Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχ. και Μηχανικών Υπολογιστών
Εργαστήριο Υπολογιστικών Συστημάτων

Παρουσίαση 1ης Άσκησης:

Ανάπτυξη παράλληλου κώδικα σε πολυπύρηνες αρχιτεκτονικές κοινής μνήμης

Ακ. Έτος 2020-2021

Συστήματα Παράλληλης Επεξεργασίας

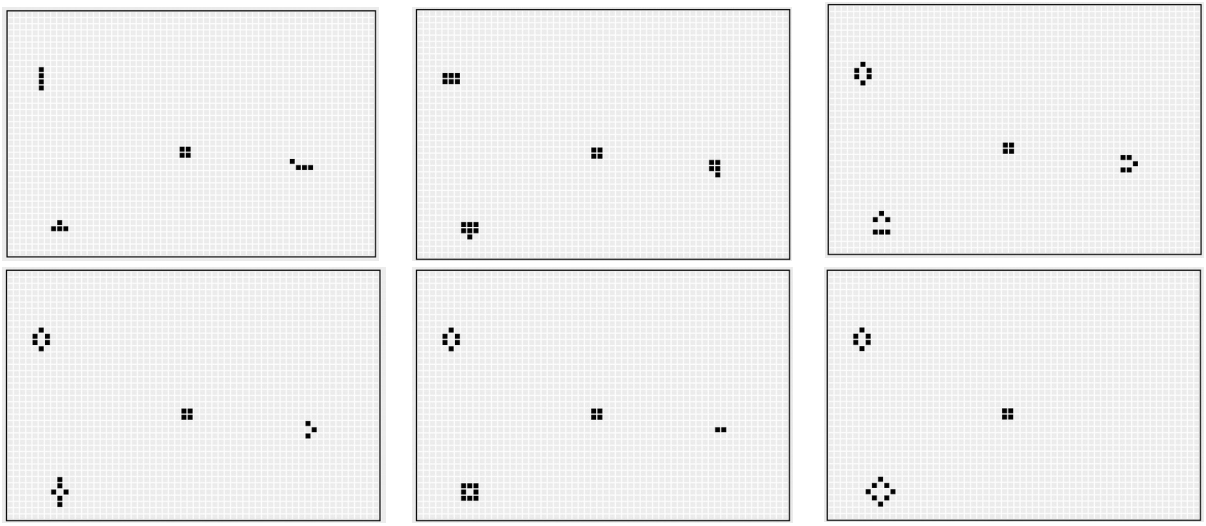
9ο Εξάμηνο

National Technical University of Athens
CSLab

Conway's Game Of Life

- Το Conway's Game of Life είναι παράδειγμα ενός κυψελικού αυτόματου (cellular automaton)
  - Σε ένα ορθογώνιο ταμπλώ, κάθε κελί έχει δύο πιθανές καταστάσεις: μπορεί να είναι *ζωντανό* ή *νεκρό*
  - Σε κάθε χρονικό βήμα/γενιά κάθε κελί εξετάζει τους γείτονές του και ενημερώνει την κατάστασή του:
    - Ένα *ζωντανό* κελί *πεθαίνει* από μοναξιά αν έχει λιγότερους από 2 ζωντανούς γείτονες
    - Ένα *ζωντανό* κελί *επιβιώνει* αν έχει 2 ή 3 ζωντανούς γείτονες
    - Ένα *ζωντανό* κελί *πεθαίνει* από υπερπληθυσμό (ή αγοραφοβία ☺) αν έχει περισσότερους από 3 ζωντανούς γείτονες
    - Ένα *νεκρό* κελί με ακριβώς 3 ζωντανούς γείτονες γίνεται *ζωντανό* λόγω αναπαραγωγής

Conway’s Game Of Life



- Εξαρτήσεις από τις τιμές των 8 γειτονικών κελιών, κατά την προηγούμενη χρονική στιγμή
- Ζητούμενα:
  - Παραλληλοποίηση αλγορίθμου στο OpenMP
  - Μέτρηση χρόνου εκτέλεσης σε 1, 2, 4, 6, 8 πυρήνες

Αλγόριθμος Floyd-Warshall (FW)

