Συγχρονισμός: Αδιέξοδο & Παρατεταμένη Στέρηση

Κεφάλαιο 6

Dr. Garmpis Aristogiannis - EPDO
TEI Messolonghi

Αδιέξοδο

- Μόνιμη αναμονή ενός συνόλου διεργασιών οι οποίες ανταγωνίζονται για πόρους του συστήματος ή για να επικοινωνήσουν μεταξύ τους
- Δεν υπάρχει αποδοτική λύση
- Εμπεριέχουν αντικρουόμενες ανάγκες για πόρους από δύο ή περισσότερες διεργασίες

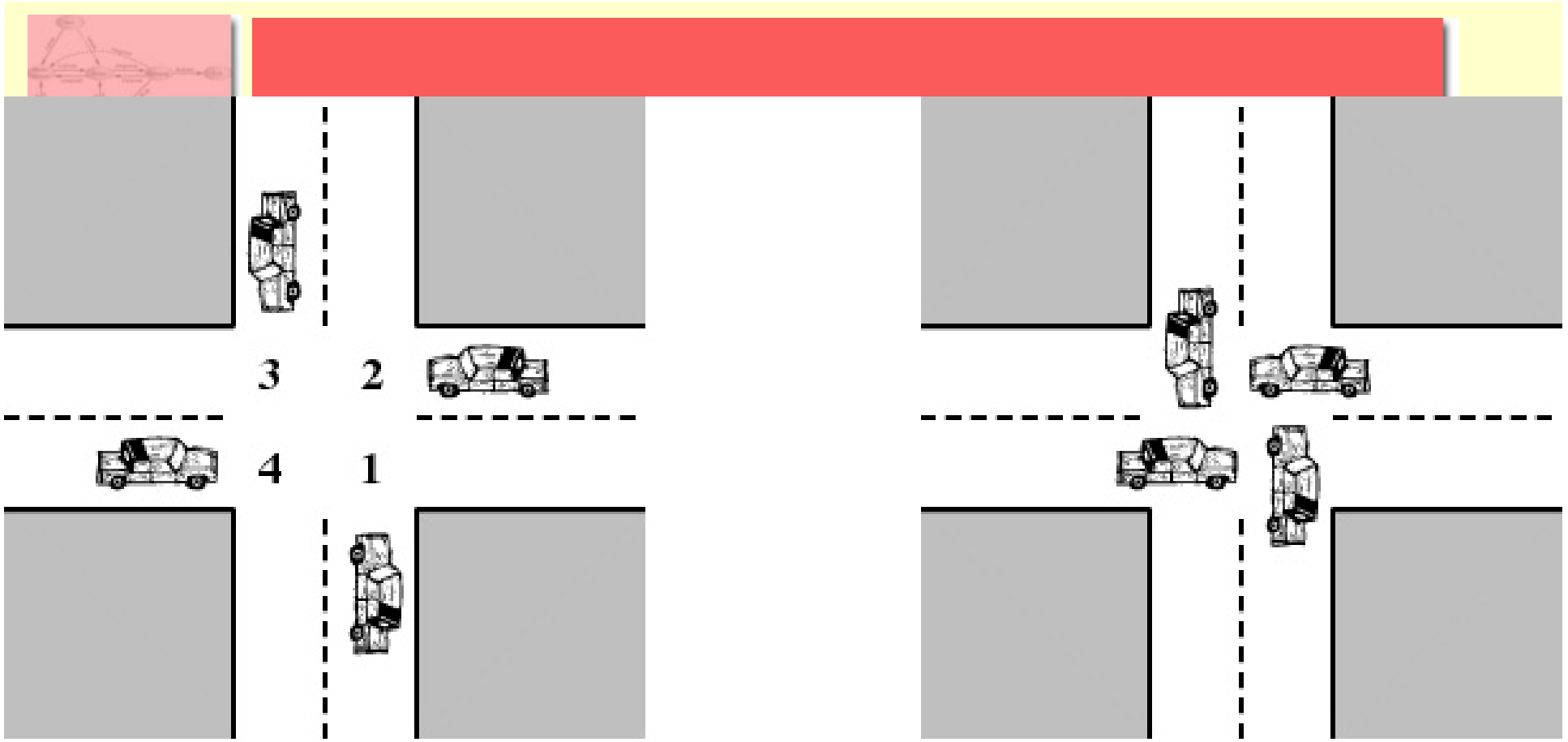


Figure 6.1 Illustration of Deadlock

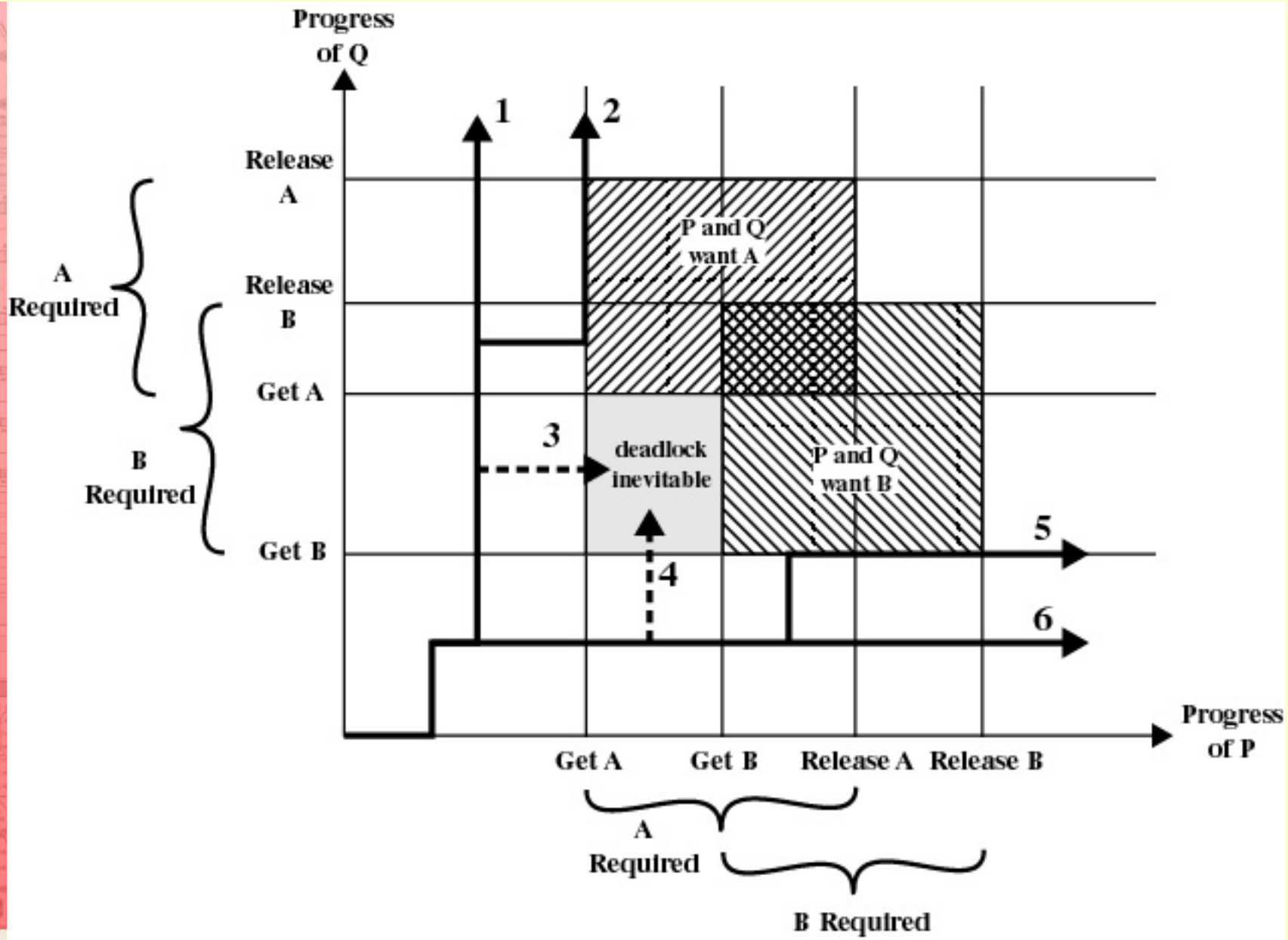


Figure 6.2 Example of Deadlock [BACO98]

