**ΣΥΣΤΗΜΑΤΑ ΑΝΑΜΟΝΗΣ**

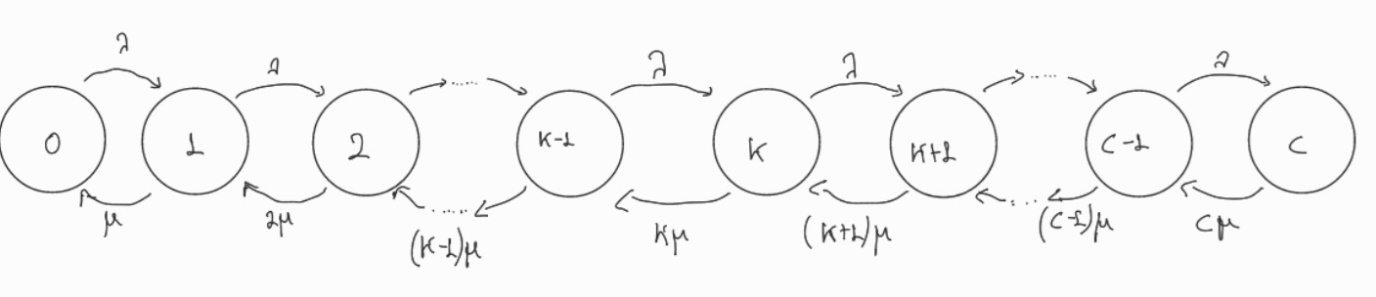
ΝΤΟΝΤΟΡΟΣ ΗΛΙΑΣ

el19206

**ΑΣΚΗΣΗ 1η**

Ερώτημα 1ο

Το διάγραμμα μεταβάσεων του συστήματος είναι το παρακάτω :



Από τις εξισώσεις ισορροπίας έχουμε :

Από την πρώτη αναδρομικά έχουμε :

Με αντικατάσταση έχουμε :

Η πιθανότητα απόρριψης πελάτη είναι η πιθανότητα το σύστημα να βρίσκεται στην κατάσταση k :

Για k = c προκύπτει ο τυπος Erlang-B :

Για να υπολογίσουμε τον μέσο αριθμό απωλειών πελατών πολλαπλασιάζουμε την πιθανότητα απόρριψης με τον ρυθμό άφιξης πελατών :

Ο κώδικας στο octave για την υλοποίηση της συνάρτησης είναι :

1. function p = erlangb\_factorial (r,c)
2. s = 0;
3. for k = 0:1:c
4. s = s + (power(r,k)/factorial(k));
5. endfor
6. p = (power(r,c)/factorial(c))/s;
7. endfunction

και τα αποτελέσματα είναι ίδια με την έτοιμη συνάρτηση του octave :

1. Erlangb\_factorial(9,9) =
2. 0.2243
3. Erlangb(9,9) =
4. 0.2243

Ερώτημα 2ο

Η συνάρτηση στο octave είναι :

1. function p = erlangb\_iterative (r,c)
2. p = 1;
3. for i=0:1:c
4. p = ((r\*p)/((r\*p)+i));
5. endfor
6. endfunction

και τα αποτελέσματα που βγάζει είναι ίδια με το παραπάνω :

1. Erlangb\_iterative(9,9) =
2. 0.2243

Ερώτημα 3ο

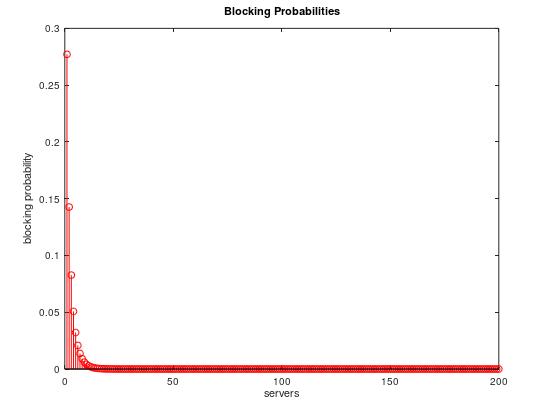
Τα αποτελέσματα που παίρνουμε είναι :

