Λειτουργική ανάλυση

Γιάννης Γαροφαλάκης

Εισαγωγή (1)

- *Λειτουργικοί νόμοι* : Απλές σχέσεις που δεν απαιτούν κατανομή χρόνων μεταξύ αφίξεων ή εξυπηρέτησης [Buzen 1976, Denning, Buzen 1978]
- Λειτουργική: *απευθείας μετρήσιμο*.

Υποθέσεις :

1. *Μετρήσιμο*: **Job flow balance** ↔ Εργοδικότητα

   Δηλαδή, σε συγκεκριμένο χρόνο T :

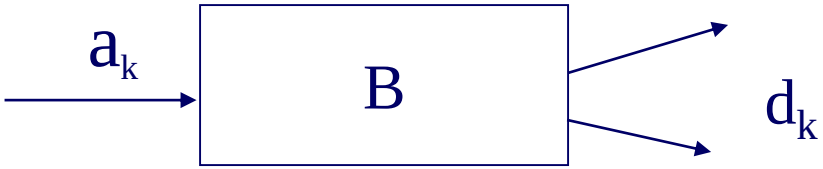
   αριθμός αφίξεων = αριθμός αναχωρήσεων.

2. *Μη μετρήσιμο* : **Ανεξαρτησία**. Η ακολουθία χρόνων εξυπηρέτησης δεν μπορεί να εντοπιστεί με μετρήσεις αν είναι ανεξάρτητες Τ.Μ. ή όχι.

Εισαγωγή (2)

- *Λειτουργικές ποσότητες* : ποσότητες που μπορούν να μετρηθούν κατά τη διάρκεια πεπερασμένης περιόδου παρατήρησης

  Παράδειγμα λειτουργικού νόμου : Νόμος του Little: $\overline{N} = \lambda T$



a_k : "βλέπει" $\overline{N}$ στο Β.

d_k : "αφήνει" $\overline{N}$ στο Β.

Άρα, οι $\overline{N}$ που βλέπει φεύγοντας, ήρθαν όλοι, όσο ο πελάτης βρισκόταν μέσα (T). Κατά την παρουσία του ήρθαν λT κατά μέσο όρο.

Άρα, $\overline{N} = \lambda T$. (*Αρκεί να ισχύει job flow balance…*)

Λειτουργικές ακολουθίες συμπεριφοράς και ιδιότητες vs. Στοχαστικές διαδικασίες (Ι)

- Ίδια αποτελέσματα με τις στοχαστικές διαδικασίες αλλά περισσότερο «διαισθητικά» και εφαρμόσιμα σε μη – στοχαστικά συστήματα.
- 3 ιδιότητες που ενδιαφέρουν:

I. Ομογενής συμπεριφορά.

- Αφίξεις
- Εξυπηρετήσεις
- Δρομολόγηση

} ανεξάρτητα από αριθμό πελατών στο σύστημα
(Εξαίρεση: $\mu = 0$ όταν δεν υπάρχουν, ή $\lambda = 0$ όταν N στο σύστημα)

- Flow – Balance
- Συμπεριφορά ενός βήματος : Μόνο ένα γεγονός τη φορά.

Λειτουργικές ακολουθίες συμπεριφοράς και ιδιότητες vs. Στοχαστικές διαδικασίες (2)

Σημαντικό : Οι λειτουργικές ιδιότητες ορίζονται για μια συγκεκριμένη ακολουθία συμπεριφοράς :

«Μέσο» = στατιστικό μέσο.

«Πιθανότητα» = σχετική συχνότητα.

Βασικές λειτουργικές σχέσεις (Ι)

- Χρόνος Παρατήρησης **T**.
- Μπορούμε να μετρήσουμε τις λειτουργικές ποσότητες (basic quantities):
  - A_i : Αριθμός αφίξεων κατά το T.
  - C_i : Αριθμός Εξυπηρετήσεων (completions) κατά το T.
  - B_i : Busy time κατά το T ($B_i \leq T$)

Βασικές λειτουργικές σχέσεις (2)

Επιπλέον, μπορούμε να πάρουμε και τις εξής λειτουργικές ποσότητες (derived quantities – performance measures) :

- Μέσος Ρυθμός Αφίξεων: $\lambda_i = \frac{A_i}{T}$
- Throughput: $X_i = \frac{C_i}{T}$
- Utilization: $U_i = \frac{B_i}{T}$
- Μέσος Χρόνος Εξυπηρέτησης: $S_i = \frac{B_i}{C_i}$

- Οι λειτουργικές ποσότητες είναι μεταβλητές που μπορεί να αλλάζουν από τη μία περίοδο παρατήρησης στην επόμενη. Όσες παραμένουν σταθερές, ονομάζονται *λειτουργικοί νόμοι*.

Νόμος Χρησιμοποίησης

- Έστω C_i ο αριθμός εξυπηρετήσεων, B_i το busy time του device κατά μία περίοδο T.
- Νόμος χρησιμοποίησης :

$$U_i = \frac{B_i}{T} = \frac{C_i}{T} \times \frac{B_i}{C_i} = X_i S_i$$

ή

$$X_i = \frac{C_i}{T} = \frac{C_i}{B_i} \times \frac{B_i}{T} = \frac{1}{S_i} U_i = \frac{U_i}{S_i}$$

ή

$$S_i = \frac{B_i}{C_i} = \frac{B_i}{T} \times \frac{T}{C_i} = U_i \frac{1}{X_i} = \frac{U_i}{X_i}$$

Νόμος Εξαναγκασμένης Ροής (Ι)

- Συνδέει το throughput του συστήματος με το throughput ενός device.
- Σε ένα ανοιχτό δίκτυο το system throughput είναι ο αριθμός των jobs που φεύγουν από το σύστημα ανά μονάδα χρόνου.
- Σε ένα κλειστό δίκτυο, καμία job δεν μπορεί να το εγκαταλείψει. Συνεπώς, ορίζεται ένα link ώστε ότι αποχωρεί από το δίκτυο, ταυτόχρονα, να εισέρχεται. Όσες διέρχονται από εκεί ορίζουν το system throughput.

