ΕΠΙΣΤΗΜΗ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

Μηχανική μάθηση, αρχιτεκτονικές και επίδοση

Ακαδημαϊκό έτος 2018-19

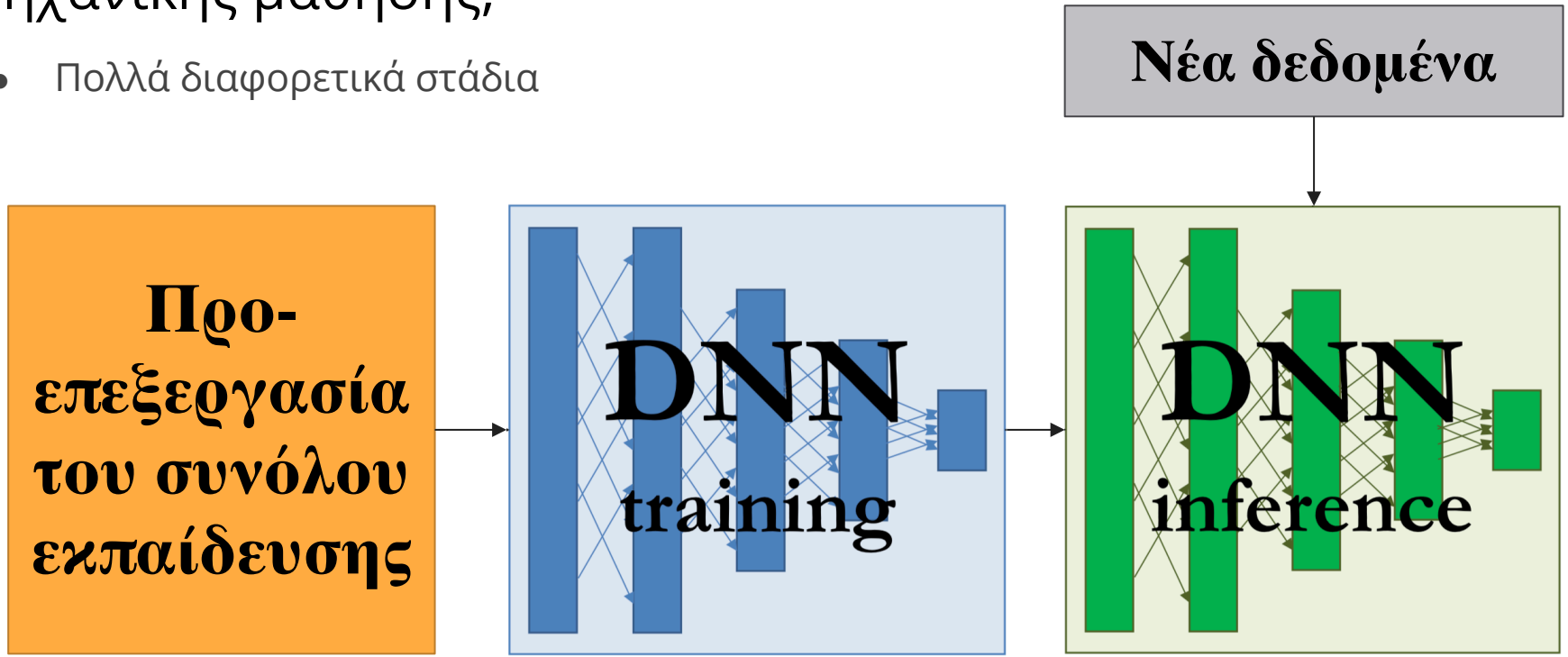
Αρχιτεκτονικές για μηχανική μάθηση

- Στη σύγχρονη μηχανική μάθηση, χρησιμοποιούνται οι παράλληλες αρχιτεκτονικές που έχουμε δει στα προηγούμενα μαθήματα
  - CPUs
  - GPUs
  - ...και άλλες αρχιτεκτονικές (TPUs, FPGAs, ASICs)

- Οι GPUs κυριαρχούν στη μηχανική μάθηση!

Με τι μοιάζει ένα pipeline μηχανικής μάθησης;

- Πολλά διαφορετικά στάδια



Πού βοηθούν οι σύγχρονες αρχιτεκτονικές

- Τόσο οι παράλληλοι επεξεργαστές, όσο και οι κάρτες γραφικών χρησιμοποιούνται στα διαφορετικά στάδια ενός pipeline μηχανικής μάθησης

- Τα διαφορετικά στάδια προωθούν:
  - Τη **χρήση διαφορετικών σύγχρονων αρχιτεκτονικών**
  - Τη **σχεδίαση νέων αρχιτεκτονικών** (για κάθε στάδιο)

