



ΤΕΙ ΑΘΗΝΑΣ ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Προγραμματισμός γενικού σκοπού σε μονάδες
επεξεργασίας γραφικών
(GPGPU)

Επόπτες:

ΓΕΩΡΓΙΟΣ ΜΠΑΡΔΗΣ
ΓΕΩΡΓΙΟΣ ΜΙΑΟΥΛΗΣ
ΧΡΗΣΤΟΣ ΓΙΑΚΟΥΜΕΤΤΗΣ

Φοιτητής:

ΓΕΩΡΓΙΟΣ ΣΟΦΙΑΝΟΣ
Α.Μ: 031053

Ημερομηνία παράδοσης:
10 Νοεμβρίου 2014

Περίληψη

Περίληψη

Σκοπός αυτής της εργασίας είναι η διερεύνηση και αξιολόγηση προτύπων και τεχνολογιών για προγραμματισμό γενικού σκοπού με χρήση μονάδων επεξεργασίας γραφικών, και η παρουσίαση εφαρμογών υψηλών υπολογιστικών απαιτήσεων οι οποίες εκμεταλλεύονται τις δυνατότητες αυτών των τεχνολογιών για επιτάχυνση και αύξηση των επιδόσεων τους.

Λόγω της μεγάλης έκτασης των δυνατοτήτων της τεχνολογίας GPGPU, επιλέχθηκαν συγκεκριμένα πεδία για έρευνα τα οποία παρουσιάζουν ιδιαίτερο ενδιαφέρον. Αντίστοιχο βάρος δόθηκε στην έρευνα και καταγραφή των εφαρμογών που χρησιμοποιούνται στα συγκεκριμένα πεδία για την επίλυση διάφορων προβλημάτων. Τέλος, επιλέχθηκαν εφαρμογές για παρουσίαση που περιέχουν αρκετές γραφικές αναπαραστάσεις των λύσεων των προβλημάτων, αλλά και εφαρμογές που μελετούν την απόδοση των υλοποιήσεων σε προβλήματα που απασχολούν τους προγραμματιστές λόγω εντατικών υπολογισμών.

Abstract

The purpose of this thesis is the research and evaluation of patterns and technologies used by general purpose programming on graphics processing units, and the demonstration of high processing computing applications which exploit the capabilities of these technologies in order to accelerate and increase their performance.

Because of the great range of possibilities for GPGPU technology, specific fields were chosen for research which have a special interest. Additional focus was given for research and document the applications that are being actively used in the specific fields as solutions for various problems. Last, some applications were chosen for demonstration that include many graphical implementations of solutions of the problems, and applications that study the performance of implementations of computational intensive problems.

Περιεχόμενα

Περίληψη	III
Εισαγωγή	VII
1 Υλοποιήσεις	1
1.1 Ιστορία	1
1.1.1 Μέλλον	2
1.1.2 Προβλήματα	2
1.2 CUDA	3
1.2.1 Εισαγωγή	3
1.2.2 Πλεονεκτήματα	4
1.2.3 Περιορισμοί	5
1.2.4 Μνήμη	5
1.2.5 CUDA streams	6
1.3 OpenCL	8
1.3.1 Εισαγωγή	8
1.3.2 Επισκόπηση	9
1.3.3 Ιστορία	9
1.3.4 Στόχοι	10
1.3.5 Μοντέλο εκτέλεσης	10
1.3.6 WebCL	11
1.4 Compute Shaders	14
1.4.1 Shaders	14
1.4.2 Εισαγωγή	15
1.4.3 Χώρος υπολογισμού	16
1.4.4 Υλοποίηση OpenGL	16
1.4.5 Υλοποίηση DirectX	18
1.5 PathScale Enzo	19
1.5.1 Εισαγωγή	19
1.6 Συμπεράσματα	21
2 Περισσότερα	23
2.1 Διαφορές CPUs και GPUs	23
2.1.1 Διαφορές προγραμματιστικού μοντέλου	24
2.1.2 Ανάγκη χώρου	24
2.2 Συμπλέγματα GPU	26
2.2.1 VirtualCL	26
2.3 Συμπεράσματα	29
3 Εφαρμογές	31

3.1	Εισαγωγή	31
3.2	Κρυπτογράφηση	32
3.2.1	Κρυπτογράφηση συμμετρικού κλειδίου	32
3.2.2	Κρυπτογράφηση δημοσίου κλειδίου	34
3.2.3	Hashcat	34
3.2.4	Κρυπτο-νομίσματα	35
3.3	Βιοπληροφορική	37
3.3.1	Εισαγωγή	37
3.3.2	Μοριακή δυναμική	38
3.3.3	GPUGRID.net	40
3.3.4	Προγράμματα	41
3.4	Κβαντική χημεία	43
3.4.1	Εισαγωγή	43
3.4.2	Προγράμματα	43
3.5	Δυναμική Ρευστών	45
3.5.1	Τυρβώδης ροή	45
3.5.2	GPU	46
3.5.3	Tsunami	46
3.5.4	Προγράμματα	48
3.6	Ψυχαγωγία	50
3.6.1	Εισαγωγή	50
3.6.2	Παιχνίδια	50
3.6.3	Υλοποιήσεις	51
3.7	Εικόνα και Βίντεο	53
3.7.1	Ιχνογράφηση ακτίνας	53
3.7.2	Μετατροπή	57
3.7.3	Μειονεκτήματα	57
3.7.4	Βίντεο	58
3.7.5	Συμπίεση δεδομένων γενετικής	59
3.7.6	Προγράμματα	59
3.7.7	Επεξεργασία εικόνας	61
3.8	Γεωπληροφορική	63
3.8.1	Σεισμική δραστηριότητα	63
3.8.2	Προγράμματα	67
3.9	Καιρός και κλίμα	69
3.9.1	Υπολογισμοί	69
3.9.2	Προγράμματα	69
3.10	Αστροφυσική	70
3.11	Συμπεράσματα	72
4	Μεθοδολογία	73
4.1	Εργαλεία που χρησιμοποιήθηκαν	73
4.1.1	X _Y TEX	73
4.1.2	Texmaker	73
4.1.3	Εταιρεία Ελληνικών Τυπογραφικών Στοιχείων	74
4.1.4	TortoiseGit	74
4.1.5	Github	75
4.1.6	Παρουσίαση	76

4.2 Άδεια χρήσης	79
4.2.1 Εισαγωγή	79
4.2.2 Επιλογή	80
5 Συμπεράσματα	83
5.1 Γενικά	83
5.2 Προτάσεις	85

Εισαγωγή

"Anyone can build a fast CPU. The trick is to build a fast system."

Seymour Cray

Τα αρχικά GPGPU πηγάζουν από την φράση General Purpose computation on Graphics Processing Units, ή αλλιώς γνωστή ως GPU Computing, δηλαδή υπολογισμός γενικού σκοπού σε μονάδες επεξεργασίας γραφικών.

Οι GPUs, είναι επεξεργαστές υψηλών επιδόσεων με δυνατότητα πολύ υψηλού υπολογισμού και διεκπεραιωτικότητας δεδομένων. Σχεδιασμένες αρχικά για γραφικά υπολογιστών με αρκετές δυσκολίες στον προγραμματισμό τους, οι σημερινές μονάδες επεξεργασίας γραφικών είναι παράλληλοι επεξεργαστές γενικής χρήσης με υποστήριξη για προσβάσιμες προγραμματιστικές διεπαφές και βιομηχανικά πρότυπα γλωσσών όπως η C.

Οι προγραμματιστές που μεταφέρουν τις εφαρμογές τους σε GPUs συνήθως πετυχαίνουν ταχύτητες πολλαπλάσιες από ότι μια αντίστοιχη εφαρμογή ειδικά βελτιστοποιημένη για κεντρική μονάδα επεξεργασίας (CPU). Ο όρος GPGPU δημιουργήθηκε από τον Mark Harris το 2002 όταν συνειδητοποίησε ότι αναπτυσσόταν μια τάση για χρήση των μονάδων επεξεργασίας γραφικών για εφαρμογές που δεν είχαν σχέση με γραφικά. Οποιοσδήποτε κοίταγε κάποιον GPGPU κώδικα 5 χρόνια πριν θα πίστευε ότι είναι απλός κώδικας γραφικών. Ο κώδικας φόρτωνε περιεργα μέρη υφών στην GPU, ζωγράφιζε τρίγωνα και τα μετακινούσε, και τέλος διάβαζε το αποτέλεσμα στην μνήμη, και επαναλάμβανε.[1]

Δεν έβγαζε νόημα για κάποιον να τον παρακολουθήσει, αλλά οι προγραμματιστές που γνώριζαν τον εσωτερικό τρόπο λειτουργίας, το θεωρούσαν θαύμα του κόσμου της πληροφορικής. Μπορούσαν να στείλουν δεδομένα σε υφές και να τις εμφανίσουν στην οθόνη. Να εμφανίσουν άλλες υφές πάνω από αυτές, και να αλλάξουν τις ιδιότητες της αντανάκλασης και άλλων λειτουργιών, και να διαβάσουν πίσω τα αποτελέσματα. Υπήρχαν όμως προβλήματα όπως ότι η μεγαλύτερη καθυστέρηση προερχόταν από την φόρτωση των δεδομένων στην GPU, και στην ανάγνωση τους από αυτήν. Επίσης δεν υπήρχε δυνατότητα διακλαδώσεων λειτουργιών και προϋποθέσεων, ή στην καλύτερη ήταν πολύ περιορισμένη. Τέλος, η γλώσσα ήταν πολύ δύσκολη να χρησιμοποιηθεί.

Τα δύο πρώτα αποτελούν ακόμα πρόβλημα. Η μεταφορά δεδομένων από την GPU προς την κύρια μνήμη είναι η μεγαλύτερη καθυστέρηση που συμβαίνει στα προγράμματα GPGPU. Αυτό σημαίνει ότι οι αλγόριθμοι μας πρέπει να είναι δομημένοι με τέτοιο τρόπο ώστε όλα τα δεδομένα να μπορούν να αποσταλούν στην GPU, να επεξεργαστούν με κάποιον τρόπο, και να μας επιστρέψει την απάντηση. Οι αλγόριθμοι που υπολογίζουν έναν αριθμό και μετά τον χρησιμοποιούν σε άλλον υπολογισμό, με επαναληπτικές διαδικασίες, δεν δουλεύουν καλά σε μια GPU. Αυτή η καθυστέρηση μας ώθησε στο να μετατρέψουμε μεγάλο μέρος κώδικα από αλγορίθμους σε νέα, φιλική μορφή για παράλληλο υπολογισμό.

Από το 2012, οι GPU έχουν αναπτυχθεί σε συστήματα πολυπύρηνων επεξεργαστών παράλληλου υπολογισμού δίνοντας μας την δυνατότητα για

πολύ αποδοτικό χειρισμό μεγάλου όγκου δεδομένων. Αυτός ο σχεδιασμός είναι πιο αποδοτικός από ότι οι κεντρικές μονάδες επεξεργασίας (CPU) για αλγόριθμους όπου η επεξεργασία μεγάλου όγκου δεδομένων γίνεται παράλληλα, όπως σε αλγορίθμους sort μεγάλων λιστών, μετασχηματισμό κυμάτων δυο διαστάσεων, προσομοίωση βιολογικών δυναμικών. Ένα παράδειγμα αυτής της μετατροπής είναι ο παράλληλος FFT (Fast Fourier Transform), που χρησιμοποιείται σε πολλές ρουτίνες επεξεργασίας ήχου και εικόνας.

Τα προβλήματα που αναφέραμε σημαίνουν ότι η επιτάχυνση μέσω GPU σε προγράμματα της αγοράς θα αργήσει να έρθει. Αν και μπορούμε να βρούμε υλοποιήσεις σε εφαρμογές καταναλωτών, η χρήση της είναι περιορισμένη σε περιοχές όπου η έρευνα έχει ολοκληρωθεί, κυρίως σε προγράμματα επεξεργασίας βίντεο/εικόνας. Προγράμματα όπως Adobe Premiere, Nero Move-it, και άλλα, χρησιμοποιούν το CUDA και άλλες GPGPU τεχνολογίες για να μειώσουν τους χρόνους της κωδικοποίησης και αποκωδικοποίησης. Άλλες εφαρμογές, όπως ο έλεγχος ορθογραφικών λαθών ή προγράμματα διαμοιρασμού αρχείων, δεν έχουν βρει ακόμα τρόπους να αυξήσουν την απόδοση μέσω του GPGPU. Οι υπολογιστές ακόμα "πνίγονται" από τους περιορισμούς I/O όταν γράφουν στους δίσκους, και αυτό συνήθως αποτελεί τον λόγο της υπολογιστικής καθυστέρησης καθώς τα αποτελέσματα πρέπει να σώζονται κάπου.[2]

Σκοπός αυτής της εργασίας είναι να διερευνήσει και αξιολογήσει πρότυπα και τεχνολογίες για προγραμματισμό γενικού σκοπού με χρήση μονάδων επεξεργασίας γραφικών, ειδικότερα όσον αφορά εφαρμογές υψηλών υπολογιστικών απαιτήσεων οι οποίες εκμεταλλεύονται τις δυνατότητες αυτών των τεχνολογιών για επιτάχυνση και αύξηση των επιδόσεων τους. Το θέμα είναι μεγάλης σημασίας καθώς είναι δυνατόν να λυθούν ή να υπολογιστούν προβλήματα που σήμερα είναι πολύ δύσκολο να λυθούν, ενώ δημιουργεί καινούριες δυνατότητες και σκέψεις για περαιτέρω ανάπτυξη του τρόπου που οι υπολογιστές μας βοηθάνε να βελτιώσουμε τον τρόπο ζωής μας.

Παράλληλος υπολογισμός

Για 30 χρόνια, ένας από τους πιο σημαντικούς τρόπους για να βελτιώσουμε την απόδοση των υπολογιστικών συσκευών των καταναλωτών ήταν η αύξηση της ταχύτητας στην οποία λειτουργεί το ρολόι ενός επεξεργαστή. Ξεκινώντας από περίπου το 1MHz το 1980, οι περισσότεροι σύγχρονοι επεξεργαστές έχουν ταχύτητες μεταξύ 1GHz και 4GHz, δηλαδή είναι περίπου 1000 φορές πιο γρήγοροι. Αν και δεν είναι ο μόνος τρόπος με τον οποίο έχουν βελτιωθεί οι επεξεργαστές, αποτελεί συνήθως μια αξιόπιστη πηγή για αύξηση της απόδοσης.

Τα τελευταία χρόνια όμως, οι κατασκευαστές έχουν αναγκαστεί να ψάξουν για εναλλακτικούς τρόπους αύξησης της υπολογιστικής δύναμης. Εξ αιτίας διάφορων περιορισμών στην κατασκευή ενσωματωμένων κυκλωμάτων, δεν είναι πλέον εύκολο να αυξάνουμε την ταχύτητα του ρολογιού του επεξεργαστή σαν τρόπο αύξησης της απόδοσης στις υπάρχουσες αρχιτεκτονικές. Στην αναζήτηση για επιπλέον υπολογιστική δύναμη για τους προσωπικούς επεξεργαστές, οι ερευνητές χρησιμοποίησαν τεχνολογίες που ήταν ήδη γνωστές από τους υπερ-υπολογιστές, στους οποίους είναι σύνηθες φαινόμενο να αποτελούνται από δεκάδες ή εκατοντάδες επεξεργαστές, οι οποίοι εκτελούν παράλληλες διεργασίες. Έτσι το 2005, οι κύριοι κατασκευαστές επεξεργαστών άρχισαν να προσφέρουν επεξεργαστές με δύο πυρήνες αντί για έναν.

Τα επόμενα χρόνια, ακολούθησαν υλοποιήσεις με τρεις, τέσσερις, έξι, ακόμα και οκτώ πυρήνες. Έχει ξεκινήσει ήδη μια μεγάλη στροφή της βιομηχανίας υπολογιστών στον παράλληλο υπολογισμό. Με την κυκλοφορία των διπύρηνων μέχρι και 8 ή 16 πυρήνων επεξεργαστών για σταθμούς εργασίας, ο παράλληλος υπολογισμός δεν είναι πλέον υπόθεση που αφορά μόνο τους εξωτικούς υπερ-υπολογιστές. Επίσης οι φορητές συσκευές όπως κινητά τηλέφωνα και φορητές συσκευές μουσικής έχουν αρχίσει να ενσωματώνουν δυνατότητες παράλληλου υπολογισμού σε μια προσπάθεια να προσφέρουν δυνατότητες πολύ ανώτερες από τους προγόνους τους. Όλο και περισσότερο, οι προγραμματιστές λογισμικού πρέπει να εξοικειωθούν με πλατφόρμες και τεχνολογίες παράλληλου υπολογισμού ώστε να προμηθεύουν με πλούσιες εμπειρίες την βάση των χρηστών τους. Το μέλλον αποτελείται από πολύ-νηματικές εφαρμογές, και από φορητές συσκευές που μπορούν ταυτόχρονα να παίζουν μουσική, να εξερευνούν το διαδίκτυο, και να παρέχουν GPS υπηρεσίες.[3]

GPU Computing

Η επιτάχυνση υπολογισμού γενικού σκοπού από μονάδες επεξεργασίας γραφικών, είναι η χρήση των GPUs μαζί με χρήση CPUs για να επιταχύνουν επιστημονικές, αναλυτικές, κατασκευαστικές, καταναλωτικές, και εμπορικές εφαρμογές. Από την πρώτη εμφάνιση της τεχνολογίας το 2007 από την NVIDIA, οι επιταχυντές GPU τώρα βρίσκονται σε κέντρα δεδομένων σε εργαστήρια αποδοτικής ενέργειας για λογαριασμό κυβερνήσεων, πανεπιστήμια, και μικρές και μεγάλες επιχειρήσεις σε όλον τον κόσμο. Οι GPUs επιταχύνουν εφαρμογές σε πλατφόρμες που εκτείνονται από αυτοκίνητα, σε κινητά τηλέφωνα, σε drones και ρομπότ.

Η επιτάχυνση μέσω μονάδων επεξεργασίας γραφικών παρέχει πρωτοφανή αποτελέσματα επιδόσεων διαχωρίζοντας και φορτώνοντας τις εντολές εντατικών υπολογισμών στην GPU, ενώ διατηρεί την εκτέλεση του υπόλοιπου κώδικα στην CPU. Από την οπτική γωνία του χρήστη, οι εφαρμογές απλώς τρέχουν πιο γρήγορα. Η επιτάχυνση εφαρμογών με χρήση GPU ξεκίνησε με την εμφάνιση των shader και την υποστήριξη αριθμών με υποδιαστολή στις μονάδες επεξεργασίας γραφικών. Ένας από τους πρώτους κώδικες που εκτελέστηκε γρηγορότερα σε GPU από ότι σε CPU είναι η υλοποίηση του αλγορίθμου LU Factorization (2005).

Αυτές οι πρώτες προσπάθειες για χρήση GPU σαν επεξεργαστή γενικού σκοπού απαιτούσαν μετατροπή των υπολογιστικών προβλημάτων σε μορφή στοιχείων γραφικών, που υποστηρίζονταν από το OpenGL ή το DirectX. Το κόστος μετατροπής ήταν πολύ μεγάλο. Στα επόμενα κεφάλαια θα μελετήσουμε τις υλοποιήσεις που ακολούθησαν, οι οποίες επέτρεψαν στους προγραμματιστές να αγνοήσουν το επίπεδο των γραφικών και να ασχοληθούν με ένα περιβάλλον προγραμματισμού υψηλής απόδοσης.



Σχήμα 1: Εκτέλεση κώδικα σε CPU και GPU

Υλοποιήσεις

1.1 Ιστορία

Παραδοσιακά, η σειρά αγωγών των γραφικών, αποτελείται από τις καταστάσεις μετατροπή και φωτισμό, συναρμολόγηση αρχέγονων, μετατροπή σε pixels, και σκίαση. Οι πρώτες GPU είχαν όλες τις λειτουργίες που χρειάζονται για να εκτελεστεί η σειρά αγωγών, αλλά με τον καιρό όλο και περισσότερες καταστάσεις έγιναν δυνατό να προγραμματιστούν με την έλευση ειδικών επεξεργαστών, όπως επεξεργαστές κορυφών και τεμάχια επεξεργαστών, που κατέστησαν κάποιες λειτουργίες πιο ευέλικτες.

Όταν οι τιμές συνέχισαν να πέφτουν ενώ η υπολογιστική δύναμη αυξανόταν, η ερευνητική κοινότητα σκέφτηκε τρόπους να αξιοποιηθεί αυτή η δύναμη για τον υπολογισμό δύσκολων λειτουργιών. Όμως, καθώς η δυνατότητα των επεξεργαστών ήταν περιορισμένη και η διεπαφή προγραμματιστικής διεπαφής (API) των οδηγών γραφικών ήταν σχεδιασμένη για να υλοποιεί συγκεκριμένα την σειρά αγωγών, έπρεπε να ληφθούν υπόψιν πολλές παράμετροι.

Για παράδειγμα, όλα τα δεδομένα έπρεπε να κωδικοποιηθούν σε υφές ως πίνακες δυο διαστάσεων που αναπαριστούν pixel, με περιεχόμενο τιμές χρωμάτων και κάποιο κανάλι alpha για την διαφάνεια. Επιπλέον, οι υφές είναι αντικείμενα μόνο προσπελάσιμα, και δεν επαναγράφονταν, κάτι που ανάγκαζε τους προγραμματιστές να αποθηκεύουν κάθε φορά καινούρια υφή με τις αλλαγές. Τέλος, οι περισσότερες GPU υποστήριζαν μόνο λειτουργίες μονής κινητής υποδιαστολής, αναγκάζοντας τους προγραμματιστές να προσομοιώνουν λογικές λειτουργίες.

Αυτοί οι περιορισμοί, ήταν ο μεγαλύτερος λόγος που ώθησε τους κατασκευαστές GPU (AMD, NVIDIA, INTEL), να δημιουργήσουν προγραμματιστικές διεπαφές ειδικές για την κοινότητα του GPGPU και να εξελίξουν τις συσκευές τους για καλύτερη υποστήριξη.

Το πεδίο του προγραμματισμού γενικής χρήσης έχει αναπτυχθεί με ταχύτατους ρυθμούς τα τελευταία χρόνια, έτσι ώστε τώρα υπάρχουν αρκετές υλοποιήσεις για τον προγραμματισμό των μονάδων επεξεργασίας γραφικών. Πρόσφατα, έχουν γίνει προσπάθειες δημιουργίας προτύπων. Ο προγραμματισμός των GPUs αναπτύχθηκε όταν το CUDA και το Stream κατέφθασαν στο τέλος του 2006. Αυτές οι διεπαφές και οι γλώσσες, σχεδιάστηκαν από τις εταιρίες κατασκευής των GPUs σε πολύ κοντινή σχέση με το υλικό, το οποίο αποτέλεσε μεγάλο βήμα προς ένα πιο εύχρηστο, ταιριαστό και μελλοντικά-ασφαλές προγραμματιστικό μοντέλο.

Η ανοιχτή γλώσσα προγραμματισμού (OpenCL) δημιουργήθηκε για να παρέχει ένα γενικό API ετερογενή υπολογισμού σε διάφορες μορφές παράλληλων συσκευών, συμπεριλαμβανομένου μονάδων επεξεργασίας γραφικών, πολυπύρηνων κεντρικών μονάδων επεξεργασίας, κ.α

1.1.1 Μέλλον

Οι πρόσφατες δραστηριότητες των μεγάλων κατασκευαστών μας δείχνουν ότι τα μελλοντικά σχέδια των μικροεπεξεργαστών και μεγάλων HPC συστημάτων θα είναι υβριδικά/ετερογενούς φύσης. Αυτά τα συστήματα θα βασίζονται στην ενσωμάτωση δύο τύπων εξαρτημάτων:

- Τεχνολογία πολυπύρηνων CPU: ο αριθμός των πυρήνων θα συνεχίσει να αυξάνεται λόγω της επιθυμίας να ενσωματώσουμε περισσότερα εξαρτήματα σε ένα τσιπ.
- Ειδικού τύπου υλικό και μαζικά παράλληλους επιταχυντές: Για παράδειγμα, οι GPUs υπερτερούν των CPUs σε απόδοση κινητής υποδιαστολής, τα τελευταία χρόνια. Επίσης ο προγραμματισμός σε αυτές έχει γίνει εύκολος, αν όχι ευκολότερος, από ότι στις CPUs

Η σχετική ισορροπία στα μελλοντικά σχέδια δεν είναι ξεκάθαρη και μπορεί να αλλάξει με την πάροδο του χρόνου. Δεν υπάρχει καμία αμφιβολία ότι οι μελλοντικές γενιές των υπολογιστικών συστημάτων, από τους φορητούς υπολογιστές μέχρι και τους υπερ-υπολογιστές θα αποτελείται από μια σύσταση ετερογενών συστημάτων.

1.1.2 Προβλήματα

Τα προβλήματα και οι προκλήσεις για τους προγραμματιστές στο καινούριο περιβάλλον των υβριδικών συστημάτων, είναι υπαρκτά. Κρίσιμα τμήματα του λογισμικού ήδη δυσκολεύονται να προλάβουν τον ρυθμό των αλλαγών. Σε μερικές περιπτώσεις, η απόδοση δεν είναι ανάλογη του αριθμού των πυρήνων, γιατί ένα μεγάλο μέρος του χρόνου ξοδεύεται στην μετακίνηση των δεδομένων παρά στους υπολογισμούς. Σε άλλες περιπτώσεις, το βελτιστοποιημένο λογισμικό για το συγκεκριμένο υλικό, παραδίδεται χρόνια μετά από την παράδοση του υλικού, και έτσι είναι απαρχαιωμένο όταν παραδοθεί. Και σε άλλες περιπτώσεις, όπως σε μερικές πρόσφατες υλοποιήσεις GPU, το λογισμικό δεν εκτελείται καθόλου γιατί το προγραμματιστικό περιβάλλον έχει αλλάξει υπερβολικά.

1.2 CUDA

1.2.1 Εισαγωγή

Το CUDA είναι μια πλατφόρμα παράλληλου υπολογισμού, που δημιουργήθηκε από την NVIDIA και υλοποιήθηκε στις κάρτες γραφικών τις οποίες παράγει η ίδια. Το CUDA δίνει στους προγραμματιστές άμεση πρόσβαση στο σετ εικονικών εντολών και την μνήμη των στοιχείων του παράλληλου υπολογισμού σε κάρτες γραφικών NVIDIA.

Αξιοποιώντας το CUDA, οι κάρτες γραφικών (GPU) μπορούν να χρησιμοποιηθούν για υπολογισμό γενικής χρήσης (δηλαδή όχι αποκλειστικά για γραφικά). Οι GPU έχουν μια αρχιτεκτονική παράλληλης εξόδου η οποία δίνει έμφαση στην εκτέλεση πολλών threads με μικρή ταχύτητα, σε αντίθεση με τις CPU όπου εκτελείται ένα thread με μεγάλη ταχύτητα.



Σχήμα 1.1: Λογότυπο NVIDIA CUDA

Η πλατφόρμα CUDA είναι προσβάσιμη στους προγραμματιστές μέσω βιβλιοθηκών, εντολών μεταγλώττισης, και προεκτάσεων σε γλώσσες προγραμματισμού βιομηχανικής κλίμακας, όπως η C, C++ και Fortran.

Οι προγραμματιστές της C/C++, χρησιμοποιούν το CUDA C/C++, μεταγλωττισμένο με το nvcc, έναν LLVM βασισμένο μεταγλωττιστή, και οι προγραμματιστές της Fortran χρησιμοποιούν το CUDA Fortran, μεταγλωττισμένο με τον μεταγλωττιστή PGI CUDA Fortran από το The Portland Group. Εκτός από τα παραπάνω, η πλατφόρμα CUDA υποστηρίζει και άλλες διεπαφές υπολογισμού, όπως το OpenCL του Khronos Group, το DirectCompute της Microsoft, και το C++ AMP.

Στην βιομηχανία των υπολογιστών, οι GPUs δεν χρησιμοποιούνται μόνο για τα γραφικά αλλά και στους υπολογισμούς φυσικής παιχνιδιών (π.χ. καπνός, φωτιά, ροή υγρών). Γνωστά παραδείγματα αποτελούν οι μηχανές PhysX και η Bullet. Το CUDA επίσης χρησιμοποιείται για να επιταχύνει

μη-γραφικές εφαρμογές στην βιοπληροφορική, στην κρυπτογραφία, και σε πολλά άλλα πεδία.[4]

Γενικότερα, η υπολογιστική δύναμη της GPU, βασίζεται στην παράλληλη αρχιτεκτονική της. Για αυτό, η πλατφόρμα του CUDA παρουσιάζει το νήμα(thread) ως το μικρότερο στοιχείο παραλληλισμού. Όμως, σε σύγκριση με την κεντρική μονάδα επεξεργασίας, τα νήματα της GPU έχουν μικρότερο κόστος χρήσης πόρων και μικρότερο κόστος δημιουργίας και αντικατάστασης.

Σημειώνεται ότι οι GPU είναι αποτελεσματικές, μόνο όταν τρέχει μεγάλος αριθμός απο τέτοια νήματα. Μια ομάδα από νήματα, που εκτελούνται παράλληλα, επικοινωνούν και συγχρονίζονται μεταξύ τους ονομάζεται block. Ο μέγιστος αριθμός των νημάτων σε ένα block είναι ένας περιορισμός που υπάρχει στην κάθε μονάδα γραφικής επεξεργασίας. Τέλος, μια ομάδα από blocks τα οποία έχουν την ίδια διάσταση και εκτελούνται απο το ίδιο πρόγραμμα CUDA παράλληλα, ονομάζεται πλέγμα.

Για να επιτρέψει βέλτιστη επίδοση για διαφορετικά πρότυπα, το CUDA εκτελεί ένα ιεραρχικό μοντέλο μνήμης, αντίθετα με τα παραδοσιακά μοντέλα που συναντάμε συνήθως στους υπολογιστές. Ο υπολογιστής και η συσκευή, έχουν τις δικές τους περιοχές μνήμης, τις οποίες ονομάζουν host memory και device memory, αντίστοιχα. Το CUDA παρέχει βελτιστοποιημένες λειτουργίες για να μεταφέρει δεδομένα από και προς αυτούς τους ξεχωριστούς χώρους. Κάθε νήμα κατέχει το δικό του αρχείο καταχώρησης, το οποίο μπορεί να προσπελαστεί και να εγγραφεί.

Επιπλέον, μπορεί να προσπελάσει το δικό του αντίγραφο της τοπικής μνήμης. Όλα τα νήματα στο ίδιο πλέγμα μπορούν να προσπελάσουν και να γράψουν στην περιοχή της κοινόχρηστης μνήμης (shared memory). Για να αποφευχθούν κίνδυνοι από ταυτόχρονη προσπέλαση, μηχανισμοί συγχρονισμού νημάτων πρέπει να χρησιμοποιηθούν. Η κοινόχρηστη μνήμη, είναι οργανωμένη σε ομάδες που ονομάζονται τράπεζες, οι οποίες μπορούν να προσπελαστούν παράλληλα. Όλα τα νήματα έχουν επίσης πρόσβαση στον χώρο μνήμης που ονομάζεται καθολική μνήμη (global memory) και στις περιοχές που ονομάζονται μνήμη σταθερών (constant memory) και μνήμη υφής (texture memory).[5]

1.2.2 Πλεονεκτήματα

Το CUDA έχει τα εξής πλεονεκτήματα σε σχέση με τους παραδοσιακούς τρόπους υπολογισμού γενικής χρήσης που εκτελούνται μέσω προγραμματιστικών διεπαφών γραφικών:

- Διασκορπισμένες προσπελάσεις - ο κώδικας μπορεί να διαβαστεί από αυθαίρετες διευθύνσεις στην μνήμη.
- Ενοποιημένη εικονική μνήμη (CUDA 6)
- Κοινόχρηστη μνήμη - το CUDA εκθέτει μια γρήγορη περιοχή κοινόχρηστης μνήμης (μέχρι 48KB για κάθε επεξεργαστή) η οποία μπορεί να μοιραστεί ανάμεσα στα threads. Αυτή μπορεί να χρησιμοποιηθεί σαν κρυφή μνήμη διαχείριση απο τον χρήστη, επιτρέποντας μεγαλύτερο εύρος δεδομένων απο ότι είναι δυνατό με τις προσπελάσεις υφών.
- Πιο γρήγορες μεταφορτώσεις και προσπελάσεις από και προς την GPU

- Πλήρης υποστήριξη για ακέραιες και bitwise λειτουργίες, για παράδειγμα τις προσπελάσεις υφών.

1.2.3 Περιορισμοί

Το CUDA δεν υποστηρίζει ολόκληρο το πρότυπο της γλώσσας C, καθώς τρέχει μέσω ενός μεταγλωττιστή C++, ο οποίος εμποδίζει συγκεκριμένα μέρη της γλώσσας C να μεταγλωττιστούν.

1.2.4 Μνήμη

Κοινή μνήμη και συγχρονισμός

Ο μεταγλωττιστής CUDA C μεταχειρίζεται τις μεταβλητές στην κοινή μνήμη με διαφορετικό τρόπο από ότι τις τυπικές μεταβλητές. Δημιουργεί ένα αντίγραφο για κάθε block που εκτελείται στην CPU. Κάθε νήμα σε αυτό το block μοιράζεται την μνήμη, αλλά τα νήματα δεν μπορούν να προσπελάσουν και να επεξεργαστούν το αντίγραφο της μεταβλητής που φαίνεται στα άλλα blocks.

Αυτό παρέχει ένα καλό τρόπο με τον οποίο τα νήματα μέσα σε ένα block μπορούν να επικοινωνούν και να συνεργάζονται στους υπολογισμούς. Επιπλέον, τα buffers κοινής μνήμης βρίσκονται πάνω στην GPU, με αυτόν τον τρόπο, η καθυστέρηση στην προσπέλαση της κοινής μνήμης είναι πολύ μικρότερη από ότι στα τυπικά buffers, καθιστώντας την κοινή μνήμη πολύ αποδοτική.[6]

Η επικοινωνία μεταξύ των νημάτων, αν και πολύ ενδιαφέρουσα, χρειάζεται έναν μηχανισμό για τον συγχρονισμό της. Για παράδειγμα, αν το νήμα A γράφει μια τιμή στην κοινή μνήμη και θέλουμε το νήμα B να κάνει κάτι με αυτήν την τιμή, δεν μπορούμε να ξεκινήσουμε το νήμα B έως ότου γνωρίζουμε ότι η εγγραφή από το νήμα A ολοκληρώθηκε. Χωρίς τον συγχρονισμό, θα είχαμε έναν αγώνα δρόμου όπου το σωστό αποτέλεσμα της εκτέλεσης θα εξαρτάται από μη ντετερμινιστικά χαρακτηριστικά του υλικού.

