Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Μηχανολόγων Μηχανικών
Τομέας Ρευστών
Εργαστήριο Θερμικών Στροβιλομηχανών
Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης

Επίλυση Διδιάστατης Ροής με Κεντροκυψελική Διατύπωση σε Μη-Δομημένα Πλέγματα. Προγραμματισμός σε Επεξεργαστές Καρτών Γραφικών.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΓΕΩΡΓΙΟΣ Δ. ΝΤΑΝΑΚΑΣ

Επιβλέπων : ΚΥΡΙΑΚΟΣ Χ. ΓΙΑΝΝΑΚΟΓΛΟΥ
Καθηγητής ΕΜΠ

Αθήνα, Φεβρουάριος 2012

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Μηχανολόγων Μηχανικών
Τομέας Ρευστών
Εργαστήριο Θερμικών Στροβιλομηχανών
Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης

Επίλυση Διδιάστατης Ροής με Κεντροκυψελική Διατύπωση σε Μη-Δομημένα Πλέγματα. Προγραμματισμός σε Επεξεργαστές Καρτών Γραφικών.

Διπλωματική Εργασία

Γεώργιος Δ. Ντανάκας

Επιβλέπων: Κ. Χ. Γιαννάκογλου
Καθηγητής Ε.Μ.Π.

Φεβρουάριος 2012

Περίληψη

Τα τελευταία χρόνια, η Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης (ΜΠΥΡ&Β) του Εργαστηρίου Θερμικών Στροβιλομηχανών (ΕΘΣ) του Εθνικού Μετσόβιου Πολυτεχνείου (ΕΜΠ) δραστηριοποιείται συστηματικά στην επίλυση προβλημάτων ρευστοδυναμικής σε επεξεργαστές καρτών γραφικών και, πρόσφατα, παράλληλων καρτών γραφικών (GPUs). Στο πλαίσιο αυτό, έχει πραγματοποιηθεί ανάπτυξη επιλυτών των εξισώσεων ροής (Euler, Navier-Stokes) στη γλώσσα προγραμματισμού CUDA C της εταιρίας NVIDIA, σε μη-δομημένα και δομημένα πλέγματα με κεντροκομβική διατύπωση επιτυγχάνοντας επιταχύνσεις μεγαλύτερες του 40x συγκριτικά με επιλύτες που κάνουν αποκλειστικά χρήση του κεντρικού επεξεργαστή.

Στην παρούσα διπλωματική εργασία, διερευνάται η χρήση μη-δομημένου πλέγματος με κεντροκυψελική διατύπωση για την επίλυση των εξισώσεων ροής σε GPUs. Αναπτύχθηκε, για το σκοπό αυτό, κώδικας, σε CUDA C, που επιλύει τις διδιάστατες εξισώσεις ροής Euler συμπιεστού ρευστού, με τη χρήση της μεθόδου πεπερασμένων όγκων σε μη-δομημένα πλέγματα, αποθηκεύοντας τα ρο·ι·κά μεγέθη στα κέντρα βάρους των όγκων ελέγχου. Η ανάπτυξη του κώδικα πραγματοποιήθηκε με αφετηρία τον υπάρχοντα, πιστοποιημένο, αντίστοιχο κώδικα της ΜΠΥΡ&Β για μη-δομημένα πλέγματα κεντροκομβικής διατύπωσης, με τον οποίο και συγκρίθηκε. Η επίλυση των εξισώσεων ροής έγινε με τη χρήση σχήματος της μεθόδου χρονοπροέλασης. Η εκτέλεση του κώδικα έγινε σε κάρτες γραφικών Tesla M2050 της NVIDIA. Η σύγκριση των αποτελεσμάτων και των χρόνων σύγκλισης του κώδικα που αναπτύχθηκε, με τον κώδικα που κάνει χρήση της κεντροκομβικής διατύπωσης, πραγματοποιήθηκε σε ίδια πλέγματα, για ίδιες συνθήκες εξωτερικής ροής. Αναδείχθηκε, με αυτόν τον τρόπο, η κεντροκυψελική μέθοδος ως υποδεέστερη σε ταχύτητα αλλά υπέρτερη σε ακρίβεια.

National Technical University of Athens
School of Mechanical Engineering
Fluids Section
Laboratory of Thermal Turbomachines
Parallel CFD & Optimization Unit

Programming of a 2D Euler equations' solver with cell-centered formulation on unstructured grids enabled to run on GPUs

Diploma Thesis

Georgios D. Ntanakas

Advisor: K. C. Giannakoglou
Professor NTUA

February 2012

Abstract

In the last years, the Parallel CFD & Optimization Unit of the Laboratory of Thermal Turbomachines of the National Technical University of Athens is developing CFD methods-software running on a single GPU and, recently, on parallel GPU systems. Solvers of the flow equations (Euler, Navier-Stokes) have been developed, written in CUDA C, a programming language by NVIDIA, using unstructured and stuctured grids with vertex-centered finite volume formulation, achieving accelerations greater than 40x comparing to solvers which run exclusively on CPUs.

In this diploma thesis, the use of a 2D unstructured grid along with a cell-centered finite volume method was developed and used to solve the flow equations. To this purpose, a new code was developed, written in CUDA C, which solves the 2D, compressible, Euler's flow equations using an unstructured grid and considering the flow variables to be stored at the centroids of the control volumes. The code development was based and validated using the existing, in-house code which uses unstructured grids and a vertex-centered scheme. The numerical solution of the flow equations was carried out by using a time-marching scheme. The runs of the code were performed on NVIDIA Tesla M2050 GPUs. The comparison of the results and times of convergence between the developed code and the code with the vertex-centered scheme was made on the same grids, with the same farfield flow conditions. The conclusion drawn is that the cell-centered method is slower but more accurate than the vertex-centered one.

Περιεχόμενα

Κατάλογος Σχημάτων

Ευχαριστίες

Η παρούσα διπλωματική εργασία σηματοδοτεί την ολοκλήρωση των προπτυχιακών μου σπουδών στη Σχολή Μηχανολόγων Μηχανικών του Ε.Μ.Π.. Θα ήθελα να εκφράσω, από τη θέση αυτή, τις ειλικρινείς ευχαριστίες μου στον επιβλέποντα Καθηγητή, Κ. Χ. Γιαννάκογλου για την ευκαιρία που μου έδωσε να ασχοληθώ με το συγκεκριμένο θέμα, για την παροχή της δυνατότητας χρήσης του εξοπλισμού της Μονάδας Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης (ΜΠΥΡ&Β) του Εργαστηρίου Θερμικών Στροβιλομηχανών (ΕΘΣ) καθώς και για την ουσιαστική καθοδήγηση και υποστήριξη που μου προσέφερε σε όλη τη διάρκεια εκπόνησης της διπλωματικής εργασίας, ιδιαίτερα σε στιγμές που παρουσιάστηκαν σημαντικά προβλήματα, παρά τον προσωπικό φόρτο εργασίας. Θερμά θα ήθελα να ευχαριστήσω, επίσης, για τον πολύτιμο χρόνο του, τον υποψήφιο διδάκτορα Ξ. Τρομπούκη, χωρίς την υπομονή και συνεχή βοήθεια του οποίου, ανεξαρτήτως ώρας και μέρας, η ολοκλήρωση της διπλωματικής εργασίας θα ήταν αδύνατη. Ευχαριστώ, ακόμα, τους Διδάκτορες Β. Ασούτη και Δ. Παπαδημητρίου και όλη την υπόλοιπη ερευνητική ομάδα της ΜΠΥΡ&Β. Τέλος, θα ήθελα να ευχαριστήσω τους γονείς μου, Δημήτρη και Παναγιώτα, την αδερφή μου και όλους τους φίλους μου που στάθηκαν δίπλα μου όλα αυτά τα χρόνια.

Γεώργιος Δ. Ντανάκας
Ιανουάριος, 2012

Κεφάλαιο 1

Εισαγωγή

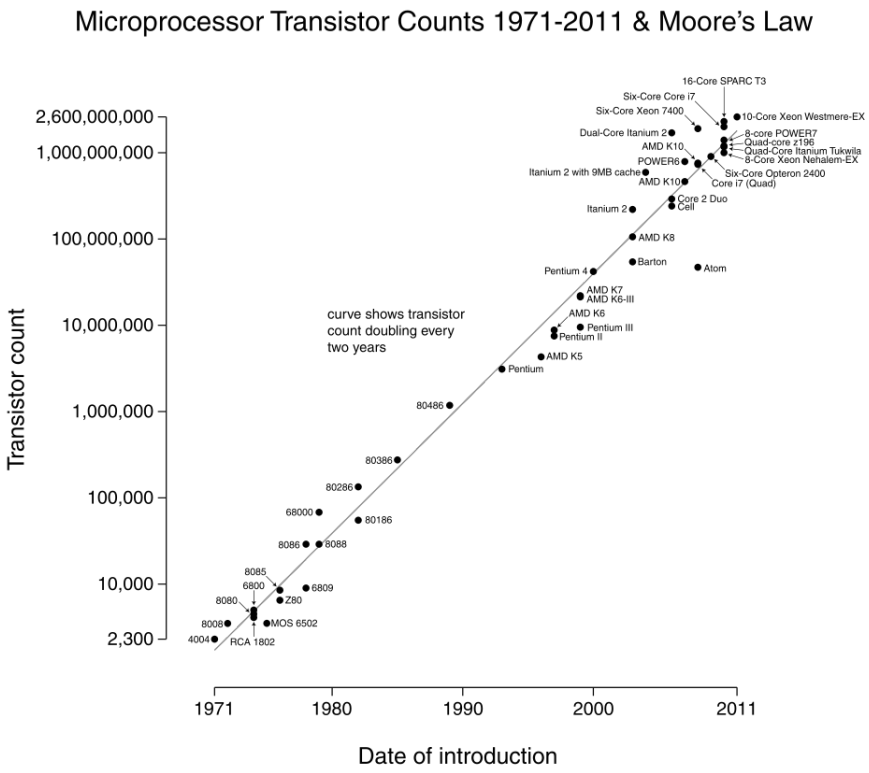
Στη σύγχρονη Υπολογιστική Ρευστοδυναμική (Computational Fluid Dynamics - CFD) είναι δεδομένη η αυξημένη ανάγκη υπολογιστικής ισχύος τόσο λόγω των πολύπλοκων γεωμετριών που τίθενται σε ρευστοδυναμική μελέτη όσο και λόγω της πολυπλοκότητας των εξισώσεων ροής και των αριθμητικών σχημάτων που επιλέγονται για την επίλυσή τους. Η ανάπτυξη των προγραμματιζόμενων καρτών γραφικών (Graphics Processing Units - **GPU**) και των δυνατοτήτων τους τα τελευταία χρόνια παρέχει πρόσφορο έδαφος για τη συνεργασία τους με την κεντρική μονάδα επεξεργασίας (Central Processing Unit - **CPU**) με σκοπό την επιτάχυνση της εκτέλεσης των επιλυτών των ρευστοδυναμικών προβλημάτων.

1.1 Συγκριτική Παρουσίαση Κεντρικών Επεξεργαστών και Καρτών Γραφικών

1.1.1 Ανάπτυξη των CPU

Πριν την είσοδο των GPU στο προσκήνιο της ανάπτυξης εφαρμογών (πλην των γραφικών), η δημιουργία λογισμικού για ένα εύρος διαφορετικών αναγκών βασιζόταν αποκλειστικά στις CPU. Η ανάγκη για γρηγορότερους υπολογισμούς οδήγησε σύντομα την έρευνα σε επίπεδο υλικού (hardware). Η ταυτόχρονη αύξηση των τρανζίστορ ανά επεξεργαστή (νομος Moore- Σχ. 1.1) οδήγησε στην αύξηση των πράξεων δεκαδικών ανά δευτερόλεπτο (FLOP) και άρα την επιτάχυνση των υπολογισμών. Χαρακτηριστικά αναφέρεται ότι από τους επεξεργαστές του 1MHz της δεκαετίας του '80 οδηγηθήκαμε σε επεξεργαστες 1-4GHz σήμερα (1000 φορες γρηγορότερους!).

Η ανάγκη για μεγαλύτερη υπολογιστική ισχύ βρήκε λύση στα συστήματα παράλληλων υπολογισμών. Αρχικά, με τη χρήση παράλληλων επεξεργαστών και στη συνέχεια, με την ενσωμάτωση διαφορετικών πυρήνων στον ίδιο επεξεργαστή (multi-core processors). Σήμερα, στο εμπόριο στο επίπεδο των προσωπικών υπολογιστών κυκλοφορούν μέχρι και 8-πύρηνα και σύντομα 16-πύρηνα μοντέλα



Σχήμα 1.1: Νόμος του Moore [41].

CPU (Σχ. 1.2). Αξίζει να σημειωθεί ότι η αύξηση της ταχύτητας επεξεργασίας που προσφέρουν είναι, κατά προσέγγιση, ανάλογη με τον αριθμό των πυρήνων.

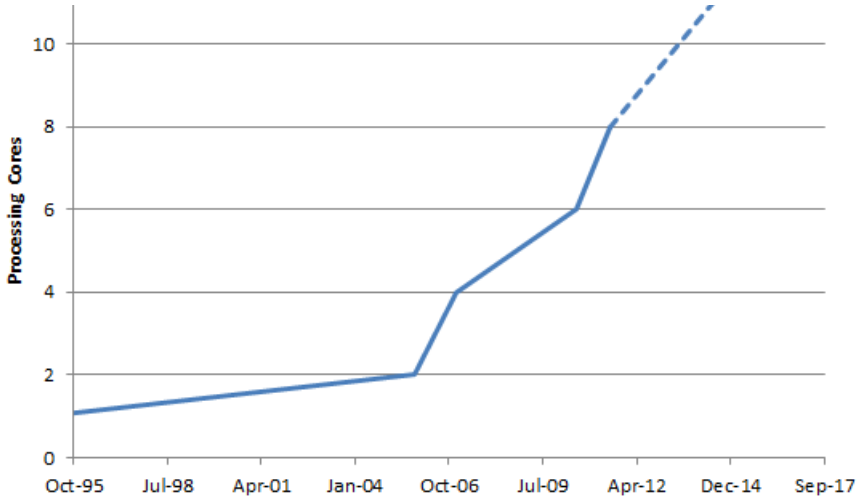
Όρια στην Ανάπτυξη των CPU

Από τις αρχές τις τελευταιάς δεκαετίας άρχισαν να διαφαίνονται τα όρια της βελτίωσης αυτής, που διατηρήθηκε σταθερή για 40 χρόνια. Παρατίθενται οι βασικότεροι από τους λόγους (όρια ή τείχη - walls) [29].

- Το **όριο ισχύος** (power wall) δηλώνει το οριακό σημείο στο οποίο έφτασε η συνολική κατανάλωση ισχύος και η κατανάλωση ισχύος ανά μονάδα επιφάνειας καθώς η συχνότητα χρονισμού του ρολογιού (clock rate) αυξάνει.

- Το **όριο συχνότητας** (frequency wall) είναι ο περιορισμός της συχνότητας λειτουργίας του επεξεργαστή από την απόσταση που χρειάζονται τα ηλεκτρικά σήματα για να διαδοθούν εντός του ολοκληρωμένου κυκλώματος. Η αντιμετώπιση αυτού του προβλήματος απαιτεί τη σμίκρυνση των διαστάσεων των ολοκληρωμένων κυκλωμάτων.

- Το **όριο μνήμης** (memory wall) αντιπροσωπεύει το αυξανόμενο χάσμα μεταξύ της ταχύτητας της μνήμης και της ταχύτητας της CPU . Στους σύγχρονους

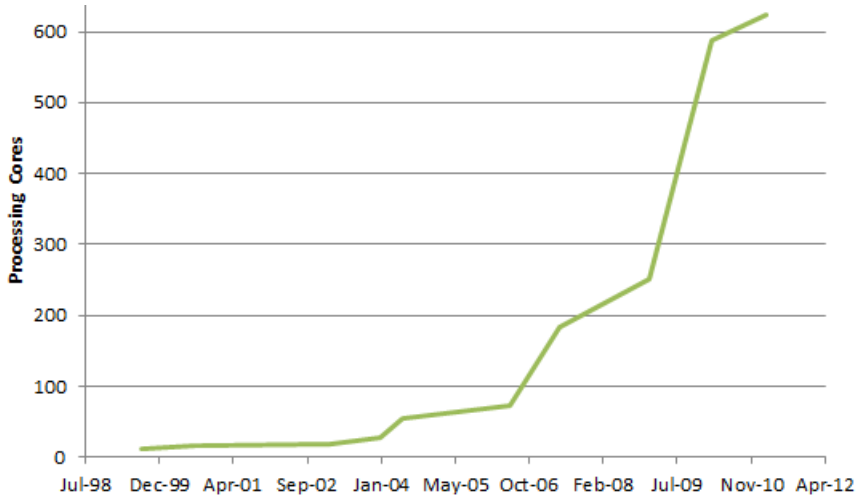


Σχήμα 1.2: Αύξηση πυρήνων των επεξεργαστών της Intel σε συνάρτηση με το χρόνο [42].

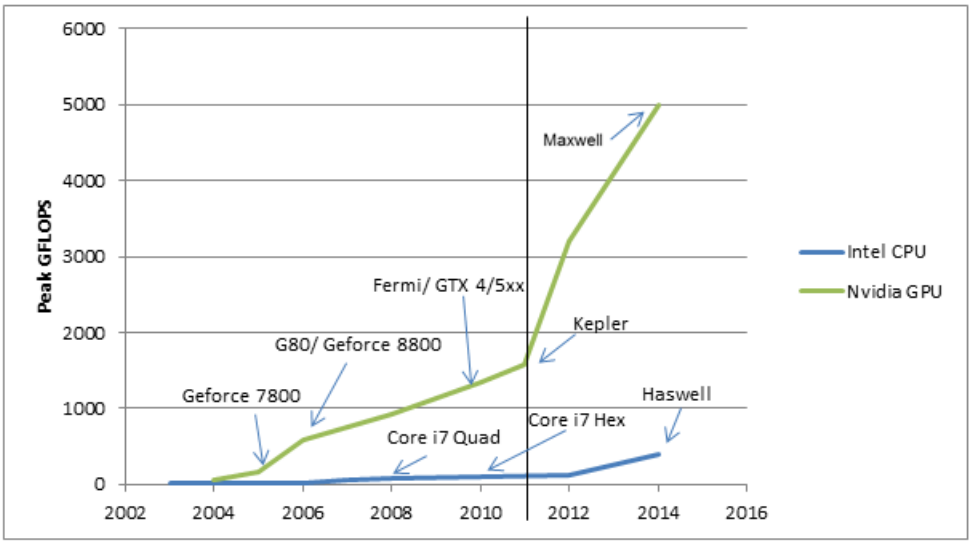
υπολογιστές μια προσπέλαση στην κύρια μνήμη απαιτεί εκατοντάδες κύκλους ρολογιού για τον επεξεργαστή.

- Το **όριο παραλληλισμού επιπέδου εντολής** (Instruction Level Parallelism - ILP wall) εκφράζει την αδυναμία να βρεθεί επαρκής παραλληλισμός σε μεμονωμένες ροές εντολών ώστε να κρατηθεί απασχολημένος ένας μονοεπεξεργαστής υψηλών επιδόσεων.

Ο συνδυασμός των παραπάνω αποτελεί αδιέξοδο για την ανάπτυξη των CPU με τους ρυθμούς που γνωρίζαμε. Είναι κοινώς αποδεκτό πως ήδη ξεκίνησε να διανύεται η περίοδος που ο νόμος του Moore (Σχ. 1.1) έπαψε να ισχύει.



Σχήμα 1.3: Αύξηση πυρήνων GPU των καρτών της Nvidia σε συνάρτηση με το χρόνο [42].



	Peak GFLOPS	Year		Peak GFLOPS	Year
Intel Pentium 4	7	2003	Nvidia Geforce 6800	54	2004
Intel Pentium D	13	2005	Nvidia Geforce 7800	165	2005
Intel Core 2 Duo	23	2006	Nvidia Geforce 8800	576	2006
Intel Core 2 Quad	51	2007	Nvidia Geforce GTX 280	933	2008
Intel Core i7 Quad	70	2008	Nvidia Geforce GTX 480	1,350	2010
Intel Core i7 Hex	109	2010	Nvidia Geforce GTX 580	1,580	2011
Intel Ivy Bridge	130	2012	Nvidia Kepler	3,200	2012
Intel Haswell	400	2014	Nvidia Maxwell	5,000	2014

Σχήμα 1.4: Ενδεικτικη σύγκριση FLOPS μεταξύ Nvidia και Intel [42].

1.1.2 Ανάπτυξη των GPU

Οι πρώτες GPU εμφανίστηκαν το 1981. Απουσίαζε οποιαδήποτε δυνατότητα επεξεργασίας των pixels (που έπρεπε να γίνει και πάλι στην CPU) και χρησίμευαν ως ένας ενδιάμεσος buffer πριν την απεικόνιση στην οθόνη. Οι GPU με δυνατότητες επεξεργασίας των pixels εμφανίστηκαν το 1984, ενσωματώνοντας σε ένα ολοκληρωμένο κύκλωμα ρουτίνες επεξεργασίας και απεικόνισης (rasterization). Περιορίζονταν, όμως, μόνο σε διδιάστατα γραφικά. Η απαίτηση για 3D παιχνίδια και εφαρμογές οδήγησε στην εξέλιξη των πρώτων σύγχρονων GPU στις αρχές του 1990, που έδιναν τη δυνατότητα επιτάχυνσης των απαιτητικών τριδιάστατων γραφικών με τρείς (3) ξεχωριστές κάρτες γραφικών από τις οποίες απουσίαζαν οι διδιάστατες δυνατότητες. Το 2001 εμφανίστηκαν οι πρώτες κάρτες (Nvidia Geforce3) που υλοποιούσαν ουσιαστικά μία παράλληλη αρχιτεκτονική όπου κάθε pixel επεξεργαζόταν ανεξάρτητα από τα υπόλοιπα, ανοίγοντας έτσι το δρόμο για την αξιοποίηση τους σε γενικότερης χρήσης εφαρμογές / υπολογισμούς (General Purpose Computation on GPU - GPGPU).

Την τελευταία δεκαετία η ραγδαία ανάπτυξη των GPU για την εκπλήρωση

αναγκών για γραφικά υψηλών προδιαγραφών σε παιχνίδια και άλλες εφαρμογές οδήγησε σε αρχιτεκτονικές που χρησιμοποιούσαν ένα πλήθος πυρήνων, εκατοντάδες φορές πάνω από το πλήθους των αντίστοιχων CPU (Σχ. 1.3) με αποτέλεσμα να αποδίδουν πολλαπλάσια FLOP (Σχ. 1.4).

1.2 Διεύρυνση της Χρήσης των GPU

Η ταχύτατη εξέλιξη στις GPU, που είχαν πλέον αποκτήσει αξιοπρόσεκτα χαρακτηριστικά, σε συνδυασμό με το κόστος τους, το οποίο είναι αποδεκτό για το μέσο χρήστη και συγκρίσιμο με αυτό των CPU, οδήγησε στη γέννηση της ιδέας να χρησιμοποιηθούν σε εφαρμογές πέρα από την επεξεργασία γραφικών. Ζητούμενο ήταν, λοιπόν, να δημιουργηθούν τα κατάλληλα εργαλεία (γλώσσα προγραμματισμού, compiler) που θα εξασφάλιζαν για το δημιουργηθέν πρόγραμμα την επικοινωνία GPU-CPU (ετερογενής προγραμματισμός) και θα εκμεταλλεύονταν την παραλληλία των πολλαπλών πυρήνων της GPU. Αρχικά, η δημιουργία τέτοιων προγραμμάτων απαιτούσε πλήρη γνώση της αρχιτεκτονικής της κάρτας και μετατροπές για την προσαρμογή του φυσικού προβλήματος στα "γραφικά" δεδομένα. Η εξέλιξη στον τομέα έκανε τις κάρτες γραφικών προσβάσιμες και φιλικότερες στο μέσο χρήστη που επιθυμούσε να προγραμματίσει χρησιμοποιώντας τις.

Στις μέρες μας έχουν υλοποιηθεί εφαρμογές GPGPU οι οποίες παρουσιάζουν σημαντική επιτάχυνση (speed-up) σε σχέση με τις αντίστοιχες εφαρμογές όταν υλοποιούνται μόνο σε CPU. Στη συνέχεια, αναφέρονται μερικά χαρακτηριστικά παραδείγματα τέτοιων εφαρμογών σε τομείς και πέρα απο αυτόν της υπολογιστικής ρευστοδυναμικής.

1.2.1 Παραδείγματα εφαρμογών ανεπτυγμένων σε GPU

Βιοπληροφορική

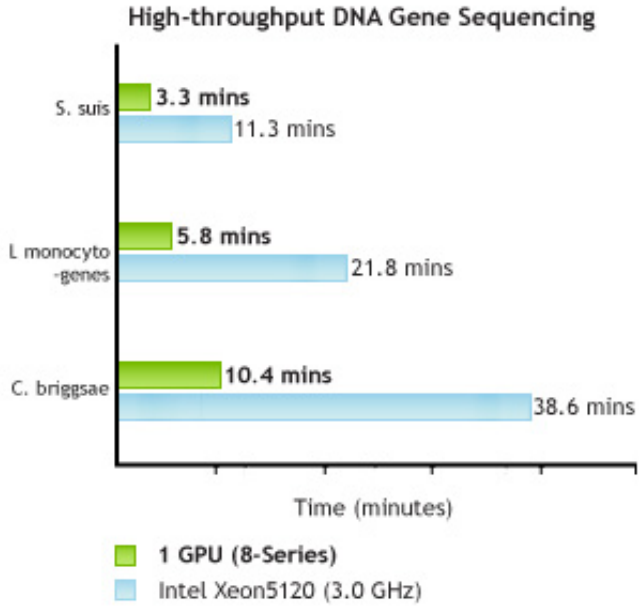
Στον τομέα της Βιοπληροφορικής [30], για έναν κώδικα (MUMmerGPU) που ασχολείται με τη στοίχιση γονιδίων του DNA παρατηρούμε την επιτάχυνση του Σχ. 1.5. Η εκτέλεση είναι 3-4 φορές ταχύτερη.

Οικονομικές Επιστήμες

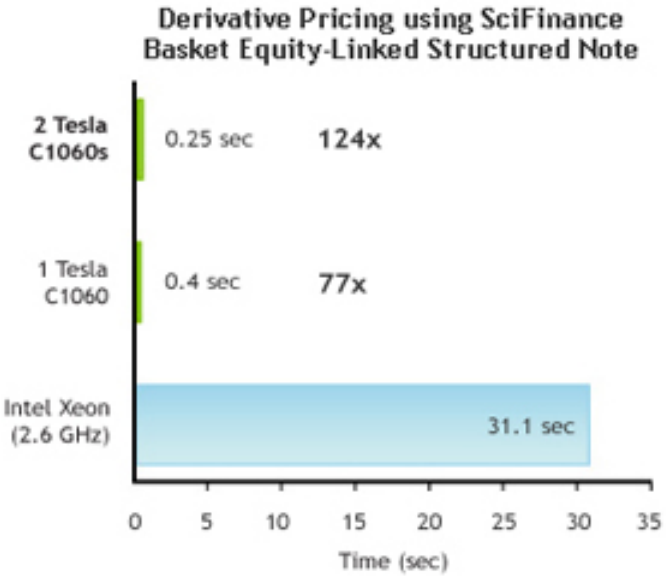
Στον τομέα των Οικονομικών Επιστημών [43], ένα λογισμικό αποτίμησης χρηματοοικονομικών παραγώγων (derivatives pricing) παρουσιάζει speed-up 77 φορές με μία GPU και 124 φορές με 2 GPU . (Σχ. 1.6)

Αριθμητική πρόβλεψη καιρού

Στον τομέα της μετεωρολογικής πρόβλεψης [35] έχουν αναπτυχθεί αριθμητικά μοντέλα υπολογισμού των καιρικών συνθηκών αλλά και των μακροπρόθεσμων



Σχήμα 1.5: Σύγκριση χρόνων υλοποίσης της MUMmerGPU με τον επεξεργαστή Xeon της Intel και μια GPU της Nvidia [30].



Σχήμα 1.6: Σύγκριση χρόνων εκτέλεσης οικονομικής εφρμογής με τον επεξεργαστή Xeon της Intel, μία και δύο GPU της Nvidia [43].

κλιματικών αλλαγών. Ο προγραμματισμός τους σε GPU οδηγεί σε επιτάχυνση ως και 20 φορές.

Σεισμική προσομοίωση

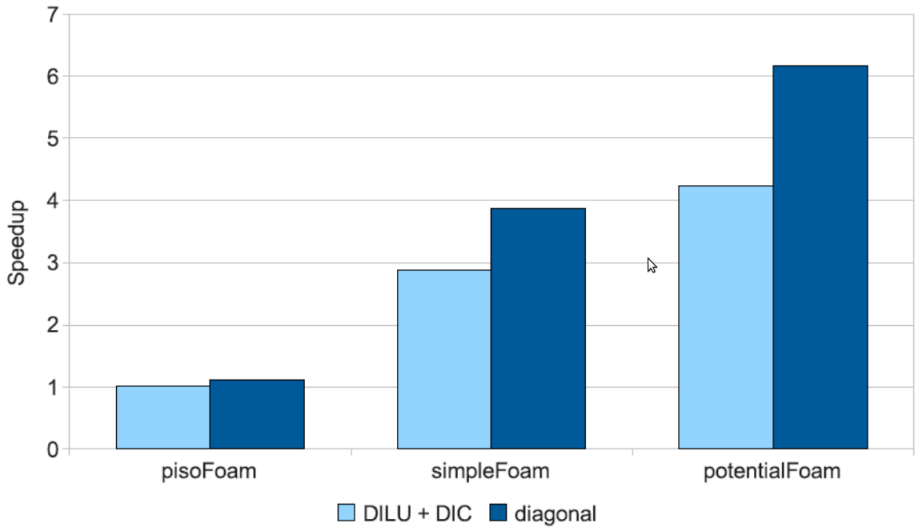
Στον τομέα αυτό καταγράφονται επιταχύνσεις ως και 20x για μοντέλα πεπερασμένων όγκων υψηλής ακρίβειας. [37]

Αστροφυσική

Ο όγκος των υπολογιστικών δεδομένων στον τομέα της Αστροφυσικής έφερε στο προσκήνιο παράλληλα συστήματα GPU για την αποτελεσματική διαχείριση των υπολογισμών. [40]

Υπολογιστική Ρευστοδυναμική

Είναι πλέον αρκετά διαδεδομένο οι εφαρμογές που ήδη υπήρχαν για την επίλυση ροών να τροποποιούνται ώστε να κάνουν χρήση των GPU. Οι αυξημένες υπολογιστικές απαιτήσεις για την επίλυση των εξισώσεων ροής (π.χ. Navier-Stokes) σε πυκνά πλέγματα οδηγούν σε μεγάλους χρόνους εκτέλεσης. Έτσι, η επιτάχυνση που προσφέρουν οι GPU κρίνεται απαραίτητη.