- Τι θέλουμε να πετύχουμε;
  - Inference με μικρό χρόνο απόκρισης
  - Training με μεγαλύτερο ρυθμό επεξεργασίας
  - Χαμηλότερο κόστος κατανάλωσης ισχύος

Πού βοηθούν οι σύγχρονες/νέες αρχιτεκτονικές;

- Επιταχύνουν τους **βασικούς υπολογιστικούς πυρήνες** των υπολογισμών της μηχανικής μάθησης
  - Βασικός υπολογιστικός πυρήνας: **πολλαπλασιασμός πίνακα με πίνακα** (matrix-matrix multiply)
  - Άλλος βασικός υπολογιστικός πυρήνας: **συνέλιξη** (convolution)
- Προσθέτουν **μονοπάτια δεδομένων** (μονοπάτια μνήμης) ειδικά για τη μηχανική μάθηση
  - Παράδειγμα: κρυφή μνήμη για αποθήκευση των βαρών του νευρωνικού δικτύου
- Δημιουργούν **λειτουργικές μονάδες** ειδικές στην εφαρμογή
  - Όχι γενικά για τη μηχανική μάθηση, αλλά για μια συγκεκριμένη εφαρμογή

Γιατί χρησιμοποιούνται οι GPUs στη μηχανική μάθηση;

Γιατί χρησιμοποιούνται οι GPUs *στην εκπαίδευση νευρωνικών δικτύων;*

CPUs vs GPUs

- Οι CPUs είναι επεξεργαστές **γενικού σκοπού**
  - Οι σύγχρονες CPUs κατά την επεξεργασία ξοδεύουν τον περισσότερο χρόνο προσπελαύνοντας κρυφές μνήμες
  - Οι σύγχρονες CPUs είναι ιδανικές για εφαρμογές με μη ομοιόμορφες ή τυχαίες προσβάσεις στη μνήμη

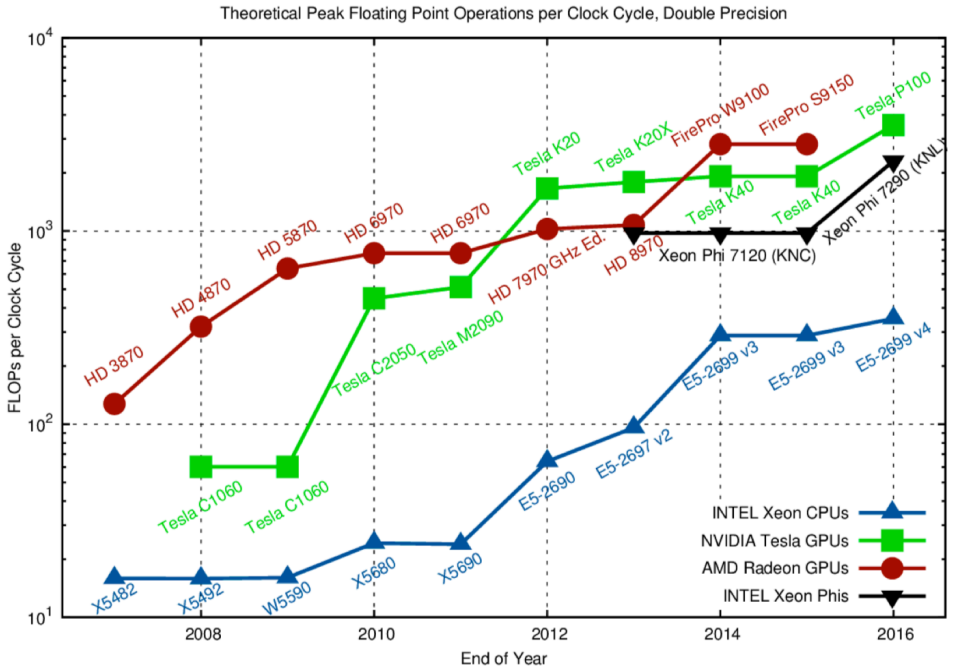
- Οι GPUs είναι επεξεργαστές **ειδικού σκοπού**
  - Οι GPUs είναι βελτιστοποιημένες για εφαρμογές μεγάλης υπολογιστικής έντασης
  - Οι GPUs είναι βελτιστοποιημένες για μοντέλα πρόσβασης στη μνήμη με ροή δεδομένων