Σταθερή μνήμη

Έχουμε αναλύσει το πως οι μοντέρνες GPUs είναι εφοδιασμένες με τεράστιες δυνατότητες υπολογιστικής δύναμης. Το υπολογιστικό πλεονέκτημα που έχουν οι μονάδες επεξεργασίας γραφικών βοήθησε στην ανάπτυξη του προγραμματισμού γενικού σκοπού. Με εκατοντάδες αριθμητικές μονάδες στην GPU, συνήθως ο περιορισμός δεν είναι η αριθμητική απόδοση, αλλά το εύρος ζώνης της μνήμης. Υπάρχουν τόσες πολλές ALUs στους επεξεργαστές γραφικών, όπου πολλές φορές δεν προλαβαίνουμε να μεταφέρουμε τα δεδομένα αρκετά γρήγορα ώστε να διατηρήσουμε έναν υψηλό ρυθμό υπολογισμού.

Η γλώσσα CUDA παρέχει άλλον έναν τύπο μνήμης γνωστή ως σταθερή μνήμη. Όπως φαίνεται από το όνομα, χρησιμοποιούμε την σταθερή μνήμη για δεδομένα που δεν αλλάζουν κατά την διάρκεια μιας εκτέλεσης πυρήνα. Το υλικό NVIDIA παρέχει 64KB σταθερής μνήμης που χρησιμοποιεί με διαφορετικό τρόπο από ότι την γενική μνήμη. Σε μερικές περιπτώσεις, η

χρήση της σταθερής μνήμης αντί της γενικής μνήμης μειώνει το εύρος ζώνης της μνήμης.

Δηλώνοντας την μνήμη σαν σταθερή, περιορίζουμε την χρήση της σε ανάγνωσης μόνο. Λόγω αυτού του περιορισμού, περιμένουμε να κερδίσουμε κάτι από αυτήν την διαδικασία. Η χρήση σταθερής μνήμης μπορεί να μας διαφυλάξει εύρος μνήμης σε σχέση με την ανάγνωση των δεδομένων από την γενική μνήμη. Υπάρχουν δύο λόγοι γιατί η ανάγνωση από την σταθερή μνήμη των 64KB μπορεί να διαφυλάξει εύρος μνήμης:

- Μια ανάγνωση από την σταθερή μνήμη μπορεί να αναμεταδοθεί σε κοντινά νήματα, σώζοντας μας έως και 15 αναγνώσεις.
- Η σταθερή μνήμη είναι cached, έτσι επόμενες αναγνώσεις της ίδιας διεύθυνσης δεν θα δημιουργήσουν επιπλέον κίνηση στην μνήμη.

Για να εξηγήσουμε αυτήν την δήλωση και το τι είναι τα κοντινά νήματα, σκεφτόμαστε την περίπτωση της ύφανσης. Στην ύφανση, το στημόνι αναφέρεται σε ένα σύνολο από νήματα, που υφαίνονται μαζί σε ένα ύφασμα. Στην αρχιτεκτονική CUDA, το στημόνι αναφέρεται σε μια συλλογή από 32 νήματα τα οποία υφαίνονται μεταξύ τους, και εκτελούνται αμφίδρομα. Σε κάθε γραμμή του κώδικα, κάθε νήμα από το στημόνι εκτελεί την ίδια διαδικασία, σε διαφορετικά δεδομένα. Όσον αφορά τον χειρισμό της σταθερής μνήμης, το υλικό NVIDIA μπορεί να αναμεταδώσει μια απλή ανάγνωση σε μια ομάδα από 16 νήματα: τα μισά από τα 32 νήματα που βρίσκονται στο στημόνι. Αν κάθε νήμα ζητάει δεδομένα από την ίδια διεύθυνση της σταθερής μνήμης, η μονάδα επεξεργασίας γραφικών θα εκτελέσει μόνο ένα αίτημα ανάγνωσης και θα αναμεταδώσει τα δεδομένα σε όλα τα νήματα. Αν η ανάγνωση γίνεται από την σταθερή μνήμη για μεγάλο όγκο πληροφοριών, θα δημιουργηθεί κίνηση που αντιστοιχεί μόνο στο 1/16 της κίνησης μνήμης που θα χρειαζόταν για την γενική μνήμη.

Τα πλεονεκτήματα όμως δεν σταματάνε εκεί. Επειδή έχουμε αποφασίσει να αφήσουμε άθικτη την μνήμη, το υλικό μπορεί να αποθηκεύσει τα σταθερά δεδομένα στην GPU. Έτσι μετά από την πρώτη ανάγνωση από την διεύθυνση της σταθερής μνήμης, όλες οι επόμενες αναγνώσεις δεν θα δημιουργήσουν επιπλέον κίνηση στην μνήμη. Δυστυχώς, μπορεί να υπάρξουν στιγμές που η απόδοση να μειώνεται λόγω της σταθερής μνήμης. Η αναμετάδοση μέρους από το στημόνι είναι ένα δίκοπο μαχαίρι. Αν και μπορεί να επιταχύνει την απόδοση όταν τα 16 νήματα διαβάζουν την ίδια διεύθυνση, μπορεί αντίστοιχα να μειώσει την απόδοση όταν τα 16 νήματα διαβάζουν διαφορετικές διευθύνσεις. Για παράδειγμα, αν τα 16 νήματα χρειάζονται διαφορετικά δεδομένα από την σταθερή μνήμη, οι 16 αναγνώσεις θα γίνονταν με την σειρά, και θα χρειαζόταν 16 φορές περισσότερο χρόνο για να εκτελέσουν την εντολή. Αν διάβαζαν από την γενική μνήμη, η εντολή θα δινόταν άμεσα. Σε αυτήν την περίπτωση, η ανάγνωση από την σταθερή μνήμη θα ήταν πιο αργή από το να γινόταν ανάγνωση της γενικής μνήμης.[7]

1.2.5 CUDA streams

Εξηγήσαμε το πως ο παράλληλος προγραμματισμός δεδομένων σε μια GPU μπορεί να δώσει εντυπωσιακά αποτελέσματα σε σχέση με την εκτέλεση σε CPU. Όμως υπάρχει ακόμα ένας τύπος παράλληλου υπολογισμού

που μπορούμε να εκμεταλλευτούμε σε μια μονάδα επεξεργασίας γραφικών NVIDIA. Ο παραλληλισμός αυτός μοιάζει με αυτόν που συμβαίνει στα πολυ-νηματικά προγράμματα CPU. Αντί να εκτελούμε την ίδια διαδικασία σε πολλά στοιχεία δεδομένων όπως στον παραλληλισμό δεδομένων, ο παραλληλισμός έργων περιλαμβάνει παράλληλη εκτέλεση δύο ή περισσότερων έργων.

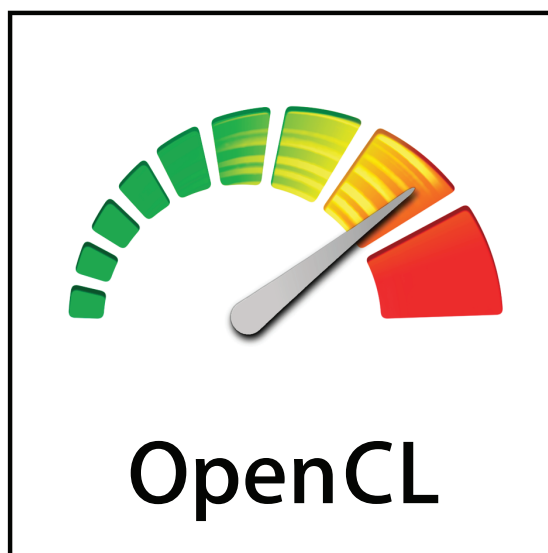
Σαν έργο μπορούμε να θεωρήσουμε μεγάλο αριθμό πραγμάτων. Για παράδειγμα, μια εφαρμογή μπορεί να εκτελεί δύο έργα: επανασχεδίαση του γραφικού περιβάλλοντος με ένα νήμα, και μεταφόρτωση μιας αναβάθμισης μέσω δικτύου με κάποιο άλλο νήμα. Αυτά τα έργα εκτελούνται παράλληλα, και ας μην έχουν τίποτε κοινό. Αν και ο παραλληλισμός έργων δεν είναι τόσο ευέλικτος όσο στις CPUs, συνεχίζει να μας προσφέρει ευκαιρίες για να αποκτήσουμε περισσότερη ταχύτητα από τις βασισμένες σε GPU εφαρμογές μας.

Τα CUDA streams, παίζουν μεγάλο ρόλο στην επιτάχυνση των εφαρμογών μας. Ένα CUDA stream αντιπροσωπεύει μια λίστα από GPU διεργασίες που θα εκτελεσθούν με συγκεκριμένη σειρά. Οι διεργασίες μπορούν να περιλαμβάνουν εκτελέσεις πυρήνων, αντιγραφές μνήμης, και εκκινήσεις/τερματισμούς συμβάντων ενός stream. Η σειρά εισαγωγής των διεργασιών ορίζει την σειρά εκτέλεσής τους, με ευκαιρίες για αυτά τα έργα να εκτελεστούν παράλληλα.

1.3 OpenCL

1.3.1 Εισαγωγή

Το πρώτο GPGPU framework δημιουργήθηκε από την NVIDIA και ήταν το CUDA. Το CUDA παρείχε στους χρήστες ένα προγραμματιστικό περιβάλλον σε C like γλώσσα για την GPU. Όμως ήταν κλειστού κώδικα και μπορεί να τρέχει μόνο σε NVIDIA κάρτες γραφικών. Λόγω της μεγάλης δημοτικότητας του CUDA, η ανάγκη για ένα ανοιχτό πρότυπο αρχιτεκτονικής που θα υποστηρίζει διάφορα είδη συσκευών από διάφορους κατασκευαστές γινόταν όλο και πιο σημαντική. Έτσι τον Ιούνιο του 2008 το Khronos Group δημιούργησε το OpenCL 1.0. Αρκετοί κατασκευαστές σταδιακά παρείχαν εργαλεία για προγραμματισμό σε OpenCL συμπεριλαμβανομένων των: Nvidia OpenCL Drivers and Tools, AMD APP SDK, Intel SDK for OpenCL applications, IBM Server with OpenCL development Kit, κ.α. Σήμερα το OpenCL επιτρέπει πολυπύρηνιο προγραμματισμό, προγραμματισμό GPU, κ.α [opencl-1]



Σχήμα 1.2: Λογότυπο OpenCL

Το OpenCL είναι ένα πλαίσιο για κατασκευή εφαρμογών που εκτελούνται σε ετερογενή συστήματα που αποτελούνται από κεντρικές μονάδες επεξεργασίας (CPU), μονάδες επεξεργασίας γραφικών (GPU), επεξεργαστές ψηφιακών σημάτων (DPS), συστοιχίες προγραμματιζόμενων θυρίδων (FPGA), και άλλους επεξεργαστές. Το OpenCL περιέχει μια γλώσσα, υποσύνολο του ISO C99 με επεκτάσεις, για τον προγραμματισμό αυτών των συσκευών, προγραμματιστικές διεπαφές εφαρμογών (API) για τον έλεγχο της πλατφόρμας και την εκτέλεση προγραμμάτων στις υπολογιστικές συσκευές. Το OpenCL παρέχει παράλληλο υπολογισμό χρησιμοποιώντας παραλληλισμό διεργασιών και δεδομένων.

Αποτελεί το πρώτο ανοιχτό, ελεύθερο από τέλη αδειών πρότυπο για cross-platform, παράλληλο προγραμματισμό μοντέρνων επεξεργαστών, που χρησιμοποιούνται συνήθως σε προσωπικούς υπολογιστές, διακομιστές, και φο-

ρητές/ενσωματωμένες συσκευές. Το OpenCL (Open Computing Language) βελτιώνει αισθητά την ταχύτητα και την απόκριση μεγάλου εύρους εφαρμογών σε διάφορες κατηγορίες αγορών από παιχνίδια και ψυχαγωγία, μέχρι επιστημονικές εφαρμογές και εφαρμογές υγείας. Συντηρείται από το μή-κερδοσκοπικό τεχνολογικό συνεταιρισμό Khronos Group. Έχει υιοθετηθεί από πολλές μεγάλες εταιρίες όπως η Apple, Intel, Qualcomm, AMD, Nvidia, Samsung, ARM Holdings.

1.3.2 Επισκόπηση

Το OpenCL ορίζει μια διεπαφή προγραμματισμού εφαρμογών με την οποία επιτρέπει στα προγράμματα που τρέχουν στον οικοδεσπότη να εκτελέσουν πυρήνες στην συσκευή υπολογισμού, και να διαχειριστούν την μνήμη της συσκευής, που είναι ξεχωριστή από την μνήμη του οικοδεσπότη. Τα προγράμματα του OpenCL είναι σχεδιασμένα ώστε να μεταγλωττίζονται την ώρα της εκτέλεσης, και με αυτόν τον τρόπο γίνεται δυνατό να εκτελεστούν σε διάφορες συσκευές. Το πρότυπο του OpenCL ορίζει διεπαφές προγραμματισμού για γλώσσα C και C++. Διεπαφές επίσης υπάρχουν και για άλλες γλώσσες, όπως Python, Julia, και Java.[8] Μια εφαρμογή του OpenCL προτύπου αποτελείται από μια βιβλιοθήκη που υλοποιεί την διεπαφή για C και C++, και έναν μεταγλωττιστή OpenCL για τις συσκευές υπολογισμού.

Ιεραρχία μνήμης

Το OpenCL ορίζει ιεραρχία τεσσάρων επιπέδων για την μνήμη των συσκευών υπολογισμού:

- Καθολική μνήμη: διαμοιράζεται σε όλες τις συσκευές υπολογισμού, αλλά έχει μεγάλη καθυστέρηση απόκρισης
- Μνήμη προσπέλασης: μικρότερη, χαμηλή καθυστέρηση απόκρισης, εγγράφημη από την κεντρική μονάδα επεξεργασίας του οικοδεσπότη, αλλά όχι των συσκευών υπολογισμού.
- Τοπική μνήμη: διαμοιράζεται σε πολλά στοιχεία υπολογισμού μιας συσκευής
- Ιδιωτική μνήμη στοιχείου (καταχωρητές)

Δεν είναι απαραίτητο για όλες τις συσκευές να υλοποιήσουν την ιεραρχία της μνήμης στο υλικό. Η συνέπεια στα διάφορα επίπεδα της ιεραρχίας είναι χαλαρή, και επιβάλλεται μόνο από κατηγορηματικά στοιχεία συγχρονισμού, όπως τα εμπόδια.

1.3.3 Ιστορία

Το OpenCL δημιουργήθηκε αρχικά από την Apple Inc., η οποία κατέχει τα πνευματικά δικαιώματα, και εξευγενίστηκε σε αρχική πρόταση σε συνεργασία με τεχνικές ομάδες της AMD, IBM, Qualcomm, Intel, και Nvidia. Η Apple καταχώρησε την πρόταση στο Khronos Group, και τον Ιούνιο του 2008 διαμορφώθηκε το Khronos Compute Working Group με αντιπρόσωπους από εταιρίες επεξεργαστών, μονάδων υλικού γραφικών, ενσωματωμένων επεξεργαστών, και λογισμικού. Αυτό το group εργάστηκε για 5 μήνες ώστε

να φέρει σε πέρας τον πρώτο προσδιορισμό για το OpenCL 1.0, ο οποίος κυκλοφόρησε τον Δεκέμβριο του 2008.

OpenCL 1.0	Η AMD, αν και αρχικά εργαζόταν πάνω στο πρότυπο Close to Metal, αποφάσισε να στραφεί και να υποστηρίξει το OpenCL. Η Nvidia ανακοίνωσε πλήρης υποστήριξη στην εργαλειοθήκη υπολογισμού GPU. Το 2009, η IBM κυκλοφόρησε την πρώτη έκδοση του μεταγλωττιστή της με υποστήριξη για OpenCL
OpenCL 1.1	Το OpenCL 1.1 επικυρώθηκε από το Khronos Group τον Ιούνιο του 2010, και προσθέτει σημαντικές λειτουργίες για βελτιωμένη ευελιξία παράλληλου προγραμματισμού, λειτουργικότητα, και επιδόσεις.
OpenCL 1.2	Το OpenCL 1.2 ανακοινώθηκε τον Νοέμβριο του 2011 από το Khronos Group, το οποίο προσθέτει αρκετές λειτουργίες σε σχέση με τις προηγούμενες εκδόσεις όσον αφορά τις επιδόσεις και χαρακτηριστικά για παράλληλο προγραμματισμό.
OpenCL 2.0	Το OpenCL 2.0 επικυρώθηκε και κυκλοφόρησε τον Νοέμβριο του 2013 από το Khronos Group και αποτελεί την τελευταία έκδοση του OpenCL.

1.3.4 Στόχοι

Ο στόχος του OpenCL είναι να κάνει ορισμένους τύπους παράλληλου προγραμματισμού πιο εύκολους, και να παρέχει ανεξαρτήτου κατασκευαστή παράλληλη εκτέλεση κώδικα μέσω επιτάχυνσης υλικού. Το OpenCL είναι το πρώτο ανοιχτό, ελεύθερο πρότυπο για παράλληλο προγραμματισμό γενικού σκοπού ετερογενών συστημάτων. Παρέχει ένα προγραμματιστικό περιβάλλον που βοηθάει τους προγραμματιστές να γράψουν αποδοτικό, φορητό κώδικα για συστήματα υψηλής απόδοσης, προσωπικούς υπολογιστές, και κινητές συσκευές χρησιμοποιώντας ένα μείγμα πολυπύρηνων CPUs, GPUs, και DSPs.

Το OpenCL παρέχει στους προγραμματιστές ένα κοινό σετ εργαλείων εύκολης χρήσης[9], ώστε αυτοί να εκμεταλλευτούν οποιαδήποτε συσκευή που περιέχει οδηγό OpenCL για την εκτέλεση παράλληλου κώδικα. Το OpenCL framework ορίζει μια γλώσσα C like για την δημιουργία των πυρήνων, και ένα σετ από APIs για την δημιουργία και την διαχείριση αυτών των πυρήνων. Οι πυρήνες είναι διαδικασίες οι οποίες μπορούν να εκτελούνται σε διαφορετικές συσκευές. Οι πυρήνες μεταγλωττίζονται από έναν μεταγλωττιστή runtime, μέσω κατάλληλου προγράμματος. Αυτό επιτρέπει στα προγράμματα να εκμεταλλεύονται όλες τις συσκευές ενός συστήματος με ένα σετ φορητών υπολογιστικών πυρήνων.

1.3.5 Μοντέλο εκτέλεσης

Τα κύρια μέρη εκτέλεσης ενός προγράμματος OpenCL είναι ο πυρήνας και το πρόγραμμα ξενιστή. Οι πυρήνες εκτελούνται στην συσκευή OpenCL και το πρόγραμμα ξενιστή, στον υπολογιστή που εκτελείται το πρόγραμμα. Ο σκοπός του προγράμματος ξενιστή είναι να δημιουργήσει και να ζητήσει την πλατφόρμα και τις ιδιότητες της συσκευής, να ορίσει το περιεχόμενο, να

κατασκευάσει τον πυρήνα, και να διαχειριστεί την εκτέλεση των πυρήνων. Όταν καταχωρηθεί ο πυρήνας από τον ξενιστή στην συσκευή, δημιουργείται ένα N διαστάσεων χώρος ευρετηρίου, με το N να είναι από 1 έως 3. Κάθε περιστατικό πυρήνα δημιουργείται στις συντεταγμένες του χώρου ευρετηρίου. Αυτό το περιστατικό ονομάζεται αντικείμενο εργασίας και ο χώρος ευρετηρίου καλείται NDRange.[10]

1.3.6 WebCL

Με την δημοτικότητα των εφαρμογών web να αυξάνεται, υπάρχει ανάγκη για αύξηση της απόδοσης παράλληλης υπολογιστικής επεξεργασίας για να επιταχύνουμε τις διεργασίες εντατικού υπολογισμού σε εφαρμογές ιστού. Αυτές οι εφαρμογές περιλαμβάνουν για παράδειγμα, παρουσίαση απεικόνισης δεδομένων, επεξεργασία εικόνας και βίντεο, παιχνίδια τριών διαστάσεων, υπολογιστική φωτογραφία, κρυπτογραφία. Παρέχοντας στους προγραμματιστές με ένα πρότυπο Javascript API και μια γλώσσα φορητού προγραμματισμού, το WebCL επιτρέπει παράλληλο υπολογισμό σε ετερογενή πολυ-πύρηνια συστήματα σε μια σωρεία συσκευών, συμπεριλαμβανομένου φορητών, σταθερών, και διακομιστών.

Το WebCL 1.0 ορίζει ένα Javascript binding στο πρότυπο OpenCL για ετερογενή παράλληλο υπολογισμό, ενώ επιτρέπει σε εφαρμογές ιστού να εκμεταλλευτούν τις δυνατότητες της GPU και τον παράλληλο υπολογισμό πολυπύρηνων CPU, μέσα από έναν Web Browser, ενεργοποιώντας σημαντική επιτάχυνση των εφαρμογών όπως επεξεργασία βίντεο και εικόνας, και ανώτερης εξομοίωσης φυσικής για παιχνίδια WebGL. Το WebCL έχει αναπτυχθεί σε στενή συνεργασία με την κοινότητα του web, και παρέχει την δυνατότητα να επεκταθούν οι δυνατότητες των HTML5 browsers ώστε να επιταχύνουν τις εφαρμογές υψηλών απαιτήσεων υπολογισμού και πλούσιου οπτικού υπολογισμού. Το WebCL έχει σχεδιαστεί και συντηρείται από το μη κερδοσκοπικό ίδρυμα Khronos Group. Οι ολοκληρωμένες προδιαγραφές της πρώτης έκδοσης του WebCL ανακοινώθηκαν τον Μάρτιο του 2014.[11]

Ανάλογα την υλοποίηση, Το WebCL επιτρέπει στις διεργασίες να εκτελούνται ταυτόχρονα με την Javascript. Συγκεκριμένα, είναι δυνατόν για την εφαρμογή να επεξεργαστεί ένα buffer καθώς αντιγράφεται από ένα αντικείμενο WebCLMemoryObject. Για να αποτρέψουμε την φθορά των δεδομένων, οι εφαρμογές θα πρέπει να μην επεξεργάζονται τα buffers που έχουν σημαδευτεί για ασύγχρονη ανάγνωση/εγγραφή, έως ότου η σχετική ουρά εντολών WebCL έχει τελειώσει την εκτέλεση της.



Σχήμα 1.3: Λογότυπο WebCL

- Khronos Launching new WebCL initiative

- Ανακοινώθηκε τον Μάρτιο του 2011
- API definitions already underway
- Javascript binding για OpenCL
 - Η ασφάλεια πρώτη προτεραιότητα
- Πολλές περιπτώσεις χρήσης
 - Μηχανές φυσικής για συμπλήρωση του WebGL
 - Επεξεργασία εικόνας και βίντεο σε browser
- Πολύ στενή σχέση με το πρότυπο OpenCL
 - Maximum flexibility
 - Foundation for higher-level middleware

Υλοποιήσεις

Αυτήν την στιγμή, μόνο μια δοκιμαστική έκδοση του Chromium browser υποστηρίζει το WebCL, γιατί η τεχνολογία είναι σχετικά καινούρια. Όμως το webCL μπορεί να εκτελεστεί και ως ένα πρόσθετο. Για παράδειγμα, η Nokia έχει δημιουργήσει ένα πρόσθετο WebCL για τον Mozilla Firefox. Η Mozilla έχει ανακοινώσει ότι δεν θα υποστηρίξει το WebCL, λόγω των Compute Shader του OpenGL ES 3.1. Σαν παράδειγμα αναφέρουμε υλοποιήσεις που προσπαθούν να ακολουθήσουν τις προδιαγραφές WebCL 1.0

- Samsung (WebKit) - <https://github.com/SRA-SiliconValley/webkit-webcl>
- Nokia (Firefox) - <http://webcl.nokiaresearch.com/>
- Motorola Mobility node-webcl (node.js) - <https://github.com/Motorola-Mobility/node-webcl>

Ασφάλεια

Το WebCL έχει δημιουργηθεί δίνοντας μεγάλη έμφαση στην ασφάλεια. Μερικά από τα φαινόμενα που μπορεί να προκύψουν κατά την μεταγλώττιση ή και την εκτέλεση ενός πυρήνα WebCL είναι τα παρακάτω

- Πρόσβαση OOB(Out of Bounds) - Οι πυρήνες του WebCL πρέπει να μην επιτρέπουν πρόσβαση σε κρίσιμα μέρη της μνήμης, χωρίς να γίνονται διακρίσεις ανάλογα με τον τύπο της μεταβλητής.(private, global, constant). Αν ανιχνευθεί την ώρα της μεταγλώττισης, η πρόσβαση OOB πρέπει να παράγει ένα λάθος μεταγλώττισης. Την ώρα της εκτέλεσης, η προσπέλαση OOB πρέπει να επιστρέφει μηδέν, και οι εγγραφές να αγνοθούν. Για λόγους ελέγχου του ορίου, η υλοποίηση μπορεί να μεταχειρίζεται τις μεταβλητές σαν ένα συνεχόμενο block μνήμης. Για τον έλεγχο των προσπελάσεων OOB, το Khronos Group ανέπτυξε τον WebCL Validator, ο οποίος αναγκάζει την αρχικοποίηση τοπικής μνήμης.
- Αρχικοποίηση μνήμης - Για να σιγουρέψουμε ότι οι εφαρμογές δεν μπορούν να προσπελάσουν παλαιότερα δεδομένα που έχουν μείνει στην μνήμη απο προηγούμενες εφαρμογές, η υλοποίηση WebCL πρέπει να αρχικοποιεί όλα τα buffers και μεταβλητές στο μηδέν πριν επιτρέψει την πρόσβαση στην εφαρμογή μας. Αυτή η ανάγκη υπάρχει ανεξάρτητα από τον τύπο της μεταβλητής και ανεξάρτητα την συσκευή στην οποία

εκτελείται ο κώδικας. Όπου είναι δυνατόν, το πρόσθετο OpenCL 1.2 "cl_khr_initialize_memory" επιτρέπει στις υλοποιήσεις WebCL να αρχικοποιήσουν την τοπική μνήμη αυτόματα πριν την εκτέλεση ενός πυρήνα.

- Άρνηση υπηρεσίας - Πυρήνες που εκτελούνται για μεγάλο χρονικό διάστημα ή πυρήνες εντατικών υπολογισμών (ή άλλες εντολές στην ουρά εντολών) μπορεί να προκαλέσουν αστάθεια στο σύστημα λόγω της υπερκατανάλωσης πόρων. Δεν είναι εύκολο να γίνει έλεγχος του συγκεκριμένου προβλήματος στο επίπεδο του WebCL. Οι απαραίτητοι μηχανισμοί, όπως μετρητές watchdog και προληπτικός πολυ-νηματικός προγραμματισμός, πρέπει να παρέχονται από τον οδηγό OpenCL και το λειτουργικό σύστημα. Σε συστήματα όπου οι παραπάνω μηχανισμοί είναι διαθέσιμοι, οι υλοποιήσεις WebCL πρέπει να τους χρησιμοποιούν για να:

1. Ανιχνεύουν προβληματικούς πυρήνες και άλλες εντολές. Μια εντολή θεωρείται προβληματική όταν εκτελείται για πολύ μεγάλο χρονικό διάστημα, ή καταναλώνει υπερβολικά πολλούς πόρους συστήματος
2. Τερματίζει τα περιεχόμενα που σχετίζονται με τις προβληματικές εντολές, πριν αυτές καταστήσουν την συσκευή OpenCL απαθής, και την αναγκάσουν να χρειαστεί επανεκκίνηση.

Όπου αυτό είναι δυνατόν, το πρόσθετο OpenCL 1.2 "cl_kh_terminate_context" μπορεί να χρησιμοποιηθεί για γρήγορο τερματισμό ενός περιεχομένου, αν για παράδειγμα κάποιος πυρήνας εκτελείται για πολύ μεγάλο χρονικό διάστημα ή ένα από τα προγράμματα τερματιστεί λόγω εξαιρέσεων.

1.4 Compute Shaders

Τα shaders υπολογισμού είναι μια κατάσταση shader που χρησιμοποιείται σχεδόν αποκλειστικά για υπολογισμούς αυθαίρετης πληροφορίας. Αν και μπορεί να χρησιμοποιηθεί για απόδοση, συνήθως χρησιμοποιείται για διεργασίες που δεν σχετίζονται άμεσα με σχεδιασμό τριγώνων και pixel.[12]

1.4.1 Shaders

Εισαγωγή

Στον τομέα των γραφικών υπολογιστών, ένα shader είναι ένα πρόγραμμα που χρησιμοποιείται για να εκτελέσει την λεγόμενη σκίαση: την παραγωγή συγκεκριμένου επιπέδου χρώματος μέσα σε μια εικόνα, ή την παραγωγή ειδικών εφέ ή μετατροπές βίντεο. Ένας όρος που περιγράφει την σκίαση είναι "ένα πρόγραμμα που μαθαίνει τον υπολογιστή πώς να ζωγραφίσει κάτι με έναν ειδικό και μοναδικό τρόπο".

Τα shader υπολογίζουν αποδόσεις εφέ σε υλικό γραφικών, με ένα μεγάλο βαθμό ευλυγισίας. Τα περισσότερα shader είναι σχεδιασμένα για χρήση σε μονάδα επεξεργασίας γραφικών (GPU), όμως αυτό δεν είναι αποκλειστική ανάγκη. Οι γλώσσες σκίασης, χρησιμοποιούνται συνήθως για να προγραμματίσουν την γραμμή σωλήνα απόδοσης της GPU. Η θέση, η απόχρωση, ο κορεσμός, η φωτεινότητα, και η αντίθεση όλων των στοιχείων, κορυφών, ή υφών, χρησιμοποιούνται για να αποδώσουν μια τελική εικόνα που μπορούμε να επεξεργαστούμε απευθείας με χρήση αλγορίθμων ορισμένων στα shader, είτε με αλλαγές από εξωτερικές μεταβλητές που εισάγει το πρόγραμμα το οποίο καλεί τον shader. Τα shader χρησιμοποιούνται πολύ στην κινηματογραφική επεξεργασία, στις εικόνες που αποδίδονται από τον υπολογιστή, αλλά και σε παιχνίδια υπολογιστών, για να παράγουν ένα μεγάλο αριθμό από εφέ. Εκτός από τα απλά μοντέλα φωτισμού, μερικά από τα πολύπλοκα εφέ επεξεργάζονται την εικόνα και προσθέτουν blur, light bloom, volumetric lightning, normal mapping, bokeh, cel shading, posterization, bump mapping, distortion, chroma keying, edge detection, motion detection, κ.α

Η σύγχρονη χρήση των shader ξεκίνησε από την Pixar, τον Μάιο του 1988. Όσο οι μονάδες επεξεργασίας γραφικών εξελίσσονταν, οι γνωστές βιβλιοθήκες γραφικών ξεκίνησαν να υποστηρίζουν τα shader. Οι πρώτες κάρτες γραφικών υποστήριζαν μόνο pixel shader, αλλά σύντομα ακολούθησε η εισαγωγή των vertex shader όταν οι προγραμματιστές κατάλαβαν τις δυνατότητες τους. Τα shader γεωμετρίας εισήχθησαν μόλις με το Direct3D 10 και το OpenGL 3.2

Τύποι

Υπάρχουν διάφοροι τύποι shader που χρησιμοποιούνται γενικά. Ενώ οι παλιές κάρτες γραφικών είχαν ξεχωριστό τρόπο επεξεργασίας στοιχείων για κάθε τύπο shader, οι καινούριες έχουν ενωμένους shader που έχουν την δυνατότητα να εκτελούν οποιονδήποτε τύπο shader. Αυτό επιτρέπει στις κάρτες γραφικών να έχουν πιο αποδοτική χρήση της επεξεργαστικής τους δύναμης.[13]

- Vertex shaders - Μετατρέπουν κάθε θέση τρισδιάστατη στον εικονικό χώρο σε δισδιάστατη. Μπορούν να επεξεργαστούν ιδιότητες όπως η θέση, το χρώμα, και συντεταγμένες υφής, αλλά δεν μπορούν να δημιουργήσουν καινούρια vertices.
- Pixel shaders - Υπολογίζουν το χρώμα και άλλες ιδιότητες ενός τεμαχίου. Οι απλές μορφές τους αποδίδουν ένα pixel εικόνας, ενώ οι πιο πολύπλοκες μορφές αποδίδουν πολλά. Στα τρισδιάστατα γραφικά, ένας pixel shader δεν μπορεί να παράγει πολύπλοκα εφέ, γιατί επεξεργάζεται μόνο ένα τεμάχιο, χωρίς κάποια γνώση της γεωμετρίας της οθόνης. Μπορούν όμως να εφαρμοστούν σε εφέ δύο διαστάσεων και χρησιμοποιούνται για επεξεργασία υφών. Για παράδειγμα είναι ο μόνος τύπος shader που μπορεί να λειτουργήσει σαν φίλτρο για μια ροή βίντεο.
- Geometry shaders - Τα shader γεωμετρίας είναι σχετικά καινούριος τύπος shader. Μπορεί να δημιουργήσει γραφικά αρχικά στοιχεία όπως γραμμές, τρίγωνα. Τα shader γεωμετρίας εκτελούνται μετά από τα vertex shader. Τυπικές χρήσεις των shader γεωμετρίας συμπεριλαμβάνουν γεωμετρική ψηφίδωση, εξώθηση σκιώδους όγκου, και απόδοση σε χάρτη κύβου. Για παράδειγμα, σε μια καμπυλωτή γραμμή, τα δεδομένα των στοιχείων εισάγονται σαν είσοδος στον shader, και αυτός αναλαμβάνει να δημιουργήσει αυτόματα επιπλέον γραμμές που δίνουν πιο εύστοχη απεικόνιση της καμπύλης.
- Tessellation shaders - Με την κυκλοφορία του OpenGL 4.0 και του Direct3D 11, ένας καινούριος τύπος shader έχει προστεθεί που ονομάζεται shader ψηφίδωσης. Επιτρέπει στα αντικείμενα κοντά στην κάμερα να έχουν καλύτερη λεπτομέρεια, ενώ τα αντικείμενα που είναι πιο μακριά να έχουν μικρότερη λεπτομέρεια αλλά να μοιάζουν όμοια στην ποιότητα.
- Compute shaders - Είναι ένας τύπος shader που χρησιμοποιείται αποκλειστικά για υπολογισμό αυθαίρετης πληροφορίας. Αν και μπορεί να χρησιμοποιηθεί για απόδοση, συνήθως χρησιμοποιείται για διεργασίες που δεν σχετίζονται άμεσα με σχεδιασμό τριγώνων και pixel.

1.4.2 Εισαγωγή

Τα shaders υπολογισμού λειτουργούν διαφορετικά από τις άλλες καταστάσεις shader. Όλες οι καταστάσεις shader έχουν προκαθορισμένου τύπου τιμές εισόδου, μερικές ενσωματωμένες και μερικές καθορισμένες από τον χρήστη. Η συχνότητα στην οποία εκτελείται μια κατάσταση shader εξαρτάται από την φύση της κατάστασης. Για παράδειγμα, τα shader κορυφών εκτελούνται μια φορά για κάθε κορυφή.

