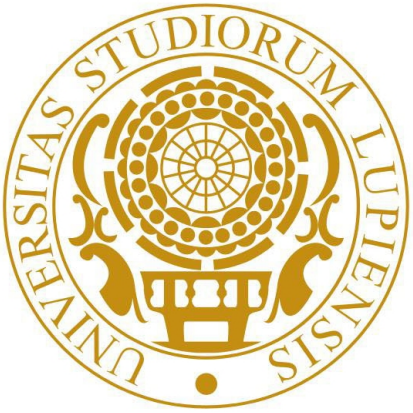


UNIVERSITÀ DEL SALENTO
DEPARTMENT OF INNOVATION ENGINEERING

MASTER'S DEGREE IN COMPUTER ENGINEERING

NETWORK TECHNOLOGIES

Network Technologies

Dott. Marco Chiarelli

Academic Year 2016/2017

Questi appunti sono stati scritti utilizzando LaTeX tramite la distribuzione MiKTeX `http://miktex.org/`

Come editor è stato usato TeXMaker 4.5 `http://www.xm1math.net/texmaker/`

Potrebbero essere state ritagliate delle immagini provenienti dalle dispense/esercizi del corso Network Technologies at UNISALENTO. Ad ogni modo i diritti di talune foto sono del prof. Giovanni Ciccarese, e non me ne assumo alcuna responsabilità dal cattivo utilizzo. Si tratta di materiale pubblicamente accessibile: `http://www.ingegneria.unisalento.it/scheda_docente/-/people/giovanni.ciccarese/materiale`

Dei contenuti rielaborati in questa opera, salvo esplicitamente scritto il contrario, il prof. Giovanni Ciccarese non se ne assume alcuna responsabilità.

Sommario

Le seguenti dispense vogliono essere un resoconto didattico del corso di Network Technologies, il quale docente è il prof. Giovanni Ciccarese, presso il CdL in Ingegneria Informatica at Unisalento. L'obiettivo dell'intero corso è la Progettazione di Reti ad Alta Disponibilità. Prima di arrivare a coprire il fulcro del corso il programma ha previsto lo studio dei Sistemi a Coda atti a descrivere degli scenari tipici di utilizzo delle Reti, in concomitanza ad una rivisitazione delle moderne tecnologie di rete allo Stato dell'Arte. In particolare nel corso vengono trattati i concetti di Affidabilità e Disponibilità e vengono approfondite le loro differenze e le loro relazioni, permettendo quindi di conoscere dei concetti che possono essere utilizzati a più ampio spettro in qualsiasi ambito ingegneristico che si possa reputare tale. Il seguente lavoro non ha la pretesa di sistematizzare in maniera esatta ciò che è stato fatto a lezione, sebbene l'impostazione di base cerca di seguire fedelmente ogni singola lezione che è stata erogata. Piuttosto cerca di raccogliere in maniera sintetica ma al contempo esaustiva gli aspetti chiave della Progettazione di Reti, fornendo un apposito background ingegneristico incrementale che permetta una maggiore comprensione e chiarezza degli argomenti principali del corso. Al contempo vengono anche forniti numerosi esercizi e vengono coperte esaustive esercitazioni su quanto fatto di teorico a lezione, lasciando un importante messaggio al lettore, una realtà vera in generale, ma ancora di più nelle Reti; ovvero che non si può prescindere dal saper destreggiarsi negli aspetti veramente pratici della progettazione per padroneggiarla veramente.

Abstract

The following work wants to be a report about the Network Technologies course, whose teacher is the prof. Giovanni Ciccarese, at Computer Engineering, Unisalento. The program covers three macro-topics: the Queueing Theory, Design of High-Availability Network and an important review about the State of Art about modern Network Technologies. In this work are described both theoretical and practical aspects. In fact there are many exercises covering these topics. It won't systematize the entire work done at lessons in an exact manner, despite the base setting try to follow each lessons that has been delivered to us. It tries to gather the key aspects of the High-Availability Network Design, and to reorder some difficult notions. Moreover it tries to give an help for the preparation of an eventual didactic exam that requires the knowledge about these notions and the related practical applications/homework/exercises.

Indice

Ringraziamenti

Un grazie particolare va ai miei compagni
d'università, Dino Sbarro, Gabriele Accarino, Giampiero D'Autilia, Matteo Settembrini, Paolo Panarese ed Emanuele Costa Cesari.

Nel frattempo.. permetto al lettore di godersi cotanta passione e motivazione verso l'Ingegneria..

Prefazione all'edizione italiana

Alcuni anni fa, subito dopo la laurea, avendo ricevuto il mandato di far nascere un filone di attività nel settore delle reti a commutazione di pacchetto presso quello che allora era l'Istituto di Elettronica e Telecomunicazioni del Politecnico di Torino, mi fu consigliato di cominciare a leggere un libro appena uscito, decantandone le qualità di profondità e chiarezza. Il libro era scritto da quella che stava emergendo come la massima autorità del settore a livello internazionale: il Prof. Leonard Kleinrock del Computer Science Department della mitica University of California a Los Angeles (UCLA). Due anni dopo, mi ritrovai in mano il libro e di fronte l'autore in persona, mentre ero ritornato studente molto lontano da casa. Non è quindi senza una certa emozione che scrivo queste poche righe di prefazione a un libro che per me, come per moltissimi, è stato una delle pietre miliari della formazione tecnica e culturale. Ciò sia per la fama dell'autore, sia per la chiarezza dell'esposizione, che antepone sempre la spiegazione intuitiva dei fenomeni ad una rigorosa trattazione matematica dei modelli. La traduzione italiana, giungendo con qualche anno di ritardo rispetto all'edizione originale, ha giustamente tralasciato quelle parti del secondo volume che hanno maggiormente sofferto per la rapidissima evoluzione di un settore in continuo divenire. Essa raccoglie in un unico testo tutta la trattazione teorica, la cui importanza continua a crescere con il passare del tempo, trovando sempre nuovi settori applicativi. Si può auspicare che la disponibilità del testo in italiano serva a diffondere sempre di più l'insegnamento della teoria delle code, rendendo così possibile un approccio quantitativo nell'analisi degli aspetti tipici delle telecomunicazioni, dell'informatica e dei trasporti. Da parte mia un augurio ai lettori: che possano trovare in questo libro il gusto per lo studio quantitativo dei fenomeni, l'interpretazione dei risultati e la conseguente comprensione del comportamento dei sistemi. Quest'ultima è, in fondo, l'essenza stessa della ricerca scientifica.

MARCO AJMONE MARSAN
Professore di Reti di Telecomunicazioni
al Politecnico di Torino
Settembre 1991

Introduzione

L'argomento principale del corso è la Progettazione di Reti ad Alta Disponibilità. Vengono coperti gli aspetti fondamentali, ovvero i concetti di **Affidabilità** e **Disponibilità** e loro interrelazioni. Saranno studiate le performance delle reti, supporti alle comunicazioni wireless, verranno fornite le basi sulla sicurezza, e sarà fatta una importante review sullo Stato dell'Arte relativo a queste problematiche. Le performance e la sicurezza sono dei requisiti che spesso entrano in conflitto tra di loro. Si pensi ad esempio a quanto possa essere differente una realtà universitaria da una realtà bancaria relativamente a questi requisiti. Nella realtà bancaria un downtime annuale accettabile è di circa 5 minuti. Si parlerà di disponibilità 5-9. Requisiti principali: {**affidabilità, disponibilità e performance**}. Requisiti QoS (Quality of Service). *Reliability* and *Availability*. Un esempio veramente forte ed indicativo del livello tecnologico delle moderne Reti può essere dato dalle comunicazioni veicolari, ove si sfrutta il protocollo *IEEE 802.11p*, che sarebbe informalmente il Wi-Fi esteso al mondo veicolare.. In questi casi vi è un forte spirito collaborativo in queste particolari reti, chiamate Reti Ad-Hoc.

Negli ultimi anni le reti giocano un ruolo molto importante sulle aziende. Sono motivo di profitto e produttività. Queste reti, molto critiche, devono soddisfare alcuni requisiti (prestazioni, QoS). Performance, mobilità, affidabilità e sicurezza sono i tipici requirements. Internet è stata messa su servizio QoS. Le reti possono essere molto diverse: si pensi ad esempio al servizio VoIP il quale utilizza Internet per trasmettere il contenuto video.

Noi ci riferiamo sempre alle IP Networks. Reliability = Affidabilità, QoS. Un campione audio deve arrivare entro 2 ms. Requisito invece stringente sulla probabilità d'errore per servizi Web, FTP, etc. Queste ultime si basano tutte sul TCP. Invece in applicazioni Real-Time si utilizza l'UDP: bisogna fare in fretta. Una rete QoS deve soddisfare vari requisiti. Di recente si parla anche di *Ethernet deterministica* (che riguarda la QoS). Banda per comunicazioni in tempo reale. BUS-CAN per comunicazioni sulle varie centraline delle auto, per il mondo veicolare. Si vogliono quindi, in alcune situazioni, privilegiare alcuni flussi di comunicazioni rispetto ad altri. {**Mobilità**, **Affidabilità**, **Disponibilità**, **Sicurezza**}. Questi sono i requisiti tipici QoS. Bisogna comunque sempre fare i conti con i costi. Reduce costs! A tal proposito si fornisce un cenno su spese *OPEX* e *CAPEX*. Nel nostro ambito, rispettivamente le prime riguardano le spese per l'operatività della rete, mentre le seconde riguardano spese per la manutenzione e mantenimento della rete.

Una rete ad Alta Disponibilità necessita di RIDONDANZA. I requisiti sulla sicurezza spesso possono entrare in conflitto con i requisiti di velocità. Bisogna scegliere di volta in volta il miglior compromesso. Problemi odierni: Sono cruciali le applicazioni di rete per le aziende. Fondamentale è in questo senso il supporto alla QoS. Stanno aumentando il numero delle applicazioni che si basano su paradigma *BYOD-BYOA*, rispettivamente *Bring-Your-Own-Device* e *Bring-Your-Own-Access*. Sempre più aziende stanno permettendo di far lavorare il dipendente con il proprio dispositivo. È naturale pensare a problemi in termini di sicurezza. Il supporto alla QoS è quindi fondamentale.

Problemi relativi all'utilizzo delle WAN (reti geografiche). Ci sarà una WAN che intercon-

nette queste reti aziendali, con un traffico 80/20, ovvero 80% traffico locale e 20% traffico che attraversa la WAN. Vecchie topologie *Hub-And-Spoke* (a stella). Servono invece diversi percorsi alternativi. Abbiamo bisogno di reti parzialmente magliate. Adesso la situazione è cambiata: il rapporto 80/20 di cui sopra è stato invertito! Le reti locali sono molto affidabili. Su una WAN non è così: ha un *Packet Loss Rate* abbastanza elevato. Oggi l'80% del traffico attraversa la WAN. Le aziende adesso stanno raggruppando i server in Data Center. Bisogna attraversare la WAN quindi per interagire con i server che ora stanno in questi Data Center. Si fa un cenno a tal proposito ai cosiddetti *Chatty Protocols*. Le aziende stanno comunque riducendo il numero dei Data Center. Per effettuare una transazione c'è bisogno di un'applicazione. Un unico data center diverrebbe presto uno SPoF (*Single Point of Failure*). In questo caso bisognerebbe attraversare la WAN molte volte. All'utente interessa proprio il tempo di risposta! Questo è quindi il prezzo da pagare quando si attraversa una WAN: i tempi di risposta sono molto elevati. Ma non tutti i protocolli sono di tipo chatty. Prendiamo l'HTTP. Oggi le applicazioni sono distribuite. Anche se il protocollo non è chatty (HTTP), comunque si richiede di scaricare molti oggetti oggigiorno. Il prezzo temporale è legato all'attraversamento della WAN. Abbiamo quindi dei problemi di prestazioni. Si parla molto oggi inoltre della virtualizzazione dei server. Corrispondono a delle vere e proprie macchine virtuali. Il traffico diventa però non più predicibile, e si richiede che la banda venga quindi dimensionata correttamente.

Dal momento che questi flussi di traffico non sono più predicibili, per far fronte ai problemi di prestazioni, l'industria del networking ha implementato delle ottimizzazioni. Ma in che modo si abbassano le prestazioni? Ci sono dei problemi con le WAN, legati a come si comporta il TCP. Quindi si creano delle caratteristiche favorevoli che consentono di migliorare le prestazioni delle reti. Infatti in queste ottimizzazioni possiamo trovare un aumento di velocità 40x per l'HTTP.

Cloud Computing. Nuvole che mettono a disposizione dei servizi. Abbiamo vari paradigmi a tal proposito: {*Software-as-a-Service* (**SaaS**), *Platform-as-a-Service* (**PaaS**), *Infrastructure-as-a-Service* (**IaaS**)}. Apparati di rete virtualizzati e messi in rete. Tutto si basa sul concetto di virtualizzazione. Provider di servizi che mettono a disposizione questi servizi. Noi ne usufruiamo mediante cloud. Non è affatto sorprendente che anche qui abbiamo vari problemi di performance, oltre che di sicurezza. È importante approfondire i concetti di PoP (*Points of Presence*). I PoP dovrebbero quindi essere interconnessi mediante reti high-performance ed highly secure.

Abbiamo quindi capito che è molto importante prestare particolare attenzione a QoS, disponibilità, affidabilità e sicurezza. 5-9 99.999%. È una sigla, un numero che ha a che fare con l'Alta Disponibilità (High-Availability). Si ha in questo caso un downtime annuo di 5.26 minuti, ovvero 25.9 secondi per mese, quindi circa 6 secondi a settimana. Per ottenere questo risultato c'è ovviamente bisogno di RIDONDANZA. Il discorso sicurezza è molto legato alla disponibilità. La sicurezza è un fattore molto importante di cui tener conto quando si trattano reti ad alta disponibilità. È molto importante quindi valutare la *Capacità di Resilienza* di un dato sistema, ovvero la sua capacità di recuperare in seguito a guasti od in generale situazioni sfavorevoli. Si tenga presente che Availability $\neq$ Reliability. L'Affidabilità $R(t)$ è la probabilità che il tempo di vita di un dispositivo sia maggiore di una certa quantità temporale τ. Invece la disponibilità (di un servizio in generale) è la frazione di tempo in cui un servizio è UP. (es 5-9, 99.999%). La sigla prima citata significa che per il 99.999% di tempo il servizio dev'essere UP.

Software Defined Networking (SDN) - NFD. Tutto questo mette in crisi le architetture tradizionali di networking. Oggi i dispositivi all'interno delle reti sono veramente tanti (sempre per far fronte a sicurezza, QoS, etc.). Serve una massiccia riconfigurazione degli apparati. E se i flussi di traffico cambiano, bisogna andare a configurare altri apparati. Alla fine non impongono queste policy. Il SDN consente di soddisfare questi requisiti in maniera programmabile. Separa il piano di controllo (Intelligenza) dal piano di forwarding (piano dati). L'intelligenza è centralizzata in un controllo SDN. Poi vi sono apparati SDN-aware che hanno adesso solo la capacità

di inoltrare i dati. Non hanno più intelligenza ma solo capacità di forwarding. Poi abbiamo ovviamente apparati che gestiscono l'intelligenza. Si è standardizzato il protocollo mediante il quale l'SDN può alla fine controllare i vari dispositivi (*OpenFlow*). In base ad esso vengono identificati i flussi dati. Gli apparati OpenFlow enabled ricevono informazioni dal nodo centrale di intelligenza (Brain).

Open-Networking-Foundation (ONF) sta sviluppando l'SDN. Poi abbiamo il paradigma NFV (*Network Functions Virtualization*). Tutto quello che accade in Europa lo gestisce l'ETSI. E l'ETSI sta dietro l'NFV, la quale è una tecnologia che virtualizza i dispositivi di rete. La realizzazione in SW dei dispositivi di rete NFV ed SDN rapresenta la ricerca attule del mondo delle reti. La Rete viene ormai vista come un vero e proprio dispositivo virtuale programmabile. Dispositivi *IEEE 802.11p*. Standard per le Comunicazioni Veicolari. Architettura EITS. Comitato europeo che standardizza tutte le comunicazioni.

Capitolo 1

Network Technologies

1.1 CONGESTIONE

Controllo della congestione. Problematica molto importante nelle reti. Senza questo sistema la rete non funzionerebbe. Non riuscirebbe a smaltire il traffico. Estensione TCP/IP ECN.

Perché si ha la congestione? Essa riguarda i nodi di comunicazione (router). Il fenomeno si presenta quando la capacità di trasmissione è superiore alla capacità di smaltimento. Quando la velocità dei sender si approssima alla velocità di smaltimento. Il throughput è il traffico smaltito dalla rete. Numero medio di pacchetti che la rete è in grado di consegnare correttamente a destinazione. Abbiamo numerosi parametri come il *Delay, delay Jitter*, etc.

Riguardo le performance, sono quindi importanti i Sistemi della congestione. Ritardi che iniziano ad aumentare. Perdita di pacchetti. Abbiamo dei buffer sui link d'uscita. Buffer in input ed in output. Quando le sorgenti iniettano tanto traffico, i router iniziano a riempirsi. La Teoria delle Code fornisce una modellazione stocastica di questi fenomeni. Ritardo di permanenza nel router $\uparrow \implies$ ritardo end-to-end $\uparrow$. Abbiamo diverse componenti di ritardo: {*Ritardo di trasmissione*, *Ritardo di propagazione*, *Ritardo di accodamento*}. Una conseguenza drastica della congestione è lo scarto dei pacchetti. A meno che non vi siano meccanismi di priorità, ovvero delle Classi di Priorità. I tipici sintomi della congestione sono lunghi ritardi e perdita di pacchetti. Si badi bene che controllo della congestione è ben differente dal controllo di flusso, il quale alla fine rallenterà i sender troppo veloci. Entrambi lo faranno, ma per obiettivi differenti. Con il controllo di flusso, sempre si rallenta il sender, ma end-to-end! I sender inviano il carico, e se i router non ce la fanno si ha bisogno del controllo di congestione. Per ogni collegamento vi sono due buffer (Input ed Output). Nei buffer di input vi è l'elaborazione (instradamento). Fatto ciò il pacchetto sarà accodato nel relativo buffer di output associato al collegamento selezionato. Tempo di accodamento e tempo di trasmissione. I Sistemi a Coda e la relativa teoria che c'è dietro, la **Teoria delle Code** è praticamente utilizzata per modellare le reti. Un sistema a coda è un sistema composto da una fila di attesa ed un centro di servizio, all'interno del quale vi sono differenti servitori, i quali alla fine serviranno i clienti. Esistono diverse discipline. Si avrà l'arrivo dei clienti dall'esterno, e ciascun cliente dovrà sempre attendere il proprio turno.

Nel nostro caso, guardando l'output buffer, possiamo immaginrlo come un sistema a coda a singolo servitore. Ritardo di accodamento e ritardo di trasmissione sono modellati dal sistema a coda. Internet tra l'altro è un insieme di code connesse tra di loro. La Teoria delle Code ci permette di valutare il delay cui sono soggetti i pacchetti, e ci permette anche conseguentemente di dimensionare in maniera corretta i link. Analisi delle prestazioni, progettazione sono quindi le tematiche principali.

Un nodo comincia a congestionarsi quando la velocità di arrivo dei clienti è superiore alla velocità di smaltimento. Nel nostro caso il cliente è il pacchetto, ed il servitore è il trasmettitore.

Se i buffer fossero di dimensione infinita, ed immaginando che non vi sia overhead aggiuntivo, possiamo definire il **CARICO** come rate medio con la quale i sender iniettano i pacchetti nella rete.

Il *throughput* è la velocità di smaltimento. Throughput e load sono stati normalizzati rispetto al massimo possibile throughput della rete. Capacità massima. Il throughput idealmente dovrebbe seguire il load. All'aumentare del carico, che oltrepassa il massimo possibile della rete, il throughput (normalizzato) rimarrebbe costante e pari ad 1. Quando il load si approccia ad 1, il ritardo se ne va invece ad infinito. Questo succede praticamente anche in una situazione ideale. Nella realtà i buffer hanno dimensione finita! Nella situazione reale, il throughput seguirà il carico fino ad un certo punto, oltrepassato il quale se il carico sale troppo, il throughput scenderà bruscamente a 0. Anche per i pacchetti correttamente ricevuti dal destinatario e riscontrati, i rispettivi taluni ACK potrebbero non tornare mai al sender, il quale in protocolli TCP ritrasmetterà (inutilmente) i pacchetti. La rete non sarà più in grado di trasmettere i pacchetti. La capacità effettiva andrà a zero. Servono quindi tecniche di controllo della congestione, attualmente implementate dal TCP.

Fondamentalmente vi sono due approcci:

- **Approccio end-to-end** nel quale la rete non fornisce alcun feedback esplicito al sender dei pacchetti. La responsabilità è lasciata agli host, agli end-system;
- **Network Assisted** nel quale la rete (i nodi di commutazione) forniscono un feedback diretto.

Si avrà il setting di un eventuale bit di controllo congestione in un pacchetto. Oppure si fornisce il Rate al Sender (*Rate based*). Oppure si inviano al sender quanti pacchetti inviare prima di fermarsi (*Credit based*). Si parlerà di *Controllo Isaritmico*. Esaurito il credito la sorgente dovrebbe fermarsi. Vi sono quindi varie possibilità. Per il momento concentriamoci sul network assisted feedback. Abbiamo detto che Internet è sostanzialmente una rete di code. In tal caso vi sono dei feedback (pacchetti della rete alla sorgente, diretti), ovvero dei pacchetti di controllo (Choke Packet). Messaggio ICMP (Source Quench), che ha sempre l'obiettivo di rallentare la sorgente quando necessario. Nella pratica non è pero usato. C'è la possibilità che il feedback arrivi alla sorgente mediante receiver. Esso va ad inserire queste informazioni in un pacchetto che lo attraversa. Tramite un apposito bit di controllo, il receiver farà l'echo al sender. In questo caso quindi serve un'AZIONE PREVENTIVA. Si noti le diverse tipologie di azioni: azione *REATTIVA* nella quale la sorgente reagisce ad una situazione di congestione, ed azione *PREVENTIVA* qualora si riesca in qualche modo ad evitare laddove possibile proprio il verificarsi della congestione. Vi sono meccanismi di gestione del flusso, di Prenotazione delle risorse. In tal caso la congestione non si presenta. Admission Control significa controllo dell'ammissione. Il sender contratta la sua necessità con la rete. Esso descriverà le caratteristiche dei suoi pacchetti con appositi descrittori del traffico, che conterranno ovviamente informazioni sul rate (pacchetti x unità di tempo). Nelle chiamate telefoniche il traffico è regolare (commutazione di circuito). Nelle reti il traffico è impulsivo, intermittente (commutazione di pacchetto). Abbiamo una caratteristica di BURSTNESS. La rete si deve quindi regolare: deve capire se ci sono delle risorse necessarie. I buffer sono i relativi ammortizzatori. Evitando sempre il buffer overflow naturalmente. Capire che burstness hanno questi particolari flussi nella rete. Rate medio e burstness (livello di intermittenza del raffico) sono quindi i parametri più significativi. Se ci sono i prerequisiti, allora il contratto sarà OK ed il flusso sarà autorizzato, altrimenti verrà rigettata la richiesta. La rete si preoccuperà poi di verificare che la sorgente rispetti la policy dei descrittori. In caso contrario, agli access node controlleremo i pacchetti, scartandoli oppure marcandoli per essere via via scartati. *Reservation* è però un modello non scalabile.

Serve una prenotazione per ogni singolo flusso. Mantenere informazioni di stato su ogni singolo flusso della rete è ovviamente una soluzione poco scalabile. La granularità di controllo non è accettabile. *IntServ*, *DiffServ* (gestire flussi appartenenti a varie classi). Adesso si parla anche di Ethernet deterministico (QoS, prenotazioni). BUS CAN delle centraline veicolari. Ma non ci sono garanzie QoS!

1.1.1 TCP TRAFFIC CONTROL

Si tratta di un protocollo a livello di trasporto che offre un servizio trasporto dati affidabile. Consegna corretta nella corretta sequenza. Per far ciò si basa su un meccanismo RQ. Stabilire una connessione TCP significa instaurare un 3-way handshake. Esiste il TCP full duplex (bidirezionale). Send buffer, receive buffer e variabili di stato sono gli elementi principali di una connessione. Il servizio TCP è detto *by-stream*. Insieme di byte che vanno a finire nel receive buffer. Si ricordi che TCP non numera i segmenti, ma i byte! (Numero di sequenza). Il funzionamento si basa su delle finestre (Send Window e Receive Window). La *Send Window* rappresenta l'insieme dei numeri di sequenza relativi a byte che possono essere mandati dall'altra parte. La prima parte della finestra contiene i cosiddetti *byte-in-flight*, ovvero i pacchetti in volo, non ancora riscontrati. Poi abbiamo i gruppi di byte liberi. Oltre un certo numero di sequenza non si può inviare. Receive window: insieme di numeri di sequenza legati ai byte che possono essere ricevuti correttamente. Al di fuori, non possono essere ricevuti.
rwnd: variabile di stato che contiene la dimensione della receive window. Molto importante per il controllo di flusso. Formato segmento TCP: campo '*window*', *Acknowledgement number*. Separazione prossimo byte atteso e campo del controllo di flusso. Con questi campi un receiver può limitare la velocità del sender. Un receiver metterà il valore di rwnd nel campo window. Quando l'altra entità TCP riceverà un corrispondente pacchetto TCP, allora dovrà fare in modo che la sua finestra di invio soddisfi alla seguente disequazione:

$$LastByteSent - LastByteAcked \leq rwnd$$

che limita sostanzialmente la mole di dati in volo (non ancora riscontrati).

In pratica, analizziamo il controllo congestione di base del TCP (end-to-end). Il TCP se ne accorge quando vi sono delle perdite dei pacchetti. Tutto si basa sulla notifica di un *elemento perdita* (congestione della rete, ovvero quando qualche router ha scartato un qualche segmento, qualche buffer si è riempito). Esaminiamo più nel dettaglio:

- **Timeout**: (timer di ritrasmissione scade, meccanismo RQ del TCP): Un timeout rappresenta un elemento di perdita, ed è anche un segnale di congestione pesante;
- **3-ACK**: un altro elemento di perdita è la ricezione di 3 ACK duplicati. Un ACK duplicato è un ACK inviato da un receiver quando riceve dati fuori sequenza. Il receiver deve quindi segnalarlo. Si riscontrano nuovamente i dati correttamente ricevuti. Lato sender, il TCP non reagisce subito, immediatamente.

Le entità TCP stanno sugli host! Il sender riceve 2 ACK duplicati, ma se ne riceve un terzo, allora effettivamente (molto probabilmente) quel pacchetto è andato perso. E qui entra in gioco il meccanismo del *FAST RETRANSMIT*, che sarebbe una trasmissione fatta in maniera tale da evitare il timeout del pacchetto, che sarebbe un altro elemento perdita come abbiamo già detto. Si basa anche su una finestra aggiuntiva, detta di congestione (*cwnd*). Tale variabile mantiene il valore della dimensione di questa finestra. Essa è legata al numero di byte in volo. Adopera un controllo di flusso congiuntamente ad un controllo di congestione. Adesso la precedente disuguaglianza diventa:

$$LastByteSent - LastByteAcked \leq \min(rwnd, cwnd)$$

In realtà la Congestion Window viene misurata in MSS (*Maximum Segment Size*). Viene misurata solo la parte dei dati! La MSS non considera ovviamente l'intestazioone TCP. cwnd è misurata in MSS. Sulla base di questo valore, il sender invierà i dati con un certo rate. La cwnd aumenta (gestita lato sender) alla ricezione degli ACK. Questo aumento dipenderà dalla velocità dell'arrivo degli ACK. Si parla di comportamento self-clocking. Si sintonizza con la situazione di congestione della rete. Se c'è congestione, il TCP sender aumenterà la cwnd lentamente. La aumenta sempre fino a che non si verifica un elemento perdita. Il TCP, per far questo, esegue un bandwidth probing. Aumenta cwnd e contemporaneamente fa il probing. Si blocca quando c'è l'elemento perdita, individuata con la sonda della banda a disposizione. Riducendo cwnd si riduce la dimensione della congestion window. Immaginiamo una connessione TCP ovviamente (3-way handshake). Il TCP inizia con un aumento esponenziale della congestion window (aumento di +1 ad ogni ACK ricevuto). Nel nostro ragionamento non prendiamo in considerazione il controllo di flusso. Ricordiamo che l'MSS è la dimensione massima del segmento. Stiamo immaginando che non vi siano errori. Lato receiver immaginiamo che non vi siano ACK ritardati (accumulati). Invio non ritardato degli ACK $\implies$ $+1$ ACK $\implies$ $++$ $cwnd$ $\implies$. aumento 2^i esponenziale della finestra di congestione. *Slow Start* è quindi un nome abbastanza fuorviante. Ovviamente il TCP non va sempre avanti con lo slow start! Termina quando la cwnd raggiunge il valore *ssthresh*, che rappresenta la soglia dello Slow Start. Si entra quindi nella fase di Congestion Avoidance. Quando si ha un timeout (congestione pesante), la cwnd viene settata ad 1, e si imposta $[ssthresh = cwnd/2]$. Valore di ssthresh dimezzato rispetto al valore di cwnd quando si è verificato il timeout; con l'altro elemento perdita (3-ACK duplicati), avviene la ritrasmissione veloce, ed anche qui $[ssthresh = cwnd/2]$. Per la cwnd invece:

- SLOW START nella versione vecchia (come il timeout);
- Nelle nuove versioni TCP (*Reno TCP*), si entra in *Fast Recovery*, nel quale la cwnd riparte da un valore più alto.

Comunque transitino i segmenti, quando si entra in Congestion Avoidance, si aumenta di 1 per ogni *RTT*. Adesso abbiamo un aumento non più esponenziale ma lineare $\implies$ $(cwnd += MSS/cwnd)$. Per ogni RTT, la finestra sarà aumentata di 1! E rimaniamo in Congestion Avoidance sino a che non si ha un elemento perdita. Se l'elemento perdita è un timeout si va in Slow Start (si ricordi che quando l'elemento perdita è di questo tipo $\implies$ si va comunque in Slow Start), altrimenti nelle nuove versioni TCP si ha il seguente settaggio:

$$\begin{cases} ssthresh = cwnd/2 \\ cwnd = ssthresh + 3 \end{cases}$$

Il TCP aumenta quindi la finestra fino a che non vi è un elemento perdita. Più o meno l'andamento di $cwnd(t)$ nel tempo sarà un "dente di sega". Questo è il controllo di congestione end-to-end.

Di recente abbiamo un meccanismo network-assisted (ove è la stessa rete che fornisce un feedback esplicito alla sorgente (sender)). Meccanismo *ECN* (RFC 3168), che ha comportato delle estensioni nel TCP. Una caratteristica opzionale è la ECN - *Explicit Congestion Notification*, che sarebbe un feedback diretto attraverso il ricevitore. Meccanismo opzionale se entrambe le entità lo vogliono e lo possono supportare! Valido per qualunque protocollo che sia in grado di fare l'echo. Nel TCP può essere supportato, ad esempio. Con l'ECN, nell'intestazione

TCP sono stati aggiunti due nuovi flag. Per l'IP invece, due bit nell'intestazione (2 bit ECN). Nell'intestazione IP vi è il *ToS* (2 byte, IPv4). Nell'IPv4 questo campo era per la QoS. Livello di servizio differente. Campo ridefinito nell'architettura DiffServ (Internet che offre dei servizi QoS). Nell'ambito della seconda architettura (DiffServ), abbiamo 8 bit, di cui 6 servono per marcare ed i restanti 2 corrispondono ai bit ECN. I 6 bit sarebbero il *Differentiated Service Codepoint* (di default abbiamo *best-effort*, identificato con la sigla esadecimale $0x00$). Gli ultimi due bit servono per l'ECN:

- La sigla 00 vuol dire *ECN-not-capable*;
- $\{01,\ 10\}$ queste due cifre significano che l'entità è *ECN-capable*;
- La sigla 11 indica invece che è stata incontrata della congestione. Questo valore non viene mai settato dagli host, ma dai router intermedi.

Quando la destinazione incontrerà questi pacchetti allora farà l'echo.

Nel TCP abbiamo invece due flag: {ECE (*ECN-Echo*), CWR (*Congestion Window Reduced*)}. Durante il 3-way handshake le due entità negoziano l'utilizzo dell'ECN. TCP lato client inizierà il 3WHS inviando il SYN. In questo segmento, andrà a settare entrambi i flag ECN. Se l'altro host è preparato, ritorna il SYN-ACK con ECE settato e CWR non settato. Altrimenti se non è preparato, ECE e CWR entrambi non saranno settati. Se il receiver riceve dei pacchetti congestionati, allora setterà ECE, e tali pacchetti saranno echoati secondo il relativo protocollo.

Meccanismo definito di recente per il mondo TCP/IP. Meccanismo network assisted $\neq$ end-to-end. Prevede di introdurre 2 flag nell'header TCP e 2 bit nell'header IP. Per l'IP si è dato significato a due bit del vecchio campo ToS; adesso 6 bit sono dedicati al DiffServ CodePoint (di default $0x00$), il quale indica che il pacchetto appartiene ad una certa classe di traffico. 2 bit per l'ECN: {00 = Host not ECN capable; 01/10 se è in grado di supportarlo; 11 = congestione incontrata!}. Un router che incontra congestione setterà il bit ad 11. Tutto è nelle mani del TCP e viene negoziato nel 3-way handshake. Per quanto riguarda il TCP, il flag ECN-Echo viene utilizzato dal receiver per dire al sender che il pacchetto è congestionato. Poi abbiamo il flag CWR (Congestion Window Reduced), il quale viene utilizzato dal sender per avvertire che ha preso provvedimenti alla ricezione di un ECE.

Supponiamo che un'entità TCP ECN-capable avvii il 3WHS con un'altra entità. Si imposta $ECN = CWR := 1$. Il receiver invierà ECE a 0 o ad 1 a seconda del supporto ad ECN, e CWR non settato. Se il receiver dovrà echoare qualche pacchetto congestionato (bit 11), allora setterà ECE ad 1. Quando il sender riceve l'echo, lo notifica, lo prende come elemento di perdita; prenderà qualche provvedimento, magari riducendo la cwnd e setterà CWR ad 1. Il receiver non smetterà di inviare echo (all'arrivo di messaggi 11), sino a che non sarà ritornato un segmento TCP con CWR pari ad 1.

1.2 X-CASTING

1.2.1 MULTICAST

Implementazione del multicast nella rete. BROADCAST ROUTING ALGORITHMS, che vengono in seguito adattati al multicast. Partiamo dall'UNICASTING. Un solo nodo sorgente ed uno destinazione, relazione 1-1. I router lungo la rotta devono trasmettere i pacchetti solo lungo una interfaccia. Un'internetwork è una serie di reti fisiche collegate tra di loro mediante switch. Operazioni di forwarding che avvengono lungo una sola interfaccia. Il MULTICASTING prevede che un pacchetto sia da inviare ad un gruppo di destinazioni, che facciano parte dello stesso gruppo multicast. Come indirizzo mittente abbiamo sempre quello del mittente stesso.

Quando un router riceverà un pacchetto multicast, dovrà mandare il pacchetto lungo le interfacce opportune per raggiungere i relativi destinatari. I router devono opportunamente ragionare con protocollo Multicast. Con esso inoltriamo un pacchetto su più di un'interfaccia. A questo punto un host che faccia parte di gruppi multicast avrà più indirizzi. n indirizzi multicast. Indirizzi che lo individueranno come una destinazione, i.e. 226.14.18.7 . Indirizzo IP che lo individua come nodo all'interno della Internet più un altro indirizzo multicast. L'indirizzo di multicast è di destinazione ovviamente. Il multicasting di default non è utilizzato su Internet. Si possono utilizzare/creare delle isole multicast. Uso non diffuso comunque. Si può emulare il multicasting (utile per streaming partite ad es.) con la tecnica del *MULTIPLE UNICASTING*. In tal caso la sorgente si dovrà preoccupare di creare n pacchetti copiati per n destinazioni es. {D1, D2, D3}. La sorgente creerà 3 pacchetti, uno per ogni destinazione. Ovviamente questo comporta uno spreco di banda (più pacchetti di quelli necessari, al contrario di quanti ne servirebbero col multicasting). Situazione meno efficiente (si pensi anche ai ritardi, alla ridondanza (copie di pacchetti multipli)). Il multicasting ha importanza per molte applicazioni!

1.2.2 BROADCAST

Gli algoritmi utilizzati vengono riusati, adattati anche per il multicast. Broadcast 48 bit dell'IPv4 ad 1, oppure a livello 2 (ARP Ethernet). Algoritmo Link State (OSPF) e Distance Vector. Abbiamo nel primo caso i LSAS (*Link State Advertisement*). Si parla comunque di broadcast parziale! Nelle reti P2P c'è bisogno di spedire un pacchetto a tutti i nodi che facciano parte di quella rete. Implementazione: FLOODING. (livello 2 nei bridge, rudimentale tecnica di routing). Ritrasmissione delle trame su tutte le interfacce, diverse da quella di provenienza. Viene fatto il flooding se non vi sono entry nel DB. Approccio semplice per implementare il broadcast. Un router prenderà il pacchetto e lo ritrasmetterà su tutte le interfacce eccetto quella di provenienza. Le maglie sono a rischio di cicli, per definizione. BROADCAST STORM. Il flooding, se ci sono delle maglie, così com'è non può essere assolutamente utilizzato; non senza degli accorgimenti. Dev'essere controllato! Flooding controllato. Un modo è il cosiddetto *SEQUENCE-NUMBER CONTROLLED FLOODING*. In questo caso nel pacchetto broadcast viene messo l'indirizzo sorgente ed il numero di sequenza. Si tiene memoria del numero di sequenza del pacchetto, e se occorre di nuovo, allora esso non verrà floodato.

RPF (*Reverse Path Forwarding*). Un router, alla ricezione di un pacchetto, lo ritrasmette a tutte le interfacce (esclusa quella di provenienza), se e solo se l'interfaccia di provenienza si trova sulla rotta a minor costo per raggiungere la sorgente in unicast. Si parla sempre della SORGENTE ORIGINARIA! C'è uno spreco! Se su alcuni link viaggiano pacchetti che verranno poi scartati, ciò darà luogo ad un impiego di banda non necessario. Se si creasse invece uno *Spanning Tree* (letteralmente "albero ricoprente" la rete), potremmo risolvere questo problema: dato questo albero abbiamo che su ogni link transiterebbe solo una copia di ogni pacchetto. Un drawback c'è però! Numero di reti fisiche attraversate dal pacchetto. Alcuni nodi potrebbero esser raggiunti in un minor numero di nodi: questo comporta quindi un ritardo maggiore. Lo spanning tree viene associato ad una particolare topologia. DEVE essere poi seguito! Ci sono vari modi per costruirlo, ed uno possibile è il cosiddetto *CENTER-BASED*:

- 1) Un nodo centrale è definito, in base a qualche criterio;
- 2) Stabilito il nodo, i vari nodi della rete si dovranno preoccupare di inviare dei messsaggi di JOIN al center node, per entrare a farne parte. Saranno delle comunicazioni unicast. I percorsi seguiti dai pacchetti di join entreranno a far parte dello spanning tree, a meno che non intercettino delle rotte già appartenenti ad esso. In tal caso la porzione di spanning tree intercettata rimarrebbe invariata. I pacchetti di join servono esclusivamente a questo. Questo consentirà di non sprecare banda;

1.2.3 MULTICASTING

Il nostro focus è su Internet. Innanzitutto è un formato di indirizzamento (indirizzamento membri del multicast). IPv4. Serve un altro componente (protocollo per raccogliere informazioni a livello di appartenenza al gruppo multicast). IGMP (*Internet Group Management Protocol*). Dopodiché c'è bisogno di un protocollo per coordinare i gruppi Multicast, per creare degli alberi multicast. IGMP mette in relazione un host con il first hop. I router multicast sanno mediante IGMP se qualche host all'interno delle sue reti faccia parte di un gruppo multicast. Il classful addressing prevedeva anche la classe D in IPv4 (multicast, bit iniziali 1110). Blocco 224.0.0.0/4. $32 - 4 = 28$. Ci sarebbero altri 28 bit, ma 5 iniziali di questi 28 sono praticamente *unused*. 23 bit utilizzati alla fine. Primi 23 bit partendo dall'LSB. *Group Identifier*.

Un'interfaccia di rete elaborerà una trama se il multicast è abilitato alla ricezione. Si ragiona a livello 2. Indirizzi MAC di tipo multicast. Vi è ovviamente una relazione tra Indirizzo MAC multicast ed IP multicast. Il pacchetto IP raggiungerà la destinazione in base al livello Data Link (con opportuno imbustamento). Si parte comunque dall'indirizzo IP multicast di destinazione. Il MAC address partirà dall'LSB proprio con i 23 bit LSB. Il prefisso fisso MSB del MAC address è 01005E (fisso per MAC multicast). Indirizzamento di componente principale.

IGMP = {query, report, ...}. L'ultima versione IGMP v3 prevede solo questi due tipi di messaggi. Il router manderà ciclicamente dei messaggi di query. Un determinato host manderà un report alla ricezione di una query IGMP da un router. Se il router si accorge che un host che prima era in un gruppo multicast NON risponde ad una query, allora capisce che vuole lasciare il gruppo. Il router invia semplicemente dei messaggi di query. Si parla di *SOFT STATE* per il protocollo IGMP (l'informazioni di stato viene sempre refreshata di volta in volta, e quindi non è permanente). Informazioni di appartenenza. I messaggi di report contengono informazioni sui gruppi multicast cui gli host intendono appartenere. L'ARP non viene imbustato nell'IP (così come in RARP), ma a livello 2. Mentre i messaggi ICMP ed IGMP vengono imbustati in IP: {224.0.0.1 per query, 224.0.0.22 per report} come indirizzo di destinazione per query ovviamente. Il protocol field sarà settato a 2 ed avrà il TTL settato ad 1. Il router del first hop raccoglie quindi le informazioni di appartenenza.

Algoritmi di routing. Pensiamo a due approcci principali: {group-shared tree, source-based tree}.

- **GROUP-SHARED TREE**: $\exists!$ albero multicast condiviso da tutti i nodi della rete. Ci riferiamo al precedente Spanning Tree dei Broadcast routing protocols. Stesso approccio CENTER-BASED. Abbiamo un Multicast-tree che mette in comunicazione router che abbiano host collegati a loro facenti parte del gruppo (o gruppi) multicast. Router centrale (Router-CORE). MULTICAST-TREE. I pacchetti di join saranno sempre trasmessi in unicast. Il multicast tree è condiviso da tutti i membri del gruppo; quindi è un unico albero;
- **SOURCE-BASED TREE**: Ci saranno diversi Multicast Tree con l'algoritmo RPF. Per come funziona il tutto, allora alberi router not-attached, comunque potrebbero ricevere i pacchetti. Soluzione: packets pruning. Il router not-attached invierà questo pacchetto al router a monte.

Distance-Vector Multicast Routing Protocol (DVMRP) è un protocollo multicast di tipo source-based tree con quindi RPF e pruning. Il PIM è il più utilizzato (Protocol-Independent Multicast):

- PIM densa (PIM-DM): La maggior parte dei router sono attached, quindi avranno degli host, dei membri ad essi collegati. L'approccio è analogo a quello del DVMRP (source-based tree);

- PIM sparsa (PIM-SM): Se pochi router sono attached, allora si utilizza il group-shared tree. Situazione in cui abbiamo più efficienza. Diversamente si sprecherebbe banda per il viaggio dei messaggi di pruning.

Come può essere utilizzato il Multicasting su Internet? In Internet non è molto diffuso il Multicast. Si possono però creare delle relative isole. Quando ragiono con protocolli di routing ragiono con tutti i router multicast. Ci sarebbe il problema con il tunneling. Rotte che comprendono router non multicast. La topologia logica (rete di overlaying) che si viene a creare comprende solo i router multicast. Si prevede che il pacchetto multicast venga imbustato in un altro pacchetto IP unicast. Gli algoritmi di routing saranno utilizzati nelle reti logiche create.

1.3 IPv6

Novità rispetto ad IPv4: spazio di indirizzamento ovviamente più grande: indirizzi 128 bit. Intestazione fissa di 40 byte. I campi opzionali sono al di fuori dell'intestazione. Per semplificare il processamento (instradamento) dei pacchetti. Non c'è più un campo che riguarda la frammantazione od il checksum. I router non possono fare la frammentazione. Solo gli host possono. Attraversare una serie di reti fisiche hop-to-hop. Ricordiamo che la MTU è la *Maximum Transfer Unit*, e sarebbe la massima unità di trasferimento sulla rete fisica. In IPv4 ci pensavano i router. Es. Ethernet da 64 a 1518 byte. Prima il router interfacciato a quella rete Ethernet frammentava il pacchetto. Ed ICMPv4 mandava *Packet Too Big* alla sorgente, alla quale ricezione il pacchetto originario veniva scartato. Questo per velocizzare il processamento (speed-up). La checksum non c'è più. Funzionalità ridondante, dal momento che è presente anche a livello di trasporto TCP ed UDP. Anche a livello 2 è presente in realtà. In IPv4 al decremento unitario del TTL veniva di volta in volta controllata la checksum. Il TTL è ovviamente anche presente in IPv6. C'è anche il supporto QoS. Due campi: *Traffic Class* e *Flow Label*, come supporto alla QoS. Il Flow Label serve ad identificare un flusso con una certa etichetta. ToS IPv4, il quale è stato sostituito dal DiffServ Code Point. Ce lo ritroviamo anche qui. Il traffico viene quindi trattato in maniera diversa. Poi abbiamo anche il supporto alla sicurezza dati, tramite degli header di intestazione. Segretezza ed integrità dati possibile.

Indirizzamento su 128 bit; anche per IPv6 non si identificano i nodi ma le interfacce dei nodi. Il fatto che sia così grande significa che alla fine gli indirizzi saranno sicuramente sufficienti. L'entità IP su un host si preoccuperà di inviarlo all'host di destinazione. Livello rete: end system. Livello trasporto: processi applicativi; livello comunicazione nodi adiacenti: data-link. Non si parlava di NAT, o di Application Gateway (Proxy). Prima classful addressing (basato sulle classi). Concetto di Subnetting. Dopodiché si è pensato ad indirizzi privati e tecniche di adattamento alle reti pubbliche (NAT). Queste tecniche non erano però ben viste dai puristi. Con il NAPT il router lavorano anche sulle porte (livello trasporto). In IPv6 NON c'è bisogno più del NAT. Possiamo adottare sistemi di sicurezza end-to-end (es. IPsec). Introduzione importantissima: *Plug & Play*, ovvero autoconfigurazione ed autoassegnamento indirizzi IP.

Tiplogie di indirizzamento: {UNICAST, MULTICAST, ANYCAST}, ove al solito il multicast si riferisce ad un set di interfacce, gruppo di interfacce. In IPv6 il BROADCAST è un concetto particolare del MULTICAST. ANYCAST identifica sempre un gruppo di interfacce. Più interfacce associate a diversi host identificate con un solo indirizzo anycast. Il pacchetto arriverà soltanto all'interfaccia più vicina (the nearest one, tipicamente); es. load-sharing, misurazione certo costo di una rotta. Ci penseranno i router. È la rete che si preoccupa di tutto. Più entries per la destinazione. Gli indirizzi Anycast NON hanno un blocco a parte di indirizzamento ma sono presi dagli Unicast (dal blocco Unicast).

Tutte le interfacce dei nodi devono avere almeno un indirizzo Unicast di tipo LINK-LOCAL. LINK vuol dire Subnet IP sostanzialmente. Sempre associati agli indirizzi IP. Site = rete aziendale, es. scope differenti. Indirizzi IP con Scope Link, utilizzato per comunicare all'interno della Subnet. È tutto sempre autoconfigurato. Ci sono sempre le Subnet!

Gli indirizzi sono 128 bit rappresentati in esadecimale, con dei punti. Abbiamo 8 sezioni separate da due punti. Ogni sezione rappresenta 4 cifre esadecimali (4 hex digits x section). Abbiamo inoltre varie tecniche di abbreviazione: *Leading zeros*, i quali possono essere omessi. Attenzione! In una URL non abbiamo un nome simbolico, ma solo l'indirizzo IPv6 tra parentesi quadre (per evitare confusione con le porte!) es. http://[IPv6]:port/index.html.

es. porta 8888. Altra tecnica di abbreviazione: quando abbiamo sezioni-adiacenti fatte da zero possiamo utilizzare una SOLA VOLTA il "::". Non si può utilizzare più di due volte la *ZERO COMPRESSION*!

Esiste anche una Mixed Notation per il periodo di transizione: gli ultimi 32 bit fornivano l'indirizzo IPv4 (IPv4-MAPPED). La struttura, come IPv4, è gerarchica (non flat come gli indirizzi a livello 2). Notazione slash (CIDR), la quale individua un prefisso che caratterizza l'indirizzo IP. Il prefisso è quindi ben individuato da questa notazione.

Sono stati già assegnati dei blocchi, assegnati in base al prefisso, che caratterizzano indirizzi speciali: {Global Unicast, Site (rete di una certa organizzazione), Link Local, Multicast, Unique local unicast, Special addresses}. La notazione slash individua un prefisso, Ogni gruppetto di 4 bit è una cifra esadecimale, quindi abbiamo 16 bit per sezione, per un totale di 16x8 = 128 bit in totale, come previsto.

1.3.1 Tipi di indirizzi

- **Global Unicast**

  Global Routing Prefix. Sta ad indicare il Site e comprenderà più Subnet preferibilmente. Questo prefisso servirà per effettuare l'instradamento. Poi c'è il *Subnet Identifier*, che sarà sfruttato dai router all'interno del sito. Poi c'è l'*Interface Identifier* utilizzato per comunicare con la relativa interfaccia. Raccomandazione: primo campo 48 bit, secondo campo 16 bit. (48+16) = 64 bit. E per l'ultimo campo abbiamo 64 bit, in maniera tale che 64+64 = 128 bit. Quest'ultimo campo è qundi di 64 bit tipicamente, ed ha a che fare con l'autoconfigurazione. $\implies \{n = 48\ bits,\ m = 16\ bits,\ q = 64\ bits\}$.

  Come viene tirato fuori l'Interface Identifier? Un'interfaccia può avere più indirizzi IP. Potrei configuarla manualmente o con un DHCP server (configurazione stateful). La configurazione potrebbe invece anche essere automatica (senza DHCP). In tal modo quel campo viene costruito a partire dall'indirizzo fisico della scheda (LAN, MAC, Ethernet). Ultima parte (ultimi 64 bits). A partire dall'indirizzo HW della scheda. Due schemi di indirizzamento fisico: {EUI-64, MAC-48 (EUI-48)}. Adesso si parla di EUI-48 (48 bits). Potremmo anche pensare di generare l'II tramite un generatore di numeri pseudocasuali (per far fronte ad eventuali problemi di privacy).

  Gli indirizzi MAC sono di 48 bit. 6 byte in tutto. Primi 24 bit OUI (*Organization Unique Identifier*) + 40 bit di *extension identifier*. Ogni scheda verrà successivamente numerata con una numerazione progressiva. (5x8 = 40). 5 byte utilizzati per la numerazione progressiva della scheda. A partire dall'indirizzo fisico EUI-64 possiamo quindi derivare l'identificatore di interfaccia. Per questo scopo si parla di EUI-modificato. Si consideri il MAC a 48 bit. Il primo byte (quello che per primo viene trasmesso in rete), quello più a sinistra; si parla del bit LSB (meno significativo). Pensiamo ad un indirizzo MAC di destinazione. Ricezione della trama. L'interfaccia ricevente deve subito capire di che indirizzo si tratta (di gruppo (Multicast), od Unicast): {9 = indirizzo individuale (UNICAST), 1

= BROADCAST $\vee$ MULTICAST}. Bit preventivo per sapere come operare (es. switch flooding). Questo primo bit ha significato I/G. Poi c'è il secondo bit trasmesso U/L: {0 = Universal, 1 = Local}. Serve 0 se è quello universamente scritto sulle RAM, oppure 1 se utilizzato localmente (impostato tramite il driver della scheda di rete). Problema di elaborazione (RFC). Hanno poi modificato l'EUI-48 in EUI-64. Si chiama modificato perché l'U/L è settato ad 1 (EUI-64 modificato). Se partiamo dall'indirizzo fisico EUI-64, si setta l'U/L ad 1. Il problema nasce se ci troviamo dinanzi un EUI-48. Dobbiamo aggiungere quindi dei bit per arrivare a 64. Si aggiungono 16 bit (sequenza esadecimale FFFE). $(4x4 = 16\ bit) + 48bit = 64bit \rightarrow$ identificatore di interfaccia.

Examples

- 2000:1456:247 4/48. Si rappresentino i primi due blocchi che si riferiscono alla prima ed alla seconda subnet (Subnet Identifier). $\{0001_{16},\ 0002_{16}\}$. (Classful Inter-Domain Routing). Nessuno impedisce di mettere la Subnet 0000;
- Supponiamo che questo sia l'indirizzo fisico: $(F5-A9-23-EF-07-14-7F-D2)_{16}$. Si deve modificare il penultimo del primo byte (+ FFFE padding se eventualmente partiamo da EUI-48);

- **Indirizzo tutti 00000...**

  Serve durante il bootstrap (durante l'avvio in una macchina). Serve ad esempio quando si parla con il DHCP, es. interazione in 4 messaggi: *Discover Message* al 255.255.255.255 (BROADCAST), come mittente 0.0.0.0. Poi c'era l'offerta, l'host ne seleziona una ed invia la richiesta (con 0.0.0.0 sempre come mittente). Poi al successivo ACK abbiamo il get. $\leftarrow$ Interazione DHCP;

- **LOOPBACK**

  Tutti 0 tranne l'ultimo che è 1 (localhost). 127.whatever, utilizzato per il testing. (es. 0000::1/128);

- **Autoconfigurazione / Link Local**

  FE80:0:0:0:[interface identifier]. Può essere utilizzato da un host quando deve comunicare all'interno della subnet; oppure quando NON ci sono router. Attenzione: lo scope è LOCAL (livello subnet). La validità è la Subnet. I router non inoltreranno pacchetti del genere;

- **MULTICAST**

  indirizzo {permanente o transitorio}. Multicast si riferisce ad un grupppo di interfacce: {0: P, 1: T}. Vari tipi di indirizzi Multicast:

  - *Link-local all nodes*: FF02::1;
  - *Link-local All-router*: FF02::2;
  - *Site-local All-router*: FF05:2;
  - *Link-local con Solicited Node*: FF02::1:FFXX:XXXX;

In IPv6 a livello Network NON vediamo ICMP, IGMP, ARP e RARP. L'ARP non prevedeva l'imbustamento a livello Network (IP). In IPv6 abbiamo invece l'ICMPv6. Non abbiamo più IGMP e l'ARP (risoluzione indirizzi data-link). Esso ingloba tutte le restanti funzionalità

rimosse. Sono stati aggiunti anche nuovi messaggi. Ciò che faceva l'ARP ora lo fa l'ICMPv6. {ARP request: ICMPv6 *Neighbor Solicitation*, ARP reply: ICMPv6 *Neighbor Advertisement*}. Nell'ARP request, l'indirizzo logico sta nella parte TARGET. Quello che vanno a vedere i vari nodi; anche nel neighbor solicitation ci sarà il campo TARGET. I messaggi NS e NA sono imbustati in ICMPv6 in IPv6. Nell'ARP veniva utilizzato il BROADCAST. Qui invece si scomoda solo un gruppetto più piccolo di nodi (*Solicited-Node Group Multicast*) per migliorare (ridurre) il tempo di elaborazione. Ne sprecheremmo infatti parecchi utilizzando FF02::1. Si prevede quindi il meccanismo Solicited Node. Ogni sistema IPv6, per ogni indirizzo IPv6 deve entrare a far parte del gruppo solicited-node multicast, fatto dagli ultimi 24 bit dell'indirizzo IPv6 e da un prefisso fisso (FF02: 0:0.0.0 :1:FF). Più nodi potranno far parte di questo gruppo ovviamente. ICMPv6 Neighbor Solicitation. Tutto ciò serve per la risoluzione di indirizzi a livello 2 (scopo dell'ex ARP). Con l'ARP si sollecitavano tutti gli indirizzi. ICMPv6 è quindi ora imbustato in IP.

I moderni switch importanti hanno il supporto per l'IGMP (IGMP Snooping). Uno switch (*IEEE 802.11d*) fa il flooding comunque. Ma sarebbe bene non inviare a tutte le stazioni quando è Multicast. Riescono a capire quali stazioni sono associate le interfacce verso le quali troviamo host attached. Quindi non si fa un flooding a priori con le trame multicast.

A livello 2 cosa succede? Con IPv4 abbiamo anche un Multicast a livello 2. Si prendono gli ultimi 32 bit dell'IP multicast IPv6 e si aggiungono al prefisso 33-33. Multicast IPv6 $\rightarrow$ Multicast-Ethernet (previsto nella trama Ethernet).

1.3.2 AUTOCONFIGURAZIONE

Quando un host parte deve assumere un indirizzo Link-local. Parte da FE80:0:0:0:[ii]. Un computer vede dall'indirizzo hardware l'II e lo appiccica a quello di prima. Deve però accertarsi che sul Link sia l'unico ad avere quell'indirizzo a livello Link-local. Gli indirizzi hardware si possono modificare! Quindi l'unicità non è più garantita. Procedura per assicurarsi l'unicità necessaria richiesta. Si avvia la procedura DAD (*Duplicate Address Detection*). Si basa sul Neighbor Solicitation. Poi si avrà il *Router Solicitation* (per il GLOBAL UNICAST), ai quali messaggi i router risponderanno con dei *Router Advertisement* (RA). Comunque i messaggi RA sono periodici. Nei RA vi sono i prefissi! Ed a quel punto (prefissi associati a quel link), avremo: $\forall \underline{prefix} :< ii >$. DAD è utilizzata anche per autoconfigurare gli indirizzi di Global Unicast. SOLO gli host si configurano automaticamente con i Global! In IPv6 i router giocano un ruolo molto importante! Autorizzeranno loro stessi gli host ad autoconfigurarsi.

1.3.3 Formato del pacchetto

Si è previsto di avere una lunghezza dell'header costante. Versione del protocollo settata a (6). Anche in IPv4 c'è (4). Poi ci ritroviamo due campi (Traffic Class e Flow Label). Con quest'ultimo si può etichettare un flusso; sulla base di questo campo un flusso potrà essere trattato in una certa maniera. Il flusso non è ben definito (es. pacchetto TCP, sessione multimediale). Identificazione del flusso per la QoS. Traffic Class equivalente al ToS (In IPv4 ToS è stato ridefinito nell'ambito dell'architettura DiffServ). Ciò detto per l'IPv4 vale per IPv6 nel Traffic Class. Questi due campi servono quindi per la QoS. Nell'IPv4 c'è l'IHC (*Internet Header Length*), dato che l'intestazione è variabile. Nell'IPv6 l'intestazione è fatta da 40 byte fissi, non ce n'è quindi bisogno. In IPv6 c'è il campo *Payload Length* direttamente, anziché il *Total Length* dell'IPv4. La frammentazione riguarda l'elaborazione a monte nell'IPv6. In IPv4 abbiamo il *Protocol*, che codifica il protocollo (servizio multiprotocollo, TCP, UDP, es ICMP anche); perché a destinazione dovrà essere correttamente deimbustato. *Next Header* in IPv6 sarà per il protocollo. Non è detto che ci possano essere solo TCP, UDP (es. *Extension Header*, la quale aggiunge nuove

funzionalità come la frammentazione). Next Header codifica l'intestazione successiva. Ma se ci sono degli header di estensione, next header punterà piuttosto a dei nuovi header imbustati nel payload IPv6. IPv6 supporta la sicurezza con IPSec. L'extension header ha sempre un puntatore all'header successivo. Necessitiamo di conoscere quindi la lunghezza. Altre opzioni disponibili, es: *Source Routing* (decisione dei router da attraversare a monte), ESP, etc.. Il payload potrebbe quindi contenere questi header aggiuntivi. Fino a massimo 6 intestazioni. Il campo Protocol ce lo ritroviamo in IPv4. In IPv6 abbiamo quindi il Next Header. All'interno del pacchetto TCP troviamo quindi i numeri di porta, per spedire i dati al corretto processo applicativo. IPv6 non permette la frammentazione ai router intermedi → aumenta le prestazioni. L'oltrepassamento dell'MTU non è oggigiorno molto frequente. In qualche RFC è definita la MTU minima.

Nel passaggio IPv4 → IPv6, a livello Network abbiamo in IPv4 dei protocolli di supporto (IGMP, ICMP, ARP e (RARP)). In IPv6 è stato tutto inglobato in ICMPv6. Vari messaggi dell'ICMPv6. Quello che prima chiamavamo ARP Request e ARP Reply sono sostituiti rispettivamente da Neighbor Solicitation e Neighbor Advertisement, per i Link-local. Per ottenere un indirizzo Global Unicast, ci si serve invece dei *Router Solicitation* e *Router Advertisement*. Poi abbiamo anche il *Redirect Message*, il quale serve nel caso in cui ad una rete fisica siano collegati due o più router. Se un host è stato configurato con un certo Default Gateway, e l'host manda un pacchetto al di fuori della rete, vi sarà una CONSEGNA INDIRETTA. Se per raggiungere una certa destinazione è presente un altro router a costo minimo, allora il router suggerisce un altro Default Gateway con il messaggio di Redirect. Nei router CISCO può essere disattivata questa opzione comunque. Poi abbiamo {*Echo Request*, *Echo Reply*}. Serve per segnalare anomalie, oppure per verificare la raggiungibilità di host e router. Messaggio *Time exceeded*. Il TTL è stato sotitutito dall'*Hop Limit*. TTL fa pensare ad un tempo. Ma in realtà è sempre decrementato unitariamente! Proprio per questo è stato rinominato Hop Limit. Il limite lo sceglie l'host di default. 8 bit, quindi massimo 255 hop. [ICMPv6].

Router Advertisement. I router periodicamente inviano questo messaggio. Ma lo inviano anche quando ricevono un Router Solicitation da un host. Nel RA ci sono vari campi. Uno è il *Source-Link-Layer Address*. Il router annuncia il proprio indirizzo MAC. Senza scomodare meccanismi di Neighbor Solicitation da parte degli host. Poi abbiamo il *prefix information*, che include il prefisso vero e proprio, l'AAC (*autonomous address configuration*), che se settato l'host può creare un indirizzo accodandovi l'Interface Identifier. Poi abbiamo il *Valid Lifetime* ed il *Preferred Lifetime*. Quando una macchina parte si configurerà il suo Link Local Address. Dopodiché sulla base di RA riceverà dei prefissi: proverà a configurarsi (prefix + ii) per formare il Global Unicast. Supponiamo che la DAD abbia avuto successo. Partiranno quindi dei timer. Quando scade il PL quell'indirizzo sarà deprecato (spirato). Se scade questo timer, questo indirizzo IPv6 diventa deprecato. A questo punto, nuove sessioni di comunicazione non potranno essere fatte. Le nuove sessioni faranno riferimento ad indirizzi non deprecati. Se spira anche il Valid Lifetime (VL), allora anche le connessioni già esistenti che utilizzano indirizzi deprecati verranno "immediatamente" abbattute. Molto utile quando bisogna riconfigurare le interfacce (es. cambio di provider). Se un sito quindi dovesse voler cambiare l'ISP, in IPv4 tutto si baserebbe sull'LPM (*Longest Prefix Match*). Eventuale rinumerazione delle interfacce (nuovi indirizzi IP). Se quindi in IPv6 ad un certo punto vogliamo riconfigurare le interfacce, nei router advertisement da mandare, settiamo i PL a 0, automaticamente! Per le nuove sessioni di comunicazione, si utilizzeranno quindi i nuovi indirizzi. Più RA ovviamente, in base al numero di reti logiche. Ricordiamo che l'Autoconfigurazione spetta soltanto agli host! I {Router Solicitation, Router Advertisement} fanno parte dell'ICMPv6. L'indirizzo Ethernet multicast si costruisce partendo con 3333 ed accodandovi gli ultimi 32 bit dell'indirizzo multicast. Nel Router Advertisement abbiamo Prefix Information, VL e PL. Nei messaggi Neighbor Solicitation

abbiamo il *Target Address* (sempre messaggio ICMPv6). La checksum è presente in ICMPv6.

1.3.4 TRANSIZIONE IPv4-IPv6

Approcci Dual-Stack. I sistemi del genere IPv6 hanno anche lo stack IPv4. Se in mezzo a due router IPv6 vi è un nodo IPv4, abbiamo IPv6-IPv4, ed il router IPv6 effettuerà una trasformazione IPv6-IPv4. Si perderanno dei campi però. Non si può fare altrimenti senza cambiare lo stack. Per evitare tutto questo esiste anche un'altra soluzione: il **Tunneling**, IPv4 header che conterrà sostanzialmente IPv6 header ed il relativo payload. Questo permette che non vengano alterate delle cose nell'intestazione.

1.4 MULTIMEDIA e QoS

Multimedia networking. Dobbiamo parlare di applicazioni multimediali, i quali hanno requisiti diversi dalle applicazioni che hanno traffico elastico (es. Web, http, email, etc.). Traffico che si adatta abbastanza bene al cambiamento del throughput. Le applicazioni multimediali sono invece NON elastiche! (Non si adattano bene alle variazioni di throughput, es. VoIP) es. PCM (codifica con frequenza pari al doppio della frequenza di Nyquist, Filtraggio a 4kHz). Meglio risparmiare in banda, filtrando sullo spettro del segnale. Un'applicazione VoIP, 64 kbit/s non ha bisogno di moltissima banda! Ma dev'essere assolutamente quella. Requisito minimo sul throughput. Applicazioni del genere sono molto sensibili al throughput, al ritardo ed al delay jitter, altrimenti l'interattività non è ovviamente usufruibile. Sono altamente sensibili a variazioni sul ritardo ed al delay jitter ma al contempo tolleranti nei confronti del Data Loss.

1.4.1 Multimedia

I requisiti di servizio delle applicazioni multimediali sono: elevata sensibilità al ritardo end-to-end ed alle variazioni sul ritardo (delay jitter). Sono tolleranti a perdite occasionali di dati. Sono molto differenti dai requisiti delle tradizionali applicazioni elastiche, tipo e-mail, Web, FTP, i quali sono tolleranti ai ritardi ma ovviamente intolleranti alle perdite dati.

Abbiamo tre ampie classi:

- ***Streaming-stored*** *audio/video*;
- ***Streaming live*** *audio/video*;
- ***Real-time interactive*** *audio/video*;

Classi di applicazioni multimediali

- **Streaming-stored audio/video**

  Gli utenti richiedono i file video compressi on-demand, immagazzinati sui server (es. Youtube). Il contenuto multimediale è pre-registrato ed immagazzinato su un server.

  - *Interattività*: Dal momento che il file multimediale è pre-registrato, l'utente può pausare, posizionarsi in avanti, all'indietro, fare fast forward et similia sul contenuto multimediale (un protocollo, come l'RTSP (*Real-Time Streaming Protocol*) è richiesto);
  - *Streaming*: L'utente tipicamente inizia a riprodurre il video pochi secondi dopo che inizia a ricevere il file. Lo streaming evita che si debba interamente scaricare il file prima che si possa iniziare a riprodurlo;

– *Riproduzione continua*: La riproduzione dovrebbe avvenire in maniera tale da rispettare l'originale temporizzazione della registrazione. La riproduzione continua è possibile utilizzando *client-side buffering* o *prefetching*.

L'ammontare di ritardo aggiunto per ogni pacchetto dovrebbe fare in modo che il ritardo totale per pacchetto rimanga costante. Il client-side buffering ed il ritardo di playout compensano le variazioni sul ritardo (delay jitter). I dati devono essere ricevuti in tempo per il suo tempo di playout. È possibile fornire un continuos playout anche quando abbiamo delle fluttuazioni sul throughput, se il throughput medio (mediato su 5-10 secondi) rimane al di sopra del video rate.

I vincoli di ritardo end-to-end sono meno stringenti che quelli per applicazioni streaming live o real-time interactive.

- **Streaming-live audio/video**

  Un utente riceve in maniera live audio/video attraverso Internet (es. Internet radio o Internet TV).

  La distribuzione di audio/video live a molti riceventi può essere efficientemente conseguita utilizzando IP multicast. Al giorno d'oggi, le distribuzioni multicast sono conseguite per mezzo di un application-layer multicast o tramite unicasting multiplo. Come negli streaming-stored multimedia, il throughput medio dovrebbe essere più grande del tasso di consumazione del video.

- **Real-time interactive audio/video**

  Gli utenti in tal caso utilizzano Internet per comunicare interattivamente con degli altri. Esempi sono il *Voice over IP* (VoIP), conferenze video. Ovviamente sono altamente sensibili al ritardo end-to-end ed al delay jitter. I dati real-time su una rete switchata richiedono la preservazione delle relazioni di tempo tra i pacchetti della sessione. Il jitter può essere rimosso da questi tre meccanismi:

  – Preponendo ad ogni chunk di dati un *numero di sequenza*. Viene incrementato di uno per ognuno dei pacchetti che il trasmettitore genera. Verrà utilizzato al ricevitore per rilevare pacchetti persi od out-of-order;
  – Preponendo ad ogni chunk di dati un *timestamp*. Il trasmettitore "incolla" questo timestamp ad ogni chunk con il tempo al quale il chunk è stato generato;
  – Ritardando il playout dei chunk al ricevitore. È richiesto un *playout buffer*, ed il tempo di arrivo è così separato dal tempo di riproduzione

Protocolli per Applicazioni Interattive Real-Time

- ***Real-Time Transport Protocol** (RTP)*

  L'*RFC 3550* definisce il protocollo RTP ed il suo compagno RTCP. RTP permette di avere un formato di pacchetto standardizzato che includa campi per i dati audio/video, numeri di sequenza e timestamp. RTP risiede tra l'UDP e l'applicazione multimediale. I pacchetti RTP non sono limitati all'unicasting ovviamente, ma possono essere mandati sfruttando un albero multicast uno-a-molti o molti-a-molti. Gli stream RTP emanati da sender multipli in una videoconferenza appartengono ad una *sessione RTP*.

  Abbiamo quattro campi principali nel pacchetto: {*payload type*, *sequence number*, *timestamp* e *source identifier*}:

- *Payload type*:

  Il payload type sono 7 bits, ed indicano il tipo di codifica correntemente in uso. Il sender può cambiare la codifica on-the-fly durante una sessione;

- *Sequence number*:

  Sono 16 bit, utilizzati dal ricevitore per rilevare pacchetti persi o fuori sequenza, per l'appunto;

- *timestamp*:

  Sono 32 bit, e vengono utilizzati per ricreare la temporizzazione adibita lato ricevente utilizzando un playout buffer. Corrisponde al tempo quando il primo byte dei dati nel pacchetto è stato campionato. Il "timestamp clock" aumenta di uno per ogni periodo di campionamento. Ad esempio,per un PCM con 8 kHz di frequenza di campionamento (periodo di campionamento pari a $125\mu sec$), si avranno 160 campioni in un chunk.

  Il timestamp aumenta di 160 per ogni pacchetto RTP quando la sorgente è attiva; Il timestamp clock continua ad aumentare anche se la sorgente è inattiva.

- *Source Identifier (SRRC)*:

  Sempre 32 bit. Valore generato in maniera random che identifica univocamente la sorgente di uno stream RTP.

- ***Real-Time Control Protocol*** *(RTCP)*

  L'RTCP lavora congiuntamente con l'RTP. Ogni partecipante in una sessione RTP invia periodicamente dei pacchetti di controllo RTCP. I pacchetti RTCP contengono sender/-receiver reports. Riporta delle statistiche utili per l'applicazione: numero di pacchetti inviati, pacchetti persi, jitter di interarrivo etc.

  I feedback possono essere utilizzati per controllare le performance. Il sender può modificare i suoi parametri di trasmissione basandosi sui feedback.

  Per una sessione RTP, tipicamente abbiamo un singolo indirizzo IP multicast. Tutti i pacchetti RTP/RTCP appartenenti ad una certa sessione utilizzano il relativo indirizzo multicast. I pacchetti RTP/RTCP vengono distinti dagli altri mediante numero di porta: La porta RTCP DOVREBBE essere settata a quella RTP più uno (*RFC 3550*).

  RTCP *bandwidth scaling*: L'ammontare del traffico RTCP aumenta linearmente con il numero di riceventi. RTCP provvede a limitare il suo traffico a circa 5% della bandwidth della sessione.

- ***Session Initiation Protocol*** *(SIP)*

  Il SIP (*Session Initiation Protocol*) è un protocollo application layer simile all'HTTP. SIP prevede meccanismi:

  - per stabilire una sessione multimediale su una rete IP;
  - per stabilire un accordo tra i partecipanti sulla codifica da utilizzare;
  - per il chiamante per determinare l'indirizzo IP del callee;
  - per la gestione delle chiamate (per esempio l'aggiunta di nuovi stream multimediali durante la chiamata, cambiamento della codifica durante la chiamate, invito di nuovi partecipanti, trasferimento chiamate e hold delle chiamate).

  Può essere utilizzato su UDP o TCP con la relativa well-known port: 5060.

Impostare una chiamata ad un indirizzo IP noto

Il messaggio SIP INVITE di Alice indica il suo numero di porta, indirizzo IP, codifica che preferisce utilizzare (es. PCM μlaw). Il messaggio di Bob 200 OK indica il suo numero di porta, indirizzo IP e codifica preferita (GSM). Si supponga che Bob non abbia modo di utilizzare la PCM μlaw a causa della mancanza del relativo encoder. Bob quindi replicherà con un messaggio *606 Not Acceptable Reply*, elencando le sue codifiche disponibili. Alice quindi può adesso inviare un nuovo messaggio INVITE, proponendo una nuova codifica. Naturalmente Bob può sempre rifiutare la chiamata.

Tipicamente, Alice conosce solo il well-known address di Bob, e non il suo indirizzo IP. Sono quindi necessari server SIP intermedi, che forniscano un servizio di traduzione. Quando un messaggio SIP passa attraverso un dispositivo SIP (includendo il sender), il dispositivo attacca una linea aggiuntiva di header, denominata *Via*. Il messaggio INVITE di Alice specifica nell'header che il SIP user agent invia/riceve messaggi SIP su UDP.

Ogni utente SIP è associato con un SIP Registration Server (*SIP registrar*). Quando un utente lancia un'applicazione SIP su un dispositivo, l'applicazione manda un messaggio di *SIP register* ad un SIP registrar, informandolo del suo indirizzo IP corrente. Il registrar informa del successo della registrazione inviando una risposta che specifica per quanto tempo la registrazione è valida (parametro *expires*). Spesso i SIP registrar ed i *SIP proxy servers* girano sulla stessa macchina (host).

Traduzione di nomi e Posizione Utenti

Alice manda il messaggio INVITE al suo SIP proxy. Il messaggio contiene l'indirizzo: *sip:bob@poly.edu*. Il server proxy è responsabile per l'inoltro del messaggio al chiamato. Il proxy ritorna il messaggio di SIP response di Bob ad Alice. Il messaggio contiene l'indirizzo IP di Bob. Il SIP proxy effettua una richiesta DNS (DNS lookup) sul SIP proxy per "poly.edu" e quindi inoltra il messaggio INVITE ad esso. Basato sulle informazioni correnti di registrazione per Bob, il SIP proxy per *domain.com* può inoltrare la richiesta a Bob. In aggiunta a questo scenario, vi è anche il meccanismo di *SIP acknowledgements*.

Un *SIP Gateway* è un'applicazione che interfaccia una rete SIP con una rete che utilizzi un altro diverso protocollo di segnalazione. Un SIP gateway termina il percorso segnalativo e termina anche il percorso media. Un gateway SIP to PSTN termina sia il signaling che il media path. Gli user agent SIP e terminali H323 possono ad esempio scambiarsi direttamente informazioni multimediali RTP.

1.4.2 QoS - Quality of Service

Si tratta di fornire un differente livello di servizio ai vari flussi; sulla rete abbiamo pacchetti con requisiti differenti. Parametri: ritardo end-to-end, delay jitter, rate loss dei pacchetti, throughput (ritardo smaltito). Reti con supporto alla QoS. Classificazione e marcatura. Abbiamo una marcature di livello 3 (Architettura DiffServ). Marcatura di lvl 2, e vi deve essere matching tra le due. Il pacchetto IP hop-by-hop dovrà attraversare delle reti fisiche. Imbustamento in trame. Ethernet, rete switched. Gli switch dovranno inoltrare le trame guardando le intestazioni di livello 2. Scheduling, Disciplina di coda. Insieme delle regole di base alla quale viene trasmesso il prossimo pacchetto. Sistema a coda: fila di attesa + 1 servitore. I clienti arrivando nel sistema a coda, o verranno trasmessi (serviti), oppure dovranno attendere il loro turno. Regolazione dei servizi da erogare ai clienti. Disciplina di coda. È possibile adottare dei sistemi a coda per modellare i ritardi: {Ritardo di elaborazione (processing), ritardo di accodamento,

ritardo di trasmissione, ed eventualmente anche il ritardo di propagazione (fuori dal router)}. Queste componenti di ritardo possono essere quindi modellate con dei sistemi a coda. Privilegiare alcuni flussi rispetto ad altri. È questa la QoS. Ad esempio (*First-Come-First-Served*) (FCFS). Coda + trasmettitore (servitore). I pacchetti sarebbero i clienti. La coda è tipicamente di dimensioni limitate (come nella realtà). I pacchetti in arrivo possono essere discardati, scartati se la coda è piena. Non è una disciplina che può evidentemente supportare la QoS. Non ci può essere un trattamento speciale per un determinato flusso. Eventuale drawback per il ritardo associato ai piccoli pacchetti. Pacchetti piccoli che seguono pacchetti grandi saranno fortemente penalizzati. I grandi utilizzeranno una maggiore banda. Non ci sarà un trattamento imparziale, mentre noi vogliamo per l'appunto la QoS, ottenibile utilizzando una disciplina a priorità. In tal caso i clienti, all'arrivo del sistema a coda, sono divisi tra N classi di priorità, alle quali sono associati N sistemi a coda (N file di attesa), in base alle classi di priorità. Quando il servitore termina l'erogazione del servizio per un certo cliente, dovrà selezionarne un altro. Il primo cliente sarà quello in fila (in testa) alla classe di maggior priorità non vuota. Se non c'è nessuno passa alla prossima classe di priorità. Sistemi di tipo *work conserving* (non viene sprecato del lavoro in questo sistema). Se ci sono dei pacchetti (clienti), questi vengono serviti (Il servitore non rimane mai idle). Due classi di priorità es. Quando i pacchetti arriveranno al sistema a coda, verranno inizialmente classificati e successivamente inseriti nella coda associata alla classe di priorità cui effettivamente appartengono, purché essa non sia piena. All'interno della singola classe di priorità si agisce in FCFS. Servizio non-preemptive (senza interruzioni), ovvero il servizio di un cliente (la trasmissione di un pacchetto) non viene interrotto se durante l'erogazione di questo servizio arriva un cliente a più alta priorità. Potenziale drawback: starvation. Se c'è un flusso continuo di pacchetti ad alta priorità, quelli a più bassa priorità rimarranno sempre in attesa! Se vi è una modalità severa, si avrà il dropping dei pacchetti a bassa priorità. Alla fine i nuovi pacchetti a bassa priorità, incontrando una coda piena verranno quindi eventualmente droppati.

Round-robin Discipline

I pacchetti in arrivo sono suddivisi in un certo numero di classi (non di priorità). Il servitore servirà le classi ciclicamente. Disciplina sempre work conserving, ma qui non c'è una disciplina a priorità. In maniera ciclica serve le varie classi di pacchetti. Ma privilegia qualche classe in particolare? NO. Il sistema è fair a patto che la dimensione media dei pacchetti sia la stessa. I pacchetti di maggiori dimensioni consumeranno più banda. WEIGHTED FAIR QUEUING (WFQ). Verranno privilegiati alcuni flussi rispetto agli altri senza avere starvation. Abbiamo sempre delle classi di elementi, però associamo questa volta un peso alle varie file di attesa. Generalizzazione del round-robin. Pesi maggiori verranno associati a classi con maggior priorità. $\frac{w_i}{\sum_i w_i}$ sarà la frazione della banda relativa alla i-esima classe. Se abbiamo capacità di trasmissione di R pkt/s, il throughput sarà: $(R\frac{w_i}{\sum_i w_i})$. Eliminiamo la starvation con questo sistema, utilizzando il round-robin ma generalizzandolo introducendo degli appositi pesi.

POLICING e SHAPING

Supporto alla QoS. Meccanismi di regolazione del traffico. Il policing è una funzionalità di controllo da parte della rete. Accordo tra il Customer ed il Provider. Il customer richiederà dei pacchetti QoS (traffico con determinate caratteristiche con certi parametri da rispettare). Il provider dovrebbe impegnarsi a conseguire quella richiesta. Ovviamente il provider dovrà effettuare POLICING! Naturalmente sui suoi router. Rate alla quale un flusso inietta pacchetti sulla rete. Di seguito alcune definizioni:

- **AVERAGE RATE**: rate (rate medio a lungo termine) alla quale i pacchetti vengono inviati nella rete es. 6000 pkt/min VS 100 pkt/s. Il flusso 2 sarà più vincolato. Entrambi hanno però il medesimo average rate;

- **PEAK RATE**: velocità di picco. Vincola la velocità su un periodo di tempo più piccolo. Peak rate di 1500 pkt/s;

- **BURST SIZE**: Burstness (periodo di tempo piccolissimo). Prende in considerazione un tempo quasi istantaneo. Numero di pacchetti limitatissimo. Non sarà mai istantanea però la trasmissione. (e.s un burst di pacchetti NON deve essere superiore a 250 pacchetti);

SHAPING: È sempre una funzione di controllo, ma dalla parte del Customer. Si tratta di sagomare il proprio profilo di traffico in base a quanto dichiarato. Rispettare l'accordo. Smussamento dei picchi. Lo può fare in vari modi: un modo è ritardando opportunamente i pacchetti. Un policer tipicamente scarta i pacchetti in eccesso. Marcatura di QoS a livello minore. Lo SHAPER invece ritarderà i pacchetti. Come implementare il POLICING & SHAPING? TOKEN BUCKET è una possibile soluzione. I pacchetti non conformi saranno OUT-OF-PROFILE, diversamente IN-PROFILE. Un POLICER può scartare o marcare con QoS a più basso trattamento. TOKEN BUCKET. Regolare, implementare il POLICING o lo SHAPING. Controllare la velocità media a lungo termine e la burstness. Immaginiamo di avere un secchio con capacità c. Token (gettoni). Supponiamo che i gettoni vengano generati alla velocità di R $token/s$. Se il secchio si riempie i token generati vengono buttati via. Immaginiamo una coda di trasmissione. Un pacchetto può essere processato solo se c'è un token disponibile per lui. Per ogni pacchetto processato, viene eliminato un token. R è quanti token vengono generati al secondo. Modellazione dell'average rate. c è la dimensione massima della burstness permessa. c limita quindi il burst di pacchetti. Con questi due parametri possiamo fare Policing e Shaping. L'accordo tra Customer e Provider si baserà sulla Token Bucket. Un Customer dichiarerà dei parametri Token Bucket, e se il Provider accetta istanzierà un Token Bucket con questi parametri R e c. P & S. Fase di accordo (Customer contratta con il provider). È importante che si accordino. Se i pacchetti sono IN-PROFILE, allora avranno sempre dei token a disposizione! Diversamente non saranno conformi (OUT-OF-PROFILE). Per questo pacchetto si ha la possibilità di marcarlo o scartarlo. SHAPING possibile: controllo del traffico inviato. Con questi meccanismi, ad un intervallo temporale T, l'ammontare dei pacchetti NON potrà essere superiore ad $(RT + c)$. Nella pratica sarà utilizzato Average Rate (velocità a lungo termine) + Burstness. Flusso a lungo termine. La velocità di generazione dei token limita la velocità a lungo termine (AR). Limiterà quindi il ritardo di accodamento e quello di trasmissione. Weighted Fair Queuing: $(R(frazione\_contemplata(garantita) = \frac{w_i}{\sum_i w_i}))$. n classi alle quali sono associate delle code in uscita. Code HW nei router della CISCO. Meccanismo Token bucket. Un token bucket $\forall$ classe. Bisogna ovviamente prevedere le file di attesa. Parametri r_i e c_i. Accodamento + trasmissione. Ritardo massimo d_{max} =? In particolare per la classe i la banda a disposizione del flusso è $(\frac{w_i}{\sum_i w_i})$. Il caso peggiore è quello che vede un burst di pacchetti che consta di c_i pacchetti, ove questa è la dimensione dell'i-esimo secchio (bucket). I c_i pacchetti entreranno tutti in coda di attesa se abbiamo a disposizione c_i token. L'ultimo pacchetto sarà quello soggetto ad un ritardo più grande: $(d_{max} = \frac{c_i}{R * w_i / \sum_i w_i})$. Velocità di servizio > velocità di arrivo $\iff$ teoria delle code in generale. Qui invece abbiamo guardato al singolo sistema a coda. Qui abbiamo il controllo del ritardo massimo. I router di Internet adottano la RED (*Random Early Discard*). Quando la dimensione del buffer supera una certa soglia, casualmente vanno a scartare dei pacchetti. Riducono il rate a qualche segmento TCP ("chiudono dei rubinetti").

1.4.3 IntServ model

Sarebbe il QoS nelle reti IP. Internet viene utilizzato con diversi requisiti. IETF ideò *IntServ*, basato sul flusso. Prevede di andare a riservare la QoS desiderata $\forall$ flusso nella rete. Ma parliamo di Reservation, in termini di buffer, banda a disposizione. Il protocollo di prenotazione richiesto è il RSVP (*Resource Reservation Protocol*). Ci sarà una fase di Admission Control, durante la quale l'RSVP invierà dei messaggi di segnalazione. Cos'è un flusso? Stream di dati unidirezionale tra due applicazioni. Il flusso del Flow Label dell'IPv6 presenta il medesimo problema di ambiguità. Si basa su una certa architettura (ISA) che comprenderà alcuni componenti. Funzioni di supporto in background, e quelle di Forwarding (es. *Classification* e *Route Selection*). Dopodiché dovrà fare l'operazione di instradamento, per fare la quale bisogna fare preliminarmente l'operazione di interrogazione (previa creazione) delle tabelle di routing. Nelle funzioni di Background troviamo: {Admission Control, Reservation Protocol, Traffic Control Database, Management Agent}. Qui stiamo nella fase di elaborazione. Elaborazione di un pacchetto (dove un pacchetto dev'essere inoltrato). Qui bisogna tenere in conto tutti i parametri del QoS. L'OSPF consente di tener conto di questi parametri! A quel punto, quando arriva un pacchetto deve essere classificato! 4 based mechanism:

- 1) Classificazione;
- 2) Dominio del link al quale inoltrarlo;
- 3) A quale coda devo mandarlo?

Classificator & Route Selection. Una volta classificato e deciso dove esser trasmesso, entra in gioco il *Packet Scheduler* con le varie discipline di coda. L'RSVP include una specifica del flusso fatta da: Specifica del traffico (*TSpec*). Ha a che fare con la specifica del traffico (parametri del token bucket, Average Rate e Burstness). Poi abbiamo *RSpec* (*Resource Specification*), che riguardano i parametri della QoS, delay, delay jitter, loss rate, throughput. Infine abbiamo la *Service Class*. Sono presenti diverse tipologie di servizio:

- **Servizio garantito**: es. VoIP, con il quale si garantisce il delay jitter, la banda ed il loss rate per un certo flusso (garantito);
- **Controlled Load**: A patto che il carico non sia pesantissimo, riesce a dare un servizio migliore rispetto al:
- **Best Effort**: servizio di default sulla Internet;

Servizi del modello QoS. Molto complesso, presenta diversi drawbacks. I router devono ragionare con l'RSVP; poi Admission Control (dove si cercherà di riservare le risorse); mantenere informazioni di stato per ogni flusso. Refresh di queste informazioni, e questo comporta un elevato traffico di controllo. Soft-states. Messaggi RSVP pesanti quando $|pacchetti| \gg some\_threshold$. Architettura evidentemente NON scalabile. Nella pratica non funziona a causa della sua complessità.

1.4.4 DiffServ model

Non si guarda più al singolo flusso, ma alle classi di traffico (*DiffServ*). Viene fatta QoS per le varie classi. Pacchetti marcati. Entra in gioco lo SLA (*Service Level Agreement*). Anche qui c'è la specifica del traffico con parametri associati al Token Bucket. Si daranno le risorse alle varie classi! NON c'è garanzia per il singolo flusso ma per le classi di traffico. Soluzione che funziona, sebbene sia però poco granulare.