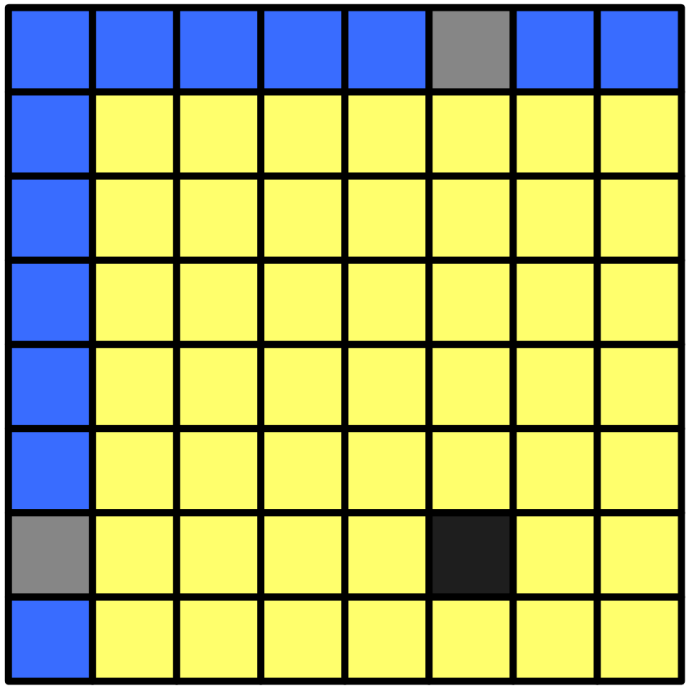
- Εύρεση **ελάχιστου μονοπατιού** ανάμεσα σε οποιοδήποτε ζεύγος κόμβων ενός κατευθυνόμενου γράφου (τα βάρη των ακμών μπορούν να είναι και αρνητικά).

```
for (k=0; k<N; k++)
   for (i=0; i<N; i++)
      for (j=0; j<N; j++)
         A[i][j] = min(A[i][j], A[i][k]+A[k][j]);
```

- Για κάθε χρονικό βήμα ***k*** υπολογίζει για κάθε ζεύγος κόμβων ***i-j*** αν υπάρχει συντομότερο μονοπάτι από τον i προς τον j περνώντας από το κόμβο k
- N: αριθμός κόμβων του γράφου
- A: πίνακας διπλανών κορυφών (αν i,j δεν συνδέονται τότε A[i][j] = ∞ αρχικά)
- Πολυπλοκότητα: $\Theta(n^3)$

Παράδειγμα: γράφος 8 κόμβων

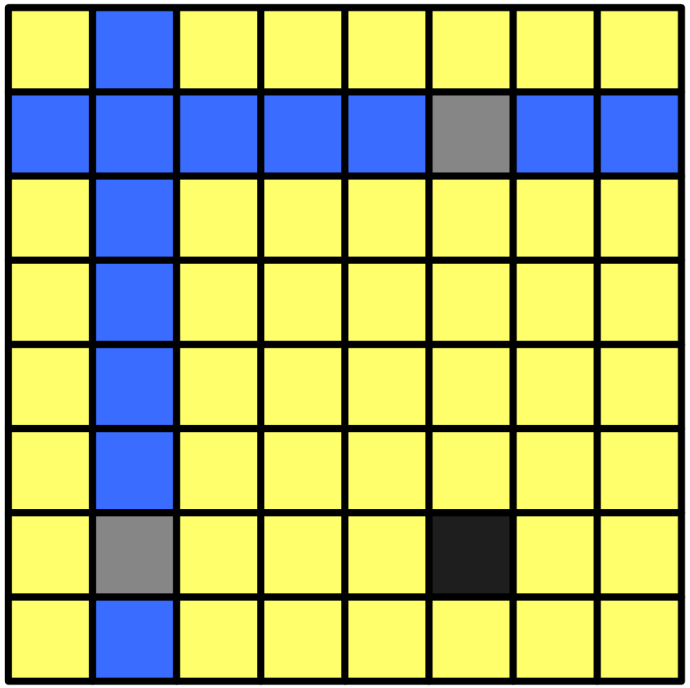
k=0



```
A[i][j] = min(A[i][j], A[i][0] + A[0][j])
```