Τα shader υπολογισμού λειτουργούν πολύ διαφορετικά. Ο "χώρος" στον οποίο ένα shader υπολογισμού λειτουργεί είναι αφηρημένος. Είναι στην κρίση του κάθε shader υπολογισμού να αποφασίσει τι σημαίνει αυτός ο "χώρος". Ο αριθμός των εκτελέσεων των shader υπολογισμού ορίζεται από την διεργασία που χρησιμοποιείται για να εκτελεστεί η υπολογιστική λειτουργία. Πιο σημαντικό από όλα, τα shader υπολογισμού δεν έχουν εισόδους καθορισμένες από τον χρήστη και ούτε καμία έξοδο. Οι ενσωματωμένες εισοδοί ορίζουν μόνο το πού στον "χώρο" της εκτέλεσης βρίσκεται ένας συγκεκριμένος shader υπολογισμού.

Έτσι, αν κάποιος shader υπολογισμού πρέπει να πάρει κάποιες τιμές σαν

είσοδο, είναι στην ευθύνη του shader να αποκτήσει τα δεδομένα, μέσω πρόσβασης υφών, αυθαίρετης φόρτωσης εικόνας, ή άλλες μορφές διεπαφής. Παρομοίως, αν ένας shader υπολογισμού υπολογίζει κάτι, θα πρέπει να το αποθηκεύσει σε μια εικόνα ή σε ένα block αποθήκευσης shader.

1.4.3 Χώρος υπολογισμού

Ο χώρος στον οποίο λειτουργεί ένα shader υπολογισμού είναι αφηρημένος. Υπάρχει η έννοια της ομάδας εργασίας. Είναι ο μικρότερος αριθμός από λειτουργίες υπολογισμού τις οποίες μπορεί να εκτελέσει ο χρήστης. Ο αριθμός των ομάδων εργασίας με τον οποίο μια λειτουργία υπολογισμού εκτελείται, ορίζεται από τον χρήστη όταν επικαλείται την λειτουργία υπολογισμού. Ο χώρος αυτών των ομάδων είναι τρισδιάστατος, οπότε έχει ένα αριθμό από ομάδες "X", "Y", "Z". Κάθε ένας από αυτούς μπορεί να είναι 1, οπότε είναι δυνατή η εκτέλεση λειτουργιών υπολογισμού δυο ή και μίας διάστασης αντί για τρεις διαστάσεις. Αυτό είναι χρήσιμο για την επεξεργασία δεδομένων εικόνας ή γραμμικών πινάκων ενός συστήματος.

Η κάθε ομάδα εργασίας μπορεί να αποτελείται από πολλούς shader υπολογισμού. Αυτό ονομάζεται τοπικό μέγεθος της ομάδας εργασίας. Κάθε shader υπολογισμού έχει τρισδιάστατο τοπικό μέγεθος (το οποίο μπορεί να είναι 1 επιτρέποντας δισδιάστατη ή μονοδιάστατη επεξεργασία). Αυτό ορίζει τον αριθμό των επικλήσεων ενός shader που θα εκτελεστούν σε κάθε ομάδα εργασίας. Για παράδειγμα, αν το τοπικό μέγεθος ενός shader υπολογισμού είναι (128, 1, 1) και εκτελεστεί με έναν αριθμό ομάδων εργασίας (16, 8, 64) τότε θα έχουμε 1,048,576 ξεχωριστές επικλήσεις shader. Κάθε επίκληση θα έχει ένα σετ από εισόδους που αναγνωρίζουν μοναδικά την κάθε επίκληση. Αυτός ο διαχωρισμός είναι χρήσιμος για διάφορες μορφές συμπίεσης και αποσυμπίεσης εικόνας. Το τοπικό μέγεθος θα είναι το μέγεθος ενός block δεδομένων εικόνας (8x8 για παράδειγμα), ενώ ο αριθμός των ομάδων θα είναι το μέγεθος της εικόνας διαιρούμενο με το μέγεθος του block. Κάθε block κατεργάζεται σαν μια μοναδική ομάδα εργασίας.

1.4.4 Υλοποίηση OpenGL

Αποστολή

Ένα αντικείμενο προγράμματος μπορεί να έχει shader υπολογισμού μέσα του. Ο shader υπολογισμού συνδέεται με καταστάσεις shader μέσω κάποιων λειτουργιών απόδοσης. Υπάρχουν δύο λειτουργίες για να ξεκινήσουν οι διαδικασίες υπολογισμού. Χρησιμοποιούν οποιονδήποτε shader υπολογισμού είναι ενεργός. Οι λειτουργίες είναι οι εξής:

- `void glDispatchCompute(GLuint num_groups_x, GLuint num_groups_y, GLuint num_groups_z);` - Οι παράμετροι `num_groups_*` ορίζουν τον αριθμό των ομάδων εργασίας, σε τρεις διαστάσεις. Αυτοί οι αριθμοί δεν μπορούν να είναι μηδέν. Υπάρχουν όρια στον αριθμό των ομάδων εργασίας που μπορούν να αποσταλούν.
- `void glDispatchComputeIndirect(GLintptr indirect);` - Η παράμετρος `indirect` είναι το αντιστάθμισμα του buffer `GL_DISPATCH_INDIRECT_BUFFER`. Ισχύουν τα ίδια όρια του αριθμού ομάδων εργασίας, όμως η αποστολή

indirect παρακάμπτει τον έλεγχο λαθών του OpenGL. Έτσι, η αποστολή με εκτός ορίων μεγέθους ομάδας εργασίας, μπορεί να προκαλέσει προβλήματα ακόμα και πάγωμα του συστήματος.

Είδοδοι

Τα shader υπολογισμού δεν μπορούν να έχουν μεταβλητές καθορισμένες απο τον χρήστη. Τα shader υπολογισμού έχουν τις παρακάτω ενσωματωμένες μεταβλητές εξόδου:

- `in uvec3 gl_NumWorkGroups;` - Αυτή η μεταβλητή περιέχει τον αριθμό των ομάδων εργασίας για την λειτουργία αποστολής
- `in uvec3 gl_WorkGroupID;` - Αυτή η μεταβλητή περιέχει την ισχύουσα ομάδα εργασίας για την επίκληση του shader.
- `in uvec3 gl_LocalInvocationID;` - Αυτή η μεταβλητή περιέχει την ισχύουσα επίκληση του shader μέσα στην ομάδα εργασίας.
- `in uvec3 gl_GlobalInvocationID;` - Αυτή η μεταβλητή αναγνωρίζει μοναδικά την συγκεκριμένη επίκληση του shader υπολογισμού ανάμεσα σε όλες τις επικλήσεις της κλήσης αποστολής υπολογισμού. Είναι μια συντόμευση για τον μαθηματικό υπολογισμό $gl_WorkGroupID * gl_WorkGroupSize + gl_LocalInvocationID$;
- `in uint gl_LocalInvocationIndex;`

Τοπικό μέγεθος

Το τοπικό μέγεθος ενός shader υπολογισμού ορίζεται απο τον shader, χρησιμοποιώντας μια ειδική δήλωση εισόδου: `layout(local_size_x = X, local_size_y = Y, local_size_z = Z) in;` Αρχικά, τα τοπικά μεγέθη είναι 1, οπότε αν θέλουμε μονοδιάστατο ή δισδιάστατο χώρο ομάδων εργασίας, μπορούμε να ορίσουμε μόνο το X ή το X και το Y. Πρέπει να είναι σταθερές εκφράσεις τιμής μεγαλύτερης του 0. Οι τιμές πρέπει να ορίζονται σε σχέση με τους περιορισμούς που υπάρχουν παρακάτω. Σε αντίθετη περίπτωση προκύπτουν λάθη. Το τοπικό μέγεθος είναι διαθέσιμο στον shader σαν σταθερά, οπότε δεν χρειάζεται να την ορίζουμε εμείς.

- `const uvec3 gl_WorkGroupSize;`

Περιορισμοί

Ο αριθμός των ομάδων εργασίας που μπορούν να αποσταλούν, ορίζεται από μια ειδική σταθερά. Αυτή η σταθερά πρέπει να διαβαστεί απο την `glGetIntegeri_v`, με τιμές ανάμεσα στο κλειστό όριο [0,2].

Προσπάθεια να καλέσουμε την `glDispatchCompute` με τιμές που ξεπερνούν το όριο είναι λάθος. Προσπάθεια κλήσης της `glDispatchComputeIndirect` είναι χειρότερα, μπορεί να διακόψει την λειτουργία του προγράμματος ακόμα και να παγώσει το σύστημα. Σημείωση: ο μικρότερος αριθμός αυτών των τιμών πρέπει να είναι 65535 σε όλους τους άξονες. Αυτό δίνει αρκετό χώρο για εργασία.

Υπάρχουν όρια στο τοπικό μέγεθος επίσης. Συγκεκριμένα, υπάρχουν δύο τύποι περιορισμών.

- Ο γενικός περιορισμός των διαστάσεων τοπικού μεγέθους, σε συνδυασμό με την `GL_MAX_COMPUTE_WORK_GROUP_SIZE`, όπως και παραπάνω. Η διαφορά είναι ότι ο μικρότερος αριθμός των τιμών είναι πολύ μικρότερος. 1024 για τον X και τον Y, και μόνο 64 για τον Z.
- Ο αριθμός των επικλήσεων μέσα σε μια ομάδα εργασίας. Δηλαδή, το προϊόν των στοιχείων X,Y,Z του τοπικού μεγέθους πρέπει να είναι μικρότερο από `GL_MAX_WORK_GROUP_INVOCATIONS`. Η μικρότερη τιμή είναι 1024.

Υπάρχει ακόμα ο περιορισμός του ολικού μεγέθους αποθήκευσης για όλες τις κοινές μεταβλητές ενός shader υπολογισμού. Ορίζεται από την `GL_MAX_COMPUTE_SHARED_MEMORY_SIZE`, που αναφέρεται σε bytes. Η μικρότερη τιμή για το OpenGL είναι 32KB.

1.4.5 Υλοποίηση DirectX

Ένας shader υπολογισμού είναι μια κατάσταση shader υπολογισμού που εξαπλώνει το Microsoft Direct3D 11 πέρα από τον προγραμματισμό γραφικών. Η τεχνολογία αυτή είναι γνωστή και ως τεχνολογία DirectCompute[13]

Όπως όλα τα προγραμματιστικά shader (για παράδειγμα shader γεωμετρίας και κορυφών), ένα shader υπολογισμού είναι σχεδιασμένο να χρησιμοποιεί μια Γλώσσα Υψηλού Προγραμματισμού Shader(HLSL) για το DirectX. Η HLSL, χρησιμοποιείται για το DirectX και μας δίνει την δυνατότητα να δημιουργήσουμε C like shaders για την γραμμή σωλήνων Direct3D.

Η HLSL δημιουργήθηκε ξεκινώντας από το DirectX 9 για την κατασκευή προγραμματιζόμενων τρισδιάστατων γραμμής σωλήνα. Μας δίνει την δυνατότητα να προγραμματίσουμε την γραμμή σωλήνα με τον συνδυασμό οδηγίων assembly, οδηγίων HLSL, και δηλώσεις καθορισμένων λειτουργιών.

Ένα shader υπολογισμού προμηθεύει υψηλής ταχύτητας υπολογισμούς γενικού προγραμματισμού, και εκμεταλλεύεται τον μεγάλο αριθμό παράλληλων επεξεργασιών που βρίσκονται στην μονάδα επεξεργασίας γραφικών (GPU). Τα shader υπολογισμού προμηθεύει διαμοιρασμό μνήμης και συγχρονισμό νημάτων, για να επιτρέψει καλύτερες μεθόδους παράλληλου προγραμματισμού.

Με την κλήση των μεθόδων `ID3D11DeviceContext::Dispatch` ή `ID3D11DeviceContext::DispatchIndirect` γίνεται η εκτέλεση εντολών σε ένα shader υπολογισμού, οι οποίες μπορούν να εκτελεστούν παράλληλα σε πολλά νήματα.

1.5 PathScale Enzo

1.5.1 Εισαγωγή

Η σουίτα PathScale ENZO™ συνδιάζει το ελεύθερο πρότυπο υβριδικού πολυπύρηνου παράλληλου προγραμματισμού (HMPP), με την δυνατότητα άμεσης παραγωγής κώδικα για τις NVIDIA Tesla GPUs. Αυτή η προσέγγιση χρησιμοποιεί την δύναμη της GPU ως επιταχυντή υλικού (HWA) για να αντικαταστήσει τις παραδοσιακές μονάδες υπολογισμού SIMD. Χρησιμοποιώντας οδηγίες HMPP με το PathScale ENZO™ επιτρέπει στον προγραμματιστή να γράφει εφαρμογές ανεξάρτητου υλικού όπου ο κώδικας για το συγκεκριμένο υλικό διαχωρίζεται από τον παραδοσιακό κώδικα. Οι εφαρμογές δεν πρέπει να γραφτούν ξανά κάθε φορά που στοχεύουμε μια διαφορετική αρχιτεκτονική.

Το PathScale ENZO™ υποστηρίζει προς το παρόν την HMPP Fortran όπου, όταν συνδυαστεί με το εκτελέσιμο του ENZO™, επιτρέπει άμεση εκτέλεση εφαρμογών ENZO™ GPGPU. Οι μελλοντικές εφαρμογές του ENZO™ θα περιέχουν υποστήριξη για HMPP C,C++ και ENZO™ C++ περιγράμματα. Για να βελτιώσει το πόσο γρήγορα τρέχει η εφαρμογή μας, το ENZO™ πρώτα αναγνωρίζει τις περιοχές του πηγαίου κώδικα της εφαρμογής που είναι κατάλληλες για τον στόχο HWA. Αυτές οι περιοχές καθίστανται περιοχές ή διεργασίες που ονομάζονται "HMPP codelets", με την χρήση οδηγιών HMPP. Οι εκδόσεις των επιταχυνόμενων από υλικό των περιοχών ή codelets ορίζονται στην ίδια πηγαία γλώσσα όπως και το υπόλοιπο πρόγραμμα, π.χ η Fortran, με την χρήση του HMPP προγραμματιστικού μοντέλου.[14]



Σχήμα 1.4: PathScale ENZO™ με χρήση επιτάχυνσης GPU NVIDIA Tesla

Ο HMPP πηγαίος κώδικας αναλύεται από το ειδικό μέρος της PathScale Fortran που μεταφράζει τις HMPP οδηγίες σε κλήσεις του ENZO™ API εκτέλεσης. Το ENZO™ API εκτέλεσης έχει την ευθύνη για την διαχείριση της ταυτόχρονης εκτέλεσης όλων των περιοχών και codelets. Οι οδηγίες HMPP επιτρέπουν επίσης να ομαδοποιήσουμε codelets. Βασίζόμενοι στην προσέγγιση codelet, αυτές οι ομάδες επιτρέπουν στον προγραμματιστή να χρησιμοποιεί δεδομένα που είναι ήδη διαθέσιμα σε έναν επιταχυντή υλικού, ώστε αυτά τα δεδομένα να μπορούν να διαμοιραστούν μεταξύ διαφορετικών codelet που εκτελούνται σε διαφορετικές στιγμές, χωρίς να χρειάζονται επιπλέον μεταφορές δεδομένων μεταξύ της μνήμης του ξενιστή και του HWA.

Όσον αφορά το μοντέλο μνήμης του ENZO™, οι διευθύνσεις μνήμης διαχειρίζονται από το επίπεδο του ξενιστή και είναι διαφορετικές στο επίπεδο HWA. Η εφαρμογή και το API εκτέλεσης του ENZO™ έχουν την δικιά τους τοπική μνήμη. Το ENZO™ το αντιμετωπίζει με τρόπο διαφανή προς τον χρήστη. Το ENZO™ είναι η προγραμματιστική κόλλα μεταξύ των προγραμματιστικών περιβαλλόντων και του προγραμματισμού γενικού σκοπού.

1.6 Συμπεράσματα

Στο συγκεκριμένο κεφάλαιο μελετήσαμε τις υλοποιήσεις που υπάρχουν την δεδομένη στιγμή στην αγορά. Οι διαφορές των υλοποιήσεων είναι σημαντικές, καθώς ενώ προσπαθούν να αντιμετωπίσουν το ίδιο πρόβλημα, ακολουθούν πολλές φορές διαφορετικές μεθόδους και αυτό τους δίνει πλεονεκτήματα και μειονεκτήματα τα οποία μπορέσαμε να αναλύσουμε σε ένα βαθμό.

Λόγω του ότι αναπτύσσονται συνεχώς, έχει ενδιαφέρον να δούμε μελλοντικά ποια τεχνολογία θα καταφέρει να διακριθεί και να βοηθήσει τους προγραμματιστές να επιταχύνουν τις εφαρμογές τους, αλλά και το αν οι εταιρίες θα καταφέρουν να αντιμετωπίσουν τα προβλήματα που ήδη υπάρχουν στις υλοποιήσεις αυτές.

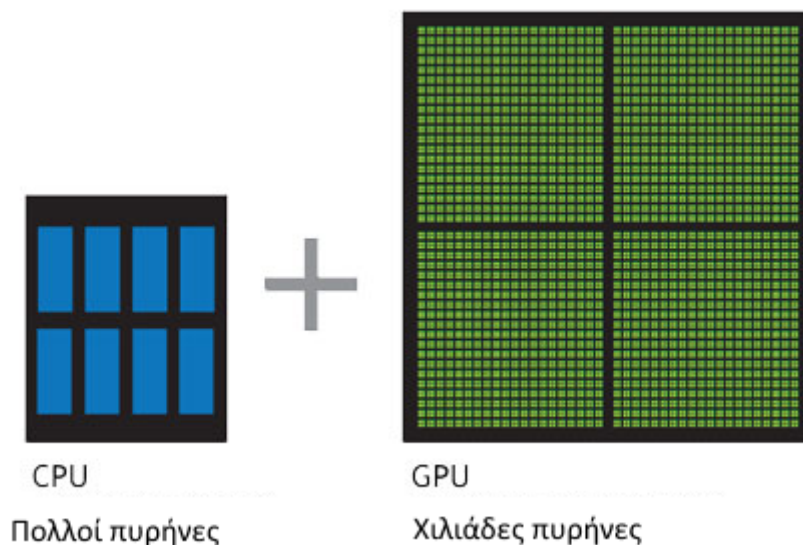
Θεωρώ ότι το OpenCL σαν τεχνολογία έχει μεγαλύτερο ενδιαφέρον από τις υπόλοιπες, καθώς είναι ελεύθερο, ανοιχτού κώδικα, ενώ στοχεύει σε χρήση από πολλές συσκευές σε αντίθεση με το CUDA που είναι φτιαγμένο αποκλειστικά για τις κάρτες γραφικών NVIDIA. Αυτό δίνει ένα επιπλέον κίνητρο σε εταιρίες και προγραμματιστές να το προωθήσουν, να το αναπτύξουν, και μέσω της συνολικής συνεργασίας της κοινότητας να δημιουργηθεί ένα πρότυπο που όπως πολλές φορές στο παρελθόν θα καταφέρει να καθιερωθεί και να αλλάξει τα δεδομένα της πληροφορικής σε παγκόσμια κλίμακα.

Στο επόμενο κεφάλαιο θα δούμε κάποια ζητήματα που έχουν ενδιαφέρον και θα ήταν λάθος να τα αγνοήσουμε τα οποία αναφέρονται στον τρόπο χρήσης της τεχνολογίας GPGPU σε σχέση με τις CPU αλλά και με χρήση πολλών μονάδων επεξεργασίας γραφικών σε συμπλέγματα.

Περισσότερα

2.1 Διαφορές CPUs και GPUs

Ένας απλός τρόπος για να κατανοήσουμε την διαφορά ανάμεσα σε μια CPU και μια GPU είναι να συγκρίνουμε πως αυτές επεξεργάζονται τις διεργασίες. Μια CPU αποτελείται από μερικούς πυρήνες βελτιστοποιημένους για σειριακή επεξεργασία σειράς, ενώ μια GPU έχει μια ογκώδης παράλληλη αρχιτεκτονική που αποτελείται από χιλιάδες μικρότερους, πιο αποδοτικούς πυρήνες σχεδιασμένους για χειρισμό πολλαπλών διαδικασιών ταυτόχρονα.



Σχήμα 2.1: Πυρήνες σε CPU και GPU αντίστοιχα

Οι GPUs έχουν δημιουργηθεί για πολύ συγκεκριμένες χρήσεις, για παράδειγμα η απεικόνιση γραφικών, πολλαπλασιασμούς πυκνού πλέγματος, απλά φίλτρα επεξεργασίας, κ.α. Είναι πολύ καλές στην διαχείριση μεγάλων καθυστερήσεων γιατί είναι σχεδιασμένες με τρόπο ώστε να ανέχονται καθυστερήσεις υφών, μια διαδικασία 1000+ κύκλων. Οι πυρήνες GPU διαθέτουν πολλά νήματα: όταν ένα νήμα εκτελέσει μια διεργασία υψηλής καθυστέρησης, (για παράδειγμα μια προσπέλαση μνήμης), αυτό το νήμα τίθεται σε αναμονή (ενώ τα υπόλοιπα νήματα συνεχίζουν να εργάζονται), έως ότου η διαδικασία τελειώσει. Αυτό επιτρέπει στις GPUs να διατηρούν τις μονάδες υπολογισμού απασχολημένες περισσότερο από ότι οι παραδοσιακοί πυρήνες.

Οι GPUs δεν αποδίδουν στον χειρισμό διακλαδώσεων, γιατί συνήθως δημιουργούν δεσμίδες από νήματα σε στημόνι, και τα αποστέλλουν στην γραμμή σωλήνα για λόγους οικονομίας εντολών προσπέλασης/αποκωδικοποίησης. Αν τα νήματα συναντήσουν κάποια διακλάδωση, μπορεί να αποκλίνουν, π.χ 2 νήματα σε ένα στημόνι 8 νημάτων μπορεί να ακολουθήσουν την διακλάδωση, ενώ τα άλλα 6 μπορεί να μην την ακολουθήσουν. Τώρα το

στημόνι έχει διαιρεθεί σε δύο στημόνια μεγέθους 2 και 6, τα οποία δεν θα εκτελούνται αποδοτικά. Το στημόνι των 2 νημάτων θα εκτελείται με 25% απόδοση, και το στημόνι των 6 νημάτων θα εκτελείται με 75% απόδοση. Μπορούμε να φανταστούμε ότι αν μια μονάδα επεξεργασίας συνεχίσει να συναντά επιπλέον διακλαδώσεις, η απόδοση τους συνεχίζει να μειώνεται. Για αυτόν τον λόγο, οι GPUs δεν είναι καλή επιλογή για τον χειρισμό διακλαδώσεων και ο κώδικας που περιέχει αυτές θα πρέπει να εκτελείται στις CPUs.

Γενικότερα, ο κώδικας που δεν θα πρέπει να εκτελείται σε GPUs είναι κώδικας με λίγο παραλληλισμό ή κώδικας με πολλές διακλαδώσεις ή συγχρονισμό, για παράδειγμα βάσεις δεδομένων, λειτουργικά συστήματα, αλγόριθμοι γραφημάτων, κ.α.

2.1.1 Διαφορές προγραμματιστικού μοντέλου

Οι πιο σημαντικές διαφορές στο προγραμματιστικό μοντέλο των GPUs είναι ότι δεν υποστηρίζουν διακοπές και εξαιρέσεις. Εκτός από αυτό, δεν υπάρχουν μεγάλες διαφορές μεταξύ της CUDA, OpenCL, και C. Επίσης όπως αναφέραμε, ο κώδικας θα πρέπει να γράφεται με όσο το δυνατόν λιγότερες διακλαδώσεις και συνεχή επικοινωνία μεταξύ των νημάτων.

Πολλά προβλήματα στον πραγματικό κόσμο έχουν πολλές διακλαδώσεις και παρατυπίες. Αλγόριθμοι γραφημάτων, λειτουργικά συστήματα, web browsers, κ.α. Ακόμα και τα γραφικά χρησιμοποιούν όλο και περισσότερο διακλαδώσεις και διαδικασίες γενικού σκοπού, αναγκάζοντας έτσι τις GPUs να γίνουν όλο και περισσότερο προγραμματιζόμενες.

2.1.2 Ανάγκη χώρου

Ο πιο σωστός τρόπος για να δούμε την επίδραση του GPGPU είναι να μελετήσουμε τους υπερ-υπολογιστές. Τα μεγάλα ερευνητικά κέντρα (Πανεπιστήμια, Εταιρίες, Κυβερνήσεις) χρειάζονται πολύ μεγάλη υπολογιστική δύναμη για να λύσουν τα πιο δύσκολα υπολογιστικά προβλήματα. Όσο μεγαλύτερο το πρόβλημα, τόσο μεγαλύτερος ο υπολογιστής που χρειάζεται. Παραδείγματα χρήσεων αυτών των υπολογιστών είναι οι διπλώσεις βιομοριακών πρωτεϊνών, η εξομοίωση εκρήξεων και πυρηνικών όπλων, αλλά και οικονομικές συναλλαγές που χρειάζονται για την διακίνηση χρήματος σε όλον τον κόσμο.

Αυτοί οι υπολογιστές συνήθως είναι μεγάλοι, τόσο σε φυσικό μέγεθος όσο και σε υπολογιστικές δυνατότητες. Για παράδειγμα ο TACC Ranger:

- Αριθμός Κόμβων: 3,936
- Αριθμός πυρήνων: 62,976
- Μέγιστη απόδοση: 579.4 TFlops
- Αριθμός ραφιών: 82

Έτσι σε 82 μονάδες έχουμε 580 TFlops υπολογιστικής απόδοσης. Σχετικά με το ίδιο παράδειγμα με χρήση NVIDIA Tesla μονάδων έχουμε

- 1 Μονάδα Tesla S1070: 4 TFlops
- 145 μονάδες: 580 TFlops

- Μονάδες σε 1 ράφι: 42
- Αριθμός ραφιών: 3.5

Δηλαδή για να έχουμε την ίδια απόδοση με την χρήση NVIDIA Tesla μονάδων θα χρειαστούμε το 1/10 του χώρου.

2.2 Συμπλέγματα GPU

2.2.1 VirtualCL

Επισκόπηση

Η πλατφόρμα VirtualCL είναι ένα περικάλυμμα για το OpenCL που επιτρέπει τις περισσότερες εφαρμογές να εκμεταλλεύονται με διαφάνεια πολλές OpenCL συσκευές σε ένα σύμπλεγμα, σαν αυτές να βρίσκονταν σε έναν προσωπικό υπολογιστή. Με την πλατφόρμα VirtualCL, οι απομακρυσμένοι κόμβοι εκτελούν διεργασίες για λογαριασμό των προγραμμάτων του ξενιστή.[**virtualcl-6**]

Το VCL είναι ευέλικτο. Οι εφαρμογές μπορούν να δημιουργήσουν περιεχόμενο OpenCL που αποτελείται από συσκευές από διάφορους κόμβους, ή πολλά περιεχόμενα, κάθε ένα αποτελούμενο από τις συσκευές διαφορετικού κόμβου. Ή οποιονδήποτε συνδυασμό από τα παραπάνω. Άλλες εφαρμογές μπορούν να διαχωριστούν σε διάφορες ανεξάρτητες διεργασίες και νήματα, κάθε μία από τις οποίες εκτελείται σε διαφορετικό σύνολο συσκευών, ενώ ταυτόχρονα χρησιμοποιούν την κοινή μνήμη του συστήματος.

Πιο εξελιγμένες εφαρμογές μπορούν να επιλέγουν τις συσκευές που θα εκτελούν τις διεργασίες τους, όμως το VCL επιτρέπει μεταβλητές συστήματος στις οποίες ορίζονται πολιτικές για τον διαμοιρασμό των συσκευών. Η αρχική ρύθμιση, είναι κάθε περιεχόμενο που δημιουργείται να περιέχει όλες τις συσκευές ενός κόμβου. Το VCL αποτελείται από τρία μέρη, την βιβλιοθήκη VCL, τον μεσίτη, και τον back-end δαίμονα. [15]

Βιβλιοθήκη

Η βιβλιοθήκη VCL είναι μια βιτρίνα συμπλέγματος για τις εφαρμογές OpenCL. Όταν συνδεθεί με τις εφαρμογές OpenCL, επιτρέπει πρόσβαση με διαφάνεια σε συσκευές OpenCL του συμπλέγματος, κρύβοντας την πραγματική τοποθεσία των συσκευών από τις εφαρμογές που τις καλούν. Η βιβλιοθήκη VCL είναι σχεδιασμένη να εκτελείται με τις περισσότερες εφαρμογές χωρίς επιπλέον παρεμβάσεις, έτσι μπορούμε να επιλέξουμε τον τρόπο που θα χρησιμοποιούνται οι συσκευές στο σύμπλεγμα, μέσω μεταβλητών συστήματος. Η βιβλιοθήκη VCL υποστηρίζει απόλυτα πολυ-νηματικό προγραμματισμό και παρέχει ασφάλεια νημάτων.[16]

Η VCL βιβλιοθήκη χρησιμοποιεί διάφορους αλγόριθμους βελτιστοποίησης. Για παράδειγμα, λόγω της καθυστέρησης δικτύου, η βιβλιοθήκη προσπαθεί να βελτιστοποιήσει την απόδοση της επικοινωνίας διατηρώντας μια ανεξάρτητη βάση δεδομένων από αντικείμενα OpenCL και εκτελεί όσες λειτουργίες είναι δυνατόν στον υπολογιστή-ξενιστή, ώστε να ελαττώσει τον αριθμό των επικοινωνιών στο ελάχιστο.[17]

Μεσίτης

Ο μεσίτης είναι μια λειτουργία δαίμονα που εκτελείται σε κάθε υπολογιστή-ξενιστή όπου οι χρήστες μπορούν να εκτελέσουν τις OpenCL εφαρμογές τους. Ο μεσίτης συνδέεται με την βιβλιοθήκη VCL μέσω υποδοχών UNIX. Η ευθύνη του περιέχει:



Σχήμα 2.2: Ένα σύμπλεγμα από υπολογιστικούς κόμβους GPU

1. Παρακολούθηση της λειτουργικότητας και διαθεσιμότητας των συσκευών στο σύμπλεγμα.
2. Αναφορά αυτών των συσκευών στις εφαρμογές που τις ζητάνε.
3. Έξυπνη κατανομή των συσκευών για τις εφαρμογές OpenCL όταν γίνεται δημιουργία περιεχομένου, για παράδειγμα η προσπάθεια να ταιριάξει ο αριθμός των συσκευών με τον αριθμό των κόμβων που έχουν σύνολο τον αριθμό που ζητήθηκε από την εφαρμογή.
4. Πιστοποίηση αυθεντικότητας, δρομολόγηση, και εξασφάλιση της ποιότητας των μηνυμάτων μεταξύ των εφαρμογών.

Back-end Δαίμονας

Η υπηρεσία του back-end δαίμονα εκτελείται σε κάθε κόμβο συμπλέγματος όπου υπάρχουν συσκευές OpenCL και υποστηρίζονται από κατάλληλους οδηγούς. Ο δαίμονας χρησιμοποιεί οποιαδήποτε OpenCL βιβλιοθήκη είναι διαθέσιμη στους κόμβους για να εκτελέσει πυρήνες για λογαριασμό των εφαρμογών πελάτη.

Εξομοιώνει όλες τις διαδικασίες OpenCL όπως αυτές ζητούνται από την βιβλιοθήκη VirtualCL. Για λόγους ασφάλειας, όσο οι συσκευές GPU και οι οδηγοί δεν επιτρέπουν διαφανής προτίμηση, οι συσκευές OpenCL δεν κοινοποιούνται από το back-end στις υπόλοιπες εφαρμογές. Κάθε συσκευή κατανέμεται σε μόνο μια εφαρμογή κάθε στιγμή.[[virtualcl-4](#)]

Προβλήματα

Όταν εκτελούμε πυρήνες OpenCL σε απομακρυσμένες συσκευές, η καθυστέρηση δικτύου είναι ο κύριος παράγοντας περιορισμού. Η ελαχιστοποίηση του αριθμού των "ταξιδιών" για βασικές λειτουργίες OpenCL είναι το πρώτο βήμα για να αντιμετωπίσουμε αυτό το πρόβλημα.

Γιαυτό αναπτύχθηκε το SuperCL, όπου μια σειρά απο πυρήνες και λειτουργίες μνήμης αποστέλλονται στις συσκευές ενός κόμβου συμπλέγματος, συνήθως μόνο σε ένα "ταξίδι". Όταν είναι απαραίτητο, επιτρέπεται η επικοινωνία με τον ξενιστή, αλλά με ασύγχρονο τρόπο, για να αποφευχθεί περαιτέρω καθυστέρηση.

Το εύρος ζώνης συχνότητας είναι επίσης περιοριστικός παράγοντας όταν περιέχονται τεράστιες διεργασίες I/O, και αυτό αντιμετωπίζεται από το SuperCL επιτρέποντας την αρχικοποίηση των buffer από αρχεία back-end, και την αποθήκευση των αποτελεσμάτων σε αρχεία back-end.[18]

2.3 Συμπεράσματα

Σε αυτό το κεφάλαιο μελετήσαμε τις σημαντικές διαφορές του υπολογισμού GPUs και CPUs, ενώ είδαμε και τον τρόπο με τον οποίο λειτουργεί ένα σύμπλεγμα μονάδων επεξεργασίας γραφικών.

Το συγκεκριμένο πρόβλημα είναι πολύ σημαντικό για εταιρίες που ασχολούνται με υπερ-υπολογιστές, καθώς η χρήση των GPUs με τον κατάλληλο προγραμματισμό, τους επιτρέπει να δημιουργήσουν υπολογιστικές λύσεις σε δύσκολα προβλήματα (π.χ μελέτες παγκόσμιου κλίματος ή ωκεανογραφικές μελέτες), οι οποίες μπορούν με την σειρά τους να ξεκλειδώσουν νέες δυνατότητες που θα επηρεάσουν την ανθρώπινη καθημερινότητα.

Ένα από τα πολλά πλεονεκτήματα της χρήσης GPUs για τους υπερ-υπολογιστές είναι η μικρότερη κατανάλωση ενέργειας και η ελευθέρωση μεγάλου όγκου χώρου, το οποίο μας επιτρέπει να προσθέσουμε επιπλέον υπολογιστική δύναμη με επιπλέον συσκευές. Οι υπερ-υπολογιστές είναι πολύ σημαντικοί για τις εταιρίες και άλλα ερευνητικά ιδρύματα τα οποία μελετούν φυσικά φαινόμενα, όπως η εξομοίωση της ανόδου της θερμοκρασίας και τις επιπτώσεις της στον πλανήτη.

Στο επόμενο κεφάλαιο θα δούμε τις σημαντικότερες υλοποιήσεις εφαρμογών σε διάφορα επιστημονικά πεδία, τον λόγο που χρειάζεται επιτάχυνση στην επίλυση των προβλημάτων αυτών, ενώ θα ερευνήσουμε εφαρμογές που παρουσιάζουν ιδιαίτερο ενδιαφέρον. Τέλος, θα δούμε το μέγεθος τις επιτάχυνσης που επιτυγχάνεται σε διάφορες εφαρμογές.

Εφαρμογές

3.1 Εισαγωγή

Τα τελευταία περίπου 20 χρόνια οι εταιρίες παραγωγής υλικού γραφικών έχουν εστιάσει στην προσπάθεια να παράγουν γρήγορες μονάδες γραφικής επεξεργασίας (GPU), ειδικότερα για την κοινότητα των φίλων των ηλεκτρονικών παιχνιδιών.