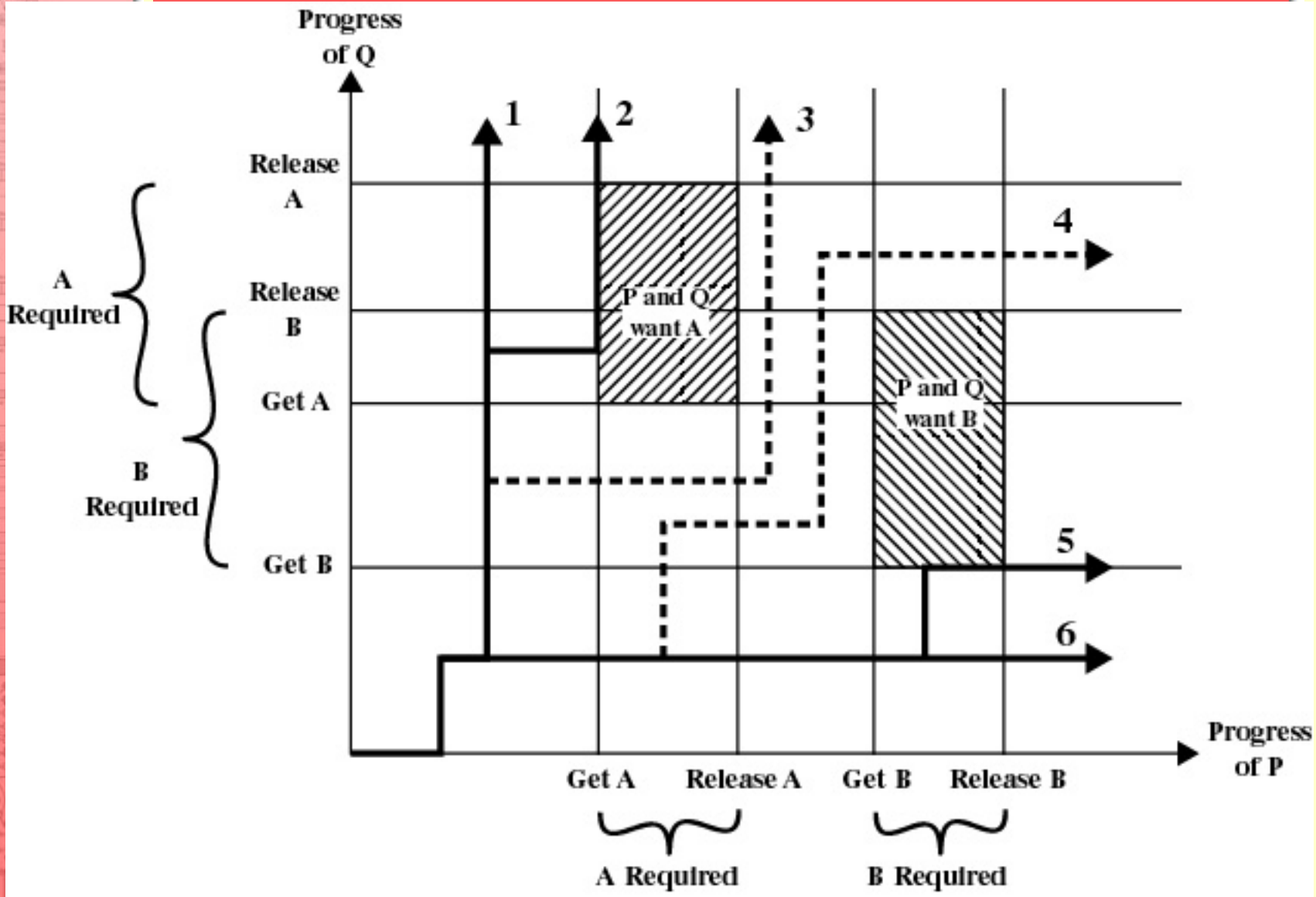


Figure 6.3 Example of No Deadlock [BACO98]