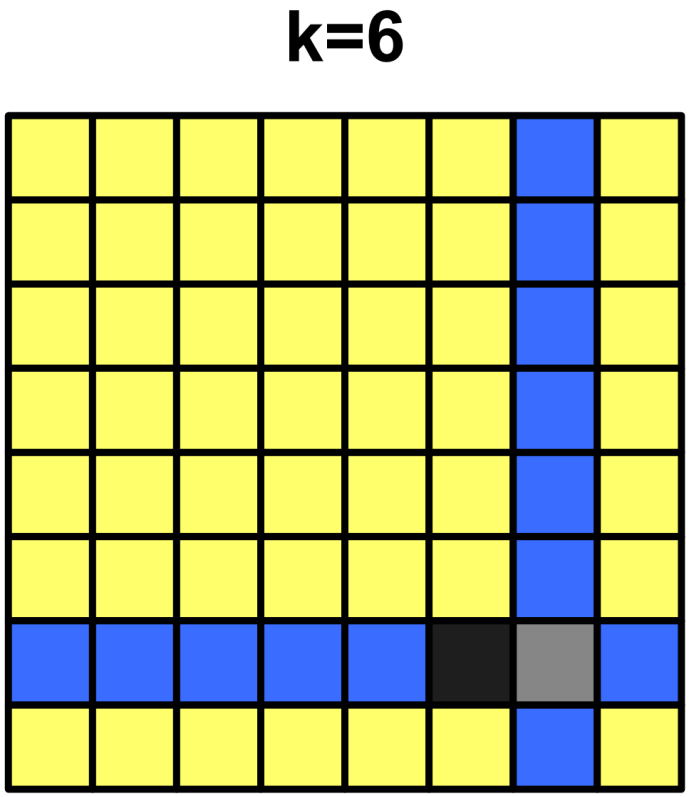
Παράδειγμα: γράφος 8 κόμβων

k=1



```
A[i][j] = min(A[i][j], A[i][1] + A[1][j])
```

Παράδειγμα: γράφος 8 κόμβων

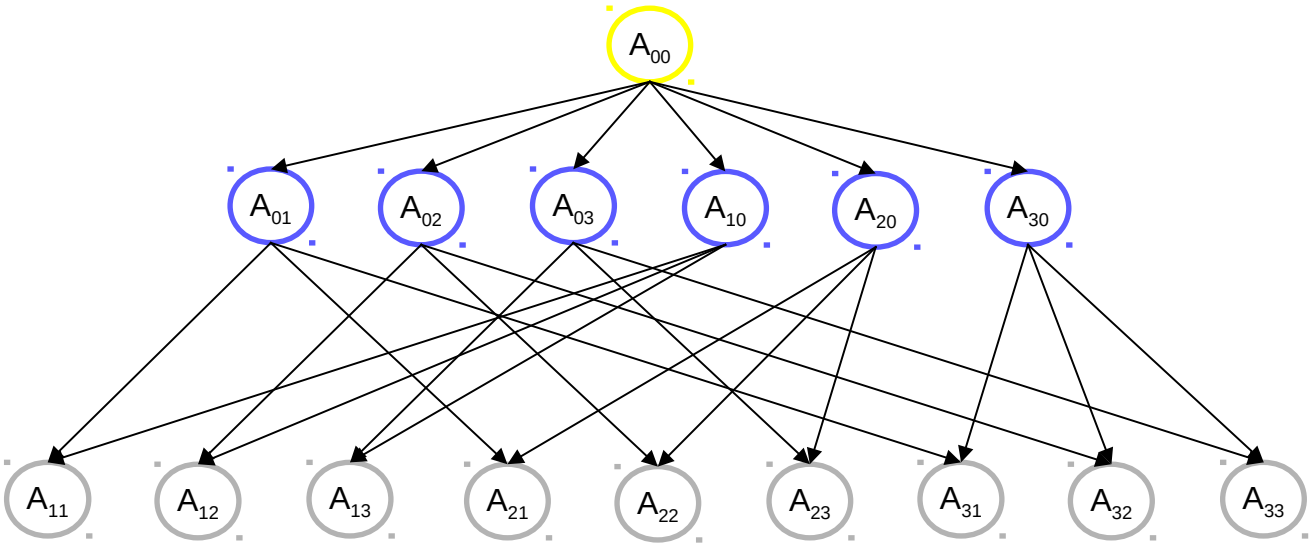



```
A[i][j] = min(A[i][j], A[i][6] + A[6][j])
```

