN si dice **viva**, se, per ogni marcatura $\mathbf{x} \in R(PN)$, esiste un'altra marcatura \mathbf{x}' che è raggiungibile da \mathbf{x} , e nella sequenza di scatti necessaria per raggiungerla è presente t .

Ciò implica che una transizione t viva può scattare infinite volte, la rete gode della proprietà di **vivezza** se ogni transazione è viva. L'ultima rete di Petri mostrata non è viva, in quanto t_1 può essere abilitata una sola volta.

nota : Una rete di Petri può essere viva, e comunque non riuscire ad evolvere in nuovi stati, d'altro canto, una rete può non essere viva, e continuare ad evolversi.

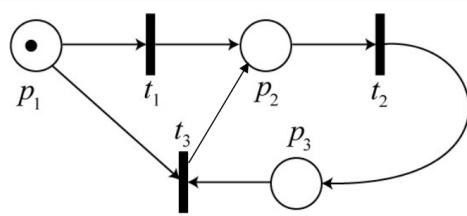
Definizione : Una rete PN è **bloccante** se esiste una marcatura $\mathbf{x}_b \in R(PN)$ in cui nessuna transizione è abilitata, ciò implica l'impossibilità di evolvere della rete (*deadlock*).

rete bloccante \implies rete non viva

rete non viva $\not\implies$ rete bloccante

La rete vista nell'esempio sopra, non è viva, ma non è nemmeno bloccante.

La seguente rete di Petri



è bloccante dato che, a partire dallo stato iniziale, se si attivano in sequenza $t_1 t_2$ si raggiunge la marcatura

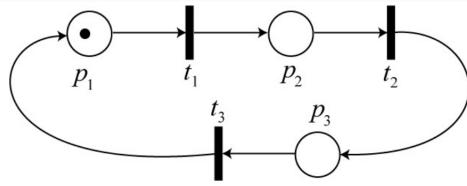
$$\mathbf{x}' \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

che non abilita alcuna transizione, in quanto

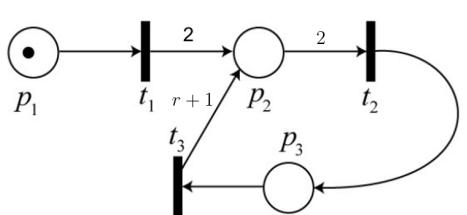
$$\mathbf{I}_3 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \implies \mathbf{x}' < \mathbf{I}_3$$

non è abilitata.

Definizione : Dato un posto p_i , e dato $k \in \mathbb{Z}^+$, questo si dice **k -limitato** se, per ogni possibile marcatura $\mathbf{x} \in R(PN)$, si ha che $x(p_i) \leq k$. Ossia, nel posto p_i non si possono accumulare più di k -token. La rete è **limitata** se ogni posto è limitato. Una rete si dice **safe** (o binaria) se ogni posto è 1-limitato. Un esempio di rete safe è la seguente:



Si consideri invece la seguente rete



al variare dei valori di r

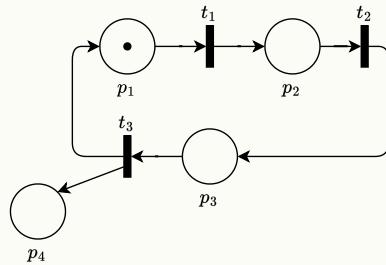
- per $r = 0$ la rete è bloccante, in quanto al susseguirsi delle transizioni t_1, t_2, t_3 , la marcatura corrente avrebbe un solo token su p_2 , e non sarebbe possibile abilitare nuovamente t_2 .
- per $r = 1$ si avrebbe una rete limitata, in particolare, i posti p_2, p_3 sarebbero 2-limitati.
- per $r > 1$ la rete sarebbe illimitata.

Un posto è illimitato se non esiste $k \in \mathbb{N}$ per cui è k -limitato, ossia, è possibile accumulare un numero infinito di token. La rete è **illimitata** se ha almeno un posto illimitato.

La proprietà di essere illimitata non influisce sulla proprietà di reversibilità

- una rete può essere illimitata (accumulare un numero infinito di token su un posot) ma essere comunque reversibile (è possibile "scaricare" quel posto un numero di volte pari al numero di token accumulati)
- una rete può essere irreversibile anche se limitata

La seguente rete di Petri è sia illimitata che irreversibile



dato che t_3 è viva e ad ogni scatto aggiunge un token su p_4 , che non può in nessun modo essere rimosso. L'insieme degli stati raggiungibili può essere espresso come segue

$$R(PN) = \left\{ \begin{bmatrix} x(p_1) \\ x(p_2) \\ x(p_3) \\ x(p_4) \end{bmatrix} \in \mathbb{N}^4 \text{ tale che } x(p_1) + x(p_2) + x(p_3) = 1 \right\}$$

Fra i posti p_1, p_2 e p_3 ci può essere contemporaneamente un singolo token, l'espressione

$$x(p_1) + x(p_2) + x(p_3) = 1$$

evidenzia che quei posti sono *conservativi*.

Definizione : In una rete di Petri, una **parte conservativa** è un sottoinsieme di posti per il quale, data una qualsiasi evoluzione ammissibile, una combinazione lineare (a coefficienti costanti in \mathbb{N}) di token si mantiene *costante*. Ossia, per ogni stato raggiungibile \mathbf{x} , si ha che

$$\Lambda \cdot \mathbf{x} = c$$

dove

- $c \in \mathbb{N}$
- $\Lambda \in \mathbb{N}^{|P|}$

$$[\lambda_1 \quad \lambda_2 \quad \dots \quad \lambda_{|P|}] \cdot \begin{bmatrix} p_1 \\ p_2 \\ \dots \\ p_{|P|} \end{bmatrix} = c$$

$$p_1\lambda_1 + p_2\lambda_2 + \dots + p_{|P|}\lambda_{|P|} = c$$

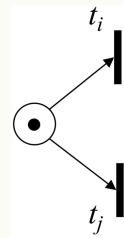
Il posto p_i appartiene alla parte conservativa della rete se $\lambda_i \neq 0$. Le parti conservative servono a modellare dei limiti fisici del sistema, se tutto l'insieme dei posti costituisce la parte conservativa ($\forall i, \lambda_i \neq 0$) allora la rete è limitata.



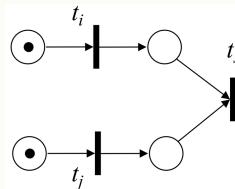
7.3.1 Strutture Fondamentali

Si individuano 3 strutture fondamentali di interconnessione posti/transazioni che descrivono determinate situazioni quando si parla di modellare un sistema reale.

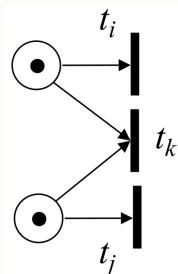
- **Transazioni in conflitto** : ci sono due transazioni adiacenti ad uno stesso posto, entrambe possono scattare singolarmente, ma non ci sono abbastanza risorse per far sì che possano scattare entrambe.



- **Parallelizzazione e sincronizzazione :** Ci sono due flussi separati che si ricongiungono su un'unica transazione adiacente a due posti da cui giungono gli archi entranti, che sono terminali dei flussi separati.



- **Transazioni in confusione :** Ci sono due flussi, questi possono sia ricongiungersi ad una transazione comune, sia continuare in parallelo, è una situazione in cui due transizioni sono in concorrenza fra loro, ma in conflitto con una terza.



nell'esempio mostrato, t_i è in concorrenza con t_j , e sia t_i che t_j sono in conflitto con t_k , si può quindi

- far scattare t_i e t_j (proseguire con la concorrenza)
- far scattare t_k (sincronizzare i flussi)

♪♫♪ ♪♫♪ ♪♫♪ ♪♫♪ ♪♫♪ ♪♫♪ ♪♫♪ ♪♫♪ ♪♫♪ ♪♫♪

7.4 Classi Particolari di Reti di Petri

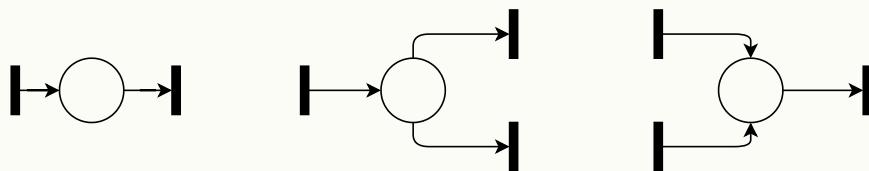
Alcune reti di Petri soddisfano alcuni vincoli che le fanno rientrare in determinate classi, rendendo più efficace un'analisi analitica. Considereremo esclusivamente reti di Petri ordinarie, ossia con pesi unitari sugli archi.

Macchina a Stati (SM)

Una rete di Petri rientra nella classe delle *State Machine* (equivalente ad un'automa a stati finiti) se

- ogni transizione, ha al più un solo posto in ingresso, e al più un solo posto in uscita

Una rete di questo tipo è conservativa, il numero di token non cambia mai, questo è scontato, per ogni transizione, è necessario un token in ingresso (un solo posto in ingresso, archi con peso unitario) e fornisce un token in uscita all'unico posto alla quale punta (sempre con arco unitario).



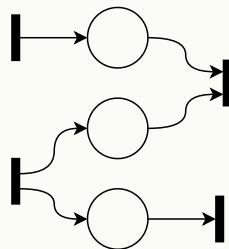
Se la marcatura iniziale ha un singolo token (c'è un solo token in un solo posto), allora la rete è safe. La rete è viva *se e solo se* esiste almeno un token, ed il grafo di Petri è fortemente connesso. Possono esserci dei conflitti (da un posto partono due archi) ma non sincronizzazioni e concorrenza (da una transazione non possono partire/congiungersi due archi).

Grafo Marcato (MG)

Risulta essere il complementare della macchina a stati, in questo caso, il vincolo è sui posti

- ogni posto può avere al più una transizione in ingresso, ed una transizioni in uscita

Può modellare la concorrenza e sincronizzazione, ma non può modellare i conflitti, in quanto da ogni posto non si potrà mai avere scelta di due transizioni.



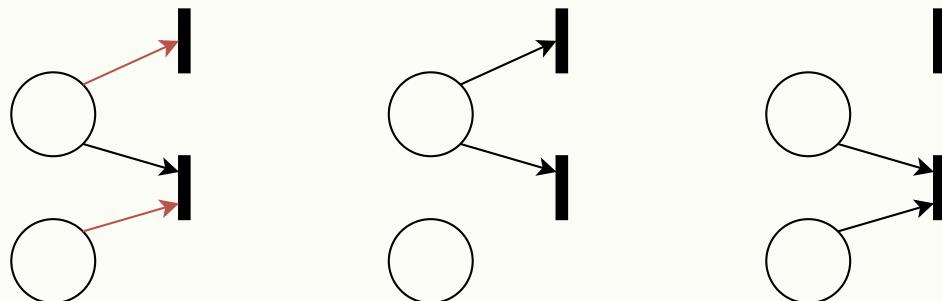
La rete è viva *se e solo se* ogni ciclo (componente fortemente connessa) del grafo di Petri, ha almeno un posto marcato nella marcatura iniziale.

Rete a Scelta Libera (FC)

Le reti a scelta libera generalizzano le SM e le MG, possono rappresentare sia concorrenza che conflitti, ma questi non devono influenzarsi fra loro. In particolare, per ogni arco che va da un posto ad una transizione

- a) o il posto è l'unico in ingresso a quella transizione (non c'è sincronizzazione di due flussi concorrenti)
- b) oppure la transizione è l'unica in uscita da quel posto (non ci sono conflitti)

Uno dei due vincoli (a o b) deve essere soddisfatto.



questo collegamento non può avvenire in una rete a scelta libera. Uno dei due vincoli va rispettato.

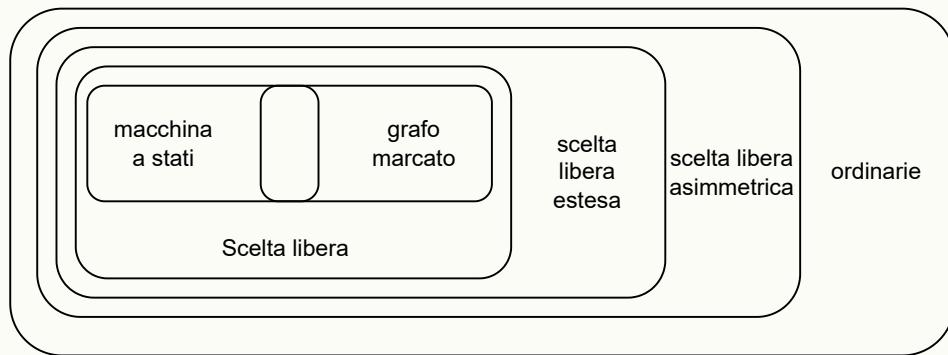
vincolo a rispettato, la transizione adesso ha un solo posto entrante, dato che da quel posto ci sono 2 archi uscenti.

vincolo b rispettato, il posto adesso ha un solo arco uscente, dato che tale arco va in una transizione con 2 archi entranti.

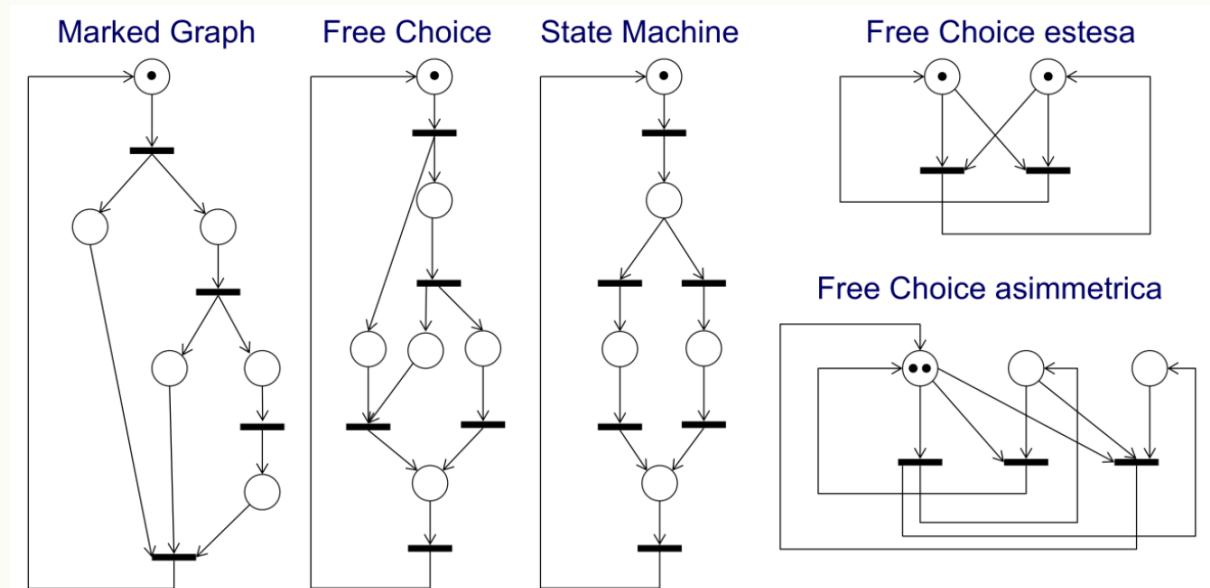
Inoltre, la rete può specificarsi in due ulteriori sottoclassi

- **FC Estesa** : Per ogni coppia di posti è vero che, o non hanno archi in uscita ad una stessa transizione, oppure, hanno *tutte* le transizioni in uscita in comune.
 - o i posti p_i e p_j non hanno transizioni in uscita in comune
 - oppure i posti p_i e p_j hanno tutte le transizioni in uscita in comune
- **FC Asimmetrica** : Per ogni coppia di posti è vero che, o non hanno archi in uscita ad una stessa transizione, oppure, tutte le transizioni in uscita di un posto, sono transizioni in uscita anche dell'altro.
 - le transizioni di uscita del posto p_i , o sono disgiunte da quelle del posto p_j , oppure ne sono un sotto-insieme.

Il seguente diagramma descrive l'ordine di gerarchia delle reti di Petri ordinarie.



Alcuni esempi di reti appartenenti alle diverse classi:



7.4.1 Esempi di Reti di Petri

Abbiamo visto che una rete di Petri può essere

- viva
- limitata
- reversibile

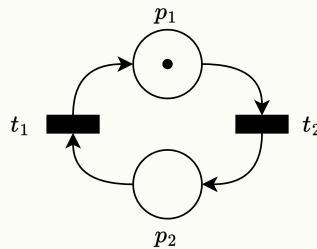
In questa sezione, considereremo diversi esempi di Reti di Petri, analizzandone (non sistematicamente, in maniera poco analitica) tali proprietà strutturali. È importante sapere che

le proprietà di limitatezza, vivezza e reversibilità sono **indipendenti** fra loro

Essendo che le proprietà sono 3, ed ognuna può essere presente o no, vedremo un esempio di tutte e $2^3 = 8$ le possibili combinazioni di queste.

Esempio 1

Si consideri la rete di Petri mostrata in figura

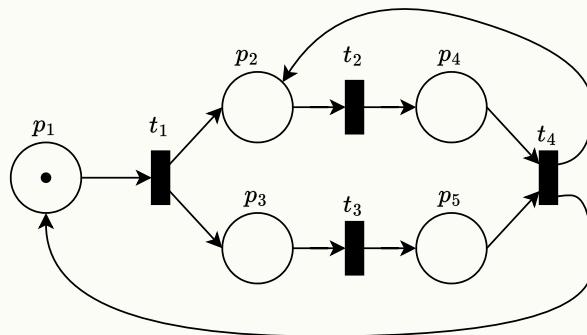


È ordinaria, inoltre è un SM, ed ha un solo token iniziale, è quindi conservativa e safe. Si può notare come la rete sia viva, in quanto ogni transizione è viva, è limitata (essendo safe) ed è reversibile, gli stati raggiungibili sono 2.

Limitata	Viva	Reversibile
Si	Si	Si

Esempio 2

Si consideri la rete di Petri mostrata in figura



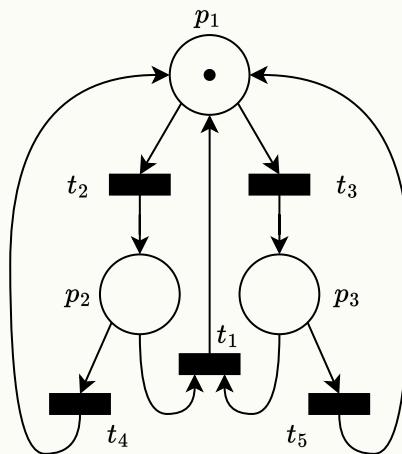
La prima osservazione riguarda la limitatezza, la transizione t_4 è viva, e può scattare un numero infinito di volte. Ad ogni scatto di questa, si aggiunge un nodo in più nel posto p_2 , i posti p_2 e p_4 non sono limitati, quindi la rete non è limitata.

Essendo ogni transizione viva, la rete è viva. Non è reversibile, dato che si possono aggiungere token su p_2 , ma non è possibile scaricarli.

Limitata	Viva	Reversibile
No	Si	No

Esempio 3

Si consideri la rete di Petri mostrata in figura

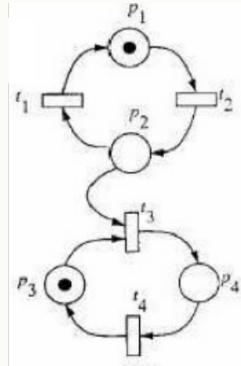


La rete è limitata in quanto vi è un solo token nella marcatura iniziale, e non ne vengono mai generati. Non è viva, dato che la transizione t_1 non può mai scattare, necessita di un token su p_2 ed un token su p_3 , ma il numero di token è 1 e la rete è conservativa. La rete è reversibile, e la verifica è immediata. Eliminando t_1 la rete diventa un SM.

Limitata	Viva	Reversibile
Si	No	Si

Esempio 4

Si consideri la rete di Petri mostrata in figura !TODO : rifare i disegni delle reti



Si noti come la sequenza t_2t_1 può scattare ciclicamente senza fine, c'è un conflitto fra t_1 e t_3 quando il token è su p_2 e p_3 , se scatta t_3 , il token su p_3 viene consumato insieme a quello su p_2 , rimane quindi un solo token su p_4 .

Quando il token è su p_4 , è abilitata solo t_4 , facendola scattare, si termina su una marcatura bloccante, con un solo token su p_3 .

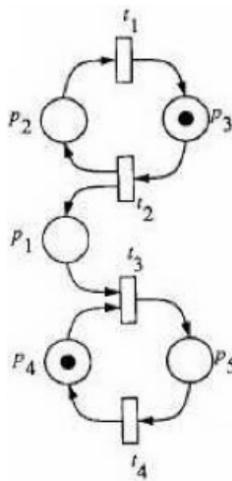
$$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \rightarrow t_2 \rightarrow \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \rightarrow t_3 \rightarrow \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \rightarrow t_4 \rightarrow \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \text{ stato bloccante}$$

La rete è quindi limitata (non si possono generare token), ma non è né reversibile né viva, in quanto esiste uno stato bloccante.

Limitata	Viva	Reversibile
Si	No	No

Esempio 5

Si consideri la rete di Petri mostrata in figura



Tale rete di Petri modella perfettamente il paradigma produttore-consamatore

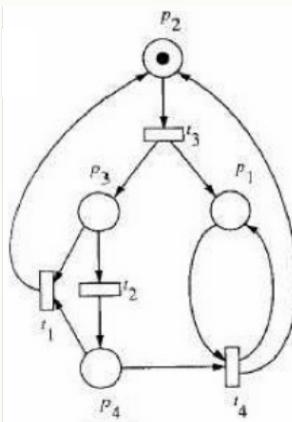
- dalla marcatura iniziale, è possibile generare k token su p_1 ripetendo ciclicamente la sequenza t_2t_1 per k volte.
- se ci sono k token su p_1 , è possibile scaricarli tutti eseguendo k volte la sequenza t_3t_4
- p_1 può accumulare indefinitivamente token (la rete non è limitata), la rete può eliminare token fino a scaricare totalmente p_1

Inoltre la rete è reversibile se t_3 scatta un numero di volte uguale a t_2 (ossia, si eliminano tanti token quanti generati, tornando ad un numero globale di 2 token). La rete è viva, dato che è un grafo marcato, ed ogni ciclo (componente fortemente connessa) contiene un token nella marcatura iniziale.