Σχήμα 1.7: Επιτάχυνση εκτέλεσης διαφορετικών περιπτώσεων της εφαρμογής OpenFOAM με χρήση GPU έναντι CPU [36].

Παρατίθενται κάποια παραδείγματα τέτοιων εφαρμογών επισημαίνοντας πως οι CFD χρήσεις των GPU είναι πολυάριθμες και η πλήρης κάλυψη τους ξεφεύγει από τους σκοπούς της παρούσας διπλωματικής εργασίας. Στο [14] παρουσιάστηκε η εφαρμογή μεθόδων για την εύρεση οριακών μεγεθών ροής. Στο [19] περιγράφηκε επιλύτης παραγώγων για μητρώα σε μη-δομημένα πλέγματα. Μία ακόμα λεπτομερής περιγραφή της επίλυσης των Navier-Stokes σε δομημένα πλέγματα παρουσιάστηκε στο [21] και επεκτάθηκε στις τρεις διαστάσεις στο [22]. Η πρώτη

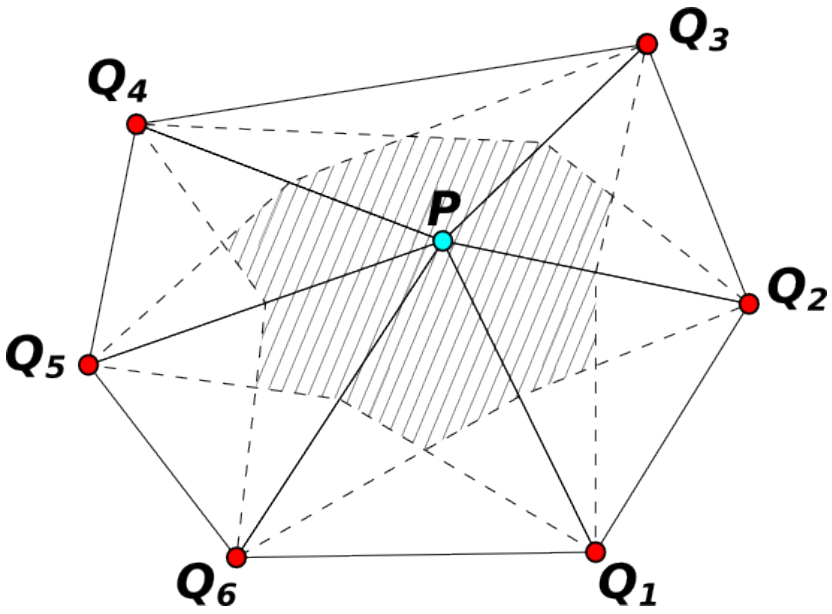
επίλυση των Euler εξισώσεων σε GPU παρουσιάστηκε στο [18] με επιτάχυνση 25x. Έχει αναφερθεί επιτάχυνση 8,5x [17] σε εφαρμογές για μη-συμπιεστή ροή σε δομημένα πλέγματα. Στην ίδια κατεύθυνση έχει δραστηριοποιηθεί και η ΜΠΥΡ&Β. Έχει γίνει χρήση διδιάστατων και τριδιάστατων πλεγμάτων για την πραγματοποίηση υπολογισμών σε συνεκτικές και μη συνεκτικές μόνιμες ροές [12]. Επίσης, έχουν γίνει οι αντίστοιχοι υπολογισμοι και σε μη-μόνιμες ροές [11] και έχει γίνει επέκταση των παραπάνω σε εφαργογές ελέγχου της ροής [13]. Στο σχ. 1.7 [36] φαίνεται η επιτάχυνση της εκτέλεσης διαφορετικών περιπτώσεων της εφαρμογής OpenFOAM με χρήση GPU έναντι CPU. Σε προσομοίωση ρευστού μέσα από σύνθετα εμπόδια [38] αναφέρεται επίλυση των τριδιάστατων εξισώσεων Navier-Stokes σε πραγματικό χρόνο. Ανάλογες εφαρμογές αναπτύσσονται και στον τομέα της αιμοδυναμικής [39].

1.3 Σκοπός και τοποθέτηση της εργασίας στο πλαίσιο της έρευνας στη ΜΠΥΡ&Β

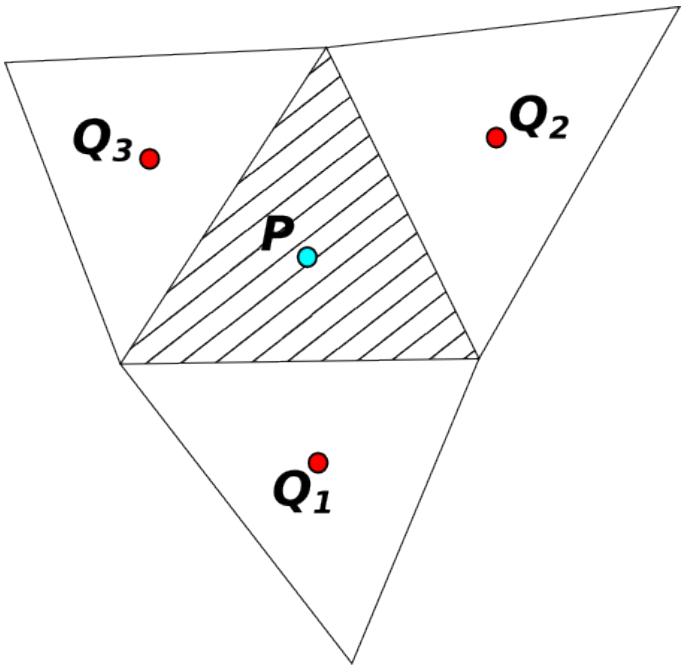
Η Μονάδα Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίσης (ΜΠΥΡ&Β) του Εργαστηρίου Θερμικών Στροβιλομηχανών (ΕΘΣ) της Σχολής Μηχανολόγων Μηχανικών του Ε.Μ.Π. ασχολείται συστηματικά τα τελευταία χρόνια με την επίλυση προβλημάτων ρευστοδυναμικής τόσο σε συστήματα παράλληλων επεξεργαστών όσο και σε GPU [11] [12] [13] και, πρόσφατα, σε συστήματα παράλληλων GPU . Έχει γίνει ανάπτυξη κωδίκων για την επίλυση ροών αλλά και για τη βελτιστοποίση αεροδυναμικών μορφών (εξελικτικοί-στοχαστικοί αλγόριθμοι). Έχουν μελετηθεί διδιάστατα και τριδιάστατα προβλήματα, για μόνιμες και μη-μόνιμες, συνεκτικές και μη-συνεκτικές ροές σε δομημένα και μη δομημένα πλέγματα.

Η παρούσα διπλωματική εργασία επιλύει διδιάστατη μόνιμη, μη-συνεκτική ροή σε μη δομημένο πλέγμα. Το πλέγμα αυτό αποτελείται από τριγωνικά στοιχεία. Η ΜΠΥΡ&Β ήδη διέθετε κώδικα ο οποίος επιλύει τη ροή με τη μέθοδο των πεπερασμένων όγκων αποθηκεύοντας τα ροϊκά μεγέθη στους κόμβους-κορυφές των τριγώνων (vertex-centered) [11] [12] [13]. Στο πλαίσιο αυτής της διπλωματικής εργασίας, δημιουργήθηκε κώδικας ο οποίος αποθηκέυει τα χαρακτηριστικά μεγέθη της ροής στα κέντρα βάρους των τριγώνων (cell-centered). Σκοπός της είναι η διερεύνηση και η σύγκριση των ταχυτήτων σύγκλισης, των επαναλήψεων ως τη σύγκλιση, των αποτελεσμάτων για τις τελικές τιμές των ροϊκών μεγεθών καθώς και των συντελεστών της άνωσης (lift) και οπισθέλκουσας (drag) που προκύπτουν από την κεντροκυψελική μέθοδο έναντι της κεντροκομβικής για ίδιες συνθήκες της ροής.

Κατα την εκτέλεση ενός τέτοιου κώδικα κάθε όγκος ελέγχου (control volume) αντλεί πληροφορία από τους γειτονικούς όγκους ελέγχου. Στην κεντροκομβική (vertex-centered) διατύπωση λογική ο αριθμός των γειτόνων είναι διαφορετικός κάθε φορά και εξαρτάται από τον αριθμό των τριγώνων που έχουν μια κοινή κορυφή (Σχ. 1.8). Η πληροφορία αυτή είναι υποχρεωτικό να μεταφέρεται σε όλη τη



Σχήμα 1.8: Όγκος ελέγχου P και γείτονες $Q_i, \quad i = 1 \cdots 6$ με κεντροκομβική διατύπωση.



Σχήμα 1.9: Όγκος ελέγχου P και γείτονες Q_1, Q_2, Q_3 με κεντροκυψελική διατύπωση.

διάρκεια του κώδικα για να γίνουν οι υπολογισμοί. Αντίθετα, στην κεντροκυψελική (cell-centered) διατύπωση, ο αριθμός των γειτόνων είναι πάντα σταθερός και ισούται με τις τρεις (3) πλευρές του τριγώνου (Σχ. 1.9). Είναι, λοιπόν, εύστοχο να μελετηθεί πως συμπεριφέρεται ο νέος κώδικας που προσαρμόζεται σε αυτή τη διαφορετική

αντιμετώπιση του πλέγματος.

1.4 Περιεχόμενα και Δομή της Εργασίας

Η παρούσα διπλωματική εργασία δομείται ως εξής:

- Στο **κεφάλαιο 2** παρουσιάζεται η αρχιτεκτονική CUDA της Nvidia, τα λειτουργικά κομμάτια μιας αντίστοιχης GPU και πως προσαρμόζεται ο προγραμματισμός για να πραγματοποιηθεί ορθά η χρήση τους. Ακόμα, προβάλλεται η εξέλιξη από την προγενέστερη αρχιτεκτονικη GT200 στη νέα Fermi.

- Στο **κεφάλαιο 3** γίνεται η διατύπωση και η αδιαστατοποιήση των διδιάστατων εξισώσεων Euler. Στη συνέχεια, παρουσιάζεται η διακριτοποίηση του χωρίου και των εξισώσεν ροής (Euler) με βάση το χωρίο, αναλύονται οι οριακές συνθήκες και καταδεικνύεται ο τρόπος υπολογισμου των χωρικών παραγώγων του διανύσματος ροής.

- Στο **κεφάλαιο 4** παρουσιάζεται η αριθμητική μέθοδος επίλυσης των εξισώσεων ροής και ο τρόπος με τον οποίο προγραμματίζεται, με την προβολή του διαγράμματος ροής και την επισήμανση άλλων προγραμματιστικών ζητημάτων.

- Στο **κεφάλαιο 5** παρουσιάζονται τα αποτελέσματα επίλυσης της ροής σε διαφορετικές περιπτώσεις και συγκρίνονται η κεντροκομβική και κεντροκυψελική διατύπωση ως προς τα αποτελέσματα και τους χρόνους σύγκλισης.

- Στο **κεφάλαιο 6** γίνεται μια σύντομη ανακεφαλαίωση, εξάγονται συμπεράσματα από τα αποτελέσματα της εκτέλεσης του κώδικα και δίνονται προτάσεις για μελλοντική επέκταση.

Κεφάλαιο 2

Αρχιτεκτονική CUDA, Δυνατότητες - Εξέλιξη

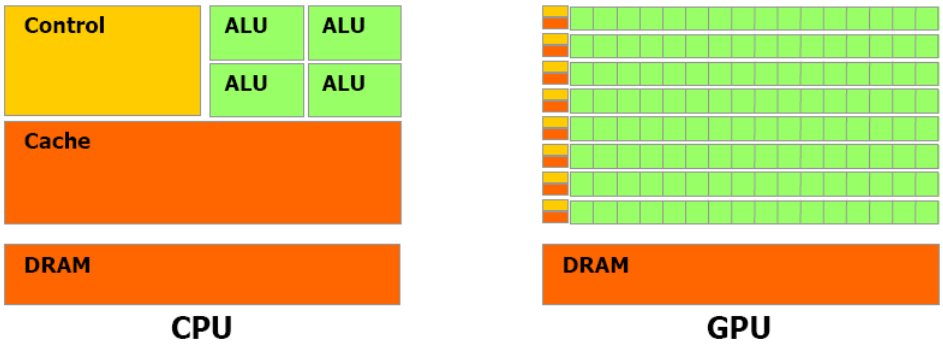
Η εταιρία η οποία δημιούργησε πρώτη πρόσφορο έδαφος για τον προγραμματισμό σε GPU, τόσο σε επίπεδο hardware δημιουργώντας κάρτες με κατάλληλες αρχιτεκτονικές, όσο και σε επίπεδο software επεκτείνοντας ήδη υπάρχουσες γλώσσες προγραμματισμού με σκοπό να γίνουν κατάλληλες για προγραμματισμό καρτών γραφικών είναι η Nvidia. Εισάγοντας την αρχιτεκτονική **CUDA** (Compute Unified Device Architecture) το Νοέμβριο του 2006 κατάφερε να εκμεταλλευτεί πλήρως τις δυνατότητες των GPU και να επιτρέψει στους χρήστες να προγραμματίσουν σε FORTRAN, C, C++, OpenCL, και DirectX, γλώσσες, δηλαδή, συμβατικού προγραμματισμού.

Στο πλαίσιο της παρούσας εργασίας αναπτύχθηκε κώδικας σε CUDA C. Η δημιουργία και τα πρώτα "τρεξίματα" έγιναν στην κάρτα GeForce GTX285 αρχιτεκτονικής GT200. Στη συνέχεια, όμως, έγινε και εκτέλεση στην Tesla M2050 που είναι αρχιτεκτονικής Fermi (αμέσως επόμενη της GT200 και τελευταίας τεχνολογίας). Κρίνεται σκόπιμο, πέρα από την σύγκριση της ταχύτητας σύγκλισης για κεντροβαρική και κεντροκυψελική διατύπωση, να συγκριθεί και η ταχύτητα σύγκλισης του κώδικα για τις δύο GPU.

Θα γίνει αρχικά συνοπτική καταγραφή των χαρακτηριστικών της GT200 και στη συνέχεια θα παρουσιαστεί αναλυτικότερα η Fermi επισημαίνοντας τις καινοτομίες και επεξηγώντας την "φιλοσοφία" της CUDA .

2.1 Γενικά περί τεχνολογίας-αρχιτεκτονικής CUDA

Η επιτάχυνση που προκύπτει με τη χρήση μιας GPU οφείλεται στην κατασκευή της που επιτρέπει την παραλληλοποίηση των υπολογισμών. Σε αντίθεση με μια CPU , η κάρτα γραφικών έχει περισσότερες μονάδες επεξεργασίας (ALU - Arithmetic Logic Unit) και λιγότερη κρυφή μνήμη (cache memory). Στο Σχ. 2.1 είναι εμφανείς οι



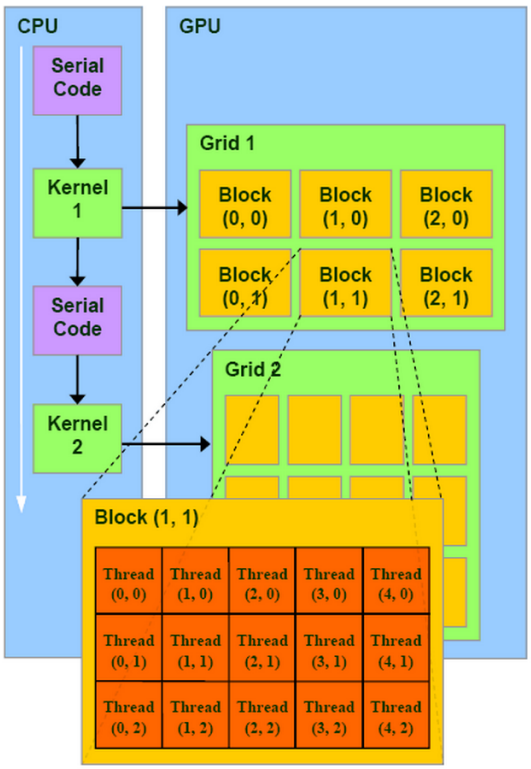
Σχήμα 2.1: Τυπική διάταξη κάρτας γραφικών συγκρινόμενη με έναν μέσο επεξεργαστή.

διαφορές στη διαμόρφωση του υλικού μεταξύ των δύο.

2.1.1 Δομή: Threads, Blocks, Grids

Το τμήμα ενός κώδικα το οποίο είναι επιθυμητό να εκτελεστεί παράλληλα ονομάζεται **kernel**. Κάθε kernel εκτελείται σε ένα **thread**. Τα threads ομαδοποιούνται σε **blocks** και αυτά με τη σειρά τους οργανώνονται σε **grids**. Στο Σχ. 2.2 φαίνεται χαρακτηριστικά η διάταξη που αναφέρθηκε.

Από τη σκοπιά της CUDA, στο υψηλότερο σημείο της ιεραρχίας της κάρτας γραφικών είναι οι streaming multiprocessors (SM). Ένας multiprocessor αποτελείται από υπολογιστικούς πυρήνες (processing cores). Ο multiprocessor είναι υπεύθυνος για να κατανέμει τα threads στους processors, να τα συγχρονίζει, να φορτώνει και να αποκωδικοποιεί τις εντολές και να τις προωθεί για εκτέλεση. Threads του ιδίου block ανατίθενται πάντα στον ίδιο multiprocessor. Για να εκτελεστούν, ομαδοποιούνται σε warps, τα οποία είναι πακέτα των 32 threads. Κάθε warp εκτελείται στον multiprocessor με την λογική SIMT (Single Instruction Multiple Data). Σύμφωνα με τον ορισμό της Nvidia, τα threads ενός warp εκτελούν ταυτόχρονα όλα την ίδια εντολή. Μόλις ολοκληρωθεί η εκτέλεση της εντολής, ο multiprocessor προωθεί την επόμενη κοινή εντολή. Σε περίπτωση που ο κώδικας του kernel περιέχει διακλαδώσεις που δεν ακολουθούνται με τον ίδιο τρόπο από όλα τα threads, τότε το κάθε branch εκτελείται σειριακά, μέχρι και πάλι όλα τα threads να συγκλίνουν στην κοινή τους ροή. Παρόλα αυτά υπάρχουν κάποιοι περιορισμοί. Το σύνολο των ενεργών threads ανά multiprocessor έχει άνω όριο.

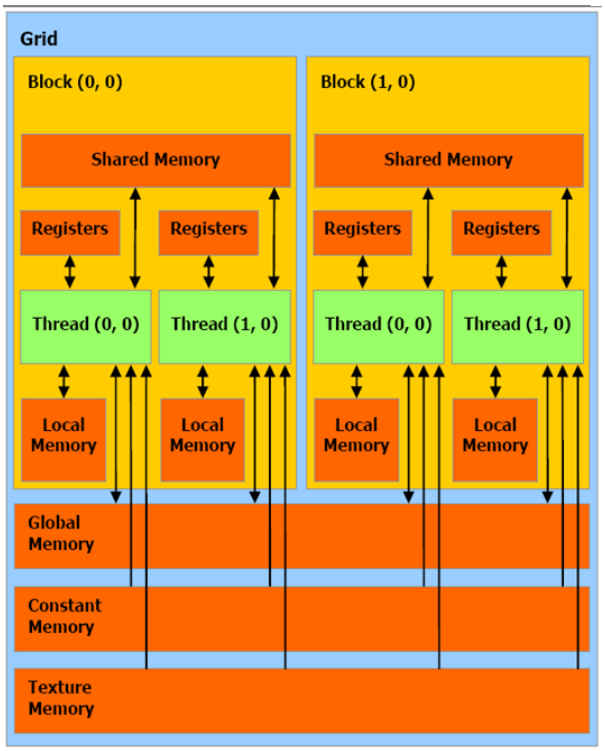


Σχήμα 2.2: Αλληλουχία grid, block, thread.

2.1.2 Μνήμες

Στην CUDA η μνήμη του υπολογιστή (RAM) αναφέρεται ως host memory, ενώ η μνήμη της κάρτας γραφικών ως device memory. Ο κώδικας που βρίσκεται στο εσωτερικό ενός kernel μπορεί να επεξεργαστεί δεδομένα που βρίσκονται μόνο στη device memory, ενώ αντίθετα κώδικας εκτός kernel δεν μπορεί να συνεργαστεί απευθείας με αυτήν. Για αυτόν τον λόγο υπάρχουν συναρτήσεις δέσμευσης, αποδέσμευσης, αντιγραφής και μεταφοράς δεδομένων ανάμεσα στην host και την device memory.

Τα threads έχουν πρόσβαση σε δεδομένα από διάφορους χώρους μνήμης της κάρτας γραφικών. Σε κάθε thread αντιστοιχεί ένα σύνολο από registers προσβάσιμους μόνο από αυτό. Ο register για κάθε πυρήνα αποτελεί την πιο γρήγορη μνήμη που είναι διαθέσιμη. Επίσης, όλα τα threads ενός block έχουν πρόσβαση σε έναν κοινό χώρο μνήμης τη shared memory. Και οι δύο αυτοί χώροι μνήμης βρίσκονται on-chip και άρα επιτυγχάνουν πολύ υψηλούς ρυθμούς διαμεταγωγής. Τέλος, υπάρχει η global memory της κάρτας γραφικών που χρησιμοποιείται για τη μεταφορά και αποθήκευση δεδομένων από και προς την μνήμη host, ενώ δύο ειδικοί χώροι μνήμης της είναι οι constant και texture memory που είναι βελτιστοποιημένες για κάποιες ειδικές χρήσεις (data filtering κλπ). Η εγγραφή τους είναι δυνατή μόνο από την πλευρά της CPU (host). (Σχ. 2.3)



Σχήμα 2.3: Είδη μνημών και με ποια δομικά στοιχεία επικοινωνούν.

2.1.3 Διασύνδεση Προγραμματισμού Εφαρμογών - API

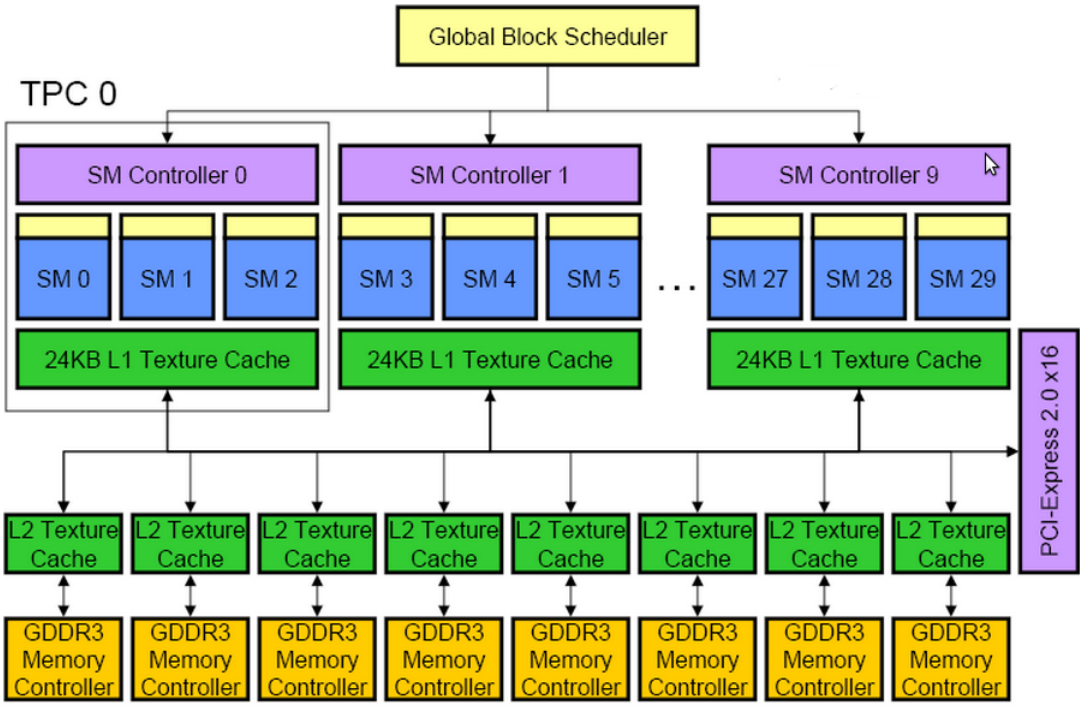
Η API (Application Programming Interface) για την CUDA C περιλαμβάνει ένα ολόκληρο σύστημα λογισμικού. Ο compiler της Nvidia (nvcc) επεξεργάζεται έναν κώδικα C με προσθήκες που ανήκουν στις εξής κατηγορίες:

- Εντολές για το αν μια συνάρτηση εκτελείται στη CPU ή τη GPU .
- Εντολές για το που είναι αποθηκευμένες οι μεταβλητές και εντολές για την αντιγραφή τους κλπ.
- Δηλώσεις σχετικά με τον παραλληλισμό της εκτέλεσης αναφορικά με τα blocks και τα grids.
- Δήλωση μεταβλητών που υποδεικνύουν τις διαστάσεις των blocks και grids και τις θέσεις των threads σε αυτά, γεγονός καθοριστικό για την εκτέλεση του προγράμματος.

Η CUDA έχει σχεδιαστεί και εξελίσσεται ώστε να ενσωματώνει τις εξελίξεις και τις βελτιώσεις σε επίπεδο hardware. Έτσι, σύμφωνα με το σύστημα μέτρησης της υπολογιστικής ικανότητας (compute capability) της Nvidia, από 1.0 σε παλαιότερες

εκδόσεις είμαστε τώρα στο 2.x. Η GT200 έχει 1.3 και διαφοροποιήθηκε με τις προηγούμενες εισάγοντας δυνατότητα εκτέλεσης πράξεων κινητής υποδιαστολής διπλής ακρίβειας ενώ η Fermi έχει 2.0.

2.2 Η Αρχιτεκτονική GT200

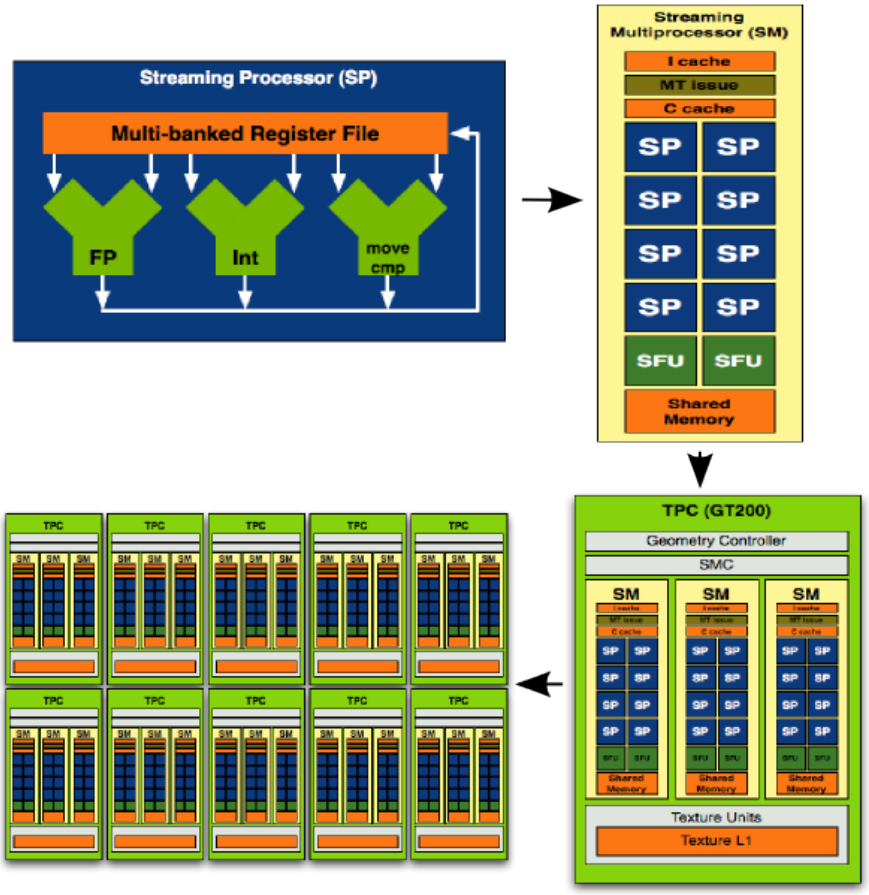


Σχήμα 2.4: Αρχιτεκτονική GT200.

Ως υπολογιστική μονάδα η GPU με GT200 αρχιτεκτονική είναι ένα πολυ-πύρηνο chip (1.4 δις τρανζίστορ - 240 πυρήνες) οργανωμένο σε ιεραρχία δύο επιπέδων που επικεντρώνεται πρωταρχικά στην επίτευξη υψηλής υπολογιστικής απόδοσης παραλληλοποιώντας τον όγκο της εργασίας και "θυσιάζοντας" ένα κομμάτι της απόδοσης του μεμονωμένου thread.

Διαμόρφωση

Μια GT200 κάρτα αποτελείται απο 10 συστοιχίες thread - Thread Processing Clusters (TPC) οι οποίες σχηματίζουν και το πρώτο επίπεδο της ιεραρχίας. Στο επόμενο επίπεδο, κάθε TPC αποτελείται από 3 Streaming Multiprocessors (SM) και ένα δίαυλο επικοινωνίας για την texture μνήμη για κάθε τριάδα SM. Κάθε SM αποτελείται από 8 Thread Processors ή Streaming Processors (SP). Τα Σχ. 2.4 και 2.5 αποδίδουν σχηματικά τα παραπάνω.



