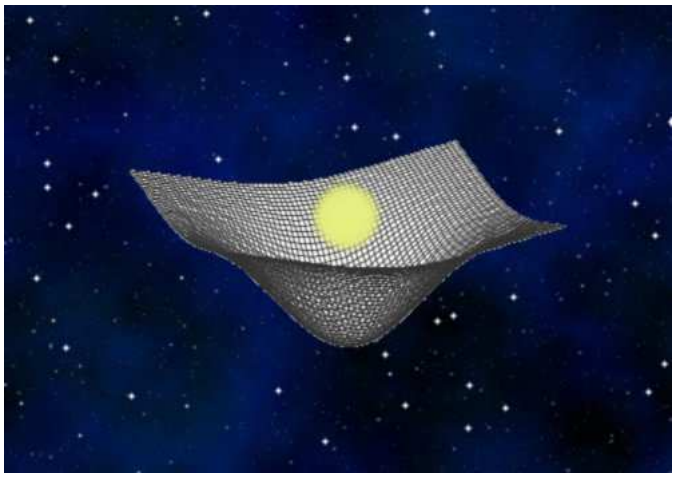
Προσομοίωση βαρυτικού πεδίου ενός μη περιστρεφόμενου αστέρα νετρονίων σε GPU



Φοιτητής: Ταντιλιάν Γκράτσια
Καθηγητές: Κ. Κόκκοτας, Ν. Στεργιούλας, Β. Zink

Πρόγραμμα Μεταπτυχιακών Σπουδών: Υπολογιστικής Φυσικής
Τμήμα: Φυσικής
Πανεπιστήμιο:
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
και
Eberhard Karls Universität Tübingen

Θεσσαλονίκη
Ιούλιος 2011

Περίληψη

Το βασικό θέμα της εργασίας είναι το πως μπορούμε να προσομοιώσουμε το βαρυτικό πεδίο ενός αστέρα νετρονίων σε GPU. Αρχικός σκοπός είναι η επίλυση των εξισώσεων του Αϊνστάιν για έναν μη περιστρεφόμενο αστέρα νετρονίων. Υιοθετούμε τον φορμαλισμό ADM 3+1 και χρησιμοποιούμε την προσέγγιση της χωρικά σύμμορφης επιπεδότητας (conformal flatness approximation - CFC), στην οποία έχουμε να επιλύσουμε μόνο ελλειπτικές εξισώσεις. Για τον υπολογισμό των εσωτερικών παραμέτρων του αστέρα χρησιμοποιούμε τις εξισώσεις ισορροπίας. Για ταχύτερη εκτέλεση, επιλέγουμε τον παράλληλο προγραμματισμό σε GPU, στη γλώσσα CUDA, ενώ χρησιμοποιούμε την αριθμητική τεχνική multigrid. Μέσω μιας βελτιστοποίησης των υπολογισμών πετύχαμε σημαντικά καλύτερους χρόνους εκτέλεσης του προγράμματος σε CUDA, σε σύγκριση με μια αρχική υλοποίηση.

Περιεχόμενα

1 Εισαγωγή **1**

2 Θεωρία **2**
2.1 CFC - Approximation 2
2.2 TOV 4
2.3 Το μοντέλο 4

3 Αριθμητικές μέθοδοι **6**
3.1 Μέθοδος Gauss - Seidel 6
3.2 Red - Black 8
3.3 Multigrid 9

4 Προγραμματισμός σε GPU **15**
4.1 GPUs 16
4.2 CUDA 18

5 Η προσομοίωση **21**

6 Βελτιστοποίηση **22**

7 Τελικά αποτελέσματα **22**

1 Εισαγωγή

Η παρούσα εργασία αποτελεί μια σύνοψη στα Ελληνικά, της πτυχιακής εργασίας με τίτλο *Simulating the gravitational field of a non-rotating neutron star on GPUs* του φοιτητή Γκράτσια Ταντιλιάν.

Σύμφωνα με τη γενική θεωρία της σχετικότητας, του Αϊνστάιν, η βαρύτητα δεν είναι μια συνηθισμένη δύναμη αλλά μια ιδιότητα της γεωμετρίας του χωροχρόνου. Χρησιμοποιώντας τη γενική θεωρία της σχετικότητας, υψηλής πυκνότητας ουράνια σώματα, όπως οι αστέρες νετρονίων, προβλέπεται να εκπέμψουν βαρυτικά κύματα κατά την περιστροφή τους, μέσω ασταθειών.

Σκοπός της εργασίας αυτής είναι να δημιουργήσουμε μια προσομοίωση του βαρυτικού πεδίου ενός αστέρα νετρονίων ώστε να προβλέψουμε τη φύση των βαρυτικών αυτών κυμάτων. Αυτό όμως, είναι το τελευταίο βήμα. Εδώ θα περιοριστούμε στο βαρυτικό πεδίο ενός στατικού αστέρα νετρονίων.

Φυσικά τέτοιες προσομοιώσεις υπάρχουν ήδη. Η πρωτοτυπία της παρούσας εργασίας έγκυται στο ότι η προσομοίωση θα γίνει σε γλώσσα προγραμματισμού CUDA και με την χρήση των GPU για την επεξεργασία, αντί της CPU. Σκοπός είναι να εκτελεστεί η προσομοίωση όσο το δυνατόν ταχύτερα.

2 Θεωρία

Όπως αναφέρθηκε προηγουμένως, η θεωρία που θα χρησιμοποιήσουμε είναι η γενική θεωρία σχετικότητας του Αϊνστάιν. Η θεωρία αυτή θεμελιώθηκε από τον Άλμπερτ Αϊνστάιν μεταξύ του 1907 και 1915. Σύμφωνα με τη θεωρία αυτή η βαρυτική αλληλεπίδραση μεταξύ δυο μαζών οφείλεται στην καμπύλωση του χωροχρόνου. Έτσι, η θεωρία της βαρύτητας μετατρέπεται σε θεωρία της γεωμετρίας του χωροχρόνου.

Η εξίσωση του Αϊνστάιν που περιγράφει την γεωμετρία του χωροχρόνου είναι μια τανυστική εξίσωση:

$$R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R = \frac{8\pi G}{c^4} T_{\mu\nu}. \tag{1}$$

Η λύση της παραπάνω εξίσωσης είναι ο μετρικός τανυστής $g_{\mu\nu}$. Για έναν επίπεδο χωρόχρονο σε καρτεσιανές συντεταγμένες, ο τανυστής αυτός είναι:

$$g_{\mu\nu} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2}$$

Έτσι λοιπόν, σκοπός μας είναι να λύσουμε την εξίσωση του Αϊνστάιν και να βρούμε τη μορφή του μετρικού τανυστή για την περίπτωση του αστέρα νετρονίων. Αρχικά θα ψάξουμε να δούμε αν μπορούμε να κάνουμε κάποιες προσεγγιστικές παραδοχές ώστε να απλοποιηθεί το πρόβλημά μας, χωρίς να αποκλίνουμε σημαντικά από την πραγματική λύση.

2.1 CFC - Approximation

Η προσέγγιση που θα κάνουμε, βασίζεται στην περιγραφή της δημοσίευσης *Dimmelmeier et al. (2002)*.

