Sistemi IV

(Corso di Informatica, 5 anno)

Linguaggi, Strumenti e modelli di Simulazione

Luciano Bononi

bononi@cs.unibo.it

http://www.cs.unibo.it/~bononi

Ricevimento: Lun-Ven 9-18 :-)

presso ufficio dottorandi, Mura Anteo Zamboni 7, Bologna

Notizie varie

- E’ nato il newsgroup! unibo.cs.sistemi4
- FAQs in rete: www.cs.unibo.it/~bononi/Sistemi4/
  - gnuplot attraverso file di comandi
  - creazione profilo utente con Modline (!)
- Lucidi lezioni I, II, III
- progetto: 3 gruppi hanno consegnato...
- domande?

Reti di Code (1)

- modelli a reti di code (Queueing Networks, QN)
  - la teoria delle reti di code è una delle principali tecniche analitiche usate nella valutazione di prestazioni
  - il modello basato su reti di code si adatta particolarmente a sistemi basati su risorse condivise e contesa: code di attesa, tempi di servizio, processi di interarrivo, ecc.
  - esiste un'analogia tra reti di code e algoritmi (?!?)
  - molti strumenti sono basati su QN (ResQ, Modline…)

Reti di Code (2)

- modelli a reti di code (Queueing Networks, QN)
  - terminologia e notazione (orientata a Modline)
  - risultati intuitivi su modelli a coda singola
  - analisi di modelli a code multiple:
    - analisi operazionale
    - analisi dei valori medi
    - metodi di convoluzione
  - modelli gerarchici (!)

Reti di Code (3)

- modelli a reti di code (Notazione)
  - popolazione di customers (jobs)
    - 1.Processo di arrivo:
      - dati arrivi ai tempi t1, t2, t3, t4… i **tempi di interarrivo** (IT) sono definiti come it1=t2-t1, it2=t3-t2…
      - Assunzione: IT formano una sequenza di v.c. indipendenti e identicamente distribuite (iid).
      - Tale assunzione deve essere RESA valida con opportuni accorgimenti nel procedimento di simulazione (prove ripetute, eliminazione fasi transienti, generatori pseudocauali std, ecc.)
      - es. di processo di interarrivo molto comune: arrivi Poissoniani.
        - » It sono iid e distribuiti esponenzialmente (memoryless prop.)
        - » si usano anche it Erlangiani o iper-esponenziali
      - In generale, molti risultati della teoria delle reti di code valgono per distribuzioni Generali (senza assunzioni richieste)

Reti di Code (4)

- modelli a reti di code (Notazione)
  - insieme di risorse e di code (attive e passive)
    - 2.Distribuzione dei tempi di servizio
      - i tempi di servizio sono v.c. iid (di solito esponenziali, ma valgono le stesse considerazioni per i tempi di interarrivo)
    - 3.numero di serventi
      - data una coda (di attesa per un servizio) possono aversi uno o più serventi (identici). Ogni gruppo di server identici forma una coda di attesa.
    - 4.Capacità del sistema
      - il massimo numero di utenti che possono essere presenti nel sistema (in coda o sotto servizio): limitato o infinito.
    - 5.Population size:
      - il max numero potenziale di utenti che possono accedere al sistema (nel corso del tempo)

Reti di Code (5)

- modelli a reti di code (Notazione)
  - insieme di risorse e di code (attive e passive)
    - 6.Disciplina di servizio: l'ordine con cui gli utenti sono serviti
      - FCFS (first come firse served)
      - LCFS (last come first served)
      - *-PR (qls. Disciplina - con Preemption and Resume)
      - RR (Round Robin con quantum di dimensione fissata)
      - PS (Processor Sharing) equivale a RR con quantum -> zero.
      - IS (Infinite Server)
      - altre discipline dinamiche (rispetto al t di servizio richiesto)
        - » SPT (Shortest Processing Time First)
        - » SRPT (Shortest Remaining Process Time First)
        - » ecc.

Reti di Code (6)

- Notazione di Kendall:
  - specifica i 6 parametri della QN visti prima
  - es. A/S/m/B/K/SD (B/K/SD = inf/inf/FCFS default)
    - A: distribuzione dei tempi di interarrivo
    - S: distribuzione del tempo di servizio
    - m: numero dei server
    - B: dimensione dell'area di buffer (capacità del sistema)
    - K: dimensione della popolazione
    - SD: disciplina di servizio
  - Simboli delle distribuzioni:
    - M (esponenziale), Ek (Erlang con parametro k), Hk (iper-esponenz. con param. k), D (deterministica), G (generale)

Reti di Code (7)

- Parametri e Variabili usate per analisi di QN
  - Tau: tempo di interarrivo (v.c.)
  - Lambda: 1/E[Tau] = mean arrival rate
    - può essere funzione dello stato del sistema
  - s: tempo di servizio di un utente (job) (v.c.)
  - mu: 1/E[s]=mean service rate di un server
    - service rate totale di m server uguali = m*mu
  - n: numero di jobs in coda (nq) o sotto servizio (ns) = (queue length) (v.c.)
  - r: tempo di risposta del sistema (= w+s) (v.c.)
  - w: waiting time= t di inizio servizio - t di arrivo (v.c.)