Οι εφαρμογές μηχανικής μάθησης έχουν χαρακτηριστικά τέτοιου τύπου

FLOP/s: CPUs vs GPUs

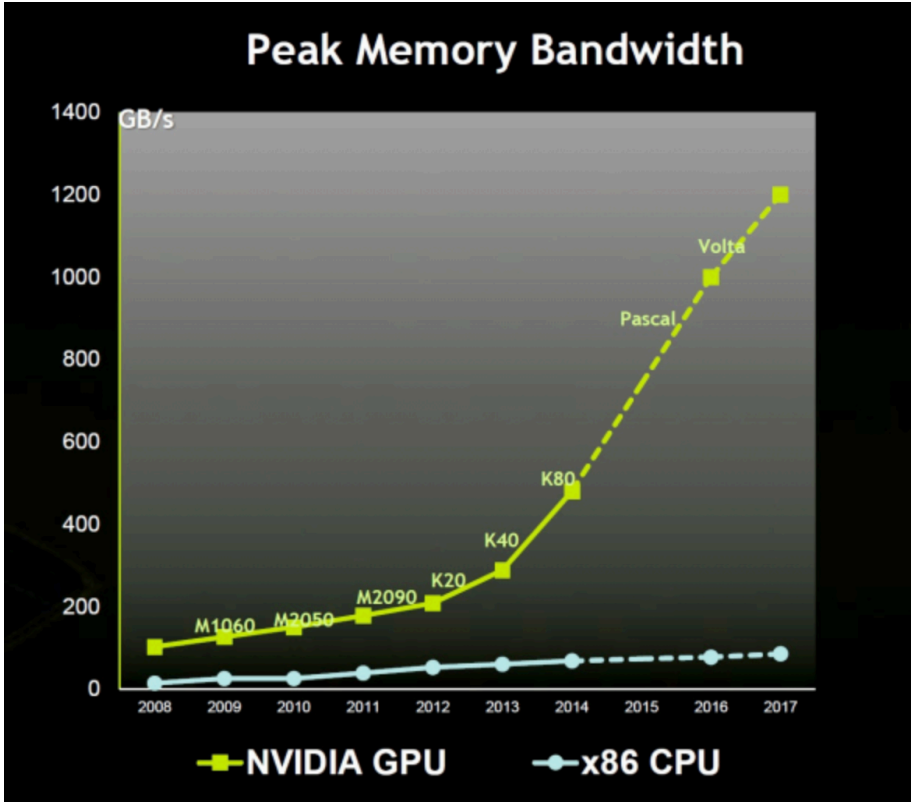
- FLOP/s: **F**loating **P**oint **Op**erations per second
  - Υπολογισμοί κινητής υποδιαστολής ανά δευτερόλεπτο
  - Μονάδα μέτρησης επίδοσης επεξεργαστών

- Οι GPUs επιτυγχάνουν περισσότερα FLOP/s από τις CPUs



Εύρος ζώνης από/προς τη μνήμη: CPUs vs GPUs

- Οι GPUs έχουν μεγαλύτερο εύρος ζώνης από/προς την κύρια μνήμη από τις CPUs
  - Εύρος ζώνης: ρυθμός μεταφοράς δεδομένων από τον επεξεργαστή προς την κύρια μνήμη και αντίστροφα
  - Ωστόσο, απαιτείται και μεταφορά δεδομένων από τη CPU στη GPU



CPUs vs GPUs:
Πώς αποφασίζω;

- Μοντέλα επίδοσης
  - Αν γνωρίζω πόσες πράξεις (κινητής υποδιαστολής) και πόσες προσβάσεις στη μνήμη κάνει ένας υπολογιστικός πυρήνας, μπορώ να αποφασίσω ποια αρχιτεκτονική είναι καλύτερη
  - Τροφοδοτώ αυτή την πληροφορία σε μοντέλα επίδοσης
  - Αρκεί να έχω άνω και κάτω όρια στην επίδοση

Το μοντέλο Roofline

Παραδείγματα υπολογισμού FLOPs και Bytes

Υπολογιστικός πυρήνας	FLOPs	Bytes
`for (i = 0 ; i < N ; i++)` `  z[i] = x[i];`	0 (καμία πράξη, μόνο αντιγραφή δεδομένων)	N (x 4 bytes) loads + N (x 4 bytes) stores
`for (i = 0 ; i < N ; i++)` `  z[i] = x[i] * y[i];`	N (μία πράξη – πολλαπλασιασμός)	2 x N (x 4 bytes) loads + N (x 4 bytes) stores

`for (i = 0 ; i < N ; i++)`
`  for (j = 0 ; j < N ; j++)`
`  y[i] = A[i][j] * x[j]`
	N^2
	
	(μία πράξη – πολλαπλασιασμός)
		N (x 4 bytes) + N^2 (x 4 bytes) loads +
		N (x 4 bytes) stores

Κλασικό μοντέλο πρόβλεψης

- Το κλασικό μοντέλο αγνοεί τις προσβάσεις στη μνήμη
  - Θεωρεί ότι ο επεξεργαστής μπορεί να τροφοδοτείται με δεδομένα από τη μνήμη χωρίς περιορισμούς