Σχήμα 2.5: Από τον SP ως τη GPU.

Διανομή εργασιών

Ο διανέμητής εργασιών (Global Block Scheduler) επιτυγχάνει παραλληλία σε επίπεδο block σε όλη την έκταση της κάρτας. Όταν ένα kernel ξεκινά, πληροφορία για ένα grid στέλνεται απο τη host CPU στη GPU . Ο διανέμητής εργασιών παίρνει την πληροφορία και επιλέγει τα blocks που θα το εκτελέσουν, επιλέγοντας σύμφωνα με τις απαιτήσεις του kernel για thread ανά block, τη shared memory ανά block, τους registers ανά thread κλπ. Σκοπός του είναι, εν τέλει, η μεγιστοποίηση των δυνατοτήτων παραλληλοποίησης.

Σε κάθε SM μπορούν να ανατεθούν ως 8 blocks. Αυτά διαχειρίζονται με τη χρήση μιας μικροαρχιτεκτονικής ομαδοποίησης, το warp (αποτελείται από 16 threads). Ο μέγιστος αριθμός wrap ανά SM είναι 32. Το μοντέλο εκτέλεσης ονομάζεται από την Nvidia Single Instruction, Multiple Thread (SIMT) και σημαίνει πως μια εντολή προσδιορίζει τη συμπεριφορά ενός μόνο ανεξάρτητου thread.

Αποθήκευση δεδομένων

Οι registers είναι χωρητικότητας 64KB και κάθε SM έχει 16000 σε όλο το εύρος της. Κάθε SP έχει 2000 αρχεία register που μπορούν να χρησιμοποιηθούν από ως 128 threads. Τα αρχεία register τμηματοποιούνται δυναμικά μεταξύ των block. Ένα μεμονωμένο thread μπορεί να έχει 4-128 registers.

Στην GT200 κάθε SM έχει 16KB μνήμης shared. Η μνήμη αυτή είναι σηματικά γρηγορότερη συγκριτικά με τη global (πλησιάζει τις ταχύτητες των registers) και χωρίζεται σε τμήματα τα οποία αποδίδονται σε ξεχωριστά block. Με σκοπό τη διατήρηση της ταχύτητας της, δεδομένα μπορούν να μεταφερθούν μόνο από τους registers στη shared και όχι από τη global.

2.3 Η Αρχιτεκτονική Fermi

2.3.1 Γενική Διάταξη

Μια GPU με αρχιτεκτονική Fermi αποτελείται απο 3 δις τρανζίστορ και 512 πυρήνες (υπερδιπλάσια της GT200). Αυτοί οργανώνονται σε 16 SM με 32 πυρήνες ο καθένας. Η RAM είναι GDDR5, σαφώς γρηγορότερη της GDDR3 που είχε η προκατέχουσα αρχιτεκτονική.

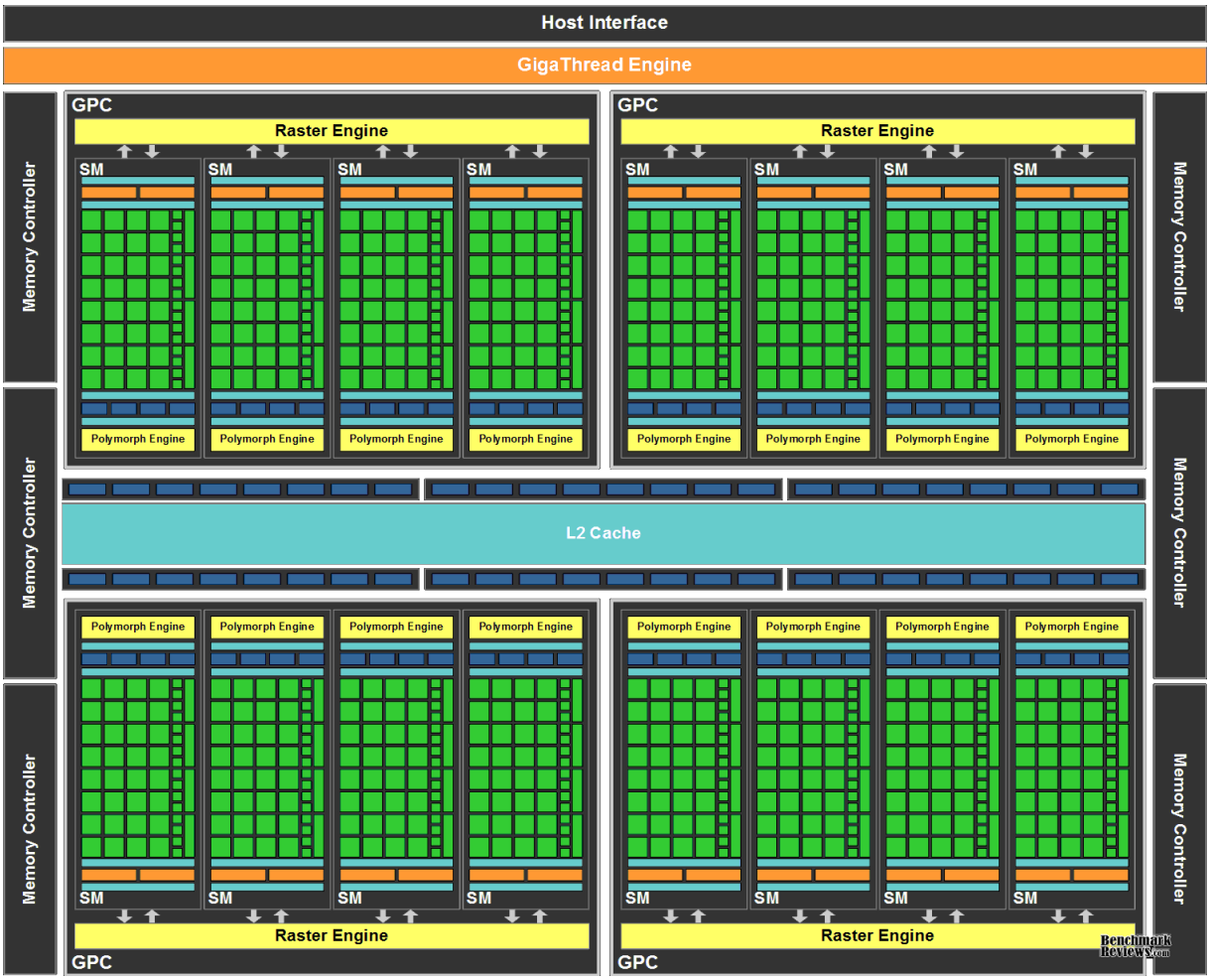
2.3.2 Τεχνολογία Πυρήνα

ALU-FPU

Οι πυρήνες 3ης γενιάς (Σχ. 2.7) υποστηρίζουν πλήρως πράξεις μεταξύ ακεραίων και δεκαδικών αριθμών από τις arithmetic logic unit (ALU) και floating point unit (FPU) αντίστοιχα. Η ALU εμφανίζεται βελτιωμένη προσφέροντας ακρίβεια 32-bit επεκτάσιμη μέχρι τα 64-bit σε σχέση με τα 24-bit της GT200. Υποστηρίζει πράξεις σύγκρισης, μετατροπής, μετακίνησης, μέτρησης πληθυσμού, λογικές κλπ. Η FPU εφαρμόζει το νέο πρωτόκολλο κινητής υποδιαστολής IEEE 754-2008 το οποίο επιτυγχάνει μέχρι και 4,2 φορές ταχύτερες πράξεις. Σε κάθε κύκλο του ρολογιού μπορούν να εκτελεστούν ως και 16 πράξεις πρόσθεσης-πολλαπλασιασμού διπλής ακρίβειας για κάθε SM.

Κάθε SM έχει 16 μονάδες φόρτωσης-αποθήκευσης που επιτρέπουν σε κάθε κύκλο του ρολογιού να υπολογίζονται οι διευθύνσεις προέλευσης και προορισμού για 16 threads.

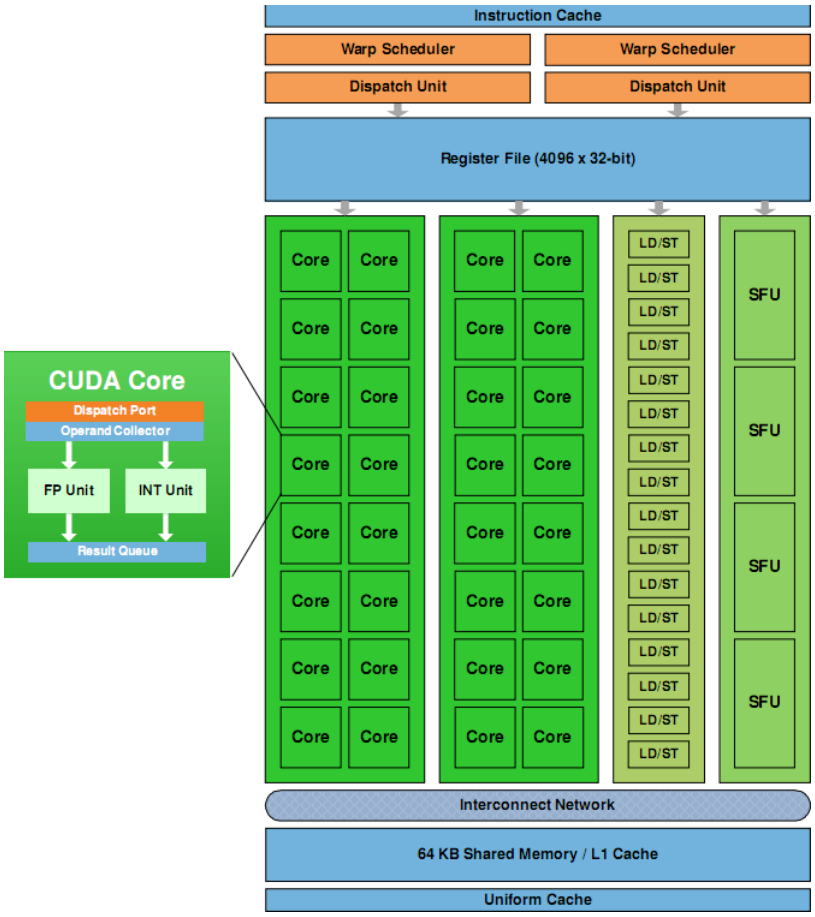
Υπάρχουν, επίσης, 4 μονάδες ειδικών συναρτήσεων (Special Function Unit - SFU) ανά SM που εκτελούν πράξεις όπως το ημίτονο, το συνημίτονο και η τετραγωνική ρίζα. Κάθε SFU εκτελέι μία εντολή ανά thread ανά κύκλο του ρολογιού (ένα warp εκτελείται σε 8 κύκλους του ρολογιού).



Σχήμα 2.6: Αρχιτεκτονική Fermi.

Διπλός warp scheduler

Ο διαχειριστής των warp σε μια Fermi (warp scheduler) ειναι διπλός. Ταυτόχρονα, συνοδεύεται από δύο μονάδες αποστολής εντολών. Ο διαχειριστής διαλέγει δύο warps και αποδίδει μία εντολή από το καθένα σε ένα σύνολο από 16 πυρήνες, 16 μονάδες φόρτωσης-αποθήκευσης ή 4 SFU. Αυτό σημαίνει ότι, πλέον, είναι δυνατό να εκτελούνται παράλληλα δύο warp ανεξάρτητα μεταξύ τους στον ίδιο SM επιτυγχάνοντας σημαντικότατη αύξηση στην αποδοτικότητα της GPU . Στο Σχ. 2.8 παρατίθεναι συνοπτικά οι αλλαγές στην τεχνολογία Fermi.



Σχήμα 2.7: SM και πυρήνας Fermi.

2.3.3 Σύστημα Μνήμης

Shared memory - L1 cache

Η shared memory, η οποιά είναι ταχύτατη (on-chip memory) και με σωστή χρήση μπορεί να αυξήσει σημαντικά την επιτάχυνση φτάνει στη Fermi τα 64 KB (4 φορές μεγαλύτερη των προγενέστερων). Αυτά μπορούν να χωριστούν σε 48 KB shared memory και 16 KB στη νέα L1 cashe ή αντίστροφα ανάλογα με τη δομή του κώδικα και το είδος μνήμης που χρησιμοποιεί. Για υπάρχουσες εφαρμογές που χρησιμοποιούσαν τη shared memory τριπλασιάζοντας τη χωρητικότητα της θα οδηγήσει σε σημαντικές βελτιώσεις στην απόδοση, ενώ για εφαρμογές που δεν έκαναν χρήση της shared κρατά γι' αυτή τα προγενέστερα 16 KB και επωφελείται πλήρως από την L1 cache.

L2 cache

Ένα άλλο είδος μνήμης που εισάγεται με τη Fermi είναι η L2 cashe με χωρητικότητα 768 KB. Είναι ενδιάμεση του kernel και της global μνήμης και χρησιμοποιείται

GPU	GT200	Fermi
Transistors	1.4 billion	3.0 billion
CUDA Cores	240	512
Double Precision Floating Point Capability	30 FMA ops / clock	256 FMA ops /clock
Single Precision Floating Point Capability	240 MAD ops / clock	512 FMA ops /clock
Warp schedulers (per SM)	1	2
Special Function Units (SFUs) / SM	2	4
Shared Memory (per SM)	16 KB	Configurable 48 KB or 16 KB
L1 Cache (per SM)	None	Configurable 16 KB or 48 KB
L2 Cache (per SM)	None	768 KB
ECC Memory Support	No	Yes
Concurrent Kernels	No	Up to 16
Load/Store Address Width	32-bit	64-bit

Σχήμα 2.8: Συνοπτική σύγκριση των αρχιτεκτονικών GT200 και Fermi.

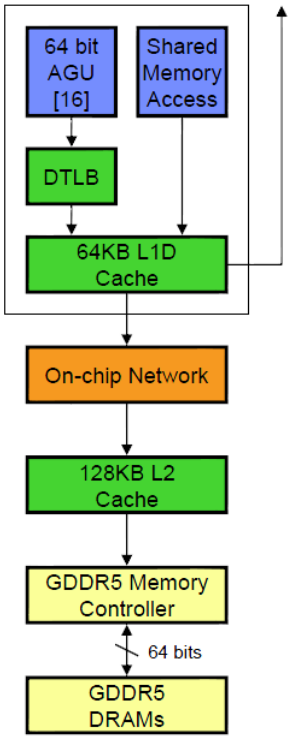
ανέξαρτητα από τις επιλογές του προγραμματιστή. Χρησιμεύει ως προσωρινή μνήμη για μια μεταβλητή όταν αυτή καλείται απο τη global πάνω από μία φορά. Έτσι, για τις επόμενες φορές που θα κληθεί η μεταβλητή, η L2 θα αποδώσει γρήγορα την τιμή της. Αλγόριθμοι, για τους οποίους οι διευθύνσεις δεδομένων δεν είναι γνωστές εκ των προτέρων και kernel που απαιτούν διαφορετικές SM να διαβάζουν τα ίδια δεδομένα επωφελούνται σημαντικά (Σχ. 2.9).

Constant και texture μνήμες

Στη βελτίωση της απόδοσης συντελούν οι constant και texture μνήμες. Στην constant μνήμη αποθηκεύονται πληροφορίες, από τον προγραμματιστή, πρίν την εκτέλεση ενός kernel και δεν μπορούν να αλλαχθούν κατά την εκτέλεσή του και ο αποθηκευτικός της χώρος είναι 64 KB. Η ανάγνωση μπορεί να είναι κοινή για μισό wrap χωρίς να χρειάζεται η ξεχωριστή ανάγνωση από κάθε thread. Δεν αποδίδει, όμως, όταν τα threads χρειάζονται να "διαβάσουν" διαφορετική πληροφορία καθώς χάνεται η παραλληλία της global και η διεργασία γίνεται σειριακά. Η texture μνήμη λειτουργεί ως προσωρινή ενδιάμεση μνήμη και χρησιμοποιείται όταν ο χρήστης επιθυμεί να διαβάσει κάτι από αυτήν και όχι από τη global αν υπάρχει ήδη εκεί. Αν δεν υπάρχει, τότε αντιγράφεται από τη global.

Διόρθωση σφαλμάτων

Άλλη μία καινοτομία είναι η υποστήριξη μνήμης για κώδικα διόρθωσης σφαλμάτων Error Correcting Code (ECC). Η ακτινοβολία που προέρχεται από τα υπολογιστικά συστήματα μπορεί να προκαλέσει μεταβολή σε ένα bit οδηγώντας σε λάθη μικρής εμβέλειας. Η ακτινοβολία αυτή αυξάνεται σε μεγάλα συστήματα (clusters). Η ECC

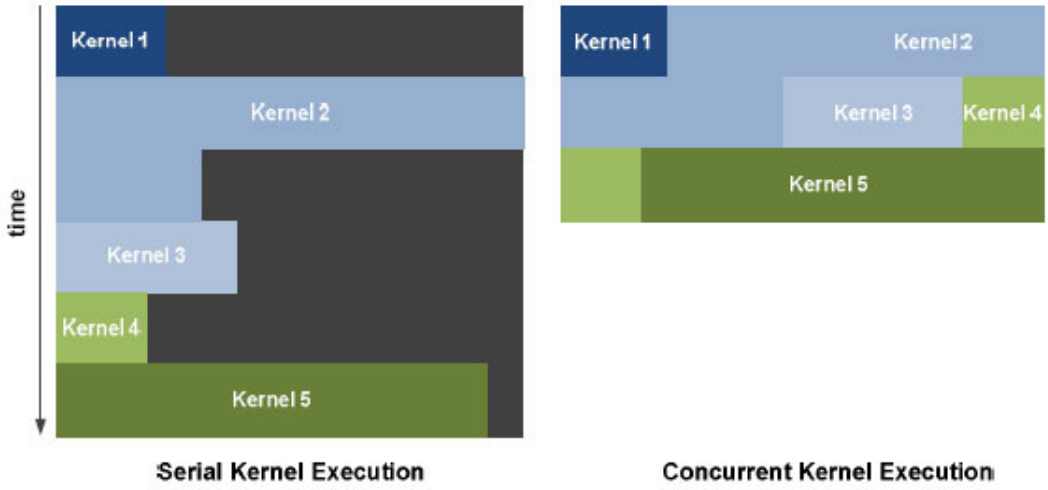


Σχήμα 2.9: Το σύστημα μνήμης στη Fermi αρχιτεκτονική.

διορθώνει αυτά τα λάθη και εξασφαλίζει τα λάθη περισσότερων bit να αναφερθούν ώστε το πρόγραμμα να προτιμήσει να ξαναεκτελεστεί παρά να επιτρέψει τη συνέχεια της εκτέλεσης με εσφαλμένα δεδομένα. Η λειτουργία αυτή υπάρχει στους register, στη shared memory, στις L1, L2 cache και την DRAM.

Thread Scheduler

Ο προγραμματιστής των threads (thread scheduler) της Fermi υποστηρίζει την ταυτόχρονη εκτέλεση διαφορετικών kernel της ίδιας εφαρμογής στη GPU. Επιτρέπεται έτσι σε προγράμματα που χρησιμοποιούν μικρά kernel να χρησιμοποιούν ολόκληρη τη GPU χωρίς να αφήνουν μεγάλο τμήμα της ανενεργό μέχρι να τελειώσει η εκτέλεση του kernel. Η σημασία αυτού φαίνεται στο Σχ. 2.10. Επιπλέον, ο scheduler αύξησε τον αριθμό ελεγχόμενων threads την ίδια χρονική στιγμή και εισήγαγε βελτιωμένες τεχνικές προγραμματισμού των block. Οι χρόνοι που απαιτούνται για τη μετάβαση από τη μια διεργασία της GPU στην άλλη (Context Switching) έχουν επιταχυνθεί ως 10 φόρες.



Σχήμα 2.10: Σχηματική απεικόνιση της επιτάχυνσης λόγω ταυτόχρονης εκτέλεσης των kernel.

2.4 Σύκριση και αναμενόμενα αποτελέσματα

Είναι σαφές ότι η τεχνολογία Fermi παρουσιάζει εμφανείς αναβαθμίσεις σχεδόν σε κάθε λειτουργικό κομμάτι μιας GPU και καινούργιες δυνατότητες. Αναμένεται, λοιπόν, ο κώδικας της παρούσας διπλωματικής να εκτελείται γρηγορότερα στην Tesla M2050 συγκριτικά με την GTX285. Κατά την παρουσίαση των χρόνων εκτέλεσης, στη συνέχεια, θα διαπιστωθεί η ορθότητα αυτής της υπόθεσης.

Κεφάλαιο 3

Διατύπωση - Διακριτοποίηση των Εξισώσεων Ροής

3.1 Διατύπωση των 2Δ Εξισώσεων Ροής

3.1.1 Συντηρητική Μορφή Εξισώσεων Euler

Η συμπιεστή, μη-συνεκτική και μόνιμη ροή μοντελοποιείται με τη βοήθεια των διαφορικών εξισώσεων Euler οι οποίες εκφράζουν τη διατήρηση της συνέχειας, της ορμής και της ενέργειας. Γράφοντάς τις σε συντηρητική μορφή (conservative form) έχουν ως εξής:

$$\frac{\partial \overrightarrow{U}}{\partial t} + \frac{\partial \overrightarrow{F}_i^{inv}}{\partial x_i} = 0, \qquad i = 1, 2 \tag{3.1}$$

και λαμβάνοντας υπόψη ότι το πρόβλημα μας αφορά τις δύο (2) διαστάσεις ισχύει: $(x_1, x_2) = (x, y)$. Οπότε η εξίσωση 3.1 μετατρέπεται στην:

$$\frac{\partial \overrightarrow{U}}{\partial t} + \frac{\partial \overrightarrow{F}_x^{inv}}{\partial x} + \frac{\partial \overrightarrow{F}_y^{inv}}{\partial y} = 0 \tag{3.2}$$

όπου με t συμβολίζεται ο χρόνος και το διάνυσμα των συντηρητικών μεταβλητών $\overrightarrow{U}$ έχει τη μορφή:

$$\overrightarrow{U} = \begin{bmatrix} \varrho \\ \varrho \overrightarrow{u} \\ E \end{bmatrix} = \begin{bmatrix} \varrho \\ \varrho u \\ \varrho v \\ E \end{bmatrix} \tag{3.3}$$

όπου:
ϱ: η πυκνότητα

$\overrightarrow{u}$: το διάνυσμα της ταχύτητας με συνιστώσα u κατά τον x άξονα και v κατά τον y
E: η ολική ενέργεια ανά μονάδα όγκου που δίνεται από τη σχέση:

$$E = \varrho e + \frac{1}{2}\varrho\left(u^2 + v^2\right) \tag{3.4}$$

Το διάνυσμα μη-συνεκτικής ροής $\overrightarrow{F}_i^{inv}$ δίνεται στο ακόλουθο μητρώο:

$$\overrightarrow{F}_i^{inv} = \begin{bmatrix} \varrho u_i \\ \varrho u_i \overrightarrow{u} + p\overrightarrow{u}_i \\ u_i\left(E + p\right) \end{bmatrix}, \qquad i = 1, 2 \tag{3.5}$$

και αναλύοντας στις δύο συνιστώσες της ταχύτητας ($u_1 = u$ και $u_2 = v$):

$$\overrightarrow{F}_x^{inv} = \begin{bmatrix} \varrho u \\ \varrho u^2 + p \\ \varrho uv \\ \left(E + p\right) u \end{bmatrix}, \qquad \overrightarrow{F}_y^{inv} = \begin{bmatrix} \varrho u \\ \varrho uv \\ \varrho v^2 + p \\ \left(E + p\right) v \end{bmatrix} \tag{3.6}$$

Παραδοχή τέλειου αέριου

Θεωρείται ότι το εργαζόμενο μέσο είναι τέλειο αέριο. Επομένως, ισχύει η καταστατική εξίσωση:

$$p = \varrho R_g T \tag{3.7}$$

όπου R_g είναι η σταθερά του τέλειου αερίου και υπολογίζεται σύμφωνα με τις ειδικές θερμοχωρητικότητες υπό σταθερή πίεση και όγκο, c_p και c_v αντίστοιχα, από την σχέση:

$$R_g = c_p - c_v \tag{3.8}$$

Ως θερμοχωρητικότητα ορίζεται η θερμότητα που απορροφά ή εκλύει ένα σώμα κατά τη μεταβολή της θερμοκρασίας του κατά έναν βαθμό Κελσίου (^oC). Η ειδική θερμοχωρητικότητα εκφράζει την αντίστοιχη ποσότητα θερμοκρασίας ανά μονάδα μάζας, ενώ οι δείκτες «υπό σταθερή πίεση» ή «υπό σταθερό όγκο» εκφράζουν μεταβολές με σταθερή πίεση ή όγκο, αντιστοιχα. Έτσι, οι μαθηματικές εκφράσεις της ειδικής θερμοχωρητικότητας ορίζονται:

$$c_p = \left(\frac{\partial q}{\partial T}\right)_p \ , \ \ c_v = \left(\frac{\partial q}{\partial T}\right)_v \tag{3.9}$$

ή ισοδύναμα με χρήση του πρώτου θερμοδυναμικού αξιώματος:

$$c_p = (\frac{\partial h}{\partial T})_p \ \ , \ \ c_v = (\frac{\partial e}{\partial T})_v \tag{3.10}$$

όπου με h συμβολίζεται η ενθαλπία του ρευστού ανά μονάδα μάζας. Η σχέση που συνδέει τη στατική ενθαλπία με την εσωτερική ενέργεια είναι η:

$$h = e + \frac{p}{\varrho} \tag{3.11}$$

Γενικά, οι ειδικές θερμοχωτητικότητες μπορούν να υπολογιστούν γνωρίζοντας δύο οποιαδήποτε θερμοδυναμικά μεγέθη. Στα τέλεια αέρια χρειάζεται μόνο η γνώση της θερμοκρασίας. Όμως η μεταβολή συναρτήσει της θερμοκρασίας είναι μικρή για τα τέλεια αέρια και για αυτό συνήθως παραλείπεται και οι ειδικές θερμοχωρητικότητες θεωρούνται σταθερές. Σύμφωνα με αυτά, ολοκληρώνοντας τις εξισώσεις 3.10 προκύπτει:

$$e = c_v T \ \ , \ \ h = c_p T \tag{3.12}$$

Τέλος, οριζεται ο εκθέτης ισεντροπικής μεταβολής γ σύμφωνα με τη σχέση:

$$\gamma = \frac{c_p}{c_v} \tag{3.13}$$

3.1.2 Αδιαστατοποίηση των Εξισώσεων Euler

Πριν γίνει η επίλυση των εξισώσεων Euler θα πραγματοποιηθεί η αδιαστατοποίησή τους. Με αυτό τον τρόπο, τα αποτελέσματα της επίλυσης δεν περιορίζονται σε μία συγκεκριμένη περίπτωση αλλά σε ένα σύνολο όμοιων περιπτώσεων.

Αρχικά, επιλέγονται καταλληλες παράμετροι αδιαστατοποίησης, τα μεγέθη αναφοράς (reference), σύμφωνα με τις οποίες προκύπτουν τα ακόλουθα αδιάστατα μεγέθη:

$$\breve{x}_i = \frac{x_i}{L_{ref}} \ \ , \ \ \breve{u}_i = \frac{u_i}{U_{ref}} \ \ , \ \ \breve{\varrho} = \frac{\varrho}{\varrho_{ref}} \tag{3.14}$$

όπου: L_{ref} είναι το μήκος αδιαστατοποίησης, U_{ref} η ταχύτητα αδιαστατοποίησης και ϱ_{ref} η πυκνότητα αδιαστατοποίησης, αντίστοιχα. Το σύμβολο "$\breve{}$" χρησιμοποιείται για να δηλώσει τα αντίστοιχα αδιάστατα μεγέθη.

Ακόμα επιλέγεται:

$$\breve{R}_g = \gamma - 1 \tag{3.15}$$

Δηλαδή:

$$(R_g)_{ref} = c_v \tag{3.16}$$

Στο συγκεκριμένο σημείο, ας αναφερθεί ότι είναι επιθυμητό η τελική μορφή των αδιάστατων εξισώσεων να διατηρήσει την αρχική μορφή της. Έτσι, τοποθετούνται τα αδιάστατα μεγέθη στις διαστατές εξισώσεις και χρησιμοποιείται η νέα μορφή τους για να προκύψουν τα μεγέθη αναφοράς των υπόλοιπων ποσοτήτων.