Νόμος Εξαναγκασμένης Ροής (2)

- Αν το T είναι τέτοιο ώστε $A_i = C_i$, τότε το device ικανοποιεί την υπόθεση Job Flow Balance.
- Έστω ότι κάθε job κάνει V_i αιτήσεις για το i-στο device. Ισχύει:

$$C_i = C_0 V_i \quad \text{ή} \quad V_i = \frac{C_i}{C_0}$$

όπου C_0, ο αριθμός των jobs που διατρέχουν το εξωτερικό link.

- Το V_i ονομάζεται *visit ratio* ή *relative throughput*, (λόγος επισκέψεων στο i device προς επισκέψεις στο out link).

Νόμος Εξαναγκασμένης Ροής (3)

- System Throughput : $X = \frac{C_0}{T}$ και

$$X_i = \frac{C_i}{T} = \frac{C_i}{C_0} \times \frac{C_0}{T} \quad \Rightarrow$$

$$X_i = X \cdot V_i \quad , \textit{ Νόμος Εξαναγκασμένης Ροής.}$$

- Ο Νόμος Εξαναγκασμένης Ροής ισχύει όποτε ισχύει το Job Flow Balance.

Νόμος Εξαναγκασμένης Ροής (4)

- Συνδυάζοντας το Νόμο Εξαναγκασμένης Ροής και το Νόμο Χρησιμοποίησης, προκύπτει:

$$\left.\begin{aligned} X_i &= X \cdot V_i \\ U_i &= X_i \cdot S_i \end{aligned}\right\} \quad U_i = X \cdot V_i \cdot S_i \;\Rightarrow\; U_i = X \cdot D_i$$

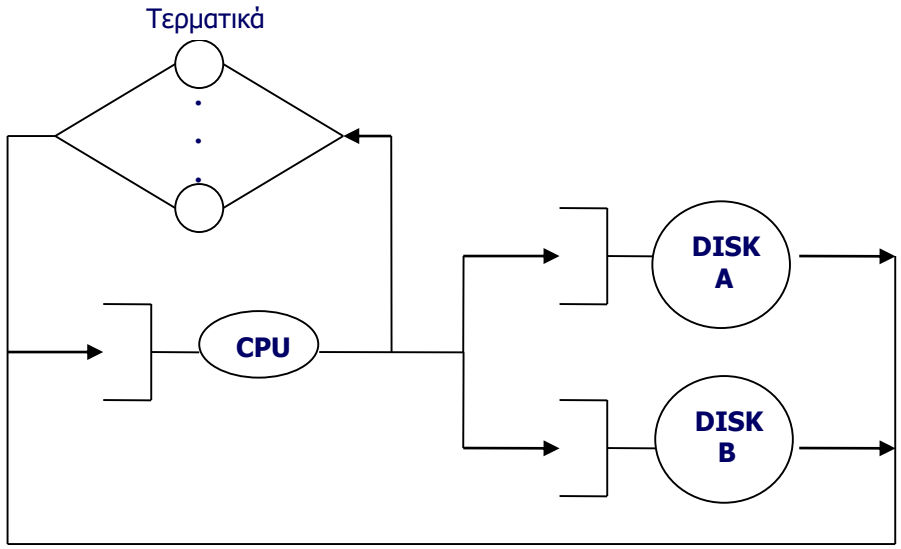
όπου, $D_i = V_i S_i$ είναι η συνολική απαίτηση service στο device-*i* για όλες τις επισκέψεις μίας job.

- Δηλαδή, το device με το μεγαλύτερο service demand D_i έχει το μεγαλύτερο Utilization και είναι bottleneck device.

Νόμος Εξαναγκασμένης Ροής (5.1)

- Παράδειγμα 1:

  Central server model ενός timesharing συστήματος.



Νόμος Εξαναγκασμένης Ροής (5.2)

Μετρώντας τα log data προκύπτουν τα εξής:

1. Κάθε πρόγραμμα απαιτεί 5 sec CPU time και κάνει 80 I/O requests στο δίσκο A και 100 στο δίσκο B.
2. Ο μέσος χρόνος σκέψης των χρηστών είναι 18 sec.
3. Ο δίσκος A απαιτεί 50 msec για μία I/O request ενώ ο δίσκος B, 30 msec.
4. Με 17 active terminals το throughput του A μετρήθηκε ίσο με 15,7 I/O requests/sec.

Θέλουμε να βρούμε το system throughput και το utilization των devices.

Νόμος Εξαναγκασμένης Ροής (5.3)

Δηλαδή:

$$D_{CPU} = 5 \text{ sec}, V_A = 80, V_B = 100, \text{ Z=18 sec}, S_A = 0,05 \text{ sec},$$

$$S_B = 0,03 \text{ sec}, N = 17, X_A = 15,7 \text{ jobs/sec}$$

$$V_{CPU} = V_A + V_B + 1 = 181$$

Επειδή οι εργασίες πρέπει να επισκεφθούν την CPU πριν τους δίσκους ή τα τερματικά.

Το πρώτο βήμα στην λειτουργική ανάλυση, γενικά, είναι να καθοριστούν τα D_i όλων των devices:

$$D_{CPU} = 5 \text{ sec},$$

$$D_A = S_A V_A = 4 \text{ sec},$$

$$D_B = S_B V_B = 3 \text{ sec}$$

Νόμος Εξαναγκασμένης Ροής (5.4)

Χρησιμοποιώντας το νόμο εξαναγκασμένης ροής, τα throughputs προκύπτουν:

$$X = \frac{X_A}{V_A} = 0{,}1963 \text{ jobs/sec},$$

$$X_{CPU} = XV_{CPU} = 35{,}48 \text{ requests/sec},$$

$$X_B = XV_B = 19{,}6 \text{ requests/sec}$$

Χρησιμοποιώντας το νόμο χρησιμοποίησης, τα utilizations προκύπτουν:

$$U_{CPU} = XD_{CPU} = 98\%,$$

$$U_A = XD_A = 78{,}4\%,$$

$$U_B = XD_B = 58{,}8\%$$

Νόμος Εξαναγκασμένης Ροής (6)

- Άλλος τρόπος περιγραφής της δρομολόγησης jobs σε ένα queueing network είναι οι *πιθανότητες μετάβασης (transition probabilities)*, p_{ij}. Είναι ισοδύναμη περιγραφή με τα V_i .