Αρχικά υιοθετούμε τον ADM (Arnowitt-Deser-Misner) 3+1 φορμαλισμό, ώστε να διαχωρίσουμε τον τετραδιάστατο χωροχρόνο στον τρισδιάστατο χώρο και στο χρόνο. Στην περίπτωση αυτή το στοιχειώδες μήκος παίρνει την μορφή:

$$ds^2 = -N^2 dt^2 + \gamma_{ij}(dx^i + \beta^i dt)(dx^j + \beta^j dt), \tag{3}$$

όπου N είναι η συνάρτηση του χρόνου (lapse function), και περιγράφει τη μεταβολή του τοπικού χρόνου κατά μήκος ενός χρονοειδούς μοναδιαίου διανύσματος, β^i είναι το διάνυσμα της χωρικής μετατόπισης (spatial shift three-vector), που περιγράφει την κίνηση του συστήματος συντεταγμένων σε μια χωροειδή υπερ-επιφάνεια και γ_{ij} είναι το χωρικό μέρος του μετρικού τανυστή.

Με την προσέγγιση της χωρικά σύμμορφης επιπεδότητας (conformal flatness approximation, CFC), προσεγγίζουμε τον μετρικό τανυστή $g_{\mu\nu}$ με το να αντικαταστήσουμε το χωρικό τμήμα του γ_{ij} με τν χωρικά σύμμορφα επίπεδη μετρική:

$$\gamma_{ij} = \phi^4 \hat{\gamma}_{ij}, \tag{4}$$

όπου $\hat{\gamma}_{ij}$ είναι η μετρική του επίπεδου χώρου. Σε καρτεσιανές συντεταγμένες ισχύει:

$$\hat{\gamma}_{ij} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{5}$$

Δηλαδή, προσεγγίζουμε την μετρική του χώρου με την μετρική του επίπεδου χώρου πολλαπλασιασμένη με έναν σύμμορφο παράγοντα ϕ^4. Με την προσέγγιση CFC και τον φορμαλισμό ADM 3+1, οι εξισώσεις του Αϊνστάιν παίρνουν την μορφή των ελλειπτικών εξισώσεων:

$$\begin{aligned}
\hat{\nabla}^2\phi &= -2\pi\phi^5 \left(\rho h W^2 - P + \frac{K_{ij}K^{ij}}{16\pi} \right), \\
\hat{\nabla}^2(N\phi) &= -2\pi N\phi^5 \left(\rho h(3W^2 - 2) + 5P + \frac{7K_{ij}K^{ij}}{16\pi} \right), \\
\hat{\nabla}^2\beta^i &= 16\pi N\phi^4 S^i + 2\hat{K}^{ij}\hat{\nabla}\left(\frac{N}{\phi^6}\right) - \frac{1}{3}\hat{\nabla}^i\hat{\nabla}_k\beta^k,
\end{aligned} \tag{6}$$

όπου ρ: η πυκνότητα της ύλης, $h = 1 + \varepsilon P/\rho$ η ειδική σχετικιστική ενθαλπία (ε η εσωτερική πυκνότητα ενέργειας), P η πίεση, $W = Nu^t$ ο παράγοντας Lorentz, $K_{ij} = \mathcal{L}_{\mathbf{n}}\gamma_{ij}$ η εξωγενής καμπυλότητα, $K = K_i{}^i$ το ίχνος της εξωγενούς καμπυλότητας και η γενικευμένη ορμή S^i.

2.2 TOV

Όπως φαίνεται από τις πραπάνω εξισώσεις, θα χρειαστούμε κάποιες πληροφορίες για τα χαρακτηριστικά του αστέρα που θα μελετήσουμε. Αυτές θα προκύψουν από την λύση των εξισώσεων TOV.

Οι εξισώσεις TOV (Tolman-Openheimer-Volkoff) για έναν σφαιρικό αστέρα, είναι οι ακόλουθες τρεις συζευγμένες διαφορικές εξισώσεις που συνδέουν τη συνάρτηση μάζας, την πυκνότητα και την πίεση του αστέρα:

$$\begin{aligned}
\frac{dm}{dr} &= 4\pi r^2 \rho(r), \\
\frac{dP}{dr} &= -\frac{\rho m}{r^2}\left(1+\frac{P}{\rho}\right)\left(1+\frac{4\pi P r^3}{m}\right)\left(1-\frac{2m}{r}\right)^{-1}, \\
\frac{d\Phi}{dr} &= -\frac{1}{\rho}\frac{dP}{dr}\left(1+\frac{P}{\rho}\right)^{-1},
\end{aligned} \tag{7}$$

όπου $m(r)$ είναι η συνάρτηση μάζας του αστέρα, $P(r)$ η πίεση, $\rho(r)$ η πυκνότητα, $\Phi(r) = \ln N$. Το στοιχειώδες μήκος είναι:

$$ds^2 = -e^{2\Phi}dt^2 + e^{2\lambda}dr^2 + r^2 d\Omega^2, \tag{8}$$

όπου $e^{2\lambda} = \left(1-\frac{2m}{r}\right)^{-1}$.

Με την χρήση του νόμου της πολυτροπικής καταστατικής εξίσωσης $P = K\rho^{\Gamma}$ λύνουμε τις εξισώσεις TOV με τη μέθοδο Runge-Kutta τέταρτης τάξης. Για έναν αστέρα νετρονίων επιλέγουμε $\Gamma = 2$ και $K = 100$ (σε αδιάστατο σύστημα μονάδων).

Από τις εξισώσεις αυτές προκύπτουν οι συναρτήσεις πυκνότητας, μάζας, πίεσης, αλλά και οι συνοριακές συνθήκες για το πρόβλημά μας.

2.3 Το μοντέλο

Καθώς αρχικός μας σκοπός είναι η προσομοίωση του βαρυτικού πεδίου ενός μη-περιστρεφόμενου αστέρα νετρονίων, οι όροι που περιλαμβάνουν κίνηση, εξαλείφονται από τις εξισώσεις.

Καταρχήν, λόγω της μη-κίνησης μπορούμε να επιλέξουμε κατάλληλο σύστημα συντεταγμένων ωστε ο όρος β^i να μηδενιστεί. Επίσης, η εξωγενής καμπυλώτητα μδενίζεται: $K_{ij} = 0$.

Υπό αυτές τις συνθήκες, οι εξισώσεις που καλούμαστε να λύσουμε, παίρνουν τη μορφή:

$$\begin{aligned} \hat{\nabla}^2\phi &= -2\pi\phi^5 \left(\rho h W^2 - P\right), \\ \hat{\nabla}^2(N\phi) &= -2\pi N\phi^5 \left(\rho h(3W^2 - 2) + 5P\right), \end{aligned} \tag{9}$$

που είναι δυο μη-γραμμικές ελλειπτικές διαφορικές εξισώσεις. Οι συναρτήσεις πίεσης και πυκνότητας, δίνονται από τις εξισώσεις TOV.

Οι άγνωστες ποσότητες είναι οι ϕ και N. Αυτό που θα κάνουμε είναι να επιλύσουμε με επαναληπτική διαδικασία την πρώτη εξίσωση για να βρούμε το ϕ και τη δεύτερη για να βρούμε το $\psi = N\phi$. Έπειτα έχουμε:

$$N = \frac{\psi}{\phi}. \tag{10}$$

Έχοντας αυτές τις συναρτήσεις, έχουμε τη λύση του προβλήματός μας $\tilde{g}_{\mu\nu}$, καθώς:

$$\tilde{g}_{\mu\nu} = \begin{pmatrix} -N^2 & 0 & 0 & 0 \\ 0 & & & \\ 0 & & \gamma_{ij} & \\ 0 & & & \end{pmatrix}, \tag{11}$$

όπου

$$\gamma_{ij} = \phi^4 \hat{\gamma}_{ij} = \phi^4 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{12}$$

Συμπερασματικά, μπορούμε να πούμε πως το μόνο που χρειαζόμαστε είναι να βρούμε τις συναρτήσεις $N(x, y, z)$ και $\phi(x, y, z)$.

