ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Σχεδιασμός& Ανάπτυξη μαθηματικής εφαρμογής σε γλώσσα GO για το Google Cloud App Engine

/

Design and Implementation of a mathematical application in GO for the Google Cloud App Engine

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΤΟΥ

ΙΩΑΝΝΗ ΣΠΑΝΑΚΗ

Βόλος, 2013

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Σχεδιασμός& Ανάπτυξη μαθηματικής εφαρμογής σε γλώσσα GO για το Google Cloud App Engine

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΙΩΑΝΝΗ ΣΠΑΝΑΚΗ

Επιβλέποντες :

Τσομπανοπούλου Παναγιώτα	Μποζάνης Παναγιώτης
Επίκουρος Καθ. - Παν. Θεσσαλίας	Αναπληρωτής Καθ. – Παν. Θεσσαλίας

Εγκρίθηκε από την διμελή εξεταστική επιτροπή την

(Υπογραφή)	*(Υπογραφή)*
.....	.....
Τσομπανοπούλου Παναγιώτα	Μποζάνης Παναγιώτης
Επίκουρος Καθ. - Παν. Θεσσαλίας	Αναπληρωτής Καθ. -Παν.Θεσσαλίας

(Υπογραφή)

.....

ΙΩΑΝΝΗΣ ΣΠΑΝΑΚΗΣ

Διπλωματούχος Μηχανικός Ηλεκτρονικών Υπολογιστών, Τηλεπικοινωνιών και Δικτύων του Τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστημίου Θεσσαλίας

© 2013. – All rights reserved

Η σελίδα αυτή είναι σκόπιμα λευκή.