- Σε δίκτυο με Job Flow Balance:

$$C_j = \sum_{i=0}^{M} C_i \, p_{ij}$$

Το 0 επισημαίνει τις επισκέψεις στο εξωτερικό link.

Νόμος Εξαναγκασμένης Ροής (7)

- Ισχύει, $C_j = \sum_{i=0}^{M} C_i p_{ij} \Rightarrow \frac{C_j}{C_0} = \sum_{i=0}^{M} \frac{C_i}{C_0} p_{ij} \Rightarrow$

$$V_j = \sum_{i=0}^{M} V_i p_{ij} \quad , \quad V_0 = 1$$

- $V_0 = 1$, διότι κάθε επίσκεψη στο εξωτερικό link ορίζεται σαν την ολοκλήρωση της job.
- Οι παραπάνω εξισώσεις είναι γνωστές σαν *visit ratio equations*. Ισχύουν εφόσον το δίκτυο είναι συνδεδεμένο λειτουργικά : Κάθε συσκευή επισκέπτεται από κάθε job τουλάχιστον 1 φορά.

Νόμος του Little

- Είναι λειτουργικός νόμος.
- Εφαρμόζουμε το Νόμο του Little για να σχετίσουμε τον αριθμό πελατών Q_i, με το χρόνο απόκρισης R_i στο device-i:

Μέσος αριθμός στο device = ρυθμός αφίξεων x μέσο χρόνο στο device.

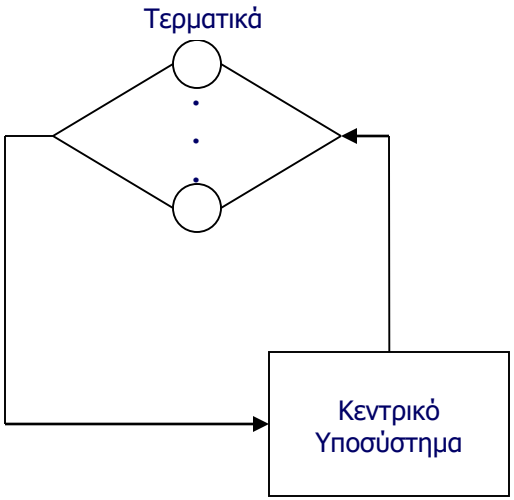
$$Q_i = \lambda_i R_i$$

Αν το job flow είναι σε ισορροπία, ο ρυθμός άφιξης είναι ίσος με το throughput ($\lambda_i = X_i$), ισχύει

$$Q_i = X_i R_i$$

Γενικός Νόμος Χρόνου Απόκρισης (Ι)

- Timesharing System



terminal υποσύστημα

κεντρικό υποσύστημα

1 user ⟶ 1 terminal.

- Ο Νόμος του Little εφαρμόζεται σε όλα τα τμήματα του συστήματος, αρκεί να υπάρχει job flow balance.
- Σε κεντρικό υποσύστημα, ισχύει : $Q = X \cdot R$

Γενικός Νόμος Χρόνου Απόκρισης (2)

$$Q = X \cdot R$$

Συνολικός αριθμός εργασιών στο υποσύστημα

Throughput συστήματος

Response time συστήματος

- Για M devices, έχουμε :

$$Q = Q_1 + Q_2 + \ldots + Q_M \xrightarrow{Q_i = X_i R_i}$$

$$XR = X_1 R_1 + X_2 R_2 + \ldots + X_M R_M \longrightarrow$$

Γενικός Νόμος Χρόνου Απόκρισης (3)

$$\longrightarrow \quad \frac{XR}{X} = \frac{X_1R_1 + X_2R_2 + \ldots + X_MR_M}{X} \quad \xrightarrow{X_i = XV_i}$$

$$R = V_1R_1 + V_2R_2 + \ldots + V_MR_M \quad \longrightarrow$$

$$R = \sum_{i=0}^{M} R_iV_i$$

- *Γενικός Νόμος Χρόνου Απόκρισης* : $R = \sum_{i=0}^{M} R_iV_i$

Γενικός Νόμος Χρόνου Απόκρισης (4)

- Ισχύει και σε περιπτώσεις που δεν ισχύει το Job Flow Balance.
- Δηλαδή, ο συνολικός χρόνος που καταναλώνει μία job σε ένα server είναι το γινόμενο του χρόνου ανά επίσκεψη, επί τον αριθμό των επισκέψεων, στο server.
- Ο συνολικός χρόνος συστήματος είναι το άθροισμα των συνολικών χρόνων στους διάφορους servers.

Interactive Response Time Law

- Έστω Z ο χρόνος think time στο terminal.
- Συνολικός χρονικός κύκλος αιτήσεων: $R+Z$.
- Κάθε χρήστης δημιουργεί περίπου $\frac{T}{R+Z}$ αιτήσεις στο χρονικό διάστημα T.
- Αν έχουμε N χρήστες (τερματικά):
  - System throughput: $X = \frac{\textit{συνολικός αριθμός αιτήσεων}}{\textit{χρόνος}} =$

$$= \frac{N\frac{T}{R+Z}}{T} = \frac{N}{R+Z} \Rightarrow R = \frac{N}{X} - Z$$

- Interactive Response Time Law: $R = \frac{N}{X} - Z$

Bottleneck analysis (1)

- Νόμος εξαναγκασμένης ροής: $X_i = X \cdot V_i$
- Νόμος Χρησιμοποίησης: $U_i = X \cdot D_i \Rightarrow U_i \propto D_i$
- Το device με το υψηλότερο φορτίο αιτήσεων για service D_i, έχει το μεγαλύτερο utilization και είναι το bottleneck device. Η βελτίωση του θα βελτιώσει το σύστημα.