1.4.5 RECAP

QoS. Modello IntServ definito dalla comunità di Internet. Flow-Based Architecture. Riserva le risorse per il singolo flusso. Protocollo RSVP, mediante il quale il terminale invia la richiesta di prenotazione. {TSpec, RSpec, Service Class}. Tre classi di servizio. Successiva fase di Admission Control, durante la quale ogni router lungo la path verificherà la disponibilità delle risorse. Le risorse già impegnate per flussi precedentemente ammessi non le può toccare. Dev'essere fatto per ogni router lungo la path. Internet è quindi una rete datagram che supporta la QoS. Questa tecnica Stateful NON è però scalabile (i router devono mantenere informazioni di stato $\forall$ router). Riguarda il singolo flusso, e se il numero di flussi è troppo grande, ci saranno troppi messaggi di controllo da elaborare ogni volta. Il job principale è quello di inoltrare pacchetti dati. Si è quindi poi optato per l'architettura DiffServ, la quale è molto più scalabile. In tal caso (servizi differenziati) esso è un modello Class-Based. Il traffico è classificato in CoS (*Class of Services*), e l'appropriata QoS è applicata alle differenti classi. I router faranno parte di un dominio DS. Dovranno semplicemente adoperare un certo trattamento ai determinati pacchetti appartenenti ad una certa CoS. DS domain. IPv4 ToS field, adesso è stato ridefinito nell'ambito dell'architettura DiffServ. Ci dovranno essere dei router di frontiera che dovranno classificare e marcare i pacchetti. Sulla base del DiffServ CodePoint i router intermedi (interior) attueranno un PHB (*Per-Hop Behaviour*). NO informazioni di stato. Bisognerà avere un accordo tra il Provider del servizio ed il Customer (SLA = Service Level Agreement). Parametri QoS (parametro Token Bucket) + classe di servizio. I customer possono essere delle organizzazioni, ma anche altri provider di differenti domini DS. I router che non si interfacciano ai customer saranno chiamati *Interior router*. Molto più semplici. Regole di queueing e di scarto. I router di confine dovranno invece classificare, marcare, ma anche effettuare delle funzioni di policing! Devono non solo implementare il PHB (queueing sulla base della CoS), ma anche classificare il traffico, marcarlo, controllare che sia conforme all'accordo SLA, e se c'è del traffico non conforme, potrà scartare questo pacchetto oppure marcarlo con QoS minore. Un router boundary potrebbe fare funzioni di Shaper. Marcatura del pacchetto. Il nodo di confine dovrà quindi fare il marking, ma dovrà anche implementare il PHB. Tecnica LLQ (*Low Latency Queueing*). Tecniche WRED (un router inizia a buttare qualche pacchetto a caso. Agisce quindi in maniera preventiva). Un provider può diventare un customer per un altro provider! SLA tra vari provider. Il gestore del secondo dominio proporrà un certo traffico al terzo dominio. Eventuali funzionalità di SHAPING sui boundary nodes.

PHB definiti:

- **EF (*Expedited Forwarding*)**: Top del servizio. es. VoIP. Richiede basso ritardo, basso jitter, banda garantita, basso loss-rate. DSCP raccomandato: 46;
- **AF (*Assured Forwarding*)**: (AF xy). $x \in \{1, 2, 3, 4\}$. Parliamo di classi di servizio AF1, AF2, AF3, AF4 (best service). AFx. y: precedenza nel discard qualora si verificassero situazioni di congestione: {AF41, AF42, AF43}. Il peggiore di tutti è praticamente l'AF13. Massima precedenza nel discard. AF corrisponde più o meno al Controlled Load dell'IntServ;
- **Best Effort**: PHB di default. (0x00 DiffServ CodePoint);
- **Class Selector**: inserito per retrocompatibilità nel ToS (old IPv4). Primi 3 bit degli 8 bit. IP precedence old field (vecchio campo). Questo Class Selector prevede dei CodePoint fatti in questa maniera (XXX000). Possibile presenza all'interno di un dominio di un vecchio router eventualmente ToS enabled.

Sarà fatto un adeguato mapping tra marcatura lvl 2 e lvl 3. Marcatura lvl 2 della trama Ethernet: {PCP, DE, VLAN ID} = TAG di 4 byte. PCP: *Priority CodePoint*, Standard implementato nell'802.1Q/p. Mapping semplice (primi 3 bit a lvl 3) = PCP del lvl 2. Top del servizio;

Capitolo 2

TEORIA DELLE CODE

2.1 PROCESSI STOCASTICI

Modello matematico di tipo probabilistico utilizzato per andare a descrivere dei fenomeni casuali che possono essere rappresentati come funzioni di un parametro che solitamente è il tempo. $\{X(t),\ t \in T\}$. Famiglia di variabili casuali $X(t)$, indicizzate dal parametro temporale $t \in T$. Potrebbe anche essere qualche altra grandezza. Le v.a. sono definite su un unico spazio campione e che assumono valori in un certo insieme S. I valori assunti sono detti stati. $S =$ spazio degli stati del processo. Un processo stocastico è un insieme di funzioni del tempo. $X(t)$ generica variabile casuale. $X(t) \ \in S$. Insieme di funzioni del tempo che vengono chiamate **Realizzazioni**. Spazio campione: possibili risultati.

$X(t) : T \mapsto S$ SAMPLE PATHS: realizzazioni. La cosa importante è che $X(t)$ è completamente specificato in termini probabilistici se posso scrivere la CDF congiunta (e la PDF). $X(t)$ famiglia di variabili casuali. Differenti realizzazioni. Un processo stocastico è completamente specificato con la CDF congiunta per un qualsiasi insieme (sottoinsieme) di v.a. estratte dal processo. Immaginiamo di estrarre n variabili casuali:

$X(t_i),\ i = 1, 2, \ldots, n$: estrazione di n variabili casuali. Per queste devo essere in grado di scrivere la CDF congiunta:

$$F_{\underline{\mathbf{X}}}(\underline{x}; t) := \Pr\{X(t_1) \leq x_1,\ X(t_2) \leq x_2,\ \ldots,\ X(t_n) \leq x_n\}$$

Tale è la CDF CONGIUNTA. Ricordiamo che data una variabile casuale ξ, abbiamo che: $F_\xi(t) := \Pr\{\xi \leq t\}$. Qui bisogna riuscire la scrivere la CDF congiunta per queste n variabili casuali estratte dal processo. In alternativa, potrei considerare la PDF congiunta, la quale è la derivata della CDF. Scriviamo la *ROW VECTOR NOTATION*, generalizzando le casistiche dimensionali:

$$\begin{cases} \underline{\mathbf{X}} = (X(t_1),\ X(t_2),\ \ldots,\ X(t_n)) \in \mathbb{R}^{n \times 1} \\ \underline{x} = (x_1,\ x_2,\ \ldots,\ x_n) \in \mathbb{R}^{n \times 1} \\ \underline{t} = (t_1,\ t_2,\ \ldots,\ t_n) \in \mathbb{R}^{n \times 1} \end{cases}$$

Ricordiamo, ancora: $[f_{\underline{\mathbf{X}}}(\underline{x}; t) = \frac{\partial F_{\underline{\mathbf{X}}}(\underline{x}; t)}{\partial \underline{x}}]$.

2.1.1 CATENE DI MARKOV

Pensiamo allo spazio degli stati S. Questo spazio può essere continuo o discreto. Una CATENA è un processo stocastico a stato discreto. Noi tratteremo le ***CATENE DI MARKOV***. t può essere continuo o discreto. Tempo continuo o tempo discreto, rispettivamente. Quando

il tempo è discreto, si parla di *SEQUENZA STOCASTICA*. Eventualmente potremmo avere: $X_n,\ n = 0, 1, 2, \ldots$ ovvero diverse componenti del sistema.

Noi studieremo le CATENE DI MARKOV a tempo continuo, dette CMTC. Categoria di processi stocastici. Per questi processi di MARKOV la relazione che intercorre tra le v.a. è molto semplice. Caratterizzazione molto semplice. Questi processi soddisfano la cosiddetta:

Definition 1. ***PROPRIETÀ DI MARKOV***

$$\Pr\{X(t) \leq x \mid X(t_n) = x_n,\ X(t_{n-1}) = x_{n-1},\ \ldots,\ X(t_0) = x_0\} = \Pr\{X(t) \leq x \mid X(t_n) = x_n\}$$

con $t > t_n > t_{n-1} > \ \ldots\ > t_0$.

Questo è un processo stocastico $\{X(t),\ t \in T\}$ che soddisfa alla proprietà di MARKOV. Si tratta di una CDF condizionata. Cosa mi dice? L'evoluzione futura del processo a partire dall'istante t_n non dipende dalla storia passata ma solo dallo stato finale del processo! L'evoluzione del processo da t_n in poi dipende soltanto da t_n. Lo stato in t_n riassume tutta la storia passata del processo. $\{X(t),\ t \in T\}$. Noi studieremo le catene di Markov a tempo continuo (stato discreto, tempo continuo).

Definition 2. ***CM omogenea***

Invariante rispetto a traslazioni temporali degli assi:

$$\Pr\{X(t) \leq x \mid x(t_n) = x_n\} = \Pr\{X(t - t_n) \leq x \mid X(0) = x_n\}$$

con $t > t_n$.

Sono sostanzialmente delle catene per modellare sistemi il quale comportamento NON dipende dal tempo di osservazione. Scelta possibile dell'arco temporale. OMOGENEITÀ.

Definition 3. ***COIMPLICAZIONE DELLA PROPRIETÀ DI MARKOV (tempo di soggiorno)***

$W_i,\ i \in S$. Uno stato i qualsiasi della mia catena dev'essere una v.a. priva di memoria. Quando una v.a. casuale soddisfa alla proprietà di assenza di memoria, vale il seguente:

$$[\Pr\{W_i > t + \tau \mid W_i > t\} = \Pr\{W_i > \tau\}]$$

Tale è la MEMORYLESS PROPERTY.

Se l'evoluzione futura del processo a partire da t_n non dipende dallo stato passato del processo, non dipenderà neanche dal tempo in cui ci è stato. L'unica distribuzione di v.a. continua che soddisfa alla proprietà di assenza di memoria è la

Definition 4. ***ESPONENZIALE NEGATIVA UNILATERA***

$$\begin{cases} [f_{W_i}(\tau) = a\,\mathrm{e}^{-a\tau},\ \tau \geq 0] \\ \begin{cases} F_{W_i}(\tau) = \Pr\{W_i \leq \tau\} = 1 - \mathrm{e}^{-a\tau},\ \tau \geq 0 \\ F^c_{W_i}(\tau) = \Pr\{W_i \geq \tau\} = \mathrm{e}^{-a\tau},\ \tau \geq 0 \end{cases} \end{cases}$$

Unica distribuzione di probabilità che soddisfa l'assenza di memoria. Vedremo in seguito come $a := -q_{ii}$ (velocità totale di uscita dallo stato i). Velocità o equivalentemente tasso di transizione.

Proprietà di assenza di memoria per una v.a. casuale (in particolare comporta conseguenze sul tempo di soggiorno in uno stato i di una CMTC):

$$\Pr\{W_i > t + \tau \mid W_i > t\} = \Pr\{W_i > \tau\}$$

Parliamone in generale.

Theorem 1. *L'unica PDF di una v.a. continua che soddisfa alla proprietà di ASSENZA di MEMORIA è l'ESPONENZIALE NEGATIVA UNILATERA (UEN).*

Dimostrazione. **UEN MEMORYLESS PROPERTY**

$$\Pr\{W_i > t + \tau \mid W_i > t\} = \frac{\Pr\{W_i > t + \tau,\ W_i > t\}}{\Pr\{W_i > t\}} = (\dots)$$

Dobbiamo dimostrare questa proprietà per la UEN: $W_i > t + \tau \supseteq W_i > t \implies$

$$(\dots) = \frac{\Pr\{W_i > t + \tau\}}{\Pr\{W_i > t\}} = \frac{\mathrm{e}^{-a(t+\tau)}}{\mathrm{e}^{-at}} = \frac{\mathrm{e}^{-at}\,\mathrm{e}^{-a\tau}}{\mathrm{e}^{-at}} = \mathrm{e}^{-a\tau} = \underline{\Pr\{W_i > \tau\}}$$

La parte sottolineata sarebbe quindi la CDF complementare $F^c_{W_i}(\tau)$ della UEN. □

A cosa è uguale il parametro a? È uguale ad $a = -q_{ii}$, che sarebbe la velocità (totale) di uscita dallo stato i, ovvero la velocità totale alla quale il processo cerca di uscire dallo stato i. Si scrive, posto $a := -q_{ii}$:

$$W_i \sim EXP(-q_{ii}),\ \forall i \in S$$

Quindi il sistema precedente diventa:

$$\begin{cases} [f_{W_i}(\tau) = -q_{ii}\,\mathrm{e}^{q_{ii}\tau},\ \tau \geq 0] \\ \begin{cases} F_{W_i}(\tau) = \Pr\{W_i \leq \tau\} = 1 - \mathrm{e}^{q_{ii}\tau},\ \tau \geq 0 \\ F^c_{W_i}(\tau) = \Pr\{W_I \geq \tau\} = \mathrm{e}^{q_{ii}\tau},\ \tau \geq 0 \end{cases} \end{cases}$$

La proprietà di assenza di memoria per W_i dice che, supponendo che il processo stia soggiornando sullo stato i da t unità di tempo, la probabilità che il processo mi soggiorni ancora per altre τ unità di tempo è pari alla probabilità che il processo soggiorni per τ unità di tempo. Posto τ = RESIDUO, definiamo $\Phi_{i,t}$ come tempo di soggiorno residuo nello stato i da t unità di tempo, ovvero il processo si trova già da t unità di tempo nello stato i! Quindi riscriviamo la MLP.

Corollary 1.

$$\Pr\{\Phi_{i,t} > \tau\} = \Pr\{W_i > \tau\}$$

La distribuzione del tempo di soggiorno residuo in un tempo t (LHS) è pari alla distribuzione del tempo di soggiorno (RHS).

$$\begin{cases} [(W_i \sim \Phi_{i,t}) \sim EXP(-q_{ii})] \\ f_{\Phi_{i,t}}(\tau) = -q_{ii}\,\mathrm{e}^{q_{ii}\tau},\ \tau \geq 0 \end{cases}$$

Nel tempo discreto la distribuzione che soddisfa la MLP è solo quella geometrica. Noi invece lavoriamo a tempo continuo. MARKOVIANITÀ: assenza di memoria nel processo markoviano. Tempo di soggiorno residuo e tempo di soggiorno non solo hanno la stessa medesima distribuzione esponenziale (UEN), ma sono distribuite anche con lo stesso parametro $-q_{ii} = a$!

Riscriviamo la proprietà di Markov a tempo continuo! (specializzazione): Stiamo lavorando con CMTC o CTMC (Continuos Time Markov Chain):

$$\Pr\{X(t_{n+1}) = x_{t_{n+1}} \mid X(t_n) = x_n,\ X(t_{n-1}) = x_{n-1},\ X(t_0) = x_0\} =$$
$$= \Pr\{X(t_{n+1}) = x_{t_{n+1}} \mid X(t_n) = x_n\}$$

$\forall x_k \in S,\ t_{n+1} > t_n > t_{n-1} > \ \dots\ > t_0$.

Chiamiamo il secondo membro probabilità di transizione: $[\Pr\{X(t_{n+1}) = x_{n+1} \mid X(t_n) = x_n\}]$. Supponendo che nel tempo t_n il processo si trovi in x_n, essa è la probabilità che al tempo t_{n+1} si trovi allo stato x_{n+1}. Ma essa non fornisce informazioni circa quello che potrebbe accadere nel frattempo tra t_n e t_{n+1}!

Definition 5. ***Probabilità di Transizione da i a j***

$$[p_{ij}(t,\theta) := \Pr\{X(\theta) = j \mid X(t) = i\}]$$

$\forall i, j \in S$, e supponendo naturalmente che $\theta > t$.

Se la CMTC è OMOGENEA, queste probabilità di transizione non dipendono dagli istanti di tempo, ma dalla differenza dei due istanti di tempo $[\tau := \theta - t]$.

$$[p_{ij}(t,\theta) := P_{ij}(\tau) = \Pr\{X(t+\tau) = j \mid X(t) = i\}]$$

Supponendo che all'istante di tempo t lo stato sia i, quella è la probabilità che dopo τ istanti di tempo lo stato sia j. Non ci riferiamo più necessariamente ai singoli istanti di tempo. Vale: $[\sum_{j\in S} p_{ij}(\tau) = 1]$. Distribuzione al tempo t: intendiamo l'insieme di queste probabilità:

Definition 6. ***Distribuzione al tempo t***

$$\pi_i(t) := \Pr\{X(t) = i\}\ \forall i \in S$$

Andiamo a considerare tutti gli stati del mio processo. Per il teorema delle probabilità totali questa probabilità la posso scrivere in tal modo:

$$\pi_i(t) := \Pr\{X(t) = i\} = \sum_{j\in S} \Pr\{X(t) = i \mid X(0) = j\} \Pr\{X(0) = j\} = \sum_{j\in S} p_{ji}(t)\pi_j(0)$$

Quindi questa distribuzione al tempo t la possiamo ottenere come somma di prodotti tra la distribuzione iniziale e la probabilità di transizione da j ad i. Possiamo inoltre, dato $\pi_j(0)$, conoscere qualsiasi CDF congiunta. Per ogni processo particolare posso scriverle facilmente... A tempo continuo le probabilità di transizione dipendono però dal tempo! $(\dots) = \cdot(t)$! Per semplificare la vita, sono state introdotte delle quantità legate a $p_{ji}(t)$, generalmente anch'esse dipendenti dal tempo, ma che, nel caso in cui la CMTC sia OMOGENEA, sono invece costanti. Queste quantità sono quindi costanti e sono chiamate TASSI o *VELOCITÀ DI TRANSIZIONE*. Ciò che NON è costante è invece la probabilità di transizione.

Tassi di Transizione

Definition 7. ***TASSI (o VELOCITÀ) di transizione***

Relazione che lega i tassi di transizione alle probabilità di transizione. Se la catena è regolare:

$$\begin{cases} \exists q_{ij} := \lim\limits_{\Delta t\to 0} \dfrac{p_{ij}(\Delta t)}{\Delta t}, \forall (i \neq j) \in S \\ \exists q_{ii} := \lim\limits_{\Delta t\to 0} \dfrac{p_{ii}(\Delta t) - 1}{\Delta t} \leq 0 \end{cases}$$

$\forall i, j\ \in S$.

Si può dimostrare che questi limiti, queste quantità, esistono se la CMTC è REGOLARE, ovvero se:

Definition 8. ***CMTC REGOLARE***

$$\forall X(0),\ [|\underline{transizioni(\Delta t < +\infty)}| < +\infty]$$

Questo deve valere per qualunque stato iniziale. $-q_{ii} \geq 0$. Supponiamo che lo stato del processo in t sia $i \iff X(t) = i$. La probabilità che in un intervallo Δt tendente a 0 ($\iff \Delta t \to 0$) vi sia una transizione al di fuori di i è: $[-q_{ii}\Delta t + o(\Delta t)]$. Ragioniamo invece su $p_{ii}(\Delta t)$: essa rappresenta la probabilità che lo stato presente sia i e tra Δt unità di tempo lo stato sia di nuovo i. Tale probabilità non mi dice nulla su tutte le possibili transizioni che ci possono essere tra t e $t + \Delta t$.

$1 - p_{ii}(\Delta t),\ \Delta t \to 0$ è la probabilità che dopo Δt unità di tempo lo stato NON sia più i, partendo da i (prob. che vi sia una transizione). Quando $\Delta t \to 0$, questa probabilità tende alla probabilità di leaving dallo stato i. Quindi invertendo la seconda equazione del precedente sistema troviamo che al limite essa è uguale a: $[-q_{ii}\Delta t + o(\Delta t)]$. La probabilità è tanto più grande quando q_{ii} è alto in modulo! Più velocemente esce, più è probabile che esca! Più grande è il valore $|q_{ii}|$, più il processo cerca di uscire velocemente, e quindi con maggior probabilità vi riuscirà $\implies |q_{ii}| \uparrow \implies$ processo più velocemente tende ad uscire.

$-q_{ii}$ è quindi la velocità alla quale un processo lascia lo stato i. In sostanza, $-q_{ii}$ rappresenta il numero medio di transizioni al di fuori di i $\forall$ unità di tempo
in cui il processo si trova nello stato i.

Prendiamo per esempio due diverse realizzazioni dello stesso processo: $\{X^{(1)}(t),\ X^{(2)}(t)\}$. Per $X^{(1)}(t)$ abbiamo diversi soggiorni nello stato i. Supponiamo che la somma delle durate di soggiorno in i facciano $1s$. Durante questa unità di tempo in cui il processo si trova in i, si trovano 7 transizioni al di fuori di i. Per $X^{(2)}(t)$, supponiamo lo stesso caso ma con 5 transizioni. Abbiamo detto che: $-q_{ii}$ è il numero medio di transizioni al di fuori dello stato i. Una transizione uscente segue un tempo di soggiorno. $-q_{ii}$ rappresenta quindi anche il tempo medio di soggiorno. Ricordando che $f_{W_i}(\tau) = -q_{ii}\, e^{q_{ii}\tau}$, abbiamo che:

$$\mathbb{E}[W_i] = \frac{1}{-q_{ii}}$$

Se calcoliamo il valor medio (la media) di $f_{W_i}(\tau)$, esce proprio ($\frac{1}{-q_{ii}}$); tanto più grande è q_{ii}, tanto basso sarà il tempo di soggiorno in media.

Significato di q_{ij}. Supponiamo che attualmente il processo si trovi in i. La probabilità che in un intervallo di tempo infinitesimo ($\Delta t \to 0$) vi sia una transizione ij, ($i \to j$), è uguale alla probabilità $q_{ij}\Delta t + o(\Delta t)$. Consideriamo $p_{ij}(t)$. Non mi dice nulla, essendo una probabilità di transizione, su cosa sia eventualmente accaduto nel rispettivo intervallo di tempo tra la transizione.

Accade quindi che:

$$[\lim_{\Delta t \to 0} \frac{p_{ij}(t)}{\Delta t} = q_{ij}]$$

(probabilità tanto più grande quanto più grande è q_{ij}). La velocità è q_{ij}. Rappresenta una velocità alla quale si verifica la transizione $i \to j$. Ecco perché viene detto TASSO (o VELOCITÀ) di TRANSIZIONE da i a j. Ma è anche in sostanza il numero medio di transizioni da i a j ($i \to j$) $\forall$ unità di tempo in cui il processo si trova nello stato ii Ragioniamo di nuovo sulle due funzioni rappresentanti due diverse realizzazioni dello stesso processo. Supponiamo sempre

che la durata complessiva di un soggiorno in i sommi temporalmente ad 1. q_{ij} si differenzia da $-q_{ii}$ in quanto rappresenta il numero medio di transizioni $(i \to j)$.

$$\sum_{(j\neq i)\in S} q_{ij} = \sum_{(j\neq i)\in S} (\lim_{\Delta t\to 0} \frac{p_{ij}(\Delta t)}{\Delta t}) = \lim_{\Delta t\to 0} \sum_{(j\neq i)\in S} (\frac{p_{ij}(\Delta t)}{\Delta t}) = (\dots)$$

Considerando uno stato i qualsiasi, questa sommatoria fa 1: $\iff \sum_{j\in S} p_{ij}(\Delta t) = 1$. Quindi:

$$(\dots) = \lim_{\Delta t\to 0} \frac{1 - p_{ii}(\Delta t)}{\Delta t} = \underline{\underline{-q_{ii}}}$$

La quale è la velocità totale di uscita dallo stato i. Introduciamo ora una seconda quantità: $[\tau_{i,j} = \frac{q_{ij}}{-q_{ii}}]$, ovvero la probabilità che il processo, lasciando lo stato i, faccia una transizione verso lo stato j.

Supponiamo che $q_{ij} = 10,\ q_{ik} = 30,\ q_{ih} = 60...$ mediamente vi siano quindi 100 transizioni uscenti. $-q_{ii} = \sum_{(j\neq i)\in S} q_{ij} = 100$. Abbiamo che:

$$\begin{cases} \tau_{i,j} = \dfrac{10}{100} = 0.1 \\ \tau_{i,h} = \dfrac{60}{100} = 0.6 \\ \tau_{i,k} = \dfrac{30}{100} = 0.3 \end{cases}$$

$\tau_{i,j}$ è quindi alla fine la probabilità che, supponendo di lasciare i, la transizione sia verso j. LASCIANDO LO STATO i, $\exists$ UNA TRANSIZIONE VERSO LO STATO j.

Distribuzione al tempo t. Ci sono dei sistemi di equazioni differenziali che legano la distribuzione al tempo t ai tassi di transizione:

$$\begin{cases} \underline{\pi_i}(t) := \Pr\{X(t) = i\},\ \forall i \in S \\ \dfrac{d\pi_i(t)}{dt} = \displaystyle\sum_{j\in S} q_{ji}\pi_j(t),\ \forall i \in S \end{cases}$$

Di queste equazioni differenziali ve n'è una $\forall s \in S$ della nostra catena. Nella maggior parte dei casi, non ci serve $\pi_i(t)$, ma una distribuzione a regime (costante). A regime serve la rispettiva distribuzione a regime (dopo un tempo molto grande). Ci sono delle condizioni in base alla quale $\pi_i(t) \to K \neq \cdot(t)$ (distribuzione di equilibrio, a regime). Ma quali sono queste condizioni?

2.1.2 Probabilità a regime

Quali sono le condizioni di esistenza delle probabilità di stato a regime? Introduciamo alcune definizioni. Consideriamo uno stato $i \in S$ della mia CMTC. Diciamo che lo stato $(i \in S)$ è *TRANSITORIO* se c'è una probabilità non nulla che il processo non torni più in quello stato dopo che esso viene lasciato. Si dirà *RICORRENTE* in caso contrario (se con probabilità 1 torni nello stato i dopo averlo lasciato). Ai fini della distribuzione di regime, a noi interessano i RICORRENTI (gli stati ricorrenti). Sia M_i il tempo medio di ritorno (o di ricorrenza nello stato i). Per tempo di ritorno si intende il tempo che passa da due ingressi consecutivi nello stato i. M_i dice quanto dura in media questo tempo (ovviamente guardando tutte le possibili realizzazioni, altrimenti sarebbe costante). Si consideri una finestra temporale (M_i), è contemplato in media un solo soggiorno in i! (UN SOLO tempo di soggiorno). Se questo valore

diverge ($\iff M_i \to +\infty$), lo stato è detto *RICORRENTE NULLO*. Se M_i converge parliamo di STATO *RICORRENTE NON NULLO*.

Consideriamo ora un sottoinsieme proprio dello spazio degli stati $A \subset S \mid \bar{A} \cup A = S$. A comprende una parte degli stati, e non può coincidere con $S \iff \bar{A} = S \setminus (A \neq \emptyset)$.

Vale:

Theorem 2. *A chiuso se:*

$$\sum_{i \in A} \sum_{j \in \bar{A}} q_{ij} = 0$$

Significa che una volta che il processo entra in A, NON esce da A! In particolare, chiamiamo *STATO TRAPPOLA* od *ASSORBENTE* uno stato i per il quale: $[q_{ii} = -q_{ii} = 0]$ (Il processo NON esce più da quello stato). Uno STATO TRAPPOLA corrisponde ad un insieme chiuso costituito da solo quello stato.

Definition 9. ***CM IRRIDUCIBILE***

Una CTMC è IRRIDUCIBILE se $\nexists$ insiemi chiusi $\iff$ Tutti gli stati COMUNICANO tra di loro. Da i a j ci arrivo (magari passando da altri stati), ma ci arrivo sempre prima o poi! $i \longleftrightarrow j \ \forall i, j \in S$.

Abbiamo quindi tre caratterizzazioni di stato: (tre diversi possibili tipi di stato):

- *TRANSITORI*;
- *RICORRENTI NULLI*;
- *RICORRENTI NON NULLI*.

Corollary 2. ***Omogeneità dei tipi di stato per CM IRRIDUCIBILE***

Qualora una CM sia IRRIDUCIBILE $\implies$ allora tutti gli stati sono dello stesso tipo.

Concetto molto forte. CATENA IRRIDUCIBILE $\iff$ tutti gli stati comunicano $\iff$ tutti gli stati sono dello stesso tipo. Ricordiamo:

$$[\frac{d\pi_i(t)}{dt} = \sum_{j \in S} q_{ji} \pi_j(t)]$$

Definition 10. ***PROBABILITÀ LIMITE***

$$[\pi_i := \lim_{t \to \infty} \pi_i(t)]$$

$\forall i \in S$.

Se una CATENA è OMOGENEA, IRRIDUCIBILE (Quando una CATENA è OMOGENEA i tassi di transizione sono COSTANTI), allora:

$$\exists \lim_{t \to \infty} \pi_i(t) := \pi_i \neq \cdot \pi_i(0)$$

Se vogliamo avere delle distribuzioni a regime, vogliamo che questa quantità esista finita (limite esistente e convergente). In particolare non devono dipendere dalle probabilità di stato iniziali. In tal caso il sistema di equazioni differenziali collassa in un sistema di equazioni algebriche lineari:

$$[\sum_{j\in S} q_{ji}\pi_j = 0]$$

$\forall i \in S$. Le derivate vanno quindi a 0 $\iff$ $\pi_i(t) \xrightarrow{t\to\infty} \pi_i$. Questo sistema è OMOGENEO (sicuramente comprende la soluzione nulla). Tante soluzioni quanti sono gli stati del processo. SE la soluzione nulla fosse l'unica soluzione, gli stati saranno tutti TRANSITORI o tutti RICORRENTI nulli e non avremmo quindi una distribuzione di regime ($\iff \nexists \pi_i$). Se così non fosse invece, in tal caso le soluzioni saranno un numero infinito, che differirranno per una costante moltiplicativa tra di loro. Se sono infinite (tutte linearmente dipendenti), tra tutte queste le filtriamo con la cosiddetta condizione di normalizzazione. In tal caso gli stati saranno tutti RICORRENTI NON NULLI.

Definition 11. ***ERGODICITÀ***

$$\begin{cases} \sum_{j\in S} q_{ji}\pi_j = 0, \ \forall i \in S \\ \sum_{i\in S} \pi_i = 1 \end{cases}$$

$\exists$SOL? Se esiste, in tal caso la CATENA è detta ERGODICA (proprietà di ERGODICITÀ). CMTC {OMOGENEA, IRRIDUCIBILE, (STATI RICORRENTI NON NULLI)}.

Innanzitutto, si verifichi l'OMOGENEITÀ guardando i tassi di transizione (diagrammi di stato). Devono essere costanti (OMOGENEA, IRRIDUCIBILE (stati dello stesso tipo)). Basterebbe, per verificare che gli stati siano TUTTI ricorrenti non nulli, verificare questa proprietà per UN SOLO STATO! Ma a questo punto, senza ragionare su M_i, ragiono sull'equazione e la risolvo, sperando di risolverla e di trovare UNA ed UNA SOLA soluzione.

Corollary 3. ***Caratterizzazione dell'Ergodicita'***
Se abbiamo una [CMTC OMOGENEA, IRRIDUCIBILE e con un numero di stati finito ($\iff |stati| < +\infty$)], sicuramente essa è ERGODICA].

Abbiamo detto che $\pi_i, \ i \in S$ sono le probabilità limite, QUANTITÀ valutate IN VERTICALE, ed hanno a che fare con le distribuzioni di regime. MA è anche la frazione di tempo a regime in cui lo stato sia i.

2.1.3 ERGODICITÀ

Condizioni di ERGODICITÀ per una CMTC: {OMOGENEA, IRRIDUCIBILE, con STATI RICORRENTI NON NULLI} $\implies$ $[\exists \lim_{t\to\infty} \pi_i(t) := \pi_i]$. Una volta veriifcate queste due proprietà, non c'è bisogno di verificare che gli stati siano tutti RICORRENTI NON NULLI. In maniera più semplice si tenta di risolvere quel sistema sperando che abbia una soluzione non banale. Diagramma dei tassi di transizione: {Cerchi = Stati, Archi = Transizioni, pesate con i tassi di transizione}. Se questi tassi non sono dipendenti dal tempo allora la catena è OMOGENEA.

$$[\underline{\pi_i} = \lim_{t\to\infty} (\pi_i(t) = \Pr\{X(t) = i\}), \ \forall i \in S$$

La distribuzione di regime ha ovviamente a che fare con le probabilità limite. Immaginiamo che la catena sia ERGODICA. π_i = probabilità che a regime lo stato sia i. Se la catena è ergodica, allora a π_i possiamo attribuire un altro significato: frazione del tempo (A REGIME) in cui lo stato sia i:

$$[\pi_i = \lim_{t\to\infty} \frac{T_i(t)}{t}]$$

dove $T_i(t)$ è il tempo trascorso dal processo nello stato i sino al tempo t. (SINGOLA REALIZZAZIONE). È come se stessimo guardando una singola realizzazione del processo. Presa una finestra $[0,t) \leftarrow T_i(t)$. Fino al tempo t. Se calcolo $\frac{T_i(t)}{t}$, allora ottengo la frazione di tempo nel quale il processo è stato nello stato i nell'intervallo di tempo definito dalla finestra temporale scelta.

es. $\pi_i = 0.2 \implies$ per il 20% il processo a regime si è trovato in i. $\pi_i\tau$ rappresenta il tempo in media nel quale il processo si è trovato nello stato i durante la finestra temporale di durata τ. π_i è da valutare sull'INSIEME delle realizzazioni.

Consideriamo $N(t)$ processo stocastico che rappresenta il numero di clienti (#) in un sistema a coda. Esaminiamo un certo numero di realizzazioni.

Dal momento che $\pi_i(t)$ è una probabilità, essa va valutata in verticale. Supponiamo che:

$$\begin{cases} n_R := |realizzazioni\ di\ N(t)| \\ n_i(t) := n_i = \big|realizzazioni\ con\ stato\ i\ nell'istante\ t\big| \end{cases}$$

$(\frac{n_i}{n_R})$ rappresenta la frazione delle n_R realizzazioni con stato pari ad i in t. Quando n_R cresce, $\iff n_R \uparrow$, $(\frac{n_i}{n_R}) \to$ probabilità che lo stato del processo sia i.

Rispettivamente:

Theorem 3. ***Probabilità Limite***

$$\lim_{n_R\to\infty} (\frac{n_i}{n_R}) = \pi_i(t) = \Pr\{X(t) = i\}$$

Tutto questo a regime! Abbiamo considerato π_i valutandola con un numero molto grande di realizzazioni. Guardando invece ad una singola realizzazione di tempo molto grande ($\iff t \to +\infty$), abbiamo che: $\underline{\pi_i} = \lim_{t\to\infty} \frac{T_i(t)}{t}$, che sarebbe la frazione in cui il processo si è trovato in i nella finestra temporale $[0,t)$.

Theorem 4. ***Mapping VERTICALE ORIZZONTALE***

Se c'è l'ERGODICITÀ abbiamo che:

$$\underline{\pi_i} = \lim_{n_R\to\infty} (\frac{n_i}{n_R}) = \pi_i(t) = \Pr\{X(t) = i\} = \lim_{t\to\infty} \frac{T_i(t)}{t}$$

Si consideri ora: $\bar{N} = \sum_{i=0}^{\infty} i\pi_i$. Questo è la media del numero di clienti. Numero medio di clienti del sistema a regime (si noti l'estremo superiore della sommatoria). Ma se il processo è ERGODICO, la media di insieme coincide con la media temporale. Nella pratica si suppone l'ergodicità, si procede in orizzontale. Si consideri $\lim_{t\to\infty} \bar{N}_t$. Senza il limite essa rappresenta il valore medio assunto da quelle funzioni: $\bar{N}_t := \frac{1}{t}\int_0^t N(\tau)d\tau$ (MEDIA TEMPORALE) durante la finestra temporale $[0,t)$. Se c'è l'ERGODICITÀ allora questa quantità, AL LIMITE di $t \to \infty$ coincide con $\bar{N} \iff [\lim_{t\to\infty} \bar{N}_t = \bar{N}]$ = VALORE DI REGIME.

ERGODICITÀ (RECAP)

$\{\pi_i,\ \pi_i\tau\}$ rappresentano rispettivamente la probabilità che a regime il processo si trovi in i, ed il tempo medio che il processo si trova in i nella finestra temporale di lunghezza τ. $M_i =$ tempo medio di ritorno nello stato i. Supponiamo di avere $\tau = M_i$. Sarebbe il tempo tra due ingressi consecutivi nello stato $i =$ tempo di ricorrenza. VALOR MEDIO M_i. È necessariamente un tempo di soggiorno. Quindi:

$\pi_i M_i$ = tempo medio trascorso nello stato i durante la finestra temporale $\tau = M_i$. Ma abbiamo un solo soggiorno durante $M_i \iff [\pi_i M_i = \mathbb{E}[W_i] = \frac{1}{-q_{ii}}]$. Invertendo troviamo: $M_i = \frac{1}{-q_{ii}\pi_i}$ (tempo medio di ricorrenza). Ricordiamo che:

$$\underline{f}_{W_i}(\tau) = -q_{ii}\, \mathrm{e}^{q_{ii}\tau},\ \tau \geq 0$$

Ed avendo un solo soggiorno $\pi_i M_i$ rappresenta il TEMPO MEDIO DI SOGGIORNO. Possiamo dare una formulazione matriciale dell'insieme di equazioni differenziali:

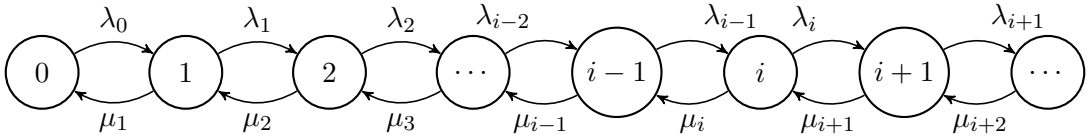
$$\begin{cases} \dfrac{d\pi_i(t)}{dt} = \displaystyle\sum_{j \in S} q_{ji}\pi_j(t),\ \forall i \in S \\ \dfrac{d\underline{\pi}(t)}{dt} = \underline{\pi}(t)\underline{Q} \end{cases}$$

$\underline{\pi}$ è il vettore riga delle probabilità di stato al tempo t, indicizzato da i. Nel caso di ERGODICITÀ ovviamente vale che: $[\underline{\pi}\underline{Q} = \underline{0}]$ (vettore nullo).

2.1.4 CMTC Nascita e Morte

Caso particolare di CMTC, utili per lo studio dei sistemi a coda. CATENE DI NASCITA E MORTE per una CMTC. Caso particolare di CMTC. Sono dette così perché si prestano bene a modellare una evoluzione, dinamica di una popolazione (dimensione della popolazione). Abbiamo incrementi o decrementi di stato UNITARI $\iff$ $\{i++,\ i--\}$. Ci interessano molto. Sono delle catene definite sullo spazio degli stati $S = \{0, 1, 2,\ \dots\}$. Tale spazio può essere di dimensione illimitata. Ma tipicamente lo spazio per noi sarà tale che $|S| < +\infty$. Catene caratterizzate da fatto che vi sono incrementi/decrementi solo unitari.

Sia $N(t) = i$, quello che può accadere è che: $\{N(t) = i \to i+1 \ \vee \ N(t) = i \to i-1\}$. Nel DTT (diagramma dei tassi di transizione), gli archi ricordiamo che rappresentano delle transizioni, e sono pesati dai tassi (o velocità) di transizione. DTT sarà fatta con cerchi che rappresentano gli stati, e gli archi rappresentanti le transizioni, pesati con i rispettivi tassi (o velocità).



Tipicamente in un DTT per una catena di questo tipo abbiamo che sopra ci sono le nascite (λ_i), e sotto vi sono le morti (μ_i). Rispettivamente:

Definition 12. ***Tassi di nascita e morte***

$$\begin{cases} \lambda_i := \textit{ tasso di nascita sullo stato } i \\ \mu_i := \textit{ tasso di morte nello stato } i \end{cases}$$

Se questi valori sono COSTANTI $\iff$ la CATENA è OMOGENEA. Se la catena è IRRIDUCIBILE: tutti gli stati comunicano $\implies$ $[\exists \pi_i = \lim_{t\to\infty} \pi_i(t)]\ \forall i \in S$. Il numero di stati però NON è finito! ($\iff$ $|S| = +\infty$). Quindi dobbiamo cercare di risolvere il sistema:

$$\begin{cases} \sum_{j=0}^{\infty} q_{ji}\pi_j = 0 \\ \sum_{i=0}^{\infty} \pi_i = 1 \ (CONDIZIONE\ DI\ NORMALIZZAZIONE) \end{cases}$$

Abbiamo che: $-q_{00} = \lambda_0$ (velocità totale di uscita dallo stato 0). Quindi: $-\lambda_0\pi_0 + \mu_1\pi_1 = 0$ per lo stato 0, poi dato che vale: $-q_{11} = \lambda_1 + \mu_1$, allora per lo stato 1: $\lambda_0\pi_0 - (\lambda_1 + \mu_1)\pi_1 + \mu_2\pi_2 = 0$. Genericamente allo stato i abbiamo:

$$"i" \rightarrow \ \lambda_{i-1}\pi_{i-1} - (\lambda_i + \mu_i)\pi_i + \mu_{i+1}\pi_{i+1} = 0$$

Queste si chiamano EQ. DI BILANCIAMENTO TOTALE. Noi scriviamo le equazioni in generale per una CMTC ERGODICA. $\sum_{j\in S} q_{ji}\pi_j = 0 \ \forall i \in S$. Proviamo a tirare fuori il termine della sommatoria per $j = i$:

$$\Longrightarrow \sum_{(j\neq i)\in S} q_{ji}\pi_j = -q_{ii}\pi_i$$

Generalizzando possiamo scrivere le:

Theorem 5. ***EQUAZIONI DI BILANCIAMENTO TOTALE***

$$\sum_{(j\neq i)\in S} q_{ji}\pi_j = \sum_{(j\neq i)\in S} q_{ij}\pi_i$$

Esse esprimono il fatto che a regime (od equilibrio), la frequenza delle transizioni entranti nello stato i eguaglia la frequenza delle transizioni uscenti dallo stato i. È sostanzialmente un altro modo di scrivere: $\sum_{j\in S} q_{ji}\pi_j = 0 \ \forall i \in S$. Il numero medio (per unità di tempo) di transizioni entranti nello stato i è uguale al numero medio di transizioni uscenti nello stato i. $q_{ji}\pi_j$ è la frequenza (quindi adimensionale) delle transizioni dallo stato j allo stato i. Numero medio di transizioni nello stato i (per unità di tempo).

q_{ji} = numero medio di transizioni da j ad i $\forall$ unità di tempo in cui il processo si trova nello stato j, mentre π_j è una frazione di tempo. Ricordiamo che scelta ad esempio una finestra temporale $\tau = 1s$, allora il prodotto $\pi_j\tau$ rappresenta il tempo medio trascorso durante questa finestra temporale ($\tau = 1s$) dal processo nello stato j. Se facciamo $q_{ji}\pi_j$ otteniamo invece il numero medio di transizioni da j ad i $\forall$ unità di tempo.

L'intera equazione dice che all'equilibrio la frequenza delle transizioni verso lo stato i è pari alla frequenza di transizioni uscenti dallo stato i. All'equilibro il flusso entrante nello stato i è pari al suo flusso uscente. Si eguaglia praticamente ($\forall$ stato), flusso entrante (IN) e flusso uscente (OUT). Quindi dobbiamo bilanciare il flusso caso per caso.

Consideriamo l'equazione di bilanciamento per il generico stato i:

$$(\lambda_{i-1}\pi_{i-1} + \mu_{i+1}\pi_{i+1} = IN) = ((\lambda_i + \mu_i) = OUT)$$

Per lo stato 0 abbiamo: $\mu_1\pi_1 = \lambda_0\pi_0$. Vale per qualunque CMTC ERGODICA.

Theorem 6. ***EQUAZIONI DI BILANCIAMENTO TOTALE generalizzate***

Possiamo considerare $(A \subset S) \supset \{statuses\}$. Si immagini di considerare come macrostati gli insiemi di stato $\{0, \ldots, i\}$. Valgono le eq. di bilanciamento totale generalizzate: Flusso uscente da A = flusso entrante in A:

$$\sum_{j\in\bar{A},i\in A} q_{ij}\pi_i = \sum_{j\in\bar{A},i\in A} q_{ji}\pi_j$$

Inoltre, ne deriva che per una BDCMTC abbiamo di conseguenza:

Corollary 4. ***EQUAZIONI DI BILANCIAMENTO TOTALE GEN. per BD-CMTC***

$$[(\lambda_i\pi_i)_{OUT} = (\mu_{i+1}\pi_{i+1})_{IN}]$$

Tentiamo ora di eguagliare il flusso sulla frontiera verticale tra due stati. Otteniamo:

Theorem 7. ***EQUAZIONI DI BILANCIAMENTO LOCALE***

$$\pi_i q_{ij} = \pi_j q_{ji}$$

QUESTE valgono soltanto IN CASI PARTICOLARI (BDCMTC). Le GLOBALI e quelle GLOBALI GENERALIZZATE valgono invece SEMPRE per una CMTC.

Si considerino queste equazioni trovate per una BDCMTC:

$$\begin{cases} \lambda_i\pi_i = \mu_{i+1}\pi_{i+1},\ i \geq 0 \\ \sum_{i=0}^{\infty} \pi_i = 1\ as\ C.N. \end{cases}$$

Quindi abbiamo: $\pi_{i+1} = \frac{\lambda_i\pi_i}{\mu_{i+1}}$. Prendiamo $(i = 0) \implies \pi_1 = \pi_0\frac{\lambda_0}{\mu_1}$. Andando avanti ricorsivamente troviamo:

$$\begin{cases} i = 1 \implies \pi_2 = \pi_1\dfrac{\lambda_1}{\mu_2} = (\pi_0\dfrac{\lambda_0}{\mu_1})\dfrac{\lambda_1}{\mu_2} \\ i = 2 \implies \pi_3 = \pi_2\dfrac{\lambda_2}{\mu_3} = (\dots) = \pi_0\dfrac{\lambda_0\lambda_1\lambda_2}{\mu_1\mu_2\mu_3} \\ i-1 \implies \underline{\pi_i} = \pi_0\dfrac{\lambda_0\lambda_1\lambda_2\dots\lambda_{i-1}}{\mu_1\mu_2\dots\mu_i} = \pi_0\prod_{k=0}^{i-1}\dfrac{\lambda_k}{\mu_{k+1}},\ i \geq 1 \end{cases}$$

Abbiamo quindi trovato le espressioni per le:

Definition 13. ***Probabilità a regime***

$$[\underline{\pi_i} = \pi_j\prod_{k=j}^{i-1}\frac{\lambda_k}{\mu_{k+1}}] = \cdot(\pi_j)$$

Adesso vogliamo mettere in gioco la condizione di normalizzazione: Quindi essa diventa:

$$[\pi_0 + \sum_{i=1}^{\infty}\pi_i = 1] \implies \pi_0 + \sum_{i=1}^{\infty}\pi_0\prod_{k=0}^{i-1}\frac{\lambda_k}{\mu_{k+1}} = 1 = \pi_0(1 + \sum_{i=1}^{\infty}\prod_{k=0}^{i-1}(\frac{\lambda_k}{\mu_{k+1}})) = 1 \implies$$

$$\pi_0 = \frac{1}{1 + \sum_{i=1}^{\infty}\prod_{k=0}^{i-1}(\frac{\lambda_k}{\mu_{k+1}})}$$

with:

$$\pi_i = [\pi_0\prod_{k=0}^{i-1}\frac{\lambda_k}{\mu_{k+1}}],\ i \geq 1$$

Corollary 5. *Se la sommatoria a denominatore di π_0 converge, allora la soluzione esiste. Quindi:*

$$\pi_i = [\frac{1}{1 + \sum_{i=1}^{\infty} \prod_{k=0}^{i-1} (\frac{\lambda_k}{\mu_{k+1}})}] \prod_{k=0}^{i-1} \frac{\lambda_k}{\mu_{k+1}}, i \geq 1$$

2.1.5 PROCESSO DI POISSON

È un caso particolare di BDCMTC nel quale i tassi di nascita sono costanti e pari a $\lambda_i = \lambda \neq \cdot(i) \ \wedge \ \mu_i = 0 \ \forall i \in S$. È sottinteso che il processo sia omogeneo. Il processo è di pura nascita. Lo spazio degli stati è il seguente: $S = \{0, 1, 2, \ \dots\}$. Tutti gli stati sono TRANSITORI (una volta uscito da uno stato NON vi ritorno più) $\implies \nexists(\pi_i \neq \cdot(t))$.

$$\begin{cases} \dfrac{d\pi_i(t)}{dt} = \displaystyle\sum_{j \in S} q_{ji}\pi_j(t), \ \forall i \in S \\ \underline{\underline{\pi_0(0)}} = 1 \end{cases}$$

dove l'ultima equazione del sistema rappresenta la condizione iniziale, sempre da rispettare. Per un processo di POISSON abbiamo che la distribuzione soddisfa alla seguente:

Definition 14. ***Funzione massa di probabilità per un Processo di POSSION***

$$\pi_i(t) = \Pr\{X(t) = i\} = \frac{(\lambda t)^i}{i!}\,\mathrm{e}^{-\lambda t}, \ t \geq 0, \ \forall i \in S$$

Questa espressione mi ricorda la distribuzione di POISSON con parametro ($a := \lambda t$). Avendo una v.a. distribuita con Poisson con parametro $a \implies a = \mathbb{E}[X]$. Disegnamo una possibile realizzazione del processo. Supposto che evolva a partire dallo stato 0. Abbiamo:

$\{\tau_1 = W_1, \ \tau_2 = W2, \ \dots, \ \tau_n = t_n - t_{n-1}\}$. Possiamo quindi scrivere la PDF del tempo di soggiorno negli stati:

$$[f_{\tau_n}(r) = \lambda\,\mathrm{e}^{-\lambda r}, \ r \geq 0]$$

ove ($\underline{\tau_n = \text{tempo di soggiorno negli stati}}$). Queste variabili casuali sono distribuite secondo Poisson. Il processo di POISSON modella gli arrivi dei pacchetti alle code dei router. Arrivi dei clienti in un sistema a coda. Supponiamo di utilizzare al posto di $X(t)$, $A(t)$. *ARRIVAL*. Posso dire che questo processo CONTEGGIA gli arrivi sino a t. PROCESSO DI CONTEGGIO DEGLI ARRIVI sino a t. τ sono tempi di INTER-ARRIVO (quanto tempo è passato tra l'arrivo del cliente (n-1)-esimo e quello n-esimo) $\iff [\tau_n = t_n - t_{n-1}]$ sono i tempi di inter-arrivo tra i clienti. SE STO UTILIZZANDO UN PROCESSO DI POISSON per modellare GLI ARRIVI, allora abbiamo che il tempo di soggiorno è sempre distribuito esponenzialmente in maniera negativa unilatera.

Corollary 6. ***Omogeneità del Processo di POISSON***

$$\begin{cases} \Pr\{A(t) = i\} = \dfrac{(\lambda t)^i}{i!}\,\mathrm{e}^{-\lambda t} \\ \Pr\{A(t) - A(s < t) = i\} = \dfrac{(\lambda \tau)^i}{i!}\,\mathrm{e}^{-\lambda \tau} \end{cases}$$

con $\tau := t - s$.

Il precedente corollario vale GRAZIE AL FATTO CHE LA CATENA È OMOGENEA.

RECAP

Processo di Poisson, caso particolare di una CMTC di nascita e morte (BDCMTC). Tipicamente utilizzato per modellare i clienti in ARRIVO ad un sistema a coda. In questo caso lo possiamo vedere come un processo di conteggio degli arrivi. I tempi di interarrivo sono

v.a. indipendenti identicamente distribuite con distribuzione esponenziale negativa unilatera. Rispettivamente, le variabili casuali $[\tau_n = t_n - t_{n-1}]$ avranno questa PDF (tempi di interarrivo):

$$PDF: \ f_{\tau_n}(r) = \lambda \, \mathrm{e}^{-\lambda r}, \ r \geq 0$$

ovvero che $\tau_n \sim EXP(\lambda)$, e che quindi $\implies \mathbb{E}[\tau_n] = \frac{1}{\lambda}$, che sarebbe quindi il tempo medio di interarrivo tra i clienti. ANALISI MARKOVIANA. Per quanto riguarda gli arrivi dei pacchetti nel processo si dice più che altro *SELF-SIMILAR* in presenza di burst. Nella realtà gli arrivi dei pacchetti non seguono ovviamente sempre il processo di Poisson. I frattali sono dietro i processi Self-Similar. Questo modello di Poisson non tiene conto solo del burstness. Se i pacchetti aumentano regolarmente, non ci sarebbe bisogno dei buffer. Gli switch ATM sono stati modellati secondo il processo di POISSON. Prevede una burstness, ma abbastanza regolare. Gli switch ATM avevano un modellato errato degli arrivi $\implies$ cattivo dimensionamento.

Theorem 8. ***RANDOM-SPLITTING***

Splitting su due processi può essere generalizzato per un $|processi| > 2$. Immaginiamo di avere un processo degli arrivi di POISSON $A(t)$, con velocità λ (velocità media di arrivo). Supponiamo di derivare due altri processi da questo: $\{A(t) \to A_1(t), \ A_2(t)\}$. Quando arriva un nuovo cliente, con probabilità p sarà assegnato ad $A_1(t)$, e con probabilità quindi $(1-p)$ al processo $A_2(t)$. Le assegnazioni sono INDIPENDENTI! Si ha: $[A(t) = A_1(t) + A_2(t)] \ A(t) = |arrivi \ da \ 0 \ a \ t|$. Gli arrivi si conservano! $\implies A(t) = A_1(t) + A_2(t)$. Si dimostra che $\{A_1(t), \ A_2(t)\}$ sono ancora di POISSON, con parametri rispettivamente: $\{\lambda_1 = \lambda p, \ \lambda_2 = \lambda(1-p)\}$. Inoltre tali processi sono indipendenti statisticamente tra di loro. Splitting casuale mi produce ancora dei processi di Poisson in virtù dell'indipendenza delle assegnazioni;

Theorem 9. ***POOLING***

Combinazione. Supponiamo $\{A_1(t), \ A_2(t)\}$ di POISSON indipendenti, con velocità $\lambda_1, \ \lambda_2$, e vogliamo fare il pooling (combino gli arrivi in un solo unico processo). Si dimostra che $A(t)$ è ancora di POISSON con parametri $\underline{\lambda = \lambda_1 + \lambda_2} \implies \underline{\underline{A(t) = A_1(t) + A_2(t)}}$. Quindi le velocità sono dei parametri associati ai processi del quale stiamo facendo la combinazione. Se ne considero n il discorso non cambia. Perfettamente generalizzabile.

2.1.6 RITARDO NELLE RETI DI DATI

Uno degli indici di prestazioni più importante è il ritardo medio (*mean delay*). Ritardo medio affinché dei dati fluiscano dall'host a destinazione. Per le applicazioni multimediali non è importante solo il ritardo medio in sé, ma proprio la sua distribuzione (CDF). Probabilità che il ritardo end-to-end sia inferiore a $100ms$ es. Teoria delle code ci fornisce degli strumenti tecnici molto importanti. Dovremo però effettuare delle ipotesi semplificative, ovvero creare un modello. Se gli arrivi si discostano un pochino dal modello di Poisson, allora le cose non andranno bene. Se invece di discostano molto, allora il modello è proprio sbagliato. Modelli che ricalcano, descrivono il sistema reale. Le richieste di chiamata alla centrale invece seguono molto bene il modello di POISSON! Oppure le richieste dati cellulari. Invece per il traffico di rete le cose non vanno affatto sempre bene. Non otterremo dei risultati molto accurati, ma in generale sufficientemente accurati. Gli switch ATM invece sono stati fallimentari, per dimensionamento per difetto dei buffer.

Consideriamo ora un generico link di comunicazione tra due nodi: inoltro dall'head node i al tail node j:

Abbiamo quattro componenti di ritardo:

- Ritardo di elaborazione;

- Ritardo di accodamento;
- Ritardo di trasmissione;
- Ritardo di propagazione.

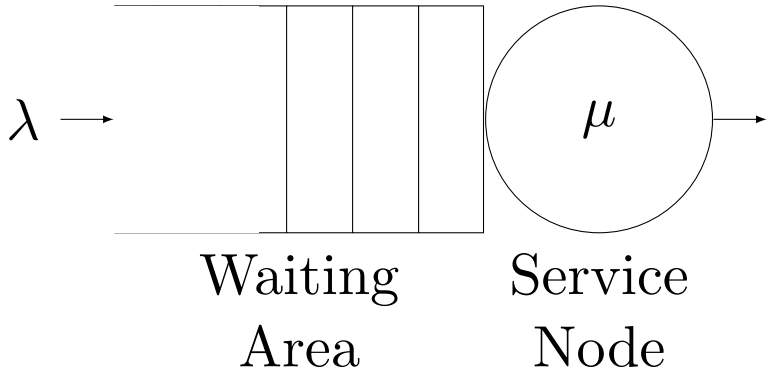
Il *processing delay* è il tempo necessario per decidere dove forwardare il pacchetto; Il *queueing delay* è il tempo di permanenza del pacchetto nel rispettivo buffer in uscita; Il *transmission delay* è il tempo necessario per trasmettere tutti i bit del pacchetto, ed infine il *propagation delay* è il tempo che ci mette un singolo bit del pacchetto a propagarsi lungo l'intero link.

Si consideri ora un singolo nodo ed un certo link bidirezionale (DUPLEX). Abbiamo in realtà due code (input e output). Il tempo del pacchetto nel buffer output sarà determinato dalla rispettiva disciplina di coda in atto. Il ritardo di elaborazione tiene conto della permanenza del pacchetto, verosimilmente, del ritardo di permanenza nel buffer input. Tipicamente però, al giorno d'oggi questa componente di ritardo è trascurata. Vi sono delle code hardware, router hardware (switch). Possiamo avere elaborazione CPU o direttamente in HW. Anche il ritardo di propagazione è tipicamente trascurabile, se i link sono sufficientemente vicini. Tipicamente il segnale si propaga a circa $(\frac{2}{3})$ della velocità della luce c. Se il link è satellitare, è chiaro che bisogna invece considerarlo (270-275 ms per salire ed altri 275 per scendere, e si consideri che un satellite può stare a 36000 km, con orbita geostazionaria).

Bisogna quindi vedere come sono fatti questi router. ASIC HW che implementano logiche di forwarding negli switch. Dipende dall'architettura dell'apparato. Il tempo di elaborazione è tipicamente trascurato quindi se la capacità elaborativa è molto alta.

Data una rete dati con una certa topologia, una rete di code è un insieme di code interconnesse che riflette questa topologia della rete dati.

2.2 SISTEMI A CODA



Un sistema a coda è un sistema costituito da una fila di attesa ed un centro di servizio. I clienti arriveranno al sistema a coda (dall'esterno), ed attenderanno il loro turno nella fila di attesa (dipendentemente dalla disciplina a coda). I router non è detto che siano tutti alla stessa velocità. Quando un qualche servitore si libera, allora dovrà servire un cliente. Notazione di KENDALL. Notazione che comprende 6 indicatori. Si presenta nella forma: {A/B/C/D/E/X $\rightarrow$ A/B/c/d/e - x}. Con il parametro A si va a caratterizzare il processo secondo il quale si susseguono gli arrivi dei clienti nel sistema a coda. Vari valori possibili per A. Se A è uguale ad M, significa *MARKOVIAN*, ed il processo degli arrivi è, come suggerisce la stessa parola, MARKOVIANO. Tempi di interarrivo v.a. indipendenti e distribuite esponenzialmente in maniera negativa unilatera. Se fossero identicamente distribuite ci troveremmo dinanzi ad un processo di POISSON.

D = *DETERMINISTIC* (tempi di interarrivo costanti), G = *General*. Vuol dire che il processo degli arrivi è di tipo generale (senza una distribuzione ben precisa). Distribuzione arbitraria generale. (MDG). Passiamo al secondo parametro, B. Con B si vanno a descrivere i tempi di servizio dei clienti nei sistemi a coda. B distribuzione dei tempi di servizio.

Vari valori: M = *Memoryless*, ove i tempi di servizio sono v.a. prive di memoria distribuite esponenzialmente in maniera negativa unilatera. D = *Deterministic*, quindi tempi di servizio costanti → Pensiamo alle reti ATM ad esempio, I pacchetti sono costanti, quindi i ritardi di trasmissione sono sempre costanti. G = *General*, al solito; Parametro C. Rappresenta il numero di servitori nel centro di servizio $\implies c := |servitori|$.

Con d (quarto parametro della denominazione di Kendall), indichiamo la capacità della fila di attesa. $\underline{d} := |buffer|$. Massimo numero di clienti nella fila di attesa (d). Se la fila di attesa è di dimensione illimitata $\iff (d = +\infty)$. Buffer di dimensione talmente grande la cui dimensione si può ritenere illimitata (parliamo sempre di modelli ovviamente, nulla di reale). In generale quindi $d \leq +\infty$. Se vale $(d = +\infty)$ potrebbe anche non riportarsi nella notazione. Su alcuni testi d potrebbe includere anche il numero di clienti presenti nel centro di servizio.

$(e := |popolazione|) \leq +\infty$. Rappresenta il numero di clienti che POSSONO arrivare nel sistema a coda. Al solito, se la dimensione della popolazione è illimitata $\iff e = +\infty$, allora potrebbe anche non riportarsi nella notazione. La X rappresenta invece la *disciplina di coda*, ovvero l'insieme delle regole che decidono il prossimo cliente da servire nella fila di attesa. es. {FCFS, LCFS, RR (round-robin), WFQ (più file di attesa in tal caso), PS (*processor sharing*)}. Quando non si esprime la X, la disciplina di coda di default è la FCFS.

Per ora studieremo l'*M/M/1*, ovvero Markovian/Memoryless/$(1 = |servitori|)$. Si suppone quindi FCFS, e $d = e = +\infty$.

Un risultato molto importante della teoria delle code è la ***Formula di Little***.

Semplicissima ma al contempo potentissima. Immaginiamo di avere un sistema generico, black-box; un sistema qualsiasi. I clienti arrivano ad una velocità di arrivo media λ.

Supponiamo che in condizioni di equilibrio nel sistema vi siano N clienti $\iff N := |clienti|$. In media, all'equilibrio N clienti. Supponiamo sempre all'equilibrio il tempo di permanenza dei clienti nel sistema sia T (tempo di permanenza medio a regime).

Theorem 10. ***FORMULA DI LITTLE***

In un qualsiasi sistema generico i clienti arrivano ad una velocità di arrivo media λ. Supponiamo che in condizioni di equilibrio nel sistema vi siano N clienti $= |clienti|$. In media, all'equilibrio N clienti. Supponiamo sempre all'equilibrio il tempo di permanenza dei clienti nel sistema sia T (tempo di permanenza medio a regime). Allora $\implies$

$$\underline{[N = \lambda T]}$$

Quelle tre grandezze vanno ovviamente intese come medie temporali (Media temporale). $N(t) = |clienti\ al\ tempo\ t|$. Osserviamo una singola realizzazione: scelta $[0, t)$ come possibile finestra temporale, abbiamo che: $N_t = \frac{1}{t}\int_0^t N(\tau)d\tau$ (valor medio). Prendiamo il limite: $\lim_{t\to\infty} N_t = N$, che rappresenta il numero medio di clienti nel sistema a coda. Ma se vale l'ERGODICITÀ, allora le medie temporali coincideranno con le medie di insieme (fatte su differenti realizzazioni).

$$\begin{cases} \pi_i(t) = \Pr\{N(t) = i\} \\ \underline{\mathbb{E}[N(t)]} = \sum_{i=0}^{+\infty} i\pi_i(t) \end{cases}$$

$N(t)$ v.a. discreta. Dobbiamo però parlare di valori a regime: $\lim_{t\to\infty} \mathbb{E}[N(t)] = \bar{N}$. È una media di insieme! Fatta sull'insieme delle realizzazioni. Quindi guardando alla formula di Little,

se c'è l'ERGODICITÀ, possiamo sostituire alle tre grandezze intese come medie temporali, le grandezze medie di insieme. [ERGODICITÀ]. Media di insieme.

Supponiamo che i pacchetti arrivino ad N nodi (Ingress Node). Normalmente ad essi sono collegati delle LAN. Abbiamo quindi una certa topologia arbitraria, rappresentante una rete di dati. All'interno vi siano dei certi nodi (NON DI ACCESSO), poiché non vi sono collegati degli host. Dati $\{\lambda_1,\ \lambda_2,\ \ldots,\ \lambda_n\}$, la velocità totale di arrivo sarà pari a: $[\lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_n]$. Essa rappresenta anche la velocità di arrivo media dei clienti in ingresso al sistema. Supponiamo che vi sua qualche meccanismo che ci permetta di valutare N (numero di pachetti medio all'interno del sistema all'equilibrio). Vale ovviamente: $[N = \lambda T] \implies [T = \frac{N}{\lambda}]$. Banalmente applichiamo Little a questo sistema. T è valutabile eventualmente utilizzando uno sniffer ad esempio. Se disponiamo di N, possiamo andare a calcolare il delay, ovvero il tempo medio affinché un pacchetto fluisca da un host mittente all'host destinazione (tempo medio di attraversamento). N_i è il numero medio di pacchetti relativi al nodo i; posso sempre applicare Little: $[T_i = \frac{N_i}{\lambda_i}]$. Focalizzandomi sui pacchetti relativi al sottosistema i quindi.

2.2.1 Sistema a coda M/M/1

Sistema a coda M/M/1. Approccio Markoviano. Diagramma tassi di transizione. *M/M/m/0* significa invece ad esempio che se tutti gli n router sono impegnati, la nostra chiamata verrà rifiutata. Con la notazione $M/M/1$ stiamo indicando un sistema a coda a singolo router, in cui il processo degli arrivi dei clienti è di POISSON a velocità λ. È un processo markoviano. Il processo degli arrivi dei clienti è quindi di POISSON (la distribuzione è la stessa per tutti, i.e. v.a. identicamente distribuite). Primo indicatore M, ovvero processo markoviano, e manca il quinto indicatore (dimensione popolazione illimitata $\iff e = +\infty$; si può quindi ritenere costante la velocità media degli arrivi). Se $e < +\infty$, non potrei parlare di POISSON, infatti quello OMOGENEO ha il parametro $(\lambda \neq \lambda(t)) \neq \cdot(t)$. Qualunque sia il tempo di interarrivo, $\tau_n \sim EXP(\lambda)$. Se tutte queste variabili casuali sono distribuite alla stessa maniera, allora il processo è effettivamente di POISSON. $(d, e = +\infty)$. Disciplina di coda di default, ovvero FCFS (Manca il sesto indicatore infatti, X).

Si suppone che i tempi di servizio siano mutuamente indipendenti (importante per la markovianità), ed indipendenti dai tempi di interarrivo tra i clienti del mio sistema a coda. λ è il parametro del processo di POISSON. $(\frac{1}{\mu})$ è invece il tempo medio di servizio. Si suppone che i tempi di servizio siano v.a. indipendenti, indipendenti dai tempi di interarrivo ed IDENTICAMENTE DISTRIBUITE! μ è la velocità di servizio di quel determinato servitore. μ è il numero medio di clienti serviti $\forall$ unità di tempo quando il servitore è costantemente occupato (che abbia sempre clienti da servire). μ è la velocità di servizio quindi (capacità in termini di servizio che può erogare). $S_n \sim EXP(\lambda) \implies$

$$\begin{cases} f_{\tau_n}(r) = \lambda \, \mathrm{e}^{-\lambda r} \iff \tau_n \sim EXP(\lambda) \\ f_{S_n}(s) = \mu \, \mathrm{e}^{-\mu s} \iff S_n \sim EXP(\mu) \end{cases}$$

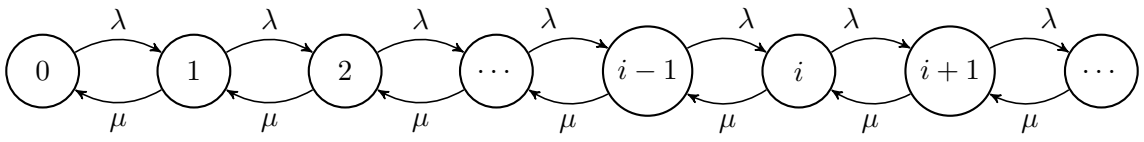
Ove i primi membri delle coimplicazioni del sistema sono le distribuzioni, rispettivamente la distribuzione dei tempi di interarrivo e quella dei tempi di servizio.

Si ricordi che quando abbiamo una v.a. distribuita in modo esponenziale, il reciproco del parametro è il suo valor medio $\implies \mathbb{E}[S_n] = \frac{1}{\mu}$, $\mathbb{E}[\tau_n] = \frac{1}{\lambda}$, ove l'ultimo rappresenta il tempo medio di interarrivo tra due pacchetti. Questo è quindi il sistema M/M/1, e lo studieremo con un approccio Markoviano.

Supponiamo $N(t)$ il numero di clienti all'interno dell'intero sistema al tempo t. Se un pacchetto è in fila di attesa ovviamente non può essere in fase di servizio. Assumeremo una catena di nascita e morte a tempo continuo (BDCMTC o CTMC-BD).

RECAP

$S = \{0, 1, 2, \dots\}$. Sistema a coda M/M/1. Processo degli arrivi dei clienti al sistema di POISSON con parametro λ, che rappresenta il numero medio di clienti in arrivo al sistema. Tempi di servizio v.a. prive di memoria indipendenti i.d. ed indipendenti dai tempi di interarrivo. μ numero medio di clienti serviti dal servitore quando esso è costantemente occupato. $N(t)$ è il numero di clienti in coda al sistema e dentro il centro di servizio al tempo t. Le variazioni di $N(t)$ sono determinate dai tempi di interarrivo e dai tempi di servizio. I tempi di interarrivo (ARRIVAL) costituiscono un processo markoviano, a differenza dei tempi di servizio. Si disegni il diagramma dei tassi di transizione:



Dimensione illimitata dello spazio degli stati ($\iff d = +\infty$). Catena di nascita e morte a tempo continuo. Abbiamo: $\{\lambda_i = \lambda,\ i \geq 0,\ \mu_i = \mu,\ i \geq 1\}$.

Valutazione (calcolo) dei tassi di transizione

Supponiamo $[N(t) = i > 0]$ (stato presente). Si ricordi che la CATENA è OMOGENEA. Siamo in uno stato i generico. Immaginiamo di non avere i tassi di transizione. La successiva transizione di stato è determinata o dall'arrivo di un nuovo cliente ($\iff ++N(t)$), oppure dalla fine dell'erogazione di un servizio ($\iff --N(t)$). L'intervallo di tempo che passa da t al prossimo tempo di arrivo del cliente è: $(\dots) := \xi_{R_t}$, ovvero il tempo di interarrivo residuo al tempo t. Dato che i tempi di interarrivo sono v.a. memoryless $\implies$ ξ_{R_t} memoryless, ed avranno la stessa distribuzione di τ_n. Tale variabile ci dice quanto manca ancora rispetto a t perché ci sia un altro arrivo.

Abbiamo quindi:

$$\begin{cases} \underline{\xi_{R_t}} \sim EXP(\lambda) \iff \tau_n \sim EXP(\lambda) \\ \eta_{R_t} \sim EXP(\mu) \iff S_n \sim EXP(\mu) \end{cases}$$

ove l'ultima implicazione rappresenta la definizione del servizio residuo al tempo t. Godono entrambe della PROPRIETÀ DI ASSENZA DI MEMORIA. Grazie al fatto che i tempi di servizio sono v.a. prive di memoria $\implies$ $\eta_{R_t} \sim EXP(\mu)$ memoryless anch'essa.

L'intervallo di tempo che passa da t e la successiva transizione di stato (determinata da $\{S_n,\ \tau_n\}$) del mio processo è: $\phi_i(t)$, ovvero il tempo di soggiorno residuo nello stato i al tempo t. Rispetto all'istante presente t, quanto tempo ancora soggiornerà il processo nello stato i? Quanto vi soggiornerà ancora? $\implies$ $\phi_i(t) = \min\{\xi_{R_t}, \eta_{R_t}\}$. MARKOVIANITÀ. I tempi di soggiorno sono v.a. distribuite esponenzialmente $\iff$ $\phi_i(t) \sim EXP(-q_{ii})$. Cerchiamo di determinare l'espressione della distribuzione di $\phi_i(t)$ in funzione di quelle due. Cerchiamo di scrivere la CDF complementare:

$$\Pr\{\phi_i(t) > \tau\} = [F^c_{\phi_i}(\tau) = \mathrm{e}^{q_{ii}\tau},\ \tau \geq 0] = \Pr\{\min\{\xi_{R_t}, \eta_{R_T}\} > \tau\} =$$

$$= [\Pr\{\xi_{R_t} > \tau,\ \eta_{R_t} > \tau\} = (\dots)$$

ove abbiamo opportunamente indicato l'evento congiunto. Dovrà congiuntamente accadere che entrambe siano maggiori di τ $\implies$ prodotto delle probabilità, data l'inter-indipendenza $\implies$

$$(\dots) = \Pr\{\xi_{R_t} > \tau\} \Pr\{\eta_{R_t} > \tau\} = \mathrm{e}^{-\lambda\tau}\, \mathrm{e}^{-\mu\tau} = \underline{\mathrm{e}^{-(\lambda+\mu)\tau}} = \mathrm{e}^{-(-q_{ii})\tau}$$

ove abbiamo sfruttato la conoscenza, rispettivamente, della CDF complementare dei tempi di interarrivo (residui) e la CDF complementare dei tempi di servizio (residui), data la MEMORYLESS.

Quindi $-q_{ii} = (\lambda + \mu)$ $\implies$ $q_{ii} = -(\lambda + \mu)$. La velocità totale di uscita è quindi pari a $-q_{ii} = \lambda + \mu$. Velocità totale. Andiamo a considerare tutte le velocità uscenti, abbiamo sempre $(\lambda + \mu)$, $\forall$ *state* i. Quindi per ottenere l'obiettivo dobbiamo considerare questa quantità:

$$[\tau_{i,i+1} = \frac{q_{i,i+1}}{-q_{ii}}] = (\dots)$$

che sarebbe la probabilità che, lasciando lo stato i il processo vada verso lo stato $i + 1$. Questo accade quando $[\xi_{R_t} < \eta_{R_t}]$. Quando accade questo, il processo degli arrivi PRECEDE il completamento (la fine) del servizio in corso.

$$(\dots) = \Pr\{\xi_{R_t} < \eta_{R_t}\} = \int_0^\infty \Pr\{\xi_{R_t} < \eta_{R_t} \mid \eta_{R_t} = y\} f_{\eta_{R_t}}(y)dy$$

ove abbiamo esplicitamente applicato il teorema delle probabilità totali nel continuo.

$$[\Pr\{y < \eta_{R_t} \leq y + dy\} = f_{\eta_{R_t}}(y)dy]$$

v.a. distribuita esponenzialmente: $\int_0^\infty 1 - \mathrm{e}^{-\lambda y} = (\dots)$;

Se consideriamo $[F_{\xi_{R_t}}(\tau) = 1 - \mathrm{e}^{-\lambda\tau}]$, dobbiamo valutare il precedente integrale derivato dal TPT in tal modo:

$$[(\dots) = \int_0^\infty (1 - \mathrm{e}^{-\lambda y})\mu\,\mathrm{e}^{-\mu y}dy = \int_0^\infty \mu\,\mathrm{e}^{-\mu y}dy - \mu\int_0^\infty \mathrm{e}^{-(\lambda+\mu)y}dy = (\dots)$$

Si badi che stiamo integrando una PDF su tutto il dominio! Quindi vale 1:

$$(\dots) = 1 - \frac{\mu}{\lambda+\mu}\int_0^\infty (\lambda+\mu)\,\mathrm{e}^{-(\lambda+\mu)y}dy = 1 - \frac{\mu}{\lambda+\mu} = \frac{\lambda+\mu-\mu}{\lambda+\mu} = \underline{\frac{\lambda}{\lambda+\mu}}$$

Molto semplicemente quindi $[q_{i,i+1} = \lambda]$! Per confronto. Potrei fare lo stesso ragionamento per $i-1$. Dato che la velocità totale è $(\lambda+\mu)$, per semplici differenze sappiamo che $q_{i,i-1} = \mu$!

Cosnideriamo ora lo stato $(i = 0) \implies N(t) = 0$. Supponiamo che il processo stocastico valga 0 nell'istante presente. In questa situazione ci può essere una variazione di stato solo in avanti! $\phi_0(t) \sim \xi_{R_t}$. Definiamo: $\underline{\phi_0(t)}$ come tempo di soggiorno residuo nello stato $(i = 0)$ al tempo t prima di cambiare stato. Essa coincide con $\underline{\xi_{R_t}}$! Quindi $\phi_0(t) \sim EXP(\lambda)$ e $[-q_{00} = \lambda] = q_{01}$.

Abbiamo ottenuto quindi i tassi di transizione per una M/M/1 con parametro λ, μ, rispettivamente per i tempi di interarrivo e $\underline{\text{tempi di servizio}}$. La CATENA è OMOGENEA (i tassi di transizione non dipendono dal tempo), IRRIDUCIBILE. Il numero di stati è però infinito! Dobbiamo quindi risolvere il sistema: $\pi_i = \pi_0(\frac{\lambda}{\mu})^i$, dove abbiamo:

$$\pi_0 = \frac{1}{1 + \sum_{i=1}^\infty \prod_{k=0}^{i-1}(\frac{\lambda_k}{\mu_k+1})} = \frac{1}{\sum_{i=0}^\infty (\frac{\lambda}{\mu})^i}$$

ove abbiamo inglobato il termine di indice 0 (1) nella sommatoria. Notiamo che ci troviamo dinanzi una serie geometrica al denominatore, la quale converge se la ragione è minore di 1:

$$\iff \frac{\lambda}{\mu} < 1 \implies \sum_{i=0}^\infty (\frac{\lambda}{\mu})^i = \frac{1}{1-\frac{\lambda}{\mu}} = (\frac{\mu}{\mu-\lambda})$$

Distribuzione a regime

Consideriamo il processo stocastico Markoviano $N(t)$ (al tempo t). Calcoliamo la DISTRIBUZIONE DI REGIME PER IL NUMERO DI CLIENTI. Ricordiamo che il sistema è *STABILE* quando $[\lambda < \mu]$, ovvero quando la velocità di arrivo è minore della velocità di servizio, ancora, quando $[\frac{1}{\lambda} > \frac{1}{\mu}]$, ovvero quando il tempo medio di di interarrivo è maggiore del tempo medio di servizio, cosa alquanto non sorprendente.

$$\pi_0 = (1 - \frac{\lambda}{\mu}) \implies \underline{\underline{\pi_i}} = (1 - \frac{\lambda}{\mu})(\frac{\lambda}{\mu})^i = (\frac{\mu-\lambda}{\mu})(\frac{\lambda}{\mu})^i,\ i \geq 0$$

Posto $\rho := \frac{\lambda}{\mu}$, possiamo scrivere: $[\pi_i = (1-\rho)\rho^i]$, ove abbiamo definito ρ come:
$\underline{\text{fattore di utilizzazione del router}}$. Frazione del tempo ($\underline{\text{a regime}}$) nel quale quel router è occupato nel SERVIRE I CLIENTI. Es. $\rho = 0.8 \implies$ significa che a regime il router è occupato per

l'80% nel servire i clienti. In un sistema Link, l'utilizzazione è la BANDA! Utilizzazione della banda del Link.

$\pi_0 = (1 - \rho)$. Tale è la probabilità che vi siano 0 clienti nel sistema a coda a regime, quindi rappresenta anche la frazione del tempo nel quale a regime nel sistema a coda vi siano 0 clienti. $1 - \pi_0$ è la frazione del tempo a regime nel quale il sistema a coda vi sia almeno un cliente! ($\iff$ sistema a coda occupato). ρ è in realtà adimensionato. Esso rappresenta l'INTENSITÀ DI TRAFFICO. Ciononostante si misura in *ERLANG*. Definiamo quindi formalmente ρ:

Definition 15. *Intensità di traffico*

ρ è definito come CARICO MEDIO DI LAVORO IN UNITÀ DI TEMPO DI SERVIZIO che arriva al sistema a coda $\forall$ unità di tempo.

$$\rho = \lambda(\frac{1}{\mu})$$

Esso si misura quindi in ERLANG. Ogni cliente richiederà al servitore un tempo di servizio ($\frac{1}{\mu}$). Ma nell'unità di tempo in media arriveranno λ clienti! ρ quindi rappresenta l'intensità di traffico. Ci sono delle apposite tabelle per il dimensionamento di questi valori (in ERLANG) ovviamente.

Quindi:

$$\frac{\lambda}{\mu} < 1 \implies \pi_i = (1 - \frac{\lambda}{\mu})(\frac{\lambda}{\mu})^i = (1 - \rho)\rho^i = \cdot(\rho), \ [i \geq 0]$$

with $\underline{(1 - \frac{\lambda}{\mu}) = 1 - \rho = \pi_0}$. Tale è la distribuzione a regime, $\forall i \geq 0$. La condizione di ERGODICITÀ è $(\rho < 1)$.

Quantità medie tipiche

Calcoliamo adesso il numero medio di clienti nella coda in condizioni di regime. (Stiamo adoperando una media di insieme):

$$\bar{N} = \lim_{t\to\infty} \mathbb{E}[N(t)] = \sum_{i=0}^{\infty} i\pi_i = \sum_{i=0}^{\infty} i(1 - \rho)\rho^i = (1 - \rho)\rho(\sum_{i=0}^{\infty} i\rho^{i-1} =$$

$$= (\frac{1}{1 - \rho})^2 = \frac{1}{(1 - \rho)^2}) = (1 - \rho)\rho\frac{1}{(1 - \rho)^2} = \frac{\rho}{1 - \rho}$$

Ove l'uguaglianza tra parentesi, rappresentante la convergenza di quella serie, si ottiene semplicemente differenziando membro a membro la serie geometrica con la sua somma.

$$\frac{\rho}{1 - \rho} = \frac{\frac{\lambda}{\mu}}{1 - \frac{\lambda}{\mu}} = (\frac{\lambda}{\mu - \lambda})$$

Tale è il numero medio di clienti nel sistema a coda a regime. Graficamente, se esprimessimo in funzione di ρ questa quantità, avremmo un andamento tale per cui quando arriviamo a $\rho = 1$, asintoticamente, il router non ce la fa più ed i tempi di ritardo salgono vertiginosamente sino a ∞, sempre asintoticamente.

Valutiamo ora il tempo medio di permanenza (a regime) di un cliente nel sistema a coda (detto anche *RITARDO MEDIO*). Quanto, in media, un cliente permane nell'intero sistema. Applichiamo Little:

$$\bar{N} = \lambda\bar{T} \implies \bar{T} = \frac{\bar{N}}{\lambda} = \frac{\lambda}{(\mu - \lambda)\lambda} = [\underline{\frac{1}{(\mu - \lambda)}}]$$

Si può dimostrare che il ritardo medio per cliente è distribuito esponenzialmente con parametro $(\mu - \lambda)$. Esso è una v.a. tale per cui $(\dots) \sim EXP(\mu - \lambda)$, in maniera tale che quindi il ritardo medio per clienti è, come abbiamo già dimostrato, $\frac{1}{(\mu-\lambda)}$. Quindi ricordiamo che λ è la velocità di arrivo, μ è la velocità di servizio, e se graficassimo questo ritardo medio in funzione di ρ, al tendere di $(p \to 1)$, il tempo di attesa medio andrebbe all'infinito. La media di insieme corrisponde alla media temporale, in virtù dell'ERGODICITÀ DELLA CATENA.

$$\begin{cases} \rho \to 0 \implies T \to (\frac{1}{\mu}) \\ \rho \to 1 \implies T \to +\infty \end{cases}$$

$\rho \to 0$ o quando $\lambda \to 0$, oppure quando $\mu \to \infty$. Quando $\rho \to 0$, il tempo di permanenza nel sistema coincide con il solo tempo di servizio $(\frac{1}{\mu})$. Supponiamo adesso di voler calcolare il tempo medio trascorso da un cliente nella sola fila di attesa: devo sostanzialmente sottrarre al tempo medio di permanenza il tempo medio di servizio:

$$\bar{W} = \bar{T} - (\frac{1}{\mu}) = \frac{\rho = (\frac{\lambda}{\mu})}{\mu - \lambda}$$

Calcoliamo adesso il numero medio di clienti NELLA SOLA fila di attesa, sempre con Little:

$$[\bar{Q} = \lambda \bar{W} = \frac{\lambda\rho}{(\mu - \lambda)} = \frac{\rho^2}{(1 - \rho)}]$$

Ricordiamo che $(\frac{1}{\mu})$ è il tempo medio di permanenza nel centro di servizio. Troviamo ora $\bar{N_s}$, ovvero il numero medio di clienti nel solo centro di servizio. Banalmente, ora che abbiamo tutti i dati necessari, potremmo scrivere semplicemente: $[\bar{N_s} = \bar{N} - \bar{Q}]$, ma applicando Little troviamo:

$$\bar{N_s} = \lambda(\frac{1}{\mu}) = \rho$$

Ove λ rappresenta anche la velocità di arrivo nel centro di servizio, ed il successivo fattore $(\frac{1}{\mu})$ rappresenta come già detto il tempo medio di servizio. $\bar{N_s} = \rho$ è quindi pari alla frazione di tempo in cui il router è impegnato. SE CI SONO dei CLIENTI il router è impegnato. $\forall$ sistema a coda in cui abbiamo un solo servitore, vale che $[\bar{N_s} = \rho]$!

Imponiamo n_S il numero di clienti nel centro di servizio. Notiamo che effettuando la media troviamo:

$$\mathbb{E}[n_S] = 0 * \Pr\{n_S = 0\} + 1 * \Pr\{n_S = 1\} = \Pr\{n_S = 1\}$$

ma c'è un cliente quando il router è occupato! Quindi è sempre pari alla frazione di tempo in cui il router è occupato. È possibile generalizzare il fatto per le code M/M, ottenendo ovviamente un differente risultato numerico.

TDM, FDM e TDM statistico

Generalmente si utilizza la TDM statistica e non FDM o TDM normale. Ricordiamo che: $\bar{T} = \frac{1}{\mu-\lambda}$. Questo è il ritardo nell'intero sistema a coda. Il MAC è un protocollo per arbitrare l'accesso multiplo ad un canale fisico. La multiplazione è una tecnica per andare a condividere un canale di trasmissione tra più utenti, suddividendo in base alla banda od alla frequenza. FDM, TDM sono tecniche di divisione STATICA. Allocazione statica dei vari sottocanali ai vari clienti. La TDM prevede una suddivisione in slot temporali, mentre la FDM prevede la trasmissione

contemporanea sovrapposta nel tempo. In ogni caso grossomodo abbiamo $\frac{C}{m}[bit/s]$. Il traffico della rete è di tipo IMPULSIVO. Ci sarebbe spreco di banda in entrambi i casi. Il calcolatore vorrebbe sempre tutta la banda. La TDM prevede trasmissioni sovrapposte in frequenza ma non nel tempo (SEPARATE NEL TEMPO). Si utilizza quindi generalmente la TDM statistica, che prevede delle trasmissioni regolate dalla statistica. Con il TDM statistico succede che la statistica sarà collegata ai vari tempi. Quando quel flusso disponibile sarà trasmesso verrà fatto alla piena capacità del link $C > \frac{C}{m}$). Supponiamo che un router debba multiplare su un link di capacità C $[bit/s]$ degli m flussi indipendenti di POISSON. Per semplicità supponiamo che le velocità di arrivo siano λ, il che significa che arrivano λ pacchetti in media per unità di tempo. m flussi. Supponiamo che le lunghezze dei pacchetti associate ai vari flussi siano v.a. indipendenti i.d. memoryless. $\underline{L\ bit}$. PDF esponenziale con valore medio di L *bit*.

- CASO 1 (***TDM***): Supponiamo che quel multiplatore adotti una FDM od una TDM. La capacità massima è $(\frac{C}{m})$. Supponiamo di non considerare il ritardo di elaborazione. Con una $\{TDM, FDM\}$ le risorse di trasmissione sarebbero allocate agli m flussi (m processi di POISSON indipendenti). Supponiamo il buffer di dimensione illimitata ($\iff d = +\infty$). $\underline{\text{M/M/1}}$. Buffer molto grande in modo tale che non rappresenti un collo di bottiglia. Il tempo medio di trasmissione di un pacchetto è: $\frac{L}{(C/m)}$. Sarebbe il valore medio del tempo di trasmissione. Il parametro della relativa PDF del tempo di servizio sarebbe quindi: $\mu = \frac{C}{mL}$. Per il singolo flusso potrei adottare un sistema a coda M/M/1 con $\{\lambda, \mu\}$ come parametri, ma ne devo considerare m di questi sottosistemi. Quindi abbiamo $\bar{T_1} = \frac{1}{\mu - \lambda}$.

- CASO 2 (***TDM statistico***): Pensiamo all'altra possibilità (TDM statistica). Utilizzo tutta la capacità del link, quindi $\implies \frac{L}{C}$ è il tempo medio di trasmissione di un pacchetto. Il reciproco, $\frac{C}{L}$ è invece la velocità del router. $\frac{L}{C} = \frac{1}{m\mu}$. Abbiamo quindi un SERVITORE a velocità $m\mu$. Si immagini di fare il POOLING, il parametro risultante sarà dato dalla somma di parametri. I clienti arrivano secondo un processo di POISSON (parametro dato dalla somma dei parametri), quindi abbiamo parametri $\{m\lambda, m\mu\}$. Il ritardo medio è quindi:

$$\bar{T_2} = \frac{1}{m\mu - m\lambda} = \frac{1}{m(\mu - \lambda)}$$

  praticamente m volte inferiore a quello utilizzato da una TDM/FDM.

