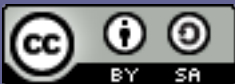


Work – Span

(και οι περιπτώσεις των λειτουργιών map και reduce)

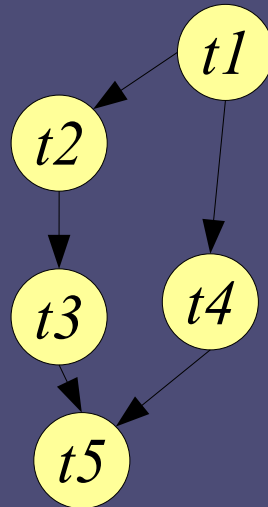
<https://mixstef.github.io/courses/parprog/>

Μ.Στεφανιδάκης



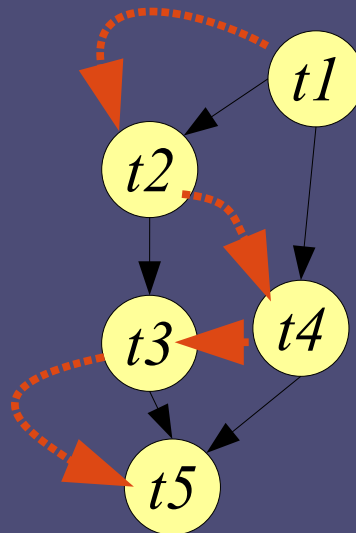
Ορισμός Work και Span

- Η παράλληλη εκτέλεση γίνεται με την ολοκλήρωση tasks
 - Ακολουθώντας τη ροή των αλληλεξαρτήσεων των δεδομένων
 - Ένας κατευθυνόμενος μη κυκλικός γράφος (DAG)



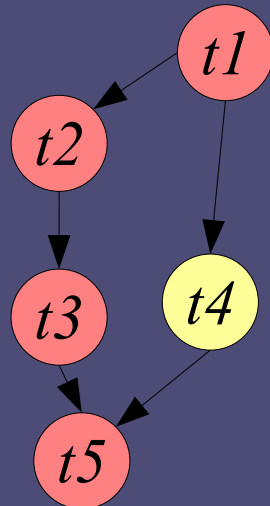
Ορισμός Work και Span

- T_1 είναι ο χρόνος σειριακής εκτέλεσης (work)
 - Μια οποιαδήποτε **έγκυρη** σειριοποίηση της δουλειάς που πρέπει να γίνει



Ορισμός Work και Span

- T_{∞} είναι ο χρόνος σε ένα ιδανικά παράλληλο σύστημα (span)
 - Διαθέσιμα **άπειρα** επεξεργαστικά στοιχεία
 - Η καλύτερη περίπτωση παραλληλίας
 - Το όριο είναι το **κρίσιμο μονοπάτι** των tasks (critical path)
 - Το μεγαλύτερο σε διάρκεια μεταξύ αρχής – τέλους



Ανάλυση Work – Span

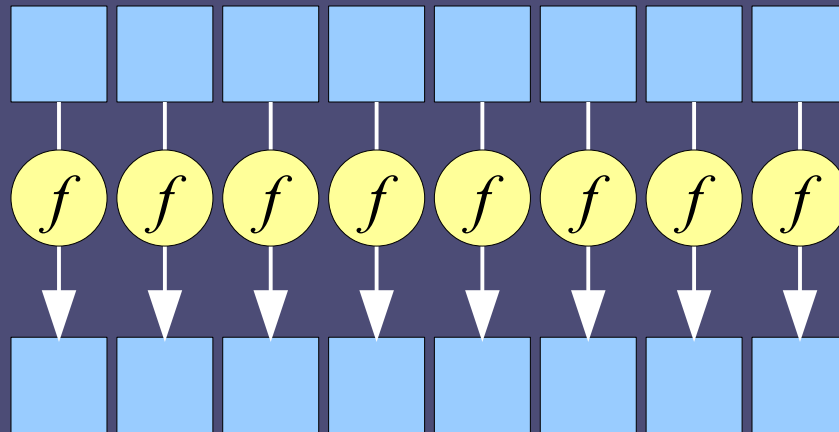
Αν T_p είναι ο χρόνος σε σύστημα με P επεξεργαστικά στοιχεία, τότε

$$T_p = O(T_1 / P + T_\infty)$$

- Το T_∞ εμποδίζει την επεκτασιμότητα
- Η αύξηση του T_1 επιβαρύνει την απόδοση
 - Συνεπώς η σχεδίαση των παράλληλων αλγορίθμων θα πρέπει να αποσκοπεί στην μείωση του T_∞ (span)
 - Αποφεύγοντας την υπέρμετρη αύξηση του T_1 (work), εκτός κι αν αυτό μειώνει δραστικά το span