Αδιαστατοποίηση πίεσης

Ισχύει:

$$p_t = p + \frac{1}{2}\varrho\,|\overrightarrow{u}|^2 \Rightarrow |\overrightarrow{u}| = \sqrt{\frac{2}{\varrho}\,(p_t - p)} \Rightarrow |\breve{\overrightarrow{u}}| = \frac{1}{U_{ref}}\sqrt{\frac{2}{\breve{\varrho}}\frac{p_{ref}}{\varrho_{ref}}\,(\breve{p}_t - \breve{p})} \quad (3.17)$$

Όμως η μορφή των σχέσεων είναι επιθυμητό να παραμένει ίδια μετά την εισαγωγή των αδιάστατων μεγεθών, δηλαδή:

$$|\breve{\overrightarrow{u}}| = \sqrt{\frac{2}{\breve{\varrho}}\,(\breve{p}_t - \breve{p})} \quad (3.18)$$

Επιλέγουμε δηλαδή:

$$U_{ref} = \sqrt{\frac{p_{ref}}{\varrho_{ref}}} \Rightarrow p_{ref} = \varrho_{ref} U_{ref}^2 \quad (3.19)$$

Αδιαστατοποίηση εσωτερικής ενέργειας

$$e = c_v T \Rightarrow \breve{e} = \left(\frac{(c_v)_{ref} T_{ref}}{e_{ref}}\right) \breve{c}_v \breve{T} \longrightarrow e_{ref} = U_{ref}^2 \quad (3.20)$$

Αδιαστατοποίηση ενθαλπίας

$$h = c_p T \Rightarrow \breve{h} = \left(\frac{(c_p)_{ref} T_{ref}}{h_{ref}}\right) \breve{c}_p \breve{T} \longrightarrow h_{ref} = U_{ref}^2 \quad (3.21)$$

Αδιαστατοποίηση ολικής ενέργειας

$$E = \frac{p}{\gamma - 1} + \frac{1}{2}\varrho\,|\overrightarrow{u}|^2 \Rightarrow \breve{E} = \frac{1}{E_{ref}}\left[(p_{ref})\,\frac{\breve{p}}{\gamma - 1} + \frac{1}{2}\left(\varrho_{ref}U_{ref}^2\right)\breve{\varrho}|\overrightarrow{\breve{u}}|^2\right] \Rightarrow \quad (3.22)$$

$$\breve{E} = \frac{\varrho_{ref}U_{ref}^2}{E_{ref}}\left(\frac{\breve{p}}{\gamma - 1} + \frac{1}{2}\breve{\varrho}|\overrightarrow{\breve{u}}|^2\right) \longrightarrow \quad (3.23)$$

$$E_{ref} = \varrho_{ref}U_{ref}^2 \quad (3.24)$$

Τελικά η αδιαστατοποίηση των υπολοίπων θερμοδυναμικών μεγεθών (δηλαδή πλην αυτών που φαίνονται στις σχέσεις 3.14) γίνεται όπως παρακάτω:

$$\breve{p} = \frac{p}{\varrho_{ref}U_{ref}^2} \;\;,\;\; \breve{e} = \frac{e}{U_{ref}^2} \;\;,\;\; \breve{h} = \frac{h}{U_{ref}^2} \;\;,\;\; \breve{E} = \frac{E}{\varrho_{ref}U_{ref}^2} \quad (3.25)$$

Διατύπωση αδιάστατων εξισώσεων

Για πρακτικούς λόγους, οι εξισώσεις συνέχειας, ορμής και ενέργειας θα γραφούν στην μορφή της εξίσωσης 3.1, με χρήση των διανυσμάτων $\overrightarrow{U}$ και $\overrightarrow{F_i}$, όπως αυτά ορίστηκαν στις σχέσεις 3.3 και 3.5.

Εξίσωση συνέχειας

$$\frac{\partial U_1}{\partial t} + \frac{\partial (F_i)_1}{\partial x_i} = 0 \Rightarrow \quad (3.26)$$

$$\frac{(U_1)_{ref}}{t_{ref}}\frac{\partial \breve{U}_1}{\partial \breve{t}} + \frac{((F_i)_1)_{ref}}{L_{ref}}\frac{\partial (\breve{F}_i)_1}{\partial \breve{x}_i} = 0 \quad (3.27)$$

Όμως

$$U_1 \;\; = \varrho \longrightarrow \quad (U_1)_{ref} = \varrho_{ref} \quad (3.28)$$
$$(F_x)_1 = \varrho u \longrightarrow \quad (F_x)_{ref} = \varrho_{ref}U_{ref} \quad (3.29)$$
$$(F_y)_1 = \varrho u \longrightarrow \quad (F_y)_{ref} = \varrho_{ref}U_{ref} \quad (3.30)$$

Επομένως

$$\left(\frac{L_{ref}}{U_{ref}t_{ref}}\right)\frac{\partial \breve{U}_1}{\partial \breve{t}} + \frac{\partial (\breve{F}_i)_1}{\partial \breve{x}_i} = 0 \quad (3.31)$$

Ορίζοντας κατάλληλα το χρονικό μέγεθος αναφοράς t_{ref}, η ποσότητα μπροστά από τον χρονικό όρο μπορεί να απαλειφθεί, ενώ παράλληλα έχει επιτευχθεί και ο αρχικός στόχος, η διαστατή και η αδιάστατη εξίσωση να είναι ίδιες. Επιλέγεται, συνεπώς :

$$t_{ref} = \frac{L_{ref}}{U_{ref}} \quad , \quad \breve{t} = \frac{t}{L_{ref}/U_{ref}} \tag{3.32}$$

Διατήρηση ορμής Οι εξισώσεις διατήρησης της ορμής είναι οι δύο εξισώσεις, για τις συνιστώσες x και y. Ακολουθεί η εξαγωγή μόνο της αδιάστατης κατά x διατήρησης της ορμής. Για την y συνιστώσα ακολουθείται η ίδια λογική.

$$\frac{\partial U_2}{\partial t} + \frac{\partial (F_i)_2}{\partial x_i} = 0 \Rightarrow \tag{3.33}$$

$$\frac{(U_2)_{ref}}{t_{ref}} \frac{\partial \breve{U}_2}{\partial \breve{t}} + \frac{((F_i)_2)_{ref}}{L_{ref}} \frac{\partial (\breve{F}_i)_2}{\partial \breve{x}_i} = 0 \tag{3.34}$$

$$U_2 = \varrho u \longrightarrow (U_2)_{ref} = \varrho_{ref} U_{ref} \tag{3.35}$$

$$(F_x)_2 = \varrho u^2 + p \longrightarrow ((F_x)_2)_{ref} = \varrho_{ref} U_{ref}^2 \tag{3.36}$$

$$(F_y)_2 = \varrho uv \longrightarrow ((F_y)_2)_{ref} = \varrho_{ref} U_{ref}^2 \tag{3.37}$$

Οπότε η αδιάστατη εξίσωση ορμής κατά x, με απλή αντικατάσταση, παίρνει τη μορφή:

$$\frac{\partial \breve{U}_2}{\partial \breve{t}} + \frac{\partial (\breve{F}_i)_2}{\partial \breve{x}_i} = 0 \tag{3.38}$$

Διατήρηση ενέργειας

$$\frac{\partial U_4}{\partial t} + \frac{\partial (F_i)_4}{\partial x_i} = 0 \Rightarrow \tag{3.39}$$

$$\frac{(U_4)_{ref}}{t_{ref}} \frac{\partial \breve{U}_4}{\partial \breve{t}} + \frac{((F_i)_4)_{ref}}{L_{ref}} \frac{\partial (\breve{F}_i)_4}{\partial \breve{x}_i} = 0 \tag{3.40}$$

Όμως

$$U_4 = E \longrightarrow (U_4)_{ref} = \varrho_{ref} U_{ref}^2 \tag{3.41}$$

$$(F_x)_4 = (E+p)u \longrightarrow ((F_x)_4)_{ref} = \varrho_{ref} U_{ref}^3 \tag{3.42}$$

$$(F_y)_4 = (E+p)v \longrightarrow ((F_y)_4)_{ref} = \varrho_{ref} U_{ref}^3 \tag{3.43}$$

Και εδώ, με απλή αντικατάσταση προκύπτει:

$$\frac{\partial \breve{U}_4}{\partial \breve{t}} + \frac{\partial (\breve{F}_i)_4}{\partial \breve{x}_i} = 0 \tag{3.44}$$

Η διαδικασία της αδιαστατοποίησης βασίστηκε στα μεγέθη αναφοράς L_{ref}, U_{ref} και ϱ_{ref}. Η επιλογή των μεγεθών αυτών είναι απόφαση του χρήστη και είναι άμεσα συνδεδεμένη με το είδος του προβλήματος προς επίλυση. Σε προβλήματα εξωτερικής αεροδυναμικής, ως μεγέθη αναφοράς μπορούμε να χρησιμοποιήσουμε τις τιμές των αντίστοιχων μεγεθών της επ΄ άπειρον ροής.

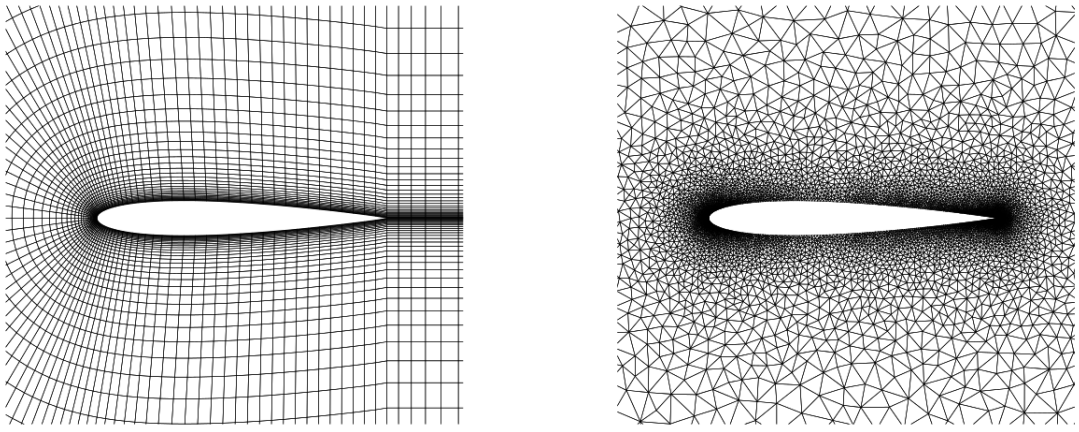
Στη συνέχεια, οποιαδήποτε αναφορά στις εξισώσεις αναφέρεται στις αδιάστατες εξισώσεις, αλλά για λόγους συντομίας παραλείπεται το αντίστοιχο σύμβολο σε κάθε μέγεθος.

3.2 Διακριτοποιήσεις

Για τον υπολογισμό των ροϊκών μεγεθών θα επιλύσουμε αριθμητικά τις εξισώσεις Euler (3.2) με τη μέθοδο των πεπερασμένων όγκων (finite volume).

3.2.1 Διακριτοποίηση του χωρίου ροής

Για την επίλυση του προβλήματος ροής είναι απαραίτητη η κατασκευή ενός πλέγματος στο χώρο γύρω ή/και μέσα στην αεροδυναμική μορφή που μελετάται για να υπολογιστούν τα ροϊκά μεγέθη στα στοιχεία του. Τα πλέγματα ανάλογα με τη δομή των στοιχείων τους μπορούν να διακριθούν σε δομημένα και μη-δομημένα (Σχ. 3.1). Στην παρούσα περίπτωση, θα χρησιμοποιήσουμε **μη-δομημένα πλέγματα τριγωνικών στοιχείων**.



Σχήμα 3.1: Σύγκριση ενός δομημένου και ενός μη-δομημένου πλέγματος γύρω απο διδιάστατη αεροτομή.

Για την επίλυση του προβλήματος με τη μέθοδο των πεπερασμένων όγκων (όπως προαναφέρθηκε) επιλέχθηκε η αποθήκευση των μεγεθών της ροής στο κέντρο

βάρους του εκάστοτε τριγώνου (κεντροκυψελικό, cell-centered) και όχι στις κορυφές (κεντροκομβικό, vertex-centered). (Σχ. 1.9)

3.2.2 Διακριτοποίηση των εξισώσεων ροής

Θεωρείται, λοιπόν, ως όγκος ελέγχου το κάθε τρίγωνο. Ολοκληρώνοντας την 3.2 στο εμβαδό τυχαίου τριγώνου με κέντρο βάρους P και εμβαδό Ω_P, προκύπτει:

$$\iint\limits_{\Omega_P} \left[\frac{\partial \overrightarrow{U}}{\partial t} + \frac{\partial \overrightarrow{F}_x^{inv}}{\partial x} + \frac{\partial \overrightarrow{F}_y^{inv}}{\partial y} \right] dx\, dy = 0 \tag{3.45}$$

και ορίζοντας ως $\underline{\overrightarrow{F}} = \left[\overrightarrow{F}_x^{inv}, \quad \overrightarrow{F}_y^{inv} \right]$:

$$\iint\limits_{\Omega_P} \frac{\partial \overrightarrow{U}}{\partial t}\, dx\, dy + \iint\limits_{\Omega_P} \nabla \cdot \underline{\overrightarrow{F}}\, dx\, dy = 0 \tag{3.46}$$

Αν τα ροϊκά μεγέθη δεν μεταβάλλονται με το χρόνο (steady πρόβλημα) τότε ισχύει: $\iint\limits_{\Omega_P} \frac{\partial \overrightarrow{U}}{\partial t}\, dx\, dy = 0$ και μπορεί να γραφεί:

$$\iint\limits_{\Omega_P} \nabla \cdot \underline{\overrightarrow{F}}\, dx\, dy = 0 \tag{3.47}$$

Κάνοντας χρήση του θεωρήματος Green-Gauss από το επιφανειακό ολοκλήρωμα της 3.47 είναι δυνατό να προκύψει το επικαμπύλιο (κλειστό) ολοκλήρωμα:

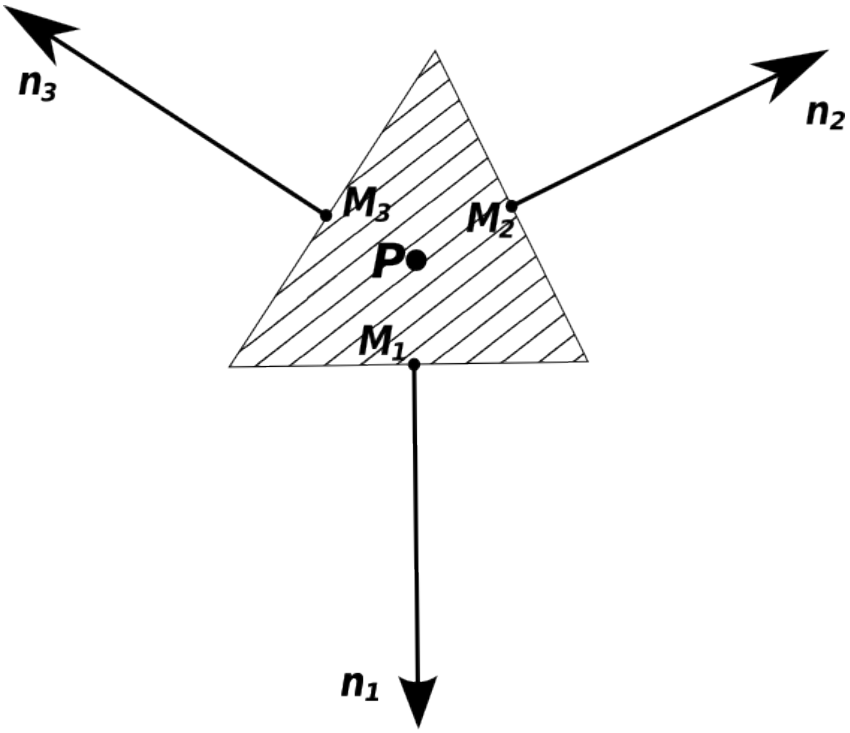
$$\oint\limits_{\partial\Omega_P} \underline{\overrightarrow{F}} \cdot \overrightarrow{\widehat{n}}\, dS = 0 \tag{3.48}$$

όπου $\overrightarrow{\widehat{n}}$ το κάθετο προς τα έξω μοναδιάιο διάνυσμα σε κάθε θέση της καμπύλης S (όριο του όγκου ελέγχου P). Στο πλαίσιο της παρούσας διπλωματικής εργασίας, ο όγκος ελέγχου είναι το ίδιο το τρίγωνο P, επομένως η 3.48 ισοδυναμεί με την:

$$\sum_{i=1}^{3} \underline{\overrightarrow{F}} \cdot \overrightarrow{n_i} = 0, \qquad \forall\, \Omega_P \tag{3.49}$$

Το άθροισμα αναφέρεται στις τρεις πλευρές του τριγώνου και το $\overrightarrow{n_i}$ έχει, πλέον, ως μέτρο το μήκος της πλευράς του τριγώνου στην οποία αναφέρεται (Σχ. 3.2). Ορίζοντας ως ροή (flux) το μέγεθος $\Phi = \underline{\overrightarrow{F}}\, \overrightarrow{n}$:

$$\sum_{i=1}^{3} \Phi_{PQ_i} = 0, \qquad \forall\, \Omega_P \tag{3.50}$$



Σχήμα 3.2: Τρίγωνο- Ογκος ελέγχου P με σχεδιασμένα τα κάθετα n_i στα μέσα M_i ($i = 1 \cdots 3$) των πλευρών του τριγώνου προς την εξωτερική πλευρά με μέτρο ίσο με το μήκος των πλευρών στις οποίες αντιστοιχούν.

συμβολίζοντας με P το βαρύκεντρο του τριγώνου στο οποίο έγινε η ολοκλήρωση και με Q_i τα τρία βαρύκεντρα των γειτονικών τριγώνων.
Το Φ_{PQ_i} σύμφωνα με τη **μέθοδο διαχωρισμού του διανύσματος ροής** (**flux vector splitting** - [23]) μπορεί να προσεγγιστεί από τη σχέση:

$$\Phi_{PQ_i} = A^+_{PQ_i} \overrightarrow{U}^L_{PQ_i} + A^-_{PQ_i} \overrightarrow{U}^R_{PQ_i} \tag{3.51}$$

Κατά σύμβαση, θεωρείται δεξιόστροφη κίνηση στο περίγραμμα του τριγώνου. Επομένως, ως αριστερή πλευρά (Left) θεωρείται το τρίγωνο P ενώ ως δεξιά (Right) οι γείτονες Q_i. Για ακρίβεια πρώτης τάξης θα γίνει χρήση των σχέσεων:

$$\overrightarrow{U}^L_{PQ_i} = \overrightarrow{U}_P \tag{3.52}$$

$$\overrightarrow{U}^R_{PQ_i} = \overrightarrow{U}_{Q_i} \quad , \qquad i = 1, 2, 3 \tag{3.53}$$

Σε επόμενη παράγραφο θα παρουσιαστεί πως μπορεί να γίνει χρήση δεύτερης τάξης ακρίβειας.
Τα $A^+_{PQ_i}$ και $A^-_{PQ_i}$ της σχέσης 3.51 ορίζονται με τη χρήση των θετικών και αρνητικών ιδιοτιμών του Ιακωβιανού μητρώου:

$$\underline{A}_{PQ_i} = \frac{\partial \left(\underline{\overrightarrow{F}} \overrightarrow{n_i} \right)}{\partial \overrightarrow{U_P}} = \underline{A}^+_{PQ_i} + \underline{A}^-_{PQ_i} \tag{3.54}$$

Ο τρόπος υπολογισμού της 3.54 παρουσιάζεται παρακάτω.
Ορίζεται το ιακωβιανό μητρώο του διανύσματος $\underline{\overrightarrow{F}}$ (3.5) ως προς τις συντηρητικές μεταβλητές $\overrightarrow{U}$ (3.3).

$$A_x \hat{=} \frac{\partial \overrightarrow{F_x}}{\partial \overrightarrow{U}}^{inv} \quad , \quad A_y \hat{=} \frac{\partial \overrightarrow{F_y}}{\partial \overrightarrow{U}}^{inv} \tag{3.55}$$

Ορίζεται επίσης:

$$\begin{aligned} \underline{A} \hat{=} A \cdot \overrightarrow{n} = A_x n_x + A_y n_y \Rightarrow \underline{A} = \frac{\partial \overrightarrow{F_x}}{\partial \overrightarrow{U}}^{inv} n_x + \frac{\partial \overrightarrow{F_y}}{\partial \overrightarrow{U}}^{inv} n_y = \frac{\partial \underline{\overrightarrow{F}}}{\partial \overrightarrow{U}}^{inv} \cdot \overrightarrow{n} \Rightarrow \\ \Rightarrow \underline{A} = \frac{\partial \left(\underline{\overrightarrow{F}}^{inv} \cdot \overrightarrow{n} \right)}{\partial \overrightarrow{U}} = \frac{\partial \overrightarrow{H}}{\partial \overrightarrow{U}}^{inv} \end{aligned} \tag{3.56}$$

Στην ανάλυση που ακολουθεί ο δείκτης "inv" παραλείπεται καθώς οι εξισώσεις αφορούν μόνο μη-συνεκτική ροή. Με την αποδοχή της θεώρησης των τελείων αερίων η καταστατική εξίσωση αυτών εύκολα αποδεικνύεται ότι ικανοποιεί την έκφραση $p = \varrho f(e)$. Εξαιτίας του συμπεράσματος αυτού και του τρόπου ορισμού των ιακωβιανών μητρώων A_x, A_y, τα διανύσματα ροής $\overrightarrow{F_x}$, $\overrightarrow{F_y}$ είναι ομογενείς συναρτήσεις πρώτου βαθμού. Κύρια ιδιότητα των συναρτήσεων αυτού του τύπου είναι:

$$\overrightarrow{F_x} = A_x \overrightarrow{U} \quad , \quad \overrightarrow{F_y} = A_y \overrightarrow{U} \tag{3.57}$$

Χρησιμοποιώντας όμως τη πορεία που ακολουθήθηκε για την εξαγωγή της σχέσης 3.56 εύκολα προκύπτει:

$$\overrightarrow{H} = \underline{A} \overrightarrow{U} \tag{3.58}$$

Ιακωβιανό μητρώο $\underline{A}$

Ισχύει ότι:

$$\overrightarrow{H} = \underline{\overrightarrow{F}} \cdot \overrightarrow{n} = \begin{bmatrix} \varrho \left(\overrightarrow{u} \cdot \overrightarrow{n} \right) \\ \varrho u \left(\overrightarrow{u} \cdot \overrightarrow{n} \right) + p n_x \\ \varrho v \left(\overrightarrow{u} \cdot \overrightarrow{n} \right) + p n_y \\ (E + p) \left(\overrightarrow{u} \cdot \overrightarrow{n} \right) \end{bmatrix} \tag{3.59}$$

Για την εύρεση του ιακωβιανού μητρώου εξυπηρετεί η μετονομασία των συντηρητικών μεταβλητών, δηλαδή:

$$\overrightarrow{U} = \begin{bmatrix} \varrho & \varrho u & \varrho v & E \end{bmatrix}^T = \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 & \mu_4 \end{bmatrix}^T \tag{3.60}$$

Εκτελώντας λοιπόν τις πράξεις το διάνυσμα ροής $\overrightarrow{H}$ γράφεται:

$$\overrightarrow{H} = \begin{bmatrix} \mu_2 n_x + \mu_3 n_y \\ \frac{\mu_2}{\mu_1}\left(\mu_2 n_x + \mu_3 n_y\right) + (\gamma-1)\left[\mu_4 - \frac{1}{2}\frac{\mu_2^2+\mu_3^2}{\mu_1}\right] n_x \\ \frac{\mu_3}{\mu_1}\left(\mu_2 n_x + \mu_3 n_y\right) + (\gamma-1)\left[\mu_4 - \frac{1}{2}\frac{\mu_2^2+\mu_3^2}{\mu_1}\right] n_y \\ \left(\gamma\mu_4 - \frac{\gamma-1}{2}\frac{\mu_2^2+\mu_3^2}{\mu_1}\right)\frac{\mu_2 n_x + \mu_3 n_y}{\mu_1} \end{bmatrix} \tag{3.61}$$

Παραγωγίζοντας κάθε όρο του διανύσματος ροής και έπειτα από τις απαραίτητες πράξεις προκύπτει:

$$\frac{\partial H_1}{\partial \mu_1} = 0$$
$$\frac{\partial H_1}{\partial \mu_2} = n_x$$
$$\frac{\partial H_1}{\partial \mu_3} = n_y$$
$$\frac{\partial H_1}{\partial \mu_4} = 0$$

$$\frac{\partial H_2}{\partial \mu_1} = -u\left(\overrightarrow{u}\cdot\overrightarrow{n}\right) + \frac{\gamma-1}{2}\left(u^2+v^2\right) n_x$$
$$\frac{\partial H_2}{\partial \mu_2} = \overrightarrow{u}\cdot\overrightarrow{n} + (2-\gamma) u n_x$$
$$\frac{\partial H_2}{\partial \mu_3} = u n_y - (\gamma-1) v n_x$$
$$\frac{\partial H_2}{\partial \mu_4} = (\gamma-1) n_x$$

$$\frac{\partial H_3}{\partial \mu_1} = -v\left(\overrightarrow{u}\cdot\overrightarrow{n}\right) + \frac{\gamma-1}{2}\left(u^2+v^2\right) n_y$$
$$\frac{\partial H_3}{\partial \mu_2} = v n_x - (\gamma-1) u n_y$$
$$\frac{\partial H_3}{\partial \mu_3} = \overrightarrow{u}\cdot\overrightarrow{n} + (2-\gamma) v n_y$$
$$\frac{\partial H_3}{\partial \mu_4} = (\gamma-1) n_y$$

$$\frac{\partial H_4}{\partial \mu_1} = \left[-\gamma\frac{E}{\varrho} + (\gamma-1)\left(u^2+v^2\right)\right](\overrightarrow{u}\cdot\overrightarrow{n})$$
$$\frac{\partial H_4}{\partial \mu_2} = \left[\frac{\gamma E}{\varrho} - \frac{\gamma-1}{2}\left(u^2+v^2\right)\right]n_x - (\gamma-1)u\,(\overrightarrow{u}\cdot\overrightarrow{n})$$
$$\frac{\partial H_4}{\partial \mu_3} = \left[\frac{\gamma E}{\varrho} - \frac{\gamma-1}{2}\left(u^2+v^2\right)\right]n_y - (\gamma-1)v\,(\overrightarrow{u}\cdot\overrightarrow{n})$$
$$\frac{\partial H_4}{\partial \mu_4} = \gamma\,(\overrightarrow{u}\cdot\overrightarrow{n})$$

Τελικά,

$$\underline{A}\,(:,1) = \begin{bmatrix} 0 \\ -u\,(\overrightarrow{u}\cdot\overrightarrow{n}) + \frac{\gamma-1}{2}\left(u^2+v^2\right)n_x \\ -v\,(\overrightarrow{u}\cdot\overrightarrow{n}) + \frac{\gamma-1}{2}\left(u^2+v^2\right)n_y \\ \left[-\gamma\frac{E}{\varrho} + (\gamma-1)\left(u^2+v^2\right)\right](\overrightarrow{u}\cdot\overrightarrow{n}) \end{bmatrix}$$

$$\underline{A}\,(:,2) = \begin{bmatrix} n_x \\ \overrightarrow{u}\cdot\overrightarrow{n} + (2-\gamma)un_x \\ vn_x - (\gamma-1)un_y \\ \left[\frac{\gamma E}{\varrho} - \frac{\gamma-1}{2}\left(u^2+v^2\right)\right]n_x - (\gamma-1)u\,(\overrightarrow{u}\cdot\overrightarrow{n}) \end{bmatrix} \tag{3.62}$$

$$\underline{A}\,(:,3) = \begin{bmatrix} n_y \\ un_y - (\gamma-1)vn_x \\ \overrightarrow{u}\cdot\overrightarrow{n} + (2-\gamma)vn_y \\ \left[\frac{\gamma E}{\varrho} - \frac{\gamma-1}{2}\left(u^2+v^2\right)\right]n_y - (\gamma-1)v\,(\overrightarrow{u}\cdot\overrightarrow{n}) \end{bmatrix}$$

$$\underline{A}\,(:,4) = \begin{bmatrix} 0 \\ (\gamma-1)n_x \\ (\gamma-1)n_y \\ \gamma\,(\overrightarrow{u}\cdot\overrightarrow{n}) \end{bmatrix}$$

Οι ιδιοτιμές του μητρώου $\underline{A}$ έχουν υπολογιστεί και δίνονται παρακάτω:

$$\lambda_1 = \overrightarrow{u}\cdot\overrightarrow{n}$$
$$\lambda_2 = \overrightarrow{u}\cdot\overrightarrow{n}$$
$$\lambda_3 = \left(\overrightarrow{u}\cdot\overrightarrow{\hat{n}} + c\right)|\overrightarrow{n}|$$
$$\lambda_4 = \left(\overrightarrow{u}\cdot\overrightarrow{\hat{n}} - c\right)|\overrightarrow{n}|$$

Όμοια παρατίθενται παρακάτω τα αριστερά και δεξιά ιδιοδιανύσματα του μητρώου

$\underline{A}$. Τα δεξιά ιδιοδιανύσματα πρέπει να ικανοποιούν τη σχέση $(A - \lambda_k I)\, r_k = 0$, σε αντίθεση με τα αριστερά που πρέπει $l_k\,(A - \lambda_k I) = 0$. Στις σχέσεις που προηγήθηκαν με I συμβολίζεται ο μοναδιαίος πίνακας.

$$
\begin{aligned}
r_1 &= \begin{bmatrix} 1 & u & v & \frac{1}{2}(u^2+v^2) \end{bmatrix}^T \\
r_2 &= \begin{bmatrix} 0 & \hat{n}_y & -\hat{n}_x & \hat{n}_y u - \hat{n}_x v \end{bmatrix}^T \\
r_3 &= \begin{bmatrix} \frac{1}{c} & \frac{u}{c} + \hat{n}_x & \frac{v}{c} + \hat{n}_y & \frac{1}{2c}(u^2+v^2) + \frac{c}{\gamma-1} + \overrightarrow{u} \cdot \overrightarrow{\hat{n}} \end{bmatrix}^T \\
r_4 &= \begin{bmatrix} \frac{1}{c} & \frac{u}{c} - \hat{n}_x & \frac{v}{c} - \hat{n}_y & \frac{1}{2c}(u^2+v^2) + \frac{c}{\gamma-1} - \overrightarrow{u} \cdot \overrightarrow{\hat{n}} \end{bmatrix}^T \\
l_1 &= \begin{bmatrix} 1 - \frac{(\gamma-1)(u^2+v^2)}{2c^2} & \frac{(\gamma-1)u}{c^2} & \frac{(\gamma-1)v}{c^2} & -\frac{(\gamma-1)}{c^2} \end{bmatrix} \\
l_2 &= \begin{bmatrix} \hat{n}_x v - \hat{n}_y u & \hat{n}_y & -\hat{n}_x & 0 \end{bmatrix} \\
l_3 &= \begin{bmatrix} -\frac{\overrightarrow{u} \cdot \overrightarrow{\hat{n}}}{2} + \frac{(\gamma-1)(u^2+v^2)}{4c} & \frac{\hat{n}_x}{2} - \frac{(\gamma-1)u}{2c} & \frac{\hat{n}_y}{2} - \frac{(\gamma-1)v}{2c} & \frac{\gamma-1}{2c} \end{bmatrix} \\
l_4 &= \begin{bmatrix} \frac{\overrightarrow{u} \cdot \overrightarrow{\hat{n}}}{2} + \frac{(\gamma-1)(u^2+v^2)}{4c} & -\frac{\hat{n}_x}{2} - \frac{(\gamma-1)u}{2c} & -\frac{\hat{n}_y}{2} - \frac{(\gamma-1)v}{2c} & \frac{\gamma-1}{2c} \end{bmatrix}
\end{aligned}
$$

Εύκολα, λοιπόν, γνωρίζοντας τα παραπάνω μπορούμε να γράψουμε το μητρώο $\underline{A}$ στην ακόλουθη μορφή:

$$\underline{A} = P \Lambda P^{-1} \tag{3.63}$$

με Λ διαγώνιο πίνακα με στοιχεία τις ιδιοτιμές του $\underline{A}$ και

$$P = \begin{bmatrix} r_1 & r_2 & r_3 & r_4 \end{bmatrix} \ , \ P^{-1} = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \end{bmatrix}$$

Τελικά, προκύπτουν τα ζητούμενα μητρώα της σχέσης 3.54

$$\underline{A}^{+}_{PQ_i} = P \Lambda^{+} P^{-1} \ \ , \ \ \underline{A}^{-}_{PQ_i} = P \Lambda^{-} P^{-1} \tag{3.64}$$

$$\left| \underline{A}_{PQ_i} \right| = \underline{A}^{+}_{PQ_i} - \underline{A}^{-}_{PQ_i} \tag{3.65}$$

Στις παραπάνω σχέσεις το μητρώο Λ^{+} περιέχει τις θετικές ιδιοτιμές, ενώ το Λ^{-} τις αρνητικές.