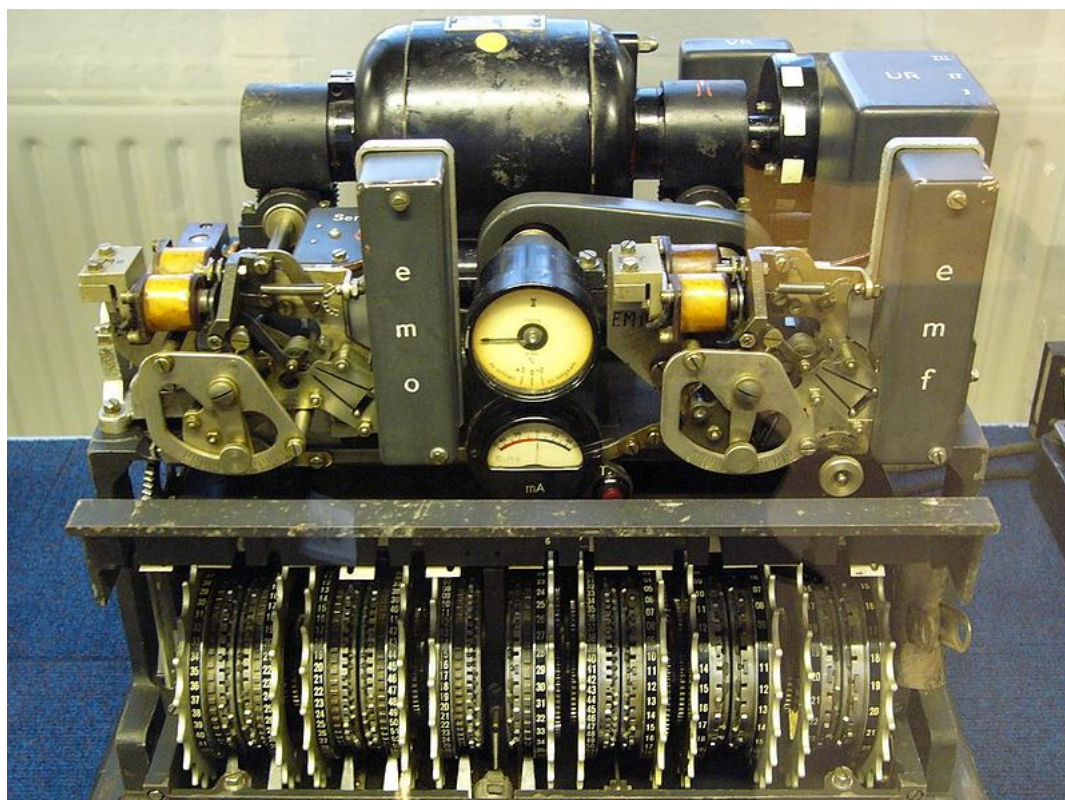
Αυτό έχει ως αποτέλεσμα πρόσφατα να δημιουργηθούν συσκευές οι επιδόσεις των οποίων ξεπερνούν τις κεντρικές μονάδες επεξεργασίας (CPU), σε συγκεκριμένες εφαρμογές, ειδικότερα σε μετρήσεις εκατομμυρίων εντολών το δευτερόλεπτο (MIPS). Έτσι, καθιερώθηκε μια κοινότητα για να αξιοποιήσει αυτήν την μεγάλη δύναμη των GPU για υπολογισμούς γενικής χρήσης (GPGPU).

Τα τελευταία δύο χρόνια έχουν εξαλειφθεί οι περισσότεροι περιορισμοί που υπήρχαν όσον αφορά το σετ εντολών και την διαχείριση μνήμης, με την ενσωμάτωση ενοποιημένων υπολογιστικών μονάδων στις κάρτες γραφικών, δίνοντας έτσι την δυνατότητα στους προγραμματιστές να δημιουργήσουν ένα πλήθος από προγράμματα με εφαρμογές σε πολλούς τομείς.

3.2 Κρυπτογράφηση

Στο πεδίο της ασύμμετρης κρυπτογράφησης, η ασφάλεια όλων των πρακτικών κρυπτοσυστημάτων βασίζεται στην δυσκολία υπολογισμού προβλημάτων, εξαρτημένη από την επιλογή των παραμέτρων. Με την όποια αύξηση των παραμέτρων όμως (συνήθως στο εύρος 1024-4096 bits), οι υπολογισμοί γίνονται όλο και πιο απαιτητικοί για τον εκάστοτε επεξεργαστή. Σε σύγχρονο υλικό, ο υπολογισμός μιας μονής εντολής κρυπτογράφησης δεν είναι κρίσιμος, όμως σε ένα σύστημα επικοινωνίας πολλών-προς-ένα, για παράδειγμα ένας κεντρικός server στο κέντρο δεδομένων μιας εταιρίας, μπορεί να αντιμετωπίσει ταυτόχρονα εκατοντάδες η και χιλιάδες ταυτόχρονες συνδέσεις και εντολές κρυπτογράφησης.

Ως αποτέλεσμα, η πιο συνήθης λύση για ένα τέτοιο σενάριο είναι η χρήση καρτών επιτάχυνσης κρυπτογράφησης. Λόγω της μικρής αγοράς, η τιμή τους φτάνει συνήθως αρκετά χιλιάδες ευρώ η δολάρια. Τελευταία, η ερευνητική κοινότητα έχει αρχίσει να εξερευνά τεχνικές για επιτάχυνση των αλγορίθμων κρυπτογράφησης με χρήση της GPU. [19]



Σχήμα 3.1: Μηχανή κρυπτογράφησης German Lorenz, χρησιμοποιήθηκε στον δεύτερο παγκόσμιο πόλεμο για να κρυπτογραφεί μηνύματα για προσωπικό πολύ υψηλής σημασίας

3.2.1 Κρυπτογράφηση συμμετρικού κλειδιού

Η κρυπτογραφία συμμετρικού κλειδιού αναφέρεται στις κρυπτογραφικές μεθόδους στις οποίες ο αποστολέας και ο αποδέκτης μοιράζονται το ίδιο

κλειδί (ή σε πιο σπάνια περίπτωση, όταν τα κλειδιά είναι διαφορετικά, αλλά βρίσκονται εύκολα μέσω απλού υπολογισμού). Αυτή ήταν η μόνη μορφή κρυπτογραφίας γνωστή μέχρι το 1976. Οι κρυπτογράφοι συμμετρικών κλειδιών υλοποιούνται σαν block ή κρυπτογράφοι ροής. Ένας κρυπτογράφος block κρυπτογραφεί τα δεδομένα σε blocks αντί για τον κάθε χαρακτήρα ξεχωριστά, δηλαδή με τον τρόπο που λειτουργεί ο κρυπτογράφος ροής.[20]



Σχήμα 3.2: Γαλλική μηχανή κρυπτογράφησης σε σχήμα βιβλίου του 16ου αιώνα

Το πρότυπο κρυπτογράφησης δεδομένων (DES) και το ανεπτυγμένο πρότυπο κρυπτογράφησης (AES) είναι κρυπτογράφοι block σχεδιασμένοι από την Αμερικάνικη κυβέρνηση. Αν και έχει σταματήσει να υποστηρίζεται σαν επίσημο πρότυπο, η χρήση του DES είναι ακόμα αρκετά δημοφιλής. Χρησιμοποιείται για μια μεγάλη σειρά από εφαρμογές, όπως κρυπτογράφηση ATM, ασφαλής απομακρυσμένη πρόσβαση, ασφαλής επικοινωνία ηλεκτρονικού ταχυδρομείου, κ.α.

Οι κρυπτογράφοι ροής, σε αντίθεση με τους block, δημιουργούν μια μεγάλη σειρά από υλικά κλειδιών, που συνδυάζονται με το απλό κείμενο χαρακτήρα προς χαρακτήρα. Σε έναν κρυπτογράφο ροής, το αποτέλεσμα παράγεται βασισμένο σε μια κρυμμένη κατάσταση που αλλάζει όσο λειτουργεί ο κρυπτογράφος. Αυτή η εσωτερική κατάσταση αρχικοποιείται με την χρήση ενός κρυμμένου κλειδιού. Παράδειγμα γνωστού κρυπτογράφου ροής είναι ο RC4.[21]

Οι κρυπτογραφικές διαδικασίες hash είναι ένας τρίτος τύπος αλγόριθμου κρυπτογράφησης. Λέχονται σαν είσοδο οποιαδήποτε μήκος κειμένου, και δίνουν αποτέλεσμα ένα μικρό κείμενο συγκεκριμένου μήκους που μπορεί να χρησιμοποιηθεί σαν ψηφιακή ταυτότητα. Όσον αφορά τις καλές διαδικασίες hash, ένας εισβολέας δεν μπορεί να βρει δύο μηνύματα που παράγουν το ίδιο hash. [22]Το MD5 για παράδειγμα, ενώ χρησιμοποιείται εδώ και πολλά

χρόνια, έχει αποδειχτεί ότι είναι εύκολο να παραβιαστεί. Το SHA-1 είναι περισσότερο διαδεδομένο ως ασφαλές διαδικασία hash κρυπτογράφησης, αλλά αναλυτές έχουν ανακαλύψει επιθέσεις για την παραβίαση του. Το 2012, μια καινούρια διαδικασία κρυπτογράφησης hash ανακαλύφθηκε, η SHA-3, η οποία χρησιμοποιεί τον αλγόριθμο Keccak για την υλοποίηση της. Σε παρακάτω υπο-ενότητα θα μελετήσουμε την χρήση προγράμματος για παραβίαση των διαδικασιών κρυπτογράφησης hash.[23]

3.2.2 Κρυπτογράφηση δημοσίου κλειδιού

Τα κρυπτοσυστήματα συμμετρικών κλειδιών χρησιμοποιούν το ίδιο κλειδί για την κρυπτογράφηση και την αποκρυπτογράφηση ενός μηνύματος, αν και ένα μήνυμα ή ομάδα μηνυμάτων μπορεί να έχουν διαφορετικό κλειδί από τα άλλα. Ένα σημαντικό μειονέκτημα των συμμετρικών κρυπτογραφημάτων είναι η διαχείριση κλειδιών, ώστε αυτά να χρησιμοποιηθούν με ασφάλεια.



Σχήμα 3.3: Η μηχανή κρυπτογράφησης Enigma, χρησιμοποιήθηκε από τον Γερμανικό στρατό και τις πολιτικές αρχές από τα τέλη του 1920 μέχρι και τον δεύτερο παγκόσμιο πόλεμο, παρείχε ένα πολύπλοκο ηλεκτρο-μηχανικό πολύ-αλφαβητικό κρυπτογράφημα. Η αποκρυπτογράφηση του αλγόριθμου αποδείχτηκε μεγάλης σημασίας για την νίκη των συμμάχων.

3.2.3 Hashcat

Για λόγους έρευνας της εργασίας επιλέχτηκε το πρόγραμμα oclHashcat ώστε να μελετήσουμε την απόδοση των κρυπτογραφικών προγραμμάτων με χρήση GPGPU. Το Hashcat είναι το πιο γρήγορο πρόγραμμα επαναφοράς κωδικών. Είναι ελεύθερο, αν και κλειστού κώδικα. Υπάρχουν εκδόσεις για Linux, OSX, και Windows, και υλοποιήσεις για CPU ή GPU. Το hashcat υποστηρίζει μεγάλο αριθμό από hashing αλγορίθμους, συμπεριλαμβανομένου MD4, MD5, SHA, Unix Crypt, κ.α

Υλοποιήσεις

Το Hashcat διανέμεται σε δύο εκδόσεις

- Hashcat - Ένα εργαλείο επαναφοράς κωδικών για CPU
- oclHashcat - Επαναφορά κωδικών με χρήση GPGPU

Πολλοί από τους αλγόριθμους που υποστηρίζει το hashcat μπορούν να βρεθούν σε μικρότερο χρόνο με την χρήση της GPU-επιτάχυνσης του oclHashcat(όπως MD5,SHA1, κ.α) Όμως, δεν επιταχύνονται όλοι οι αλγόριθμοι από την χρήση GPU. Το Bcrypt είναι ένα τέτοιο παράδειγμα. Λόγω παραγόντων όπως διακλαδώσεις δεδομένων, σειριακοποίηση, μνήμη, κ.α, το oclHashcat δεν είναι απόλυτος αντικαταστάτης του Hashcat.

Επιθέσεις

Το Hashcat υποστηρίζει πολλούς τύπους επιθέσεων για την επαναφορά πολύπλοκων κωδικών από τον χώρο κλειδιού ενός hash. Αυτοί οι τύποι περιλαμβάνουν:

- Επίθεση Brute-Force
- Επίθεση Combinator
- Επίθεση λεξικού
- Επίθεση αποτυπώματος
- Υβριδική επίθεση
- Επίθεση μάσκας
- Επίθεση Permutation
- Επίθεση βασισμένη σε κανόνες
- Επίθεση με χρήση πίνακα
- Επίθεση Toggle-Case

3.2.4 Κρυπτο-νομίσματα

Τα κρυπτονομίσματα είναι μια μορφή συναλλαγής με χρήση κρυπτογραφίας για ασφάλεια και τον έλεγχο της δημιουργίας καινούριων νομισμάτων. Το πρώτο κρυπτο-νόμισμα που δημιουργήθηκε ήταν το Bitcoin το 2009. Από τότε, πολλά κρυπτονομίσματα έχουν δημιουργηθεί. Ένα χαρακτηριστικό τους είναι ότι δεν υπάρχει κεντρικός έλεγχος, σε αντίθεση με άλλα συστήματα ηλεκτρονικού χρήματος όπως το Paypal. Ακόμα ένα χαρακτηριστικό είναι ότι οι συναλλαγές καταγράφονται δημόσια, για παράδειγμα στο Bitcoin, οι συναλλαγές καταγράφονται στην block αλυσίδα. [24]

Η δημιουργία κρυπτονομισμάτων γίνεται μέσω ειδικών προγραμμάτων που υπολογίζουν συγκεκριμένα hashes. Τα κρυπτονομίσματα χρησιμοποιούν έναν αλγόριθμο "Proof of work" ώστε να γνωρίζουν ότι ο χρήστης έχει εξορύξει τα νομίσματα που αυτός αναφέρει. (π.χ Scrypt, SHA-256, κ.α). Η εξόρυξη ενός block είναι μια διαδικασία που μοιάζει πολύ με την επαναφορά κωδικών. Για αυτό οι GPUs, είναι πιο αποδοτικές από τις CPUs - όταν μια τυπική CPU έχει μέχρι 8 πυρήνες, μια GPU μπορεί να έχει εκατοντάδες, όπου η καθεμία από αυτές υπολογίζει ένα διαφορετικό "hash".

Υπάρχουν δύο τρόποι για εξόρυξη εικονικών νομισμάτων.



Σχήμα 3.4: Λογότυπο Bitcoin, το πρώτο και το πιο επιτυχημένο κρυπτο-νόμισμα

- Ατομική εξόρυξη - Η ατομική προσπάθεια για εξόρυξη νομισμάτων με χρήση μιας ή περισσότερων GPU
- Εξόρυξη κοινοπραξίας - Η προσπάθεια από πολλούς ανθρώπους να εξορύξουν το ίδιο block, διαιρώντας τα κέρδη.

Οι ταχύτητες της εξόρυξης μετρώνται σε KH/s (kilohashes ανα δευτερόλεπτο) και MH/s (megahashes ανα δευτερόλεπτο). Για παράδειγμα, μια Radeon 4870 εξορύσσει ένα νόμισμα SHA-256 με ταχύτητα 80-100 MH/s, ενώ ένα νόμισμα Scrypt με ταχύτητα μόλις 130-140 KH/s.

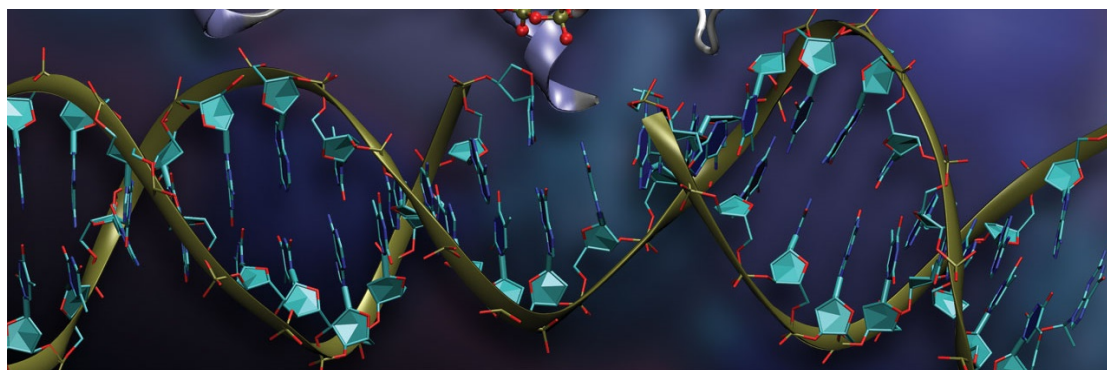
Η χρήση των GPUs για εξόρυξη εικονικών νομισμάτων, οδήγησε την αγορά των καρτών γραφικών σε αύξηση των τιμών, αφού για πολύ καιρό ήταν μια αποδοτική λύση σε αντίθεση με την χρήση CPUs. Πρόσφατα όμως, με την παρουσίαση εξειδικευμένων καρτών (ASICS - Application Specific Integrated Circuit), η χρήση των GPUs για εξόρυξη κρυπτο-νομισμάτων άρχισε να μειώνεται, αφού η απόδοσή τους είναι έως και 1000 φορές μικρότερη. Αυτό ήταν επίσης ένας λόγος για τον απλό χρήστη να απομακρυνθεί από την εξόρυξη νομισμάτων, αφού δημιουργήθηκαν μικρότερες κοινοπραξίες που απαιτούν μεγαλύτερη υπολογιστική δύναμη.

3.3 Βιοπληροφορική

3.3.1 Εισαγωγή

Η συνεχής αύξηση της ποσότητας βιολογικών δεδομένων, η ανάγκη για ανάλυση τους και το συνεχές ενδιαφέρον από την επιστημονική κοινότητα για την κατανόηση των δομικών λειτουργιών των βιολογικών μορίων, αποτέλεσαν τους κύριους λόγους για την ανάπτυξη της βιοπληροφορικής. Για να κατανοήσουμε τις κυτταρικές και βιομοριακές λειτουργίες, τα βιολογικά δεδομένα πρέπει να συνενωθούν για να σχηματίσουν μια ακριβής εικόνα. Οι ερευνητές της βιοπληροφορικής, έχουν αναπτύξει υπολογιστικές τεχνικές για την επεξεργασία των βιολογικών δεδομένων, όπως νουκλεοτιδικές αλληλουχιών, αλληλουχίες αμινο οξέων, τρισδιάστατων δομών, όπως επίσης βιολογικών σημάτων και εικόνων. Μεγάλες ερευνητικές προσπάθειες του πεδίου συμπεριλαμβάνουν αναγνώριση προτύπων, ευθυγράμμιση αλληλουχιών, ανάλυση πρωτεϊνικών δομών, φυλογενητική ανάλυση, μοριακή δυναμική, ανάλυση γονιδιώματος, σχεδιασμός φαρμάκων και ανάπτυξη φαρμάκων. Επίσης, υπάρχουν προφητικές τεχνικές ειδικές για τις εκφράσεις γονιδίων, και την αλληλεπίδραση πρωτεϊνών.

Η Βιοπληροφορική παίζει μεγάλο ρόλο σε πολλές πτυχές της βιολογίας.



Σχήμα 3.5: Βιολογία και πληροφορική

Στην πειραματική μοριακή βιολογία, οι τεχνικές βιοπληροφορικής όπως επεξεργασία εικόνας και σήματος, επιτρέπει την εξόρυξη χρήσιμων αποτελεσμάτων από μεγάλο όγκο δεδομένων. Στο πεδίο της γενετικής και γονιδιωμικής, συμβάλλει στην αλληλουχία και υποσημείωση γονιδιωμάτων και την παρατήρηση των μεταλλάξεων τους.[25] Παίζει μεγάλο ρόλο στην εξόρυξη τεχνικών όρων και στην κατασκευή βιολογικών και γονιδιακών οντολογιών για την οργάνωση και αναζήτηση βιολογικών δεδομένων. Έχει επίσης μεγάλο ρόλο στην ανάλυση των γονιδίων και στην ρύθμιση πρωτεϊνών. Τα εργαλεία της Βιοπληροφορικής συμβάλουν στην σύγκριση γενετικών και γονιδιακών δεδομένων και γενικότερα στην κατανόηση των αναπτυξιακών πτυχών της μοριακής βιολογίας. Σε πιο εσωτερικό επίπεδο, συμβάλλει στην ανάλυση και κατηγοριοποίηση των βιολογικών διαδρόμων και δικτύων τα οποία είναι σημαντικό κομμάτι της συστημικής βιολογίας. Στην Δομική βιολογία, συμβάλλει στην εξομοίωση και μοντελισμό του DNA, RNA, και δομές πρωτεϊνών όπως

και μοριακών αλληλεπιδράσεων.

3.3.2 Μοριακή δυναμική

Η Βιοπληροφορική είναι ένα επιστημονικό πεδίο που εστιάζει στην εφαρμογή της τεχνολογίας υπολογιστών στην διαχείριση βιολογικών δεδομένων. Με το πέρασμα του χρόνου, οι εφαρμογές βιοπληροφορικής έχουν χρησιμοποιηθεί για να αποθηκεύσουν, αναλύσουν και να ενσωματώσουν βιολογικές και γενετικές πληροφορίες, χρησιμοποιώντας ένα μεγάλο εύρος μεθοδολογιών. Μια από τις πλέον γνωστές τεχνικές για την κατανόηση των φυσικών κινήσεων των ατόμων και των μορίων, είναι η μοριακή δυναμική. Η μοριακή δυναμική είναι μια μέθοδος εξομοίωσης των φυσικών κινήσεων των ατόμων και των μορίων κάτω από συγκεκριμένες συνθήκες. Έχει ρόλο κλειδί σε επιστήμες όπως η βιολογία, η χημεία, η φυσική, ιατρική. Λόγω της πολυπλοκότητας τους, οι υπολογισμοί της μοριακής δυναμικής χρειάζονται μεγάλες ποσότητες μνήμης και υπολογιστικής δύναμης, και για αυτό η εκτέλεση τους είναι συχνά μεγάλο πρόβλημα.[26]

Οι εξομοιώσεις της μοριακής δυναμικής χρησιμοποιούν πολύπλοκους αριθμητικούς υπολογισμούς, που πολλές φορές οδηγούν σε αριθμητικά λάθη. Πριν την ανακάλυψη του προγραμματισμού γενικής χρήσης, οι GPU χρησιμοποιούνταν μόνο για διαδικασίες απεικόνισης των μοριακών δομών, και η εκτέλεση των αλγορίθμων μοριακής δυναμικής μπορούσε να διαρκέσει από ώρες, έως και μέρες. Η λύση προήλθε από το GPGPU, καθώς οι GPU έχουν πολλές αριθμητικές μονάδες που μπορούν να εκτελεστούν παράλληλα.

Στο πεδίο της μοριακής δυναμικής, έχουν αναπτυχθεί πολλές εφαρμογές εφαρμογές βασισμένα στο GPGPU, που υποστηρίζουν εξομοιώσεις σε πολλαπλές μονάδες. Αυτή η καινοτομία δημιουργεί ευκαιρίες για το μέλλον, ειδικά για μικρότερες ερευνητικές ομάδες. Μειώνει τον χρόνο που απαιτείται για διαδικασίες και τα απαραίτητα κονδύλια για έρευνα-ανάπτυξη, προάγει την ανάπτυξη καινούριων εφαρμογών και την επιστημονική πρόοδο.[27]

Μετάβαση από CPU σε GPU

Η διαφορά στην αρχιτεκτονική μεταξύ CPU και GPU, είναι ότι στην τελευταία είναι δυνατή η εκτέλεση πολλαπλών παράλληλων διεργασιών, κάτι που επιτρέπει την καλύτερη εκτέλεση πολύπλοκων αλγορίθμων και καλύτερη διαχείριση μεγάλου όγκου δεδομένων. Επίσης, μια μονάδα επεξεργασίας γραφικών έχει λιγότερες ενεργειακές απαιτήσεις, έτσι η δημιουργία υπερ-υπολογιστών με χρήση GPU εξαλείφει την ανάγκη για τεράστιους χώρους γεμάτους με υπολογιστές. Η εγκατάσταση μιας επιπλέον μονάδας, αντιγράφει τον παραλληλισμό του προγραμματισμού, χωρίς καμία επιπλέον ενέργεια. Επιπλέον, η GPU έχει εντυπωσιακές δυνατότητες υπολογισμού floating point και μεγάλο εύρος ζώνης μνήμης, δίνοντας την δυνατότητα για βελτιστοποιημένη πρόσβαση στην μνήμη, ελεγχόμενη εκτέλεση επιλογών, και διαχείριση πόρων, με χρήση λίγων γραμμών κώδικα. Συγκεκριμένα, οι εφαρμογές μοριακής δυναμικής, κβαντικής χημείας, η απεικόνιση των αποτελεσμάτων τους, τρέχουν μέχρι και 5 φορές πιο γρήγορα.

Από την αρχή του GPU προγραμματισμού μέχρι και σήμερα, η προγραμματιστική ανάπτυξη συνεχίζεται αδιάκοπα. Ο αριθμός των εφαρμογών βασισμένων σε αρχιτεκτονικές GPU, φτάνουν τις 200, το οποίο είναι αύξηση της τάξεως πάνω από 60% μέσα σε δύο χρόνια. Οι καλύτερες εφαρμογές βασισμένες σε GPGPU έχουν σχεδιαστεί για την μοριακή δυναμική, τον σχεδιασμό φαρμάκων, κβαντική χημεία, το κλίμα, την φυσική σύμφωνα με την Nvidia[28]

Πίνακας 3.1: Μοριακή δυναμική

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
ACEMD	Εξομοίωση μοριακών μηχανικών πεδίων ισχύος με χρήση GPU		
AMBER	Σουίτα προγραμμάτων για εξομοίωση μοριακής δυναμικής σε βιομόρια		
CHARMM	Πακέτο μοριακής δυναμικής για εξομοίωση σε βιομόρια		
DESMOND	Δυναμικές εξομοιώσεις βιολογικών συστημάτων υψηλών ταχυτήτων		
DL-POLY	Εξομοίωση μακρομορίων, πολυμερών, ιονικών συστημάτων, κ.α σε υπολογιστή κατανεμημένης παράλληλης μνήμης		
ESPResSo	Πακέτο λογισμικού υψηλής ευελιξίας για ανάλυση και απόδοση επιστημονικών μοριακών δυναμικών.		
Folding@Home	Ένα πολύ γνωστό κατανεμημένο υπολογιστικό σύστημα που μελετάει διπλώσεις πρωτεϊνών και σχετικές αρρώστιες		
GPUGrid.net	Ένα κατανεμημένο υπολογιστικό σύστημα που χρησιμοποιεί GPUs για εξομοιώσεις μορίων		
GROMACS	Εξομοίωση βιοχημικών μορίων με διαδραστικότητα περίπλοκων δεσμών		
HALMD	Εξομοιώσεις μεγάλης κλίμακας απλών και πολύπλοκων υγρών		
HOOMD-Blue	Πακέτο λογισμικού εξομοίωσης σωματιδίων για GPUs		
LAMMPS	Πακέτο λογισμικού κλασσικής μοριακής δυναμικής		
NAMD	Σχεδιασμένο για εξομοίωση υψηλής απόδοσης σε μεγάλα μοριακά συστήματα		
OpenMM	Βιβλιοθήκη και εφαρμογή για μοριακή δυναμική υψηλής υπολογιστικής απόδοσης με χρήση GPUs		

3.3.3 GPUGRID.net

"I hope mankind will acknowledge people like you, its real heroes."

Grzegorz Granowski, Volunteer & Donor

Το GPUGRID είναι ένα εθελοντικό κατανεμημένο σύστημα, το οποίο στοχεύει στην βιοϊατρική έρευνα από το πανεπιστήμιο Universitat Pompeu Fabra της Ισπανίας. Το GPUGRID αποτελείται από πολλές μονάδες επεξεργασίας γραφικών, που συνεργάζονται μεταξύ τους για να παραδώσουν υψηλών επιδόσεων εξομοιώσεις βιομορίων. Οι μοριακές εξομοιώσεις που εκτελούνται από τους εθελοντές του, αποτελούν μερικούς από τους πιο συνήθεις τύπους εξομοιώσεων που εκτελούνται από τους επιστήμονες του πεδίου, αλλά ταυτόχρονα είναι από τους πιο απαιτητικούς σε υπολογιστική δύναμη και συνήθως απαιτούν υπερ-υπολογιστές.



Σχήμα 3.6: Βιολογία και πληροφορική

Το σύστημα ερευνά μεταξύ άλλων τα παρακάτω προβλήματα

- Εξομοίωση της ωρίμανσης πρωτεολυτικών του HIV - Μια από τις πιο σημαντικές πτυχές της ωρίμανσης του HIV είναι το πώς η πρωτεΐνη "ψαλιδιών", δημιουργείται. Η απάντηση σε αυτό το ερώτημα χρειάζεται εξομοιώσεις μοριακής δυναμικής στο όριο των μοντέρνων υπολογιστικών δυνατοτήτων. Το GPUGRID μας επιτρέπει να λύσουμε αυτό το πρόβλημα και έχουμε καταφέρει να δείξουμε ότι τα πρώτα "ψαλίδια" κόβονται από το "σκοινί" που είναι δεμένα. Αυτό το γεγονός συμβαίνει στην αρχή της ωρίμανσης, και αν σταματήσουμε την ωρίμανση των πρωτεολυτικών, τότε θα σταματήσουμε και την ωρίμανση του HIV σαν σύνολο.[29]
- Ανακάλυψη του ρόλου των μεμβρανών λιπιδίων στην δραστηριότητα ενζύμων.
- Μοριακή εξομοίωση αισθητήρων ντοπαμίνης κάτω από φυσιολογικές ιονικές δυνάμεις.
- Αποκάλυψη των μηχανισμών αντίδρασης φαρμάκων καρκίνου παχέος εντέρου - Ο καρκίνος είναι βασικά ή ανεξέλεγκτη ανάπτυξη ιστών και εισβολή από μεταλλαγμένα κύτταρα σε έναν οργανισμό. Σε αντίθεση με τις παραδοσιακές χημειοθεραπείες ή ραδιοθεραπείες, οι νεότερες θεραπείες στοχεύουν σε συγκεκριμένους στόχους κακοήθων κυττάρων. Αυτό επιτυγχάνεται με τον εντοπισμό ορισμένων πρωτεϊνών που εκφράζονται διαφορετικά σε ογκογεννητικά κύτταρα. Με την βοήθεια του GPUGRID, επιτυγχάνεται η επεξήγηση των μοριακών μηχανισμών που συμβαίνουν στα μεταλλαγμένα μόρια των κυττάρων.

Η εκτέλεση του GPUGRID στις GPUs, καινοτομεί στον εθελοντικό υπολογισμό, παραδίδοντας εφαρμογές υπερ-υπολογιστών, σε υποδομές χαμηλού κόστους. Η απόδοση των μονάδων γραφικής επεξεργασίας, καταγράφεται και συγκρίνεται σε σχέση με άλλους χρήστες, ανάλογα με την διάρκεια ολοκλήρωσης των WU (Work Units)[30].