Επαναχρησιμοποιήσιμοι Πόροι

- Χρησιμοποιούνται από μία διεργασία κάθε φορά και δεν εξαντλείται από την χρήση του
- Η διεργασία δεσμεύει πόρους οι οποίοι αργότερα απελευθερώνονται για να χρησιμοποιηθούν από άλλες διεργασίες
- Επεξεργαστές, κανάλια I/O, κύρα και δευτερεύουσα μνήμη, αρχεία, βάσεις δεδομένων και σημαφόροι
- Αδιέξοδο δημιουργείται όταν μία διεργασία παρακρατά έναν πόρο και απαιτεί κάποιον άλλον για να συνεχίσει την εκτέλεση της

Παράδειγμα Αδιεξόδου

Process P

Step	Action
p_0	Request (D)
p_1	Lock (D)
p_2	Request (T)
p_3	Lock (T)
p_4	Perform function
p_5	Unlock (D)
p_6	Unlock (T)

Process Q

Step	Action
q_0	Request (T)
q_1	Lock (T)
q_2	Request (D)
q_3	Lock (D)
q_4	Perform function
q_5	Unlock (T)
q_6	Unlock (D)

Figure 6.4 Example of Two Processes Competing for Reusable Resources

Παράδειγμα Αδιεξόδου (2)

- Υπάρχουν 200Kbytes ελεύθερα για χρήση και δημιουργείται η ακόλουθη σειρά γεγονότων

P1

. . .

Request 80K bytes;

. . .

Request 60K bytes;

P2

. . .

Request 70K bytes;

. . .

Request 80K bytes;

- Αδιέξοδο δημιουργείται αν και οι δύο διεργασίες εκτελέσουν την δεύτερη αίτηση χορήγησης μνήμης

Καταναλώσιμοι Πόροι

- Δημιουργούνται (παράγονται) και καταστρέφονται (καταναλώνονται) από μία διεργασία
- Διακοπές, σήματα, μηνύματα, πληροφορία σε ενδιάμεσους αποθηκευτικούς χώρους (buffers) I/O
- Αδιέξοδο δημιουργείται όταν μία διεργασία πού θέλει να πραγματοποιήσει λήψη ενός μηνύματος αναστέλλεται μέχρι να το παραλάβει (Blocking Receive message)
- Αιτία αδιεξόδου μπορεί να είναι ένας σπάνιος συνδυασμός γεγονότων

Παράδειγμα Αδιεξόδου

P1

```
. . .
Receive(P2);
. . .
Send(P2, M1);
```

P2

```
. . .
Receive(P1);
. . .
Send(P1, M2);
```

Συνθήκες για Αδιέξοδο

- Αμοιβαίος Αποκλεισμός
  - Μόνο μία διεργασία μπορεί να χρησιμοποιήσει έναν πόρο κάθε χρονική στιγμή
- Παρακράτηση και Αναμονή(Hold-and-wait)
  - Η διεργασία απαιτεί όλους τους απαιτούμενος πόρους για να εκτελεστεί σε μία χρονική στιγμή

Συνθήκες για Αδιέξοδο

- Μη προεκχώρηση (No preemption)
  - Αν μία διεργασία που απαιτεί παρακρατεί έναν πόρο και δεν ικανοποιηθεί μία αίτηση της για κάποιον άλλο πόρο, η διεργασία πρέπει να απελευθερώσει τους πόρους που παρακρατεί
  - Αν μια διεργασία απαιτήσει πόρους οι οποίοι παρακρατούνται από άλλη διεργασία το λειτουργικό σύστημα μπορεί να αναστήλλει την δεύτερη διεργασία και να απαιτήσει την απελευθέρωση των πόρων
  - Κανένας πόρος δεν μπορεί να αποσπαστεί βίαια από την διεργασία που τον κατέχει

Συνθήκες για Αδιέξοδο

- Κυκλική Αναμονή
  - Μπορεί να αποτραπεί ορίζοντας μια γραμμική διάταξη των κατηγοριών πόρων