Περιεχόμενα

Κεφάλαιο 1:Εισαγωγή……11

1.1 Αντικείμενο της Διπλωματικής……11

1.2 Οργάνωση Κειμένου……...11

Κεφάλαιο 2: CLOUD COMPUTING……..……12

2.1 Εισαγωγή-Ορισμός……...……12

2.2 Πλεονεκτήματα……12

2.3 Κύρια Χαρακτηριστικά……13

2.4 Μοντέλα Υπηρεσιών……..……14

2.5 Μοντέλα Ανάπτυξης…….…….....15

2.6 Παραδείγματα Χρήσης του «νέφους».....16

2.7 Επιπλοκές.....17

Κεφάλαιο 3: Google Application Engine (GAE/AppEngine)……..……..19

3.1 Εισαγωγή……...19

3.2 Ανάπτυξη Λογισμικού & Υπηρεσίες ……19

3.3 Κλιμάκωση Πόρων-Απόδοση…….....22

3.4 «Ζώνη» Δωρεάν Πόρων……..……...23

3.5 Διαχείριση της Εφαρμογής (Κονσόλα Διαχειριστή).....24

3.6 Προστασία Δεδομένων 25

Κεφάλαιο 4: Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ «Go»……...26

4.1 Εισαγωγή……...……....26

4.2 Στόχοι-Αρχές……..……..……26

4.3 Εκδόσεις……..……..…….....27

4.4 Σύστημα Τύπων & Δομές Ελέγχου……..…….....27

4.5 Block, Δηλώσεις& Εμβέλεια……..……29

4.6 Συγχρονισμός……..…….....30

ΚΕΦΑΛΑΙΟ 5: Η εφαρμογή“GO linear algebra calculator”.....32

5.1 Εισαγωγή.....32

5.2 Οργάνωση Κώδικα.....32

5.3 Είσοδος-Έξοδος.....33

5.4 Επεξεργασία Δεδομένων.....34

5.5 Περιορισμοί.....35

5.6 Αλγόριθμοι.....35

ΚΕΦΑΛΑΙΟ 6: Δοκιμαστικοί Πίνακες.....39

6.1 Εισαγωγή……..….39

6.2 SHERMAN2: Oil reservoir simulation……..….39

6.3 SHERMAN4: Oil reservoir simulation……..….40

6.4 BCSSTK27: BCS Structural Engineering……..…...40

6.5 BCSSTM19: BCS Structural Engineering……..…41

6.6 Hilbert 500……..……42

6.7 Wilkinson 500…….…..43

Βιβλιογραφία.....44

Κεφάλαιο 1: Εισαγωγή

1.1 Αντικείμενο της Διπλωματικής

Ο στόχος αυτής της διπλωματικής εργασίας είναι η επισκόπηση των δυνατοτήτων της πλατφόρμας υπολογιστικού «νέφους» (Cloud) Google Application Engine και της αναδυόμενης γλώσσας προγραμματισμού «Go» και το συνδυασμό τους για την δημιουργία μιας διαδικτυακής εφαρμογής που προσανατολίζεται στην επίλυση συστημάτων γραμμικών εξισώσεων σε μορφή πίνακα. Η εφαρμογή βασίζεται στη χρήση 5 επαναληπτικών μεθόδων κατ'επιλογή του χρήστη. Προσφέρει 5 εναλλακτικές μεθόδους, τις Gauss-Seidel, Jacobi, SOR, Steepest Gradient Descent και Conjugate Gradient. Επίσης 2 βοηθητικές μεθόδους , την Power Iteration Method για την εύρεση φασματικής ακτίνας πίνακα και την συνάρτηση "Det" για τον υπολογισμό ορίζουσας πίνακα. Η ανάπτυξη και διάθεση της στο περιβάλλον του Google AppEngine εγγυάται υψηλή προσβασιμότητα και ικανοποιητική διαθεσιμότητα σε σχέση με το μηδενικό κόστος.

1.2 Οργάνωση Κειμένου

Στο 2ο Κεφάλαιο παρουσιάζονται τα κύρια χαρακτηριστικά του υπολογιστικού «νέφους»(cloud computing) καθώς και τα βασικά μοντέλα ανάπτυξης και υπηρεσιών τα οποία το χαρακτηρίζουν. Στο 3ο Κεφάλαιο δίνεται ένα περίγραμμα της λειτουργίας και τονίζονται βασικά χαρακτηριστικά της πλατφόρμας Google Application Engine βάσει της οποίας αναπτύχθηκε, λειτουργεί και διατίθεται η εφαρμογή. Στο 4ο Κεφάλαιο δίνεται μια περιγραφή και επισκόπηση των δυνατοτήτων της γλώσσας Go, με τη χρήση της οποίας έγινε η συγγραφή του κώδικα της εφαρμογής. Στο 5ο Κεφάλαιο αναλύεται η ίδια η εφαρμογή όσον αφορά το σχεδιασμό, την ανάπτυξη, τη λειτουργικότητα και την απόδοση της, ενώ στο 6Ο Κεφάλαιο παρατίθενται αποτελέσματα από δοκιμαστικούς πίνακες που δόθηκαν σαν είσοδος στην εφαρμογή.

ΚΕΦΑΛΑΙΟ 2: CLOUD COMPUTING

2.1 Εισαγωγή-Ορισμός

Επιστημονικά ο όρος «νέφος» χρησιμοποιείται για να περιγράψει μια ευρεία συγκέντρωση στοιχείων τα οποία οπτικά και εξ αποστάσεως ομοιάζουν με νέφος (με την ευρέως γνωστή έννοια του μετεωρολογικού νέφους) καθώς επίσης και για οποιαδήποτε συλλογή στοιχείων όταν δε γίνεται για αυτά περεταίρω επισκόπηση των ιδιοτήτων τους . Ο όρος είναι ιδιαίτερα δόκιμος για να περιγράψει την τεχνολογία του cloud computing («υπολογιστικό νέφος»). Αυτό διότι οι χρήστες του «νέφους» έχουν τη δυνατότητα πρόσβασης και χρήσης του μεγάλου εύρους των υπηρεσιών που προσφέρει χωρίς να χρειάζεται να γνωρίζουν τους μηχανισμούς και τις λεπτομέρειες των υποδομών που επιτελούν τη λειτουργικότητα αυτή.

Προχωρούμε σε ένα τυπικό ορισμό του «cloud» ως εξής:

Το cloud computing είναι ένα διαδικτυακό μοντέλο για την παροχή εύκολης, κατ'απαίτηση, πρόσβασης σε μια πηγή παραμετροποιήσιμων πόρων, όπως δίκτυο, servers, αποθηκευτικός χώρος, εφαρμογές και άλλες υπηρεσίες, οι οποίοι μπορούν να δεσμευθούν/αποδεσμευθούν με τον ελάχιστο δυνατό κόπο και επικοινωνία με τον πάροχο.