Σημειώνουμε, πως για έναν σφαιρικά συμμετρικό χωροχρόνο, η προσέγγιση CFC είναι ακριβής, παρόλα αυτά αναφερόμαστε σε αυτή ως προετοιμασία για μελλοντική επέκταση του κώδικα για μη-σφαιρικά-συμμετρικές περιπτώσεις. Επίσης, λύνοντας τις εξισώσεις TOV αποκτούμε και τη λύση για τις συναρτήσεις $N(r)$ και $\phi(r)$, όμως δεν τις χρησιμοποιούμε, παρά μόνο την $P(r)$.

3 Αριθμητικές μέθοδοι

Προβλήματα αναλυτικά, πολλές φορές πρέπει να αντικατασταθούν με αντίστοιχα διακριτά προβλήματα, που επιδέχονται λύσεις αριθμητικές με αρκετά καλή προσέγγιση των αναλυτικών λύσεων. Η διαδικασία αυτή ονομάζεται διακριτοποίηση. Για παράδειγμα, οι περισσότερες διαφορικές εξισώσεις λύνονται με αριθμητικό τρόπο. Η λύση θα πρέπει να αντικατασταθεί με ένα σύνολο δεδομένων που αντιπροσωπεύουν την τιμή της λύσης σε διάφορα σημεία του χώρου, παρόλο που η λύση είναι συνεχής στην πραγματικότητα.

Η αριθμητική μέθοδος που θα χρησιμοποιήσουμε είναι μια επαναληπτική μέθοδος. Στις επαναληπτικές μεθόδους ξεκινάμε από μια αρχική υπόθεση της λύσης και ακολουθούμε μια επαναληπτική διαδικασία μέχρι να συγκλίνει η λύση μας σε μια σταθερή λύση που θα είναι κοντά στην πραγματική.

Παραδείγματα επαναληπτικών μεθόδων είναι η μέθοδος Newton και η μέθοδος Gauss-Seidel. Εμείς θα χρησιμοποιήσουμε την μέθοδο Gauss-Seidel που περιγράφεται στην επόμενη παράγραφο.

3.1 Μέθοδος Gauss - Seidel

Θα μελετήσουμε πως επιλύουμε μια διαφορική εξίσωση Poisson με τη μέθοδο Gauss-Seidel.

Έστω ότι έχουμε την διαφορική εξίσωση με μερικές παραγώγους: $\nabla^2\phi = f$ που μπορεί να γραφεί και ως:

$$\frac{\partial^2\phi}{\partial x} + \frac{\partial^2\phi}{\partial y} + \frac{\partial^2\phi}{\partial z} = f \tag{13}$$

Έστω ότι u είναι η αριθμητική λύση της εξίσωσης. Διακριτοποιώντας τη δεύτερη μερική παράγωγο με σφάλμα τάξης $O(h^2)$, μπορούμε να τη

γράψουμε στην παρακάτω μορφή:

$$\left.\frac{\partial^2\phi}{\partial x}\right|_{i,j,k} = \frac{u_{i-1,j,k} - 2u_{i,j,k} + u_{i+1,j,k}}{h_x^2},$$

$$\left.\frac{\partial^2\phi}{\partial y}\right|_{i,j,k} = \frac{u_{i,j-1,k} - 2u_{i,j,k} + u_{i,j+1,k}}{h_y^2}, \tag{14}$$

$$\left.\frac{\partial^2\phi}{\partial z}\right|_{i,j,k} = \frac{u_{i,j,k-1} - 2u_{i,j,k} + u_{i,j,k+1}}{h_z^2},$$

όπου h_i είναι το σταθερό βήμα του πλέγματος ως προς την $i-$διεύθυνση. Τοποθετώντας τη λύση μας σε ένα τρισδιάστατο πλέγμα, τα (i, j, k) είναι οι συντεταγμένες της λύσης στο πλέγμα.

Αντικαθιστώντας τους τελεστές με την αντίστοιχη αριθμητική μορφή και για $h_x = h_y = h_z = h$, η εξίσωση (13) γράφεται ως:

$$\begin{aligned}\tfrac{u_{i-1,j,k}-2u_{i,j,k}+u_{i+1,j,k}}{h^2} + \tfrac{u_{i,j-1,k}-2u_{i,j,k}+u_{i,j+1,k}}{h^2} + \\ \tfrac{u_{i,j,k-1}-2u_{i,j,k}+u_{i,j,k+1}}{h^2} = f.\end{aligned} \tag{15}$$

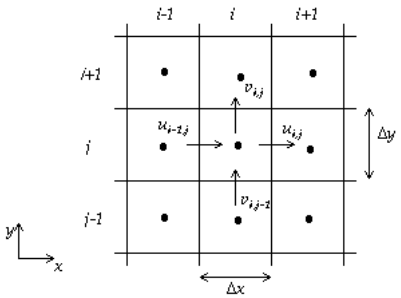
Λύνωντάς την ως προς το $u_{i,j,k}$ έχουμε:

$$\begin{aligned}u_{i,j,k} = A_i(u_{i+1,j,k} + u_{i-1,j,k}) + A_j(u_{i,j+1,k} + u_{i,j-1,k}) \\ + A_k(u_{i,j,k+1} + u_{i,j,k-1}) - A_f \cdot f,\end{aligned} \tag{16}$$

όπου

$$\begin{aligned}A_i &= \tfrac{h_y^2 h_z^2}{2D}, \\ A_j &= \tfrac{h_x^2 h_z^2}{2D}, \\ A_k &= \tfrac{h_x^2 h_y^2}{2D}, \\ A_f &= \tfrac{h_x^2 h_y^2 h_z^2}{2D}, \\ D &= h_x^2 h_y^2 + h_x^2 h_z^2 + h_y^2 h_z^2.\end{aligned} \tag{17}$$

Με τη χρήση της παραπάνω εξίσωσης, υπολογίζουμε τη λύση στο σημείο (i, j, k) εάν γνωρίζουμε τις τιμές γύρω από το σημείο αυτό. Παρακάτω βλέπουμε μια αναπαράσταση της μεθόδου για δυο διαστάσεις.



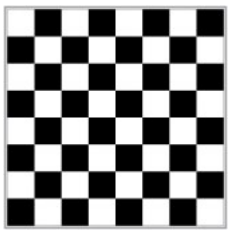
Σχήμα 1:
Μέθοδος Gauss-Seidel για δυο διαστάσεις

Για να λύσουμε τις διαφορικές μας εξισώσεις με τη μέθοδο που μελετάμε, θα πρέπει να ορίσουμε τις συνοριακές συνθήκες, να κάνουμε μια αρχική υπόθεση για τη λύση και να εκτελέσουμε όσες επαναλήψεις χρειαστούμε ώστε να επιτύχουμε την ακρίβεια που επιθυμούμε.

3.2 Red - Black

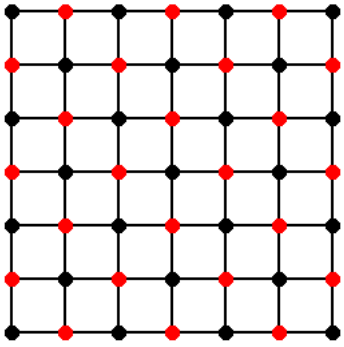
Όπως φαίνεται και από την παραπάνω παρουσίαση της αριθμητικής μεθόδου που θα χρησιμοποιήσουμε, για την επίλυση της εξίσωσης σε κάθε σημείο χρειαζόμαστε τις άνω, κάτω, δεξιά και αριστερά γειτονικές τιμές. Όμως υπάρχει κίνδυνος να γίνεται ανάγνωση και εγγραφή τιμών στην ίδια θέση μνήμης, πράγμα που θα προκαλούσε επιπλοκές, ειδικά στην περίπτωση παράλληλου προγραμματισμού. Έτσι, διαχωρίζουμε την επίλυση σε δυο μέρη.