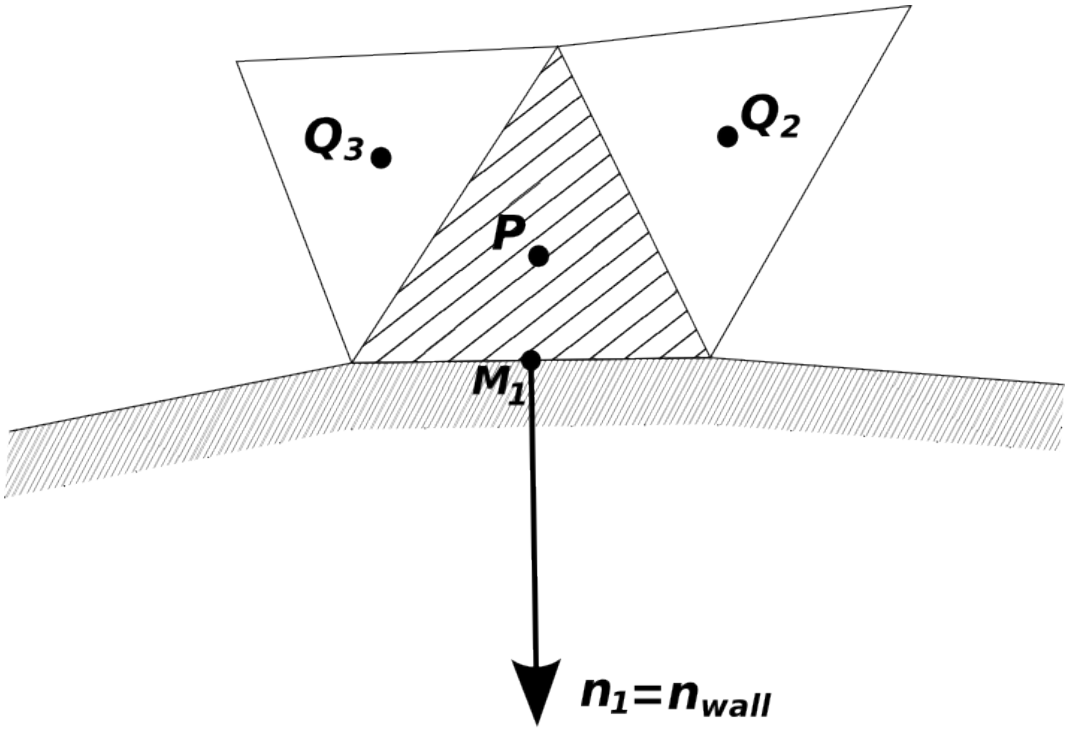
3.3 Επιβολή οριακών συνθηκών

3.3.1 Στερεά τοιχώματα

Το μη-συνεκτικό διάνυσμα ροής $\left(\overrightarrow{\Phi}_{wall}^{inv}\right)$ δια μέσου τού τοιχώματος είναι:

$$\overrightarrow{\Phi}_{wall} = \overrightarrow{H}\left(\overrightarrow{U}_{wall}, \overrightarrow{n}_{wall}\right) \tag{3.66}$$

Από τη σχέση 3.59 προκύπτει:



Σχήμα 3.3: Όγκος ελέγχου που εφάπτεται στο στερεό τοίχωμα.

$$\overrightarrow{\Phi}_{wall}^{inv} = \begin{bmatrix} \varrho\left(\overrightarrow{u}\cdot\overrightarrow{n}\right) \\ \varrho u\left(\overrightarrow{u}\cdot\overrightarrow{n}\right) + pn_x \\ \varrho v\left(\overrightarrow{u}\cdot\overrightarrow{n}\right) + pn_y \\ \left(E+p\right)\left(\overrightarrow{u}\cdot\overrightarrow{n}\right) \end{bmatrix}_{wall} \tag{3.67}$$

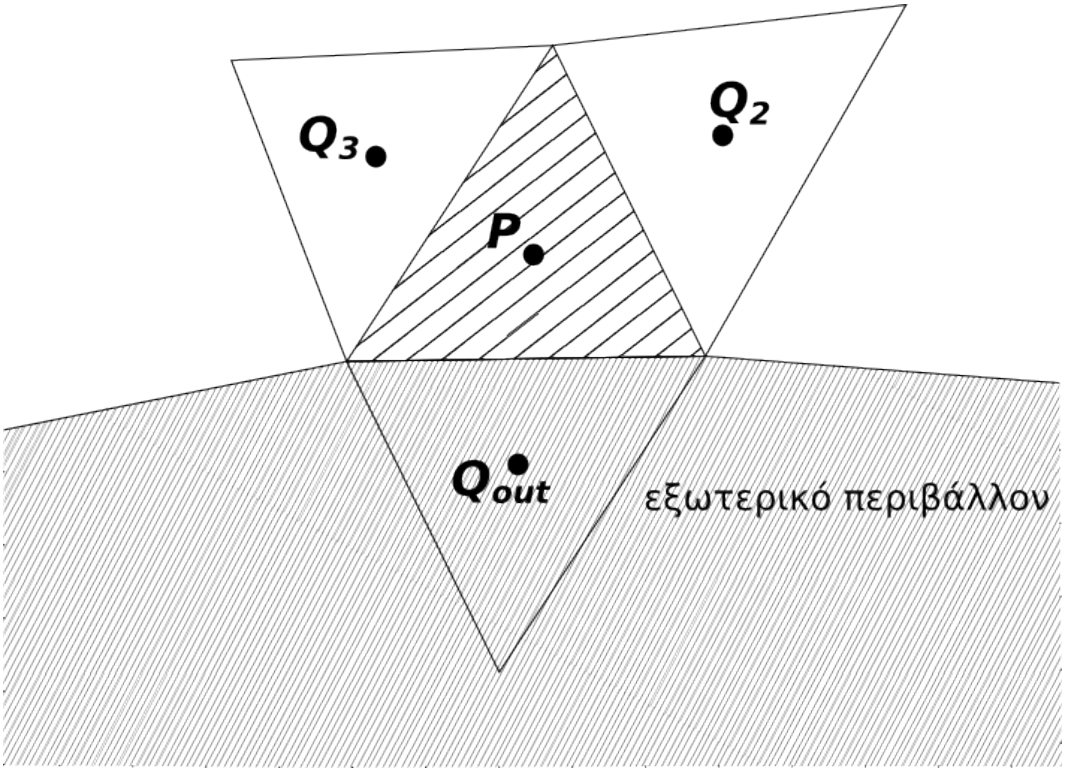
Για ατριβές ρευστό, για τις πλευρές που εφάπτονται στο τοίχωμα ισχύουν οι συνθήκες μη-εισχώρησης. Η συνθήκη μη-εισχώρησης υπαγορεύει στο ρευστό να κινείται εφαπτομενικά του τοιχώματος, δηλαδή

$$\overrightarrow{u}\cdot\overrightarrow{n} = 0 \tag{3.68}$$

Το μη-συνεκτικό διάνυσμα ροής διαμέσου του τοιχώματος δίνεται από την ακόλουθη έκφραση.

$$\overrightarrow{\Phi}_{wall} = \begin{bmatrix} 0 \\ pn_x \\ pn_y \\ 0 \end{bmatrix}_{wall} \tag{3.69}$$

3.3.2 Εξωτερικό όριο - "άπειρο"



Σχήμα 3.4: Όγκος ελέγχου που εφάπτεται στο εξωτερικό περιβάλλον.

Στη συγκεκριμένη περίπτωση, θεωρούμε πως στο "άπειρο" υπάρχει φανταστικό τρίγωνο πλήρως συμμετρικό με το τρίγωνο - όγκο ελέγχου ώς προς την πλευρά που εφάπτεται στο εξωτερικό όριο (Σχ. 3.4). Για τις παρακάτω εξισώσεις, θα θεωρηθεί ο δείκτης "out" για το συμμετρικό τρίγωνο.

Σε εφαρμογές μεμονωμένης αεροτομής, δηλαδή σε εφαρμογές που η ροή γύρω από την αεροτομή επηρεάζεται μόνο από την παρουσία της ίδιας, τα μεγέθη που συνήθως δίνονται για το κλείσιμο των εξισώσεων ροής είναι η πυκνότητα, το διάνυσμα της ταχύτητας και ο αριθμός $Mach$ της επ' άπειρον ροής $(\varrho_{far},\ |\overrightarrow{u}_{far}|,\ \theta_{far},\ M_{far})$. Επομένως:

$$\varrho_{out} = \varrho_{far} \tag{3.70}$$

$$(\varrho u)_{out} = \varrho_{far} \, |\overrightarrow{u}_{far}| \cos\theta_{far} \tag{3.71}$$

$$(\varrho v)_{out} = \varrho_{far} \, |\overrightarrow{u}_{far}| \sin\theta_{far} \tag{3.72}$$

$$\left.\begin{aligned} M_{far} = \frac{|\overrightarrow{u}_{far}|}{c} = \frac{|\overrightarrow{u}_{far}|}{\sqrt{\gamma(\gamma-1)T_{far}}} \\ \frac{p_{out}}{\varrho_{far}} = (\gamma-1)T_{far} \end{aligned}\right\} \Rightarrow \tag{3.73}$$

$$M_{far} = |\overrightarrow{u}_{far}| \sqrt{\frac{\varrho_{far}}{p_{out}}} \Rightarrow p_{out} = \varrho_{far} \left(\frac{|\overrightarrow{u}_{far}|}{M_{far}} \right)^2 \tag{3.74}$$

$$E_{out} = \frac{p_{out}}{\gamma - 1} + \frac{1}{2} \varrho_{far} \, |\overrightarrow{u}_{far}|^2 \tag{3.75}$$

Δηλαδή τελικά:

$$\overrightarrow{U_{out}} = \begin{bmatrix} \varrho_{far} \\ \varrho_{far} \, |\overrightarrow{u}_{far}| \cos\theta_{far} \\ \varrho_{far} \, |\overrightarrow{u}_{far}| \sin\theta_{far} \\ \frac{\varrho_{far} \left(\frac{|\overrightarrow{u}_{far}|}{M_{far}} \right)^2}{\gamma - 1} + \frac{1}{2} \varrho_{far} \, |\overrightarrow{u}_{far}|^2 \end{bmatrix} \tag{3.76}$$

Το $\overrightarrow{\Phi}_{far}$ υπολογίζεται τώρα κατά τα γνωστά για τα εσωτερικά βαρύκεντρα.

3.4 Δεύτερης τάξης ακρίβεια - Χρήση παραγώγων

Είναι δυνατό να αυξηθεί η ακρίβεια υπολογισμού των ροϊκών μεγεθών στα μέσα M_i, $i = 1, 2, 3$ των πλευρών του όγκου ελέγχου χρησιμοποιώντας τις τιμές των ροϊκών μεγεθών στα βαρύκεντρα (ήδη γνωστές) και υπολογίζοντας τις χωρικές παραγώγους τους για το εκάστοτε τρίγωνο. Η μέθοδος που θα χρησιμοποιηθεί είναι μια προσαρμογή της μεθόδου ελαχίστων τετραγώνων (least-squares method).

3.4.1 Εσωτερικά Τρίγωνα

Επιβάλλεται να ισχύει η παρακάτω ισότητα ώστε να είναι κατά το δυνατόν ακριβείς οι χωρικές παράγωγοι των ροϊκών μεγεθών:

$$\overrightarrow{U}_{Q_i} = \overrightarrow{U}_P + \overrightarrow{\nabla U_P} \cdot \overrightarrow{PQ_i}, \qquad i = 1, 2, 3 \tag{3.77}$$

όπου $\overrightarrow{PQ_i}$ το διάνυσμα που ενώνει το βαρύκεντρο του υπόψη τριγώνου με αυτό του γειτονικού τριγώνου.
Αναπτύσσοντας:

$$\overrightarrow{U}_{Q_i} - \overrightarrow{U}_P = \frac{\partial \overrightarrow{U_P}}{\partial x} \overrightarrow{PQ_i}\Big|_x + \frac{\partial \overrightarrow{U_P}}{\partial y} \overrightarrow{PQ_i}\Big|_y \qquad i = 1, 2, 3 \tag{3.78}$$

Σχηματίζεται η ποσότητα L:

$$L = \frac{1}{2}\sum_{i=1}^{3} \frac{\overrightarrow{U_{Q_i}} - \overrightarrow{U_P} - \frac{\partial \overrightarrow{U_P}}{\partial x}\overrightarrow{PQ_i}\Big|_x - \frac{\partial \overrightarrow{U_P}}{\partial y}\overrightarrow{PQ_i}\Big|_y}{\left|\overrightarrow{PQ_i}\right|^2} \tag{3.79}$$

η οποία είναι επιθυμητό να παίρνει την ελάχιστη τιμή της ώστε οι παράγωγοι που χρησιμοποιούνται να είναι ακριβείς.
Παραγωγίζοντας την ποσότητα L ως προς τις παραγώγους των ροϊκών μεγεθών και θέτοντας τη ίση με το μηδέν για να προκύψει το ελάχιστο:

$$\frac{\partial L}{\partial \left.\frac{\partial \overrightarrow{U}}{\partial x}\right|_P} = \sum_{i=1}^{3}\left[\frac{\overrightarrow{U_{Q_i}} - \overrightarrow{U_P} - \frac{\partial \overrightarrow{U_P}}{\partial x}\overrightarrow{PQ_i}\Big|_x - \frac{\partial \overrightarrow{U_P}}{\partial y}\overrightarrow{PQ_i}\Big|_y}{\left|\overrightarrow{PQ_i}\right|^2}\left(-\overrightarrow{PQ_i}\Big|_x\right)\right] = 0 \tag{3.80}$$

$$\frac{\partial L}{\partial \left.\frac{\partial \overrightarrow{U}}{\partial y}\right|_P} = \sum_{i=1}^{3}\left[\frac{\overrightarrow{U_{Q_i}} - \overrightarrow{U_P} - \frac{\partial \overrightarrow{U_P}}{\partial x}\overrightarrow{PQ_i}\Big|_x - \frac{\partial \overrightarrow{U_P}}{\partial y}\overrightarrow{PQ_i}\Big|_y}{\left|\overrightarrow{PQ_i}\right|^2}\left(-\overrightarrow{PQ_i}\Big|_y\right)\right] = 0 \tag{3.81}$$

Κάνοντας τις πράξεις καταλήγουμε:

$$\left[\sum_{i=1}^{3}\frac{\left(\overrightarrow{PQ_i}\Big|_x\right)^2}{\left|\overrightarrow{PQ_i}\right|^2}\right]\left.\frac{\partial \overrightarrow{U}}{\partial x}\right|_P + \left[0\sum_{i=1}^{3}\frac{\left(\overrightarrow{PQ_i}\Big|_x\right)\left(\overrightarrow{PQ_i}\Big|_y\right)}{\left|\overrightarrow{PQ_i}\right|^2}\right]\left.\frac{\partial \overrightarrow{U}}{\partial y}\right|_P = \tag{3.82}$$

$$\sum_{i=1}^{3}\left[\frac{\overrightarrow{PQ_i}\Big|_x}{\left|\overrightarrow{PQ_i}\right|^2}\left(\overrightarrow{U_{Q_i}} - \overrightarrow{U_P}\right)\right] \tag{3.83}$$

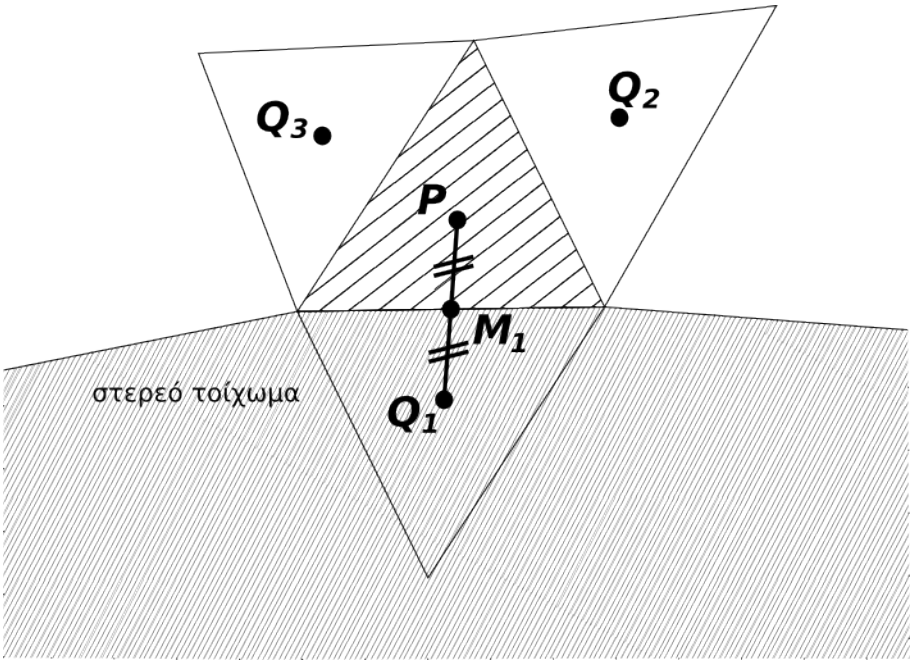
$$\left[\sum_{i=1}^{3}\frac{\left(\overrightarrow{PQ_i}\Big|_x\right)\left(\overrightarrow{PQ_i}\Big|_y\right)}{\left|\overrightarrow{PQ_i}\right|^2}\right]\left.\frac{\partial \overrightarrow{U}}{\partial x}\right|_P + \left[\sum_{i=1}^{3}\frac{\left(\overrightarrow{PQ_i}\Big|_y\right)^2}{\left|\overrightarrow{PQ_i}\right|^2}\right]\left.\frac{\partial \overrightarrow{U}}{\partial y}\right|_P = \tag{3.84}$$

$$\sum_{i=1}^{3}\left[\frac{\overrightarrow{PQ_i}\Big|_y}{\left|\overrightarrow{PQ_i}\right|^2}\left(\overrightarrow{U_{Q_i}} - \overrightarrow{U_P}\right)\right] \tag{3.85}$$

Οι εξισώσεις 3.83, 3.85 αποτελούν σύστημα 2x2 από το οποίο υπολογίζονται τελικά τις χωρικές παραγώγους του διανύσματος ροής για το τρίγωνο P $\left(\left.\frac{\partial \overrightarrow{U}}{\partial x}\right|_P$ και $\left.\frac{\partial \overrightarrow{U}}{\partial y}\right|_P\right)$.

3.4.2 Τρίγωνα με πλευρά κοινή με τοίχωμα

Στην περίπτωση που μία (ή και περισσότερες) πλευρά του τριγώνου εφάπτεται σε τοίχωμα (έστω η πλευρά 1), τότε θεωρείται ότι η πλευρά αυτή είναι κοινή του P με



Σχήμα 3.5: Φανταστικό τρίγωνο στην πλευρά του όγκου ελέγχου που εφάπτεται στο στερεό τοίχωμα.

το φανταστικό τρίγωνο Q_1 (Σχ. 3.5). Αξίζει να αναφερθεί πως για τα τρίγωνα στο τοίχωμα μας ενδιαφέρει μόνο η παράγωγος της πίεσης για λόγους που αναφέρθηκαν στην παράγραφο 3.3.

Για το τρίγωνο Q_1 θεωρούμε ότι $p_{Q_1} = p_P$, όπου p η πίεση. Επίσης, επιλέγεται το φανταστικό Q_1 ώστε να ισχύει ότι $\overrightarrow{PQ_1} = 2\overrightarrow{PM_1}$ κατά το σχ. 3.5. Αντικαθιστώντας στις εξισώσεις 3.83, 3.85 τα γεωμετρικά μεγέθη του φανταστικού τριγώνου και βάζοντας τις τιμές της πίεσης αντί για το διάνυσμα των ροϊκών μεγεθών προκύπτει το σύστημα των εξισώσεων 3.87, 3.89 από το οποίο υπολογίζεται η κλίση της πίεσης στα τρίγωνα

που εφάπτονται με το στερεό τοίχωμα.

$$\left[\sum_{i=1}^{3}\frac{\left(\left.\overrightarrow{PQ_i}\right|_x\right)^2}{\left|\overrightarrow{PQ_i}\right|^2}\right]\left.\frac{\partial p}{\partial x}\right|_P+\left[0\sum_{i=1}^{3}\frac{\left(\left.\overrightarrow{PQ_i}\right|_x\right)\left(\left.\overrightarrow{PQ_i}\right|_y\right)}{\left|\overrightarrow{PQ_i}\right|^2}\right]\left.\frac{\partial p}{\partial y}\right|_P= \tag{3.86}$$

$$\sum_{i=1}^{3}\left[\frac{\left.\overrightarrow{PQ_i}\right|_x}{\left|\overrightarrow{PQ_i}\right|^2}\left(p_{Q_i}-p_P\right)\right] \tag{3.87}$$

$$\left[\sum_{i=1}^{3}\frac{\left(\left.\overrightarrow{PQ_i}\right|_x\right)\left(\left.\overrightarrow{PQ_i}\right|_y\right)}{\left|\overrightarrow{PQ_i}\right|^2}\right]\left.\frac{\partial p}{\partial x}\right|_P+\left[\sum_{i=1}^{3}\frac{\left(\left.\overrightarrow{PQ_i}\right|_y\right)^2}{\left|\overrightarrow{PQ_i}\right|^2}\right]\left.\frac{\partial p}{\partial y}\right|_P= \tag{3.88}$$

$$\sum_{i=1}^{3}\left[\frac{\left.\overrightarrow{PQ_i}\right|_y}{\left|\overrightarrow{PQ_i}\right|^2}\left(p_{Q_i}-p_P\right)\right] \tag{3.89}$$

3.4.3 Τρίγωνα με πλευρά κοινή με το "άπειρο"

Στην περίπτωση που το τρίγωνο του πλέγματος εφάπτεται στο εξωτερικό περιβάλλον, γίνεται χρήση της ίδιας μεθόδου με την περίπτωση που εφάπτεται με στερεό τοίχωμα, δημιουργώντας ένα φανταστικό τρίγωνο στην πλευρά του τριγώνου που είναι κοινή με το "άπειρο". Η διαφορά έγκειται στο ότι υπολογίζεται η κλίση ολόκληρου του διανύσματος ροϊκών μεγεθών και όχι μόνο της πίεσης αυτήν τη φορά.

3.4.4 Προσέγγιση του ροϊκού διανύσματος στο μέσο M των πλευρών

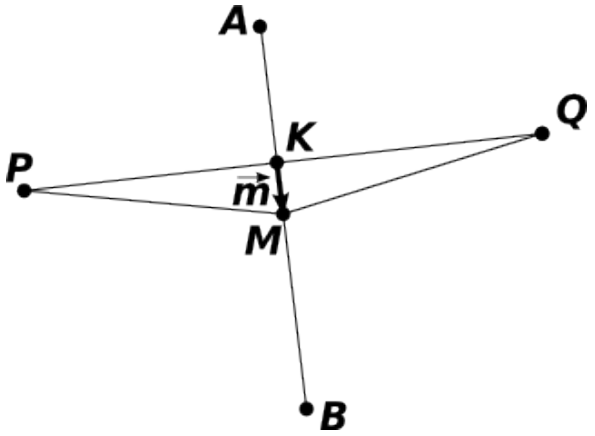
Αρχικά, υπολογίζονται τα ροϊκά μεγέθη και η κλίση τους στο K που είναι το σημείο τομής των ευθύγραμμων τμημάτων (PQ) και (AB), όπου (AB) η κοινή πλευρά των όγκων ελέγχου P (βαρύκεντρο κεντρικού τριγώνου) και Q (βαρύκεντρο ενός εκ των τριών γειτονικών τριγώνων) (Σχ. 3.6).

$$\overrightarrow{U_K}=f\overrightarrow{U_P}+(1-f)\,\overrightarrow{U_Q} \tag{3.90}$$

$$\nabla\overrightarrow{U_K}=f\nabla\overrightarrow{U_P}+(1-f)\,\nabla\overrightarrow{U_Q} \tag{3.91}$$

Το f είναι συντελεστής που υπολογίζεται από τη σχέση:

$$f=\frac{\left|\overrightarrow{PK}\right|}{\left|\overrightarrow{PQ}\right|} \tag{3.92}$$



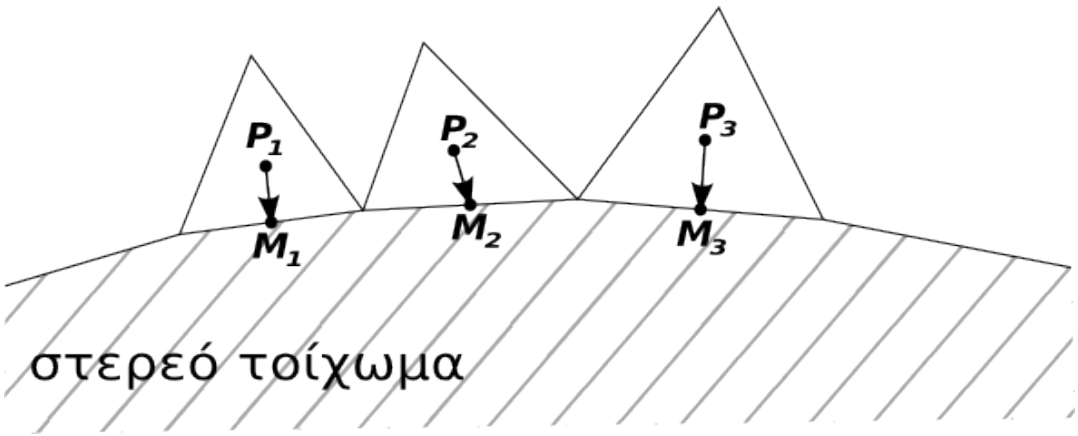
Σχήμα 3.6: Υπολογισμός του ροϊκού διανύσματος στο μέσο M του (AB) με χρήση των ροϊκών μεγεθών στα βαρύκεντρα P και Q.

Ο υπολογισμός των ροϊκών μεγεθών στο M γίνεται με προεκβολή από το K.

$$\overrightarrow{U_M} = \overrightarrow{U_K} + \overrightarrow{m} \cdot \nabla \overrightarrow{U_K} \tag{3.93}$$

όπου, όπως φαίνεται στο σχ. 3.6, το $\overrightarrow{m}$ είναι το διάνυσμα από το K στο M.

3.5 Υπολογισμός συντελεστών άνωσης - οπισθέλκουσας



Σχήμα 3.7: Προεκβολή της πίεσης από τα βαρύκεντρα των τριγώνων που εφάπτονται στο στερεό τοίχωμα στο μέσο της αντίστοιχης πλευράς.

Στα τριγωνικά στοιχεία P_i, σχ. 3.97, γίνεται προεκβολή της πίεσης από το βαρύκεντρο στο μέσο του τμήματος που εφάπτεται στο τοίχωμα με χρήση της κλίσης της πίεσης στον αντίστοιχο όγκο ελέγχου.

$$p_M = p_P + \overrightarrow{PM} \cdot \nabla p \tag{3.94}$$

Αθροίζεται το σύνολο των πιέσεων που προέκυψαν από την προεκβολή για κάθε όγκο ελέγχου σε συνάρτηση με την κατεύθυνση κάθε ευθύγραμμου τμήματος που εφάπτεται στο τοίχωμα. Έτσι, ισχύει:

$$F_x = \sum_{i=1}^{N} p_M n_x \tag{3.95}$$

$$F_y = \sum_{i=1}^{N} p_M n_y \tag{3.96}$$

όπου N το πλήθος των τριγωνικών στοιχείων που εφάπτονται σε στερεό τοίχωμα. Από τις σχέσεις 3.97, 3.98 προκύπτουν οι συντελεστές άνωσης και οπισθέλκουσας αντίστοιχα.

$$C_{Lift} = \frac{F_x cos\left(a_\infty\right) + F_y sin\left(a_\infty\right)}{\frac{1}{2}\varrho_\infty u_\infty^2} \tag{3.97}$$

$$C_{Drag} = \frac{-F_x sin\left(a_\infty\right) + F_y cos\left(a_\infty\right)}{\frac{1}{2}\varrho_\infty u_\infty^2} \tag{3.98}$$

όπου a_∞: η γωνία προσβολής της ροής στην αεροτομή και ο δείκτης "∞" συμβολίζει τις τιμές των μεγεθών στο άπειρο.