- Χρόνος εκτέλεσης = operations * processor speed
  - processor speed = seconds per operation

Κλασικό μοντέλο πρόβλεψης

- Έστω επεξεργαστής με επίδοση 500 MFLOPS
  - 500MFLOPS = 500 x 10^6 FLOP / sec
  - processor speed = 1 / (500 x 10^6) sec / FLOP = 2 x 10^{-9} sec/FLOP

Υπολογιστικός πυρήνας	**FLOPs**	**Χρόνος εκτέλεσης**
`for (i = 0 ; i < 1000 ; i++)` `  z[i] = x[i] * y[i];`	1000 (μία πράξη – πολλαπλασιασμός)	1000 FLOPs x 2 x 10^{-9} = 2 μsec

`for (i = 0 ; i < 1000 ; i++)`
`  for (j = 0 ; j < 1000 ; j++)`
`  y[i] = A[i][j] * x[j]`
	10^6
	
	(μία πράξη – πολλαπλασιασμός)
		10^6 FLOPs x 2 x 10^{-9}
		= 2 msec

Κλασικό μοντέλο πρόβλεψης

- Το κλασικό μοντέλο αγνοεί τις προσβάσεις στη μνήμη
  - Θεωρεί ότι ο επεξεργαστής μπορεί να τροφοδοτείται με δεδομένα από τη μνήμη χωρίς περιορισμούς

- Στην πραγματικότητα:
  - Οι προσβάσεις στη μνήμη είναι πιο ακριβές από τις πράξεις
  - Το εύρος ζώνης του διαδρόμου μνήμης είναι περιορισμένο
    - Ο ρυθμός με τον οποίο η μνήμη τροφοδοτεί με δεδομένα τον επεξεργαστή είναι περιορισμένος

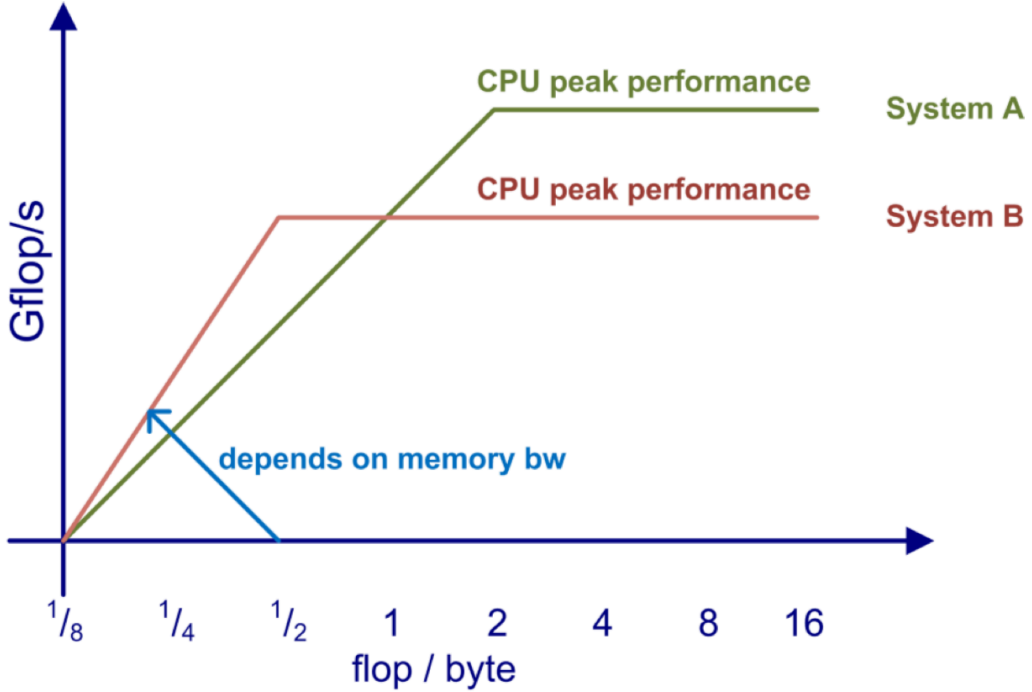
Το μοντέλο roofline

- Το μοντέλο roofline λαμβάνει υπόψην:
  - Το **εύρος ζώνης** (bandwidth-BW) σε **bytes/sec**
  - Την **ένταση σε υπολογισμούς** (operational intensity - OI) σε **operations/byte**