Exercise

The network of a small company has been designed according to a Hub-and-Spoke hierarchical topology (see figure below). The LAN in each branch office is connected to the main office via a router and a link with capacity C equal to 128 $Kbps$ (1 $Kbit = 1000$ $bits$). In one of the sites the outgoing traffic flow has been mainly generated by CAD applications up to now. With regard to that traffic, which can be modeled by a Poisson process with rate $\lambda_0 = 3$ pkt/s, it is required that the queuing delay in the router (for the queued packets) is less than $0.5s$ with probability greater than 0.99. Following a new configuration of the business processes, it is necessary to install a number of computers for office automation. It is expected that each of them generates a flow of packets towards the central office at an average rate $\lambda = 1$ pkt/s. For the latter flow, it is required that the average time spent in the router is less than or equal to $1s$. Packet sizes can be modeled by independent random variables which are exponentially distributed with a mean value $D = 500$ $bytes$.

- 1) Assume that the output queue of the router has an infinite size and is shared by all the traffic flows according to the FCFS policy. Determine the maximum number M of new computers that can be installed;
- 2) Considering the above value M, evaluate the packet loss probability in case the output queue size is limited to 10 packets.

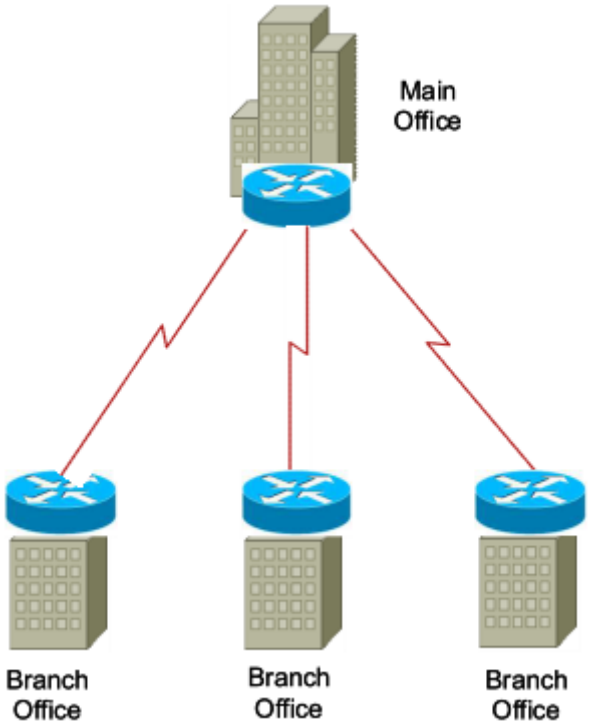


Figura 2.1: Connection to the Main Office

"È stata progettata una rete di una piccola compagnia con una topologia gerarchica *Hub-And-Spoke*. La LAN in ogni ufficio secondario è connessa all'ufficio principale tramite un router ed un link con capacità $C = 128\ Kbps$. Dove ($1\ Kbit = 1000\ bits$). In uno dei siti il flusso del traffico in uscita è generato principalmente da applicazioni CAD fino ad adesso. Per quel traffico, che può essere modellato come un processo di Poisson con rate $\lambda_0 = 3\ pkt/s$, è richiesto che il queueing delay nel router (per i pacchetti in coda) sia meno di 0.5 secondi con probabilità più grande di 0.99. A seguito di una nuova configurazione dei processi di business, è anche necessario installare un numero di computer per l'automazione degi uffici. Ci si aspetta che ognuno di loro generi un flusso di pacchetti verso l'ufficio centrale ad un average rate di $\lambda = 1\ pkt/s$. Per quest'ultimo flusso, è richiesto che l'average time speso nel router sia minore od uguale ad 1s.

Le dimensioni dei pacchetti possono essere modellate da variabili casuali indipendenti che sono esponenzialmente distribuite con valore medio $D = 500bytes$.

- Si assuma che la coda di output del router abbia capacità infinita e sia condivisa da tutti i flussi di traffico secondo disciplina di coda FCFS. Si determini il massimo numero M di nuovi computer che possono essere installati;
- Considerando il valore M, si valuti la *packet loss probability* nel caso la dimensione della coda di output sia limitata a 10 pacchetti.

Abbiamo una topologia Hub-And-Spoke. (centro-e-raggi, topologia stellare). $C = 128\ Kbps$. Dove $(1\ Kbit = 1000\ bits) \implies C = 128000\ bit/s$. $\lambda_o = 3\ pkt/s \ \wedge \ \lambda = 1\ pkt/s$. Si richiede che il ritardo di accodamento nel router sia minore di $0.5s$ con probabilità maggiore di 0.99. Le lunghezze dei pacchetti sono distribuite esponenzialmente di media 500 bytes. Politica FCFS. Quanto è il numero massimo di computer installabili M?

Supponiamo di indicare con W la v.a. che rappresenta il ritardo di accodamento; il vincolo è il seguente, con $(\tau = 0.5s)$, $\underline{s = threshold} = 0.99$:

$$\Pr\{W < (\tau = 0.5s) \mid si\ faccia\ coda\} > (0.99 = s)$$

Indichiamo con $\bar{R}$ il ritardo di accodamento per l'ultimo flusso. Deve valere: $\bar{R} \leq 1s$. Cerchiamo di risolvere il sistema con M/M/1. Abbiamo già delle ipotesi, ma dovremo farne di aggiuntive. Modelliamo quindi il router con un sistema a coda M/M/1:

Abbiamo un buffer output, ed un trasmettitore collegato in serie che è associato al link di uscita. Ipotesi: flusso secondo il quale arrivano i pacchetti di POISSON (pacchetti applicazioni CAD). Stiamo trascurando il ritardo di elaborazione router. Capacità di elaborazione talmente elevata da ritenersi trascurabile (NO BOTTLENECK). Inoltre: $(d = +\infty)$. Il router opera con FCFS (quella contemplata dal sistema a coda M/M/1). Ipotesi aggiuntiva: certo numero di flussi, M riguardanti i computer dell'Office Automation. Processo degli arrivi di POISSON. Ipotesi di indipendenza abbastanza scontata, dal momento che i vari utenti non si influenzeranno a vicenda. $M\lambda$. Ipotesi aggiuntiva: supponiamo che gli arrivi dei pacchetti relativi ai computer Office Automation siano indipendenti da quelli delle applicazioni CAD. Effettuando il POOLING, abbiamo che la distribuzione risultante è sempre un processo di POISSON con parametro dato dalla somma dei parametri: $\lambda' = \lambda_0 + M\lambda$. Quindi λ' è la velocità di arrivo dei clienti nel sistema a coda, μ è la velocità di servizio del router, ovvero il numero medio di pacchetti elaborati dal router per unità di tempo quando il router è costantemente occupato. Abbiamo dei tempi di servizio indipendenti, identicamente distribuiti ed indipendenti dai tempi di interarrivo. Nel sistema i tempi di servizio corrispondono ai tempi di trasmissione. Questo tempo di servizio è legato al parametro (valore medio) D della distribuzione esponenziale che caratterizza la lunghezza dei pacchetti. Quindi abbiamo che il tempo medio di trasmissione è $\frac{1}{\mu} = \frac{D}{C}[s]$, ed abbiamo quindi $\mu = \frac{C}{D} = \frac{1}{D/C} = 32\ [pkt/s]$, ovvero abbiamo 32 pacchetti in media quando il router è costantemente occupato.

- **Parte 1**

  Tutte le ipotesi del sistema a coda M/M/1 sono rispettate. Possiamo quindi utilizzarne i risultati: $\{\lambda', \mu\}$. Dobbiamo andare ad imporre quelle due condizioni. Per un M/M/1, il ritardo per cliente è $\bar{R} = \frac{1}{(\mu - \lambda)}$. Il ritardo per cliente è una v.a. distribuita esponenzialmente $\iff [R \sim EXP(\mu - \lambda)]$. Guardando adesso all'altra condizione, si dimostra che:

  $$[F_W(y) = \Pr\{W \leq y\} = 1 - \rho\, e^{-\mu(1-\rho)y},\ y \geq 0]$$

  Tale è la CDF del Ritardo del cliente nella CODA! (Nella sola coda). W sarebbe il tempo di permanenza del cliente nella SOLA coda del sistema a coda M/M/1. Distribuzione valutata su tutti i clienti (anche per quelli che non ci interessano). Dobbiamo condizionarla al fatto che si faccia coda. Quando $(y = 0)$, esce $\Pr\{W \leq 0\} = (1 - \rho)$. Ovvero la probabilità che il tempo di permanenza in fila di attesa sia nullo è pari a $(1 - \rho)$. $\Pr\{W = 0\} = 1 - \rho$. Si tratta di una v.a. mista. ρ è il fattore di utilizzazione del router. Nel sistema a coda M/M/1, $\rho = (\frac{\lambda'}{\mu})$. Nell'M/M/1, $(1 - \rho) = \pi_0^{(a)}$, ovvero la frazione di

tempo a regime nel quale nel sistema a coda non vi sono clienti. La probabilità che il tempo di permanenza sia nullo è PARI alla probabilità che all'arrivo la fila di attesa SIA VUOTA!

$$[\Pr\{W = 0\} = 1 - \rho = \pi_0]$$

Questa è pari alla probabilità che un cliente AL SUO ARRIVO vada subito al router. In generale: $\pi_i^{(a)} \neq \pi_i$, laddove il primo membro della disuguaglianza riflette la visione dei clienti all'arrivo, valutata solo sugli istanti di arrivo, mentre il secondo membro riflette la visione all'esterno del sistema, considerando tutto l'asse temporale. π_i, ponendo l'ERGODICITÀ, rappresenta la frazione di tempo nel quale vi sono i clienti a regime. Mentre $\pi_i^{(a)}$ è la probabilità calcolata valutando SOLTANTO GLI ISTANTI DI ARRIVO! Frazione degli ARRIVI che trovano il sistema (a regime) nello stato i. Infatti potremmo avere ad esempio: $\{\{\pi_1 = \frac{1}{3},\ \pi_0 = \frac{2}{3}\} \ \wedge\ \{\pi_0^{(0)} = 1,\ \pi_1^{(0)} = 0,\ \pi_2^{(0)} = 0\}\}$.

Theorem 11. ***P.A.S.T.A. POISSON ARRIVALS SEE TIME AVERAGES***

$$\begin{cases} \pi_i = \lim\limits_{t\to\infty} \Pr\{N(t) = i\} \\ \pi_i^{(a)} = \lim\limits_{t\to\infty} \Pr\{N(t) = i \mid \textit{un arrivo subito dopo } t\} \end{cases}$$

Se gli arrivi dei clienti si susseguono con processo di POISSON, abbiamo che: $\pi_i^{(a)} = \pi_i,\ \bar{N} = \sum_{i=0}^{\infty} i\pi_i$

Dove la prima equazione è una quantità calcolata sull'insieme delle realizzazioni. (Media di insiemi). Ma se vale L'ERGODICITÀ, allora le medie d'insieme coincidono con le medie TEMPORALI. π_i rappresenta la frazione del tempo nella quale vi sono a regime i clienti nel sistema; $\pi_i^{(a)}$ rappresenta la frazione dei clienti che, all'arrivo trovano il sistema nello stato i. Coincide con la probabilità (a regime), che all'arrivo un cliente trovi i clienti nel sistema. Rappresenta cosa vedono i clienti all'arrivo.

Per i nostri scopi, $[\pi_0^{(a)} = \Pr\{W = 0\}] = [\underline{1 - \rho = \pi_0}]$, ove la parte sottolineata è la probabilità che vi siano 0 clienti, in un istante qualsiasi, mentre il primo membro è la probabilità che un cliente, al suo arrivo, trovi 0 clienti nel sistema. Grazie al teorema appena visto, esse coincidono.

ρ è la frazione del tempo in cui il router è occupato $\implies$ probabilità che il router sia occupato. $(1-\rho)$ è la probabilità che il router sia quindi libero. $\Pr\{W \leq \tau \mid si\ fa\ coda\} = (\dots)$ è la probabilità che il tempo di permanenza in FILA DI ATTESA sia inferiore (od uguale) a τ, quando si fa coda. Quindi abbiamo:

$$(\dots) = 1 - \Pr\{W > \tau \mid si\ fa\ coda\} = 1 - \frac{\Pr\{W > (y := \tau),\ si\ fa\ coda\}}{\Pr\{si\ faccia\ coda\}} = (\dots)$$

Notiamo che $\{W > y\} \supseteq \{si\ fa\ coda\}$, ed inoltre, dal teorema appena visto (PASTA) vale che: $1 - \pi_0^{(a)} = 1 - \pi_0 = \rho = \Pr\{si\ fa\ coda\}$, ovvero la probabilità che all'arrivo di un cliente il servitore sia occupato coincide con la probabilità di fare coda. Ciò implica: $\implies$

$$(\dots) = 1 - \frac{\Pr\{W > \tau \ \ si \ fa \ coda\}}{\Pr\{si \ fa \ coda\}} = 1 - \frac{\Pr\{W \geq \tau\}}{\rho} =$$

$$= 1 - \frac{\rho \, \mathrm{e}^{-\mu(1-\frac{\lambda}{\mu})y}}{\rho} = 1 - \mathrm{e}^{-\mu(1-\frac{\lambda}{\mu})y}, \ y \geq 0$$

Concludiamo che:

$$\Pr\{W \leq y \mid si \ fa \ coda\} = 1 - \mathrm{e}^{-\mu(1-\frac{\lambda}{\mu})y} \iff W|_{\{si \ fa \ coda\}} \sim EXP(\mu - \lambda)$$

quindi nell'M/M/1, $R = W|_{\underline{\{si \ fa \ coda\}}}$. Si riferisce proprio alla medesima distribuzione esponenziale, con il medesimo parametro!

Consci che $[R = \frac{1}{\mu - \lambda'}]$, allora procediamo:

$$1 - \mathrm{e}^{-(\mu-\lambda')\tau} > s \implies \mathrm{e}^{-(\mu-\lambda')\tau} < (1-s) \implies -(\mu-\lambda')\tau < \log(1-s) \implies (\dots)$$

$$(\dots) \implies -\mu\tau + \lambda'\tau < \log(1-s) \implies [\lambda'\tau < \log(1-s) + \mu\tau] \implies (\dots)$$

$$(\dots) \implies \lambda_0 + M\lambda < \frac{\log(1-s)}{\tau} + \mu \implies M\lambda < \mu + \frac{\log(1-s)}{\tau} - \lambda_0 \implies (\dots)$$

$$(\dots) \implies M < \frac{1}{\lambda}[\mu + \frac{1}{\tau}\log(1-s) - \lambda_0] = [32.8]$$

ove il valore si è ottenuto appositamente sostituendovi i dati al secondo membro dell'ultima disequazione.

Quindi $\lfloor M \rfloor = M_{max} = 32$ computer massimi. Si ottiene $M \leq 32.8$, quindi possiamo installare MASSIMO 32 COMPUTER. D'altro canto dobbiamo rispettare anche il vincolo sull'average router time:

$$\bar{R} = \frac{1}{\mu - \lambda'} \implies \frac{1}{\mu - \lambda'} \leq 1s \implies ((\mu - \lambda') > 0) \geq 1s \implies (\dots)$$

ove l'ultima parentesizzazione maggiore di 0 indica la condizione di STABILITÀ, che naturalmente deve essere sempre rispettata.

$$(\dots) \implies \mu - (\lambda_0 + M\lambda) \geq 1 \implies \mu - \lambda_0 - M\lambda \geq 1 \implies M \leq \frac{\mu - \lambda_0 - 1}{\lambda} =$$

$$= \frac{1}{\lambda}[\mu - \lambda_0 - 1] \implies M \leq 28$$

Ne si conclude che $M_{max} = \min\{28, 32\} = 28$ computer. Quindi: $\underline{\underline{[M \leq M_{max} = 28]}}$.

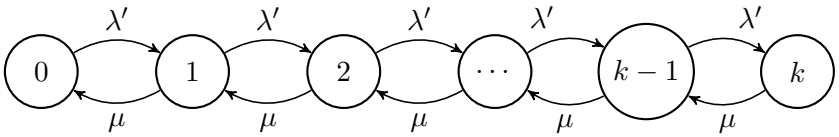
- **Parte 2**:

M/M/1/k

Consideriamo che $d = 10$ *pkt* (10 pacchetti), $d < +\infty$ ed abbiamo quindi un buffer di dimensione finita. $\iff$ $(d = 10) < +\infty$. In questo caso NON possiamo studiare il sistema con un modello M/M/1! Ci serve invece un *M/M/1/k*, nel quale la dimensione del buffer è limitata. Pensiamo a k come la capacità di tutto il sistema (includendo il centro di servizio). Stiamo quindi considerando un sistema a CODA CON PERDITA. Quando un elemento arrivando trova la coda piena, il sistema interviene con il meccanismo di DISCARD (avviene il fenomeno *packet loss*). In questo caso λ' è la velocità di arrivo dei clienti al sistema (POISSON). λ_s sarà sempre la velocità di ingresso alla coda e di uscita (numero medio di clienti). $\rho = \frac{\lambda_s}{\mu}$ è il fattore di utilizzazione del servitore. Abbiamo che:

$$\Pr\{perdita\} = \frac{\lambda^l}{\lambda'} = \pi_k^{(a)} = \pi_k$$

grazie a PASTA. Disegnamo il relativo DTT della catena:



Sia $N(t)$ il numero di clienti nel sistema a tempo t. $\Pr\{\phi_i(t) > \tau\} = (\dots)$, dove $\phi_i(t)$ indica il tempo di soggiorno residuo nello stato i al tempo t, indica la probabilità che per almeno τ unità di tempo non ci sia alcun arrivo e NON vi sia la fine del servizio in corso, condizionata con il fatto che attualmente vi sono i clienti nel sistema.

Indichiamo con $\xi_{Rt} \sim EXP(\lambda')$ la v.a. tempo di interarrivo residuo al tempo t, e con $\eta_{Rt} \sim EXP(\mu)$ la v.a. tempo di servizio residuo al tempo t. Tutte indipendenti ed indipendenti tra di loro, grazie all'ipotesi di (inter)-indipendenza.

$$(\dots) = \Pr\{\xi_{Rt} > \tau,\ \eta_{Rt} > \tau\} = \Pr\{\xi_{Rt} > \tau\}\Pr\{\eta_{Rt} > \tau\} = \mathrm{e}^{-\lambda'\tau}\,\mathrm{e}^{-\mu\tau} = \mathrm{e}^{-(\lambda'+\mu)\tau}$$

Quindi abbiamo di nuovo: $-q_{ii} = \mu + \lambda' \implies$

$$\tau_{i,i+1} = \frac{q_{i,i+1}}{-q_{ii}} = \frac{q_{i,i+1}}{(\mu + \lambda')}$$

$\tau_{i,i+1}$ è la probabilità che lasciando lo stato i, il processo vada verso $i + 1$. Sfruttando il teorema delle probabilità totali nel continuo, abbiamo: $q_{i,i+1} = (\mu + \lambda')\frac{\lambda'}{(\mu+\lambda')} = \lambda'$. Quindi sarà λ' fino a $k - 1$. La CMTC è una catena omogenea, IRRIDUCIBILE e dal momento che $|stati| < +\infty \implies$ allora essa è SICURAMENTE ERGODICA! CATENA SEMPRE ERGODICA. Sistema STABILE.

$$\begin{cases} [\pi_i = \pi_0(\frac{\lambda'}{\mu})^i] \\ [\pi_0 = \dfrac{1}{1 + \sum_{i=1}^{k}(\frac{\lambda'}{\mu})^i}] \end{cases}$$

$i = 1, 2, \ \ldots, \ k$.

Notiamo che:

$$\sum_{i=0}^{k} \rho^i = \begin{cases} k+1, \ \rho = 1 \\ \dfrac{1-\rho^{k+1}}{(1-\rho)}, \ p \neq 1 \end{cases}$$

Quindi nel nostro caso abbiamo:

$$\pi_0 = \frac{1}{\sum_{i=0}^{k} \rho^i} = \frac{1}{\frac{1-\rho^{k+1}}{(1-\rho)}} = \frac{1-\rho}{1-\rho^{k+1}}$$

Quindi abbiamo come Distribuzione di regime:

$$\pi_i = \frac{1-\rho}{1-\rho^{k+1}} \rho^i, \ i = 0, 1, 2, \ \ldots, \ k$$

Si noti che vale anche per $i = 0$. Allora:

$$[\Pr\{perdita\} = \underline{\pi_k^{(a)} \overset{PASTA}{=} \pi_k} = [\frac{1-\rho}{1-\rho^{k+1}} \rho^k]]$$

da valutarsi con $k = 10 + 1 = d + 1 < +\infty$, in quanto include anche il centro di servizio. Facendo i calcoli esce $0.07 \implies (70\%)$.

La probabilità di perdita si poteva trovare, come precedentemente enunciato, anche effettuando il rapporto $\frac{\lambda_L}{\lambda'}$. Banalmente, per SPLITTING si ha: $[\lambda_L = \lambda' - \lambda_S]$. Le perdite comunque mi fanno venire meno la caratteristica di POISSON.

$$\Pr\{k \ arrivi \ in \ \tau\} = \frac{(\lambda\tau)^k}{k!} \, \mathrm{e}^{-\lambda\tau}$$

ma la sequenza degli ingressi a valle della diramazione NON è più di POISSON! $\rho = (\frac{\lambda_S}{\mu})$ rappresenta il fattore di utilizzazione, quindi la frazione del tempo a regime in cui il servitore è occupato. $\rho = 1 - \pi_0 \implies \pi_0 = 1 - \rho$.

$\lambda_S = \mu(1 - \pi_0)$ rappresenta la velocità delle partenze (*departure*) dei clienti dal mio sistema. Abbiamo: $[\pi_0 = \frac{1-\rho}{1-\rho^{k+1}}]$.

$$\lambda_L = \lambda' - \lambda_S = \lambda' - \mu(1 - \pi_0)$$

Quindi λ_S è la velocità secondo la quale i clienti partono dal sistema, od anche la media dei clienti che partono all'unità di tempo. Guardando il DTT, vi è una partenza quando FINISCE un servizio! Determinando le frequenze delle transizioni a sinistra, potremo sapere le frequenza delle partenze. La frequenza delle partenze si ottiene sommando tutte le frequenze delle transizioni $\pi_i q_{ij}$, sfruttando opportunamente l'inversa della CONDIZIONE DI NORMALIZZAZIONE:

$$\pi_1\mu + \pi_2\mu + \ (\ldots) \ + \pi_k\mu = \mu \sum \pi_i = [\lambda_S = \mu(1 - \pi_0)]$$

Stessa cosa mettendoci a valle della diramazione ed a monte della coda: (ARRIVI DI CLIENTI dall'esterno del sistema). λ_S rappresenta qui il numero medio di clienti che entrano nel sistema per unità di tempo.

$$\lambda_S = [\pi_0\lambda' + \pi_1\lambda' + \ \dots\ + \pi_{k-1}\lambda'] = \lambda'$$

Valgono le equazioni di bilanciamento locale, valide solo per una CMTC di nascita e morte (BDCMTC):

$$\pi_1\mu = \pi_0\lambda' \implies \pi_{k+1}\mu = \pi_k\lambda'$$

ovvero le sommatorie sono identiche.

$\frac{\lambda_L}{\lambda'}$ rappresenta la probabilità di perdita.

Theorem 12. ***Probabilità di perdita***

$$\Pr\{perdita\} = \frac{1-\rho}{1-\rho^{k+1}}\rho^k = \pi_k^{(a)} = \pi_k$$

ove π_k rappresenta la probabilità che il sistema si trovi nello stato k in un istante qualsiasi. Abbiamo anche:

$$\Pr\{perdita\} = \frac{(\lambda_L = \lambda' - \lambda_S)}{\lambda'} = 1 - \frac{\lambda_S}{\lambda'} = (\dots)$$

Ove λ_S è stato trovato applicando Little al primo sottosistema, ottenendo quindi $\lambda_S = \mu(1-\pi_0)$.

Dimostrazione.

$$(\dots) = 1 - \frac{\mu(1-\pi_0)}{\lambda'} = 1 - \frac{1}{\rho}(1 - \frac{1-\rho}{1-\rho^{k+1}}) = 1 - \frac{1}{\rho}(\frac{1-\rho^{k+1}-1+\rho}{1-\rho^{k+1}})$$

$$1 - \frac{1}{\rho}\rho\frac{(1-\rho^k)}{(1-\rho^{k+1})} = \frac{1-\rho^{k+1}-1+\rho^k}{1-\rho^{k+1}} = \frac{\rho^k(1-\rho)}{(1-\rho^{k+1})};$$

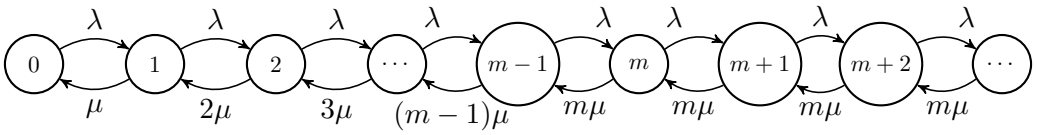
□

La probabilità di perdita di un cliente deve quindi alla fine coincidere con il rapporto $\frac{\lambda_L}{\lambda'}$.

2.2.2 Sistemi a coda M/M/m

Il sistema a coda M/M/m rappresenta una generalizzazione del sistema M/M/1. Abbiamo m router. Anche in questo caso, il processo degli arrivi è un processo di POISSON con velocità λ. (λ = velocità di arrivo dei clienti). In media arrivano λ clienti per unità di tempo. Anche in questo caso i tempi di servizio sono v.a. indipendenti (mutuamente), i.d. con una distribuzione esponenziale negativa unilatera, con tempo medio di servizio ($\frac{1}{\mu}$), ed indipendenti statisticamente dai tempi di interarrivo dei clienti nel sistema a coda. La velocità del centro di servizio sarà variabile (il centro di servizio include ora m servitori), ovvero dipende dal numero di router attivi in un certo istante di tempo. Se i è il numero di clienti nel sistema a coda in un certo istante di tempo, $i < m \implies i\mu$ velocità del centro di servizio. i sono i clienti

IN TUTTO IL SISTEMA! In totale quindi i servitori attivi serviranno $i\mu$ utenti $\forall$ unità di tempo. $i \geq m$ tutti gli m servitori saranno attivi $\implies$ velocità $m\mu$. Quindi $i \geq m$ centro di servizio alla massima velocità possibile. $m\mu$ = capacità TOTALE DI SERVIZIO DEL SISTEMA (del centro di servizio). Come nella M/M/1, la dimensione del buffer e la popolazione sono illimitate ($\iff d, e < +\infty$). Disciplina di queueing di default (FCFS). Modellabile come uno stack FIFO. Se arriva un cliente e c'è un certo numero di router liberi, sarà assegnato in maniera random ad uno di questo. Approccio MARKOVIANO basato sul processo stocastico che si riferisce al numero di clienti nell'intero sistema a coda al tempo t, ovvero $N(t)$. La sua evoluzione sarà determinata dalle v.a. tempi di interarrivo e tempi di servizio. Dato che per queste ipotesi queste v.a. sono prive di memoria e statisticamente indipendenti $\implies$ siamo dinanzi una CMTC, il cui diagramma dei tassi di transizione è il seguente:



CMTC nascita e morte (BDCMTC), con $|stati| = +\infty$ (infiniti), fila di attesa di dimensione illimitata. Tassi di nascita pari a λ, tassi di morte $m\mu$, che sarebbe il tasso di morte nello stato i prima che si saturino i servitori disponibili, dopodiché ($i \geq m$), sempre pari a $m\mu$.

Calcolo dei tassi (velocità) di transizione

Supponiamo $N(t) = i \neq 0$ (i clienti nel mio intero sistema a coda), ovvero qualsiasi stato diverso dallo stato banale. Il successivo cambiamento di stato sarà legato od all'arrivo di un nuovo cliente od al completamento di uno dei servizi eseguiti in parallelo al tempo t. Comunque abbiamo i clienti totali. Possibili casi: $\{i++,\ i--\}$. L'intervallo di tempo da t al successivo arrivo lo indichiamo con $\underline{\xi_R(t) \sim EXP(\lambda)}$, ovvero il tempo di interarrivo residuo (stessa caratteristiche dei tempi di interarrivo). Il tempo che passa tra l'istante presente t ed il successivo completamento del servizio, sarà $\eta_R(t)$, e sarà pari al minimo dei tempi di servizio residui in t. Ci saranno un certo numero di servizi. $\eta_R(t) = \min\{...\}$, ovvero al minimo dei tempi di servizio residui legati a quei servizi che al tempo t stanno procedendo in parallelo. Quando $i < m$, ci saranno i servizi che stanno procedendo in parallelo. Anche i tempi di servizio residui sono $\underline{\text{indipendenti}}$ fra di loro per ipotesi, dal momento che i servizi procedono in parallelo senza influirsi. Quindi $\eta_R(t) \sim EXP(\sum_i \mu_i)$. Quando $i < m \implies \eta_R(t) \sim EXP(i\mu)$. Se invece $i \geq m$, $\eta_R(t) \sim EXP(m\mu)$. A questo punto, il tempo che passa dall'istante presente t e la successiva transizione di stato, sarà $\phi_i(t)$ (tempo di soggiorno residuo nello stato i al tempo t) prima di cambiare stato $\implies$

$$\phi_i(t) = \min\{\xi_R(t),\ \eta_R(t)\} \implies \phi_i(t) \sim EXP(-q_{ii})$$

ove l'ultima implicazione è valida grazie al fatto che queste variabili casuali sono i.i.d. ed indipendenti tra di loro. Abbiamo:

$$-q_{ii} = \begin{cases} (\lambda + i\mu),\ i < m \\ (\lambda + m\mu),\ i \geq m \end{cases}$$

(Velocità totale di uscita $\underline{\text{dallo stato i}}$). Parametro $-q_{ii}$. Quindi, quella quantità rappresenta con che velocità in totale esco dallo stato. Ma se $\underline{i < m}$, perché $q_{i,i+1} = \lambda,\ q_{i,i-1} = i\mu$. Solito giochetto. (Se $i \geq m,\ q_{i,i-1} = m\mu$). Consideriamo:

$$\tau_{i,i+1} = \frac{q_{i,i+1}}{-q_{ii}} = \Pr\{\xi_R(t) < \eta_R(t)\} = (\dots)$$

ove l'ultima quantità rappresenta la probabilità che il prossimo arrivo preceda la fine del servizio (il completamento del servizio). Dobbiamo quindi applicare il teorema delle probabilità totali nel continuo, ottenendo alla fine:

$$(\dots) = \begin{cases} \dfrac{\lambda}{\lambda + i\mu}, \ i < m \\ \dfrac{\lambda}{\lambda + m\mu}, \ i \geq m \end{cases}$$

CMTC OMOGENEA, IRRIDUCIBILE, ma non sappiamo tuttavia se è ERGODICA.

$$\begin{cases} [\pi_0 = \dfrac{1}{1 + \sum_{i=1}^{m-1} (\frac{\lambda}{\mu})^i \frac{1}{i!} + (\frac{m^m}{m!}) \sum_{i=m}^{\infty} (\frac{\lambda}{m\mu})^i}] \\ \begin{cases} \pi_i = \pi_0 (\dfrac{\lambda}{\mu})^i \dfrac{1}{i!}, \ i < m \\ \pi_i = \pi_0 (\dfrac{\lambda}{\mu})^i \dfrac{1}{m! m^{i-m}}, \ i \geq m \end{cases} \end{cases}$$

Dobbiamo capire qual'è la CONDIZIONE MAX di ERGODICITÀ (si ha una condizione di regime). La sommatoria ad infiniti termini (serie) a denominatore di π_0 deve convergere di nuovo. La condizione di ERGODICITÀ è quindi: $(\frac{\lambda}{m\mu}) < 1$ (convergenza della serie geometrica). Quando $\rho := \frac{\lambda}{m\mu} < 1$ questa sommatoria (serie) converge, e si ha una distribuzione che è quella di regime. $[\lambda < m\mu]$ è proprio la condizione di STABILITÀ, la quale indica che la velocità di arrivo deve essere minore della capacità totale di servizio. Serve sostanzialmente per evitare l'"ingolfamento". ρ è il fattore di utilizzazione del generico router (Non del centro di servizio). Ognuno dei router avrà un fattore di utilizzazione pari a ρ, quando il carico è EQUAMENTE DISTRIBUITO. $\frac{\lambda}{m}$ sarà la capacità di arrivo del singolo cliente. Servono in media lo stesso numero di clienti $\forall$ unità di tempo. Tutti lavoreranno alla stessa maniera. Ogni router si vedrà arrivare quindi $\frac{\lambda}{m}$ clienti per unità di tempo. Fattore di utilizzazione; frazione di tempo quando il servitore è occupato. Infatti, applicando Little troviamo: $(\frac{\lambda}{m})(\frac{1}{\mu})$, trovando la frazione di tempo nel quale il router è occupato per unità di tempo nella quale il router è costantemente occupato, quando il carico di lavoro è EQUAMENTE DISTRIBUITO. $\frac{\lambda}{\mu}$ è l'*intensità di traffico* (arrivano λ clienti, $\frac{1}{\mu}$ è il lavoro medio richiesto dal cliente). Quindi sfruttiamo ancora Little e troviamo: $\lambda(\frac{1}{\mu})$, trovando così il CARICO MEDIO DI LAVORO.

$$[\frac{\lambda}{\mu} = m(\rho := \frac{\lambda}{m\mu}) = m\rho]$$

Si usi questo fatto per vedere a cosa converge la sommatoria geometrica:

$$[\pi_0 = \frac{1}{\sum_{i=0}^{m-1} (m\rho)^i \frac{1}{i!} + (\frac{m^m}{m!} \frac{\rho^m}{1-\rho} = \frac{(m\rho)^m}{m!(1-\rho)})}]$$

e tramite questa espressione possiamo ovviamente trovare le π_i.

ERGODICITÀ se: $[\lambda < m\mu]$. Abbiamo trovato la distribuzione di regime. Nel caso M/M/1, abbiamo poi ricavato tutti i numeri medi di clienti e tempi di servizio, etc. Nel caso M/M/1 siamo partiti da $\bar{N}$.

Quantità medie tipiche

Adesso partiamo da $\bar{T}$, ovvero il ritardo medio per cliente a regime. Terminologia tempo di risposta. $\bar{T}$ tempo di risposta. Tempo di permanenza del cliente nel sistema a coda. Abbiamo:

$$(\bar{T} := \mathbb{E}[T]) = \mathbb{E}[t_s] + \mathbb{E}[W] = \frac{1}{\mu} + (\dots)$$

ove W è il tempo di accodamento, ovvero il queueing delay. Adesso per valutare il tempo medio di permanenza in fila di attesa utilizziamo il concetto di *MEDIA CONDIZIONATA*:

$$(\dots) = \mathbb{E}[\mathbb{E}[W \mid k \; clienti \; all'arrivo]] = \cdot(k)$$

dove $\{k \; clienti \; all'arrivo\}$ sarebbe l'evento nel quale troviamo k clienti all'arrivo in testa al sistema. Tempo di permanenza, sapendo che all'arrivo il cliente trova k clienti nel sistema. Quindi:

$$\bar{T} = \frac{1}{\mu} + \sum_{k=m}^{\infty} \mathbb{E}[W \mid k \; clienti \; all'arrivo]\pi_k^{(a)}$$

Notiamo che k parte da m, perché se $k < m$ quella quantità sarebbe nulla! Il tempo medio di permanenza nella fila di attesa è nullo quindi se $k < m$! Se arrivato nel sistema a coda trovo un numero $(k < m)$ di clienti, allora VADO DIRETTAMENTE nel CENTRO DI SERVIZIO! Sommatoria che parte da m quindi. Volendo la si potrebbe fare partire da 0, senza alcuna perdità di generalità, dal momento che i primi $(m-1)$ termini sarebbero comunque nulli. Abbiamo:

$$\bar{T} = \frac{1}{\mu} + \sum_{k=m}^{\infty} \frac{k-m+1}{m\mu}(\pi_k^{(a)} = \pi_k) = (\dots)$$

Dove l'uguaglianza tra parentesi deriva da PASTA. Abbiamo sfruttato il fatto che: $[\mathbb{E}[W|k] = \frac{k-m+1}{m\mu}]$. Siamo nel caso $k \geq m$, ove vi sono m clienti al servizio, $(k-m)$ clienti in fila di attesa ed io. Dopo che vi sarà il completamento di $(k-m)$ servizi mi troverò in testa alla fila di attesa. Ma dovrò quindi attendere il completamento di un ulteriore servizio (ecco perché il termine $+1$). Ma il tempo tra il completamento del servizio corrente ed il successivo è $(\frac{1}{m\mu})$! ($\leftarrow$ tempo medio di servizio). $\implies$

$$(\dots) = \frac{1}{\mu} + \sum_{k=m}^{\infty} (\frac{k-m+1}{m\mu})\pi_0 \frac{m^m}{m!}\rho^k = \frac{1}{\mu} + \pi_0 \frac{m^m}{m!} \sum_{k=m}^{\infty} \frac{k-m+1}{m\mu}\rho^k = (\dots)$$

A questo punto possiamo porre: $i := k - m + 1$, ed otteniamo:

$$(\dots) = \frac{1}{\mu} + \pi_0 \frac{m^m}{m!m\mu} \sum_{i=1}^{\infty} i(\rho^{i-1+m} = \rho^{i-1}\rho^m) =$$

$$= \frac{1}{\mu} + \frac{(m\rho)^m}{m!m\mu}\pi_0 \sum_{i=1}^{\infty} i\rho^{i-1} = [\frac{1}{\mu} + \frac{\pi_0(m\rho)^m}{m!m\mu}\frac{1}{(1-\rho)^2}]$$

Abbiamo quindi trovato il RITARDO MEDIO PER CLIENTE. Volendo potremmo anche trovare con Little: $\bar{N} = \lambda\bar{T} = (\dots)$. Spesso in letteratura si esprime $\bar{T}$ in funzione della probabilità di attesa:

$$\Pr\{accodamento \; (queueing)\} = \sum_{k=m}^{\infty} (\pi_k^{(a)} = \pi_k)$$

sarebbe ovvero la sommatoria di queste probabilità. Si deriva quindi la *C di ERLANG*. (La B di ERLANG è invece collegata alla probabilità di perdita (M/M/m/0)).

Siamo in un sistema a coda M/M/m. Ritardo medio per cliente:

$$\begin{cases} \bar{T} = \dfrac{1}{\mu} + \dfrac{\pi_0(m\rho)^m}{m!m\mu}\dfrac{1}{(1-\rho)^2} \\ \rho = \dfrac{\lambda}{m\mu},\ \rho < 1 \end{cases}$$

ove l'ultima equazione del sistema indica la condizione di STABILITÀ. Questa quantità ($\bar{T}$), viene solitamente espressa in letteratura in funzione della probabilità di ATTESA, nella quale dobbiamo considerare la situazione in cui abbiamo almeno m clienti nel sistema a coda quando arriva un cliente:

$$\Pr\{queueing\} = \sum_{k=m}^{\infty}(\pi_k^{(a)} = \pi_k) = \sum_{k=m}^{\infty}\pi_k =$$

$$\sum_{k=m}^{\infty}\pi_0(\frac{\lambda}{m\mu} = \rho)^k\frac{m^m}{m!} = \pi_0\frac{m^m}{m!}\sum_{k=m}^{\infty}\rho^k = (\dots)$$

Siamo dinanzi una sommatoria geometrica, quindi:

$$(\dots) = \pi_0\frac{m^m}{m!}(\frac{\rho^m}{1-\rho}) = \frac{\pi_0(m\rho)^m}{m!(1-\rho)} := C(m, \frac{\lambda}{\mu})$$

Abbiamo ottenuto quest'espressione $C(m, (\frac{\lambda}{\mu}))$, nota come funzione *C di ERLANG* con parametro: $\{m, (\frac{\lambda}{\mu})\}$. Ci sono delle tabelle per questa formula. Abbiamo ricavato un modello per la M/M/m. Possono essere utilizzate per modellare quei sistemi di servizio che prevedono diverse risorse. Sistemi di servizio con un certo numero di risorse, corrispondenti ai router (m), laddove ($i \geq m \implies queueing$).

Modello M/M/m. Dobbiamo però ritrovare le ipotesi dietro all'M/M/m nel mondo reale. Call-center ad esempio. Processo secondo il quale si susseguono gli arrivi al sistema a coda dev'essere ben modellabile come un processo di POISSON. Durata media delle telefonate (tempi medi di servizio) ($\frac{1}{\mu}$). In un call center mi ritrovo queste ipotesi. Ci vorrebbero anche le ipotesi sulla dimensione del buffer e della popolazione. ($d, e = +\infty$). Dimensionamento del numero massimo di operatori (servitori) in base ad un predeterminato massimo valore da non scavalcare relativo alla probabilità di accodamento. $\{\lambda, \frac{1}{\mu}\} \to m$. Problema di progetto. Potremo anche voler affrontare un problema di ANALISI. Durata media delle telefonate ($\frac{1}{\mu}$). [ANALISI $\to$ PROGETTO]. In funzione della C di ERLANG, troviamo il valore del ritardo medio per cliente:

$$\bar{T} = \frac{1}{\mu} + C\frac{1}{m\mu}(\frac{1}{1-\rho}),\ \rho < 1$$

Ricordiamo che ($C\ di\ ERLANG = \Pr\{queuing\}$). Adesso, noto il valore del ritardo medio per cliente, applico Little e trovo:

$$\bar{N} = \lambda\bar{T} = \frac{\lambda}{\mu} + C\frac{\lambda}{m\mu}(\frac{1}{1-\rho}) = [m\rho + C\rho(\frac{1}{1-\rho})]$$

Troviamo ora il tempo medio di permanenza nella sola fila di attesa, dove ($\rho := \frac{\lambda}{m\mu}$):

$$\bar{W} = \bar{T} - (T_s = \frac{1}{\mu}) = C\frac{1}{m\mu}(\frac{1}{1-\rho})$$

E adesso il numero medio di clienti nella sola fila di attesa, $\bar{N_q}$

$$[\bar{N_q} = \lambda \bar{W} = C\rho(\frac{1}{1-\rho})]$$

Dove abbiamo nuovamente applicato Little al sottosistema costituito dalla sola fila di attesa. Ora passiamo al centro di servizio, ragionando con gli stessi argomenti:

$$\begin{cases} (T_s = \frac{1}{\mu}) \\ \bar{N_s} = m\rho := \frac{\lambda}{\mu} = \lambda(\frac{1}{\mu}) \end{cases}$$

ρ è il FATTORE DI UTILIZZAZIONE. A REGIME ($\rho < 1$). Ricordiamo che λ è sempre la velocità di arrivo.

Confronto tra tre sistemi MARKOVIANI

Questi tre sistemi MARKOVIANI hanno la stessa capacità di servizio e stessa velocità tutti tra di loro:

- 1): Un singolo sistema M/M/1 con velocità di arrivo $m\lambda$ e velocità di servizio del singolo servitore pari a $m\mu$. Parametri: $\{m\lambda, m\mu\}$;
- 2): m code M/M/1, ogni coda con velocità di arrivo λ e di servizio $\mu \implies \{m\lambda, m\mu\}$ come parametri;
- 3) Sistema M/M/m con velocità di arrivo $m\lambda$ ed m servitori. Ricordiamo che qui vale:

$$[\frac{1}{\mu} + C\frac{1}{m\mu}(\frac{1}{1-\rho}) = \bar{T},\ (\rho < 1)]$$

  Abbiamo parametri: $\{m\lambda, m\mu\}$.

Tre sistemi che hanno tutti la stessa velocità di arrivo e di servizio. Confrontiamo i ritardi medi di permanenza per i clienti. Dobbiamo quindi stilare una graduatoria: Effettuiamo un confronto:

$$\begin{cases} \bar{T_1} = \frac{1}{m\mu - m\lambda} = \frac{1}{m(\mu - \lambda)} \\ \bar{T_2} = \frac{1}{\mu - \lambda} \\ \bar{T_3} = ? \end{cases}$$

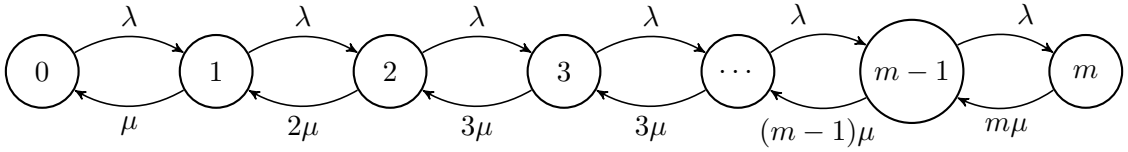
Dovremmo fare i conti con l'espressione di $\bar{T}$ precedentemente trovata. Ma, intuitivamente la classifica sarebbe: $\{1, 3, 2\}$. Il primo sarebbe il migliore, sebbene poco attuabile nella pratica. Ci accontentiamo del terzo sistema. Primo sistema non attuabile come umani come router. Il primo lavora sempre alla capacità di servizio, mentre il terzo ha capacità di servizio $m\mu$ solo quando $i \geq m$!

M/M/m/0

In questo sistema abbiamo 0 come dimensione del buffer $\iff (d = 0) < +\infty$. NO POISSON modeling! In letteratura troviamo anche M/M/m/m $(d = m)$. Nel primo caso con d si intende la dimensione del buffer, mentre $(d = m)$ intende il numero totale di clienti nel sistema. Ovviamente M/M/m/0 $\neq$ M/M/m, in quanto se $i \geq m$, i clienti in arrivo vengono irrimediabilmente persi. m router e velocità μ ciascuno $(\frac{1}{\mu})$ ritardo medio per cliente. v.a. memoryless EXP (UNI-NEG). statisticamente indipendenti i.d. ed indipendenti statisticamente dai tempi di interarrivo (MARKOV). $\lambda_D = \lambda_S$, dove D sta per *departure*. Si può verificare quindi il fenomeno di LOSS. Sistema con blocco, ma con perdita (Non si può fare fila in coda). λ_S sarà anche la velocità di partenza! Le perdite fanno perdere la proprietà di POISSON per il processo che modella il numero di clienti. Di conseguenza il processo secondo il quale entrano i clienti nel centro di servizio NON è più di POISSON, il quale modellerebbe la distribuzione di massa in tal modo:

$$\Pr\{k \ arrivi \ in \ \tau\} = \frac{(\lambda\tau)^k}{k!} \, \mathrm{e}^{-\lambda\tau}$$

Ma in questo caso è 0 se $k \geq m$! $N(t)$ è il numero di clienti nel sistema a coda al tempo t. Sostanzialmente coincide con il numero di clienti nel centro di servizio del sistema. $N(t)$. Sia una CMTC, dal momento che abbiamo indipendenza tra λ e μ. Il diagramma DTT è il seguente:



Mi trovo dinanzi ad una catena di nascita e morte (BDCMTC) con insieme degli stati FINITO ($\iff |stati| < +\infty$). Abbiamo i seguenti tassi di transizione, trovati rispetto al DTT relativi all'M/M/m (lì avevamo però un numero di stati infinito):

$$\begin{cases} \lambda_i = \lambda \ \forall i = 0, 1, \ \ldots, \ m-1 \\ \mu_i = i\mu, \ i = 1, \ \ldots, \ m \end{cases}$$

ove l'ultimo range di indici relativo al tassi di morte è tale per cui sarebbe 0 altrimenti. La catena è ERGODICA $\iff$ esiste sicuramente la distribuzione di regime:

$$\begin{cases} [\pi_i = \pi_0 (\frac{\lambda}{\mu})^i (\frac{1}{i!})], \ i = 1, \ \ldots, m \\ [[\pi_0 = \frac{1}{1 + \sum_{k=1}^{m} (\frac{\lambda}{\mu})^k (\frac{1}{k!})}] = \underline{\frac{1}{\sum_{k=0}^{m} (\frac{\lambda}{\mu})^k (\frac{1}{k!})}}] \end{cases}$$

ove nell'ultimo passaggio sottolineato dell'equazione abbiamo inglobato l'1 nella sommatoria. Ora sappiamo già che è ERGODICA (la sommatoria è una semplice SOMMA di un numero finito di elementi, quindi convergerà sicuramente).

$$[\pi_i = \frac{(\frac{\lambda}{\mu})^i (\frac{1}{i!})}{\sum_{k=0}^{m} (\frac{\lambda}{\mu})^k (\frac{1}{k!})}]$$

$\leftarrow$ ovvero la probabilità di regime, per $\underline{0 \leq i \leq m}$. Quando $(i = 0)$, trovo esattamente π_0 in evidenza.

$\bar{N}$ sia il numero medio di clienti nel sistema. $\implies \bar{N} = \sum_{i=0}^{m} i\pi_i$. Possiamo applicare anche Little al centro di servizio: $\implies \bar{N} = \lambda_S(\frac{1}{\mu})$. Abbiamo:

$$[\lambda_S = \sum_{i=0}^{m-1} \lambda\pi_i] = \underline{\lambda(1-\pi_m)} =$$

$$= [\lambda\pi_0 + \lambda\pi_1 + \lambda\pi_2 + \ \cdots + \ \lambda\pi_{m-1}]$$

dove il termine sottolineato si riferisce alla NORMALIZZAZIONE (per via della condizione di NORMALIZZAZIONE). Stiamo sostanzialmente sommando le frequenze di transizione, considerando quelle transizioni che si riferiscono all'evento {*entra un cliente nel mio sistema*}. Sommatoria delle frequenze. Ogni qualvolta vi è una transizione, significa che è arrivato un cliente che è entrato nel centro di servizio. Per trovare il numero medio di clienti che entrano nel mio sistema a coda, basta sommare le frequenze di queste transizioni. Ad un certo punto, i clienti se ne escono $\implies [\lambda_D = \lambda_S]$. Potremo effettuare lo stesso ragionamento:

$$\pi_1\mu + \pi_2(2\mu) + \pi_3(3\mu) + \ \cdots + \ \pi_m(m\mu) = \lambda_D = \lambda_S$$

Ad esempio $\pi_1\mu$ rappresenta il numero medio di volte che accade questo evento di interesse per unità di volte o frequenze di transizioni. $\pi_i q_{ij}$ è anche chiamato flusso, FLUSSO dallo stato i allo stato j. Frequenza di transizione. $\{\pi_j q_{ji}, \pi_i q_{ij}\} \mid \pi_1\mu = \pi_0\lambda$. Quest'ultima quantità rappresenta quante volte in media un cliente all'arrivo trova 0 clienti. Queste sono le cosiddette equazioni di bilanciamento locale, e sono valide solo per una CMTC nascita e morte. Ad esempio: $[\pi_1\mu = \pi_0\lambda]$, $\pi_{m-1}\lambda = \underline{\pi_m m\mu}$, ovvero la frequenza con la quale un cliente arriva e si becca l'unico router disponibile.

Una importante quantità è la cosiddetta probabilità di perdita (legata alla B di ERLANG).

$$\Pr\{LOSS\} = \underline{\pi_m^{(a)} = \pi_m} = \frac{(\frac{\lambda}{\mu})^m \frac{1}{m!}}{\sum_{i=0}^{m} (\frac{\lambda}{\mu})^i (\frac{1}{i!})} = B(m, \ \frac{\lambda}{\mu})$$

Tale quantità è la B di ERLANG con parametri m, $\frac{\lambda}{\mu}$. Ancora più importante della prima. Con questa formula sono state dimensionate opportunamente le centrali telefoniche. Certe quantità di servizi ma NON è possibile fare fila di attesa nel caso in cui $i \geq m$. Per un call center NON si potrebbe utilizzare ovviamente. Si dimostra che quanto scritto, vale per il sistema **M/G/m/0**, in generale, ovvero anche quando abbiamo una distribuzione arbitraria dei tempi di servizio.

Typical scenario

Fino a poco tempo fa i RAS erano dei router con delle porte seriali alle quali si collegavano i modem (SLIP). Il computer chiedeva al modem di collegarsi ad un altro utente, ed era necessario che ci fosse ovviamente un'interfaccia libera. Ad ogni borchia era associato un numero di telefono. Allora le applicazioni non richiedevano tutta questa banda.

Circuiti telefonici tra un modem ed un altro modem. PPP = *Point to Point Protocol*. Tramite PPP venivano inoltrati i vari pacchetti IP. Questo sistema può essere studiato con un M/M/m/0. Ipotesi da fare nel sistema reale. I clienti arrivano con un processo di POISSON a velocità λ. m porte corrispondenti agli m router. I circuiti telefonici non si creerebbero se $i \geq m$. Perdita. Le richieste di chiamata si devono susseguire in un modo tale da essere ben modellate con un processo di POISSON. Velocità medie delle richieste. I tempi di servizio dovevano essere v.a. i.i.d. (EXP-UNINEG). $(\frac{1}{\mu})$. Qui i tempi di servizio sono i tempi di collegamento. Anche

indipendenti dai tempi di interarrivo, ovviamente. M/G/m/0. Anche se non fossero (EXP-UNINEG), funzionerebbero lo stesso per questo tipo di sistema, difatti quanto un utente sta collegato, non influenza i vari tempi.

Posso effettuare un problema di progetto. Dati λ, $\frac{1}{\mu}$, ove l'ultimo parametro rappresenta la durata media dei tempi di collegamento, dobbiamo risolvere il seguente problema:

$$\min m \mid \Pr\{LOSS\} < \underline{S}$$

ove la quantità sottolineata è un'apposita threshold. È interesse del provider che il suo sistema funzioni bene. Si utilizzi la B di ERLANG ($B(m, \frac{\lambda}{\mu})$). Se B è elevata, dobbiamo evidentemente aumentare $m \iff m \uparrow$, e pensare ad un problema di PROGETTO.

Con l'ADSL si utilizza l'ATM (dimensionamento del sistema). L'*Access Concentrator* è un router del provider collegato mediante interfaccia ATM alla rete ATM. Varie possibilità di PPP: {PPPoA, PPPoE}. [DSLAM = *Digital Subscriber Link Access Multiplexer*]. Multiplatore dell'accesso da parte di linee che giungono da diversi clienti. Dal router ADSL ci si collega sino al DSLAM, che si trova in centrali, mediante DOPPINO. Nel caso PPPoA, il PPP agisce tra i due estremi (router ADSL ed Access Concentrator). Il DSLAM è come se fosse uno switch ATM. Ricordiamo lo stack: $\{IP \to PPP \to AAL5 \to PHY\}$. Il PPP è poggiato sull'ATM. Attraverso le celle ATM, il pacchetto giungerà all'Access Concentrator e verrà adeguatamente deimbustato. Con il PPPoE, la sessione PPP è tra il mio computer e l'Access Concentrator. IN TAL CASO NON SI COMPORTANO PIU' COME ROUTER! Access Concentrator ed il primo router fanno da BRIDGE REMOTI! In tal caso lo stack sarebbe: $\{IP \to PPP \to Eth. \to PHY\}$. Pacchetto IP imbustato in PPP a livello di computer. La trama Ethernet, quando arriva al router bridge, tramite le celle ATM viaggerà e verrà alla fine deimbustato (all'altra estremità del circuito virtuale). L'Access Concentrator ovviamente avrà m router.

Exercise

Si debba progettare un servizio di accesso remoto basato su VPN IPSec di cui possano usufruire i dipendenti di un'azienda.

Si assuma che le richieste di instaurazione di una VPN arrivino secondo un processo di Poisson a velocità $\lambda = 200\ req/s$ e che le connessioni sicure durino in media $\frac{1}{\mu} = 30\ min.$ Determinare il numero minimo **m** di VPN contemporanee che il concentratore di VPN deve essere in grado di supportare affinché la probabilità di rifiuto di una richiesta in arrivo sia minore di $S = 0.01$.

Il candidato introduca le eventuali ulteriori ipotesi necessarie al fine di poter utilizzare il modello stocastico che intende adottare per la progettazione.

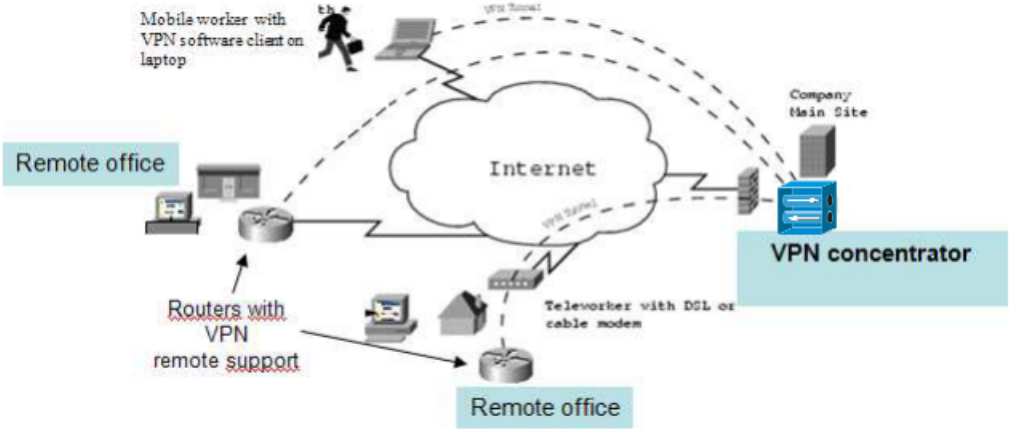


Figura 2.2: VPN IPSec

Risoluzione:

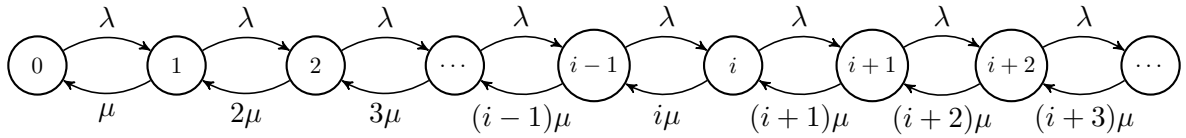
$S = 0.01$. VPN IPSec. Accesso da remoto ad Internet e si basa sulla presenza di un *VPN Concentrator* (concentratore di reti virtuali). IPSec, SSL. Meccanismi di sicurezza. Concentratore tramite VPN. Un router normale ne supporta fino ad una centinaia di VPN (Grandi aziende come CISCO, IBM). Le richieste di instaurazione arrivano secondo un processo di POISSON. $\lambda = 200\, req/h$. Inoltre $\frac{1}{\mu} = 30\, min$ (tempo medio di servizio). $m = ? \mid \Pr\{LOSS\} < (S = 0.01)$.

M/M/m/0. Il processo degli arrivi di POISSON. Ci viene chiesto il numero di VPN contemporanee da supportare. Sistema con PERDITA (LOSS). ed ipotesi standard sui vari tempi. m router. m numero di VPN contemporanee. FCFS come disciplina di coda. Non c'è una coda però, quindi non conta tanto. ($\iff d = 0, (e = +\infty)$).

Vogliamo utilizzare le tabelle (B di ERLANG). Il CARICO MEDIO di LAVORO (INTENSITÀ DI TRAFFICO) è $(\frac{\lambda}{\mu}) = (200x0.5) = 100 \implies (118 = m)$. $S = 0.1$ (percentuale di perdita). [ERLANG] Sebbene sia in realtà una quantità adimensionata.

2.2.3 Sistemi a coda M/M/inf

Sistema a coda costituito da un insieme illimitato di router. NON ABBIAMO FILA DI ATTESA! (Risorse illimitate). I clienti NON sono costretti a permanere in fila di attesa. Essa è quindi praticamente irrilevante. $N(t)$ al solito è il numero di clienti nel sistema a tempo t. Spazio degli stati di dimensione illimitata $\iff S = \{0, 1, 2, \dots\}$. Stesse ipotesi. $N(t)$ CATENA DI MARKOV a tempo continuo (CMTC), nascita e morte, con il seguente DTT:



Abbiamo:

$$\begin{cases} \lambda_i = \lambda, \ i \geq 0 \\ \mu_i = i\mu, \ i \geq 1 \end{cases}$$

Tassi di nascita λ, tassi di morte $i\mu$. Numero di stati illimitato. La CATENA è OMOGENEA, IRRIDUCIBILE. Dobbiamo vedere se è ERGODICA:

$$\begin{cases} [\pi_i = \pi_0 (\frac{\lambda}{\mu})^i \frac{1}{i!}] \\ [\pi_0 = \dfrac{1}{1 + \sum_{k=1}^{\infty} (\frac{\lambda}{\mu})^k \frac{1}{k!}}] \end{cases}$$

Anzitutto come sempre inglobiamo il termine 1 nella sommatoria dell'ultima equazione. Essa converge sempre. Abbiamo:

$$\pi_0 = \frac{1}{\sum_{k=0}^{\infty} (\frac{\lambda}{\mu})^k \frac{1}{k!}} = \mathrm{e}^{-\frac{\lambda}{\mu}}$$

ove si è sfruttato il fatto che: $[\sum_{i=0}^{\infty} x^i \frac{1}{i!} = \mathrm{e}^x]$. Quindi NO CONDIZIONE DI ERGODICITÀ. Tale serie CONVERGE SEMPRE. Converge sempre a patto che $\{\lambda, \mu\}$ siano finiti ($\iff \lambda, \mu < +\infty$). Abbiamo:

$$\pi_i = \pi_0 (\frac{\lambda}{\mu})^i \frac{1}{i!} = (\frac{\lambda}{\mu})^i \frac{1}{i!} \mathrm{e}^{-\frac{\lambda}{\mu}}, \ i \geq 0$$

CATENA SEMPRE ERGODICA. Non dobbiamo soddisfare nulla. Notiamo subito che riflette una DISTRIBUZIONE DI POISSON di parametro $\frac{\lambda}{\mu}$. Quindi: $[\bar{N} = (\frac{\lambda}{\mu})]$, anche se l'avrei potuto pure trovare con Little questo risultato: $\bar{N} = \lambda(\frac{1}{\mu})$. Quanto abbiamo detto vale per tempi di servizio distribuiti genericamente (M/G/∞). Tale sistema a coda è utilizzato per la modellazione dei tempi di propagazione. I tempi di propagazione sono considerati all'incirca costanti e pari a circa ($\sim \frac{2}{3}c$).

Capitolo 3

Affidabilità e Disponibilità

3.1 Starting Point

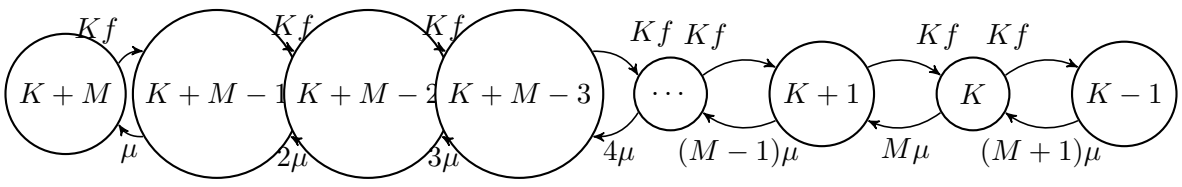
3.1.1 Exercise

CSP = *Content Service Provider*. Abbiamo una batteria di K server, tutti equivalenti. Server soggetti a guasti. M server di backup. Quando un server si blocca viene sostituito con un server di backup, SE DISPONIBILE. Se i K server funzionano tutti siamo in "*Normal State*", Se inferiori a K il sistema si blocca e siamo in "*Failure State*". Ce ne devono essere almeno K.

{TTF per il processo expneg. con parametro f_p, TTF memoria expneg. con parametro f_m, TTF disco expneg. con parametro f_d}.

DTT, probabilità di regime, frazione di tempo Failure State, MTTF del sistema; durata media dell'intervallo di tempo tra un istante in cui il servizio è attivato e l'istante successivo corrispondente alla sospensione del servizio.

In totale ci sono $\underline{K+M}$ server (server attivi + server di backup). La situazione normale sarebbe il seguente tableau: $\{K, 0, M\}$, ove i tre parametri indicano rispettivamente i server funzionanti della server farm, quelli in riparazione e quelli di backup. Ad un certo punto potrebbe accadere che nella server farm vi siano K server ed M in riparazione. Un ulteriore guasto della server farm porterebbe il sistema ad entrare in Failure State. Dinamica relativa al sistema. Definizione di Stato: processo stocastico: numero di server nella server farm e nei server di backup. Questa definizione di stato ci va bene per un processo Markoviano. Le tre v.a. sono indipendenti. Il tempo di guasto di un server sarà il minimo di queste tre v.a. dei TTF. Se quindi $f = f_p + f_m + f_d$, allora il tempo di guasto del server sarà tale che $TTF \sim EXP(f)$. I tempi di guasto dei vari server sono v.a. indipendenti. Le riparazioni procedono in parallelo tra di loro. Tempi di riparazione indipendenti dai tempi di guasto $\iff$ PROCESSO MARKOVIANO, il cui DTT è il seguente:



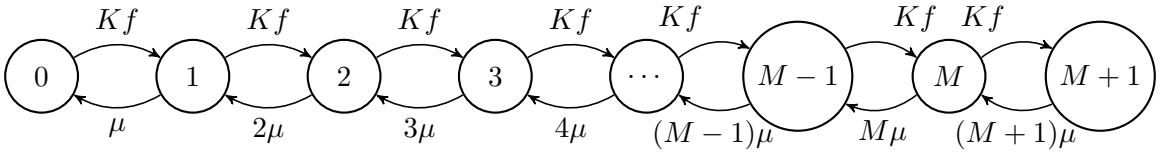
Il numero di stati è finito ($\iff |stati| < +\infty$). Si suppone che le riparazioni avvengano in parallelo, altrimenti se ci fosse una Single Repair Facility μ sarebbe costante. Quindi abbiamo: $N(t) = K + M$, ovvero K server nella server farm che si possono guastare. Il tempo di guasto è dato da $\rightarrow$

$$\phi_{K+M}(t) = \underline{\min(\dots)}$$

ove la parte sottolineata indica i tempi di guasto residui al tempo t. Se siamo nello stato K, vi sono M in riparazione e 0 nella stanza dei server di backup. Può accadere che ad un certo punto si guasti un ulteriore server della server farm $\implies$ FAILURE STATE. La velocità totale di uscita dallo stato $K+M$ sarebbe proprio Kf.

- **NORMAL STATE** fino a K;
- **FAILURE STATE** $K-1$;

Oscillazione tra normal state e failure state. La seguente Catena di Markov vagherà tra questi stati. Non esiste una singola definizione di stato. Ce ne possono essere differenti. Alcune volte delle differenti definizioni di stato potrebbero portare allo stesso DTT. Considerando ad esempio $N(t)$ come il numero di server in riparazione avremo il seguente DTT:



ove cambierebbero solo le label rispetto al precedente. Non è importante il modo in cui si chiamino gli stati, ma le loro transizioni! PROCESSO STOCASTICO: numero di server in riparazione al tempo t; CATENA ERGODICA (OMOGENEA, IRRIDUCIBILE con $|stati| < +\infty$). In funzione della distribuzione di regime dobbiamo trovare altre quantità:

π_{M+1} sarebbe la frazione di tempo a regime ove il servizio è sospeso, quindi $\underline{1-\pi_{M+1}} = \underline{\pi_0 + \cdots + \pi_M}$ sarebbe invece la frazione di tempo a regime nel quale il sistema si trova nello stato NORMAL, per via della condizione di NORMALIZZAZIONE. Il LHS è proprio la DISPONIBILITÀ.

$\underline{1-\pi_{M+1}}$ AVAILABILITY del mio sistema di servizio. MTTF: durata (media) dell'intervallo che è costituito dall'istante di tempo di funzionamento alla sospensione. MTTF = *Mean Time To Failure*. Rappresenta un processo stocastico, sebbene non markoviano, con stati: $S = \{ON,\ OFF\}$, ove ON sta per NORMAL ed OFF per FAILURE. Ad un certo punto la catena sarà in M. L'MTTF è la durata media dell'intervallo di tempo che va dall'inizio dello stato M sino all'istante di Failure, ovvero un istante immediatamente prima delle sospensione del servizio. Invece MTTR significa *Mean Time To Recovery*.

Definition 16. ***AVAILABILITY***
L'AVAILABILITY è la frazione del tempo in cui il sistema sta ad erogare il servizio:

$$A := [\frac{MTTF}{MTTF + MTTR}]$$

Noi parleremo di ALTA DISPONIBILITÀ. 5-9 (99.999%), che rappresenta un downtime annuo di circa 5 minuti. Disponibilità molto alta in realtà bancarie. L'MTTR corrisponde al tempo medio di soggiorno della mia catena nello stato $M+1$. Il valor medio del tempo di soggiorno è $\frac{1}{(M+1)\mu}$.

Consideriamo il seguente risultato:

$$MTTF = \frac{1}{(M+1)\mu\pi_{M+1}} - (\dots)$$

dove $(\dots)$ rappresenta il tempo medio di soggiorno da sottrarre (durata media di quell'intervallo). Quindi:

$$MTTF = (\dots) = \frac{1}{(M+1)\mu\pi_{M+1}} - \frac{1}{(M+1)\mu} = \frac{1}{(M+1)\mu}\frac{1-\pi_{M+1}}{\pi_{M+1}} \implies$$

$$\implies [\frac{MTTF}{\frac{1}{(M+1)\mu}} = \frac{1-\pi_{M+1}}{\pi_{M+1}}] \implies \frac{MTTF}{MTTR} = \frac{1-\pi_{M+1}}{\pi_{M+1}}$$

Vale il BILANCIAMENTO LOCALE. Abbiamo che:

- $\frac{1}{(M+1)\mu}$ è il tempo medio di soggiorno della mia catena nello stato $M+1$;
- $\frac{1}{(M+1)\mu\pi_{M+1}}$ è il tempo medio di ricorrenza (o di ritorno) nello stato $M+1$;

La differenza ci ritorna l'MTTF. Abbiamo sfruttato il fatto che la frequenza dell'evento ritorno nello stato di normal state è: $\pi_{M+1}(M+1)\mu$, mentre a titolo informativo la frequenza dell'evento ingresso nello stato di failure è $\pi_M K f$. Le due frequenze sono ovviamente uguali in virtù dell'applicazione dell'equazione di bilanciamento locale, dato che questa CMTC è di tipo Nascita e Morte.

Se si volesse utilizzare la tecnica degli stati assorbenti (vedasi **CMTC with Absorbing States**), allora bisognerebbe considerare come STATO ASSORBENTE {M+1}, e bisognerebbe tenere opportunamente in conto le condizioni iniziali del processo.

Sistemi di servizio che possono trovarsi in due stati: {NORMAL STATE, FAILURE STATE}, mappati rispettivamente in $SYS : \{ON, OFF\}$. Prendiamo in considerazione due sistemi:

- *Primo sistema*:
  Tale sistema mediamente per $1s$ si trovi nello stato OFF, e per $9s$ nello stato ON. La disponibilità di questo sistema è: $A = \frac{9s}{10s} = 90\%$, ovvero per il 90% del tempo il sistema è perfettamente funzionante, nello stato di ON.
- *Secondo sistema*:
  Tale sistema invece è UP per $0.9s$, e down per $0.1s$. Abbiamo sempre una disponibilità di $A = \frac{0.9s}{1s} = 90\%$, come il primo sistema;

La disponibilità è la medesima per i due sistemi, ma l'affidabilità invece è maggiore nel primo (non ha a che fare con il tempo di Recovery).

Definition 17. ***AFFIDABILITÀ***
L'Affidabilità è formalmente definita come:

$$R(t) := \Pr\{X > t\} = Reliability(t)$$

Non sarebbe nient'altro che la CDF complementare della v.a. X, la quale rappresenta il tempo di vita del sistema. Non pensiamo alla riparazione o sostituzione. Si parte dall'affidabilità dei singoli componenti. Per l'analisi di affidabilità sono molto utili gli RBD, ovvero i *Reliability Block Diagram*.

3.1.2 Ridondanza

Web server SW processo applicativo che si guasta con un failure rate γ_p. In esecuzione con una macchina che si guasta indipendentemente a failure rate γ_m. Il failure rate corrisponde ad un parametro della distribuzione esponenziale. Definizione di failure rate dipendente dal tempo. Vi è un meccanismo automatico di *Failure Detection*, basato sul polling. Il tempo medio necessario per rilevare il failure sul server sia $\frac{1}{\delta_p}$ e che $\frac{1}{\delta_m}$ sia il tempo medio di rilevazione failure macchina.

Quando la macchina ha un malfunzionamento, il processo applicativo è migrato su una macchina di riserva, se DISPONIBILE. $\frac{1}{\tau_m}$ è il tempo medio necessario per avviare la macchina di backup. Se invece si ha un guasto soltanto del processo server, viene automaticamente riavviato sulla stessa macchina. Il tempo medio di restart del server software è $\frac{1}{\tau_p}$. Tipicamente $[\tau_p > \tau_m] \implies \frac{1}{\tau_p} < \frac{1}{\tau_m}$. C'è una piccola probabilità $(1-c)$ che il restart del processo sulla stessa macchina non vada a buon fine, nel qual caso è avviato sulla macchina di riserva. Questo schema di (re)start automatico dopo i guasti è anche chiamato "*cold replication*". Quando una macchina crasha, è necessario un recovery più complesso dal relativo rate μ. Il Web server è considerato disponibile quando sia il processo server che la macchina sulla quale esso gira sono entrambi funzionanti. Si valuti la *Steady-State Availability* del server, assumendo che non vi siano ulteriori guasti del processo o della macchina fino a che non siano stati adeguatamente trattati (risolti).

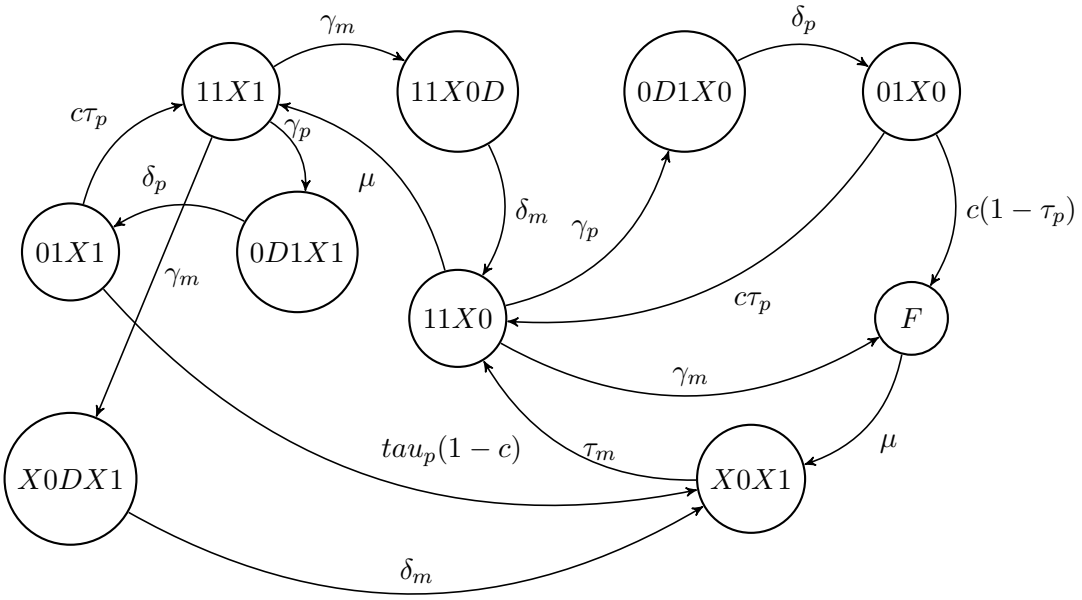
Quindi la DISPONIBILITÀ viene meno quando nè il processo nè la macchina sono funzionanti. AFFIDABILITÀ di un sistema cold-replicated $\implies$ RIDONDANZA. RIDONDANZA o *REPLICATION ATTIVA* quando vi sono oltre ai server operativi anche dei server di backup, tutti quanti soggetti alla stessa quantità di lavoro (in tal caso anche quelli di backup lavorano come quelli normali). Si suppone che il failure rate sia il medesimo per entrambi i tipi. Il contrario è la REPLICATION PASSIVA $\rightarrow$ {WARM REPLICATION, COLD REPLICATION}. Si immagini di avere due server, due macchine. Uno attivo è l'altro di backup. Con la WARM, di tanto in tanto quello di backup interagisce con quello attivo per ricevere le sue strutture dati. Un malfunzionamento di quello attivo fa sì che si attivi quello di backup. Ma il relativo tempo del recovery è molto basso. Ci piace questo, però a trasferire queste strutture dati periodicamente, una parte delle capacità elaborative sarà riservata a questa mansione. Potrebbe invece eseguire altri jobs al posto di questi. Vantaggi in tempi di Recovery, però il problema è legato ad un throughput più basso del server attivo. Con quella a freddo abbiamo un tempo di recovery più alto, ma sull'attivo sfruttiamo perlomeno tutta la capacità elaborativa. Parametri: {performance, affidabilità della rete (dei vari dispositivi di cui la rete si compone) (*reliability R(t), fault tolerance*), Sicurezza dati}. L'esercizio in questione propone la ridondanza di un Server. Potrebbero esserci dei *WATCH-DOG*, ovvero dei "cani da guardia" che controllano il server primario. Controllo delle eventuali degradazioni delle prestazioni in un cluster - Policing, disciplina decidibile.

Studiamo il sistema con un approccio Markoviano. DTT della CMTC che modella il sistema. Faremo l'ipotesi di distribuzione esponenziale per le v.a. in gioco (Anche per la distribuzione relativa al Failure Detection Time).

$$\begin{cases} F_p \sim EXP(\gamma_p),\ F_m \sim EXP(\gamma_m) \\ D_p \sim EXP(\delta_p),\ D_m \sim EXP(\delta_m) \\ R_p \sim EXP(\tau_p),\ R_m \sim EXP(\tau_m) \end{cases}$$

Rispettivamente, le prime due equazioni della prima riga indicano la distribuzione dei time to failure di un processo o della macchina, le successive due della seconda riga indicano la distribuzione dei time to failure detection del processo o della macchina, e le ultime due della terza riga indicano la distribuzione del time to (re)start del processo sulla stessa macchina

(SAME MACHINE) o del riavvio del processo sulla macchina di riserva (SPARE MACHINE). $(1-c)$ rappresenta invece la probabilità che il riavvio del processo sulla stessa macchina fallisca. Quindi c è il *coverage factor*, ovvero la probabilità di successo del riavvio del processo. Infine abbiamo: $C_r \sim EXP(\mu)$, ovvero il tempo di riparazione per una macchina andata in crash (crashed machine recovery). Il DTT è il seguente:



La definizione di stato si basa sui seguenti stati degli elementi {(Processo primario, Macchina primaria), (Processo secondario, Macchina secondaria)}, riassumibili nel seguente apposito tableau:

$$\begin{bmatrix} P_p & P_m \\ S_p & S_m \end{bmatrix}$$

Tale è la rappresentazione del DTT di una CMTC che modella un Web Server con Cold Replication con una macchina di backup. $\{P_p,\ P_m\}$ rappresentano rispettivamente gli stati del processo primario e della macchina primaria sulla quale questo processo è in esecuzione. I pedici $\{p, m\}$ si riferiscono rispettivamente al processo od alla macchina. Gli altri due indici, $\{S_p,\ S_m\}$ stanno per SPARE o secondary. E stanno per lo stato del processo server di backup e della relativa macchina sulla quale esso sta in esecuzione, al solito. $P \rightarrow primary,\ S \rightarrow spare.$ "1" $\rightarrow$ processo/macchina up; "0" $\rightarrow$ " processo/macchina down. Se $P_p = 1 \ \vee P_m = 1 \implies$ il processo/macchina primario/a è up. $P_p = 0 \vee\ P_m = 0 \implies$ processo/macchina primaria down. Vale anche per gli altri ovviamente. "$0D$" $\rightarrow$ processo/macchina è failed (guasta), con malfunzionamento, ma il rispettivo guasto non è ancora stato rilevato (To be detected). "X" $\rightarrow$ don't care. Valori possibili. Possiamo pensare di partire dallo stato: {1,1,X,1} $\implies$ $P_p = P_m = 1$. (processo e macchina primaria entrambi UP). Macchina secondaria funzionante e non ci interessa del processo secondario. Ai fini delle transizioni di stato, ci può essere un malfunzionamento o del processo primario, o della macchina primaria od ancora quella di backup. γ_p è il parametro della distribuzione esponenziale che modella il TTF del processo. Supponiamo si sia verificato un malfunzionamento del processo primario. Macchina primaria/secondaria entrambe UP $\rightarrow 0D = P_p \rightarrow P_p = 0$. Altra transizione. Macchine ancora entrambe UP. Dopo che il sistema di Failure Detection avrà fatto per l'appunto detection del failure, a velocità δ_p, lo stato sarà: {0,1,X,1}. A questo punto si tenta di riavviare il processo (come accade tipicamente nei SO)

sulla stessa macchina. Abbiamo un Coverage Factor pari a c, ovvero pari alla probabilità che il riavvio vada a buon fine. $(1-c)$ è per contro, la probabilità che vada male il riavvio. Se il riavvio sulla stessa macchina ha successo, migriamo conseguentemente verso lo stato iniziale {1,1,X,1} a velocità $c\tau_p$. Altrimenti con velocità $(1-c)\tau_p$ migriamo verso lo stato {X,0,X,1}. Siamo in situazione di macchina primaria Down. Con certezza la macchina per me è in crash. Macchina inutilizzabile. Siamo quindi in {X,0,X,1}. Macchina 1 in crash. Bisogna quindi cambiare macchina. τ_p è il parametro che caratterizza la distribuzione esponenziale relativa al restart time della macchina stessa. Consideriamo la v.a. tempo di soggiorno residuo nello stato: {0,1,X,1}, di indice 4. $\phi_4 \sim EXP(\cdot)$. Consideriamo:

$$\Pr\{\phi_4(t) > \tau\} = \Pr\{R_{Rp} > \tau\} = \mathrm{e}^{-\tau_p \tau}$$

dove R_{Rp} rappresenta il restart time del processo (residuo), e tale è la probabilità che esso sia maggiore di τ. Indica quanto manca ancora affinché il restart finisca. R_{Rp} sarà distribuita proprio come il Restart Time del processo sulla macchina attiva. R_p è il restart time, mentre R_{Rp} è il restart time residuo. Nell'istante t stiamo quindi in questo stato. Questo restart potrebbe andar bene od andar male. Velocità totale di uscita banalmente τ_p. Quindi:

$$\tau_{i,j} = \frac{q_{i,j}}{-q_{ii}} = \tau_{4,1} = \frac{q_{4,1}}{-q_{4,4}} = \frac{q_{4,1}}{\tau_p} \implies$$
$$\implies \tau_{4,1} = c \implies q_{4,1} = c\tau_p$$

La macchina è andata in crash, a questo punto. Bisogna quindi cambiare macchina. Ci vuole un altro tempo che indichiamo con R_m, ovvero il restart time del processo sulla macchina secondaria. Con velocità τ_m arriveremo da {X,0,X,1} → {1,1,X,0} (Non ho più una macchina di riserva). Quella primaria si è sostanzialmente scambiata con la secondaria (funzionante). A velocità μ dopodiché, avremo la riparazione e migriamo verso lo stato iniziale, nuovamente: {1,1,X,1}. Possono accadere però altre cose. Dobbiamo anche considerare che, o accada un malfunzionamento della macchina primaria o della secondaria. {1,1,X,1} → {X,0D,X,1} → {X,0,X,1} a velocità δ_m, quest'ultimo passaggio. Poi arriveremo verso {1,1,X,0} ad avvenuta sostituzione. γ_m è il parametro che caratterizza la distribuzione esponenziale della F_m, ovvero il TTF della macchina. Sempre γ_m per la failure. Partiamo da {1,1,X,0}. NO macchina secondaria a disposizione. O malfunzionamento del processo primario o della macchina secondaria. Se accade un malfunzionamento del processo primario, migrerò con velocità γ_p verso {0D,1,X,0}, ed a velocità δ_p verso {0,1,X,0}. Se si verifica un failure della macchina andremo in FAILURE direttamente a velocità $(1-c)\tau_p$. Ma se a partire da {1,1,X,0} si guasta completamente la macchina, allora migreremo direttamente in **FAILURE** totale a velocità γ_m. Notiamo che siamo in situazione di SRF (*Single Repair Facility*).

Studio di fattibilità. Non è importante l'etichetta che attribuiamo agli stati quanto il loro significato. A questo punto li etichettiamo. Procedura di Labelizing. Distribuzioni di regime. Sistema di 10 equazioni (9 eq. + 1 eq. normalizzazione). Sfruttiamo la conoscenza del valore dei diversi parametri in gioco. Si valuti quindi la Steady-State Availability del Server. Stati nei quali il Web Server funziona. Gli stati dove $P_p = 1$ sono:

$$\left\{ \begin{bmatrix} 1 & 1 \\ X & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ X & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ X & 0D \end{bmatrix} \right\}$$

A questo punto, abbiamo che: $[A = \pi_1 + \pi_6 + \pi_7]$, secondo la nuova notazione indiciale rappresentata nella seguente maniera, nella notazione *State name* → *State index*:

- {1,1,X,1} → 1;

- $\{0D,1,X,1\} \rightarrow 2$;
- $\{X,0D,X,1\} \rightarrow 3$;
- $\{0,1,X,1\} \rightarrow 4$;
- $\{X,0,X,1\} \rightarrow 5$;
- $\{1,1,X,0D\} \rightarrow 6$;
- $\{1,1,X,0\} \rightarrow 7$;
- $\{0D,1,X,0\} \rightarrow 8$;
- $\{0,1,X,0\} \rightarrow 9$
- $\{1,1,X,1\} \rightarrow 10$;

$$\begin{cases} \frac{1}{\gamma_p} = 10\ days, \frac{1}{\gamma_m} = 20\ days \\ \frac{1}{\delta_p} = 1s, \frac{1}{\delta_m} = 0.4/0.5s \\ \frac{1}{\tau_m} = 2\ min, \frac{1}{\tau_p} = 30s \end{cases}$$

$$\begin{cases} \pi_1 = \frac{1}{E},\ \pi_2 = \frac{1}{E}\frac{\gamma_p}{\delta_p},\ \pi_3 = \frac{1}{E}\frac{\gamma_m}{\delta_m},\ \pi_4 = \frac{1}{E}\frac{\gamma_p}{\tau_p} \\ \pi_5 = \frac{1}{E}\frac{[\gamma_m + (1-c)\tau_p][\mu + 2\gamma_m + (1-c)\tau_p]}{\mu\tau_m} \\ \pi_6 = \frac{1}{E}\frac{\gamma_m}{\delta_m},\ \pi_7 = \frac{1}{E}\frac{2\gamma_m + (1-c)\gamma_p}{\mu} \\ \pi_8 = \frac{1}{E}[2\gamma_m + (1-c)\gamma_p]\gamma_p,\ \pi_9 = \frac{1}{E}\frac{[2\gamma_m + (1-c)\gamma_p]\gamma_p}{\mu\tau_p} \\ \pi_{10} = \frac{1}{E}\frac{[2\gamma_m + (1-c)\gamma_p][\gamma_m + (1-c)\gamma_p]}{\mu^2} \end{cases}$$

dove:

$$E = 1 + \frac{\gamma_p}{\delta_p} + 2\frac{\gamma_m}{\delta_m} + \frac{\gamma_p}{\delta_p} + \frac{[\gamma_m + (1-c)\gamma_p][\mu + 2\gamma_m + (1-c)\gamma_p]}{\mu\tau_m} +$$

$$+ \frac{2\gamma_m + (1-c)\gamma_p}{\mu}[1 + \frac{\gamma_p}{\delta_p} + \frac{\gamma_p}{\tau_p}] + \frac{[2\gamma_m + (1-c)\gamma_p][\gamma_m + (1-c)\gamma_p]}{\mu^2}$$

ed otteniamo alla fine:

$$A = \pi_1 + \pi_6 + \pi_7 = \frac{1}{E}[1 + \frac{\gamma_p}{\delta_p} + \frac{2\gamma_m + (1-c)\gamma_p}{\mu}]$$

Abbiamo così trovato la Steady-State Availability A.

3.2 Affidabilità

{Affidabilità, Disponibilità}. Come può esser valutata l'Affidabilità di un Sistema. Affidabilità di un Dispositivo / servizio di Rete. Concetti legati ma differenti.