Limitata	Viva	Reversibile
No	Si	Si

Esempio 6

Si consideri la rete di Petri mostrata in figura

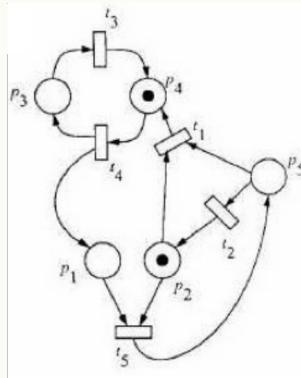


La rete non è limitata, si noti come (partendo dalla marcatura iniziale), la sequenza di scatti $t_3t_2t_4$ aggiunge un token su p_1 e può essere eseguita ciclicamente un numero indefinito di volte. Inoltre, non è viva, dato che la transizione t_1 non è mai abilitata, dato che la marcatura in cui p_3 e p_4 hanno almeno un token non è mai raggiungibile. Non è reversibile dato che ogni token aggiunto su p_1 non può essere eliminato.

Limitata	Viva	Reversibile
No	No	No

Esempio 7

Si consideri la rete di Petri mostrata in figura

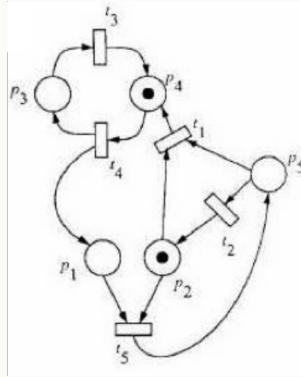


A partire dalla marcatura iniziale, la sequenza t_3t_4 può essere eseguita un numero indefinito di volte ciclicamente, aumentando sempre il numero di token su p_1 , che rende la rete non limitata. Si noti come t_1 non è mai abilitata, quindi la rete non è viva. La rete è reversibile dato che l'esecuzione ciclica della sequenza t_5t_2 scarica p_1 .

Limitata	Viva	Reversibile
No	No	Si

Esempio 8

Si consideri la rete di Petri mostrata in figura



Inizialmente l'unica transizione abilitata è t_1 , in seguito, l'unica sequenza ammissibile (ciclicamente) è $t_2t_1t_4t_3$, la marcatura iniziale

$$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

non è più raggiungibile dopo il primo scatto, la rete non è quindi reversibile. La rete è viva e limitata.

Limitata	Viva	Reversibile
Si	Si	No

♪ ♫ ♪ ♪ ♫ ♪ ♪ ♪ ♪ ♪ ♪ ♪ ♪

7.5 Albero di Raggiungibilità

Nella sezione precedente, abbiamo analizzato le proprietà di alcune reti di Petri, tramite una rapida analisi della topologia della rete, si vuole dare una metodologia più sistematica. Il seguente strumento, permette di analizzare e studiare la raggiungibilità (e le altre proprietà) di una rete di Petri $PN = (P, T, A, w, \mathbf{x}_0)$.

Definizione : Data PN , si definisce **albero di raggiungibilità** una rappresentazione grafica degli stati rappresentata con un albero, in cui i nodi sono le possibili marcature, e c'è un arco orientato da un nodo ad un altro se una transizione permette tale evoluzione dello stato.

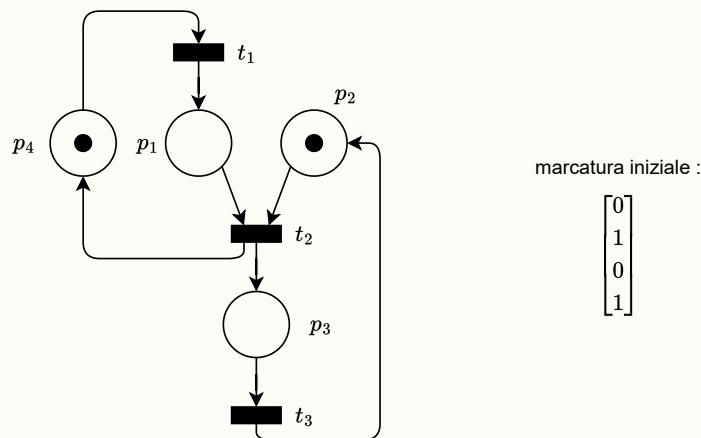
Il nodo radice rappresenta la marcatura iniziale della rete, l'albero viene disegnato a partire da questo, tracciando tutte le possibili evoluzioni, quando si incontra una marcatura già presente nell'albero, si denota il fatto che sia stata già visitata, e si lascia come foglia. Sono foglie anche le marcature che non abilitano alcuna transizione.

Tramite l'analisi dell'albero di raggiungibilità si può studiare

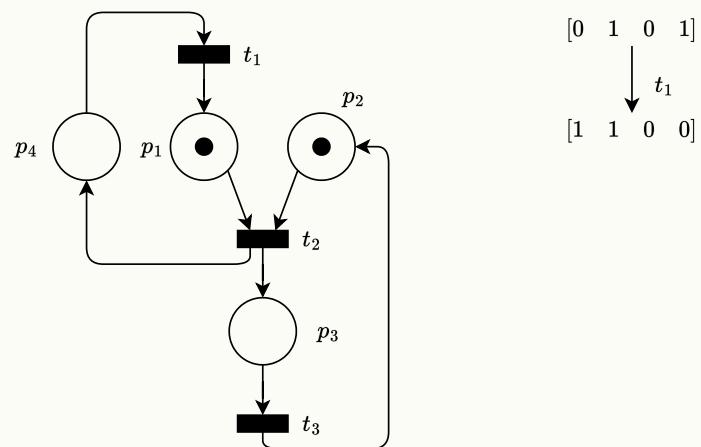
- l'insieme di raggiungibilità $R(PN)$, ossia, tutti i nodi dell'albero
- la reversibilità
- la vivezza
- la limitatezza

Esempio 1 : Rete limitata

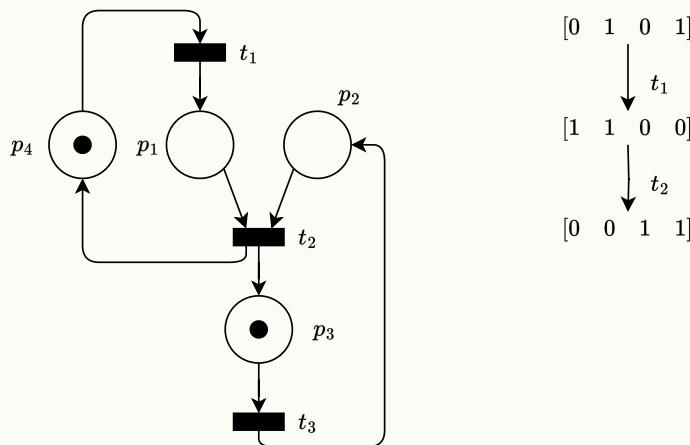
Si vuole tracciare l'albero di raggiungibilità della seguente rete di Petri



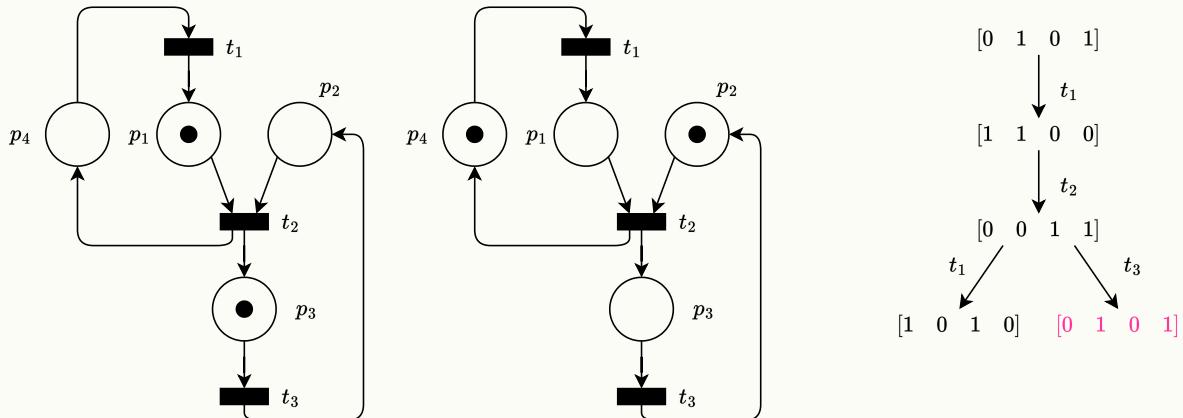
L'unica transizione abilitata è t_1



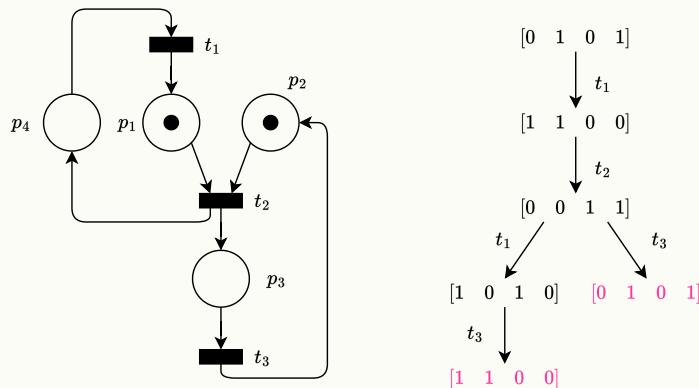
L'unica abilitata poi, è t_2



una volta abilitata t_2 , possono scattare sia t_3 che t_1



Appare una marcatura già visitata (evidenziata in rosa), quindi, si mantiene come foglia, e si continua la diramazione considerando la marcatura $[1 \ 0 \ 1 \ 0]$, da questa, l'unica transazione abilitata è t_3 .

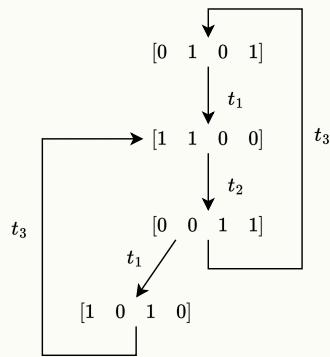


Anche qui si ricade in una marcatura già vista, l'albero di raggiungibilità è quindi concluso. I nodi presenti rappresentano l'insieme delle marcature raggiungibili:

$$R(PN) = \left\{ \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \right\}$$

Essendo $R(PN)$ un insieme finito, se ne conclude che la rete è limitata, in particolare è safe, dato che è 1-limitata. Inoltre si può vedere che se si trasforma l'albero di raggiungibilità (grafico) in un grafo

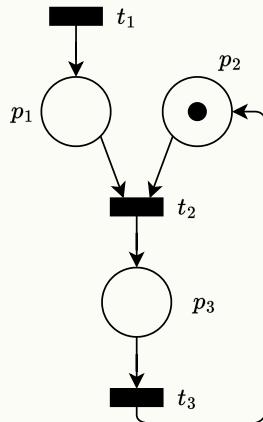
(unendo in un solo nodo le ripetizioni delle stesse marcature), si nota che questo è fortemente connesso, da tutte le foglie, si può risalire la nodo radice, l'albero è quindi reversibile.



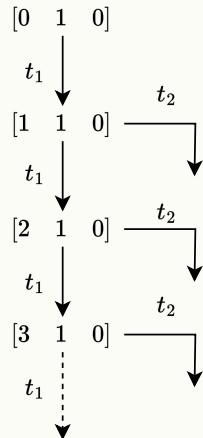
Inoltre da ogni nodo (marcatura) dell'albero, è possibile eseguire una sequenza che faccia scattare tutte le transizioni, la rete è quindi viva.

Esempio 2 : Rete illimitata

Si consideri la seguente rete di Petri



La rete è illimitata, si possono accumulare nodi su p_1 , ma è anche reversibile, dato che si possono scaricare eseguendo ciclicamente t_2t_3 . Essendo illimitata, il numero di nodi dell'albero di raggiungibilità è infinito.



In particolare, i token si conservano su p_2 e p_3 , ma non su p_1 .

$$R(PN) = \left\{ \begin{bmatrix} x(p_1) \\ x(p_2) \\ x(p_3) \end{bmatrix} \in \mathbb{N}^3 \text{ tale che } x(p_2) + x(p_3) = 1 \right\}$$

Per rappresentare un numero arbitrariamente grande di nodi, si fa utilizzo del particolare termine ω , che viene definito in modo da rispettare le seguenti proprietà

$$\omega + k = \omega \quad \omega - k = \omega \quad \forall \omega \in \mathbb{N}$$

Tenendo ciò in mente, è necessario dare la definizione di *ricoprimento*, è una relazione che coinvolge due marcature \mathbf{x}_1 , \mathbf{x}_2 di una Rete di Petri. Diciamo che \mathbf{x}_1 ricopre \mathbf{x}_2 se per ogni posto, il numero di token su tale posto nella marcatura \mathbf{x}_1 è maggiore o uguale al numero di token sullo stesso posto nella marcatura \mathbf{x}_2 , e per almeno un posto è strettamente maggiore.

$$\mathbf{x}_1 = \begin{bmatrix} x_1(p_1) \\ x_1(p_2) \\ \vdots \\ x_1(p_{|P|}) \end{bmatrix} \text{ ricopre } \mathbf{x}_2 = \begin{bmatrix} x_2(p_1) \\ x_2(p_2) \\ \vdots \\ x_2(p_{|P|}) \end{bmatrix} \implies \forall i, x_1(p_i) \geq x_2(p_i) \wedge \exists j \ x_1(p_j) > x_2(p_j)$$

Se una sequenza di scatti S porta \mathbf{x}_2 a \mathbf{x}_1 , e \mathbf{x}_1 ricopre \mathbf{x}_2 , allora la sequenza S sarà nuovamente ammissibile in \mathbf{x}_2 . L'implicazione è chiara

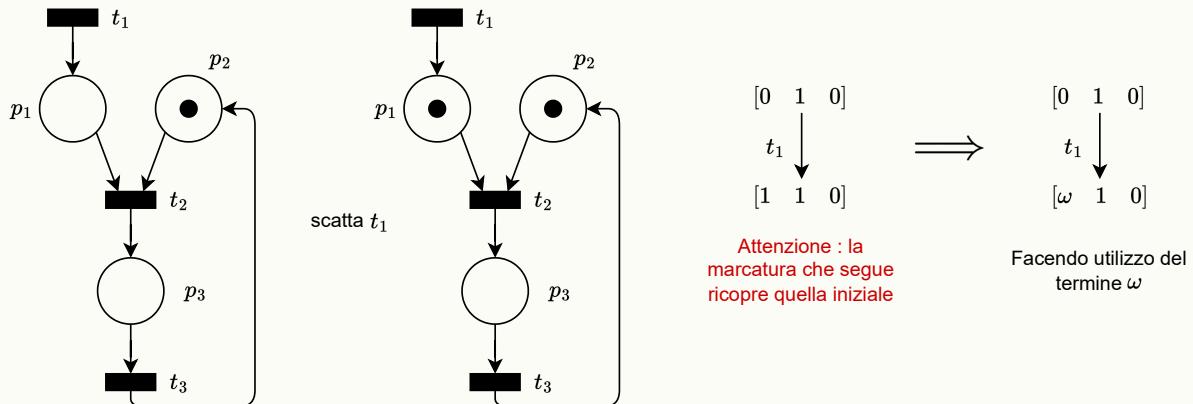
i posti di \mathbf{x}_1 che hanno un numero di token strettamente maggiore rispetto a \mathbf{x}_2 , potranno *accumulare token all'infinito* se viene eseguita ciclicamente la sequenza S .

Esempio

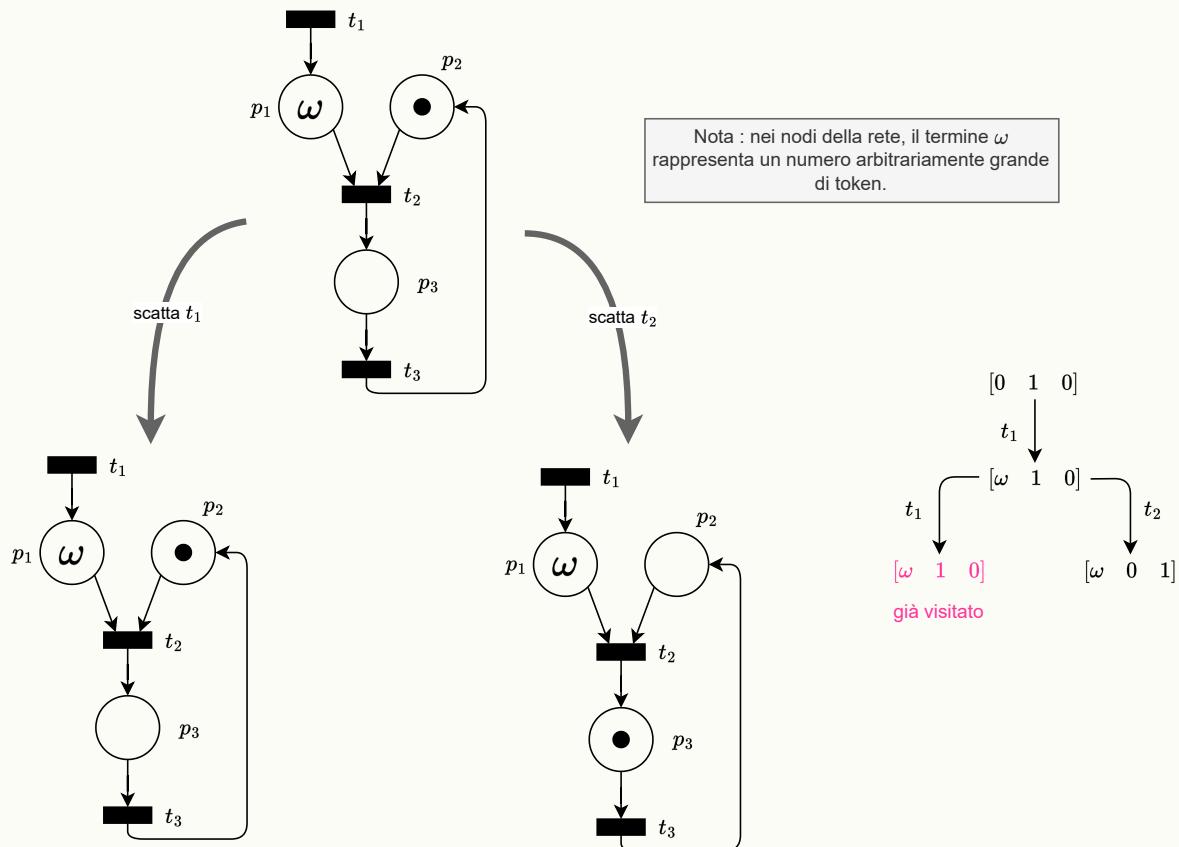
$\begin{bmatrix} 2 \\ 1 \\ 0 \\ 1 \end{bmatrix}$	ricopre	$\begin{bmatrix} 2 \\ 1 \\ 1 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 56 \\ 22 \\ 12 \\ 16 \end{bmatrix}$	non ricopre	$\begin{bmatrix} 0 \\ 0 \\ 13 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 2 \\ 1 \\ 0 \\ 1 \end{bmatrix}$	non ricopre	$\begin{bmatrix} 2 \\ 1 \\ 0 \\ 1 \end{bmatrix}$
--	---------	--	--	-------------	---	--	-------------	--

Durante la costruzione dell'albero di raggiungibilità, quando un nodo \mathbf{x} dell'albero è collegato ad un'altro nodo \mathbf{y} che lo ricopre, si sostituisce ω nei posti in cui $y(p_j) > x(p_j)$.

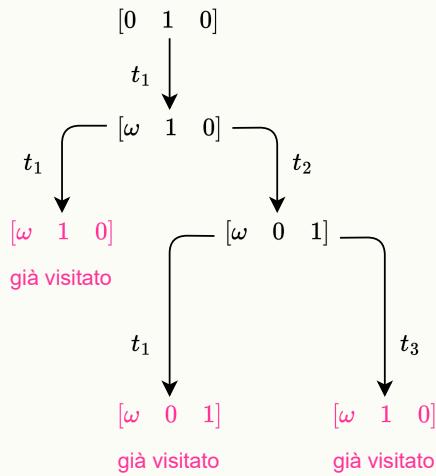
Torniamo all'esempio dell'albero con marcature infinite, con l'utilizzo del termine ω si può rendere finito, partendo dalla prima marcatura, si considera la prima (ed unica) transazione disponibile.



La marcatura $[\omega \ 1 \ 0]$ rappresenta tutte le marcature in cui i posti 2 e 3 hanno 1 e 0 token, e nel posto 1 c'è un arbitrario numero di token. Da questo insieme di marcature, si può proseguire facendo scattare nuovamente t_1 , oppure t_2 .



A questo punto, da $[\omega \ 0 \ 1]$, si può procedere chiamando nuovamente t_1 (aggiungendo un token su p_1), trovando nuovamente lo stato $[\omega \ 0 \ 1]$, oppure facendo scattare t_3 , e ritrovandosi nello stato (sempre già visitato) $[\omega \ 1 \ 0]$.



L'albero di raggiungibilità è completato.

♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪ ♪♪♪

7.6 Analisi Matriciale

Si può considerare la matrice di incidenza \mathbf{C} per osservare le proprietà strutturali di una rete di Petri, si ricordi che data una marcatura \mathbf{x} ed un vettore di sequenze \mathbf{s} , lo stato successivo è

$$\mathbf{x}' = \mathbf{x} + \mathbf{Cs}$$

Sappiamo che nella rete può essere presente un insieme di posti in cui il numero di token si conservi (se tale insieme copre tutti i posti, la rete è limitata), tale vincolo si esprime con la diseguaglianza

$$\gamma^T \cdot \mathbf{x} = c \text{ costante per ogni } \mathbf{x} \in R(PN)$$

Dove $\gamma \in \mathbb{N}^{|P|}$, e γ^T è il trasposto di γ , ossia semplicemente il vettore riga, quindi $\gamma^T \cdot \mathbf{x}$ è uno scalare. Il vettore γ è detto **P-Invariante**, per ovvie ragioni, le sue componenti devono essere intere e non negative.

$$\begin{bmatrix} \gamma_1 & \dots & \gamma_{|P|} \end{bmatrix} \cdot \begin{bmatrix} x(p_1) \\ \vdots \\ x(p_{|P|}) \end{bmatrix} = \gamma_1 x(p_1) + \dots + \gamma_{|P|} x(p_{|P|}) = c \text{ costante}$$

Ovviamente anche lo stato iniziale \mathbf{x}_0 deve essere conservativa

$$\gamma^T \cdot \mathbf{x}_0 = c$$

è quindi vero che

$$\gamma^T \cdot \mathbf{x}_0 = \gamma^T \cdot \mathbf{x} \quad \forall \mathbf{x} \in R(PN)$$

Ma $\mathbf{x} = \mathbf{x}_0 + \mathbf{Cs}$ per qualche sequenza ammissibile \mathbf{s} quindi

$$\gamma^T \cdot \mathbf{x}_0 = \gamma^T \cdot (\mathbf{x}_0 + \mathbf{C})$$

ma allora

$$\gamma^T \cdot \mathbf{x}_0 = \gamma^T \cdot \mathbf{x}_0 + \gamma^T \cdot \mathbf{C}$$

Ne consegue che

$$\gamma^T \cdot \mathbf{C} = \mathbf{0}$$

ma allora è possibile trovare un *P-Invariante* studiando **il nucleo della matrice di incidenza C**.