1. Erlangb\_factorial(1024,1024) =
2. NaN
3. Erlangb\_iterative(1024,1024) =
4. 0.024524

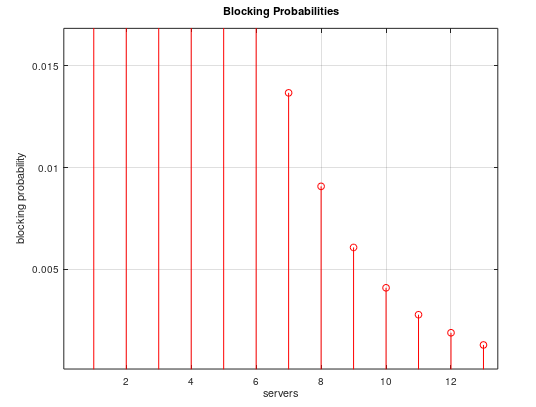
Η erlangb\_factorial προσπαθεί να υπολογίσει το 1024! το οποίο είναι υπερβολικά μεγάλος αριθμός οπότε δεν μπορεί να βρει τελικό αποτέλεσμα.

Ερώτημα 4ο

1. Έχοντας ως πρότυπο τον πιο απαιτητικό χρήστη τότε η συνολική κυκλοφοριακή ένταση του δικτύου είναι :
2. Το διάγραμμα πιθανότητα απόρριψης πελάτη ως προς το αριθμό τηλεφωνικών γραμμών είναι :



1. Αν μεγεθύνουμε το παραπάνω διάγραμμα βλέπουμε ότι για να έχουμε πιθανότητα απόρριψης μικρότερη από 1% πρέπει ο αριθμός τηλεφωνικών γραμμών να είναι μεγαλύτερος η ίσος με 8.



Ο κώδικας για ολόκληρη την άσκηση 1 :

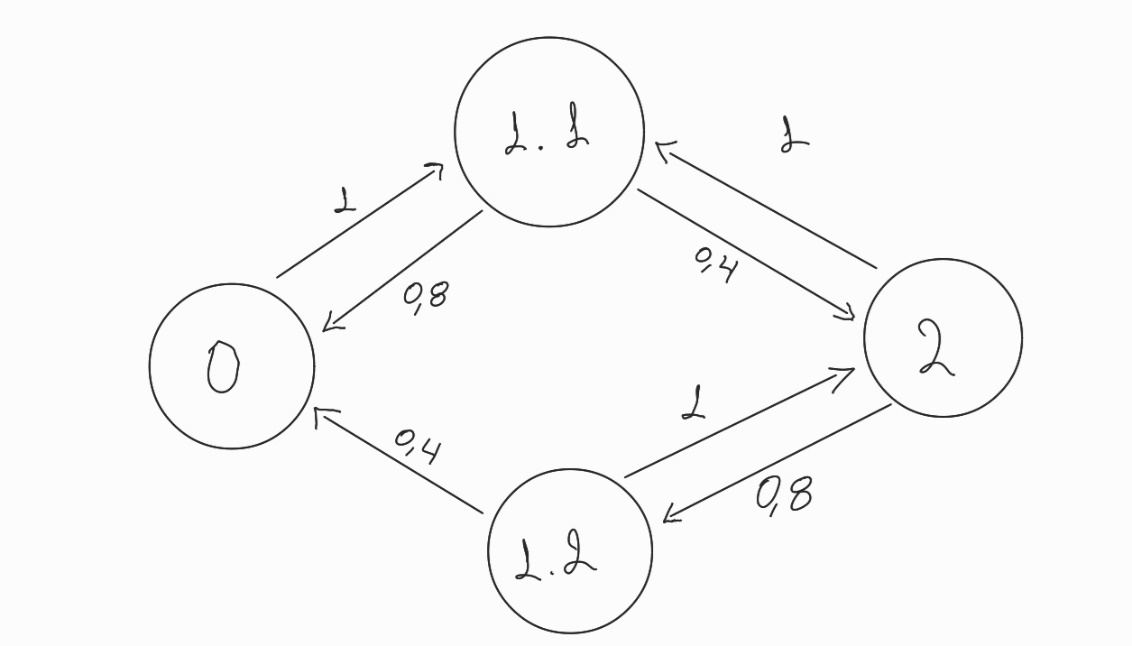
1. # Exercise 1
2. clc;
3. clear all;
4. close all;
5. pkg load queueing;
7. % r = lamda / mu
8. % c: number of servers
10. function p = erlangb\_factorial (r,c)
11. s = 0;
12. for k = 0:1:c
13. s = s + (power(r,k)/factorial(k));
14. endfor
15. p = (power(r,c)/factorial(c))/s;
16. endfunction