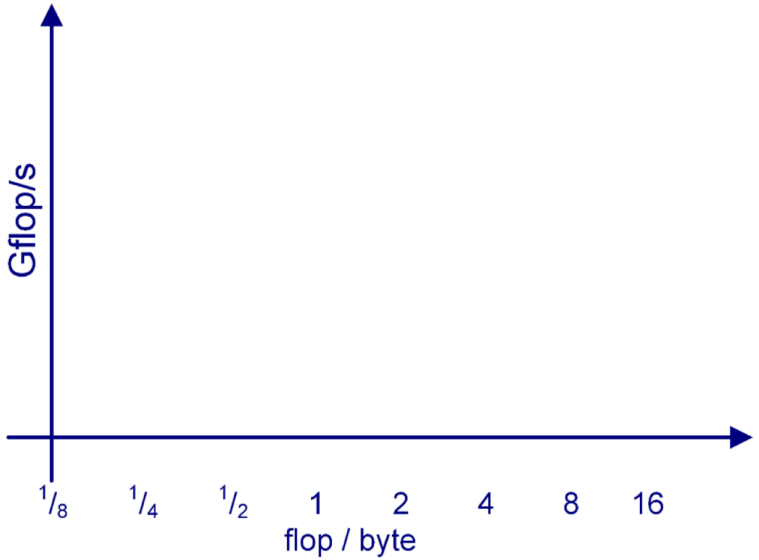
- Χρόνος εκτέλεσης = operations * **max**(processor speed, 1 / (OI * BW))
  - processor speed = seconds per operation
  - OI = operations / byte
  - BW = bytes / sec

Το μοντέλο roofline

- Το μοντέλο roofline συσχετίζει την εφαρμογή με την αρχιτεκτονική

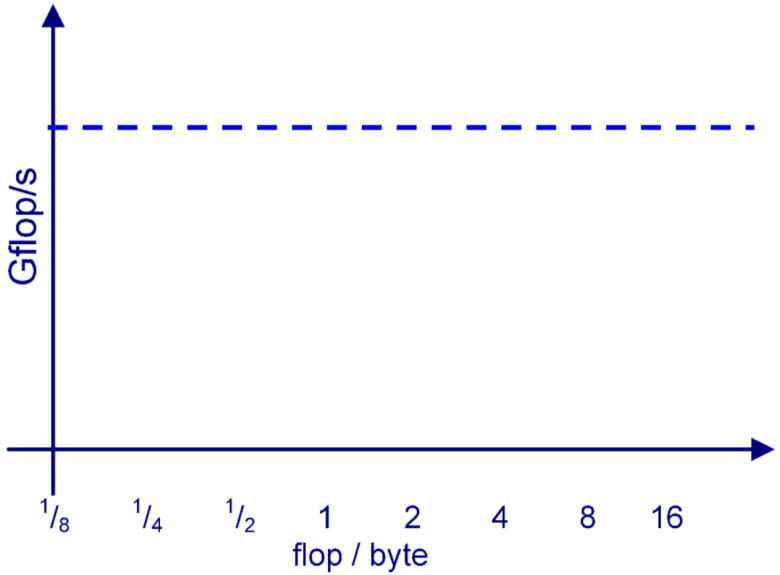


Φτιάχνοντας το μοντέλο roofline



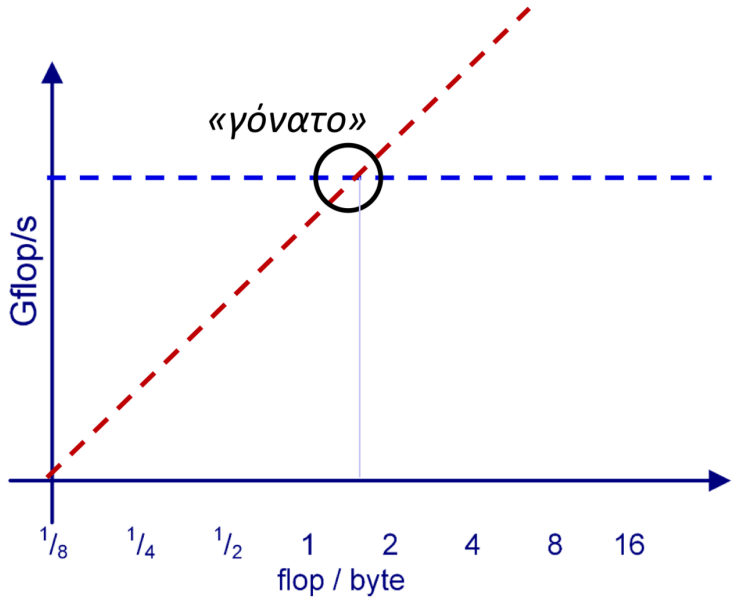
Φτιάχνοντας το μοντέλο roofline

- Βήμα 1: Μέτρηση της υπολογιστικής ισχύος (processor speed) του επεξεργαστή (σε GFLOP/s)