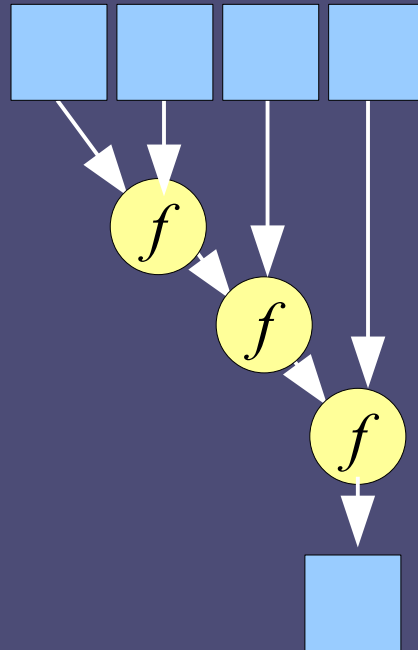
Η λειτουργία map ξανά

- Εφαρμογή μιας συνάρτησης σε κάθε στοιχείο μιας ακολουθίας δεδομένων
 - $Work = O(n)$
 - $Span = O(1)$ (αν η f έχει σταθερό κόστος)



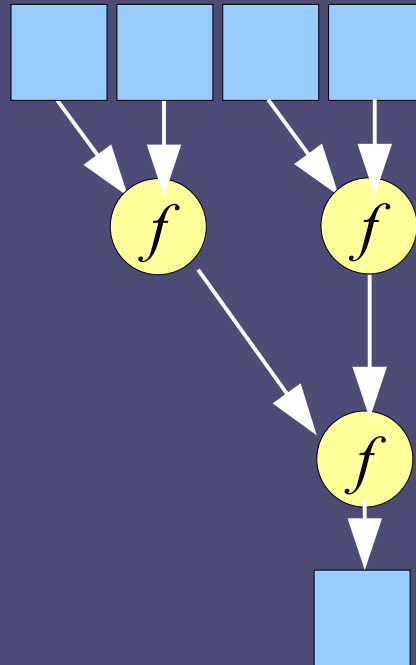
Η λειτουργία reduce

- Συνδυάζει όλα τα στοιχεία μιας συλλογής (collection) σε ένα μοναδικό στοιχείο μέσω τελεστή f



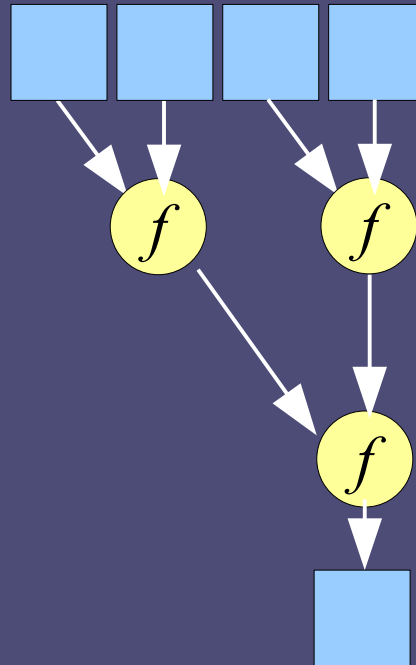
Παραλληλοποίηση της reduce

- Δεν είναι πάντα δυνατή
 - Θα πρέπει ο τελεστής f να είναι προσεταιριστικός
 - Όταν $((x1 \circ x2) \circ x3) \circ x4 = (x1 \circ x2) \circ (x3 \circ x4)$