Reti di Code (8)

- Condizione di stabilità:
  - se il numero di jobs nel sistema cresce all'infinito il sistema si dice instabile
  - condizione di stabilità: Lambda < m*mu
  - il ritmo degli arrivi deve essere minore del ritmo con cui i job vedono soddisfare le loro richieste
  - N.B. tale condizione vale per sistemi a popolazione e buffer infiniti (altrimenti il sistema è sempre stabile)
- Numero di job nel sistema
  - n = nq + ns : il numero di job nel sistema è dato dal numero di job in coda + il numero di job sotto servizio. Siccome n, nq e ns sono v.c. la stessa relazione vale per medie e varianze. (purchè il service rate sia indipendente dalla queue size)

Reti di Code (9)

- Numero di job (t) con buffer finiti
  - se non c'è perdita di job a causa di buffer limitati
    - E[n] = arrival rate * mean response time = lambda*E[r]
    - E[nq] = arrival rate x mean waiting time = lambda*E[w]
    - Queste due equazioni definiscono la LEGGE di LITTLE
- r = w+s dove r,w,s sono v.c. e quindi
  - E[r]=E[w]+E[s]
  - se il service rate non dipende dal nq vale Cov(w,s)=0 e quindi Var[r]=Var[w]+Var[s]

Reti di Code (10)

- Legge di Little
  - E’ il teorema più usato in teoria delle QN
    - visione BlackBox del sistema
    - E[n]=arrival rate x mean response time
    - Assunzione: il numero di job che entrano nel sistema è uguale al numero di job che escono dopo il servizio
      - no creazione di Job interni (fork, create, split)
      - no perdita (drop) di job per carenza di buffer
    - può essere applicata a porzioni del sistema che soddisfano le richieste

Reti di Code (11)

- Tipi di processi stocastici
  - Processo Stocastico: definisce una v.c. i cui valori sono associati a una dipendenza dal tempo
  - es. n(t)=numero job nel sistema al tempo t
  - eseguire k simulazioni del sistema a partire da uno stato iniziale s0 e da un tempo iniziale t0 significa definire una sequenza di istanti t1, t2, t3… nei quali lo stato s1, s2, s3 include valori n(t1), n(t2)…
  - La v.c. n(t) definisce un processo stocastico per ognuna delle k simulazioni

Reti di Code (12)

- Tipi di processi stocastici
  - I processi stocastici sono utili nella teoria delle QN
    - a) **Processi a tempo continuo/discreto**
    - se gli stati del sistema possono assumere un insieme finito di valori --> processo stocastico a stato discreto (stochastic chain). Es. n(t)
    - se gli stati del sistema possono assumere un insieme infinito di valori --> processo stocastico a stato continuo (es. w(t))

Reti di Code (13)

- Tipi di processi stocastici
  - b) **Processi Markoviani**: sono processi stocastici il cui valore al tempo (t+1) dipende solo dal valore al tempo t.
  - “semplici” da analizzare: non devo conoscere la storia
  - Processo Markoviano a stato discreto = catena di Markov
  - N.B. per prevedere il futuro di un processo Markoviano a tempo continuo basta conoscere lo stato attuale, e non serve a nulla sapere da quanto tempo siamo nello stato attuale. Ciò è possibile solo se lo stato ha la memoryless property (v.c. con distribuzione esponenziale). Questo limita l’applicabilità delle catene di Markov.
  - Reti di Code M/M/m possono essere descritte con processi markoviani: r è un processo markoviano e nq è una catena di Markov.

Reti di Code (14)

- Tipi di processi stocastici
  - c) **Processi Birth and Death**: sono processi markoviani (tempo continuo) nei quali lo spazio degli stati è discreto, e nei quali da ogni stato è possibile passare a stati “adiacenti”
  - posso rappresentare gli stati come interi ...(n-1) <-> (n) <-> (n+1)…
  - es. nq(t) con un solo server e interarrivi singoli (no bulk)
  - un arrivo di un nuovo job -> (n -> n+1)
  - un servizio completato da un job -> (n -> n-1)

Reti di Code (15)

- Tipi di processi stocastici
  - d) **Processi Poissoniani**: se ta sono iid e con distribuzione esponenziale, allora il numero di arrivi in (t, t+x) è v.c. con distribuzione poissoniana -> il processo di arrivi si dice poissoniano o sequenza poissoniana.
  - Gli arrivi sono memoryless dato che gli interarrivi sono esponenziali
  - fondere k sequenze poissoniane con rate lambda(i) risulta in una sequenza poissoniana con rate medio lambda=sommatoria(i=1,k) lambda(i)
  - suddividere una sequenza poissoniana in k seq. tc pi=prob. sequenza i -> ogni sottosequenza è poisson con rate pi*lambda

Reti di Code (16)

- Tipi di processi stocastici
  - d) Processi Poissoniani: (continua)
  - se l’arrivo a un server è poisson(Lambda) e il tempo di servizio è esponenziale con media 1/mu allora le partenze dal server sono ancora poissoniane purchè lamba<mu (il rate di arrivo è minore del rate di servizio)
  - se l’arrivo a una coda su m server è poisson(Lambda) e il tempo di servizio di ogni server è esponenziale con media 1/mu(i) allora le partenze dal server sono ancora poissoniane(Lamnda) purchè lamba<sommatoria(mu(i)) (il rate di arrivo è minore del rate aggregato di servizio)
- set: Markov procs > Birth/death > Poisson

Reti di Code (17)

- Analisi di code singole
  - birth/death processes (risultati usati per M/M/1 522)
    - n(t) = k -> n(t+1) = k-1 se death, oppure k+1 se birth
    - occorre derivare un'espressione per descrivere la probabilità che il sistema si trovi in uno stato t.c. n(t)=k, per ogni valore possibile di k
    - siano esponenziali sia interarrivi che servizi
    - Teorema: la probabilità stazionaria pn = ((LAMBDA0*…*LAMBDAn-1)/(MU1*…*MUn))*p0
    - siccome SUMpn=1, allora p0=1/(1+Sum[n=1..inf](Prod[j=0..n-1] (LAMBDAj/MUj+1))
    - quindi posso trovare tutte le p stazionarie dati Lambda, Mu