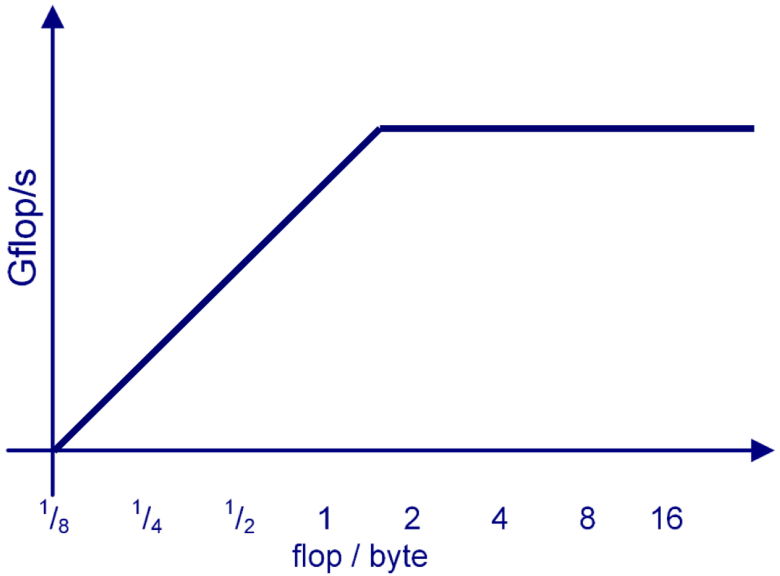


Φτιάχνοντας το μοντέλο roofline

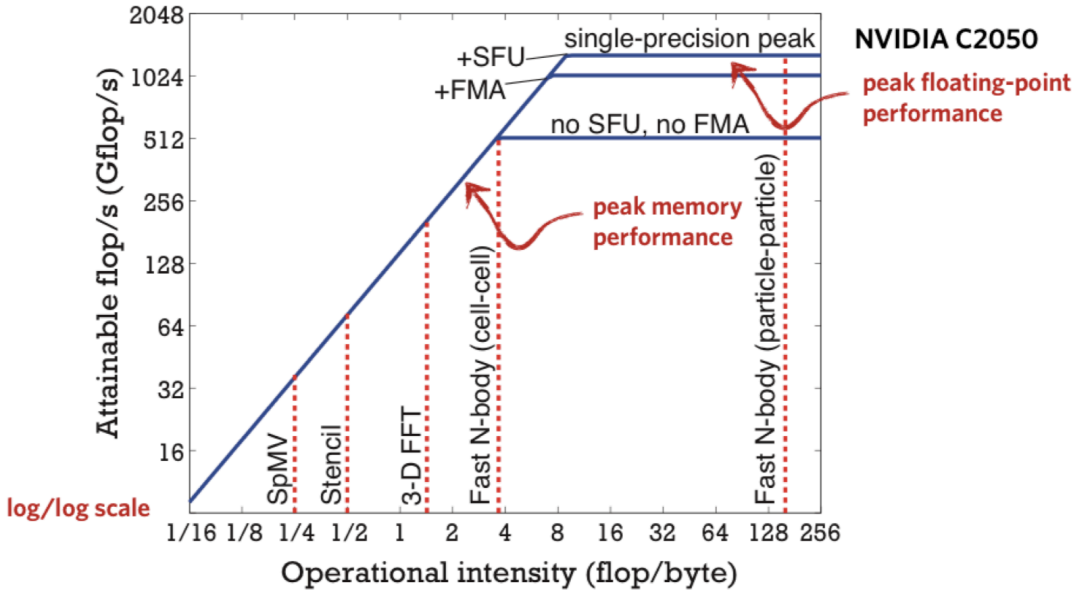
- Βήμα 2: Μέτρηση του εύρους ζώνης (BW) του διαύλου μνήμης (σε bytes/s)