Φανταστείτε το πλέγμα ως μια σκακιέρα. Έχοντας κατά νου την αριθμητική μέθοδο, εύκολα παρατηρούμε ότι τα λευκά σημεία είναι μεταξύ τους ανεξάρτητα, όπως και τα μαύρα. Οι τιμές εξαρτώνται μόνο από τις τιμές του άλλου χρώματος.



Σχήμα 2:
Η σκακιέρα ως ανάλογο της μεθόδου

Για να αποφύγουμε λοιπόν, τις επιπλοκές, εκτελούμε την μέθοδο πρώτα μόνο για σημεία στα ‘‘μαύρα κουτιά’’, και έπειτα στα ‘‘λευκά’’. Η τεχνική αυτή ονομάζεται *Επίλυση κόκκινου-μαύρου* (*Red-Black Relaxation*) και αναπαριστάται καλύτερα στην ακόλουθη εικόνα.



Σχήμα 3:
Αναπαράσταση της τεχνικής κόκκινου-μαύρου

3.3 Multigrid

Παρόλο που η μέθοδος που παρουσιάστηκε είναι αρκετά ακριβής και σταθερή, είναι επίσης αρκετά αργή. Ειδικά σε περίπτωση πλέγματος μεγάλης ανάλυσης. Φανταστείτε ένα πλέγμα με $100 \times 100 \times 100$ σημεία. Θα χρειαζόταν 1.000.000 πράξεις για κάθε κύκλο της μεθόδου. Πρωτού προχωρήσουμε στην υλοποίηση μπορούμε να κάνουμε κάποιες βελτιώσεις στον αλγόριθμό μας.

Μια πολύ καλή μέθοδος που θα επιταχύνει τους υπολογισμούς είναι η μέθοδος πολλαπλών πλεγμάτων (*multigrid*). Στη μέθοδο αυτή, μειώ-

νουμε τον αριθμό των υπολογισμών με το να αλλάζουμε την ανάλυση του πλέγματος.

Η διαφορική εξίσωση που επιλύουμε, μπορεί να γραφεί με τη μορφή πινάκων ως $Au = f$ όπου A είναι ο τελεστής του αριστερού μέλους της εξίσωσης, u είναι η αριθμητική λύση και f είναι η συνάρτηση πηγής (δεξί μέλος).

Ο ορίζουμε τη συνάρτηση υπολοίπου r (residual) ως:

$$r = f - Au. \tag{18}$$

Προφανώς αν η u είναι η ακριβής λύση, τότε $r = 0$. Αλλιώς, η r είναι η συνάρτηση του αριθμητικού σφάλματος. Για την ακρίβεια, το μέγεθος $\| r \|_\infty$ είναι αυτό που μας ενδιαφέρει. Αριθμητικά, αν έχουμε έναν πίνακα με τις τιμές του r σε κάθε σημείο του πλέγματος, τότε $\| r \|_\infty = \max(r_i)$.

Τώρα, ας υποθέσουμε ότι v είναι η ακριβής λύση του προβλήματος. Μπορούμε να συνδέσουμε την ακριβή λύση με την αριθμητική λύση με την σχέση:

$$v = u + e, \tag{19}$$

όπου e είναι το σφάλμα. Αυτό σημαίνει ότι η εξίσωση $Av = f$ ικανοποιείται. Έτσι, μπορούμε να πούμε:

$$\left.\begin{array}{l} Av = f \\ v = u + e \end{array}\right\} \Rightarrow A(u + e) = f \Rightarrow \\ Au + Ae = f \Rightarrow Ae = f - Au. \tag{20}$$

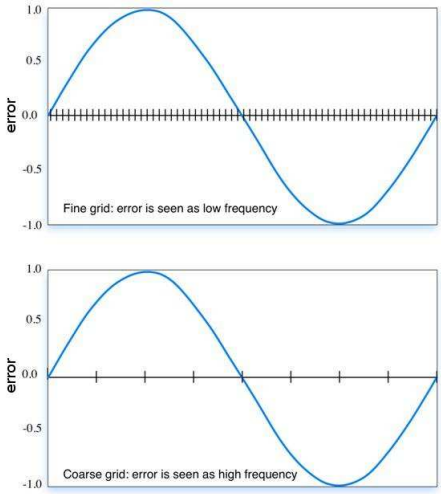
Όμως, γνωρίζουμε ότι $f - Au = r$ οπότε:

$$Ae = r, \tag{21}$$

που σημαίνει ότι μπορούμε να χρησιμοποιήσουμε το r ως συνάρτηση πηγής για να λύσουμε μια άλλη εξίσωση ως προς το σφάλμα. Αφού βρούμε το σφάλμα, επιστρέφουμε στην αρχική μας λύση και τη βελτιώνουμε - διορθώνουμε. Παρόλα αυτά, ακόμα δεν φαίνεται για ποιό λόγο να μπούμε σε αυτή τη διαδικασία.

Αν προσέξουμε καλύτερα την αριθμητική μέθοδο, θα δούμε ότι η λύση σε ένα σημείο, επιρεάζεται μόνο από τα γειτονικά σημεία. Δεν επιρεάζεται

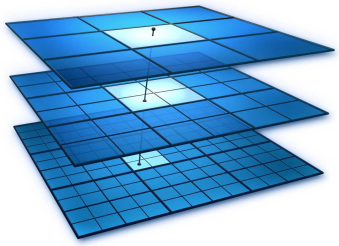
από τις πληροφορίες παραπέρα. Έτσι, αν έχουμε ένα σφάλμα ‘‘υψηλής συχνότητασ’’, πολύ εύκολα ανιχνεύεται και εξαλείφεται. Αν όμως έχουμε ένα σφάλμα πολύ ‘‘χαμηλής συχνότητασ’’, τότε θα χρειαστούν πάρα πολλές επαναλήψες για να εξαλειφθεί. Έτσι η μέθοδος θα συγκλίνει πολύ αργά.



Σχήμα 4:
Το ίδιο σφάλμα εμφανίζεται με διαφορετική συχνότητα ανάλογα με την ανάλυση του πλέγματος

Ένας καλός τρόπος για να το αντιμετωπίσουμε αυτό είναι να αλλάξουμε την ανάλυση του πλέγματος. Αυτό συμβαίνει επειδή ένα σφάλμα χαμηλής συχνότητας σε πλέγμα υψηλής ανάλυσης, εμφανίζεται ως πλέγμα υψηλής συχνότητας σε πλέγμα χαμηλής ανάλυσης, όπως φαίνεται και στο Σχήμα 4.

Έτσι, αυτό που κάνουμε στην τεχνική *multigrid* είναι αφού ορίσουμε το αρχικό πρόβλημα με τις συνοριακές συνθήκες, να κάνουμε μερικές επαναλήψεις. Έπειτα, παίρνουμε τη συνάρτησηυπολοίπου r και την εισάγουμε σε ένα πλέγμα μισής ανάλυσης. Αφού κάνουμε κι εκεί μερικές επαναλήψεις για το σφάλμα, επαναλαμβάνουμε τη διαδικασία για το σφάλμα του σφάλματος. Μόλις φτάσουμε στο υψηλότερο επίπεδο, δηλαδή αυτό με το μικρότερο δυνατό αριθμό σημείων, επιστρέφουμε στα προηγούμενα, διορθώνοντας τις λύσεις, μέχρι που φτάνουμε στην αρχική μας λύση και την διορθώνουμε με το ήδη διορθωμένο σφάλμα.