Κεφάλαιο 4

Επίλυση των Εξισώσεων Ροής και Προγραμματισμός της

4.1 Αριθμητική επίλυση των εξισώσεων ροής

Η επίλυση της εξίσωσης 3.50 γίνεται με το επαναληπτικό σχήμα **Jacobi** [44] και τη βοήθεια της εισαγωγής του **ψευδοχρόνου**. Προστίθεται στην 3.50 ο ψευδοχρονικός όρος και έτσι προκύπτει η:

$$\left.\frac{\Delta\overrightarrow{U}}{\Delta\tau}\Omega_P\right|^{n+1} + \left.\sum_{i=1}^{3}\Phi_{PQ_i}\right|^{n+1} = 0 \tag{4.1}$$

όπου με $\Delta\tau$ συμβολίζεται το ψευδοχρονικό βήμα και με $n+1$ η ψευδοχρονική στιγμή στην οποία βρισκόμαστε.

Το $\Delta\tau$ υπολογίζεται από τη σχέση:

$$\Delta\tau = \frac{CFL}{\frac{1}{h_{min}}\left(|\overrightarrow{u}| + c\right)} \tag{4.2}$$

όπου CFL (Courant-Friedrichs-Lewy) [24]: συντελεστής επιλεγμένος από το χρήστη,
h_{min}: το μήκος του μικρότερου ύψους του τριγωνικού όγκου ελέγχου,
$|\overrightarrow{u}|$: το μέτρο της ταχύτητας και
c: ο αριθμός Mach.

Χρησιμοποιώντας την ανάπτυξη κατά Taylor με ακρίβεια πρώτης τάξης:

$$\left.\sum_{i=1}^{3}\Phi_{PQ_i}\right|^{n+1} = \left.\sum_{i=1}^{3}\Phi_{PQ_i}\right|^{n} + \left.\sum_{i=1}^{3}\frac{\partial\Phi_{PQ_i}}{\partial\overrightarrow{U}^{L}_{PQ_i}}\Delta\overrightarrow{U}^{L}_{PQ_i}\right|^{n+1} + \left.\sum_{i=1}^{3}\frac{\partial\Phi_{PQ_i}}{\partial\overrightarrow{U}^{R}_{PQ_i}}\Delta\overrightarrow{U}^{R}_{PQ_i}\right|^{n} \tag{4.3}$$

Κάνοντας χρήση των 3.52 και 3.53 (απλή ακρίβεια) μπορεί να γραφεί:

$$\left.\sum_{i=1}^{3}\Phi_{PQ_i}\right|^{n+1} = \left.\sum_{i=1}^{3}\Phi_{PQ_i}\right|^{n} + \left.\sum_{i=1}^{3}\frac{\partial\Phi_{PQ_i}}{\partial\overrightarrow{U}_{P}}\Delta\overrightarrow{U}_{P}\right|^{n+1} + \left.\sum_{i=1}^{3}\frac{\partial\Phi_{PQ_i}}{\partial\overrightarrow{U}_{Q_i}}\Delta\overrightarrow{U}_{Q_i}\right|^{n} \tag{4.4}$$

Συνδυάζοντας τις 4.1 και 4.4 προκύπτει:

$$\sum_{i=1}^{3} \frac{\partial \Phi_{PQ_i}}{\partial \overrightarrow{U}_P} \Delta \overrightarrow{U}_P \Big|^{n+1} + \sum_{i=1}^{3} \frac{\partial \Phi_{PQ_i}}{\partial \overrightarrow{U}_{Q_i}} \Delta \overrightarrow{U}_{Q_i} \Big|^{n} + \frac{\Omega_P}{\Delta \tau} \Delta \overrightarrow{U_P} \Big|^{n+1} = - \sum_{i=1}^{3} \Phi_{PQ_i} \Big|^{n} \Leftrightarrow \quad (4.5)$$

$$\Leftrightarrow \Delta \overrightarrow{U}_P \Big|^{n+1} = \left(- \sum_{i=1}^{3} \Phi_{PQ_i} \Big|^{n} - \sum_{i=1}^{3} \frac{\partial \Phi_{PQ_i}}{\partial \overrightarrow{U}_{Q_i}} \Delta \overrightarrow{U}_{Q_i} \Big|^{n} \right) \left(\sum_{i=1}^{3} \frac{\partial \Phi_{PQ_i}}{\partial \overrightarrow{U}_P} + \frac{\Omega_P}{\Delta \tau} \right)^{-1} \quad (4.6)$$

Υπολογίζοντας με αυτό τον τρόπο το $\Delta \overrightarrow{U_P}$ ανανεώνουμε το $\overrightarrow{U_P}$ για το επόμενο ψευδοχρονικό βήμα και ξεκινά η επόμενη επανάληψη. Ισχύει, δηλαδη:

$$\overrightarrow{U_P} \Big|^{n+1} = \overrightarrow{U_P} \Big|^{n} + \Delta \overrightarrow{U_P} \Big|^{n+1} \quad (4.7)$$

Οι επαναλήψεις σταματούν όταν

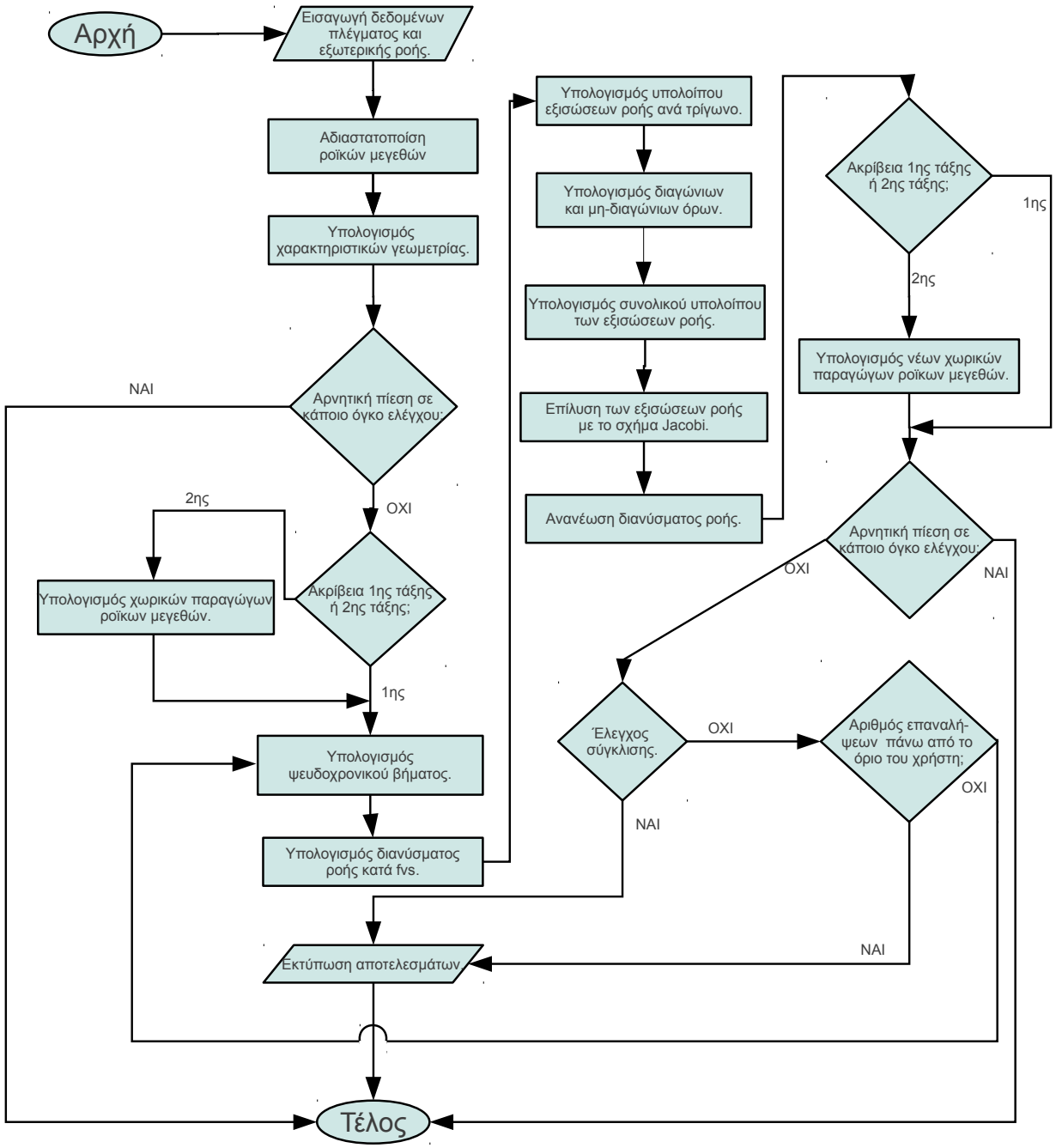
$$\left| \Delta \overrightarrow{U_P} \Big|^{n+1} \right| < \overrightarrow{z} \quad (4.8)$$

όπου $\overrightarrow{z} = \begin{bmatrix} z_1 & z_2 & z_3 & z_4 \end{bmatrix}^T$ με z_1, z_2, z_3, z_4 μικρές θετικές ποσότητες όσο κοντά στο 0 επιλέξει ο χρήστης ανάλογα με την επιθυμητή σύγκλιση.

4.2 Γενική εποπτεία κώδικα

Στο σχ. 4.1 παρουσιάζεται το διάγραμμα ροής του κώδικα που αναπτύχθηκε στην παρούσα διπλωματική εργασία με σκοπό την επίλυση της ροής όπως διατυπώθηκε στο κεφάλαιο 3. Παρακάτω, θα παρατεθεί μια σύντομη επεξήγηση του διαγράμματος.

Αρχικά, ο κώδικας αποθηκεύει σε πίνακες τα δεδομένα του πλέγματος που του παρέχονται απο τα .nod και .ele αρχεία. Συνοπτικά, αυτά αφορούν το πλήθος των κόμβων, τις συντεταγμένες τους, τη συνδεσιμότητα μεταξύ τους και ένα χαρακτηρισμό για το αν είναι εσωτερικοί ή οριακοί και τι είδους. Επίσης, από το .ini αρχείο λαμβάνονται πληροφορίες σχετικά με τη ροή (γωνία προσβολής, ταχύτητα) αλλά και άλλες όπως την τιμή του CFL (4.2) ή το μέγιστο αριθμό επαναλήψεων που θα πραγματοποιηθούν αν δεν πραγματοποιηθεί νωρίτερα σύγκλιση. Στη συνέχεια, τα μεγέθη της ροής αδιαστατοποιούνται κατά την παράγραφο 3.1.2. Αφού είναι ήδη αποθηκευμένα τα στοιχεία του πλέγματος, με τη βοήθειά τους υπολογίζονται άλλα χρήσιμα γεωμετρικά μεγέθη που θα χρειαστούν στη συνέχεια και τα αποθηκεύουμε και αυτά. Ενδεικτικά, αναφέρουμε τα διανύσματα $\overrightarrow{n_i}$ (σχ. 3.2) και $\overrightarrow{PQ_i}$ (3.83, 3.85). Σε περίπτωση που επιλεγεί η χρήση διπλής τάξης ακρίβειας (ενδείκνυται) υπολογίζονται οι χωρικές παράγωγοι του διανύσματος ροής σύμφωνα με την παράγραφο 3.4. Διαφορετικά το βήμα αυτό παραλείπεται. Εδώ αρχίζει ο κύκλος



Σχήμα 4.1: Λογικό διάγραμμα του αλγόριθμου επίλυσης της ροής.

των επαναλήψεων. Υπολογίζεται το ψευδοχρονικό βήμα $\Delta\tau$ (4.2). Το επόμενο βήμα είναι ο υπολογισμός του διανύσματος ροής Φ_{PQ_i} (3.51). Τα υπόλοιπα των εξισώσεων

ροής ανά τρίγωνο-όγκο ελέγχου αθροίζονται για να προκύψει το δεξί μέλος της 4.6, ενώ υπολογίζονται και οι διαγώνιοι και μη-διαγώνιοι όροι για τον πρώτο και δεύτερο όρο της 4.6 αντίστοιχα. Στη συνέχεια, πραγματοποιείται η ανανέωση του διανύσματος ροής κατά την 4.7 και ο υπολογισμός της χωρικής παραγώγου του αν πρόκειται για δεύτερης τάξης ακρίβεια. Στο παρόν σημείο, ελέγχεται αν έχει πραγματοποιηθεί η σύγκλιση και αν ο αριθμός της τρέχουσας επανάληψης είναι μεγαλύτερος του μέγιστου που έχει θέσει ο χρήστης. Αν η απάντηση και στα δύο ερωτήματα είναι αρνητική ξεκινά μια νέα επανάληψη. Διαφορετικά, τυπώνονται σε αρχεία τα επιθυμητά αποτελέσματα και ολοκληρώνεται ο κώδικας.

4.3 Προγραμματιστικά ζητήματα

Η επίλυση της ροής προγραμματίστηκε, όπως προαναφέρθηκε, σε CUDA C, μια γλώσσα που δημιουργήθηκε και εξακολουθεί να βρίσκεται υπό ανάπτυξη από την Nvidia. Η μεταγλώττιση (compiling) δημιουργεί ένα εκτελέσιμο αρχείο που εκτελείται σε έναν κεντρικό επεξεργαστή και σε κάρτες της ίδιας εταιρίας.

Σκοπός, φυσικά, της επιλογής αυτής είναι η επιτάχυνση της εκτέλεσης. Το χρονικό κέρδος που προκύπτει οφείλεται στην παραλληλοποίηση κάποιων εργασιών στα εκατοντάδες threads της GPU. Από την άλλη, είναι χρονοβόρα και μη-συμφέρουσα σε μνήμη η επικοινωνία μεταξύ επεξεργαστή και κάρτας γραφικών. Το ερώτημα που τίθεται, λοιπόν, είναι ποια τμήματα του παραπάνω αλγόριθμου συμφέρει να τρέξουν παράλληλα στην κάρτα γραφικών και πως αυτά θα διαχωριστούν.

Η επιλογή που έγινε είναι: οι πράξεις που αφορούν όλους τους όγκους ελέγχου-τρίγωνα, οι οποίες είναι ίδιες για όλους (ή για υποσύνολα των όγκων ελέγχου) και θα απαιτούσαν βρόχους (loops) στο σύνολο των τριγώνων, να εκτελεστούν από τη GPU παράλληλα. Κατά την εκτέλεση του κώδικα, κάθε τρίγωνο αποδίδεται σε ένα thread το οποίο αναλαμβάνει να εκτελέσει τις πράξεις που απαιτούνται. Κρίνεται σκόπιμο να αναφερθεί ξανά ότι στον προϋπάρχοντα κώδικα στον οποίο στηρίχθηκε ο κώδικας που αναπτύχθηκε κατά την παρούσα διπλωματική εργασία, κάθε thread αναλάμβανε έναν κόμβο και όχι ένα τρίγωνο. Η επιλογή αντιστοίχησης thread-τριγώνου προκαλεί μεν μικρές καθυστερήσεις κάθε φορά που απαιτείται η επικοινωνία CPU-GPU, γεγονός που επικαλύπτεται σε πολύ μεγάλο βαθμό από το γεγονός ότι επαναλήψεις που θα εκτελούνταν σειριακά σε αριθμό ίσο με το πλήθος των όγκων ελέγχου (απο μερικές χιλιάδες ως αρκετές δεκάδες χιλιάδες) γίνονται τώρα ταυτόχρονα. Στην επόμενη υποπαράγραφο παρουσιάζεται με ποιό τρόπο επιλέγεται ο όγκος ελέγχου που θα αποδοθεί σε ένα thread.

Επίσης, στη συνέχεια, θα αναφερθούν ενδεικτικά ζητήματα που λόγω της ιδιομορφίας του παράλληλου προγραμματισμού διαχειρίζονται έτσι ώστε να εξασφαλιστεί η επιτάχυνση της σύγκλισης.

4.3.1 Καταμερισμός όγκων ελέγχου στα threads

Η ταυτότητα κάθε όγκου ελέγχου-τριγώνου είναι ένας αριθμός ο οποίος είναι μοναδικός για το συγκεκριμένο πλέγμα και κυμαίνεται μεταξύ του ένα (1) και του πλήθους των όγκων ελέγχου. Στο εξής, η ταυτότητα αυτή θα συμβολίζεται με τη συντομογραφία **ip**. Είναι προγραμματιστικά εφικτό με μια απλή εντολή για κάθε thread να γνωρίζουμε τα εξής:

- **block_dim**: τη διάσταση του κάθε block, δηλαδή από πόσα threads αποτελείται (βλ. παρ. 2.1.1). Η τιμή αυτή επιλέγεται από το χρήστη στην αρχή του κώδικα.
- **block_id**: την αρίθμηση του block (που ανήκει το συγκεκριμένο thread) αναφορικά με τα υπόλοιπα
- **thread_id**: την αρίθμηση του thread αναφορικά με τα υπόλοιπα που ανήκουν στο ίδιο block

Η ip του τριγώνου που θα το αναλάβει ένα συγκεκριμένο thread προκύπτει από την παρακάτω σχέση:

$$ip = block\_dim \cdot block\_id + thread\_id \tag{4.9}$$

Κάποιες φορές, κατά τη διάρκεια εκτέλεσης του κώδικα χρειάζεται να χρησιμοποιηθεί αντίστροφη λογική. Για παράδειγμα, είναι γνωστή εκ των προτέρων η ip των τριγώνων που έχουν έστω μια κοινή πλευρά με το τοίχωμα και είναι επιθυμητό τα threads που αντιστοιχούν στα τρίγωνα αυτά να εκτελέσουν ένα διαφορετικό σύνολο πράξεων από τα υπόλοιπα εσωτερικά τρίγωνα. Σε αυτήν την περίπτωση χρειάζεται να εντοπίσουμε σε ποιά threads αντιστοιχούν αυτά τα τρίγωνα (υπολογισμός block_id και thread_id).

$$block\_id = \frac{ip}{block\_dim} \tag{4.10}$$

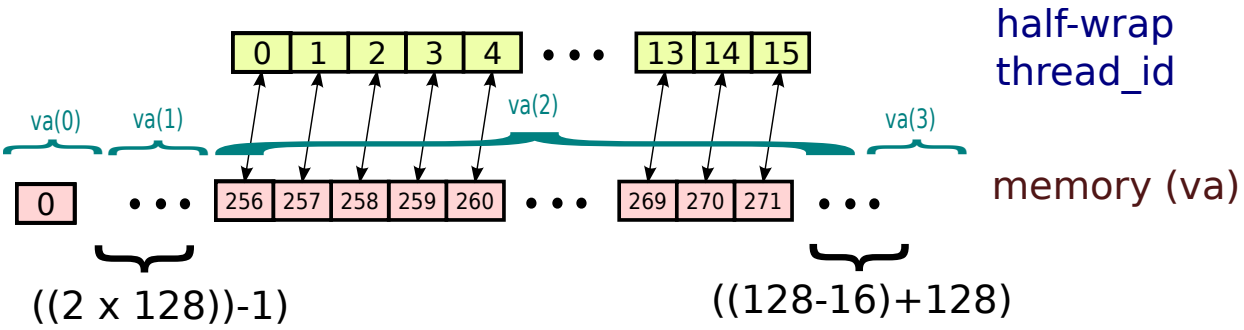
$$thread\_id = ip - block\_dim \cdot block\_id \tag{4.11}$$

Επισημαίνεται ότι στη διαίρεση της 4.10 λαμβάνεται μόνο το ακέραιο πηλίκο της ενώ το υπόλοιπο αγνοείται.

4.3.2 Αποθήκευση πινάκων στη GPU

Όπως έχει αναφερθεί (παρ. 2.1.1), τα threads δεν λειτουργούν τελείως ανεξάρτητα το ένα με το άλλο αλλά ομαδοποιούνται σε warps δηλαδή υπο-ομάδες των 32, εκτελώντας ταυτόχρονα όλα την ίδια εντολή (kernel). Συχνά, αυτή η εντολή απαιτεί την ανάγνωση ενός συγκεκριμένου στοιχείου ενός πίνακα που είναι αποθηκευμένος στη global μνήμη. Επειδή, ακριβώς, αυτή η εντολή εκτελείται από όλο το warp, για να επιτύχουμε όσο το δυνατόν γρηγορότερη προσπέλαση στη μνήμη πρέπει οι τιμές των μεταβλητών που πρέπει (για τη συγκεκριμένη εντολή) να διαβαστούν από

κάθε thread του warp να βρίσκονται κατά το δυνατόν σε κοντινές θέσεις μνήμης. Επομένως, η αποθήκευση ενός πίνακα πραγματοποιείται με τέτοιο τρόπο ώστε να επιτευχθεί κατά το δυνατόν το παραπάνω.



Σχήμα 4.2: Προσπέλαση της μνήμης για την ανάγνωση του διανύσματος ροής σύμφωνα με την αποθήκευση του κώδικα.

Παρατίθεται ένα παράδειγμα αποθήκευσης πίνακα για την πληρέστερη επεξήγηση του παραπάνω. Από τις μεταβλητές που χρειάζεται να διαβαστούν συχνότερα κατά τη διάρκεια εκτέλεσης του κώδικα είναι αυτές του συντηρητικού διανύσματος ροής:

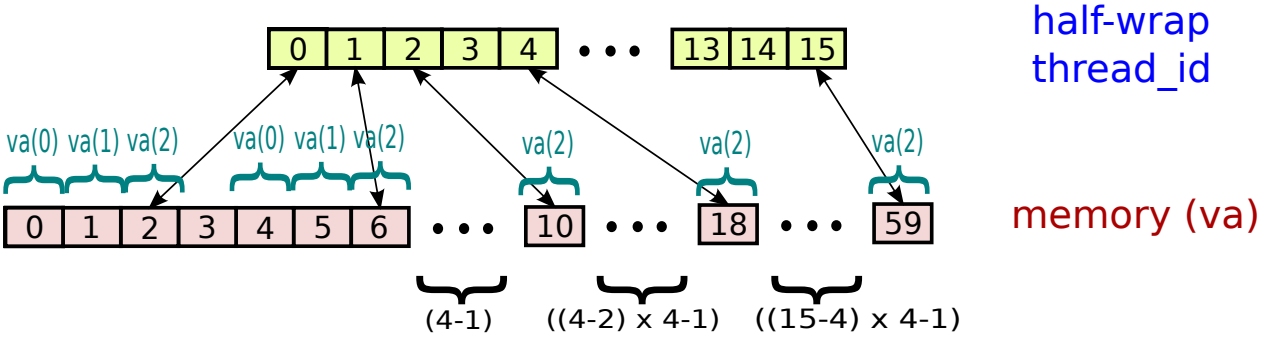
$$\overrightarrow{U} = \begin{bmatrix} \varrho \\ \varrho u \\ \varrho v \\ E \end{bmatrix} \tag{4.12}$$

Το διάνυσμα αυτό επιλέχθηκε να αποθηκευτεί σε έναν πίνακα με το όνομα va με τον εξής τρόπο:

$$\overrightarrow{U} = \begin{bmatrix} \varrho \\ \varrho u \\ \varrho v \\ E \end{bmatrix} = \begin{bmatrix} va\left[thread\_id + block\_dim \cdot 0\right] \\ va\left[thread\_id + block\_dim \cdot 1\right] \\ va\left[thread\_id + block\_dim \cdot 2\right] \\ va\left[thread\_id + block\_dim \cdot 3\right] \end{bmatrix} \tag{4.13}$$

Για να γίνει πιο συγκεκριμένο το παράδειγμα, επιλέγεται $block\_dim = 128$ (η επιλογή αυτή δεν προκαλεί βλάβη στη γενικότητα). Ο παραπάνω τρόπος αποθήκευσης πρακτικά σημαίνει πως θα αποθηκευτεί στη σειρά μία 128άδα από τα 128 πρώτα στοιχεία του πίνακα που αντιστοιχούν σε 128 όγκους ελέγχου, μία 128άδα για τα 128 δεύτερα στοιχεία κ.ο.κ. Η διαδικασία συνεχίζεται μέχρι να καλυφθούν όλοι οι όγκοι ελέγχου. Θεωρείται, τώρα, πως τα threads του ίδου half-warp ($thread\_id = 0, 1 \ldots 15$) θέλουν να διαβάσουν το 3ο στοιχείο του διανύσματος ροής ($\varrho v = va\,(2)$). Καταχρηστικά, στο παράδειγμα, θα θεωρηθεί πως δεν υπάρχουν άλλα threads. Όπως φαίνεται από το σχ. 4.2 οι θέσεις μνήμης που διαβάζονται είναι παραπλήσιες με αποτέλεσμα η ανάγνωση να γίνεται γρήγορα.

Σε περίπτωση που το διάνυσμα της ροής αποθηκευόταν σειριακά (συνηθισμένη τακτική για σειριακό - μη-παράλληλο προγραμματισμό), δηλαδή για κάθε τρίγωνο οι 4 μεταβλητές του διανύσματος ροής μαζί η μία μετά της άλλη, η ανάγνωση θα γινόταν κατά προσέγγιση όπως στο σχ. 4.3. Αποθηκεύονται, δηλαδή, μαζί όλα τα στοιχεία του πίνακα για κάθε όγκο ελέγχου ξεχωριστά. Επισημαίνοντας ότι το σύνολο των threads είναι χιλιάδες φορές περισσότερα απο 16, είναι κατανοητό πόσο άτακτη είναι η προσπέλαση της μνήμης και πόσο αυτό καθυστερεί τον κώδικα.



Σχήμα 4.3: Προσπέλαση της μνήμης για την ανάγνωση του διανύσματος ροής αν αυτό αποθηκευτεί σειριακά.

4.3.3 Ημι-παράλληλη άθροιση - Χρήση CUDA streams και shared μεταβλητών.

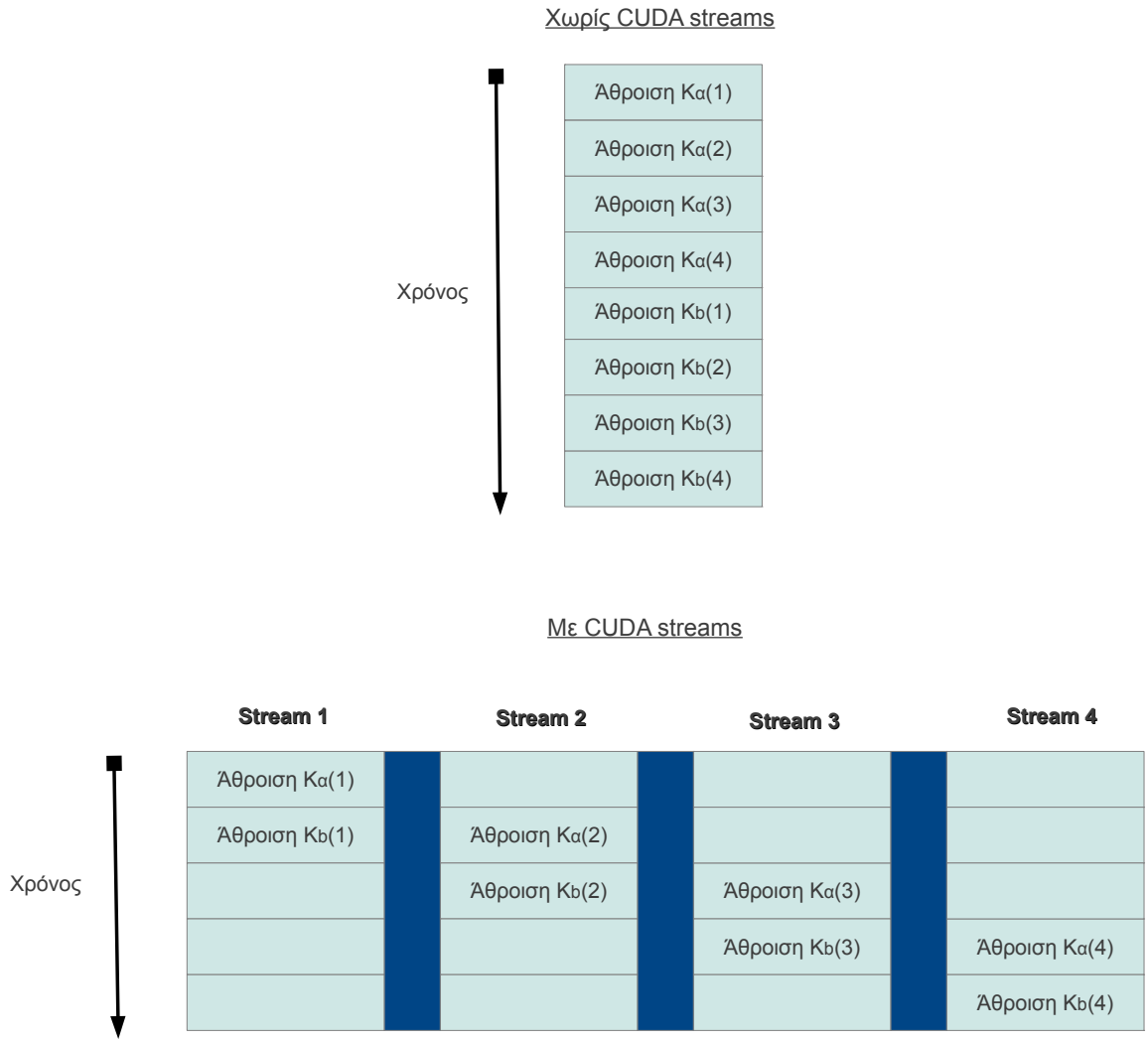
Κατά την εκτέλεση του κώδικα παρουσιάζεται σε κάθε επανάληψη η ανάγκη να αθροιστούν τα υπόλοιπα των εξισώσεων ροής όλων των όγκων ελέγχου. Η ίδια ανάγκη παρουσιάζεται μετά τη σύγκλιση, όταν θέλουμε να υπολογίσουμε τους συντελεστές άνωσης και οπισθέλκουσας που απαιτούν, αρχικά, την άθροιση των πιέσεων στην επιφάνεια της αεροτομής (βλ. παρ. 3.5). Αυτά τα προς άθροιση μητρώα έχουν υπολογιστεί στην GPU για τον κάθε όγκο ελέγχου από τα αντίστοιχα threads και βρίσκονται αποθηκευμένα στη global μνήμη. Η απευθείας άθροιση τους είναι αδύνατη καθώς δεν υπάρχει άμεση επικοινωνία μεταξύ των threads.