Φτιάχνοντας το μοντέλο roofline



Εφαρμογές στο μοντέλο roofline



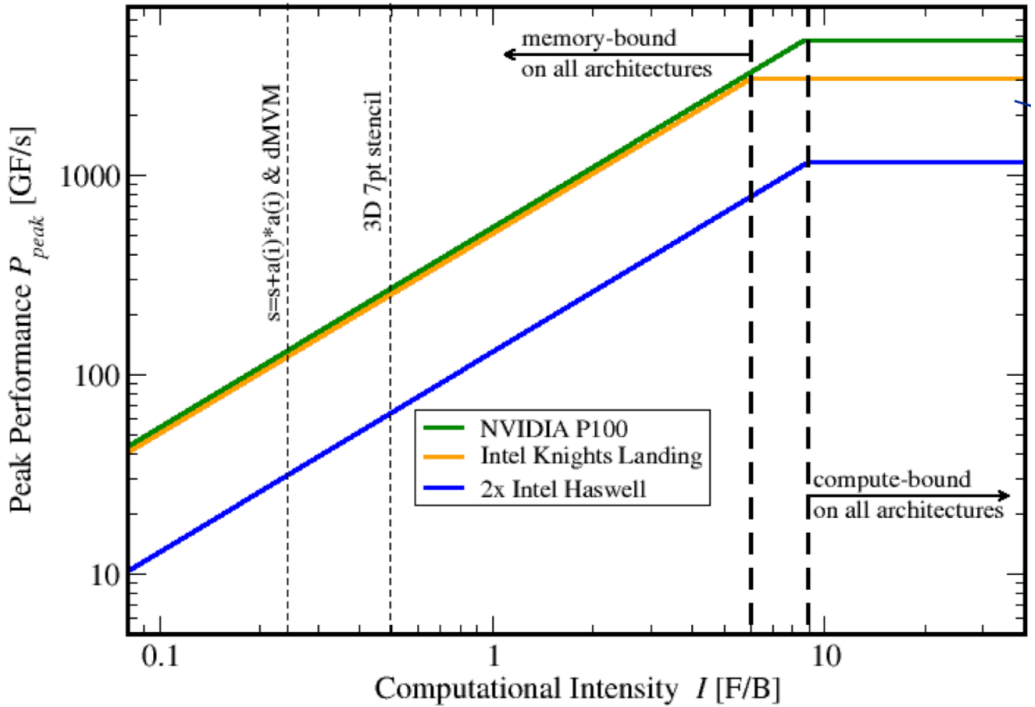
Μοντέλο roofline

- Έστω επεξεργαστής με επίδοση 500 MFLOPS
  - 500MFLOPS = 500 x 10^6 FLOP / sec
  - processor speed = 1 / (500 x 10^6) sec / FLOP = 2 x 10^{-9} sec/FLOP
  - Bandwidth = 4GB/s = 4 x 10^9 bytes/sec

Υπολογιστικός πυρήνας	FLOPs	bytes	OI	1/(OI x BW)	Κλασικό μοντέλο	Roofline
`for (i = 0 ; i < 1000 ; i++)` `  z[i] = x[i] * y[i];`	1000	3 x 1000 x 4 bytes = 12000 bytes	1/12	3 x 10^{-9}	1000 x 2 x 10^{-9} = 2 μsec	1000 x max(2 x 10^{-9} , 3 x 10^{-9}) = **3 μsec**

`for (i = 0 ; i < 1000 ; i++)`
`  for (j = 0 ; j < 1000 ; j++)`
`  y[i] = A[i][j] * x[j]`
	10^6	2 x 1000 x 4 bytes + 1000 x 1000 x 4 bytes = 40008000 bytes	~1/4	10^{-9}	10^6 x 2 x 10^{-9} = 2 msec	10^6 x max(2 x 10^{-9} , 10^{-9}) = **2 msec**