Σχήμα 5:
Τα επίπεδα διαφορετικών αναλύσεων στο multigrid

Ο τελεστής που μετατρέπει ένα πλέγμα σε ένα άλλο μισής ανάλυσης, ονομάζεται restriction και είναι ένας τελεστής παρεμβολής. Η τυπική διαδικασία για ένα διδιάστατο πλέγμα αναπαρίσταται με το σχήμα:

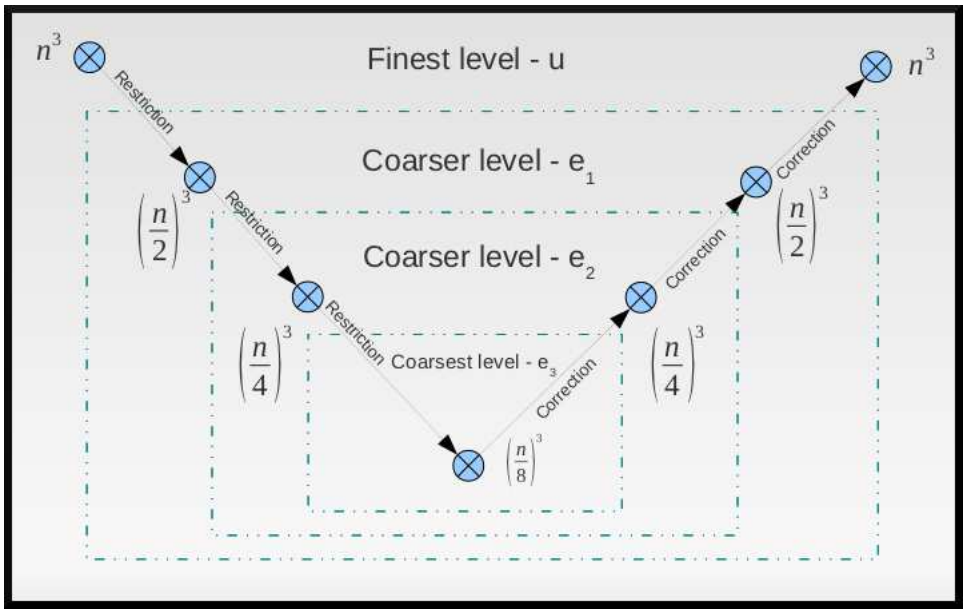
$$I_h^H = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \tag{22}$$

Το παραπάνω σχήμα ονομάζεται και *full weighting restriction*.

Από την άλλη μεριά, ο τελεστής που μετατρέπει ένα πλέγμα σε ένα άλλο διπλάσιας ανάλυσης, λέγεται correction και είναι επίσης ένας τελεστής παρεμβολής. Σχηματικά, μπορεί να αναπαρασταθεί ως (για 2 διαστάσεις):

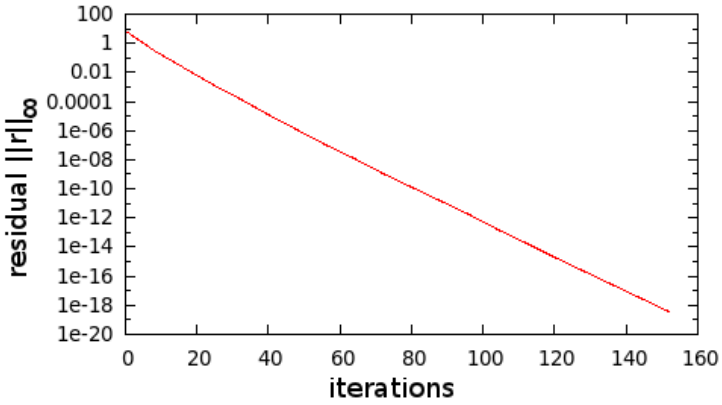
$$I_H^h = \frac{1}{4}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \tag{23}$$

Στην τεχνική *multigrid* ξεκινάμε από ένα βασικό επίπεδο και βρίσκουμε τη λύση. Στη συνέχεια πάμε σε ανώτερα επίπεδα (μικρότερη ανάλυση) όπου λύνουμε για το σφάλμα. Έπειτα, επιστρέφουμε πίσω στο βασικότερο επίπεδο και διορθώνουμε τη λύση. Ο κύκλος αυτός ονομάζεται κύκλος - V (V-cycle) και αναπαρίσταται στο παρακάτω γράφημα.

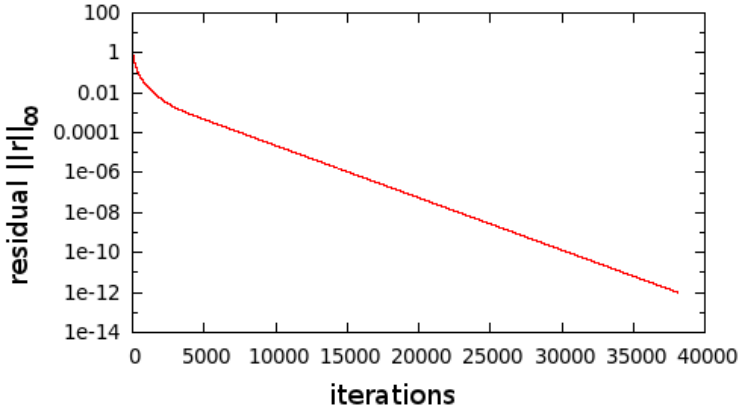


Σχήμα 6:
Ένας πλήρης κύκλος -V (V-cycle)

Για να δούμε την διαφορά μεταξύ Multigrid και απλής μεθόδου, λύνουμε τη διαφορική εξίσωση Poisson για ακρίβεια δώδεκα δεκαδικών ψηφίων. Στα σχήματα 7 και 8 φαίνεται η ταχύτητα σύγκλισης της μεθόδου Gauss-Seidel με και χωρίς Multigrid.



Σχήμα 7:
Gauss-Seidel με Multigrid



Σχήμα 8:
Gauss-Seidel χωρίς Multigrid

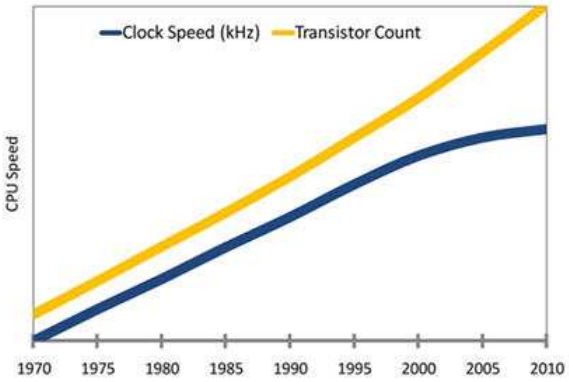
Παρατηρούμε ότι για να πετύχουμε ακρίβεια δώδεκα δεκαδικών ψηφίων, χωρίς Multigrid χρειαστήκαμε περίπου 35000 επαναλήψεις, ενώ με τη μέθοδο Multigrid μόνο 90 επαναλήψεις. Είναι μια αξιόλογη επιτάχυνση, χωρίς κάποιο κόστος, εκτός από την διαδικασία υλοποίησης της τεχνικής.