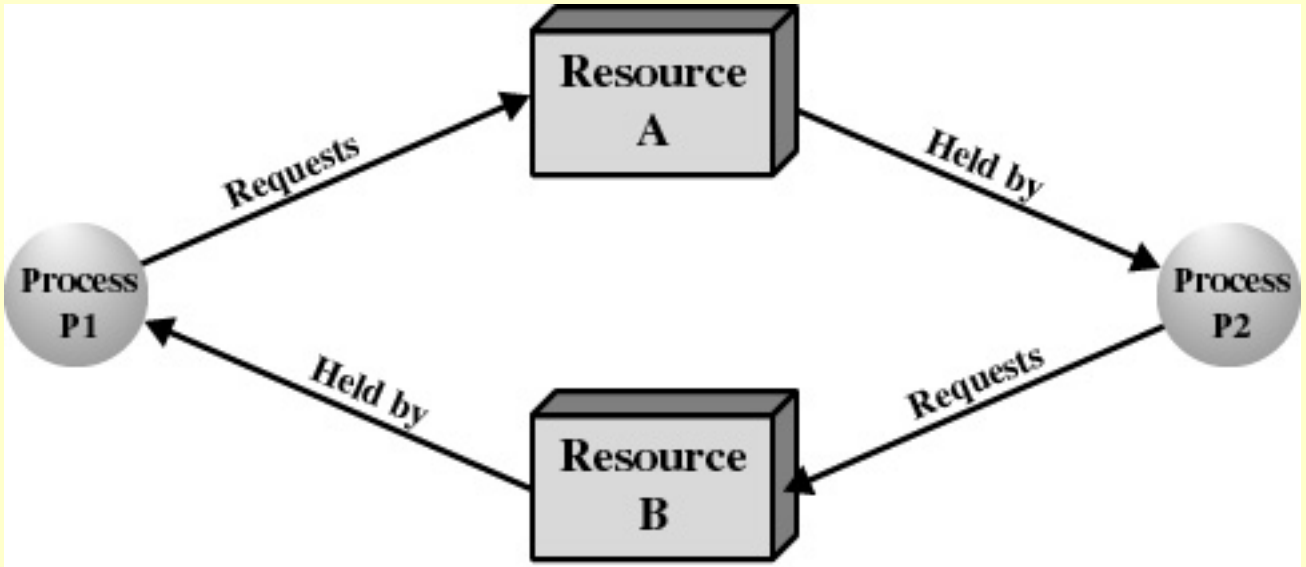


Figure 6.5 Circular Wait

Αποτροπή Αδιεξόδου

- Αποφασίζεται δυναμικά αν η υλοποίηση της αίτησης για την εκχώρηση ενός πόρου μπορεί να οδηγήσει σε πιθανό μελλοντικό αδιέξοδο
- Απαιτεί γνώση των μελλοντικών αιτήσεων για εκχώρηση πόρων

Δύο προσεγγίσεις για την Αποτροπή Αδιεξόδου

- Μη εκκίνηση μιας διεργασίας που οι απαιτήσεις της μπορεί να οδηγήσουν σε μελλοντικό αδιέξοδο
- Μη εκχώρηση ενός επαυξητικού πόρου σε μία διεργασία, αν η εκχώρηση μπορεί να οδηγήσει σε αδιέξοδο

Άρνηση εκχώρησης πόρων

- Αναφέρεται σαν ο αλγόριθμος του τραπεζίτη
- Κατάσταση Συστήματος είναι η τρέχουσα εκχώρηση πόρων σε διεργασίες
- Ασφαλής κατάσταση έχουμε όταν υπάρχει τουλάχιστον μια ακολουθία εκτέλεσης που δεν οδηγεί σε αδιέξοδο
- Ανασφαλής κατάσταση είναι η κατάσταση που δεν είναι ασφαλής