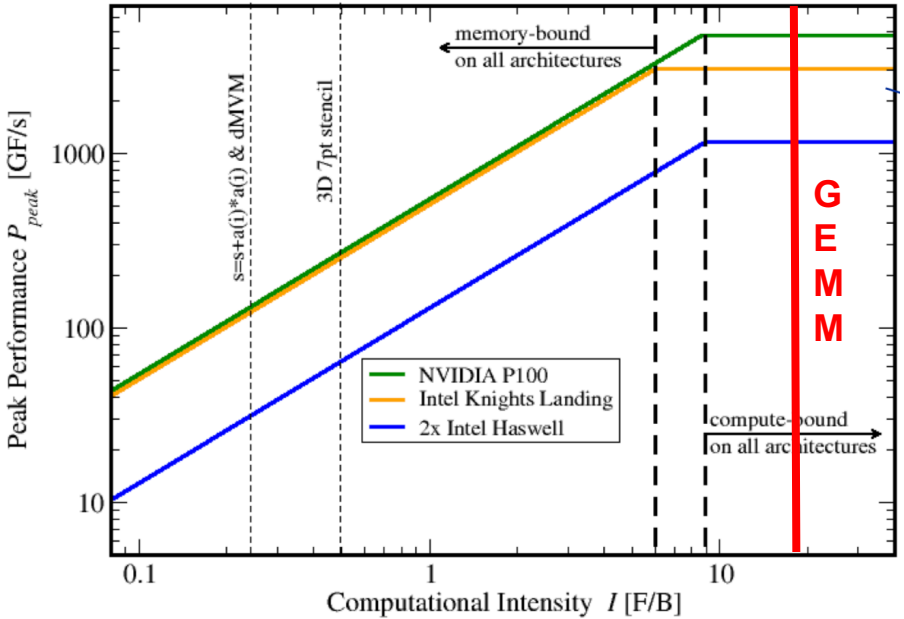
Το μοντέλο roofline σε CPUs και GPUs



Βαθιά νευρωνικά δίκτυα και GPUs

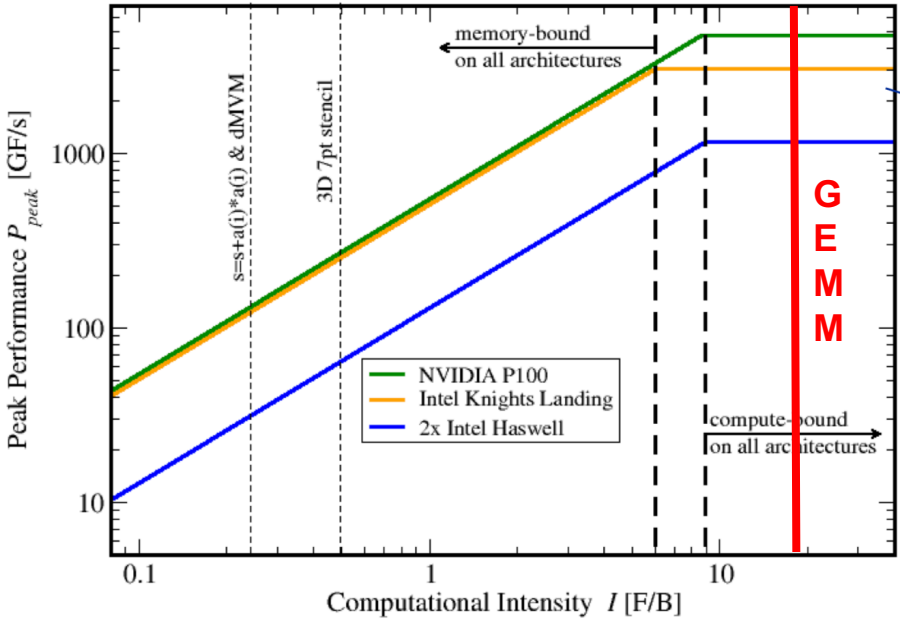
Τι περιορίζει την επίδοση των βαθιών νευρωνικών δικτύων;

- Στα βαθιά νευρωνικά δίκτυα, οι περισσότεροι υπολογισμοί ανάγονται σε πολλαπλασιασμούς πινάκων
- Πολλαπλασιασμός πίνακα με πίνακα (Matrix Multiplication - GEMM)
  - $O(N^3)$ πράξεις
  - $O(N^2)$ προσβάσεις στη μνήμη
  - Operational Intensity: O(N)
- Για την εκπαίδευση βαθιών νευρωνικών δικτύων, **οι GPUs επιτυγχάνουν καλύτερη επίδοση**



Γιατί προτιμάμε τις GPUs για την εκπαίδευση βαθιών νευρωνικών δικτύων

- Ο πολλαπλασιασμός πινάκων εμφανίζεται τόσο στην εκπαίδευση όσο και στο inference
- Είναι υπολογιστικός πυρήνας με ένταση στους υπολογισμούς (όχι στη μνήμη)
- Για την εκπαίδευση βαθιών νευρωνικών δικτύων, **οι GPUs επιτυγχάνουν καλύτερη επίδοση**



Βαθιά νευρωνικά δίκτυα και πολλαπλασιασμός πινάκων

- Παραδοσιακά, πρόκειται για την πιο “ακριβή” πράξη στα βαθιά νευρωνικά δίκτυα

- Ο πολλαπλασιασμός με πινάκων με πολυπλοκότητα **O(n^3)** δεν κλιμακώνει τόσο όσο άλλες πράξεις
  - Άρα μπορεί να αποτελέσει εμπόδιο στην κλιμάκωση καθώς τα μεγέθη προβλημάτων κλιμακώνουν

- Τα βαθιά νευρωνικά δίκτυα είναι ακόμα σε ανοδική τάση και καταναλώνουν μεγάλο κομμάτι των κύκλων υπολογισμού

- Ερώτημα: θα είναι οι περισσότεροι υπολογισμοί στο μέλλον πολλαπλασιασμοί πινάκων;
  - Πώς επηρεάζει αυτό την αρχιτεκτονική;

Σχεδίαση νέων αρχιτεκτονικών με βάση τον πολλαπλασιασμό πινάκων

- Σημαντική ευκαιρία για το σχεδιασμό νέων αρχιτεκτονικών, εξειδικευμένων στη συγκεκριμένη πράξη
  - Google TPU : μια υπολογιστική μηχανή για γρήγορο πολλαπλασιασμό πινάκων
    - Χρησιμοποιείται ευρέως για inference

- Οι εξειδικευμένες αρχιτεκτονικές μπορούν να βοηθήσουν στη χαμηλότερη κατανάλωση ενέργειας