CAPITOLO

8

COMPLEMENTI

8.1 La Trasformata di Laplace

La trasformata di Laplace è una *trasformata integrale*, nello specifico, è una funzione che associa ad una funzione di variabile reale, una funzione di variabile complessa.

Definizione (Trasformata di Laplace) : : Sia f una funzione di variabile reale, nulla in $(-\infty, 0)$, si chiama trasformata di Laplace di f la funzione

$$\mathcal{L}[f](p) = \int_0^{+\infty} e^{-px} f(x) dx \quad p \in \mathbb{C}$$

Essendo $p = \alpha + i\beta$ una variabile complessa, la funzione integranda si può riscrivere

$$\int_0^{+\infty} e^{-px} f(x) dx = \int_0^{+\infty} e^{-(\alpha+i\beta)x} f(x) dx$$

Ricordando l'identità di Eulero

$$e^{ix} = \cos(x) + i \sin(x)$$

Si ha

$$e^{-(\alpha+i\beta)x} = e^{-\alpha x} \cdot e^{-\beta ix} = \quad (8.1)$$

$$e^{-\alpha x} \cdot (\cos(-\beta x) + i \sin(-\beta x)) = e^{-\alpha x} \cdot (\cos(\beta x) - i \sin(\beta x)) = \quad (8.2)$$

$$e^{-\alpha x} \cos(\beta x) - ie^{-\alpha x} \sin(\beta x) \quad (8.3)$$

Quindi

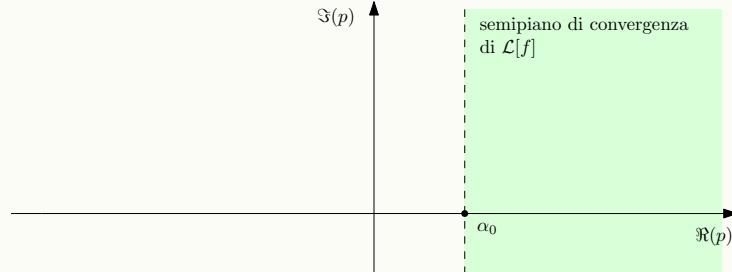
$$\begin{aligned} \mathcal{L}[f](p) &= \mathcal{L}[f](\alpha + i\beta) = \int_0^{+\infty} e^{-(\alpha+i\beta)x} f(x) dx = \\ &= \int_0^{+\infty} e^{-\alpha x} \cos(\beta x) f(x) - ie^{-\alpha x} \sin(\beta x) f(x) dx = \\ &= \int_0^{+\infty} e^{-\alpha x} \cos(\beta x) f(x) dx - i \int_0^{+\infty} e^{-\alpha x} \sin(\beta x) f(x) dx \end{aligned}$$

Se l'integrale $\mathcal{L}[f](\alpha + i\beta)$ converge per un certo $\alpha \in \mathbb{R}$, allora converge per $p = \alpha + i\beta$ per ogni altro $\beta \in \mathbb{R}$. Se per f esiste almeno un $p \in \mathbb{C}$ tale che $\mathcal{L}[f](p) < \infty$, allora f si dice *trasformabile secondo Laplace*.

In generale, se $\mathcal{L}[f](p) < \infty$ per $p = p_0$, allora è definita anche nel semipiano complesso

$$\{p \in \mathbb{C} \mid \Re(p) > \Re(p_0)\}$$

Sia α_0 l'estremo inferiore dell'insieme $\{\alpha \in \mathbb{R} \mid \mathcal{L}[f](p) < \infty \wedge \Re(p) > \alpha\}$, allora il semipiano $\{p \in \mathbb{C} \mid \Re(p) > \alpha_0\}$ è detto **semipiano di convergenza**.



Vediamo un esempio di trasformata, si consideri

$$H(x) = \begin{cases} 1 & \text{se } x \geq 0 \\ 0 & \text{altrimenti} \end{cases}$$

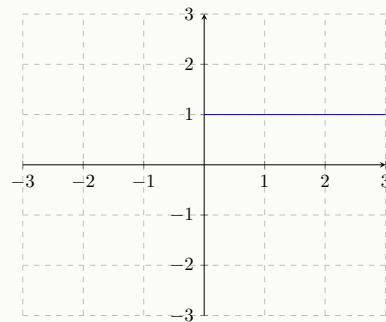


Figura 8.1: Funzione di Heaviside

Si calcola

$$\begin{aligned} \mathcal{L}[H](p) &= \int_0^{+\infty} e^{-px} \cdot 1 \, dx = \lim_{T \rightarrow +\infty} \mathcal{L}[H](p) = \int_0^T e^{-px} \cdot 1 \, dx = \lim_{T \rightarrow +\infty} \left[-\frac{e^{-px}}{p} \right]_0^T = \\ &= \lim_{T \rightarrow +\infty} -\frac{e^{-pT}}{p} - \left[-\frac{e^{-p0}}{p} \right] = \lim_{T \rightarrow +\infty} -\frac{e^{-pT}}{p} + \frac{1}{p} = \frac{1}{p} \end{aligned}$$

Il cui semipiano di convergenza risulta essere $\Re(p) > 0$.

8.1.1 Proprietà della Trasformata

Linearità

La trasformazione di Laplace gode della proprietà di *linearità*, siano $f(p)$ e $g(p)$ due funzioni trasformabili, siano $\lambda, \mu \in \mathbb{C}$ due costanti complesse, se la funzione $\lambda \cdot f(p) + \mu \cdot g(p)$ è trasformabile, allora

$$\mathcal{L}[\lambda \cdot f + \mu \cdot g](p) = \lambda \mathcal{L}[f](p) + \mu \mathcal{L}[g](p)$$

Il semipiano di convergenza sarà uguale all'intersezione dei due semipiani di convergenza delle funzioni di partenza, più precisamente se

- f ha come semipiano di convergenza $\Re(p) > \alpha$
- g ha come semipiano di convergenza $\Re(p) > \beta$
- allora $\lambda \cdot f + \mu \cdot g$ ha come semipiano di convergenza $\Re(p) > \max\{\beta, \alpha\}$

Ritardo

Sia f una funzione trasformabile, si consideri una costante reale $a > 0$, la funzione $g(x) = f(x - a)$ è detta funzione *ritardata*.

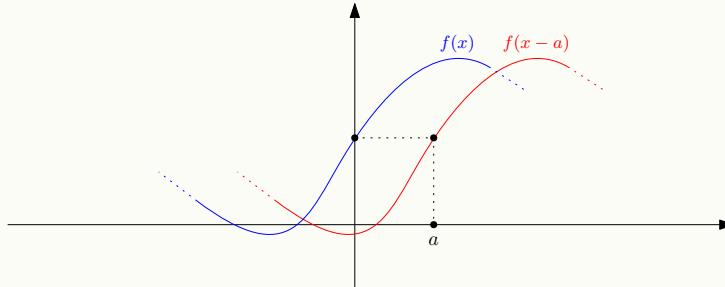


Figura 8.2: funzione ritardata

Per il calcolo della trasformata di $g(x) = f(x - a)$ si considera il cambio di variabile

$$\begin{aligned} t &= x - a \\ x &= t + a \end{aligned}$$

Si ricordi come, se f è nulla in $(-\infty, 0)$, allora g sarà nulla in $(0, a)$.

$$\begin{aligned} \mathcal{L}[g](p) &= \int_0^{+\infty} e^{-px} g(x) dx = \int_a^{+\infty} e^{-pt} f(x-a) dx = \int_a^{+\infty} e^{-p(t+a)} f(t) dx = \\ &= \int_a^{+\infty} e^{-pt-pa} f(t) dx = \int_a^{+\infty} e^{-pt} e^{-pa} f(t) dx = e^{-pa} \int_a^{+\infty} e^{-pt} f(t) dx = e^{-pa} \mathcal{L}[f](p) \end{aligned}$$

Dunque si ricavano le cosiddette *formule del ritardo* :

$$\mathcal{L}[f(x-a)](p) = e^{-pa} \mathcal{L}[f(x)](p)$$

$$\mathcal{L}[e^{ax} f(x)](p) = \mathcal{L}[f](p-a)$$

Trasformazione di una derivata e di una primitiva

La seguente proprietà risulta cruciale nell'utilizzo della trasformata di Laplace per la risoluzione di equazioni differenziali. Le dimostrazioni dei seguenti risultati non saranno trattate in quanto non sono argomento di questo corso.

Sia f una funzione derivabile, la cui derivata è continua in $[0, \infty)$. Sia inoltre f' trasformabile, con semipiano di convergenza $\Re(p) > \alpha$, allora anche f è trasformabile, ha semipiano di convergenza $\Re(p) > \max\{\alpha, 0\}$, e vale la seguente identità :

$$\mathcal{L}[f'](p) = p\mathcal{L}[f](p) - f(0)$$

Si generalizza per derivate di ordine maggiore

$$\mathcal{L}[f''](p) = p^2 \mathcal{L}[f](p) - pf(0) - f'(0)$$

Analogamente, sia $F(x) = \int_0^x f(t) dt$, se f è trasformabile ed ha semipiano di convergenza $\Re(p) > \alpha$, allora anche F lo è, ha semipiano di convergenza $\Re(p) > \max\{\alpha, 0\}$ e vale che

$$\mathcal{L}[F](p) = \frac{1}{p} \mathcal{L}[f](p)$$

Convoluzione

Siano f e g due funzioni integrabili secondo Riemann e nulle in $(-\infty, 0)$, l'operatore $*$ detto **convoluzione** è definito nel modo seguente

$$(f * g)(x) = \int_0^{+\infty} f(x-t)g(t) dt = \int_0^{+\infty} f(t)g(x-t) dt$$

Se f è trasformabile, e $|g|$ lo è, nello stesso semipiano, allora $f * g$ è trasformabile e vale

$$\mathcal{L}[f * g](p) = \mathcal{L}[f](p) \cdot \mathcal{L}[g](p)$$

Derivata ed Integrale della trasformata di Laplace

Essendo $\mathcal{L}[f](p) = \int_0^{+\infty} f(x)e^{-px} dx$ si ha

$$\begin{aligned} \frac{d}{dp} \mathcal{L}[f](p) &= \frac{d}{dp} \int_0^{+\infty} f(x)e^{-px} dx \\ \frac{d}{dp} \mathcal{L}[f](p) &= \int_0^{+\infty} \frac{d}{dp}(f(x)e^{-px}) dx \\ \frac{d}{dp} \mathcal{L}[f](p) &= \int_0^{+\infty} -xf(x)e^{-px} dx \\ \frac{d}{dp} \mathcal{L}[f](p) &= \mathcal{L}[-xf(x)](p) \end{aligned}$$

Generalizzando, per ogni $n \geq 0$

$$\boxed{\frac{d^n}{dp^n} \mathcal{L}[f](p) = \mathcal{L}[-(1)^n x^n f(x)](p)}$$

Esempio di calcolo : Si vuole trovare

$$\mathcal{L}[x \sin(\omega x)](p)$$

Essendo

$$\mathcal{L}[\sin(\omega x)](p) = \frac{\omega}{p^2 + \omega^2}$$

Ho che

$$\mathcal{L}[x \sin(\omega x)](p) = -\frac{d}{dp} \mathcal{L}[\sin(\omega x)](p) = -\frac{d}{dp} \left(\frac{\omega}{p^2 + \omega^2} \right) = \frac{2p\omega}{(p^2 + \omega^2)^2}$$

Trascurando il procedimento, la formula per l'integrale di una trasformata è la seguente

$$\boxed{\int_p^{+\infty} \mathcal{L}[f](s) ds = \mathcal{L}\left[\frac{f(x)}{x}\right](p)}$$

8.1.2 Trasformata inversa

La funzione che associa ad ogni funzione trasformabile la sua trasformata, è iniettiva, se $F(p)$ è una trasformata di Laplace, esiste un'unica funzione f tale che $\mathcal{L}[f](p) = F(p)$. Data F , è possibile ottenere la funzione di base su cui si è effettuata la trasformata, tale operazione è detta *trasformazione inversa* di Laplace, si indica con \mathcal{L}^{-1}

$$\begin{aligned} \mathcal{L}[f] &= F \\ \mathcal{L}^{-1}[F] &= f \end{aligned}$$

Le formule di trasformazione derivate dalle proprietà (raggruppate alla fine di questa sezione), se lette al contrario valgono come formule di anti-trasformata.

Esempio di calcolo : Ricordando che $\mathcal{L}[e^{-ax}](p) = \frac{1}{p+a}$, si vuole calcolare la trasformata inversa di

$$\frac{2}{p+3}$$

$$\mathcal{L}^{-1}\left[\frac{2}{p+3}\right](x) = \quad (8.4)$$

$$2 \cdot \mathcal{L}^{-1}\left[\frac{1}{p+3}\right](x) = \quad (8.5)$$

$$2 \cdot e^{-3x} \cdot H(x) \quad (8.6)$$

Una funzione risultante da un anti trasformata va moltiplicata per la funzione di Heaviside $H(x)$ in quanto deve essere nulla in $(-\infty, 0)$. **Esempio di calcolo :** Si vuole trovare l'anti trasformata di

$$F(p) = \frac{1}{p(p^2 + 1)}$$

Riscrivo la funzione

$$\frac{1}{p(p^2 + 1)} = \frac{1}{p^3 + p} = \frac{1}{p} - \frac{p}{p^2 + 1}$$

Applicando la linearità ho

$$\mathcal{L}^{-1}\left[\frac{1}{p} - \frac{p}{p^2 + 1}\right](x) = \mathcal{L}^{-1}\left[\frac{1}{p}\right](x) - \mathcal{L}^{-1}\left[\frac{p}{p^2 + 1}\right](x) = \quad (8.7)$$

$$H(x) - \cos(x) \cdot H(x) = H(x)(1 - \cos(x)) \quad (8.8)$$

8.1.3 Trasformate note

Funzione	Trasformata	Semipiano di convergenza
1	$\frac{1}{p}$	$\Re(p) > 0$
e^{-ax}	$\frac{1}{p+a}$	$\Re(p) > -\Re(a)$
x	$\frac{1}{p^2}$	$\Re(p) > 0$
x^n	$\frac{n!}{p^{n+1}}$	$\Re(p) > 0 \ n \in \mathbb{N}$
$\sin(\omega x)$	$\frac{\omega}{p^2 + \omega^2}$	$\Re(p) > 0$
$\cos(\omega x)$	$\frac{p}{p^2 + \omega^2}$	$\Re(p) > 0$
δ	1	$p \in \mathbb{C}$
$\cosh(ax)$	$\frac{p}{p^2 - a^2}$	$\Re(p) > \Re(a) $
$\sinh(ax)$	$\frac{a}{p^2 - a^2}$	$\Re(p) > \Re(a) $

8.1.4 Funzione di trasferimento

Come già accennato, la trasformata di Laplace è utile nella risoluzione di equazioni differenziali. Si consideri il seguente problema di Cauchy

$$a_0 y''(t) + a_1 y'(t) + a_2 y(t) = b(t) \quad \begin{cases} y(0) = \alpha \\ y'(0) = \beta \end{cases}$$

Si applica la trasformata all'equazione, ottenendo

$$\mathcal{L}[a_0 y'' + a_1 y' + a_2 y](p) = \mathcal{L}[b](p)$$

si applica la linearità

$$a_0 \mathcal{L}[y''](p) + a_1 \mathcal{L}[y'](p) + a_2 \mathcal{L}[y](p) = \mathcal{L}[b](p)$$

Chiamo

$$\mathcal{L}[y](p) = Y(p) \quad \mathcal{L}[b](p) = B(p)$$

ed applico le proprietà della trasformazione di una derivata

$$a_0(p^2Y(p) - p\alpha - \beta) + a_1(pY(p) - \alpha) + a_2Y(p) = B(p)$$

$$a_0p^2Y(p) - a_0p\alpha - a_0\beta + a_1pY(p) - a_1\alpha + a_2Y(p) = B(p)$$

esplicito $Y(p)$:

$$Y(p)(a_0p^2 + a_1p + a_2) = B(p) + a_0p\alpha + a_0\beta + a_1\alpha$$

$$Y(p) = \frac{1}{(a_0p^2 + a_1p + a_2)}B(p) + a_0p\alpha + a_0\beta + a_1\alpha$$

Pongo

$$S(p) = \frac{1}{(a_0p^2 + a_1p + a_2)}$$

Tale S è detta **funzione di trasferimento**, se le condizioni iniziali sono entrambe nulle, ossia $\alpha = \beta = 0$, si ha

$$Y(p) = S(p) \cdot B(p)$$

$$\mathcal{L}[y](p) = S(p) \cdot \mathcal{L}[b](p)$$

Ricordando la convoluzione di una trasformata, si ha che

$$y(t) = \mathcal{L}^{-1}[S \cdot B](t)$$

$$y(t) = \mathcal{L}^{-1}[S](t) * \mathcal{L}^{-1}[B](t)$$

$$y(t) = \mathcal{L}^{-1}[S](t) * b(t)$$

Le seguenti formule hanno un significato fisico notevole, supponiamo che vi sia un sistema fisico caratterizzato da un ingresso $b(t)$, ed un uscita $y(t)$, ad esempio, $b(t)$ è una forza, e $y(t)$ il moto di una particella. Trovare esplicitamente il moto y non è banale, è possibile quindi applicare la trasformata, passando nel dominio complesso di Laplace, per poi risolvere l'equazione ed applicare l'anti trasformata, trovando così il moto.

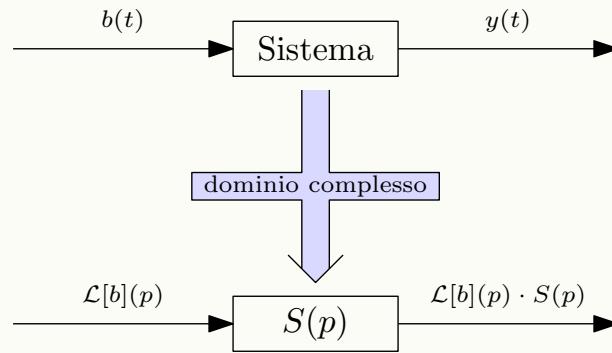


Figura 8.3: Funzione di Trasferimento

La funzione $S(p)$ quindi caratterizza totalmente il sistema fisico nel dominio di Laplace, in quanto basta moltiplicarla alla trasformata del segnale in ingresso per ottenere la trasformata del segnale in uscita.

Esempio di calcolo : Si consideri il seguente problema di Cauchy

$$y''(t) + 4y'(t) + 3y(t) = 0 \quad \begin{cases} y(0) = 0 \\ y'(0) = 1 \end{cases}$$

Si applica la trasformazione di Laplace

$$\mathcal{L}[y''](p) + 4\mathcal{L}[y'](p) + 3\mathcal{L}[y](p) = 0$$

Chiamando $\mathcal{L}[y](p) = Y(p)$, si ha

$$p^2Y(p) - py(0) - y'(0) + 4pY(p) - 4y(0) + 3Y(p) = 0$$

$$Y(p)(p^2 + 4p + 3) - 1 = 0$$

$$Y(p) = \frac{1}{p^2 + 4p + 3} = \frac{1}{(p+1)(p+3)} = \frac{A}{(p+1)} + \frac{B}{(p+3)}$$

dove $A = \frac{1}{p+1}$ se $p = -3$ e $B = \frac{1}{p+3}$ se $p = -1$, quindi

$$A = \frac{1}{(-3)+1} = -\frac{1}{2}$$

$$B = \frac{1}{(-1)+3} = \frac{1}{2}$$

Quindi

$$Y(p) = -\frac{1}{2} \frac{1}{p+3} + \frac{1}{2} \frac{1}{p+1}$$

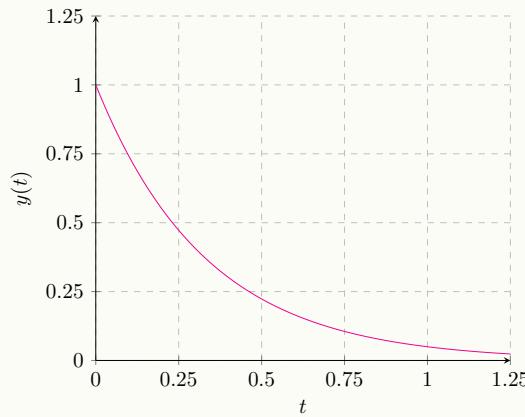
Si applica l'anti trasformata

$$y(t) = \mathcal{L}^{-1}\left[-\frac{1}{2} \frac{1}{p+3} + \frac{1}{2} \frac{1}{p+1}\right](t)$$

$$y(t) = -\frac{1}{2} \mathcal{L}^{-1}\left[\frac{1}{p+3}\right](t) + \frac{1}{2} \mathcal{L}^{-1}\left[\frac{1}{p+1}\right](t)$$

Ricordando che $\mathcal{L}[e^{-ax}](p) = \frac{1}{p+a}$ si ha

$$y(t) = \left(\frac{1}{2}e^{-t} - \frac{1}{2}e^{-3t}\right)H(t)$$



8.1.5 Zeri e Poli

Riprendiamo in analisi la seguente EDO che descrive un sistema

$$a_0y''(t) + a_1y'(t) + a_2y(t) = 0 \quad \begin{cases} y(0) = y_0 \\ y'(0) = 0 \end{cases}$$

Nella seguente notazione, verrà utilizzata s per la variabile complessa, applicando la trasformata si ottiene

$$a_0(s^2Y(s) - sy_0) + a_1(sY(s) - y_0) + a_2Y(s) = 0$$

$$a_0 s^2 Y(s) - a_0 s y_0 + a_1 s Y(s) - a_1 y_0 + a_2 Y(s) = 0$$

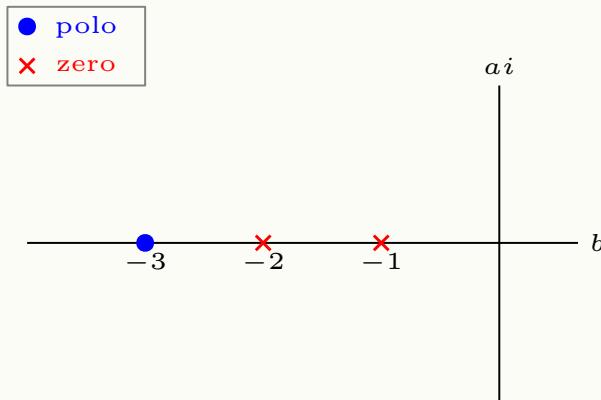
Si risolve per $Y(s)$

$$Y(s) = \frac{y_0(a_0 s + a_1)}{(a_0 s^2 + a_1 s + a_2)} = \frac{p(s)}{q(s)}$$

Il polinomio al denominatore $q(s) = (a_0 s^2 + a_1 s + a_2)$ si dice **equazione caratteristica** e le sue radici, dette **poli**, determinano il carattere della risposta nel tempo del sistema. Le radici del numeratore $p(s)$ sono dette **zeri** del sistema. Ai poli, la funzione $Y(s)$ va all'infinito, sugli zeri invece va a zero. Il numero di zeri è sempre minore o uguale al numero dei poli.