Προσδιορισμός Ασφαλούς Κατάστασης
Αρχική Κατάσταση

	R1	R2	R3
P1	3	2	2
P2	6	1	3
P3	3	1	4
P4	4	2	2

Claim Matrix

	R1	R2	R3
P1	1	0	0
P2	6	1	2
P3	2	1	1
P4	0	0	2

Allocation Matrix

R1	R2	R3
9	3	6

Resource Vector

R1	R2	R3
0	1	1

Available Vector

(a) Initial state

Προσδιορισμός Ασφαλούς Κατάστασης
Η P2 εκτελείται μέχρι τέλους

	R1	R2	R3
P1	3	2	2
P2	0	0	0
P3	3	1	4
P4	4	2	2

Claim Matrix

	R1	R2	R3
P1	1	0	0
P2	0	0	0
P3	2	1	1
P4	0	0	2

Allocation Matrix

R1	R2	R3
6	2	3

Available Vector

(b) P2 runs to completion

Dr. Garmpis Aristogiannis - EPDO
TEI Messolonghi

Προσδιορισμός Ασφαλούς Κατάστασης
Η P1 εκτελείται μέχρι τέλους

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	3	1	4
P4	4	2	2

Claim Matrix

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	2	1	1
P4	0	0	2

Allocation Matrix

R1	R2	R3
7	2	3

Available Vector

(c) P1 runs to completion

Dr. Garmpis Aristogiannis - EPDO
TEI Messolonghi

Προσδιορισμός Ασφαλούς Κατάστασης
Η P3 εκτελείται μέχρι τέλους

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	0	0	0
P4	4	2	2

Claim Matrix

	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	0	0	0
P4	0	0	2

Allocation Matrix

R1	R2	R3
9	3	4

Available Vector

(d) P3 runs to completion

Προσδιορισμός Ανασφαλούς Κατάστασης

	R1	R2	R3
P1	3	2	2
P2	6	1	3
P3	3	1	4
P4	4	2	2

Claim Matrix

	R1	R2	R3
P1	1	0	0
P2	5	1	1
P3	2	1	1
P4	0	0	2

Allocation Matrix

R1	R2	R3
9	3	6

Resource Vector

R1	R2	R3
1	1	2

Available Vector

(a) Initial state

Προσδιορισμός Ανασφαλούς Κατάστασης

	R1	R2	R3
P1	3	2	2
P2	6	1	3
P3	3	1	4
P4	4	2	2

Claim Matrix

	R1	R2	R3
P1	2	0	1
P2	5	1	1
P3	2	1	1
P4	0	0	2

Allocation Matrix

R1	R2	R3
0	1	1

Available Vector

(b) P1 requests one unit each of R1 and R3

Αποφυγή Αδιεξόδου

- Η μέγιστη απαίτηση ποσότητας ενός πόρου πρέπει να δηλωθεί στο λειτουργικό σύστημα από πριν
- Οι διεργασίες πρέπει να είναι ανεξάρτητες. Δεν πρέπει να απαιτείται συγχρονισμός
- Απαιτείται να υπάρχει σταθερός αριθμός πόρων προς εκχώρηση
- Καμία διεργασία δεν μπορεί να τερματιστεί ενώ παρακρατεί πόρους