Reliability $R(t)$. Affidabilità. Indipendentemente dalla rete / servizi di rete. L'Affidabilità può esser applicata dappertutto. L'Affidabilità va PROGETTATA. Reliability Engineering. Di mezzo ci sono queste tecniche, che si servono principalmente dei
diagrammi a blocchi dell'affidabilità, detti RBD. Utilizzo delle CATENE DI MARKOV. Dato S un sistema, componente, apparato, switch lvl 2 o lvl 3, l'affidabilità esprime la capacità, relativamente a quel sistema, di funzionare correttamente per un certo periodo di tempo. X è il time to failure del sistema (lifetime del sistema). Indichiamo con $f(t)$ la PDF di X, ovvero la densità di probabilità. Sia invece $F(t)$ la CDF di X (funzione di distribuzione cumulativa). L'Affidabilità $R(t)$ è definita come:

Definition 18. ***Affidabilità***

$$R(t) := \Pr\{X > t\} = 1 - F(t) = F^c(t)$$

Non è nient'altro che la CDF complementare, ed indica la probabilità che il TTF sia maggiore di un certo tempo t, non meglio definito per il momento Probabilità che S sia funzionante correttamente in $[0, t)$. $\Pr\{S \textit{ correctly working in } [0, t)\}$. Tempo di vita almeno pari a t, ovvero che S sopravviva per almeno t unità di tempo. Normalmente si suppone che il sistema lavori correttamente nell'istante iniziale, ovvero $\iff R(0) = 1$ (con probabilità unitaria il lifetime sia maggiore di 0). Potrebbe anche accadere che in alcuni casi $\Pr\{X = 0\} = p \neq 0$ (che un dispositivo NON funzioni all'inizio). Normalmente si ha invece: $\Pr\{X = 0\} = (0 = p)$. Tipicamente quindi $R(0) = 1$. Poi si suppone che: $[\lim_{t\to+\infty} R(t) = 0] \iff$ un sistema NON possa funzionare indefinitamente. Inizialmente funzioni correttamente e prima o poi si guasterà. Sostanzialmente, graficamente parlando, $\Pr\{X > t\}$ rappresenta l'area sottesa dalla relativa PDF $f(t)$ in $[t, +\infty)$. $R(t)$ è la CDF complementare di X, quindi vale:

$$\begin{cases} [R(t) = \int_t^{+\infty} f(x)dx] \\ [R'(t) = -\dfrac{dF(t)}{dt} = -f(t)] \end{cases}$$

Ovvero che la derivata dell'affidabilità, $R'(t)$, non è nientemeno che la PDF del TTF X o lifetime cambiata di segno.

Qual'è invece il MTTF? Valor medio del time to failure? (tempo medio di vita):

$$MTTF = \mathbb{E}[X] = \int_0^{\infty} tf(t)dt = \int_0^{\infty} R(t)dt$$

ovvero che il MTTF è sostanzialmente l'integrale dell'affidabilità. Si dimostra integrando per parti.

$$\underline{\mathbb{E}[X]} = \int_0^{+\infty} [1 - F(t)]dt - \int_{-\infty}^{0} F(t)dt$$

L'integrazione per parti dimostra ciò. Questo vale generalmente quando $\mathbb{E}[X] \leq\geq 0$. Ma se il lifetime è maggiore di 0, come nel nostro caso, dal momento che è una variabile aleatoria soltanto a valori positivi (v.a. positiva), allora banalmente vale che:

$$\int_{-\infty}^{0} F(t)dt = 0 \implies \mathbb{E}[X] = \int_0^{+\infty} [1 - F(t)]dt = \int_0^{\infty} \underline{R(t)}dt$$

ovvero che il MTTF è l'integrale dell'affidabilità, come già detto. Ricordiamo che MTTF sarebbe il *Mean Time To Failure*, ed il MTBF significa *Mean Time Between Failure*. Sono sostanzialmente la stessa cosa ma nominate in modo diverso. A volte nei paper della CISCO si preferisce utilizzare il MTBF. I valori possono tranquillamente raggiungere 40 anni. Esistono metodi di calcolo che favoriscono il confronto (Metodi Standard).

Definiamo il: FAILURE RATE (istantaneo). (Instantaneous) Failure Rate. Partiamo dalla PDF della v.a. X TTF o lifetime:

$$f(t) = \frac{dF(t)}{dt} = \lim_{\Delta t \to 0} [\frac{F(t+\Delta t) - F(t)}{\Delta t}]$$

A numeratore abbiamo: $\Pr\{t < X \leq t + \Delta t\}$, per Δt sufficientemente piccolo. Per $\Delta t \to 0$, abbiamo che $f(t)\Delta t$ rappresenta la probabilità che il tempo di vita di quella v.a. vari tra t e $t + \Delta t$. Per $\Delta t \to 0$ il prodotto $f(t)\Delta t \to \Pr\{t < X \leq t + \Delta t\}$. Questa è una probabilità NON condizionata. Adesso consideriamo: $\Pr\{t < X \leq t + \Delta t \mid X > t\} = (\dots)$, che sarebbe la probabilità che, dato che il sistema è sopravvissuto per t unità di tempo, non sopravviva per ulteriori Δt unità di tempo. Abbiamo che:

$$(\dots) = \frac{\Pr\{t < X \leq t + \Delta t,\ X > t\}}{\Pr\{X > t\}} = [\frac{\Pr\{t < X \leq t + \Delta t\}}{\Pr\{X > t\}}] \implies$$

ove si è sfruttato il seguente fatto:

$$\{X > t\} \subset \{t < X \leq t + \Delta t\} \implies \Pr\{t < X \leq t\Delta t \cap X > t\} = \Pr\{t < X \leq t + \Delta t\}$$

A denominatore abbiamo l'Affidabilità $R(t)$. Definiamo ora l'IFT $h(t)$ come:

Definition 19. ***(Instantaneous) Failure Rate***
L'IFT (Instantaneous Failure Rate) è definito come:

$$h(t) := \lim_{\Delta t \to 0} [\frac{1}{\Delta t} \frac{\Pr\{t < X \leq t + \Delta t\}}{R(t)}] = \cdot(t)$$

Questa è la definizione del Failure Rate istantaneo. $\Delta t \to 0 \implies h(t)\Delta t$ rappresenta la probabilità che, supponendo che il dispositivo sia sopravvissuto per t unità di tempo, non sopravviva per ulteriori Δt unità di tempo. Ma notiamo che:

$$h(t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} \frac{F(t+\Delta t) - F(t)}{R(t)} = \frac{f(t)}{R(t)}$$

ovvero corrisponde al rapporto tra la PDF e la CDF complementare della v.a. X. Sappiamo inoltre che: $f(t) = -R'(t) \implies$

$$h(t) = -\frac{(R'(t) = -f(t))}{R(t)} = \frac{-R'(t)}{R(t)}$$

Ma essendo l'affidabilità una probabilità $\implies R(t) \in [0, 1]$. Quindi ne consegue che: $h(t) \geq f(t)$. Ma a questo punto, indipendentemente da $\Delta t \to 0$, abbiamo che: $[h(t)\Delta t \geq f(t)\Delta t]$. Vero indipendentemente dal valore di Δt. Quindi, ricapitolando, $f(t)\Delta t$ rappresenta la probabilità che il tempo di vita sia compreso tra t e $t+\Delta t$, mentre $h(t)\Delta t$ indica la probabilità che, sapendo che il dispositivo sia sopravvissuto sino a t, muoia nei prossimi Δt. $h(t)$ è quindi sensibilmente più grande di $f(t)$, indipendentemente da Δt.

Ci serve un modo teorico per ricavare questi valori. Abbiamo tre macro-fasi di vita del dispositivo, sintetizzabili in un grafico di $h(t)$ in funzione del tempo t. Abbiamo un andamento

cosiddetto a *CURVA di VASCA DA BAGNO*. Abbiamo la prima fase di mortalità infantile, che include l'avvenimento di eventuali difetti di fabbrica, quindi $h(t)$ è alto. Poi abbiamo la seconda fase, ove $h(t)$ è tipicamente basso, ovvero il ciclo di vita utile del dispositivo. Qualunque malfunzionamento qui deriva da PROBLEMI ESOGENI, ove lo stress è dovuto a cause esterne. Poi abbiamo un'ultima fase di senilità, ove comprensibilmente $h(t)$ torna ad esser nuovamente alto in quanto vi è l'USURA del dispositivo da tener in conto. $h(t) = \frac{f(t)}{R(t)} = \frac{-R'(t)}{R(t)}$ rappresenta quindi la velocità alla quale un sistema tende a rompersi.

Il Failure Rate NON è un qualcosa di costante, ma varia con il tempo! Nel calcolo pratico si ipotizza sempre che siamo nella seconda fase di vita del dispositivo, ovvero quella di vita utile.

I seguenti elementi: $\{MTTF = \mathbb{E}[X],\ R(t),\ h(t)\}$ sono in stretta relazione tra di loro. Sia N_0 il numero di dispositivi messi ad operare tutti insieme nell'istante $(t_0 = 0)$. Siano nelle stesse condizioni di lavoro di partenza. Per tutti INIZIA il tempo di vita. Same working conditions. Al generico istante t, possiamo osservare lo stato di questi dispositivi. $N_S(t)$ sia il numero di dispositivi SURVIVED (sopravvissuti), per i quali il tempo di vita NON è terminato. Vale: $N_F(t) = \underline{N_0 - N_S(t)} \implies [N_S(t) + N_F(t) = N_0]$ (popolazione iniziale). Se dovessi vedere come varia questa funzione $N_S(t)$ al variare del tempo, avrei una funzione a gradino decrescente. I guasti, ovvero le variazioni del grafico, avvengono in istanti t_i. L'intervallo $[0, t_1]$ comprende il tempo di vita del primo dispositivo che si è guastato, e così via... Set di dati corrispondenti ai TTF dei vari dispositivi. Mediando (aritmeticamente), otteniamo:

$$\underline{\hat{\mathbb{E}}[X]} = \frac{t_1 + t_2 + \ \cdots + \ t_{N_0}}{N_0} = \frac{1}{N_0} \sum_{i=0}^{N_0} t_i$$

Sarebbe la media empirica dei tempi di guasto. Se $N_0 \to +\infty$, quella media empirica tende all'$\mathbb{E}[X]$. Con una popolazione sufficientemente grande, $\hat{\mathbb{E}[X]} \to \mathbb{E}[X]$. Abbiamo una BUONA STIMA procedendo in questo modo. È richiesto un numero sufficientemente grande di dispositivi iniziali. Come stimiamo invece l'Affidabilità? $\iff \hat{R}(t) =?$ Si immagini che $X \sim EXP(\lambda)$. A questo punto abbiamo:

$$h(t) = \frac{-R'(t)}{R(t)} = \frac{f(t)}{R(t)} = \frac{\lambda\, \mathrm{e}^{-\lambda t}}{\mathrm{e}^{-\lambda t}} = \lambda (\neq \cdot(t))$$

Quindi abbiamo un IFR $h(t)$ costante e pari a λ quando $X \sim EXP(\lambda)$. Calcoliamo $\mathbb{E}[X]$:

$$MTTF = \mathbb{E}[X] = \int_0^\infty R(t)dt = \frac{1}{\lambda} \int_0^\infty \lambda\, \mathrm{e}^{-\lambda t} dt = (\frac{1}{\lambda}) * 1$$

Se la distribuzione del TTF è esponenziale, $\lambda = h(t)$ ed $(\frac{1}{\lambda})$ è il suo valore medio. TTF è ovviamente il lifetime. $\mathbb{E}[X]$ è l'inverso del FAILURE RATE in tal caso, che è costante.

Tipicamente il TTF di una macchina potrebbe avere come parametro: $[f = f_p + f_m + f_d]$. In tal caso:

$$\begin{cases} f = f_p + f_m + f_d \\ X \sim EXP(f) \\ R(t) = \mathrm{e}^{-ft} \end{cases}$$

Diamo una definizione:

Definition 20. ***funzione di affidabilità EMPIRICA***

$$\hat{R}(t) = \frac{N_S(t)}{N_0}$$

È chiaro che se $N_0 \to +\infty \implies \hat{R}(t) \to R(t)$.

Definition 21. ***failure rate empirico***

$$\hat{h}(t) = -(\frac{1}{\Delta t})\frac{[\hat{R}(t+\Delta t) - \hat{R}(t)]}{\hat{R}(t)}$$

Al solito, $N_0 \to +\infty \implies \hat{h}(t) \to h(t)$.

3.3 Disponibilità

Definition 22. ***Disponibilità***

Si definisce formalmente la DISPONIBILITÀ ISTANTANEA:

$$\underline{A(t)} := \Pr\{in\ t\ il\ dispositivo\ sia\ UP\}$$

Inizieremo dalla DISPONIBILITÀ ISTANTANEA, per poi passare dalla DISPONIBILITÀ IN UN INTERVALLO sino ad arrivare alla DISPONIBILITÀ A REGIME.

AVAILABILITY = DISPONIBILITÀ. Dobbiamo pensare alla riparazione/sostituzione di un componente del dispositivo. *Instantaneous Availability* $A(t)$, probabilità che un sistema stia funzionando correttamente al tempo t. Questo indipendentemente dal numero di guasti o riparazioni che possano essere avvenuti prima di t. Consideriamo la v.a. $I(t)$ indicatrice:

$$I(t) := \begin{cases} 1,\ sistema\ UP \\ 0,\ sistema\ DOWN \end{cases}$$

È un indicatore dello stato del sistema. Il valor medio di $I(t)$, per un noto lemma, è:

$$\underline{A(t)} = \Pr\{I(t) = 1\} = \underline{\mathbb{E}[I(t)]}$$

$I(t)$ è una v.a. binaria $\iff$ può assumere solo valori 0 e 1. Pensiamo ad una singola realizzazione di $I(t)$. Indichiamo con x_i la durata dell'i-esimo periodo di up e con D_i la durata dell'i-esimo periodo di down $\iff \{x_i, D_i\} \to \{UP,\ DOWN\}$. Definiamo la:

Definition 23. ***Interval Availability***

$$\bar{A}(t) := \underline{\frac{1}{t}\int_0^t A(x)dx}$$

Somiglia molto ad una media temporale. Questa quantità è la frazione di tempo nella quale il sistema è UP limitatamente alla finestra temporale $[0, t]$.

$$\bar{A}(t) = \frac{1}{t}\int_0^t \mathbb{E}[I(x)]dx = \frac{1}{t}\underline{\mathbb{E}[\int_0^t I(x)dx]}$$

Ove l'ultima uguaglianza deriva per linearità, ed il termine sottolineato è il
Mean Total Uptime (MTU). Questo integrale ha a che fare con il total Uptime in $[0, t]$, ovvero limitatamente a quell'intervallo: $\int_0^t I(x)dx$. Questa quantità sarà alla fine uguale all'MTTF.

Definiamo:

Definition 24. ***STEADY-STATE AVAILABILITY***

$$A := \lim_{t \to +\infty} \bar{A}(t)$$

ovvero il limite dell'Interval Availability. Ricordiamo che MTTF, MTTR significano rispettivamente: *Mean Time To Failure* e *Mean Time To Recovery/Repair*, sebbene per l'ultimo acronimo si preferisca la terminologia *Recovery*, perché si preferisce includere anche i tempi di Detection e di sopralluogo.

Una formula che troviamo in letteratura è:

$$[A = \frac{MTTF}{MTTF + MTTR}]$$

ove A rappresenta la frazione del tempo ove il sistema è UP, ovvero la durata media del ciclo UP-DOWN. Sta erogando il servizio per il quale è stato pensato. Troviamo in realtà anche questa formula:

$$A = \frac{MTBF}{\underline{MTBF + MTTR}}$$

Se abbiamo sistemi che una volta guasti vengono riparati e sostituiti. Si utilizzano in realtà entrambe le notazioni. Quindi $MTBF = MTTF$, non c'è nessuna differenza. Si utilizza l'$MTTF$ quando NON abbiamo possibilità di riparazione.

$$A = \frac{durata\ media\ periodo\ UP}{durata\ media\ periodo\ UP + durata\ media\ periodo\ DOWN} = \frac{MTTF}{MTTF + MTTR}$$

Esistono dei grandi numeri per la $MTBF$, ovvero la Mean Time Between Failure. Ma in realtà NON è realistico avere un dispositivo che funzioni per 40 anni. Ma se tutti i produttori adottassero lo stesso procedimento, allora le quantità sarebbero perlomeno confrontabili. Per vedere la durata realistica si dovrebbe procedere assumendo un numero elevato di componenti e fare la media aritmetica dei tempi di guasto. Valutazione dell'Affidabilità di un Sistema a partire dall'Affidabilità dei singoli componenti. Se tutti questi attuassero questa procedura alla stessa maniera, i valori sarebbero però confrontabili. Se volessimo una valutazione realistica dell'Affidabilità di un Sistema, dovremmo ovviamente partire da Affidabilità Realistiche dei singoli componenti. Per alcuni componenti si sanno i valori realistici. Nella realtà non abbiamo il ragionevole tempo di osservare N_0, con $(N_0 \to +\infty) \uparrow$. Popolazione iniziale eccessivamente elevata.

Tipicamente il Failure Rate di un sistema (funzione che varia nel tempo, $h(t)$), è costante $\Longleftarrow$ se la distribuzione della v.a. X è esponenziale. Avremo un andamento che segue abbastanza bene il profilo di una vasca da bagno.

La curva suggerisce l'esistenza di tre fasi: *Mortalità Infantile*, *Useful Life* e *Wearout Phase* Nella fase di Mortalità Infantile, come suggerisce lo stesso nome, i Failure sono dovuti a prolbemi ENDOGENI. Eventuali progettazioni non fatte bene. All'inizio abbiamo un Failure Rate abbastanza elevato. Poi abbiamo la vita utile e poi di nuovo dei valori alti nella wearout phase (dovuti tipicamente all'usura). Se la prima fase è superata, possiamo sempre andare incontro a failure, ma nella Useful Life essi sono dovuti a problemi ESOGENI (Cause di stress per un dispositivo).

Nella Useful Life il Failure Rate del dispositivo è costante; questo fatto corrisponde ad un'ipotesi. Durante la Vita Utile sono le condizioni di Stress che provocano malfunzionamenti. Di tanto in tanto lo stress raggiunge dei picchi, oltrepassando lo Stress Max che il dispositivo può tollerare. Situazioni di stress (particolari) $\Longrightarrow$ il dispositivo va incontro a malfunzionamenti. Se λ è il Failure Rate, è auspicabile pensare che lo stress sia un processo di POISSON (a velocità λ). $N_t := |picchi\ in\ [0,t]|$. Se il processo dei picchi è di POISSON abbiamo:

$$[\Pr\{N(t) = \underline{k}\} = \frac{(\lambda t)^k}{k!}\, \mathrm{e}^{-\lambda t}]$$

POISSON, con $[k \geq 0]$. Quando il failure rate è costante $\iff$ la v.a. X è distribuita esponenzialmente con parametro λ. Si consideri la probabilità che $\Pr\{X > t\}$, ovvero l'AFFIDABILITÀ. La probabilità che $X > t$, che il tempo di vita del dispositivo sia maggiore di t, è UGUALE alla probabilità che fino a t non ci siano stati picchi:

$$\implies (\dots) = \Pr\{N_t = 0\} = \mathrm{e}^{-\lambda t}$$

(la distribuzione è esponenziale). Se il failure rate è costante, la distribuzione del lifetime è esponenziale. Superata la fase Useful Life abbiamo la fase del LOGORIO, quando il dispsositivo comincia ad usurarsi. Se NON ci fosse quest'usura, la durata media dell'Intervallo di Useful Life sarebbe realisticamente molto grande.

{MTTF, =MTBF} $\leftarrow$ quando si valutano queste quantità si suppone sempre di stare in Useful Life. Valori grandi.

3.4 RBD (Reliability Block Diagram)

Reliability Block Diagram, ovvero diagrammi a blocchi dell'Affidabilità. Strumento teorico per valutare l'Affidabilità di un sistema a partire dall'Affidabilità degli elementi di cui esso si compone. Per tirare fuori gli RBD dobbiamo partizionare un sistema in elementi con specifici task. Parliamo di un sistema con *STRUTTURA SEMPLICE* per riferirci ad un sistema che può essere modellato mediante un RBD costituito da blocchi combinati secondo strutture {SERIE, PARALLELO, k-out-of-n ("k su n")}, ed in cui i vari blocchi sono indipendenti l'uno dall'altro. I blocchi modellano l'Affidabilità dei componenti. L'Affidabilità è: $\Pr\{X > t\}$. Blocchi indipendenti $\implies$ elementi indipendenti $\implies$ tempi di guasto v.a. indipendenti. Il tempo di guasto in un certo blocco non va ad influire sui tempi di guasto degli altri, e dualmente, non è influenzato dai tempi di guasto di un altro blocco.

3.4.1 Struttura SERIE

Due blocchi in serie. Con il rettangolo tratteggiato stiamo considerando il sistema costituito dai blocchi in serie:

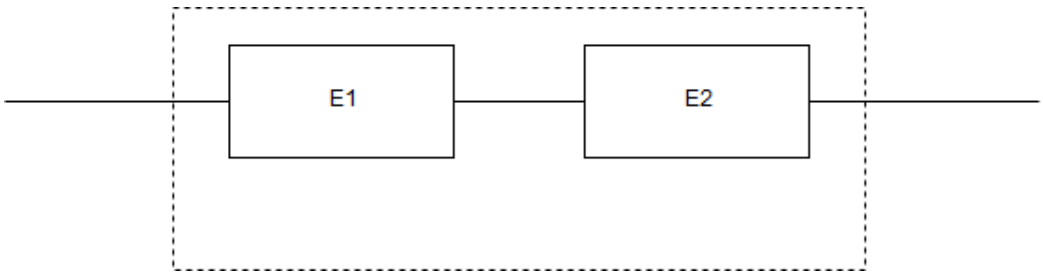


Figura 3.1: Struttura Serie

Siano: $\{R_1(t),\ R_2(t)\}$ le rispettive due Affidabilità degli elementi $\{E_1, E_2\}$, due elementi di un sistema, ai quali corrispondono due blocchi di Affidabilità in serie. L'intero sottosistema funziona bene se i due sotto-elementi funzionano. Ogni elemento della struttura serie deve funzionare. *PATH* che porta dall'Ingresso all'Uscita. Se un elemento NON funziona è come se nel mezzo ci siano circuiti aperti. Se un elemento è guasto, è considerato come *CIRCUITO*

APERTO. Sia X il time to failure dell'intero sottosistema, X_1 il lifetime associato ad E_1 ed X_2, diametralmente, sia il lifetime associato ad E_2. Abbiamo:

$$R_{series}(t) = \Pr\{X > t\} = \Pr\{X_1 > t,\ X_2 > t\} = \Pr\{X_1 > t\}\Pr\{X_2 > t\} = \underline{R_1(t)R_2(t)}$$

ove si è sfruttato l'indipendenza dei blocchi per scrivere la probabilità dell'evento INTERSECT $\iff$ la probabilità congiunta è il PRODOTTO delle probabilità. Il termine sottolineato è quindi il prodotto delle due Affidabilità. Possiamo naturalmente generalizzare, scrivendo che:

Theorem 13. ***Affidabilità sottosistema SERIE***
L'Affidabilità di un sottosistema serie costiuito da n componenti è:

$$R_{series}(t) := \prod_{i=1}^{n} R_i(t)$$

Questa è anche detta Product law of Reliability. Ad esempio se abbiamo 5 componenti in serie (elementi) tali per cui:

$$R_i(t) = R = 0.970\ \forall i = 1, 2, \ldots, 5 \implies R_{series} = (0.97)^5 = 0.859$$

Se ne avessimo 10 con questa stessa Affidabilità dei singoli componenti, avremmo invece $R_{series} = (0.97)^{10} = 0.738$. Si supponga ora di avere un sistema costituito da: {HARD DISK, PROCESSORE, MEMORIA}, rispettivamente mappati nell'RBD come: $\{E_1, E_2, E_3\}$. Sia $X \sim EXP(f)$, dove $f = f_p + f_{mem} + f_{HD}$. Allora l'Affidabilità del computer è: $[R(t) = \mathrm{e}^{-ft},\ f = f_p + f_{mem} + f_{HD}]$. Procedendo con gli RBD, troviamo:

$$\begin{cases} R_{proc}(t) = \mathrm{e}^{-f_p t} \\ R_{mem}(t) = \mathrm{e}^{-f_{mem} t} \\ R_{HD}(t) = \mathrm{e}^{-f_{HD} t} \end{cases}$$

Facendo il prodotto abbiamo:

$$R_{series}(t) = R_{proc}(t)R_{mem}(t)R_{HD}(t) = \mathrm{e}^{-(f_p + f_{mem} + f_{HD})t} = \mathrm{e}^{-f_p t}\,\mathrm{e}^{-f_{mem} t}\,\mathrm{e}^{-f_{HD} t}$$

I vari elementi combinati in serie devono quindi funzionare, e DEVONO funzionare molto bene per un'alta affidabilità. Quindi, se consideriamo due elementi in serie abbiamo che: $R(t) = R_1(t)R_2(t)$. Se consideriamo:

$$\begin{cases} R'(t) = R_1'(t)R_2(t) + R_1(t)R_2'(t) \\ \left[\begin{aligned} h_{series}(t) &= \frac{-R'(t)}{R(t)} = -\frac{R_1'(t)R_2(t)}{(R(t) = R_1(t)R_2(t))} - \frac{R_1(t)R_2'(t)}{R_1(t)R_2(t)} = \\ &= -\frac{R_1'(t)}{R_1(t)} - \frac{R_2'(t)}{R_2(t)} = h_1(t) + h_2(t) = \sum_{i=1}^{2} h_i(t) \end{aligned} \right. \end{cases}$$

Generalizzando abbiamo che:

Theorem 14. ***Failure Rate sottosistema SERIE***
Il Failure Rate di un sottosistema serie costiuito da n componenti è:

$$h_{series}(t) = \sum_{i=1}^{n} h_i(t)$$

in generale considerando n elementi. Il failure rate totale corrispondente è la sommatoria dei vari failure rati associati ai singoli elementi. Abbiamo: {Prodotto delle $R_i(t)$, sommatoria degli $h_i(t)$}.

3.4.2 Struttura PARALLELO

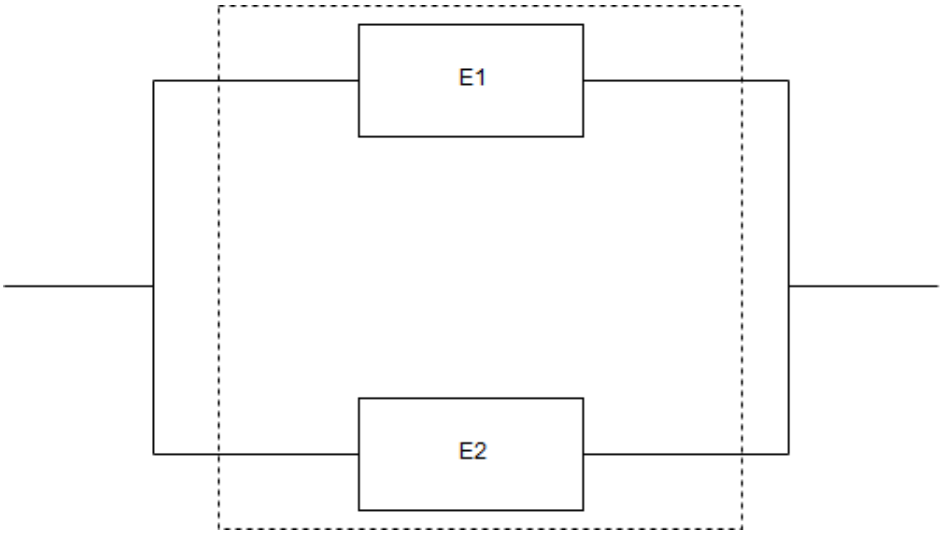


Figura 3.2: Struttura Parallelo

Siano sempre: $\{R_1(t) \rightarrow E_1(t),\ R_2(t) \rightarrow E_2(t)\}$. Valutiamo l'Affidabilità del sistema parallelo. Il sistema funziona se ALMENO uno degli elementi funziona.

$$R_{parallel}(t) = \Pr\{X > t\} = 1 - \Pr\{X \leq t\} = 1 - \Pr\{X_1 \leq t,\ X_2 \leq t\} = (\dots)$$

Dato che parliamo di blocchi indipendenti abbiamo: $\implies \Pr\{X > t\} = 1 - \Pr\{X_1 \leq t\} \Pr\{X_2 \leq t\}$. Quindi:

$$(\dots) = 1 - (1 - \underline{(\Pr\{X_1 > t\} = R_1(t))})(1 - \underline{(\Pr\{X_2 > t\} = R_2(t))}) =$$

$$= 1 - \prod_{i=1}^{2} (UNRELIABILITY)_i$$

Ove i termini sottolineati sono le due affidabilità, $R_1(t), R_2(t)$. Si può notare una definizione alternativa, eseguendo i prodotti opportuni:

$$(\dots) = \underline{[\Pr\{X_2 > t\} + \Pr\{X_1 > t\} - \Pr\{X_1 > t\} \Pr\{X_2 > t\}]}$$

Formalizziamo:

Theorem 15. ***Affidabilità sottosistema PARALLELO***
L'Affidabilità di un sottosistema parallelo costiuito da n componenti è:

$$R_{parallel}(t) := 1 - \prod_{i=1}^{n} (1 - R_i(t))$$

Si potrebbe in realtà fornire una definizione alternativa sfruttando il *principio di inclusione-esclusione*, od il teorema della probabilità totale.

Esempio con singole affidabilità uguali a quelle del precedente esempio: $R_{parallel}(t) = 1 - (1-0.97)^2 = 0.9991$, con due componenti. Mettendo 5 componenti in parallelo invece troviamo: $R_{parallel}(t) = 1 - (1-0.97)^5 = 0.9999999757$, ovvero un'altissima affidabilità. RIDONDANZA. Elementi RIDONDANTI. Struttura RIDONDANTE. I blocchi in tal caso sono in parallelo.

Per quanto concerne il failure rate, supponiamo di avere due dispositivi in parallelo, con parametri indipendenti tra di loro. L'Affidabilità complessiva è:

$$R(t) = R_2(t) + R_1(t) - R_1(t)R_2(t)$$

Calcoliamo quindi il failure rate, omettendo la dipendenza funzionale temporale onde alleggerire la notazione $\iff R_i := R_i(t),\ i = 1, 2$:

$$h_{parallel}(t) = -\frac{R'(t)}{R(t)} =$$

$$= -\frac{R_2'}{R_2 + R_1 - R_1R_2} - \frac{R_1'}{R_2 + R_1 - R_1R_2} + \frac{R_1'R_2}{R_2 + R_1 - R_1R_2} + \frac{R_1R_2'}{R_2 + R_1 - R_1R_2} =$$

$$= -\frac{R_1'}{R_2 + R_1 - R_1R_2}[1 - R_2] - \frac{R_2'}{R_2 + R_1 - R_1R_2}[1 - R_1]$$

Di per sé non ha un particolare significato esplicito. Se ci restringiamo al caso in cui abbiamo dispositivi identici, con quindi la stessa affidabilità ($\iff R_1 = R_2 := R$), otteniamo invece:

$$h_p := h_{parallel}(t) = -\frac{R'}{2R - R^2} - \frac{R'}{2R - R^2} = -2\frac{R'(1-R)}{R(2-R)} = -2h\frac{(1-R)}{2-R}$$

Ancora una volta, sarebbe possibile formalizzare il tutto sfruttando il teorema della probabilità totale.

3.4.3 Struttura k-out-of-n

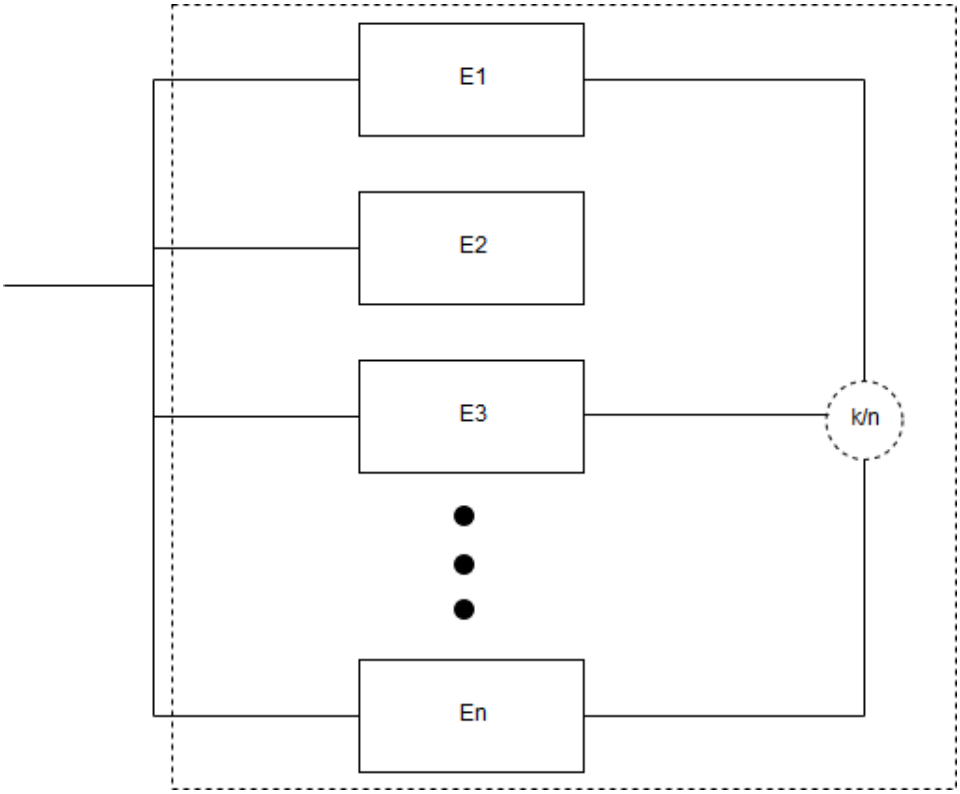


Figura 3.3: Struttura Parallelo

Sia l'Affidabilità $= (\cdot(t))$, ovvero stiamo sempre pensando ad un certo tempo. L'intero sistema funziona solo se ALMENO k su n elementi funzionano. Sistema UP quando ci sono k server funzionanti, ad esempio. Supponiamo per semplicità (wlog) che le Affidabilità dei vari elementi siano uguali: $[R_i(t) = R_e(t)]\ \forall i = 1, 2, \ldots, n$. Abbiamo:

$$R_{k/n}(t) = \Pr\{\text{"}almeno\ k\ elementi\ sopravvivono\ fino\ al\ tempo\ t\text{"}\} = \Pr\{X > t\} =$$

$$= \Pr\{\cup_{i=k}^{n}[\text{"}i\ elementi\ sopravvivono\ fino\ al\ tempo\ t,\ exactly\text{"}]\} = (\ldots)$$

Ma gli eventi in gioco sono DISGIUNTI! $\Longrightarrow$

$$(\ldots) = \sum_{i=k}^{n} \Pr\{\text{"}exactly\ i\ elements\ survive\ until\ time\ t\}$$

Questa non è nient'altro che una sequenza di n trial di Bernoulli indipendenti, quindi una DISTRIBUZIONE BINOMIALE. Definiamo quindi:

Definition 25. ***Affidabilità sottosistema k-out-of-n***
L'Affidabilità di un sottosistema k-out-of-n costiuito da n componenti è:

$$R_{k/n}(t) := \Pr\{X > t\} = \sum_{i=k}^{n} \binom{n}{i} R_e(t)^i [1 - R_e(t)]^{n-i}$$

Bernoulli Trials. Ne si considerino tutte le $\binom{n}{i}$ combinazioni possibili di configurazioni ove almeno k sono funzionanti.

Workstation and File Servers (Exercise)

Consider a network service based on a system that consists of n workstations and m file servers. The network connecting these devices is assumed to be fault-free. The system is considered to be operational so long as at least k workstations and l file servers are operational. Let $R_w(t)$ denote the reliability of a single workstation and Rf(t) the reliability of a single file server. Assuming that all devices fail independently of each other, evaluate system reliability.

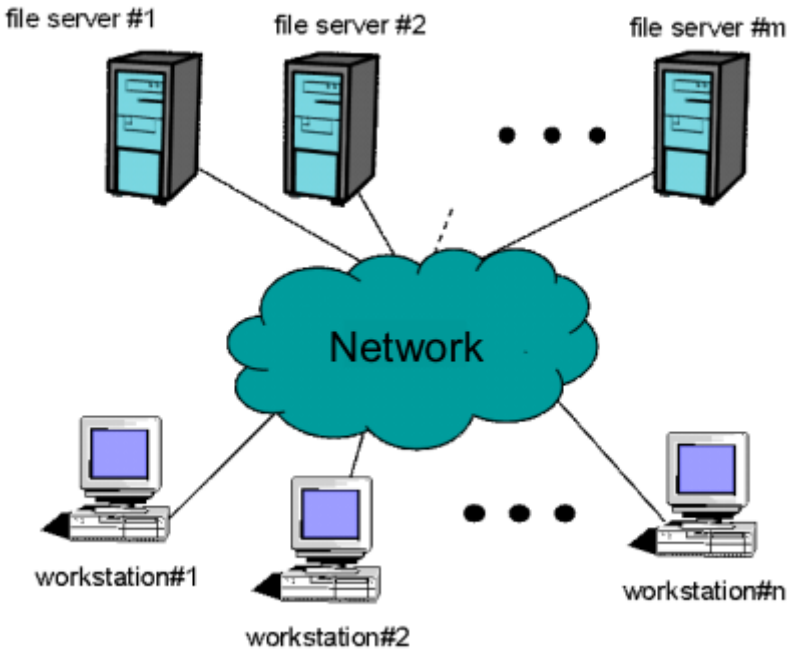


Figura 3.4: Workstations & File Servers

Si supponga di avere n workstation ed m File Server. La rete che connette questi dispositivi è ad Affidabilità Unitaria (*Fault-Free*). Il sistema è considerato essere operativo quando almeno k workstation ed l file server sono operativi. I due macrosistemi sono ovviamente collegati in serie. Abbiamo il seguente RBD:

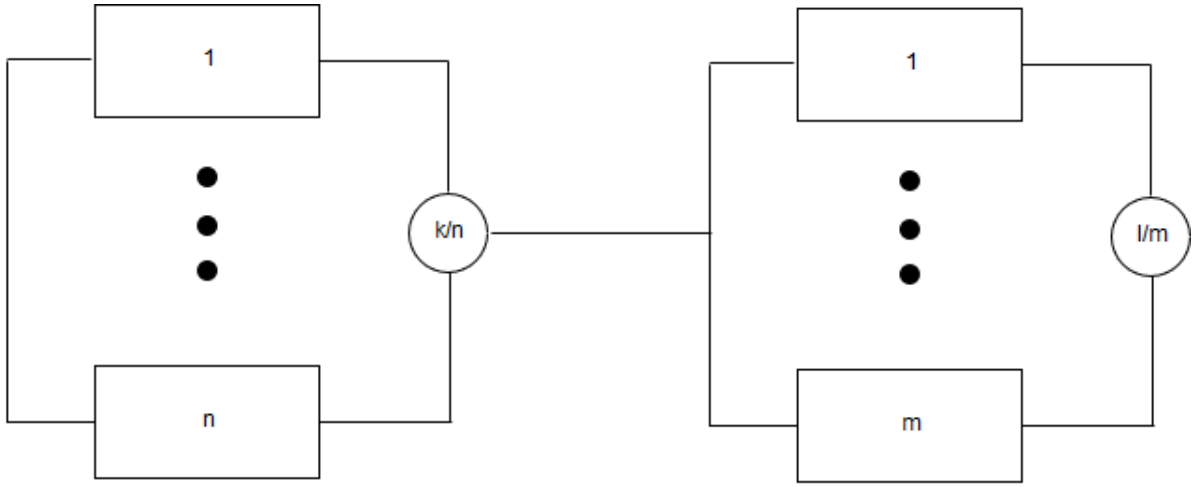


Figura 3.5: Workstation & File Servers RBD

Dunque risolviamo:

$$[R(t) = (\sum_{i=k}^{n} \binom{n}{i} R_w(t)^i [1 - R_w(t)]^{n-i})(\sum_{i=l}^{m} \binom{m}{i} R_f(t)^i [1 - R_f(t)]^{m-i})]$$

3.4.4 Network Lecce-Torino (Exercise)

Consider the network in the figure below:

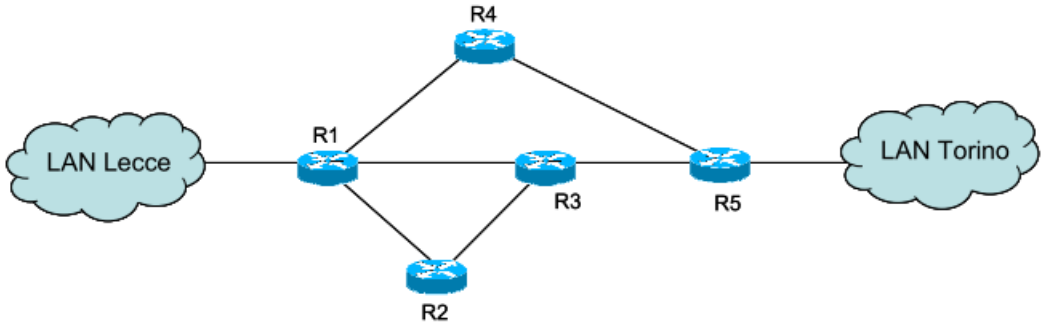


Figura 3.6: Network Lecce-Torino

Let $R_L(t)$ denote the reliability of links and assume that routers are fault-free. Evaluate the reliability of the path from Lecce to Torino. Possible routes for packets:

- R1 - R4 - R5;
- R1 - R3 - R5;

- R1 - R2 - R3 - R5;

Risoluzione:
Network Lecce-Torino. $R_L(t)$ rappresenta l'affidabilità dei Link. Router fault-tree. Possibili rotte: $\{R_1 - R_4 - R_5,\ R_1 - R_3 - R_5,\ R_1 - R_2 - R_3 - R_5\}$. Si enuncino i casi limite della struttura k-out-of-n:

$$\begin{cases} R_{1/n} = R_{parallel}(t) \\ R_{n/n} = R_{series}(t) \end{cases}$$

Ipotesi fault-free. Router ridondati. Affidabilità talmente elevata da ritenersi unitaria, quindi fault-free. Si consideri quindi solo l'Affidabilità dei Link. Sistema semplice modellabile con blocchi in serie, in parallelo o k/n. I link avranno in realtà diverse affidabilità a seconda della lunghezza etc. Modelliamo questo sistema reale con un RBD, andando a considerare l'Affidabilità dei singoli link. Non modelliamo il link d'ingresso, ma volendo potremmo anche associargli un RB apposito (*Reliability Block*). Facciamo un'IPOTESI IMPORTANTE: [Blocchi indipendenti]. Modello RBD che modella l'Affidabilità di questa rete. L'RBD è il seguente:

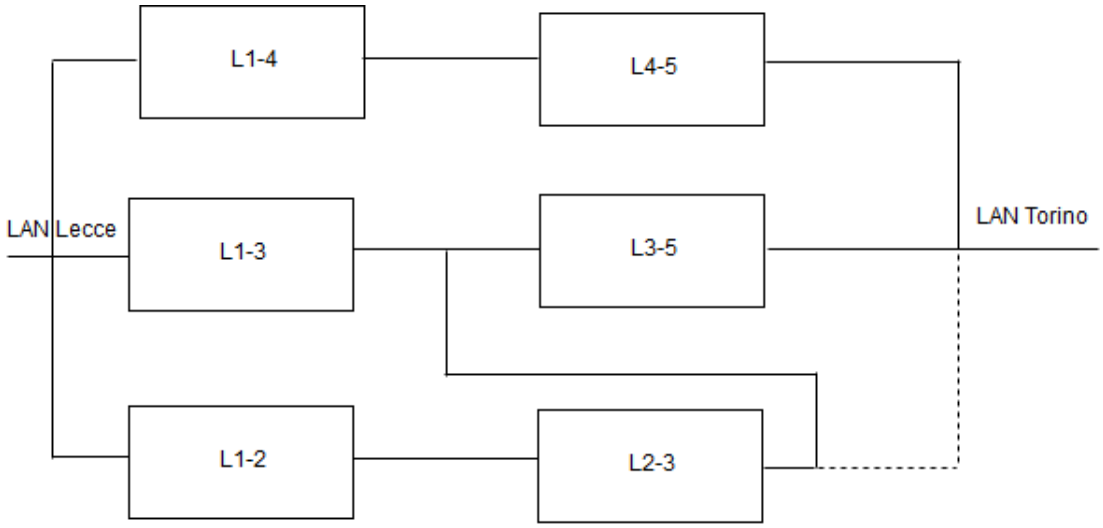


Figura 3.7: Network Lecce-Torino RBD

Sistema a struttura semplice, ovvero modellabile con blocchi di affidabilità combinati secondo strutture serie, parallelo, o k/n. Poniamo: $R_L(t) := (R \neq constant) = \cdot(t)$. Stiamo considerando la stessa affidabilità per tutti i link. Se avessimo blocchi ripetuti, essi non saranno assolutamente indipendenti! Rappresentiamo lo stesso elemento. Concetto di riduzione del modello iniziale in un modello a struttura semplice.

$$R_{collegamento}(t) = 1 - (1 - R^2)[1 - R[1 - (1 - R)(1 - R^2)]]$$

Approccio di calcolo ricorsivo Top-Down. Elementi relativi ai blocchi. Ci sono delle tabelle con dei valori caratteristici.

Network Lecce-Torino with Key-Item Method

Consideriamo un esercizio con diagrammi con struttura NON semplice. Sempre Network Lecce-Torino. Reti fisiche. Router IP. Reti fisiche come dei link. Reti fisiche collegati da router IP. Si denoti con $R_L(t)$ l'Affidabilità dei Link. Due router IP collegati alla medesima rete fisica sono detti ADIACENTI. Possibili rotte: $\{R_1 - R_3 - R_4,\ R_1 - R_3 - R_2 - R_4,\ R_1 - R_2 - R_4,\ R_1 - R_2 - R_3 - R_4\}$. Ci sono dei blocchi ripetuti fondamentalmente.

Consider the network in the figure below.

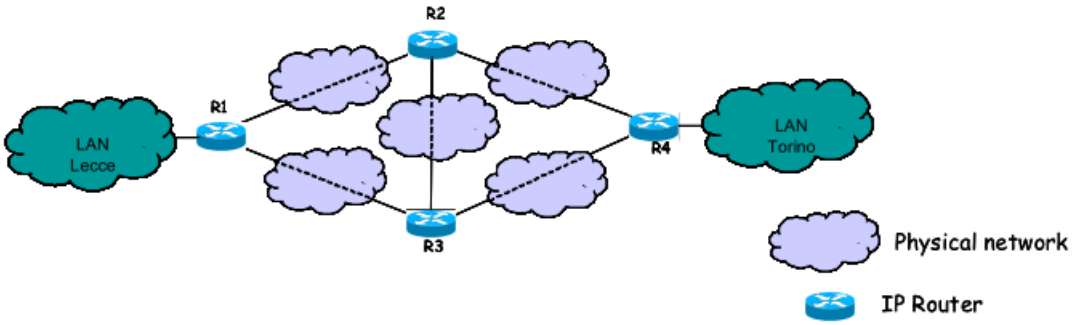


Figura 3.8: Network Lecce-Torino #Key-Item

Let $R_L(t)$ denote the reliability of links and assume that routers are fault-free. Evaluate the reliability of the path from Lecce to Torino. Possible routes for packets:

- R1-R3-R4;
- R1-R3-R2-R4;
- R1-R2-R4;
- R1-R2-R3-R4.

Risoluzione:

Consideriamo il relativo RBD:

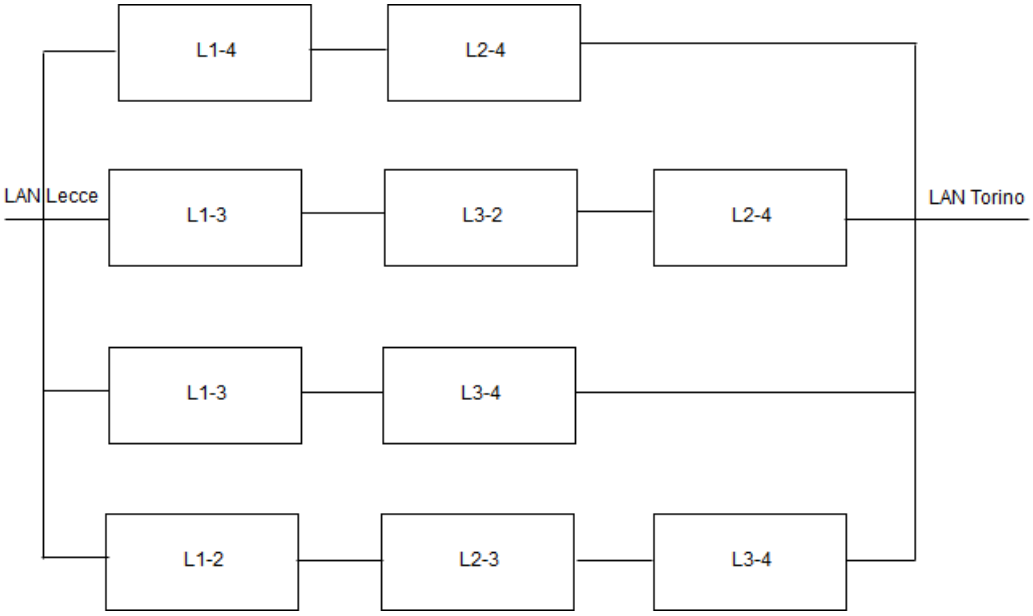


Figura 3.9: Network Lecce-Torino with Key-Item Method RBD

Il sistema è riducibile. L'RBD risultante potrebbe essere rappresentato anche nel seguente modo:

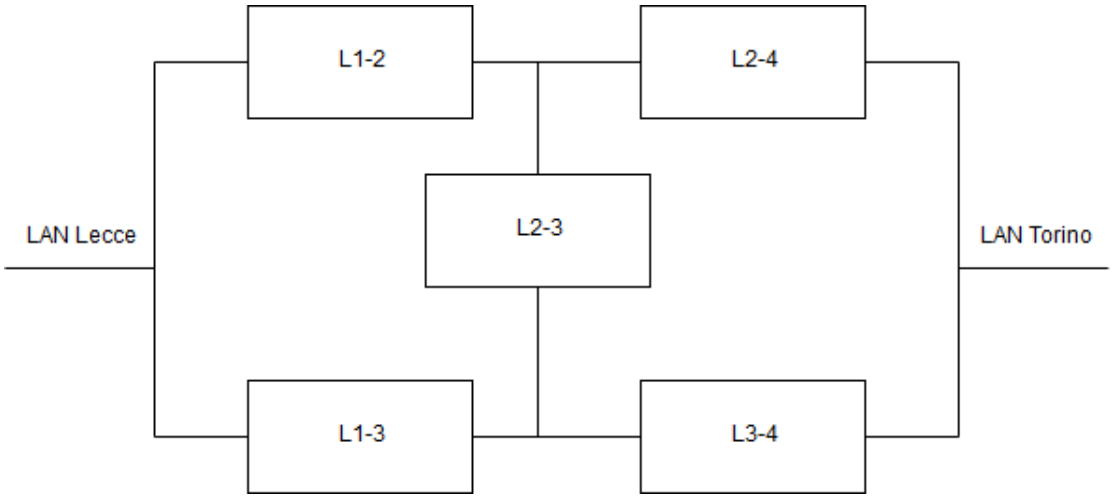


Figura 3.10: Network Lecce-Torino with Key-Item Method Reduced RBD

Sistema NON con struttura semplice, sebbene abbiamo dei blocchi indipendenti. Si sfrutti il *Metodo Key-Item*, ovvero il Key-Item Method, in letteratura. Sostanzialmente si basa sul teorema delle probabilità totali. Condizionare rispetto allo stato di funzionamento dell'elemento chiave in gioco. Consideriamo quindi un elemento chiave $E_i(t)$ ($\underline{E_i \text{ is the key element}}$).

$$R_S(t) = \Pr\{\underline{S \textit{ sia } UP \textit{ in } [0,t]}\} =$$

$$= \Pr\{S \textit{ sia } UP \textit{ in } [0,t] \mid E_i \textit{ sia } UP \textit{ in } [0,t]\}(\Pr\{E_i \textit{ sia } UP \textit{ in } [0,t]\} = R_i(t)) +$$

$$+ \Pr\{S \textit{ sia } UP \textit{ in } [0,t] \mid E_i \textit{ sia } DOWN \textit{ in } [0,t]\}(\Pr\{E_i \textit{ sia } DOWN \textit{ in } [0,t]\} = (1 - R_i(t)))$$

$\Pr\{X > t\}$ è la probabilità il componente sopravviva sino a t. Notiamo che:

$$\begin{cases} [\Pr\{E_i \textit{ sia } UP \textit{ in } [0,t]\} = R_i(t)] \\ [\Pr\{E_i \textit{ sia } DOWN \textit{ in } [0,t]\} = (1 - R_i(t))] \end{cases}$$

Se funziona sempre in $[0,t] \implies E_i$ diviene un CORTOCIRCUITO, diversamente nel secondo caso otteniamo un CIRCUITO APERTO. Potrebbe talvolta essere necessario applicare iterativamente il teorema delle probabilità totali sul nuovo scenario ottenuto.

$$\begin{cases} \Pr\{S \textit{ sia } UP \textit{ in } [0,t] \mid E_i \textit{ sia } UP \textit{ in } [0,t]\} \rightarrow E_i \;\; CORTOCIRCUITO \\ \Pr\{S \textit{ sia } UP \textit{ in } [0,t] \mid E_i \textit{ sia } DOWN \textit{ in } [0,t]\} \rightarrow E_i \;\; CIRCUITO \; APERTO \end{cases}$$

Ove le notazioni a destra si riferiscono al comportamento da schematizzare nel conseguente RBD.

- E_i UP: $\rightarrow R_a(t)$;
- E_i DOWN: $\rightarrow R_b(t)$

$\implies [R_s(t) = R_a(t)R_i(t) + R_b(t)(1 - R_i(t))]$. Si considerino due casi quindi, entrambi da risolvere con la tecnica degli RBD, e laddove necessario si applichi ricorsivamente il TPT.

L_{2-3} è nel nostro caso un candidato E_i:

- a) link 2-3 UP in $[0,t]$;
- b) link 2-3 DOWN in $[0,t]$

Dobbiamo prima considerare il caso a, poi il caso b. Se lo CORTOCIRCUITO, il risultante RBD sarà la serie di due blocchi parallelo: $R_s = [1 - (1 - R)^2]^2$. Nel secondo caso invece otteniamo che dobbiamo rendere l'elemento chiave un CIRCUITO APERTO $\implies$ conseguente RBD semplice costituito dal parallelo di due serie $\implies R_s = [1 - (1 - R^2)^2]$. Quindi alla fine l'Affidabilità della PATH da Lecce a Torino è:

$$R_{path}(t) = R_a(t)R_{L_{2-3}}(t) + R_b(t)[1 - R_{L_{2-3}}(t)]$$

Nel caso il link chiave fosse unidirezionale, si potrebbe procedere in un solo verso. In tal caso avremmo sempre una struttura non semplice, ma in tal caso sarebbe consigliabile scegliere come elemento chiave L_{3-4}: Adesso abbiamo quindi considerato cosa significa Affidabilità di un collegamento.

3.4.5 SYSTEM RELIABILITY

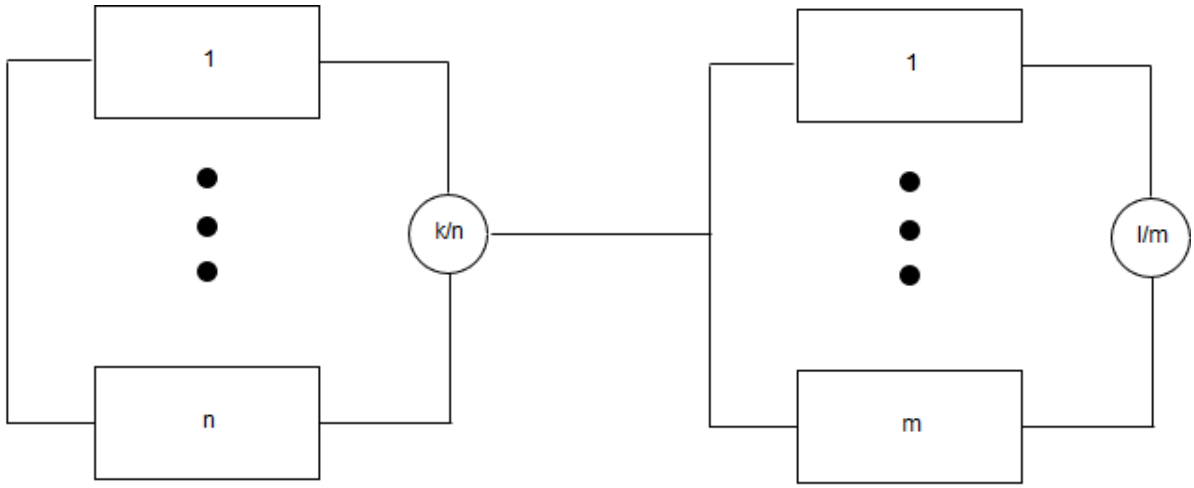


Figura 3.11: Workstation & File Servers RBD

Supponiamo che dalle distribuzioni empiriche risulti che i TTF (time to failure) siano distribuiti esponenzialmente, rispettivamente:

$$\begin{cases} W_S \sim EXP(\lambda_W) \\ F_S \sim EXP(\lambda_f) \end{cases}$$

$\{\lambda_W, \lambda_f\}$. Determiniamo l'Affidabilità del sistema ed il MTTF. Caso semplice: $n = 2 \wedge m = 1 \ \wedge \ k = l = 1$. Riduzione a:

$$R_i(t) = R_f(t)[1 - (1 - R_w(t))^2] = \mathrm{e}^{-\lambda_f t}[1 - (1 - \mathrm{e}^{-\lambda_W t})^2] = (\dots)$$

dove abbiamo: $\{R_f(t) = \mathrm{e}^{-\lambda_f t}, \ R_W(t) = \mathrm{e}^{-\lambda_W t}\}$. Quindi:

$$(\dots) = \mathrm{e}^{-\lambda_f t}[1 - (1 + \mathrm{e}^{-2\lambda_W t} - 2\,\mathrm{e}^{-\lambda_W t})] = \mathrm{e}^{-\lambda_f t}[2\,\mathrm{e}^{-\lambda_W t} - \mathrm{e}^{-2\lambda_W t}] =$$
$$= [2\,\mathrm{e}^{-(\lambda_f + \lambda_W)t} - \mathrm{e}^{-(\lambda_f + 2\lambda_W)t}]$$

Questa è l'Affidabilità del sistema. Adesso abbiamo esplicitato la distribuzione di probabilità dell'Affidabilità. Il failure rate è: $\rightarrow$

$$[h(t) = \frac{-R'(t)}{R(t)}]$$

Inoltre,

$$MTTF = \mathbb{E}[X] = \int_0^\infty R(t)dt = \int_0^\infty (2\,\mathrm{e}^{-(\lambda_f + \lambda_W)t} - \mathrm{e}^{-(\lambda_f + 2\lambda_W)t})dt =$$
$$= 2\int_0^\infty \mathrm{e}^{-(\lambda_f + \lambda_W)t}dt - \int_0^\infty \mathrm{e}^{-(\lambda_f + 2\lambda_W)t}dt = \frac{2}{\lambda_f + \lambda_W} - \frac{1}{\lambda_f + 2\lambda_W}$$

Quanto detto per l'Affidabilità può anche esser esteso per la Disponibilità con gli ABD (*Availability Block Diagram*), a patto che ttf e ttr siano v.a. indipendenti tra di loro $\implies \nexists$ Single Repair Facility (SRF) $\implies \forall$ elemento $\exists!$ Repair Facility. Altrimenti i vari ttr sarebbero dipendenti mutuamente. Se invece il sistema ha abbastanza risorse per le riparazioni degli elementi $\implies$ ttr v.a. indipendenti.

3.4.6 ABD (Availability Block Diagram)

Abbiamo:

$$\begin{cases} A_s(t) = \prod_{i=1}^{n} A_i(t), \ serie \\ A_p(t) = 1 - \prod_{i=1}^{n} (1 - A_i(t)), \ parallelo \end{cases}$$

Vale per Disponibilità istantanea, disponibilità in un intervallo e disponibilità a regime (che sarebbe il limite per $t \to +\infty$ della disponibilità in un intervallo). Supponendo $n = 2$, $m = 2$, $l = k = 1$, si calcoli la Disponibilità del Sistema, supponendo che il $MTTF$ di una workstation sia $MTTF_W$ e quella del File Server sia $MTTF_f$. Analogamente per $MTTR_W$, $MTTR_f$. Si calcoli la disponibilità a regime, sfruttando i diagrammi a blocchi della disponibilità.

La STEADY-STATE AVAILABILITY è:

$$A_{SS} = A_f[1 - (1 - A_w)^2]$$

ma $A_f, A_w =? \implies$

$$\begin{cases} A_f = \dfrac{MTTF_f}{MTTF_f + MTTR_R} \\ A_W = \dfrac{MTTF_W}{MTTF_W + MTTR_W} \end{cases}$$

Il tempo di Restore include anche il tempo di rilevazione del malfunzionamento e di sopralluogo, oltre a quello ovviamente dell'effettiva riparazione.

3.5 CMTC e Affidabilità

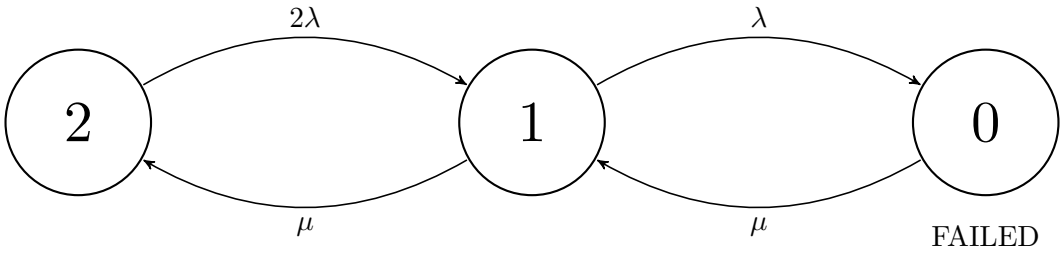
CMTC, Catene di Markov a Tempo Continuo per il calcolo dell'Affidabilità. Catene di Markov con *Stati ASSORBENTI*. Il sistema reale potrebbe ovviamente essere più complesso. Un servizio di Rete è basato su un sistema ridondante parallelo con 2 dispositivi. Il sistema è FALLITO quando entrambi i dispositivi sono guasti. Assumiamo:

$$\begin{cases} TTF \sim EXP(\lambda) \\ TTR \sim EXP(\mu) \end{cases}$$

$MTTF =? \ R(t) =?$. Interazioni più complesse che NON ci consentono di procedere con i normali RBD. Qui non solo i ttr NON sono indipendenti $\iff$ Single Repair Facility, ma è proprio l'interazione NON descrivibile mediante RBD.

3.5.1 Stati Assorbenti

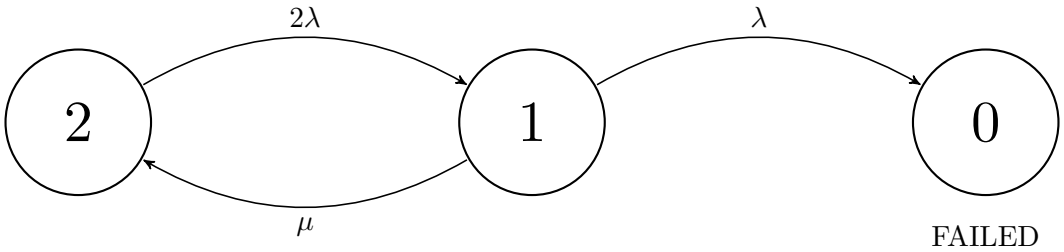
PROCESSO STOCASTICO $N(t) = |\{dispositivi\ funzionanti\ al\ tempo\ t\}|$. Devo calcolare l'Affidabilità del Sistema di servizio (dell'intero servizio). Questa servirà poi per il calcolo dell'$MTTF = \mathbb{E}[X]$. Voglio studiare questo sistema con un processo stocastico, il quale è una CMTC, il cui DTT è il seguente:



I ttf sono tali per cui $ttf \sim EXP(\lambda)$, mentre il ttr è tale per cui $ttr \sim EXP(\mu)$. Abbiamo $S = \{0, 1, 2\}$ come spazio degli stati. Lo stato 0 rapresenta il fatto che non vi sono dispositivi funzionanti (stato FAILED). Ma qui contempliamo anche la possibilità che da FAILURE si torni nello stato UP. Possiamo anche considerare la DISPONIBILITÀ.

$$\{\pi_2,\ \pi_1,\ \pi_0\} \implies A = \pi_2 + \pi_1$$

(quando il sistema sarà nello stato di NORMAL). Ma dobbiamo ora calcolare l'Affidabilità e l'MTTF. Computational Model. Modello di calcolo. Modello che si utilizza per calcolare una certa quantità. {Affidabilità, MTTF}. Modello di calcolo a partire dal modello di disponibilità (AVAILABILITY MODEL). La DTT diventa, a fronte di un opportuno detach del ramo di transizione che porta dallo stato di failure a normal:



Sostanzialmente per quello che ci serve calcolare, partiamo dall'AM e facciamo un detach del ramo di transizione 0-1. $X,\ \Pr\{X > t\}$ è la nostra Affidabilità. Il sistema entra in FAIL quando si ha un ingresso nello stato 0 (Transizione 1-0), ovvero nello stato ASSORBENTE. È finito in tal caso il tempo di vita. Supponiamo che il sistema EVOLVA con due dispositivi funzionanti (evolva a partire dallo stato 2) $\implies$

$$\begin{cases} \pi_2(0) = 1 \\ \pi_1(0) = \pi_0(0) = 0 \end{cases}$$

Il ttf X corrisponde al *time-to-absorption* (il tempo per entrare nello stato assorbente) $\iff$ (TEMPO DI VITA = TEMPO DI ASSORBIMENTO). Cosicché $MTTF = MTTA$. Tutti gli altri stati sono stati TRANSITORI ($\nexists$ NON ESISTE distribuzione di regime). Ora mi servo di distribuzioni transitorie:

$$\underline{\Pr\{X > t\}} = 1 - \underline{\Pr\{X \leq t\}} = 1 - \underline{\pi_0(t)}$$

ove la probabilità $\pi_0(t)$ corrisponde alla probabilità che prima di t si sia entrati nello stato assorbente. Abbiamo $X = ttf = tta$. C'è un solo stato assorbente, ve ne sono due transitori e quindi $\implies \nexists \pi_i = \lim_{t\to+\infty} \pi(t)$.

$$[\frac{d\pi_i(t)}{dt} = \sum_{j\in S} q_{ji}\pi_j(t),\ \forall i \in S]$$

Potremmo utilizzare queste equazioni ed applicarle a questa catena. In particolare, agli stati $\{0, 1, 2\}$. Abbbiamo:

$$\begin{cases} \dfrac{d\pi_2(t)}{dt} = -2\lambda\pi_2(t) + \mu\pi_1(t) \\ \dfrac{d\pi_1(t)}{dt} = -(\lambda+\mu)\pi_1(t) + 2\lambda\pi_2(t) \\ \dfrac{d\pi_0(t)}{dt} = \pi_1(t)\lambda \end{cases}$$

In particolare si risolvano ovviamente con le opportuni condizioni di evoluzione iniziale. Si potrebbe risolvere tal sistema con le TL, ovvero con le trasformate di Laplace per trovare alla fine $\underline{\pi_0(t)}$. Quindi:

$$\begin{cases} s\pi_2^\star(s) - \pi_2(0) = -2\lambda\pi_2^\star(s) + \mu\pi_1^\star(s) \\ s\pi_1^\star(s) - \pi_1(0) = -(\lambda+\mu)\pi_1^\star(s) + 2\lambda\pi_2^\star(s) \\ s\pi_0^\star(s) - \pi_0(0) = \lambda\pi_1^\star(s) \end{cases}$$

L'Affidabilità, una volta trovato $\pi_0(t)$, sarà quindi: $1 - \pi_0(t)$. Calcolata $R(t)$, il $\underline{MTTF} = \int_0^\infty R(t)dt$. Integrando opportunamente troviamo:

$$[\underline{MTTF} = \int_0^\infty R(t)dt = \frac{3}{2\lambda} + \frac{\mu}{2\lambda^2}] = MTTA$$

ovvero pari al tempo che la catena ci mette per entrare nello stato assorbente. Se ci avesse chiesto di calcolare solo l'MTTF, avremmo potuto procedere in altro modo: Definiamo una nuova quantità:

$$\begin{cases} L_i(t) := \displaystyle\int_0^t \pi_i(x)dx \\ [\pi_i(t) = \Pr\{X(t) = i\}] \end{cases}$$

ove l'espressione tra quadre rappresenta una distribuzione di probabilità. La prima equazione rappresenta invece il tempo medio trascorso nello stato i sino a t. Se consideriamo la finestra temporale $[0, t]$, $L_i(t)$ rappresenta il tempo in media in cui il processo si è trovato in i. Definiamo la variabile indicatrice $I_i(t)$ come:

$$I_i(t) := \begin{cases} 1,\ X(t) = i \\ 0,\ X(t) \neq i \end{cases}$$

Sostanzialmente essa è una VARIABILE INDICATRICE che è per l'appunto indicatrice del fatto che nell'istante t il processo si trovi in i o meno. Sfruttando il lemma della variabile indicatrice otteniamo:

$$L_i(t) = \int_0^t \Pr\{I_i(x) = 1\}dx = \int_0^t \mathbb{E}[I(x)]dx = \mathbb{E}[\int_0^t \underline{I(x)}dx]$$

ove la quantità sottolineata, $I(x) \in \{0,1\}\ \forall x$, è una funzione integranda che varia tra 1 e 0, discretamente. Se ne facciamo l'integrale otteniamo il tempo totale nel quale il processo si è trovato nello stato i sino a t. Consideriamo una singola realizzazione:

$$x_1 * 1 + x_2 * 2 = \underline{x_1 + 2x_2}$$

è il tempo trascorso dal processo nello stato i. Il tempo medio si ottiene mediando sull'insieme delle singole realizzazioni. Con l'operatore $\mathbb{E}[\cdot]$ davanti otteniamo proprio il tempo medio trascorso dal processo nello stato i. $L_i(t)$ rappresenta quindi questa quantità. Definita questa quantità deriveremo un sistema di equazioni differenziali che contempla invece quelle $L_i(t)$ e dato che abbiamo a che fare con Stati assorbenti, partizioneremo lo spazio degli stati in Stati assorbenti e Stati transitori.

Definition 26. ***TEMPO MEDIO DI ASSORBIMENTO***

$$\underline{L_i(\infty)} := \lim_{t \to +\infty} L_i(t)$$

Fino all'assorbimento $(t \to \infty) \uparrow$. Se gli stati sono transitori, ovviamente $L(\infty) < +\infty$. Invece per gli stati assorbenti è ragionevole ipotizzare che $L(\infty) = +\infty$. Poi sfruttando il seguente fatto: $L_2(\infty) + L_1(\infty) = MTTF$, troveremo gli stessi risultati fondamentalmente.

Determinare l'Affidabilità $R(t)$ ed il MTTF del sistema. Si può procedere con le trasformate di Laplace per determinare $\pi_0 \implies R(t) = 1 - \pi_0(t)$. Integrando $R(t)$ opportunamente troviamo $MTTF$. Se chiede soltanto di determinare il MTTF, c'è in realtà un altro modo. $L_i(t) = \int_0^t \pi_i(\tau)d\tau$, che rappresenta il tempo medio trascorso dal processo nello stato i durante la finestra temporale $[0,t]$. Sappiamo che $MTTF = MTTA$ (*Mean Time To Absorption*), ove termina il tempo di vita del sistema $\implies$ Condizione di sistema down = Ingresso della catena nello stato di guasto, malfunzionamento. $L_i(t)$ è una primitiva di $\pi_i(t)$. È proprio la funzione integrale, sostanzialmente. Possiamo quindi scrivere: $[\frac{dL_i(t)}{dt} = \pi_i(t)]$. Consideriamo le equazioni differenziali che legano le probabilità in TRANSITORIO con i tassi di transizione:

$$\frac{d\pi_i(t)}{dt} = \sum_{j \in S} q_{ji}\pi_j(t) \implies \int_0^t \frac{d\pi_i(\tau)}{dt}d\tau = \int_0^t \sum_{j \in S} q_{ji}\pi_j(\tau)d\tau = (\dots),\ \forall i \in S$$

ove abbiamo adeguatamente integrato da 0 a t. Quindi:

$$(\dots) = \pi_i(t) - \pi_i(0) = \sum_{j \in S} q_{ji}(\int_0^t \pi_j(\tau)d\tau = L_j(t)) \implies$$

$$\implies \frac{dL_i(t)}{dt} - \pi_i(0) = \sum_{j \in S} q_{ji}L_j(t),\ \forall i \in S$$

Valgono per una qualsiasi CMTC. A questo punto,

$$\sum_{j \in S} q_{ji}L_j(t) + \pi_i(0) = \frac{dL_i(t)}{dt},\ \forall i \in S$$

Dove riconducendoci in forma matriciale abbiamo: $\implies$

$$\frac{d\bar{L}(t)}{dt} = \bar{L}(t)\bar{Q} + \bar{\pi}(0)$$

Avendo compattato tutto in forma matriciale. Notiamo che: $\bar{L}(t) \in \mathbb{R}^{1 \times n}$. Come queste equazioni possono essere utilizzate per determinare il MTTF del sistema: Considero la CMTC

con stati ASSORBENTI. Si consideri la seguente partizione: $S = \underline{N} \cup \underline{A}$, dove S rappresenta lo spazio degli stati, N è il sottoinsieme degli stati transitori ed S il sottoinsieme degli stati assorbenti. Adesso si consideri uno stato transitorio. Per uno stato transitorio, consideriamo:

$$\lim_{t\to+\infty} L_i(t) = L_i(\infty) < +\infty$$

Se $\exists i \in N \iff N \neq \emptyset \implies L_i(\infty) < +\infty$ (Il limite converge). Invece per gli stati assorbenti, $L_i(\infty) = +\infty$. Il processo vagherà per gli stati transitori e ad un certo punto entrerà negli stati assorbenti. $L_i(t) \to L_i(\infty) < +\infty \implies \frac{dL_i(t)}{dt} \to 0$ $(t \to +\infty)$. Consideriamo quindi uno stato i transitorio:

$$(\lim_{t\to+\infty} \frac{dL_i(t)}{dt} = 0) = \lim_{t\to+\infty} [\sum_{j\in S} q_{ji} L_j(t)] + \pi_i(0) = 0$$

Abbiamo effettuato una riduzione da un sistema di equazioni differenziali ad un sistema di equazioni algebriche. Prima avevamo un sistema di equazioni differenziali. Adesso abbiamo un sistema di equazioni algebriche (lineari peraltro). Quindi:

$$0 = \sum_{j\in S} q_{ji} L_j(\infty) + \pi_i(0),\ \forall i \in N$$

In forma matriciale abbiamo:

$$\bar{L}_N(\infty)\bar{Q}_N = -\bar{\pi}_N(0)$$

ove il primo fattore del primo membro è un vettore riga delle $L_i(\infty)$ relativi gli stati transitori, MENTRE IL SECONDO FATTORE è la sottomatrice dei tassi di transizione relativi agli stati transitori. IDEM per il vettore riga delle condizioni iniziali. Abbiamo compattato tutte le equazioni relativi agli stati transitori. Abbiamo quindi adoperato una restrizione. In particolare, $\bar{Q}_N$ è una restrizione di $\bar{Q}$ su N.

Qui abbiamo tre stati: {{2,1} transitori, {0} assorbente (Nessun dispositivo operativo)}. Quindi {2,1} transitori. Scriviamo la matrice dei tassi di transizione:

$$\bar{Q} = \begin{bmatrix} -2\lambda & 2\lambda & 0 \\ \mu & -(\mu+\lambda) & \lambda \\ 0 & 0 & 0 \end{bmatrix}$$

Notiamo che per quanto concerne lo stato assorbente, NON si esce da esso, quindi i tassi di transizione sono praticamente tutti nulli. Una volta entrato NON ve ne si esce più. Consideriamo la restrizione agli stati transitori $\implies$

$$(\bar{Q}_N = \begin{bmatrix} -2\lambda & 2\lambda \\ \mu & -(\mu+\lambda) \end{bmatrix}) \in \mathbb{R}^{2\times 2}$$

Abbiamo:

$$\begin{cases} \bar{L}_N(\infty) = \begin{bmatrix} L_2(\infty) & L_1(\infty) \end{bmatrix} \\ \bar{\pi}_N(0) = \begin{bmatrix} \pi_2(0) & \pi_1(0) \end{bmatrix} \end{cases}$$

Poniamo: $\{L_2 := L_2(\infty),\ L_1 := L_1(\infty)\}$. Ipotizziamo che il sistema evolva a partire dallo stato 2 $\iff \{\pi_2(0) = 1,\ \pi_1(0) = 0\}$. Abbiamo:

$$\begin{bmatrix} L_2 & L_1 \end{bmatrix} \begin{bmatrix} -2\lambda & 2\lambda \\ \mu & -(\mu+\lambda) \end{bmatrix} = -\begin{bmatrix} \pi_2(0) & \pi_1(0) \end{bmatrix}$$

Noi vogliamo calcolare $\underline{L_i(\infty)}$ per gli stati transitori. $L_i(\infty) = MTTA_i$, ove la quantità sottolineata rappresenta il tempo medio trascorso dal processo nello stato i PRIMA dell'assorbimento. Dobbiamo considerare TUTTI gli stati transitori in cui ho vagato.

$$MTTF = \underline{MTTA} = \sum_{i \in N} MTTA_i$$

Il processo partità da uno degli stati transitori, e poi giungerà verso lo stato assorbente. Eseguendo opportunamente i prodotti matriciali otteniamo:

$$\begin{cases} L_2(-2\lambda) + L_1\lambda = (-\pi_2(0) = -1) \\ L_2(2\lambda) - L_1(\mu + \lambda) = (0 = -\pi_1(0)) \end{cases} \implies$$

$$\implies \begin{cases} L_1 = \dfrac{-1 + 2\lambda L_2}{\mu} \\ \left[L_2(2\lambda) - \dfrac{(2\lambda L_2 - 1)(\mu + \lambda)}{\mu} = 0 \implies \right. \\ \left[L_2(2\lambda)\mu - 2\lambda L_2\mu - 2\lambda^2 L_2 + 2\lambda L_2\mu + \mu + \lambda = 0 \right. \end{cases} \implies$$

$$\begin{cases} L_1 = \dfrac{-1 + 2\lambda\frac{\mu+\lambda}{2\lambda^2}}{\mu} = \dfrac{-2\lambda + 2\mu + 2\lambda}{\mu\lambda} = \dfrac{1}{\lambda} \\ L_2 = \dfrac{\mu + \lambda}{2\lambda^2} = \dfrac{1}{2\lambda} + \dfrac{\mu}{2\lambda^2} \end{cases}$$

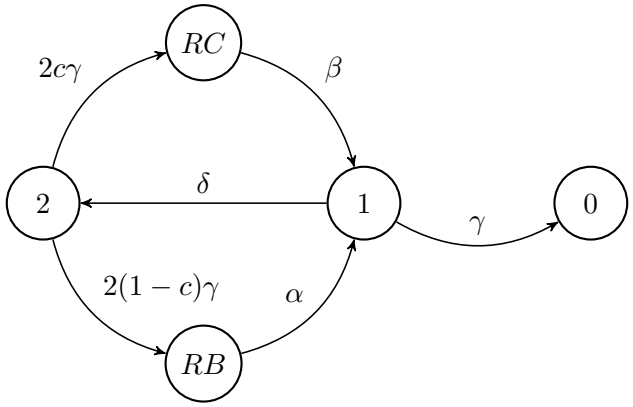
Assemblando opportunamente troviamo:

$$MTTF = MTTA = MTTA_2 + MTTA_1 = L_2(\infty) + L_1(\infty) = \frac{1}{2\lambda} + \frac{\mu}{2\lambda^2} + \frac{1}{\lambda} = \frac{3}{2\lambda} + \frac{\mu}{2\lambda^2}$$

Per questo approccio naturalmente servono comunque le condizioni INIZIALI relative agli stati transitori.

Exercise

$\implies$ {Stato DOWN ed UP nel sistema}. Computational Model {Affidabilità, Disponibilità}. ($n = 2$) devices workload. $\forall$ dispositivo soggetto a guasto con $MTTF = \frac{1}{\gamma}$. Recovery con probabilità c (Coverage Factor). Recovery ha un periodo di tempo in media pari a $(\frac{1}{\beta})$. La recovery può anche NON andare con successo con probabilità $(1 - c)$. Reboot lungo in tal caso, con tempo medio $(\frac{1}{\alpha})$. I dispositivi guasti hanno necessità di essere riparati (Single Repair Facility). La riparazione di un dispositivo non influenza il corretto funzionamento dell'altro. SRF per ipotesi. Se $\nexists$ sistemi funzionanti $\implies$ sistema DOWN e torna UP quando almeno un dispositivo è riparato. CMTC DTT. Modello di calcolo. NON possiamo procedere con gli RBD $\iff$ abbiamo SRF. Ma possiamo procedere con la CMTC con stati assorbenti. $MTTR = (\frac{1}{\delta})$. Analizziamo il DTT:



Quindi abbiamo c la probabilità di successo. Il tempo di soggiorno residuo nello stato 2 all'istante t è tale che: $\underline{\phi_2(t)} \sim EXP(2\gamma)$. Abbiamo inoltre: $\underline{\tau_{2,RC}} = (\frac{q_{2,RC}}{-q_{2,2}})$. Il LHS è la probabilità che lasciando lo stato 2, migriamo verso lo stato RC. Ma sappiamo già che essa è uguale a c! Di conseguenza:

$$\underline{c} = \frac{q_{2,RC}}{-q_{2,2}} \implies q_{2,RC} = c(2\gamma) = 2c\gamma$$

CMTC con questi stati:

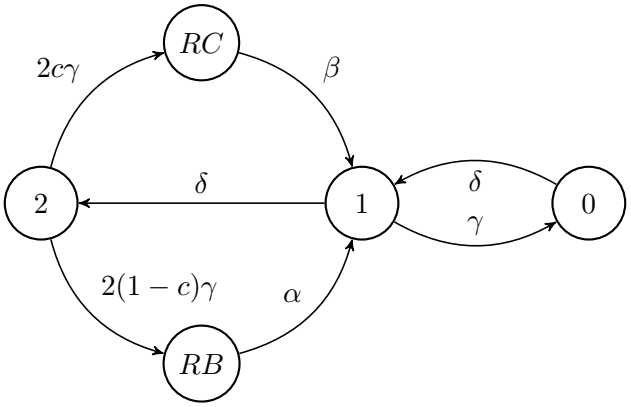
- *"i": 0,1,2*: "i" dispositivi. Working devices;
- RC: Recovery in atto;
- RB: Reboot in atto;

Spazio con 5 stati della mia catena. Recovery che dura in media $(\frac{1}{\beta})$. Se va male, con probabilità $(1-c)$, abbiamo un Reboot in atto che dura in media invece $(\frac{1}{\alpha})$. Velocità α per migrare da RB ad 1. Lo stato 1 indica che abbiamo un dispositivo funzionante, e l'altro è in riparazione. δ è la VELOCITÀ DI RIPARAZIONE. Dallo stato 1 si potrebbe avere una riparazione od un guasto. Dipende chi avviene prima. $(\frac{1}{\delta}) = MTTR$. Modello di calcolo per l'Affidabilità: TIME TO ABSORPTION. Sappiamo che $MTTF = MTTA$ (lifetime) X TTF del sistema. Ci serve $\pi_0(t)$, dal momento che: $R(t) = 1 - \pi_0(t) \iff$

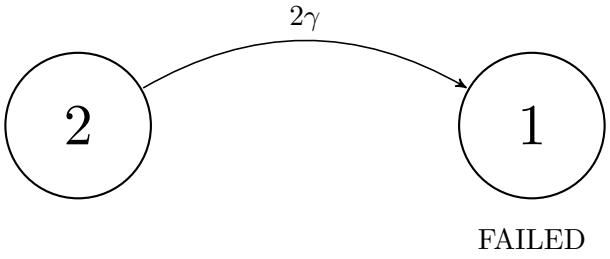
$$\underline{R(t) = \Pr\{X > t\}} = 1 - \Pr\{X \leq t\} = \underline{1 - \pi_0(t)}$$

Immaginiamo che evolva dallo stato 2. Poi, avendo $\pi_0(t)$, con il metodo delle TL, posso integrare l'Affidabilità $R(t)$ trovando così il: $MTTF = \int_0^\infty R(t)dt$. Approccio fattibile con l'altro metodo per il SOLO calcolo dell'MTTF. Contesto più restrittivo $\implies$ configurazione NON accettabile.

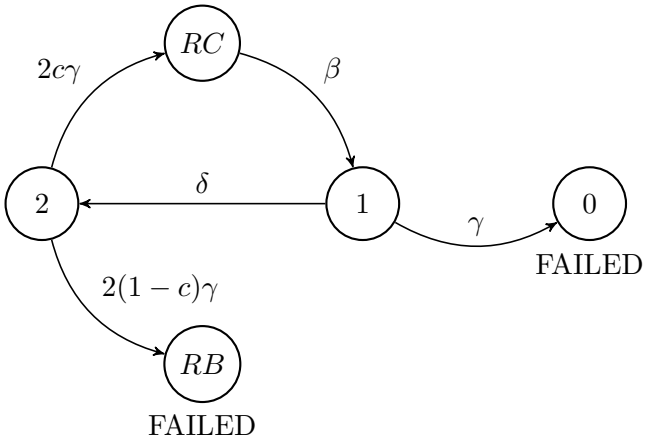
Il modello di calcolo dipende dai REQUISITI del sistema, ovvero {affidabilità, protezione, disponibilità}. Reliability Requirements. Vediamo ora il Computational Model per la DISPONIBILITÀ. Si tratta di aggiungere un ramo di transizione da 0 ad 1 nel precedente DTT, indicando che dallo stato di FAILURE si può prevedere una riparazione per tornare nello stato 1. Quindi se avessimo dovuto calcolare la disponibilità, quel ramo avrebbe tasso di transizione δ, perché abbiamo un SRF, altrimenti avremmo avuto 2δ. $[A = \pi_2 + \pi_1]$. Il DTT sarebbe stato quindi:



Ma noi stiamo parlando del modello di calcolo per l'Affidabilità. $R(t) = \Pr\{X > t\}$ è la probabilità che un sistema operi correttamente per almeno t unità di tempo.
SENZA INTERRUZIONI!! Le interruzioni relative agli ingressi negli stati {RB, RC} non sono poi in realtà catastrofiche! Non sono quindi considerati Stati ASSORBENTI. Interruzioni tollerate nei reliability requirements. DISPOSITIVO DOWN (OOO) $\iff \nexists$ dispositivi funzionanti. *Out Of Service*, *Out Of Order*. Se invece NON venissero accettate interruzioni, 1 dispositivo NON sarebbe tollerato. Adesso 1 è lo stato assorbente. Contesto restrittivo ove la riconfigurazione NON è accettata. Si figuri quindi il Reboot:



Se invece fosse tollerata la riconfigurazione, ci sarebbe nuovamente il ramo da RC ad 1, e mancherebbe invece il ramo da RB ad 1. Gli stati assorbenti in tal caso sarebbero RB e 0:



Abbiamo quindi visto 3 modelli di calcolo per l'Affidabilità che si riferiscono a tre contesti diversi. Se il contesto cambia, cambia ovviamente anche il Modello di Calcolo.

3.6 Markov Reward Model (MRM)

Eventi catastrofici $\leftrightarrow$ transizioni particolari verso stati assorbenti. Per questi stati, abbiamo $-q_{ii} = 0$, ovvero velocità totale di uscita NULLA, come è ragionevole aspettarsi. *J2EE* (Web, Application, DB). 5-9 (High Availability). Approccio gerarchico (CMTC).

3.6.1 Markov Reward Model (MRM)

Modelli di Markov con RICOMPENSA. Consideriamo una CMTC: $\{X(t),\ t \geq 0\}$. Homogeneous finite-state CMTC. Abbiamo quindi un numero di stati finito $\iff |statuses| = n < +\infty$. Pensiamo di associare ad ognuno degli stati della mia catena un *reward rate*. $r_i = (\dots)$ reward rate $\forall i \in S$. Il reward potrebbe anche essere negativo $\iff (\dots) < 0$. Quando il processo soggiorna nello stato i, accumula un reward $\underline{r_i \tau_i}$ (ricompensa accumulata dal processo nello stato i per τ_i unità di tempo). Il termine sottolineato costituisce quindi proprio un reward. Una CMTC che supporta queste condizioni/convenzioni è detta MRM. MRM $\rightarrow$ alcune quantità di interesse.

Definition 27. ***Reward Rate istantaneo***
Definiamo l'Instantaneous Reward Rate come:

$$[z(t) := r_{X(t)}]$$

Reward Rate istantaneo. Pedice $X(t)$. $X(t)$ è lo stato corrente al tempo t della mia CATENA, ovviamente variabile aleatoria. Reward associato allo stato in cui la catena si trova nell'istante di tempo t, $(X(t))$. Dato che $X(t)$ è una v.a. $\implies z(t)$ sarà una v.a. e possiamo quindi definire il suo valore medio come:

Definition 28. ***EXPECTED REWARD RATE***
Definiamo l'ERR (Expected Reward Rate) come:

$$ERR := \mathbb{E}[z(t)] = \sum_{i \in S} r_i \underline{\pi_i(t)}$$

$i \in S$ ovviamente. Ricordiamo che a tal proposito, $\pi_i(t) = \Pr\{X(t) = i\}$. $z(t)$ è una v.a. discreta. Abbiamo l'ERR. Possiamo considerare anche il valore a regime di questa quantità...

