Φροντιστήριο 1 – Σκελετοί Λύσεων

Λύση Άσκησης 1

(α) Χρησιμοποιούμε τις επιπλέον μεταβλητές PC1, PC2, (program counters) οι οποίες παίρνουν ως τιμές ονόματα των γραμμών του κώδικα όπως φαίνεται πιο κάτω.

```
Bool T[2] = [0,0];

while True do {
A1   T[0] = 1;
A2   while T[1] = 1 do {/* nothing /*;}
A3   CRITICAL;
A4   T[0] = 0;
}
||
while True do {
B1   T[1] = 1;
B2   while T[0] = 1 do {/* nothing /*;}
B3   CRITICAL;
B4   T[1] = 0;
}
```

Ο γράφος προγράμματος δίνεται από την πλειάδα (V, S , T, p) όπου

V = {PC1, PC2, T[2]}

S = σύνολο όλων των δυνατών αναθέσεων τιμών στις πιο πάνω μεταβλητές

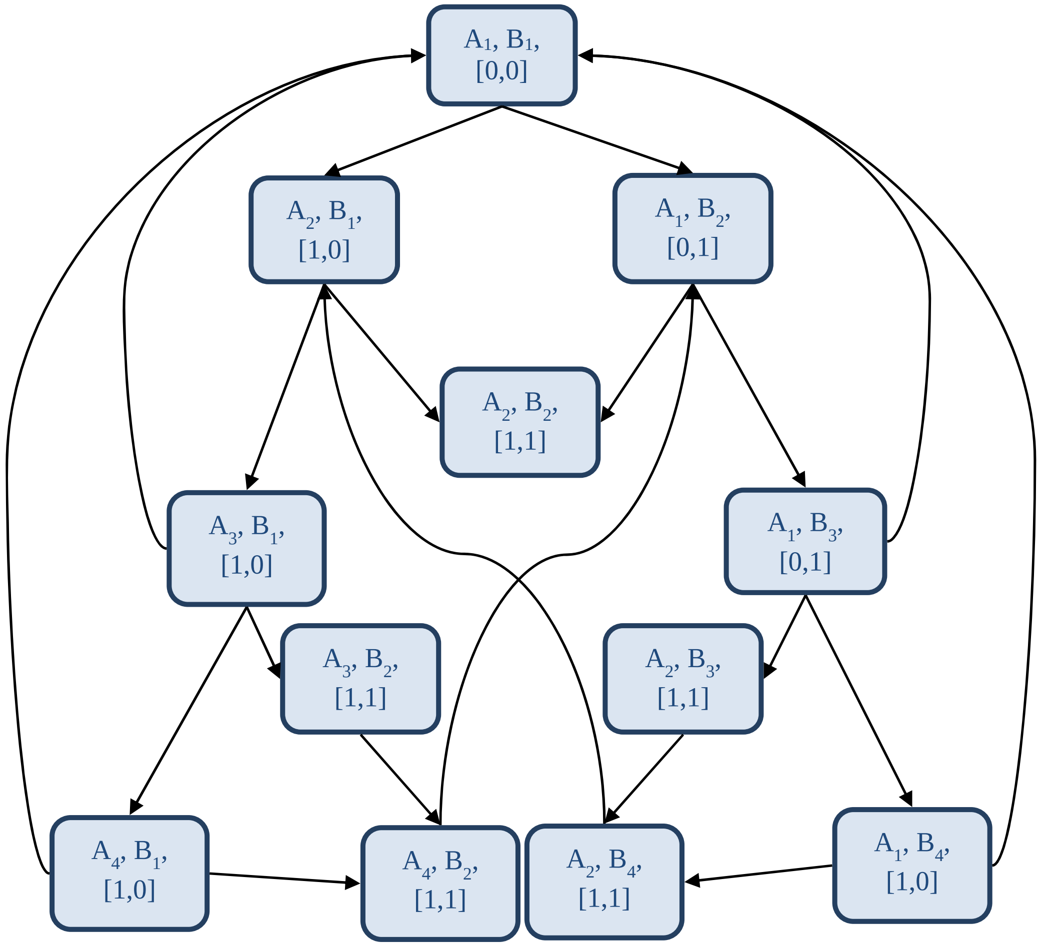
T = {

T1 :	PC1= A1	$\rightarrow$	(PC1, T[0]) := (A2, 1) ,
T2 :	PC1= A2 $\wedge$ (T[1] == 0)	$\rightarrow$	PC1 := A3 ,
T3 :	PC1= A3	$\rightarrow$	PC1 := A4 ,
T4 :	PC1= A4	$\rightarrow$	(PC1, T[0]) := (A1, 0) ,
T5 :	PC2= B1	$\rightarrow$	(PC2, T[1]) := (B2, 1) ,
T6 :	PC2= B2 $\wedge$ (T[0] == 0)	$\rightarrow$	PC2 := B3 ,
T7 :	PC2= B3	$\rightarrow$	PC2 := B4 ,
T8 :	PC2= B4	$\rightarrow$	(PC2, T[1]) := (B1, 0) ,

}

p $\equiv$ (PC1 = A1) $\wedge$ (PC2 = B1) $\wedge$ (T = [0,0])

(β) Ακολουθεί το σύστημα μεταβάσεων όπου οι τριάδες (A,B,[x,y]) αναφέρονται στις τιμές των μεταβλητών (PC1, PC2 και T) που ισχύουν στην κάθε κατάσταση.



(γ) Για να είναι κατάλληλος ο αλγόριθμος για διασφάλιση αμοιβαίου αποκλεισμού, θα πρέπει να είναι αδύνατο και οι δύο διεργασίες να βρίσκονται ταυτόχρονα στα σημεία του κώδικα A_3 και B_3. Δηλαδή δεν θα πρέπει να υπάρχει κατάσταση στο σύστημα μεταβάσεων, όπου η μεταβλητή PC1 = A_3 και η PC2 = B_3. Από τη μελέτη του