FW Task graph

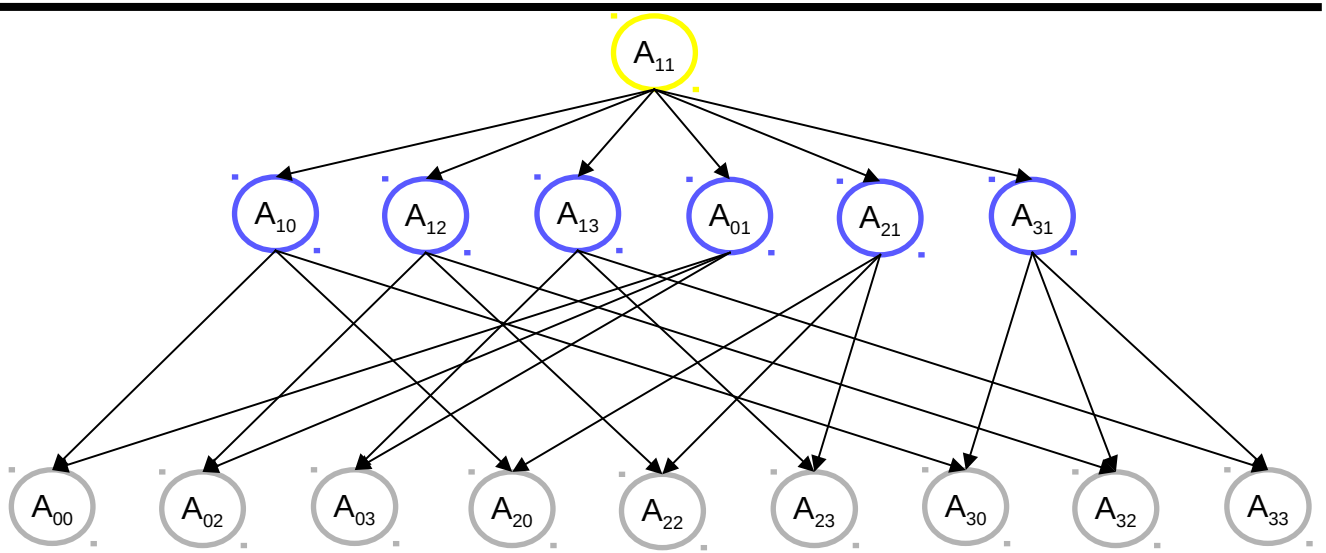
k=0

A_{00}	A_{01}	A_{02}	A_{03}
A_{10}	A_{11}	A_{12}	A_{13}
A_{20}	A_{21}	A_{22}	A_{23}
A_{30}	A_{31}	A_{32}	A_{33}



k=1

A_{00}	A_{01}	A_{02}	A_{03}
A_{10}	A_{11}	A_{12}	A_{13}
A_{20}	A_{21}	A_{22}	A_{23}
A_{30}	A_{31}	A_{32}	A_{33}



Σειριακή και παράλληλη εκτέλεση

- Για μεγάλα N (ο A δεν χωράει στην cache), ο FW είναι memory bound:
  - Ο πίνακας A πρέπει να μεταφέρεται από την κύρια μνήμη σε κάθε επανάληψη k
  - Οι πράξεις που γίνονται είναι πολύ απλές (σύγκριση / πρόσθεση) σε ακέραιους ή πραγματικούς απλής ακρίβειας
- Παράλληλη εκτέλεση:
  - Τα loops i, j είναι παράλληλα
  - Ο αλγόριθμος δεν κλιμακώνει καλά σε αρχιτεκτονικές κοινής μνήμης

Εναλλακτικές υλοποιήσεις: recursive

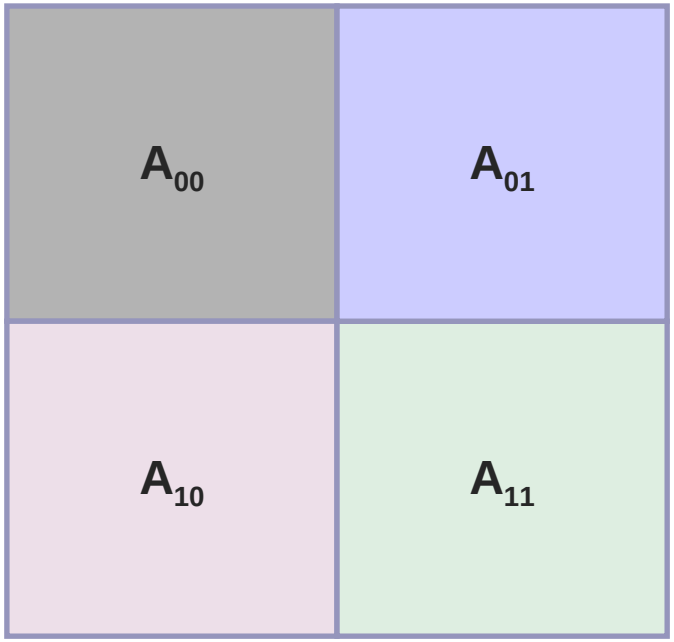
- J.-S. Park, M. Penner, and V. K. Prasanna, **“Optimizing Graph Algorithms for Improved Cache Performance,”** *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 15, NO. 9, SEPTEMBER 2004.*

```
FWR (A, B, C)
  if (base case)
    FWI (A, B, C)
  else
    FWR (A11, B11, C11);
    FWR (A12, B11, C12);
    FWR (A21, B21, C11);
    FWR (A22, B21, C12);
    FWR (A22, B21, C12);
    FWR (A21, B21, C11);
    FWR (A12, B11, C12);
    FWR (A11, B11, C11);
```