Εμβαθύνοντας στην έννοια της κατ’απαίτηση πρόσβασης, που αποτελεί και θεμελιώδες χαρακτηριστικό του «νέφους», εννοούμε ότι ο χρήστης έχει τη δυνατότητα να επιλέξει, από ένα κατάλογο υπηρεσιών στον οποίο γνωστοποιούνται, εκ μέρους του πάροχου , το κόστος και η λειτουργικότητα κάθε υπηρεσίας και να κάνει άμεσα χρήση αυτής/αυτών καθώς και να επωμιστεί το κόστος χωρίς περεταίρω επαφή με τον πάροχο.

2.2 Πλεονεκτήματα

Η τεχνολογία του υπολογιστικού νέφους βασίζεται στην συντονισμένη κοινή χρήση πόρων ώστε να επιτυγχάνονται «οικονομίες κλίμακας». Με τον οικονομικό αυτόν όρο εννοούμε την μείωση του μέσου κόστους ενός προϊόντος όταν αυξάνεται το μέγεθος της παραγωγής ή της διαθεσιμότητάς του. Μεγάλες υποδομές με υπολογιστικό υλικό και το λογισμικό που το πλαισιώνει γίνονται διαθέσιμες σε μεγάλο αριθμό από χρήστες που επωμίζονται από κοινού το κόστος λειτουργίας, καθιστώντας το προσιτό για καθέναν από αυτούς. Ο τρόπος ομοιάζει με την εμπορική διάθεση προϋπαρχόντων δημόσιων αγαθών όπως το ηλεκτρικό ρεύμα μιας και το κόστος είναι ευθέως ανάλογο της έντασης της χρήσης της υπηρεσίας.

Για τις επιχειρήσεις που χρειάζονται να κάνουν χρήση υπολογιστικών υποδομών τα οφέλη είναι πολύ σημαντικά. Κυρίως, αντί η ανάγκη να καλυφθεί από αρχική επένδυση, αγοράζοντας τις υποδομές αυτές, γίνεται μέρος των λειτουργικών εξόδων αφού η επιχείρηση πλέον μπορεί να «ενοικιάσει» τις υποδομές αυτές από παρόχους . Επιπλέον, οι επιχειρήσεις οι οποίες προσφέρουν τις υπηρεσίες τους μέσω ιδιωτικών servers οφείλουν να διατηρούν περισσότερους υπολογιστικούς πόρους από αυτούς που χρειάζονται κατά μέσο όρο προκειμένου να είναι συνεπής με τους πελάτες τους σε περιόδους αυξημένης ζήτησης, έχοντας έτσι χαμηλό ποσοστό χρησιμοποίησης για τις υποδομές αυτές, δυσκολεύοντας την απόσβεση. Αυτό το πρόβλημα μπορεί να λυθεί με τη μεταφορά της προσφερόμενης υπηρεσίας στο «νέφος», το οποίο υπόσχεται ταχεία κλιμάκωση των προσφερόμενων πόρων ανά πάσα στιγμή, χωρίς την παρέμβαση του πελάτη.

Ακόμα, η ύπαρξη τέτοιων μεγάλων υπολογιστικών κέντρων που χρησιμοποιούνται από κοινού σε υψηλή χρησιμοποίηση μειώνει σημαντικά την ανάγκη για διάσπαρτες υποδομές ανά οργανισμό που συνολικά είναι πλεονασματικές ως προς τις ανάγκες που καλύπτουν. Έτσι η κατανάλωση ενέργειας για υπολογιστικούς πόρους χαμηλώνει σε παγκόσμιο επίπεδο κάνοντας τη τεχνολογία περιβαλλοντικά φιλική.