4 Προγραμματισμός σε GPU

Το επόμενο βήμα είναι να επιλέξουμε μια κατάλληλη γλώσσα προγραμματισμού ώστε να έχουμε βέλτιστα αποτελέσματα. Πρώτα απ΄ όλα ας σκεφτούμε το υλικό που θα χρησιμοποιήσουμε.

Καθώς τα μοντέλα μας γίνονται όλο και πιο περίπλοκα, χρειαζόμαστε όλο και περισσότερη υπολογιστική ισχύ. Από την άλλη, όσο εξελίσσεται η τεχνολογία και έχουμε ισχυρότερους υπολογιστές, καλούμαστε να αντιμετωπίσουμε όλο και μεγαλύτερες προκλήσεις.

Σύμφωνα με τον νόμο του Moore ο αριθμός των τρανζίστορ σε ένα ολοκληρωμένο κύκλωμα περίπου διπλασιάζεται κάθε ενάμισι με δυο χρόνια. Παρόλα αυτά τα τελευταία χρόνια έχει παρατηρηθεί ένας χαμηλότερος ρυθμός αύξησης της υπολογιστικής ισχύος. Ο λόγος είναι ότι η ανάπτυξη της υπολογιστικής ισχύος στους επεξεργαστές πλησιάζει σε κάποια φυσικά όρια.



Σχήμα 9:
Ο νόμος του Moore

Υπάρχουν πολλοί λόγοι για τους οποίους έχει επιβραδυνθεί η ανάπτυξη των επεξεργαστών. Ένας από αυτούς είναι και οι περιορισμοί στις διαστάσεις. Χρειάζεται σχεδίαση όλο και μικρότερων κυκλωμάτων, αλλά αυτό παρουσιάζει τεχνικές δυσκολίες.

Ένας άλλος λόγος είναι η ψύξη. Για να αυξηθεί η απόδοση του επεξεργαστή απαιτείται και περισσότερη ενέργεια. Όμως ήδη οι επεξεργαστές τροφοδοτούνται με ρεύμα πάνω από 200A. Χωρίς επαρκή ψύξη, αναπτύσσονται υψηλές θερμοκρασίες και λιώνουν τα κυκλώματα.

Όλα τα παραπάνω μπορούν να αντιμετωπισθούν με κάποιον τρόπο. Ωστόσο, υπάρχει και ο περιορισμός της ατομικής κλίμακας. Δεν μπορούμε να κατασκευάσουμε μικρότερα κυκλώματα διότι κβαντικά φαινόμενα, όπως το φαινόμενο σήραγγας, προκαλούν δυσλειτουργίες.

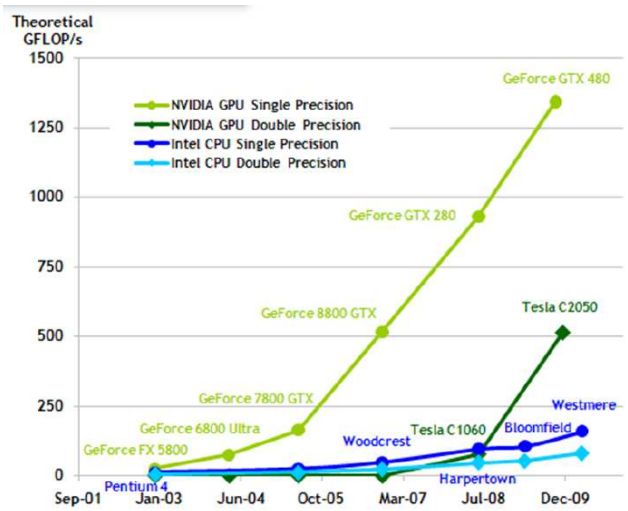
Η λύση στο πρόβλημα αυτό είναι ο παράλληλος προγραμματισμός. Με τον παράλληλο προγραμματισμό χρησιμοποιούμε πολλούς επεξεργαστές, οπότε μειώνεται και η απαίτηση της ισχύος του κάθε επεξεργαστή. Ωστόσο, χρειαζόμαστε πολλούς υπολογιστές και γενικά ακριβές εγκαταστάσεις. Σήμερα, η βιομηχανία υπολογιστών επενδύει στην παραγωγή πολυπύρηνων επεξεργαστών. Έτσι, κυκλοφορούν επεξεργαστές με πυρήνες μέχρι και 16.

Με τον παράλληλο προγραμματισμό αντιμετωπίζονται αρκετές από τις δυσκολίες, υπάρχει όμως ένα κόστος στην εγκατάσταση ενός ολόκληρου δικτύου υπολογιστών. Υπάρχει όμως και μια άλλη επιλογή.

4.1 GPUs

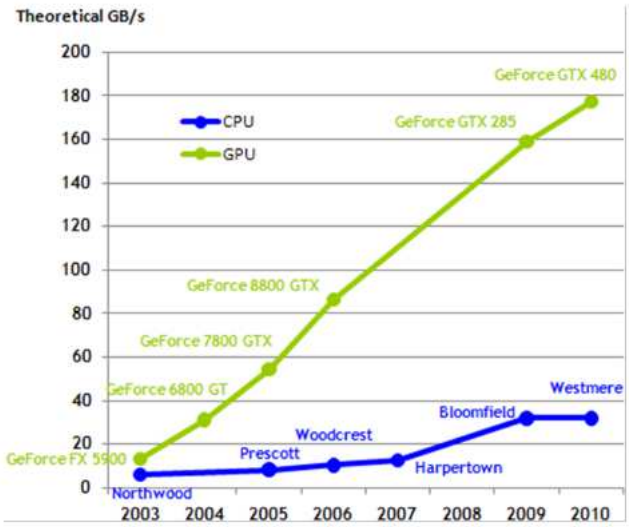
Η βιομηχανία παιχνιδιών είναι μια από τις πιο απαιτητικές αλλά και μεγάλες βιομηχανίες του κόσμου. Τα παιχνίδια γίνονται όλο και πιο ρεαλιστικά με υψηλή ευκρίνεια και ποιότητα κίνησης, με προσομοίωση φυσικών φαινομένων στον χώρο που εξελίσσεται το παιχνίδι, βέλτιστη διάχυση φωτός και λοιπά. Όλα αυτά απαιτούν υψηλή επεξεργαστική ισχύ και μάλιστα σε παράλληλη επεξεργασία.

Οι ανάγκες αυτές οδήγησαν τη βιομηχανία των υπολογιστών στην κατασκευή ειδικών καρτών γραφικών. Οι κάρτες γραφικών της NVIDIA επιδέχονται υψηλού επιπέδου παραλληλισμό με πολλούς μικροεπεξεργαστές, με πολλούς πυρήνες, έχοντας υψηλές επιδόσεις, τόσο στην επεξεργασία όσο και στην ταχύτητα μνήμης.



Σχήμα 10:
Μέγιστη απόδωση των CPU και GPU

Στο σχήμα 10 βλέπουμε τη μέγιστη απόδοση των επεξεργαστών (CPU) και των επεξεργαστών γραφικών (GPU) για συσκευές απλής και διπλής ακρίβειας. Στο σχήμα 11 φαίνεται η αντίστοιχη επίδοση στις ταχύτητες μνήμης.



Σχήμα 11:
Η ταχύτητα μνήμης των CPU και GPU

Ο λόγος που οι επεξεργαστές γραφικών είναι τόσο αποδοτικοί έναντι των συμβατικών επεξεργαστών, είναι επειδή οι επεξεργαστές αυτοί είναι ειδικά σχεδιασμένοι για παράλληλη επεξεργασία δεδομένων - ακριβώς ό,τι απαιτεί η επεξεργασία γραφικών - και έτσι έχουν περισσότερα κυκλώματα αφιερωμένα σε αριθμητικές πράξεις και λιγότερα στη διαχείριση μνήμης. Αυτό φαίνεται σχηματικά στο σχήμα 12.