Esempio : Supponiamo che $a_0 = 1$, $a_2 = 2$ e $a_1 = 3$, l'equazione diventa

$$Y(s) = \frac{y_0(s+3)}{(s^2 + 3s + 2)} = \frac{y_0(s+3)}{(s+1)(s+2)}$$



Dato un sistema descritto da un EDO, una volta ottenuta la sua evoluzione nel tempo grazie alla trasformazione di Laplace, è utile studiare il *valore finale* della risposta, detto anche *stato stazionario*, sia $y(t)$ tale risposta, si vuole determinare l'evoluzione del sistema per t che tende ad infinito.

Teorema (valore finale) : : Sia $y(t)$ una funzione e $Y(s)$ la sua trasformata di Laplace, se $\lim_{t \rightarrow \infty} y(t) < \infty$ allora

$$\lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} s Y(s)$$

Definizione (Stabilità) : Un sistema dinamico è *stabile* se e solo se tutti i poli della sua funzione di trasferimento hanno parte reale negativa.

8.1.6 Coefficiente di Smorzamento e Pulsazione Naturale

Il coefficiente di smorzamento ζ e la pulsazione naturale ω_n sono due parametri fondamentali che caratterizzano il comportamento di un sistema dinamico, e sono strettamente legati alla posizione dei poli nel piano complesso s della funzione di trasferimento.

Una EDO del secondo ordine, se analizzata nel dominio di Laplace, avrà al più due poli.

Claim : La trasformata di Laplace di una EDO di ordine n avrà al più n poli.

La generica trasformazione di Laplace di una EDO omogenea di ordine 2 ha la seguente forma

$$Y(s) = \frac{a_0 s \alpha + a_0 \beta + a_1 \alpha}{(a_0 s^2 + a_1 s + a_2)}$$

La funzione caratteristica ha $a_0 s^2 + a_1 s + a_2$ due poli, siano questi s_1, s_2 , si pongono

$$s_1 = -\zeta \omega_n + \omega_n \sqrt{\zeta^2 - 1}$$

$$s_2 = -\zeta \omega_n - \omega_n \sqrt{\zeta^2 - 1}$$

I termini ζ e ω_n sono, rispettivamente, il coefficiente di smorzamento e la pulsazione naturale del sistema descritto dalla EDO. In generale, una tipica funzione di trasferimento del secondo ordine è della forma

$$G(S) = K \frac{1}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

Il calcolo di tali termini avviene confrontando i valori del denominatore a quelli della funzione generica.

Esempio

Si ha la seguente funzione di trasferimento

$$10 \frac{1}{s^2 + 4s + 16}$$

Si ha

$$K = 10$$

$$2\zeta\omega_n = 4$$

$$\omega_n^2 = 16$$

Allora

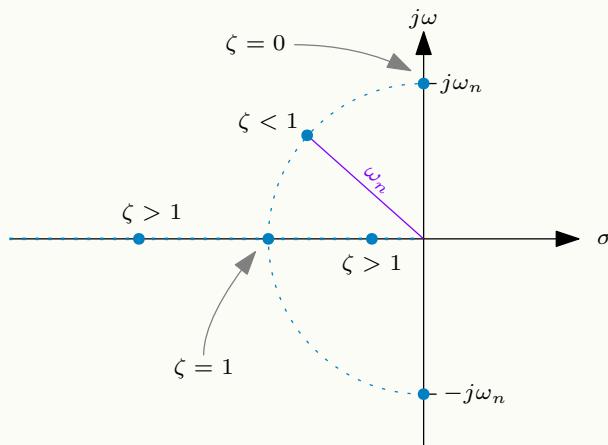
$$\omega_n = \sqrt{16} = 4 \implies \quad (8.9)$$

$$2\zeta \cdot 4 = 4 \implies 2\zeta = 1 \implies \zeta = \frac{1}{2} \quad (8.10)$$

Un sistema può essere

- **sotto smorzato** se $\zeta < 1$
- **criticamente smorzato** se $\zeta = 1$
- **sovra smorzato** se $\zeta > 1$

Soffermiamoci sulla rappresentazione polare dei poli della funzione di trasferimento, ω_n e ζ sono numeri reali, i poli quindi, per definizione, saranno valori reali se $\zeta \geq 1$, in quanto il termine sotto radice nella definizione di questi si annullerebbe, differentemente, se $\zeta = 0$, il termine a sinistra si annullerebbe, e rimarrebbe esclusivamente il valore immaginario.



I poli sono, o entrambi reali, oppure complessi e coniugati.

- finché ζ è minore di 1, i poli, definiti $-\zeta\omega_n \pm \omega_n\sqrt{\zeta^2 - 1}$ avranno parte immaginaria, e si troveranno sul semicerchio di circonferenza ω_n
- se $\zeta = 0$, allora i poli avranno parte reale nulla e si troveranno sull'asse immaginario
- se $\zeta \geq 1$, allora il termine sotto radice non sarà immaginario quindi i poli vivranno sull'asse reale. Se ciò avviene, il sistema è stabile.

8.1.7 Espansione in Fratti Semplici

Consideriamo una generica funzione di trasferimento di un sistema, nella forma

$$F(S) = \frac{N(s)}{D(s)}$$

Tali funzioni saranno sempre di tale forma, $N(s)$ e $D(s)$ sono i polinomi, sia r il grado di N e sia q il grado di D , si avranno

- r zeri $z_1, z_2 \dots, z_r \in \mathbb{C}$
 - di molteplicità m_j con $j = 1 \dots, r$
- q poli $p_1, p_2 \dots, p_q \in \mathbb{C}$
 - di molteplicità n_i con $i = 1 \dots, q$

La funzione $F(S)$ può essere scritta nella forma

$$F(S) = K \frac{(s - z_1)^{m_1}(s - z_2)^{m_2} \dots (s - z_r)^{m_r}}{(s - p_1)^{n_1}(s - p_2)^{n_2} \dots (s - p_r)^{n_q}}$$

Applicare l'anti-trasformata ad una funzione di questo tipo è difficile, infatti è possibile riscrivere $F(s)$ come segue

$$\begin{aligned} F(S) &= \frac{C_{1,1}}{(s - p_1)} + \dots + \frac{C_{1,n_1}}{(s - p_1)^{n_1}} + \dots \\ &= \frac{C_{2,1}}{(s - p_2)} + \dots + \frac{C_{2,n_2}}{(s - p_2)^{n_2}} + \dots \\ &\quad \vdots \\ &= \frac{C_{q,1}}{(s - p_q)} + \dots + \frac{C_{q,n_q}}{(s - p_q)^{n_q}} + \dots \end{aligned}$$

In tale forma è semplice calcolare l'anti trasformata grazie alla proprietà di linearità, infatti, il risultato è noto

$$\mathcal{L}^{-1} \left[\frac{C_{i,j}}{(s - p_i)^j} \right] (t) = \frac{C_{i,j}}{(j-1)!} t^{j-1} e^{p_i t} \cdot H(t)$$