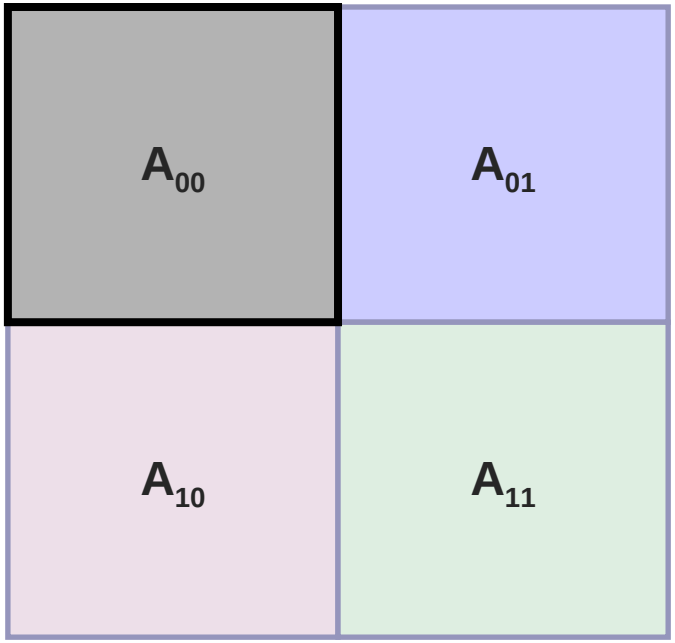
```
FWI (A, B, C)
for (k=0; k<N; k++)
   for (i=0; i<N; i++)
      for (j=0; j<N; j++)
         A[i][j] = min(A[i][j], B[i][k]+C[k][j]);
```

- Καλείται ως: FWR(A, A, A);