19. function p = erlangb\_iterative (r,c)
20. p = 1;
21. for i=0:1:c
22. p = ((r\*p)/((r\*p)+i));
23. endfor
24. endfunction
26. display("Erlangb\_factorial(9,9) =");
27. disp(erlangb\_factorial(9,9));
29. display("Erlangb(9,9) =");
30. disp(erlangb(9,9));
32. display("Erlangb\_iterative(9,9) =");
33. disp(erlangb\_iterative(9,9));
35. display("Erlangb\_factorial(1024,1024) =");
36. disp(erlangb\_factorial(1024,1024));
38. display("Erlangb\_iterative(1024,1024) =");
39. disp(erlangb\_iterative(1024,1024));
41. P = zeros(0,200);
43. for i = 1:1:200
44. P(i) = erlangb\_iterative (i\*(23/60),i)
45. endfor
47. figure(1);
48. stem(P,'b',"linewidth",0.4);
49. title("Blocking Probabilities")
50. xlabel("Servers");
51. ylabel("Blocking probability");

**ΑΣΚΗΣΗ 2η**

Ερώτημα 1o

Το διάγραμμα ρυθμών μεταβάσεων του συστήματος στην κατάσταση ισορροπίας είναι :



1. Για το σύστημα μας ισχύουν οι παρακάτω εξισώσεις :

Αντικαθιστώντας τις τιμές :

προκύπτουν οι εργοδικές πιθανότητες :

1. Η πιθανότητα απόρριψης πελάτη από το σύστημα είναι :
2. Ο μεσος αριθμός πελατών στο σύστημα είναι :

Ερώτημα 2o

1. Τα thresholds που χρησιμοποιήσαμε είναι :
2. threshold\_0 = lamda/1;
3. threshold\_1a = lamda/(lamda+m1);
4. threshold\_1b = lamda/(lamda+m2);
5. threshold\_2\_first = lamda/(lamda+m2+m1);
6. threshold\_2\_second = (m1+lamda)/(lamda+m2+m1);

threshold\_0: Από την κατάσταση 0 είναι δυνατή να πραγματοποιηθεί μόνο άφιξη επομένως θα ισούται με 1.

threshold\_1a: Από την κατάσταση 1\_1 μπορούμε να έχουμε είτε αναχώρηση είτε άφιξη και να μεταβούμε στην κατάσταση 0 με ρυθμό µ1 είτε στην κατάσταση 2 με ρυθμό λ αντίστοιχα. Επομένως ο παρονομαστής θα είναι λ + µ1 και ο αριθμητής λ.

threshold\_1b: Από την κατάσταση 1\_2 μπορούμε να έχουμε είτε αναχώρηση είτε άφιξη και να μεταβούμε στην κατάσταση 0 με ρυθμό µ2 είτε στην κατάσταση 2 με ρυθμό λ αντίστοιχα. Επομένως ο παρονομαστής θα είναι λ + µ2 και ο αριθμητής λ.