συστήματος μεταβάσεων στο μέρος (β) οδηγούμαστε στο συμπέρασμα ότι είναι αυτό δεν είναι δυνατό. Επομένως, ο αλγόριθμος διασφαλίζει αμοιβαίο αποκλεισμό. Εντούτοις, ο αλγόριθμος είναι δυνατό να εμφανίσει αδιέξοδα: επιτρέπει στις διεργασίες να εισέλθουν σε μια κατάσταση όπου T=[1,1] από όπου καμιά διεργασία δεν είναι σε θέση να εισέλθει στο κρίσιμό της τμήμα αφού οι συνθήκες των μεταβάσεων T2 και T6 δεν ικανοποιούνται.

Λύση Άσκησης 2

Χρησιμοποιούμε τις επιπλέον μεταβλητές PC1, PC2, (program counters) οι οποίες παίρνουν ως τιμές ονόματα των γραμμών του κώδικα όπως φαίνεται πιο κάτω.

```
left
     l1: if (y1=n-k) then stop;
     l2: y3 := y3*y1;
     l3: y1--;
     l4: goto l1
||

right
     r1: if (y2=k) then stop;
     r2: y2++;
     r3: await y2 <= n-y1
     r4: y3 := y3/y2;
     r5: goto r1;
```

Ο γράφος προγράμματος δίνεται από την πλειάδα (V, S , T, p) όπου

V = {PC1, PC2, y1,y2,y3,n,k}

S = σύνολο όλων των δυνατών αναθέσεων τιμών στις πιο πάνω μεταβλητές

T = {

T1 :	PC1= l1, y1 = n-k	$\rightarrow$	stop ,
T2 :	PC1= l1, y1 != n-k	$\rightarrow$	PC1 := l2 ,
T3 :	PC1= l2	$\rightarrow$	(PC1, y3) := (l3 , y3*y1) ,
T4 :	PC1= l3	$\rightarrow$	(PC1,y1 := (l4 , y1-1) ,
T5 :	PC1= l4	$\rightarrow$	PC1 := l1 ,
T6 :	PC2= r1, y2 = k	$\rightarrow$	stop ,
T7 :	PC2= r1, y2 != k	$\rightarrow$	PC2 := r2 ,
T8 :	PC2= r2	$\rightarrow$	(PC2, y2) := (r3, y2+1) ,
T9 :	PC2= r3, y2 <= n-y1	$\rightarrow$	PC2 := r4 ,
T10 :	PC2= r4	$\rightarrow$	(PC2,y3) : = (r5,y3/y2) ,
T11 :	PC2= r5	$\rightarrow$	PC2 := r1

}

p = $y1 = n \wedge y2 = 0 \wedge y3 = 1 \wedge PC1 = l1 \wedge PC2 = r1$,   n>0, k>0