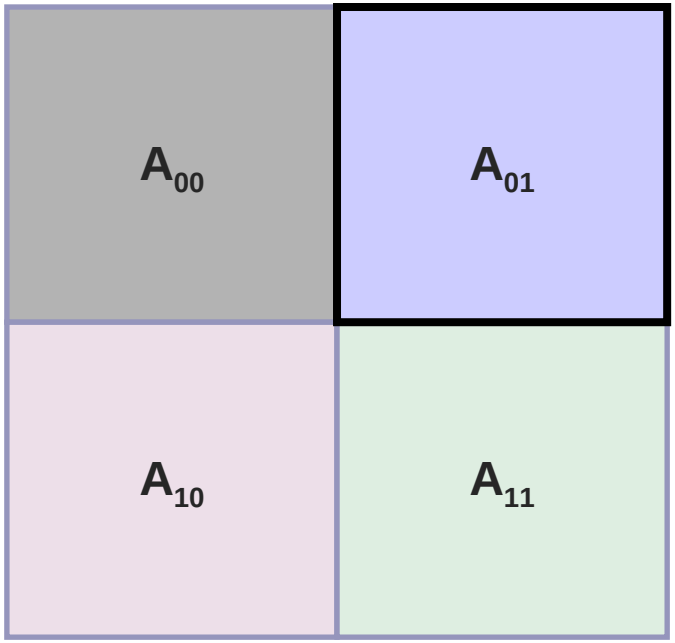
Εναλλακτικές υλοποιήσεις: recursive



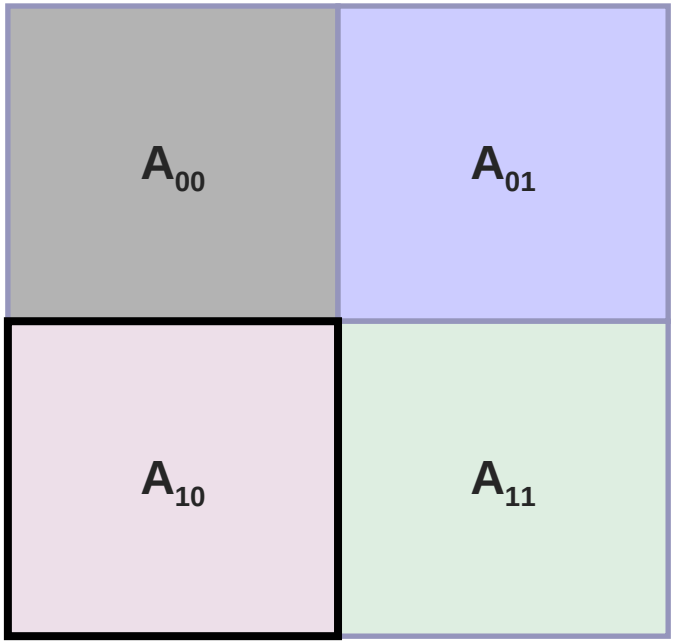
Εναλλακτικές υλοποιήσεις: recursive



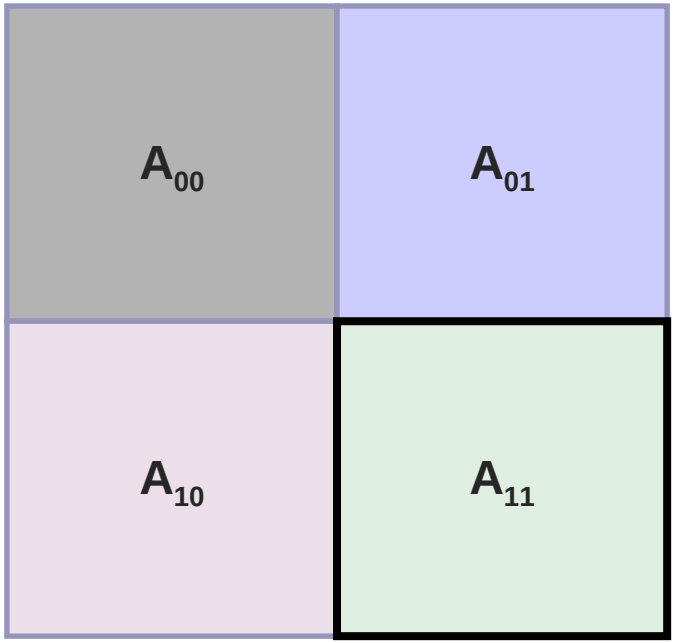
Εναλλακτικές υλοποιήσεις: recursive



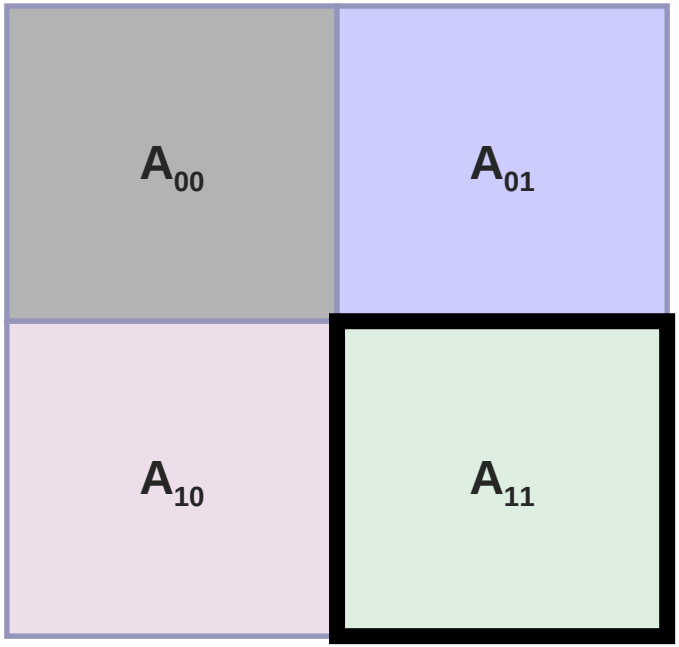
Εναλλακτικές υλοποιήσεις: recursive



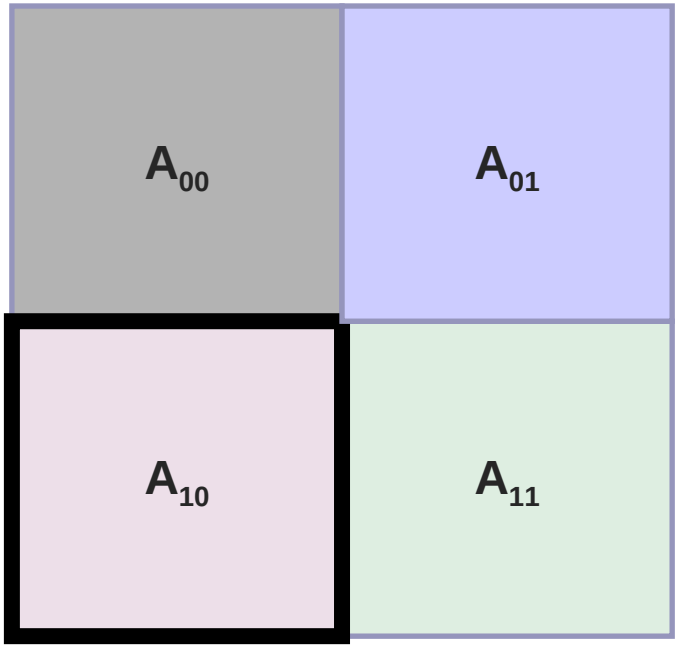
Εναλλακτικές υλοποιήσεις: recursive



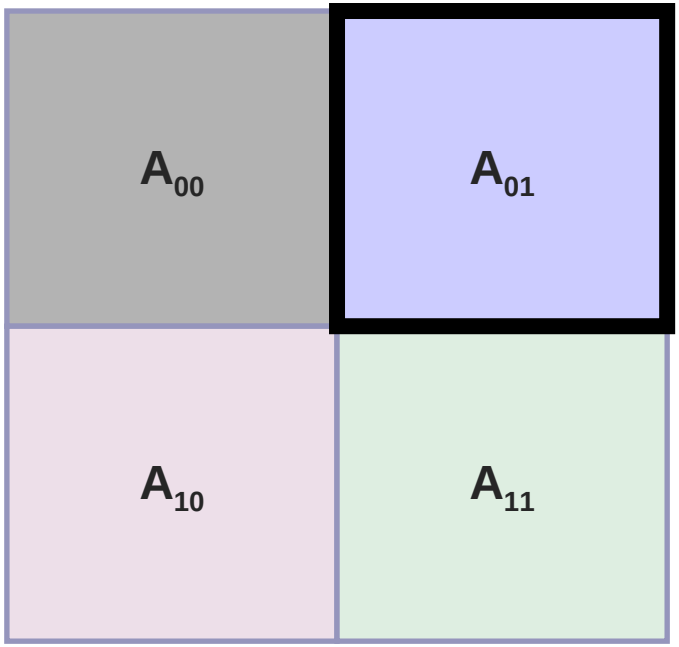
Εναλλακτικές υλοποιήσεις: recursive



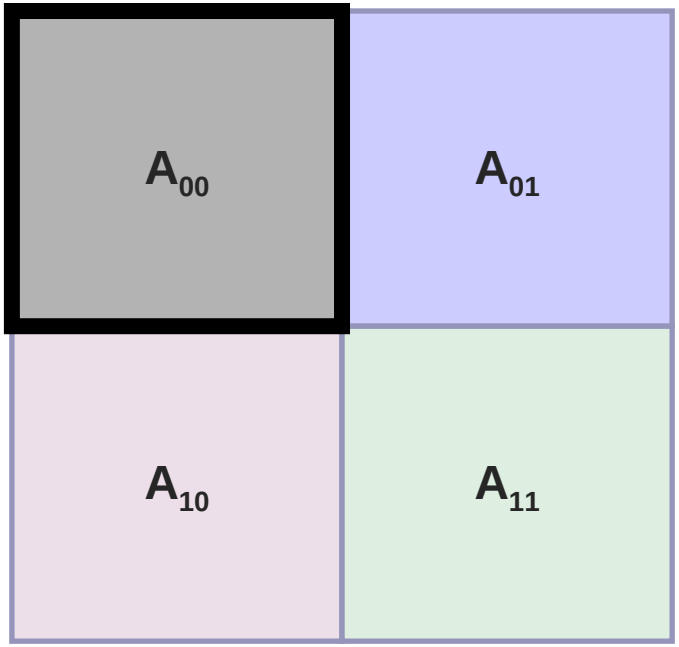