A questo punto è necessario poter calcolare i coefficienti $C_{i,j}$. [da continuare](#)

~~~ ~~~ ~~~ ~~~ ~~~ ~~~ ~~~ ~~~ ~~~ ~~~ ~~~

## 8.2 Elementi di Teoria dei Sistemi e Controlli Automatici

Possiamo rappresentare un sistema dinamico (a tempo continuo) come un processo che date  $m$  entrate restituisce  $p$  uscite, convenzionalmente, definiamo  $u$  l'entrata del sistema e  $y$  l'uscita.

$$\bar{u}(t) = \begin{bmatrix} u_1(t) \\ \vdots \\ u_m(t) \end{bmatrix} \quad \bar{y}(t) = \begin{bmatrix} y_1(t) \\ \vdots \\ y_p(t) \end{bmatrix}$$



Verranno trattati i sistemi a dimensione finita, essi sono caratterizzati da delle cosiddette *variabili di stato* (o da un vettore di stato)

$$\bar{x}(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{bmatrix} \in \mathbb{R}^n$$

Un sistema ad  $n$  variabili di stato è detto di ordine  $n$ .

Un sistema verrà analizzato per un intervallo di tempo  $t \geq t_0$ , quindi a partire da un istante di partenza  $t_0$ , per poter calcolare  $y(t)$  per  $t > t_0$  è *necessario* conoscere  $\bar{x}(t_0)$ . Un sistema dinamico è rappresentato come segue

$$\begin{cases} \dot{x}(t) = f(x(t), u(t), t) \\ y(t) = g(x(t), u(t), t) \end{cases}$$

nel caso scalare,  $f$  e  $g$  descrivono il comportamento del sistema, generalmente sono grandezze associate ad accumuli di energie, masse ecc... Per brevità di notazione verrà scritto

$$\begin{cases} \dot{x} = f(x, u, t) \\ y = g(x, u, t) \end{cases}$$

Il sistema si dice **tempo invariante** se  $f$  e  $g$  non dipendono da  $t$ .

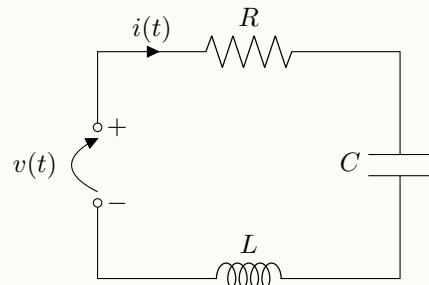
Il sistema si dice **lineare** se  $f$  e  $g$  sono lineari in  $x, u$ .

Il sistema si dice **monovariabile** se l'ingresso e l'uscita sono funzioni ad una variabile. Altrimenti si dice multivariabile.

Ci occuperemo di sistemi **LTI**, ossia lineari e tempo invarianti. L'andamento delle variabili di stato  $\bar{x}$  è detto *movimento dello stato*, mentre l'andamento delle variabili  $\bar{y}$  è il *movimento dell'uscita*. Per i sistemi stazionari è lecito assumere che l'istante iniziale  $t_0$  sia 0.

### Esempio

Si consideri il circuito  $RLC$  mostrato in figura



le tensioni ai margini della resistenza, del condensatore e dell'induttore sono precisamente

$$\begin{aligned} v_L(t) &= Li'(t) \\ v_R(t) &= Ri(t) \\ v_C(t) &= \frac{1}{C} \int_0^t i(t) dt \end{aligned}$$

La legge di Kirchoff delle tensioni stabilisce che

$$v(t) = v_L(t) + v_R(t) + v_C(t)$$

quindi

$$v(t) = Li'(t) + Ri(t) + \frac{1}{C} \int_0^t i(t) dt$$

derivando rispetto a  $t$  si ottiene

$$\begin{cases} v'(t) = Li''(t) + Ri'(t) + \frac{1}{C} i(t) \\ y(t) = i(t) \quad \text{uscita} \end{cases} \quad v$$

Tale modello è quello di un sistema lineare tempo invariante, la tensione rappresenta l'ingresso del sistema ed in base a questa è possibile ricavarne la corrente. Consideriamo una tensione costante di  $v(t) = v_0$

$$0 = Li''(t) + Ri'(t) + \frac{1}{C} i(t)$$

Si pongono i termini

$$\alpha = \frac{R}{2L} \quad \omega_0 = \frac{1}{\sqrt{LC}} \quad \zeta = \frac{R}{2} \sqrt{\frac{C}{L}}$$

$$\omega_n = \omega_0 \sqrt{1 - \zeta^2}$$

La soluzione di tale equazione è nota

$$i(t) = i_0 e^{-\alpha t} \sin(\omega_n t)$$

### 8.2.1 Equilibrio

Consideriamo il generico sistema

$$\begin{cases} \dot{x} = f(x, u) \\ y = g(x, u) \end{cases}$$

Se esiste un'ingresso  $u(t) = u_0$  costante per cui il movimento dello stato è costante  $x(t) = x_0$ , allora  $x_0$  è detto *stato di equilibrio*. Tutti gli stati di equilibrio si trovano al variare di  $u_0$ . Analogamente per le *uscite di equilibrio* in cui  $y(t) = y_0$  per un ingresso costante  $u(t) = u_0$ . Per trovare l'equilibrio di un sistema si risolve l'equazione

$$f(x, u_0) = 0$$

trovando  $x_0$ , per poi sostituire  $x_0$  e  $u_0$  nella variabile d'uscita

$$y_0 = g(x_0, u_0)$$

### 8.2.2 Rappresentazione con Matrici

Un sistema dinamico monovariabile (SISO) con  $n$  variabili di stato  $x_1, x_2, \dots, x_n$  è rappresentato (nel caso generale) come segue

$$\begin{cases} \dot{x}_1 = a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n + b_1u \\ \dot{x}_2 = a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n + b_1u \\ \vdots \\ \dot{x}_n = a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n + b_1u \\ y = c_1x_1 + c_2x_2 + \dots + c_nx_n + du \end{cases}$$

L'uscita è una combinazione lineare di variabili di stato e dell'ingresso (chiameremo anche "controllo" la variabile di ingresso  $u$ ). Per comodità si può rappresentare il sistema in forma matriciale

$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \in \text{Mat}(n \times n) \quad B = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

$$C = [c_1 \quad \dots \quad c_n] \quad D = d \in \mathbb{R}$$

Il sistema si scrive in forma compatta

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

L'equilibrio si pone con  $u(t) = u_0$

$$Ax + Bu_0 = 0 \implies Ax = -Bu_0$$

Se il determinante della matrice  $A$  è diverso da zero, allora esiste un unico stato di equilibrio

$$\det(A) \neq 0 \implies x_0 = -A^{-1}Bu_0$$

Quindi, sostituendo l'ingresso costante e lo stato di equilibrio all'uscita si ottiene l'unica uscita costante  $y_0$

$$y_0 = Cx_0 + Du_0 = C(-A^{-1})Bu_0 + Du_0 = (-CA^{-1}B + D)u_0 = \mu u_0$$

Il termine  $\mu = (-CA^{-1}B + D)$  è detto **guadagno statico**. È un numero reale in quanto

- $-CA^{-1}$  è il prodotto fra una matrice  $n \times 1$  con una matrice  $n \times n$ , quindi è una matrice  $n \times 1$
- $-CA^{-1}B$  è il prodotto di una matrice  $n \times 1$  con una matrice  $1 \times n$ , è quindi uno scalare.
- $-CA^{-1}B \cdot D$  è il prodotto di due scalari



Se il determinante della matrice  $A$  è uguale a 0, allora il sistema ricade in uno dei seguenti casi

- esistono infiniti stati ed uscite di equilibrio
- non esistono stati o uscite di equilibrio

Un sistema SISO di ordine 1 (caso scalare) ammette una soluzione nota, il sistema

$$\begin{cases} \dot{x}(t) = ax(t) + bu(t) \\ y(t) = cx(t) + du(t) \end{cases}$$

Dato  $x(0) = x_0$  e  $t \geq 0$  ha variabile di stato

$$x(t) = x_0 \cdot e^{at} + \int_0^t e^{(at-\tau)} bu(\tau) d\tau$$

Nel caso generale di un sistema SISO di ordine  $n$ , ossia

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \quad \text{con} \quad \begin{array}{ll} A \in \text{mat}(n \times n) & B \in \text{mat}(n \times 1) \\ C \in \text{mat}(1 \times n) & D \in \mathbb{R} \end{array}$$

Dato  $x(0) = \bar{x}_0 \in \text{mat}(n \times 1)$  e per  $t \geq 0$  il vettore di stato vale

$$x(t) = \bar{x}_0 \cdot e^{At} + \int_0^t e^{(At-\tau)} Bu(\tau) d\tau$$

Può sembrare complicato, analizziamo i singoli termini della formula

- $A \cdot t$  è il prodotto di uno scalare per una matrice  $n \times n$ , ogni entrata della matrice sarà moltiplicata alla variabile  $t$ , è quindi a sua volta una matrice  $n \times n$ .
- l'esponenziale di una matrice verrà definito a breve, se  $A$  è una matrice  $n \times n$ , allora anche  $e^A$  lo è.
- $Bu(t)$  è il prodotto di una matrice  $n \times 1$  con uno scalare  $u(t)$ , quindi  $B$  vedrà ogni entrata moltiplicata per  $u(t)$ .  $Bu(t)$  è una matrice  $n \times 1$ .
- $e^{(At-\tau)}$  è una matrice  $n \times n$  e viene moltiplicata per  $Bu(t)$  che è una matrice  $n \times 1$ , quindi  $e^{(At-\tau)} \cdot Bu(t)$  sarà una matrice  $n \times 1$ . Analogamente, anche  $\bar{x}_0 \cdot e^{At}$  lo sarà.
- $x(t)$  sarà quindi una somma di due matrici  $n \times 1$ .

**Definizione (Matrice esponenziale) :** Sia  $A$  una matrice, si definisce la *matrice esponenziale*  $e^A$  come segue

$$e^A = \sum_{i=0}^{\infty} \frac{A^k}{k!} = I + \frac{A^2}{2} + \frac{A^3}{6} + \dots$$

dove  $I = A^0$  è la matrice identità (tutte le entrate sono 1).

Nella formula compare il termine  $e^{At}$ , la sua espressione segue dalla definizione appena data

$$e^{At} = \sum_{i=0}^{\infty} \frac{A^k t^k}{k!} = I + \frac{A^2 t^2}{2} + \frac{A^3 t^3}{6} + \dots$$

L'evoluzione dello stato può essere rappresentata anche come segue

$$x(t) = x_l(t) + x_f(t)$$

dove

$$x_l(t) = \bar{x}_0 \cdot e^{At}$$

$$x_f(t) = \int_0^t e^{(At-\tau)} Bu(\tau) d\tau$$

Il termine  $x_l(t)$  è detto **movimento libero** e rappresenta l'evoluzione "autonoma" del sistema quando l'ingresso  $u(t)$  è nullo. Dipende linearmente da  $\bar{x}_0$ .

Il termine  $x_f(t)$  è detto **movimento forzato** e rappresenta il comportamento del sistema influenzato dall'ingresso  $u(t)$ , da cui appunto, dipende linearmente.

Lo stesso concetto si applica per l'uscita del sistema  $y$ , questa infatti può essere un *uscita libera* (l'ingresso è nulla) o un *uscita forzata*.

$$y(t) = y_l(t) + y_f(t)$$

$$y(t) = \bar{C}x_0 \cdot e^{At} + \int_0^t C e^{(At-\tau)} Bu(\tau) d\tau + Du(t)$$

$$y_l(t) = \bar{C}x_0 \cdot e^{At}$$

$$y_f(t) = \int_0^t C e^{(At-\tau)} Bu(\tau) d\tau + Du(t)$$

Ovviamente i sistemi possono avere anche più ingressi e più uscite, nel caso generale di un sistema *MIMO* di ordine  $n$  con  $m$  ingressi e  $p$  uscite si ha

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$$

dove

$$\begin{aligned} x &\in \mathbb{R}^n \\ u &\in \mathbb{R}^m \\ y &\in \mathbb{R}^p \end{aligned}$$

e

$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \in Mat(n \times n) \quad B = \begin{bmatrix} b_{11} & \dots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{n1} & \dots & b_{nm} \end{bmatrix} \in Mat(n \times m)$$

$$C = \begin{bmatrix} c_{11} & \dots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{p1} & \dots & c_{pn} \end{bmatrix} \in Mat(p \times n) \quad D = \begin{bmatrix} d_{11} & \dots & d_{1m} \\ \vdots & \ddots & \vdots \\ d_{p1} & \dots & d_{pm} \end{bmatrix} \in Mat(p \times m)$$

~~\*~ ~~~\*~ ~~~\*~ ~~~\*~ ~~~\*~ ~~~\*~ ~~~\*~ ~~~\*~ ~~~\*~ ~~~\*~

## 8.3 Stabilità di un Sistema

La *stabilità* è una proprietà associata ai sistemi dinamici, in particolare ci occuperemo di

- stabilità dell'equilibrio
- stabilità del sistema

La stabilità dell'equilibrio è una proprietà che possono avere gli stati di equilibrio di un sistema. Informalmente, Uno stato di equilibrio si dice stabile se, dopo una piccola perturbazione, il sistema tende spontaneamente a ritornare alla sua condizione iniziale. In altre parole, le piccole deviazioni dall'equilibrio vengono corrette automaticamente.

**Definizione (Stabilità) :** Sia  $x_0$  uno stato di equilibrio, esso è stabile se e solo se

$$\forall \varepsilon > 0 \exists \delta(\varepsilon) > 0 \text{ tale che} \\ \|x(0) - x_0\| \leq \delta(\varepsilon) \implies \|x(t) - x_0\| \leq \varepsilon, \forall t \geq 0$$

$\delta(\varepsilon)$  è un intorno più piccolo di  $\varepsilon$ .

Per ogni valore infinitesimale  $\varepsilon$ , se  $x(0) = x(t_0)$  (il vettore di stato nell'istante iniziale) ha una distanza  $\delta(\varepsilon) < \varepsilon$  da un punto di equilibrio  $x_0$ , allora la distanza fra ogni possibile valore dello stato  $x(t)$   $\forall t \geq 0$  avrà una distanza minore o uguale ad  $\varepsilon$  dal punto di equilibrio.

In termini più qualitativi, un punto di equilibrio è localmente stabile se qualunque evoluzione che nasce da uno stato iniziale vicino rimane vicina per tutti gli istanti successivi, comunque si scelga il massimo scostamento ammesso dall'equilibrio per tutta l'evoluzione del sistema nel tempo. A perturbazioni limitate corrispondono variazioni limitate dell'evoluzione.

vettore di stato :  $x = (x_1, x_2)$

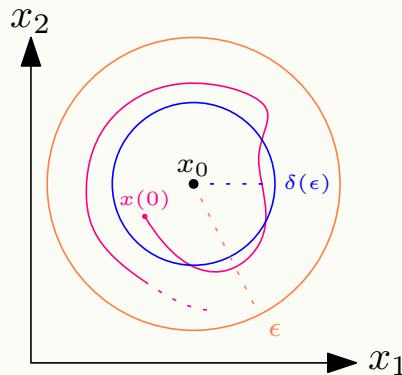


Figura 8.4: vettore di stato in un sistema stabile di ordine 2

Se la condizione non è verificata, il punto di equilibrio  $x_0$  si definisce **instabile**. La distanza iniziale fra  $x(0)$  ed il punto di equilibrio  $x_0$  rappresenta la *perturbazione*.

Un punto di equilibrio può essere anche **asintoticamente stabile** se, si verifica la condizione di stabilità, ed in più è anche vero che al passare del tempo l'evoluzione dello stato converge al punto di equilibrio.

$$\exists \delta_a > 0 \mid \|x(0) - x_0\| < \delta_a \implies \lim_{t \rightarrow +\infty} \|x(t) - x_0\| = 0$$

Le condizioni introdotte devono essere verificate per un qualsiasi valore iniziale  $x(0)$ , ed una volta definito l'intorno di  $\delta(\varepsilon)$  (o  $\delta_a$  nel caso asintotico), qualunque scelta di  $x(0)$  (ossia, qualunque perturbazione) deve comportare le condizioni di soddisfacimento.

La stabilità non è un valore correlato ad uno stato di equilibrio ma è una caratteristica del sistema nella sua globalità

Il sistema è in una **condizione di equilibrio** se, dato un ingresso costante  $u(t) = u_0$  ed uno stato di equilibrio  $x_0$ , all'istante iniziale non riceve perturbazione, ossia il valore del vettore di stato all'inizio è già quello dello stato di equilibrio.

$$\begin{aligned} x_0 &\text{ è uno stato di equilibrio} \\ x(0) &= x_0 \\ u(t) = u_0 &\text{ è l'ingresso che causa lo stato di equilibrio} \end{aligned}$$

vettore di stato :  $x = (x_1, x_2)$

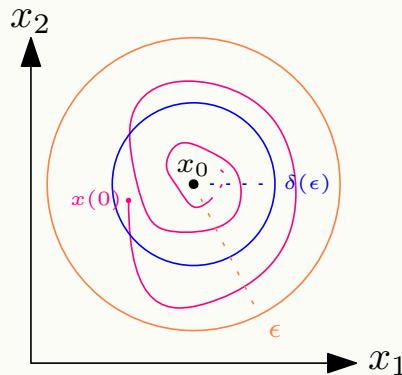


Figura 8.5: vettore di stato in un sistema stabile asintoticamente di ordine 2

In tal caso il sistema *stabile*, non essendo perturbato rimane nello stato di equilibrio

$$x(t) = x_0 \cdot e^{At} + \int_0^t e^{(A(t-\tau))} Bu_0 d\tau = x_0 \quad \forall t \geq 0$$

Il sistema parte in **condizioni di perturbazione** quando lo stato iniziale  $x(0)$  dista di un certo valore  $\delta x$  dallo stato di equilibrio  $x_0$

$$x(0) = x_0 + \delta x$$

Il movimento è perturbato e si avrà che  $x(t) \neq x_0 \quad \forall t \geq 0$

$$\begin{aligned} x(t) &= e^{At}(x_0 + \delta x) + \int_0^t e^{A(t-\tau)} Bu_0 d\tau = \\ &e^{At}\delta x + e^{At}x_0 + \int_0^t e^{A(t-\tau)} Bu_0 d\tau = \end{aligned}$$

i due termini a destra della somma sono quelli della condizione di equilibrio :

$$e^{At}x_0 + \int_0^t e^{A(t-\tau)} Bu_0 d\tau = x_0$$

quindi si avrà che

$$x(t) = e^{At}\delta x + x_0$$

$$\Leftrightarrow x(t) - x_0 = e^{At}\delta x$$

Quest'ultima equazione rende chiaro il concetto che *la stabilità dipende da  $e^{At}$* .

- **stabilità** : se  $e^{At}$  è limitata, allora  $\lim_{t \rightarrow \infty} x(t) - x_0$  sarà un valore finito, quindi l'evoluzione dello stato non si scosta di più di un certo valore dallo stato di equilibrio.
- **stabilità asintotica** : se  $\lim_{t \rightarrow \infty} e^{At} = 0$ , allora  $\lim_{t \rightarrow \infty} x(t) - x_0 = 0$ , quindi l'evoluzione dello stato tenderà ad avvicinarsi sempre di più al valore di equilibrio.
- **instabilità** : se  $e^{At}$  diverge, allora per  $t \rightarrow \infty$  lo stato del sistema si allontanerà sempre di più dallo stato di equilibrio.

### Proprietà dei sistemi asintoticamente stabili

1. Un sistema asintoticamente stabile, se spostato dallo stato di equilibrio (perturbazione), tende a tornarci.
2. Ha un *unico stato di equilibrio* ed il determinante di  $A$  è 0.
3. Il movimento libero del sistema tende a zero e si annulla (dato che la matrice  $e^{At}$  tende a zero) quindi il movimento dello stato dipende *asintoticamente* solo dall'ingresso  $u(t)$

$$\lim_{t \rightarrow \infty} x(t) = x_f(t)$$

4. Se l'ingresso è nullo  $u(t) = 0$ , allora, per un qualsiasi stato iniziale  $x(0)$

$$\lim_{t \rightarrow \infty} x(t) = \lim_{t \rightarrow \infty} x(0) \cdot e^{At} + \int_0^t e^{(At-\tau)} B 0 d\tau = \quad (8.11)$$

$$\lim_{t \rightarrow \infty} x(0) \cdot e^{At} = 0 \quad (8.12)$$

Quindi

$$\lim_{t \rightarrow \infty} x(t) = 0$$

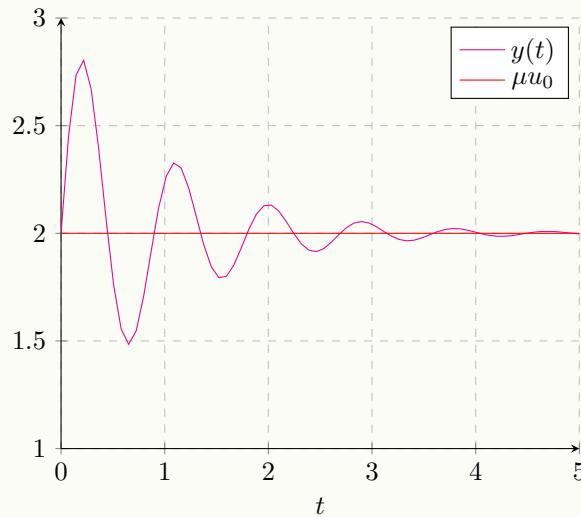
$$\lim_{t \rightarrow \infty} y(t) = 0$$

La condizione si verifica anche se  $u$  è del tipo  $u(t) = \begin{cases} \text{qualsiasi se } 0 \leq t < t^* \\ 0 \text{ se } t \geq t^* \end{cases}$

5. Se l'ingresso è costante allora  $u(t) = u_0$  allora l'uscita  $y(t)$  tende a

$$y_0 = \mu u_0 = (-CA^{-1}B + D)u_0$$