Rank	User name	WU id	Timestamp (h)	GPU description
1	Retvari Zoltan*	10083132	4.59	[2] NVIDIA GeForce GTX 780 Ti (3071MB) driver: 344.11
2	valterc	10106939	4.81	NVIDIA GeForce GTX 780 Ti (3071MB)
3	petebe	10092106	5.20	[3] NVIDIA GeForce GTX TITAN (4095MB) driver: 335.28
4	Matt	10092250	5.24	[2] NVIDIA GeForce GTX 780 Ti (3072MB) driver: 344.11
5	Justin	10093651	5.31	[2] NVIDIA GeForce GTX 780 Ti (3072MB) driver: 340.52
6	jgis	10087293	5.33	NVIDIA GeForce GTX 780 Ti (3072MB) driver: 344.11
7	TJ	10106709	5.33	[2] NVIDIA GeForce GTX 780 Ti (3072MB) driver: 337.88
8	Pubodee	10106064	5.49	NVIDIA GeForce GTX 780 Ti (3072MB) driver: 344.11
9	[VENETO] sabayonino	10094096	5.50	[2] NVIDIA GeForce GTX 780 (3071MB)

Σχήμα 3.7: Βιολογία και πληροφορική

3.3.4 Προγράμματα

Πίνακας 3.2: Βιοπληροφορική

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
BarraCUDA	Λογισμικό χαρτογράφησης ακο- λουθίας	6-10x	NAI
CUDASW++	Λογισμικό ανοιχτού κώδικα για αναζητήσεις Smith-Waterman σε πρωτεϊνικές βάσεις δεδομένων με χρήση GPUs	10-50x	NAI
CUSHAW	Ευθυγραμμιστής παράλληλων μι- κρών προσπελάσεων	10x	NAI
G-BLASTN	Επιταχυνόμενο από GPU εργα- λείο ευθυγράμμισης νουκλεοτι- δίων βασισμένο στο ευρέως διαδε- δομένο NCBI-BLAST	4-15x	
GPU-BLAST	Τοπική αναζήτηση με γρήγορους ευρετικούς αλγόριθμους k-tuple	3-4x	

Συνέχεια πίνακα 3.2			
Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
mCUDA-MEME	Πολύ γρήγορη κλιμακωτή ανακάλυψη μοτίβων βασισμένη στο MEME	4-10x	NAI
MUMmer GPU	Πρόγραμμα υψηλής απόδοσης τοπικής ευθυγράμμισης ακολουθίας	3-10x	
NVBIO	Βιβλιοθήκη ανοιχτού κώδικα C++ αποτελούμενη από στοιχεία επαναχρησιμοποιήσιμα σχεδιασμένα για να επιταχύνουν εφαρμογές βιοπληροφορικής με χρήση CUDA.	4-5x	NAI
NVBowtie	Μια σε μεγάλο βαθμό ολοκληρωμένη εφαρμογή του ευθυγραμμιστή Bowtie2 πάνω από το NVBIO	2.75x-8.35x	NAI
PEANUT	Καθορισμός ανάγνωσης για ακολουθίες DNA ή RNA σε γνωστή αναφορά γονιδιώματος.	10x	
REACTA	Ο σκοπός του REACTA είναι ο ποσοτικός προσδιορισμός της συμβολής της γενετικής διακύμανσης στην φαινοτυπική διακύμανση σε πολύπλοκα χαρακτηριστικά.	2-4x	NAI
SeqNFind	Ακολουθία επόμενης γενιάς και συγκρίσεις γονιδιωμάτων	400x	NAI
SOAP3	Λογισμικό βασισμένο σε GPU για ευθυγράμμιση μικρών προσπελάσεων με αναφορά ακολουθίας. Μπορεί να βρει όλες τις ευθυγραμμίσεις με k ασυμφωνίες, όπου το k είναι ένας αριθμός από το 0 έως το 3	10x	NAI
SOAP3-dp	Πολύ γρήγορο εργαλείο βασισμένο σε GPU για ευθυγραμμίσεις μικρών προσπελάσεων μέσω δυναμικού προγραμματισμού υποβοηθούμενου από ευρετήριο	28-64x	NAI
UGENE	Λογισμικό ανοιχτού κώδικα Smith-Waterman για SSE/CUDA, επαναλήψεις βασισμένες σε πίνακα δεικτών	6-8x	NAI
WideLM	Ταιριάζει πολυάριθμα γραμμικά μοντέλα σε μια σταθερή σχεδίαση και απάντηση	150x	NAI

3.4 Κβαντική χημεία

3.4.1 Εισαγωγή

Η κβαντική χημεία είναι μια διακλάδωση της χημείας όπου η κύρια εστίαση είναι η εφαρμογή κβαντικής μηχανικής σε μοντέλα φυσικής και σε πειράματα χημικών συστημάτων. Η ιστορία της κβαντικής χημείας ουσιαστικά ξεκινάει με την ανακάλυψη των καθοδικών ακτίνων από τον Michael Faraday, την υπόδειξη του 1877 από τον Ludwig Boltzmann ότι τα επίπεδα ενέργειας σε ένα σύστημα φυσικής μπορεί να είναι διακριτά, και την κβαντική υπόθεση του 1900 από τον Max Planck ότι κάθε ατομικό σύστημα ακτινοβολίας ενέργειας μπορεί να διαιρεθεί σε έναν αριθμό διακριτών στοιχείων έτσι ώστε κάθε από αυτά τα στοιχεία ενέργειας να είναι αναλογικό στην συχνότητα με την οποία ακτινοβολούν ενέργεια και μια αριθμητική τιμή που ονομάζεται σταθερά Planck.[31]

Η Κβαντική χημεία περιλαμβάνει βαθιά χρήση πειραματικών και θεωρητικών μεθόδων:

- Οι πειραματικοί κβαντικοί χημικοί βασίζονται στην φασματοσκοπία, μέσω της οποίας η πληροφορία σχετικά με την κβάντιση της ενέργειας σε μοριακή κλίμακα μπορεί να αποκτηθεί. Τυπικές μέθοδοι είναι η υπέρυθρη φασματοσκοπία και η πυρηνική μαγνητική αντήχηση (NMR) φασματοσκοπίας
- Θεωρητική κβαντική χημεία, το έργο της οποίας βρίσκεται κάτω από την κατηγορία της υπολογιστικής χημείας, στοχεύει να υπολογίσει τις προβλέψεις της κβαντικής θεωρίας καθώς τα άτομα και τα μόρια μπορούν να έχουν μόνο διακριτές ενέργειες. Λόγω του ότι αυτή η διαδικασία, όταν εφαρμοστεί σε πολύ-ατομικά είδη δημιουργεί υπολογιστικά προβλήματα, αυτοί οι υπολογισμοί γίνονται με την χρήση υπολογιστών αντί για την αναλυτική μέθοδο, και άλλες παραδοσιακές μεθόδους.[32]

Κατά αυτούς τους τρόπους, η κβαντική χημεία μελετάει τα χημικά φαινόμενα. Στις αντιδράσεις, η κβαντική χημεία μελετάει την σταθερή κατάσταση των ατόμων και των μορίων, τις μη σταθερές καταστάσεις, και την μετάβαση των καταστάσεων ανάμεσα στις χημικές αντιδράσεις. Στους υπολογισμούς, η μελέτη της κβαντικής χημείας περιλαμβάνει χρήση ημι-εμπειρικών και άλλων μεθόδων που βασίζονται στις αρχές της κβαντικής μηχανικής, και αντιμετωπίζουν προβλήματα σχετικά με τον χρόνο. Πολλές μελέτες κβαντικής χημείας υποθέτουν ότι οι πυρήνες βρίσκονται σε κατάσταση ηρεμίας και πολλοί υπολογισμοί περιέχουν επαναληπτικές μεθόδους.[33] Οι στόχοι της κβαντικής χημείας περιλαμβάνουν την αύξηση της ακρίβειας των αποτελεσμάτων για μικρά μοριακά συστήματα, και αύξηση του μεγέθους των μορίων που μπορούν να επεξεργαστούν. Ο χρόνος υπολογισμού αυξάνεται όσο αυξάνεται ο αριθμός των ατόμων.

3.4.2 Προγράμματα

Πίνακας 3.3: Κβαντική Χημεία

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
Abinit	Επιτρέπει τον υπολογισμό της ολικής ενέργειας, πυκνότητα φόρτισης και ηλεκτρονική δομή του συστήματος αποτελούμενο από ηλεκτρόνια και πυρήνες μέσα στο DFT	1.3-2.7x	NAI
ADF	Λογισμικό Θεωρίας Λειτουργικής Πυκνότητας (DFT) που επιτρέπει υπολογισμούς ηλεκτρονικών δομών πρώτης αρχής	1.5-2x	NAI
BigDFT	Εκτελεί θεωρία λειτουργικής πυκνότητας λύνοντας τις εξισώσεις Kohn-Sham περιγράφοντας τα ηλεκτρόνια σε ένα υλικό	2-25x	NAI
CP2K	Πρόγραμμα που εκτελεί ατομιστικές και μοριακές εξομοιώσεις στερεών καταστάσεων, υγρών, μοριακών και βιολογικών συστημάτων	2-7x	NAI
GAMESS-UK	Πρόγραμμα για απόδοση υπολογισμών γενικού σκοπού (SCF-gradient, DFT-gradient, MCCF-gradient)	8x	NAI
GAMESS-US	Εργαλείο υπολογιστικής χημείας που χρησιμοποιείται για εξομίωση ατομικής και μοριακής ηλεκτρονικής δομής	1.3-2.9x	NAI
Gaussian (Σε ανάπτυξη)	Προφητεύει ενέργειες μοριακές δομές, και δονούμενες συχνότητες μοριακών συστημάτων	-	NAI
GPAW	Πλέγμα πραγματικού χώρου DFT κώδικα γραμμένο σε C και Python	8x	NAI
LATTE	Υπολογισμούς πυκνότητας πίνακα	-	NAI
MOLCAS	Μέθοδοι για υπολογισμό γενικών ηλεκτρονικών δομών σε μοριακά συστήματα σε καταστάσεις εδάφους και ενθουσιώδης καταστάσεις	1.1x	NAI
MOPAC2013	Ημι-εμπειρική κβαντική χημεία	2x	OXI
NWChem	Υπολογισμοί	3-10x	NAI
Octopus	Χρησιμοποιείται για εικονικό πειραματισμό και υπολογισμούς κβαντικής χημείας	1.5-8x	-
Q-CHEM	Πακέτο υπολογιστικής χημείας σχεδιασμένο για συμπλέγματα υψηλής υπολογιστικής απόδοσης	8-14x	-

Συνέχεια πίνακα 3.3			
Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
QUICK	Είναι ένα πακέτο λογισμικού κβαντικής χημείας με υποστήριξη επιτάχυνσης GPU	10-100x	NAI
TeraChem	Λογισμικό κβαντικής χημείας σχεδιασμένο για να εκτελείται σε μια NVIDIA GPU	44-650x	NAI

3.5 Δυναμική Ρευστών

Στην φυσική, η δυναμική ρευστών είναι ένα υποσύνολο της μηχανικής ρευστών που ασχολείται με την ροή ρευστών - την φυσική επιστήμη των ρευστών (υγρών και αερίων) σε κίνηση. Έχει ακόμα περισσότερα υποσύνολα, όπως η αεροδυναμική (η μελέτη του αέρα και άλλων αερίων σε κίνηση), υδροδυναμική (η μελέτη των υγρών σε κίνηση). Η δυναμική ρευστών έχει μεγάλο εύρος εφαρμογών, συμπεριλαμβανομένου υπολογισμό ισχύων και στιγμών σε ένα αεροπλάνο, έρευνα της ροής μάζας του πετρελαίου μέσα σε σωλήνες μεταφοράς, πρότυπα πρόβλεψης καιρικών φαινομένων, κατανόηση ενός nebulae στο διαγαλαξιακό διάστημα, και μοντελοποίηση εκρήξεων. Μερικές από τις αρχές της δυναμικής ρευστών χρησιμοποιούνται ακόμα και στον σχεδιασμό έργων κυκλοφορίας, όπου η κυκλοφορία θεωρείται ένα συνεχόμενο ρευστό.[34] [35]

Οι δυναμικές ρευστών παρέχουν μια συστηματική σχεδίαση, που περιλαμβάνει εμπειρικούς νόμους από μετρήσεις ροής και χρησιμοποιείται για να λύσει πρακτικά προβλήματα. Η λύση στα προβλήματα δυναμικής ρευστών συνήθως περιλαμβάνει υπολογισμούς διάφορων ιδιοτήτων των ρευστών, όπως η ταχύτητα, πίεση, πυκνότητα, και θερμοκρασία, όπως αυτά αντιδρούν μέσα στον χώρο και στον χρόνο.

3.5.1 Τυρβώδης ροή

Η τυρβώδης ροή είναι μια ροή που χαρακτηρίζεται από επανάληψη, και τυχαία συμπεριφορά. Η ροή στην οποία η ταραχή δεν εμφανίζεται ονομάζεται ελασματώδης. Μερικά από τα φαινόμενα μπορεί να εμφανίζονται και στην ελασματώδη ροή. Μαθηματικά, η τυρβώδης ροή εκπροσωπείται μέσω μιας αποσύνθεσης Reynolds, στην οποία η ροή αναλύεται στο άθροισμα ενός μέσου στοιχείου, και ενός στοιχείου άναρχης κατάστασης.

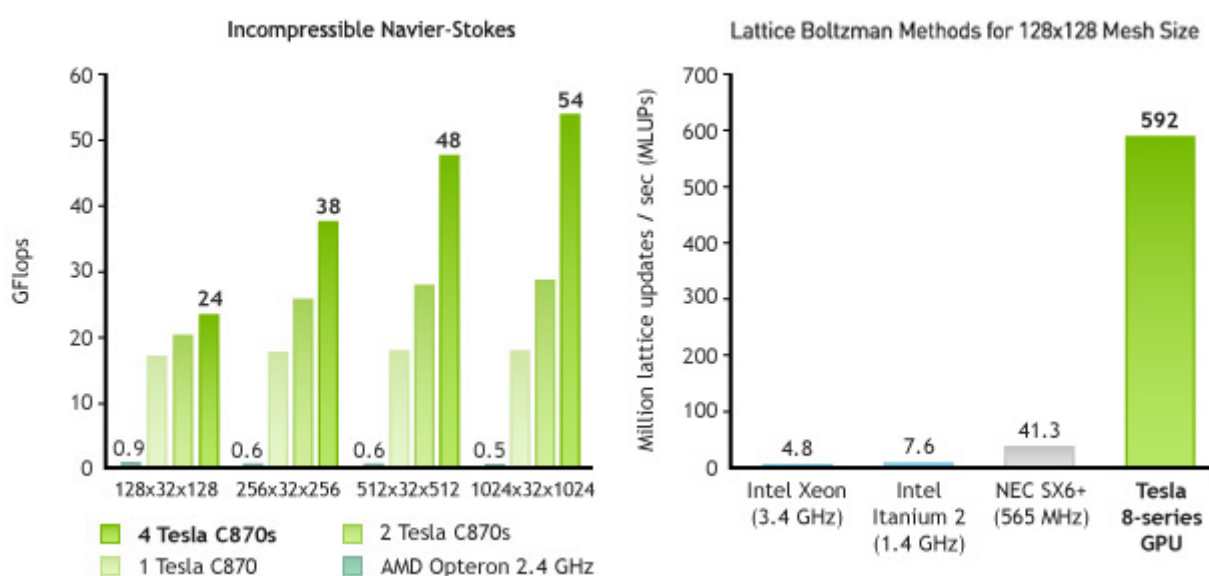
Πιστεύεται ότι η τυρβώδης ροή μπορεί να περιγραφεί μέσω εξισώσεων Navier-Stokes, εξομοίωση άμεσης αρίθμησης (DNS), και περιορίζεται από την ισχύ του υπολογιστή που χρησιμοποιείται για την εξομοίωση, όσο και από την αποδοτικότητα του αλγορίθμου. Τα αποτελέσματα του DNS πολλές φορές συμφωνούν με τα πειραματικά δεδομένα για κάποιες ροές.[36] [37]

Οι περισσότερες ροές, έχουν αριθμούς Reynolds πολύ υψηλούς για την χρήση του DNS (έχοντας υπόψη την υπολογιστική δύναμη για τις επόμενες δεκαετίες). Το οποιοδήποτε αεροσκάφος που μεταφέρει ανθρώπους, με ταχύτητα πάνω από 72 χιλιόμετρα την ώρα, είναι αρκετά εκτός ορίων για την χρήση εξομοίωσης DNS ($Re = 4$ εκατομμύρια). Τα φτερά του αεροσκάφους

έχουν αριθμούς Reynolds πάνω από 40 εκατομμύρια. Για να λυθούν αυτά τα προβλήματα, οι τυρβώδεις ροές είναι αναγκαιότητα για το μεσοπρόθεσμο μέλλον. Οι Navier-Stokes εξισώσεις σε συνδυασμό με τα μοντέλα ταραχών, παρέχουν ένα μοντέλο των αποτελεσμάτων της τυρβώδους ροής. Άλλη μια υποσχόμενη μεθοδολογία είναι η μεγάλη εξομίωση eddy (LES).

3.5.2 GPU

Αρκετά σχέδια υπάρχουν τα τελευταία χρόνια σε μοντέλα Navier-Stokes και οι μέθοδοι Lattice Boltzman έχουν αποδείξει πολύ μεγάλες επιταχύνσεις με χρήση υπολογισμού GPGPU. Μερικά από τα αποτελέσματα φαίνονται στο παρακάτω διάγραμμα. Υπάρχουν επίσης πολλά έργα σε μοντελοποίηση καιρικών φαινομένων και μοντελοποίηση ωκεανών με χρήση GPU.



Σχήμα 3.8: Υπολογισμός δυναμικής ρευστών σε GPU

3.5.3 Tsunami

Το tsunami, γνωστό και ως σεισμικό θαλάσσιο κύμα, είναι μια σειρά από κύματα νερού που προκαλούνται από το εκτόπισμα μεγάλου όγκου νερού, συνήθως ενός ωκεανού ή μιας μεγάλης λίμνης. Οι σεισμοί, οι ηφαιστειακές δραστηριότητες, και άλλες υποθαλάσσιες εκρήξεις (συμπεριλαμβανομένου πυροδοτήσεων πυρηνικών συσκευών), συγκρούσεις από μετεωρίτες, ή και άλλες διαταράξεις πάνω ή κάτω από το νερό έχουν την δυνατότητα να προκαλέσουν ένα tsunami.

Τα κύματα tsunami δεν μοιάζουν με τα συνηθισμένα θαλάσσια κύματα, γιατί το μήκος κύματος είναι πολύ μεγαλύτερο. Αντί να εμφανίζεται σαν μια θραύση κυμάτων, το tsunami αρχικά μπορεί να μοιάζει με έναν τύπο παλιρροιακού κύματος, κύματα μεγάλου ύψους που δημιουργούνται ειδικά από σεισμικές δραστηριότητες, και για αυτόν τον λόγο ονομάζεται σεισμικό θαλάσσιο κύμα. Ο όρος αυτός προτιμάται από τους γεωλόγους και ωκεανογράφους αν και δεν έχει ακριβώς την σημασία της λέξης tsunami. Ο όρος

tsunami προέρχεται από τα δύο γιαπωνέζικα γράμματα 津(tsu) που σημαίνει λιμάνι, και 波(nami) που σημαίνει κύμα. Ο όρος πήρε το όνομα του από τους ψαράδες, όπου ενώ βρίσκονταν στην θάλασσα και δεν συναντούσαν κάποιο φαινόμενο, όταν γυρνούσαν στο λιμάνι βρίσκανε το χωρίο τους κατεστραμμένο από το tsunami. Ακόμα και ο όρος σεισμικό θαλάσσιο κύμα δεν είναι πολύ ακριβής, καθώς άλλες δυνάμεις εκτός από τους σεισμούς μπορούν να δημιουργήσουν τέτοια κύματα, μεταξύ τους οι ηφαιστειακές δραστηριότητες, καθίζηση γης στον ωκεανό, ακόμα και η απότομη αλλαγή πίεσης. [38]

Τα tsunami γενικά αποτελούνται από μια σειρά κυμάτων με περιόδους που εκτείνονται από λεπτά μέχρι ώρες, και φτάνουν σε μια μορφή τρένου κυμάτων. Κύματα ύψους δεκάδων μέτρων μπορούν να δημιουργηθούν από μεγάλα γεγονότα. Αν και η πρόσκρουση των tsunami περιορίζεται στις ακτές, η καταστροφική τους δύναμη μπορεί να είναι τεράστια και επηρεάζουν ολοκληρωτικά κόλπους οκεανών. Το 2004 το tsunami στον Ινδικό ωκεανό ήταν ανάμεσα στα πιο καταστροφικά φυσικά φαινόμενα στην ιστορία του ανθρώπου, με τουλάχιστον 290.000 νεκρούς και αγνοούμενους σε 14 χώρες που συνορεύουν με τον Ινδικό ωκεανό.

Χαρακτηριστικά

Τα tsunami μπορούν να προκαλέσουν ζημιές με δύο μηχανισμούς: την ισχύ ενός τοίχου νερού που ταξιδεύει σε μεγάλη ταχύτητα, και την ισχύ του μεγάλου όγκου νερού όπου μεταφέρει συντρίμια στην ξηρά, ακόμα και με κύματα που δεν φαίνονται να είναι μεγάλα.

Ενώ τα καθημερινά κύματα έχουν μήκος κύματος περίπου 100 μέτρα και ύψος περίπου 2 μέτρα, ένα tsunami στον βαθύ ωκεανό έχει πολύ μεγαλύτερο μήκος κύματος έως και 200 χιλιόμετρα. Ένα τέτοιο κύμα ταξιδεύει με πολύ περισσότερο από 800 χιλιόμετρα την ώρα, αλλά λόγω του μεγάλου μήκους κύματος χρειάζεται 20-30 λεπτά για να ολοκληρώσει ένα κύκλο, και έχει πλάτος περίπου 1 μέτρο. Αυτό κάνει τα tsunami δύσκολο να εντοπιστούν σε μεγάλο βάθος.

Όταν το tsunami πλησιάζει την ακτή και τα νερά γίνονται πιο ρηχά, το κύμα συμπιέζεται και η ταχύτητα του μειώνεται σε περίπου 80 χιλιόμετρα την ώρα. Το μήκος κύματος του μειώνεται σε λιγότερο από 20 χιλιόμετρα, αλλά το πλάτος του αυξάνεται εξαιρετικά. Λόγω του ότι το κύμα έχει ακόμα την ίδια μεγάλη περίοδο, το tsunami μπορεί να χρειαστεί ολόκληρα λεπτά για να φτάσει το μέγιστο ύψος. Με εξαίρεση τα πολύ μεγάλα tsunami, το κύμα δεν σπάει, αλλά εμφανίζεται σαν ένα γρήγορα αυξανόμενο παλιρροιακό κύμα. Το 80% των tsunami συναντώνται στον Ειρηνικό ωκεανό, αλλά μπορούν να εμφανιστούν σε όποια περιοχή έχει μεγάλο όγκο νερού, συμπεριλαμβανομένου των μεγάλων λιμνών.

Προειδοποιήσεις και προβλέψεις

Κάθε κύμα, έχει μια θετική και μια αρνητική κορυφή. Στην περίπτωση του tsunami, μπορεί να φτάσει οποιαδήποτε από τις δύο. Αν η θετική κορυφή φτάσει πρώτη στην στεριά, τότε μια ξαφνική πλημμύρα θα εμφανιστεί στην στεριά. Αν όμως η αρνητική κορυφή φτάσει πρώτη στην στεριά, τότε

θα δημιουργηθεί μια υποχώρηση του νερού, που θα εμφανίσει την βυθισμένη επιφάνεια. Αυτή η επιφάνεια μπορεί να φτάσει εκατοντάδες μέτρα. Ένα τυπικό κύμα tsunami έχει περίοδο περίπου 12 λεπτών. Αυτό σημαίνει ότι αν η αρνητική κορυφή φτάσει πρώτη στην στεριά, η θάλασσα θα υποχωρήσει, εμφανίζοντας την βυθισμένη επιφάνεια μέσα σε 3 λεπτά. Μετά από 6 λεπτά τα κύματα του tsunami μετατρέπονται σε θετική κορυφή, όπου η θάλασσα επανέρχεται και δημιουργεί καταστροφές στην στεριά. Η διαδικασία αυτή επαναλαμβάνεται μόλις φτάσει το επόμενο κύμα.

Το tsunami δεν μπορεί να προβλεφθεί με ακρίβεια, ακόμα και αν το μέγεθος και η περιοχή ενός σεισμού είναι γνωστή. Οι γεωλόγοι, ωκεανογράφοι, και σεισμολόγοι αναλύουν τους σεισμούς και βασισμένοι σε πολλούς παράγοντες προειδοποιούν ή όχι για περιπτώσεις tsunami. Όμως, υπάρχουν προειδοποιητικές πινακίδες σε περιοχές που είναι ευπαθείς σε tsunami, και τα αυτοματοποιημένα συστήματα παρέχουν προειδοποιήσεις αμέσως μετά από ένα σεισμό, ώστε να υπάρχει αρκετός χρόνος για να σωθούν ζωές. Ένα από τα πιο πετυχημένα συστήματα χρησιμοποιεί αισθητήρες χαμηλής πίεσης, που συνεχώς ελέγχουν την πίεση του νερού.

3.5.4 Προγράμματα

Πίνακας 3.4: Δυναμικές Ρευστών

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
Altair AcuSolve	Λογισμικό CFG γενικού σκοπού		NAI
ANSYS Fluent	Λογισμικό CFG γενικού σκοπού		NAI
Autodesk Moldflow	Λογισμικό ψεκασμού με πλαστικό καλούπι		NAI
Barracuda	Λογισμικό προσομοίωσης υδροποιημένου στρώματος		NAI
Fluidyna Culises for OpenFOAM	Βιβλιοθήκη για λογισμικά CFG γενικού σκοπού		NAI
Fluidyna LBultra	Λογισμικό CFG γενικού σκοπού		NAI
Prometech Particleworks	Βασισμένο σε σωματίδια λογισμικό CFD		NAI
Turbostream Ltd.	Λογισμικό CFD για ροές μηχανών τουρμπίνας		NAI
Vratis ARAEL	Λογισμικό CFG γενικού σκοπού βασισμένο στο FVM με υποστήριξη OpenFOAM		NAI
Vratis SpeedIT extreme for OpenFOAM	Βιβλιοθήκη για λογισμικά CFG γενικού σκοπού		NAI
DualSPHysics	Λογισμικό CFD βασισμένο σε SPH		NAI
FEFLO (GMU-Lohner)	Λογισμικό CFG γενικού σκοπού για συμπιεζόμενες και αποσυμπιεζόμενες ροές		NAI
NASA FUN3D	Λογισμικό CFG γενικού σκοπού		NAI

Συνέχεια πίνακα 3.4			
Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
S3D(Sandia and OakRidge NL)	Άμεσος αριθμητικός λύτης (DNS) για τυρβώδη καύσεις		NAI
SD++ (Stanford-Jameson)	Λογισμικό CFG γενικού σκοπού για συμπιεζόμενες ροές.		NAI

3.6 Ψυχαγωγία

3.6.1 Εισαγωγή

Μια μηχανή φυσικής είναι ένα πρόγραμμα υπολογιστή που παρέχει εξομοίωση συγκεκριμένων συστημάτων φυσικής, όπως δυναμική άκαμπτων σωμάτων, ανίχνευση σύγκρουσης, δυναμική υγρών, για χρήση σε πεδία όπως τα γραφικά υπολογιστών, παιχνίδια, κινούμενα σχέδια, ταινίες. Μια από τις κύριες χρήσεις τους είναι στα παιχνίδια υπολογιστών, στην οποία περίπτωση η εξομοίωση γίνεται σε πραγματικό χρόνο. Ο όρος χρησιμοποιείται γενικότερα για να περιγράψει οποιοδήποτε σύστημα λογισμικού που εξομοιώνει φυσικά φαινόμενα, όπως επιστημονικές εξομοιώσεις υψηλής απόδοσης.

Οι μηχανές φυσικής έχουν χρησιμοποιηθεί αρκετά στους υπερ-υπολογιστές από την δεκαετία του '80 για να εκτελέσουν μοντελοποίηση δυναμικών υγρών, όπου αναθέτουμε διανύσματα ισχύος σε σωματίδια, για να δείξουμε την κυκλοφορία. Λόγω των υψηλών απαιτήσεων σε ταχύτητα και ακρίβεια, ειδικοί επεξεργαστές δημιουργήθηκαν που είναι γνωστοί ως επεξεργαστές διανυσμάτων για να επιταχύνουν τους υπολογισμούς. Οι τεχνικές μπορούν να χρησιμοποιηθούν για να μοντελοποιήσουν πρότυπα καιρού για την πρόβλεψη καιρού, δεδομένα σήραγγας αέρα για σχεδιασμό αεροπλάνων και υποβρυχίων, και ανάλυση θερμικής απόδοσης για καλύτερο σχεδιασμό ψηκτρών για επεξεργαστές. Φυσικά μεγάλο ρόλο παίζει η ακρίβεια των υπολογισμών, αφού μικρές αποκλίσεις μπορούν να αλλάξουν δραστικά τα αποτελέσματα των υπολογισμών. Οι κατασκευαστές ελαστικών χρησιμοποιούν εξομοιώσεις φυσικής για να μελετήσουν πώς οι καινούριοι τύποι ελαστικών θα αποδίδουν σε συνθήκες βρεγμένου και στεγνού οδοστρώματος, χρησιμοποιώντας καινούρια υλικά και κάτω από διαφορετικές συνθήκες βάρους.[39]

Υπάρχουν γενικά δύο τύποι μηχανών φυσικής. Οι πραγματικού χρόνου, και οι υψηλής ακρίβειας. Οι υψηλής ακρίβειας απαιτούν περισσότερη υπολογιστική δύναμη για να υπολογίσουν φυσικά φαινόμενα με ακρίβεια και χρησιμοποιούνται συνήθως από επιστήμονες αλλά και σε κινούμενα σχέδια. Οι πραγματικού χρόνου - χρησιμοποιούνται σε παιχνίδια υπολογιστών και σε άλλες μορφές διαδραστικού υπολογισμού - χρησιμοποιούν απλοποιημένους υπολογισμούς με μειωμένη ακρίβεια ώστε να επιτρέπουν στο παιχνίδι να αντιδράει σε αποδεκτό ρυθμό για την εμπειρία χρήσης.[40]

3.6.2 Παιχνίδια

Γενικά

Στα περισσότερα παιχνίδια, η ταχύτητα των επεξεργασιών και η εμπειρία χρήσης είναι πιο σημαντικά από την ακρίβεια της εξομοίωσης. Αυτό μας οδηγεί σε σχεδιασμούς μηχανών φυσικής που παράγουν αποτελέσματα σε πραγματικό χρόνο αλλά αντιγράφουν φυσικά φαινόμενα μόνο για απλές περιπτώσεις. Τις περισσότερες φορές, η εξομοίωση είναι σχεδιασμένη να παρέχει μια φαινομενικά σωστή εκτίμηση, παρά απόλυτη ακρίβεια. Όμως μερικές μηχανές, απαιτούν μεγαλύτερη ακρίβεια σε σκληρές μάχης ή σε παιχνίδια τύπου παζλ. Οι κινήσεις χαρακτήρων στο παρελθόν χρησιμοποιούσαν φυσική άκαμπτων σωμάτων γιατί είναι γρηγορότερο και πιο εύκολο να

υπολογιστεί, όμως τα τελευταία χρόνια τα παιχνίδια και οι ταινίες έχουν αρχίσει να χρησιμοποιούν φυσική μαλακών σωμάτων. Αυτού του τύπου η εξομοίωση χρησιμοποιείται επίσης για εφέ σωματιδίων, κίνηση υγρών και υφασμάτων. Μια μορφή εξομοίωσης δυναμικής υγρών χρησιμοποιείται για να εξομοιώσει νερό και άλλα υγρά αλλά και την ροή της φωτιάς και του καπνού στον αέρα.[41]



Σχήμα 3.9: Λογότυπο της μηχανή φυσικής havok

Ανίχνευση σύγκρουσης

Η ανίχνευση σύγκρουσης συνήθως αναφέρεται στο υπολογιστικό πρόβλημα της ανίχνευσης της διασταύρωσης δύο η περισσότερων αντικειμένων. Αν και το θέμα έχει σχέση περισσότερο με την χρήση του στα παιχνίδια και σε άλλες εξομοιώσεις φυσικής, έχει και χρήσεις στην ρομποτική. Εκτός από την ανίχνευση του αν δύο αντικείμενα έχουν συγκρουστεί, τα συστήματα ανίχνευσης μπορούν να υπολογίσουν τον χρόνο της σύγκρουσης (Time Of Impact), και να αναφέρουν ένα σύνολο από σημεία διασταύρωσης. Η αντίδραση σύγκρουσης είναι η εξομοίωση του τι συμβαίνει όταν ανιχνευθεί μια σύγκρουση.

3.6.3 Υλοποιήσεις

Μονάδα Επεξεργασίας Φυσικής

Η μονάδα επεξεργασίας φυσικής (PPU) είναι ένας μικροεπεξεργαστής αποκλειστικά σχεδιασμένος για να χειρίζεται τους υπολογισμούς φυσικής, ειδικά σε μηχανές φυσικής των παιχνιδιών υπολογιστών. Η ιδέα είναι ότι αυτοί οι ειδικοί επεξεργαστές ελαφρύνουν το φόρτο εργασίας των CPUs, όπως μια κάρτα γραφικών εκτελεί υπολογισμούς γραφικών. Ο όρος αρχικά δημιουργήθηκε από την εταιρία Ageia για να περιγράψει τους επεξεργαστές PhysX στους καταναλωτές. Η NVIDIA απέκτησε την Ageia Technologies το 2008 και συνεχίζει να αναπτύσσει την πλατφόρμα PhysX και στο υλικό αλλά και στο λογισμικό. Από την έκδοση 2.8.3, η υποστήριξη για κάρτες PPU σταμάτησε, και δεν κατασκευάζονται πλέον.

Υπολογισμοί γενικής χρήσης σε GPUs

Η επιτάχυνση υλικού για υπολογισμούς φυσικής χρησιμοποιείται πλέον από τις GPUs που υποστηρίζουν υπολογισμό γενικής χρήσης. Η εκτέλεση φυσικών υπολογισμών σε GPUs είναι συνήθως αρκετά πιο γρήγορη από ότι σε μια CPU, έτσι η απόδοση των παιχνιδιών βελτιώνεται και η ροή εικόνας μπορεί να είναι πολύ πιο γρήγορη. Όμως η χρήση υπολογισμών φυσικής σε

ένα παιχνίδι δημιουργεί επιπλέον φόρτο στην GPU. Έτσι, η χρήση ξεχωριστής μονάδας επεξεργασίας γραφικών για εκτελέσεις υπολογισμών φυσικής μπορεί να αποδώσει τα βέλτιστα αποτελέσματα. Το PhysX εκτελείται γρήγορα και αποδίδει μεγαλύτερο ρεαλισμό όταν εκτελείται στην GPU, αποφέροντας 10-20 φορές περισσότερα εφέ και οπτική πιστότητα απο ότι οι υπολογισμοί φυσικής που εκτελούνται σε μια κεντρική μονάδα επεξεργασίας τελευταίας τεχνολογίας. Το PhysX χρησιμοποιεί ετερογενή υπολογισμό για να αποδώσει την καλύτερη εμπειρία χρήσης. Καθώς το παιχνίδι εκτελείται, το σύστημα PhysX εκτελεί μέρη της τεχνολογίας στην CPU αλλά και άλλα μέρη στην GPU. Αυτό γίνεται ώστε να χρησιμοποιείται αποδοτικά το υλικό του υπολογιστή ώστε να παρέχουν την καλύτερη δυνατή εμπειρία στον χρήστη. Το πιο σημαντικό, είναι ότι η τεχνολογία PhysX μπορεί να κλιμακώνεται με την χρήση GPU, σε αντίθεση με άλλες ανταγωνιστικές υλοποιήσεις φυσικής.

Σύγκριση

Οι πιο σημαντικές μηχανές γραφικών που χρησιμοποιούνται σήμερα είναι οι παρακάτω:

- PhysX - Είναι μια μηχανή φυσικής πραγματικού χρόνου, από την NVIDIA. Είναι κλειστού κώδικα και χρησιμοποιείται σε πολλά παιχνίδια υπολογιστών και κονσολών. Υποστηρίζει μεγάλο αριθμό συσκευών.
- Havok - Είναι μια μηχανή φυσικής που χρησιμοποιείται σε πολλά παιχνίδια υπολογιστών.
- ODE - Είναι μια μηχανή φυσικής που υποστηρίζει κυρίως ανίχνευση συγκρούσεων, δυναμικές άκαμπτων σωμάτων. Αποτελεί ανοιχτό και ελεύθερο λογισμικό. Έχει χρησιμοποιηθεί σε πολλά παιχνίδια και εφαρμογές. Αποτελεί δημοφιλή επιλογή για εφαρμογές εξομοίωσης ρομποτικής.
- Newton Game Dynamics - Είναι μια μηχανή φυσικής ανοιχτού κώδικα που εξομοιώνει άκαμπτα σώματα σε παιχνίδια και άλλες εφαρμογές πραγματικού χρόνου.
- Bullet - Είναι μια μηχανή φυσικής που εξομοιώνει ανίχνευση συγκρούσεων, δυναμική άκαμπτων και μαλακών σωμάτων. Χρησιμοποιείται σε παιχνίδια υπολογιστών αλλά και για οπτικά εφέ σε ταινίες. Η βιβλιοθήκη της bullet physics είναι ελεύθερη και ανοιχτού κώδικα κάτω από την άδεια zlib.