Εναλλακτικές υλοποιήσεις: recursive



Εναλλακτικές υλοποιήσεις: recursive



Εναλλακτικές υλοποιήσεις: recursive



Εναλλακτικές υλοποιήσεις: recursive

```
FWR (A, B, C)
  if (base case)
    FWI (A, B, C)

  else
```

```
FWR (A00, B00, C00);
```
1

```
FWR (A01, B00, C01);

FWR (A10, B10, C00);
```
2

```
FWR (A11, B10, C01);
```
3

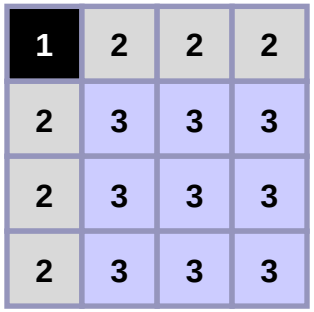
```
FWR (A11, B10, C01);
```
4

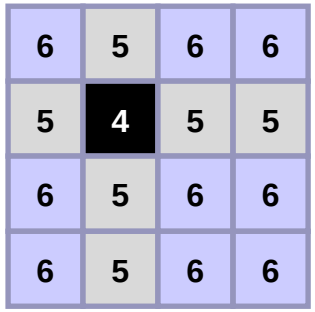
```
FWR (A10, B10, C00);

FWR (A01, B00, C01);
```
5