➢ ***Βήμα 1***: Προσδιορισμός του bottleneck device για μελέτη βελτίωσης απόδοσης.

Έστω ότι βρίσκουμε ότι το device b είναι το bottleneck.

Δηλαδή, $D_b = D_{\max} = \max(D_1, D_2, \ldots, D_M)$

Bottleneck analysis (2)

Τότε, το throughput και οι χρόνοι απόκρισης του συστήματος περιορίζονται από τις παρακάτω σχέσεις:

$$X(N) \leq \min\left(\frac{1}{D_{\max}}, \frac{N}{D+Z}\right)$$

$$R(N) \geq \max(D, ND_{\max} - Z)$$

Το $D = \sum D_i$ είναι το άθροισμα των service devices (εκτός των terminals).

Bottleneck analysis (2.1)

- Απόδειξη:

Παρατηρήσεις :

1. Το utilization οποιουδήποτε device δεν μπορεί να είναι μεγαλύτερο από 1. Με αυτόν τον τρόπο τίθεται ένα όριο για το μέγιστο δυνατό throughput. (1)
2. Το response time του συστήματος με N users δεν μπορεί να είναι μικρότερο από αυτό του συστήματος με 1 user. (2)
3. Η σχέση $R=\frac{N}{X}-Z$ μπορεί να χρησιμοποιηθεί για τη μετατροπή του ορίου του throughput σε όριο για το response time και αντίστροφα.

Bottleneck analysis (2.2)

Βασισμένοι στις προηγούμενες παρατηρήσεις, έχουμε, για το bottleneck device:

$$U_b = XD_{\max} \xrightarrow{(1)} \quad (U_b \text{ είναι το πιο κοντινό } U \text{ στο } 1\ldots)$$

$$U_b \le 1 \Rightarrow XD_{\max} \le 1 \Rightarrow X \le \frac{1}{D_{\max}} \qquad (3)$$

Με **1** χρήστη στο σύστημα, δεν υπάρχει αναμονή στην ουρά:

$$R(1) = D_1 + D_2 + \ldots D_M = D$$

Με περισσότερους χρήστες, μπορεί να υπάρχει αναμονή:

$$R(N) \ge D \qquad (4)$$

Bottleneck analysis (2.3)

Ισχύει από την (1):

$$R(N) = \frac{N}{X(N)} - Z \geq ND_{\max} - Z \xrightarrow{(2)}$$

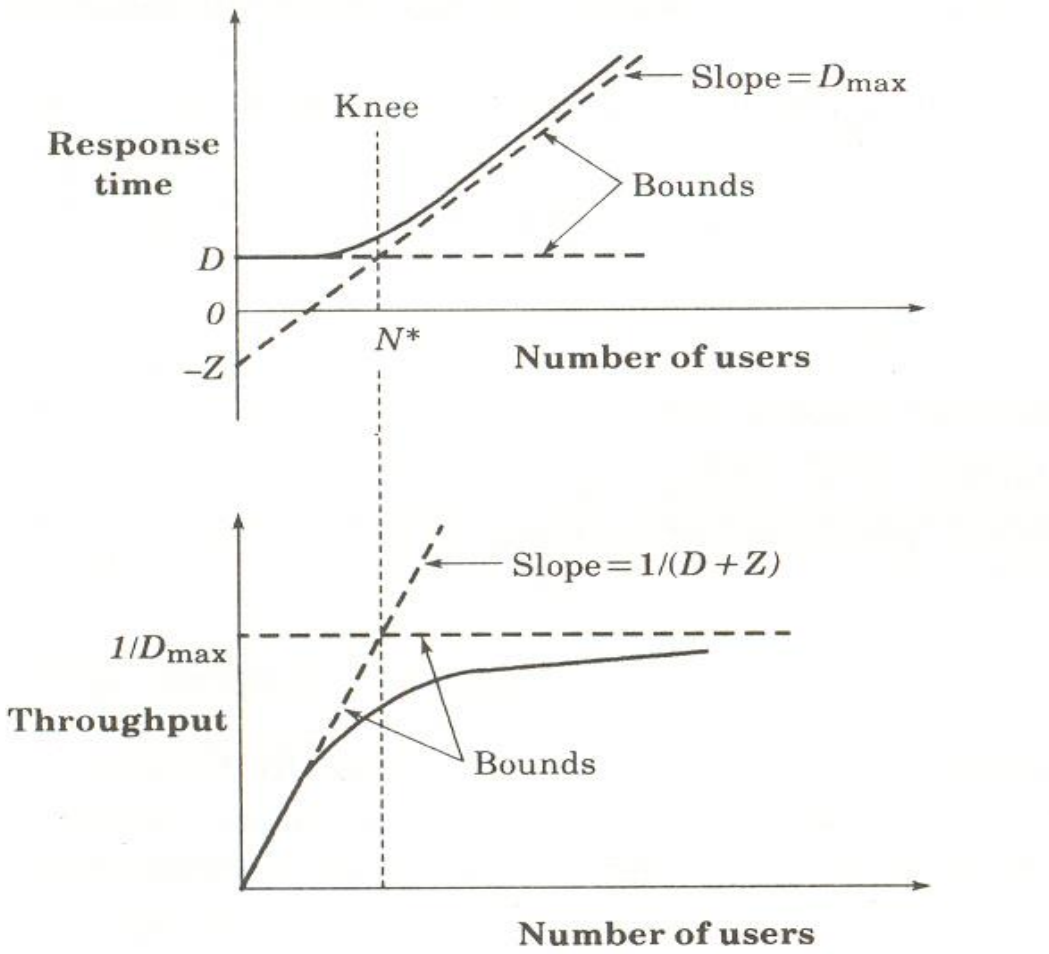
$$X(N) = \frac{N}{R(N)+Z} \leq \frac{N}{D+Z}$$