3.7 Εικόνα και Βίντεο

3.7.1 Ιχνογράφηση ακτίνας

Πραγματικότητα

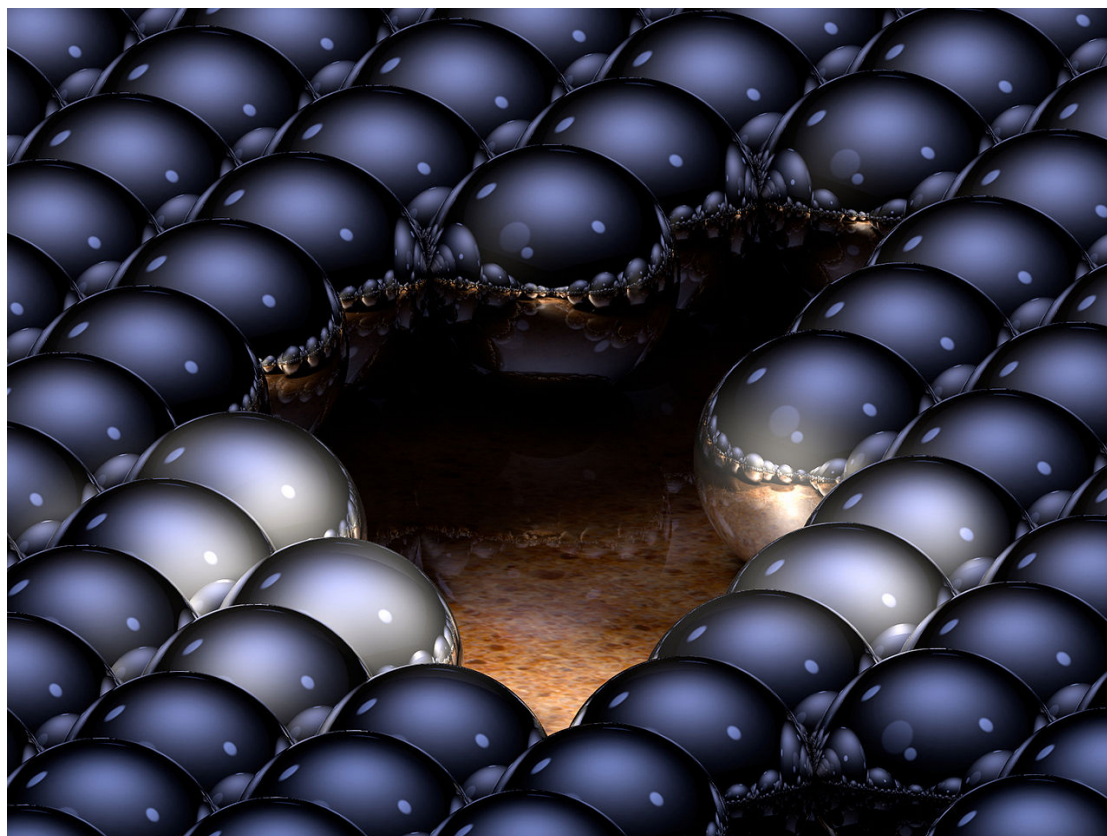
Στην φύση, μια πηγή φωτός εκπέμπει μια ακτίνα φωτός όταν ταξιδεύει, προς μια επιφάνεια που εμποδίζει την πρόοδο της. Μπορούμε να σκεφτούμε την "ακτίνα" σαν μια ροή φωτονίων που ταξιδεύουν προς το ίδιο μονοπάτι. Στο απόλυτο κενό, αυτή η ακτίνα θα είναι μια ευθεία γραμμή, αν αγνοήσουμε την δράση της σχετικότητας. Ένας συνδυασμός της απορρόφησης, ανάκλασης, διάθλασης και φθορισμού μπορεί να σχηματιστεί από αυτήν την ακτίνα. Μια επιφάνεια μπορεί να απορροφήσει μέρος της ακτίνας, που θα έχει ως αποτέλεσμα την μείωση της έντασης της ανακλώμενης ακτίνας. Μπορεί επίσης να δημιουργήσει αντανάκλαση όλης ή μέρος της ακτίνας, προς μία ή παραπάνω κατευθύνσεις. Αν η επιφάνεια έχει διάφανες ιδιότητες, θα διαθλάσει την ακτίνα στην ίδια και σε άλλη μια κατεύθυνση, και θα απορροφήσει μόνο μέρος της ακτίνας, ίσως αλλάζοντας και το χρώμα της. Έπειτα, οι ακτίνες μπορεί να φτάσουν σε επιπλέον επιφάνειες, όπου οι ιδιότητες τους θα επηρεάσουν ξανά την πρόοδο τους. Πολλές από αυτές τις ακτίνες, ταξιδεύουν με τρόπο που φτάνουν στο μάτι μας, επιτρέποντας μας να δούμε την σκηνή και να συνεισφέρουν στην τελική εικόνα.

Γραφικά υπολογιστών

Στα γραφικά των υπολογιστών, η ιχνογράφηση ακτίνας είναι μια τεχνική για παραγωγή μιας εικόνας που χρησιμοποιεί ιχνογράφηση στο μονοπάτι του φωτός μέσα στον χώρο μιας εικόνας και εξομοιώνει τα εφέ τις επίδρασης του στα εικονικά αντικείμενα. Η τεχνική είναι δυνατόν να παράγει πολύ υψηλό βαθμό εικονικού ρεαλισμού, συνήθως μεγαλύτερο από τις τυπικές μεθόδους, αλλά με πολύ μεγαλύτερο υπολογιστικό κόστος. Αυτό κάνει την ιχνογράφηση ακτίνας πιο χρήσιμη για εφαρμογές όπου η εικόνα μπορεί να δημιουργηθεί σε βάθος του χρόνου, όπως οι σταθερές εικόνες και τα εφέ ταινιών, ενώ είναι συνήθως λιγότερο επιθυμητή σε εφαρμογές πραγματικού χρόνου, όπως τα παιχνίδια όπου η ταχύτητα είναι σημαντική. Η ιχνογράφηση ακτίνας μπορεί να εξομοιώσει ένα μεγάλο αριθμό από οπτικά εφέ, όπως η αντανάκλαση και η διάθλαση, διασκόρπιση και φαινόμενα διασποράς.

Ιστορία

Ο πρώτος αλγόριθμος ιχνογράφησης ακτίνας παρουσιάστηκε από τον Arthur Appel το 1968. Ο αλγόριθμος ονομάστηκε από τότε διάχυση ακτίνας. Ένα σημαντικό πλεονέκτημα της ιχνογράφησης ακτίνας προς τους παραδοσιακούς scanline αλγορίθμους είναι η δυνατότητα να αντιμετωπίζει επιφάνειες όπως οι κώνοι και οι σφαίρες. Η επόμενη μεγάλη ανακάλυψη ήρθε από τον Turner Whitted το 1979. Ενώ οι παλιοί αλγόριθμοι υπολόγιζαν μόνο την ακτίνα έως την στιγμή που θα έφτανε στο πρώτο εμπόδιο, ο Whittman συνέχισε την διαδικασία. Όταν μια ακτίνα φτάνει σε μια επιφάνεια, μπορεί να παράγει ως και τρεις τύπους ακτίνων: αντανάκλαση,



Σχήμα 3.10: Κάθε ακτίνα χρησιμοποιεί αντανάκλαση έως και 16 φορές

διάθλαση, σκίαση. Αυτή η επαναληπτική ιχνογράφηση, έδωσε περισσότερο ρεαλισμό στις εικόνες.

Μειονεκτήματα

Ένα σοβαρό μειονέκτημα της ιχνογράφησης ακτίνας είναι η απόδοση. Οι τυπικοί αλγόριθμοι (scanline, κ.α) χρησιμοποιούν στοιχεία συνοχής για να διαμοιράσουν τους υπολογισμούς μεταξύ των εικονοστοιχείων, ενώ η ιχνογράφηση ακτίνας συνήθως ξεκινάει μια καινούρια διαδικασία, θεωρώντας την κάθε ακτίνα σαν ξεχωριστή οντότητα. Αυτός ο διαχωρισμός όμως δίνει και πλεονεκτήματα, όπως η ικανότητα να έχουμε περισσότερες ακτίνες για να δημιουργήσουμε anti-aliasing διαστήματος και να βελτιώσουμε την ποιότητα της εικόνας όπου χρειάζεται.

Αν και χειρίζεται τα οπτικά εφέ όπως η διασκόρπιση με ακρίβεια, η παραδοσιακή ιχνογράφηση ακτίνας δεν είναι πάντα φωτορεαλιστική. Ο πραγματικός φωτορεαλισμός επιτυγχάνεται όταν η εξίσωση ανταπόδοσης προσεγγίζεται πολύ κοντά ή εκτελείται ολοκληρωτικά. Η εκτέλεση της εξίσωσης ανταπόδοσης δίνει αληθινό φωτορεαλισμό, καθώς η εξίσωση περιγράφει κάθε φυσικό εφέ ή ροή φωτός. Συνήθως όμως, λόγω υπολογιστικών περιορισμών δεν είναι δυνατόν να συμβεί.

Ο ρεαλισμός των μεθόδων ανταπόδοσης μπορεί να αξιολογηθεί ως προσέγγιση της εξίσωσης. Η ιχνογράφηση ακτίνας, αν περιοριστεί στον αλγόριθμο του Whitted δεν είναι απαραίτητα η πιο ρεαλιστική. Οι μέθοδοι που ιχνογραφούν ακτίνες, αλλά περιέχουν επιπλέον τεχνικές (καταγραφή φω-

τονίων, ιχνογράφηση μονοπατιού), δίνουν μεγαλύτερη ακρίβεια εξομοίωσης του φωτισμού στον πραγματικό κόσμο.

Είναι επίσης δυνατό να προσεγγίσουμε την εξίσωση χρησιμοποιώντας διάχυση ακτίνας με διαφορετικό τρόπο από ότι χρησιμοποιείται συνήθως στην ιχνογράφηση ακτίνας. Για λόγους απόδοσης, οι ακτίνες μπορούν να συμπλεχθούν σε σχέση με την κατεύθυνση τους, με ειδικό υλικό να υπολογίζει τις ακτίνες.

Προγράμματα

Η μαζική παράλληλη υπολογιστική δύναμη των GPUs ταιριάζει τέλεια με την παράλληλη φύση της ιχνογράφησης ακτίνας, και βελτιώνει αισθητά τον χρόνο ανταπόδοσης για διάφορες βιομηχανίες, χρησιμοποιώντας διάφορους τύπους τεχνικών. Οι GPUs είναι ένα απαραίτητο εργαλείο για τις εταιρίες ανταπόδοσης. Παρακάτω βλέπουμε μια λίστα από προγράμματα που χρησιμοποιούν αυτήν την υπολογιστική δύναμη για να επιταχύνουν αισθητά τις υλοποιήσεις τους και να ορίσουν νέες διαδραστικές δυνατότητες.



Σχήμα 3.11: Ανταπόδοση φωτορεαλιστικής εικόνας σε 3,5 λεπτά με την χρήση του Furryball

- Nvidia Optix - Είναι ένα εργαλείο ιχνογράφησης ακτίνας για προγραμματιστές υπολογιστών που δημιουργούν εφαρμογές που δίνουν γρήγορα αποτελέσματα. Σε αντίθεση με υλοποιήσεις με υπαγορευμένη αισθητική, ή που περιορίζουν τις δομές δεδομένων τους ή και της γλώσσας ανταπόδοσης τους, η μηχανή Optix είναι υπερβολικά γενικευμένη, δίνοντας την δυνατότητα στους προγραμματιστές να επιταχύνουν όλη διεργασία ιχνογράφησης ακτίνας θέλουν και να την εκτελέσουν σε μονάδες επεξεργασίας γραφικών
- Arion - Είναι ένα προϊόν επόμενης γενιάς που χρησιμοποιείται για εξομοίωση φωτός με τεχνολογία CUDA για να επιταχύνει την ανταπόδοση σε μια

ή πολλές μονάδες επεξεργασίας γραφικών. Το Arion είναι ένας εξομοιωτής φωτός βασισμένος σε φυσικά φαινόμενα, ενώ μπορεί να χρησιμοποιήσει CPUs και GPUs για παραγωγή ανταπόδοσης υψηλών απαιτήσεων.

- **Furryball** - Είναι ένα πρόγραμμα πραγματικού χρόνου ανταπόδοσης τελικής εικόνας με ανεπτυγμένες δυνατότητες ανταπόδοσης, το οποίο εκτελείται σαν πρόσθετο στις υλοποιήσεις Maya και Autodesk 3ds Max, με ανταπόδοση μέσω δικτύου και υποστήριξη πολλαπλών GPUs. Το Furryball είναι βασισμένο στο Nvidia Optix, ενώ συνδυάζει την ταχύτητα των GPU με την ποιότητα και τις δυνατότητες της ανταπόδοσης CPU. Η δύναμη του έχει αποδειχθεί σε πραγματικές ταινίες και παραγωγές παιχνιδιών από πολλές εταιρίες.
- **Lightworks** - Εκμεταλλεύεται το Nvidia Optix για να δημιουργήσει μια καινούρια γενιά πολύ γρήγορων μηχανών ιχνογράφησης ακτίνας για αρχιτεκτονικές, βιομηχανικές ανάγκες, και ανάγκες εσωτερικής διακόσμησης χώρων.
- **Octane Renderer** - Είναι μια διαδραστική μηχανή ανταπόδοσης βασισμένη σε GPU, που παράγει φωτορεαλιστικές ανταποδόσεις με μεγάλη ταχύτητα. Αυτό επιτρέπει στους χρήστες να δημιουργήσουν έργα υψηλής αισθητικής, σε ένα κλάσμα του χρόνου που χρειάζεται από τις μηχανές ανταπόδοσης σε CPU. Η εταιρία που δημιουργεί το Octane Renderer είναι μια από τις πρώτες που ασχολήθηκαν με τις μηχανές φωτορεαλιστικής ανταπόδοσης και έπαιξε μεγάλο ρόλο στην εξέλιξη τους.
- **V-ray** - Είναι μια μηχανή ανταπόδοσης εικόνας που χρησιμοποιεί τις διαθέσιμες GPUs που βρίσκονται στο σύστημα, αντί για την CPU. Χρησιμοποιεί OpenCL, και βρίσκεται ως πρόσθετο στο πρόγραμμα Autodesk 3ds Max.

Πίνακας 3.5: Μοντελοποίηση

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
Autodesk 3ds Max + NVIDIA Iray	Τρισδιάστατη μοντελοποίηση, κινούμενα σχέδια, και ανταπόδοση		NAI
Autodesk Maya	Τρισδιάστατη μοντελοποίηση, κινούμενα σχέδια, και ανταπόδοση		NAI
Autodesk Motion Builder	Κίνηση χαρακτήρων και καταγραφή κίνησης		NAI
Autodesk Mudbox	Τρισδιάστατη απεικόνιση αγαλμάτων		NAI
Cebas finalRender	Ανταπόδοση με χρήση GPU		NAI
CentiLeo GPU Render	Ανταπόδοση με χρήση GPU		NAI
Chaos V-Ray RT	Ανταπόδοση με χρήση GPU		NAI
Jawset TurbulenceFD	Πρόσθετο για εξομοίωση φυσικής		NAI
Maxon Cinema 4D	Τρισδιάστατη μοντελοποίηση, κινούμενα σχέδια, και ανταπόδοση		NAI

Συνέχεια πίνακα 3.5			
Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
NewTek Lightwave	Τρισδιάστατη μοντελοποίηση, κινούμενα σχέδια, και ανταπόδοση		NAI
Otoy Octane Render	Ανταπόδοση με χρήση GPU		NAI
Pixologic Sculptris	Τρισδιάστατη απεικόνιση αγαλμάτων		NAI
Redshift Renderer	Επιταχυνόμενο από GPU πρόγραμμα ανταπόδοσης		NAI
Side Effects Houdini	Τρισδιάστατη μοντελοποίηση, κινούμενα σχέδια, και ανταπόδοση		NAI
The Foundry Mari	Τρισδιάστατη ζωγραφική		NAI

3.7.2 Μετατροπή

Η μετατροπή είναι η άμεση μετατροπή ψηφιακής κωδικοποίησης, όπως για παράδειγμα σε αρχεία μιας ταινίας(π.χ .mp4, avi), αρχεία ήχου(.wav .mp3), ή κωδικοποίηση χαρακτήρων. (π.χ UTF-8, ISO/IEC 8859). Αυτή γίνεται σε περιπτώσεις που η συσκευή στόχος δεν υποστηρίζει την μορφή του αρχείου ή χρειαζόμαστε μικρότερο μέγεθος αρχείου, ή για να μετατρέψουμε αρχεία παλαιότερης μορφής σε μια πιο σύγχρονη για καλύτερη υποστήριξη σε μελλοντικές εφαρμογές.

Η μετατροπή χρησιμοποιείται συχνά στα λογισμικά προβολής βίντεο για να ελαττώσουμε το μέγεθος του αρχείου βίντεο. Μια διαδικασία που γίνεται συχνά είναι η μετατροπή από αρχεία MPEG-2(DVD) σε αρχεία μορφής MPEG-4, που ενσωματώνει σύγχρονους αλγόριθμους για καλύτερη ποιότητα εικόνας σε συνδυασμό με μικρότερο μέγεθος αρχείου.[42]



Σχήμα 3.12: H264, το πιο δημοφιλές πρότυπο συμπίεσης βίντεο

3.7.3 Μειονεκτήματα

Το μεγαλύτερο μειονέκτημα της μετατροπής σε απωλεστικές μορφές αρχείου είναι η μειωμένη ποιότητα. Τα τεχνουργήματα συμπίεσης συσσωρεύονται, οπότε κάθε διαδικασία μετατροπής δημιουργεί μια βαθμιαία απώλεια ποιότητας, που είναι γνωστή ως ψηφιακή απώλεια. Για αυτόν τον λόγο, η μετατροπή συνήθως δεν συνίσταται εκτός και αν δεν μπορούμε να την αποφύγουμε.

3.7.4 Βίντεο

Η συμπίεση βίντεο χρησιμοποιεί σύγχρονες τεχνικές για να μειώσει πλεονασμούς στα δεδομένα βίντεο. Οι περισσότεροι αλγόριθμοι συμπίεσης βίντεο συνδυάζουν συμπίεση εικόνας και προσωρινή αποζημίωση κίνησης. Ο ήχος κωδικοποιείται παράλληλα με διαφορετικούς αλγόριθμους συμπίεσης άλλα συνήθως συνδυάζεται σε ένα πακέτο όταν ολοκληρωθεί η διαδικασία.

Οι περισσότεροι αλγόριθμοι συμπίεσης βίντεο χρησιμοποιούν απωλεστική συμπίεση, καθώς το ασυμπίεστο βίντεο απαιτεί πολύ μεγάλους ρυθμούς μετάδοσης. Αν και οι περισσότεροι αλγόριθμοι έχουν ένα παράγοντα συμπίεσης 3, μια τυπική συμπίεση βίντεο MPEG-4 μπορεί να έχει παράγοντα συμπίεσης από 20 έως και 200. Όπως σε όλες τις απωλεστικές διαδικασίες συμπίεσης, υπάρχει ένα δίλημμα μεταξύ της ποιότητας του βίντεο, το κόστος της επεξεργασίας της συμπίεσης και της αποσυμπίεσης, και των απαιτήσεων του συστήματος. Υπερβολικά συμπιεσμένο βίντεο μπορεί να δημιουργήσει οπτικά τεχνουργήματα.[43]

Συνήθως ο τρόπος με τον οποίο λειτουργεί ένας αλγόριθμος συμπίεσης βίντεο είναι με ομάδες γειτονικών εικονοστοιχείων(pixel), που ονομάζονται blocks. Αυτές οι ομάδες από pixels, συγκρίνονται μεταξύ τους από μια εικόνα στην επόμενη, και ο αλγόριθμος αποστέλλει μόνο τις αλλαγές μεταξύ αυτών των block. Σε περιοχές του βίντεο που υπάρχει μεγαλύτερη κίνηση, ο αλγόριθμος πρέπει να συμπίεσει περισσότερα δεδομένα για να προλάβει τον μεγαλύτερο αριθμό εικονοστοιχείων που αλλάζουν. Συνήθως σε σκηνές με φωτιά, εκρήξεις, καπνούς, το αποτέλεσμα της συμπίεσης έχει μεγαλύτερη απώλεια ποιότητας, η αύξηση του ρυθμού μετάδοσης.

Μερικοί από τους σύγχρονους αλγόριθμους μετατροπής είναι οι παρακάτω

- Lagarith - Είναι ένας μη απωλεστικός αλγόριθμος ανοιχτού κώδικα που δίνει έμφαση στην ταχύτητα, στην υποστήριξη διαφόρων χώρου χρωμάτων(YV12,RGB,YUY2). Είναι ιδανικός για επεξεργασία και αποθήκευση αρχείων. Αν και υπάρχουν κάποιοι καλύτεροι μη απωλεστικοί αλγόριθμοι συμπίεσης, ο lagarith είναι ο πιο γρήγορος και έτσι έχει κερδίσει την υποστήριξη της κοινότητας.
- VP9 - Είναι ένα ελεύθερο πρότυπο ανοιχτού κώδικα που αναπτύσσεται από την Google. Είναι ο διάδοχος του VP8. Σκοπός του είναι να μειώσει περισσότερο τον χώρο που απαιτείται από το βίντεο διατηρώντας την ίδια ποιότητα.
- H.264 - Είναι ένα πρότυπο συμπίεσης βίντεο, που είναι ίσως το πιο επιτυχημένο για την καταγραφή, συμπίεση, και αναμετάδοση περιεχομένου βίντεο. Χρησιμοποιείται για αναμετάδοση ψηφιακού σήματος τηλεόρασης, δορυφόρων, και από υπηρεσίες αναμετάδοσης στο διαδίκτυο.
- H.265 - Είναι ένα πρότυπο συμπίεσης βίντεο, διάδοχος του επιτυχημένου H.264. Σκοπός του είναι ο διπλασιασμός της συμπίεσης διατηρώντας την ίδια ποιότητα. Υποστηρίζει αναλύσεις έως και 8192x4320.
- Daala - Είναι μια τεχνολογία συμπίεσης από το ίδρυμα Xiph.Org. Χρησιμοποιεί περιτύλιξη μεταμορφώσεων για να μειώσει τα οπτικά τεχνουργήματα. Ο στόχος είναι η απόδοση του να ξεπεράσει τις δυνατότητες του VP9 και του H.265.

3.7.5 Συμπίεση δεδομένων γενετικής

Οι γενετικοί αλγόριθμοι συμπίεσης είναι η τελευταία γενιά μη απωλεστικών αλγορίθμων για συμπίεση δεδομένων (συνήθως αλληλουχίες nucleotides) χρησιμοποιώντας συμβατικούς αλγορίθμους συμπίεσης αλλά και γενετικούς αλγόριθμους βελτιστοποιημένους στο συγκεκριμένο τύπο δεδομένων. Το 2012, μια ομάδα επιστημόνων από το John Korpins University ανακοίνωσε έναν αλγόριθμο συμπίεσης γενετικής (HarZipper), ο οποίος χρησιμοποιεί HarMap δεδομένα για να πετύχει συμπίεση 95% μείωση στο μέγεθος αρχείου, πετυχαίνοντας καλύτερη συμπίεση σε πολύ καλύτερο χρόνο από ότι οι γνωστοί μη απωλεστικοί αλγόριθμοι. Άλλοι γενετικοί αλγόριθμοι συμπίεσης π.χ GenomeZip πετυχαίνουν μεγαλύτερη συμπίεση καταφέροντας αποθήκευση 6 δισεκατομμυρίων ανθρώπινων ζευγαριών γονιδιώματος σε 2.5 megabyte.

3.7.6 Προγράμματα

Πίνακας 3.6: Κωδικοποίηση βίντεο

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
ArcVideo Core	Εξαιρετικά γρήγορο και υψηλής απόδοσης σύστημα επεξεργασίας βίντεο και κωδικοποίησης		NAI
ArcVideo Live	Υψηλής πυκνότητας σύστημα επεξεργασίας βίντεο πραγματικού χρόνου και κωδικοποίησης		NAI
Cinnafilm Tachyon	Μετατροπή προτύπων		NAI
Digimetrics Aurora	Δοκιμές αυτοματοποιημένου βίντεο και μετρήσεις ήχου		NAI
Elemental Live	Ζωντανή μετάδοση επεξεργασίας βίντεο και κωδικοποίησης		NAI
Elemental Server	Επεξεργασία και κωδικοποίηση βίντεο βασισμένο σε αρχεία		NAI
isovideo Viarte	Μετατροπή προτύπων βίντεο		NAI
MainConcept CUDA H.264/AVC Encoder SDK	Κωδικοποίηση βίντεο με χρήση αλγορίθμου H.264		NAI
Snell Alchemist on Demand	Μετατροπή προτύπων βίντεο		NAI
Sorenson Squeeze	Εφαρμογή μετατροπής κωδικοποίησης βίντεο και plug-In		NAI
Telestream Vantage	Μετατροπή κωδικοποίησης βίντεο και επεξεργασία		NAI

Πίνακας 3.7: Επεξεργασία βίντεο

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
Adobe Photoshop CC	Επεξεργασία εικόνας		NAI
Adobe Premiere Pro CC	Επεξεργασία Βίντεο		NAI
Apple Final Cut Pro	Επεξεργασία Βίντεο		NAI
Avid Media Composer	Επεξεργασία Βίντεο		NAI
Grass Valley Edius	Επεξεργασία Βίντεο		NAI
Harris Velocity	Επεξεργασία Βίντεο		NAI
Quantel Qube	Αναμετάδοση επεξεργασίας βίντεο		NAI
Sony Vegas Pro	Επεξεργασία Βίντεο		NAI

Πίνακας 3.8: Σύνθεση, Τελειοποίηση, εφέ

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
ABSoft Neat Video	Πρόσθετο μείωσης θορύβου βίντεο		NAI
Adobe After Effects CC	Γραφικά και εφέ κίνησης		NAI
Autodesk Flame Premium	Τελείωμα και βελτίωση χρώματος		NAI
Autodesk Smoke	Τελείωμα και επεξεργασία		NAI
Boris FX Continuum Complete	Πρόσθετο οπτικών εφέ		NAI
Cinnafilm Dark Energy Plug-in	Διαχείριση χρωμάτων		NAI
CoreMelt complete	Πρόσθετο οπτικών εφέ		NAI
eyeon Fusion	Εφέ και σύνθεση		NAI
GenArts Monsters GT	Πρόσθετο οπτικών εφέ		NAI
GenArts Sapphire	Πρόσθετο οπτικών εφέ		NAI
HS-ART DIAMANT-Film Restoration	Βελτίωση εικόνας και αφαίρεση αντικειμένων		NAI
Neat Video Open FX	Πρόσθετο μείωσης θορύβου βίντεο		NAI
NewBlueFX Video Essentials	Πρόσθετο οπτικών εφέ βίντεο		NAI
NewBlue Titler Pro	Πρόσθετο για υποστήριξη υποτίτλων βίντεο		NAI
Pixelan AnyFX	Πρόσθετο οπτικών εφέ βίντεο		NAI
Re:Vision Effects	Πρόσθετο οπτικών εφέ βίντεο		NAI

Συνέχεια πίνακα 3.8			
Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
Red Giant Effects Suite	Πρόσθετα οπτικών εφέ βίντεο		NAI
Red Giant Magic Bullet Looks	Εργαλεία χρωμάτων και τελειώματος		NAI
The Foundry Nuke and NukeX	Εργαλεία σύνθεσης με τρισδιάστατο ανιχνευτή		NAI
Video Copilot Software Element 3D	Σύστημα στοιχείων τρισδιάστατων αντικειμένων		NAI
Video Copilot Optical Flares	Πρόσθετο οπτικών εφέ για το After Effects		NAI
Video Copilot Twitch	Πρόσθετο οπτικών εφέ για το After Effects		NAI

3.7.7 Επεξεργασία εικόνας

Οι εικόνες αποθηκεύονται στον υπολογιστή με την μορφή πλέγματος εικονοστοιχείων. Αυτά τα εικονοστοιχεία περιέχουν πληροφορία για το χρώμα και την φωτεινότητα. Οι επεξεργαστές εικόνας μπορούν να αλλάξουν τα εικονοστοιχεία για να βελτιώσουν την εικόνα με πολλούς τρόπους. Τα εικονοστοιχεία μπορούν να αλλάξουν σαν ομάδα, ή ξεχωριστά, από τους ανεπτυγμένους αλγορίθμους των επεξεργαστών εικόνας. Αυτό ισχύει περισσότερο για τις εικόνες τύπου bitmap, ενώ τα προγράμματα διανυσματικών γραφικών όπως Adobe Illustrator ή Inkscape, χρησιμοποιούνται για να δημιουργήσουν και να επεξεργαστούν διανυσματικές εικόνες, που αποθηκεύονται με γραμμές, καμπύλες, και κείμενο, αντί για εικονοστοιχεία. Οι διανυσματικές εικόνες είναι πιο εύκολο να επεξεργαστούν καθώς περιέχουν πληροφορίες για κάθε στοιχείο ξεχωριστά, ενώ μπορούμε να τις απεικονίσουμε κλιμακωτά, σε όποια ανάλυση θέλουμε.[44]

Τα προγράμματα επεξεργασίας εικόνας συνήθως προσφέρουν δυνατότητες αυτόματης βελτίωσης εικόνας οι οποίες διορθώνουν την απόχρωση και την φωτεινότητα της εικόνας καθώς και άλλα χαρακτηριστικά επεξεργασίας, όπως αφαίρεση κόκκινων ματιών, βελτίωση οξύτητας, χαρακτηριστικά μεγέθυνσης, και αυτόματη περικοπή. Αυτές οι διαδικασίες ονομάζονται αυτόματες γιατί συνήθως εκτελούνται χωρίς την διαδραστικότητα του χρήστη ή προσφέρονται με ένα πάτημα ενός πλήκτρου ή επιλέγοντας μια επιλογή από ένα μενού.

Συμπίεση

Πολλές μορφές αρχείων εικόνας χρησιμοποιούν συμπίεση για να μειώσουν το μέγεθος του αρχείου και να μειώσουν τον απαιτούμενο χώρο. Η ψηφιακή συμπίεση εικόνων μπορεί να εκτελεστεί στην κάμερα, ή στον υπολογιστή με την χρήση ενός επεξεργαστή εικόνας. Όταν για παράδειγμα αποθηκεύουμε εικόνες σε μορφή JPEG, η συμπίεση έχει ήδη συμβεί. Οι κάμερες και οι υπολογιστές έχουν δυνατότητες για να επεξεργαστούν το ποσοστό της συμπίεσης.



Σχήμα 3.13: Παράδειγμα οπτικών εφέ ενός επεξεργαστή εικόνας

Κάποιοι άλλοι αλγόριθμοι συμπίεσης, όπως των αρχείων PNG, είναι μη απωλεστικοί, που σημαίνει ότι δεν χάνεται πληροφορία όταν το αρχείο αποθηκεύεται. Σε αντίθεση ο αλγόριθμος JPEG είναι απωλεστικός, και όσο μεγαλύτερη είναι η συμπίεση, τόσο μεγαλύτερη πληροφορία χάνεται, μειώνοντας τελικά την ποιότητα της εικόνας σε σημείο που δεν μπορεί να την επαναφέρουμε. Το JPEG χρησιμοποιεί γνώσεις για τον τρόπο που λειτουργεί ο εγκέφαλος μας στην αναγνώριση εικόνων για να κάνει αυτήν την απώλεια πληροφορίας δυσκολότερη να παρατηρηθεί.

Βελτίωση εικόνας

Στα γραφικά υπολογιστών, η διαδικασία για βελτίωση της ποιότητας της ψηφιακής εικόνας που είναι αποθηκευμένη στον υπολογιστή είναι μια από τις πιο σημαντικές λειτουργίες ενός επεξεργαστή εικόνας. Πολλά προγράμματα έχουν την δυνατότητα να προσθέτουν φίλτρα για να μετατρέπουν τις εικόνες με διάφορους τρόπους. Αυτά τα προγράμματα περιέχουν διαδικασίες για εξομάλυνση της οξύτητας της εικόνας και θόλωση εικόνας με διά-

φορους τρόπους. Οι επεξεργαστές εικόνas έχουν επίσης την δυνατότητα να προσθέτουν ειδικά οπτικά εφέ που δημιουργούν περίεργα αποτελέσματα. Οι εικόνes μπορούν να μετακινηθούν και να επεξεργαστούν με διάφορους τρόπους. Μερικά από τα οπτικά εφέ είναι γεωμετρικές μετατροπές, καλλιτεχνικά εφέ, εφέ υφών, και συνδυασμοί αυτών.[45]

Προγράμματα

Πίνακας 3.9: Βελτίωση εικόνας

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
Adobe SpeedGrade CC	Βαθμολόγηση χρώματος		NAI
Assimilate Scratch	Βαθμολόγηση χρώματος και τελειώματος		NAI
Blackmagic DaVinci	Βαθμολόγηση χρώματος		NAI
Cinnafilm Dark Energy	Εφαρμογή και πρόσθετα για βελτίωση εικόνας		NAI
Digital Vision Nucoda	Βαθμολόγηση χρώματος		NAI
HS-ART DIAMANT-Film	Επαναφορά εικόνας & βελτίωση		NAI
Restoration	Επαναφορά εικόνας		NAI
Marquise Technologies Rain	Βαθμολόγηση χρώματος		NAI
Red Digital Cinema	Βαθμολόγηση χρώματος		NAI
SGO Mistika	Βαθμολόγηση χρώματος και τελειώματος		NAI
The Pixel Farm PFClean	Επαναφορά εικόνας & βελτίωση		NAI

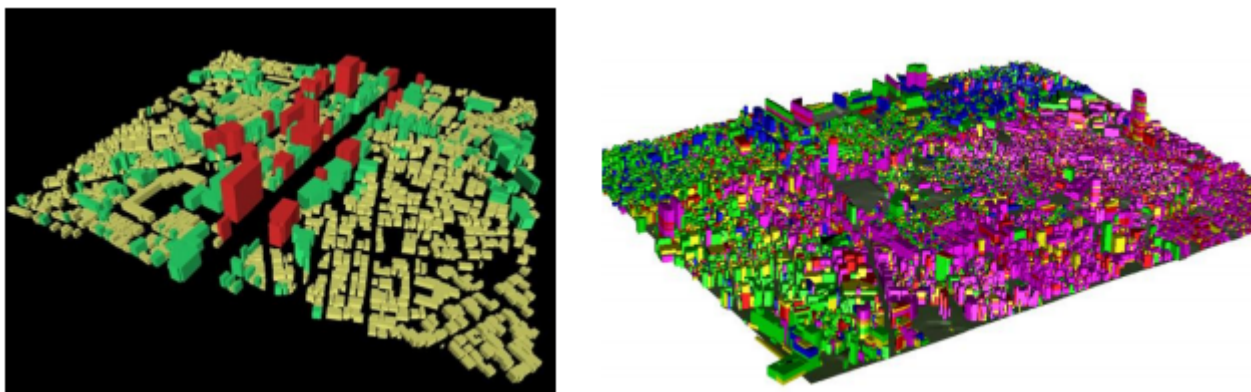
3.8 Γεωπληροφορική

3.8.1 Σεισμική δραστηριότητα

Εισαγωγή

Η ενσωματωμένη εξομοίωση σεισμικής δραστηριότητας είναι μια ανώτερη τεχνολογία για προσδιορισμό και απεικόνιση της δομικής βλάβης που δημιουργείται σε ένα σενάριο σεισμικής δραστηριότητας. Στο IES, όλα τα κτήρια που βρίσκονται στην περιοχή δοκιμής απεικονίζονται σαν δομικά μοντέλα, και η δομική ζημιά μπορεί να προσδιοριστεί με γραμμική και μη-γραμμική δυναμική δομική ανάλυση. Επειδή υπάρχουν πολλά κτήρια σε μια περιοχή δοκιμής, και κάθε κτήριο έχει διαφορετικά χαρακτηριστικά, ο υπολογισμός υψηλής ανάλυσης είναι απαραίτητος ώστε να εκτελεστεί η

ανάλυση σε σχετικά μικρό χρόνο.[46]



Σχήμα 3.14: Ενσωματωμένη εξομοίωση σεισμικής δραστηριότητας

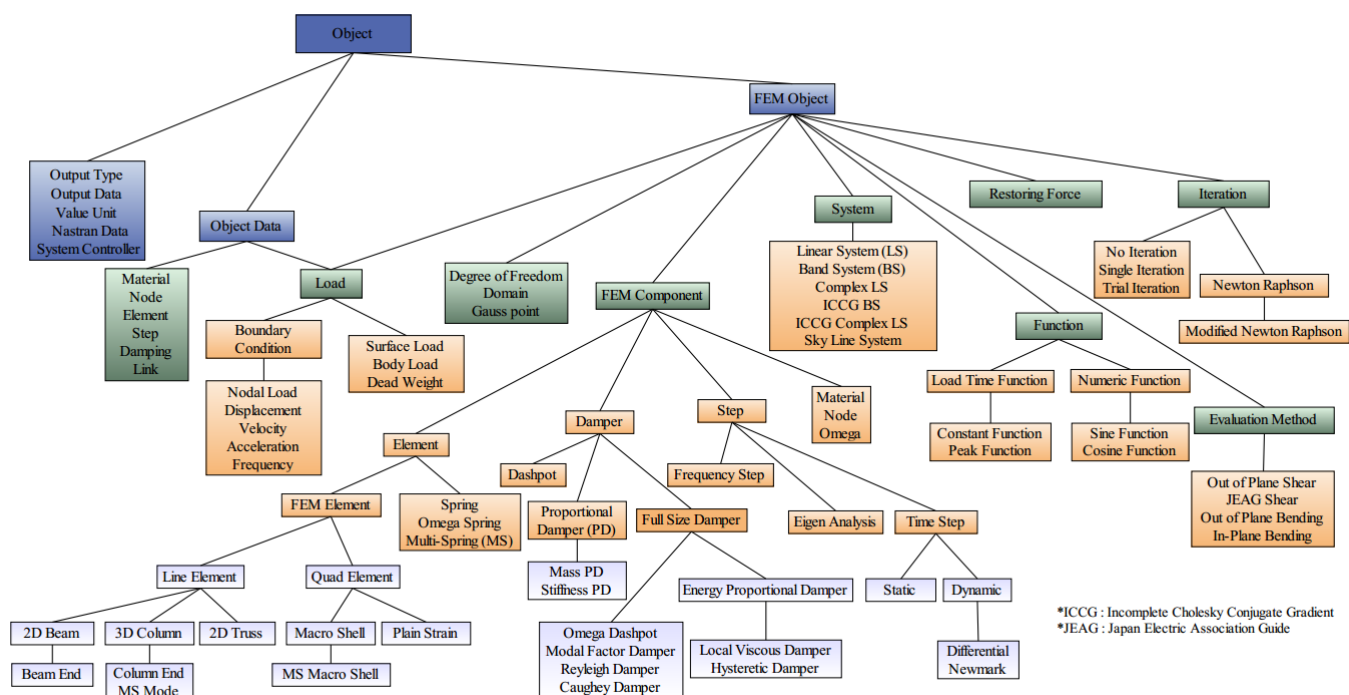
Ένας καλός υποψήφιος για τον IES, είναι το OBASAN (Object Based Structural Analysis). Σε μια εξομοίωση σεισμικής δραστηριότητας, κάθε κτήριο σε μια περιοχή δοκιμών μπορεί να παρασταθεί ως δομικό μοντέλο για ανάλυση του κάθε αντικειμένου. Το γεωγραφικό πληροφοριακό σύστημα παρέχει δεδομένα δομικών σχημάτων στην μορφή των στοιχείων και πολυγώνων, και άλλα συστήματα βάσεων δεδομένων παρέχουν τύπους κτηρίων (π.χ δομή οπλισμένου σκυροδέματος, δομή χάλυβα, ξύλινη δομή, κ.α) Βασισμένο στον τρόπο σχεδιασμού της Ιαπωνίας, η ανάλυση μοντέλων των κτηρίων στον IES δημιουργούνται αυτόματα από τις διαστάσεις και τον τύπο του κάθε κτηρίου, και τότε το OBASAN μπορεί να χρησιμοποιηθεί για να αναλύσει αυτά τα μοντέλα δοκιμής ανάλυσης. Η δοκιμή ζημιά σε όλα τα κτήρια μιας περιοχής δοκιμής μπορεί να προβλεφθεί και η ολική ζημιά σε μια περιοχή πόλης μπορεί να απεικονιστεί από αυτήν την εξομοίωση σεισμού.[47][48]

Τα πιο πολύπλοκα δομικά μοντέλα μπορούν να αυξήσουν την αξιοπιστία του προσδιορισμού δομικής βλάβης, π.χ τα μοντέλα μπορούν να χρησιμοποιηθούν για δυναμική ανάλυση δομικής απόκρισης. Μοντέλα δομικών αντικειμένων, μπορούν να χρησιμοποιηθούν για να δώσουν ένα πιο αξιόπιστο αποτέλεσμα.