Definition 29. ***STEADY-STATE EXPECTED REWARD RATE***
Definiamo lo Steady-state expected reward rate come:

$$\mathbb{E}[z] = \sum_{i \in S} r_i \pi_i$$

Ma ricordiamo che affinché $\exists(\pi_i = \lim_{t \to +\infty} \pi_i(t))$, la CATENA deve anche ESSERE ERGODICA! In tal caso, per un MRM è sufficiente che sia IRRIDUCIBILE. Poniamo di trovarci dinanzi una ERGODIC CMTC. Definiamo:

Definition 30. ***ACCUMULATED reward*** $\Upsilon(t)$ ***in*** $[0, t]$
Definiamo la ricompensa accumulata in $[0, t]$ *come:*

$$\Upsilon(t) := \int_0^t z(\tau) d\tau$$

Dove ovviamente vale: $z(t) = r_{X(t)}$. Proprio per questo motivo, siamo nuovamente autorizzati ad effettuare una media, trattandosi di una variabile casuale:

$$\underline{\mathbb{E}[\Upsilon(t)]} = \mathbb{E}[\int_0^t z(\tau)d\tau] = \int_0^t \underline{\mathbb{E}[z(\tau)]}d\tau =$$

$$= \int_0^t \sum_{i\in S} r_i\pi_i(\tau)d\tau = \sum_{i\in S} r_i \int_0^t \pi_i(\tau)d\tau = \sum_{i\in S} r_i L_i(t)$$

Ove vari passaggi si sono ottenuti per linearità della media e della sommatoria/integrale. Procediamo alla seguente definizione:

Definition 31. ***EXPECTED Accumulated REWARD*** $\Upsilon(t)$ ***in*** $[0,t]$

Definiamo la ricompensa accumulata attesa in $[0,t]$ *come:*

$$\underline{\mathbb{E}[\Upsilon(t)]} = \mathbb{E}[\int_0^t z(\tau)d\tau] = \sum_{i\in S} r_i L_i(t)$$

(Per quanto tempo in media sono stato in quegli stati nella finestra temporale $[0,t]$). Otteniamo il REWARD medio accumulato dagli stati...

3.6.2 Analisi Combinata

Questo modello serve per affrontare l'ANALISI COMBINATA delle performance (prestazioni) di un sistema (Server che elabora i job = $\cdot(throughput)$), (Job in media elaborati dal server $\forall$ unità di tempo). Abbiamo le seguenti caratteristiche da tenere in conto:

- PERFORMANCE;
- DEPENDABILITY

Ove l'ultimo elemento sottolineato si compone di {Affidabilità (Reliability) $\vee$ Disponibilità (Availability)}. L'Analisi Combinata prevede di analizzare uno dei seguenti scenari: {PERFORMANCE+Reliability $\vee$ PERFORMANCE+Availability}. Particolarmente importante per l'analisi dei cosiddetti Degradable Systems (DEGRADABLE), ovvero sistemi con un certo numero di dispositivi funzionanti (RIDONDANZA ATTIVA), quindi tutti quanti lavorano, per aumentare la capacità elaborativa del sistema. Con un DEGRADABLE, quando un dispositivo si guasta in ridondanza attiva, il sistema si riconfigura ed entra (opera) in modalità DEGRADED. Continua a funzionare ma con capacità elaborativa RIDOTTA. (DEGRADED MODE). Se i tempi di servizio sono troppo elevati in DM, si potrebbe preferire chiudere TUTTI i server.

ANALISI COMBINATA con un unico modello, detto MODELLO COMPOSITO. Ma procedendo così si potrebbero avere problemi di SCALABILITÀ. Modello che considera sia eventi importanti relativi alle performance {arrivo di un job, completamento di un job}, sia quelli collegati ad esempio alla Disponibilità (guasti, etc.). COMPOSITE MODEL. Problemi: *Largeness*, *Stiffness*. La Largeness è un problema relativo all'avere un modello con tanti stati. Stiffness si riferisce al fatto che potrei avere delle difficoltà nell'ambito del mio sistema a causa delle differenze tra i rate relativi agli eventi collegati con le performance e quelli collegati agli eventi relativi alla DEPENDABILITY. Enorme differenza tra i rate $\implies$ INSTABILITÀ NUMERICA nel modello $\implies$ Non tollerabile nelle simulazioni numeriche ai calcolatori. Si accetta quindi un piccolo errore in relazione a ciò che avviene nelle performance, se consideriamo un intervallo molto grande relativo alla dependability. Situazione di *QUASI STEADY-STATE*

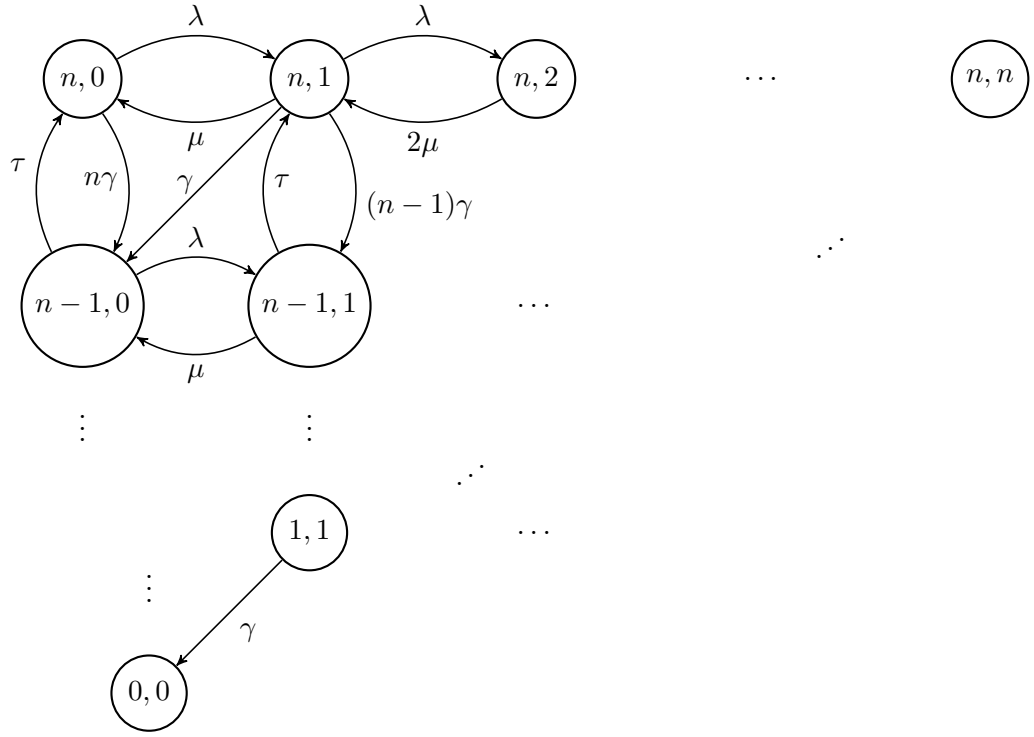
(QSS). Struttura gerarchica. ∀ stato associato al Modello di Dependability abbiamo un differente modello per le performance. Si compie però un certo errore, che può essere tanto piccolo quanto più grande è la frequenza degli eventi collegati alle performance e tanto più piccola è la frequenza con la quale accadono invece gli eventi relativi alla dependability.

Composite Model / Server Farm

Un Service provider fornisce capacità di calcolo ai suoi clienti. Sistema con perdita se tutti i server sono BUSY (occupati). Richieste arrivano secondo un processo λ-POISSON. Probabilità di perdita = ?

$$\begin{cases} TTF \sim EXP(\gamma) \\ TTR \sim EXP(\tau) \end{cases}$$

Abbiamo una SRF, ovvero una Single Repair Facility. Approccio gerarchico possibile, ma l'analisi è fattibile anche con il modello composito, che effettua l'analisi combinata delle performance e della disponibilità. ANALISI COMBINATA. Modello composito che tenga contemporaneamente in conto eventi delle due nature insieme. Problemi di SCALABILITÀ, Largeness e Stiffness. Differenza tra frequenze relative a problematiche di performance e quelle relative alle problematiche di dependability. Un approccio utilizzato è quello della MODELLAZIONE GERARCHICA (più modelli a più livelli). Indice di prestazioni ∀ stato del modello di disponibilità. Questa misura è proprio il reward rate. Con il modello composito avremmo il seguente DTT bidimensionale:



La variabile di stato è: $\{|dispositivi\ funzionanti|\ ,\ |job\ in\ esecuzione|\}$. Abbiamo i seguenti vari tassi di interesse: $\{\gamma, \lambda, \tau, \mu\}$. Si potrebbe ad esempio avere una transizione: $(n, 0) \rightarrow (n, 1)$, con ciò intendendo che, sempre con n server funzionanti, si ha l'arrivo di un nuovo job da elaborare. Potrebbe arrivare poi un'altra richiesta: $(\dots) \rightarrow (n, 2)$ migration, e così via. Ma potrebbe accadere che, prima ancora che accada qualsiasi altro evento, si faccia la

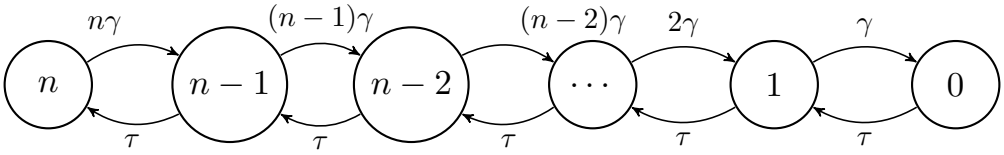
transizione verso $(n, 0)$ nuovamente. Oppure in $(n, 1)$ potrebbe accadere che si guasti uno dei server funzionanti (degli n). A guastarsi potrebbe essere proprio quello sul quale si sta eseguendo l'unico job disponibile, oppure qualcuno idle. Nel primo caso, si ha una transizione verso $(n-1, 0)$. Se INVECE non è quel server a guastarsi, ci sarebbe una transizione verso $(n-1, 1)$. Il modello composito tiene conto del fatto che i job si riducono quando il server che si guasta è proprio quello sul quale si sta eseguendo il job. Nel modello gerarchico non teniamo invece conto di questo. Piccolo errore che commettiamo. A titolo informativo abbiamo che il passaggio $(n, 2) \to (n, 1)$ avviene a velocità 2μ. Dovrei considerare la v.a. min etc. Ovvero dovrei considerare: $\phi_{n,1}(t) \sim EXP(-q_{ii})$ la v.a. tempo di soggiorno residuo, partendo dal considerare la v.a. $\min(\dots)$ che entra in gioco quando cerchiamo di determinare la CDF complementare di questo tempo di soggiorno residuo, etc, ovvero: $\Pr\{\phi_{n,1}(t) > \tau'\}$. Probabilità che in τ' non accade nessun evento che ne causi la transizione dello stato. Uscirà un esponenziale che ha come esponente la somma dei parametri. Probabilità congiunta $\iff$ Prodotto delle probabilità. Se adottassi questa strategia (modello composito), sarebbe veramente difficile calcolare tutte le quantità. Potrei calcolare la distribuzione di regime con un calcolatore $\implies$ ma avrei problemi di INSTABILITÀ:

$$P_L = \pi^{(a)}_{n,n} + (\dots) + \pi^{(a)}_{0,0} \overset{PASTA}{=} (\dots)$$

La LOSS PROBABILITY corrisponderebbe alla somma di termini del tipo $\pi^{(0)}_{i,i}$, etc. Per via di POISSON vale PASTA. Somma delle probabilità su tutti gli stati diagonali sostanzialmente. Approccio che si basa sul modello composito.

Approccio gerarchico

Dobbiamo derivare un modello di disponibilità di più alto livello e $\forall$ stato di questo modello deriviamo un modello di basso livello relativo alle performance. Availability Model. Come stato consideriamo: $|server\ funzionanti|$ al tempo t. Questa definizione di stato corrisponde ad una CMTC con il seguente DTT:

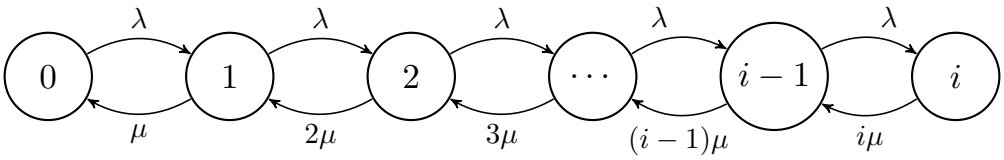


τ perché SRF. Numero di server funzionanti al tempo t è la nostra definizione di stato. τ è la velocità alla quale viene riparato un server. Catena OMOGENEA, IRRIDUCIBILE e $|stati| < +\infty \iff$ ERGODICA $\iff$ possiamo scrivere la distribuzione di regime:

$$\begin{cases} \pi_i = \pi_0 (\frac{\tau}{\gamma})^i \frac{1}{i!},\ 0 < i \leq n \\ [\pi_0 = \frac{1}{\sum_{i=0}^{n} (\frac{\tau}{\gamma})^i \frac{1}{i!}}] \end{cases}$$

Calcolate con le apposite formule...

Adesso $\forall$ stato i dobbiamo considerare un modello per le prestazioni. Obiettivo: probabilità di perdita di una richiesta nell'esecuzione di un job. Consideriamo quindi uno stato i del modello di disponibilità. Il sottomodello prevede come stato: $|job\ in\ esecuzione|$ al tempo t. PERFORMANCE MODEL, valido quando vi sono i server funzionanti. Il DTT è il seguente:



($i = 2$). Potrebbe arrivare un nuovo job oppure si potrebbe avere il completamento di uno dei due job in esecuzione. i server funzionanti! Più di i server in esecuzione NON li possiamo avere. i servitori che corrispondono a server funzionanti a disposizione. Si nota subito che questo DTT è molto simile a quello della M/M/m/0. CATENA OMOGENEA, IRRIDUCIBILE e $|stati| < +\infty \iff$ ERGODICA $\iff$ scriviamo la probabilità a regime:

$$\begin{cases} P_j = P_0(\frac{\lambda}{\mu})^j \frac{1}{j!} \\ [P_0 = \frac{1}{\sum_{j=0}^{i} (\frac{\lambda}{\mu})^j \frac{1}{j!}} \end{cases}$$

dove ovviamente vale: $j = 0, \ \ldots, \ i \iff 0 < j \leq i$. P_j è la probabilità che vi siano esattamente j job in esecuzione. Espressione trovata già per la M/M/m/0. In tal caso $m := i$. A questo punto possiamo calcolare, utilizzando il modello di prestazioni quando siamo nel macrostato i server funzionanti, la probabilità di perdita P_L:

$$\underline{P_L(i)} = \underline{P_i^{(a)} \overset{PASTA}{=} P_i} =$$

$$= [\frac{(\frac{\lambda}{\mu})^i \frac{1}{i!}}{\sum_{j=0}^{i} (\frac{\lambda}{\mu})^j \frac{1}{j!}}] = B(i, \frac{\lambda}{\mu})$$

Dove la prima quantità sottolineata è la probabilità di perdita quando abbiamo i server in esecuzione. Quindi la seconda uguaglianza sottolineata, valida in virtù dell'applicazione di PASTA, coinvolge la probabilità che vi siano i job in esecuzione. In tal caso verrebbe scartata la richiesta. La formula trovata è nientemeno che la B di ERLANG per la M/M/m/0 con parametri $(m, \frac{\lambda}{\mu})$.

M/M/m/m. Consideriamo m utenti in tutto il sistema. Questo indice di prestazioni si riferisce al fatto che i server funzionanti sono i. Questa misura sarà proprio il reward rate da associare allo stato i del modello di disponibilità (Availability Model). E questa probabilità si calcola con la B di ERLANG $B(i, \frac{\lambda}{\mu})$.

$$r_i := P_L(i), \ i = 1, 2, \ \ldots, \ n$$

Banalmente, $\rightarrow i = 0 \implies P_L(0) = 1$. Adesso applichiamo la formula di reward rate a regime:

$$[\mathbb{E}[z] = \sum_{i=0}^{n} r_i \pi_i] = [\sum_{i=1}^{n} P_L(i)\pi_i + \underline{(r_0 = 1)}\pi_0]$$

Questa quantità sarà proprio la mia probabilità di perdita. Ricordiamo che $\mathbb{E}[z]$ è il valor medio del reward rate a regime. z è l'Instantaneous Reward Rate $\implies z(t) = r_{X(t)}$ (reward rate associato allo stato nel quale il processo si trova a tempo t). z rappresenta quindi la probabilità di perdita istantanea (all'istante t). Ricordiamo che π_i sono le MACROPROBABILITÀ, ovvero le probabilità di stato a regime relative al modello di disponibilità di più alto livello, mentre le $P_L(i)$, che poi sarebbero i reward rate i-esimi associati agli stati i del modello di disponibilità, si calcolano sfruttando invece le probabilità di stato a regime del modello di più basso livello.

Questo è un modo elegante per formalizzare il tutto in termini di MRM. Reward rate associati ai vari stati. Approccio gerarchico si potrebbe pensare come una sorta di Probabilità Totale. Ma ragioniamo in termini di MRM. Qui però non descriviamo alcuni eventi: (che un job vada meno quando un server vada meno. Ma frequenza molto basse!) eventi che coinvolgono due stati contemporaneamente. Tempi di interarrivo associati alle due problematiche molto differenti. CATENA ERGODICA. Non è una vera situazione di Steady State. Si parla in letteratura di *QUASI STEADY-STATE*. Dato che i tempi di interarrivo associati agli eventi di disponibilità sono di ordini di grandezza differenti (più grandi), possiamo considerare il valore a regime.

RAS Remote Access Server con un modello gerarchico (approccio gerarchico).

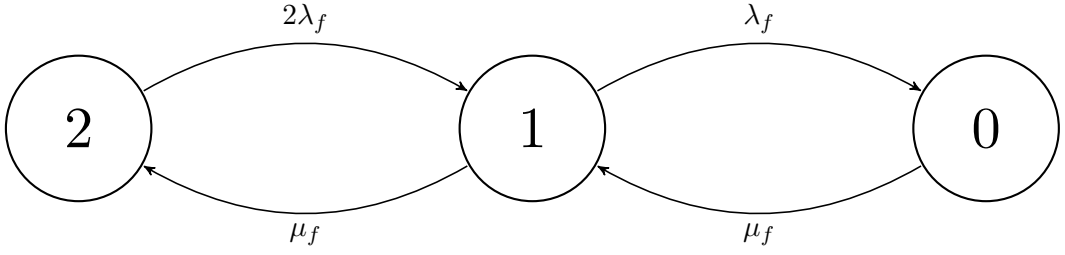
$$t = \frac{\hat{\theta} - \theta}{\hat{\delta}(\hat{\theta})} \frac{\delta^\star}{\sqrt{R}} \sim TSTUDENT$$

Considerando $\Upsilon_1, \Upsilon_2, \ \ldots, \ \Upsilon_R$, abbiamo: $\hat{\theta} = \frac{\sum_{i=1}^{R} \Upsilon_i}{R}$.

3.6.3 RECAP

Modellazione gerarchica. (Possibili sistemi parallelo). Il throughput NON aumenta lienarmente con il numero di processi. Più probabile che accada qualche malfunzionamento. Trade/Off. Più processori $\implies$ più intervalli di down del sistema. (In questo intervallo il sistema non può elaborare i job). Modellazione gerarchica, MRM, Analisi combinata di {Performance + Dependability}. Negli esempi visti entrambi i modelli erano rappresentabili mediante CMTC. Ma non è sempre così! Consideriamo ad esempio degli switch lvl 2 (*Equipment Room*). Centro stella di edificio (Campi blu/Pannelli blu). Parlando di progettazione, nell'ER si trovano switch multilayer per la ridondanza! Meglio non prevedere ridondanze interne negli switch multilayer. A livello di piano è meglio ridondare internamente gli switch lvl 2 per il centro stella di piano invece. Tipicamente sono ridondati i componenti cruciali: {alimentatore (FANS), ventole (power supply), hypervisor (intelligenza)} = SWITCH RIDONDATO. Parliamo di *IN THE BOX REDUNDANCY* (IBR). Si valuti la disponibilità di uno switch con IBR. Lo switch (lvl 2) è considerato non disponibile quando uno o più sottosistemi si sono guastati (includendo le unità ridondanti). Si supponga che guasti e riparazioni siano indipendenti per i tre sottosistemi. Guasti e riparazioni indipendenti. A questo punto potremmo utilizzare i famosi RBD (ABD in realtà, in questo caso). Modelliamo la disponibilità con tre sottosistemi serie: {fans, power supply, Hypervisor} $\rightarrow \{A_f, A_{ps}, A_h\}$. La disponibilità totale è: $A = [A_f A_{ps} A_h]$. Scegliendo un approccio gerarchico questo potrebbe essere il modello di livello più alto.

- **FANS**: Supponiamo che per la disponibilità del sistema di raffreddamento (fans) si abbia un *Parallel Redundant System* (PRS) con SRF, al quale corrisponde il modello di disponibilità sintetizzato nel seguente DTT:



Sia: μ_f la velocità di riparazione (repair rate); λ_f il failure rate di una ventola. COOLING SUBSYSTEM. Questo è quindi l'availability model per il sistema di raffreddamento. Per questo modello la variabile di stato è: $|ventole\ funzionanti|$ al tempo t, e valgono le seguenti:

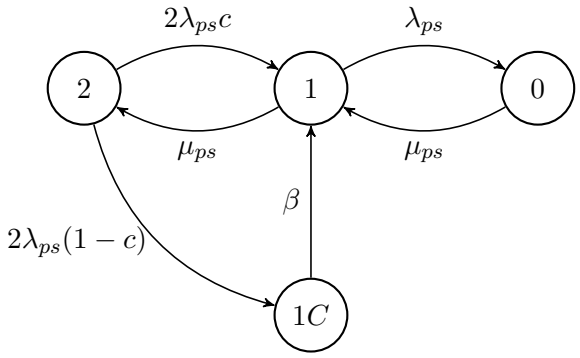
$$\begin{cases} TTF \sim EXP(\lambda_f) \\ TTR \sim EXP(\mu_f) \end{cases}$$

Ricordiamo che abbiamo un SRF. Risulta:

$$A_f = \frac{MTTF}{MTTF + MTTR} = 1 - \pi_0 = \pi_2 + \pi_1$$

in quanto il sistema raffredderebbe lo switch sia in stato 1 che in stato 2.

- **POWER SUPPLY**: Consideriamo l'altro sottosistema (degli alimentatori). Power SUPPLY SUBSYSTEM. Si abbia questo modello di disponibilità (Availability Model):

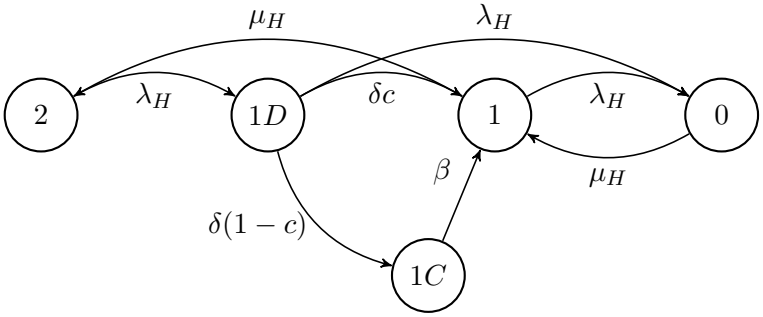


availability model with imperfect coverage. Per il sottosistema di alimentazione si abbia ridondanza con imperfect coverage: se si guasta uno dei due alimentatori, si cerca di ripristinarlo (può andare bene o male). λ_{ps} è il failure rate, con valor medio ($\frac{1}{\lambda_{ps}}$). Software di recovery installato. Se la procedura di recovery va bene, il sistema switcherà sull'altro. Abbiamo:

$$\begin{cases} TTF \sim EXP(\lambda_{ps}) \\ TTR \sim EXP(\mu_{ps}) \end{cases}$$

($\frac{1}{\beta}$) è il tempo medio di reboot del sistema. Di nuovo abbiamo: $A_{ps} = \pi_2 + \pi_1$.

- **HypervisoR**: Qui abbiamo un availability model con imperfect coverage e detection delay. LAVORANO UNA ALLA VOLTA! (Hypervisor subsystem). DTT:



($\frac{1}{\beta}$) è il tempo medio di reboot. ($\frac{1}{\delta}$) è il ritardo medio di detection. A velocità λ_H si guasta l'unico hypervisor funzionante. Si parta dallo stato 2. Lo stato 1D non è contemplato nella valutazione della disponibilità, che è in tal caso:

$$[A_D = \pi_2 + \pi_1]$$

Stiamo contemplando il caso in cui la rilevazione possa avere successo oppure no. Se va bene si ha una transizione verso $\rightarrow 1$. Se va male allora abbiamo bisogno del reboot; si va nello stato 1C a tasso $\delta(1-c)$, ove $(1-c)$ è l'anticoverage factor.

Concludiamo notando che l'approccio gerarchico funziona quindi anche molto bene in altre realtà.

Scenario tipico: due interfacce di rete $\implies$ ridondanze sull'interfaccia di rete. *Linux HA* (High Availability). Supporta la WARM-COLD REDUNDANCY. Tipicamente abbiamo: $[\frac{1}{\delta} \ll \frac{1}{\beta}]$. NEAR-COINCIDENT FAULTS RBD.

Capitolo 4

Reti di Code

4.1 Starting Point

4.1.1 Client-Server System Exercise

Si consideri un sistema client-server con M clienti. Ogni utente, indipendentemente agli altri, dopo un *think period*, o periodo di riflessione, il quale è esponenzialmente distribuito con media di $\frac{1}{\lambda}$ secondi, invia una richiesta al server. Il server esegue le richieste con disciplina di coda FCFS. Si assume che il *service time*, ovvero il tempo di servizio per richiesta sia esponenzialmente distribuito con media $\frac{1}{\mu}$ secondi e che un user generi una nuova richiesta solo dopo che la precedente sia stata servita, completata (sempre dopo un think time preliminare).

Gli utenti comunque, sono un po' impazienti, e possono cancellare le richieste inviate al server: ogni utente, indipendentemente dagli altri, decide di cancellare la richiesta mandata dopo un intervallo di tempo (partendo dall'istante quando la richiesta è stata inviata) la cui lunghezza è esponenzialmente distribuita con parametro γ. La richiesta, comunque, può essere cancellata solo se sta ancora aspettando.

Una volta che la richiesta è stata cancellata, l'utente genera una nuova richiesta dopo un nuovo periodo di riflessione.

Si assuma che i tempi di servizio, think times e tempi di cancellazione siano variabili casuali statisticamente indipendenti.

- a) Si disegni il DTT della CMTC che modelli tal sistema;

- b) Si valuti la probabilità a regime (steady-state) che vi siano i richieste in esecuzione od in attesa sulla CPU, l'utilizzazione della CPU, il throughput medio, il numero medio di utenti in *thinking state*;

- c) Assumendo che le richieste non possano essere cancellate, si tracci il grafico del tempo di risposta medio in funzione del numero di clienti, M, nel caso $\frac{1}{\lambda} = 15s$, $\frac{1}{\mu} = \ s$.

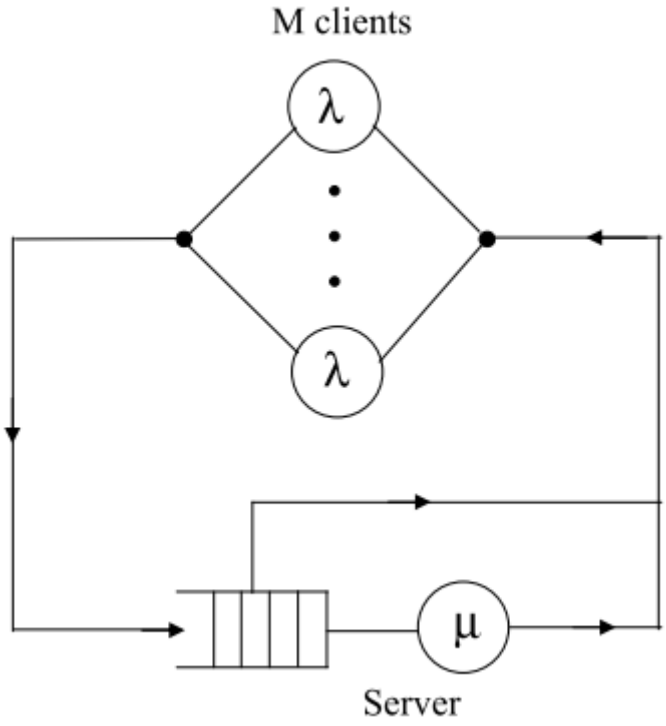


Figura 4.1: Client-Server System

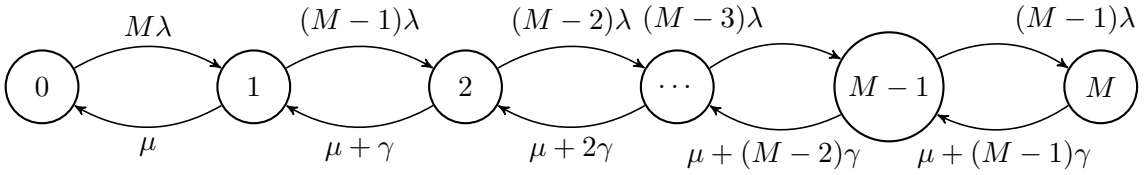
Client/Server con M utenti. $\forall$ utente, in media dopo $\frac{1}{\lambda}s$ (EXP NEG) invia una richiesta al server. Il server esegue le richieste con FCFS. $\frac{1}{\mu}s$ è il tempo di servizio. Un utente genera una richiesta dopo che la precedente è stata servita. Ogni utente, indipendentemenete dagli altri, dopo un periodo di riflessione ($\frac{1}{\lambda}$), può quindi inviare le richieste al server. Gli utenti sono tuttavia impazienti e possono cancellare le richieste: a partire dall'invio delle richieste, dopo un tempo di durata media $\frac{1}{\gamma}s$, cancella la richiesta. Può cancellare solo se la richiesta NON sta venendo eseguita. Tempi di servizio, cancellazione e di riflessione sono rispettivamente i.i.d. e statisticamente indipendenti tra di loro; Quindi DTT, utilizzazione CPU, ...

Modellazione di un sistema Web. Non è contemplata la possibilità che un utente si scocci e vada via. Condizioni di saturazione: Studio del sistema sotto-stress. $\max\{M\}$ per avere determinate prestazioni. Numero di utenti massimo che il server può sopportare, tollerare affinché si abbiano delle predeterminate prestazioni. $t \to \frac{1}{\lambda}$ per cliccare su un certo Hyperlink.

Sistema reale Questo sistema reale descritto può essere modellato come una RETE DI CODE CHIUSA. Sistema a coda. Modello stocastico che rappresenta questo sistema reale descritto. Varie entità in gioco: {{utenti}, server}. Si parla di: {tempi di riflessione, tempi di servizio, tempi di cancellazione}. Server Web. Processo applicativo. Clienti + servitore + fila di attesa. Una rete di code è un insieme di code interconnesse tra di loro secondo una certa topologia. APERTA quando i clienti possono arrivare dall'esterno e partire verso l'esterno di questo sistema. CHUSA quando NON c'è la possibilità che arrivino dall'esterno o partano verso l'esterno. In tal caso il numero dei clienti è costante (Non si arriva e non si parte). Situazioni intermedie rendono il sistema non stabile. I blocchi di ritardo sono modellabili con delle code $[\cdot/\mathrm{M}/\infty]$. Questa è una RETE DI CODE CHIUSA. $M = |clienti|$. M utenti nel sistema reale. Clienti + servitore. Un cliente si può trovare o nei blocchi di ritardo oppure nel sistema a coda. La presenza di un cliente nel blocco di ritardo rappresenta il fatto che nel sistema reale gli utenti possono avere un tempo di riflessione. Abbiamo M unità di ritardo. Se un cliente è in un blocco di ritardo, il corrispondente utente nel sistema reale si trova in un periodo di riflessione. Il pas-

saggio di un cliente dal blocco di ritardo verso il sistema a coda rappresenta il fatto che l'utente ha cliccato su un hyperlink: ha inviato una richiesta al server. Fila di attesa. Il passaggio di un cliente dalla fila di attesa verso il centro di servizio corrisponde al fatto che il server ha preso in considerazione la richiesta Web. Quando nel modello il cliente parte, andrà nuovamente nel blocco di ritardo. Il corrispondente utente nel sistema reale, avendo usufruito dell'erogazione del servizio, starà di nuovo pensando, scegliendo il successivo hyperlink da cliccare. C'è anche la possibilità che un cliente vada via dalla fila di attesa PRIMA ancora che venga servito. Ciò corrisponde alla situazione reale al fatto in cui un utente cancelli la propria richiesta in attesa, in coda. Possibilità che i clienti vadano via dalla fila di attesa. Anche quando un cliente va via dalla fila di attesa, giungerà nuovamente nei blocchi di ritardo. Modello stocastico che rappresenta bene il sistema reale.

Rete di code. Potrei risolverla con le tecniche adibite alla risoluzione di queste reti di code. Ma procederemo con i processi stocastici. Consideriamo il processo stocastico: $N(t) = |clienti\ del\ sistema\ a\ tempo\ t|$ (Nel sistema a coda). Dinamica determinata dalle v.a.: {tempi di riflessione, tempi di servizio, tempi di annullamento}. Sono tutte v.a. exp-neg i.i.d. e statisticamente indipendenti tra di loro $\implies$ CMTC. DTT:



I clienti che NON si trovano nel sistema a coda saranno in una condizione di ARRIVO nel sistema a coda. Il tempo che ci mette per uscire dal blocco di ritardo e giungere nel sistema a coda è $(\dots) \sim EXP - NEG(\lambda)$ i.i.d. indipendente dalle altre. Consideriamo uno stato non estremo. $0 < i < M$. Supponiamo che al tempo t, $N(t) = i = |clienti\ in\ fila\ di\ attesa| + |clienti\ nel\ centro\ di\ servizio|$. Avremo rispettivamente $i-1$ clienti in coda ed 1 nel centro di servizio. Se lo stato a tempo t è i, avremo un certo numero $(M-i)$ in condizioni di arrivo. Se lo stato è i, la successiva transizione di stato potrà essere determinata da: un arrivo nel sistema a coda, quando finisce un servizio in corso, oppure a causa di un annullamento di una delle $(i-1)$ richieste pendenti. In questo caso, si avrà una transizione di stato (variazione di stato). $t \to$ successivo arrivo. Indichiamo al solito con $\xi_{R(t)}|_i \sim EXP((M-i)\lambda) = \min\{(\dots)\}$, ovvero pari al minimo delle v.a. che rappresentano il tempo di arrivo residuo del k-esimo cliente nel blocco di ritardo (NO TEMPO DI INTERARRIVO!). Tempo che passa invece dall'istante t al successivo completamento dell'erogazione del servizio è $\eta_{R(t)}|_i \sim EXP(\mu)$. Il tempo che passa dall'istante presente t ed il successivo annullamento della richiesta è una v.a. definita in tal modo: $\theta_{R(t)}|_i \sim EXP[(i-1)\gamma] = \min\{(\dots)\}$, ovvero pari al minimo delle v.a. che rappresentano il tempo di annullamento residuo al tempo t. Ricordiamo: $\phi_i(t) \sim EXP(-q_{ii})$, ovvero il tempo di soggiorno nello stato i. Inoltre, $\Pr\{\phi_i(t) > \tau\}$, la sua CDF complementare, è pari alla probabilità che in τ NON ci sia nessun arrivo, NON ci sia la fine del servizio in corso e NON vi sia annullamento di una richiesta; il tutto condizionato con il fatto di trovarci nello stato i. Per indipendenza, essa è pari al prodotto delle rispettive probabilità, ovvero:

$$\Pr\{\phi_i(t) > \tau\} = [\Pr\{\xi_{R(t)} > \tau \mid i\} \Pr\{\eta_{R(t)} > \tau \mid i\} \Pr\{\theta_{R(t)} > \tau \mid i\}] =$$
$$= \mathrm{e}^{-(M-i)\lambda\tau}\, \mathrm{e}^{-\mu\tau}\, \mathrm{e}^{-(i-1)\gamma\tau} = e^{\underline{[-[(M-i)\lambda+\mu+(i-1)\gamma]=x]\tau}}$$

Cosicché $(-q_{ii} = [x]) > 0$, ovvero abbiamo trovato la VELOCITÀ TOTALE DI USCITA. Procediamo al solito modo:

$$\underline{\tau_{i,i+1}} = \frac{q_{i,i+1}}{-q_{ii}} = (\dots)$$

dove $[-q_{ii} = [(M-i)\lambda + \mu + (i-1)\gamma]]$; il termine sottolineato è invece la probabilità che, lasciando lo stato i, migriamo verso lo stato $i+1$.

$$(\dots) = \Pr\{\xi_{R(t)} < \underline{\min\{\eta_{R(t)},\ \theta_{R(t)}\}}\} = (\dots)$$

ove tale quantità rappresenta la probabilità che il successivo arrivo preceda il verificarsi dei due altri eventi. Tramite il teorema delle probabilità totali, otterremo alla fine:

$$(\dots) = \frac{(M-i)\lambda}{(M-i)\lambda + \mu + (i-1)\gamma} \implies$$

$$\begin{cases} [q_{i,i+1} = (M-i)\lambda] \\ [q_{i,i-1} = \mu + (i-1)\gamma] \end{cases}$$

Abbiamo quindi capito perché i tassi di transizione sono proprio questi.

$$N(t) = 0 \implies \underline{\phi_0(t)} \sim EXP(-q_{00}) = \underline{\xi_{R(t)}|_0} \sim EXP(M\lambda)$$

$\implies \underline{-q_{00} = M\lambda}$. Se invece consideriamo lo stato $N(t) = M$ (tutti i clienti nel sistema a coda), avremo 0 clienti invece nel blocco di ritardo. La successiva transizione di stato sarà determinata o dall'annullamento di una richiesta o dalla fine dell'erogazione del servizio in corso:

$$\phi_M(t) = \min\{\eta_{R(t)}, \theta_{R(t)}\} \sim EXP((M-1)\gamma + \mu)$$

$\implies -q_{MM} = (M-1)\gamma + \mu$. Quindi in realtà possiamo considerare valida la definizione di $-q_{ii}$ per $1 < i \leq M$, in realtà.

CATENA $\underline{\text{ERGODICA}} \iff$ OMOGENEA, IRRIDUCIBILE e $|stati| < +\infty \iff$ numero di stati finito. Nel sistema a coda più di M clienti NON ci possono essere ($\iff |stati| < +\infty$).

CMTC tempo continuo, nascita e morte $\implies$

$$\begin{cases} [\pi_i = \pi_0 \lambda^i \dfrac{M!}{(M-i)!} \dfrac{1}{\prod_{k=0}^{i-1}(\mu + k\gamma)}] \\ \pi_0 = \dfrac{1}{1 + \sum_{i=1}^{M} \lambda^i \frac{M!}{(M-i)!} \frac{1}{\prod_{k=0}^{i-1}(\mu+k\gamma)}} \end{cases}$$

ove π_0 lo otteniamo con la condizione di NORMALIZZAZIONE. $i-1$ richieste pendenti ed un'altra in esecuzione nella CPU (del Server). ρ è l'UTILIZZAZIONE, ovvero la frazione di tempo in media nella quale il servitore è impegnato a servire i clienti: $[\rho = 1 - \pi_0]$. Adesso dobbiamo valutare il throughput medio (throughput della CPU). Numero medio di richieste servite dalla CPU $\forall$ unità di tempo. Quando la CPU termina il servizio della richiesta, il cliente partirà dal centro di servizio. Applichiamo $\underline{\text{Little}}$ ed otteniamo:

$$[\rho = \frac{\lambda_S}{\mu} \implies \lambda_S = \mu\rho = \mu(1-\pi_0)]$$

Se consideriamo il prodotto $\pi_i[\mu + (i-1)\gamma]$, otteniamo la frequenza delle transizioni di questo tipo. Numero medio di volte che accade questa transizione $\forall$ unità di tempo. Consideriamo allora: $\underline{\pi_i\mu} + \pi_i(i-1)\gamma$. Il primo termine sottolineato è la frequenza delle transizioni che portano da i ad $i-1$ A CAUSA della fine di un servizio, mentre il secondo termine sottolineato rappresenta la frequenza delle transizioni $i \to i-1$ A CAUSA dell'annullamento di una richiesta.

$$\begin{cases}[\lambda_S = \mu\pi_1 + \mu\pi_2 + \mu\pi_3 + \cdots + \mu\pi_M = \mu(1-\pi_0) = \mu\rho] \\ [\lambda_R = \gamma\pi_2 + 2\gamma\pi_3 + 3\gamma\pi_4 + \cdots + (M-1)\gamma\pi_M]\end{cases}$$

Ove la seconda equazione rappresenta la velocità di partenza dei clienti dalla fila di attesa. Per trovare il numero medio di clienti in stato di riflessione, quindi, applichiamo nuovamente Little:

$$\bar{N}_R = \underline{(\lambda_S + \lambda_R)}\frac{1}{\lambda}$$

$$\bar{N}_R = 0\pi_M + 1\pi_{M-1} + 2\pi_{M-2} + \cdots + M\pi_0$$

Non è infatti difficile riconoscere ivi una media. Si può dimostrare che i due termini sono praticamente uguali e consistenti utilizzando le equazioni di BILANCIAMENTO LOCALE:

$$\pi_{i-1}(M-i+1)\lambda = \pi_i[(\mu + (i-1)\gamma)]$$

Assumendo che le richieste NON possano essere cancellate, abbiamo che:

$$\underline{\mathbb{E}[R]} + \frac{1}{\lambda}$$

Il primo termine sottolineato è il tempo medio di risposta, ove con R intendiamo il tempo di risposta del sistema a coda, mentre il secondo addendo è al solito il tempo medio di riflessione. Un utente invierà una richiesta ogni $(\mathbb{E}[R] + \frac{1}{\lambda})$. Abbiamo M richieste. M utenti invieranno M richieste alla seguente velocità:

$$(\frac{M}{\mathbb{E}[R] + \frac{1}{\lambda}}) = \underline{\lambda_S} = \mu\rho$$

Dato che un utente NON può inviare una richiesta PRIMA che la precedente sia soddisfatta. Ove adesso il termine sottolineato rappresenta la velocità di arrivo delle richieste. Cosicché abbiamo:

$$\frac{M}{\mu\rho} = \mathbb{E}[R] + \frac{1}{\lambda} \implies \mathbb{E}[R] = \frac{M}{\mu\rho} - \frac{1}{\lambda} \implies \mathbb{E}[R] = \cdot(M)$$

Se abbiamo 1 solo cliente $\iff M \to 1 \implies \mathbb{E}[R] = \frac{1}{\mu}$ NO fila di attesa. $M \uparrow (\cdot \to \infty) \implies \rho \to 1 \implies \mathbb{E}[R] = (\frac{M}{\mu} - \frac{1}{\lambda})$, la quale quantità corrisponderebbe all'ASINTOTO OBLIQUO del grafico di $\mathbb{E}[R]$ in funzione di M.

L'asintoto viene in letteratura chiamato *HEAVY LOAD asymptote*, mentre la retta ovviamente corrisponderebbe al *LIGHT LOAD asymptote*. Il valore di M, $M^\star$ per il quale i due asintoti si incontrano, viene chiamato in letteratura *SATURATION NUMBER*, ovvero il valore di $M = M^\star$ per il quale il sistema si considera in SATURAZIONE. Carico che può essere tollerato da un servizio, da un server:

$$\frac{1}{\mu} = \frac{M}{\mu} - \frac{1}{\lambda} \implies \frac{M}{\lambda} = \frac{1}{\mu} + \frac{1}{\lambda} \implies M^\star = (1 + \frac{\mu}{\lambda})$$

Tale valore rappresenta quindi il valore di M per il quale il sistema è in saturazione. Se $\frac{1}{\lambda} = 15s$, e nel caso in cui $\frac{1}{\mu} = 1s$, abbiamo $\implies M^\star = 16 = 1 + (15 = \frac{\mu}{\lambda})$.

Ricordiamo che:

$$\underline{R(t) = \Pr\{X > t\}}$$

è l'AFFIDABILITÀ.

4.1.2 RECAP

M/M/1. Stato: $N(t) := |clienti\ nel\ sistema\ a\ coda|$ (nell'intero sistema a coda). $N_q(t) := |clienti\ in\ fila\ di\ attesa|$. CATENA DI MARKOV (OMOGENEA). Noto lo stato presente è possibile determinare l'evoluzione futura del processo in termini probabilistici, senza conoscere la storia passata (lo stato presente riassume tutta l'evoluzione passata). CATENE = stato discreto. Dell'evoluzione futura fa parte anche il tempo di soggiorno residuo $\phi_i(t) \sim EXP(-q_{ii})$. Se noto lo stato del processo nell'istante presente non riesco a determinare la distribuzione del tempo di soggiorno residuo, nel tempo in cui mi trovo, possiamo dire che NON riusciamo a determinare l'evoluzione futura. Evoluzione futura in termini probabilistici.

M/M/1. $|clienti\ in\ fila\ di\ attesa| = \underline{N_q(t) = 0}$. Supponiamo vi siano 0 clienti (informazione che ho sullo stato presente del processo). Proviamo a determinare la distribuzione del tempo di soggiorno residuo in questo istante, MA non so se ci sono clienti nel centro di servizio.

Supponiamo che vi sia un cliente nel centro di servizio (servitore occupato). Se termina quel servizio, si avrà una transizione di stato? NO. $\underline{\phi_0}(t)$ (0 clienti in fila di attesa). Se c'è 1 cliente, $\phi_0(t) \sim EXP(\lambda)$. Tempo di interarrivo residuo. Ma se non c'è nessuno, quella v.a. non sarà mica pari al tempo di interarrivo residuo. La transizione di stato si ha se arriva un cliente e ne arriva un altro. (Quindi se mettiamo insieme l'informazione invece riusciamo). Ma se includiamo quell'informazione torniamo alla precedente definizione di stato.

4.2 RETI DI CODE

Una rete dati di tipo switched può essere modellata mediante una rete di code. Queste code interagiscono tra di loro. Uno stream di pacchetti in partenza da una coda potrà andare a visitare un'altra coda dopo un merging con pacchetti provenienti da un'altra coda. [MERGING = POOLING]. Quindi una volta acceduti alla rete (superata la coda di accesso), si crea (si viene a creare) una correlazione tra i tempi di interarrivo dei pacchetti e quelli di trasmissione. Quindi viene meno l'ipotesi iniziale di indipendenza tra tempi di servizio e tempi di interarrivo. Adesso i tempi di servizio corrispondono ai tempi di trasmissione. Immaginiamo che i pacchetti arrivino alla rete con un processo di λ-POISSON. Nodo di accesso. Supponiamo che le lunghezze dei pacchetti siano v.a. equidistribuite statisticamente (i.d.), mutuamente indipendenti (con un certo valor medio), ed indipendenti dai tempi di interarrivo dei pacchetti al nodo di accesso. Tempi di trasmissione $\sim EXP(\cdot)$. Data una certa capacità, le lunghezza dei pacchetti sono collegate ai tempi di trasmissione. Supponiamo che tempi di elaborazione e propagazione siano trascurabili. FCFS disciplina di coda. Serve però un'ipotesi aggiuntiva riguardo la dimensione dei buffer ($d = +\infty$), ovvero illimitata.

Sistema reale: insieme dei due router con linee random (coda in serie). Il primo sistema è un M/M/1, per le ipotesi fatte. Si osservi il ramo che collega il primo centro di servizio con l'ingresso nel secondo buffer, e si effettuino delle analisi sulla eventuale dipendenza tra tempi di servizio e tempi di interarrivo. È evidente che le due grandezze sono in *CORRELAZIONE*. Non posso quindi parlare di coda M/M/1 per il secondo sottosistema. A parte che non sappiamo che processo degli arrivi sia. Lo sapremo mediante teorema di <u>BURKE</u>.

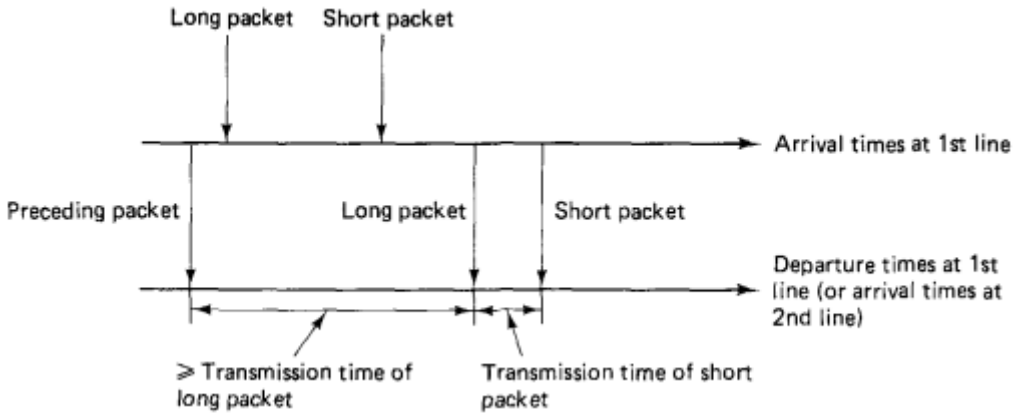


Figura 4.2: Timing Diagram of packet arrivals and departures completions in a system of two transmission lines in tandem

Supponiamo che ad un certo istante termini la trasmissione di un pacchetto precedente. Poi, arrivi un pacchetto grande alla prima coda, il quale, trovandola vuota e notando disponibile il centro di servizio, ivi vi entra subito e viene iniziato il servizio (trasmissione). Finita la trasmissione di questo pacchetto, possiamo osservare che il tempo di interarrivo alla seconda coda (distanza temporale tra l'istante di arrivo del precedente pacchetto e l'istante di arrivo di questo), è risultato maggiore del tempo di trasmissione del pacchetto grande. Questo sarebbe già sufficiente per compromettere l'indipendenza. Si noti comunque che, se durante la trasmissione arriva un piccolo pacchetto alla prima coda, trovando il centro di servizio occupato, è costretto ad attendere in fila di attesa. Quando finirà la trasmissione del pacchetto grande, il piccolo potrà essere trasmesso. Ma a questo punto osserveremo un tempo di interarrivo (tra il grande ed il piccolo pacchetto) alla seconda coda praticamente uguale al tempo di trasmissione del piccolo pacchetto. Notiamo che vale: $\underline{t_{IA} \geq t_T}$, ovvero il tempo di interarrivo al secondo router è maggiore uguale del tempo di trasmissione nel primo router (in particolare sarà più grande se e solo se il secondo pacchetto trova la prima coda vuota (incluso il centro di servizio) al suo arrivo). Dal momento che nelle reti dati tipicamente avremo una lunghezza costante dei pacchetti, saranno quindi correlati anche i tempi di interarrivo al secondo sottosistema con i suoi tempi di servizio, dal momento che le lunghezze dei pacchetti influenzano i tempi di trasmissione, e quindi di servizio, anche se capacità dei link è differente tra i due. $\Longrightarrow$ NO INDIPENDENZA. Leonard Kleinrock, il pioniere nella Ricerca delle Reti delle Code, ha risolto brillantemente questo problema. Andare a combinare (MERGING) su una certa coda flussi di pacchetti provenienti da altre code ha l'effetto di ripristinare l'indipendenza tra tempi di interarrivo a tempi di trasmissione. Se nel nostro esempio osserviamo al secondo router pacchetti di un altro router (nodo), non c'è più correlazione! Ripristino dell'ipotesi di INDIPENDENZA. La possibilità che accada questo evento dipende dal livello di comunicazione e di traffico! Più la rete è magliata (più merging di differenti flussi), e più è intenso il traffico, più è probabile che accadano eventi di questo tipo. La rete deve essere *DENSAMENTE CONNESSA* ed il traffico deve essere intenso. Cosicché per una rete densamente connessa e per intensità di traffico medio-alta, possiamo ritenere valida la cosiddetta ***APPROSSIMAZIONE DI INDIPENDENZA DI KLEINROCK***. Intensità di traffico medio-alta inoltre! Possiamo mantenere sempre valida l'indipendenza tra tempi di interarrivo e tempi di trasmissione (servizio).

4.2.1 RETE DI CODE APERTA

Rete di code: insieme di code interconnesse tra di loro. Supponiamo di avere il seguente scenario:

Stiamo rappresentando con una rete di code un sistema di Elaborazione.

Clienti/Servitori/Fila di Attesa. Questa è la topologia della rete di code. Stiamo modellando un sistema di elaborazione. I clienti corrispondono alle richieste di esecuzione dei job (la presenza di un clientre in μ_1 significa che si sta eseguendo quel job). I job stanno arrivando dall'esterno. Nodi alias stazioni di servizio. Il nodo 0 corrisponde al mondo esterno, dal quale possono arrivare i clienti. SINGLE CLASS OPEN. Una rete si dice aperta quando i clienti possono arrivare dall'esterno e fluire (partire) verso l'esterno. Per una CHIUSA (CLOSED) è il contrario di quella aperta. Per una CHIUSA il numero di clienti è costantè (*ISARITHMIC CONGESTION CONTROL*). $\exists M$ percorsi nella rete. Una rete di code, quando in essa i clienti arrivano esattamente se partono gli altri: 1 entra $\leftrightarrow$ 1 esce, può essere considerata chiusa. A volte possiamo volutamente limitare ad M il numero di pacchetti che transitano.

SINGLE CLASS OPEN. Tornando al nostro caso, i clienti arrivando dall'esterno nel nostro sistema a coda, possono visitare solo il nodo 1 $\iff$ $\underline{\underline{p_{01} = 1}}$. Ma potrebbero $\exists(p_{0i} \neq 1)$. Leggi delle probabilità:

$$[p_{11} + p_{10} + p_{12} + p_{13} + p_{14} = 1]$$

Similmente per gli altri nodi. Una volta visitato il nodo 2, il cliente potrà visitare soltanto il nodo 1. Questo vale anche per gli altri nodi 3-4! ($\iff$ $[p_{21} = p_{31} = p_{41} = 1]$. SINGLE CLASS = {singola classe di clienti}. Ma potrebbero esserci più classi di clienti. Per una rete di code con vari nodi definiamo:

$$[\underline{p_{ij} := ROUTING\ PROBABILITY}]$$

o PROBABILITÀ DI ROUTING. (collegate ai protocolli di instradamento). Si potrebbero definire delle probabilità per differenti classi. Per altri stream potrei scegliere delle differenti destinazioni. Si modellino i vari stream nella rete ($\exists K$ classi ad esempio) come delle "classi di clienti". Sia: $N = |nodi|$. Indichiamo con K il vettore riga: $\underline{K} := (K_1, K_2, \ldots, K_N)$, ove tale vettore rappresenta il numero di clienti NEI VARI NODI. Difatti, $K_i := |clienti\ al\ nodo\ "i"|$. E quindi: $K = \sum_{i=1}^{N} K_i := |clienti\ nella\ rete|$. Definiamo: $[m_i \geq 1] := |servitori\ paralleli\ al\ nodo\ "i"|$. Indichiamo con μ_i la velocità del singolo servitore al nodo "i" (In generale possiamo avere velocità differenti per il nodo "i"). Noi considereremo servitori alla stessa velocità. Introduciamo il concetto di [Soluzione in forma PRODOTTO].

p_{ij} = probabilità di routing $\implies$ $p_{0,j}$ = probabilità che un cliente in arrivo dall'esterno vada a visitare il nodo j. Similmente, $p_{i,0}$ = probabilità che un cliente in partenza dal nodo i, lasci la rete. Cosicché: $pi_{i,0} = 1 - \sum_{j=1}^{N} p_{ij}$. λ_{0i} = velocità di arrivo dei clienti dall'esterno al nodo "i". Indichiamo con λ la velocità totale di arrivo dei clienti dall'esterno $\iff$ $\lambda = \sum_{i=1}^{N} \lambda_{0i}$. La velocità totale sarà tale che: $\lambda_{0i} = \lambda p_{0i}$. Indicando con λ_i la velocità TOTALE di arrivo dei clienti al nodo i e con λ_j la velocità di partenza delle altre code, otteniamo:

$$[\lambda_i = \lambda_{0i} + \sum_{j=1}^{N} \lambda_j p_{ji}]$$

con $i = 1, 2, \ldots, N$. Dividendo membro a membro per λ otteniamo:

$$\frac{\lambda_i}{\lambda} = \frac{\lambda_{0i}}{\lambda} + \sum_{j=1}^{N} \frac{\lambda_j}{\lambda} p_{ji}$$

con $i = 1, 2, \ldots, N$. Queste sono le cosiddette EQUAZIONI DEL TRAFFICO. Definiamo allora:

Definition 32. ***Visit Ratio***
Definiamo il VISIT Ratio come:

$$[e_i := \frac{\lambda_i}{\lambda}]$$

Esso rappresenta il numero medio di visite al nodo "i" $\forall$ arrivo dall'esterno. Quante volte un cliente IN ARRIVO DALL'ESTERNO visita il nodo "i" prima di lasciare la rete. Ad esempio:

$$\lambda_i = 20 \; clienti/s, \; \lambda = 2 \; clienti/s, \; \frac{\lambda_i}{\lambda} = (10/2 = 5) \; clienti/s$$

Diamo ora un'altra definizione:

Definition 33. ***Service Demand***
Definiamo D_i service demand la quantità totale di servizio che in media un cliente richiede in un certo nodo:

$$\begin{cases} [e_i = p_{0i} + \sum_{j=1}^{N} e_j p_{ji}] \\ [D_i := e_i(\frac{1}{\mu_i})] \end{cases}$$

Si noti che D_i è una quantità temporale!

Consideriamo la seguente rete di code: Due code a singolo router in serie e supponiamo che il processo degli arrivi al nodo "1" sia di λ-POISSON, che i tempi di servizio dei clienti alla prima ed alla seconda coda siano v.a. $(\ldots) \sim EXP(\mu_1)$ per la prima e $(\ldots) \sim EXP(\mu_2)$ per la seconda, mutuamente indipendenti ed indipendenti dal processo degli arrivi alle code (tempi di interarrivo dei clienti alle code). QUESTE NON SONO RETI DATI! Ma generiche reti di code. Supponiamo ($d = +\infty$) e disciplina di coda FCFS. Con le ipotesi viste la prima coda è M/M/1. Riguardo alla seconda coda, il processo degli arrivi dei clienti alla seconda coda corrisponde al tempo di partenza (al processo delle partenze dei clienti del nodo 1). Quindi è una $\cdot$/M/1 (Il processo degli arrivi NON può essere considerato, descritto indipendentemente dal resto della rete). Processo delle partenze. Questo fatto ci fa comprendere l'importanza di caratterizzare i processi delle partenze dei clienti delle code. Teorema di BURKE che fa la caso nostro:

Theorem 16. ***Teorema di BURKE***
Si consideri un sistema a coda M/M/1 $\vee$ M/M/m $\vee$ M/M/($m = \infty$). Sia λ la velocità di arrivo dei clienti (processi degli arrivi dei clienti). Supponiamo di stare in condizioni di regime:

- *a) Il processo delle partenze è di POISSON, a velocità λ;*
- *b) $\forall$ istante t, |clienti nel sistema| NON dipende dalla sequenza delle partenze fino a t. Esso dipende SICURAMENTE dagli arrivi che ci sono stati fino a t, SICURAMENTE dai tempi di servizio dei clienti che SONO stati effettivamente serviti;*

Per dimostrarlo si dovrebbero utilizzare le CATENE DI MARKOV *REVERSIBILI* (Ragionare andando nel verso opposto come tempo $\iff$ guardando l'evoluzione a ritroso). Determinare la cosiddetta *OCCUPANCY DISTRIBUTION*, ove intendiamo la probabilità che vi siano K_1 clienti nella "coda 1" e K_2 clienti nella "coda 2" a regime (probabilità congiunta):

$$\Pr\{K_1 \textit{ clienti nella "coda 1"},\ K_2 \textit{ clienti nella "coda 2"}\} =$$
$$= \Pr\{K_1, K_2\} := \underline{\pi(K_1, K_2)}$$

Notazione con pedici anche possibile. Distribuzione congiunta dei clienti nelle VARIE CODE. Una rete di code si dice STABILE se $\forall$ sistema a coda di cui si compone è STABILE. CONDIZIONE DI STABILITÀ:

$$[(\rho < 1) \iff (\frac{\lambda}{\mu} < 1) \iff \lambda < \mu] \implies$$
$$\implies \begin{cases} \rho_1 = \dfrac{\lambda}{\mu_1} < 1 \iff \lambda < \mu_1 \\ \rho_2 = \dfrac{\lambda}{\mu_2} < 1 \iff \lambda < \mu_2 \end{cases}$$

Imposta la stabilità delle code e delle reti di code, utilizzo il teorema di BURKE. Sfruttiamo la parte a) del teorema. Processi degli arrivi in 2 di POISSON. Se guardiamo "2", isolatamente sarebbe un M/M/1 adesso (grazie a BURKE). Sfruttiamo adesso la parte b) di BURKE. Consideriamo le v.a. casuali che rappresentano il numero di clienti in "1" ed in "2":

$$\Pr\begin{cases} N_1(t) = K_1 \\ N_2(t) = K_2 \end{cases}$$

Abbiamo $\{N_1(t), N_2(t)\},\ \forall t$. Risulta:

$$N_1(t) \neq \cdot N_2(t)$$

grazie a Burke $\implies$ $N_1(t), N_2(t)$ sono v.a. indipendenti $\implies$

$$\pi_{K_1,K_2} := \pi(K_1, K_2) = \pi(K_1)\pi(K_2)$$

pari ovvero al prodotto delle due distribuzione marginali dei sistemi a coda visti in isolamento. Soluzione in FORMA PRODOTTO della OCCUPANCY DISTRIBUTION. In realtà i singoli sistema a coda NON stanno in isolamento! È una rete di code infatti. Riprendendo l'esempio delle M/M/1, abbiamo:

$$\{\lambda, \mu\} \implies \underline{\pi(K_1, K_2)} = [(1-\rho_1)\rho_1^{K_1}][(1-\rho_2)\rho_2^{K_2}]$$

dove abbiamo: $\{(\rho_1 = \frac{\lambda}{\mu_1}),\ (\rho_2 = \frac{\lambda}{\mu_2})\}$. Ove abbiamo semplicemente utilizzato le espressioni trovate per la M/M/1. Allo stesso risultato si poteva arrivare senza sfruttare Burke ed analizzando con approccio Markoviano le seguenti reti di code, utilizzando una definizione di stato bidimensionale (multi-indice) (al tempo t). Sfruttiamo l'EQ. DI BILANCIAMENTO TOTALE dei flussi per una CATENA ERGODICA. Dovremo però aggiungere ovviamente la condizione di normalizzazione. Si arriva a scrivere:

$$\pi_{K_1,K_2} = (1-\frac{\lambda}{\mu_1})(1-\frac{\lambda}{\mu_2})(\frac{\lambda}{\mu_1})^{K_1}(\frac{\lambda}{\mu_2})^{K_2} =$$
$$= \underline{[((1-\frac{\lambda}{\mu_1})(\frac{\lambda}{\mu_1})^{K_1} = \pi_1(K_1))]}\underline{[((1-\frac{\lambda}{\mu_2})(\frac{\lambda}{\mu_2})^{K_2} = \pi_2(K_2))]} = \underline{\pi(K_1)\pi(K_2)}$$

Proviamo a sostituirle nelle equazioni di bilanciamento.

Giustificazione matematica. $\underline{(K_1, K_2)}$. Quindi ai fini della Occupancy Distribution, la distribuzione congiunta è pari al prodotto delle distribuzioni marginali.

4.2.2 Feed-Forward (Reti di code ACICLICHE)

Tutte le reti di code che comprendono code del tipo: $\{\cdot/M/1, \cdot/M/m, \cdot/M/\infty\}$ le quali ricevono arrivi sia da altre code della rete, sia dall'esterno, secondo processi di POISSON, e che non permettono a nessun cliente di ritornare ad una coda già visitata (SENZA CICLI), sono risolubili in forma prodotto. ACICLICHE $\iff \nexists$ CICLI. Per come sono fatte queste reti di code i processi degli arrivi sono di POISSON alle varie CODE.

SEARCH ENGINE - MOTORI DI RICERCA

Si consideri una porzione di rete in cui sono presenti tre motori di ricerca organizzati gerarchicamente in due livelli (vedi figura): a livello più alto e' presente il motore MC, mentre a livello gerarchico più basso sono presenti i motori MA e MB. Gli utenti sono connessi al livello gerarchico più basso. Una richiesta di ricerca viene presentata da un generico utente al proprio motore. Questo, se dopo averla elaborata, non e' in grado di soddisfarla, la reindirizza verso il motore MC che e' sicuramente in grado di risolverla. Si assuma che:

- i) le richieste di ricerca da parte degli utenti del gruppo A siano presentate in accordo ad un processo di Poisson con parametro λ_A req/min;
- ii) le richieste di ricerca da parte degli utenti del gruppo B siano presentate in accordo ad un processo di Poisson con parametro λ_B req/min;
- iii) in ogni motore un unico processore elabori le richieste in modalità FCFS e con tempi che sono v.c. indipendenti distribuite secondo una ddp esponenziale negativa a valor medio pari, rispettivamente per i tre motori M_A, M_B, e M_C, a T_A sec, T_B sec e T_C sec;
- iv) una richiesta sia soddisfatta con probabilità α da un motore di livello basso;
- v) sia trascurabile il tempo necessario ad inoltrare una richiesta al motore M_C.

Calcolare:

- a) il fattore di utilizzazione del processore nel motore M_C;
- b) il tempo medio di risoluzione di una richiesta;
- c) il tempo medio di risoluzione di una richiesta da parte degli utenti del gruppo A.

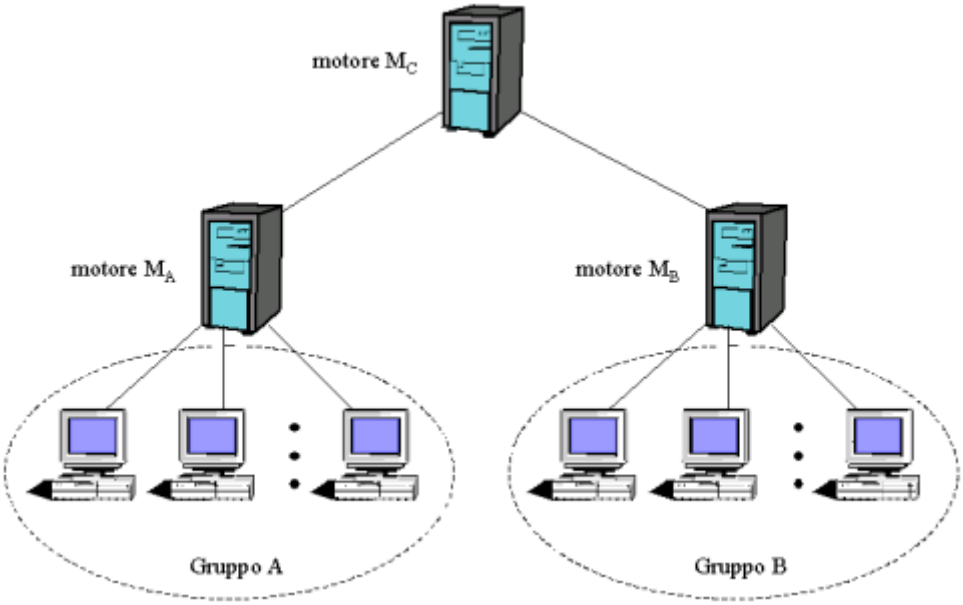


Figura 4.3: Hierarchical Search Engines

Risoluzione:

Tre motori di ricerca organizzati gerarchicamente in tre livelli. Gli utenti sono connessi al livello gerarchico più basso. $\{\lambda_A, \lambda_B\}$ req/min, processi di POISSON. Richieste in modalità FCFS. Per quanto concerne i tempi di servizio, siamo dinanzi delle ddp esponenziali con i seguenti tempi di servizio medi, rispettivamente: $\{T_A, T_B, T_C\}s$. M_C =? (fattore di utilizzazione). C'è da aggiungere un'ipotesi sulle dimensioni delle file di attesa: File di attesa illimitate $\iff$ $(d = +\infty)$. Se qualcuna fosse un collo di bottiglia ci sarebbe scritta la dimensione ($\iff$ $d < +\infty$). Reti di code ACICLICHE. Processo degli arrivi dall'esterno di POISSON. Processo degli arrivi alle varie code di POISSON. Applicabile/applicato il teorema di BURKE. Abbiamo: $\{\mu_A = \frac{1}{T_A},\ \mu_B = \frac{1}{T_B},\ \mu_C = \frac{1}{T_C}\}$. Parametri della ddp esponenziale negativa. Popolazione illimitata $\iff$ $(e = +\infty)$. Processo degli arrivi di POISSON. Rete ACICLICA $\iff$ feedforward network. Mi trovo dinanzi una rete di code. I processi degli arrivi sono di POISSON. Le reti A e B sono delle M/M/1. Viste in isolamento sono delle M/M/1. $\{K_A,\ K_B,\ K_C\}$ ci servono per definire la Occupancy Distribution (Burke + Splitting/Pooling). Code M/M/1 viste in isolamento $\iff$ grazie alla Soluzione in forma prodotto. Dobbiamo applicare le EQUAZIONI DEL TRAFFICO. La velocità di arrivo al sistema M_C sarà:

$$\lambda_A(1-\alpha) + \lambda_B(1-\alpha) = (1-\alpha)(\lambda_A + \lambda_B) := \lambda_C$$

I clienti NON si perdono! Se λ_A arrivano nel primo sistema, λ_A ne partiranno! Lo stesso dicasi per M_C. Per la STABILITÀdella rete di code dobbiamo imporre che $\forall$ sistema vi sia STABILITÀ. Stabilità di ogni singolo sistema a coda:

$$\begin{cases} \rho_A = \dfrac{\lambda_A}{\mu_A} < 1 \iff \lambda_A < \mu_A \\ \rho_B = \dfrac{\lambda_B}{\mu_B} < 1 \iff \lambda_B < \mu_B \\ \rho_C = \dfrac{\lambda_C}{\mu_C} < 1 \iff \lambda_C < \mu_C \end{cases}$$

Analogamente... tre motodi di ricerca. Livello didattico. Nel calcolare λ_C abbiamo anche risolto la prima domanda: $\rho_C = \frac{\lambda_C}{\mu_C}$, ovvero il FATTORE DI UTILIZZAZIONE del servitore nel nodo M_C.

Tempo medio di risoluzione di una richiesta. L'arrivo nella rete di code di una richiesta corrisponde all'invio di una richiesta da parte di un utente. Tempo medio di risoluzione di una richiesta = tempo medio di permanenza nell'intero sistema. Clienti dall'esterno + Clienti in partenza ← risoluzione di una richiesta. Si applichi Little:

$$[\bar{N}_{TOT} = \lambda_{TOT}\bar{R}] \implies \bar{R} = \frac{\bar{N}_{TOT}}{\lambda_{TOT}}$$

$\bar{R}$ è quello che mi interessa. Sappiamo che: $\underline{\lambda_{TOT} = \lambda_A + \lambda_B}$, ovvero la velocità totale di arrivo dei clienti dall'esterno. Per un M/M/1 $\cdot(\lambda, \mu)$ il numero medio di clienti è: $(\bar{N} = \frac{\lambda}{\mu - \lambda})$. Lo dobbiamo però applicare ai vari sottosistemi. Somma dei numeri medi di clienti (a regime ovviamente):

$$[N_{TOT} = \bar{N}_A + \bar{N}_B + \bar{N}_C]$$

sapendo che:

$$\begin{cases} \bar{N}_A = \dfrac{\lambda_A}{\mu_A - \lambda_A} \\ \bar{N}_B = \dfrac{\lambda_B}{\mu_B - \lambda_B} \\ \bar{N}_C = \dfrac{\lambda_C}{\mu_C - \lambda_C} \end{cases}$$

Quindi abbiamo:

$$\bar{R} = \frac{N_{TOT}}{(\lambda_A + \lambda_B = \lambda_{TOT})}$$

Per sapere invece il tempo medio di risoluzione di una richiesta da parte degli utenti del gruppo A $\iff$ R_A =?, sapendo che per un M/M/1 vale:

$$\begin{cases} \bar{N} = \dfrac{\lambda}{\mu - \lambda} \\ \bar{T} = \dfrac{1}{\mu - \lambda} \end{cases}$$

Allora si riconosce subito la media di una v.a.:

$$\bar{R}_A = \alpha[\frac{1}{\mu_A \lambda_A}] + (1 - \alpha)[\frac{1}{\mu_A \lambda_A} + \frac{1}{\mu_C - \lambda_C}] =$$

$$= \frac{\alpha}{\mu_A - \lambda_A} + \frac{1}{\mu_A - \lambda_A} - \frac{\alpha}{\mu_A - \lambda_A} + \frac{1}{\mu_C - \lambda_C} - \frac{\alpha}{\mu_C - \lambda_C} =$$

$$= [\frac{1}{\mu_A - \lambda_A} + \frac{(1-\alpha)}{\mu_C - \lambda_C}]$$

Legittimo applicare Little al sottosistema M_C. Potremmo in realtà tentare un altro approccio. Potrei applicare Little a tutto il sistema ma considerando il numero di clienti del gruppo A. Sappiamo che:

$$\bar{N}_C = \frac{\lambda_C}{\mu_C - \lambda_C}$$

Ma non stiamo diversificando i gruppi di clienti. La frazione di clienti A in M_C è: $\frac{\lambda_A(1-\alpha)}{\lambda_C}$. Se facciamo questo rapporto, questa quantità corrisponderà proprio alla frazione dei clienti del gruppo A in M_C. Possiamo quindi scrivere la formula per sapere il numero medio di clienti del gruppo A in tutta la rete di code:

$$\bar{N}_{TOT,A} =? = \frac{\lambda_A}{\mu_A - \lambda_A} + (\frac{\lambda_A(1-\alpha)}{\lambda_C})(\frac{\lambda_C}{\mu_C - \lambda_C}) = \frac{\lambda_A}{\mu_A - \lambda_A} + \frac{\lambda_A(1-\alpha)}{\mu_C - \lambda_C} =$$

$$= \lambda_A[\frac{1}{\mu_A - \lambda_A} + \frac{(1-\alpha)}{\mu_C - \lambda_C}]$$

Dobbiamo ora applicare Little:

$$\frac{\bar{N}_{TOT,A}}{\lambda_A} = \bar{R}_A = [\frac{1}{\mu_A - \lambda_A} + \frac{(1-\alpha)}{\mu_C - \lambda_C}]$$

avendo considerato come velocità di arrivo solo λ_A, ovviamente.

4.2.3 Reti di code CICLICHE

Fino ad ora abbiamo visto le reti ACICLICHE $\iff$ Non permettiamo ad un cliente di visitare una coda già visitata. L'assenza di cicli è fondamentale per preservare la caratteristica di POISSON. Ciononostante esistono delle reti di code con CICLI per le quali continua comunque a valere la Soluzione in forma prodotto per la Occupancy Distribution. Reti di code con CICLO. Parliamo di rete aperta naturalmente. Due code $\cdot$/M/1 collegate in serie e con CICLO. Stiamo facendo l'ipotesi che i tempi di servizio siano v.a. distribuite esponenzialmente, mutuamente indipendenti e statisticamente indipendenti dai tempi di interarrivo dei clienti, distribuiti secondo λ-POISSON.

JACKSON NETWORK

JACKSON NETWORK. $\{p := p_1,\ p_2 := 1 - p\}$. Ci favorisce la possibilità di calcolare la Occupancy Distribution sempre utilizzando la Soluzione in forma prodotto. Occupancy Distribution utilizzando una soluzione di stato bidimensionale $\{N_1(t),\ N_2(t)\}$. Processo stocastico. Due code $\cdot$/M/1. Fila di attesa illimitata $\iff$ $(d = +\infty)$. Altrimenti non potrei parlare di M/M/1. Le ACICLICHE sono un sottoinsieme delle reti di JACKSON. Una rete di JACKSON senza cicli è una rete ACICLICA. Vale:

$$\lambda_1 = \lambda + \lambda_2\alpha \overset{[\lambda_2=\lambda_1]}{=} \lambda + \lambda_1\alpha \implies$$

$$\implies [\lambda_1 = \frac{\lambda}{1-\alpha} = \lambda_2]$$

Dobbiamo imporre anche qui la STABILITÀ dei singoli sottosistemi:

$$\begin{cases} \rho_1 := \dfrac{\lambda_1}{\mu_1} < 1 \iff \lambda_1 < \mu_1 \\ \rho_2 := \dfrac{\lambda_2}{\mu_2} < 1 \iff \lambda_2 < \mu_2 \end{cases}$$

Adottiamo l'approccio precedente. Dobbiamo srivere le EQ. di bilanciamento totale dei flussi. Qui ovviamente NON possiamo applicare BURKE. I processi degli arrivi non sono di POISSON. Abbiamo scritto le equazioni di bilanciamento totale dei vari flussi. Dovrei risolvere il sistema aggiungendo la condizione di NORMALIZZAZIONE. Ricordiamo che: $|stati| = +\infty$. Proviamo ad utilizzare questa distribuzione:

$$[\pi_{K_1,K_2} = \underline{(1 - \frac{\lambda_1}{\mu_1})(\frac{\lambda_1}{\mu_1})^{K_1}(1 - \frac{\lambda_1}{\mu_2})(\frac{\lambda_2}{\mu_2})^{K_2}}]$$

Ove i termini sottolineati sommano ad 1. Si può verificare che, dato che sostituite nella eq. di bilanciamento totale dei flussi essa è risolta $\iff$ è proprio quella la distribuzione. Prodotto delle distribuzioni marginali dei singoli sottosistemi, considerando che la loro distribuzione degli arrivi sia di POISSON, anche se in realtà non lo è! Sono sicuro che gli arrivi NON sono distribuiti secondo POISSON, ma i singoli sottosistemi, ai fini del calcolo della Occupancy Distribution, è proprio come se fossero degli M/M/1 in ISOLAMENTO! λ-POISSON autorizzati. Esempio di reti di code per le quali vale anche la Soluzione in forma prodotto $\rightarrow$ JACKSON NETWORK

Dobbiamo formalizzare in generale queste particolari classi. Reti di Jackson, dal nome del ricercatore che ha studiato queste particolari reti di code. Ipotesi:

- -) Ci sono N nodi: $|nodi| = N$;
- -) $\exists!$ classe di clienti. Non stiamo considerando più classi al momento;
- -) RETE APERTA cosicché i clienti possono arrivare dall'esterno e partire verso l'esterno;
- -) λ_{0i}-POISSON processi degli arrivi al nodo i dall'esterno. λ_{0i} può anche essere 0, ma affinché la rete sia aperte deve valere:

  $$\exists i \mid (\lambda_{0i} > 0) \neq 0$$

  Se $\lambda_{0i} = 0$, la coda i non prevede arrivi dall'esterno. Ricordiamo alcuni fatti:

  - λ: velocità totale di arrivo dei clienti dall'esterno;
  - $\lambda_{0i} = \lambda p_{0i}$: velocità totale di arrivo dei clienti dall'esterno al nodo i;

  Se $\forall i,\ \lambda_{0i} = 0 \implies$ la rete è CHIUSA!

- -) Supponiamo che le file di attesa siano illimitate $\iff$ $(d_i = +\infty\ \forall i)$. Al nodo i ci sono m_i servitori identici, dove $[m_i \geq 1]$. Velocità di servizio μ_i. μ_i sia la velocità di servizio del generico servitore al nodo i. $(m_i = +\infty)$ è un caso contemplato. $\{\cdot/\mathrm{M}/1,\ \cdot/\mathrm{M}/\mathrm{m},\ \cdot/\mathrm{M}/\infty\}$;
- -) i tempi di servizio sono distribuiti esponenzialmente, mutuamente indipendenti e statisticamente indipendenti dal processo degli arrivi dei clienti alla coda (considerando sempre una certa coda i);

Sostanzialmente, stiamo considerando $\cdot/\mathrm{M}/m_i$, con $\{\frac{1}{\mu_i},\ \mu_i\ PAR\}$. Invece di considerare m_i router identici nel sistema a coda i, possiamo ipotizzare la presenza di un unico servitore che operi con:

$$\mu_i(K_i) = \begin{cases} K_i\mu_i,\ K_i < m_i \\ \underline{m_i\mu_i},\ K_i \geq m_i \end{cases}$$

È una velocità variabile governata da questa legge. m_i servitori nel centro di servizio. La massima capacità di servizio possibile è per forza di cose il termine sottolineato. Possiamo considerare anche un servitore a velocità variabile, la cui velocità varia col numero di clienti presenti (nell'intero sistema a coda). Parliamo di *LOAD DEPENDENT SERVICE-RATE* (LDSR). Discipline di coda per le varie code FCFS. $\{\cdot/\mathrm{M}/m_i)$, dove $(m_i \overset{CBE}{=} +\infty)$. Varie casistiche:

$$\begin{cases} m_i = 1 \\ m_i > 1 \\ m_i \leq +\infty \end{cases}$$

Alla fine del servizio presso la coda i, un cliente va dalla coda "i" alla coda "j" con probabilità relativa di routing (un cliente, lasciando la coda "i", visita la coda "j" con probabilità p_{ij}), dove $i, j = 1, 2,\ \ldots,\ (N = |nodi|)$, e potrà lasciare la rete con probabilità: $\underline{p_{i0}} = 1 - \sum_{i=1}^{N} p_{ij},\ i = 1, 2,\ \ldots,\ N$, ove il termine sottolineato rappresenta la probabilità di routing di un cliente da "i" verso l'esterno. PROBABILITÀ DI ROUTING. Ricordiamo che il nodo 0 è l'esterno.

In una rete di Jackson le probabilità di routing sono uguali $\forall$ classe. Nelle BCMP generalmente sono differenziate in base alle varie classi. Ricordiamo le EQUAZIONI DEL TRAFFICO:

$$[\lambda_i = \lambda_{0i} + \sum_{j=1}^{N} \lambda_j p_{ji}]$$

Adesso consideriamo il nodo i per il quale $\exists i \mid m_i < +\infty$. m servitori. Numero di router (servitori) finito. $m_i = |servitori|$ (in parallelo). Possiamo calcolare il fattore di utilizzazione del generico servitore: $[\rho_i = \frac{\lambda_i}{m_i\mu_i}]$. "i" con m_i servitori, tale per cui $(m_i > 1) < +\infty$. $\rho_i < 1$ per la stabilità della coda "i" (fattore di utilizzazione del generico servitore quando il carico è equamente distribuito). Dobbiamo imporre che:

$$\forall i\ |m_i < +\infty \implies \rho_i < 1$$

Notiamo infatti che se $m_i = +\infty$ allora la STABILITÀ è sempre soddisfatta. $\{\cdot/\mathrm{M}/1$, $\cdot/\mathrm{M}/\mathrm{m}$, $\cdot/\mathrm{M}/\infty\}$. Tre tipologie di sistemi a coda per le JN. $\underline{K} = (K_1, K_2,\ \ldots,\ K_N) \in \mathbb{R}^{N\times 1}$. Indichiamo con K il vettore di stato, che considera il numero di clienti nelle varie code della JN. Mi interessa valutare la Occupancy Distribution: $\pi(K_1, K_2,\ \ldots,\ K_N)$ in condizioni di regime. Vi siano: $\{K_1$ clienti al nodo "1", K_2 clienti al nodo "2", $\ldots$, K_N clienti al nodo "N"$\}$. Per queste tipologie di reti di code la OD è esprimibile mediante Soluzione in forma prodotto $\implies$

$$\pi(K_1, K_2,\ \ldots,\ K_N) = \pi(K_1)\pi(K_2)\ \ldots\ \pi(K_N)$$

Nelle JN i processi degli arrivi NON sono di POISSON. Possiamo però considerare i sistemi in isolamento.

Exercise

Un servizio di rete si basa su due server. Il processo stocastico secondo il quale si susseguono le richieste di esecuzione di un "job" da parte degli utenti può essere ben modellato da un processo di Poisson a velocità λ. In figura 1 è riportato lo schema che illustra la modalità di esecuzione dei job. Si noti come, dopo aver utilizzato il server #1, con probabilità p1 un job lascia il sistema (job eseguito!), con probabilità $p2 = 1 - p1$, invece, un job utilizza il server#2. Utilizzato il server#2, un job viene nuovamente inserito nella coda del server#1.

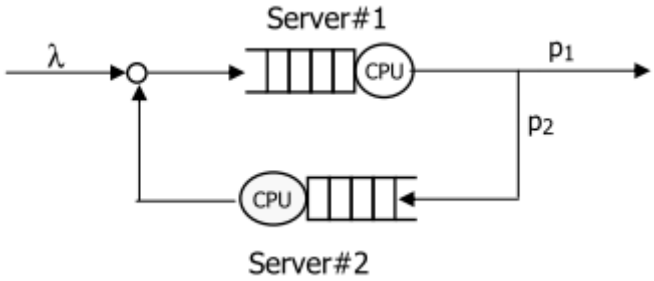


Figura 4.4: Jackson Network

Si assuma che i singoli intervalli di tempo durante i quali viene utilizzato la CPU del server#1 ed i singoli intervalli di tempo durante i quali viene utilizzata la CPU del server#2 siano descritti da variabili casuali indipendenti e distribuite esponenzialmente a valor medio, rispettivamente, $\frac{1}{\mu_1}$ e $\frac{1}{\mu_2}$. Si assuma, inoltre, che i sistemi a coda in figura abbiano una capacità illimitata ed una disciplina di servizio di tipo FCFS. Si calcoli:

- a) Il fattore di utilizzazione della CPU del server#1;
- b) il valore medio del tempo di esecuzione di un job, a regime;
- c) il valore medio del tempo totale di CPU del server#1 richiesto da un job, a regime;
- d) il valore medio del tempo totale di CPU del server#2 richiesto da un job, a regime.

Risoluzione:
(λ-POISSON) processi degli arrivi. p_1 che il job sia eseguito; $p_2 = 1 - p_1$ che utilizzi il server 2. Se il job utilizza il server 2 ritorna in feedback all'ingresso di "1". $\{\{\lambda\}, \{\mu_1, \mu_2\}\}$. $(d = +\infty)$, FCFS. Ci troviamo dinanzi una JN con code $\cdot$/M/1. Ma potrebbero anche essere $\{\cdot/M/m, \cdot/M/\infty\}$. M/M/1 in isolamento ai fini della OD. Definiamo con λ_1 velocità totale di arrivo dei clienti al nodo "1". Analogamente per "2", con λ_2. Applicando le equazioni del traffico troviamo:

$$\begin{cases} \lambda_1 = \lambda + \lambda_2 \\ \lambda_2 = \lambda_1 p_2 \end{cases} \implies \begin{cases} \lambda_1 = \dfrac{\lambda}{1 - p_2} \\ \lambda_2 = \dfrac{\lambda}{(1 - p_2 = p_1)} p_2 \end{cases}$$

Dobbiamo imporre le Stabilità dei sottosistemi: $\implies$

$$\begin{cases} [\underline{\rho_1 = \frac{\lambda_1}{\mu_1} = \frac{\lambda}{p_1\mu_1}}] < 1 \\ [\underline{\rho_2 = \frac{\lambda_2}{\mu_2} = \frac{\lambda p_2}{p_1\mu_2}}] < 1 \end{cases}$$

ove i termini sottolineati rappresentano il fattore di utilizzazione del server 1 e quello del server 2, rispettivamente. Little. $\bar{N} = \bar{N}_1 + \bar{N}_2$. Dove:

$$\begin{cases} \bar{N}_1 = \frac{\lambda_1}{\mu_1 - \lambda_1} = \frac{\lambda}{p_1(\mu_1 - \frac{\lambda}{p_1})} = \frac{\lambda}{p_1\mu_1 - \lambda} \\ \bar{N}_2 = \frac{\lambda_2}{\mu_2 - \lambda_2} = \frac{\lambda p_2}{p_1(\mu_2 - \frac{\lambda p_2}{p_1})} = \frac{\lambda p_2}{p_1\mu_2 - \lambda p_2} = \frac{\lambda}{\frac{p_1}{p_2}\mu_2 - \lambda} \end{cases}$$

Ricordiamo che λ è la velocità totale di arrivo dei clienti dall'esterno. Dobbiamo quindi applicare Little: $\Longrightarrow$

$$\bar{R} = \frac{\bar{N}_1 + \bar{N}_2}{\lambda} = (\frac{1}{p_1\mu_1 - \lambda} + \frac{1}{\frac{p_1}{p_2}\mu_2 - \lambda})$$

Quindi questo è il tempo medio di risposta. Dal punto di vista del tempo medio di risposta = tempo medio di permanenza, vi è un'equivalenza a due M/M/1 posti in serie, con velocità di servizio rispettivamente $\mu_1' = p_1\mu_1$ e $\mu_2' = \frac{p_1}{p_2}\mu_2$. RITARDO MEDIO il MEDESIMO, ma la distribuzione $\Pr\{R \leq t\}$ è DIFFERENTE!!