threshold\_2\_first: Από την κατάσταση 2 μπορούμε να έχουμε άφιξη και απόρριψη του πελάτη με ρυθμό λ, επομένως ο αριθμητής του threshold\_2\_first θα είναι λ.

threshold\_2\_second: Από την κατάσταση 2 μπορούμε να έχουμε αναχώρηση και να μεταβούμε στην κατάσταση 1\_2 με ρυθμό µ1, επομένως ο αριθμητής του threshold\_2\_second θα είναι λ+µ1.

Για τα δυο τελευταία μπορούμε να έχουμε αναχώρηση και να πάμε στην κατάσταση 1\_1 με ρυθμό µ2, άρα ο παρονομαστής θα είναι λ + µ1 + µ2

1. Το κριτήριο σύγκλισης είναι η διαφορά μεταξύ δυο διαδοχικών μέσων αριθμών πελατών να είναι κάτω από 0,001%
2. Οι πιθανότητες που υπολογίζει η προσομοίωση είναι :
3. 0.2517
4. 0.2168
5. 0.1939
6. 0.3375

Οι οποίες είναι ίσες με τις τιμές που υπολογίσαμε παραπάνω αλλά υπάρχουν μικρές αποκλίσεις

Ο κώδικας για ολόκληρη την άσκηση 2 :

1. # Exercise 2
3. clc;
4. clear all;
5. close all;
6. pkg load queueing;
8. lamda = 1;
9. m1 = 0.8;
10. m2 = 0.4;
12. threshold\_0 = lamda/1;
13. threshold\_1a = lamda/(lamda+m1);
14. threshold\_1b = lamda/(lamda+m2);
15. threshold\_2\_first = lamda/(lamda+m2+m1);
16. threshold\_2\_second = (m1+lamda)/(lamda+m2+m1);
18. current\_state = 0;
19. arrivals = zeros(1,4);
20. total\_arrivals = 0;
21. maximum\_state\_capacity = 2;
22. previous\_mean\_delay = 0;
23. delay\_counter = 0;
24. time = 0;
26. while 1 > 0
27. time = time + 1;
29. if mod(time,1000) == 0
30. for i=1:1:4
31. P(i) = arrivals(i)/total\_arrivals;
32. endfor
34. delay\_counter = delay\_counter + 1;
36. mean\_delay = 0\*P(1) + 1\*P(2) + 1\*P(3) + 2\*P(4);
38. delay\_table(delay\_counter) = mean\_delay;
40. if abs(mean\_delay - previous\_mean\_delay) < 0.00001
41. break;
42. endif
43. previous\_mean\_delay = mean\_delay;
44. endif
46. random\_number = rand(1);
48. if current\_state == 0
49. if random\_number < threshold\_0
50. current\_state = 1;
51. arrivals(1) = arrivals(1) + 1;
52. total\_arrivals = total\_arrivals + 1;
53. endif
54. elseif current\_state == 1
55. if random\_number < threshold\_1a
56. current\_state = 3;
57. arrivals(2) = arrivals(2) + 1;
58. total\_arrivals = total\_arrivals + 1;
59. else
60. current\_state = 0;
61. endif
62. elseif current\_state == 2
63. if random\_number < threshold\_1b
64. current\_state = 3;
65. arrivals(3) = arrivals(3) + 1;
66. total\_arrivals = total\_arrivals + 1;
67. else
68. current\_state = 0;
69. endif
70. else
71. if random\_number < threshold\_2\_first
72. arrivals(4) = arrivals(4) + 1;
73. total\_arrivals = total\_arrivals + 1;
74. elseif random\_number < threshold\_2\_second
75. current\_state = 2;
76. else
77. current\_state = 1;
78. endif
79. endif
81. endwhile
83. display(P(1));
84. display(P(2));
85. display(P(3));
86. display(P(4));