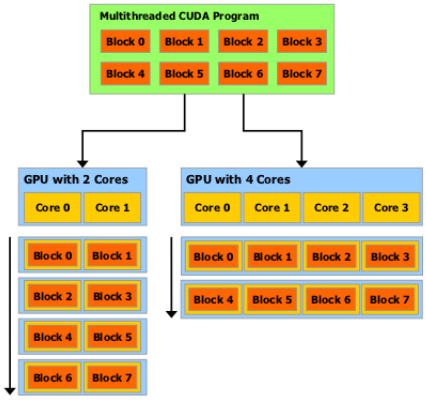
Σχήμα 12:
Η αρχιτεκτονική των CPU και GPU

Ένα από τα βασικά πλεονεκτήματα των GPU είναι ότι μπορούμε να έχουμε όλη αυτή την επεξεργαστική ισχύ σε μια απλή κάρτα γραφικών και πολύ πιο οικονομικά, από μεγάλα συστήματα υπερ-υπολογιστών.

4.2 CUDA

Η γλώσσα προγραμματισμού που θα χρησιμοποιήσουμε είναι η CUDA της NVIDIA. Η CUDA είναι μια γλώσσα προγραμματισμού που είναι ιδιαίτερα εύκολη για κάποιον που έχει μια βάση στον προγραμματισμό σε C.

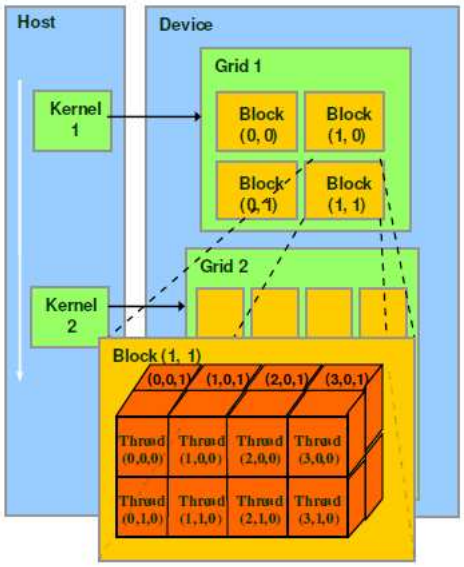
Στον προγραμματισμό σε CUDA τα νήματα (threads) ομαδοποιούνται σε μπλόκ τα οποία προσαρμόζονται ανάλογα με τους πυρήνες της συσκευής, όπως φαίνεται στο παρακάτω σχήμα.



Σχήμα 13:
Προσαρμογή των μπλοκ ανάλογα με τους πυρήνες της GPU

Έτσι, το πρόγραμμα είναι αυτο-προσαρμόσιμο ανάλογα με τις δυνατότητες της συσκευής.

Το σημαντικότερο που πρέπει να γίνει κατανοητό από έναν προγραμματιστή σε CUDA είναι η δομή των νημάτων μέσα στη συσκευή. Αυτή περιγράφεται καλύτερα στο σχήμα 14:



Σχήμα 14:
Κατανομή νημάτων σε πλέγμα και μπλοκ

Υπάρχει, το βασικό διδιάστατο πλέγμα, όπου κατανέμονται τα τρισδιάστατα μπλοκ και το καθένα έχει τις δικές του συντεταγμένες στο πλέγμα. Μέσα

σε κάθε μπλοκ το κάθε νήμα έχει τις συντεταγμένες του. Έτσι, το κάθε νήμα έχει έναν μοναδικό αριθμό που το χαρακτηρίζει, και εξαρτάται από τις συντεταγμένες του μπλοκ και του νήματος μέσα στο μπλοκ.

Για να ορίσουμε μια συνάρτηση που θα εκτελεστεί στην κάρτα γραφικών, αρκεί να χρησιμοποιήσουμε την κωδική λέξη __global__:

```
__global__ void function(float x){
...
}
```

και για να την καλέσουμε χρησιμοποιούμε την παρακάτω μορφή:

`function`$<<< dg, db >>>$`(x);`

όπου dg και db είναι οι διαστάσεις του πλέγματος και των μπλοκ, αντίστοιχα.

5 Η προσομοίωση

Έχοντας δει τη θεωρία και τις αριθμητικές μεθόδους αλλά και τον τρόπο προγραμματισμού που θα χρησιμοποιήσουμε, ας δούμε πως θα τα συνδυάσουμε όλα αυτά για να επιλύσουμε το πρόβλημά μας.

Η βασική δομή του προγράμματος είναι παρόμοια με αυτή της C/C++. Αρχικά, ορίζουμε τις συναρτήσεις που θα χρειαστούμε στο κυρίως πρόγραμμα. Οι συναρτήσεις αυτές είναι:

- Συνάρτηση για επίλυση της εξίσωσης TOV
- Συνάρτηση για επίλυση ελλειπτικών εξισώσεων
- Συνάρτηση για αλλαγή ανάλυσης ενός πλέγματος σε μισή (Restriction)
- Συνάρτηση για αλλαγή ανάλυσης πλέγματος σε διπλάσια (Prolongation)
- Συνάρτηση για τον υπολογισμό του r (residual) και του $\|r\|_\infty$
- Συνάρτηση για την εξαγωγή των αποτελεσμάτων σε αρχεία.
- μερικές βοηθητικές συναρτήσεις, όπως για αντιγραφή από την κεντρική μνήμη στη μνήμη της συσκευής.

Στο κεντρικό πρόγραμμα (main) αρχικά διαβάζουμε κάποιες σταθερές από εξωτερικά αρχεία, στη συνέχεια υπολογίζουμε τις συναρτήσεις πυκνότητας, μάζας και πίεσης, από τις εξισώσεις TOV και αρχικοποιούμε την συνάρτηση πηγής, τα σφάλματα και τα σύνορα της λύσης. Στη συνέχεια γίνονται 30 κύκλοι V και σε κάθε κύκλο γίνονται οι ανάλογοι υπολογισμοί και οι μετατροπές.

Τέλος, όταν ολοκληρωθεί η διαδικασία επίλυσης, εξάγουμε τη λύση σε εξωτερικά αρχεία και καθαρίζουμε τη μνήμη από τα pointers που είχαμε δεσμεύσει.

Επειδή υπάρχει και μη-γραμμικός όρος στο δεξί μέλος της διαφορικής εξίσωσης, αυτό αντιμετωπίζεται με το να κάνουμε αρχική υπόθεση για την λύση, να την θεωρίσουμε σταθερά στο δεξί μέλος της εξίσωσης, και μόλις βρούμε μια νέα λύση να την αντικαταστήσουμε στο δεξί μέλος. Επαναλαμβάνουμε αυτή τη διαδικασία μέχρι που να συγκλίνει η λύση μας και να επιτύχουμε την ακρίβεια που επιθυμούμε (πέντε δεκαδικά ψηφία).

6 Βελτιστοποίηση

Κατά τη διαδικασία βελτιστοποίησης, αναζητούμε τρόπους να ελαχιστοποιήσουμε τις όποιες καθυστερήσεις, ώστε το πρόγραμμά μας να είναι όσο το δυνατόν ταχύτερο.

Μια διαδικασία που είναι ιδιαίτερα χρονοβόρα είναι η ανταλλαγή δεδομένων μεταξύ κύριας μνήμης και της μνήμης συσκευής. Το πρόβλημα αυτό αντιμετωπίζεται με το να φτιάχνουμε επιπλέον συναρτήσεις που θα εκτελεστούν στη συσκευή και θα κάνουν τις απαραίτητες αρχικοποιήσεις χωρίς να χρειαστεί να γίνεται αυτό στην κύρια μνήμη.