Μία προφανής λύση θα ήταν η αντιγραφή των μητρώων από τη global μνήμη στη RAM της CPU και η σειριακή άθροιση τους από τον επεξεργαστή. Προσπαθώντας, όμως, να γίνει καλύτερη εκμετάλλευση των δυνατοτήτων της GPU ακολουθείται η παρακάτω διαδικασία. Ορίζεται μία shared μεταβλητή. Υπενθυμίζεται ότι οι μεταβλητές shared είναι ορατές μόνο από threads που ανήκουν στο ίδιο block. Το πλεονέκτημα τους, όμως, είναι ότι η προσπέλαση της shared μνήμης είναι σημαντικά ταχύτερη από αυτήν της global. Στη θέση της shared αποθηκεύεται το ένα από τα στοιχεία του μητρώου (το ίδιο πάντα) προς άθροιση. Στη συνέχεια, γίνεται συγχρονισμός των threads και οι τιμές τους στο πρώτο μισό του block αθροίζονται μία προς μία με τις αντίστοιχες του δεύτερου μισού του block. Αν, όπως στο προηγούμενο παράδειγμα, $block\_dim = 128$ τότε αθροίζονται τα threads

(ξεκινώντας την αρίθμηση από το 0) 0 με 64, 1 με 65, κ.ο.κ, 63 με 128. Η διαδικασία συνεχίζεται με αυτό τον τρόπο μέχρι να προκύψει το τελικό άθροισμα του block καταλαμβάνοντας 7 βήματα $(128 \to 64 \to 32 \to 16 \to 8 \to 4 \to 2 \to 1)$.

Όπως αναφέρθηκε, αυτό που απαιτείται να αθροιστεί είναι μητρώα και όχι απλές μεταβλητές για τον κάθε όγκο ελέγχου. Στο σημείο αυτό επιλέγεται να γίνει χρήση τόσων CUDA streams όσο είναι το πλήθος των μεταβλητών του μητρώου για τον κάθε όγκου ελέγχου. Τα streams, αφού πραγματοποιήσουν την άθροιση ανά block για κάθε στοιχείο του προς άθροιση μητρώου με τη σειρά μεταφέρουν τα αποτελέσματα στη CPU η οποία αναλαμβάνει την τελική άθροιση των αποτελεσμάτων όλων των blocks. Το στοιχείο που συντελεί στην επιτάχυνση είναι ότι χάρη στη χρήση των streams η εκτέλεση του παραπάνω kernel (άθροιση ανά block) για κάθε στοιχείο του μητρώου γίνεται χωρίς να απαιτείται συγχρονισμός μεταξύ των block και άρα χωρίς να παραμένει ανενεργή η GPU. Μια σχηματική απεικόνιση του παραπάνω παρουσιάζεται στο σχ. 4.4. Θεωρείται ότι το προς άθροιση μητρώο είναι το $K\left(i\right), i = 1 \ldots 4$ το οποίο έχει σε κάθε όγκο ελέγχου τέσσερα στοιχεία, γι' αυτό και επιλέγεται η χρήση τεσσάρων streams. Οι δείκτες a, b αφορούν την ένδειξη για το ποιό block αφορά, καθώς το παράδειγμα περιλαμβάνει μόνο δύο blocks.

Αυτή η διαδικασία που αξιοποιεί κατά το δυνατόν τις δυνατότητες της GPU για αύξηση της ταχύτητας αλλά και η αναπόφευκτη χρήση της CPU για την εκτέλεση ορισμένων πράξεων οδήγησε στην επινόηση της προσφώνησης ημι-παράλληλη.



Σχήμα 4.4: Χρονικό κέρδος με τη χρήση CUDA streams.

Κεφάλαιο 5

Παρουσίαση Αποτελεσμάτων - Σύγκριση Επιδόσεων

Στο παρόν κεφάλαιο παρουσιάζονται τα αποτελέσματα που προκύπτουν από την εκτέλεση του κώδικα. Δοκιμάστηκαν διαφορετικές αεροτομές, διαφορετικά πλέγματα, τόσο πυκνά (με πολλούς όγκους ελέγχου), όσο και λιγότερο πυκνά (με λιγότερους όγκους ελέγχου) με διαφορετικές συνθήκες ροής (αριθμός Mach - γωνία προσβολής). Πέρα από την επίλυση της ροής, εξετάζεται ο αριθμός των επαναλήψεων που χρειάστηκε για να επέλθει η σύγκλιση αλλά και ο χρόνος εκτέλεσης. Θα γίνει σύγκριση των παραπάνω στοιχείων μεταξύ του κεντροκυψελικού κώδικα της παρούσας διπλωματικής εργασίας και του προϋπάρχοντος κεντροκομβικού της ΜΠΥΡ&Β. Επίσης, θα συγκριθούν οι χρόνοι μεταξύ των καρτών GTX285 και Tesla M2050 της Nvidia. Με εξαίρεση την τελευταία σύγκριση όλα τα άλλα τρεξίματα έγιναν στην κάρτα Tesla M2050.

5.1 Παρουσίαση αποτελεσμάτων επίλυσης της ροής

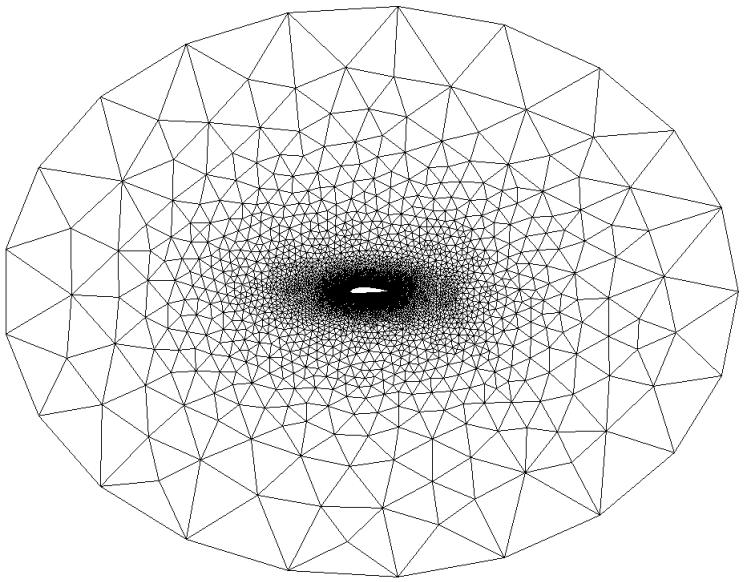
Πριν γίνει οποιαδήποτε σύγκριση θα παρουσιαστούν τρεξίματα σε δύο διαφορετικές αεροτομές, χρησιμοποιώντας διαφορετικά πλέγματα ώστε να καταδειχθεί ποια είναι τα αποτελέσματα του κώδικα και ποιά χρήσιμα μεγέθη μπορούν να εξαχθούν από την εκτέλεσή του. Οι τιμές των μεγεθών παρακάτω είναι όλες αδιάστατες.

5.1.1 Αεροτομή 1

Η αεροτομή κατά τα πρότυπα της National Advisory Committee for Aeronautics είναι η **NACA 4415** [45]. Δοκιμάστηκε η ίδια αεροτομή με δύο διαφορετικά πλέγματα σε δύο διαφορετικές συνθήκες ροής.

Περίπτωση i

Γι' αυτήν την περίπτωση μελετάται ροή με αριθμό $Mach$ εξωτερικής ροής $M_\infty = 0,6$ και γωνία προσβολής $a_\infty = 5^o$. Το CFL επιλέγεται έτσι ώστε να μεγιστοποιείται



Σχήμα 5.1: Ροή γύρω από την αεροτομή 1 (περίπτωση i). Το πλέγμα που χρησιμοποιήθηκε.

η ταχύτητα σύγκλισης. Έτσι, γι΄ αυτήν και για κάθε επόμενη περίπτωση έγιναν δοκιμές ώστε κάθε φορά να έχουμε την ταχύτερη δυνατή σύγκλιση. Το πλέγμα που χρησιμοποιήθηκε έχει 7733 κόμβους ή 15245 τρίγωνα. Στο σχ. 5.1 έχουμε την πλήρη εικόνα του αν και λόγω της μακρινής εστίασης υστερεί σε λεπτομέρεια.

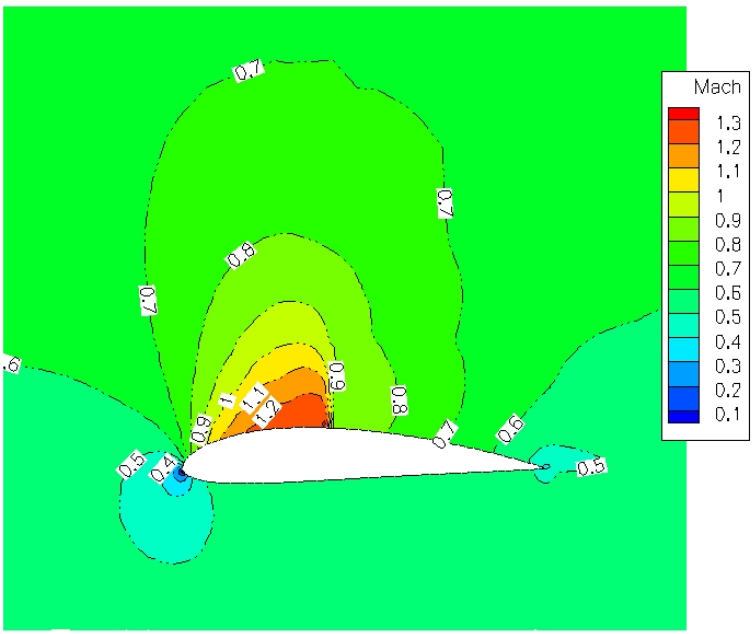
Αφού ολοκληρωθεί η εκτέλεση του κώδικα, ο χρήστης μπορεί να δει στην οθόνη πληροφορίες για το πλέγμα, το χρόνο εκτέλεσης του κώδικα και των επιμέρους κομματιών και των αριθμό των επαναλήψεων που χρειάστηκαν για να επιτευχθεί η επιθυμητή σύγκλιση. Για τα συγκεκριμένα δεδομένα χρειάστηκαν 999 επαναλήψεις για να συγκλίνει το πρόβλημα και $4,7$ δευτερόλεπτα. Η σύγκλιση πραγματοποείται όταν τα z_1, z_2, z_3, z_4 της εξίσωσης 4.8 γίνουν μικρότερα του 10^{-14}. Η συγκεκριμένη ακρίβεια σύγκλισης επιλέγεται για όλες τις περιπτώσεις. Ο χρόνος σύγκλισης περιλαμβάνει:

- $0,2$ δευτερόλεπτα για τον υπολογισμό των χωρικών παραγώγων των ροϊκών μεγεθών (παρ. 3.4),

- $1,0$ δευτερόλεπτο για τον υπολογισμό ροών (fluxes) κατά flux vector splitting (3.51) και

- $3,0$ για την επίλυση κατά Jacobi (παρ. 4.1).

Επιπλέον, στο τέλος εκτέλεσης του κώδικα τυπώνονται αρχεία που περιέχουν τις πληροφορίες που προέκυψαν από την εκτέλεση του. Αποθηκεύονται, έτσι, για κάθε όγκο ελέγχου τα ροϊκά μεγέθη από τα οποία μπορούν να εξαχθούν γραφικά, για παράδειγμα, η κατανομή του αριθμού Mach (σχ. 5.2). Στην παραπάνω απεικόνιση επιλέχθηκε να γίνει εστίαση κοντά στην αεροτομή καθώς η διακύμανση των μεγεθών είναι εντονότερη εκεί.

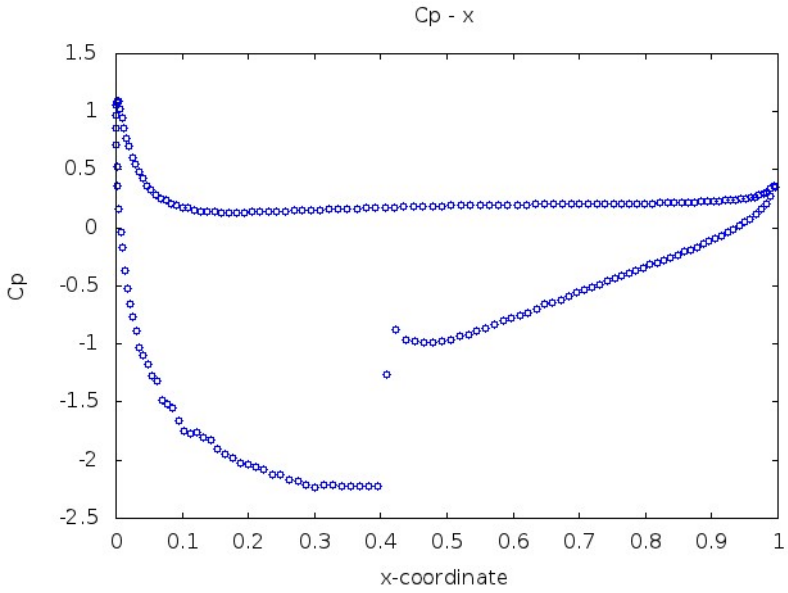


Σχήμα 5.2: Ροή γύρω από την αεροτομή 1 (περίπτωση i). Κατανομή του αριθμού Mach γύρω από την αεροτομή (κεντροκυψελική διατύπωση).

Ένα άλλο γράφημα που προκύπτει είναι αυτό του συντελεστή πιέσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με τη συντεταγμένη κατά x (σχ. 5.3). Ο συντελεστής προκύπτει από τη σχέση:

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\varrho_\infty u_\infty^2} \tag{5.1}$$

όπου ο δείκτης "∞" συμβολίζει τις τιμές των μεγεθών στο άπειρο.
Οι συντελεστές άνωσης και οπισθέλκουσας παίρνουν αντίστοιχα τις τιμές:
$C_{Lift} = 1,26$ και $C_{Drag} = 0,05$. Στο γράφημα 5.2 αλλά και στη γραφική παράσταση 5.3 διακρίνεται το κύμα κρούσης που εμφανίζεται στην αεροτομή καθώς και το σημείο στο οποίο εμφανίζεται.



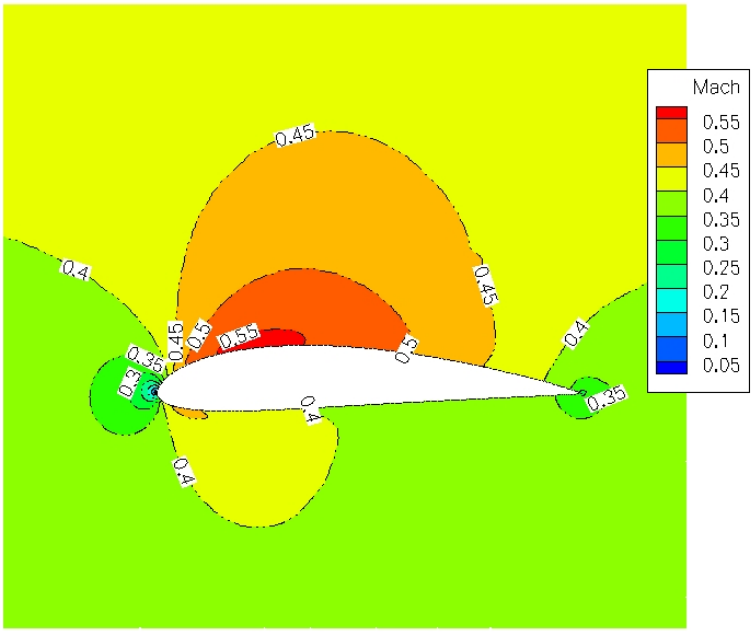
Σχήμα 5.3: Ροή γύρω από την αεροτομή 1 (περίπτωση i). Συντελεστής πίεσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με την τετμημένη της αντίστοιχης θέσης (κεντροκυψελική διατύπωση).

Περίπτωση ii

Επιλέγεται ένα πλέγμα τώρα 10 φορές περίπου πυκνότερο (με 146853 τρίγωνα). Οι συνθήκες της εξωτερικής ροής είναι $M_\infty = 0,4$ και $a_\infty = 0^o$. Ο κώδικας συγκλίνει στις 4952 επαναλήψεις σε $228,5$ δευτερόλεπτα. Οι συντελεστές άνωσης και οπισθέλκουσας που προκύπτουν είναι:$C_{Lift} = 0,53$ και $C_{Drag} = 0,01$. Στα σχ. 5.4 και 5.5 βρίσκονται οι αντίστοιχες γραφικές απεικονίσεις όπως και στο προηγούμενο παράδειγμα. Αυτό που παρατηρείται είναι ότι με την αύξηση των όγκων ελέγχου (περίπου 10x) αυξήθηκαν και οι επαναλήψεις για να συγκλίνει ο κώδικας (3x) αλλά και κυρίως ο χρόνος εκτέλεσης (48x).

5.1.2 Αεροτομή 2

Επιλέγεται αυτή τη φορά η NACA 0012, μια συμμετρική αεροτομή με $M_\infty = 0,5$ και $a_\infty = 0^o$. Αναμένονται συμμετρικά γραφήματα και μηδενικοί συντελεστές άνωσης και οπισθέλκουσας. Ο κώδικας για ένα πλέγμα 48665 τριγώνων - όγκων ελέγχου συγκλίνει μετά από 1197 επαναλήψεις σε χρόνο $15,8$ δευτερόλεπτα. Στα σχ. 5.6 και 5.7 έχουμε την κατανομή του αριθμού Mach γύρω από την αεροτομή και το συντελεστή πίεσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με την τετμημένη της αντίστοιχης θέσης αντίστοιχα. Παρατηρείται: η συμμετρικότητα του σχ. 5.6 και η σύμπτωση του C_p για την πάνω και κάτω πλευρά της αεροτομής. Επίσης, πράγματι προκύπτουν $C_{Lift} = 0,00$ και $C_{Drag} = 0,00$.



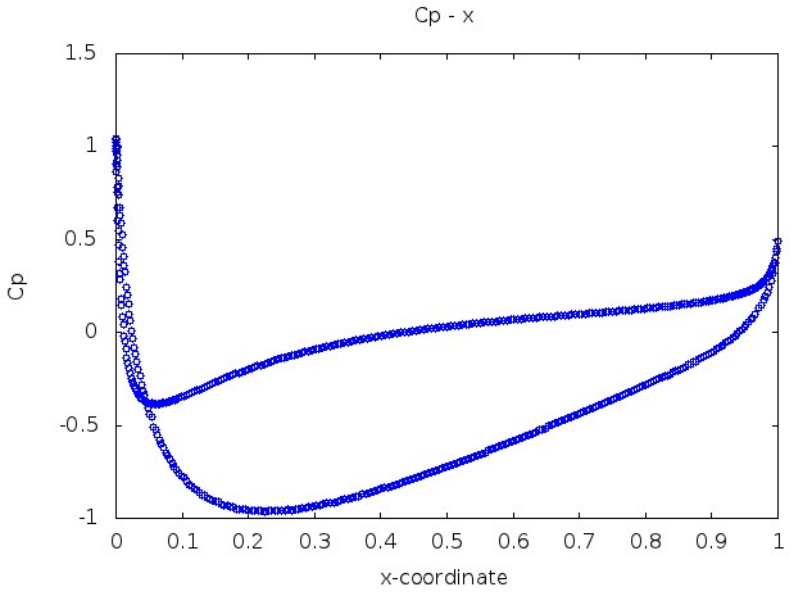
Σχήμα 5.4: Ροή γύρω από την αεροτομή 1 (περίπτωση ii). Κατανομή του αριθμού Mach γύρω από την αεροτομή (κεντροκυψελική διατύπωση).

5.2 Αντιπαραβολή κεντροκυψελικής - κεντροκομβικής μεθόδου

Ένα από τα εναύσματα για την επιλογή του θέματος της παρούσας διπλωματικής εργασίας ήταν η σύγκριση της κεντροκυψελικής διατύπωσης με αυτή της κεντροκομβικής τόσο ως προς τα αποτελέσματα όσο και ως προς την ταχύτητα εκτέλεσης. Θα παρουσιαστούν στη συνέχεια τα αποτελέσματα και οι χρόνοι της κεντροκομβικής μεθόδου για της περιπτώσεις i και ii για την αεροτομή 1 της παραγράφου 5.1.1 και της περίπτωσης για την αεροτομή 2 της παραγράφου 5.1.2. Στη συνέχεια θα καταγραφούν κάποιες διαπιστώσεις αναφορικά με τη σύγκριση των αποτελεσμάτων.

5.2.1 Αεροτομή 1

Επαναλαμβάνεται ότι η αεροτομή είναι η NACA 4415.



Σχήμα 5.5: Ροή γύρω από την αεροτομή 1 (περίπτωση ii). Συντελεστής πίεσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με την τετμημένη της αντίστοιχης θέσης (κεντροκυψελική διατύπωση).

Περίπτωση i

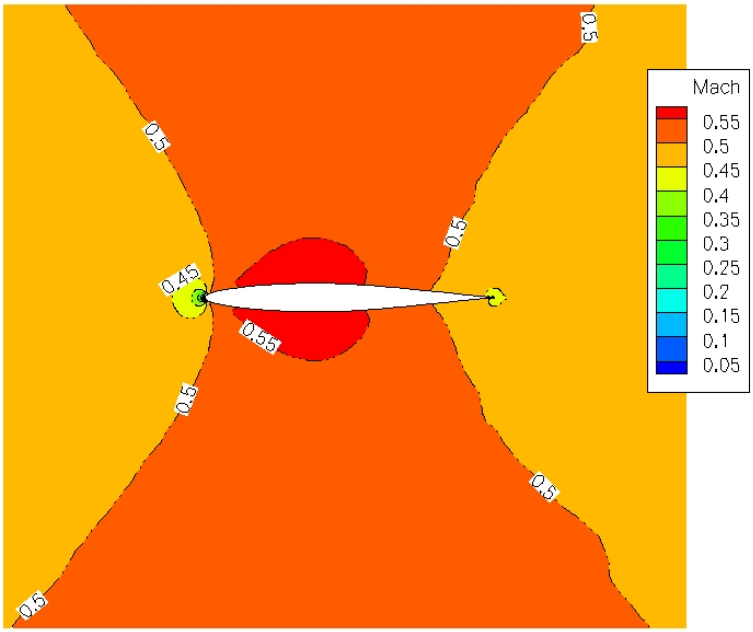
Τα δεδομένα της εξωτερικής ροής είναι $M_\infty = 0,6$ και $a_\infty = 5^o$. Για να πραγματοποιηθεί η σύγκλιση χρειάζονται 991 επαναλήψεις και $3,0$ δευτερόλεπτα. Στο σχ. 5.8 απεικονίζεται ο συντελεστής πίεσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με την τετμημένη της αντίστοιχης θέσης σχεδιασμένο στο ίδιο γράφημα με την ίδια περίπτωση για την κεντροκυψελική διατύπωση. Οι συντελεστές άνωσης και οπισθέλουσας παίρνουν αντίστοιχα τις τιμές:
$C_{Lift} = 1,16$ και $C_{Drag} = 0,08$.

Περίπτωση ii

Σε αυτή την περίπτωση τα δεδομένα της εξωτερικής ροής είναι :$M_\infty = 0,4$ και $a_\infty = 0^o$. Ο κώδικας συγκλίνει σε 4770 επαναλήψεις και $128,8$ δευτερόλεπτα. Το σχ. 5.9 είναι το αντίστοιχο με την παραπάνω περίπτωση. Ακόμα, $C_{Lift} = 0,51$ και $C_{Drag} = 0,02$.

5.2.2 Αεροτομή 2

Για την αεροτομή αυτή θα αναφερθούν μόνο οι επαναλήψεις για τη σύγκλιση: 1021 και ο χρόνος: $7,7$ δευτερόλεπτα καθώς όλα τα υπόλοιπα αποτελέσματα παραμένουν ίδια.



Σχήμα 5.6: Ροή γύρω από την αεροτομή 2. Κατανομή του αριθμού Mach γύρω από την αεροτομή (κεντροκυψελική διατύπωση).

5.2.3 Παρατηρήσεις

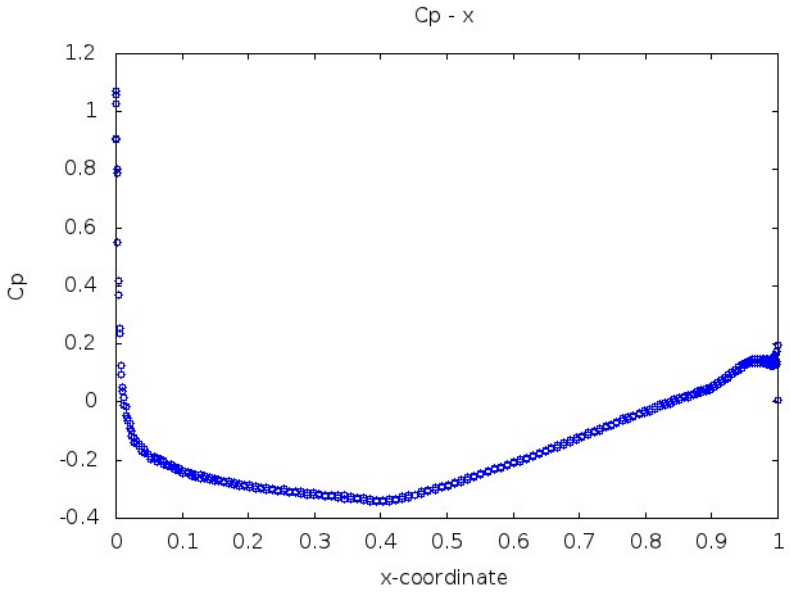
Πριν την παράθεση των παρατηρήσεων για τις διαφοροποιήσεις μεταξύ των δύο μεθόδων ορίζονται τα παρακάτων μεγέθη για ένα διδιάστατο μη-δομημένο πλέγμα αποτελούμενο αποκλειστικά από τριγωνικά στοιχεία:

- n_nod : ο αριθμός των κόμβων
- n_tri: ο αριθμός των τριγωνικών στοιχείων
- n_seg: ο αριθμός των ευθυγράμμων τμημάτων (segments)
- n_nei: ο αριθμός των γειτονικών κόμβων για έναν κόμβο στην κεντροκομβική μέθοδο (αυτοί που ενώνονται μαζί του με ένα ευθύγραμμο τμήμα)

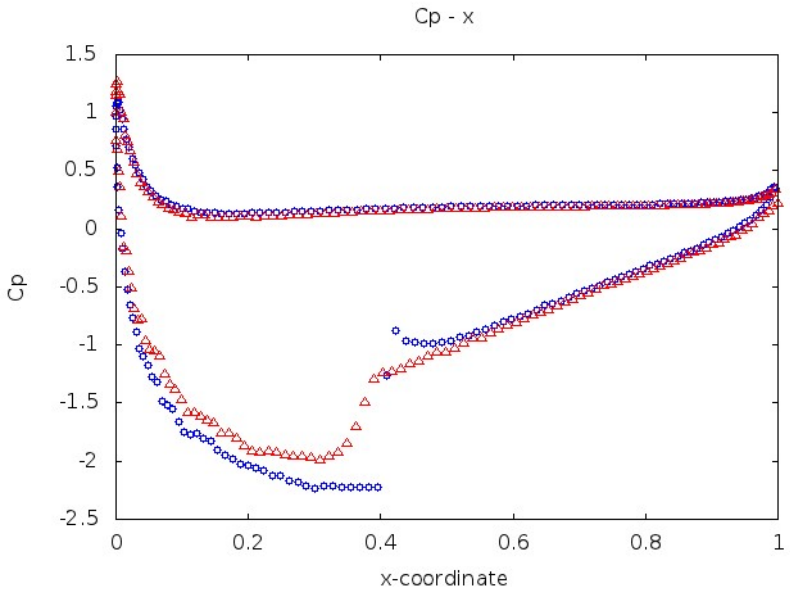
Όταν ένα τέτοιο πλέγμα είναι σχετικά μεγάλο αποδεικνύεται ότι ισχύουν προσεγγιστικά οι παρακάτω σχέσεις:

$$n\_tri \approx 2 \cdot n\_nod \tag{5.2}$$

$$n\_seg \approx 3 \cdot n\_nod \tag{5.3}$$



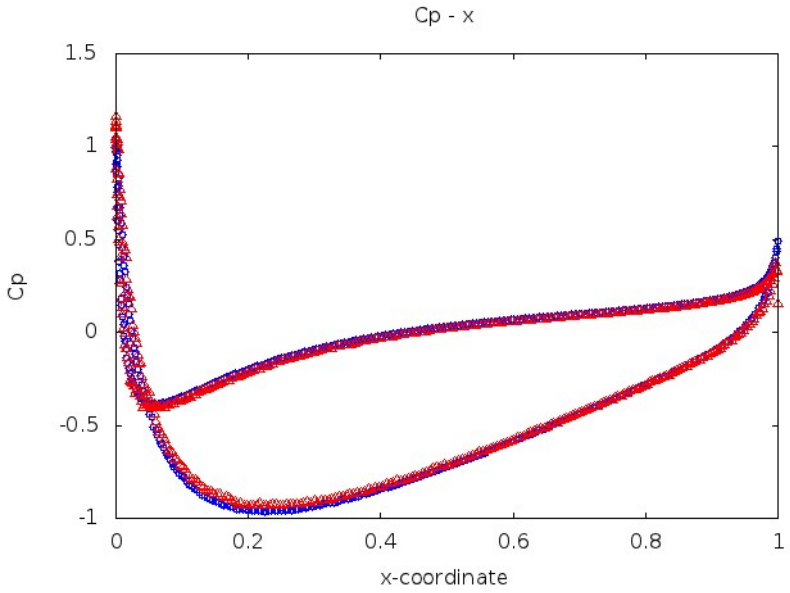
Σχήμα 5.7: Ροή γύρω από την αεροτομή 2. Συντελεστής πίεσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με την τετμημένη της αντίστοιχης θέσης (κεντροκυψελική διατύπωση).