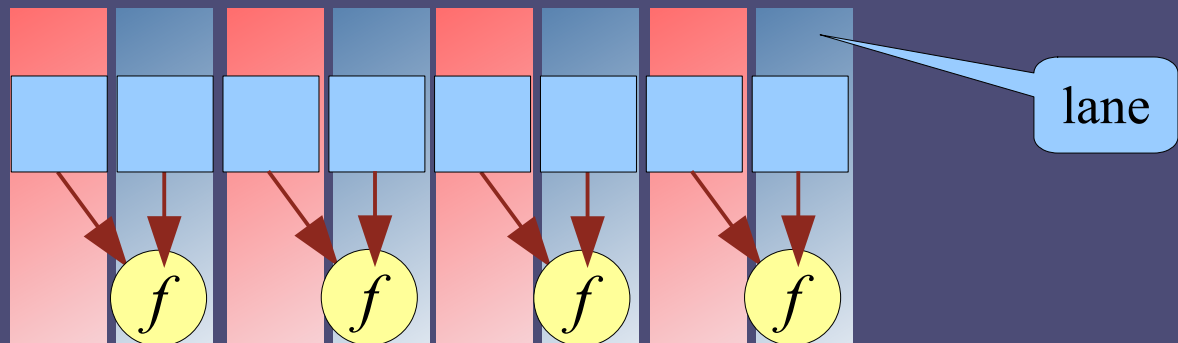
Reduce: Work και Span

- Στην ιδανική περίπτωση
 - $Work = O(n)$ – όσο και η σειριακή εκδοχή
 - $Span = O(\log n)$