$$\lim_{t \rightarrow \infty} y(t) = \mu u_0$$



6. Se l'ingresso  $u(t)$  è limitato, allora anche l'uscita  $y(t)$  è limitata.

$$\|u(t)\| < K \implies \exists H \mid \|y(t)\| < H$$

ta proprietà è detta *stabilità esterna* e tutti i sistemi asintoticamente stabili ne godono. Un sistema però, può godere della proprietà di stabilità esterna anche senza essere asintoticamente stabile

$$\begin{aligned} \text{stabilità asintotica} &\implies \text{stabilità esterna} \\ \text{stabilità esterna} &\not\implies \text{stabilità asintotica} \end{aligned}$$

### 8.3.1 Studio di $e^{At}$ per la Stabilità

Consideriamo il generico sistema dinamico

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$$

Le proprietà della matrice  $A$  decretano la stabilità del sistema. Si individuano 4 casi distinti.

**Caso 1 :  $A$  diagonale**

Supponiamo che  $A$  sia una matrice diagonale

$$A = \begin{bmatrix} s_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & s_n \end{bmatrix}$$

I termini  $s_1, s_2 \dots, s_n$  sono gli autovalori di  $A$ , allora

$$e^{At} = \begin{bmatrix} e^{s_1 t} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & e^{s_n t} \end{bmatrix}$$

Allora

- $\forall i, s_i \leq 0$  se tutte le entrate della matrice sono  $e^{s_i t}$  ed  $s_i$  è minore o uguale a zero, allora per  $t \rightarrow \infty$  ogni termine  $e^{s_i t}$  tenderà ad un valore finito, quindi il sistema è **stabile**.
- $\forall i, s_i < 0$  se tutte le entrate della matrice sono  $e^{s_i t}$  ed  $s_i$  è strettamente minore di 0, allora ogni termine della matrice tenderà a zero, quindi  $A$  tenderà a zero, allora il sistema è **asintoticamente stabile**.
- $\exists i | s_i > 0$  se un singolo termine  $s_i$  è maggiore di zero, allora  $e^{s_i t}$  tenderà ad infinito, la matrice non è limitata quindi il sistema è **instabile**.

**Caso 2 :  $A$  ha autovalori reali distinti**

Siano  $s_1, s_2 \dots, s_n$  gli autovalori di  $A$ , sono tutti distinti quindi

$$s_1 \neq s_2 \neq \dots \neq s_n$$

Sappiamo allora che esiste una matrice  $M$  per cui

$$A = M \tilde{A} M^{-1} \quad \text{dove} \quad \tilde{A} = \begin{bmatrix} s_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & s_n \end{bmatrix}$$

Allora si studia l'andamento della matrice  $M \tilde{A} M^{-1}$ .

**Esempio :** Si ha il sistema

$$\begin{cases} \dot{x}_1 = -2x_1 + 6x_2 \\ \dot{x}_2 = -2x_1 + 5x_2 \end{cases} \quad A = \begin{bmatrix} -2 & 6 \\ -2 & 5 \end{bmatrix}$$

trovo gli autovalori

$$P(\lambda) = \det(sI - A) = \det \begin{bmatrix} s+2 & -6 \\ 2 & s-5 \end{bmatrix} = (s+2)(s-5) + 12 = (s-2)(s-1) \implies \begin{cases} s_1 = 1 \\ s_2 = 2 \end{cases}$$

Bisogna trovare la matrice  $M$ , si ricordi essere la matrice le cui righe sono gli autovettori  $v, u$  di  $A$

$$\begin{aligned} \begin{bmatrix} -2 & 6 \\ -2 & 5 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = s_1 \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} &\implies \begin{cases} -2v_1 + 6v_2 = v_1 \\ -2v_1 + 5v_2 = v_2 \end{cases} \implies 2v_2 = v_1 \implies v = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \\ \begin{bmatrix} -2 & 6 \\ -2 & 5 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = s_2 \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} &\implies \begin{cases} -2u_1 + 6u_2 = 2u_1 \\ -2u_1 + 5u_2 = 2u_2 \end{cases} \implies u_1 = 3/2u_2 \implies u = \begin{bmatrix} 3 \\ 2 \end{bmatrix} \\ M = \begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix} & M^{-1} = \begin{bmatrix} 2 & -3 \\ -1 & 2 \end{bmatrix} \\ \hat{A} = M^{-1} A M = \begin{bmatrix} 2 & -3 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} -2 & 6 \\ -2 & 5 \end{bmatrix} \begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \end{aligned}$$

quindi

$$e^{At} = M e^{\hat{A}t} M^{-1} = \begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} e^t & 0 \\ 0 & e^{2t} \end{bmatrix} \begin{bmatrix} 2 & -3 \\ -1 & 2 \end{bmatrix} = \begin{bmatrix} 4e^t - 3e^{2t} & -6e^t + 6e^{2t} \\ 2e^t - 2e^{2t} & -3e^t + 4e^{2t} \end{bmatrix}$$

Il sistema è instabile.

**Caso 3 :  $A$  ha autovalori complessi distinti**

Consideriamo il caso in cui  $A$  è una matrice  $2 \times 2$ , i suoi autovalori  $s_1, s_2$  sono del tipo

$$s_1 = \sigma + j\omega \quad s_2 = \sigma - j\omega$$

Si ricordi che

se una matrice quadrata a valori reali ha autovalori complessi, allora essi compaiono come coppie di complessi coniugati.

In tal caso, ci sono 3 possibili casi

1.  $\operatorname{Re}(s_i) \leq 0, \forall i$  Se ogni autovalore ha parte reale minore o uguale a zero, il sistema è stabile.
2.  $\operatorname{Re}(s_i) < 0, \forall i$  Se ogni autovalore ha parte reale strettamente minore di zero, il sistema è asintoticamente stabile.
3.  $\exists i | \operatorname{Re}(s_i) > 0$  se esiste un solo autovalore con parte reale positiva il sistema è instabile.

**Caso 4 :  $A$  ha autovalori multipli**

Supponiamo che  $A$  abbia degli autovalori multipli, ossia, esistono almeno due autovalori identici, prendiamo in esame una matrice diagonale

$$A = \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix} \quad s_1 = s_2 = \alpha$$

L'andamento di  $e^{At}$  dipende da  $\alpha$ , consideriamo una matrice non diagonalizzabile

$$A = \begin{bmatrix} \alpha & 1 \\ 0 & \alpha \end{bmatrix} \quad s_1 = s_2 = \alpha$$

Allora

$$e^{At} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} \alpha & 1 \\ 0 & \alpha \end{bmatrix} t + \begin{bmatrix} \alpha^2 & 2\alpha \\ 0 & \alpha^2 \end{bmatrix} \frac{t^2}{2!} + \dots$$

La matrice conterrà termini del tipo  $e^{\alpha t}$  o  $te^{\alpha t}$ , quindi la stabilità dipenderà da  $\alpha$

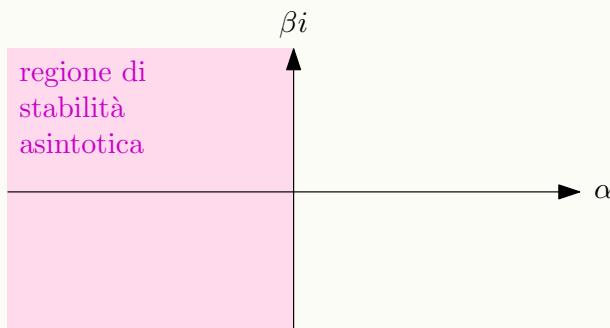
1.  $\alpha = 0$  instabilità
2.  $\alpha < 0$  stabile asintoticamente
3.  $\alpha > 0$  instabilità

Generalizzando ad una matrice  $n \times n$ , ci saranno autovalori multipli con diverse molteplicità, e la matrice conterrà termini del tipo

$$t^k e^{sk}$$

dove  $k$  dipende dalla molteplicità di  $s$ , anche in questo caso

- $\operatorname{Re}(s_i) < 0, \forall i$  Se ogni autovalore ha parte reale strettamente minore di zero, il sistema è asintoticamente stabile.
- $\exists i | \operatorname{Re}(s_i) > 0$  se esiste un solo autovalore con parte reale positiva il sistema è instabile.
- nel caso gli autovalori hanno tutti parte reale minore di zero, ma ce ne sta uno con parte reale uguale a zero, il sistema potrebbe essere stabile (non asintoticamente) oppure instabile.





Esistono anche altri criteri per lo studio della stabilità basati sullo studio della matrice  $A$

- **criterio 1** : Se  $A$  è triangolare il sistema è stabile asintoticamente
- **criterio 2** : Se la traccia di  $A$  è negativa il sistema è stabile asintoticamente, se invece è positiva il sistema è instabile.
- **criterio 3** : Se il determinante di  $A$  è diverso da zero allora il sistema è asintoticamente stabile.

In generale, se  $A$  ha tutti gli autovalori con parte reale strettamente negativa, il sistema è asintoticamente stabile.

### Criterio di Routh

Il criterio esposto in questa sezione permette lo studio della stabilità di un sistema tramite la costruzione di una apposita tabella.



## 8.4 Raggiungibilità e Osservabilità

**Definizione (raggiungibilità) :** Con *raggiungibilità* si definisce la proprietà di un sistema dinamico di poter assumere un prefissato valore agendo sull'ingresso.

Si consideri un generico sistema lineare, uno stato  $\tilde{x}$  si dirà *raggiungibile* se esiste un ingresso  $\tilde{u}(t)$  tale che il movimento forzato del sistema con tale ingresso risulti proprio  $\tilde{x}$ . La raggiungibilità dipende esclusivamente dall'equazione di stato e non ha nulla a che vedere con l'uscita  $y$ .

Dato un sistema

$$\dot{x} = Ax + Bu$$

Sia  $n$  l'ordine (quindi l'ordine di  $A$ ), la **matrice di raggiungibilità** associata al sistema è

$$R = [B \ AB \ A^2B \ A^3B \ \dots \ A^{n-2}B \ A^{n-1}B]$$

Se la matrice  $B$  ha dimensione  $n \times m$  (dove  $m$  è il numero di ingressi), La matrice di raggiungibilità  $R$  avrà dimensione  $n \times nm$ .  $R$  è la matrice la cui immagine è l'insieme di tutti gli stati raggiungibili, definiamo quindi  $X^R$  lo *spazio di raggiungibilità*.

$$X^R = \text{Im}(R)$$

Il sistema si dice **completamente raggiungibile** se il rango di  $R$  è uguale ad  $n$ , in tal caso, ogni possibile stato del sistema è raggiungibile.

$$\forall \tilde{x} \in \mathbb{R}^n \ \exists \tilde{u} \text{ t.c. } x(t) = x_0 \cdot e^{at} + \int_0^t e^{(at-\tau)} b \tilde{u}(\tau) d\tau = \tilde{x}$$

Se  $m = 1 \implies R$  è quadrata, sarà sufficiente che  $\det(R) \neq 0$ . Se il sistema non è completamente raggiungibile, il rango di  $R$  (denotato  $n_r$ ) sarà inferiore ad  $n$ .

$$R = \begin{bmatrix} n & B \\ & \vdots \\ & n & AB \\ & & n & A^2B \\ & & & \ddots \\ & & & n & A^{n-1}B \end{bmatrix}$$

Quando un sistema non è completamente raggiungibile, è possibile *isolare* la sua componente raggiungibile. Bisogna operare un *cambio di variabili* di stato per mezzo di un'apposita matrice di trasformazione  $T_r$ , si definisce

$$\hat{x}(t) = T_r x(t)$$

L'equazione del movimento dello stato si modificherà

$$\hat{\dot{x}} = \hat{A}\hat{x} + \hat{B}u$$

Quindi verranno modificate anche le matrici  $A$  e  $B$ . Come saranno fatte queste nuove matrici?

$$\begin{aligned} \hat{A} &= T_r A T_r^{-1} \\ \hat{B} &= T_r B \end{aligned}$$

La matrice  $T_r$  deve essere invertibile, dovrà avere dimensione  $n \times n$ . Le prime  $n_r$  colonne di  $T_r$  saranno composte da  $n_r$  vettori linearmente indipendenti della matrice di raggiungibilità  $R$ , le restanti colonne saranno un completamento, ossia la matrice  $T_r$  deve avere le colonne tutte linearmente indipendenti (il determinante deve essere diverso da zero).

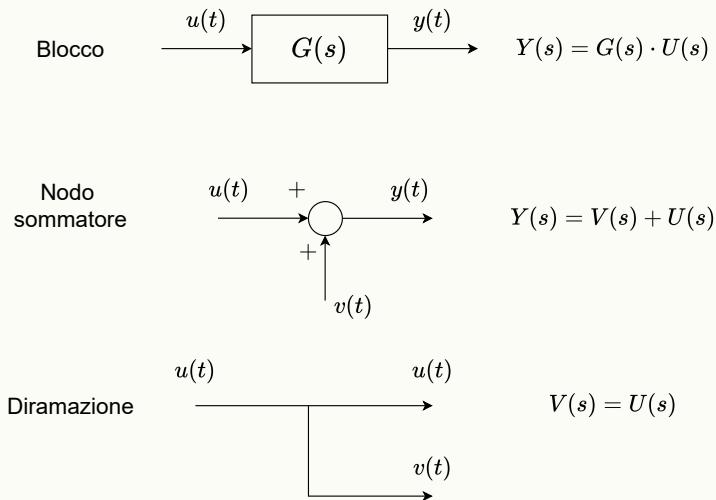
$$T_r = \begin{bmatrix} n & \begin{array}{|c|} \hline n_r \\ \hline \text{colonne} \\ \text{linearmente} \\ \text{indipendenti} \\ \text{prese da } R \\ \hline \end{array} & n & \begin{array}{|c|} \hline n - n_r \\ \hline \text{completamento} \\ \hline \end{array} \end{bmatrix}$$

Continuare

.....

## 8.5 Schemi a Blocchi

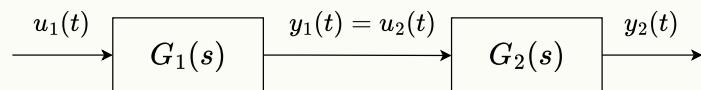
Gli schemi a blocchi costituiscono una rappresentazione grafica e schematica dei sistemi dinamici costituiti da più sistemi interconnessi fra loro, mettendone in relazione le interazioni fra essi. Durante la sezione, se  $x(t)$  è una generica funzione nel tempo, rappresenteremo con  $X(s)$  la sua trasformata di Laplace (lettera maiuscola e variabile complessa  $s$ ). Gli elementi di base degli schemi a blocchi sono i seguenti



In base alla disposizione degli elementi negli schemi a blocchi esistono diverse regole di elaborazione al fine di manipolare e trovare la funzione di trasferimento globale del sistema.

### Blocchi in Serie

Quando sono presenti due blocchi in serie :



L'uscita globale del sistema segue la seguente formulazione

$$Y_2(s) = G_2(s)U_2(s) = G_2(s)Y_1(s) = G_2(s)G_1(s)U_1(s)$$

La funzione di trasferimento *globale* del sistema è data dal prodotto delle due funzioni di trasferimento

$$\begin{aligned} G(s) &= G_1(s)G_2(s) \\ Y_2(s) &= U(s)G(s) \end{aligned}$$

### Blocchi in Parallello

Quando sono presenti due blocchi in parallelo, come mostrato in figura 8.6, l'uscita globale del sistema segue la seguente formulazione

$$Y(S) = Y_1(s) + Y_2(s) = U(s)G_1(s) + U(s)G_2(s) = U(s)[G_1(s) + G_2(s)]$$

### Blocchi in Retroazione

Uno schema a *retroazione*, anche detto ad *anello chiuso* è fondamentale nella teoria dei controlli automatici, modella l'azione del misurare un errore presente fra un valore che si vuole far assumere ad una variabile (riferimento) ed il suo valore effettivo (uscita del sistema).

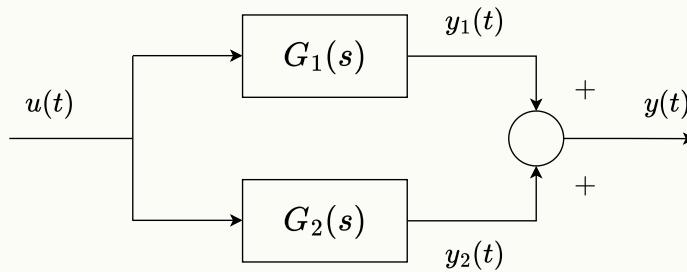


Figura 8.6: blocchi in parallelo

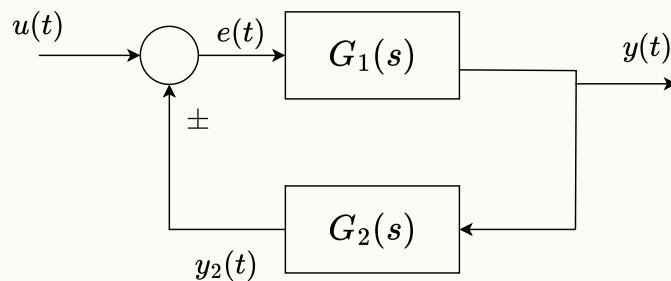


Figura 8.7: blocchi in retroazione

Si osservi la figura 8.7, la funzione di trasferimento globale è formulata come segue

$$Y(s) = E(s)G_1(s) = G_1(s)[U(s) \pm Y_2(s)] = G_1(s)[U(s) \pm G_2(s)Y(s)] \implies$$

$$Y(s)[1 \pm G_1(s)G_2(s)] = G_1(s)U(s) \implies$$

$$Y(s) = \frac{G_1(s)}{1 \pm G_1(s)G_2(s)}U(s)$$

### Regole di Elaborazione

In questa sezione segue una lista di regole di elaborazione degli schemi a blocchi, che ne modificano il diagramma lasciandolo equivalente a quello originale.

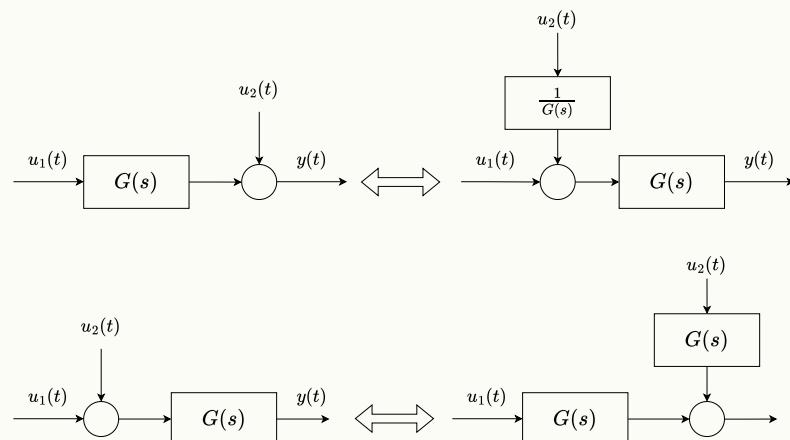


Figura 8.8: regole sullo spostamento del nodo sommatore

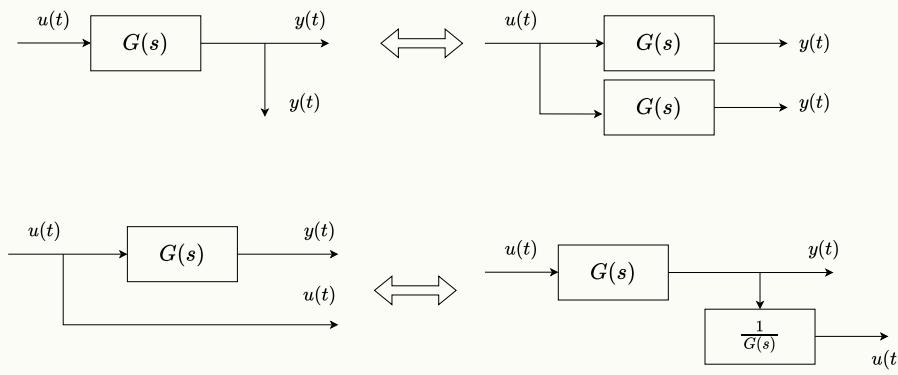


Figura 8.9: regole sullo spostamento della diramazione

La rielaborazione dello schema è lecita soltanto ai fini del calcolo della funzione di trasferimento complessiva.

### Analisi della Stabilità nei Sistemi Interconnessi

Si consideri un sistema composto da due blocchi in serie con le funzioni di trasferimento

$$G_1(s) = \frac{N_1(s)}{D_1(s)} \quad G_2(s) = \frac{N_2(s)}{D_2(s)}$$

La funzione di trasferimento complessiva sappiamo essere

$$G(s) = G_1(s)G_2(s) = \frac{N_1(s)N_2(s)}{D_1(s)D_2(s)} = \frac{N(s)}{D(s)}$$

In generale i poli della funzione  $G$  è l'insieme dei poli di entrambe le funzioni, ciò implica che, se entrambe le funzioni  $G_1, G_2$  sono asintoticamente stabili (la parte reale di ogni polo è negativa), allora anche  $G$  sarà asintoticamente stabile.

Può accadere però, che vi siano delle cancellazioni, se ad esempio ci sono fattori in comune fra  $N_1(s), D_2(s)$  oppure fra  $N_2(s), D_1(s)$

esempio

$$\begin{aligned} G_1(s) &= \frac{s+3}{(s+1)s} & \Rightarrow G(s) &= \frac{(s+3)}{(s+3)(s+\pi)(s+1)s} = \frac{1}{(s+\pi)(s+1)s} \\ G_2(s) &= \frac{1}{(s+3)(s+\pi)} \end{aligned}$$

Anche se  $s_1 = -3$  era il polo di una delle due funzioni, non lo è di quella complessiva

In generale, la dinamica del sistema complessivo è "nascosta", se un polo con parte reale positiva è stato cancellato, il sistema è comunque non asintoticamente stabile anche se la funzione complessiva non presenta poli con parte reale positiva.

- Se vengono cancellati poli con parte reale positiva il sistema non è asintoticamente stabile
- Se vengono cancellati solo poli con parte reale negativa e  $G(s)$  non ha poli con parte reale positiva il sistema è asintoticamente stabile

$$\begin{array}{ccc} u(t) & \xrightarrow{\frac{1}{s-1}} & \xrightarrow{\frac{s-1}{s+2}} y(t) \end{array} \quad G(S) = \frac{s-1}{(s-1)(s+2)} = \frac{1}{s+2}$$

Il sistema è instabile anche se  $G(s)$  mostra una dinamica asintoticamente stabile.

In conclusione, il sistema complessivo è stabile se tutti i sotto sistemi sono stabili, gli autovalori del sistema complessivo sono l'unione degli autovalori dei sottosistemi.

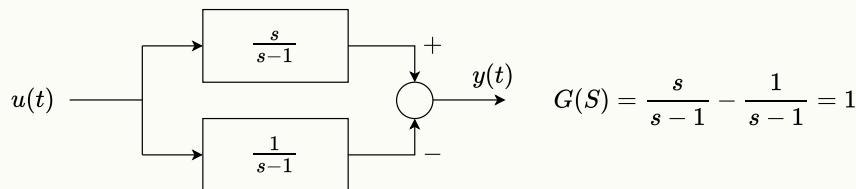
Si consideri un sistema composto da due blocchi in serie con le funzioni di trasferimento

$$G_1(s) = \frac{N_1(s)}{D_1(s)} \quad G_2(s) = \frac{N_2(s)}{D_2(s)}$$

La funzione di trasferimento complessiva sappiamo essere

$$G(s) = G_1(s) + G_2(s) = \frac{N_1(s)D_2(s) + N_2(s)D_1(s)}{D_1(s)D_2(s)} = \frac{N(s)}{D(s)}$$

Le radici del polinomio  $D_1(s)D_2(s)$  sono l'unione delle radici dei singoli polinomi, è quindi vero che i poli di  $G(s)$  sono l'unione dei poli di  $G_1(s)$  e  $G_2(s)$ . Anche in questo caso però possono verificarsi delle situazioni di instabilità nascosta.



Il sistema è instabile anche se  $G(s)$  mostra una dinamica asintoticamente stabile.

In conclusione, proprio come per le interconnessioni in serie, il sistema complessivo è stabile se tutti i sotto sistemi sono stabili, gli autovalori del sistema complessivo sono l'unione degli autovalori dei sottosistemi.

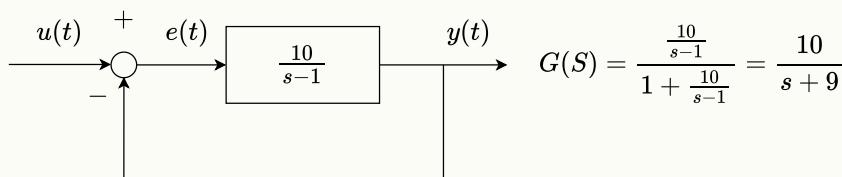
Passiamo infine al caso della connessione in retroazione, date le funzioni di trasferimento

$$G_1(s) = \frac{N_1(s)}{D_1(s)} \quad G_2(s) = \frac{N_2(s)}{D_2(s)}$$