OBASAN

Το OBASAN, είναι μια τεχνολογία που δημιουργήθηκε για την ανάλυση των δομικών μοντέλων ώστε να δέχεται τα αποτελέσματα των αντικειμένων (αξονική δύναμη, παραμόρφωση, καταπόνηση) και τα αποτελέσματα των κόμβων (π.χ ταχύτητα, επιτάχυνση, ισχύς) κάθε κτηρίου σε μια εξομοίωση σεισμικής δραστηριότητας, και έτσι το OBASAN μπορεί να παρέχει διάφορους τύπους αποτελεσμάτων δομικής ανάλυσης. Υπάρχουν δυνατότητες για χρήση υπολογισμού υψηλής απόδοσης ώστε να αυξηθεί η αποδοτικότητα και η δυνατότητα του συστήματος εξομοίωσης σεισμών (IES).

Το OBASAN αναπτύχθηκε με αντικειμενοστραφή προγραμματισμό σε γλώσσα C++. Η γλώσσα C++ μπορεί να επεκταθεί ώστε να χρησιμοποιεί παράλληλο προγραμματισμό στην GPU. Όπως φαίνεται στο διάγραμμα, η



Σχήμα 3.15: OBASAN

αρχιτεκτονική του OBASAN έχει οργανωθεί συστηματικά και είναι εύκολο να κατανοηθεί. Λόγω του ότι υπάρχουν πολλά κτήρια σε μια εξομοίωση σεισμικής δραστηριότητας σε μια περιοχή δοκιμών, ο IES χρησιμοποιεί την αρχιτεκτονική OpenMPI για να ενεργοποιήσει παράλληλο υπολογισμό στην κεντρική μονάδα επεξεργασίας (CPU). Στην CPU, ένα υποσύνολο (π.χ 10) κτηρίων μπορεί να εκτελεστεί με τον μέγιστο αριθμό νημάτων σε έναν παράλληλο υπολογισμό. Όπως και στον IES όπου η αρχιτεκτονική OpenMPI χρησιμοποιείται για τον παράλληλο υπολογισμό των κτηρίων σε μια περιοχή δοκιμών, στο OBASAN χρησιμοποιείται η αρχιτεκτονική CUDA για την παράλληλη επεξεργασία της δυναμικής ανάλυσης δομικής απόκρισης(DSRA).

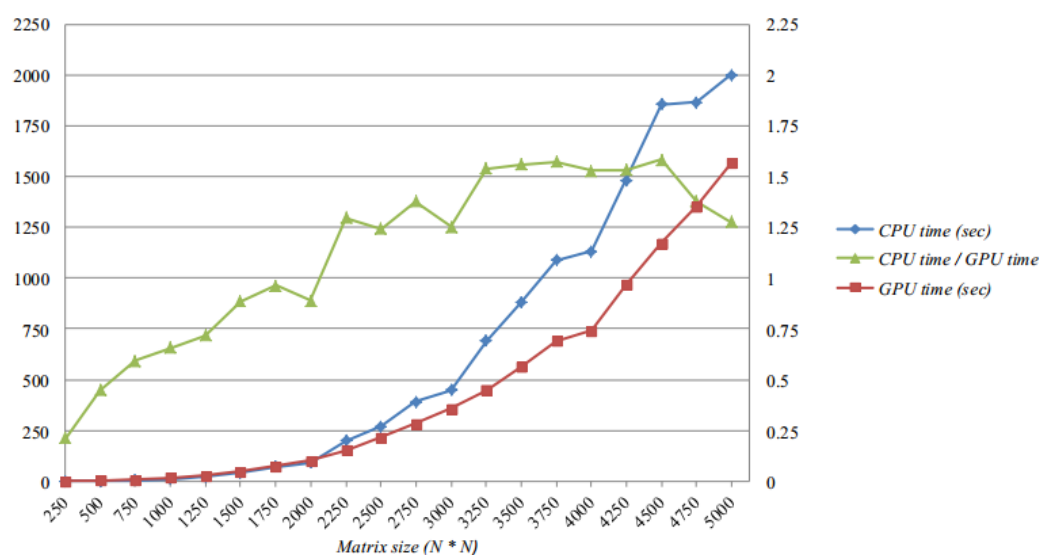
Λόγω του ότι η DSRA έχει να κάνει με πιο πολύπλοκα μοντέλα, η δομική ανάλυση των μοντέλων πρέπει να παρέχει μεγάλο αριθμό παραμέτρων, που εκφράζονται με μαθηματικές εξισώσεις που περιέχουν μεγάλους πίνακες. Για ένα τόσο προχωρημένο υπολογιστικό έργο, η τεχνική HPC είναι η συνηθέστερη προσέγγιση για την επίλυση των μαθηματικών εξισώσεων σε μικρό χρόνο. Στο HPC, η τεχνολογία GPGPU είναι σχεδιασμένη ώστε να αντιμετωπίζει μεγάλο αριθμό δεδομένων και μεγάλο αριθμό υπολογισμών σε μαθηματικές εξισώσεις. Στο GPGPU, η αρχιτεκτονική CUDA μπορεί να επιλεγεί για να λύσει αυτές τις μαθηματικές εξισώσεις της DSRA. Το κάθε βήμα της ανάλυσης της DSRA, απαιτεί πολλαπλασιασμούς πινάκων για την δομική ανάλυση.

Στο παρακάτω σχήμα μπορούμε να δούμε ένα παράδειγμα χρόνου εκτέλεσης πολλαπλασιασμού πινάκων. Οι πίνακες A,B και C υποθέτουμε ότι είναι τετράγωνοι πίνακες και το μέγεθος τους είναι $N * N$. Ο κώδικας C++ του πολλαπλασιασμού πινάκων εκτελείται σε μια CPU (Intel Xeon), ενώ οι εντολές CUDA σε μια GPU(Tesla M2050).

<i>Matrix size</i> <i>(N * N)</i>	<i>CPU time</i> <i>(sec)</i>	<i>GPU time</i> <i>(sec)</i>	<i>Matrix size</i> <i>(N * N)</i>	<i>CPU time</i> <i>(sec)</i>	<i>GPU time</i> <i>(sec)</i>
250 * 250	0.1343	0.0006	2750 * 2750	389.9368	0.2829
500 * 500	1.2111	0.0027	3000 * 3000	447.8122	0.3573
750 * 750	4.2088	0.0071	3250 * 3250	692.1258	0.4498
1000 * 1000	10.3524	0.0157	3500 * 3500	878.4347	0.5626
1250 * 1250	20.8688	0.0290	3750 * 3750	1086.8954	0.6909
1500 * 1500	41.5084	0.0468	4000 * 4000	1133.5968	0.7412
1750 * 1750	71.0964	0.0737	4250 * 4250	1484.2185	0.9687
2000 * 2000	91.2003	0.1026	4500 * 4500	1855.4903	1.1724
2250 * 2250	198.0169	0.1525	4750 * 4750	1866.9938	1.3547
2500 * 2500	269.6148	0.2171	5000 * 5000	2000.5736	1.5680

Σχήμα 3.16: GPU and CPU matrix multiplication times

Στην DRSA, το μέγεθος πίνακα ορίζεται απο τον συνολικό αριθμό των μοιρών της ελευθερίας (DOF) σε ένα κτήριο, οπότε το μέγεθος του πίνακα εξαρτάται απο τις διαστάσεις του κάθε κτηρίου. Τα μικρότερα κτήρια όπως τα σπίτια αναπαρίστανται απο μικρότερους πίνακες και τα μεγαλύτερα κτήρια αναπαρίστανται απο μεγαλύτερους πίνακες. Όπως μπορούμε να δούμε στον πίνακα, οι GPUs είναι πολύ πιο γρήγορες απο τις CPUs σε κάθε περίπτωση πολλαπλασιασμού πινάκων. Από τα δεδομένα του πίνακα, μπορούμε να απεικονίσουμε τα αποτελέσματα σε γράφημα. Στο παρακάτω γράφημα βλέπουμε την σχέση που έχει ο χρόνος εκτέλεσης με το μέγεθος του πίνακα, τόσο στην CPU όσο και στην GPU. Οι μεγαλύτεροι πίνακες χρειάζονται περισσότερο χρόνο για να εκτελέσουν τον υπολογισμό πινάκων. Όσο μεγαλύτερο το μέγεθος του πίνακα, τόσο μεγαλύτερο το πλεονέκτημα της GPU έναντι της CPU. Για ένα πίνακα 2000 * 2000 στοιχείων, οι GPUs είναι περίπου 1000 φορές γρηγορότερες απο τις CPUs. Για αυτό ο προγραμματισμός γενικού σκοπού είναι καλή επιλογή για την δυναμική ανάλυση δομικής απόκρισης.



Σχήμα 3.17: GPU and CPU matrix multiplication times

Τα παραπάνω αποτελέσματα, είναι μια απλή τεχνική CUDA. Ο προγραμ-

ματισμός υψηλής απόδοσης της DSRA μπορεί να αυξηθεί με την χρήση βιβλιοθηκών CUDA. Οι CUDA βιβλιοθήκες, π.χ CUBLAS(Γραμμική Άλγεβρα),CUFFT(Μετασχηματισμός Fast Fourier), μπορούν να λύσουν τις μαθηματικές εξισώσεις της DSRA. Γενικότερα, όλες οι βιβλιοθήκες CUDA είναι βελτιστοποιημένες για εκτέλεση υψηλής απόδοσης. Επιπλέον, το OBASAN με την χρήση CUDA μπορεί να επιτύχει καλή κλιμάκωση στον IES.

Συμπεράσματα

Για να μειώσουν την κοινωνική και οικονομική επίπτωση των σεισμών, οι ερευνητές προσπαθούν να εφαρμόσουν καινούριες τεχνολογίες και τα τελευταία δεδομένα στον IES. Η αριθμητική εξομοίωση χρησιμοποιείται για να προσδιορίσει πιθανές ζημιές σε σενάρια σεισμών. Σε μια εξομοίωση σεισμών, η σωστή πρόβλεψη της ζημιάς στην κοινωνία είναι πολύ σημαντική στην διαχείριση κρίσεων. Αυτή η προσέγγιση στο μαθηματικό πρόβλημα με την χρήση τεχνολογιών προγραμματισμού υψηλής απόδοσης και CUDA, μπορεί να εφαρμοστεί σε διάφορα πεδία της μηχανικής. [49][50]

Με την χρήση των προσδιορισμών, ένα πιο σταθερό πλάνο μπορεί να χρησιμοποιηθεί για κατασκευή καινούριων κτηρίων για να εμποδίσει δομικές ζημιές σε σενάρια σεισμών. Επιπλέον, σχέδια για εκκένωση κτηρίων και μεταφορά ατόμων μπορούν να δημιουργηθούν για να αναλύσουν τις εξόδους κινδύνου, κάτι που μπορεί να προστατέψει και να σώσει ανθρώπινες ζωές.

3.8.2 Προγράμματα

Πίνακας 3.10: Γεωπληροφορική

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
DigitalGlobe Advanced Ortho Series	Γεωδιαστημική Οπτικοποίηση	50x	NAI
Eternix Blaze Terra	Γεωδιαστημική Οπτικοποίηση	50x	NAI
Exelis (ITT) ENVI	Γεωδιαστημική Οπτικοποίηση	70x	NAI
GAIA	ανεπτυγμένη γεωδιαστημική δυνατότητα, παραγωγή χάρτη θερμότητας, και κατανεμημένες υπηρεσίες ανταπόδοσης	-	NAI
GeoWeb3d Desktop	Γεωδιαστημική Οπτικοποίηση	-	NAI
Incogna GIS	Γεωδιαστημική Οπτικοποίηση	50x	NAI
LuciadLightspeed	Γεωδιαστημική Οπτικοποίηση και ανάλυση		NAI
Manifold Systems	Γεωγραφικό πληροφοριακό σύστημα, επεξεργασία διανυσμάτων & ανάλυση	-	NAI
MrGeo	Γεωδιαστημική Οπτικοποίηση	-	NAI

Συνέχεια πίνακα 3.10			
Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
OpCoast SNEAK	Ηλεκτρομαγνητικά σήματα μοντελοποίησης για πολύπλοκα αστικά περιβάλλοντα	100x	NAI
PCI Geomatics GXL	Γεωδιαστική Οπτικοποίηση	20-60x	NAI

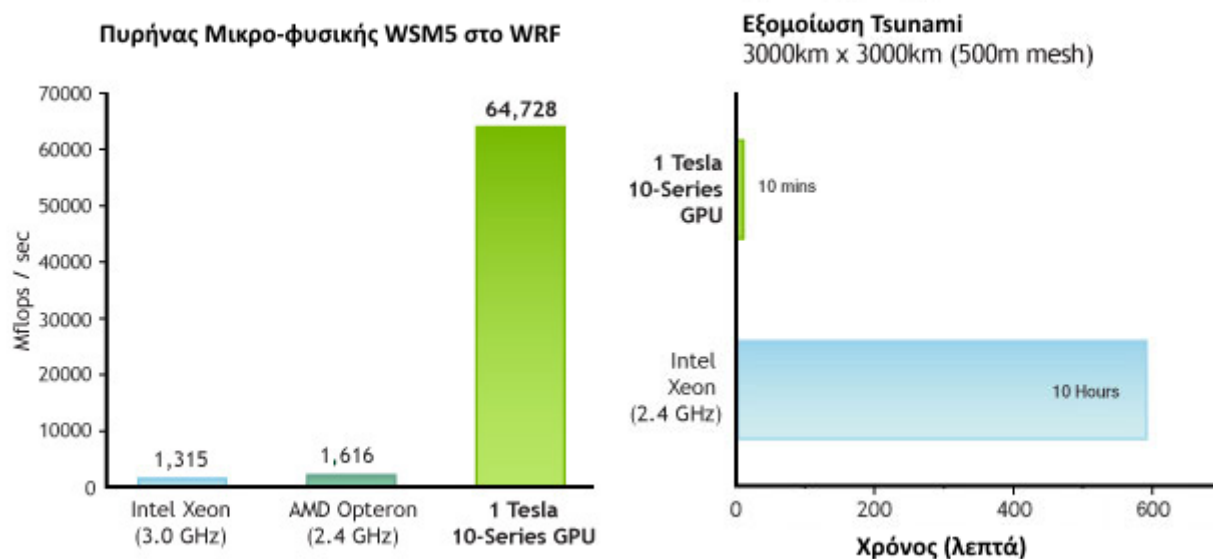
Πίνακας 3.11: Σεισμική δραστηριότητα

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
cceleware AxRTM AxKTM	Σεισμική επεξεργασία		NAI
CGGV GeoVation	Σεισμική επεξεργασία		NAI
CGG-Veritas Inside Earth	Σεισμική ερμηνεία		NAI
ffA Geoteric	Σεισμική ερμηνεία		NAI
ffA SEA3D Pro	Σεισμική ερμηνεία		NAI
ffA SVI Pro	Σεισμική ερμηνεία		NAI
GeoMage Multifocusing	Σεισμική απεικόνιση		NAI
GeoStar Seismic Suite	Σεισμική επεξεργασία		NAI
HUE Headwave Suite	Σεισμική ερμηνεία		NAI
HUE HUESpace	Σεισμική ερμηνεία		NAI
OpenGeo Solutions OpenSeis	Σεισμική επεξεργασία		NAI
Paradigm Echos RTM	Σεισμική επεξεργασία		NAI
Paradigm SKUA	Μοντελοποίηση δεξαμενών		NAI
Panorama Tech	Σεισμική επεξεργασία, μοντελοποίηση		NAI
Roxar RMS	Μοντελοποίηση δεξαμενών		NAI
Schlumberger Omega2 RTM	Σεισμική επεξεργασία		NAI
Seismic City Prestack Interpretation	Σεισμική επεξεργασία		NAI
SpectraSeis	Σεισμική επεξεργασία		NAI
Stoneridge Technologies GAMPACK	Εξομοίωση δεξαμενών		NAI
Tsunami A2011	Σεισμική επεξεργασία / πακέτο απεικόνισης		NAI
Tsunami RTM	Σεισμική επεξεργασία		NAI

3.9 Καιρός και κλίμα

3.9.1 Υπολογισμοί

Υπολογισμοί δυναμικής ρευστών σε εφαρμογές όπως μοντελοποίηση κλίματος και μοντελοποίηση ωκεανών όπως το WRF (Weather Research and Forecasting model) και εξομοιώσεις tsunami έχουν δείξει μεγάλες επιταχύνσεις που επιτρέπουν εξοικονόμηση χρόνου και βελτιώσεις στην ακρίβεια.



Σχήμα 3.18: Επιτάχυνση WRF - Εξομοίωση Tsunami

3.9.2 Προγράμματα

Πίνακας 3.12: Α

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
cuweather Cinemative HD	Γραφικά καιρικών φαινομένων		OXI
Accuweather Storyteller	Γραφικά καιρικών φαινομένων		OXI
Meteo Earth	Γραφικά καιρικών φαινομένων		OXI
Weather Central Fusion Studio	Γραφικά καιρικών φαινομένων		OXI
WSI TrueView Max	Γραφικά καιρικών φαινομένων		OXI

3.10 Αστροφυσική

Οι πρώτες καταγραφές της θεωρητικής αστρονομίας χρονολογούνται το 1550-1292 π.Χ. Οι υπολογισμοί που βρέθηκαν σχηματισμένοι σε αιγυπτιακούς τάφους δείχνουν ότι υπήρχαν τεχνικές για την αναγνώριση και την καταγραφή προτύπων στον ουρανό, ένα πεδίο της αστρονομίας που είναι χρήσιμο ακόμα και σήμερα για την καταγραφή και χαρτογράφηση.

Μπορεί οι σημερινοί ερευνητές να μην χρησιμοποιούν τα αρχαία εργαλεία, αλλά η ανάπτυξη τους έγινε σταδιακά. Το μέγεθος των τηλεσκοπίων χρειάστηκε 400 χρόνια για να μεγαλώσει από 1 τετραγωνικό μέτρο σε 110 τετραγωνικά μέτρα. Οι ψηφιακοί υπολογιστές εμφανίστηκαν στο τέλος του 1940 με υπολογιστική ταχύτητα περίπου 100 floating point λειτουργιών το δευτερόλεπτο (FLOPS), και εξελίχθηκαν σε περίπου $3 * 10^{16}$ FLOPS σε λιγότερο από 65 χρόνια. Αυτή η επανάσταση των υπολογιστών συνεχίζεται ακόμα και σήμερα, και έχει οδηγήσει σε ένα καινούριο κομμάτι έρευνας στο οποίο οι εγκαταστάσεις δεν βρίσκονται στο ψηλότερο βουνό του κόσμου, αλλά στο διπλανό δωμάτιο. Οι Αστρονόμοι κατανόησαν γρήγορα ότι μπορούν να χρησιμοποιήσουν τους υπολογιστές για να καταγράψουν, αναλύσουν, αρχειοθετήσουν, τις τεράστιες ποσότητες πληροφοριών που καταγράφονται από τις εκστρατείες παρατηρητών.[51]

Την μεγαλύτερη όμως επίδραση στον τρόπο με τον οποίο αντιμετωπίζουν οι αστρονόμοι τα αναπάντητα ερωτήματα τους, έχει το πεδίο της εξομώωσης. Με την χρήση των υπολογιστών, είναι δυνατόν να μελετήσουμε την λειτουργία του Διαγαλαξιακού κενού, την φυσική των μαύρων τρυπών, κ.α

Πίνακας 3.13: Φυσική

Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
Chemora	Το Chemora είναι ένα σύστημα που εκτελεί εξομοιώσεις συστημάτων που περιγράφονται από διαφορικές εξισώσεις που εκτελούνται σε επιταχυνόμενα υπολογιστικά συμπλέγματα		NAI
Chroma	Δικτυωτή Κβαντική Χρωμοδυναμική (LQCD)		NAI
ENZO	Τρισδιάστατος AMR κώδικας μορφής block για σχηματισμό κοσμολογικής μορφής		NAI
GTC	Εξομοιώνει μικρο-διαταραχές και μεταφορές σε μαγνητικά περιορισμένη σύντηξη πλάσματος		NAI
GTS	Εξομοιώνει μικρο-διαταραχές και την κίνηση των φορτισμένων σωματιδίων και αλληλεπιδράσεις σε σύντηξη πλάσματος		NAI

Συνέχεια πίνακα 3.13			
Όνομα	Περιγραφή	Επιτάχυνση	Multi-GPU
MILC	Δικτυωτή Κβαντική Χρωμοδυναμική (LQCD) εξομοιώνει πως δομούνται τα σωματίδια στοιχείων και δεσμεύονται από την δυνατή ισχύ ώστε να δημιουργήσουν μεγαλύτερα σωματίδια όπως πρωτόνια και νετρόνια		NAI
PIConGPU	Ένα σχεσιακό σύστημα σωματιδίων σε κελί που περιγράφει την δυναμική ενός πλάσματος υπολογίζοντας την κίνηση των ηλεκτρονίων και ιόντων που περιγράφεται από την εξίσωση Maxwell-Vlasov.		NAI
QUDA	Βιβλιοθήκη για δικτυωτούς QCD υπολογισμούς με χρήση GPUs		NAI
RAMSES	Εξομοιώνει προβλήματα αστροφυσικής σε διαφορετικές κλίμακες(π.χ σχηματισμούς αστερισμών, δυναμικές γαλαξίων, σχηματισμό κοσμολογικής μορφής)		NAI

3.11 Συμπεράσματα

Αυτό το κεφάλαιο ήταν το πιο σημαντικό της πτυχιακής εργασίας. Στο συγκεκριμένο κεφάλαιο μελετήσαμε όλες τις εφαρμογές της τεχνολογίας που θεωρώ ότι έχουν ενδιαφέρον και έγινε έρευνα σε εφαρμογές που ήδη υπάρχουν και χρησιμοποιούνται από εταιρίες, πανεπιστήμια, κυβερνήσεις σε όλον τον κόσμο. Μεγάλο βάρος δόθηκε στην έρευνα όλων των προγραμμάτων για κάθε επιστημονικό πεδίο και στον χρόνο τον οποίο επιταχύνεται η εκτέλεση της λύσης του προβλήματος.

Άξιο σημείο αναφοράς είναι το ότι το κάθε επιστημονικό πεδίο έχει διαφορετικές υπολογιστικές απαιτήσεις, που εκμεταλλεύεται ή όχι τις δυνατότητες του υπολογισμού GPGPU διαφορετικά. Για παράδειγμα, είδαμε ότι η δυναμική ρευστών έχει μεγάλες υπολογιστικές ανάγκες, ενώ η επεξεργασία εικόνας / βίντεο συνήθως δεν έχει τόσο μεγάλες ανάγκες. Επίσης είδαμε ότι πολλές εφαρμογές χρησιμοποιούν καταμεμημένα δίκτυα για να εκτελεστούν σε πλήθος υπολογιστών σε όλον τον κόσμο, για παράδειγμα στον τομέα της Βιοπληροφορικής και της Ιατρικής, και μπορούν να προσελκύσουν πλήθος εθελοντών που γοητεύονται από την ιδέα της ομαδικής προσπάθειας μέσω χρήσης υπολογιστικών πόρων, για την επίλυση προβλημάτων που απασχολούν την ανθρωπότητα.

Στο επόμενο κεφάλαιο θα αναλυθούν τα συμπεράσματα που έβγαλα από την διαδικασία της έρευνας και της ανάπτυξης της εργασίας, ιδέες για μελλοντικές εργασίες, αλλά και τα προβλήματα που αντιμετωπίσα κατά την ανάπτυξη της.

Μεθοδολογία

4.1 Εργαλεία που χρησιμοποιήθηκαν

Για την εκπόνηση της πτυχιακής εργασίας, χρησιμοποιήθηκαν αρκετά εργαλεία για λόγους ευχρηστίας αλλά και για εκπαιδευτικούς σκοπούς, σε μια προσπάθεια να αποκτηθούν γνώσεις για τεχνολογίες που πιστεύω πως χρειάζονται σε έναν απόφοιτο πληροφορικής.

4.1.1 X_YTeX

Το X_YTeX είναι μια μηχανή τυπογραφίας τύπου TeX η οποία χρησιμοποιεί κωδικοποίηση Unicode και υποστηρίζει σύγχρονες τεχνολογίες γραμματοσειρών όπως οι Opentype, Graphite και Apple Advanced Typography. Έχει σχεδιαστεί από τον Jonathan Kew και διανέμεται κάτω από την ελεύθερη άδεια λογισμικού X11. Ενώ δημιουργήθηκε αποκλειστικά για το Mac OS X, πλέον είναι διαθέσιμο για όλες τις γνωστές πλατφόρμες. Υποστηρίζει κωδικοποίηση Unicode και τα αρχεία κειμένου είναι εξαρχής σε μορφή UTF-8.

Το X_YTeX μπορεί να χρησιμοποιήσει τις γραμματοσειρές που είναι εγκατεστημένες στο σύστημα, και να κάνει χρήση των ανεπτυγμένων τυπογραφικών δυνατοτήτων τους. Υποστηρίζει και μικρο-τυπογραφία, δηλαδή μια σειρά από μεθόδους που βελτιώνουν την αισθητική του κειμένου και το καθιστούν πιο ευανάγνωστο. Οι μέθοδοι συμπεριλαμβάνουν την μείωση μεγάλων κενών μεταξύ των λέξεων (expansion), την επέκταση των γραμμών όταν τελειώνουν με κάποιο μικρό σύμβολο, όπως η τελεία ή ένα στρογγυλό γράμμα όπως το "ο" (protrusion), και ο χωρισμός γραμμών (hyphenation). Αν και το X_YTeX είναι υποδεέστερο του L^ATeX στην μικρο-τυπογραφία, επιλέχτηκε για αυτήν την πτυχιακή εργασία λόγω των πολλών άλλων πλεονεκτημάτων, όπως η χρήση unicode χαρακτήρων αλλά και η ευκολία διαχείρισης γραμματοσειρών.

4.1.2 Texmaker

Το Texmaker είναι ένα επεξεργαστής κειμένου για L^ATeX και X_YL^ATeX. Είναι cross-platform ανοιχτού κώδικα με ενσωματωμένο PDF Viewer. Το Texmaker είναι μια εξ ολοκλήρου εφαρμογή Qt. Ο επεξεργαστής κειμένου έχει άρτια υποστήριξη unicode, λειτουργία ελέγχου ορθογραφίας, αυτόματη συμπλήρωση κειμένου, κ.α. Επίσης υποστηρίζει regular expressions για την εύρεση και αντικατάσταση κειμένου.

Το Texmaker περιέχει λειτουργίες για αυτόματη χρήση των παρακάτω λειτουργιών:

- Παραγωγή νέου εγγράφου, γράμματος, βιβλίου, κ.α
- Παραγωγή πινάκων, περιβάλλοντος σχημάτων, κ.α
- Εξαγωγή κειμένου σε HTML ή ODT διάταξης

Μερικές από τις ετικέτες L^AT_EX και μαθηματικά σύμβολα μπορούν να χρησιμοποιηθούν με ένα click. Ο Texmaker βρίσκει αυτόματα τα λάθη και προειδοποιήσεις και τα εμφανίζει στο αρχείο καταγραφής.

4.1.3 Εταιρεία Ελληνικών Τυπογραφικών Στοιχείων

Η ΕΕΤΣ διαθέτει στο κοινό μια συνεχώς διευρυνόμενη ποικιλία ελληνικών ψηφιακών πολυτονικών γραμματοσειρών η οποία αποτελείται από σημαντικά ιστορικά δείγματα, αλλά και νέους σχεδιασμούς που σέβονται την τυπογραφική παράδοση και αποφεύγουν τις ανιστόρητες αντιγραφές λατινικών προτύπων. Επιπλέον, η ΕΕΤΣ έχει αναλάβει ειδικές παραγγελίες για την Ακαδημία Αθηνών, την Αρχαιολογική Εταιρεία, το Ινστιτούτο Επεξεργασίας του Λόγου κ.α.

Το Τμήμα Μαθηματικών του Πανεπιστημίου Αιγαίου έχει ιδρύσει το εργαστήριο Ψηφιακής Τυπογραφίας και Μαθηματικού Λογισμικού. Σκοπός αυτού του εργαστηρίου, μεταξύ άλλων, είναι και η υποστήριξη της επιστημονικής κοινότητας σε θέματα που σχετίζονται με το σύστημα στοιχειοθεσίας T_EX και τα παράγωγά του. Το T_EX είναι το de facto εργαλείο για επιστημονικά κείμενα διεθνώς. Μετά την ενσωμάτωση στο L^AT_EX της υποστήριξης των ελληνικών από τον Απόστολο Συρόπουλο και την παρουσίαση μιας πλήρους ελληνικής γραμματοσειράς σε Metafont από τον Claudio Beccari, παρουσιάστηκε η ανάγκη για μια γραμματοσειρά σε scalable μορφή. Έτσι αναπτύχθηκε η πρώτη ελληνική γραμματοσειρά σε Type1 μορφή με πλήρη υποστήριξη για το L^AT_EX με το όνομα «Κέρκης». Πέραν του σχεδιασμού φτιάχτηκε το βασικό αρχείο κωδικοποίησης για ελληνικές γραμματοσειρές ακολουθώντας το Unicode Standard.

Η ΕΕΤΣ παρέχει δωρεάν πλήθος γραμματοσειρών στο διαδίκτυο οι οποίες σκοπεύουν στην αναπαράσταση χαρακτήρων από τον 15ο έως τον 21ο αιώνα. Η επιλογή γραμματοσειράς από την ΕΕΤΣ μου επέτρεψε να δημιουργήσω ένα καλλιτεχνικά και οπτικά αποδεκτό αποτέλεσμα για την πτυχιακή μου εργασία, και να διατηρήσω τον επιστημονικό χαρακτήρα της. Επίσης βοήθησε να αποφευχθούν προβλήματα με άδειες γραμματοσειρών που δεν θα επέτρεπαν την δημιουργία αρχείου PDF για την παρουσίαση της εργασίας.

4.1.4 TortoiseGit

Το TortoiseGIT είναι μια εφαρμογή που υλοποιεί ένα σύστημα διαχείρισης εκδόσεων λογισμικού. Το σύστημα αυτό, είναι ένα υποσύνολο του ελέγχου αναθεωρήσεων. Ο έλεγχος αναθεωρήσεων χρησιμοποιείται για την διαχείριση των αλλαγών σε κείμενα, προγράμματα υπολογιστών, ιστοσελίδες, και άλλες συλλογές από πληροφορίες. Οι αλλαγές συνήθως αναγνωρίζονται από έναν αριθμό η από ένα γράμμα, το οποίο ονομάζεται αριθμός αναθεώρησης.