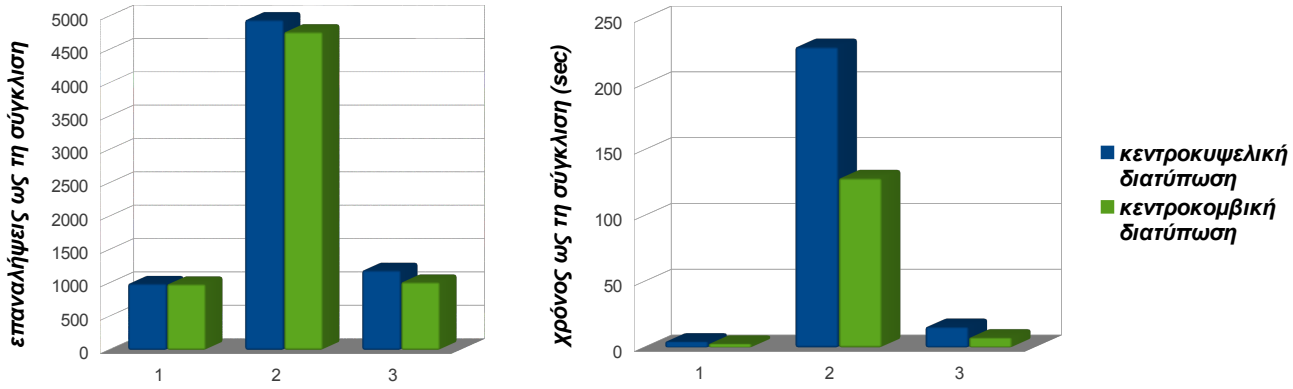
Σχήμα 5.8: Ροή γύρω από την αεροτομή 1 (περίπτωση i). Συντελεστής πίεσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με την τετμημένη της αντίστοιχης θέσης στην κεντροκομβική και κεντροκυψελική διατύπωση. Οι (μπλε) κύκλοι αφορούν την κεντροκυψελική και τα (κόκκινα) τρίγωνα την κεντροκομβική.

Σύγκριση χρόνου-επαναλήψεων σύγκλισης

Συγκρίνοντας τους χρόνους και τις επαναλήψεις για την επίτευξη της σύγκλισης μεταξύ των δύο διατυπώσεων (κεντροκομβική- κεντροκυψελική) προκύπτει το σχ.



Σχήμα 5.9: Ροή γύρω από την αεροτομή 1 (περίπτωση ii). Συντελεστής πίεσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με την τετμημένη της αντίστοιχης θέσης στην κεντροκομβική και κεντροκυψελική διατύπωση. Οι (μπλε) κύκλοι αφορούν την κεντροκυψελική και τα (κόκκινα) τρίγωνα την κεντροκομβική.



Σχήμα 5.10: Σύγκριση επαναλήψεων και χρόνου εκτέλεσης του κώδικα ως τη σύγκλιση για τις περιπτώσεις i, ii της αεροτομής 1 και για την αεροτομή 2 (1, 2, 3 αντίστοιχα στο γράφημα) για την κεντροκομβική και την κεντροκυψελική διατύπωση.

5.10. Η διαφορά μεταξύ των επαναλήψεων σύγκλισης για τις δύο μεθόδους κρίνεται αμελητέα καθώς η κεντροκομβική μέθοδος εμφανίζει μείωση στις επαναλήψεις που απαιτούνται της τάξης του 6%. Παρατηρώντας, όμως, τους χρόνους σύγκλισης ο χρόνος της κεντροκομβικής μεθόδου είναι κατά μέσο όρο 43% μικρότερος. Όπως αναφέρθηκε, υπάρχει δυνατότητα να καταγραφούν οι χρόνοι των επιμέρους τμημάτων του κώδικα. Παρατίθενται οι επιμέρους χρόνοι σε δευτερόλεπτα για τις

δύο διατυπώσεις για τρία από τα κυριότερα τμήματα του κώδικα (αεροτομή 1 - περίπτωση ii).

τμήμα του κώδικα	κεντροκυψελική μέθοδος	κεντροκομβική μέθοδος
υπολογισμός παραγώγων	9,9	1,8
υπολογισμός διανύσματος ροής	51,4	49,7
επίλυση εξισώσεων ροής με σχήμα Jacobi	154,6	71,1
συνολικός χρόνος	228,5	128,8

Παρατηρείται ότι η επίλυση των εξισώσεων ροής με το σχήμα Jacobi καταλαμβάνει κατά μέσο όρο το 61% του συνολικού χρόνου εκτέλεσης του κώδικα και ότι στο συγκεκριμένο τμήμα εντοπίζεται η μεγάλη διαφοροποίηση των δύο μεθόδων. Η εξίσωση που επιλύεται στο συγκεκριμένο τμήμα είναι η 4.6. Ανατρέχοντας σε αυτή και υπολογίζοντας το σύνολο των πολλαπλασιασμών (το υπολογιστικό κόστος προσθέσεων/αφαιρέσεων είναι αμελητέο συγκριτικά με αυτό των πολλαπλασιασμών/διαιρέσεων) που περιλαμβάνει το δεξί μέλος για την κεντροκομβική μέθοδο είναι: $n\_nei + 1$. Πολλαπλασιάζοντας με το σύνολο των κόμβων n_nod και το 4, που είναι ο αριθμός των Jacobi εξισώσεων άνα κόμβο (όσα στοιχεία έχει το διάνυσμα ροής) προκύπτει: $n\_nod \cdot 4 \cdot (n_nei + 1)$. Αναπτύσσοντας:

$$n\_nod \cdot 4 \cdot (n_nei + 1) = \tag{5.4}$$
$$= 4 \cdot n\_nod \cdot n\_nei + 4 \cdot n\_nod = \tag{5.5}$$
$$= 4 \cdot 2 \cdot n\_seg + 4 \cdot n\_nod = \tag{5.6}$$
$$= 4 \cdot 2 \cdot 3 \cdot n\_nod + 4 \cdot n\_nod = \tag{5.7}$$
$$= 28 \cdot n\_nod \tag{5.8}$$

Με αντίστοιχη λογική για την κεντροκυψελική μέθοδο (αριθμός γειτόνων πάντα 3):

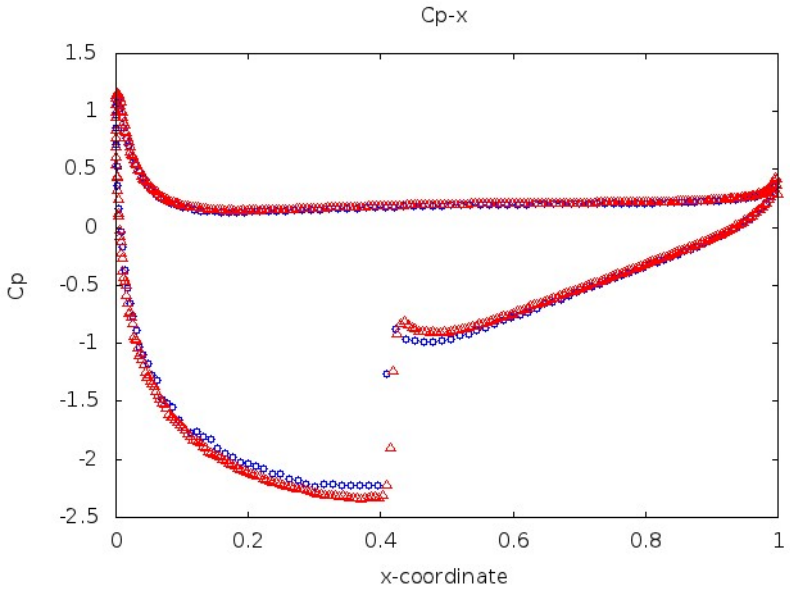
$$4 \cdot n\_tri\,(3 + 1) = \tag{5.9}$$
$$= 4 \cdot 2 \cdot n\_n\_nod \cdot 4 = \tag{5.10}$$
$$= 32 \cdot n\_nod \tag{5.11}$$

Διαπιστώνεται, δηλαδή, ότι στην κεντροκυψελική μέθοδο γίνονται 15% περισσότεροι πολλαπλασιασμοί. Με τον ίδιο τρόπο σκέψης, παρατηρείται ότι οι αντιστροφές του πίνακα της 4.6, πριν γίνει η επίλυση της εξίσωσης Jacobi, είναι n_nod για την κεντροκομβική μέθοδο και $n\_tri = 2 \cdot n\_nod$ για την κεντροκυψελική μέθοδο.

Τα παραπάνω παραδείγματα είναι χαρακτηριστικά και μπορούν να εξηγήσουν ένα μέρος της διαφοράς του χρόνου σύγκλισης μεταξύ των δύο μεθόδων.

Σύγκριση αποτελεσμάτων

Στην περίπτωση ii για την αεροτομή 1 και στην περίπτωση της αεροτομής 2 τα αποτελέσματα αναφορικά με τα ροϊκα μεγέθη μετά τη σύγκλιση είναι ταυτόσημα. Παρατηρώντας, όμως, το σχ. 5.8 της περίπτωσης i είναι εμφανής μια σημαντική διαφοροποίηση γύρω από το σημείο που η κεντροκυψελική μέθοδος αποδίδει το κύμα κρούσης. Διαφοροποίηση υπάρχει και στο συντελεστή άνωσης ($1,16$ η κεντροκομβική έναντι $1,26$ της κεντροκυψελικής διατύπωσης). Σημειώνεται ότι η περίπτωση αυτή είναι η πιο απαιτητική από τις τρεις λόγω ταχύτητας της εξωτερικής ροής και γωνιάς προσβολής. Για να μην υπάρχει αμφιβολία για την ορθότητα του αποτελέσματος, εκτελούμε τον κώδικα της κεντροκομβικής μεθόδου για την ίδια περίπτωση (NACA 4415, $M_\infty = 0,6$, $a_\infty = 5^o$) με το 10 φορές πυκνότερο πλέγμα της περίπτωσης ii (73727 κόμβοι έναντι των 7733 κόμβων της περίπτωσης i) . Το αποτέλεσμα παρουσιάζεται στο σχ. 5.11. Τώρα τα γραφήματα για τις δύο μεθόδους



Σχήμα 5.11: Ροή γύρω από την αεροτομή 1 (περίπτωση i). Συντελεστής πίεσης C_p στην επιφάνεια της αεροτομής σε συνάρτηση με την τετμημένη της αντίστοιχης θέσης στην κεντροκομβική και κεντροκυψελική διατύπωση. Οι μπλε κύκλοι αφορούν την κεντροκυψελική (αραιό πλέγμα) και τα κόκκινα τρίγωνα την κεντροκομβική (πυκνότερο πλέγμα).

είναι ταυτόσημα. Εξάγεται, λοιπόν, το συμπέρασμα ότι η πυκνότητα του πλέγματος της περίπτωσης i δεν ήταν ικανοποιητική για την κεντροκομβική μέθοδο ενώ ήταν για την κεντροκυψελική. Παρακάτω επιχειρείται μια προσπάθεια για την εξήγηση αυτού του συμπεράσματος.

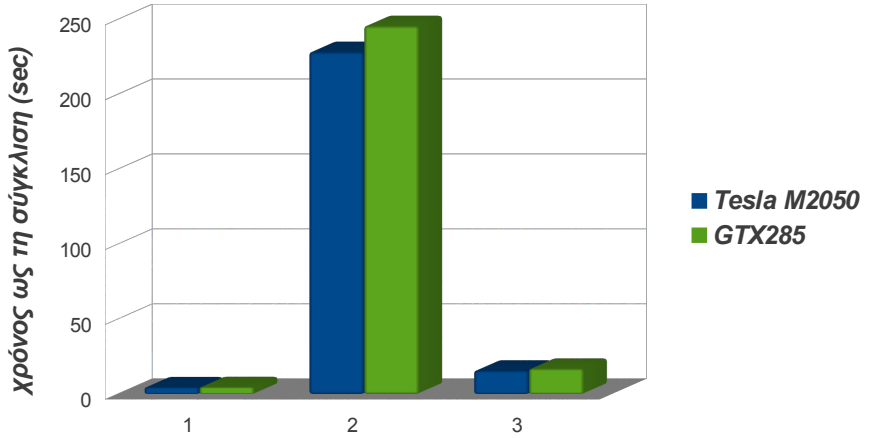
Η ειδοποιός διαφορά των δύο μεθόδων είναι η διαμόρφωση των όγκων ελέγχου. Για την κεντροκομβική διατύπωση κάθε όγκος ελέγχου αντιστοιχεί σε έναν κόμβο

ενώ για την κεντροκυψελική μέθοδο ο όγκος ελέγχου ταυτίζεται με το τρίγωνο. Επειδή, όμως, κατά τη σχέση 5.2, σε ένα μεγάλο πλέγμα τα τρίγωνα είναι κατά προσέγγιση διπλάσια σε πλήθος από τους κόμβους, ισχύει ότι και οι όγκοι ελέγχου της κεντροκυψελικής μεθόδου είναι διπλάσιοι από αυτούς της κεντροκομβικής. Εκεί οφείλεται η μεγαλύτερη ακρίβεια της μεθόδου. Αν, δηλαδή, θεωρηθεί μέτρο της πυκνότητας ενός πλέγματος ο αριθμός των όγκων ελέγχου, είναι δυνατό να ειπωθεί ότι ένας κώδικας "αντιλαμβάνεται" το ίδιο πλέγμα πυκνότερο όταν χρησιμοποιεί την κεντροκυψελική μέθοδο έναντι της κεντροκομβικής.

Προκύπτει, τελικά, σύμφωνα με τα παραπάνω, ότι απαιτείται προσοχή κατά την προσπάθεια σύγκρισης του χρόνου σύγκλισης των δύο μεθόδων στο ίδιο πλέγμα (για τις ίδιες συνθήκες εξωτερικής ροής) με σκοπό να βρεθεί η ταχύτερη καθώς δεν προσφέρουν την ίδια ακρίβεια.

5.3 Σύγκριση επιδόσεων καρτών

Όπως αναφέρθηκε, παρουσιάζει ενδιαφέρον και η σύγκριση των ταχυτήτων εκτέλεσης του κώδικα μεταξύ των καρτών GTX285 και Tesla M2050 που έχουν αρχιτεκτονικές GT200 και Fermi αντίστοιχα. Πραγματοποιούμε την εκτέλεση των τριών περιπτώσεων της παραγράφου 5.1. Οι χρόνοι που προκύπτουν είναι $4,8$, $245,8$ και $17,1$ δευτερόλεπτα αντίστοιχα. Είναι, δηλαδή ταχύτερη η Tesla M2050 σε ποσοστό που φτάνει ως και το $7,5\%$. Επισημαίνεται ότι ο κώδικας που έτρεξε στις δύο κάρτες είναι ο ίδιος. Προβλέπεται ότι σε μια υπολογιστική διαδικασία που θα απαιτούσε τη χρήση και των νέων χαρακτηριστικών της Fermi (νέες μνήμες κ.λ.π.) η διαφοροποίηση θα ήταν εντονότερη.



Σχήμα 5.12: Σύγκριση χρόνων εκτέλεσης του κώδικα για τις περιπτώσεις i, ii της αεροτομής 1 και για την αεροτομή 2 (1, 2, 3 αντίστοιχα στο γράφημα) στις κάρτες GTX285 και Tesla M2050.

Κεφάλαιο 6

Ανακεφαλαίωση - Συμπεράσματα

Η παρούσα διπλωματική εργασία εντάσσεται στην ερευνητική δραστηριότητα της Μονάδας Παράλληλης Υπολογιστικής Ρευστοδυναμικής & Βελτιστοποίησης (ΜΠΥΡ&Β) του Εργαστηρίου Θερμικών Στροβιλομηχανών (ΕΘΣ) της Σχολής Μηχανολόγων Μηχανικών του Ε.Μ.Π.. Η ΜΠΥΡ&Β, τα τελευταία χρόνια, ασχολείται συστηματικά με την αριθμητική επίλυση προβλημάτων Υπολογιστικής Ρευστοδυναμικής δημιουργώντας λογισμικό που εκμεταλλεύεται την παραλληλία που προσφέρουν οι κάρτες γραφικών. Στο πλαίσιο αυτό, έχουν δημιουργηθεί κώδικες που επιλύσουν τις εξισώσεις ροής σε μη-δομημένα και δομημένα πλέγματα με κεντροκομβική διατύπωση [11] [12] [13] [9]. Στόχος, της παρούσας διπλωματικής ήταν η δημιουργία κώδικα που επιλύει τις διδιάστατες εξισώσεις ροής (Euler) με τη μέθοδο των πεπερασμένων όγκων με τη χρήση μη-δομημένου πλέγματος με κεντροκυψελική διατύπωση και η διερεύνηση της συγκεκριμένης τεχνικής.

Κατά την εκπόνηση της εργασίας, διατυπώθηκαν και διακριτοποιήθηκαν οι εξισώσεις ροής χρησιμοποιώντας ως όγκους ελέγχου τα τριγωνικά στοιχεία του πλέγματος και ως βάση της χωρικής διακριτοποίησης το κέντρο βάρους τους. Πάνω στην ίδια γεωμετρική λογική, διατυπώθηκαν οι οριακές συνθηκες και διαμορφώθηκαν οι σχέσεις υπολογισμού των χωρικών παραγώγων των ροϊκών μεγεθών με προσαρμογή της μεθόδου ελαχίστων τετραγώνων. Η μέθοδος που επιλέχθηκε για την επίλυση των εξισώσεων ροής είναι αυτή της χρονοπροέλασης, για την εφαρμογή της οποίας οι εξισώσεις ροής μετασχηματίστηκαν με τη χρήση του σχήματος Jacobi.

Στη συνέχεια, με βάση τα παραπάνω δεδομένα δημιουργήθηκε ο αντίστοιχος κώδικας που πραγματοποιεί την επίλυση της ροής με γνώμονα τον κώδικα της ΜΠΥΡ&Β που έκανε χρήση της κεντροκομβικής διατύπωσης για μη-δομημένα πλέγματα. Η γλώσσα στην οποία έγινε η σύνταξη του κώδικα είναι η CUDA C και η εκτέλεση έγινε σε κάρτες γραφικών Tesla M2050 της εταιρίας NVIDIA. Κατά τον προγραμματισμό, έγινε προσπάθεια εκμετάλλευσης των δυνατοτήτων που προσφέρει η CUDA C για το βέλτιστο χειρισμό της κάρτας γραφικών και προσαρμογή στις ιδιομορφίες της παράλληλης εκτέλεσης σημαντικού μέρους των εντολών με στόχο

την επιτάχυνση της σύγκλισης του κώδικα.

Τα αποτελέσματα από την εκτέλεση του κώδικα οδήγησαν σε χρήσιμα συμπεράσματα για τη μέθοδο της κεντροκυψελικής διατύπωσης σε μη-δομημένα πλέγματα. Αρχικά, πραγματοποιήθηκε η πιστοποίηση του κώδικα αναφορικά με τα αποτελέσματα του με βάση τη σύγκριση τους με τα αποτελέσματα του προϋπάρχοντα κώδικα της ΜΠΥΡ&Β για μη-δομημένα πλέγματα με κεντροκομβική διατύπωση. Ακολούθησε η σύγκριση μεταξύ των δύο μεθόδων τόσο ως προς τα αποτελέσματα, όσο και ως προς τους χρόνους σύγκλισης.

Η σύγκριση των χρόνων για την επίτευξη της σύγκλισης μεταξύ των δύο μεθόδων για ίδια πλέγματα σε ίδιες συνθήκες εξωτερικής ροής ανέδειξε, αρχικά, την κεντροκομβική μέθοδο ως ταχύτερη. Αποδείχθηκε, όμως, πως εξαιτίας της χρήσης σχεδόν διπλάσιων όγκων ελέγχου κατά τη χρήση της κεντροκυψελικής μεθόδου, τα αποτελέσματα της είναι πιο ακριβή όταν γίνεται χρήση του ίδιου πλέγματος, καθώς το πλέγμα νοητά "πυκνώνει". Αυτό πρακτικά σημαίνει πως για να ανταποκριθεί η κεντροκομβική μέθοδος στην ακρίβεια της κεντροκυψελικής μεθόδου θα έπρεπε να κάνει χρήση πυκνότερου πλέγματος, γεγονός που θα αύξανε το χρόνο ως την επίτευξη της σύγκλισης.

Ολοκληρώνοντας, ακολουθούν κάποιες προτάσεις για μελλοντική έρευνα. Η κεντροκυψελική μέθοδος θα ήταν εξίσου ενδιαφέρον να χρησιμοποιηθεί για την επίλυση των τριδιάστατων εξισώσεων ροής με τη χρήση επεξεργαστή κάρτας γραφικών. Θα μπορούσε, ακόμα, να προγραμματιστεί κατάλληλα ώστε να επιλύει τις Navier-Stokes εξισώσεις ροής (όχι μόνο τις εξισώσεις Euler) και τα αποτελέσματα που θα προκύψουν να συγκριθούν με αυτά άλλων μεθόδων, όχι μόνο αναφορικά με τη διατύπωση του μη-δομημένου πλέγματος αλλά ευρύτερα. Τέλος, μια άλλη πρόταση είναι η προσαρμογή και χρήση της κεντροκυψελικής διατύπωσης στη βελτιστοποίηση αεροδυναμικών μορφών, κυρίως μέσω στοχαστικών αλγορίθμων.

Βιβλιογραφία

[1] Κ.Χ. Γιαννάκογλου, Μέθοδοι Βελτιστοποίησης στην Αεροδυναμική, Αθήνα (2006), Ε.Μ.Π

[2] Κ.Χ. Γιαννάκογλου 'Συνεκτικές ροές στις Στροβιλομηχανές', Ε.Μ.Π

[3] Κ. Μαθιουδάκης, 'Λειτουργία αεριοστροβίλων και ατμοστροβίλων' Εκδόσεις ΕΜΠ (2η Έκδοση), Αθήνα, 2009, Ε.Μ.Π

[4] Κ.Δ. Παπαηλίου, Κ.Μ. Μαθιουδάκης, Κ.Χ. Γιαννάκογλου 'Εισαγωγή στις θερμικές στροβιλομηχανές', Αθήνα, 2000, Ε.Μ.Π

[5] Ξ. Σ. Τρομπούκης, Υπολογιστική ανάλυση και παραμετρική διερεύνηση της τεχνικής συνεχούς αναρρόφησης για τον έλεγχο οριακών στρωμάτων, Διπλωματική Εργασία, Εργαστήριο Θερμικών Στροβιλομηχανών ΕΜΠ, Αθήνα, 2007

[6] Ε. Ι. Φουντής, Προγραμματισμός σε Κάρτες Γραφικών και Εφαρμογή στην Αεροδυναμική Βελτιστοποίηση, Διπλωματική Εργασία, Εργαστήριο Θερμικών Στροβιλομηχανών ΕΜΠ, Αθήνα, 2009

[7] Ξ. Τρομπούκης. Αριθμητική επίλυση προβλημάτων αεροδυναμικής-αεροελαστικότητας σε επεξεργαστές καρτών γραφικών, Διδακτορική διατριβή, Εργαστήριο Θερμικών Στροβιλομηχανών, Ε.Μ.Π., Αθήνα, σε εξέλιξη

[8] Γ. Βαλσαμάκης, Αριθμητική επίλυση μη-μόνιμου πεδίου ροής σε κάρτες γραφικών με απεικόνιση του σε πραγματικό χρόνο, Διπλωματική Εργασία, Εργαστήριο Θερμικών Στροβιλομηχανών ΕΜΠ, Αθήνα, 2010

[9] Ι. Σ. Καββαδίας, Προγραμματισμός επιλύτη 3Δ εξισώσεων ροής ατριβούς ρευστού σε δομημένα πλέγματα, σε κάρτες γραφικών, Διπλωματική Εργασία, Εργαστήριο Θερμικών Στροβιλομηχανών ΕΜΠ, Αθήνα 2011

[10] Γ. Δ. Ρήγας. Προσομοίωση και χαμηλού κόστους βελτιστοποίηση του ενεργητικού ελέγχου ροής ρευστού γύρω από αεροτομή, σε κάρτες γραφικών, Διπλωματική εργασία, Εργαστήριο Θερμικών Στροβιλομηχανών ΕΜΠ, Ιούλιος 2010.

[11] V. G. Asouti, X. S. Trompoukis, I. C. Kampolis, K. C. Giannakoglou. Unsteady CFD computations using vertex-centered finite volumes for unstructured grids on Graphics Processing Units, International journal for numerical methods in fluids, Int. J. Numer. Meth. Fluids (2010).

[12] I. C. Kampolis, X. S. Trompoukis, V. G. Asouti, K. C. Giannakoglou. CFD-based analysis and two-level aerodynamic optimization on graphics processing units, Computer Methods in Applied Mechanics and Engineering, Volume 199, Issues 9-12, 15 January 2010

[13] X.S. Trompoukis, V.G. Asouti, I.C. Kampolis, K.C. Giannakoglou. CUDA implementation of Vertex-Centered, Finite Volume CFD methods on Unstructured Grids with Flow Control Applications, GPU Computing Gems Vol. 2, Editor: Wei-mei Hwu, Morgan Kaufmann, 2011 (to be published).

[14] N. Goodnight, C. Woolley, G. Lewin, D. Luebke, G. Humphreys, A multigrid solver for boundary value problems using programmable graphics hardware, Graphics Hardware (2003) 1–11.

[15] T. Brandvik, G. Pullan. Acceleration of a 3D Euler solver using commodity graphics hardware. AIAA Paper 2008-607, 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2008.

[16] T. Brandvik, G. Pullan. Acceleration of a two-dimensional Euler flow solver using commodity graphics hardware. Proceedings of the Institution of Mechanical Engineers, Pt C: Journal of Mechanical Engineering Science 2007; 221(12):1745–1748.

[17] J. Cohen, M. Molemaker. A fast double precision CFD code using CUDA. Proceedings of Parallel CFD 2009, Moffett Field, CA, May 2009.

[18] T. R. Hagen and J. R. Natvig, J. Solving the Euler equations on graphics processing units. Computational Science—ICCS 2006; 3994:220–227

[19] J. Bolz, I. Farmer, E. Grinspun, P. Schröder, Sparse matrix solvers on the GPU: conjugate gradients and multigrid. ACM Transactions on Graphics 2003; 22(3):917–924.

[20] J. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Kruger, A. Lefohn, T. Purcell, A survey of general-purpose computation on graphics hardware. Computer Graphics Forum 2007; 26(1):80–113.

[21] M. Harris, Fast fluid dynamics simulation on the GPU. GPU Gems, chapter 38. Pearson Education, Inc.: Boston, 2004; 637–665.

[22] K. Crane,I. Llamas, S. Tariq. Real-time simulation and rendering of 3D fluids. GPU Gems 3, chapter 30. Pearson Education, Inc.: Boston, 2007; 633–675.

[23] B. van Leer, Flux-Vector Spliting for the Euler Equations, Institute for Computer Applications in Science and Engineering and Leiden State University, Leiden, The Netherlands, 1982

[24] R. Courant, K. Friedrichs, H. Lewy, "Über die partiellen Differenzengleichungen der mathematischen Physik" (in German), Mathematische Annalen, 1928

[25] J. Sanders, E. Kandrot, NVIDIA, CUDA by example, An introduction to General-Purpose GPU Programming, 2011

[26] NVIDIA, NVIDIA CUDA C Programming Guide, Version 4.0, 2010

[27] NVIDIA, NVIDIA CUDA C Programming, Best Practices Guide, CUDA Toolkit 2.3, 2009

[28] NVIDIA, NVIDIA CUDA Architecture, Introduction & Overview, Version 1.1, 209

[29] K. Asanovic, R. Bodik, B. Christopher Catanzaro, J. J. Gebis, P. Husbands, K. Keutzer, D. A. Patterson, W. L. Plishker, J. Shalf, S. W. Williams and K. A. Yelick. The landscape of parallel computing research: A view from berkeley. Technical Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley, Dec 2006

[30] M. C Schatz, C. Trapnell, A. L. Delcher and A. Varshney. High-throughput sequence alignment using Graphics Processing Units, BMC Bionformatics, 474(1471-2105), 10 December 2007,

[31] NVIDIA GeForce® GTX 200 GPU Architectural Overview, Second-Generation Unified GPU Architecture for Visual Computing, Technical Brief

[32] NVIDIA Fermi Compute Architecture Whitepaper

[33] D. P. Director, The Top 10 Innovations in the New NVIDIA Fermi Architecture, and the Top 3 Next Challenges, Parallel Computing Research Laboratory (Par Lab), U.C. Berkeley, 2009

[34] P. N. Glaskowsky, NVIDIA's Fermi: The First Complete GPU Computing Architecture. 2009

[35] J. Michalakes, M. Vachharajani, GPU Acceleration of Numerical Weather Prediction. Parallel Processing Letters Vol. 18 No. 4. World Scientific. Dec. 2008. pp. 531–548.

[36] V. Tossavainen, GPU-accelerated CFD Simulations with OpenFOAM, GPU-accelerated CFD Simulations CSC, Espoo, 2011-05-17 3/18,

[37] D. Komatitscha, G. Erlebacherc, D.Göddeked, D. Michéaa, High-order finite-element seismic wave propagation modeling with MPI on a large GPU cluster. J. Comput. Phys. 229, 20 (October 2010), 7692-7714.

[38] Y. Liu, X. Liu, E. Wu, Real-time 3D fluid simulation on GPU with complex obstacles, Computer Graphics and Applications, 2004. PG 2004. Proceedings. 12th Pacific Conference on , vol., no., pp. 247- 256, 6-8 Oct. 2004

[39] M. Bisson, M. Bernaschi, S. Melchionna, S. Succi, E. Kaxiras. Multiscale Hemodynamics Using GPU Clusters. Communications in Computational Physics, 11 (2012), pp. 48-64.

[40] C. J. Fluke, D. G. Barnes, B. R. Barsdell, A. H. Hassan, Astrophysical Supercomputing with GPUs: Critical Decisions for Early Adopters, PASA, 28 (2011) 15-27

[41] http://en.wikipedia.org/wiki/Moore's_law

[42] http://www.theverge.com/2011/11/3/2535607/cpu-and-gpu-the-convergent-technology

[43] http://www.scicomp.com/parallel_computing/GPU_OpenMP

[44] http://en.wikipedia.org/wiki/Jacobi_method

[45] http://en.wikipedia.org/wiki/NACA_airfoil