La funzione di trasferimento complessiva sappiamo essere

$$G(s) = \frac{G_1(s)}{1 + G_1(s)G_2(s)} = \frac{\frac{N_1(s)}{D_1(s)}}{1 + \frac{N_1(s)N_2(s)}{D_1(s)D_2(s)}} = \frac{N_1(s)D_2(s)}{D_1(s)D_2(s) + N_1(s)N_2(s)}$$

I poli della funzione complessiva  $G(s)$  sono le radici del polinomio  $D_1(s)D_2(s) + N_1(s)N_2(s)$ , non è quindi più vero che i poli di  $G(s)$  sono l'unione dei poli di  $G_1(s)$  e  $G_2(s)$ . La stabilità asintotica dei sottosistemi non implica la stabilità asintotica del sistema complessivo, e la stabilità asintotica del sistema complessivo non implica la stabilità asintotica dei sottosistemi.



Il sistema complessivo è asintoticamente stabile  
anche se il sottosistema è instabile

### 8.5.1 Risposta allo Scalino

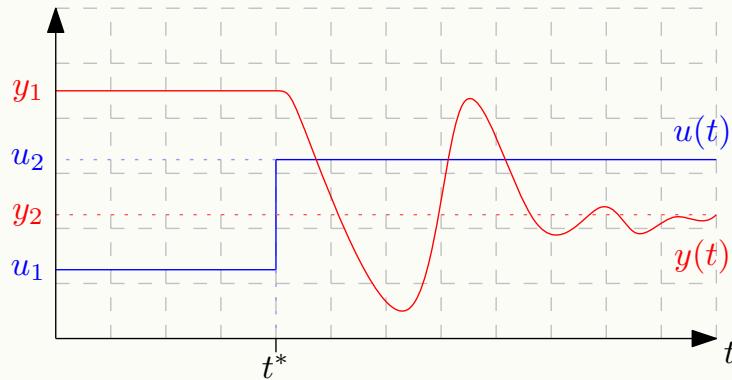
Dato un sistema dinamico, la sua funzione di trasferimento presenta dei parametri che descrivono l'evoluzione dell'uscita del sistema, quando viene dato come ingresso la funzione scalino

$$H(t) = \begin{cases} 1 & \text{se } t \geq 0 \\ 0 & \text{se } t < 0 \end{cases}$$

In generale, la risposta ad un ingresso che varia istantaneamente da un valore costante ad un altro.

$$u(t) = \begin{cases} u_1 & \text{se } t \geq t^* \\ u_2 & \text{se } t < t^* \end{cases}$$

Nei sistemi asintoticamente stabili, la risposta allo scalino descrive una *transizione* da uno stato di equilibrio a un altro (**risposta indiciale**).



Denominiamo l'uscita del sistema a **regime** per  $t \rightarrow \infty$

$$\lim_{t \rightarrow +\infty} y(t) = y(\infty) = \bar{y}$$

Si definisce poi *risposta all'impulso* la risposta del sistema se l'ingresso è la delta di diraq  $\delta$ . E si definisce *risposta alla rampa* la risposta del sistema alla funzione rampa  $u(t) = t$ .

- l'impulso è la derivata dello scalino, quindi la risposta all'impulso è la derivata della risposta allo scalino.
- la rampa è l'integrale dello scalino, la risposta alla rampa è l'integrale della risposta allo scalino.

|             |             |               |                 |
|-------------|-------------|---------------|-----------------|
| funzione    | $\delta(t)$ | $H(t)$        | $H(t) \cdot t$  |
| trasformata | 1           | $\frac{1}{s}$ | $\frac{1}{s^2}$ |

Si considereranno adesso differenti casi di sistemi, in particolare del primo e del secondo ordine, ed in base ai parametri della funzione di trasferimento, essi manifesteranno differenti comportamenti nella risposta.

### Sistemi del primo ordine

**Definizione :** Un sistema dinamico si dice **strettamente proprio** se, la sua funzione di trasferimento ha il numeratore di grado strettamente inferiore al grado del denominatore. Tale proprietà dà informazione sulla velocità di risposta del sistema.

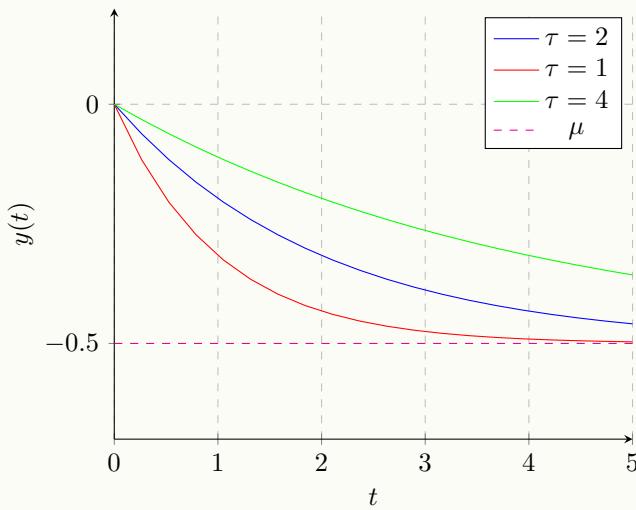
Si consideri un sistema strettamente proprio del primo ordine (una variabile di stato), avrà una funzione di trasferimento della forma

$$G(s) = \frac{\mu}{1 + s\tau}$$

Asintoticamente stabile se  $\tau > 0$ . L'andamento nel tempo di tale sistema quando l'ingresso è lo scalino risulta essere

$$\mathcal{L}^{-1}\left[\frac{1}{s} \frac{\mu}{1 + s\tau}\right] = \mathcal{L}^{-1}\left[\frac{\mu}{1 + s\tau} + \frac{\mu}{s}\right] = \mu(1 - e^{-t/\tau})$$

Definiamo **tempo di assestamento** il tempo necessario al movimento dell'uscita per assestarsi all'interno di un intervallo nella quale continuerà ad oscillare senza uscirne.

Figura 8.10: risposta allo scalino nel caso  $\mu = -0.5$ 

Il polo della funzione è  $-\frac{1}{\tau}$ , più  $\tau$  è piccolo, più il tempo di assestamento sarà minore. Se il valore di  $\tau$  è negativo, il polo sarà positivo quindi non ci sarà assestamento dato che il sistema è instabile (ovviamente  $e^{-\frac{t}{\tau}}$  diverge in questo caso).

Un sistema che *non* è strettamente proprio ha la trasformata di Laplace della forma

$$\mu \frac{1+sT}{1+s\tau}$$

Il suo andamento nel tempo è

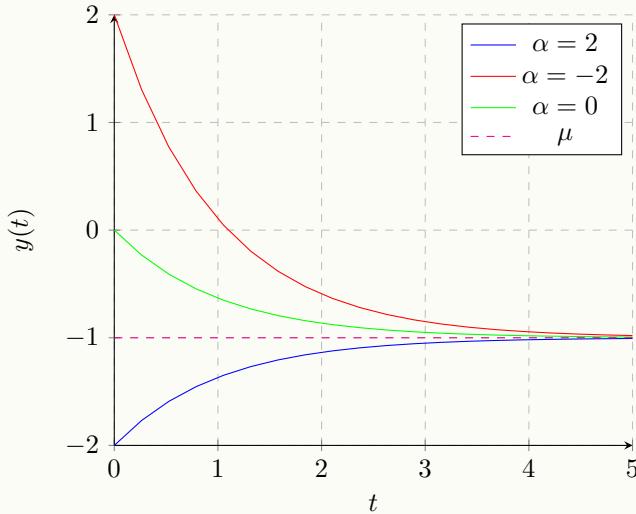
$$\mathcal{L}^{-1}\left[\frac{1}{s}\mu \frac{1+sT}{1+s\tau}\right] = \mathcal{L}^{-1}\left[\frac{\mu}{s} + \frac{\mu(T-\tau)}{1+s\tau}\right] = \mu\left(1 + \frac{T-\tau}{\tau}e^{-\frac{t}{\tau}}\right)$$

Si pone in questo caso il fattore  $\alpha$  tale che

$$\alpha\tau = T$$

Allora il movimento dell'uscita è

$$y(t) = \mu\left[1 + (\alpha - 1)e^{-\frac{t}{\tau}}\right]$$

Figura 8.11: risposta allo scalino nel caso  $\mu = -1$ ,  $\tau = 1$

Lo stato iniziale del sistema è  $\alpha\mu$  differentemente dal caso del sistema strettamente proprio in cui il valore iniziale è 0. La velocità di assestamento dipenderà sia da tale stato iniziale, che dal valore di  $\tau$ .

### Sistemi del secondo ordine

Consideriamo adesso un sistema del secondo ordine (vettore di stato  $x \in \mathbb{R}^2$ ), si vuole dare una descrizione del comportamento dell'uscita in base ad alcuni parametri della sua funzione di trasferimento. In tal caso, indetifichiamo 4 differenti casi, la funzione di trasferimento può avere

- Poli reali e nessuno zero
- Poli reali ed uno zero
- Poli complessi e nessuno zero
- Poli complessi ed uno zero

#### Poli reali e nessuno zero

La funzione di trasferimento è del tipo

$$G(s) = \frac{\mu}{(1+s\tau_1)(1+s\tau_2)}$$

Si ricordi che la condizione di stabilità asintotica si ha quando i poli hanno parte reale negativa, ossia  $\tau_1 > 0 \wedge \tau_2 > 0$ . Per calcolare facilmente l'anti trasformata, occorre l'espansione della funzione in fratti semplici.

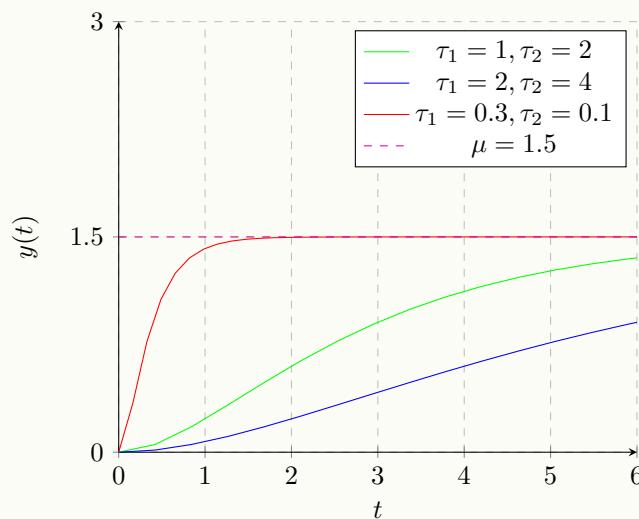
$$\begin{aligned} y(t) &= \mathcal{L}^{-1}[G(s)\frac{1}{s}] = \mathcal{L}^{-1}\left[\frac{\mu}{s(1+s\tau_1)(1+s\tau_2)}\right] \\ \frac{\mu}{s(1+s\tau_1)(1+s\tau_2)} &= \frac{A}{s} + \frac{B}{(1+s\tau_1)} + \frac{C}{(1+s\tau_2)} \end{aligned}$$

quindi

$$s = 0 \implies A = \frac{\mu}{(1+s\tau_1)(1+s\tau_2)} = \mu$$

$$s = -\frac{1}{\tau_1} \implies B = \frac{\mu}{s(1+s\tau_2)} = \frac{\mu}{(-\frac{1}{\tau_1})(1+(-\frac{1}{\tau_1})\tau_2)} = \frac{\mu\tau_1^2}{\tau_2 - \tau_1}$$

$$s = -\frac{1}{\tau_2} \implies C = \frac{\mu}{s(1+s\tau_1)} = \frac{\mu\tau_2^2}{\tau_1 - \tau_2}$$



Allora

$$y(t) = \mathcal{L}^{-1} \left[ \frac{\mu}{s} + \frac{\frac{\mu\tau_1^2}{\tau_2 - \tau_1}}{(1+s\tau_1)} + \frac{\frac{\mu\tau_2^2}{\tau_1 - \tau_2}}{(1+s\tau_2)} \right] = \mu \left( 1 - \frac{\tau_1}{\tau_1 - \tau_2} e^{-t/\tau_1} + \frac{\tau_2}{\tau_1 - \tau_2} e^{-t/\tau_2} \right)$$

Si noti come a regime permanente la risposta tende a  $y(\infty) = \lim_{t \rightarrow \infty} y(t) = \mu$ . Inoltre  $y(0) = \dot{y}(0) = 0$ .

In questo caso in assenza di zeri, i poli più vicini all'asse immaginario sono quelli più influenti sull'andamento della risposta.

### Poli complessi e nessuno zero

La funzione di trasferimento è della forma

$$G(s) = \frac{\varrho}{(s + \sigma + j\omega)(s + \sigma - j\omega)}$$

I poli sono complessi e coniugati, e sono  $-\sigma \pm j\omega$ , è chiaro che si ha stabilità asintotica se  $\sigma > 0$ . Si pone

$$\mu = G(0) = \frac{\varrho}{\sigma^2 + \omega^2}$$

L'uscita del sistema nel dominio di Laplace è

$$Y(s) = G(s) \frac{1}{s} = \frac{\varrho}{s(s + \sigma + j\omega)(s + \sigma - j\omega)}$$

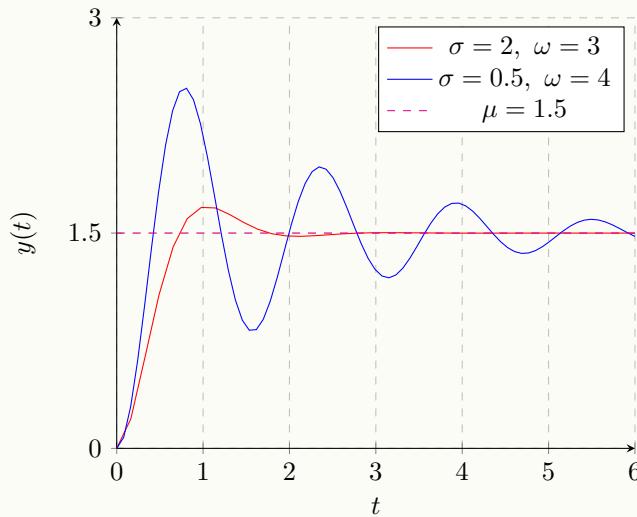
Con una diversa parametrizzazione (della quale non mostreremo il procedimento), si può riscrivere

$$Y(s) = \mu \left[ \frac{1}{s} - \frac{s + \sigma}{(s + \sigma)^2 + \omega^2} - \frac{\sigma}{\omega} \frac{\omega}{(s + \sigma)^2 + \omega^2} \right]$$

Applicando le anti trasformate note si mostra che

$$y(t) = \mathcal{L}^{-1}[Y(s)] = \mu \left[ 1 - e^{-\sigma t} \left( \cos(\omega t) + \frac{\sigma}{\omega} \sin(\omega t) \right) \right]$$

L'andamento della risposta consisterà in delle oscillazioni smorzate che si attenueranno su  $\mu$  andando verso l'infinito.



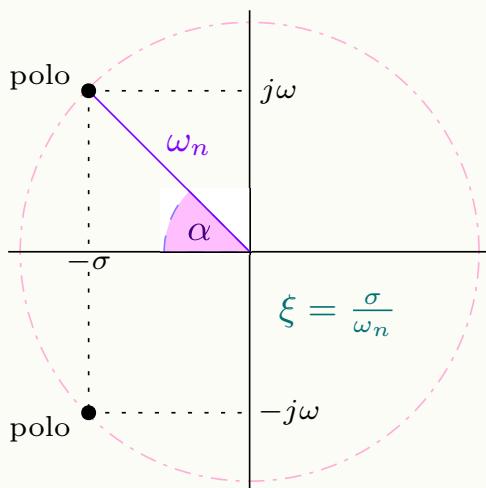
Più  $\omega$  è grande, più la frequenza delle oscillazioni è maggiore. Più  $\sigma$  è grande, più sarà rapida la risposta ad attenuarsi sul valore a regime  $\mu$ .

Si consideri la seguente parametrizzazione, si pone

$$\omega_n^2 = \sigma^2 + \omega^2$$

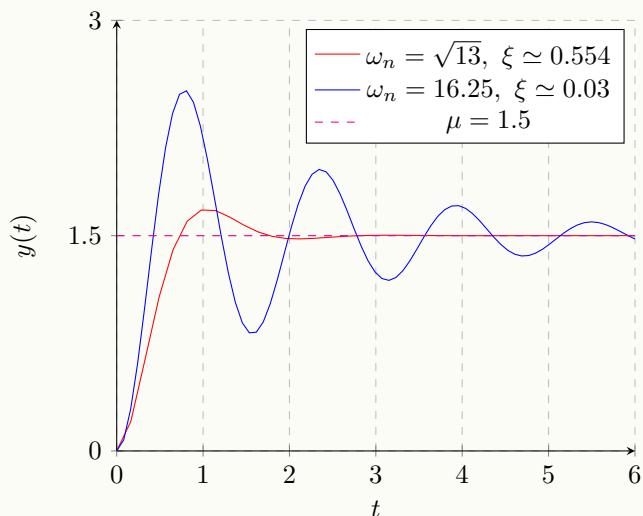
$$\omega_n \xi = \sigma$$

$$\omega_n \sqrt{1 - \xi^2} = \omega$$



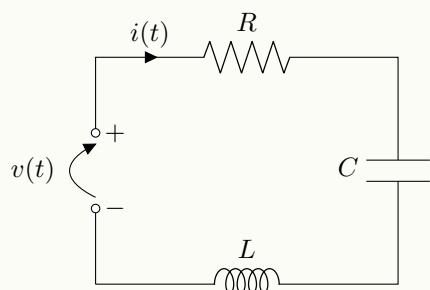
In tale modo  $\omega_n$  è il modulo dei poli ed è detta **pulsazione naturale**. Sia  $\alpha$  l'argomento dei poli, allora  $\xi = \cos \alpha$ , tale valore è detto **fattore di smorzamento**, ed è compreso fra zero ed 1. Se è uguale a 0, le oscillazioni non sono smorzate (sistema non asintoticamente stabile), se è uguale ad 1, le oscillazioni sono assenti. Più  $\xi$  è vicino ad 1 più lo smorzamento delle oscillazioni è forte.

$$G(s) = \frac{\varrho}{s^2 + 2\xi\omega_n s + \omega_n^2}$$



### Esempio circuito RLC

Si consideri il circuito  $RLC$  mostrato in figura



Il sistema dinamico equivalente è il seguente

$$\begin{cases} Cx_1 = v - x_2 \\ Lx_2' = x_1 - Rx_2 \\ y = Rx_2 \end{cases}$$

La funzione di trasferimento equivalente è la seguente

$$G(s) = \frac{1}{s^2 + \frac{R}{L}s + \frac{1}{LC}} \frac{R}{LC}$$

Si hanno pulsazione naturale e fattore di smorzamento

$$\omega_n = \frac{1}{\sqrt{LC}} \quad \xi = \frac{R}{2} \sqrt{\frac{C}{L}} \quad \mu = R$$

### Poli Dominanti ed Equivalenti

In una funzione di trasferimento, i poli detti *dominanti* sono quelli che forniscono il contributo maggiore nella risposta del sistema. Consideriamo una generica funzione di trasferimento (per semplicità, con poli reali e distinti)

$$Y(s) = G(s) \frac{1}{s} = \frac{\alpha_0}{s} + \frac{\alpha_1}{1+s\tau_1} + \cdots + \frac{\alpha_n}{1+s\tau_n}$$

L'andamento nel tempo è

$$y(t) = \alpha_0 + \frac{\alpha_1}{\tau_1} e^{-\frac{t}{\tau_1}} + \cdots + \frac{\alpha_n}{\tau_n} e^{-\frac{t}{\tau_n}}$$

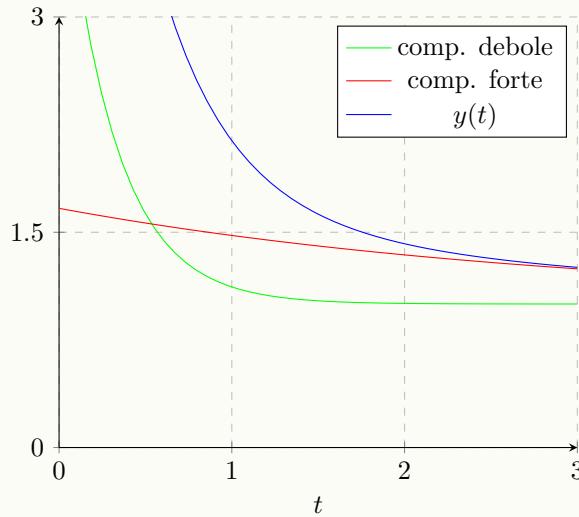
Sia  $\tau_i$  lo specifico valore per cui  $\tau_i > \tau_j \ \forall j$ , allora la componente

$$\frac{\alpha_i}{\tau_i} e^{-\frac{t}{\tau_i}}$$

Darà il maggior contributo all'andamento dell'uscita

$$y(t) \simeq \alpha_0 + \frac{\alpha_i}{\tau_i} e^{-\frac{t}{\tau_i}}$$

Il polo dominante  $-\frac{1}{\tau_i}$  sarà quello più vicino all'asse immaginario. Nel caso di poli complessi e coniugati, saranno entrambi dominanti.



È inoltre possibile approssimare una funzione di trasferimento con un *polo equivalente*, si consideri la seguente funzione (nell'ipotesi che i poli siano tutti reali negativi)

$$G(s) = \frac{\mu}{(1+s\tau_1)(1+s\tau_2)\dots(1+s\tau_n)} \quad \tau_i > 0$$

Si considera

$$\tau_e := \sum_{i=1}^n \tau_i$$

La funzione approssimata al polo equivalente sarà

$$G_e(s) = \frac{\mu}{(1 + s\tau_e)}$$

Ad esempio, si consideri

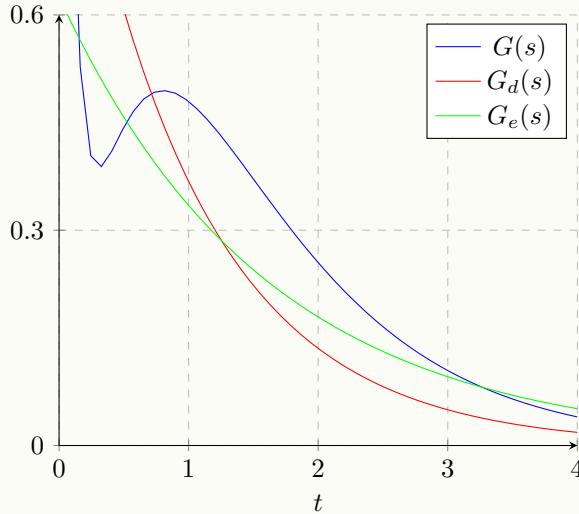
$$G(s) = \frac{1}{(1 + s)(1 + 0.1s)(1 + 0.5s)}$$

Si ha

$$G_e(s) = \frac{1}{1 + 1.6s}$$

La funzione approssimata al polo dominante è invece

$$G_d(s) = \frac{1}{1 + s}$$



## 8.6 Analisi in Frequenza

Dato un sistema dinamico lineare completamente descritto da una funzione di trasferimento, si vuole studiare la risposta dato un generico ingresso  $u(t)$ . Sia  $G(s)$  la funzione in questione, assente di cancellazioni

$$G(s) = \frac{N(s)}{D(s)}$$

Sia  $u(t)$  un ingresso qualsiasi che ammette una trasformata di Laplace razionale  $U(s)$ . La risposta del sistema è

$$Y(s) = U(s) \frac{N(s)}{D(s)}$$

è possibile tramite l'espansione in fratti semplici mettere in evidenza i termini **semplicemente stabili**, **asintoticamente stabili** ed **instabili**.

$$Y(s) = \sum_{i=0}^{n_{ss}} \frac{P_i}{s - p_i} + \sum_{i=0}^{n_{as}} \frac{P_i}{s - p_i} + \sum_{i=0}^{n_{ns}} \frac{P_i}{s - p_i}$$

La **risposta transitoria** è identificata dai termini della risposta asintoticamente stabile che tendono a zero per  $t \rightarrow \infty$ .

$$\lim_{t \rightarrow \infty} y_{trans}(t) = \mathcal{L}^{-1} \left[ \sum_{i=0}^{n_{as}} \frac{P_i}{s - p_i} \right] = 0$$

La risposta a **regime permanente**, composta dai termini semplicemente stabili ed instabili, costituisce il comportamento del sistema a lungo termine.

$$y_{reg}(t) = \mathcal{L}^{-1} \left[ \sum_{i=0}^{n_{ss}} \frac{P_i}{s - p_i} + \right] + \mathcal{L}^{-1} \left[ \sum_{i=0}^{n_{ns}} \frac{P_i}{s - p_i} \right]$$

**Esempio:** Si consideri il sistema descritto dalla funzione di trasferimento

$$G(s) = \frac{2(s+1)}{(s+2)(s+10)}$$

Si consideri l'ingresso

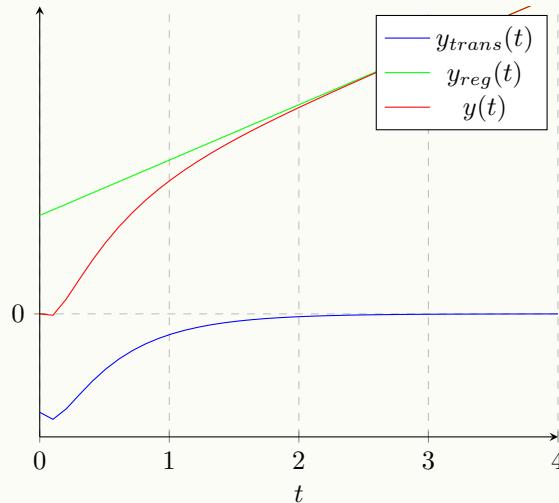
$$u(t) = 4t \cdot H(t)$$

Essendo  $U(s) = 4\mathcal{L}[t] = \frac{4}{s^2}$ , la risposta del sistema è

$$Y(s) = \frac{8(s+1)}{s^2(s+2)(s+10)}$$

Si espande in fratti semplici

$$\begin{aligned} Y(s) &= \frac{2}{5s^2} - \frac{1}{4(s+2)} + \frac{9}{100(s+10)} + \frac{4}{25s} \\ Y_{trans}(s) &= \frac{9}{100(s+10)} - \frac{1}{4(s+2)} \\ y_{trans}(t) &= \left[ -\frac{1}{4}e^{-2t} + \frac{9}{100}e^{-10t} \right] H(t) \\ Y_{reg}(s) &= \frac{9}{100s^2} + \frac{4}{25s} \\ y_{reg}(t) &= \frac{9t}{100} + \frac{4}{25}H(t) \end{aligned}$$



**Teorema (Risposta in Frequenza) :** Nell'ipotesi di asintotica stabilità, si consideri un ingresso sinusoidale

$$u(t) = A \sin(\omega t) H(t)$$

La risposta a regime permanente del sistema avrà la **stessa pulsazione** dell'ingresso.

$$y(t) \simeq B \sin(\omega t + \phi) H(t)$$

Data una funzione di trasferimento  $G(s)$  di un sistema, definiamo  $G(j\omega)$  (al variare di  $\omega$ ) la sua **risposta in frequenza**<sup>1</sup>.

Si consideri la funzione di trasferimento

$$G(s) = \frac{1}{1+s}$$

e l'ingresso  $u(t) = 10 \sin(2t)H(t)$ , la frequenza dell'ingresso è 2, si analizza la risposta del sistema quando la frequenza dell'ingresso è 2

$$G(2j) = \frac{1}{1+2j} = \frac{1}{5} - j\frac{2}{5}$$

Si calcola modulo ed argomento della risposta  $G(2j)$

$$|G(2j)| = \sqrt{1/25 + 4/25} = 1/\sqrt{5}$$

$$\arg(G(2j)) = \arctan(-2) = -63\frac{\pi}{180}$$

Il fatto interessante è che la risposta del sistema (a regime permanente) è uguale all'ingresso, moltiplicato per il modulo della sua risposta, e sfasato dell'argomento della risposta.

$$y(t) \simeq |G(2j)|10 \sin(2t + \arg(G(2j))) = \frac{10}{\sqrt{5}} \sin(2t - 63 \cdot \pi/180)$$

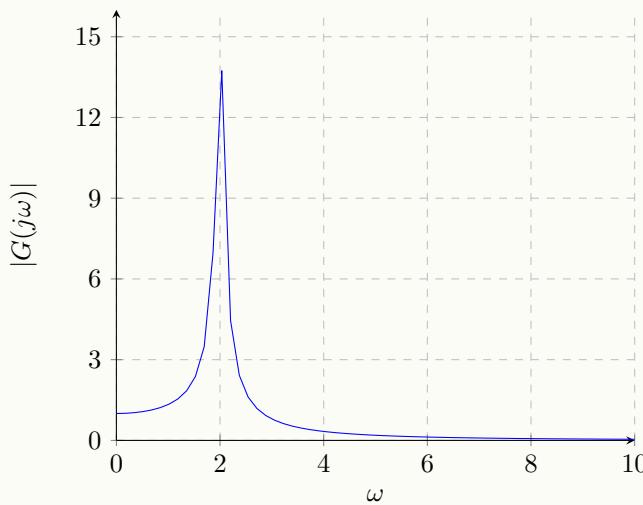
Si può rappresentare graficamente la risposta in frequenza al variare di  $\omega$ , visualizzando l'"intensità" della risposta del sistema in base alla pulsazione dell'ingresso.

Si consideri il seguente esempio, si ha una funzione di trasferimento

$$G(s) = \frac{1}{1 + 1/10s + \frac{s^2}{4}}$$

Il modulo della risposta in frequenza è

$$|G(j\omega)| = \frac{1}{|1 + 1/10j\omega + \frac{(j\omega)^2}{4}|} = \left( \sqrt{\left(1 - \frac{\omega^2}{4}\right)^2 + \frac{1}{100}\omega^2} \right)^{-1}$$



I punti di *massimo* della risposta in frequenza si dicono **risonanza**, nell'esempio sovrastante, avviene per  $\omega = 2$ .

Si è visto come

- Se l'ingresso è una sinusoide, la risposta avrà la medesima pulsazione dell'ingresso.

Si vuole considerare un caso *più generale* di ingresso e studiarne la risposta in frequenza.

---

<sup>1</sup>si ricordi che con  $j$  si indica l'unità immaginaria

## Ingresso multi sinusoidale

Si considera un caso più generale in cui l'ingresso, piuttosto che essere un'armonica è una *combinazione lineare* di  $n$  armoniche:

$$u(t) = \sum_{k=1}^n c_k \sin(\omega_k t + \gamma_k)$$

Per il principio di sovrapposizione degli effetti, e per il teorema della risposta in frequenza, a regime permanente la risposta sarà

$$y(t) = \sum_{k=1}^n c_k |G(j\omega_k)| \cdot \sin(\omega_k t + \gamma_k + \arg(G(j\omega_k)))$$

## Ingresso periodico

Si consideri un caso ancora più generale in cui  $u$  è una funzione periodica di periodo  $t$ , ammette quindi uno sviluppo in **serie di Fourier**

$$u(t) = c_0 + \sum_{k=1}^{\infty} c_k \sin\left(k \frac{2\pi}{T} t + \gamma_k\right)$$

Esattamente come nel caso multi sinusoidale, valgono i principi di principio di sovrapposizione degli effetti e teorema della risposta in frequenza, a regime permanente la risposta sarà