Μια άλλη διαδικασία βελτιστοποίησης είναι να διαβάσουμε τις πληροφορίες από την κάρτα γραφικών και να δούμε ποιες είναι οι μέγιστες επιτρεπτές διαστάσεις του κάθε μπλοκ. Ορίζοντας αυτές ως διαστάσεις των μπλοκ, δεν υπάρχει μικροεπεξεργαστής που μένει ανεκμετάλλευτος. Έτσι, μεγιστοποιείται η απόδοση.

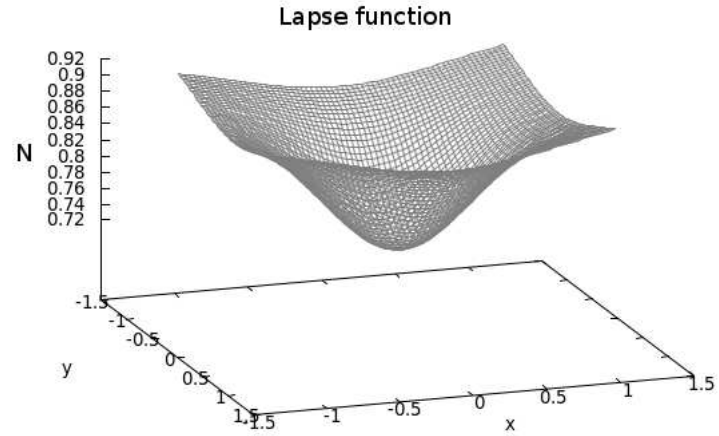
Υλοποιώντας τις παραπάνω βελτιστοποιήσεις, επιτύχαμε επιτάχυνση περίπου 5 φορές.

7 Τελικά αποτελέσματα

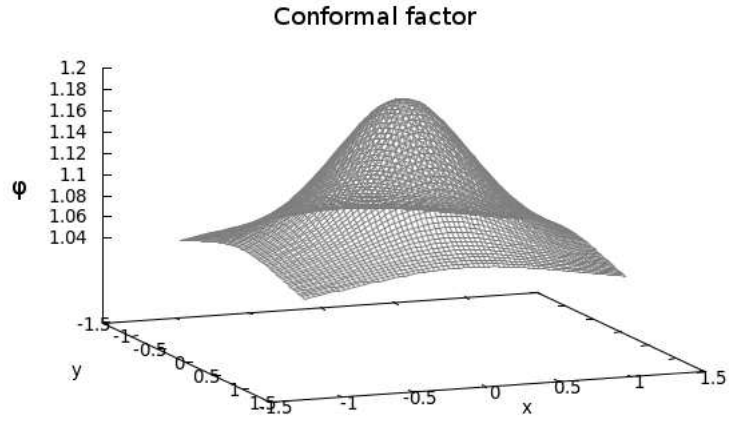
Οι αρχικές δοκιμές έγιναν σε έναν υπολογιστή με 480 πυρήνες CUDA, διπλής ακρίβειας. Το αποτέλεσμα ήταν η λύση να προκύψει σε $\boxed{8.96}$ δευτερόλεπτα. Όμως αυτό ήταν χωρίς την βελτιστοποίηση. Το ίδιο πρόγραμμα σε έναν υπολογιστή με 336 πυρήνες απλής ακρίβειας, χρειάστηκε περίπου 2.75 δευτερόλεπτα. Αποτέλεσμα λογικό μιας και οι επεξεργαστές απλής ακρίβειας είναι 4.2 φορές ταχύτεροι από αυτούς διπλής ακρίβειας.

Μετά τη βελτιστοποίηση, το πρόγραμμα έδωσε τα ίδια αποτελέσματα σε χρόνο $\boxed{0.47}$ δευτερόλεπτα σε υπολογιστή με 336 CUDA-cores απλής ακρίβειας.

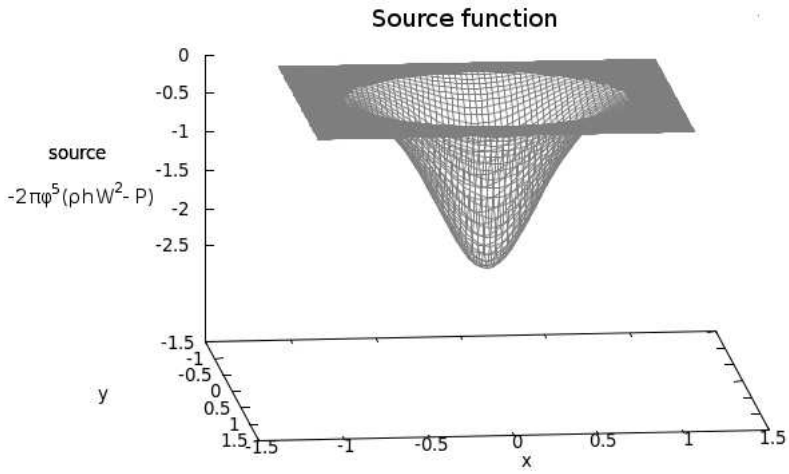
Παρακάτω φαίνονται και γραφικά τα αποτελέσματα:



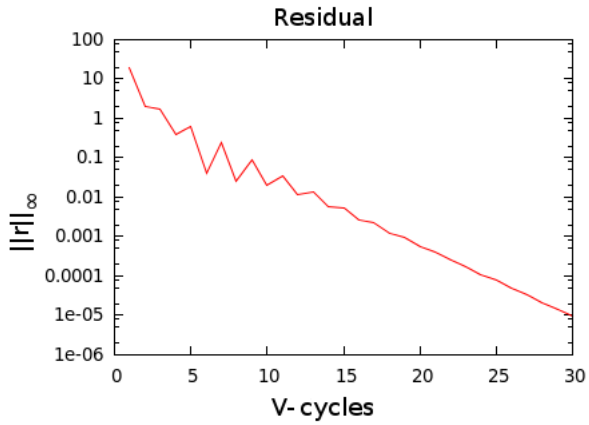
Σχήμα 15:
Η συνάρτηση N (lapse function).



Σχήμα 16:
Η αριθμητική προσέγγιση της λύσης ϕ.



Σχήμα 17:
Η πηγή (s) για την λύση φ.



Σχήμα 18:
Η ακρίβεια της αριθμητικής λύσης συναρτήσει των κύκλων-V

Οι αρχικές διαταραχές στη σύγκλιση της συνάρτησης υπολοίπου $\|r\|_\infty$ ωφείλονται στη μη-γραμμικότητα του προβλήματος σε συνδυασμό με το γεγονός ότι αρχικά κάνω μια πολύ προσεγγιστική υπόθεση για τη λύση.

Πηγές

1. *Relativistic simulations of rotational core collapse I. Methods, initial models, and code tests H. Dimmelmeier, J.A. Font, and E. Müller, A&A 388, 917 - 935 (2002)*

2. *Black Holes, White Dwarfs and Neutron stars: THE PHYSICS OF COMPACT OBJECTS - Stuart L. Shapiro, Saul A. Teukolsky, Wiley-Interscience, 1983*

3. *A Multigrid Tutorial, Second Edition - William L. Briggs, Van Emden Henson, Steve F. McCormic, SIAM, 2000*

4. *Wikipedia (http://www.wikipedia.org/)*

5. *NVIDIA CUDA C Programming Guide v3.1 - NVIDIA, www.nvidia.com*