2.3 Κύρια Χαρακτηριστικά

Κατά το Εθνικό Ινστιτούτο Προτύπων και Τεχνολογίας των Η.Π.Α (ειδική έκδοση 800-145), μια πλατφόρμα υπολογιστικού «νέφους» οφείλει να επιδεικνύει τα εξής 5 χαρακτηριστικά :

-Εξυπηρέτηση κατ'απαίτηση: ο πελάτης πρέπει να μπορεί να προμηθευτεί τους επιθυμητούς πόρους ή υπηρεσίες της πλατφόρμας μονομερώς χωρίς περεταίρω συνεννόηση με τον εκάστοτε πάροχο.

-Ευρεία προσβασιμότητα μέσω δικτύου:Οι δυνατότητες της πλατφόρμας οφείλουν να είναι προσβάσιμες μέσω δικτύου και προτυποποιημένων μηχανισμών που διευκολύνουν τη χρήση από ετερογενή μεταξύ τους μέσα όπως κινητά, tablet και Η/Υ.

-Συγκεντρωμένοι πόροι: Οι υπολογιστικοί πόροι του παρόχου συγκεντρώνονται (όχι υποχρεωτικά σε φυσική τοποθεσία αλλά «λογικά») ώστε να εξυπηρετούν πολλαπλούς πελάτες ταυτόχρονα με το μοντέλο της «πολύ-ενοικίασης»(multi-tenancy) της συνολικής υποδομής, δεσμεύοντας και αποδεσμεύοντας πόρους σε πελάτες ανά πάσα στιγμή ανάλογα με τις ανάγκες τους.

-Ταχεία ελαστικότητα: Η πλατφόρμα οφείλει να μπορεί να αυξάνει και να μειώνει τους πόρους που έχουν δεσμευθεί σε πελάτη με υψηλή ταχύτητα ώστε να επιδεικνύει συνέπεια σε διαστήματα υψηλής ζήτησης μεν αλλά να μην επιβαρύνει τον πελάτη όταν αυτοί οι πόροι είναι περιττοί.

-Επιμετρούμενες υπηρεσίες: Οι πλατφόρμες οφείλουν να ελέγχουν και να βελτιστοποιούν τις παρεχόμενες υπηρεσίες χρησιμοποιώντας μηχανισμούς μέτρησης κατάλληλους για τον τύπο της υπηρεσίας (π.χ. το μέγεθος του αποθηκευτικού χώρου, τον χρόνο χρήσης της κεντρικής μονάδας επεξεργασίας, το πλήθος των ενεργών χρηστών κ.α.). Οι πόροι που χρησιμοποιούνται για μια υπηρεσία παρακολουθούνται και ελέγχονται με διαφανή τρόπο σε πελάτη και πάροχο.

2.4 Μοντέλα Υπηρεσιών

Οι πλατφόρμες υπολογιστικού «νέφους» προσφέρουν 3 βασικά μοντέλα υπηρεσιών: Υπηρεσίες Υποδομής(Infrastructure as a Service/IaaS), Υπηρεσίες (προγραμματιστικής) Πλατφόρμας (Platform as a Service/PaaS) και Υπηρεσίες Λογισμικού (Software as a Service/SaaS), τα οποία λειτουργούν σαν «στοίβα» εφόσον το ένα επίπεδο χρησιμοποιεί βασικές υπηρεσίες του προηγούμενου (IaaS->PaaS->SaaS).

Αναλυτικότερα:

-IaaS : Σε αυτό το μοντέλο υπηρεσίας ο πάροχος προσφέρει στοιχεία υπολογιστικής και δικτυακής υποδομής. Ανάμεσα σε αυτά δεσπόζει η «εικονική μηχανή», η οποία είναι ένα αφαιρετικό στοιχείο υπολογιστικής υποδομής το οποίο έχει ανάμεσα σε άλλα τη δυνατότητα να αυξάνει και να μειώνει του πόρους που χρησιμοποιεί, οι οποίοι ανήκουν σε φυσικές μηχανές και είτε είναι ποσοστό μιας είτε συγκέντρωση πολλών. Άλλα στοιχεία που προσφέρονται είναι: «αναπαραστάσεις» μονάδων αποθήκευσης, firewall, εξισορροπητές φόρτου, διευθύνσεις IP κ.α. Ο πελάτης κάνει χρήση της υποδομής μεταφορτώνοντας εκεί συμβατό λειτουργικό σύστημα και άλλο απαραίτητο λογισμικό για του οποίου τη συντήρηση είναι υπεύθυνος ο ίδιος . Η κοστολόγηση γίνεται με τον τρόπο που αναφέραμε παραπάνω. Παραδείγματα τέτοιου είδους πλατφόρμας αποτελούν τα : Amazon EC2, Google Compute Engine, HP Converged Cloud, Windows Azure.