Συνδυάζοντας τα παραπάνω όρια με τις (3), (4) προκύπτουν τα όρια για το $X(N)$ και για το $R(N)$:

$$X(N) \leq \min\left(\frac{1}{D_{\max}}, \frac{N}{D+Z}\right)$$

$$R(N) \geq \max(D, ND_{\max} - Z)$$

Bottleneck analysis (2.4)



Bottleneck analysis (2.5)

Η προηγούμενες γραφικές αναπαριστούν τα ασυμπτωτικά όρια. Παρατηρείται, ότι το knee εμφανίζεται για την ίδια τιμή του αριθμού χρηστών:

$$D = N^* D_{\max} - Z \Rightarrow N^* = \frac{D+Z}{D_{\max}}$$

Αν ο αριθμός χρηστών είναι μεγαλύτερος από N^* μπορούμε με βεβαιότητα να πούμε ότι υπάρχει αναμονή στο σύστημα.

Bottleneck analysis (2.6)

Συνέχεια παραδείγματος Ι :

Έχουμε $D_{CPU} = 5,\ D_A = 4,\ D_B = 3,\ Z = 18$.

Άρα,

$$D = D_{CPU} + D_A + D_B = 12 \Rightarrow$$

$$D_{\max} = D_{CPU} = 5$$

Τα όρια είναι:

$$X(N) \le \min\left\{\frac{N}{30}, \frac{1}{5}\right\}$$

$$R(N) \ge \max\{12, 5N - 18\}$$

Το knee εμφανίζεται για N^*: $12 = 5N^* - 18 \Rightarrow N^* = 6$

Δηλαδή, αν υπάρχουν περισσότεροι από 6 χρήστες σίγουρα θα υπάρχει αναμονή κάπου.

Bottleneck analysis (2.7)

Συνέχεια παραδείγματος 1 :

Πόσα τερματικά μπορούν να υποστηριχθούν αν το response time πρέπει να μην ξεπερνά τα 100 sec ;

Με τη χρησιμοποίηση των ασυμπτωτικών ορίων, έχουμε:

$$R(N) \geq \max\{12, 5N - 18\}$$

Το response time θα είναι μεγαλύτερο από 100 αν

$$12,5N - 18 \geq 100 \Rightarrow N \geq 23,6.$$

Συνεπώς, το σύστημα δεν μπορεί να υποστηρίξει περισσότερους από 23 χρήστες ώστε να είναι $R(N) \leq 100$

Operational Laws

- Utilization Law : $U_i = X_i S_i = X D_i$
- Forced Flow Law : $X_i = X V_i$
- Little's Law : $Q_i = X_i R_i$
- General Response Time Law : $R = \sum_{i=1}^{M} R_i V_i$
- Interactive Response Time Law : $R = \frac{N}{X} - Z$
- Asymptotic bounds :

$$R \geq \max\left\{D, N D_{\max} - Z\right\}$$

$$X \leq \min\left\{\frac{1}{D_{\max}}, \frac{N}{D+Z}\right\}$$

Mean Value Analysis

Ανάλυση ανοικτών δικτύων (Ι)

- Μοντέλα για Transaction Processing συστήματα (ρυθμός αφίξεων ανεξάρτητος από φορτίο συστήματος).
- Αφίξεις Poisson με μέσο ρυθμό λ .
- Τα service centers είναι
  - fixed capacity (single server εκθετικοί)
  - ή
  - delay centers (infinite server εκθετικοί)

Ανάλυση ανοικτών δικτύων (2)

- Σε όλα τα fixed capacity service centers:
  - Response Time: $R_i = S_i(1+Q_i)$
  - $R_i = S_i + S_i Q_i$

service

service των Q_i jobs που βλέπει στην device i

  - Δεν είναι operational law διότι υποθέτει memoryless service, κάτι που δεν είναι λειτουργικά δοκιμαζόμενο.

Ανάλυση ανοικτών δικτύων (3)

- Λόγω του job flow balance, ισχύει: $X = \lambda$

  Σε κάθε device *i*:

  - Forced flow law : $X_i = XV_i$
  - Νόμος χρησιμοποίησης: $U_i = X_i S_i = XV_i S_i = \lambda D_i$
  - Με τη χρήση του N. Little:

$$Q_i = X_i R_i = X_i S_i (1 + Q_i) = U_i (1 + Q_i) \Rightarrow$$

$$Q_i = \frac{U_i}{1 - U_i} \Rightarrow \qquad R_i = \frac{S_i}{1 - U_i}$$

Ανάλυση ανοικτών δικτύων (4)

- Σε όλα τα delay centers:
  - Ισχύει, $R_i = S_i$.
  - Και, $Q_i = X_i R_i = X V_i S_i = X D_i = U_i$.

Ανάλυση ανοικτών δικτύων - Αλγόριθμος (Ι)

Inputs:

- $\lambda = X$: Εξωτερικό arrival rate = system throughput
- M : αριθμός συσκευών (χωρίς τερματικά).
- S_i : service time per visit στο i -στό device.
- V_i : αριθμός επισκέψεων στο i -στό device.

Outputs:

- N : μέσος αριθμός εργασιών στο δίκτυο
- Q_i : μέσος αριθμός εργασιών στο i –στό device.
- R_i : response time στο i -στό device.
- R : system response time.
- U_i : χρησιμοποίηση του i -στού device.