Αναγνώριση Αδιεξόδου

	R1	R2	R3	R4	R5
P1	0	1	0	0	1
P2	0	0	1	0	1
P3	0	0	0	0	1
P4	1	0	1	0	1

Request Matrix Q

	R1	R2	R3	R4	R5
P1	1	0	1	1	0
P2	1	1	0	0	0
P3	0	0	0	1	0
P4	0	0	0	0	0

Allocation Matrix A

R1	R2	R3	R4	R5
2	1	1	2	1

Resource Vector

R1	R2	R3	R4	R5
0	0	0	0	1

Available Vector

Figure 6.9 Example for Deadlock Detection

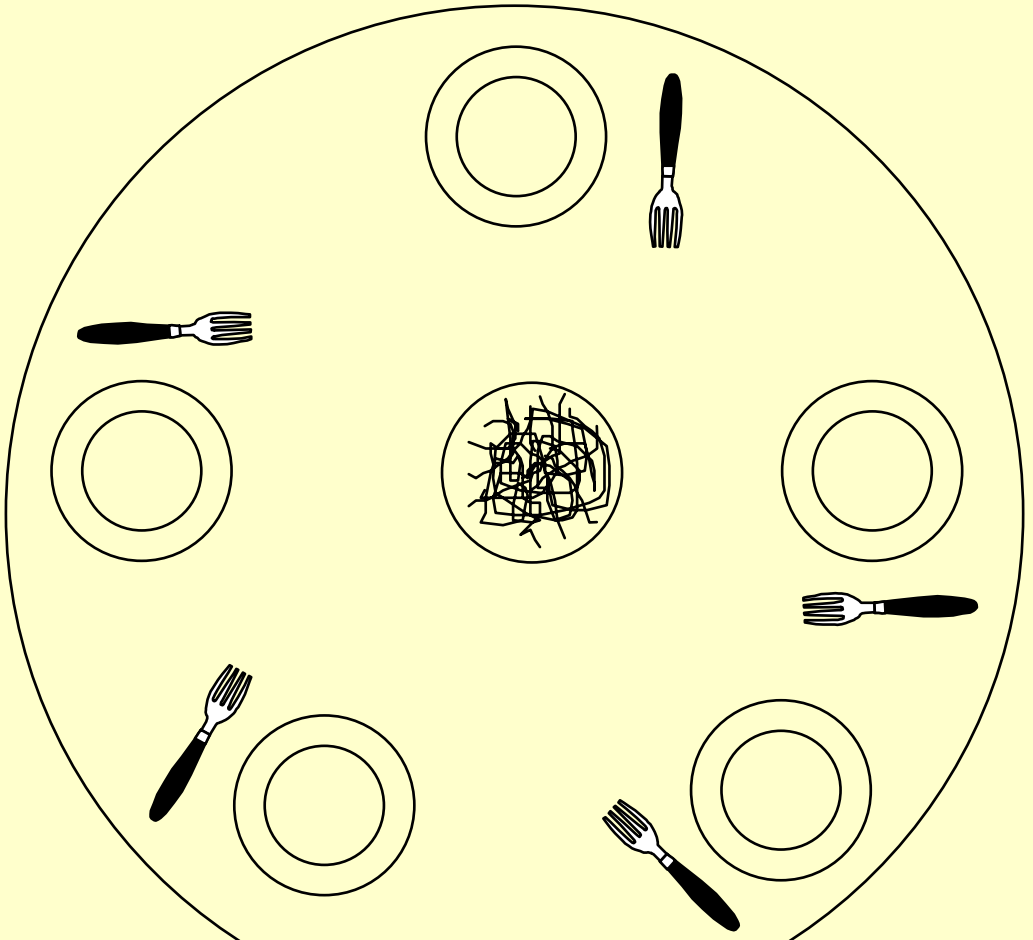
Στρατηγικές αν αναγνωριστεί Αδιέξοδο

- Διακοπή όλων των διεργασιών που βρίσκονται σε αδιέξοδο
- Υπαναχώρηση κάθε διεργασίας που βρίσκεται σε αδιέξοδο σε κάποιο προηγούμενο σημείο ανάνηψης(checkpoint) και επανεκκίνηση όλων των διεργασιών
  - Μπορεί να ξαναπραγματοποιηθεί το αρχικό αδιέξοδο
- Διακοπή των διεργασιών που συμμετέχουν στο αδιέξοδο με τη σειρά μέχρι την διακοπή του αδιεξόδου
- Ακολουθιακή προεκχώρηση πόρων μέχρι την διακοπή του αδιεξόδου

Επιλογή Διεργασιών σε Αδιέξοδο

- Επιλέγεται η διεργασία που:
  - έχει καταναλώσει τον ελάχιστο επεξεργαστικό χρόνο
  - έχει δημιουργήσει την λιγότερη έξοδο
  - εκτιμάται ότι θα απαιτήσει τον μεγαλύτερο χρόνο εκτέλεσης για να ολοκληρωθεί
  - Έχει παρακρατήσει τους λιγότερους πόρους
  - Έχει την μικρότερη προτεραιότητα

Το πρόβλημα των Συνεστιαζόμενων Φιλοσόφων



Μηχανισμός Συγχρονισμού UNIX

- Σωληνώσεις (Pipes)
- Μηνύματα
- Κοινή μνήμη
- Σημαφόροι
- Σήματα

Μηχανισμοί Συγχρονισμού Windows 2000/XP

- Διεργασίες
- Νήματα
- Αρχεία
- Είσοδος Κονσόλας
- Ειδοποίηση αλλαγής αρχείου
- Mutex
- Σημαφόροι (Semaphore)
- Γεγονότα (Event)
- Χρονόμετρα με αναμονή (Waitable timer)