$$y(t) = G(0)c_0 + \sum_{k=1}^{\infty} c_k |G(jk \frac{2\pi}{T})| \cdot \sin(\frac{2\pi}{T}t + \gamma_k + \arg(G(jk \frac{2\pi}{T})))$$

## Ingresso generico

Nel caso di un generico ingresso, esso può essere scritto come

$$u(t) = \int_0^\infty C(\omega) \sin(\omega t + \gamma(\omega)) d\omega$$

Dove  $C(\omega)$  è lo *spettro di ampiezza*, ossia il modulo della trasformata di Fourier di  $u$ , e  $\gamma(\omega)$  è lo *spettro di fase*, ossia l'argomento della trasformata di Fourier di  $u$ .

$$\begin{aligned} C(\omega) &= |\mathcal{F}[u](\omega)| \\ \gamma(\omega) &= \arg(\mathcal{F}[u](\omega)) \end{aligned}$$

Richiamo sulla trasformata di fourier di una generica funzione  $u(t)$ :

$$\mathcal{F}[u](\omega) = \int_{\mathbb{R}} u(t) e^{-2\pi t\omega} dt$$

Per il principio di sovrapposizione degli effetti e per il teorema della risposta in frequenza, a regime permanente la risposta sarà

$$y(t) = \int_0^{\infty} C(\omega) |G(j\omega)| \sin(\omega t + \gamma(\omega) + \arg(G(j\omega))) d\omega$$



## 8.7 Banda Passante e Margine di Fase

Si consideri un sistema lineare con funzione di trasferimento  $G(s)$ . La funzione di risposta armonica del sistema lineare è  $G(j\omega)$ .

Se applichiamo un ingresso  $u(t)$ , la risposta armonica dell'uscita  $y(t)$  sarà

$$Y(j\omega) = G(j\omega)X(j\omega)$$

E vale che

$$\begin{aligned}|Y(j\omega)| &= |G(j\omega)| + |X(j\omega)| \\ \arg(Y(j\omega)) &= \arg(G(j\omega)) + \arg(X(j\omega))\end{aligned}$$

È come se il sistema  $G$  fungesse da *filtro* per il segnale di ingresso. La **banda passante** è l'intervallo di frequenze per cui il modulo della funzione di trasferimento è maggiore di una certa soglia, tipicamente -3 dB rispetto al valore massimo.

### Calcolo della banda passante

1. Data una funzione  $G(s)$  si considera il modulo di  $G(j\omega)$  al variare di  $\omega$ .
2. Si sceglie una soglia di attenuazione, tipicamente -3 dB rispetto al valore massimo del modulo di  $G(j\omega)$ .
3. Si trovano le frequenze  $\omega_1, \omega_2$  per cui  $|G(j\omega)|$  è uguale alla soglia scelta, dette frequenze di taglio.
4. La banda passante è data dalla differenza tra le frequenze di taglio

Si definisce poi  $\omega_c$  la **pulsazione critica**, ossia la frequenza per cui

$$|G(j\omega_c)| = 1 = 0dB$$

Si definisce poi il **Margine di fase**

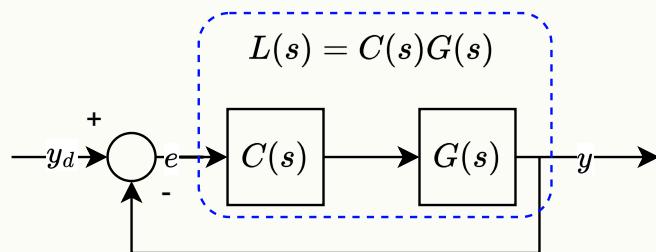
$$\varphi_m = 180^\circ - \arg(G(j\omega_c))$$

In sostanza, il margine di fase indica di quanti gradi la fase del sistema deve ancora diminuire per raggiungere  $-180^\circ$ , valore che corrisponde alla condizione limite di stabilità. Perché il margine di fase è importante?

- **Stabilità:** Un margine di fase positivo garantisce la stabilità del sistema a ciclo chiuso. Più alto è il margine di fase, maggiore è la stabilità e minore è la tendenza del sistema ad oscillare.
- **Prestazioni:** Il margine di fase influenza anche le prestazioni del sistema. Un margine di fase troppo basso può portare a una risposta lenta e a un elevato errore a regime. Un margine di fase troppo alto può rendere il sistema eccessivamente smorzato, con una risposta lenta e poco reattiva.

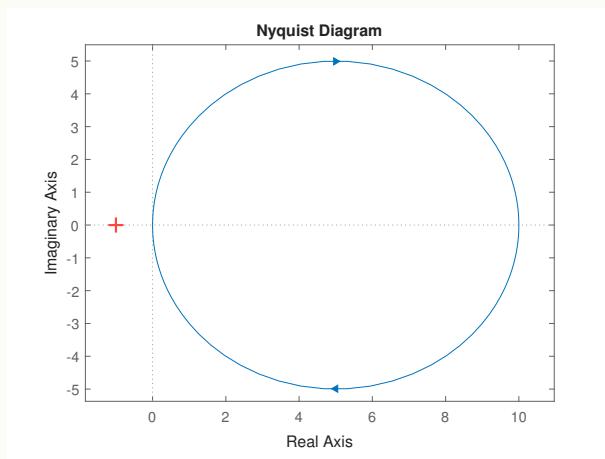
#### 8.7.1 Criterio di Nyquist

Il criterio di Nyquist è un metodo *grafico* che permette di stabilire l'asintotica stabilità di un sistema ad anello chiuso  $F(s) = \frac{L(s)}{1+L(s)}$  analizzando esclusivamente  $L(s)$ .



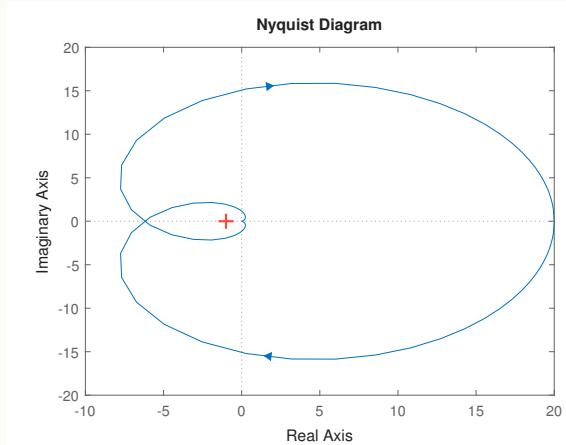
Il diagramma di Nyquist coincide con il diagramma polare di  $L(j\omega)$  reso simmetrico rispetto l'asse reale. È una curva chiusa orientata per  $\omega$  crescenti. Definiamo  $n$  il numero di giri antiorari del diagramma intorno al punto critico  $-1 + 0j$  sottratto al numero di giri orari, e  $p$  il numero di poli di  $L(s)$  con parte reale positiva. Il sistema  $F(s)$  ad anello chiuso è asintoticamente stabile se,  $n$  è ben definito (il diagramma non passa sul punto critico) ed è uguale a  $p$ .

**Esempio 1 :** Si consideri la funzione  $L(s) = \frac{10}{1+s}$ , il relativo diagramma di Nyquist è



La funzione ha  $p = 0$  poli con parte reale positiva e  $n = 0$  giri antiorari intorno al punto critico, essendo  $p = n$ , allora  $F(s) = \frac{L(s)}{1+L(s)}$  è asintoticamente stabile.

**Esempio 2 :** Si consideri la funzione  $L(s) = 20 \frac{1-s/2}{(s+1)^3}$ , il relativo diagramma di Nyquist è



La funzione di trasferimento ha 3 poli con parte reale negativa ( $s = -1$ ), quindi  $p = 0$ . Il diagramma fa un 2 giri orari intorno al punto critico e 0 giri antiorari, quindi  $n = -2$ , essendo  $p \neq n$  la funzione  $F(s) = \frac{L(s)}{1+L(s)}$  non è asintoticamente stabile.

Tornando ai controllori, se  $G(s)$  è un processo fisico, e  $C(s)$  il controllore, vogliamo che il controllore perfetto inverta la dinamica del sistema, ossia

$$C(s)G(s) = 1 \implies C(s) = \frac{1}{G(s)}$$

Ciò sta a significare che, qualsiasi riferimento dato al controllore, il sistema lo riprodurrà. È però necessario che il controllore **non abbia** più zeri che poli. Si consideri il processo fisico

$$G(s) = \frac{10 + 10s}{(1 + 2s)(1 + 0.1s)}$$

Si vuole progettare un controllore per tale processo. La funzione che inverte totalmente la dinamica del sistema è

$$C_0(s) = \frac{(1 + 2s)(1 + 0.1s)}{10 + 10s}$$

Avendo il numeratore di grado più alto del denominatore, *non è realizzabile*, quindi è necessario pensare ad un altro controllore, ad esempio

$$C_1(s) = \frac{0.1(1 + 2s)(1 + 0.1s)}{(1 + s)(1 + 0.01s)}$$

Infatti

$$C_1(s)G(s) = \frac{1}{1 + 0.01s}$$

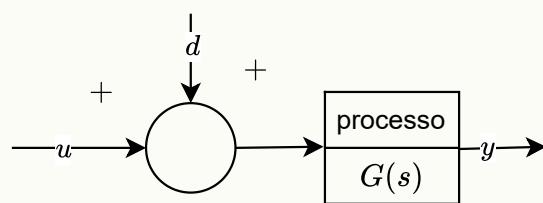
Il sistema si comporta da filtro passa-basso, lascierà inalterate le frequenze dei segnali comprese fra 0 e 100hz.



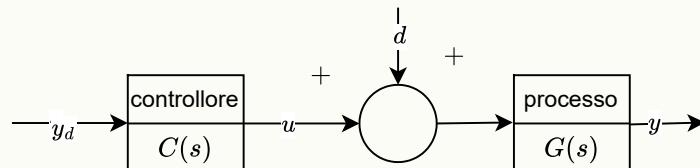
## 8.8 Analisi dei Sistemi Retro Azionati

In questa sezione verranno affrontati i sistemi LTI retro azionati, e verranno presentate delle proprietà utili nella *progettazione* di un controllore.

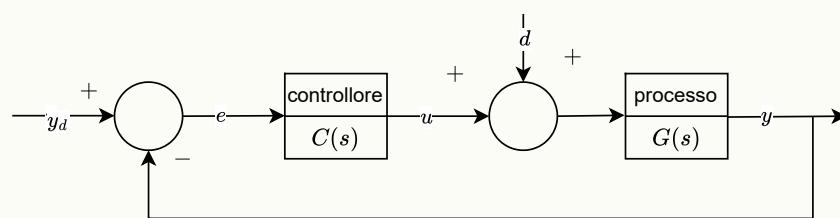
Nel classico schema dei sistemi di controllo, vi è un generico *processo*  $G(s)$ , alla quale dare un ingresso  $u$ , che ne scaturisce una risposta  $y$ , il processo può essere soggetto da un *disturbo*  $d$



Un *controllore* si pone come ingresso per il processo, dato un segnale di riferimento  $y_d$ , il suo scopo è quello di generare l'ingresso  $u$  tale che l'uscita del sistema sia proprio il riferimento  $y_d$ . In un sistema ad *anello aperto* il controllore funziona senza un feedback.



Differentemente, un sistema ad *anello chiuso* misura l'uscita del sistema e la confronta con il segnale di riferimento, valutando l'**errore**  $e = y_d - y$ , che verrà utilizzato dal controllore per generare il segnale di controllo adeguato.



Consideriamo il caso generale in cui il riferimento  $y_d$  varia nel tempo, l'errore vale:

$$e(t) = y_d(t) - y(t)$$

nel dominio di Laplace

$$E(s) = Y_d(s) - Y(s)$$

Il segnale di controllo  $u$  è dato dal prodotto fra l'errore e la funzione di trasferimento del controllore

$$U(s) = C(s)E(s)$$

L'uscita è data da

$$Y(s) = G(s)C(s)E(s) + D(s)$$

Dove  $D(s)$  è la trasformata di Laplace del disturbo  $d$  (assunto noto). Sostituendo l'espressione di  $E$  si ha

$$Y(s) = G(s)C(s)[Y_d(s) - Y(s)] + D(s)$$

Si raccoglie  $Y(s)$

$$Y(s)(1 + G(s)C(s)) = G(s)C(s)Y_d(s) + D(s)$$

Quindi l'uscita del sistema è

$$Y(s) = Y_d(s) \frac{G(s)C(s)}{1 + G(s)C(s)} + D(s) \frac{C(s)}{1 + G(s)C(s)}$$

Si definisce  $L(s) = G(s)C(s)$  il **guadagno in anello aperto**. In termini di questo, l'errore vale

$$E(s) = Y_d(s) \frac{1}{1 + L(s)} - D(s) \frac{G(s)}{1 + L(s)}$$

Se l'uscita del sistema è

$$Y(s) = Y_d(s) \frac{G(s)C(s)}{1 + G(s)C(s)}$$

Vorremmo idealmente seguire il segnale di riferimento

$$\frac{Y(s)}{Y_d(s)} = 1 \implies \frac{G(s)C(s)}{1 + G(s)C(s)} = 1 \implies \frac{L(s)}{1 + L(s)} = 1$$

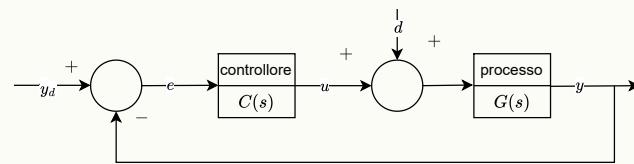
Definiamo  $F(s) = \frac{L(s)}{1+L(s)}$ , è la funzione di sensitività, che indica quanto l'uscita del sistema è influenzata da variazioni nel segnale di riferimento. Dato il disturbo  $D(s)$  si definisce

$$M(s) = \frac{D(s)}{1 + L(s)}$$

È la funzione di sensitività al disturbo, che indica quanto l'uscita del sistema è influenzata da disturbi esterni. Vorremmo che  $M(s) = 0$ , significa che il disturbo non ha alcuna influenza sull'uscita del sistema, ovvero il sistema è completamente immune ai disturbi.

### 8.8.1 Analisi di $F(s)$

In un generico sistema retroazionato



L'**analisi statica** di un sistema dinamico ci permette di studiare il comportamento del sistema a regime permanente. Sia  $L(s) = G(s)C(s)$ , si vuole analizzare la funzione  $F(s) = \frac{L(s)}{1+L(s)}$ , si ricordi che il *caso ideale* ricade quando  $F(s) \simeq 1$ . Nel caso generale

$$L(s) = \frac{\mu}{s^g} \frac{\prod_i (1 + sT_i)}{\prod_i (1 + s\tau_i)}$$

Essendo che per  $s \rightarrow 0$  il termine a destra diventa 1, a regime permanente vale  $L(s)$  vale  $\frac{\mu}{s^g}$ . Si consideri come valore di ingresso lo scalino

$$w(t) = A \cdot H(t)$$

Per un dato valore  $A$ . Calcoliamo il valore della funzione di sensitività a regime permanente ( $\frac{A}{s}$  è la trasformata di laplace dell'ingresso  $w(t)$ )

$$\lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} sF(s) \frac{A}{s} = \quad (8.13)$$

$$\lim_{s \rightarrow 0} \frac{L(s)}{1 + L(s)} = A \lim_{s \rightarrow 0} \frac{\frac{\mu}{s^g}}{1 + \frac{\mu}{s^g}} = \quad (8.14)$$

$$A \lim_{s \rightarrow 0} \frac{\mu}{s^g + \mu} \quad (8.15)$$

Quindi il valore a regime permanente può valere

$$y(\infty) = \begin{cases} A \frac{\mu}{1+\mu} & \text{se } g = 0 \text{ il guadagno è } \mu_F \simeq 1 \text{ se } \mu \gg 1 \\ A & \text{se } g > 0 \text{ il guadagno è } \mu_F = 1 \\ 0 & \text{se } g < 0 \text{ (caso pessimo)} \end{cases}$$

Un sistema retroazionato con funzione  $F(s) = \frac{L(s)}{1+L(s)}$  è stabile asintoticamente se, dato

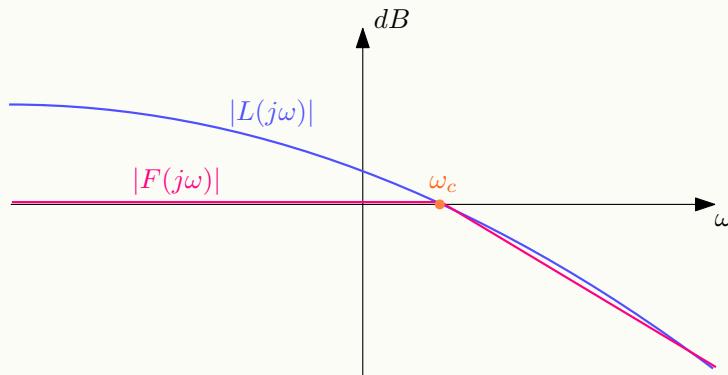
$$L(s) = \frac{\text{Num}(s)}{\text{Den}(s)}$$

$$F(s) = \frac{\text{Num}(s)}{\text{Num}(s) + \text{Den}(s)}$$

Si ha che il polinomio  $\text{Num}(s) + \text{Den}(s)$  ha radici con parte reale negativa. Vediamo ora l'analisi della **risposta in sequenza** di  $F(s)$

$$|F(j\omega)| = \frac{|L(j\omega)|}{|1 + L(j\omega)|} \simeq \begin{cases} 1 & \text{se } |L(j\omega)| \gg 1 \\ |L(j\omega)| & \text{se } |L(j\omega)| \ll 1 \end{cases}$$

È chiaro che laddove il valore del modulo della risposta in frequenza di  $L$  è molto maggiore di 1, la risposta in frequenza tende ad 1 (0 dB nel diagramma di Bode).



L'andamento sarà qualitativamente quello mostrato in figura, quando il modulo di  $L$  scende sotto una certa soglia, l'andamento del modulo di  $F$  inizierà a "seguirlo". È chiaro che il punto in cui  $|F(j\omega)|$  inizierà a calare sarà la *pulsazione critica* trattata nella sezione 8.7, ossia il punto in cui il modulo della risposta armonica di  $L$  è 1 (ossia 0dB)

$$|L(j\omega_c)| = 1 = 0dB$$

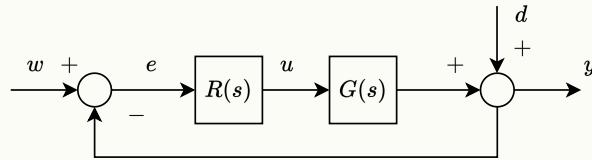
$F(s)$  si comporta come un filtro passa basso

Lascia passare le frequenze comprese in  $[0, \omega_c]$  con un guadagno  $\mu_f$  presentato prima (dipende dai valori di  $g$ ). I poli dominanti di  $F(s)$  sono in corrispondenza di  $\omega_c$ .

In un sistema di controllo in anello chiuso con retroazione, la precisione statica dipende da  $g$  e  $\mu$ . La precisione dinamica dipende dalla frequenza critica  $\omega_c$  e dal margine di fase  $\varphi_m = 180^\circ - \arg(F(j\omega_c))$

### 8.8.2 Esempio di Progettazione

Si consideri il seguente schema a blocchi



Il processo  $G(s)$  è descritto dalla funzione di trasferimento

$$G(s) = \frac{10}{(1+10s)(1+5s)(1+s)}$$

Il riferimento  $w$  e l'errore  $d$  sono scalini e possono assumere i seguenti valori

$$\begin{cases} w(t) = AH(t) & A \in [-1, 1] \\ d(t) = BH(t) & B \in [-5, 5] \end{cases}$$

*Si vuole progettare un controllore che soddisfi le seguenti specifiche*

- $|e(\infty)| \leq 0.1$  l'errore a regime permanente deve essere al più 0.1
- $w_c \geq 0.2$  vincolo sulla frequenza critica
- $\varphi_m \geq 60^\circ$  vincolo sul margine di fase

Il generico controllore ha la forma

$$R(s) = R_1(s)R_2(s) \frac{\mu_R}{s^r} \frac{\prod_i (1+sT_i)}{\prod_i (1+s\tau_i)}$$

Consideriamo il progetto statico (**a regime permanente**), solamente il primo termine  $R_1(s)$  ha influenza. La funzione ad anello aperto sarà

$$L(s) = R_1(s)G(s) = \frac{\mu_R}{s^r} \frac{10}{(1+10s)(1+5s)(1+s)}$$

L'errore a regime permanente è composto da due parti, l'errore dovuto al disturbo sul riferimento  $e_w(t)$  e l'errore dovuto al disturbo  $d$  sulla variabile controllata, ossia  $e_d(t)$

$$e(\infty) = e_w(\infty) + e_d(\infty)$$

Data questa uguaglianza, l'errore a regime permanente è limitato come segue

$$|e(\infty)| \leq |e_w(\infty)| + |e_d(\infty)|$$

Calcoliamo  $e_w(\infty)$ . Sappiamo che

$$e_w = w - y$$

quindi

$$E_w(s) = W(s) - Y(s)$$

inoltre

$$Y(s) = W(s) \frac{L(s)}{1 + L(s)}$$

Quindi

$$E_w(s) = W(s) - W(s) \frac{L(s)}{1 + L(s)}$$

Essendo che  $w(t) = AH(t)$ , si ha che  $W(s) = \frac{A}{s}$

$$E_w(s) = \frac{A}{s} - \frac{A}{s} \frac{\frac{\mu_R}{s^r} \frac{10}{(1+10s)(1+5s)(1+s)}}{1 + \frac{\mu_R}{s^r} \frac{10}{(1+10s)(1+5s)(1+s)}}$$

Consideriamo il caso  $r = 0$

$$E_w(s) = \frac{A}{s} - \frac{A}{s} \frac{\mu_R \frac{10}{(1+10s)(1+5s)(1+s)}}{1 + \mu_R \frac{10}{(1+10s)(1+5s)(1+s)}}$$

$$E_w(s) = \frac{1}{s} \left[ A - A \frac{\mu_R \frac{10}{(1+10s)(1+5s)(1+s)}}{1 + \mu_R \frac{10}{(1+10s)(1+5s)(1+s)}} \right]$$

Si calcola  $e_w(\infty) = \lim_{s \rightarrow 0} s E_w(s)$

$$\lim_{s \rightarrow 0} s \frac{1}{s} \left[ A - A \frac{\mu_R \frac{10}{(1+10s)(1+5s)(1+s)}}{1 + \mu_R \frac{10}{(1+10s)(1+5s)(1+s)}} \right] = \lim_{s \rightarrow 0} \left[ A - A \frac{\mu_R \frac{10}{(1+10s)(1+5s)(1+s)}}{1 + \mu_R \frac{10}{(1+10s)(1+5s)(1+s)}} \right] =$$

$$\left[ A - A \frac{\mu_R 10}{1 + \mu_R 10} \right] = A \left[ 1 - \frac{10\mu_R}{1 + 10\mu_R} \right] = \frac{A}{1 + 10\mu_R}$$

Quindi

$$\text{☞ } e_w(\infty) = \frac{A}{1 + 10\mu_R} \implies |e_w(t)| = \frac{|A|}{1 + 10\mu_R} \leq \frac{1}{1 + 10\mu_R} \text{ ☞}$$

Seguendo lo stesso identico ragionamento si ha che

$$\text{☞ } e_d(\infty) = \frac{B}{1 + 10\mu_R} \implies |e_d(t)| = \frac{|B|}{1 + 10\mu_R} \leq \frac{5}{1 + 10\mu_R} \text{ ☞}$$

Ma allora

$$\begin{aligned} |e(\infty)| &\leq |e_w(\infty)| + |e_d(\infty)| \implies \\ |e(\infty)| &\leq \frac{1}{1 + 10\mu_R} + \frac{5}{1 + 10\mu_R} = \frac{6}{1 + 10\mu_R} \end{aligned}$$

Quindi seguendo le specifiche del progetto

$$|e(\infty)| \leq 0.1 \iff \frac{6}{1 + 10\mu_R} \leq 0.1 \iff \mu_R \geq 5.9$$

Nel caso  $r = 1$  l'errore a regime è nullo per qualsiasi  $\mu_R$ . Avendo trovato il valore minimo di  $\mu_R$ , si pone (arbitrariamente seguendo il vincolo)

$$R_1(s) = 8$$

Si ha quindi che

$$L(s) = 8G(s)R_2(s)$$

Si pone  $L'(s) = 8G(s)$

$$L'(s) = \frac{80}{(1 + 10s)(1 + 5s)(1 + s)}$$

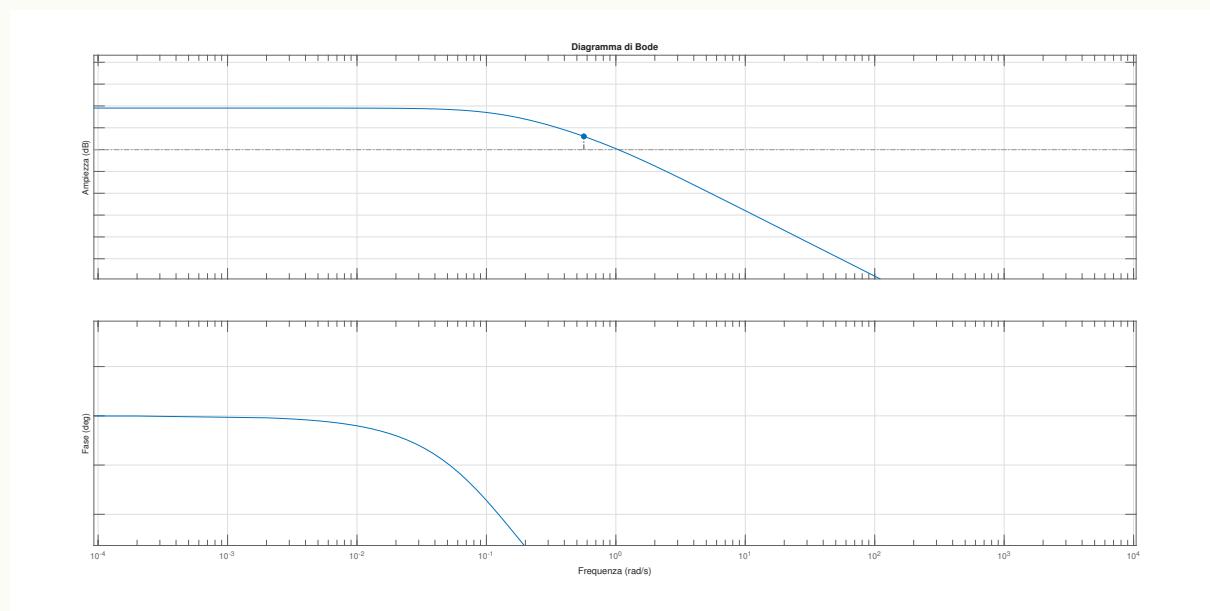
A questo punto si deve definire  $R_2(s)$  in modo che rispetti le specifiche di progetto, ossia

- $w_c \geq 0.2$  vincolo sulla frequenza critica
- $\varphi_m \geq 60^\circ$  vincolo sul margine di fase

Si va a **tentativi**, si provi con  $R_2(s) = 1$

$$|L(j\omega_c)| = 1 \implies \left| \frac{80}{(1 + 10j\omega_c)(1 + 5j\omega_c)(1 + j\omega_c)} \right| = 1 \implies \omega_c \simeq 1.0409$$

Il vincolo sulla frequenza critica è rispettato. I diagrammi di Bode sono



Il margine di fase è chiaramente negativo, quindi il vincolo  $\varphi_m \geq 60^\circ$  non è rispettato, allora  $R_2(s) = 1$  non va bene come regolatore.

Capire come scegliere un regolatore corretto. Libro Controlli Automatici (G. Marro) : pagina 318