Ανάλυση ανοικτών δικτύων - Αλγόριθμος (II)

Λύση:

- $D_i = S_i \cdot V_i$
- $U_i = X \cdot D_i$
- $X_i = X \cdot V_i$
- $R_i = \begin{cases} S_i/(1-U_i) & Fixed\ capacity\ centers \\ S_i & Delay\ centers \end{cases}$
- $Q_i = \begin{cases} U_i/(1-U_i) & Fixed\ capacity\ centers \\ U_i & Delay\ centers \end{cases}$
- $R = \sum_{i=1}^{M} R_i \cdot V_i$
- $N = \sum_{i=1}^{M} Q_i$

MVA - Ανάλυση Κλειστών Δικτύων (1)

- Γενικά εφαρμόζεται για πολλά είδη service κατανομών και τρόπων εξυπηρέτησης.
- Θεωρούμε fixed capacity service centers.
  - Αν έχουμε κλειστό δίκτυο με N εργασίες, οι Reiser και Lavenberg (1980), έδειξαν:

$$R_i(N) = S_i[1 + Q_i(N-1)]$$

όπου $Q_i(N-1)$, ο μέσος αριθμός εργασιών στο i-στο device, όταν είναι $N-1$ jobs στο δίκτυο.

MVA - Ανάλυση Κλειστών Δικτύων (2)

- Δηλαδή, όταν προσθέσουμε ένα πελάτη στο δίκτυο, ενώ υπάρχουν ήδη $N-1$ πελάτες, μόλις φθάσει στο i-στο device, θα βρει $Q_i(N-1)$ jobs μέσα.

Θα περιμένει χρόνο $S_i Q_i(N-1)$ μέχρι να εξυπηρετηθεί και θα ξοδέψει χρόνο S_i κατά την εξυπηρέτηση του.

MVA - Ανάλυση Κλειστών Δικτύων (3)

- System Response Time (Γενικός Ν. Χρόνου Απόκρισης):

$$R(N) = \sum_{i=1}^{M} V_i R_i$$

- System Throughput (Interactive Response Time Law):

$$X(N) = \frac{N}{R(N) + Z}$$

- Device throughputs βάσει των jobs:

$$X_i(N) = X(N) V_i$$

MVA - Ανάλυση Κλειστών Δικτύων (4)

- Μήκη ουρών με τη χρήση του N. Little:

$$Q_i(N) = X_i(N) R_i(N) = X(N) V_i R_i(N)$$

MVA - Ανάλυση Κλειστών Δικτύων (5)

- Θεωρούμε delay centers.
  - Ισχύει, $R_i(N)=S_i$
  - Device throughputs βάσει των jobs:

$$X_i(N)=X(N)V_i$$

  - Μήκη ουρών με τη χρήση του N. Little:

$$Q_i(N)=X_i(N)R_i(N)=X(N)V_iR_i(N)$$

MVA - Ανάλυση Κλειστών Δικτύων (6)

- Η MVA ισχύει, αν είναι Product Form Network:
  - Job Flow Balance.
  - One-Step behavior.
  - Device homogeneity.
  - Εκθετικοί χρόνοι εξυπηρέτησης.

MVA - Αλγόριθμος (1)

- MVA Algorithm

Inputs:

- N : αριθμός χρηστών (jobs).
- Z : think time.
- M : αριθμός συσκευών (χωρίς τερματικά).
- S_i : service time per visit στο i -στό device.
- V_i : αριθμός επισκέψεων στο i -στό device.

Outputs:

- X : system throughput.
- Q_i : μέσος αριθμός jobs στο i –στό device.
- R_i : response time στο i -στό device.
- R : system response time.
- U_i : χρησιμοποίηση του i -στού device.

MVA - Αλγόριθμος (2)

- Αρχικοποίηση:

$$\text{FOR } i=0 \text{ TO } M \text{ DO } Q_i=0$$

- Επαναλήψεις:

FOR $n=1$ TO N DO
BEGIN

$$\text{FOR } i=1 \text{ TO } M \text{ DO } R_i = \begin{cases} S_i(1+Q_i) & \text{fixed capacity} \\ S_i & \text{delay centers} \end{cases}$$

$$R=\sum_{i=1}^{M} R_i V_i$$

$$X=\frac{n}{Z+R}$$

$$\text{FOR } i=1 \text{ TO } M \text{ DO } Q_i = XV_iR_i$$

END

MVA - Αλγόριθμος (3)

- Device Throughputs: $X_i = XV_i$
- Device Utilizations: $U_i = XS_iV_i$

Approximate MVA (1)

- Η MVA είναι αναδρομικός αλγόριθμος. Για μεγάλο N , εμφανίζει μεγάλο υπολογιστικό κόστος.
- Η προσέγγιση Schweitzer (Schweitzer's Approximation) (1979) αποφεύγει την αναδρομή του MVA.
- Schweitzer's Approximation:
  1. Εκτίμηση του Q_i με N jobs.
  2. Υπολογισμός των R_i , X_i .
  3. Υπολογισμός του Q_i . Αν η διαφορά με το προηγούμενο Q_i είναι μικρή, η νέα τιμή είναι καλή εκτίμηση.

Approximate MVA (2)

- Υπόθεση: Όσο μεγαλώνει ο αριθμός των jobs στο δίκτυο, το μήκος ουράς σε κάθε device αυξάνεται ανάλογα.

  Δηλαδή, $\frac{Q_i(N)}{N} = a_i$ (constant) $\forall N$

  Πιο συγκεκριμένα, $\frac{Q_i(N-1)}{N-1} = \frac{Q_i(N)}{N} \Rightarrow$

  $$Q_i(N-1) = \frac{N-1}{N} Q_i(N)$$

Approximate MVA (3)

- Συνεπώς, οι MVA εξισώσεις γράφονται:

$$R_i(N) = \begin{cases} S_i\left(1 + \frac{N-1}{N} Q_i(N)\right) & \text{fixed capacity} \\ S_i & \text{delay centers} \end{cases}$$

$$X(N) = \frac{N}{Z + \sum V_i R_i(N)}$$

$$Q_i(N) = X(N) \cdot V_i \cdot R_i(N)$$

Approximate MVA (4)

- Συγκλίνει;
- Οι αρχικές τιμές των $Q_i(N)$ επηρεάζουν τον αριθμό των iterations και όχι το αποτέλεσμα.
- Αρχική τιμή: $Q_i(N)=\frac{N}{M}$.