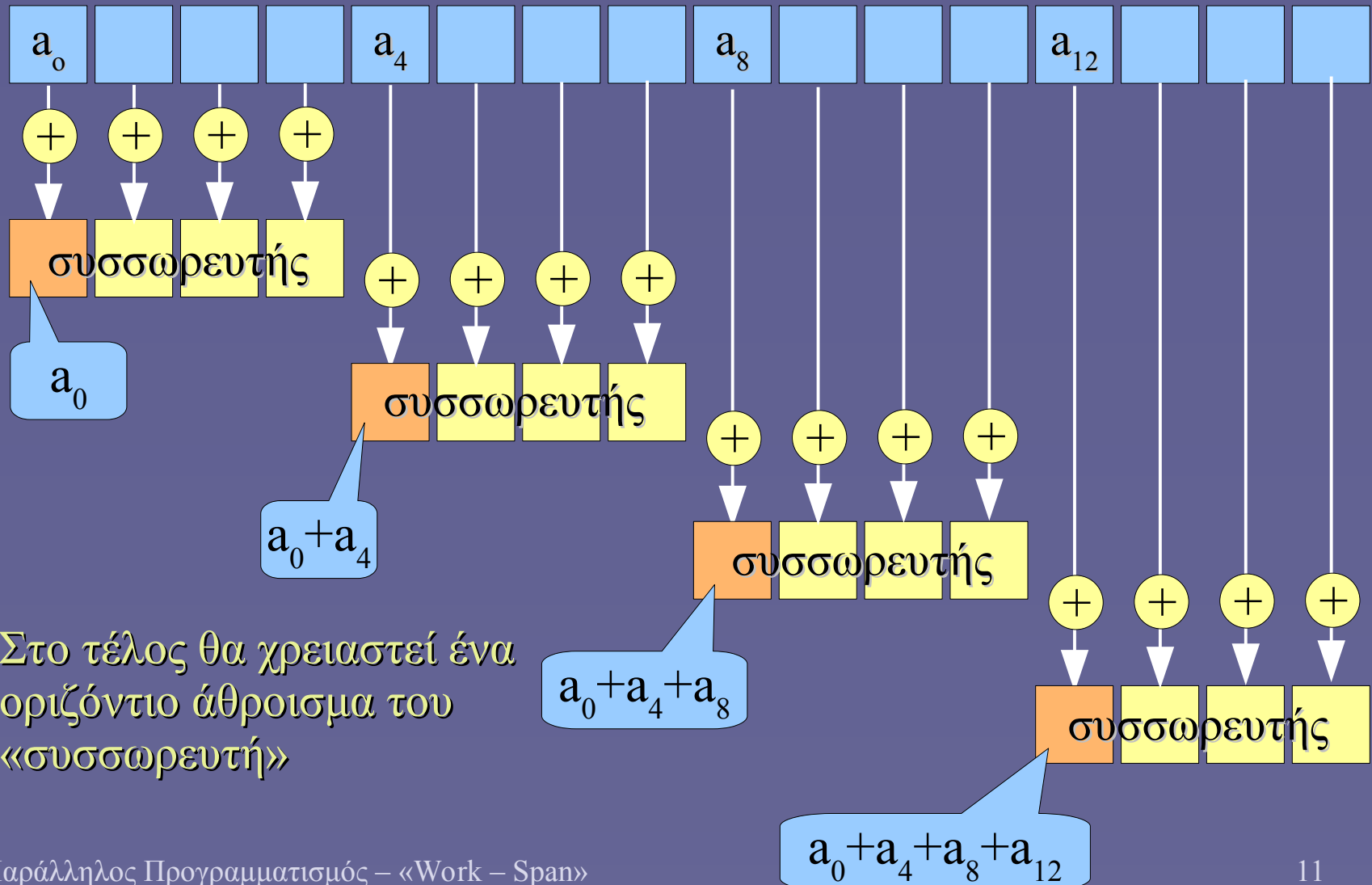
Reduce και εντολές SSE/AVX

- Για την άμεση υλοποίηση θα ήταν απαραίτητες εντολές με μη-κάθετες πράξεις
 - «Οριζόντιες» λειτουργίες
 - Διάσχιση κάθετων λωρίδων (**cross-lane operations**)
 - Μη αποδοτικές πράξεις σε σχέση με τις «κάθετες»
 - Πολλές εντολές έχουν περιορισμούς στη διάσχιση λωρίδων
 - Χρειάζεται υλοποίηση με όσο το δυνατόν περισσότερες «κάθετες» πράξεις
 - Ιδίως στα κρίσιμα για την απόδοση σημεία του κώδικα



Παράδειγμα reduction: Άθροισμα πίνακα

(έστω εύρος πράξης = 4)



Στο τέλος θα χρειαστεί ένα οριζόντιο άθροισμα του «συσσωρευτή»

Πώς αθροίζω «οριζόντια» μια ποσότητα `__m256`;

- Ποσότητα `__m256 = 8 packed floats`
 - h g f e d c b a
 - Το αποτέλεσμα πρέπει να είναι **ένας και μοναδικός** αριθμός float = h + g + f + e + d + c + b + a
- 1η μέθοδος
 - Αποθηκεύουμε την δάδα σε ένα array 8 floats στη μνήμη
 - Στη συνέχεια αθροίζουμε σειριακά τους 8 αριθμούς

```
float partial_sums[8], result;  
  
// move 8-float __m256 accumulator to float array, unaligned  
_mm256_storeu_ps(partial_sums, accumulator);  
  
// serially add 8 parts to final result  
result = 0.0;  
for (int i=0; i<8; i++)  
    result += partial_sums[i];
```

Εναλλακτική «οριζόντια» άθροιση __m256

```
// horizontal sum of 8 packed floats, let *pa = h g f e d c b a
// add high and low 128-bit halves into a __m128 number t1 = h+d g+c f+b e+a
__m128 t1 = _mm_add_ps(_mm256_extractf128_ps(*pa,1),_mm256_castps256_ps128(*pa));
// create t2, a shuffled version of t1,      t2 = g+c h+d e+a f+b
__m128 t2 = _mm_shuffle_ps(t1,t1,_MM_SHUFFLE(2,3,0,1));
// add t1 and t2, result t3 = h+d+g+c g+c+h+d f+b+e+a e+a+f+b
__m128 t3 = _mm_add_ps(t1,t2);
// set t2 to upper parts of t2 and t3, now t2 = g+h h+d h+d+g+c g+c+h+d
t2 = _mm_movehl_ps(t2,t3);
// add lowest parts of t2 and t3, now t3 = g+h h+d h+d+g+c g+c+h+d+e+a+f+b
t3 = _mm_add_ss(t2,t3);
// copy to a single float
fsum = _mm_cvtss_f32(t3);
```

Βιβλιογραφία

- Michael McCool, James Reinders, and Arch Robison. 2012. *Structured Parallel Programming: Patterns for Efficient Computation* (1st ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- Intel® Intrinsic Guide (<https://software.intel.com/sites/landingpage/IntrinsicGuide/>)