Calcoliamo le Service demand per i due nodi:

$$\begin{cases} D_1 = (\frac{1}{\mu_1})(e_1 = \frac{\lambda_1}{\lambda}) = \frac{1}{\mu_1}\frac{\lambda_1}{\lambda} = \frac{1}{\mu_1}\frac{\lambda}{p_1\lambda} = \frac{1}{\mu_1 p_1} \\ D_2 = (\frac{1}{\mu_2})(e_2 = \frac{\lambda_2}{\lambda}) = \frac{1}{\mu_2}\frac{\lambda p_2}{p_1}\frac{1}{\lambda} = \frac{1}{\mu_2}\underline{\frac{p_2}{p_1}} \end{cases}$$

Dove e_i rappresenta il numero medio di visite dall'esterno al nodo "i". Nel caso "2" ad esempio, il termine sottolineato indica $e_2 = \frac{p_2}{p_1}$. Notiamo che il nodo "1" verrà visitato sempre una volta in più prima di lasciare la rete: $+1 \implies \frac{p_2}{p_1} + 1 = (\frac{1}{p_1})$. Potrei considerare: ξ il numero totale di visite al server 1. $i-1$ volte visiti il nodo "2", e la volta successiva il cliente lascia la rete. Se $p_1 = \Pr\{si\ lasci\ la\ rete\}$, allora vale:

$$\mathbb{E}[\xi] = \sum_{i=1}^{+\infty} i \Pr\{\xi = i\} = \sum_{i=1}^{+\infty} i p_2^{i-1} p_1 =$$

$$= p_1 \sum_{i=1}^{+\infty} i p_2^{i-1} = p_1 \frac{1}{(1-p_2)^2} = p_1 \frac{1}{p_1^2} = \underline{\frac{1}{p_1}}$$

(INDEPENDENT BERNOULLI TRIALS).

4.2.4 Reti BCMP

Reti di code {CICLICHE, ACICLICHE, JACKSON}. Per queste tipologie di code si ha una sola classe di clienti. Rete dati modellabile come rete di code, ma bisognerebbe differenziare le varie classi! $p_{ij} = \cdot(stream\ di\ pacchetti)$. Indici di prestazioni: {ritardo medio}. Reti di code BCMP e teorema BCMP. Va a definire una classe di reti di code per le quali

vale ancora la Soluzione in forma prodotto per la Occupancy Distribution. Ma in questo caso i fattori del prodotto non corrispondono necessariamente alle distribuzioni marginali. Ma i fattori si ottengono a partire dall'analisi dei vari sistemi a coda visti in isolamento. Le BCMP comprendono le reti di JACKSON e quindi anche le ACICLICHE. C'è la possibilità di avere più classi di clienti R.

- -) $R = |classi|$. Le classi possono essere APERTE o CHIUSE.
  - **APERTE**: i clienti al loro interno possono partire dall'esterno ed arrivare dall'esterno;
  - **CHIUSE**: altrimenti

  La BCMP è detta *MIXED* se alcune classi sono aperte ed altre chiuse;
- -) class switching possibile. Possibile che un cliente appartenente ad una certa classe, partendo da un nodo, visiti un certo nodo cambiando la sua classe;
- -) $N = |nodi|$. $p_{ir,js},\ r, s = 1, 2,\ \ldots,\ (R = |classi|),\ i, j = 1, 2,\ \ldots,\ N = |nodi|$ di reti di code. Si definirà quindi una *matrice di routing*:

  Definition 34. ***Matrice di Routing***

  Per indicare le probabilità che un cliente appartenente ad una certa classe, partendo da un nodo, visiti un certo nodo cambiando la sua classe, si definisce:

  $$\underline{\underline{P}} := [p_{ir,js}]$$

  Tipicamente è una matrice SPARSA. A rigore sarebbe in realtà un tensore quadridimensionale. Se togliamo il class switching il tensore si ridurrebbe a 3 dimensioni.
- -) Per quanto concerne la disciplina di coda, sino ad adesso abbiamo visto la FCFS, adesso ne sono consentite altre: {{PS, LCFS-PR, IS}, FCFS}. Ne spieghiamo qualcuna:
  - **PS** = *processor sharing*: tutti i clienti sono serviti contemporaneamente, con time slicing;
  - **LCFS-PR** = *Last-Come-First-Served with Preemptive-Resume*: Se un cliente viene servito ed arriva un altro cliente, vi è l'interruzione del servizio corrente ed il cliente corrente va in fila di attesa. Rientrerà nel servizio quando $\nexists$ clienti nel sistema a coda;
  - **IS** = *Infinite Server*: Come suggerisce lo stesso nome, prevede INFINITI (∞) servitori! $\iff$ ($m = +\infty$) Come nell'M/M/∞.

Vi è un'altra differenza (caratteristica aggiuntiva) riguardo la distribuzione dei tempi di servizio. Fino ad adesso abbiamo visto la distribuzione esponenziale. Adesso è permessa qualsiasi distribuzione purché la trasformata di Laplace della PDF sia razionale. COX ha dimostrato che qualunque distribuzione di v.a. si può approssimare sufficientemente bene mediante le seguenti reti di stadi: abbiamo una Rete di Stadi esponenziali. Il tempo di permanenza di un cliente in μ_i è distribuito esponenzialmente. Possiamo associare delle variabili casuali $\{\eta_1, \eta_2,\ \ldots,\ \eta_n\}$ che sono tutte distribuite esponenzialmente $\iff$ $\eta_i \sim EXP(\mu_i)\ \forall i = 1,\ \ldots,\ n$, ed indicano il tempo di permanenza in ciascuno stadio i-esimo. $\{a_i, b_i\}$ sono delle probabilità. La distribuzione associata all'attraversamento (al tempo di attraversamento) dell'intero sistema si chiama

distribuzione di COX e mediante essa possiamo approssimare (giocando opportunamente con il numero di stadi) sufficientemente bene una qualsiasi distribuzione. A questo punto il vincolo della trasformata di Laplace razionale della PDF non è più limitativo dal momento che la distribuzione di COX ha una TL della PDF razionale (PDF associata alla v.a. tempo di attraversamento).

Quindi possiamo pensare ad un unico servitore che rappresenti tutto il sistema. Il servitore è unico! Il servizio parte quando entra nel sistema e finisce quando il cliente esce dall'ultimo stadio. Solo alla fine un altro cliente va al servizio. In alcuni casi è ammessa una velocità del servizio che dipenda dal numero di clienti nella coda (M/M/m), o dal numero di clienti di una CERTA CLASSE nella coda.

(Probabilità di routing). La disciplina NON è a priorità! Ancora non la stiamo esaminando (interruzioni e senza interruzioni). Per queste reti che prevedono le forme prodotto NON è possibile. Consideriamo il caso di rete BCMP aperta (*OPEN-BCMP*). Vediamo che i processi degli ARRIVI sono ammesssi in questa rete. Consideriamo due casi, due possibilità:

- *CASO 1*: i clienti arrivano alla rete da un'unica sorgente, con tempi di interarrivo indipendenti e distribuiti esponenzialmente. Il Rate di arrivi (λ) può dipendere dal numero di clienti K nella rete $\iff \lambda = \lambda(K)$. Se $[\lambda \neq \lambda(K)]$ (si riferisce al rate di arrivo), avremo un processo di POISSON omogeneo. Un cliente in arrivo visiterà il nodo i in classe r con probabilità $p_{0,ir}$ (giungendo dall'esterno). Ovviamente dovrà valere il seguente vincolo:

$$[\sum_{i=1}^{N}\sum_{r=1}^{R} p_{0,ir} = 1]$$

- *CASO 2*: Se non facessi delle semplificazioni dovrei trattare delle sottocatene di Markov ergodiche. Eviteremo quindi il class switching. Supponiamo che non vi sia CS quindi. In tal caso si ha un processo degli arrivi $\forall$ classe di clienti. La velocità di arrivo λ_r (indicizzata su r) potrà dipendere da $K_r \implies \lambda_r = \lambda_r(K_r),\ r = 1, 2, \ldots, R$. Se $\lambda_r \neq \lambda_r(K_r)$ avremo un processo di POISSON omogeneo per la classe r.

Con le BCMP si può benissimo modellare un *J2EE*, con Studio di Affidabilità e Disponibilità annesso. Quindi nel primo caso abbiamo un unico processo degli arrivi, mentre nel secondo caso non abbiamo class switching ed abbiamo un processo degli arrivi differente $\forall$ classe della rete. $K_r = |clienti\ della\ classe\ r|$. Tipi di nodi ammessi nella BCMP: Nodi: {type 1, type 2, type 3, type 4}.

- type 1: FCFS node. Coda con tempi di servizio distribuiti esponenzialmente con lo stesso valor medio (SAME $\frac{1}{\mu_i}$) per tutte le classi. La velocità di servizio (μ_i) può dipendere dal numero di clienti totali nella coda (nel sistema a coda). LOAD DEPENDENT SERVICE-RATE (LDSR) possibile. Può servire per modellare dispositivi I/O, HDD. Lo utilizzeremo per modellare tempi di accodamento e trasmissione dei router. Considereremo quindi: $\{\cdot/\mathrm{M}/m_i$, FCFS$\}$;

- type 2: PS node (processor sharing). Utilizzato per modellare le CPU. Code con servitore unico e disciplina PS. Tutti i clienti sono serviti contemporaneamente. Sono possibili distribuzioni dei tempi di servizio diverse per classi diverse. Differenza rispetto al type 1. E qui non ci deve essere per forza l'esponenziale {DSD for DSC}. Vale sempre il vincolo sulle TL della PDF dei tempi di servizio. La velocità di servizio per una classe può dipendere dal numero di clienti nel nodo od anche dal numero di clienti di quella classe nel nodo. (Anche qui LDSR possibile);

- type 3: Detto IS (*Infinite Server*) node oppure *Delay Node*, perché solitamente utilizzato per modellare il ritardo (ritardo di elaborazione). Tutto quello scritto per il type 2 vale anche per il type 3. Code del tipo: $\cdot/\mathrm{G}/\infty$. Sempre un servitore a disposizione. In fila di attesa NON c'è mai nessuno. (Infiniti servitori $\implies \nexists$ fila di attesa). È come se non ci fosse alcuna disciplina;

- type 4: LCFS-PR (LCFS with Preemptive Resume). Coda con un unico servitore con quella disciplina di servizio. Contempliamo quindi le {$\cdot/\mathrm{G}/1$ - LCFS-PR}. Unico servitore con quella disciplina di coda.

Teorema BCMP

Andiamo direttamente ad una versione semplificata del teorema BCMP (*Baskett, Chandy, Muntz and Palacios*). Versione semplificata. Serve per modellare le reti di dati. Consideriamo BCMP (reti di code) aperte con load independent arrival rate. Quindi sostanzialmente processi di arrivo omogenei. Reti BCMP aperte con $[\lambda \neq \lambda(K)] \implies$ processi degli arrivi omogenei sostanzialmente.

Consideriamo che le velocità di servizio siano tali che $[\mu_i \neq \mu_i(K)]$. Casi ovviamente estendibili. Consideriamo:

$$\begin{cases} N = |nodi| \\ R = |classi| \\ K_i = |clienti\ totali\ nel\ nodo\ i| \\ K_{ir} = |clienti\ di\ classe\ r\ nel\ nodo\ i| \\ \underline{K} = (K_1, K_2,\ \ldots,\ K_N) \\ \begin{cases} K_i = \displaystyle\sum_{r=1}^{R} K_{ir} \\ K = \displaystyle\sum_{i=1}^{N} K_i = \sum_{i=1}^{N}\sum_{r=1}^{R} K_{ir} \end{cases} \end{cases}$$

Con queste quantità, la Occupancy Distribution è uguale a:

$$\Pr\{\underline{K}\} = \pi(K_1, K_2,\ \ldots,\ K_N) = \prod_{i=1}^{N} \pi_i(K_i)$$

questa è la OD per questa OPEN-BCMP. $\pi_i(K_i)$ è caratterizzato in questo modo:

$$\pi_i(K_i) = \begin{cases} \underline{(1-\rho_i)\rho_i^{K_i}},\ type\ 1,\ type\ 2,\ type\ 4 \\ \mathrm{e}^{-\rho_i}\dfrac{\rho_i^{K_i}}{K_i!},\ type\ 3 \end{cases}$$

Per quanto concerne l'equazione che presenta il termine sottolineato, è bene specificare che abbiamo una restrizione sul type 1: è necessario avere esclusivamente un unico servitore ($\underline{m_i = 1\ \forall i}$). La formula ricorda tanto la M/M/1. Ma vale anche per type (1,2,4). Reti: $\cdot/\mathrm{M}/(1 = m_i)$. Per la terza invece abbiamo reti $\cdot/\mathrm{G}/\infty$. ρ_i è il fattore di utilizzazione del nodo i. Vale: $\rho_i := \sum_{r=1}^{R} \rho_{ir}$. ($\forall$ classe di servizio), Se ho m_i servitori nel type 1, $\pi_i(K_i)$ avrà la stessa espressione trovata per la $\mathrm{M}/\mathrm{M}/m_i$ (λ_i, μ_i). Le velocità di arrivo le calcoliamo con le EQUAZIONI DEL TRAFFICO. Considerando $(m_i = 1)$, dobbiamo imporre che $(\rho_i < 1)$ per

la STABILITÀ della CODA. Anche per il type 1 con più servitori ($m_i \geq 1$), avremo: $\rho_i = \frac{\lambda_i}{m_i \mu_i}$ nel caso $m_i \geq 1$. Quindi:

$$\rho_{ir} = \begin{cases} \dfrac{\lambda_{ir}}{\mu_i},\ type\ 1,\ \mu_i \neq \mu_{ir} \\ \dfrac{\lambda_{ir}}{\mu_{ir}},\ type\ \{2,3,4\},\ (\mu_i = \mu_{ir}) = \cdot(IDX\ r) \end{cases}$$

Nel secondo caso infatti (seconda equazione), le distribuzioni dei tempi di servizio possono essere differenti per le varie classi. Ricordiamo che vale: $[\underline{\lambda_i} = \sum_{r=1}^{R} \lambda_{ir}]$, differenziando ed includendo rispetto alle varie classi. Il termine sottolineato rappresenta la velocità totale di arrivo al nodo i.

4.2.5 RECAP

[BCMP $\supset$ JACKSON $\supset$ ACICLICHE]. $\pi_i(K_i)$, reti $\{\cdot/\text{M}/1,\ \cdot/\text{G}/1\text{-PS},\ \cdot/\text{G}/1\text{-LCFS-PR}\}$. Per queste code, ai fini dell'Occupancy Distribution, si comportano come se fossero delle M/M/1 in isolamento ($1 = m_i$ servitori $\forall i$). Nel type 1 abbiamo la FCFS, mentre nel type 2, type 4 non abbiamo quella disciplina di coda! È come se ai fini del calcolo della OD, avessimo invece proprio la disciplina FCFS. Si comportano proprio come se fossero M/M/1 in isolamento.

Per quanto riguarda il secondo caso, per un M/M/∞ abbiamo $[\text{e}^{-\rho_i} \frac{\rho_i^{K_i}}{K_i!}]$. Ma vale sostanzialmente per tempi di servizio non distribuiti esponenzialmente (per le $\cdot/\text{G}/\infty$). La motivazione è collegata proprio al teorema BCMP. Tornando al primo caso, ipotizziamo una rete di code BCMP con un processo degli arrivi dall'esterno che NON dipenda dal carico (Stiamo considerando un processo degli arrivi di POISSON omogeneo). Il BCMP vale anche se c'è un solo sistema a coda! Consideriamo $\cdot/\text{M}/1$. Adesso sono sicuro che mi trovo dinanzi una M/M/1. Adesso i processi degli arrivi sono di POISSON. Per il type 2, M/G/1-PS. Stesso discorso per il type 4. Le formule trovata per un M/M/1, sono valide anche per un {M/G/1-PS, M/G/1-LCFS-PR}, ove i termini sottolineati si riferiscono rispettivamente a code del tipo type 2 e type 4.

4.3 Reti di dati

Ulteriore passaggio verso la modellazione. Obiettivo: modellare una rete di dati switched. Consideriamo $(i \to j)$. Ci saranno 4 componenti di ritardo su questo link. Dobbiamo considerare:

- -) tempi di accodamento in i;
- -) tempi di trasmissione su questo link;
- -) tempi di propagazione;
- -) tempi di elaborazione

Modellando queste componenti di ritardo per il link $(i \to j)$, consideriamo nel primo blocco sicuramente $\cdot/\text{M}/1$ e agli ultimi due blocchi utilizzeremo rispettivamente: $\{\cdot/\text{G}/\infty$ e $\cdot/\text{G}/\infty\}$. I sistemi a coda sono dei modelli per modellare il ritardo.

Virtual Circuit (a commutazione di pacchetto). Per un certo flusso la ROTTA sarà sempre la stessa. Stiamo immaginando qui di avere 3 flussi: $\{X_{s1}, X_{s2}, X_{s3}\} \leftarrow$ velocità di arrivo. Per il link $(i \to j)$ abbiamo il flusso di pacchetti $s_1 - s_2 \implies X_{s_{i-j}} = X_{s1} + X_{s2}$. A questo punto, per valutare λ_{ij}, dobbiamo trovare la somma delle X_s estesa a tutti gli stream S che attraversano il link $(i \to j)$:

$$[\underline{\lambda_{ij}} = \sum_{stream\ s\ in\ (i,j)} X_s]$$

X_{si} è il rate (velocità) del flusso i-esimo. La rete Internet è a commutazione di pacchetto ma di tipo Datagram (Rotte non sempre le stesse). Per le DATAGRAM: $f_{ij}(s_1)X_{s,1}$ rappresenta la Velocità di arrivo, considerando le opportune frazioni dei pacchetti. Qui per calcolare le velocità di arrivo totale dobbiamo considerare le seguenti formule:

$$\underline{\lambda_{ij}} = \sum_{stream\ s\ in\ (i,j)} f_{ij}(s)X_s$$

ove la somma è estesa al solito a tutti gli stream che attraversano il link $(i \to j)$. Biforcazione o fork possibile.

Velocità di arrivo totale dei vari link sia su una VC che su una DATAGRAM. Entrambe a commutazione di pacchetto. Rete di code BCMP. Versione semplificata del teorema per modellare una rete di dati come una rete di code. Modellare ogni link della rete con il seguente schema:

Ipotesi sul sistema reale. Rete BCMP (più flussi) $\implies$ ($\forall$ flusso $\exists$ classe associata ad esso $\iff$ $(s \to r)$). Sia $R = |flussi\ di\ pacchetti|$. Rete BCMP con R classi di clienti. Consideriamo il generico flusso s, al quale corrisponde una certa velocità $X_s \iff (s \to X_s)$. Per quanto concerne le velocità totali abbiamo: $\underline{\lambda_{0,r}}$ è la velocità di arrivo dall'esterno dei clienti di classe r; $\lambda_{0,hr}$ è la velocità di arrivo dall'esterno dei clienti di classe r alla coda di ingresso h della rete; $p_{0,hr}$ è la probabilità che un cliente dall'esterno di classe r vada verso la coda h. Immaginiamo che h sia il nodo di ingresso $\implies p_{0,hr} = 1$:

$$X_s = \lambda_{0,r} \iff [\lambda_{0,r}p_{0,hr} = \underline{\lambda_{0,hr}}]$$

Difatti quella probabilità è 1 per quella coda che rappresenta il nodo di ingresso. $\nexists$ biforcazioni all'ingresso della rete. Per applicare il teorema BCMP dovremo imporre che questi pacchetti arrivino secondo un processo di POISSON. Un'altra ipotesi riguarda i tempi di trasmissione. Nel primo blocco modelliamo il ritardo di accodamento e quello di trasmissione. Dobbiamo ipotizzare che i tempi di trasmissione siano distribuiti esponenzialmente (Con la stessa distribuzione per tutte le classi). Parametro $(\frac{1}{\mu_{ij}})$ come valor medio. Supponiamo che i buffer del router siano molto grandi $(d = +\infty)$ da poter ritenersi illimitati. Valga l'approssimazione di Kleinrock. Indipendenza tra i tempi di interarrivo e tempi di servizio. (Reti magliate ed intensità di traffico medio-alta). Vale quando la rete è abbastanza magliata e intensità di traffico medio-alta. Con queste ipotesi possiamo applicare il teorema BCMP. I due successivi blocchi di ritardo modellano il tempo di propagazione e di elaborazione (quest'ultimo è il router j) (ADN). Per elaborazione si intende il ritardo per decidere il routing. Il router analizzerà i vari link e si occuperà di decidere consultando le tabelle di forwarding. Il tempo di propagazione si trascura solitamente. Al posto dell'$\cdot$/M/1 andrebbe bene un $\cdot$/M/∞. Anche se non siamo vicini alla distribuzione esponenziale per le lunghezze dei pacchetti, tutto alla fine va ancora bene. Applichiamo quindi le formule BCMP. Sia: $\bar{R}$ il tempo medio di attraversamento della rete da parte di un pacchetto. POOLING di tutti i flussi che attraversano lo stesso link. Abbiamo: $\bar{R} = \frac{N}{\gamma}$, ove abbiamo posto $\gamma := \sum_{tutti\ gli\ stream\ s} X_s$ ovvero la velocità totale di arrivo dei pacchetti. Dato che abbiamo a che fare con valor medi... indichiamo inoltre con $\{\mathbb{E}[T_{ij}^p],\ \mathbb{E}[T_{ij}^e]\}$ rispettivamente il valor medio del tempo di propagazione ed il valor medio del tempo di elaborazione, entrambi sul link $(i \to j)$. Sfruttando le formule dell'M/M/1 e di Little otteniamo:

$$\bar{N}_{ij} = \frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}} + \lambda_{ij}\,\mathbb{E}[T^p_{ij}] + \lambda_{ij}\,\mathbb{E}[T^e_{ij}]$$

Abbiamo trovato il numero medio di pacchetti sul link $(i,j) \implies \bar{N} = \sum_{(i,j)} \bar{N}_{ij}$. Dividiamo per la velocità totale di arrivo dei pacchetti (γ) e troviamo $\bar{R}$. Sempre per il fatto della forma prodotto. (M/M/1) in isolamento (BCMP TH.). Se introduco un modello M/G/∞, un po' di errore lo commetto. Le formule che ottengo per l'M/G/∞ sono analoghe a quelle M/M/∞.

Theorem 17. $\bar{R}_{path}$

Per calcolare questa quantità, dovrei considerare tutti i link associati a questo percorso:

$$\bar{R}_{path} := [\sum_{link(i,j)\ lungo\ path} \frac{1}{\mu_{ij} - \lambda_{ij}}] + \mathbb{E}[T^p_{ij} + \mathbb{E}[T^e_{ij}]]$$

Vale per tutte le classi! ($\forall r = s$). Qualunque sia la sua classe. Tanto è inglobata nelle varie quantità ($\mu_{ij},\ \lambda_{ij}$).

4.3.1 Exercise

Consider the network in the figure below. There are four sessions: ACE, ADE, BCEF and BDEF sending Poisson traffic at rates {100, 200, 500, 600} pkt/min, respectively. Packets lengths are exponentially distributed with mean 1000 $bits$. All transmission lines have capacity 50 $kbits/s$, and there is a propagation delay of 2 $msec$ on each line. Using the Kleinrock independence approximation,

- a) find the average number of packets in the system, the average delay per packet (regardless of session) and the average delay per packet of each session;
- b) derive the CTMC that can be used to find the approximate delay distribution related to the packets belonging to the ACE session

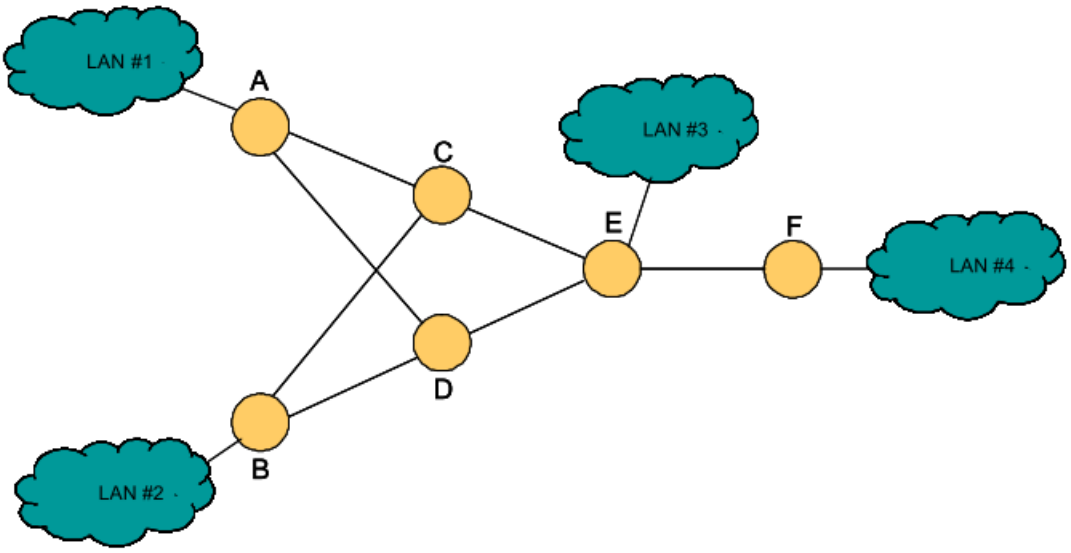


Figura 4.5: Data Network

Flussi in realtà bidirezionali. Ma a livello didattico concentriamoci nel caso NON DUPLEX ($\iff$ Flussi unidirezionali). Abbiamo varie sessioni:

$$\begin{cases} ACE, 1 \\ ADE, 2 \\ BCEF, 3 \\ BDEF, 4 \end{cases}$$

Quattro servizi che generano pacchetti alle seguenti velocità. $\{100, 200, 500, 600\}$ $[pkt/min]$. $L = 1000\ bit$ (lunghezza pacchetti), $C = 50\ kbps$. $\mathbb{E}[T_{ij}^p] = dp = 2ms$ (propagation delay, PD). ($\forall ij$. Qualunque sia il link i, j). Il PD è in realtà una costante, non un valor medio. Supponiamo quindi che tutti i link siano alla stessa velocità. Supponiamo valida l'approssimazione di Kleinrock. 4 flussi di pacchetti $\iff$ 4 sessioni di comunicazioni. Numeriamo le sessioni e calcoliamo:

$$\begin{cases} X_1 = (\frac{100}{60} = \frac{5}{3})\ pkt/s \\ X_2 = (\frac{200}{60} = \frac{10}{3})\ pkt/s \\ X_3 = (\frac{500}{60} = \frac{25}{3})\ pkt/s \\ X_4 = (\frac{600}{60} = 10)\ pkt/s \end{cases}$$

Ci serve andare a calcolare le velocità totali di arrivo sui VARI link attraversati.

$$\begin{cases} \lambda_{AC} = X_1 \\ \lambda_{CE} = X_1 + X_3 = 10\ pkt/s \\ \lambda_{AD} = X_2 \\ \lambda_{BD} = X_4 = 10\ pkt/s \\ \lambda_{DE} = X_2 + X_4 = \frac{40}{3}\ pkt/s \\ \lambda_{BC} = X_3 \\ \lambda_{EF} = X_3 + X_4 = \frac{55}{3}\ pkt/s \end{cases}$$

Dobbiamo andare a considerare la velocità di servizio su ogni link. Ci troviamo dinanzi $\cdot$/M/1. Ma anche relativi ai router dei blocchi $\cdot$/G/∞. Dividiamo $\frac{C}{L} = \mu_{ij} = (\frac{50000}{1000} = 50)\ pkt/s$. Dobbiamo calcolare il γ (velocità totale di arrivo dei clienti):

$$\gamma = \sum_{i=1}^{4} X_i = \frac{70}{3}\ pkt/s$$

in quanto abbiamo 4 flussi diversi. Con $\bar{N} \simeq 1.84\ pkt$ pacchetti, abbiamo quindi:

$$\bar{R} = \frac{\bar{N}}{\gamma} = \frac{1.84}{70/3\ pkt/s} = \frac{3(1.84)}{70} msec\ (millisec)$$

(poco credibile oggi). Dobbiamo ora calcolare il ritardo medio $\forall$ stream (per i vari stream). Per le varie sessioni ($\bar{R}_{path}$ formula), abbiamo:

$$\bar{R}_p = [50ms,\ 53ms,\ 87ms,\ 90ms]$$

$\Pr\{R \leq t\}$, dove R è il ritardo, tempo di attraversamento della rete.

Distribuzione del ritardo. Supponiamo di voler determinare la distribuzione del ritardo dei pacchetti associati alla sessione ACE. CMTC che con uno stato assorbente può essere utilizzata per modellare il ritardo ($ACE = AC + CE$) (link attraversato):

Distribuzione di ritardo = tempo di permanenza di un pacchetto nella rete in questione. Abbiamo trascurato i ritardi di elaborazione. $\cdot$/M/1, che tiene conto dei ritardi di accodamento + il ritardo di trasmissione. Presso il nodo "C" arriveranno pacchetti anche dal nodo "B" (da un altro stream) (BCEF). Determinare la distribuzione del tempo di risposta in una rete di code aperta. C'è un metodo per approssimare questa distribuzione (del tempo di risposta), e si basa sulla conoscenza della distribuzione del tempo di risposta relativa ai vari sottosistemi di cui la rete si compone. Reti di code di JACKSON. Tempo di risposta = tempo di attraversamento. Vogliamo approssimare questa distribuzione (del tempo di risposta). È possibile andare a derivare da questa rete di JACKSON una CMTC con stato assorbente. Il time-to-absorption (la sua distribuzione), rappresenta un'approssimazione del tempo di risposta della rete. Se è ACICLICA (rete di JACKSON senza CICLI) si ha una coincidenza tra TTA e tempo di risposta della rete. Consideriamo questa rete CICLICA:

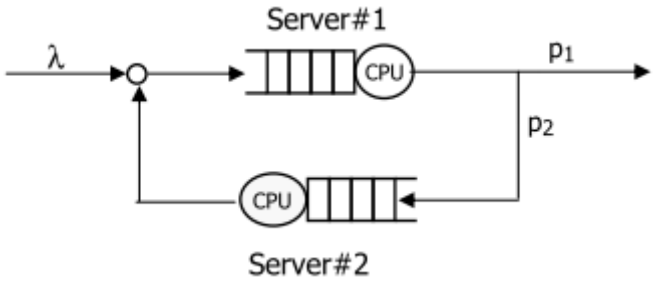
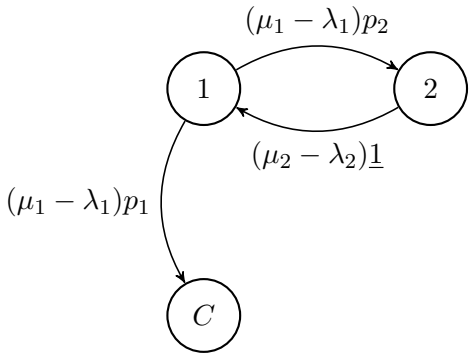


Figura 4.6: Jackson Network

Due $\cdot$/M/1. Questo metodo si basa sulla conoscenza delle distribuzioni dei vari tempi di risposta dei singoli sottosistemi di cui si compone la rete. M/M/1. R = tempo di permanenza. Sappiamo che: $R \sim EXP(\mu - \lambda)$. $F(t)$ CDF, quindi la CDF del tempo di risposta in un M/M/1 è:

$$F(t) = \Pr\{R \leq t\} = 1 - \mathrm{e}^{-(\mu-\lambda)t}$$

Sulla base di queste conoscenze, possiamo dedurre, ricavare la seguente CMTC:



con: $\{\{C\},\{1,2\}\}$ rispettivamente come Stato assorbente e Stati transitori. Lo STARTING STATE è lo stato 1, ovvero lo stato dalla quale evolve la CMTC. $R \sim EXP(\mu - \lambda)$. Rappresenta la permanenza del cliente sul sistema. Tempo di permanenza di un determinato cliente. In particolare, il tempo di soggiorno della CMTC in questo stato corrisponde al tempo di permanenza nel primo M/M/1. ($R_1 \sim EXP(\mu_1 - \lambda_1)$). Il tempo di soggiorno della catena nello stato 2 rappresenta il tempo di permanenza del cliente nel sistema nel sistema a coda 2 ($R_2 \sim EXP(\mu_2 - \lambda_2)$). Se invece lasciasse la rete, si avrebbe una transizione verso lo stato assorbente {C}. Il TTA qui corrisponde al ritardo del cliente in questa rete (tempo di permanenza). Se riesco a determinare la distribuzione del TTA per questa catena riesco a determinare proprio il tempo di permanenza nella mia rete. Ma è un'approssimazione questa, in quanto abbiamo dei cicli! Quindi i processi degli arrivi NON sono di POISSON.

$$\underline{\Pr\{T_a \leq t\}} \simeq \underline{\Pr\{R \leq t\}}$$

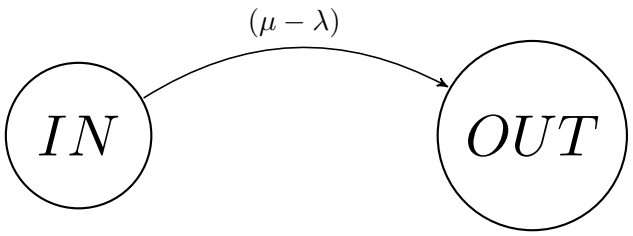
ove il LHS sottolineato rappresenta $\pi_C(t)$ in TRANSITORIO, mentre il RHS sottolineato rappresenta la distribuzione del ritardo nella mia rete. A quanto pare vi è un'approssimazione possibile. Per trovare $\pi_C(t)$ si può procedere con il metodo della trasformata di Laplace, diversamente ci sono dei programmi di simulazione e non. Si richiami il concetto di affidabilità:

$$\Pr\{R > t\} = 1 - \underline{\pi_0(t)} = 1 - \Pr\{R \leq t\}$$

4.3.2 Response-Time Blocks (RTB)

Li utilizziamo per trovare queste distribuzioni. Blocchi mediante i quali costruiamo le CMTC con stati assorbenti. Dobbiamo considerare tanti blocchi quante sono le componenti di ritardo.

- **RTB *M/M/1***: Ci sono quindi anzitutto quelle per la M/M/1 (λ, μ), ove per stabilità deve valere: $(\lambda < \mu)$. Il RTB per questo sistema è il seguente:

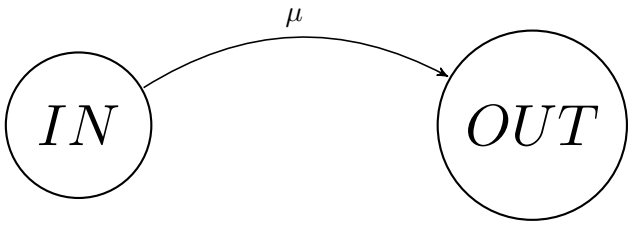


  Lo stato IN rappresenta il fatto che il cliente è entrato nella coda. Il tempo di soggiorno nello stato {IN} rappresenta il ritardo nella rete. {OUT} = {partenza del cliente nella coda}. (Tante serie di blocchi quanti sono i nodi in gioco):

$$\underline{F(t)} = 1 - \mathrm{e}^{-(\mu-\lambda)t}$$

  Ove il termine sottolineato rappresenta la CDF del tempo di risposta. Lo stato {OUT} rappresenta la partenza del cliente. OUT nel precedente esercizio era {2,C}=OUT. In tali casi saranno da pesare le velocità di uscita complessive (con probabilità eventualmente, corrispondenti alle cosiddette probabilità di routing);

- **RTB *M/M/*∞**: Il tempo di soggiorno nella mia catena corrisponde al tempo di servizio $\iff \nexists$ code nella M/M/∞. (infiniti servitori, NO fila di attesa).



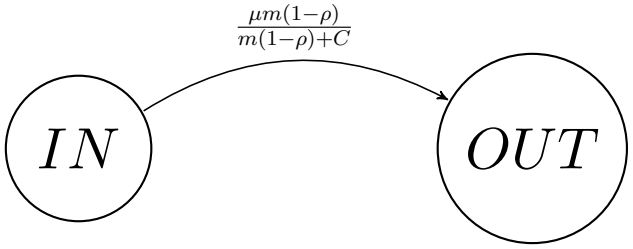
Tempo di risposta corrisponde al tempo di servizio:

$$F(t) = \Pr\{R \leq t\} = 1 - \mathrm{e}^{-\mu t}$$

{IN} rappresenta quindi lo stato di permanenza nel mio sistema;

- **RTB *M/M/m***:

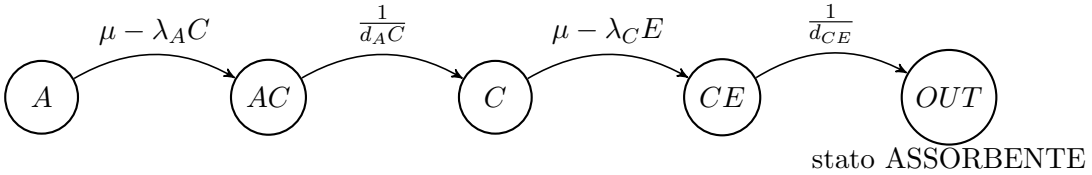
  Per ultimo, ma non di importanza, abbiamo l'RTB di una M/M/m:



  ove C è la C di ERLANG, ovvero la probabilità di accodamento;

Ma lo stato OUT può corrispondere allo stato IN di un altro blocco! Se effettivamente OUT è l'uscita della rete sarà assorbente, altrimenti NO.

Adesso consideriamo la rete precedente. Quella è la rete di code che modella la sessione ACE sostanzialmente:



Il nodo E interfaccia i pacchetti verso la LAN 3. (Rete ACICLICA). Trascuriamo il ritardo di elaborazione, al solito. Velocità elevatissima! A volte si può ritenere legittimo trascurare il ritardo di accodamento e trasmissione della LAN di destinazione! Molto veloce! TTA in tal caso coincide esattamente con il ritardo di attraversamento (tempo di attraversamento) della rete, dal momento che la rete è ACICLICA.

Capitolo 5

Network Technologies 2

5.1 (Rapid)STP - Spanning Tree Protocol

{M/M/1, M/M/∞}. (STP)$\rightarrow$(RSTP) rapida. La RSTP (Rapid) è la versione veloce dell'STP. È un protocollo che consente di trasformare una topologia magliata in una ad albero (bridge). L'STP porta a dei tempi di convergenza molto elevati. *IEEE 802.1D* è lo standard che va a definire i *Transparent Bridge*. Oggi i bridge sono HW (lvl 2). Prima erano a livello SW. Oggi abbiamo i router HW ASIC elettronici (matrici ASIC).

L'STP serve per evitare che vi siano dei loop a lvl 2. Queste maglie esisteranno per via dei link ridondanti, introdotti per aumentare l'Affidabilità/Disponibilità. Dato che gli switch/-bridge IEEE 802.1D non tollerano topologie magliate, allora serve l'STP. Il pericolo è quando loro fanno il flooding. Fase di forwarding, MAC Multicast/Broadcast $\implies$ flooding su tutte le altre porte. MAC Unicast $\implies$ funzioni di forwarding, previa consultazione delle sue tabelle di forwarding/routing. Si possono creare dei LOOP ad esempio, con il flooding. Tante copie dello stesso pacchetto. Gli switch lvl 2 NON tollerano topologie magliate. Una rete Fault-Tolerant richiede invece topologie magliate. Ma alla fine dobbiamo avere quella ad albero. Switched LAN. $\forall$ porta $\exists!$ LAN. Come l'STP si occupa di effettuare questa trasformazione? Trasforma alcune porte dello switch in *BLOCKING*, altre in modalità *FORWARDING*. Ogni interfaccia dello switch è numerata. Una porta bloccata non potrà forwardare/ricevere trame $\iff$ No elaborazione. Esiste anche lo stato *DISABLED*. Quando è disabilitata, è come se non ci fosse proprio nella topologia ai fini dell'STP. Disabilitata quando o c'è un guasto, o viene disabilitata da un *Network Manager* oppure non è collegata a nessun dispositivo. Esistono anche switch remoti! Che interconnettono LAN geografiche. A scala geografica ci potrebbe essere un enorme spreco, se disabilitassimo alcune porte. Link di comunicazine geografici $\implies$ bridge remoti. Se NON utilizzassimo un link geografico, avremmo un notevole spreco di banda. DRAWBACK. Le path risultanti dall'STP potrebbero essere più lunghe rispetto ad altre path che si potevano scegliere. L'STP converge via via in caso di guasti a nuovi ST (*Spanning Tree*). Switched LAN = reti LAN interconnesse tra switch. Quando c'è qualche guasto, alcune altre porte che stavano in Blocking potrebbero tornare in Forwarding. Con le nuove convergenze riusciamo a farle comunicare.

5.1.1 Funzionamento STP

Modi di funzionamento: tre fasi principali. Alta Disponibilità $\implies$ rete fault-tolerant $\implies$ ridondanza. VLAN (*Virtual LAN*). Se le VLAN attraversassero più piani in tal caso, ci sarebbe bisogno dell'STP. Cosa accade quando una VLAN rimane confinata in un solo piano/od attraversa più piani? RSTP possibile. Abbiamo tre fasi principali:

- *Elezione del* ***ROOT BRIDGE***: $\exists!$ "highlander", root bridge. Tutte le porte del root bridge saranno settate a forwarding;

- *Selezione della* ***ROOT PORT***: $\forall$ bridge $\neq$ root, seleziono la porta verso la quale giunge a costo minimo al root bridge. La ROOT PORT di ogni bridge viene settata in FORWARDING;

- *Selezione del* ***BRIDGE DESIGNATO*** *e della* ***DESIGNATED PORT***: $\forall$ segmento LAN alla quale saranno collegati due o più bridge, viene selezionato un bridge detto *BRIDGE DESIGNATO*. Sarà quello più vicino al ROOT BRIDGE. La porta del bridge designato con la quale quel bridge è collegato a quel segmento viene detta *PORTA DESIGNATA*. Solo il BRIDGE DESIGNATO sarà autorizzato ad inoltrare le trame dati!

Un segmento LAN. LAN = rete locale che si basa su un canale Broadcast (canale ad accesso multiplo ad accesso circolare). Si risolve col protocollo MAC (Nell'Ethernet abbiamo il *CSMA/CD*). Il canale è a diffusione circolare ma la trasmissione non sarà inviata in Broadcast. Sentiranno tutti ma lo elaborerà solo uno quel determinato pacchetto. Una stazione è un dispositivo che implementa almeno il sottolivello MAC. Un ripetitore NON è una stazione! (Implementa solo il Physical). Anche il ROOT BRIDGE è collegato a segmenti LAN. Ma tutte le sue porte sono in Forwarding $\implies$ tutte le porte del root bridge sono designate. La porta designata sarà ovviamente messa in stato di Forwarding. $\forall$ segmento abbiamo il bridge designato.

Perché funzioni l'STP abbiamo bisogno di scambiare dei messaggi di controllo (BPDU) - *Bridge Protocol Data Unit*. Parliamo di un protocollo standardizzato dall'ISO. Problema del supporto multiprotocollo. Ogni livello ha il problema del multiprotocollo (es. LLC per 802.1). Ma l'IP non è stato standardizzato dall'ISO $\implies$ serve lo SNAP. Codici per l'LLC abbiamo 042H sia per il DSAP che il SSAP. È l'LLC che trasporta la BPDU. Le BPDU possono essere di due tipi:

- ***Configuration BPDU*** utilizzata per la definizione della topologia loop-free (ad albero);

- ***Topology Change Notification BPDU*** **(TCN BPDU)** con l'obiettivo di scatenare nuovamente l'STP per far convergere ad un nuovo albero;

Il campo *BPDU type* fa la differenza tra i due tipi {CONFIG, TCN}. Quelle di CONFIG hanno anche altri fondamentali campi dell'STP. {TC/ . . . /TCA}. servono per lo scatenamento delle reazioni alle notifiche TCN (C)BPDU $\implies$ hello messages!

- *Root Bridge ID*: identificativo del bridge che
  si assume sia il ROOT BRIDGE {REAL/SUPPOSED}, per via della presenza di una fase transitoria;

- *Root Path Cost*: Path a minor costo verso il root bridge a partire dallo switch che sta trasmettendo questo messaggio di configurazione. Inviato ovviamente NON significa che l'abbia generato lui! Messaggi di Hello = CBPDU. Costo della path a minor costo;

- *Bridge ID*: identificativo del transmitting bridge;

- *Port ID*: analogamente per questo transmitting bridge: la porta dalla quale STA INVIANDO il messaggio;

- {*Message Age, Max Age, Hello Time, Forward Delay*} sono impostati dal ROOT BRIDGE. Nell'*Hello Time* imposterà il tempo di inter-ricezione dell'hello message. Di default pari ad (1s). Ogni 1s hello time da parte del root bridge. Passato l'hello time si aspetta un altro *Max Age*. Scaduto anche questo tempo, manda delle TCN BPDU. Forward delay (default 15s);

Identificativi (IDS):

- ***Bridge Identifier e Root Identifier***: (*Bridge Priority + Bridge MAC Address*). Di default la priorità è settata a 32768. Viene settata (incrementata o decrementata a steps di 4096 unità);
- ***ID di porta***: (*Port priority + Port number*). default 128 la priorità. Passi di 16 unità possibili. Dividiamo in due la rappresentazione decimale (es. 128.21). 1 byte per la priorità ed 1 byte per il numero di porta;

{Root ID, Bridge ID, Port ID}. Poi avevamo il Root Path Cost. Alla porta di uno switch è associato un costo, settato manualmente da un Network Manager oppure sono dei costi raccomandati dalla IEEE, a seconda della velocità della porta (es. Ethernet). 16 bit per il costo. Fino a 65535 (Unsigned). Fino a 10Tb/s. (Revised *802.1D*) (Switched LAN). Alle porte sono associate dei costi. Un bridge sarà in grado di calcolare il root path cost a partire dalle porte sulle quale sta ricevendo. Costo porta di ricezione + costo che troviamo nel messaggio di hello. Additività. Bisognerà preoccuparsi ovviamente di aggiornare via via il Root Path Cost per inviare a valle (ad un livello più basso della topologia).

Spiegazione FASI STP

- 1) Elezione del ROOT BRIDGE. Parte la rete. Ogni switch pensa di essere il root bridge. Nel Root Path Cost metteremo ovviamente 0! Root ID = Bridge ID inizialmente. Se un bridge riceve su una certa porta un messaggio di Hello, farà un confronto con il proprio Root ID. Vince l'ID più basso! (Bridge ID più basso). Root/Bridge ID. Il bridge con maggior Bridge ID tra questi ora non invierà più messaggi di hello, e setterà il proprio Root Path Cost a partire dal costo della porta dalla quale sta ricevendo. Se la priorità è lasciata di default, il confronto verrà fatto a livello di indirizzo MAC. In questo modo $\exists!$ alla fine Root Bridge, con il (Bridge ID) = (Root ID) più basso. Prima di forwardare un messaggio di hello, un bridge non root modificherà opportunamente il Root Path Cost. Alla fine uno soltanto sarà eletto. ID: {Bridge Priority + Bridge MAC}. Ad uno switch è associato un indirizzo MAC. Avrà un MAC $\forall$ porta. Per la sua operatività non serve il MAC in realtà. Per quanto riguarda le trame dati difatti, gli indirizzi MAC sono trasparenti! Da qui il nome TRANSPARENT BRIDGE. Prima abbiamo l'elezione del Root Bridge;
- 2) fase successiva: elezione della Root Port. $\forall$ bridge non root dovrà selezionare la Root Port. $\exists$ certa sequenza di condizioni che devono essere verificate in caso di condizioni di parità:
  - -) Supponiamo che si verifichi una situazione di parità sulla porta (Rotta a minor costo);
  - -) Selezioniamo come Root Port quella sulla quale i messaggi hanno il Bridge ID più basso;

  - -) Se abbiamo stessi Bridge ID abbiamo il confronto con la Port ID. Porta dalla quale si riceve il messaggio di hello con il più piccolo Port ID. Porta di INVIO. Porta tramite la quale quel transmitting bridge sta inviando il messaggio di hello;
  - -) Alla fine si passa alla quarta condizione discriminante: identificativo della porta di ricezione. Vince quella minore;

  La prima condizione prevede la discriminante come Least Cost (Root Path Cost);

- 3) Selezione della PORTA DESIGNATA. $\forall$ segmento LAN ci sarà da selezionare un bridge designato. Quale sarà il bridge designato? Quello che sta comunicando su quel segmento LAN delle BPDU a più basso costo;

Se una porta è bloccata NON vi è inoltro. Un Hub (ripetitore) è come se non ci fosse (è trasparente per loro). Un hub NON potrebbe interconnettere LAN a differenti velocità, né LAN a differenti MAC (avrebbe bisogno dello stack data layer a lvl 2). Ci sarebbe inoltre bisogno dello *STORE & FORWARD* (S&F). Quest'ultimo lo possono fare switch, router (logica di elaborazione lvl 2, lvl 3).

Alla fine sarà il Root Bridge ad inviare messaggi di hello. Valori dettati dal Root Bridge. Se un Bridge non root si aspetta di ricevere messaggi di hello con quella temporizzazione, aspetta un pochino di tempo (pari a Max Age). Passato anche questo tempo (tipicamente 20s), il suddetto bridge scatena l'STP, mandando TCN. Se una porta, prima in stato di Blocking (per evitare i loop), deve passare a Forwarding, tutto ciò non avviene immediatamente. Prima quella porta viene messa in *LISTENING STATE* per evitare loop. Quella porta si potrebbe accorgere che il Root Bridge è "obsoleto". Passato un quantitativo di tempo pari a Forward Delay, entra in *LEARNING STATE*, nella quale sarà autorizzata ad utilizzare il *Backward Learning*. Dopo un altro tempo pari al Forward Delay, finalmente la porta entrerà in *Forwarding State*. Durata massima (pensando ai tempi di default), $20 + 15 + 15 = 50s$. Si è quindi definito l'RSTP. Se avessimo più switch naturalmente avremmo dei transitori molto lunghi, e questi valori temporali potrebbero essere pesanti, soprattutto se si parla di una rete di grandi dimensioni! Potrebbe creare lei stessa dei loop. Pericolo di *Broadcast Storm* (invocazioni troppo frequenti dell'STP):

TCA settati (*Topology Change Acknowledgements*) dal notifier del cambiamento della topologia. Modifica dell'Aging Time (Apprendimento Annullato). La convergenza è pesante, ed è stata risolta dall'RSTP. *IEEE 802.1AX*. Raggruppare link paralleli tra stazioni. Quattro link a 100 $Mbit/s$ ad esempio. Raggruppare ad un link logico con banda più o meno costante (grossomodo 400 $Mbit/s$) (LACP) = *Link Aggregation Control Protocol*. In parallelo NON abbiamo il problema della convergenza (non se ne accorgerà l'STP). Il costo del Link cambierà ad un guasto, aumenterà di preciso. Il costo del Link Logico invece diminuisce al raggruppamento. Il protocollo lavora soltanto con Full-Duplex alla stessa trasmissione.

RSTP = *Rapid STP*, definito nell'*IEEE 802.1W*. Lavora come l'STP, ma cambia la terminologia: *Discarding* $\supset$ {Blocking, Listening, Disabled}. Si può scendere giù di 10s, oppure di 12 o 2s in alcuni casi. Con l'STP abbiamo la selezione della Root Port. Se una root port viene meno, viene scelta la porta *Alternative* a minor costo. Meccanismi aggiuntivi che lo rendono molto più veloce. Se ci sono degli Hub Repeater ivi prevediamo solo dei link punto-punto! Altrimenti stessi tempi dell'STP;

Agire sulla topologia modificando il campo *Priorità*.

VLAN = Virtual LAN. Configurare su una Switched LAN diverse sotto-LAN logiche. Configurazione LAN virtuali. Differenti VLAN comunicano tra di loro solo sfruttando il lvl 3. L'STP non andrà a guardare le LAN virtuali. Prima avevamo quindi l'esecuzione di una sola istanza dell'STP. Per effettuare *Load Balancing* si è introdotto l'MST (*Multiple STP*). C'è da pensare a vari parametri $\forall$ VLAN (Virtual LAN). Quindi è stato modificato il campo Bridge Priority

(primi 16 bit). Bridge ID: (Priority ID + Bridge MAC). BP non più di 16 bit. 12 bit = identificativo della VLAN. 4 bit più a monte: valore effettivo della priorità. Il campo di 16 bit è stato quindi diviso in 2: 32768 default priority. Incrementi consigliati sempre di 4096. (4 bit = BP + 12 bit = *System ID Extension*). 4 bit $\implies$ 16 valori di priorità possibili. Non è comunque limitativo. L'intero Bridge Priority sarà quindi modificato MA sarà comunque da affiancare all'indirizzo MAC per formare il Bridge ID ed eventualmente Root ID.

RECAP

STP. Strategie per evitare che un malintenzionato possa collegarsi ad uno switch di piano ed alterando la BPDU con priorità molto basse, modificando quindi lo Spanning Tree. Oltre ad alterare lo Spanning Tree potrebbe anche intercettare il traffico. Se lui fosse il ROOT, il traffico transiterebbe tutto verso di lui. Potrebbe di conseguenza anche creare una situazione DoS. Due opzioni, proposte in ambito CISCO:

- Configurare *fast port* (generalmente user port), ma non va bene perché toglierebbe la possibilità di avere un PC con meccanismi di switching;
- *Root Guard* disabilita una porta che diventerebbe STP root port;

L'STP di default è sempre attivo sugli switch. Si può disattivare ma è meglio tenerlo prevalentemente attivo.

All'avvio di una rete switchata parte l'STP a lavorare (vedasi *Cisco Packet Tracer*). Di default sugli switch lvl 2 o multilayer è configurato sulla VLAN 1 (sulla quale sono associate tutte le porte). ID Bridge: {Priorità + Indirizzo MAC}. Con l'avvento delle VLAN adesso abbiamo la suddivisione della Priorità in: {BPV: *Bridge Priority Value* di 4 bit + *System ID Extension*} $\leftarrow$ relativo al Multiple Spanning Tree (MST). Gli ultimi 12 bit sono l'identificativo delle VLAN. es. Bridge Priority: 32769: priorità effettiva: 32768 + (1 = VLAN ID). Link punto-punto. Costo 19 porte *Fast Ethernet*. Se cambiamo priorità, l'algoritmo si riavvia. Bisogna specificare la VLAN. Ci vuole un certo tempo perché converga l'algoritmo (STP).

5.2 LAN Virtuali (VLAN)

Un'organizzazione ha bisogno di avere LAN parallele (indipendenti). Come realizzarle? Si utilizzino degli apparati come switch per collegare le varie stazioni interessate ai rispettivi switch. La tecnologia delle LAN virtuali consente questa configurazione a livello SW, mediante VLAN, anche se i dispositivi sono fisicamente collegati. Una certa serie di VLAN può esserci anche con un singolo switch, con un'operazione opportuna di configurazione. Ma le stazioni LAN potrebbero non essere separate fisicamente! Sono separate logicamente. Potrebbero anche esserci due o più switch $\implies$ le VLAN in questo caso si estendono su più switch $\iff$ stazioni collegate a switch diversi fanno parte della stessa VLAN. VLAN che si estendono su più piani, su più switch. Essenzialmente una VLAN è un dominio di *BROADCAST* $\neq$ dominio di *COLLISIONE*. Nel primo se una stazione trasmette un pacchetto in broadcast, nonostante tutti possano sentirlo, un meccanismo di arbitraggio virtuale del canale fa in modo che giunga a destinazione e venga elaborato solo da uno. Si ricordi che uno switch è una stazione! (Almeno lvl 2). {Stazioni = Router, Host, Switch}. Attraverso il full-duplex il MAC è bypassato. È lo switch che stabilisce il dominio di BROADCAST. Gli switch separano i domini di collisione. Se tutte le stazioni di una LAN fossero collegate ad un Repeater, allora otterremmo un unico dominio di collisione, più grande, che coincide in tal caso con il dominio di BROADCAST. Un dominio di BROADCAST contiene uno o più domini di collisione.

Una LAN virtuale corrisponde ad un dominio di BROADCAST. Possiamo già pensare ad una motivazione per le VLAN. Contenere le dimensioni dei domini di broadcast. Far fronte ai protocolli *Broadcast Intensive* (ARP utilizza il Broadcast, ma non è BI!). Altre motivazioni: VLAN per raggruppare gli utenti in Gruppi di Lavoro. Esigenza di avere LAN separate $\iff$ motivazioni di sicurezza. Evitare che il traffico di una certa serie di utenti transiti, riguardi altre VLAN. Due VLAN comunicano tra di loro solo se vi è un dispositivo lvl 3 che le interconnette (router IP con Subnet IP). Controllano il traffico mediante le ACL, diversamente le due VLAN non potrebbero comunicare. Conflitto di competenze: Mantenere dispositivi sensibili, importanti su una LAN virtuale. Separare determinate tipologie di traffico. *Telecommunication Outlet* nella stessa stanza. Si sfrutta un'unica presa di rete (es. telefono VoIP con L2 switch incorporato). Questioni di convenienza (es. TAG VLAN). Nel TAG ci saranno i Campi di *Priorità*. Marcatura a lvl 2, parlando di priorità delle trame appartenenti ad un certo flusso.

Si aggiunge un cosiddetto *Tag* per le VLAN. Sorta di francobolli che identificano la VLAN alla quale la stazione appartiene. Sulla base di questi tag lo switch ricevente capirà come gestirlo. Un link tra due switch attraverso il quale viaggiano trame taggate viene chiamato *TRUNK*. E le porte agli estremi del collegamento si chiamano *porte TRUNK*.

Configurazione di VLAN. Operazione di configurazione. Ci sono due tipologie di porte: {*ACCESS*, *TRUNK*}:

- ***Access***: Le prime trasmettono e ricevono trame NON taggate. Una porta ACCESS potrà gestire una sola VLAN alla quale essa corrisponde. Tipicamente alle porte access ci collego gli host;
- ***Trunk***: A queste porte invece possono essere configurate differenti VLAN. A quale VLAN è permesso attraversare la linea trunk è deciso in fase di configurazione;

La VLAN 1 è quella di default e $\forall$ port $\in$ VLAN 1. La VLAN nativa normalmente corrisponde alla VLAN 1, e le trame associate alla VLAN nativa NON sono taggate! Anche le ACCESS in realtà possono collegare due switch. Associazione: {porte $\leftrightarrow$ VLAN}. Un link trunk tra due switch può essere sempre sostituito con n link access, ove n è il numero di VLAN che viaggiano normalmente sul link trunk. Link Access associato alla VLAN. In questi casi NON abbiamo più trame taggate, ma ci stiamo giocando n link! E costano un bel po' sugli switch.

5.2.1 Configurazione

Configurazione. Comandi CISCO:

```
show VLAN brief
```

VLAN1 = VLAN di default. Per aggiungere delle VLAN devo anzitutto crearle:

```
Sw#vlan database
Sw(vlan)#vlan 2 name Ammin
VLAN 2 added:
        Name: Ammin
```

Si creino le nuove VLAN. Dopodiché dobbiamo cominciare ad associare le porte:

```
Sw(config)#int GigabitEthernet 0/1
Sw(config-if)#switchport access vlan 2
Sw(config-if)#exit
```

Porta di tipo ACCESS associata alla VLAN 2. Per la modalità trunk invece, nel link trunk viaggeranno trame taggate:

```
Sw(config)#interface GigabitEthernet 0/1
Sw(config-if)#switchport mode trunk
Sw(config-if)#switchport trunk allowed vlan 1,4,5
Sw(config-if)#exit
```

Prima entriamo nell'interfaccia delle porte ovviamente. Di default le porte sono tutte ACCESS per la VLAN 1. Oppure:

```
Sw(config-if)#switchport trunk allowed vlan all
```

Le reti *Token Ring* sono state create dall'IBM, ma non ebbero molto successo per via dei costi elevati. Preferibile oggigiorno l'Ethernet perché più semplice con il suo potentissimo CSMA/CD.

Standard che specificano le LAN virtuali: *IEEE 802.1Q - 2005*. VLAN specificate da questi standard. Ma questi standard prevedono le VLAN per porta (*Per-port VLAN*). Tutte le stazioni collegate allo switch mediante quelle porte faranno parte alle VLAN associate a quelle porte. Si abbassa però la Flessibilità. Richiede un certo peso per il Network Manager. Bassa flessibilità su maggior controllo da parte del Network Manager. I produttori possono anche fornire soluzioni alternative, ma non rispettano ovviamente gi standard i quali servono per interoperare invece. L'802.1-Q specifica anche il formato per il tag, i quali servono per identificare la VLAN di appartenenza quando i pacchetti attraversano i link trunk. Il TAG consiste in 4 byte. 2 byte *Type/Length* che si differenziano sulla base dell'*IEEE 802.3*. Per il PADDING serve il Length. Per far sì che la trama abbia una certa lunghezza minima.

[64,1518] byte. Il TAG consiste in 4 byte (2 byte + 2 byte). Primi 16 bit (2 byte) TPID (*Tag Control Protocol Identifier*); se codice 81-00 significa che dopo vi sarà il Tag della VLAN: TCI (*Tag Control Information*) $\supset$ {PCP (*Priority Code Point*): priorità della trama a lvl 2 per la QoS. 7 livelli di priorità: Il 7 è il più alto, mentre il più basso è l'1, mentre quello 0 è di default. Tramite questa priorità si differenzia il servizio fornito; DEI (*Drop Eligible Indicator*) di 1 bit. Qualora si verifichino fenomeni, situazioni di congestione, sono i primi frame (quelli preferibili, candidati) a venire scartati; *VLAN* ID (VID), valore di 12 bit. Le VLAN sono numerate da 1 a 4095 (esattamente 4095 VLAN diverse possibili). Ma il VID può valere anche 0 (quella trama NON è associata a nessuna VLAN). Quel TAG è stato scritturato solo per fornire priorità alla trama (nel PCP)}. La VLAN 1 è spesso utilizzata per una VLAN di gestione (*management VLAN*). I vari apparati dovranno ovviamente essere gestiti.

L'*IEEE 802.1-V* va invece a definire le VLAN per protocollo. Si immagini di configurare un protocollo ad una VLAN. Lo switch, ricevendo una trama, va a vedere la rispettiva PDU. Prevale la IEEE 802.1-V su quella Port-Based. Prima avevamo un solo *Filtering Database*. Adesso i moderni switch supportano l'IVL (*Independent Virtual LAN*): $\forall$ VLAN $\exists!$ Filtering Database. Con questa modalità possiamo associare ad una stazione più VLAN.

Configurazione link trunk. Transitano le trame relative a quelle VLAN permesse. MVRP (*Multiple VLAN Registration Protocol*). Gli switch stessi capiscono quali trame VLAN possono transitare. In base alle VLAN che vi sono agli estremi. Può coesistere una situazione mista. Le trame che partono da una VLAN 1 NON sono taggate su un link trunk. Importante in alcuni scenari. Le BPDU viaggiano NON taggate!

Più istanze STP. Originariamente si poteva creare una sola istanza dell'STP. L'STP NON vede le VLAN! Più recentemente si è data la possibilità di avere più istanze dell'STP $\rightarrow$ (MST) per *Multiple Spanning Tree*. Decidere le associazioni VLAN - Spanning Tree. Tipicamente uno Spanning Tree può coinvolgere più VLAN quindi. Load balancing a lvl 2. {PVST, PVST+} sono protocolli proprietari Cisco. Mentre l'MST è uno standard. LO standard IEEE è comunque stato derivato dall'MST.

5.2.2 Private Virtual LAN (PVLAN)

LAN Virtuali. VLAN Private. Non vi sono degi standard dietro. I produttori si sono inventate le PVLAN. Molto utili. Suddividere una VLAN già esistente in ulteriori sottodomini di BROADCAST, sebbene le subnet IP rimangano sempre quelle di partenza. Ogni sottodominio ha una: {VLAN Primaria, VLAN Secondaria}. La primaria è condivisa da tutti i sottodomini. La VLAN secondaria è quella che fa la differenza. È costituita da *Porte Isolate* che possono comunicare con le cosiddette *Porte Promiscue*. Le *Porte COMMUNITY* possono comunicare solo con le porte nella stessa Community VLAN e con le porte promiscue. {porte promiscue}: porte della VLAN primaria. Una porta promiscua può comunicare con tutte le porte: {$\exists!$ VLAN ISOLATA $\wedge$ $\exists$ VLAN COMMUNITY (multiple)} $\leftarrow$ VLAN secondarie. È una tecnologia che gioca un ruolo importante nelle DMZ. Queste VLAN private possono estendersi su più switch. Esiste anche qui il concetto di link TRUNK. Suddivisione in sottodomini. Le porte promiscue le gioco per firewall, router. Attraverso le VLAN viaggiano le trame appartenenti alla stessa VLAN community.

- -) In una VLAN regolare il broadcast raggiunge tutte le porte della VLAN;
- -) Si ricordi che le porte VLAN isolate comunicano soltanto alle porte promiscue, oppure alle porte trunk se esse permettono di raggiungere una porta promiscua. Le porte promiscue sono utilizzate per accedere al mondo esterno mediante router. Sul link trunk abbiamo il tagging. Accedendo verso una porta promiscua non abbiamo più bisogno di TAG (PVLAN).

Si ricordi che qui non abbiamo più il concetto standardizzato di {ACCESS, TRUNK}. Dominio delle PVLAN. I produttori si sono inventati questa furba tecnologia.

Quindi, per quanto riguarda il broadcast, adesso esso raggiunge solo porte isolate, link trunk (o porte promiscue). Mentre per il broadcast delle VLAN Community, essi raggiungono porte Community (della stessa community), link trunk ed al solito porte promiscue. Questa tecnologia bloccante a lvl 2 serve per proteggere interi blocchi lvl 2 di DMZ.

5.3 NETWORK DESIGN

Design di una rete di Campus ad Alta Disponibilità (High Availability). Gli Access Point sono tipicamente Lite. Architettura basata sul controllore. Unico controllore per tutto il campus (SPoF e bottleneck). $\forall$ access point ho un tunnel verso un unico controllore. C'è un tunnel dall'access point al controllore. Unico tunnel. Possibilità di Controllo ridondante.

5.3.1 Progettazione di rete ad Alta Disponibilità

- **Analisi dei Requisiti**: Prima fase. Requisiti legati al Business di un'Azienda, alla Mission di un'Organizzazione. Requisiti tecnici: {throughput, delay, delay jitter} che consentono di ottenere gli obiettivi della mission. {Disponibilità, Sicurezza dei Dati. Strettamente legati alla Disponibilità};
- Caratterizzare la rete esistente (che tipo di HW, SW è presente). Magari ci saranno delle applicazioni già presenti (flussi di traffico presenti, correnti e quelli futuri);
- **PROGETTAZIONE LOGICA**: Topologia rete, protocolli di routing, strategia di sicurezza e strategia di management (http). Progettata logicamente la rete si passa al:
- **livello fisico**:

– Tecnologie e dispositivi per reti di campus;
– Tecnologie e dispositivi per WAN.

Ci si può avvalere di programmi tipo Packet Tracer per testare il progetto della rete. Si possono (devono) ovviamente applicare delle variazioni in retroazione.

5.3.2 FAULT-TOLERANCE

Una rete può essere progettata per le performance, per la sicurezza, per la Disponibilità. Alcuni requisiti ovviamente vanno in conflitto tra di loro. Adesso ci occupiamo di *Fault-Tolerance*. Bisogna pensare alla ridondanza. Abbiamo diversi tipi di Ridondanza:

- *Backbone*;
- *Device*;
- *ITB (In The Box)*;
- *Interfaccia di Rete*;
- *Stazione duale (tandem, watch dog)*;
- *Sistemi Cluster*

La soluzione fault-tolerant dev'essere la più semplice possibile. Ridondanza del default router. Molto importante! Sicuramente il router di default va ridondato! Il primo protocollo proposto ed utilizzato è l'HSRP (*Hot Standby Router Protocol*) della CISCO (proprietario ovviamente). Poi abbiamo il VRRP, LGBP;

HSRP

L'HSRP fa sì che si crei un router virtuale (detto anche router fantasma). L'*Active Router* e lo *Standby Router* sono router fisici! Il fantasma no! A quello fantasma corrisponde un indirizzo IP e MAC. Ogni host nella rete avrà come default gateway l'indirizzo di quello fantasma. Più di due router possibili, ma router attivi e standby saranno soltanto due! Tutti apparterranno allo stesso gruppo comune. $\exists$ FASE di elezione, basata sul concetto di priorità (100 di default). Qui il router con la più alta priorità diventa quello Attivo. In caso di parità, diventa Attivo quello con l'indirizzo IP più grande. Alla fine della fase di elezione, quello Attivo avrà il ruolo del router virtuale (fantasma), ed assumerà anche IP e MAC (indirizzi) di quello fantasma. Quindi l'interfaccia del router attivo avrà due indirizzi IP! Sempre lui dovrà rispondere alle richieste ARP per il router di default (Manderà la *ARP reply*).

Al termine dell'elezione, periodicamente l'Attivo e lo Standby manderanno dei messaggi di hello. Se il router attivo va down, subentra lo standby come quello Attivo. Ma bisogna eleggere quello di standby tra i router del gruppo HSRP. Se cade quello di standby, di nuovo bisogna eleggerne un altro di standby. Ci saranno dei timer opportuni. I messaggi HSRP sono inviati all'IP multicast 224.0.0.2 sulla porta UDP 1985. Se appropriamente tunato, converge in 1 secondo. Il MAC del router virtuale può essere configurato manualmente oppure creato automaticamente come well-known. Per quanto riguarda l'indirizzo IP, bisognerà configurare opportunamente le interfacce dei router Standby ed Attivo:

```
Router-X> interface ethernet 0
Router-X> ip address 10.1.1.1 255.255.255.0
Router-X> standby 24 ip 10.1.1.5
```

(24 è l'HSRP Group!) A questo punto la elezione che, si basa sull'indirizzo IP qualora la priorità sia la medesima, avrà sicuramente successo. Una volta che un router è diventato Attivo, non molla l'asso anche se subentra qualcuno migliore di lui (Priorità od IP). Ma può subentrare con la funzionalità *preempt*, se opportunamente configurata con:

```
Router-X> interface ethernet 0
Router-X> ip address 10.1.1.1 255.255.255.0
Router-X> standby 24 preempt
Router-X> standby 24 priority 105
Router-X> standby 24 ip 10.1.1.5
```

Naturalmente sono le interfacce che devono gestire il gruppo HSRP. Funzionalità *TRACK*. Se entrambi i router (Standby ed Attivo) sono collegati in Seriale ad un altro router, e la tratta Active - Router *Z* va in down, allora entra in funzione Track che va automaticamente ad abbassare la Priorità del Router "down":

```
R-X>interface ethernet 0
R-X-if>ip address 10.1.1.5 255.255.255.0
R-X-if>standby 1 preempt
R-X-if>standby 1 priority 105
R-X-if>standby 1 ip 10.1.1.10
R-X-if>standby 1 track Serial 0
R-X-if>no shutdown
R-X-if>exit
R-X>interface serial 0
R-X-if>ip address 10.6.2.5 255.255.255.0
```

Se va giù quell'interfaccia si diminuisce la priorità (di default vi sono decrementi di 10). Naturalmente si opererà sull'interfaccia. *1* qui indica il relativo gruppo HSRP, mentre *Serial 0* è l'interfaccia di interesse. La track ha senso ovviamente solo se c'è la Preempt!

Multi-HSRP

l'HSRP normale porta ad uno spreco della banda! Il Router di Standby NON è autorizzato a forwardare pacchetti IP che arrivano dagli host! Il router Standby non farà niente per instradare verso l'esterno! $\Longleftrightarrow$ banda sprecata. I link geografici devono essere adeguatamente sfruttati. Sull'Internet ci sono dei protocolli di routing dinamici (ECMP - *Equal Cost Multipath Routing*). Nelle tabelle di routing ci potrebbero essere path col medesimo costo. Load balancing. Con il *multi-HSRP* faremo *load Sharing* (Balancing). Un router può essere attivo per un gruppo e standby per un altro. Paradigma a stella. Per fare il Load Sharing bisognerà configurare opportunamente gli host, ovviamente. Operazione molto scomoda. Si utilizza nella pratica VLAN+DHCP. Si configuri lo switch con due VLAN (metà parte VLAN 10 e l'altra metà VLAN 20). Per-port VLAN. Le due VLAN corrispondono ad IP differenti. Stiamo pensando all'utilizzo del DHCP. Ovviamente i link switch-router saranno link TRUNK. Settare opportunamente le VLAN ID di appartenenza dopo l'accesso alle interfacce.

Attenzione ad utilizzare l'HSRP su reti switchate! Se va giù il link di comunicazione TRA switch ci sono problemi in entratata con l'ECMP. Sorgono dei problemi di irraggiungibilità (*reachability problem*) per gli host. Il problema si risolve raddoppiando il link di comunicazione (facendo girare STP opportunamente per i loop), oppure si aggregano quei due in un unico link mediante LACP - Link Aggregation Control Protocol. *IEEE 802.1AX* per aggregare i link. Nella realtà non sono collegati questi switch! Ma sono messi in stack tra di loro.

Poi abbiamo il VRRP - *Virtual Router Redundancy Protocol*. Definito in *RFC 2338*. Molto simile all'HSRP (Master → Active, Slave → Standby); GLBP *Gateway Load Balancing Protocol*. Ci consente di evitare il multi-HSRP. Si crea un unico gruppo GLBP, nel quale entreranno a far parte le interfacce. Unico gruppo. Tutti gli host saranno configurati con l'IP del virtual router. L'AVG (*Active Virtual Gateway*) viene eletto, ed è lui l'elemento preposto a rispondere alle *ARP Request*. In maniera ciclica (round-robin) verranno scelti i MAC dei restanti membri del gruppo, denominati AVP (*Active Virtual Forwarder*).

Sicurezza in HSRP

Problema della Security. Indirizzi IP multicast alla quale vengono mandati i pacchetti HSRP: 224.0.0.102 sulla porta UDP 1985. Problema della sicurezza dati. C'è la possibilità che venga sferrato un attacco DoS qualora vi sia *Spoofing HSRP*. Stazione utilizzata da un hacker che manda dei pacchetti hello HSRP con priorità 255 (max priority). Molto probabile che esso diventi il router Attivo. Enhancement necessari. Di default lo schema di autenticazione è un testo in chiaro nel messaggio di hello. Il router accetta il pacchetto solo se corrisponde questo campo di *Authentication*. Molto debole, soggetto a possibili sniffing. La stringa di default è "cisco". Ma un hacker potrebbe ugualmente sniffare con un *Packet Sniffer*. Troubleshooter per problemi di rete, utile ai Network Manager. Switch lvl 2 (bridge). Per la trama broadcast e multicast esso farebbe un flooding. A lvl 2 ci sarebbe un MAC Multicast. Rilevazione trama + Spoofing. Supporto dell'*IGMP Snooping* per le trame Multicast (hello HSRP nel messaggio IP). Ma con questo protocollo si riesce a capire mediante quale porta si riescono a raggiungere determinati gruppi multicast. Abbiamo un enhancement MAC (*Message Authentication Code*) con MD5. Facciamo vedere i comandi rispettivamente quello di default e l'enhancement MD5:

```
...
Router-X> interface ethernet 0
Router-X> ip address 10.1.1.1 2
Router-X> standby 24 authentication text string
Router-X> standby 24 authentication md5 key-string string
...
```

Ove 24 è al solito l'identificativo del gruppo HSRP. *string* → password mediante la quale vengono hashati i messaggi (in questo caso gli hello HSRP). Possibile pure un filtraggio a livello MAC (switch avanzati). Access List a livello più basso (*ACL lvl 2*). La versione HSRPv2 supporta l'MD5.

5.3.3 Network Design in pratica

Switched LAN. Rete ad alta disponibilità. Fault-Tolerant → Ridondanza. Abbiamo visto ridondanza sul default router. Ridondanza sulle dorsali verticali. Certo numero di piani. Su ogni piano, nel Telecommunication Closet (centro-stella di piano) ci si mette uno switch lvl 2 (switch di piano) che raccoglierà le istanze di piano. Switch di accesso. Poi abbiamo l'*Equipment Room* (locale tecnico), ove abbiamo il centro stella di edificio (S1-P, S1-S). Apparati attivi nelle ER. La maggior parte dei flussi di traffico passa da qui. Bisogna quindi ridondare opportunamente. Devono essere abbastanza potenti! Poi abbiamo le dorsali verticali. Cavi, a rigore. LIU (*Lightguide Interconnection Unit*). Apparati di centro-stella. *CAVEDIO* (primaro & secondario). Le due dorsali scorrono in cavedi differenti. I produttori consigliano di mettere negli ER degli switch multilayer (lvl 2 - lvl 3). Centro-stella. Apparati molto più potenti. Switch di distribuzione. *Livello distribuzione*. Ridondanza sulle dorsali. Si creano dei possibili LOOP → STP → (R)STP ovviamente. Root bridge S1-P, ma interviene S1-S se malfunziona il primo.

Modifica automatica della priorità (previo identificativo). In particolare si avrà il decremento della priorità. 32768 di default, a decrementi di 4096. Confronti sul Root Path Cost. Path a stesso costo $\implies$ link alla stessa velocità. Spanning-Tree.

Ridondanze sull'interfaccia (fault-tolerant). Si fa girare l'HA-Linux (High Availability). Si ha lo switching in automatico qualora l'interfaccia (una delle due) vada giù. Se va giù uno switch del lvl distribuzione, con questo sistema è tutto OK. Switch $\leftarrow$ bridge. Oggi gli switch sono in HW (ASIC). Pensare anche alla capacità elaborativa! Gli switch del livello distribuzione devono essere ben potenti. Utilizzo delle VLAN (LAN virtuali). Per far comunicare host appartenenti a VLAN differenti abbiamo bisogno necessariamente del lvl 3. Possibilità 3 porte access al posto di 1 trunk la quale richiederebbe che il router soprastante sia *VLAN-Aware*. Link geografico che porta dal Router alla vera e propria Internet. Dobbiamo pensare anche eventualmente alla ridondanza dei link geografici. Pensare all'utilizzo dell'HSRP e del VRRP in questo caso. ECMP (Equal Cost Multipath Routing). Il discorso HSRP ha a che fare solo con il traffico in uscita. Pensare a più gruppi HSRP! ($\forall$ VLAN $\exists!$ HSRP), tipicamente.

Switch livello distribuzione lvl 2. Multi-layer switch. I router tradizionali lavorano non in hardware, ma con le CPU. I multilayer lavorano a lvl hardware! Matrice ASIC. I multilayer lavorano a lvl 3 come dei router! Lavorano in HW solo per il protocollo IP! Commutano ad altissima velocità a livello IP per reti di campus. Per i link geografici non ne avremmo bisogno (capacità trasmissiva bassa). Il multilayer lavora in HW per gli IP (maggioranza dei pacchetti). {IPX, AppleTalk} $\leftarrow$ legacy lvl 3 protocols. Multilayer switch. I router tradizionali vanno bene quando il $|VLAN| \ll +\infty$, il traffico è basso e le ACL sono limitate.

NO-FAULT TOLERANT NETWORK

Immaginiamo che le varie entità organizzative siano disposte per piani, su reti differenti.

- Le VLAN non attraversano i piani. Associare ai vari piani differenti Subnet IP, configurate $\forall$ piano. Indirizzo IP specifico x piano (SW-1) multilayer. Non c'è bisogno che lavoriamo a lvl 2. Qui abbiamo domini di broadcast differenti. In questo caso non abbiamo bisogno che lavori a lvl 2 $\rightarrow$ lvl 3 $\leftarrow$ router. Router che fa L'Internetworking tra host associati alla stessa Subnet IP. Ogni interfaccia dà verso una rete fisica differente. Con un router normale andrebbe tutto bene! Ma con un multilayer dovrei ragionare con LAN virtuali! (VLAN). *vlan database* (...) etc. Dobbiamo creare l'associazione VLAN $\rightarrow$ Subnet IP. Trattiamo le interfacce VLAN come se fossero fisiche! In questo caso le porte diventano Access VLAN. Bisogna creare delle LAN Virtuali nel Multilayer. Prima si associa la porta alla VLAN, con il corrispettivo tipo. Quindi:

  - 1) Creare VLAN;
  - 2) Associazione porta-VLAN;
  - 3) Assegnare indirizzo IP $\forall$ VLAN $\leftrightarrow$ utilizzando il comando *interface VLAN 2, ip address*;

  Questo se ragiono con le VLAN! Siamo su un Multilayer, che mi impone di ragionare con le VLAN. Bisogna fornire un indirizzo IP alle VLAN! Comando switch lvl 2. Complichiamo ora la situazione:

- VLAN interpiano. Adesso le entità organizzative si trovano su più piani! In questa situazione (SW-1) dovrà funzionare anche a lvl 2! NON solo a lvl 3. Sui vari piani potremmo avere lo stesso dominio di broadcast. Una VLAN corrisponde ad un dominio di broadcast. Adesso le porte saranno trunk! Le trame adesso viaggeranno taggate.

Dobbiamo pensare anche a VLAN aggiuntive host/Router. Switch multilayer = {Router con $|interfacce| = |VLAN|$ + switch lvl 2}. La VLAN 1 è sempre presente (default VLAN). Fasi:

- 1) Creare VLAN che servono;
- 2) (...) TRUNKING;
- 3) Associare indirizzo IP ALLA VLAN! Come se fosse un'interfaccia fisica;

Tipicamente su un link trunk c'è sempre quella di default. La si aggiunge per sicurezza. La VLAN 1 è utilizzata per la Gestione della Rete. È quindi importante che passi la 1 $\iff$ (*Network Management*): *switchport trunk allowed* (...).

FAULT-TOLERANT NETWORK

Prevediamo ridondanza nell'ER (Multilayer switch ridondato) e ridondanza di dorsale verticale. Se VLAN NON attraversano i piani $\iff$ entità organizzative separate $\forall$ piano. Lavoreremo in HW però. {{HSRP, VRRP}, GLBP}. Ma anche se ragioniamo con Router devo comunque lavorare con le VLAN!

OSPF possibile per convogliare il traffico Internet (WAN) $\rightarrow$ Server. High Availability.

- Fault-tolerant network con entità organizzative non spalmate sui piani. Porte switch multilayer. Accesso ed associazione VLAN alle varie porte. Prima si creano le VLAN (*vlan database* (...)). Si associano le VLAN alle varie porte dello switch. Poi si associano gli indirizzi IP (Subnet IP) alle interfacce VLAN. Modalità configurazione VLAN (*interface VLAN 4, ip address* (...)). Lo stesso dicasi per le altre interfacce. Presenti i router virtuali (HSRP in gioco, uno sarà Active e l'altro Standby). All'interfaccia virtuale VLAN 4 ad esempio dovremmo alzarle la priorità HSRP. Lo stesso per le altre VLAN ed agli altri gruppi HSRP associati. Si potrebbe utilizzare routing statico per convogliare il traffico sulla WAN (consegna indiretta). L'OSPF è invece gerarchico, suddiviso in aree. Le aree rappresentano i domini di routing. Dobbiamo decidere per i router a che aree apparterranno. Supponiamo che il costo di default sia 10. I costi sono associati alle interfacce! Parliamo delle interfacce che collegano i due multilayer! *interface vlan 5* $\iff$ dobbiamo sempre ragionare con le VLAN! Comandi OSPF. Complementare i bit della Submask (*Subnet Mask*). Gli IP rimangono i medesimi. Bisogna annunciare tutti i prefissi! Altrimenti ad esempio il server NON sarà raggiungibile. Comando *redistribute connected*. Il prefisso deve comunque essere annunciato! Però vogliamo che il server non appartenga al dominio di routing, dato che è un punto terminale. Quindi è opportuno che non viaggino su quell'interfaccia pacchetti di controllo OSPF (OSPF algoritmo *Link-State* (LS)). Nel caso del secondo multilayer, quello collegato al Router verso la WAN, dovremo sia annunciare che entrare a far parte del dominio di routing.

- Caso più complicato. Rete fault-tolerant, ma le stazioni di una stessa entità organizzativa dell'azienda possono risiedere su piani diversi! I multilayer devono adesso lavorare anche a lvl 2! (Problematiche relative ai domini di broadcast). Link lvl 2 anziché lvl 3. Significa che i multilayer dovranno comportarsi sia da router che da switch. HSRP/VRRP richiesto. In questo caso dovremmo per forza utilizzare l'STP! Spanning Tree Protocol. Possibili maglie a lvl 2. Collegamenti raddoppiati per problematiche dell'Affidabilità. STP va gestito opportunamente. Imponiamo che SW-1 sia il Bridge Active per tutte le VLAN, e che sia contemporaneamente Root Bridge per l'STP. Decrementi di priorità per l'STP di 4096. Le porte adesso saranno Trunk! ($\iff$ Link Trunk). Adesso bisognerà creare

tutte le VLAN (*vlan database*). Dopodiché si iniziano ad assegnare gli indirizzi IP alle varie LAN virtuali (VLAN). Vogliamo utilizzare HSRP/VRRP. SW-1 deve essere Router Active per tutte le VLAN! Ci deve ovviamente essere corrispondenza con i gruppi HSRP anche sul SW-2! Multilayer = (Router + switch collegato). Anche in questo caso possiamo pensare all'OSPF. Dobbiamo fare in modo che il traffico uscente verso l'esterno (WAN Internet) viaggi verso il link tra i due multilayer.

5.3.4 Progettazione Gerarchica

Modelli proposti per la progettazione della rete di Campus ad Alta Disponibilità. È bene basarsi sulla *progettazione gerarchica*. Vantaggi in termini di costo. Il costo dell'apparato dipende ovviamente dalle funzionalità interne. Possiamo avere una topologia modulare. I cambiamenti NON hanno impatto sugli altri dispositivi. Scalabilità assicurata e troubleshooting favorito. Comprensione migliore dei pattern di traffico. Riduce il numero di adiacenze delle CPU. Con una rete switchata a lvl 2 non abbiamo gerarchie. Un pacchetto broadcast a lvl 2 viene floodato su tutte le interfacce (CPU degli switch sprecate). Algoritmi di routing gerarchici (protocolli interior ed exterior). *Intra-domain* (Link State OSPF oppure DV), oppure *Inter-domain* (BGP). Il BGP è di tipo Exterior.

I produttori utilizzano la *three-layer Architecture*: {***Core***, ***Access***, ***Distribution***}. Alcune volte Core e Distribution sono collassate. Disponibilità 5-9. Downtime annuo di circa 5 minuti. Massimo downtime (come servizio di rete). Leggasi a tal proposito: *h-campus White-paper CISCO*. PIX Firewall CISCO. Livello {Access → Distribution → Core} in ordine.

- Il livello *Core* rappresenta il Backbone della rete. Dev'essere molto veloce e resiliente! Estremamente pronto a riprendersi dai cambiamenti. Il Core è un livello cruciale! Da lui dipende la connettività di tutta la rete. Filosofia "less is more". Molto snello ma veloce. Come è implementato? Tramite due switch ML collegati tra di loro. Segmenti LACP (802.11-AX IEEE). Due ML molto potenti. Ognuno di questi è collegato con due link ai due ML di edificio (Building). Nell'ER ci saranno due ML. Un ML del livello Core sarà collegato (in fibra monomodale) a quelli ER. Meglio costruire triangoli anziché quadrati (Si utilizzi ECMP per redistribuire il traffico). L'ECMP prevede di fare Load Sharing tra queste path. Qualora si dovesse avere qualche malfunzionamento i tempi di convergenza sono ben differenti rispetto alle costruzioni di quadrati. (ECMP - Equal Cost Multi-Path). Schema ben meditato dai vari produttori;

- Livello Distribuzione: Aggregare switch di accesso dello strato di accesso. Tipicamente il livello Distribution è una coppia di ML (Il funzionamento router/switch dipende dalla localizzazione delle VLAN se sono su più piani o meno). Stacked switch possibili eventualmente. A livello Distribution è molto importante il supporto a meccanismi QoS (per eventuali applicazioni Real-Time). I ML dovranno supportare questi meccanismi. Due ML per ridondare il dispostivo (per problematiche di disponibilità): HSRP/VRRP;

- Livello ACCESS: Raccoglie le utenze dei vari piani. Switch lvl 2 NON ridondati fisicamente ma con la IBR (In The Box Redundancy). Ridondanze: {Ventole, Hyervisor, Alimentazione}. Se i ML del livello distribuzione lavorano come router, non ci sarà bisogno dell'STP e tutti i link potranno trasportare traffici. Diversamente potrebbe non essere così per via dell'STP. Ridondanza IBR necessaria per il livello Access;

Gli switch lvl 2 dovranno supportare il QoS. *Confine di fiducia*, idealmente posto sui terminali, ai fini della QoS. Terminali come host, VoIP. Così marcheranno le trame a piacimento. Manipolazione opportuna maligna del DiffServ. Implementazione di servizi di sicurezza. *IEEE*

802.1X Access Control. Sia in rete Wireless (Access Point) che anche in reti Switchate (Autenticazione, Autorizzazione e ...). Ovviamente è presente anche la Crittografia. Controllo degli Accessi (a switch lvl 2). Altri meccanismi di sicurezza possibili. Redirezione su un Web Server che farà da proxy. Configurazione liste di accesso MAC (MAC filtering e DHCP Snooping, IGMP Snooping). *DHCP Snooping*. Meccanismo per evitare che qualcosa a caso possa diventare server DHCP. IGMP Snooping per ottimizzare le prestazioni se giunge un pacchetto Multicast. Suddiviso in gruppi. PoE (*Power over Ethernet*). Alimentazione tramite Ethernet dei VoIP oppure Access Point. Gli Access Point sono messi in posti difficili da raggiungere. *Wireless Access Point* (WAP) dotati tipicamente di PoE. Tra PoE e PoE+ cambia il livello di tensione.