Approximate Algorithm (1)

- Approximate algorithm

Inputs:

- N : αριθμός χρηστών (jobs).
- Z : think time.
- M : αριθμός συσκευών (χωρίς τερματικά).
- S_i : service time per visit στο i-στο device.
- V_i : αριθμός επισκέψεων στο i-στο device.
- ε : μέγιστο επιτρεπόμενο λάθος στα $Q_i(N)$.

Outputs:

- X : system throughput.
- Q_i : μέσος αριθμός jobs στο i-στο device.
- R_i : response time στο i-στο device.
- U_i : χρησιμοποίηση του i-στού device.
- R : system response time.

Approximate Algorithm (2)

- Αρχικοποίηση:

$$X = 0$$

$$\text{FOR } i = 0 \text{ TO } M \text{ DO } Q_i = N / M.$$

- Επαναλήψεις:

$$\text{WHILE } \max_i \{ |Q_i - X R_i V_i| \} > \varepsilon \text{ DO}$$

BEGIN

$$\text{FOR } i = 1 \text{ TO } M \text{ DO } R_i = \begin{cases} S_i (1 + \frac{N-1}{N} Q_i) & \text{fixed capacity} \\ S_i & \text{delay centers} \end{cases}$$

$$R = \sum_{i=1}^{M} R_i V_i$$

$$X = \frac{N}{Z + R}$$

$$\text{FOR } i = 1 \text{ TO } M \text{ DO } Q_i = X V_i R_i$$

END

Approximate Algorithm (3)

- Device Throughputs: $X_i = XV_i$
- Device Utilizations: $U_i = XS_iV_i$

Balanced Job Bounds (1)

- Zahorjan, Sevcik, Eager, Galler (1982).
- Άνω και κάτω όρια βασισμένα στην παρατήρηση ότι ένα σύστημα σε ισορροπία (balanced) έχει καλύτερη απόδοση από ένα σύστημα που δεν βρίσκεται σε ισορροπία.
- Σύστημα σε ισορροπία : σύστημα χωρίς bottleneck. Οι συνολικοί χρόνοι εξυπηρέτησης των αιτήσεων είναι ίσοι σε όλα τα devices.
- Η απόδοση ενός unbalanced συστήματος μπορεί να βελτιωθεί με την αντικατάσταση του bottleneck από ένα πιο γρήγορο device, μέχρι να δημιουργηθεί κάποιο άλλο bottleneck.

Balanced Job Bounds (2)

- Ο χρόνος απόκρισης και το throughput ενός timesharing συστήματος φράσσονται ως εξής:

$$\max\left\{ND_{\max}-Z, D+(N-1)D_{avg}\frac{D}{D+Z}\right\}\le R(N)\le D+(N-1)D_{\max}\frac{(N-1)D}{(N-1)D+Z}$$

$$\frac{N}{Z+D+(N-1)D_{\max}\frac{(N-1)D}{(N-1)D+Z}}\le X(N)\le \min\left\{\frac{1}{D_{\max}}, \frac{N}{Z+D+(N-1)D_{avg}\frac{D}{D+Z}}\right\}$$

όπου, $D_{avg}=D/M$. Τα όρια είναι αυστηρά.

Balanced Job Bounds (3)

- Υποθέσεις: Όλα τα service centers είναι fixed capacity. Τα τερματικά είναι delay centers. Τα balanced job bounds προκύπτουν από τα εξής βήματα:

1. Βρίσκουμε έκφραση για τα X, R ενός balanced συστήματος.
2. Με δεδομένο ένα unbalanced σύστημα, κατασκευάζουμε ένα αντίστοιχο "best case" balanced, με ίδιο αριθμό devices και άθροισμα απαιτήσεων service. Από αυτό προκύπτουν τα άνω όρια.
3. Κατασκευάζουμε ένα αντίστοιχο "worst case" balanced, ώστε κάθε device έχει demand ίσο με bottleneck και ο αριθμός των devices προσαρμόζεται ώστε το άθροισμα των απαιτήσεων να είναι ίδιο σε balanced και unbalanced συστήματα. Από αυτό προκύπτουν τα κάτω όρια.

Balanced Job Bounds (4.1)

- Απόδειξη:
  Βήμα 1

Έστω σύστημα με κεντρικό υποσύστημα balanced:

$$D_i = \frac{D}{M}$$

Οι χρόνοι απόκρισης κάθε device, με MVA είναι: (εκθετικοί servers)

$$R_i(N) = S_i[1 + Q_i(N-1)] \qquad i = 1, 2, \ldots, M.$$

Το σύστημα είναι balanced, άρα:

$$Q_i(N-1) = \frac{Q(N-1)}{M}$$

όπου $Q(j)$ είναι ο συνολικός αριθμός jobs στο subsystem όταν υπάρχουν j jobs στο σύστημα. Ο αριθμός των jobs στα terminals είναι $j - Q(j)$

Balanced Job Bounds (4.2)

Η απόκριση του συστήματος δίνεται:

$$R(N) = \sum_{i=1}^{M} V_i R_i(N) = \sum_{i=1}^{M} \frac{D}{M}\left[1 + \frac{Q(N-1)}{M}\right] \Rightarrow$$

$$R(N) = D\left[1 + \frac{Q(N-1)}{M}\right] = D + \frac{D}{M} Q(N-1)$$

Διαδικασία να βάλουμε bound στο $Q(N)$:

1. Αν αντικαταστήσουμε το σύστημα με N workstations έτσι ώστε κάθε χρήστης να έχει το δικό του workstation και κάθε workstation είναι ίδιο με το αρχικό σύστημα, το νέο σύστημα θα έχει καλύτερο R και X . Το νέο περιβάλλον έχει N single user systems. Κάθε χρήστης κάνει κύκλους με Z χρόνο σκέψης και D χρόνο computing.