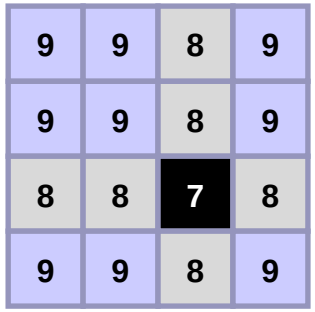
```
FWR (A00, B00, C00);
```
6

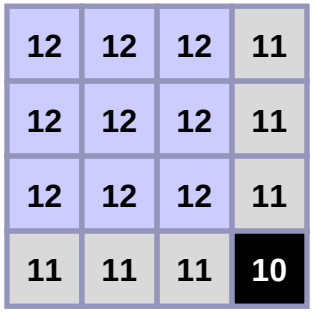
Παραλληλία

Εναλλακτικές υλοποιήσεις: tiled



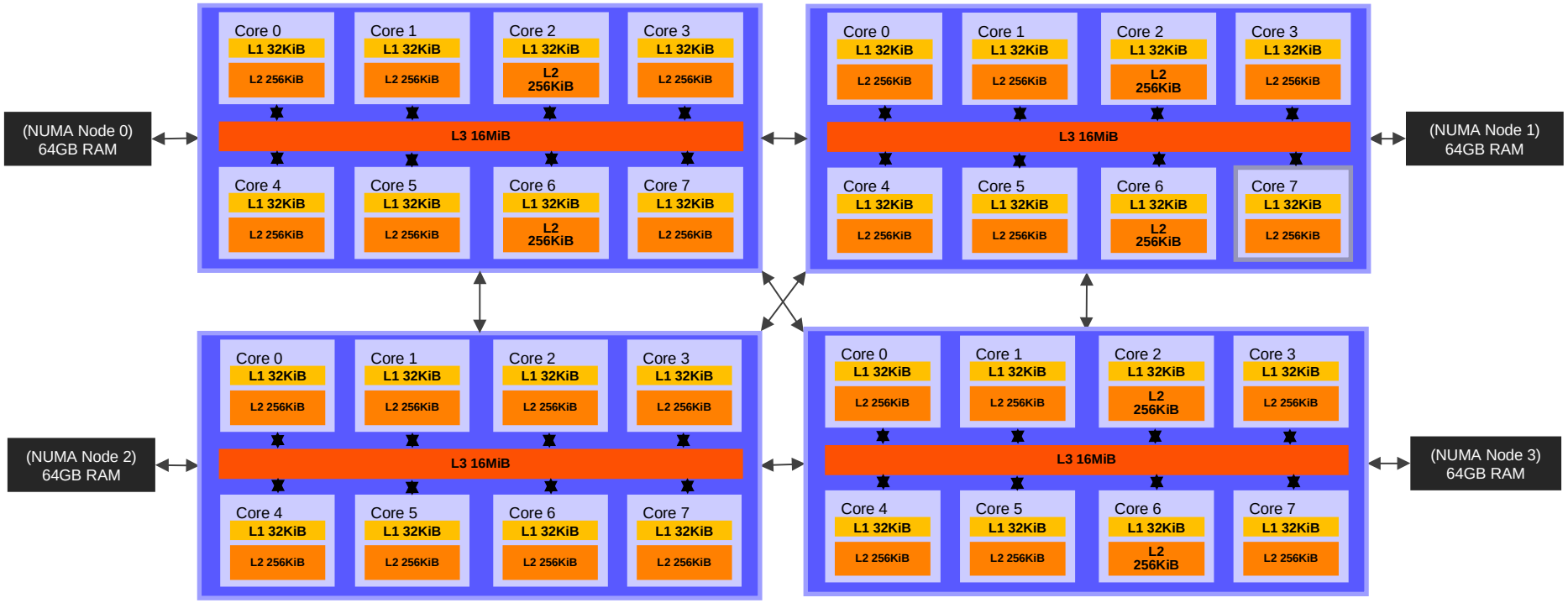






Περιβάλλον εκτέλεσης

- *sandman*: 4 x Intel Xeon E5-4620 (Sandy Bridge)
  - Συνολικά 32 πυρήνες (και 64 threads)



Οδηγίες

- Για χρήση του sandman:

  $ qsub –q serial –l nodes=sandman:ppn=64 <script>

- **ΠΡΟΣΟΧΗ**: Χρησιμοποιείτε τα μηχανήματα της ουράς **parlab** για ανάπτυξη προγραμμάτων/έλεγχο ορθής λειτουργίας/εκσφαλμάτωση
- Μπορείτε να χρησιμοποιήσετε OpenMP ή TBBs για την εκπόνηση της άσκησης
- Μπορείτε να χρησιμοποιείτε τα μηχανήματα της ουράς parlab για την ανάπτυξη του παράλληλου κώδικα
- Θα βρείτε τον κώδικα της άσκησης στον scirouter στο path:

  /home/parallel/pps/2020-2021/a1/FW-serial

- Θα βρείτε παραδείγματα και οδηγίες μεταγλώττισης/εκτέλεσης για τα TBBs στον scirouter στο path:

  /home/parallel/pps/2020-2021/a1/tbb-workspace