RECAP

Prevedere a livello Distribution e Core ridondanze, ma non IBR (Best Practice). Meglio di no, ai fini del Convergence-Time. Meglio la ridondanza del dispositivo (esterna). Se si guasta un hypervisor ridondato, i tempi di convergenza sono maggiori. Reti ad Alta Disponibilità (100-200 ms senza IBR, 1-3 secondi con IBR). Downtime massimo di 5 minuti annuo. [SCALPING pacchetti di azioni. Trading estremo. Tick]; per il livello di Accesso è presente la IBR, anche per problemi di risparmio (il costo potrebbe cominciare ad aumentare, NON si ridonda il dispositivo di accesso), a meno che non ci troviamo nel *Data Center*, ove non ci sono le stazioni host utenti ma Server! Qui potrebbe essere utile ridondare esternamente. Il Server, contrariamente agli host, deve essere quanto più raggiungibile possibile.

ECMP permette di fare Load-Sharing. Più next hop possibili. Attuazione del load-sharing. Il Core lavora a lvl 3. È il Distribution che può lavorare anche a lvl 2. Due Next Hop possibili a livello Distribution, dal livello Core. Se vi sono dei guasti a livello HW (dei link), ce ne si accorge molto velocemente $\implies$ (tempi di convergenza estremamente veloci). È la scheda HW che si accorge del malfunzionamento a livello fisico. Nel caso della topologia a quadrato, i malfunzionamenti faranno sì che perderemmo molti più pacchetti ed i tempi di convergenza saranno inevitabilmente maggiori (sia per la via di convergenza dell'algoritmo di routing che tempi maggiori per via di costi maggiori! Non abbiamo DIAGONALI (Abbiamo algoritmi di routing, IGRP (CISCO) che permette Load-Sharing per link a differenti costi).

Importanza del link di collegamento (uno o più, utilizzo LACP) che collega gli ML del livello Distribution. Switch lvl 3. Subnet IP. Problemi di ridondanza. Se le VLAN attraversano i piani (ML lavorerà anche a lvl 2), Il link sarà di tipo trunk! Se NON ci fosse quel link di collegamento, si creerebbero degli strani pattern di traffico! Abbbiamo l'STP in gioco, ricordiamo. Due ML a livello Distribution e VLAN spalmate su più piani. Lvl 3 + lvl 2. Utilizzo HSRP/VRRP. Serve l'(R)STP. Se non ci fosse il collegamento, pacchetti di un piano viaggerebbero su switch di piani differenti. All'interno di un Telecommunication Closet (ER, livello distribuzione), il dispositivo lvl Access avrà gli switch collegati ad entrambi (con due collegamenti) gli ML del rispettivo lvl distribuzione. Bisogna però vedere quanta utenza per piano c'è! Dipende dalle porte dello switch. NON si collegano in cascata! Se uno di questi va giù, la rete verrebbe splittata in due! L'HSRP/VRRP riguarda solo il traffico in uscita. In entrata abbiamo la problematica ECMP. La rete IP verrebbe splittata. Non ci sarebbe più una rotta possibile! ICMP ritornerebbe il messaggio (*destination unreachable*), ed il pacchetto verrebbe scartato. Tutto questo si può risolvere utilizzando la tecnologia Stacked, non utilizzando quindi la porta di rete. Come se tutti fossero un UNICO switch. Possibile utilizzo LACP (IEEE 802.1AX).

Dimensionamento Link

Dimensionamento capacità Link. Access & Distribution. *Over Subscription*. Ogni 20 porte al Gigabit, si aggiunge un Gigabit dall'Access → Distribution (20:1). Invece livello Distribution → Core rapporto 4:1, preferibilmente 1:1 nei Data Center. Attenzione che con l'Over Subscription andremo a contemplare la congestione! Si risolve il tutto con meccanismi QoS in caso di congestione.

5.4 Parte Wireless

IEEE 802.11. Standard alla base del Wi-Fi; progettare una rete di Accesso. *CAPWAP* protocollo con la quale gli access point comunicano con il controller. Architettura centralizzata! Elementi di una rete wireless: {smartphone, tablet, PC, notebook} ← host wireless mobile e non. Normalmente tutti gli host wireless sono mobile. Link wireless: canale per comunicare con un altro host wireless oppure con una base station. Gli host wireless devono comunicare. Rete infrastrutturale (presenza di base station). Anche la Rete Cellulare (3G e 4G) è basato su modalità infrastruttura. Tutte le comunicazioni devono comunque passare per l'Access Point. Le diverse stazioni possono comunicare tra di loro mediante *infrastructured network*. *Coverage Area* e *Raggio di Trasmissione*. Mediante il raggio di trasmissione si può ricavare l'Area di Copertura, entro la quale tutte le stazioni riescono a ricevere pacchetti dalla loro station, se non vi sono interferenze. Esistono differenti tipologie di raggi:

- *Raggio di trasmissione*;
- *Raggio di Carrier-Sensing (o di Detection)*;
- *Raggio di Interferenza*;

Il Wireless è broadcast, quindi è richiesto un protocollo MAC che gestisca le collisioni. *HANDOFF* (o *HANDOVER*). In modalità infrastruttura gli host devono essere associati ad una base station (ripetitore nel caso del cellulare). Roaming nel caso cellulare $\implies$ passare da una base station alla successiva. Altra modalità prevista: ***AdHoc***. *Wireless AdHoc Network*, ove non abbiamo più la base station. NO infrastruttura. In tal caso tutte le facilities devono supportare le tecnologie di comunicazione. Ma bisogna sempre tener conto della Coverage Area! Quindi è richiesta una certa capacità di instradamento! Tutte le stazioni (host, end-system) dovranno comportarsi anche da router (funzioni di forwarding). La rete dev'essere auto-informante. Più hop wireless eventualmente richiesti. Tassonomia wireless: ({Infrastructured, not Infrastructured} x {Single Hop, Multi Hop}). *Wireless Mesh Network* $\in$ infrastructured $\wedge$ multi-hop. Le *VANET* sono un particolare esempio di rete MANET *Mobile AdHoc Network*. Nel caso VANET i nodi mobili sono i veicoli.

A seconda della tecnologia avrei diverse Aree di Copertura e Data Rate. La Coverage Area sferica (circolare) è solo ideale: ci sono degli ostacoli. Versioni diverse di Wi-Fi l'*802.11ae* è quello attuale. Differenti Data Rate (Mbps). Utilizzando parabole riesco a raggiungere distanze molto elevate (PPP technologies). Decine di km. Tecnologie cellulari (4G, LTE, etc.). Si parla già di 5G (2019-2020 come possibile data di uscita, in Giappone).

RECALL

Vi è un canale radio tra u e v se la potenza di irradiazione è superiore ad un certa soglia β (threshold) $\iff (P_r) \mid [P_r > \beta]$. Più è alto il Data Rate più alta sarà la soglia! Più è alto il Data Rate, più l'impatto del rumore sarà maggiore. Il rumore ha un impatto che aumenta

all'aumentare del Data Rate. Costellazioni troppo vicine e fitte. La potenza ricevuta sarà legata alla potenza trasmessa ed alla *Path Loss* dalla famosa

Definition 35. ***Equazione di Trasmissione di Friis***

$$[P_r(d) = \frac{G_t G_r \lambda^2}{(4\pi)^2 d^2} P_t]$$

La potenza ricevuta è quindi inversamente proporzionale al quadrato della distanza. d è la distanza, G_r è il Guadagno Antenna in ricezione, e G_t, analogamente è il guadagno Antenna in trasmissione. Si suppone il LoS (*Line of Sight*).

Effetti atmosferici, Meccanismi che dipendono anche dalla frequenza del segnale. Ostacoli lungo la LoS, Atmosferici, Riflessione del segnale sui vari ostacoli (oggetti di dimensioni molto maggiori della lunghezza d'onda del segnale). {*Shadowing* + *Reflections*). Effetti della degradazione del segnale. Più è alta la frequenza, più il segnale tende a comportarsi come la luce (effetto negativo) $\implies$ shadowing aumentato. {*Scattering*, *diffrazione*}. Segnale incidente che viene splittato in sottosegnali multipli (Splitting). Diffrazione legata agli spigoli degli oggetti (piccole dimensioni). Differenti sottocopie che si propagano. Diffrazione possibilmente utilizzata in maniera costruttiva.

Con lo scattering, diffrazione, riflessione abbiamo in ogni modo tante repliche del segnale che avranno seguito differenti path, e saranno quindi stati soggetti a diversi ritardi. Ovviamente più lunga è la Path, più il ritardo sarà maggiore. *MultiPath Propagation.* Improbabile ricevere lungo la LoS! LoS + repliche ricevute, Bisognerà vedere come giocano questi fattori! Dipende dagli argomenti. Normalmente pensiamo ad una deformazione, degradazione del segnale. Interferenza intersimbolica. Sovrapposizione di impulsi LoS e multipath, magari anche di segnali diversi. Allargare gli impulsi per far fronte a queste interferenze. Distanziarli nel tempo $\implies$ abbassare il data rate $\implies$ abbassare la banda. Limitare il data rate necessario.

Se il trasmettitore e ricevitore fossero vicini, si potrebbe sfruttare una *sequenza di training* nota al ricevitore (ORIGINALE) ed inviata dal trasmettitore, e si potrebbe sfruttare una procedura di *Equalizzazione* per correggere, compensare la propagazione MultiPath. Dipende dalla velocità di mobilità. Tecnica basata sugli equalizzatori va bene in ASSENZA di mobilità. Il MultiPath è collegato ovviamente all'Ambiente Circostante. Modello del canale che varia costantemente. Potenza in ricezione che varia molto velocemente al valor medio. *FADING*: dissolvenza del segnale {short-term fading, long-term fading}. Valor medio che varia nel tempo = long-term fading. Si può correggere.

Antenna Diversity. In posti molto vicini, il campo elettromagnetico potrebbe essere naturalmente differente! {RX1 e RX2}. Antenna Diversity. Access Point. In realtà quelle antenne si sfruttano anche in trasmissione. Tecnologie MIMO (*Multiple Input Multiple Output*).

5.4.1 Funzionamento

Ad ogni Access Point è associato un BSS (*Basic Service Set*). Set di stazioni minime che devono comunicare tra di loro. SNR fattore importante, per il calcolo della probabilità di errore sul simbolo. Data una certa modulazione digitale, affinché decresca il BER (*Bit Error Rate*), aumentiamo anche l'SNR aumentando la potenza in gioco. Data una modulazione digitale $\implies$ $\{P \uparrow, SNR \uparrow, BER \downarrow\}$. Dato invece un certo SNR (e BER), quindi non avendo la possibilità di variare la potenza, posso però scegliere la modulazione digitale più opportuna! Variare la modulazione digitale, che va a codificare meno bit per simbolo onde diminuire il BER. Dipende il tutto dall'Ambiente di Propagazione. Se l'Access Point non va al massimo dipende molto probabilmente dall'Ambiente Circostante.

Protocollo MAC CSMA-CD (*Carrier Sensing Multiple Access, with Collision Detection*). Sente sempre il canale anche quando trasmette. Se ciò che sente coincide con quanto trasmette è tutto OK, diversamente abbiamo una COLLISIONE! Sequenza di JAMMING. Effetto cattura. Anche se vi è una collisione, può essere che comunque tutto vada in porto! Dipende sempre dalle potenze in gioco. NO velocità differenti. Anche se sono parzialmente sovrapposte, non è detto che tutte e due vengano scartate $\iff$ *effetto CATTURA* (Fenomeno non deterministico, dipende dall'elettronica delle schede).

Il CSMA-CD NON funziona in ambienti wireless. Difficile gestire la collisione $\implies$ trasmissione + ricezione contemporanea $\implies$ *hidden terminal problem.* Per le reti Wi-Fi Infrastruttura abbiamo un rimedio per questo terminale nascosto. Con il CSMA-CD non si spreca più banda quando è rilevata una collisione! Nelle reti wireless invece il segnale continua inevitabilmente ad esser trasmesso. La COLLISIONE si verifica al RICEVITORE! È il ricevitore che riceve le trasmisioni! (Raggio di comunicazione = Raggio di trasmissione). In Wi-Fi si è trovato un rimedio. Per le VANET (reti AdHoc in realtà) è veramente cruciale. A rigore si dovrebbe parlare di *Carrier Sensing Range* > raggio di trasmissione, entro il quale si potrebbe decodificare il contenuto informativo. Esiste anche un problema aggiuntivo, quello del *Terminale Esposto*. Non abbiamo collisione, ma abbiamo comunque spreco di banda. Referring. Rimanda la trasmissione. Spreco di banda.

WiFi. Vari standard che si riferiscono al Wi-Fi (*IEEE 802.11*), in Legacy che prevedeva 1-2 Mbps {b/a/g/ac}. {Frequenza, Data Rate, Layer Fisico, Modulazione e Compatibilità}. ISM = *Industrial Scientific Medical* è una Banda ad Accesso Libero, sostanzialmente. Quella del 2,4 GHz è molto intasata come tecnologia di banda. Quale Data Rate sarà supportato dipende realmente dall'SNR, BER e dalla modulazione scelta {FHSS, DSSS}. Si riesce a far fronte al problema delle Interferenze. FHSS utilizzato in ambito militare. A seconda degli standard abbiamo tutte le caratteristiche sopracitate. L'*IEEE 802.11-D* (2009) utilizza la tecnologia MIMO (4x4 in TX ed RX). Con l'*IEEE 802.11-ac* (2013) si arriva addirittura sino a 6,9 Gbps (con MIMO 8x8). Attualmente siamo alla seconda generazione.

{Stazione, Access Point (per riferirsi alla base station)}. Parliamo di modalità Infrastruttura. BSS = gruppo di stazioni che lavorano con lo stesso Access Point. BSSID corrisponde normalmente con il MAC dell'Access Point. Più BSS possono essere messi insieme per formare un *Extended BSS* (Extended Service Set), identificato da un certo SSID consistente di 32 caratteri alfanumerici. Un sistema di distribuzione collega diversi Access Point (quindi differenti BSS) a formare un unico ESS.

{Portale: bridge per un'altra rete}. Alla fine tutto l'ESS è un dominio di broadcast. Protocollo MAC richiesto: CSMA-CA (*Collision Avoidance*). Nelle reti AdHoc mancano le base station (Access Point), ma non è supportato il multi-hop. 802.11 in modalità AdHoc single-hop. Il multi-hop non fa parte dello standard (*Relaying*). Terminologia: IBSS = *Independent BSS.* È imposto da quella stazione che inizia la comunicazione. Stazione che inizia la rete ed impone eventualmente l'IBSS. Canali 802.11 b/g parzialmente sovrapposti.

FDMA (*Frequency Division Multiple Access*). Accesso Multiplo. Sorgenti distribuite nello spazio. Banda affettata in sottocanali. Sottocanali della banda. NO interferenze. Invece nell'802.11 b/g abbiamo canali sovrapposti. Se sono dei canali distanti si può fare. Throughput aggregato maggiore, ovviamente, se i BSS lavorano su canali differenti. *Maximum EIRP*, potenza minima ($dbM \sim mW \implies (100mW = 20dbM \wedge 50mW = 10dbM)$).

BSS, modalità infrastruttura. Le stazioni si devono associare ad un certo Access Point! Si deve fare lo *SCANNING*. Le stazioni (base) inviano un *BEACON* ogni 100 ms normalmente. Questi BEACON contengono un *timestamp* per sincronizzare le trasmissioni (e quindi anche le varie stazioni che vogliamo comunicare); conterrà anche il {Beacon Interval; tutte le trasmissioni passeranno dall'Access Point. Questo avviene anche attraverso il *Traffic Indicator Map* (lista

di trame bufferizzate per una certa lista di stazioni), parametri di trasmissione che consentono l'effettiva associazione}.

Ricerca reti wireless: scanning per i beacon!

- *SCANNING PASSIVO* $\iff$ una stazione aspetta un beacon;
- *SCANNING ATTIVO* probe da parte della stazione. Viene sollecitato un AP mediante questa sonda:
  - *Directed Probe* nella quale cercheremo un certo SSID;
  - *Broadcast Probe* utile per Service Discovery.

C'è bisogno di una fase di Autenticazione. Captive Portal, Website redirection, oppure IEEE 802.1X

Richiesta di associazione. L'Access Point tipicamente blocca le richieste da parte delle stazioni wireless. Autenticazione *Shared-Key* (with nonce). L'approccio tipico è però quello basato sullo standard *IEEE 802.1X*. Anziché del WEP ora si utilizza il WPA2-PSK con AES (*Advanced Encryption Standard*). IEEE 802.1X + EAP (*Extensible Authentication Protocol*). L'EAP prevede tanti metodi di autenticazione (proprietari & non proprietri). L'802.1X mette in comunicazione stazione ed AP. Il *RADIUS* è il protocollo che mette in comunicazione AP e server RADIUS. Roaming + handover: Esiste uno standard, l'*IEEE 802.1F* che consente di mettere in comunicazione due AP in contesa per una stazione. Nella realtà molte cose sono proprietarie! Protocolli proprietari.

5.4.2 CSMA-CA (Collision Avoidance)

IEEE 802.11 MAC (CSMA-CA) *Collision Avoidance*. Funzionalità protocolli MAC: accesso al mezzo. C'è la possibilità di effettuare una prenotazione della risorse virtuale mediante pacchetti {RTS, CTS}. MAC PDU, controllo errori, segmentazione e reassembly delle trame. Tre tipologie di frame: {Controllo, Data Transfer, Management}. Gli {RTS, CTS} consentono di far fronte al problema del terminale nascosto. Tre metodi per l'Accesso la canale. Uno obbligatorio. Non funziona qui il CSMA-CD! Serve il CSMA-CA (Collision Avoidance). {Problema del terminale nascosto e quella del terminale esposto}; metodo opzionale con RTS, CTS, ed un altro metodo *Contention Free* basato sulla tecnica del Polling (applicazioni Real-Time). L'Access Point fa il polling alle differenti stazioni. Polling: richiedere loro se hanno qualcosa da trasmettere. Passato a livello di definizione. Non implementato. DCF (*Distributed Coordination Function*) ingloba i primi due metodi. PCF è il terzo (*Point Coordination Function*).

Le stazioni devono essere SINCRONIZZATE! Il tempo è suddiviso in slot. Lo slot time può essere differente per via delle diverse varianti. Nella modalità Infrastruttura sono gli Access Point a sincronizzare le varie stazioni con i beacon (pacchetti) che contengono il timestamp. Nelle AdHoc c'è un meccanismo che permette la mutua sincronizzazione. "Mobile Communication".

Sistemi sincronizzati. Il protocollo MAC prevede che quando si deve accedere al canale una stazione debba aspettare un certo tempo. Controllato attraverso differenti IFS (*Inter Frame Space*), ciascuno con differenti priorità. L'IFS più piccolo è il SIFS (*Short IFS*). Priorità più elevata. Utilizzato quando c'è da inviare gli ACK, CTS, polling response (DCF). PIFS per il PCF = (SIFS + 1 slot time). Il DIFS (*Distributed IFS*) è a priorità più bassa $\iff$ DIFS = SIFS + 2 slot time. La priorità NON è legata al traffico, ma è la priorità nell'Accesso al canale. Al SIFS è legata la priorità più alta, al DIFS quella più bassa.

Funzionamento

CSMA/CA (Collision Avoidance). C'è una versione leggermente modificata. Quando una stazione (host od access point) ha bisogno di accedere al canale, deve prima ascoltarlo! Deve basarsi sul CCA (segnale *Clear Channel Assessment*). C'è la possibilità di ritrasmettere i pacchetti dati. Qualora il canale fosse libero per almeno DIFS unità di tempo, può trasmettere immediatamente il pacchetto. Se invece risulta occupato oppure occupato prima di DIFS, allora quella stazione deve attendere un tempo aggiuntivo dopo che si è liberato. Aspetterà DIFS ulteriore tempo, poi la stazione entrerà in contesa (Contention). Questa fase impone di aspettare un ulteriore tempo aggiuntivo, espresso in slot. (Random backoff time (multiplo dello slot time)) scelto all'interno della finestra di Contesa (*Contention Window*). es $[0, 7]$. 5 può essere il mio numero di contesa. Decremento lo slot time. Quando lo slot time arriva a 0, allora la stazione è libera di trasmettere. Decrementi unitari di slot time $\forall$ slot. Se durante la fase di decremento il canale si occupa, si freeza il tempo, e si riprenderà a decrementarlo solo quando si libererà di nuovo il canale.

Con questo meccanismo si evita la Collisione, e se il canale è libero per almeno DIFS si può trasmettere immediatamente, questo dà l'idea di un canale abbastanza vuoto. Traffico basso come intensità. Se il canale si occupa prima, il canale è abbastanza congestionato (fase di contesa necessaria). CA = Collision Avoidance. Meccanismo di attesa aggiuntiva. CCA = Clear Channel Assessment: segnale di canale libero. Sempre nel Range di Sensing. Meccanismo di Basic DCF. Quando il backoff time arriva a 0, la stazione trasmette immediatamente. Se alla fine vi è una collisione, questa viene rilevata con gli ACK, e se qualche ACK non è riscontrato (ricevuto) allora vi è una ritrasmissione. Dopo la Collisione, la CW viene raddoppiata. Ha senso, perché evidentemente ci sono molti tentativi di accesso al canale. Ad ogni ritrasmissione (a collisione avvenuta), viene raddoppiata la CW (Contention Window).

ACK (acknowledgements). Immaginiamo trasmissioni dati. Se una destinazione ha ricevuto correttamente un pacchetto dati, dovrà riscontrarlo con un ACK. Dopo un SIFS dall'avvenuta ricezione. Maggiore priorità nell'Accesso al canale $\iff SIFS < DIFS$. Maggior priorità. Se un'altra stazione rileva il canale occupato da un ACK, comunque deve entrare in contesa (fine trasmissione + DIFS + Backoff time). Il protocollo è di tipo S&W (Stop and Wait), di tipo Unicast in relazione ai pacchetti dati. Se il pacchetto è Broadcast, allora gli ACK non servono.

Backoff esponenziale (2-exp). CW si raddoppia $\forall$ collisione. Probabilità di Collisione Wireless elevata. Il MAC funziona così: $\exists$ numero max di ritrasmissioni. $\forall i,\ CW_i \leq CW_{max}$. Ci si basa tutto sul sistema degi ACK. Protocollo S&W. Pacchetti RTS/CTS per far fronte al problema del terminale nascosto. Sender/Receiver ed altre stazioni. RTS = *Request to Send*; campo *duration* che si riferisce alla durata dell'intera trasmissione. Una volta trasmesso l'RTS, il destinatario dovrebbe replicare con un CTS = *Clear to Send* dopo un SIFS. Il ricevitore è prioritario rispetto ad altre stazioni (SIFS). Ricevuto il CTS, dopo un SIFS il sender è pronto ad inviare. Ricevuto il pacchetto dati, il ricevitore aspetta SIFS e poi manda l'ACK. La collisione si può verificare nell'RTS! Il pacchetto RTS contiene il campo duration. Ogni stazione che riceva l'RTS, estrapola il valore del campo duration, e lo mette nella sua NAV (*Net Allocation Vector*). Quindi questa stazione aspetta per NAV (RTS) e per NAV (CTS), inclusivamente. Questo meccanismo fa fronte al problema del terminale nascosto. Il Raggio di Carrier Sensing è molto più grande di quello di trasmissione. Basato sulla rilevazione della portante, e si basa sul fatto che sappiamo, rileviamo che qualcuno sta parlando (sua portante). Carrier-Sensing Range o Detection Range. Raggio di trasmissione $\neq (\cdot)$ (più piccolo). Con l'RTS/CTS abbiamo il problema risolto (meccanismo aggiuntivo). Tutte le trasmissioni (dati) passano per l'Access Point. Non si possono inviare pacchetti dati direttamente alle altre stazioni. RTS. Tutte le stazioni che riceveranno l'RTS, guarderanno al campo duration e faranno il deferring (settaggio del NAV e successivo idling). Nel mondo delle AdHoc questo meccanismo non funziona più (es

VANET, reti veicolari). Nelle AdHoc abbiamo problemi con il raggio di interferenza / area di interferenza, centrata sul ricevitore. L'Area di Interferenza dipende dalle distanze sender/receiver. Tutti i nodi all'interno dell'Area di Interferenza sono per definizione nodi nascosti! Con le reti infrastrutturali invece non si pone proprio il problema! Infatti nodi nascosti facenti parte dell'Area di Interferenza e non coperti dall'Area RTS/CTS non saranno associati! Lavoreranno a frequenze differenti.

La collisione si può verificare nell'invio dell'RTS! Ma se il pacchetto dati ha delle dimensioni confrontabili con l'RTS, non ha senso! Sempre meglio avere collisioni sull'RTS che su un pacchetto dati! Threshold utilizzata (Soglia). Se i pacchetti dati superano questa soglia, allora si utilizza effettivamente il meccanismo aggiuntivo RTS/CTS.

Struttura frame *IEEE 802.11*. [frame control, duration, *address 1*, *address 2*, *address 3*, seq control (utile per gestire i duplicati, quando l'ACK si perde, ad esempio), payload, CRC]. Altri campi di controllo: (type, subtype, etc.). Il significato dei campi indirizzo dipende dai campi *To AP* e *From AP*. Gli Indirizzi 1 e 2 hanno a che fare con il ricevitore fisico della trama ed il trasmettitore fisico della trama, rispettivamente. FISICO $\neq$ LOGICO, a meno che non stiamo nelle AdHoc. *address 3* contiene invece il router logico, Access Point, BSSID.

5.4.3 Mobilità in IPv6

Lo standard al riguardo dell'*Architettura Centralizzata* è l'*IEEE 802.1d* Backward Learning. Handover sulla stessa Subnet IP non causa problemi. Quando un nodo mobile passa da una rete IP ad un'altra, il suo indirizzo IP cambierebbe! Possibile abbattimento delle connessioni TCP. MIPv6 soluzione adottata: *Mobile IPv6*. Parliamo di un Mobile Node che si muove. *Correspondent Node*: qualche altro nodo che comunica con il Mobile Node. Il Correspondent Node utilizzerà un indirizzo IP basato su quello del Mobile Node, chiamato *Home Address*. Il Mobile Node avrà una sua *Home Link*: rete fisica alla quale è agganciato. *Home Address Prefix*. Subnet IP sulla Home Link (Subnet A) indirizzata dal prefisso Home Address Prefix (HAP). Home Address corrispondente al Mobile Node. A questa Subnet sarà collegato un Router detto *Home Agent*, e funzionerà come proxy, e redirigerà le connessioni (traffico) destinate al Mobile Node qualora questo non risieda più nella Home Link. Supponiamo che il Mobile Node si sia spostato in una *Foreign Link*. Associatosi a questo nuovo abbiamo Access Point, ascolterà i Router Advertisement della nuova Foreign Link (mondo IPv6). Acquisirà un certo indirizzo IPv6 nuovo (CoA - *Core of Address*). Bisogna fare in modo di recapitare questo CoA all'Home Agent per redirigere correttamente il traffico. Tutta questa procedura viene fatta dal Mobile Node mediante *Binding Update* (BU). L'Home Agent dovrà intercettare tutti i pacchetti dedicati all'Home Address del Mobile Node. Dopo il BU il messaggio sarà autenticato dall'Home Agent e processato. Successivamente farà il *Binding Ack* al Mobile Node, e l'informazione sarà inserita nella sua *Binding Cache* (BC). Inviato l'ACK ed inserita la Entry, invierà un PNA (*Proxy Neighbor Advertisement*) (il quale sostituisce l'ARP). Neighbor Solicitation + Neighbor Advertisement.

Si viaggerà col tunnel IP! Tutto transiterà tramite l'Home Agent! I pacchetti nel tunnel viaggeranno in maniera sicura tramite *IPSec*. Il routing NON è ottimizzato perché si passa per l'Home Agent di volta in volta (banda sprecata)! Si può ottimizzare il routing una volta che si è fatta l'Associazione ed il Tunneling, comunicando direttamente al Correspondent Node l'associazione CoA / Home Address (Binding Cache sul CN) (sul Corresponding Agent).

5.4.4 Architetture

Architettura Autonoma

L'Access Point (*FatAP*) implementa completamente e termina le funzioni dell'802.11, in maniera tale che i frame sulla wired LAN siano frame 802.3. Ogni AP è gestito indipendentemente. I FatAP possono prevedere *VLAN Tagging*, basato sull'SSID che i client usano per associarsi all'AP (***Multiple SSID***), e fornisce anche funzioni router-like, tipo come server DHCP.

I FatAP hanno anche altre capacità, come ad esempio le *Access Control Lists* (ACL), funzioni QoS, collegate alla priorità IEEE 802.1p. Il principale punto debole del FatAP è la complessità, che lo rende utile solo in piccole installazioni di rete.

Architettura Centralizzata

Una motivazione importante è la locazione degli AP. Mirando a fornire una connettività radio ottimale per le end stations, gli AP sono tipicamente installati in aree difficili da raggiungere. I Network Manager preferiscono installare gli AP
solo una volta e che non vi sia bisogno di intervenire con complesse manutenzioni.

Gli AP sono per questo collegati ad un *WLAN Controller* (WLC) mediante l'utilizzo di un **tunnel sicuro**. Questo tunnel deve assicurare un basso ritardo su questi pacchetti. Il protocollo utilizzato per comunicare è il CAPWAP - *Control And Provisioning of Wireless Access Points*. CAPWAP è responsabile per la scoperta e l'elezione di un WLC da parte di un AP. I pacchetti di controllo CAPWAP sono criptati. Il carico wireless 802.11 degli AP verso il WLC è incapsulato in pacchetti CAPWAP. Anche questi pacchetti dati possono essere criptati, ma questo porterebbe ad una degradazione del throughput.

Con l'architettura *Split Mac* l'implementazione delle funzioni MAC è divisa tra l'AP ed il WLC. GLi AP sono leggeri, nel senso che detengono solo una minima parte delle funzionalità del MAC. I vendor si comportano differentemente al riguardo. Tipicamente, gli AP:

- Gestiscono funzioni MAC real-time, tipo la generazione dei beacon, risposta alle sonde, processazione di frame di controllo (RTS, CTS)...
- Lasciano tutte le funzioni NON real-time (autenticazione, associazione) da processare al WLC.

Gli AP forniscono crittografia wireless mentre utilizzano WLC per scambiarsi le chiavi.

Il WLC invece, gestisce i firmware e le configurazioni degli AP controllati, effettua RRM - *Radio Resource Management*, basata sulla configurazione e sul monitoraggio degli AP controllati. Attraverso i messaggi di controllo CAPWAP, gli AP mandano statistiche (numero di tentativi di trasmissione, numero di frame erronei, ...) al WLC; Per esempio, se due AP controllati da un WLC si interferiscono l'uno con l'altro, il WLC può mandare un segnale ad un AP per ridurre la sua forza.

Il WLC gestisce il rinforzo del QoS e fornisce un filtraggio ACL-based. Gestisce la mobilità a livello 2 e livello 3; in altre parole funge da Mobile IP Home Agent. Comunque l'estensione delle varie funzionalità dipende anche dalle implementazioni dei vendors.

5.4.5 Progettazione

La prima generazione dei prodotti 802.11ac supportano data rate fino a 1.3 Gbps. La seconda generazione di prodotti 802.11ac supportano data rate fino a 3.5 Gbps. Operando alla massima capacità, le apparecchiature 802.11ac sono capaci di superare di gran lunga le prestazioni fornite da un 1000BASE-T uplink. Le prossime tecnologie forse permetteranno

di utilizzare 10GBASE-T, o possibilmente dual (od anche quad) 1000BASE-T uplinks, per supportare il multi-gigabit.

Dovrebbe essere utilizzato il cablaggio di categoria 6A. Si potrebbero considerare fibre multimodali OM3/OM4 laddove i data rates siano superiori a 10 Gbps e per locazioni esterne dove le distanze siano superiori a 100 m.

L'*ISO/IEC TR-24704* ha proposto ciò che è considerato un ottimo schema di posizionamento degli access point wireless. La progettazione è basata su un array di celle esagonali (tight-fitting hexagonal cells). L'Area di Copertura di ogni cella è limitata ad un raggio di 12 metri. *TR-24704* raccomanda di terminare il cavo per ogni cella ad un armadio ubicato quanto più possibile al centro della cella. *TIA TSB-162-A* suggerisce un una griglia quadrata delle aree di cablaggio, ognuna larga circa 18 metri. In previsione dell'evoluzione dell'802.11ac, la revisione di questo standard propone un cablaggio di categoria 6A. TIA-4966 raccomanda che la densità di AP all'interno di grandi spazi indoor debba essere basata sull'occupazione media.

La progettazione dell'architettura di cablaggio deve essere attentamente basata sull'ambiente RF, i livelli di interferenza e le sorgenti, capacità/bisogni futuri, requisiti di cablaggio e di potenza richiesti. Idealmente, l'architettura di cablaggio e l'analisi di copertura dovrebbero lavorare a braccetto per fornire la massima capacità e flessibilità per soddisfare le necessità correnti/future degli utenti. Quando si ha posizionato un AP, è fortemente consigliato effettuare un'indagine RF per ottimizzare la posizione dell'AP all'interno della cella. Un analisi dell'ambiente RF è quindi molto consigliata. In ambienti di minima capacità e bassi requirements, una semplice valutazione della propagazione RF può essere sufficiente. Alcuni programmi permettono ai progettisti di reti di fornire in input il layout del sito, di condurre modellazione degli AP, e di comparare simulazioni ed ispezioni RF.

La debolezza del segnale può essere dovuta a numerose variabili, tipo la presenza di un materiale ostacolante per la RF, come ad esempio armadi, oppure proprio degli impedimenti fisici come dei grandi muri. Aggiungere AP molto spesso vuol dire migliorare significativamente la copertura.

Oltre ad assicurare un adeguato segnale RF, dovrebbe essere considerato anche il *throughput aggregato*. Lo spazio deve essere diviso in celle rettangolari, come raccomandato dall'*TIA TSB-162-A* o dall'*ISO/IEC TR 24704*. Le posizioni degli AP e la densità possono essere modificati in virtù dell'analisi dell'occupazione.

È consigliato fornire almeno due cablaggi da ogni cella, dal momento che ogni cella della griglia potrebbe anche avere due o più AP. Oltre alla progettazione dell'ambiente RF e della gestione della capacità, c'è un vasto numero di fattori in gioco da considerare nel cablaggio e nel posizionamento degli access point wireless (accessibiltà, requisiti di potenza, estetica, ...).

5.4.6 Reti Wireless AdHoc

È un insieme di mobile host capaci di formare una rete temporanea senza il supporto di un'infrastruttura fissa: {*infrastructureless*, *selforganizing*, *self-configuring*}. Le operazioni di network, come routing e gestione delle risorse, sono effettuate in maniera ***cooperativa*** e ***distribuita***.

A causa del range di trasmissione limitato, il *routing multi-hop* è tradizionalmente utilizzato. Ogni nodo può comportarsi da host e da router: un pacchetto è forwardato da un nodo ad un altro fino a che non raggiunge la destinazione.

Importanti scenari di utilizzo:

- Applicazioni militari;
- Operazioni di emergenza;

- ***Comunicazioni veicolari***;
- Comunicazioni subacquee

A causa della *mobilità* associata ai nodi, la topologia di rete può sperimentare *continui cambiamenti*: Differenti livelli di potenza tra i nodi introduce *link asimmetrici*. Le risorse sono tipicamente limitate, vincolate (larghezza di banda, potenza batteria, etc.).

Principali problemi delle reti Wireless AdHoc:

- Schema di accesso al mezzo;
- Routing;
- Multicasting;
- Protocollo a livello di trasporto;
- Fornitura QoS;
- Auto-organizzazione;
- Sicurezza;
- Gestione dell'energia;
- Indirizzamento e service discovery;
- Scalabilità

L'efficacia degli handshake RTS/CTS è basata sull'assunzione che i nodi nascosti siano all'interno del raggio di trasmissione dei ricevitori (in modo tale che essi possano riceverlo correttamente). Qualche nodo fuori dal range di trasmissione del ricevitore potrebbe comunque interferire con il ricevitore. I nodi all'interno del *raggio di interferenza* (R_i) del ricevitore sono chiamati *nodi nascosti*.

Abbiamo tre raggi radio per quanto riguarda le comunicazioni wireless:

- ***Raggio di Trasmissione*** (R_{tx}): rappresenta il raggio all'interno del quale un pacchetto è correttamente ricevuto se non ci sono altre interferenze da altre stazioni radio. Il raggio di trasmissione è principalmente determinato dalla potenza di trasmissione e dalle proprietà di propagazione radio, come ad es. l'attenuazione;
- ***Raggio di Carrier Sensing*** (R_{cs}): raggio all'interno del quale un trasmettitore può iniziare la carrier sense detection. Questo raggio è principalmente determinato dalla antenna sensitivity e dalla potenza di trasmissione;
- ***Raggio di Interferenza*** (R_i): definisce l'*area di interferenza*:

  $$A_i = \pi R_i^2$$

  attorno al ricevitore. Tutti i nodi localizzati in quest'area sono i nodi nascosti del ricevitore. Quando il ricevitore sta ricevendo un pacchetto, se un nodo nascosto comincia una trasmissione, avverrà una collisione al ricevitore;

Il raggio di trasmissione e quello di Carrier Sensing sono fissi e sono infuenzati dalle proprietà radio. Il raggio di interferenza non è fisso ma è relativo alla distanza trasmettitore-ricevitore e può andare ben oltre il raggio di trasmissione.

5.4.7 VANET

Le VANET non sono nient'altro che reti AdHoc ove i nodi sono però veicoli. L'*IEEE 802.11p* è l'emendamento approvato per il supporto alle comunicazioni veicolari. Il principale problema delle VANET è il routing. L'alta velocità dei veicoli causa frequenti cambiamenti della topologia. Sarebbe molto difficile mantenere aggiornate le tabelle di routing. Il traffico sul canale wireless incrementerebbe a causa dello scambio dei messaggi di controllo. IDEA: *Routeless Routing* → ***Intelligent Broadcasting***.

Contention Based Forwarding (CBF). Solo un nodo è il contenitore dei pacchetti. Tutti i nodi che ricevono un dato pacchetto e sono nella direzione ***source*** → ***destination*** calcolano il *waiting time* e partecipano al processo di contesa. Il nodo al quale per primo scade il timer inoltra il pacchetto in avanti. Ogni nodo che ascolta la conversazione abbandona i pacchetti (discard con ACK implicito).

5.5 MPLS

MPLS. Principi di progetto QoS. MPLS fornisce QoS. C'è un campo della sua intestazione con il quale è possibile fornire QoS: rete in grado di fornire un differente livello di servizio in base alle varie applicazioni. Meccanismi QoS: {classification and marking, disciplina di coda, POLICING & SHAPING = serve a sagomare il traffico in modo da rispettare il profilo di traffico dichiarato}. Resource reservation (IntServ, definito dall'IETF). Es. Fase di Admission Control (controllare che vi siano risorse). Marcatura lvl 3: CodePoint DiffServ, lvl 2: PCP CodePoint. Marcatura ulteriore a livello 2: MPLS. MPLS EXP (Experimental). MPLS è una rete fatta da router. Problematiche di routing... DiffServ (PHB, Per-Hop Behavior). IntServ non era scalabile. Riguardava i vari flussi. DiffServ definisce invece delle classi di servizio. I flussi appartenevano alle varie classi di servizio. Per-Hop Behavior (PHB): comportamento vario nel forwarding del pachetto. Vari PHB: {Expedited Forwarding (EF) → DSCP = 46, Assured Forwarding: {AFxy: AF1, AF2, AF3, AF4. Caso peggiore: AF13, Caso migliore: AF41}, Default: best effort (rete Internet), Class Selector. Questo PHB è stato definito per avere retrocompatibilità con il vecchio servizio ToS}.

5.5.1 Funzionamento

Essa è una tecnologia di Internetworking basata su delle label, mediante le quali si può fare forwarding. Reti Internet a commutazione di pacchetto: {Datagram, Virtual Circuit (VC)}. Le ultime si basano sulla presenza di circuiti virtuali, instaurati a priori. *Label Swapping*. Tabella di instradamento: quale label utilizzare nel prossimo hop. MPLS lavora così. MPLS nell'ambito di Internet: fornisce delle sembianze connection-oriented alla normale Internet. La label è un indice che viene utilizzato per accedere ai campi delle tabelle di forwarding. Oggi i router sono molto veloci! Potrebbe essere vista come falsa motivazione di utilizzo per MPLS. Utilizzo alternativo: VPN, Traffic Engineering. Normalmente si sceglierebbe la Path a minor costo nel forwarding/routing. Ci potrebbero essere delle path non utilizzate! Sfruttare altre path. Bisognerebbe però fare *Source Routing* (SR). È la sorgente che poi alla fine sceglie la path da utilizzare. Questo consente di sfruttare altri link, tipicamente costosi in ambito metropolitano. È bene utilizzarli al meglio!

Con MPLS si possono emulare degli switch lvl 2 virtuali! *Virtual Private LAN*. Si realizza una Switched LAN ove non c'è questo dispositivo lvl 2! Switch lvl 2 virtuali. Switched LAN a livello metropolitano / urbano. Interconnettere due nodi sfruttando uno switch lvl 2. Sedi eventuali che sono separate da suolo pubblico. Bridge remoti per interconnettere LAN differenti,

in remoto. VPN con MPLS, pensando al routing (lvl 3). Rete che commuta a lvl 2, senza protocolli di routing lvl 3.

Con MPLS possiamo trasportare pacchetti (frame) {IPv4, IPv6, Ethernet (con VPLAN), PPP}. Dominio MPLS: {insieme contiguo di router che parlano MPLS}. *Label Switch Router*. LSR *Ingress/Egress* node, A dei pacchetti che giungono sull'Ingress Node LSR, si appiccica una label (od eventuale stack di label). Rete che si basa su delle label, conservate a lvl 3, sulla base delle quali viene fatta la commutazione (con Label Swapping). Qui accade la stessa cosa delle reti VC. *Label Switched Path* (LSP). {Ingress LSR, Egress LSR}. Quando si riceve un pacchetto all'ingresso, bisognerà capire che path seguire. *Forwarding Equivalence Class* (FEC), classe che raggruppa flussi di pacchetti che seguono tutti una stessa certa rotta (vengono trattati allo stesso modo per quanto riguarda l'instradamento). FEC: gruppi/flussi di pacchetti. Stessi parametri QoS. Definire Path e parametri QoS = FEC. Se parliamo di Path, dobbiamo pensare ad una serie di etichette (locali!). FEC, Path, serie di etichette, label. Queste etichette sono distribuite utilizzando il protocollo LDP (*Label Distribution Protocol*). Nel caso TE, viene utilizzata un'estensione dell'RSVP, detto RSVP-TE (utilizzato nell'ambito dell'architettura DiffServ). Bisognerà configurare tutti i router lungo una certa path. Sarà importante verificare che vi siano le risorse necessarie! La FEC si riferisce ad un gruppo di pacchetti che sono trattati alla stessa maniera (parametri QoS) e seguono una stessa rotta. Nodi MPLS-Aware. Cosa farà un nodo di ingresso? Quando riceve un pacchetto, deve capire a quale FEC associare quel pacchetto. Saranno utilizzate alla fine delle label (o stack di label) da appiccicare, sulla base del FEC. Si ricordi che le label hanno un significato LOCALE! Si utilizzeranno per il Label Swapping (LS). Significato locale ma univoco! Con le VC non è il caso di scegliere delle label con significato globale! Sarebbe difficile da gestire il controllo sull'univocità in particolare. Con il significato locale la SCALABILITÀ è garantita.

Label MPLS

Le label sono di 32 bit. Primi 20 bit effettivi. 3 bit per il supporto QoS (EXP). Bit S: vale 1 per l'entry più vecchia, 0 altrimenti. Campo TTL (time-to-live) → stessa funzione del campo TTL dell'IP. Stack di label possibile. Problema di fissare dei valori nel campo *Protocol Type* del lvl 2. Immaginiamo che vi sia un pacchetto che attraversi una rete Ethernet. PT che riflette il protocollo delle PDU trasportato. Se aggiungiamo un pacchetto IP, dobbiamo considerare l'MPLS! 8847. *Ethertype Value*. Ethernet v2. Se si sfrutta il sottolivello MAC, il supporto è fornito dall'LLC, con l'estensione SNAP (SNAP/LLC). Se invece viaggiano su PPP, il codice è 0281 (sempre trasporto MPLS).

VPL (*Virtual Private LAN*). Organizzazione che deve interconnettere le sedi. Potrebbe dotarsi di una rete privata: {Apparato attivo + link di comunicazione}. Acquisto dell'Infrastruttura e costi di manutenzione. Tutto ciò sarebbe estremamente costoso. VPN (*Virtual Private Network*). Emulazione di una rete privata mediante Internet, il quale è una rete condivisa. Sulla rete privata i traffici andranno poi separati. Customer che si rivolge ad un fornitore di servizio VPN. Il provider si deve preoccupare della separazione del traffico sulle diverse VPN.

MPLS in termini di VPN. Service Provider al centro (MPLS VPN). Terminologia associata al servizio MPLS VPN. PE = *Provider Edge*, ovvero il router collegato al *Customer Edge* (CE). Servono come collegamento al lato provider. Gli altri router del provider che non son connessi ad un CE sono detti (P) *Provider* router. Esiste un altro modello, detto *ATM Relay*. VPN con ATM o Frame Relay (lvl 2 technologies).

- Overlay Model. ATM è una rete a commutazione di pacchetto VC (Virtual Circuit). Si stabiliscono dei circuiti virtuali, tra i nodi del Customer. Nodi da interconnettere eventualmente tramite ATM. Connessioni Full-Meshed. 53 byte, lunghezza fissa dei pacchetti

ATM. Si noti l'assenza dell'amministratore di rete ATM. La rete ATM offrirebbe un servizio lvl 2 (Comunicazione di nodi adiacenti), nonostante vi siano problematiche di routing (a lvl 3). Tutti i CE vengono visti come Nodi Adiacenti. Modello VPN Overlay. I nodi di commutazione ATM (ATM Switch) NON vengono proprio visti dal mondo IP. L'amministratore si dovrebbe però preoccupare di instaurare $(n-1)$ circuiti virtuali, qualora si aggiunga un nuovo CE. Overlay Network. Router CE visti come Router IP. Algoritmo DS (*Distance Vector*) oppure LS (*Link State*). ROUTER ADIACENTI! Anche se ovviamente non lo sono.

Advertisement dei prefissi della VPN. VPN (rete privata virtuale che inerconnette le varie sedi del customer) con:

- MPLS P2P VPN. Con il modello MPLS P2P (peer-to-peer) adesso vi sarà adiacenza tra Customer Edge (CE) e PE. Servono BGP di tipo Interior (iBGP). Non abbiamo più il modello Overlay. Dobbiamo sfruttare i PE, i quali tramite iBGP trasporteranno questi prefissi. Distribuzione dei prefissi. Non c'è più tutto quell'onere a carico dell'amministratore di rete. Se il Customer deve aggiungere un nuovo sito (CE), sarà sufficiente collegarlo (fare il peering) con un PE. Penserà il BGP a trasportare tutti i prefissi (verranno annunciati tutti tramite BGP). SCALABILE.

Rete MPLS condivisa da più Customer. Creare/configurare diverse VPN ai vari Customer. Diverse istanze VRF (*Virtual Routing/Forwarding*). VPN: rete privata virtuale. Si dovrà comportare come una rete privata! Traffici separati, ed eventualmente cifrati! Serve separazione. Se vogliamo anche cifratura utilizziamo IPSec. I traffici dovranno subire diverse routing instances. Diverse tabelle di routing a seconda delle VPN. È un must il collegare un sito allo stesso PE con differenti interfacce! $\forall VPN\ \exists!$ interfaccia sul PE. Per ogni interfaccia dovrà esistere quindi un unico VRF $\iff \forall interface\ \exists! VRF \iff \forall VPN\ \exists! VRF$. VRF servirà a capire, per quella VPN, cosa fare di un determinato pacchetto IP. Il tutto si comprenderà mediante IP di destinazione. L'LSR Ingress dovrà appicccare le label rispettive! Appiccicare al pacchetto IPv4 due etichette: *VPN Label* ed *IGP Label*. Quest'ultima collega i due PE attraveso router di tipo P, fino a giungere sul PE detto Egress LSR, che "strapperà" le etichette. Eventualmente con Label Swapping $\forall$ attraversamento di un router P. Sulla base di queste etichette il pacchetto viaggerà mediante un protocollo Interior. Arrivato il pacchetto a destinazione, si sfrutterà la VPN Label per identificare la VPN (associazione del pacchetto ad una certa VPN).

5.5.2 QoS

MPLS che consente di fornire QoS tramite quei 3 bit EXP (Experimental). QoS: dobbiamo idealmente pensare ad una QoS end-to-end. Ma non è così semplice! Perché è difficile andare a fidarsi degli utenti della rete. Dovrebbe essere l'utente a marcare/classificare il traffico generato. SLA (*Service Level Agreement*), con il DiffServ es. PHB (AF41). Problema di fiducia: si definiscono dei *Trust Boundaries*: punti ove è consentito che si marchino i pacchetti. Altre considerazioni: QoS in ambito di campus. Non solo applicabile a link geografici, ove la banda è bassa. Over Subscription 20:1 (Campus). La congestione entra in gioco! Traffico contemporaneo. I meccanismi di congestione sono quindi importanti anche a livello locale (Campus) $\leftrightarrow$ Trust Boundaries: punti ove la marcatura lvl 2 / lvl 3 è accettata. Idealmente il Trust Boundaries dovrebbe essere sugli endpoint. VoIP giustificabile (EF necessario). Marcatura diretta del telefono. Problema generale comunque di fiducia. Marcatura a livello Access di Campus. Bisogna dotarsi di apparati che supportino questi meccanismi QoS (CISCO ex). Potrebbero marcare anche a lvl 3! È importante fare il policing alla sorgente, ed intervenire in

casi di attacchi maligni (SCAVENGER: Più basso livello possibile. Anche minore rispetto al Best-Effort).

Raccomandazioni presenti su documenti CISCO (basati su RFC), basati sulle varie classi di servizio. 11 classi di servizio, al quale associare un certo PHB. CodePoint associabile. La Best Effort corrisponde a tutto a 0. Vecchio sistema IPP (IP-*Precedence*). La voce ha dei requisiti ancora più stringenti della videoconferenza!

Valori anche per la marcatura a lvl 2. Raccomandazioni per il PCP dell'IEEE 802.1Q. Debba coincidere con il MPLS EXP. Marcatura a lvl 2. Valori scelti in modo tale che potessero esser tirati fuori direttamente (valori uguali all'IPP). Lo 0000 non corrisponde alla più bassa priorità. Coerenza tra servizio QoS lvl 3 e quello lvl 2: disciplina di accodamento importante {inf, sup} per le bande associate alle varie Classi di Traffico. 25% della banda di un link per best effort e max 33% per il traffico Audio/Video (Weighted Fair Queueing) con WFQ. Range di traffico indicato per le varie bande. $\forall$ link di uscita $\exists$ molteplici code HW! es. 5 code HW per il link di uscita. Se ne avessimo 11, potremmo scegliere proprio le raccomandazioni in atto. Capability in termini di code HW sui link di uscita. QoS livello end-to-end. Paramtri QoS da rispettare (SLA). Service Level Agreement. Ad esempio due sedi da interconnettere con {latenza 150 ms, packet loss probability 1%, jitter 30 ms}. Latenza end-to-end! Latenza tipicamente inferiore a quella massima end-to-end!

Mapping tra marcature (ciò che abbiamo pensato come CoS e ciò che è possibile fare nella rete del provider). $|CoS|$ del provider $\neq |CoS|$ customer, tipicamente. Mapping non banale, non indifferente. Tutto questo mapping sarà incluso, deciso nello SLA (Service Level Agreement):

5.6 Security

5.6.1 IPSec

Security (*IPSec*). Criterio di progettazione per la sicurezza delle applicazioni. VPN = Virtual Private Network. Un'organizzazione può interconnettere diverse sedi con delle reti private. È possibile emulare delle reti private con una rete condivisa come Internet. Si tratta sempre di un'emulazione. Attraverso questa rete condivisa saranno trasportate più VPN. Necessità di separazione. Possibilmente il traffico dev'essere cifrato. MPLS con sistema VRF. Idealmente la crittografia ci dovrebbe essere, altrimenti il traffico potrebbe essere sniffato.

IPSec e SSL/TLS. Realizzazione di VPN mediante IPSec. Due tipologie di VPN: {*Site-to-site*, *Remote User*} VPN.

- *Site-to-site VPN*: Le prime sono utilizzate per interconnettere due gateway (anche chiamata gateway-to-gateway). Diversi router presenti durante il percorso. IPSec è in esecuzione sui gateway. Qui tutte le comunicazioni sono cifrate prima che attraversino la VPN. Potremmo basarci anche su degli ISP (Internet Service Provider). Message Authentication (es. MD5). I benefici che IPSec fornisce non sono necessari a livello host-to-host;

- *Remote User VPN*: Il processo VPN non starà sul router di confine dei vari siti (gateway), ma sarà in esecuzione proprio sul computer che invierà dei messaggi all'azienda. Computer che utilizzerannno per collegarsi alla sede principale. *VPN Gateway* sulla sede principale, ove saranno concentrate tutte le connessioni. Remote User VPN. In questo caso il dipendente è come se fosse dentro la rete aziendale! Host-to-gateway VPN. Due tipologie di tunnel (virtuali).

IPSec ha a che fare con una suite di protocolli:

- *Authentication Header* (AH), che non fornisce sicurezza. Quindi non molto utilizzati. MAC (*Message Authentication Code*). Messaggio inviato da una CERTA sorgente ed indica che il messaggio non sia stato alterato;

- *Encapsulation Security Payload* (ESP);

IPSec a livello network! La AH viene lasciata perdere. *Security Association* (SA). UNIDIREZIONALE! Half duplex. IPSec opera in: {*Transport Mode*, *Tunnel Mode*}:

- In Transport, IPSec NON protegge l'intestazione del pacchetto. Lo utilizziamo quando vogliamo delle comunicazioni end-to-end. Endpoint A e B. A ha in esecuzione IPSec e B pure. SA A $\rightarrow$ B. Su A ci sarà un software che si preoccuperà di mantenere le Informazioni di Stato necessarie. A livello di cifratura, un pacchetto IP viene concatenato con un *ESP trailer*. Payload IP + ESP trailer cifrato. Viene aggiunto nell'intestazione tra header IP ed il resto, una porzione chiamata *ESP Header*. Tutto dall'ESP header sino all'ESP trailer viene hashato per l'appunto nell'*ESP Auth Trailer*, finale;

- In Tunnel Mode, supponendo di essere un site-to-site, viene cifrato tutto quanto! Adesso l'header IP originario associato all'IP Payload viene cifrato! Viene aggiunto un nuovo header IP contenente informazioni sul tunnel! Come prima, dall'ESP header all'ESP trailer viene tutto hashato. {IP SRC = R1, IP DST = R2} esternamente. A metà abbiamo: {IP SRC = A, IP DST = B}.

Se avessimo un VPN host-to-gateway, il tunnel partirebbe dall'host A! Adesso l'unica differenza in tunnel mode è che la configurazione di indirizzi è: {IP SRC = A, IP DST = R2} nell'header esterno, mentre {IP SRC = A, IP DST = B} nell'header più interno (cifrato). Algoritmo MAC (Message Authentication Code). Tutte queste informazioni sono informazioni di stato! Un'entità IPSec magari ha più SA associati! Vengono mantenute nel SED (*Security Association Database*). Occorrerà sapere pure quali pacchetti dovranno basarsi sull'IPSec e quali no! (SPD) - *Security Policy Database*. In questo SPD si dice se un pacchetto ad una certa destinazione ha bisogno o meno dell'IPSec. Ci sarà un protocollo, detto IPSec IKE che si occuperà di automatizzare il processo di configurazione (*Internet Key Exchange*).

Site-to-site. Altre informazioni importanti: SPI (*Security Parameter Index*). Sia su R1 che su R2 dovranno essere memorizzate. Accedendo al SAD, R1 capirà che cosa fare (favorirà opportunamente il pacchetto da inviare). A destinazione, bisognerà capire a quale SA appartiene quel determinato pacchetto! Nell'ESP header ci saranno SPI e Seqno (numero di sequenza). R2 ha tutte le informazioni di stato per quelle SA nel suo SAD. Nell'intestazione abbiamo quindi SPI e Seqno, quest'ultimo per evitare attacchi Replay. Nel trailer vi sono: {Padding, padding length + next header}. I cifratori agiscono per blocchi di una predeterminata lunghezza. Quindi ci vuole padding, ed anche mantenere traccia della sua lunghezza! Nel next header ci sarà il protocollo che ha generato quel payload IP. Nel next header più esterno ci sarà il protocol type dell'ESP. Nel next header invece ci sarà l'informazione del protocollo {IPv4, IPv6}. Relativo quindi alla tipologia del pacchetto IP (od IPv4 od IPv6).

La struttura dati che IPSec gestisce è l'SPD, contiene le regole per dire quali pacchetti sono trattati con IPSec e quali con IPvX. SPLIT TUNNELING utilizzato: il traffico verso Internet passa sempre *unencrypted*. Per un mobile worker, l'accesso verso Internet viene rediretto sempre verso la rete aziendale! NO accesso diretto verso Internet. Maggior controllo di ciò che fanno i dipendenti. Questo discorso vale sempre per Remote User. Dispositivi blindati. Comunque bisogna passare dalla rete aziendale; VPN Gateway e VPN Concentrator.

5.6.2 Firewall

Firewall: "tagliafuoco" sul confine della rete aziendale. Tipicamente abbiamo dei firewall distribuiti. Varie tipologie di firewall:

- *Stateless Firewall*: SLPF Stateless packet filter. Sono soggetti ad IP Spoofing;
- *Stateful Packet Filter*: Check della connessione TCP. Lo stateless agisce solo in base agli indirizzi del pacchetto. Gli stateful tengono invece conto della connessione TCP, al loro SYN infatti vengono create delle informazioni di stato. Ma anche gli stateful hanno dei problemi! Sono soggetti al DoS. Tabelle con un numero max di entries. Attacco DoS possibile. Non vanno oltre a lvl 4!
- Esistono anche tipologie di firewall a livello applicativo;

RFC2827 per proteggere le ACL da attacchi IP Spoofing. Replicazione eventuale anche nel router dell'ISP. *Application Gateway* = Proxy. Drawback: problemi di prestazioni. Firewall a livello applicativo: $\forall$ applicazione $\exists$ proxy. Proxy server HTTP Software, processo applicativo che potrebbe costituire un bottleneck. Il proxy agirà da Server nei miei confronti. *Origin Server*. Il proxy si comporterà come client nei confronti dell'Origin Server (OS) e stabilirà una connessione TCP con esso. Si sfrutta il servizio DNS per eventuali associazioni.

IDS (*Intrusion Detection System*), Deep inspection (Probe utilizzate) fino a livello applicativo. Possono accorgersi di un attacco e generare un allarme che invieranno al Network Manager che eventualmente potrebbe anche prendere delle contromisure. NIDS. Network IDS. Se ne mettono tipicamente più di uno. Sensore IDS centrale che manderà alla fine l'allarme. I router possono fungere da firewall con le ACL. Nella DMZ vi sono i Server pubblici. Alla fine non ci sarà un solo firewall! Vi sarà un firewall distribuito tipicamente, onde evitare persino attacchi interni! Parametri di sicurezza.

J2EE (Java EE). Firewall tra L3 e la server farm (Application, Web e Data). Well-firewall capability. PIX della CISCO, adesso sono chiamati ASA. Oppure un SW da installare su un host. Firewall 1 della CheckPoint.

5.6.3 IPSec e Firewall

- ***TRUSTED***:

  VPN Gateway. IPSec Gateway avrà un'interfaccia collegata al WAN Router, ed un'altra alla Intranet aziendale (Intranet). Problema legato all'eventuale troppa fiducia data ai Remote User ed ai Remote Site. Variante possibile: IPSec collegato al firewall anziché al WAN Router. No vantaggi in termini di sicurezza, ma possibile Logging. Con l'ulteriore variante:

- ***UNTRUSTED***:

  Abbiamo un controllo aggiuntivo. Una volta decifrato il traffico, passa dall'Internet Firewall e poi c'è un Optional NIDS! Come prima abbiamo la variante Logging, che non aggiunge particolari vantaggi in termini di sicurezza. Eventuale collo di bottiglia sull'Internal Firewall.

- In alcuni casi la funzionalità firewall è già inclusa nell'IPSec Gateway (VPN Gateway), ad esempio il CISCO ASA. In ogni caso serve comunque il Corporate Firewall. Tipicamente esso sarà anche ridondato. Adesso non costituisco più colli di bottiglia. (RAS = NAS) serve per l'accesso remoto NON gestito da VPN.

5.6.4 Sicurezza nelle Applicazioni

Progettazione per la sicurezza nelle applicazioni. Posta (email) & http. Nel primo caso, abbiamo un Corporate Firewall che gestisce il traffico Internal → External. Esso dovrà prendere le mail ricevute da un *Outside SMTP Server* e le collocherà su un *External SMTP Server*, che le immagazzinerà opportunamente prima di inviarle all'*Internal SMTP Server* (SMTP, POP3 o IMAP) e viceversa. Eventuale replicazione degli antivirus su tutti gli SMTP. Eventuale dedica di alcune macchine per lo SCANNING.

A livello web, tipicamente in un'architettura Java EE, abbiamo un Web Server nella DMZ ed i server Application e Database in Server Interni. (Eventualmente collassati). Si configurino le ACL opportunamente.

Capitolo 6

Training Facility

6.1 Homeworks 1

6.1.1 Esercizio 1

In fig.1 sono illustrate due topologie di rete per il collegamento tra due sedi di un'azienda. Valutare in quale caso l'affidabilità del collegamento tra le due sedi è maggiore, supponendo che i router IP siano fault-free e che i link di collegamento (una rete fisica di una internetwork IP è considerata un "link" tra due nodi IP) abbiano la stessa affidabilità.

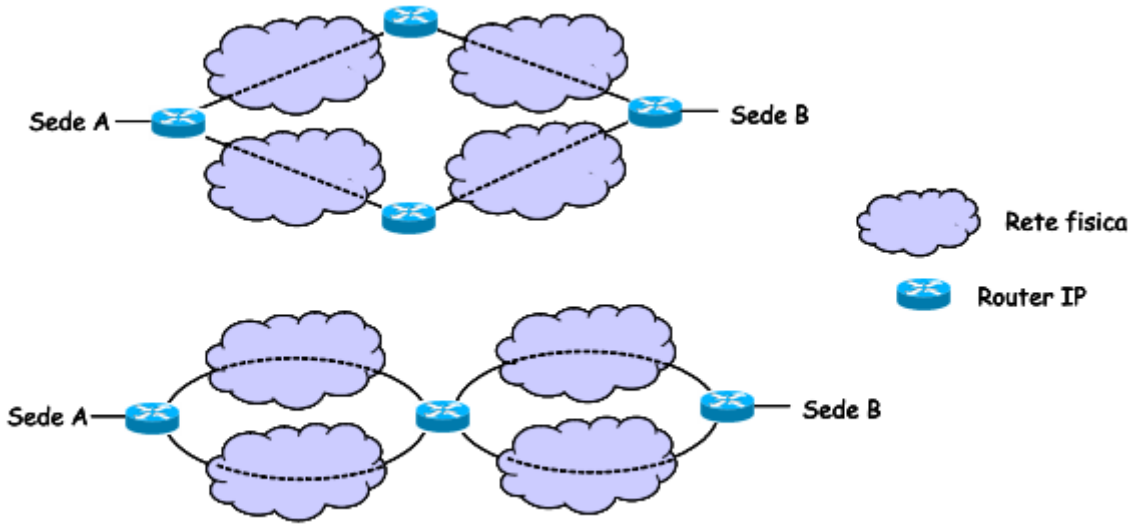


Fig.1

Procediamo naturalmente utilizzando la tecnica degli RBD, date le ipotesi in gioco ed introducendo le opportune ipotesi di indipendenza dei vari link (dei guasti dei vari link in particolare), ponendo $R := R(t)$:

- *parallelo di due SERIE*: $[R_1 = 1 - (1 - R^2)^2]$;
- *serie di due PARALLELO*: $[R_2 = [1 - (1 - R)^2]^2]$;

Naturalmente, è di gran lunga migliore l'affidabilità del secondo collegamento (coefficienti maggiori in numero). Il numero di possibili percorsi interrotti è minimizzato, per via della maggiore opportunità di collegamento a fronte di guasti.

6.1.2 Esercizio 2

Si consideri un servizio di rete basato su una batteria di m file server. Il servizio è erogato (UP) fino a che almeno l file server sono funzionanti. Ciascun utente accede al servizio tramite una workstation. La porzione di internetwork mostrata in figura collega la LAN lato utente alla LAN lato server. Supponendo che:

- l'affidabilità dei router IP e delle due LAN sia molto alta;
- l'affidabilità di ciascun link di collegamento tra i router sia $R_l(t)$ (una rete fisica di una internetwork IP è considerata un "link" tra due nodi IP);
- l'affidabilità di ciascun file server sia pari a $R_f(t)$;
- l'affidabilità di ciascuna workstation sia pari a $R_w(t)$;
- tutti i malfunzionamenti siano indipendenti l'uno dall'altro

si valuti l'affidabilità del sistema tramite il quale un utente accede al servizio.

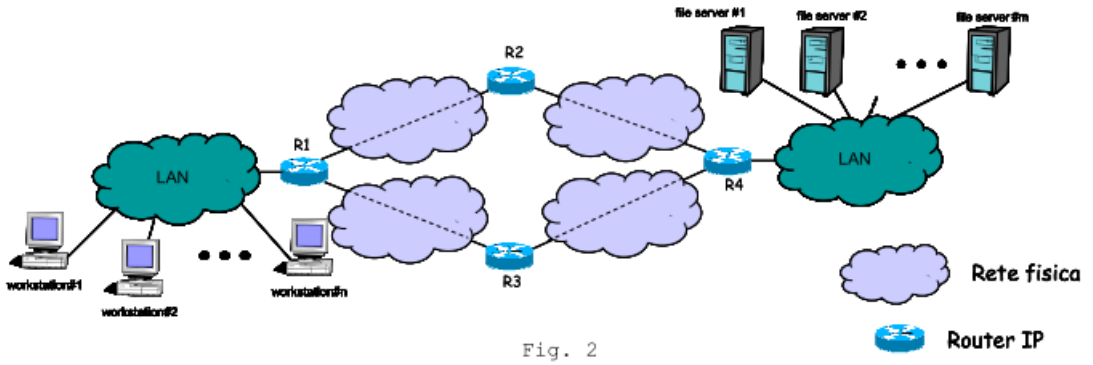


Fig. 2

Risoluzione: m file server. Servizio UP quando abbiamo l ne sono funzionanti. Blocchi indipendenti e struttura semplice $\Longrightarrow$ RBD utilizzabile:

$$R_w(t)[1-(1-R_l(t))^2]^2 \sum_{i=l}^{m} \binom{m}{i} R_f(t)^i [1-R_f(t)]^{m-i}$$

6.1.3 Esercizio 3

In fig.3 è illustrata la topologia di rete per il collegamento tra le tre sedi di un'azienda. Valutare l'affidabilità del collegamento della sede A con la sede C supponendo che i router IP siano fault-free, che i link di collegamento abbiano la stessa affidabilità e che non vi siano loop per le rotte. Fig.3 Si supponga, adesso, che il link R2-R3 abbia una capacità C_1 (bit/sec) molto minore della capacità C_2 (bit/sec) dei rimanenti link e che siano attive due sessioni di comunicazione: una prima sessione S1 i cui pacchetti viaggiano dalla sede A alla sede C lungo

la rotta R1-R4-R3, una seconda sessione S2 i cui pacchetti viaggiano dalla sede B alla sede C lungo la rotta R2-R4-R3 (visto che $C_1 << C_2$). Si assuma che i pacchetti associati alla sessione S1 arrivino al router R1 secondo un processo di Poisson a velocità X_{S1}, che i pacchetti associati alla sessione S2 arrivino al router R2 secondo un processo di Poisson a velocità X_{S2}. Le lunghezze dei pacchetti siano v.c. i.i.d. con distribuzione esponenziale a valor medio L bit. Si supponga, infine, di poter trascurare i ritardi di elaborazione e propagazione e che i buffer dei router abbiano una dimensione molto grande. Si derivi la CMTC omogenea che puo' essere utilizzata per lo studio della distribuzione del ritardo di pacchetto associato alla sessione S1.

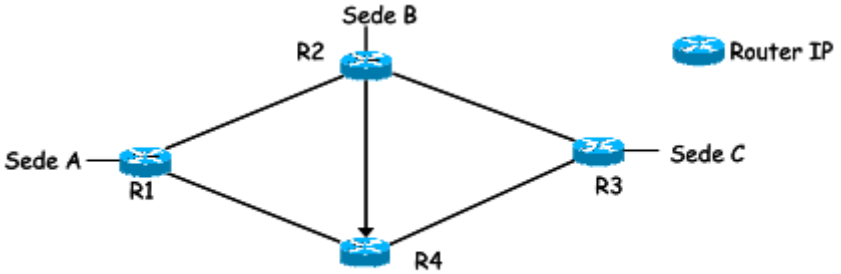


Fig.3

Risoluzione: supponiamo router fault-free e link di collegamento alla stessa affidabilità, e NO LOOP tra le rotte!

Si valuti l'affidabilità supponendo L_{2-4} HALF-DUPLEX (unidirezionale), e supponendo che NON vi siano LOOP. NO struttura semplice, sebbene siano blocchi indipendenti a valle della riduzione. Applichiamo il *Key-Item Method*, supponendo L_{4-3} come elemento chiave. Alla fine avremo:

$$R_S(t) = R_a(t)R_i(t) + R_b(t)(1 - R_i(t))$$

- Consideriamo il primo SCENARIO: L_{4-3} cortocircuitato: abbiamo il parallelo di un semplice blocco con la serie di un blocco e del parallelo di due blocchi:

$$[R_a(t) := 1 - (1 - R)(1 - R[1 - (1 - R)^2])]$$

- Nel secondo scenario, L_{4-3} è un circuito aperto. Dato che se si attraversa L_{1-4} non si può risalire per via dell'unidirezionalità di L_{2-4}, e d'altro canto se si prende L_{2-4} a valle dell'attraversamento di L_{1-2} non si può comunque risalire sulla sinistra per via dell'ipotesi dell'assenza di LOOP, tale scenario collassa nella serie di due semplici blocchi: $R_b(t) := R^2$.

Riassemblando opportunamente troviamo:

$$R_S(t) = [1 - (1 - R)(1 - R[1 - (1 - R)^2])]R + R^2(1 - R)$$

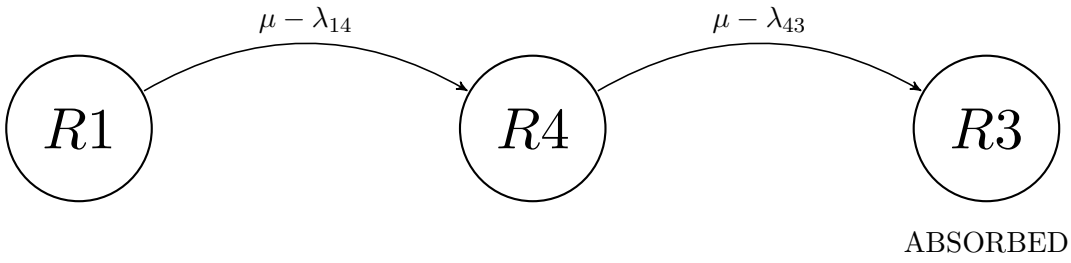
Per quanto riguarda il secondo punto, eseguiamo il seguente mapping tra rotte e velocità di arrivo dei vari flussi:

$$\begin{cases} X_{S1} - POISSON \\ X_{S2} - POISSON \end{cases} \mapsto \begin{cases} R1 - R4 - R3 \\ R2 - R4 - R3 \end{cases}$$

Sia $C_1 \ll C_2$ la capacità del link R2-R3 ($[C_i] = [bit/sec]$). Abbiamo che: $L \sim EXP(L\ bit)$. Si trascurino i ritardi di propagazione ed elaborazione, e si supponga che i buffer dei router abbiano una dimensione molto gramde ($\iff d_1, d_2 = +\infty$). La seguente è la CMTC omogenea per lo studio della distribuzione del ritardo dei paccchetti associati alla sessione S_1, con rotta: R1-R4-R3. Sfruttando le EQUAZIONI del TRAFFICO troviamo le velocità di arrivo sui vari link che ci servono, in concomitanza anche alla derivazione del rate di trasmissione:

$$\begin{cases} \lambda_{14} = X_{S1} \\ \lambda_{43} = \underline{X_{S1} + X_{S2}} \\ \dfrac{C}{L} = \mu[\dfrac{bit}{sec\ bit} = \dfrac{bit}{sec}] \end{cases}$$

Abbiamo la seguente CMTC, con STATI TRANSITORI: {R1, R4} e STATO ASSORBENTE {R3}, rappresentato dal seguente DTT:



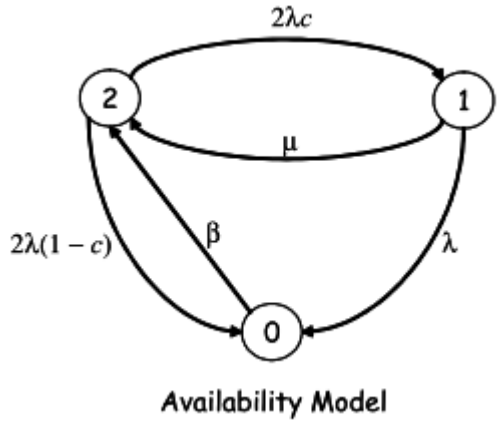
I tassi sono riassunti di seguito:

$$\begin{cases} \mu - \lambda_{14},\ R1 - R4 \\ \mu - \lambda_{43},\ R4 - R3 \end{cases}$$

6.2 Homeworks 2

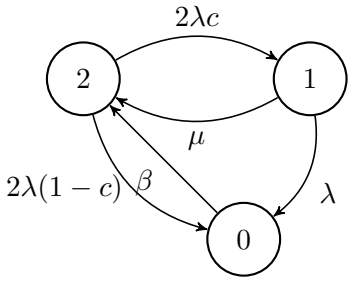
6.2.1 Esercizio 1

Un servizio di rete è basato su un sistema ridondante parallelo con due dispositivi. La CMTC illustrata in figura 3 rappresenti il modello di disponibilità del sistema. Valutare il Mean Time To Failure (MTTF) del sistema supponendo che lo stato iniziale della CMTC sia lo stato 2. Valutare, in funzione delle probabilità di regime, il numero medio di "Failure" che si hanno durante una finestra temporale di durata pari a T unità di tempo.



	State	Meaning
UP states	2	Two devices are working
	1	One device is working
	0	Failure State

Tale è il DTT:

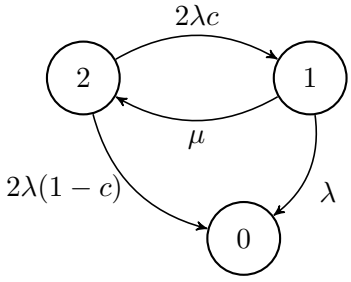


Sistema ridondante parallelo con due dispositivi. c coverage factor, $\lambda = -q_{22}$ velocità totale di uscita dallo stato 2. STARTING STATE = $\{2\}$.

$$\begin{cases} T_f \sim EXP(\lambda) \\ T_r \sim EXP(\beta) \end{cases}$$

Abbiamo due dispositivi. $\pi_2(0) = 1 \iff \{2\}$ STARTING STATE.

Il modello di disponibilità è quello sintetizzato nel precedente DTT. Per quanto concerne il solo calcolo dell'MTTF, possiamo procedere con la tecnica degli stati assorbenti e dei tempi medi di assorbimento. Consideriamo quindi il relativo modello di calcolo per l'affidabilità: si effettui il detach del ramo di transizione $0 \to 2$. Adesso la CMTC contiene uno stato assorbente: $\{0\}$:



FAILED

Naturalmente la catena in questa configurazione NON è però ERGODICA $\iff \nexists(\pi_i = \lim_{t\to+\infty} \pi_i(t) = \Pr\{X(t) = i\})$.

- a) **MTTF**

  Per il primo punto, utilizziamo i risultati dell'esercizio visto precedentemente: **CMTC with Absorbing States**, e procediamo con la tecnica dei tempi medi di assorbimento. Scriviamo la matrice generatore dei tassi di transizione:

$$\bar{Q} = \begin{bmatrix} -2\lambda & 2\lambda c & 2\lambda(1-c) \\ \mu & -(\mu+\lambda) & \lambda \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3}$$

  Adoperiamo ora una restrizione della matrice $\bar{Q}$ considerando solo gli stati transitori, sapendo che $S = N \cup A$:

$$\bar{Q}_N = \begin{bmatrix} -2\lambda & 2\lambda c \\ \mu & -(\mu+\lambda) \end{bmatrix}$$

Sappiamo che:

$$\begin{cases} L_1(\infty) < +\infty \\ L_2(\infty) < +\infty \\ L_0(\infty) = +\infty \end{cases}$$

Quindi scriviamo l'equazione matriciale: $\bar{L}_N(\infty)\bar{Q}_N = -\bar{\pi}_N(0)$, avendo cura di alleggerire la notazione: $L_i(\infty) := L_i \ \forall i$:

$$\begin{bmatrix} L_2 & L_1 \end{bmatrix} \begin{bmatrix} -2\lambda & 2\lambda c \\ \mu & -(\mu+\lambda) \end{bmatrix} = -\begin{bmatrix} \pi_2(0) & \pi_1(0) \end{bmatrix}$$

Sapendo che: $\{\pi_2(0) = 1,\ \pi_1(0) = \pi_0(0) = 0\} \implies$

$$\begin{cases} L_2(-2\lambda) + L_1\mu = -(\pi_2(0) = 1) \\ L_2(2\lambda c) - L_1(\mu+\lambda) = -(\pi_1(0) = 0) \end{cases}$$

Ricordiamo che $[\lim_{t\to+\infty} L_i(t) = L_i(\infty)]$. Abbiamo:

$$\begin{cases} L_1 = \dfrac{-1+2l\lambda L_2}{\mu} \\ L_2(2\lambda c) - \dfrac{(2\lambda L_2 - 1)}{\mu}(\mu+\lambda) = 0 \end{cases} \implies \begin{cases} \begin{bmatrix} 2L_2\lambda\mu c - 2\lambda\mu L_2 - 2\lambda^2 L_2 + \mu + \lambda = 0 \\ 2L_2\lambda(\mu c - \mu - \lambda) + \mu + \lambda = 0 \\ L_2 = \dfrac{-(\mu+\lambda)}{2\lambda(\mu c - \mu - \lambda)} = \dfrac{(\lambda+\mu)}{2\lambda(\mu+\lambda-\lambda c)} \end{bmatrix} \\ L_1 = \dfrac{(-1+2\lambda L_2)}{\mu} = \dfrac{-1+\frac{2\lambda(\lambda+\mu)}{2\lambda(\mu+\lambda-\lambda c)}}{\mu} = \\ = \dfrac{-\mu-\lambda+\lambda c+\lambda+\mu}{(\mu+\lambda-\lambda c)\mu} \end{cases}$$

Quindi abbiamo:

$$MTTA_1 + MTTA_2 = MTTF_1 + MTTF_2 = L_1(\infty) + L_2(\infty) := L_2 + L_1 =$$
$$= \frac{(\lambda+\mu)}{2\lambda(\mu+\lambda-\mu c)} + \frac{c}{(\mu+\lambda-\mu c)} = \frac{\lambda+\mu+2\lambda c}{2\lambda(\mu+\lambda-\mu c)}$$

Dimensionalmente torna!

- b) **Numero medio di "Failure" in finestra temporale** T:

  Per completare l'esercizio, riprendiamo il modello di calcolo per la disponibilità, ove esiste la distribuzione di regime, e calcoliamo il numero medio di failure che si hanno durante una finestra temporale di durata $T[s]$. Sommiamo le frequenze delle transizioni relative all'evento: "Ingresso nello stato di failure", esprimibile ovviamente in funzione delle probabilità di regime:

$$f_F := \pi_2 2\lambda(1-c) + \pi_1\lambda = \lambda[2\pi_2(1-c) + \pi_1]$$

Moltiplicando per T troviamo effettivamente il numero medio di entrate nello stato di FAILURE durante T:

$$\bar{N}_F = f_F T = T\lambda[2\pi_2(1-c) + \pi_1]$$

Anche qui dimensionalmente torna tutto.

6.2.2 Esercizio 2

Un Internet Service Provider (ISP) dispone di un sistema per l'accesso remoto degli utenti basato su una batteria di N Remote Access Server (RAS), ciascuno dei quali è dotato di m porte. Il sistema è in grado di riconoscere le richieste di accesso associate ad una classe privilegiata di utenti. Si supponga che tali richieste, ad alta priorità, e che le rimanenti richieste, a bassa priorità, arrivino secondo processi indipendenti di Poisson a velocità, rispettivamente, $\{\lambda_1,\ \lambda_2\}$. Si assuma, inoltre, che i tempi di collegamento siano v.c. indipendenti e distribuite esponenzialmente con valor medio $\frac{1}{\mu}$. Quando una richiesta ad alta priorità arriva ed una porta qualsiasi sui vari apparati RAS è disponibile ("idle"), la richiesta è accettata e la porta è assegnata ad essa, altrimenti la richiesta è scartata. Una richiesta a bassa priorità è accettata solo se $(g+1)$ o più porte sono disponibili, altrimenti la richiesta è scartata. I RAS sono soggetti a malfunzionamenti. Il "time to failure" ed il "time to repair" per un RAS siano v.c. indipendenti e distribuite esponenzialmente con parametro, rispettivamente, pari a: $\{f,\ \beta\}$. Una singola "repair facility" sia condivisa da tutti i RAS. Introducendo eventuali ipotesi che si ritengono necessarie, il candidato valuti la probabilità P_H che una richiesta ad alta priorità sia scartata. Suggerimento. Se si trascurano gli effetti che hanno sulla quantità a regime P_H gli eventi corrispondenti al guasto di un RAS quando ci sono collegamenti in corso sulle sue porte, il problema può essere affrontato con un approccio gerarchico. Il candidato immagini, adesso, di essere collegato ad Internet tramite l'ISP di cui sopra. Valuti la probabilità P_I che il proprio collegamento sia interrotto improvvisamente a seguito del guasto del RAS.

Risoluzione:

(ISP) Internet Service Provider, con batteria di $N := |RAS|$ (Remote Access Server). Ogni RAS è dotato di $m := |porte|$. Si ha il seguente mapping:

$$\begin{cases} \lambda_1 - POISSON \\ \lambda_2 - POISSON \end{cases} \mapsto \begin{cases} HPRIO\ ARRIVALS \\ LPRIO\ ARRIVALS \end{cases}$$

I tempi di collegamento (servizio) siano v.c. indipendenti e distribuite esponenzialmente con valor medio $(\frac{1}{\mu})$. Se arriva una HPRIO-req, allora se ci sono porte disponibili è accettata, altrimenti la richiesta viene scartata. Se invece arriva una LPRIO-req, sono necessarie $(g+1)$ porte per accettare la richiesta, altrimenti anch'essa viene scartata. Abbiamo:

$$\begin{cases} ttf \sim EXP(f) \\ ttr \sim EXP(\beta) \end{cases}$$

Si noti che abbiamo una SRF (Single Repair Facility). Definiamo: $P_H = \Pr\{LOSS\}_{HPRIO}$. Si immagini che gli eventi relativi alla dependability (disponibilità ed affidabilità) siano di gran lunga più rari che quelli relativi alle performance (richieste di collegamento), ed accettando inoltre di trascurare gli effetti che hanno sulle quantità a regime P_H gli eventi comprendenti il guasto di un RAS quando vi sono collegamenti in corso sulle sue m porte, allora il sistema

può essere analizzato mediante APPROCCIO GERARCHICO. Immaginiamo che i tempi di collegamento siano indipendenti dai tempi di arrivo delle richieste, e che i tempi di guasto siano indipendenti dai tempi di riparazione (ipotesi più che plausibile, ma non scontata). Approccio gerarchico. Sia suddiviso il modello in:

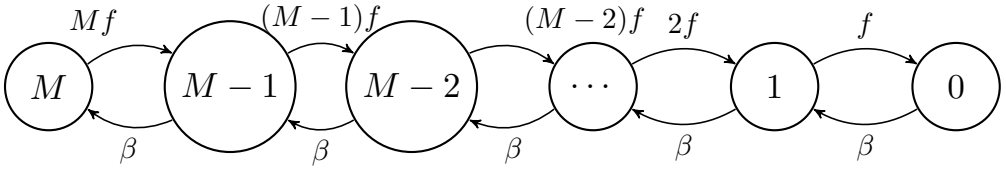
- -) Modello di disponibilità (*Availability Model*);
- -) Performance Model

Cominciamo con l'analizzare, per l'appunto gerarchicamente, il sistema:

- *Availability Model*:

  Diamo come definzione di stato del modello di disponibilità la seguente:

  $N(t) := |RAS\ funzionanti\ in\ t|$. Tale è il DTT del modello di disponibilità (a livello più alto):



  Calcoliamo la velocità totale di uscita, supponendo che $N(t) = M - i$. Definiamo le v.a. $\{\xi_{Rt}, \eta_{Rt}\}$ rispettivamente tempo di riparazione residuo e tempo di guasto residuo. Ricordiamo che:

$$\begin{cases} \eta_{jRt} \sim EXP(f) \\ \eta_{Rt} \sim EXP((M-i)f) \end{cases}$$

  Infatti, di RAS funzionanti ve ne sono $M - i$ con guasti indipendenti tra di loro, quindi $\eta_{Rt} \sim EXP(\sum_k^{M-i} f_k = f = (M-i)F)$. Calcoliamo $(\phi_{i':=M-i}(t) = \min(\eta_{Rt}, \xi_{Rt})) \sim EXP(-q_{i'i'})$:

$$F^c_{\phi_{M-i}}(\tau) = \Pr\{\phi_{M-i}(t) > \tau\} = \Pr\{\min(\eta_{Rt}, \xi_{Rt}) > \tau\} = \Pr\{\xi_{Rt} > \tau,\ \eta_{Rt} > \tau\} =$$

$$= \Pr\{\xi_{Rt} > \tau\} \Pr\{\eta_{Rt} > \tau\} = \mathrm{e}^{-\beta\tau}\, \mathrm{e}^{-[(M-i)f\tau]} =$$

$$= \mathrm{e}^{-[\beta+(M-i)f]\tau} \implies q_{i'i'} = -[\beta + (M-i)f]$$

  Calcoliamo ora i tassi di transizione:

$$\tau_{i',i'+1} = \frac{q_{i',i'+1}}{-q_{i'i'}} = \Pr\{\xi_{Rt} < \eta_{Rt}\} = \int_0^\infty \Pr\{\xi_{Rt} < \eta_{Rt} \mid \eta_{Rt} = y\} \Pr\{\eta_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-\beta y})(M-i)f\, \mathrm{e}^{-(M-i)fy} dy =$$

$$= \int_0^\infty (M-i)f\, \mathrm{e}^{-(M-i)fy} dy - (M-i)f \int_0^\infty \mathrm{e}^{-[\beta+(M-i)f]y} dy =$$

$$= 1 - \frac{(M-i)f}{\beta + (M-i)f} = \frac{\beta + (M-i)f - (M-i)f}{\beta + (M-i)f} \implies q_{i',i'+1} = \beta$$

Analogamente..

$$\tau_{i',i'-1} = \frac{q_{i',i'-1}}{-q_{i'i'}} = \Pr\{\eta_{Rt} < \xi_{Rt}\} = \int_0^\infty \Pr\{\eta_{Rt} < \xi_{Rt} \mid \xi_{Rt} = y\} \Pr\{\xi_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-(M-i)fy}) \beta \, \mathrm{e}^{-\beta y} dy = \int_0^\infty \beta \, \mathrm{e}^{-\beta y} dy - \beta \int_0^\infty \mathrm{e}^{-[(M-i)f+\beta]y} dy =$$

$$= 1 - \frac{\beta}{(M-i)f + \beta} = \frac{(M-i)f + \beta - \beta}{(M-i)f + \beta} = \frac{(M-i)f}{(M-i)f + \beta} \implies q_{i',i'-1} = (M-i)f$$

Per quanto concerne le probabilità a regime abbiamo:

$$\begin{cases} \pi_i = \pi_0 (\frac{\beta}{f})^i \frac{1}{i!} \\ [\pi_0 = \frac{1}{\sum_{i=0}^{M} (\frac{\beta}{f})^i \frac{1}{i!}} \end{cases}$$

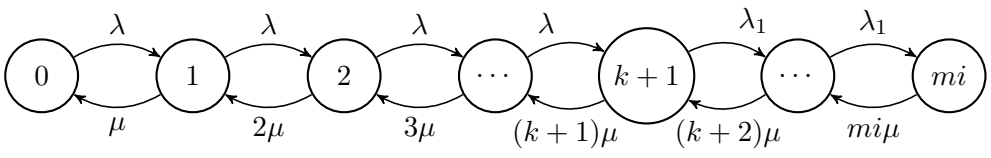
con $0 \leq i \leq M$. La formula è già stata vista in realtà nella M/M/m/0, ovvero la B di ERLANG.