Balanced Job Bounds (4.3)

Κάθε job έχει πιθανότητα $\frac{D}{D+Z}$ να είναι στο κεντρικό υποσύστημα και συνεπώς:

$$\frac{Q(N)}{N} \geq \frac{D}{D+Z} \qquad (1)$$

2. Τώρα, θεωρούμε ένα άλλο περιβάλλον, ίδιο με το προηγούμενο, με τη διαφορά ότι κάθε χρήστης έχει workstation N φορές αργότερο από το αρχικό σύστημα. Το νέο περιβάλλον έχει συνολική υπολογιστική ισχύ ίδια με το αρχικό, αλλά δεν υπάρχει διαμοιρασμός (sharing). Δηλαδή, οι χρήστες περνούν περισσότερο χρόνο στο κεντρικό σύστημα:

$$\frac{Q(N)}{N} \leq \frac{ND}{ND+Z} \qquad (2)$$

Balanced Job Bounds (4.4)

Από τις (1), (2) προκύπτει το εξής όριο για τον αριθμό στα devices:

$$\frac{D}{D+Z} \le \frac{Q(N)}{N} \le \frac{ND}{ND+Z} \Rightarrow$$

$$(N-1)\frac{D}{D+Z} \le Q(N-1) \le (N-1)\frac{(N-1)D}{(N-1)D+Z} \xrightarrow{R(N)}$$

$$D+\frac{D}{M}(N-1)\frac{D}{D+Z} \le R(N) \le D+\frac{D}{M}(N-1)\frac{(N-1)D}{(N-1)D+Z}$$

Balanced Job Bounds (4.5)

Βήμα 2

Έστω ότι έχουμε unbalanced σύστημα τέτοιο ώστε, οι απαιτήσεις εξυπηρέτησης στο i – στο device να είναι D_i . Θεωρούμε την εξής διάταξη:

$$D_1 \leq D_2 \leq \ldots \leq D_M$$

Κατ' αυτόν τον τρόπο το M – στο device είναι το αργότερο (bottleneck device) και το πρώτο το γρηγορότερο. Εκτελούμε το παρακάτω πείραμα:

Balanced Job Bounds (4.6)

1. Κάνουμε το bottleneck device λίγο γρηγορότερο και το γρηγορότερο device, λίγο αργότερο. Η performance θα βελτιωθεί.
2. Συνεχίζουμε τη διαδικασία μέχρι το bottleneck να γίνει ίσο με το bottleneck κάποιου άλλου device. Συνεπώς, διαθέτουμε 2 bottleneck devices.
3. Παίρνουμε τα δύο πιο αργά και τα δύο πιο γρήγορα devices στο σύστημα και εφαρμόζουμε το βήμα 1. Αυτό θα βελτιώσει την performance μέχρι και το τρίτο device να μπει στο bottleneck group. Την ίδια στιγμή προσθέτουμε ένα device στο group των γρήγορων devices.
4. Στο τέλος, θα έχουμε ένα balanced σύστημα όπου κάθε device θα έχει:

$$D_i = D_{avg} = \frac{1}{M}\sum_{i=1}^{M} D_i = \frac{D}{M}$$

Balanced Job Bounds (4.7)

Το νέο balanced σύστημα έχει καλύτερη performance. Η παρατήρηση αυτή, οδηγεί στα ακόλουθα όρια της απόδοσης του unbalanced συστήματος:

$$R(N) \geq D + (N-1)D_{avg}\frac{D}{D+Z} \quad (3)$$

$$X(N) \leq \frac{N}{Z + D + (N-1)D_{avg}\frac{D}{D+Z}} \quad (4)$$

Balanced Job Bounds (4.8)

Βήμα 3

Ένα άλλο πείραμα στο unbalanced σύστημα:

1. Τα devices με D_i ίσο με $D_{\max}$ (bottleneck) απομακρύνονται από το σύνολο των devices που πρόκειται να δεχθούν αλλαγές. Από τα υπόλοιπα, παίρνουμε το γρηγορότερο και το αργότερο. Έστω ότι αυτά είναι τα devices 1 και k , αντίστοιχα. Ισχύει, k = M-1, εκτός κι αν, $D_M = D_{M-1}$,οπότε και παίρνουμε το μεγαλύτερο k με $D_k \neq D_M$.
   Μειώνουμε κατά ΔD το γρηγορότερο και αυξάνουμε κατά ΔD το k.
2. Συνεχίζουμε την διαδικασία, μέχρι το γρηγορότερο να γίνει ίσο με 0 και το αργό να γίνει ίσο με $D_{\max}$. Όποιο από τα devices φτάσει πρώτο το όριο της, απομακρύνεται από το σύνολο των devices που πρόκειται να δεχθούν αλλαγές.
3. Επαναλαμβάνουμε το βήμα 1 για το νέο σύνολο devices, κ.ο.κ.

Balanced Job Bounds (4.9)

4. Στο τέλος θα έχουμε ένα balanced σύστημα όπου $M' = \frac{D}{D_{\max}}$ devices έχουν demands ίσα με $D_{\max}$.
   Τα υπόλοιπα έχουν 0 και απομακρύνονται από το σύστημα.

5. Αφού κάθε αλλαγή χειροτερεύει την performance, τελικά θα έχουμε χειρότερη performance από το unbalanced. Δηλαδή,

$$D + (N-1)D_{\max} \frac{(N-1)D}{(N-1)D + Z} \geq R(N)$$

$$\frac{N}{Z + D + (N-1)D_{\max} \frac{(N-1)D}{(N-1)D + Z}} \leq X(N)$$

Συνδυάζοντας τις (3), (4) με τις παραπάνω προκύπτουν τα ασυμπωτικά όρια.

Balanced Job Bounds (4.10)

$$\max\left\{ND_{\max}-Z, D+(N-1)D_{avg}\frac{D}{D+Z}\right\}\le R(N)\le D+(N-1)D_{\max}\frac{(N-1)D}{(N-1)D+Z}$$

$$\frac{N}{Z+D+(N-1)D_{\max}\frac{(N-1)D}{(N-1)D+Z}}\le X(N)\le \min\left\{\frac{1}{D_{\max}}, \frac{N}{Z+D+(N-1)D_{avg}\frac{D}{D+Z}}\right\}$$