Για παράδειγμα, αν τα αρχεία είχαν αρχικό αριθμό το "αναθεώρηση 1", μετά τις αλλαγές θα αποκτήσει τον αριθμό "αναθεώρηση 2", κτλ. Κάθε αναθεώρηση σχετίζεται με μια χρονοσήμανση, και με το άτομο το οποίο πραγματοποίησε την αλλαγή. Οι αναθεωρήσεις μπορούν να συγκριθούν, να

αποκαθιστούν, και με κάποιους τύπους αρχείων, να συγχωνευτούν.

Η ανάγκη για ένα λογικό τρόπο οργάνωσης και ελέγχου αναθεωρήσεων υπάρχει σχεδόν από τότε που εφευρέθηκε η γραφή, αλλά ο έλεγχος αναθεωρήσεων έγινε πιο σημαντικός και πολύπλοκος όταν ξεκίνησε η εποχή των υπολογιστών. Η αρίθμηση των εκδόσεων βιβλίων είναι ένα από τα παραδείγματα της προηγούμενης εποχής. Σήμερα, τα πιο πολύπλοκα εργαλεία ελέγχου αναθεωρήσεων είναι αυτά που χρησιμοποιούνται στην δημιουργία λογισμικού, στα οποία μια ομάδα απο ανθρώπους μπορεί να αλλάξει τα ίδια αρχεία.

Μπορεί το σύστημα ελέγχου να είναι ένα ξεχωριστό πρόγραμμα, αλλά ο έλεγχος αναθεωρήσεων βρίσκεται ενσωματωμένος και μέσα σε διάφορους τύπους λογισμικού όπως οι επεξεργαστές κειμένου, αλλά και συστήματα διαχείρισης περιεχομένου π.χ Wikipedia, Wordpress, κ.α.

Ο έλεγχος αναθεωρήσεων δίνει την δυνατότητα να επαναφέρουμε ένα κείμενο σε μια προηγούμενη αναθεώρηση, το οποίο είναι πολύ σημαντικό για να υπάρχει έλεγχος στις αλλαγές, να διορθώνονται τα λάθη, και να προστατεύεται το περιεχόμενο από βανδαλισμό και ανεπιθύμητα μηνύματα.

4.1.5 Github

Το Github είναι μια διαδικτυακή υπηρεσία αποθηκευτικού χώρου Git, που επιτρέπει πλήρη κατανεμημένο έλεγχο αναθεωρήσεων και διαχείριση πηγαίου κώδικα μέσω Git, ενώ προσθέτει επιπλέον λειτουργίες σε αυτό. Το Github προσφέρει ένα web-based γραφικό περιβάλλον, εφαρμογές σταθερών υπολογιστών, αλλά και εφαρμογές κινητών συσκευών. Επίσης παρέχει έλεγχο πρόσβασης και αρκετά χαρακτηριστικά συνεργασίας, όπως wiki, διαχείριση έργων, παρακολούθηση λαθών, και αιτήματα χαρακτηριστικών για κάθε σχέδιο.

Η ανάπτυξη του Github ξεκίνησε το 2007. Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η Ruby on Rails και η Erlang. Η ιστοσελίδα ανακοινώθηκε τον Απρίλιο του 2008 απο τον Tom Preston-Werner, τον Chris Wanstrath, και τον Pj Hyett.

Το Github παρέχει δωρεάν λογαριασμούς, που συνηθίζεται να χρησιμοποιούνται για έργα λογισμικού ανοιχτού κώδικα. Ως το 2014, οι χρήστες του φτάνουν τα 3.4 εκατομμύρια, καθιστώντας το το μεγαλύτερο αποθηκευτικό χώρο πηγαίου κώδικα στον πλανήτη.



Σχήμα 4.1: Λογότυπο Github

4.1.6 Παρουσίαση

Blender και Bullet Physics

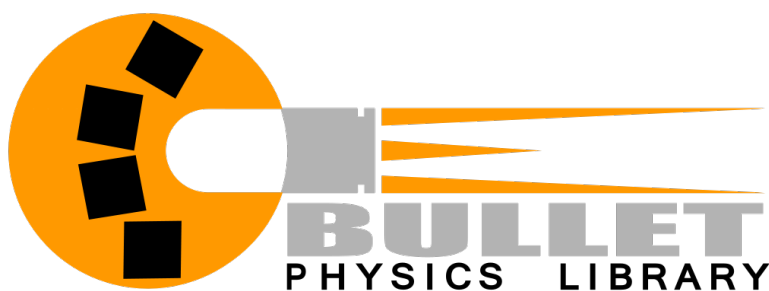
Το Blender είναι ένα επαγγελματικό πρόγραμμα τρισδιάστατων γραφικών το οποίο είναι ελεύθερο και ανοιχτού κώδικα. Χρησιμοποιείται για την δημιουργία ταινιών, οπτικών εφέ, τέχνης, τρισδιάστατων μοντέλων εκτύπωσης, διαδραστικές τρισδιάστατες εφαρμογές, και παιχνίδια υπολογιστών. Τα χαρακτηριστικά του Blender περιέχουν τρισδιάστατη μοντελοποίηση, δημιουργία υφών, εξομοίωση ρευστών και καπνού, εξομοίωση μαλακών σωμάτων, ανταπόδοση σκηνών. Περιέχει επίσης και μηχανή κατασκευής παιχνιδιών.



Σχήμα 4.2: Λογότυπο Blender 3D

Το Blender επιλέχθηκε για την παρουσίαση της πτυχιακής εργασίας, με εκπαιδευτικό παράδειγμα για το πως μπορεί να επιταχυνθεί η ανταπόδοση γραφικών και εξομοίωση φυσικής σε ένα παράδειγμα γραφικών.

Η μηχανή εξομοίωσης φυσικής επιλέχθηκε για την παρουσίαση της πτυχιακής εργασίας. Η Bullet είναι μια μηχανή φυσικής που εξομοιώνει αντίχνηυση συγκρούσεων, δυναμική άκαμπτων και μαλακών σωμάτων. Χρησιμοποιείται σε παιχνίδια υπολογιστών αλλά και για οπτικά εφέ σε ταινίες. Η βιβλιοθήκη της bullet physics είναι ελεύθερη και ανοιχτού κώδικα κάτω από την άδεια zlib. Η επιλογή της μηχανής φυσικής Bullet έγινε λόγω της μεγάλης δημοτικότητας που έχει και της καταξιωμένης εφαρμογής της σε γνωστές ταινίες και παιχνίδια, συμπεριλαμβανομένου των: Toy Story 3, Grand Theft Auto IV,V, 2012(ταινία), Hancock, Megamind, και λόγω της ενσωμάτωσής της στο πρόγραμμα γραφικών Blender.



Σχήμα 4.3: Λογότυπο Blender 3D

Netbeans και JOCL

Το Netbeans είναι ένα περιβάλλον ανάπτυξης λογισμικού που χρησιμοποιείται κυρίως σε συνδυασμό με την γλώσσα προγραμματισμού Java, αλλά

και με άλλες γλώσσες, όπως PHP, C/C++, και HTML5. Αποτελεί επίσης μια πλατφόρμα ανάπτυξης για επιτραπέζιες εφαρμογές Java, κ.α. Το Netbeans είναι ανοιχτού κώδικα και υποστηρίζει έλεγχο εκδόσεων.

Ο λόγος της επιλογής του Netbeans είναι ότι υπάρχει υποστήριξη για Java Bindings για προγραμματισμό OpenCL μέσω της βιβλιοθήκης JOCL. Το JOCL υποστηρίζει όλες τις υλοποιήσεις OpenCL που υπάρχουν αυτήν την στιγμή.

- AMD APP SDK με υποστήριξη για OpenCL 1.2
- NVIDIA οδηγούς με υποστήριξη για OpenCL 1.1
- OpenCL σε OSX
- Intel OpenCL SDK

Το JOCL χρησιμοποιήθηκε για προγραμματισμό εκπαιδευτικού παραδείγματος με την χρήση OpenCL και μελετήθηκε η απόδοση διαφόρων παραδειγμάτων σε CPU και GPU αντίστοιχα. Καθώς η Java είναι μια γλώσσα που έχουμε διδαχτεί ενδελεχώς κατά την διάρκεια των σπουδών μας, θεωρώ ότι είναι χρήσιμη και ίσως η πιο εύκολη μετάβαση από τον παραδοσιακό προγραμματισμό σε προγραμματισμό GPU. Αντίστοιχα bindings παρέχονται και για την τεχνολογία CUDA, μέσω της βιβλιοθήκης Jcuda.

Sfera

Το Sfera είναι ένα εκπαιδευτικό παιχνίδι με μεγάλη επιστημονική σημασία, ανεπτυγμένο από τον προγραμματιστή David Dade Bucciarelli. Το Sfera χρησιμοποιεί κυρίως OpenCL και OpenGL, και εκτελεί ανταπόδοση γραφικών με ιχνογράφηση ακτίνας πραγματικού χρόνου. Χρησιμοποιεί την μηχανή φυσικής Bullet. Το sfera βασίζεται στο πρόγραμμα SmallLinuxGPU v2.0. Μερικά από τα χαρακτηριστικά του είναι τα εξής:

- Πολλαπλές μηχανές ανταπόδοσης: μονο-νηματική, πολυ-νηματική, μία ή περισσότερες GPUs
- Πολλαπλά υλικά που εκπέμπουν φως: καθρεύτες, γυαλί, μέταλλο, κράμα, ματ
- Σχεδιασμός Υφών
- Βάθος πεδίου
- Φορητό: υλοποιήσεις σε Windows, Linux, MacOS, κ.α

Compubench

Το Compubench είναι ένα πρόγραμμα μετρήσεων απόδοσης OpenCL για τον έλεγχο και την σύγκριση της παράλληλης υπολογιστικής απόδοσης των CPUs, GPUs και επιταχυντές επιτραπέζιων και φορητών συσκευών. Περιλαμβάνει πολλές δοκιμές απόδοσης όπως αναγνώριση προσώπων, εξομίωση επιφάνειας ωκεανών, εξομίωση σωματιδίων, σύνθεση και μετατροπή βίντεο, εξόρυξη κρυπτο-νομισμάτων, κ.α.

GPU Caps Viewer

Το GPU Caps Viewer είναι ένα ελεύθερο και πλούσιο σε χαρακτηριστικά πρόγραμμα το οποίο περιγράφει τις δυνατότητες της GPU, συμπεριλαμβανομένου του τύπου της GPU, του μεγέθους της μνήμης, την έκδοση της

υποστηριζόμενης έκδοσης OpenGL, ενώ περιέχει και διαδικασίες ελέγχου σταθερότητας και έντασης. Το GPU Caps Viewer περιέχει αρκετά παραδείγματα της χρήσης της τεχνολογίας OpenCL με χρήση γραφικών, ενώ βρίσκεται ανάμεσα στα πρώτα προγράμματα που υποστήριξαν την τεχνολογία OpenCL.

Artifacts και demoscene

Το παράδειγμα παρουσίασης Artifacts είναι μια παρουσίαση οπτικών εφέ και μουσικής, που κέρδισε την πρώτη θέση στον αντίστοιχο διαγωνισμό της demoscene στο TokyoDemoFest 2013. Είναι ένα καλό παράδειγμα των δυνατοτήτων της GPU τόσο από την οπτική γωνία των γραφικών όσο και από την οπτική γωνία της παρουσίασης των δυνατοτήτων της υπολογιστικής δύναμης των GPU.

4.2 Άδεια χρήσης

4.2.1 Εισαγωγή

Η άδεια χρήσης λογισμικού, είναι ένα νομικό μέσο (συνήθως της μορφής του δικαίου των συμβάσεων) το οποίο ορίζει την χρήση και την διαμοίραση του λογισμικού. Σύμφωνα με τον Ευρωπαϊκό νόμο, όταν κάποιος γράφει ένα λογισμικό, η πνευματική ιδιοκτησία για αυτό το έργο προστατεύεται από τον νόμο, όπως ένα έργο λογοτεχνίας ή τέχνης. Ο νόμος της πνευματικής ιδιοκτησίας, δίνει στον ιδιοκτήτη του έργου συγκεκριμένα δικαιώματα, και βάζει όρια στο πώς οι άλλοι μπορούν να χρησιμοποιήσουν το έργο. Η πνευματική ιδιοκτησία στο λογισμικό προέρχεται από την προστασία των γραπτών έργων, και είναι σημαντικό να γνωρίζουμε ότι το λογισμικό του υπολογιστή αντιμετωπίζεται από τον νόμο σαν ένα έργο λογοτεχνικό. Η ιδιοκτησία πνευματικών δικαιωμάτων ενός έργου, είτε βιβλίου είτε λογισμικού, σημαίνει ότι ο ιδιοκτήτης, δηλαδή ο εκδότης ή ο εργοδότης, αποφασίζει για το ποιος μπορεί να το αντιγράψει, να το τροποποιήσει και να το διανέμει. Εξ αρχής, μόνο ο ιδιοκτήτης μπορεί να το κάνει αυτό.[52] Όποιος αντιγράφει, τροποποιήσει ή διανέμει έργο που ανήκει σε κάποιον άλλον χωρίς την άδεια του, μπορεί να βρεθεί αντιμέτωπος με τον νόμο.

Για να αποκτήσει κάποιος άδεια για τα παραπάνω, έχουμε συγκεκριμένες άδειες που θεωρούνται συμβόλαια, μεταξύ του εκδότη του λογισμικού και του χρήστη, ο οποίος μπορεί να το χρησιμοποιήσει όπως αναγράφεται στους όρους της άδειας χρήσης. Σημείωση: σε περίπτωση που δεν συμφωνεί ο χρήστης με τους όρους της άδειας, δεν μπορεί να χρησιμοποιήσει, αντιγράψει, τροποποιήσει, ή διανέμει το λογισμικό. Σε αντίθετη περίπτωση, παραβιάζει τον νόμο περί πνευματικών δικαιωμάτων.

Συνήθως οι άδειες χρήσεις λογισμικού εντάσσονται σε μια από τις παρακάτω κατηγορίες, ενώ η διαφορά τους βρίσκεται στους όρους με τους οποίους ο τελικός χρήστης μπορεί στην συνέχεια να διαμοιράσει το λογισμικό.

Άδειες αποκλειστικής ιδιοκτησίας

Το σήμα κατατεθέν του λογισμικού αποκλειστικής ιδιοκτησίας είναι ότι ο εκδότης επιτρέπει την χρήση μιας ή περισσότερων αντίτυπων του λογισμικού, κάτω από μια συμφωνητικού τύπου άδεια χρήσης (EULA), αλλά η ιδιοκτησία των αντίτυπων παραμένει στον εκδότη. Αυτό το χαρακτηριστικό αυτής της άδειας, σημαίνει ότι συγκεκριμένα δικαιώματα σχετικά με το λογισμικό έχουν παρακρατηθεί από τον εκδότη του λογισμικού. Γιαυτό είναι συνηθισμένο, στις EULA να υπάρχουν όροι που ορίζουν τις χρήσεις του λογισμικού, όπως ο αριθμός των εγκαταστάσεων που επιτρέπονται από τους όρους της διανομής.

Η μεγαλύτερη επίδραση αυτού του τύπου άδειας, είναι ότι, αν η ιδιοκτησία του λογισμικού παραμένει στον εκδότη, τότε ο τελικός χρήστης πρέπει να αποδεχτεί την άδεια του λογισμικού, ενώ σε περίπτωση που δεν το κάνει, δεν μπορεί να χρησιμοποιήσει το λογισμικό. Σε αυτού του τύπου άδειες περιέχεται συνήθως και μια εκτενής λίστα με λειτουργίες που απαγορεύονται,

όπως για παράδειγμα reverse engineering, ταυτόχρονη χρήση λογισμικού από πολλούς χρήστες, κ.α

Άδειες ελεύθερου και ανοιχτού κώδικα

Οι άδειες ελεύθερου και ανοιχτού κώδικα, κατατάσσονται σε μια από τις δύο κατηγορίες:

- **Permissive άδειες** - Αυτές οι άδειες έχουν λιγότερες απαιτήσεις για το πώς θα χρησιμοποιηθεί και διανεμηθεί το λογισμικό. Παράδειγμα permissive άδειας είναι η άδεια BSD και η άδεια MIT, που δίνουν απόλυτη ελευθερία για χρήση, μελέτη, και ιδιωτική τροποποίηση του λογισμικού, και περιέχει μόνο ελάχιστες απαιτήσεις για την διανομή. Αυτό δίνει σε κάποιον την δυνατότητα να χρησιμοποιήσει τον κώδικα σαν μέρος ενός λογισμικού κλειστού κώδικα, ή λογισμικού αποκλειστικής ιδιοκτησίας.



Σχήμα 4.4: Λογότυπο Open Source Initiative

- **Copyleft άδειες** - Αυτές οι άδειες έχουν ως στόχο να διατηρήσουν τις ελευθερίες που δίνονται στον χρήστη. Ένα παράδειγμα copyleft άδειας χρήσης λογισμικού, είναι το GPL. Αυτή η άδεια, στοχεύει στο να δώσει στους χρήστες την ελευθερία για χρήση, μελέτη, και ιδιωτική τροποποίηση του λογισμικού, και αν ο χρήστης εμμένει στους όρους και προϋποθέσεις του GPL, ελευθερία για διανομή του λογισμικού και κάθε τροποποίησης που έχει γίνει σε αυτό. Για παράδειγμα, κάθε τροποποίηση που γίνεται από τον χρήστη πρέπει να συνοδεύεται από τον πηγαίο κώδικα των αλλαγών όταν το λογισμικό διανεμηθεί, και η άδεια για οποιαδήποτε παράγωγο του έργου να μην έχει τους οποιεσδήποτε περιορισμούς εκτός από αυτούς που το GPL ορίζει.



Σχήμα 4.5: Λογότυπο Free Software Foundation

4.2.2 Επιλογή


Μετά από εκτενή αναζήτηση και έρευνα πάνω στις άδειες χρήσεις λογισμικού, αποφασίστηκε να χρησιμοποιηθούν δύο άδειες για την εκπόνηση της πτυχιακής εργασίας. Μία για τον πηγαίο κώδικα και μία για το τελικό αποτέλεσμα.

Πηγαίος κώδικας

Η άδεια χρήσης EUPL 1.1 κρίθηκε η καλύτερη επιλογή για την πτυχιακή εργασία. Η EUPL είναι μια άδεια λογισμικού ελεύθερου, ανοιχτού κώδικα. Έχει εγκριθεί το 2009 από το Open Source Initiative, καθώς εκπληρώνει τις προϋποθέσεις του Open Source Definition (OSD)[[osi-definition](#)]. Η άδεια εκπληρώνει επίσης τις προϋποθέσεις του Free Software Foundation (FSF) κάτι το οποίο μπορεί να συνοψίσει τέσσερις σημαντικές ελευθερίες στην άδεια

- Ελευθερία για χρήση ή εκτέλεση για οποιαδήποτε χρήση και για οποιονδήποτε αριθμό χρηστών
- Ελευθερία απόκτησης του πηγαίου κώδικα (για μελέτη του τρόπου με τον οποίο λειτουργεί το λογισμικό)
- Ελευθερία αναδιανομής αντιτύπων του λογισμικού
- Ελευθερία μετατροπής, προσαρμογής, βελτίωσης του λογισμικού σύμφωνα με συγκεκριμένες ανάγκες, και αναδιανομή αυτών των μετατροπών.

Πτυχιακή εργασία

Για την επιλογή της άδειας χρησιμοποιήθηκε ο οδηγός στο site της creativecommons <http://creativecommons.org/choose/> Επέλεξα την άδεια Attribution-ShareAlike 4.0 International  γιατί θεωρώ ότι η πτυχιακή εργασία μου μπορεί να βοηθήσει στο μέλλον και άλλους φοιτητές να γνωρίσουν την τεχνολογία του L^AT_EX, αλλά ταυτόχρονα είναι και συμβατή με την άδεια χρήσης του πηγαίου κώδικα.

Συμπεράσματα

5.1 Γενικά

Η χρήση της τεχνολογίας GPGPU είναι πλέον διαδεδομένη και ενισχύεται συνεχώς από προγραμματιστικές προσπάθειες που γίνονται σε όλον τον κόσμο. Η σημερινή μορφή του διαδικτύου βοηθάει στην εξάπλωση των καινοτόμων τεχνολογιών και μέσω μηχανισμών αλληλοβοήθειας (forums, stackoverflow, ειδικές ιστοσελίδες, κ.α) οι προγραμματιστές μπορούν να λύσουν τις απορίες τους και να εισέλθουν δυναμικά σε έναν καινούριο προγραμματιστικό μοντέλο.

Οι πλατφόρμες CUDA και OpenCL δεν αποτελούν εξαίρεση, καθώς υπάρχει τεράστιο υποστηρικτικό υλικό στο διαδίκτυο και στα βιβλιοπωλεία, ενώ τα εργαλεία ανάπτυξης λογισμικού που χρησιμοποιούνται για την ανάπτυξη GPGPU εφαρμογών γίνονται συνεχώς και πιο φιλικά προς τον χρήστη - προγραμματιστή. Ταυτόχρονα, ειδικά σεμινάρια προγραμματισμού για τις τεχνολογίες αυτές γίνονται όλο και πιο διαδεδομένα, με αποκορύφωση την ετήσια διοργάνωση της NVIDIA "GPU Technology Conference", όπου ακόμα και αν ο προγραμματιστής δεν μπορεί να παρευρεθεί στο σεμινάριο, μπορεί να βρει σχεδόν ολόκληρο το υλικό δωρεάν στο διαδίκτυο, μαζί με τις παρουσιάσεις και τα βίντεο της διοργάνωσης.

Οι εφαρμογές με επιτάχυνση μέσω GPUs έχουν αρχίσει να ωριμάζουν όσον αφορά επιστημονικά πεδία έρευνας όπως βιοπληροφορική, χημεία, ιατρική, επεξεργασία εικόνας και βίντεο, αλλά σε άλλα πεδία υπάρχει ακόμα ανάγκη για έρευνα και ανάπτυξη λογισμικού, με σκοπό την επιτάχυνση της επίλυσης διάφορων προβλημάτων. Για παράδειγμα, η επιτάχυνση GPU δεν χρησιμοποιείται αρκετά σε επιτραπέζιες λειτουργίες των λειτουργικών συστημάτων, όπως η αναζήτηση σε ένα ευρετήριο. Θεωρώ ότι υπάρχει αρκετός χώρος και ευκαιρίες για ανάπτυξη εφαρμογών οπτικής απεικόνισης, ειδικότερα όσον αφορά την υλοποίηση σε εφαρμογές διαδικτύου.

Η έρευνα μου στις ήδη υπάρχουσες εφαρμογές μου έδωσε την εντύπωση ότι ενώ υπάρχουν εφαρμογές και υλοποιήσεις που χρησιμοποιούν επιτάχυνση GPU, αυτές είναι μόνο ένα μικρό ποσοστό σε σχέση με τις εφαρμογές που υπάρχουν και αναπτύσσονται οι οποίες χρησιμοποιούν αποκλειστικά εκτέλεση μέσω CPU. Ο λόγος για αυτό είναι σίγουρα ότι η τεχνολογία έχει μόλις αρχίσει να ωριμάζει, αλλά μεγάλο ρόλο παίζει και η διάσπαση των τεχνολογιών σε CUDA και OpenCL, κάτι το οποίο διχάζει τις εταιρίες και αποτελεί πρόβλημα στην επιλογή πλατφόρμας. Το OpenCL, δείχνει να επωφελείται περισσότερο από αυτήν την κατάσταση, αφού εκτελείται στις περισσότερες συσκευές συμπεριλαμβανομένου και στις συσκευές NVIDIA και αυτό φαίνεται από το πλήθος των εφαρμογών που αναπτύσσονται σε σχέση με τις εφαρμογές CUDA. Πρόσφατα, η τελευταίες κάρτες της NVIDIA ξεπέρασαν σε απόδοση OpenCL τις αντίστοιχες κάρτες της AMD, κάτι που δείχνει την πρόθεση και των δύο εταιριών να συνεχίσουν να υποστηρίζουν την πλατφόρμα OpenCL.

Τέλος, το πλεονέκτημα της πλατφόρμας DirectCompute με το οποίο ο

προγραμματιστής γράφει κατευθείαν εντολές OpenGL και DirectX για εντατικούς υπολογισμούς, μπορεί να αλλάξει την σκηνή μελλοντικά, και να κερδίσει έδαφος από τις ανταγωνιστικές υλοποιήσεις.

5.2 Προτάσεις

Αυτή η εργασία μπορεί να αποτελέσει την βάση για επόμενες εργασίες, όπως για ανάπτυξη εφαρμογών διαδικτύου αλλά και εκπαιδευτικές εφαρμογές επιτάχυνσης μέσω GPU, ή ακόμα και να αναπτυχθεί περισσότερο με καταγραφή και ανάλυση των πεδίων επιστημονικών και όχι μόνο, στα οποία υπάρχουν ελλείψεις εφαρμογών με επιτάχυνση GPU. Για παράδειγμα, οι περισσότεροι κωδικοποιητές βίντεο και ήχου δεν υποστηρίζουν ακόμα επιτάχυνση GPU, και οι μεταγλωττιστές προγραμματιστικών γλωσσών δεν έχουν εκμεταλλευτεί ακόμα τις δυνατότητες των GPUs. Επιπλέον, η χρήση του \LaTeX που έγινε για την συγκεκριμένη εργασία, θα μπορούσε να εκμεταλλευτεί τις δυνατότητες της επιτάχυνσης GPU για να μεταγλωττίσει και σε συνεργασία με την CPU να παράγει πιο γρήγορα το τελικό αρχείο παρουσίασης.

Θεωρώ ένα από τα πιο σημαντικά και πρόσφατα ζητήματα, την ανάπτυξη εφαρμογών WebCL, η οποία θα παίξει μεγάλο ρόλο στο προσεχές μέλλον για εφαρμογές διαδικτύου και για αυτό θα πρότεινα την άμεση εκμετάλλευση του για εκπαιδευτικές και όχι μόνο εφαρμογές. Η χρήση της κάρτας γραφικών για επιτάχυνση υπολογισμών μέσα σε browsers που λόγω της φύσης τους είναι δύσκολο να αποδώσουν τα μέγιστα των δυνατοτήτων του υπολογιστή, ανοίγει νέες ευκαιρίες για εφαρμογές γραφικών αλλά και δίνει ένα επιπλέον προβάδισμα στην γλώσσα Javascript και βοηθάει στην μείωση του φόρτου εργασίας των νημάτων του browser.

Τέλος, η χρήση του Blender με την μηχανή φυσικής Bullet 3.0 η οποία βρίσκεται ακόμα σε δοκιμαστικό στάδιο και υποστηρίζει επιτάχυνση μέσω GPU, μπορεί να χρησιμοποιηθεί για εργασίες αλλά και για παραδείγματα εκπαιδευτικού τύπου στο μάθημα των γραφικών του ΤΕΙ, ώστε να προετοιμάσει και να εφοδιάσει τους φοιτητές με γνώσεις γραφικών αλλά και προγραμματισμού εφαρμογών ψυχαγωγίας όπως παιχνίδια ηλεκτρονικών υπολογιστών με πραγματικά εντυπωσιακή εξομοίωση φυσικής, κίνησης ρευστών και καπνού, δηλαδή ιδιότητες που μέχρι και πριν λίγα χρόνια ήταν αδύνατο να διδαχθούν λόγω περιορισμών στην υπολογιστική δύναμη αλλά και στην έλλειψη περιβαλλόντων ανάπτυξης ειδικών εφαρμογών.

Βιβλιογραφία

- [1] David B. Kirk (NVIDIA) and Wen-mei W. Hwu (University of Illinois). *Programming Massively Parallel Processors*. 2010 (Σελίδα VII).
- [2] Wen-Mei Hwu. *GPU Computing Gems: Emerald Edition*. 2011 (Σελίδα VIII).
- [3] David Kirk Wen-Mei Hwu. *Programming Massively Parallel Processors: A Hands-on Approach*. 2010 (Σελίδα IX).
- [4] Massimiliano Fatica Greg Ruetsch. *CUDA Fortran For Scientists and Engineers*. 2013 (Σελίδα 4).
- [5] Nicholas Wilt. *CUDA HANDBOOK: A COMPREHENSIVE GUIDE TO GPU PROGRAMMING*. 2013 (Σελίδα 4).
- [6] Rob Farber. *CUDA Application Design and Development*. 2011 (Σελίδα 5).
- [7] Edward Kandrot Jason Sanders. *CUDA by Example: An Introduction to General-Purpose GPU Programming*. 2010 (Σελίδα 6).
- [8] Aaftab Munshi (Editor). *The OpenCL specifications*. 2013 (Σελίδα 9).
- [9] Takashi Nakamura Ryoji Tsuchiyama. *The OpenCL Programming Book*. 2012 (Σελίδα 10).
- [10] David R. Kaeli Benedict Gaster Lee Howes. *Heterogeneous Computing with OpenCL*. 2011 (Σελίδα 11).
- [11] Matthew Scarpino. *OpenCL in Action*. 2013 (Σελίδα 11).
- [12] OpenGL various editors. *Compute Shaders*. 2014. URL: https://www.opengl.org/wiki/Compute_Shader (Σελίδα 14).
- [13] NVIDIA various editors. *DirectCompute*. 2013. URL: <http://www.nvidia.com/object/directcompute.html> (Σελίδες 14, 18).
- [14] PathScale Enzo. *PathScale Enzo UserGuide*. 2014. URL: <http://www.pathscale.com/pdf/PathScale-ENZ0-1.0-UserGuide.pdf> (Σελίδα 19).
- [15] MPI. *The Message Passing Interface (MPI) standard*. 2014. URL: <http://www.mcs.anl.gov/research/projects/MPI/> (Σελίδα 26).
- [16] OpenMP. *The OpenMP API specification for parallel programming*. 2014. URL: <http://www.OpenMP.org>. (Σελίδα 26).
- [17] A. Munshi (Ed). Khronos Group. *OpenCL, The OpenCL Specification*. 2010. URL: <http://www.khronos.org/opencv> (Σελίδα 26).
- [18] A. Barak and A. Shiloh. *The MOSIX management system for Linux cluster, multi-clusters, GPU clusters and Clouds*. 2010. URL: <http://www.mosix.org/pub/MOSIX%20wp.pdf> (Σελίδα 28).
- [19] Jan Pelzl Christof Paar. *Understanding Cryptography, A Textbook for Students and Practitioners*. 2010. URL: <http://www.cryptography-textbook.com/> (Σελίδα 32).
- [20] William Stallings. *Cryptography and Network Security: Principles and Practice*. 2013 (Σελίδα 33).
- [21] David Kahn. *The Codebreakers*. 1967 (Σελίδα 33).

- [22] NIST. "*NIST Selects Winner of Secure Hash Algorithm (SHA-3) Competition*". 2012 (Σελίδα 33).
- [23] Bruce Schneier. "*The Data Encryption Standard (DES)*". 2000 (Σελίδα 34).
- [24] Bitcoin Community. "*Bitcoin cryptocurrency*". 2014. URL: <https://bitcoin.org/en/> (Σελίδα 35).
- [25] Arthur Lesk. *Introduction to Bioinformatics*. 2013 (Σελίδα 37).
- [26] Conrad Bessant and Darren Oakley. *Building Bioinformatics Solutions*. 2014 (Σελίδα 38).
- [27] Paurush Praveen Sinha. *Bioinformatics with R Cookbook*. 2014 (Σελίδα 38).
- [28] Jean-Michel Claverie and Cedric Notredame. *Bioinformatics for Dummies 2nd Edition*. 2006 (Σελίδα 39).
- [29] GPUGRID.net. *Volunteer computing for biomedicine*. Online; accessed 27-September-2014. 2014. URL: <http://www.gpugrid.net/about.php> (Σελίδα 40).
- [30] GPUGRID.net. *Volunteer computing for biomedicine*. Online; accessed 27-September-2014. 2014. URL: <http://www.gpugrid.net/science.php> (Σελίδα 41).
- [31] Ira N. Levine. *Quantum Chemistry*. 2013 (Σελίδα 43).
- [32] Shan Gao. *Quantum Mechanics: A Comprehensible Introduction for Students*. 2014 (Σελίδα 43).
- [33] Jim Al-Khalili and Johnjoe McFadden. *Life on the Edge: The Coming of Age of Quantum Biology*. 2014 (Σελίδα 43).
- [34] Anatoly I. Ruban and Jitesh S. B. Gajjar. *Fluid Dynamics: Part 1: Classical Fluid Dynamics*. 2014 (Σελίδα 45).
- [35] Joel H. Ferziger and Milovan Peric. *Computational Methods for Fluid Dynamics*. 2013 (Σελίδα 45).
- [36] Dr J. F. Douglas and Dr John Gasiorok. *Fluid Mechanics*. 2011 (Σελίδα 45).
- [37] Brian J. Hoskins and Ian N. James. *Fluid Dynamics of the Mid-Latitude Atmosphere (Advancing Weather and Climate Science)*. 2014 (Σελίδα 45).
- [38] Jiyuan Tu and Guan-Heng Yeoh. *Computational Fluid Dynamics: A Practical Approach*. 2012 (Σελίδα 47).
- [39] Jason Gregory. *Game Engine Architecture, Second Edition*. 2014 (Σελίδα 50).
- [40] Krishna Kumar. *Learning Physics Modeling with Physx*. 2014 (Σελίδα 50).
- [41] Sanjay Madhav. *Game Programming Algorithms and Techniques: A Platform-Agnostic Approach*. 2013 (Σελίδα 51).
- [42] Pat Bitton and Tatyana Kropina. *Movie-Making in One Hour: Tips for Trouble-Free Video Shooting and Editing*. 2014 (Σελίδα 57).
- [43] Gael Chandler. *Cut by Cut, 2nd edition: Editing Your Film or Video*. 2012 (Σελίδα 58).
- [44] Adobe Creative Team. *Adobe Photoshop CS6 Classroom in a Book*. 2012 (Σελίδα 61).

- [45] Barbara Obermeier and Ted Padova. *Photoshop Elements 12 All-in-One For Dummies*. 2013 (Σελίδα 63).
- [46] Ichimura T Hori M and Oguni K. *Development of Integrated Earthquake Simulation for Estimation of Strong Ground Motion, Structural Responses and Human Action in Urban Areas*. 2006 (Σελίδα 64).
- [47] Hori M and Ichimura T. *Current state of integrated earthquake simulation for earthquake hazard and disaster*. 2008 (Σελίδα 64).
- [48] Lalith M and Hori M. *Application of high performance computation for the prediction of urban area earthquake disaster*. 2011 (Σελίδα 64).
- [49] Hori M Sobhaninejad G and Kabeyasawa T. *Enhancing integrated earthquake simulation with high performance computing*. 2011 (Σελίδα 67).
- [50] Hwu WW Kirk DB. *Programming Massively Parallel Processors*. Elsevier. Morgan Kaufmann. 2013 (Σελίδα 67).
- [51] S. Portegies Zwart and J. B'edorf. "“Computational Gravitational Dynamics with Modern Numerical Accelerators”". In: *ArXiv e-prints* (Sept. 2014). "arXiv": 1409.5474 ("astro-ph.IM") (Σελίδα 70).
- [52] European Parliament. *EUPL Guideline*. Online; accessed 6-November-2014. 2013. URL: https://joinup.ec.europa.eu/sites/default/files/ckeditor_files/files/EUPL%201_1%20Guidelines%20EN%20Joinup.pdf (Σελίδα 79).