- *Performance Model*:

  Una porta per una richiesta a bassa priorità si occupa solo se $(g+1)$ porte sono ancora disponibili. Potremmo considerare come definizione di stato: $Y(t) := |\textit{porte occupate in } t|$, od analogamente: $Y(t) := |\textit{richieste in esecuzione in } t|$. Una richiesta verrebbe scartata ovviamente (sia LPRIO che HPRIO) qualora tutte le porte fossero busy, che in tutto sono proprio mi (considerando che vi siano i server funzionanti al tempo t). È ragionevole ipotizzare che i tassi di transizione (di morte) in questa CMTC, se vi sono j richieste in esecuzione, siano pari proprio a $(j\mu)$, per gli argomenti trattati sempre (minimo di v.a. esponenziali indipendenti). Per scrivere il DTT, ragioniamo sul seguente tableau indicante le richieste rimanenti:

$$\begin{cases} mi \\ mi - 1 \\ mi - 2 \\ \vdots \\ \begin{bmatrix} mi - k = g + 1 \\ k := mi - g - 1 \end{bmatrix} \end{cases}$$

Il DTT è allora il seguente:



Solita storia per i tassi di transizione. Se consideriamo uno stato j prima di $(mi-g-1)$, allora la velocità di arrivo delle richieste, effettuando un POOLING alla POISSON è: $\lambda := \lambda_1 + \lambda_2$. Al solito:

$$\begin{cases} \xi_{Rt} \sim EXP(\lambda) \\ \eta_{Rt} \sim EXP(j\mu) \end{cases}$$

$$(\phi_j(t) = \min(\xi_{Rt}, \eta_{Rt})) \sim EXP(-q_{jj}) \implies$$

$$\implies F_{\phi_j}(\tau) = \Pr\{\phi_j(t) > \tau\} = \Pr\{\min(\xi_{Rt}, \eta_{Rt}) > \tau\} = \Pr\{\xi_{Rt} > \tau,\ \eta_{Rt} > \tau\} =$$

$$= \Pr\{\xi_{Rt} > \tau\} \Pr\{\eta_{Rt} > \tau\} = \mathrm{e}^{-\lambda\tau}\, \mathrm{e}^{-j\mu\tau} = \mathrm{e}^{-(\lambda+j\mu)\tau} \implies -q_{jj} = (\lambda + j\mu)$$

Calcoliamo ora i tassi di transizione:

$$\tau_{j,j+1} = \frac{q_{j,j+1}}{-q_{jj}} = \Pr\{\xi_{Rt} < \eta_{Rt}\} = \int_0^\infty \Pr\{\xi_{Rt} < \eta_{Rt} \mid \eta_{Rt} = y\} \Pr\{\eta_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-\lambda y}) j\mu\, \mathrm{e}^{-j\mu y} dy = \int_0^\infty j\mu\, \mathrm{e}^{-j\mu y} dy - j\mu \int_0^\infty \mathrm{e}^{-(\lambda+j\mu)} dy =$$

$$= 1 - \frac{j\mu}{\lambda + j\mu} \int_0^\infty (\lambda + j\mu)\, \mathrm{e}^{-(\lambda+j\mu)y} dy = 1 - \frac{j\mu}{\lambda + j\mu} =$$

$$= \frac{\lambda + j\mu - j\mu}{\lambda + j\mu} = \frac{\lambda}{\lambda + j\mu} \implies q_{j,j+1} = \lambda$$

Analogamente:

$$\tau_{j,j-1} = \frac{q_{j,j-1}}{-q_{jj}} = \Pr\{\eta_{Rt} < \xi_{Rt}\} = \int_0^\infty \Pr\{\eta_{Rt} < \xi_{Rt} \mid \xi_{Rt} = y\} \Pr\{\xi_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-j\mu y})\, \mathrm{e}^{-\lambda y} dy = \int_0^\infty \lambda\, \mathrm{e}^{-\lambda y} dy - \lambda \int_0^\infty \mathrm{e}^{-(\lambda+j\mu)y} dy =$$

$$= 1 - \frac{\lambda}{(\lambda + j\mu)} \int_0^\infty (\lambda + j\mu)\, \mathrm{e}^{-(\lambda+j\mu)y} dy = \frac{\lambda + j\mu - \lambda}{\lambda + j\mu} = \frac{j\mu}{\lambda + j\mu} \implies q_{j,j-1} = j\mu$$

Facciamo una sintesi:

$$\tau_{j,j+1} = \begin{cases} \dfrac{\lambda}{\lambda + j\mu},\ j < mi - g \\ \dfrac{\lambda_1}{\lambda_1 + j\mu},\ j \geq mi - g \end{cases}$$

$$\begin{cases} P_j = \begin{cases} P_0 (\dfrac{\lambda}{\mu})^j \dfrac{1}{j!},\ j < mi - g \\ P_0 (\dfrac{\lambda_1}{\mu})^j \dfrac{\lambda^{mi-g-1}}{\lambda_1^{mi-g-1} j!},\ j \geq mi - g \end{cases} \\ P_0 = \dfrac{1}{1 + \sum_{j=1}^{mi-g-1} (\frac{\lambda}{\mu})^j \frac{1}{j!} + \frac{\lambda^{mi-g-1}}{\lambda_1^{mi-g-1}} \sum_{j=mi-g}^{mi} (\frac{\lambda_1}{\mu})^j \frac{1}{j!}} \end{cases}$$

Con la seguente procedura faremo un mapping tra i due modelli:

$$\underline{P_L(i)} := P_{mi}^{(a)} \overset{PASTA}{=} P_{mi} =$$

$$= \frac{(\frac{\lambda_1}{\mu})^{mi} \frac{\lambda^{mi-g-1}}{\lambda_1^{mi-g-1}(mi)!}}{\sum_{j=0}^{mi-g-1} (\frac{\lambda}{\mu})^j \frac{1}{j!} + \frac{\lambda^{mi-g-1}}{\lambda_1^{mi-g-1}} \sum_{j=mi-g}^{mi} (\frac{\lambda_1}{\mu})^j \frac{1}{j!}}$$

ove abbiamo opportunamente inglobato il termine 1 nella prima sommatoria. Effettuiamo il seguente assegnamento:

$$\begin{cases} r_i := P_L(i),\ i = 1, 2,\ \ldots, M \\ \left[\mathbb{E}[z] = \sum_{i=0}^{M} r_i \pi_i = \sum_{i=0}^{M} P_L(i)\pi_i = \right. \\ \left. = \sum_{i=1}^{M} P_L(i)\pi_i + (r_0 = 1) \right. \end{cases}$$

Calcoliamo ora la probabilità P_I che il collegamento di un utente su un RAS sia interrotto improvvisamento a seguito del guasto di quest'ultimo. Ridefiniamo opportunamente i tempi residui, indicando rispettivamente con $\{\eta_{Rt}, \xi_{Rt}\}$ il tempo di failure residuo ed il tempo di servizio residuo:

$$\begin{cases} \eta_{Rt} \sim EXP(f) \\ \xi_{Rt} \sim EXP(\mu) \end{cases}$$

$$\Pr\{\eta_{Rt} < \xi_{Rt}\} = \int_0^\infty \Pr\{\eta_{Rt} < \xi_{Rt} \mid \xi_{Rt} = y\} \Pr\{\xi_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-fy}) \mu \, \mathrm{e}^{-\mu y} dy = \int_0^\infty \mu \, \mathrm{e}^{-\mu y} - \mu \int_0^\infty \mathrm{e}^{-(f+\mu)y} dy =$$

$$= 1 - \frac{\mu}{(f+\mu)} = \frac{f+\mu-\mu}{(f+\mu)} = [\frac{f}{f+\mu}]$$

6.2.3 Esercizio 3

Un servizio di rete è erogato tramite un sistema ridondante parallelo costituito da due dispositivi, con un dispositivo active, l'altro in standby. Il failure rate dell'unità attiva è differente da quello dell'unità in standby ed anche gli effetti di un guasto dell'unità attiva sono differenti dagli effetti di un guasto dell'unità in standby. Si supponga che:

- il time to failure dell'unità attiva sia una v.c. $(\ldots) \sim EXP(\lambda)$;
- il time to failure dell'unità in standby sia una v.c. $(\ldots) \sim EXP(\lambda_S)$;
- il time to restoration di una unità failed sia una v.c. $(\ldots) \sim EXP(\mu)$

Quando l'unità attiva si guasta, con probabilità c (coverage factor) lo switch di protezione ripristina il servizio con successo, commutando sull'unità in standby, la quale diventa l'unità attiva. Con probabilità $(1-c)$, invece, la procedura di ripristino non ha successo ed è necessario un reboot dello switch la cui durata è una v.c. $(\ldots) \sim EXP(\beta)$. Un guasto dell'unità in standby mentre l'unità attiva è ancora operativa è rilevato immediatamente con probabilità c_S.

Con probabilità $(1-c_S)$, invece, il guasto non è rilevato. In tal caso, vi sarebbe un guasto latente dell'unità di riserva. Al fine di rivelare il guasto latente dell'unità in standby, una routine di diagnostica è eseguita ad intervalli di tempo che sono v.c. $(\dots) \sim EXP(\alpha)$.

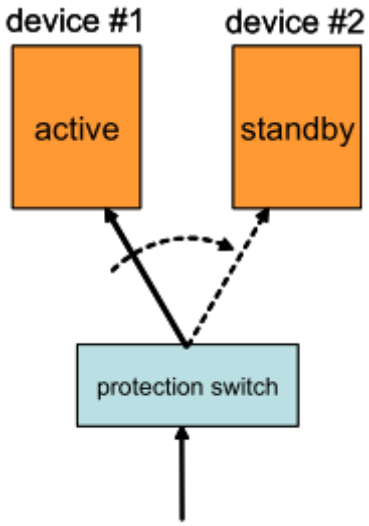
- a) Definire una CMTC che modelli il sistema e disegnarne il diagramma dei tassi di transizione;

- b) Valutare la frequenza delle riparazioni (supponendo di aver calcolato la distribuzione di regime);

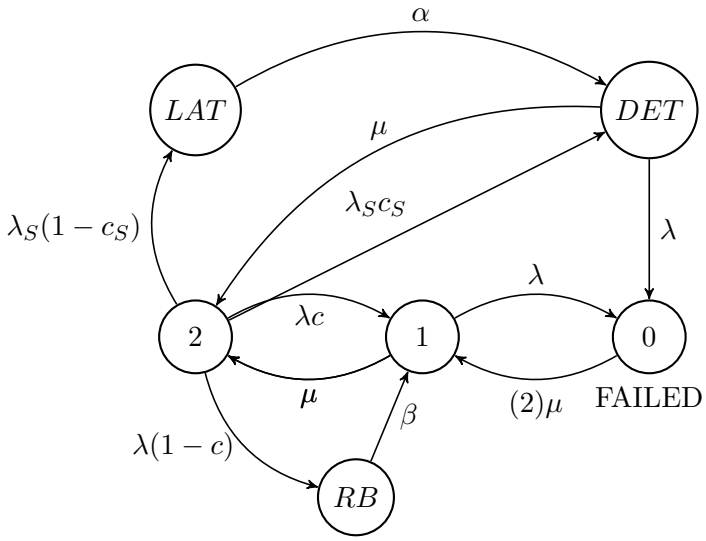
- c) Il servizio non è erogato (sistema DOWN) durante la fase di reboot e quando entrambe le unità sono guaste:

  - Determinare la steady-state unavailability del servizio (supponendo di aver calcolato la distribuzione di regime);

  - Impostare (non risolvere) il sistema di equazioni $\bar{L}_N(\infty)\bar{Q}_N = -\bar{\pi}_N(0)$ (stato iniziale: entrambi i dispositivi non sono guasti) che consente di valutare il Mean Time To Failure del servizio.



Il DTT è il seguente:



Previo mapping delle seguenti variabili:

$$\begin{cases} \xi_{1Rt} \sim EXP(\lambda) \\ \xi_{2Rt} \sim EXP(\lambda_S) \end{cases}$$

$\rightarrow$ {tempo di guasto residuo al tempo t (ACTIVE), tempo di guasto residuo al tempo t (STANDBY)}, calcoliamo la velocità totale di uscita dallo STARTING STATE: {2}, sapendo che ovviamente $(\phi_2(t) = \min(\xi_{1Rt},\ \xi_{2Rt})) \sim EXP(-q_{22})$:

$$F_{\phi_2}(\tau) = \Pr\{\min(\xi_{1Rt},\ \xi_{2Rt}) > \tau\} = \Pr\{\xi_{1Rt} > \tau,\ \xi_{2Rt} > \tau\} =$$

$$= \Pr\{\xi_{1Rt} > \tau\} \Pr\{\xi_{2Rt} > \tau\} = \mathrm{e}^{-\lambda\tau}\, \mathrm{e}^{-\lambda_S\tau} = \mathrm{e}^{-(\lambda+\lambda_S)\tau}$$

Ciò ovviamente implica che $\implies -q_{22} = \lambda + \lambda_S$.
Sappiamo empiricamente che:

$$\begin{cases} q_{21} = c\lambda \\ q_{2RB} = (1-c)\lambda \\ q_{2DET} = c_S\lambda_S \\ q_{2LAT} = (1-c_S)\lambda_S \end{cases}$$

Valutiamo le frequenze delle riparazioni (supponendo ovviamente di aver calcolato la distribuzione di regime). Anzitutto, la CMTC è OMOGENEA (tassi di transizione costanti), irriducibile (tutti gli stati comunicano tra di loro) con un numero di stati finito ($\iff |stati| < +\infty$) $\implies$ ERGODICA $\iff \exists \pi_i = \lim_{t\to+\infty} \pi_i(t)$. Quindi per sapere la frequenza delle riparazioni è sufficiente sommare le frequenze delle transizioni di stato che caratterizzano una riparazione:

$$f_R := \pi_1\mu + \pi_0\mu + \pi_{DET}\mu = \mu(\pi_1 + \pi_0 + \pi_{DET})$$

Servizio NON EROGATO: (sistema DOWN) durante la fase di reboot e quando entrambe le unità sono guaste. Sempre supponendo di disporre della distribuzione di regime, abiamo che la *STEADY-STATE UNAVAILABILITY* è:

$$SSU := \bar{A} = \pi_0 + \pi_{RB}$$

Si noti che il tasso $0 \to 1$ dipende dal fatto se abbiamo una SRF (Single Repair Facility) o meno (minimo di v.a. esponenziali indipendenti $\to$ ancora esponenziale con parametro somma dei ttr dei due dispositivi). Comunque non ci interessa, dal momento che per calcolare l'MTTF del sistema dobbiamo switchare sul modello di affidabilità, che non prevede assolutamente riparazioni. Siamo quindi obbligati a fare il detach del ramo di transizione sopracitato. Per quanto concerne $RB \to 1$, probabilmente non è considerato poi un guasto catastrofico, un failure molto dannoso dal punto di vista dell'affidabilità; quindi un'attenta analisi dei requisiti potrebbe risparmiarne la cancellazione nel RCM (*Reliability Computational Model*). Impostiamo il sistema: $\bar{L}_N(\infty)\bar{Q}_N = -\bar{\pi}_N(0)$, con stato iniziale: $\{2\} \iff \pi_2(0) = 1$. Scriviamo la matrice generatore della catena, ovvero la matrice dei tassi di transizione:

$$\bar{Q} = \begin{bmatrix} -(\lambda+\lambda_S) & \lambda c & 0 & \lambda(1-c) & \lambda_S(1-c_S) & \lambda_S c_S \\ \mu & -(\mu+\lambda) & \lambda & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \beta & 0 & -\beta & 0 & 0 \\ 0 & 0 & 0 & 0 & -\alpha & \alpha \\ \mu & 0 & \lambda & 0 & 0 & -\mu \end{bmatrix}$$

Si noti che nella CMTC non abbiamo considerato un eventuale ramo di transizione $LAT \to 0$. Infatti, se la routine di diagnostica fosse eseguita sul primo server, un eventuale guasto di quest'ultimo abbatterebbe il relativo processo di diagnostica in corso, rendendo di fatto impossibile la rilevazione del malfunzionamento del server standby che avesse avuto un malfunzionamento. Ma consideriamo $\alpha \ll \lambda$, in maniera tale che sia molto meno probabile che accada. Comunque è solamente questione di non appesantire il modello. Naturalmente in un'analisi più dettagliata tali questioni andrebbero tenute in conto, magari con l'aggiunta di un watch-dog a parte con il quale effettuare diagnostica e controllo.

Adoperiamo la restrizione della matrice sul set degli stati transitori, ricordando l'ovvia relazione di unione: $S = N \cup A$:

$$\begin{cases} \bar{Q}_N = \begin{bmatrix} -(\lambda+\lambda_S) & \lambda c & \lambda(1-c) & \lambda_S(1-c_S) & \lambda_S c_S \\ \mu & -(\mu+\lambda) & 0 & 0 & 0 \\ 0 & \beta & -\beta & 0 & 0 \\ 0 & 0 & 0 & -\alpha & \alpha \\ \mu & 0 & 0 & 0 & -\mu \end{bmatrix} \\ \bar{L}_N = \begin{bmatrix} L_2 & L_1 & L_{RB} & L_{LAT} & L_{DET} \end{bmatrix} \\ \bar{\pi}_N = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{cases}$$

$\bar{L}_N(\infty)\bar{Q}_N = -\bar{\pi}_N(0) \implies$

$$\begin{cases} -L_2(\lambda+\lambda_S) + L_1\mu + L_{DET}\mu = -1 \\ L_2\lambda c - L_1(\mu+\lambda) + L_{RB}\beta = 0 \\ L_2\lambda(1-c) - L_{RB}\beta = 0 \\ L_2\lambda_S(1-c_S) - L_{LAT}\alpha = 0 \\ L_2\lambda_S c_S + L_{LAT}\alpha - L_{DET}\mu = 0 \end{cases}$$

6.3 MTTF of a fiber link

Transmission links can be deployed with protection channels, wherein if the primary link is disrupted, the system switches to a protection channel. Fiber cable is typically the transmission

medium of choice for these link systems. Figures 1, 2 and 3 show the structure of link systems. Link systems can be generally classified as one of three cases:

- ***Case A****: Unprotected link* is the single-fiber system with no backup (shown in Fig. 1);
- ***Case B****: Partially protected link* has redundant fiber media backup. Redundant circuits are supposed to take different physical paths (shown in Fig. 2);
- ***Case C****: Fully protected link* has fiber media, transceiver, and power backup, with the transceivers being in hot standby (shown in Fig. 3).

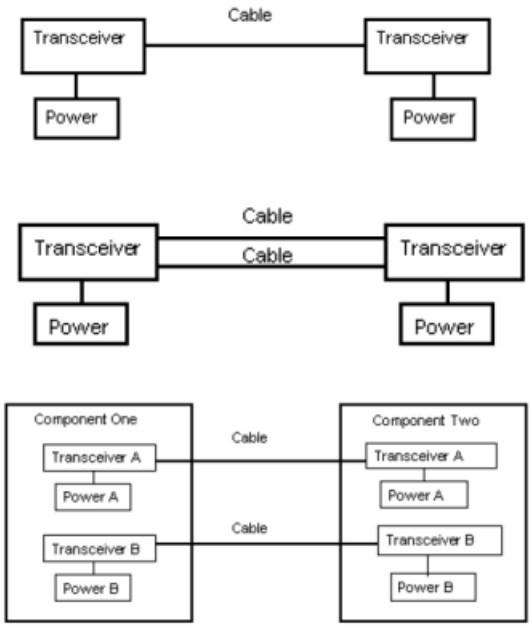


Figura 6.1: Unprotected link, Partially protected link, Fully protected link

For long optical fiber links, cable cuts are the dominant failure scenario (table 1 shows the optical fiber MTTF at different distances). Instead, for short fiber links (less than about 5 miles), the optical fiber MTTF is quite high, so that the transceiver and power systems become the dominant contribution to failure, rather than the fibers. Assuming that the Transceiver MTTF is equal to 8.0 $years$, that the Power MTTF is equal to 8.0 $years$ and that the Times To Failure of a fiber cable, a power supply and a transceiver are independent, exponentially distributed random variables, evaluate:

- a) the MTTF of a $10 - mile$ link without protection;
- b) the MTTF of a $10 - mile$, partially protected link;
- c) the MTTF of a $10 - mile$ link, fully protected link.

Table 1

Link Length (Miles)	Optical Fiber MTTF (Years)
1	222.79
5	45.56
10	22.78
20	11.39

Risoluzione:

$$\begin{cases} ttf_{FIBER\_CAB} \sim EXP(h_F = \dfrac{1}{MTTF_F = 22.78y}) \\ ttf_{TRANS} \sim EXP(h_T = \dfrac{1}{MTTF_T = 8.0y}) \\ ttf_{POWER} \sim EXP(h_P = \dfrac{1}{MTTF_P = 8.0y}) \end{cases}$$

ove h_i sono failure rate. In particolare, la distribuzione del lifetime $(ttf)_i$ è esponenziale $\implies$ failure rate costante e pari a $(\lambda_i := h_i(t))$. Il valor medio è $\frac{1}{h_i(t)} = (\frac{1}{\lambda_i})$.

- **Unprotected link**:

  In tal primo caso abbiamo la serie di tre macrocomponenti in serie, costituiti rispettivamente da:

  - serie tra power e transceiver a monte;
  - link di trasmissione;
  - serie tra transceiver e power a valle;

  Nelle configurazioni serie i failure rate si sommano per dar luogo al failure rate complessivo del sistema:

$$h_L = 2h_P + 2h_t + h_F = (\frac{1}{22.78}y^{-1}) + 2(\frac{1}{8}y^{-1}) + 2(\frac{1}{8}y^{-1}) =$$
$$= (\frac{1}{22.78}y^{-1} + \frac{1}{2}y^{-1}) = 0.5439y^{-}1 \implies \underline{MTTF_L} = \frac{1}{h_L} = \frac{1}{0.5439}y = 1.8386y$$

- **Partially protected link**:

  In tal caso la configurazione rimane la medesima del caso precedente, ma nell'intermezzo come sistema di trasmissione abbiamo il parallelo tra due link in fibra ottica. Calcoliamo prima l'affidabilità:

$$R(t) = R_P(t)^2 R_T(t)^2[1 - (1 - R_F(t))^2] = \mathrm{e}^{-2h_P t}\,\mathrm{e}^{-2h_T t}[-\,\mathrm{e}^{-2h_F t} + 2\,\mathrm{e}^{-h_F t}] =$$
$$= -\,\mathrm{e}^{-[2h_P+2h_T+2h_F]t} + 2\,\mathrm{e}^{-[2h_P+2h_T+h_F]t}$$

  Quindi integrando opportunamente l'affidabilità, troviamo l'MTTF:

$$MTTF = \mathbb{E}[X] = \int_0^\infty R(t)dt = 2\int_0^\infty \mathrm{e}^{-[2h_P+2h_T+h_F]t}dt - \int_0^\infty \mathrm{e}^{-[2h_P+2h_T+2h_F]t}dt =$$

$$= \frac{2}{2h_P+2h_T+h_F} - \frac{1}{2h_P+2h_T+2h_F} = \frac{2}{2(\frac{1}{8})+2(\frac{1}{8})+\frac{1}{22.78}} - \frac{1}{\frac{1}{2}+\frac{2}{22.78}} = 1.976y$$

Ragionevolmente maggiore!

- **Fully protected link**:

  Dobbiamo fare il parallelo tra i due interi sistemi, ove ogni singolo sistema è quello del caso 1, ovvero l'unprotected link, che ha come affidabilità banalmente:

  $$R_{NP} = R_P^2 R_T^2 R_F = \mathrm{e}^{-[2h_P+2hT+h_F]t}$$

  Dunque, adoperando il parallelo su R_{NP}, doppiandolo opportunamente, abbiamo:

  $$1-(1-R_{NP})^2 = 1-(1+\mathrm{e}^{-2[2h_P+2h_T+h_F]t} - 2\,\mathrm{e}^{-[2h_P+2h_T+h_F]t}) =$$

  $$= 2\,\mathrm{e}^{-[2h_P+2h_T+h_F]t} - \mathrm{e}^{-2[2h_P+2h_T+h_F]t}$$

  Integrando opportunamente per trovare l'MTTF troviamo:

  $$MTTF = \mathbb{E}[X] = \int_0^\infty R(t)dt = 2\int_0^\infty \mathrm{e}^{-[2h_P+2h_T+h_F]t}dt - \int_0^\infty \mathrm{e}^{-2[2h_P+2h_T+h_F]t} =$$

  $$= \frac{2}{2h_P+2h_T+h_F} - \frac{1}{2[2h_P+2h_T+h_F]} =$$

  $$= \frac{3}{2[2h_P+2h_T+h_F]} = \frac{3}{2(0.5439y^{-1})} = 2.75787y$$

  Decisamente migliore!

6.4 Response Time Distribution 2

Un servizio di rete è basato su due centri di servizio. In figura 1 è riportata la rete di Jackson che descrive lo schema secondo il quale i job sono eseguiti. La $P\{accodamento\}$ sia pari, rispettivamente, a P_A per il sistema a coda "A" e a P_B per il sistema a coda "B". Si derivi la CMTC che consente di determinare un'approssimazione della distribuzione del tempo di esecuzione di un job.

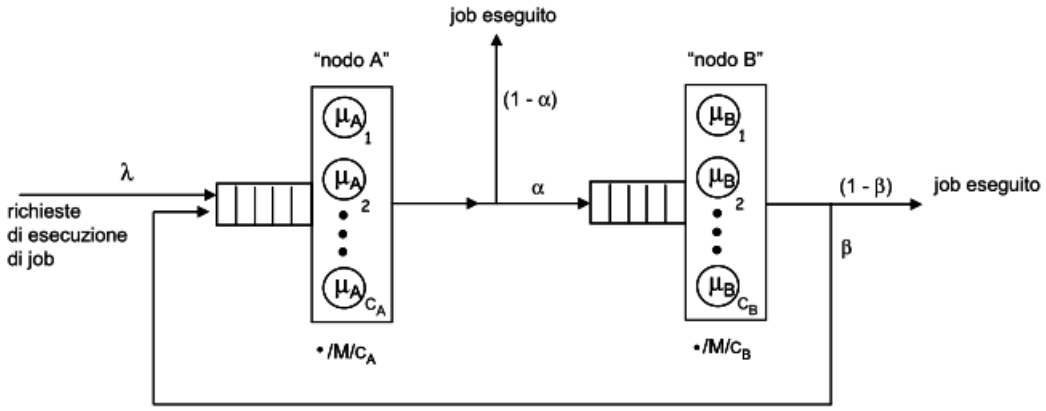


Figura 1

Sia: $P_A := \Pr\{queueing\}_A$, $P_B := \Pr\{queueing\}_B$. Dallo studio fatto sappiamo che sono calcolabili mediante C di ERLANG per delle M/M/m. Valgano le ipotesi standard BCMP, dal momento che ci troviamo dinanzi delle code $\cdot/\mathrm{M}/c_{i\in A,B}$, ovvero con arrivi non markoviani (poissoniani). Per A abbiamo:

$$T_A = \frac{1}{\mu_A} + P_A \frac{1}{C_A \mu_A (1-\rho_A)} = \frac{C_A(1-\rho_A) + P_A}{\mu_A C_A (1-\rho_A)}$$

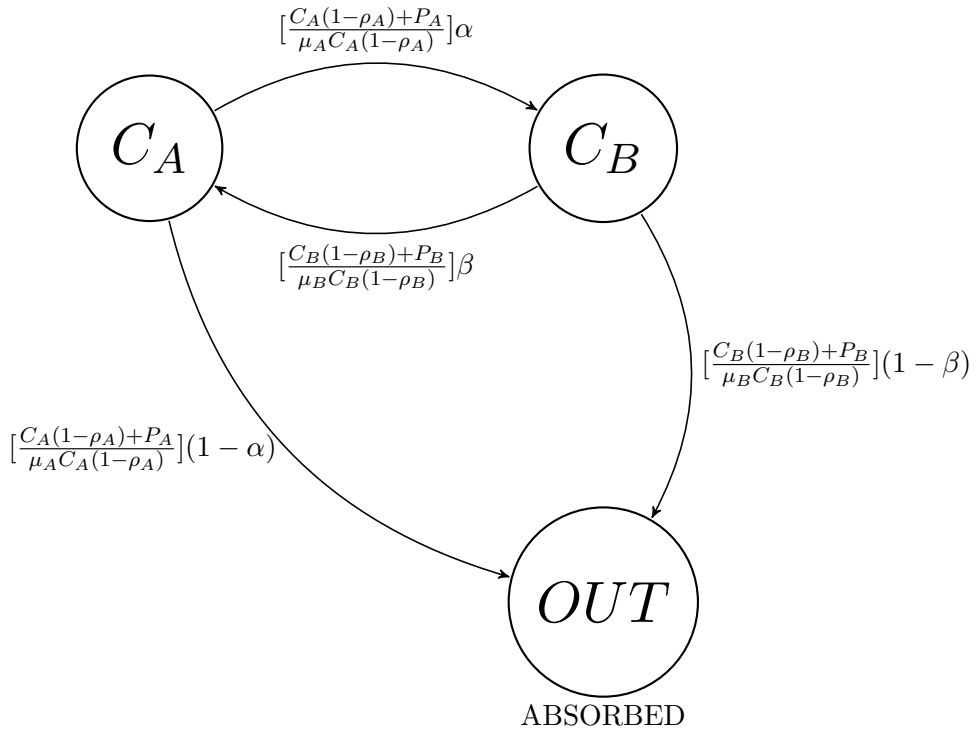
Di conseguenza il tasso di ritardo per questa CMTC, con STATO ASSORBENTE {OUT} è:

$$q_{AB} = \frac{1}{T_A} = [\frac{C_A(1-\rho_A) + P_A}{\mu_A C_A (1-\rho_A)}]\alpha$$

Analogamente per l'altro sistema a coda. Quindi i tassi alla fine saranno i seguenti:

$$\begin{cases} q_{AB} = \frac{1}{T_A}\alpha = [\frac{C_A(1-\rho_A) + P_A}{\mu_A C_A (1-\rho_A)}]\alpha \\ q_{AOUT} = \frac{1}{T_A}(1-\alpha) = [\frac{C_A(1-\rho_A) + P_A}{\mu_A C_A (1-\rho_A)}](1-\alpha) \\ q_{BA} = \frac{1}{T_B}\beta = [\frac{C_B(1-\rho_B) + P_B}{\mu_B C_B (1-\rho_B)}]\beta \\ q_{BOUT} = \frac{1}{T_B}(1-\beta) = [\frac{C_B(1-\rho_B) + P_B}{\mu_B C_B (1-\rho_B)}](1-\beta) \end{cases}$$

Il DTT della CMTC è il seguente:



6.5 Server Load Balancer

Un Server Load Balancer (SLB) (vedi figura) è utilizzato da un Content Service Provider per bilanciare il carico sui k server di una server farm. L'algoritmo adottato dal SLB per distribuire le richieste di servizio dei client è il seguente: una nuova richiesta viene inoltrata al server che sta rispondendo al minor numero di richieste e, nel caso in cui più server stiano rispondendo allo stesso ed, al contempo, più basso numero di richieste, la richiesta è inoltrata ad uno qualsiasi tra questi server. Si supponga:

- a) che le richieste di servizio arrivino al SLB secondo un processo indipendente di Poisson a velocità λ $[req/s]$;
- b) che sia trascurabile l'intervallo temporale tra l'istante in cui una richiesta arriva al SLB e l'istante in cui il server selezionato riceve tale richiesta;
- c) che i server rispondano ad una richiesta non appena la ricevono e con tempi di servizio che sono variabili casuali indipendenti e distribuite esponenzialmente a valor medio $\frac{1}{\mu}$ secondi;
- d) che un generico server non possa rispondere contemporaneamente a più di M richieste e che sia persa qualsiasi richiesta in arrivo quando la server farm è satura (ciascun server sta rispondendo a M richieste).

Nel caso in cui $k = M = 2$,

- 1) Disegnare il diagramma dei tassi di transizione di una CMTC che modella il sistema e discutere l'ergodicità della catena;
- 2) Calcolare:

– la probabilità che una richiesta di servizio non sia persa;

– il numero medio di richieste a cui un server risponde contemporaneamente;

- 3) La sequenza delle richieste ricevute da un server possono essere modellate con un processo di Poisson? Motivare la risposta.

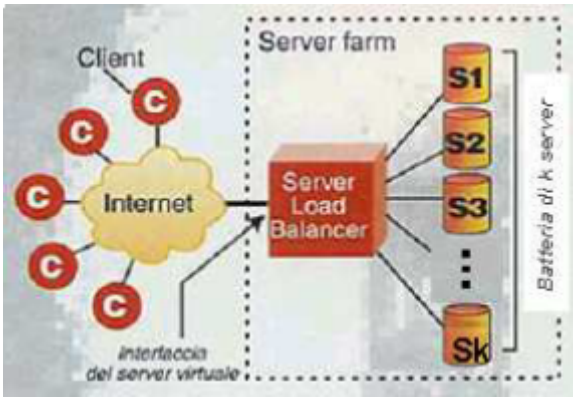
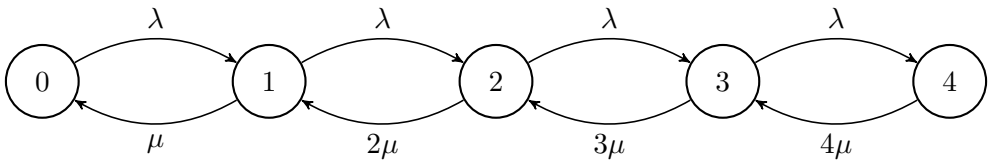


Figura 6.2: Service Load Balancer

Risoluzione:

(SLB) as *Service Load Balancer*, (ISP) as *Content Service Provider*. k server di una server farm. Consideriamo la seguente definizione di stato: $N(t) := |richieste\ al\ tempo\ t|$. Abbiamo la seguente CMTC, con $[k = M = 2]$, rappresentata mediante il seguente DTT:



La quale è OMOGENEA, IRRIDUCIBILE e con numero di stati finito ($\iff |stati| < +\infty$), dunque è ERGODICA. Probabilità che una richiesta di servizio in arrivo non sia persa:

$$P_{NOT4} := 1 - \pi_4^{(a)} \overset{PASTA}{=} 1 - \pi_4 =$$

$$= 1 - [\Pr\{N(t) = 4\} = \frac{(\frac{\lambda}{\mu})^4 \frac{1}{4!}}{\sum_{i=0}^{4} (\frac{\lambda}{\mu})^4 \frac{1}{4!}}]$$

Calcoliamo ora il numero medio di richieste a cui un server risponde contemporaneamente, ovvero il numero medio di clienti in un server sostanzialmente. Dal momento che i server sono praticamente identici, equifunzionali ed equiprobabili, non c'è nessuna ragione per la quale un arrivo, a parità di richieste sui singoli server, debba preferibilmente esser preso in considerazione dall'uno o dall'altro. In taluna circostanza infatti, l'algoritmo adopera una scelta casuale. Quindi la velocità di arrivo ad ognuno di essi è: $(\frac{\lambda_S}{2})$, sebbene queste velocità NON costituiscano un processo di POISSON, dal momento che vi sono perdite in gioco. Dobbiamo sostanzialmente sommare tutte le frequenze di transizione che riguardano l'ingresso di una richiesta ad uno qualsiasi dei due server:

$$\begin{cases} \lambda_S = \pi_1\mu + \pi_2(2\mu) + \pi_3(3\mu) + \pi_4(4\mu) \\ \lambda_S = \lambda\pi_0 + \lambda\pi_1 + \lambda\pi_3 + \lambda\pi_4 \end{cases}$$

Ove le due equazioni del sistema sono per l'appunto equivalenti, dal momento che valgono le equazioni di *BILANCIAMENTO LOCALE* per catene di nascita e morte: $\lambda \sum_{i=0}^{k+M-1} \pi_i = \underline{\lambda(1-\pi_4)}$. Dopodiché, dal momento che arrivano metà richieste ad uno e metà all'altro, abbiamo che la velocità di arrivo al singolo server è $(\frac{\lambda_S}{2})$, ovvero il relativo throughput per server. Applicando Little banalmente troviamo:

$$\bar{N}_S = \frac{\lambda_S}{2}(\frac{1}{\mu})$$

ove $(\frac{1}{\mu})$ è il tempo medio di permanenza di una richiesta all'interno del singolo server. In linea generale, per un tal modello che rispecchi il sistema reale del genere, vale la seguente equazione:

$$\bar{N}_S = \frac{[\lambda_S = \lambda(1-\pi_M)]}{k}(\frac{1}{\mu})$$

Appendici

6.6 Velocità di Uscita e Tassi di Transizione

6.6.1 M/M/1

Ricordiamo che: $\phi_i(t) = \min\{\xi_{Rt}, \eta_{Rt}\}$ è il tempo di soggiorno residuo nello stato i al tempo t. Sappiamo che: $\phi_i(t) \sim EXP(-q_{ii})$. Nelle dispense avevamo visto il caso: $\xi_{Rt} < \eta_{Rt}$ onde valutare il relativo tasso di transizione $q_{i,i+1}$. Una volta trovato questo si poteva tranquillamente risalire all'altro tasso per differenza, dato che si conosce $-q_{ii}$, ma per esercizio ne riportiamo il calcolo indipendente anche di $q_{i,i-1}$:

$$\tau_{i,i-1} = \frac{q_{i,i-1}}{-q_{ii}} = \int_0^\infty \Pr\{\eta_{Rt} < \xi_{Rt} \mid \xi_{Rt} = y\} f_{\xi_{Rt}}(y)dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-\mu y})\lambda\, \mathrm{e}^{-\lambda y}dy = \int_0^\infty \lambda\, \mathrm{e}^{-\lambda y}dy - \lambda \int_0^\infty \mathrm{e}^{-(\mu+\lambda)y}dy =$$

$$= 1 - \frac{\lambda}{\mu+\lambda} = \frac{\mu+\lambda-\lambda}{\mu+\lambda} = \frac{\mu}{\mu+\lambda} \implies q_{i,i-1} = \mu$$

Ricordiamo che sempre nell'M/M/1, l'equazione della distribuzione del ritardo di accodamento è il seguente:

$$F_W(y) = \Pr\{W \le y\} = 1 - \rho\, \mathrm{e}^{-\mu(1-\frac{\lambda}{\mu})} = 1 - \rho\, \mathrm{e}^{-(\mu-\lambda)y}$$

6.6.2 M/M/m

Partiamo sempre da: $\phi_i(t) = \min\{\xi_{R(t)}, \eta_{R(t)}\}$. Si ricordi che qui $[\eta_{R(t)} = \min_i(\eta_{R_i(t)})]$, dal momento che vi sono m servitori uguali ed indipendenti, ovvero che non si influiscono a vicenda. Dal momento che hanno tempi di servizio uguali e sono v.a. i.i.d., allora il minimo di quelle variabili, onde determinare la relativa velocità totale di uscita $-q_{ii}$, è distribuito esponenzialmente con parametro dato dalla somma dei parametri.

$$\Pr\{\phi_i(t) > \tau\} = F^c_{\phi_i}(\tau) = \Pr\{\xi_{R(t)} > \tau,\ \eta_{R(t)} > \tau\} =$$

$$\Pr\{\xi_{R(t)} > \tau\} \Pr\{\eta_{R(t)} > \tau\} = \mathrm{e}^{-\lambda\tau}\, \mathrm{e}^{-i\mu\tau} = [\mathrm{e}^{-\tau(\lambda+i\mu)}] \implies -q_{ii} = (\lambda + i\mu)$$

Calcoliamo ora i tassi di transizione, tenendo a mente che:

$$-q_{ii} = \begin{cases} (\lambda + i\mu),\ i < m \\ (\lambda + m\mu),\ i \ge m \end{cases}$$

Poniamoci nel caso $i < m$, per semplicità, ovvero nel primo caso. Notiamo che possiamo confondere la notazione $\{\xi_{R(t)},\ \eta_{R(t)}\} \to \{\xi_{Rt},\ \eta_{Rt}\}$, onde evitare di appesantire notevolmente la notazione:

$$\tau_{i,i+1} = \frac{q_{i,i+1}}{-q_{ii}} = \int_0^\infty \Pr\{\xi_{Rt} < \eta_{Rt} \mid \eta_{Rt} = y\} \Pr\{\eta_{Rt} = y\} dy = \int_0^\infty (1 - \mathrm{e}^{-\lambda y}) i\mu \, \mathrm{e}^{-i\mu y} dy =$$

$$= \int_0^\infty i\mu \, \mathrm{e}^{-i\mu y} dy - i\mu \int_0^\infty \mathrm{e}^{-(\lambda + i\mu)y} dy = 1 - \frac{i\mu}{(\lambda + i\mu)} \int_0^\infty (\lambda + i\mu) \, \mathrm{e}^{-(\lambda + i\mu)y} dy =$$

$$= 1 - \frac{i\mu}{(\lambda + i\mu)} = \frac{\lambda + i\mu - i\mu}{(\lambda + i\mu)} = \frac{\lambda}{(\lambda + i\mu)} \implies q_{i,i+1} = \lambda$$

Analogamente per il tasso di transizione inverso:

$$\tau_{i,i-1} = \frac{q_{i,i-1}}{-q_{ii}} = \int_0^\infty \Pr\{\eta_{Rt} < \xi_{Rt} \mid \eta_{Rt} = y\} \Pr\{\eta_{Rt} = y\} dy = \int_0^\infty (1 - \mathrm{e}^{-i\mu y}) \lambda \, \mathrm{e}^{-\lambda y} dy =$$

$$= \int_0^\infty \lambda \, \mathrm{e}^{-\lambda y} dy - \lambda \int_0^\infty \mathrm{e}^{-(\lambda + i\mu)y} dy = 1 - \frac{\lambda}{(\lambda + i\mu)} \int_0^\infty (\lambda + i\mu) \, \mathrm{e}^{-(\lambda + i\mu)y} dy =$$

$$= 1 - \frac{\lambda}{\lambda + i\mu} = \frac{\lambda + i\mu - \lambda}{\lambda + i\mu} = \frac{i\mu}{\lambda + i\mu} \implies q_{i,i-1} = i\mu$$

Il procedimento è il medesimo anche nell'altro caso: $i \geq m$.

6.6.3 Server Farm Exercise

Chiamiamo rispettivamente con: $\{\xi_{Rt},\ \eta_{Rt}\}$ le v.a. tempi di guasto e di riparazione residui al tempo t. Abbiamo che il singolo i-esimo server ha un tempo di guasto residuo pari a:

$$\eta_{iRt} \sim EXP(f = f_p + f_m + f_d)$$

Dal momento che il minimo di una v.a. esponenziale è pari ad una v.a. ancora esponenziale con parametro dato dalla somma dei parametri. Per gli stessi motivi, avendo K server sempre in funzione, indipendenti tra di loro (non si pestano i piedi, sostanzialmente), avremo che: $\eta_{Rt} \sim EXP(Kf)$. Stesso discorso dicasi per i SERVER IN RIPARAZIONE, che sono sostanzialmente i (fissiamo tale notazione):

$$\begin{cases} \xi_{jRt} \sim EXP(\mu) \\ \xi_{Rt} \sim EXP(i\mu) \end{cases}$$

Calcoliamo la velocità totale di uscita: $-q_{K+M-i|K+M-i}$:

$$\phi_{i':=K+M-i}(t) = \min\{\eta_{Rt}, \xi_{Rt}\} \implies F_{\phi_{i'}}(\tau) = \Pr\{\phi_{i'}(t) > \tau\} = \Pr\{\xi_{Rt} > \tau,\ \eta_{Rt} > \tau\} =$$

$$= \Pr\{\eta_{Rt} > \tau\} \Pr\{\xi_{Rt} > \tau\} = \mathrm{e}^{-Kf\tau} \, \mathrm{e}^{i\mu\tau} = \mathrm{e}^{-(Kf + i\mu)\tau} \implies -q_{i'i'} = Kf + i\mu$$

Mancano ancora i tassi di transizione:

$$\tau_{i',i'+1} = \frac{q_{i',i'+1}}{-q_{i'i'}} = \int_0^\infty \Pr\{\xi_{Rt} < \eta_{Rt} \mid \eta_{Rt} = y\} \Pr\{\eta_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-i\mu y}) Kf \, \mathrm{e}^{-Kfy} dy = \int_0^\infty Kf \, \mathrm{e}^{-Kfy} dy - Kf \int_0^\infty \mathrm{e}^{-(Kf + i\mu)y} dy =$$

$$= 1 - \frac{Kf}{(Kf + i\mu)} \int_0^\infty (Kf + i\mu) \, \mathrm{e}^{-(Kf + i\mu)y} dy =$$

$$= \frac{Kf + i\mu - Kf}{Kf + i\mu} = \frac{i\mu}{Kf + i\mu} \implies q_{i',i'+1} = i\mu$$

Analogamente vale:

$$\tau_{i',i'-1} = \frac{q_{i',i'-1}}{-q_{i'i'}} = \int_0^\infty \Pr\{\eta_{Rt} < \xi_{Rt} \mid \xi_{Rt} = y\} \Pr\{\xi_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-Kfy}) i\mu \, \mathrm{e}^{-i\mu y} dy = \int_0^\infty i\mu \, \mathrm{e}^{-i\mu y} dy - i\mu \int_0^\infty \mathrm{e}^{-(Kf+i\mu)y} dy =$$

$$= 1 - \frac{i\mu}{(Kf + i\mu)} \int_0^\infty (Kf + i\mu) \, \mathrm{e}^{-(Kf+i\mu)y} dy =$$

$$= 1 - \frac{i\mu}{(Kf + i\mu)} = \frac{Kf + i\mu - i\mu}{(Kf + i\mu)} \implies q_{i',i'-1} = Kf$$

Sostanzialmente i tassi di transizione sono ovviamente i medesimi prendendo in considerazione la definizione di stato alternativa con la quale si è poi proseguito lo svolgimento dell'esercizio, ovvero: $N(t) := |server\ in\ riparazione|$. Ricordiamo che differenti definizioni di stato possono poi portare allo stesso DTT.

6.6.4 Client-Server System Exercise

Supponiamo che il numero di clienti attuale nel sistema a coda, data la definizione di stato scelta, sia: $N(t) = i$. Utilizziamo le stesse notazioni usate nell'esercizio, ma evitando di appesantire inutilmente la notazione:

$$\tau_{i,i+1} = \frac{q_{i,i+1}}{-q_{ii}} = \Pr\{\xi_{Rt} < \min(\eta_{Rt}, \theta_{Rt})\} =$$

$$= \int_0^\infty \Pr\{\xi_{Rt} < \min(\eta_{Rt}, \theta_{Rt}) \mid \min(\eta_{Rt}, \theta_{Rt}) = y\} \Pr\{\min(\eta_{Rt}, \theta_{Rt}) = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-(M-i)\lambda y})[\mu + (i-1)\gamma] \, \mathrm{e}^{-[\mu+(i-1)\gamma]y} dy =$$

$$= \int_0^\infty [\mu + (i-1)\gamma] \, \mathrm{e}^{-[\mu+(i-1)\gamma]y} dy - [\mu + (i-1)\gamma] \int_0^\infty \mathrm{e}^{-[(M-i)\lambda+\mu+(i-1)\gamma]y} dy =$$

$$= 1 - \frac{\mu + (i-1)\gamma}{[(M-i)\lambda + \mu + (i-1)\gamma]} = \frac{(M-i)\lambda}{[(M-i)\lambda + \mu + (i-1)\gamma]} \implies q_{i,i+1} = (M-i)\lambda$$

Analogamente nell'altro caso:

$$\tau_{i,i-1} = \frac{q_{i,i-1}}{-q_{ii}} = \Pr\{\min(\eta_{Rt}, \theta_{Rt}) < \xi_{Rt}\} =$$

$$= \int_0^\infty \Pr\{\min(\eta_{Rt}, \theta_{Rt}) < \xi_{Rt} \mid \xi_{Rt} = y\} \Pr\{\xi_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-[(i-1)\gamma+\mu]y})(M-i)\lambda \, \mathrm{e}^{-(M-i)\lambda y} dy =$$

$$= \int_0^\infty (M-i) \, \mathrm{e}^{-(M-i)\lambda y} dy - (M-i)\lambda \int_0^\infty \mathrm{e}^{-[(i-1)\gamma+\mu+(M-i)\lambda]y} dy =$$

$$= 1 - \frac{(M-i)\lambda}{[(i-1)\gamma + \mu + (M-i)\lambda]} \int_0^\infty [(i-1)\gamma + \mu + (M-i)\lambda] \, \mathrm{e}^{-[(i-1)\gamma+\mu+(M-i)\lambda]y} dy =$$

$$= 1 - \frac{(M-i)\lambda}{[(i-1)\gamma + \mu + (M-i)\lambda]} = \frac{(i-1)\gamma + \mu}{[(i-1)\gamma + \mu + (M-i)\lambda]} \implies q_{i,i-1} = (i-1)\gamma + \mu$$

6.6.5 Composite Model Exercise

Nonostante il DTT sia praticamente identico a quello della M/M/m/0, con l'unica differenza nello starting state, rieseguiamo il calcolo della velocità totale di uscita e dei tassi di transizione, onde fortificare la nostra preparazione in tal genere di calcoli:

$$\begin{cases} \phi_i(t) = \min(\xi_{Rt}, \eta_{Rt}) \\ \begin{cases} \eta_{Rt} \sim EXP(i\mu) \\ \xi_{Rt} \sim EXP(\tau) \end{cases} \end{cases}$$

Procediamo con il calcolo della velocità totale di uscita:

$$F_{\phi_i}(\tau') = \Pr\{\min(\xi_{Rt}, \eta_{Rt}) > \tau'\} = \Pr\{\xi_{Rt} > \tau',\ \eta_{Rt} > \tau'\} = \underline{\Pr\{\xi_{Rt} > \tau'\} \Pr\{\eta_{Rt} > \tau'\}} =$$

$$= \mathrm{e}^{-\tau\tau'} \mathrm{e}^{-i\gamma\tau'} = \mathrm{e}^{-(\tau+i\gamma)\tau'} \implies q_{ii} = (\tau + i\gamma)$$

Notiamo che il termine sottolineato sta ad indicare che tale possibilità è concessa dall'indipendenza delle variabili in questione. Calcoliamo ora i tassi:

$$\tau_{i,i+1} = \frac{q_{i,i+1}}{-q_{ii}} = \Pr\{\xi_{Rt} < \eta_{Rt}\} = \int_0^\infty \Pr\{\xi_{Rt} < \eta_{Rt} \mid \eta_{Rt} = y\} \Pr\{\eta_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-\tau y}) i\gamma \, \mathrm{e}^{-i\gamma y} dy = \int_0^\infty i\gamma \, \mathrm{e}^{-i\gamma y} dy - i\gamma \int_0^\infty \mathrm{e}^{-(\tau+i\gamma)y} dy =$$

$$= 1 - \frac{i\gamma}{\tau + i\gamma} \int_0^\infty (\tau + i\gamma) \, \mathrm{e}^{-(\tau+i\gamma)y} dy = 1 - \frac{i\gamma}{\tau + i\gamma} = \frac{\tau + i\gamma - i\gamma}{\tau + i\gamma} = (\frac{\tau}{\tau + i\gamma}) \implies q_{i,i+1} = \tau$$

Analogamente:

$$\tau_{i,i-1} = \frac{q_{i,i-1}}{-q_{ii}} = \Pr\{\eta_{Rt} < \xi_{Rt}\} = \int_0^\infty \Pr\{\eta_{Rt} < \xi_{Rt} \mid \xi_{Rt} = y\} \Pr\{\xi_{Rt} = y\} dy =$$

$$= \int_0^\infty (1 - \mathrm{e}^{-i\gamma y}) \tau \, \mathrm{e}^{-\tau y} dy =$$

$$= \int_0^\infty \tau \, \mathrm{e}^{-\tau y} dy - \tau \int_0^\infty \mathrm{e}^{-(i\gamma+\tau)y} dy = 1 - \frac{\tau}{i\gamma + \tau} \int_0^\infty (i\gamma + \tau) \, \mathrm{e}^{-(i\gamma+\tau)y} dy =$$

$$= 1 - \frac{\tau}{i\gamma + \tau} = \frac{i\gamma + \tau - \tau}{i\gamma + \tau} = \frac{i\gamma}{\tau + i\gamma} \implies q_{i,i-1} = i\gamma$$

6.7 Tips and Tricks

6.7.1 MTTF in Reliability/Availability Computational Models

Talvolta potreste incappare nel dubbio che l'MTTF che si calcoli in un modello di calcolo per l'Affidabilità, magari con stati assorbenti, possa essere uguale all'MTTF calcolato in un modello di calcolo per la Disponibilità. Questo intuitivamente è vero, ma bisogna ricordarsi di partire dalle condizioni iniziali giuste nel modello di Affidabilità! Riprendiamo l'esercizio: **CMTC with Absorbing States**. Introduciamo il modello di calcolo per la Disponibilità, semplicemente partendo dal modello dell'Affidabilità e reintroducendo il ramo della riparazione dallo stato di FAILURE. Scriviamo anzitutto la distribuzione di regime, ricordandoci che nel

caso in questione vi è una Single Repair Facility, e che quindi il tasso di riparazione complessivo è sempre μ:

$$\begin{cases} \pi_i = \pi_0(\frac{\mu}{\lambda})^i \frac{1}{i!} \\ \pi_0 = \frac{1}{\sum_{i=0}^{2}(\frac{\mu}{\lambda})^i \frac{1}{i!}} = \frac{1}{1+\frac{\mu}{\lambda}+(\frac{\mu}{\lambda})^2\frac{1}{2}} \end{cases}$$

Ricordando la formula ricavata nel **Server Farm Exercise**, la quale affermava che l'MTTF si può calcolare tranquillamente effettuando la differenza tra il tempo medio di ricorrenza nello stato di FAILURE ed il tempo di soggiorno nello stato di FAILURE, corrispondente quest'ultimo al MTTR:

$$MTTF = \frac{1}{\pi_0\mu} - \frac{1}{\mu} = \frac{1}{\frac{\mu}{1+\frac{\mu}{\lambda}+(\frac{\mu}{\lambda})^2\frac{1}{2}}} - \frac{1}{\mu} =$$

$$= \frac{2\lambda^2+2\lambda\mu+\mu^2}{2\lambda^2\mu} - \frac{1}{\mu} = \frac{1}{\mu} + \frac{1}{\lambda} + \frac{\mu}{2\lambda^2} - \frac{1}{\mu} = \frac{1}{\mu} + \frac{\mu}{2\lambda^2}$$

Notiamo che tra l'altro, tale termine non coincide assolutamente con l'MTTF ricavato nel caso di quello calcolato a lezione mediante tecnica degli Stati Assorbenti oppure la L_i-tecnica (mediante tempi medi di assorbimento). L'intoppo sta nel fatto che nel modello di calcolo dell'Affidabilità con stati assorbenti, siamo liberi di scegliere la condizione iniziale! Entrambe le tecniche in questo modello prenderanno in considerazione anche il transitorio e le condizioni iniziali dunque! Naturalmente avremo soluzioni diverse al variare delle condizioni iniziali, dal momento che la media è fatta lungo tutto l'asse temporale, includendo così sia situazioni di transitorio che di regime. SOLTANTO se in questo particolare caso partiamo dallo stato adiacente a quello assorbente $\{1\}$, vi è una coincidenza esatta tra gli MTTF calcolati nei due modelli. L'interpretazione è lasciata al lettore.

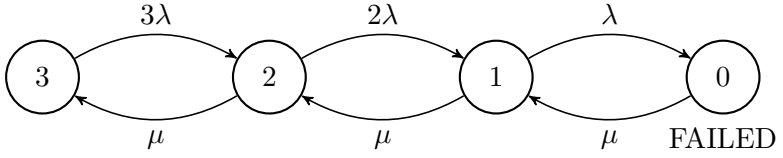
Infatti, se partissimo da $\pi_1 = 1$, otterremmo il seguente sistema nel modello di calcolo dell'Affidabilità:

$$\begin{cases} \begin{cases} -2L_2\lambda + L_1\mu = 0 \\ L_1 = \frac{2L_2\lambda}{\mu} \end{cases} \\ \begin{cases} L_2(2\lambda) - L_1(\lambda+\mu) = -1 \\ L_2(2\lambda) - \frac{2L_2\lambda(\lambda+\mu)}{\mu} = -1 \end{cases} \\ L_2(2\lambda\mu) - 2L_2\lambda_2^L(2\lambda\mu) = -\mu \end{cases} \implies \begin{cases} L_2 = \frac{\mu}{2\lambda^2} \\ L_1 = \frac{2\mu\lambda}{\mu 2\lambda^2} = \frac{1}{\lambda} \end{cases}$$

Sommando i tempi medi di assorbimento troviamo:

$$MTTF = MTTA = MTTA_1 + MTTA_2 = [L_1 + L_2 = \frac{1}{\lambda} + \frac{\mu}{2\lambda^2}]$$

Proviamo a fare lo stesso esercizio supponendo che si abbia un terzo stato, onde osservare una possibile generalizzazione in questa configurazione, rappresentata dal seguente DTT:



Calcoliamo quindi l'MTTF nei due modelli:

$$\begin{cases} MTTF = \dfrac{1}{\pi_0} - \dfrac{1}{\mu} \\ \left[\begin{array}{l} \pi_0 = \dfrac{1}{1 + \frac{\mu}{\lambda} + \frac{\mu^2}{\lambda^2}\frac{1}{2} + \frac{\mu^3}{\lambda^3}\frac{1}{6}} \\ \dfrac{1}{\pi_0\mu} = \dfrac{(1 + \frac{\mu}{\lambda} + \frac{\mu^2}{\lambda^2}\frac{1}{2} + \frac{\mu^3}{\lambda^3}\frac{1}{6})}{\mu} = \dfrac{6\lambda^3 + 6\lambda^2\mu + 3\lambda\mu^2 + \mu^3}{6\lambda^3\mu} = \\ \dfrac{1}{\mu} + \dfrac{1}{\lambda} + \dfrac{\mu}{2\lambda^2} + \dfrac{\mu^2}{6\lambda^3} = [\dfrac{1}{\pi_0\mu} := \sum_{i=0}^{n} (\dfrac{\mu^{i-1}}{\mu^i})\dfrac{1}{i!}] \end{array} \right. \end{cases}$$

Riassemblando opportunamente i termini troviamo:

$$MTTF = \frac{1}{\pi_0\mu} - \frac{1}{\mu} =$$

$$= \frac{1}{\lambda} + \frac{\mu}{2\lambda^2} + \frac{\mu^2}{6\lambda^3} := \sum_{i=1}^{n} (\frac{\mu^{i-1}}{\mu^i})\frac{1}{i!}$$

Proviamo ora a calcolare la stessa quantità nel modello di affidabilità:

$$\bar{Q} = \begin{bmatrix} -3\lambda & 3\lambda & 0 & 0 \\ \mu & -(\mu + 2\lambda) & 2\lambda & 0 \\ 0 & \mu & -(\mu + \lambda) & \lambda \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Ricordando la partizione: $S = N \cup A$, restringiamo la matrice al set degli stati transitori:

$$\bar{Q}_N = \begin{bmatrix} -3\lambda & 3\lambda & 0 \\ \mu & -(\mu + 2\lambda) & 3\lambda \\ 0 & \mu & -(\mu + \lambda) \end{bmatrix}$$

Impostiamo l'equazione matriciale:

$$\begin{bmatrix} L_3 & L_2 & L_1 \end{bmatrix} \begin{bmatrix} -3\lambda & 3\lambda & 0 \\ \mu & -(\mu + 2\lambda) & 3\lambda \\ 0 & \mu & -(\mu + \lambda) \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}$$

E risolviamo il sistema che rappresenta:

$$\begin{cases} L_3(-3\lambda) + L_2\mu = 0 \\ L_3(3\lambda) - L_2(\mu + 2\lambda) + L_1\mu = 0 \\ 2L_2\lambda - L_1(\lambda + \mu) = -1 \end{cases} \implies$$

$$\Longrightarrow \begin{cases} \begin{cases} L_3(-3\lambda) + L_2\mu + L_3(3\lambda) - L_2(\mu + 2\lambda) + L_1\mu - 2L_2\lambda - L_1(\lambda + \mu) = -1 \\ L_2[\mu - \mu - 2\lambda + 2\lambda] + L_1(\mu - \lambda - \mu) = -1 \\ L_1 = \dfrac{1}{\mu} \end{cases} \\ \begin{cases} 2\lambda L_2 - \dfrac{\lambda + \mu}{\lambda} = -1 \\ 2\lambda^2 L_2 - \lambda - \mu = -\lambda \\ L_2 = \dfrac{\mu}{2\lambda^2} \end{cases} \\ \begin{cases} -3\lambda L_3 + \dfrac{\mu\mu}{2\lambda^2} = 0 \\ L_3 = \dfrac{\mu^2}{6\lambda^3} \end{cases} \end{cases}$$

Quindi assemblando i termini troviamo:

$$MTTF = L_1 + L_2 + L_3 = [\frac{1}{\mu} + \frac{\mu}{2\lambda^2} + \frac{\mu^2}{6\lambda^3}]$$

Si può ovviamente generalizzare per una catena che abbia distribuzione di regime uguale all'M/M/m/0.

6.7.2 Failure Rate e MTTF in parallelo

Attenzione quando si manipolano i failure rate in concomitanza con l'MTTF in parallelo! Quando si effettua un collegamento parallelo, il suo failure rate $h(t)$ non è più il reciproco del valor medio del lifetime del sistema risultante! Infatti questo accade soltanto per sistemi che hanno distribuzione del lifetime esponenziale, come ad esempio i risultanti da configurazioni serie, i quali a valle della combinazione hanno ancora un lifetime distribuito esponenzialmente, con parametro dato dalla somma dei parametri (i failure rate $h_i(t)$ dei sistemi componenti. Con la configurazione parallelo, il sistema risultante ha un lifetime che è costituito dalla combinazione lineare di altri esponenziali, ma non è un esponenziale puro, quindi non vi è un collegamento diretto visibile con il $MTTF$. Come esempio, riprendiamo l'esercizio **MTTF of a fiber link** e proviamo a calcolare la configurazione del *Partially protected link* scenario:

$$1 - (1 - \mathrm{e}^{-h_F t})^2 = 1 - 1 - \mathrm{e}^{-2h_F t} + 2\,\mathrm{e}^{-h_F t} = 2\,\mathrm{e}^{-h_F t}\,\mathrm{e}^{-2h_F t}$$

Quindi integrando opportunamente troviamo:

$$MTTF_L = \int_0^\infty R_F(t)dt = 2\int_0^\infty \mathrm{e}^{-h_F t}dt - \int_0^\infty \mathrm{e}^{-2h_F t}dt =$$

$$= [\frac{2}{h} - \frac{1}{2h}] = \frac{1}{h}(2 - \frac{1}{2}) = \frac{3}{2}MTTF_F = \frac{3}{2}(22.78y)$$

Tuttavia, e qui sta l'errore, non possiamo dire che il failure rate del sistema risultante sia il reciproco di questo MTTF appena trovato! $\iff h_L \neq \frac{1}{MTTF_L}$!! Supponendo che ciò fosse vero, procediamo con il calcolo del MTTF del sistema complessivo, sommando i vari failure rate (operazione lecita, dal momento che tali sistemi sono in serie), trovando:

$$h = 2h_P + 2h_F + h_L = 0.52926y^{-1} \implies \underline{MTTF_{PP}} = \frac{1}{h} = 1.8994y$$

Alquanto basso come MTTF! È vero che abbiamo raddoppiato solo il link di comunicazione in fibra, dal momento che gli altri sistemi in serie costituiscono il bottleneck. Tuttavia è abbastanza improbabile che l'MTTF dell'intero sistema rispetto al caso *Unprotected* migliori soltanto di qualche mese! Infatti il link in fibra ha un MTTF di gran lunga superiore a quello degli altri. Tra l'altro nemmeno coincide con l'h calcolato nell'esercizio, il quale è sicuramente corretto avendo proceduto preliminarmente calcolando l'affidabilità complessiva sfruttando le proprietà dell'affidabilità in configurazione parallelo e serie, ed avendo integrato A POSTERIORI, trovando il vero MTTF del sistema *Partially protected link*.

6.8 Trasformate di Laplace

6.8.1 Proprietà e Trasformate di Funzioni Notevoli

Tratto da *Wikipedia*:

- **Proprietà**:
  - Linearità:
    $\mathcal{L}\{af(t) + bg(t)\} = a\mathcal{L}\{f(t)\} + b\mathcal{L}\{g(t)\}$
  - Derivazione:
    $\mathcal{L}\{f'\} = s\mathcal{L}\{f\} - f(0^+)$
  - Integrazione:
    $\mathcal{L}\{\int_0^t f(\tau)d\tau\} = \frac{1}{s}\mathcal{L}\{f(t)\}$
  - Prodotto di Convoluzione:
    $\mathcal{L}\{f \star g\} = \mathcal{L}\{f\}\mathcal{L}\{g\}$
- **Trasformate di Funzioni notevoli**:
  - *Delta di Dirac*:
    $\mathcal{L}\{\delta(t)\} = 1$
  - *Funzione gradino di Heaviside*:
    $\mathcal{L}\{\Theta(t)\} = \frac{1}{s}$
  - *Funzione esponenziale*:
    $\mathcal{L}\{\mathrm{e}^{\alpha t}\} = \frac{1}{s-\alpha}$
  - *Seno*:
    * $\mathcal{L}\{\sin(\alpha t)\} = \frac{\alpha}{s^2+\alpha^2}$
    * $\mathcal{L}\{e^{\beta t}\sin(\alpha t)\} = \frac{\alpha}{(s-\beta)^2+\alpha^2}$
  - *Coseno*:
    * $\mathcal{L}\{\cos(\alpha t)\} = \frac{s}{s^2+\alpha^2}$
    * $\mathcal{L}\{e^{\beta t}\cos(\alpha t)\} = \frac{s-\beta}{(s-\beta)^2+\alpha^2}$

A noi in realtà interessano le antitrasformate di Laplace, ma ovviamente viene tutto in automatico riguardandole bene.

6.8.2 CMTC with Absorbing States

Verifica TL:

$$\begin{cases} s\pi_2^\star(s) - (\pi_2(0) = 1) = -2\lambda\pi_2^\star(s) + \mu\pi_1^\star(s) \\ s\pi_1^\star(s) - (\pi_1(0) = 0) = -(\lambda + \mu)\pi_1^\star(s) + 2\lambda\pi_2^\star(s) \\ s\pi_0^\star(s) - (\pi_0(0) = 0) = \lambda\pi_1^\star(s) \end{cases}$$

$$\begin{cases} \pi_2^\star(s) = \dfrac{\mu\pi_1^\star(s) + 1}{(s + 2\lambda)} \\ \begin{cases} \pi_1^\star(s)(s + \lambda + \mu) = 2\lambda\pi_2^\star(s) \\ \pi_1^\star(s) = \dfrac{2\lambda[\mu\pi_1^\star(s) + 1]}{(s + 2\lambda)(s + \lambda + \mu)} \\ \pi_1^\star(s)[(s + 2\lambda)(s + \lambda + \mu)] = 2\lambda\mu\pi_1^\star(s) + s\lambda \\ \pi_1^\star(s) = \dfrac{2\lambda}{[(s + 2\lambda)(s + \lambda + \mu) - 2\lambda\mu]} \end{cases} \\ \pi_0^\star(s) = \dfrac{2\lambda^2}{s[(s + 2\lambda)(2 + \lambda + \mu) - 2\lambda\mu]} \end{cases}$$

Quindi troviamo alla fine quello che ci serve, ovvero:

$$\pi_0^\star(s) = \frac{2\lambda^2}{s[s^2 + s\lambda + s\mu + 2s\lambda + 2\lambda^2 + 2\lambda\mu - 2\lambda\mu]} = \frac{2\lambda^2}{s[s^2 + s(3\lambda + \mu) + 2\lambda^2]}$$

Il seguito dell'esercizio vorrebbe che:

$$MTTF = \int_0^\infty (1 - \mathcal{L}^{-1}\{\pi_0^\star(s)\}(t))dt$$

Quindi da adesso si dovrebbe procedere espandendo $\pi_0^\star(s)$ in fratti semplici ed antitrasformando secondo Laplace membro a membro, ma non lo faremo nel seguito per brevità. Poi, ricordando che:

$$R(t) = \Pr\{X > t\} = 1 - \Pr\{X \leq t\} = 1 - \pi_0(t)$$

Si può tranquillamente integrare e trovare l'MTTF. Si ricordi che l'ultima espressisone vale ovviamente soltanto se lo $\{0\}$ è uno STATO ASSORBENTE!

6.8.3 Homeworks 2-1

Si tenti l'approccio mediante trasformate di Laplace per ricavare $\pi_0(t)$ in seguito antitrasformando opportunamente:

$$\begin{cases} \dfrac{d\pi_2(t)}{dt} = -2\lambda\pi_2(t) + \pi_1(t)\mu \\ \dfrac{d\pi_1(t)}{dt} = -\pi_1(t)(\mu + \lambda) + 2\pi_2(t)c\lambda \\ \dfrac{d\pi_0(t)}{dt} = 2\pi_2(t)\lambda(1 - c) + \pi_1(t)\lambda \end{cases}$$

Trasformando il sistema, membro a membro per ogni equazione, otteniamo:

$$
\begin{cases}
\begin{cases}
s\pi_2^\star(s) - (\pi_2(0)=1) = -2\lambda\pi_2^\star(s) + \pi_1^\star(s)\mu \\
\pi_2^\star(s) = \dfrac{\pi_1^\star(s)\mu + 1}{(s+2\lambda)}
\end{cases} \\
\begin{cases}
s\pi_1^\star(s) - (\pi_1(0)=0) = -\pi_1^\star(s)(\mu+\lambda) + 2\dfrac{(\pi_1^\star(s)\mu+1)}{(s+2\lambda)}\lambda c \\
s(s+2\lambda)\pi_1^\star(s) + \pi_1^\star(s)(\mu+\lambda)(s+2\lambda) - 2\pi_1^\star(s)\mu\lambda c - 2\lambda c = 0 \\
\pi_1^\star(s) = \dfrac{2\lambda c}{[(s+2\lambda)(s+\lambda+\mu) - 2\mu\lambda c]}
\end{cases} \\
s\pi_0^\star(s) - (\pi_0(0)=0) = 2\lambda(1-c)\pi_2^\star(s) + \pi_1^\star(s)\lambda
\end{cases}
$$

$$
\begin{cases}
\pi_1^\star(s) = \dfrac{2\lambda c}{[(s+2\lambda)(s+\lambda+\mu) - 2\mu\lambda c]} \\
\begin{cases}
\pi_2^\star(s) = \dfrac{\frac{2\lambda c\mu}{[(s+2\lambda)(s+\lambda+\mu)-2\lambda\mu c]} + 1}{(s+2\lambda)} = \dfrac{2\lambda c\mu + (s+2\lambda)(s+\lambda+\mu) - 2\lambda c\mu}{(s+2\lambda)[(s+2\lambda)(s+\lambda+\mu) - 2\lambda\mu c]} = \\
= \dfrac{s+\lambda+\mu}{[(s+2\lambda)(s+\lambda+\mu) - 2\lambda\mu c]}
\end{cases} \\
\begin{cases}
\pi_0^\star(s) = \dfrac{2\lambda(1-c)(s+\lambda+\mu)}{s[(s+2\lambda)(s+\lambda+\mu) - 2\lambda\mu c]} + \dfrac{2\lambda c}{[(s+2\lambda)(s+\lambda+\mu) - 2\lambda\mu c]} = \\
= \dfrac{2\lambda[s+\lambda+\mu-cs-c\lambda+c]}{s[(s+2\lambda)(s+\lambda+\mu) - 2\lambda\mu c]} = \\
= \dfrac{2\lambda[s(1-c) - c(\lambda+\mu) + \lambda + \mu + c = [s(1-c) + (\lambda+\mu)(1-c) + c]]}{s[(s+2\lambda)(s+\lambda+\mu) - 2\lambda\mu c]}
\end{cases}
\end{cases}
$$

Si procede ovviamente allo stesso modo dell'esercizio precedente, espandendo in fratti semplici la seguente espressione di $\pi_0^\star(s)$, antitrasformandola secondo Laplace, e ciò che si ottiene lo si deve sottrarre da 1 ed integrando opportunamente, ottenendo così l'MTTF.

6.9 Sigle Network Technologies

Si ringrazia anticipatamente il *Dott. Andrea Camisa* per il prezioso aiuto:

6.9.1 Sigle IEEE

- **IEEE 802.1: Architettura e descrizione generale delle LAN**
  - 802.1aq Shortest Path Bridging (SPB), evoluzione di 802.1w e 802.1s;
  - 802.1AX Rettifica di 802.3ad per il Link Aggregation Control Protocol (LACP) (riposiziona alcuni livelli 802.1 come 802.1X security al di sopra del livello di Link Aggregation);
  - 802.1D Funzioni dei bridge e del protocollo Spanning Tree Protocol (STP);
  - 802.1p Priorità di traffico e filtraggio dei pacchetti multicast da parte dei bridge;
  - 802.1Q Etichetta (tag) per la realizzazione delle VLAN e la differenziazione dei tipi di traffico (QoS di livello 2, vedi 802.1p);
  - 802.1s Multiple Spanning Tree (MST) per le VLAN;
  - 802.1v VLAN per porta e per protocollo;
  - 802.1w Rapid Spanning Tree Protocol (RSTP);

  - 802.1X Autenticazione di livello 2 nelle reti (usato anche in 802.11), definisce l'incapsulamento di Extensible Authentication Protocol (EAP) in IEEE 802.

- **IEEE 802.3: tecnologia per reti locali (LAN)**

  La seguente è derivata da Ethernet. Definisce CSMA/CD.

  - 802.3ad Link Aggregation Control Protocol (LACP), rettificato da 802.1AX;
  - 802.3ac Allungamento delle trame con etichetta VLAN (vedi 802.1Q);
  - 802.3af Power over Ethernet (PoE).

- **IEEE 802.11: Specifiche MAC (Media Access Control) e PHY (Physical Layer) per implementare WLAN (Wireless Local Area Network)**

  - Vedi slide Wireless a pag 17;
  - 802.11i Sicurezza delle WLAN;
  - 802.11f Standard per lo scambio di informazioni tra AP per Handoff/Handover/Roaming;
  - 802.11p Standard per VANET.

6.9.2 Sigle varie divise per argomento

- **Enti vari**

  - IEEE: *Institute of Electrical and Electronics Engineers*;
  - ANSI: *American National Standards Institute*;
  - ISO: *International Standard Organization*;
  - IETF: *Internet Engineering Task Force*;
  - RFC: *Request for Comments*;
  - IANA: *Internet Assigned Numbers Authority*;
  - ICANN: *Internet Corporation for Assigned Names and Numbers*;
  - EMEA: *Europe Middle East Africa*.

- **Sigle varie**

  - ISO/OSI: Modello *Open System Interconnection* della ISO;
  - SAP: *Service Access Point*;
  - SSAP: *Source Service Access Point*;
  - DSAP: *Destination Service Access Point*;
  - PDU: *Protocol Data Unit*;
  - RTT: *Round Trip Time*;
  - ACK: Acknowledgement.

- **Sigle di livello 2**

  - LACP: *Link Aggregation Control Protocol*;

  - ATM: *Asynchronous Transfer Mode*:
    * CBR: *Constant Bit Rate*;
    * UBR: *Unspecified Bit Rate*;
    * AAL: *ATM Adaption Layer*;
    * VC: *Virtual Circuit*;
    * VCI: *Virtual Channel Identifier*;
    * VPI: *Virtual Path Identifier*.
  - MAC: *Media Access Control* (identifica un protocollo di accesso e per estensione il livello data link);
  - LLC: *Logical Link Control* (sottolivello del livello 2 per le trame 802.3);
  - OUI: *Organization Unique Identifier* (usato negli indirizzi MAC e nel primo campo a 3 byte dell'estensione SNAP);
  - SFD: *Starting Frame Delimiter* (il byte che sta dopo il preambolo nelle trame Ethernet e 802.3);
  - FCS: *Frame Check Sequence* (i 4 byte che stanno alla fine di una trama Eth/802.3 per il controllo del pacchetto); itemIFS *Inter Frame Spacing* (terminologia 802.3);
  - IPG: *Inter Packet Gap* (terminologia Ethernet);
  - SNAP: *Subnetwork Access Protocol Extension*;
  - BLAN: Bridged LAN (reti locali estese per mezzo di bridge o switch);
  - ARP: *Address Resolution Protocol*;
  - RARP: *Reverse Address Resolution Protocol*:

- **Spanning Tree Protocol**
  - STP: *Spanning Tree Protocol*;
  - RSTP: *Rapid Spanning Tree Protocol*;
  - MST: *Multiple Spanning Tree* (MST regions: regioni con switch che supportano tutti 802.1s);
  - SST: *Single Spanning Tree* (SST regions: regioni con switch che non supportano tutti 802.1s);
  - BPDU: *Bridge Protocol Data Unit*;
  - TCN: *Topology Change Notification*;
  - TC: *Topology Change*;
  - TCA: *Topology Change Acknowledgement*.

- **Virtual LAN**
  - VLAN: *Virtual Local Area Network*:
    * VID: *VLAN Identifier*;
    * TPID: *Tag Protocol Identifier* (il campo nelle trame Ethernet/802.3 che dice che subito dopo c'è il TCI, valore 81-00);
    * TCI: *Tag Control Information*;
  - GVRP: *GARP VLAN Registration Protocol* (una particolare istanza di GARP per definire le VLAN sugli switch);

- MVRP: *Multiple VLAN Registration Protocol*;
- GARP: *Generic Attribute Registration Protocol* (protocollo usato dagli switch per comunicare attributi tra loro);
- IVL: *Independent Virtual LAN* (IVL switch: con filtering database indipendenti per le VLAN, permettono ad un dispositivo di essere presente contemporaneamente su più VLAN):
  - *Filtering Database*, creati per ogni VLAN;
  - FID: *Filtering Identifier* (identificatore di un Filtering Database).
- SVL: *Shared Virtual LAN* (SVL switch: con filtering database condiviso tra le VLAN, non permettono quello che fanno fare gli IVL);
- PVST: *Per VLAN Spanning Tree* (Protocollo proprietario cisco);
- PVST+: *PVST Plus*;
- PVLAN: *Private VLAN*.

- **Sigle di livello 1**:
  - FDDI: *Fiber Distributed Data Interface*;
  - PHY: *Physical Layer*;
  - CSMA/CD: *Carrier Sense Multiple Access / Collision Detection*;
  - CSMA/CA: *Carrier Sense Multiple Access / Collision Avoidance*;
  - DSL: *Digital Subscriber Line*;
  - ADSL: *Asymmetric DSL*;
  - ASIC: *Application Specific Integrated Circuit*.

- **Sigle di livello 3**:
  - IP: *Internet Protocol*;
  - IPX: *Internetwork Packet Exchange*;
  - CIDR: *Classless Inter-Domain Routing*;
  - LIS: *Logical IP Subnet* (rete logica IP);
  - MTU: *Maximum Transmission Unit* (dimensione max del datagramma IP);
  - ToS: *Type of Service* (vecchio nome del campo di 8 bit usato dapprima in IPP e poi in DiffServ);
  - DS: *Differentiated Services* (nuovo nome del campo ToS in DiffServ);
  - IPP: *IP Precedence* (i primi 3 bit nel ToS);
  - DSCP: *DiffServ Code Point* (i primi 6 bit nel DS):
    - CS: *Class Selector* (da CS0 [000 000] a CS7 [111 000], mantenendo gli ultimi 3 bit a 0); *AFxy*: *Assured Forwarding* (da AF1x [001xxx] a AF4x [100xx], scegliendo per gli ultimi 3 bit: 1 [010], 2 [100], 3 [110] mantenendo l'ultimo bit a 0);
    - *Expedited Forwarding*: (101 000);
  - ECN: *Explicit Congestion Notification* (ultimi 2 bit nel DS);
    - ECT: *ECN-Capable Transport*;
    - CE: *Congestion Encountered*.

- **Protocolli per livello 3**:
  - AS: *Autonomous System*;
  - IGP: *Interior Gateway Protocol*;
  - RIP: *Routing Information Protocol*;
  - OSPF: *Open Shortest Path First*:
    - LSA: *Link State Advertisement*
  - EGP: *Exterior Gateway Protocol*;
  - BGP: *Border Gateway Protocol*;
  - ICMP: *Internet Control Message Protocol*;
  - IGMP: *Internet Group Management Protocol*;
  - TTL: *Time To Live*;
  - NTP: *Network Time Protocol*;
  - DHCP: *Dynamic Host Configuration Protocol*;
  - HSRP: *Hot Standby Routing Protocol*;
  - VRRP: *Virtual Router Redudancy Protocol*;
  - GLBP: *Gateway Load Balancing Protocol*:
    - AVG: *Active Virtual Gateway*;
    - AVF: *Active Virtual Forwarders*;
  - ECMP: *Equal-Cost Multipath*.
- **Multicast**
  - RPF: *Reverse Path Forwarding*;
  - DVMRP: *Distance-Vecotr Multicast Routing Protocol* (source-based with RPF and pruning);
  - PIM: *Protocol-Independent Multicast*:
    - PIM-DM: *PIM-Dense Mode*;
    - PIM-SM: *PIM-Sparse Mode*.
- **Sigle livello 4**
  - UDP: *User Datagram Protocol*;
  - TCP: *Transmission Control Protocol*:
    - Cwnd: *congestion window*;
    - Rwnd: *receive window*;
    - Ssthresh: *slow-start threshold*;
    - MSS: *Maximum Segment Size* (max dimensione del payload TCP);
    - AIMD: *Additive Increase, Multiplicative Decrease*;
    - ECE: *ECN-Echo* flag;
    - CWR: *Congestion Window Reduced*.
- **IPv6**

  - EUI: *Extended Unique Identifier* (per gli indirizzi MAC EUI-64);
  - DAD: *Duplicate Address Detection*;
  - ESP: *Encapsulating Security Payload* (header aggiuntivo IPv6);
  - RA: *Router Advertisements*.

- **Multimedia networking**
  - RSTP: *Real-Time Streaming Protocol*;
  - RTP: *Real-time Transport Protocol*;
  - RTCP: *RTP Control Protocol*;
  - SIP: *Session Initiation Protocol*;
  - PSTN: *Public Switched Telephone Network*;
  - VoIP: *Voice over IP*.

- **Quality of Service**
  - PCP: *Priority Code Point* (nella terminologia 802.1Q/p, nella 802.1p è User Priority);
  - DEI: *Drop Eligible Indicator*;
  - WFQ: *Weighted Fair Queuing*;
  - IntServ: *Integrated Services*:
    - RSVP: *Resource Reservation Protocol*:
      - Tspec: *Traffic specification*;
      - Rspec: *Resource specification*.
    - ISA: *Integrated Services Architecture*.
  - DiffServ: *Differentiated Services*:
    - PHB: *Per-Hop Behaviour*;
    - CoS: *Classes of Service*;
    - TCB: *Traffic Conditioning Block*;
    - LLQ: *Low Latency Queuing*;
    - RED: *Random Early Detection*;
    - WRED: *Weighted RED*;
    - MQC: *Modular QoS CLI* (Command Line Interface).
  - SLA: *Service Level Agreement*;
  - MPLS: *Multiprotocol Label Switching*:
    - MPLS-TE: *MPLS-Traffic Engineering*;
    - LSR: *Label Switch Router*;
    - LSP: *Label Switched Path*;
    - FEC: *Forwarding Equivalence Class*;
    - LDP: *Label Distribution Protocol* (senza MPLS-TE);
    - RSVP-TE: *Resource Reservation Protocol-Traffic Engineering* (con MPLS-TE);
    - EXP: *Experimental* (i bit di MPLS per la QoS);
    - CE: *Customer Edge*;
    - C: *Customer*;

    * PE:*Provider Edge*;
    * P: *Provider*;
    * VRF: *Virtual Routing/Forwarding*

- **Wireless**
  - MANET: *Mobile Ad Hoc Network*;
  - VANET: *Vehicular Ad Hoc Network*:
    * CBF: *Contention-Based Forwarding*;
    * WT: *Waiting Time*.
  - Tipi di reti mobile:
    * GSM: *Global System for Mobile Communication* (in origine Groupe Spécial Mobile);
    * UMTS: *Universal Mobile Telecommunication System*;
    * HSDPA: *High-Speed Downlink Packet Access*;
    * HSPA: *High-Speed Packet Access*;
    * LTE: *Long-Term Evolution*.
  - SNR: *Signal-to-Noise Ratio*;
  - BER: *Bit Error Rate*:
    * PSK: *Phase Shift Keying*;
    * QPSK: *QPSK*;
    * QAM: *Quadrature Amplitude Modulation*.
  - BSS: *Basic Service Set*:
    * BSSID: *BSS Identifier* (MAC address dell'AP);
  - STA: *Station*;
  - AP: *Access Point*;
  - DS: *Distribution System*;
  - ESS: *Extended Service Set*:
    * SSID: *Service Set Identifier*;
  - IBSS: *Independent Basic Service Set* (BSS indipendenti sulla stessa frequenza nel caso di reti Ad Hoc);
  - EIRP: *Equivalent Isotropic Radiated Power*;
  - WPA: *WiFi Protected Access*;
  - WEP: *Wired Equivalent Privacy*;
  - AES: *Advanced Encryption Standard*;
  - EAP: *Extensible Authentication Protocol*;
  - RADIUS: *Remote Authentication Dial-In User Service*;
  - DCF: *Distributed Coordination Function*;
  - PCF: *Point Coordination Function*;
  - IFS: *Inter-Frame Space*:
    * SIFS: *Short IFS*;

        * PIFS: *PCF IFS*;
        * DIFS: *DCF IFS*;
        * EIFS: *Extended IFS*.
    - CCA: *Clear Channel Assessment* (segnale per capire quando un canale è libero);
    - CW: *Contention Window*;
    - CRC: *Cyclic Redudancy Check*;
    - RTS: *Request To Send*;
    - CTS: *Clear To Send*;
    - NAV: *Net Allocation Vector*;
    - MIPv6 *Mobile IPv6*:
        * MN: *Mobile Node*;
        * CN: *Correspondent Node*;
        * HA: *Home Agent*;
        * CoA: *Care-of-Address*;
        * BU: *Binding Update*;
    - WLC: *WLAN Controller*;
    - CAPWAP: *Control and Provisioning of Wireless Access Points*

- **Sicurezza**
    - VPN: *Virtual Private Network*;
    - IPsec: *IP security*:
        * AH: *Authentication Header*;
        * ESP: *Encapsulation Security Payload*;
        * SA: *Security Association*:
            · SPI: *Security Parameter Index* (identificatore della SA)
        * SAD: *Security Association Database*;
        * SPD: *Security Policy Database*;
        * IPsec IKE: *Internet Key Exchange*.
    - SSL: *Secure Sockets Layer*;
    - TLS: *Transport Layer Security*;
    - IPS: *Intrusion Prevention System*;
    - IDS: *Intrusion Detection System*:
        * HIDS: *Host-based IDS*;
        * NIDS: *Network IDS*.

Thanksgiving

Si ringrazia:

- il prof. *Giovanni Ciccarese* per il corso di Network Technologies tenutosi in AA 2016/2017 del corso di laurea in Ingegneria Informatica at UNISALENTO, ed anche per disponibilità e chiarimenti. Contatto: *gianni.ciccarese@unisalento.it*;
- me stesso, *Marco Chiarelli*, studente del II anno di Ingegneria Informatica at UNISALENTO. Contatti: {*marco_chiarelli@yahoo.it*, *marcochiarelli.nextgenlab@gmail.com*, *marco.chiarelli @studenti.unisalento.it*};
- *Andrea Camisa*, collega universitario che ha fornito le sigle NT in appendice. Contatto: *andrea.camisa@studenti.unisalento.it*;
- La mia squadra di studio: {*Gabriele Accarino*, *Matteo Settembrini*, *Paolo Panarese*, *Emanuele Costa Cesari*, *Dino Sbarro*};
- Google, ovviamente.

Credits

/marcochiarelli

Bibliografia

[1] J. Kurose, K.W. Ross, *Computer Networking. A Top-Down Approach, sixth edition*

[2] B.A. Forouzan *Data Communication and Networking, fifth edition*

[3] P. Oppenheimer *Top Down Networking Design, third edition*