-PaaS: Σε αυτό το μοντέλο υπηρεσίας ο πάροχος προσφέρει υπολογιστική-προγραμματιστική πλατφόρμα η οποία ως επί το πλείστον περιλαμβάνει λειτουργικό σύστημα , περιβάλλον εκτέλεσης γλώσσας προγραμματισμού, βάση δεδομένων και δικτυακό server. Οι πελάτες μπορούν εκεί να αναπτύσσουν και να λειτουργούν λογισμικό. Στις περισσότερες πλατφόρμες PaaS προσφέρεται και η δυνατότητα για αυτόματη κλιμάκωση των υπολογιστικών και αποθηκευτικών πόρων ώστε να

καλύπτονται οι ανάγκες τις εφαρμογής του πελάτη. Παραδείγματα αποτελούν τα : AWS Elastic Beanstalk (Amazon), Google Application Engine, AppScale, Windows Azure Cloud Services.

-SaaS: Σε αυτό το μοντέλο υπηρεσίας οι πελάτες αποκτούν πρόσβαση σε «έτοιμο» λογισμικό εφαρμογών και σε βάσεις δεδομένων που προσφέρονται από τον πάροχο. Ο πάροχος είναι υπεύθυνος για την ορθή λειτουργία των υποκείμενων υποδομών και του περιβάλλοντος εκτέλεσης του λογισμικού. Εδώ η κοστολόγηση γίνεται με τη μορφή εγγραφής για χρήση της υπηρεσίας συνήθως ανά μήνα ή έτος. Παραδείγματα αποτελούν τα: Google Apps, Microsoft Office 365, Hewlett-Packard enterprise software SaaS, Salesforce.

-NaaS: Αυτό το μοντέλο υπηρεσίας το οποίο τυποποιήθηκε πρόσφατα (μόλις το 2012) περιλαμβάνει υπηρεσίες δικτυακής συνδεσιμότητας και μεταφοράς δεδομένων ακόμα και ανάμεσα σε διαφορετικές πλατφόρμες «νεφών». Επίσης είναι επιφορτισμένο με τη βελτιστοποίηση της δέσμευσης πόρων αντιμετωπίζοντας δίκτυο και υπολογιστική υποδομή σαν ένα ενιαίο σύνολο. Τα πιο συνηθισμένα πακέτα υπηρεσιών που προσφέρει αφορούν ελαστικά και εκτεταμένα εικονικά ιδιωτικά δίκτυα (δηλαδή ιδιωτικά δίκτυα που έχουν επεκταθεί ώστε να περιλαμβάνουν κόμβους που ανήκουν στο διαδίκτυο τηρώντας όμως την «ιδιωτικότητα») καθώς και δέσμευση εύρους ζώνης κατ'απαίτηση (συνήθως με αυτοματοποιημένο τρόπο για την κάλυψη των αναγκών του οργανισμού-πελάτη). Παραδείγματα αποτελούν τα: AristaNetworks NaaS, Aryaka NaaS, Tata Communications NaaS.

2.5 Μοντέλα Ανάπτυξης

Τα κυριότερα μοντέλα ανάπτυξης μιας πλατφόρμας υπολογιστικού «νέφους», είναι 4 και διαφέρουν ως προς τη χρήση, τη διαχείριση και την οργάνωση.

Αναλυτικότερα:

-Δημόσιο «νέφος»: Είναι το κυριότερο και χαρακτηριστικότερο μοντέλο ανάπτυξης υπολογιστικού «νέφους». Μεγάλοι οργανισμοί όπως οι Amazon, Google, Microsoft κ.α. υλοποιούν μεγάλης κλίμακας υποδομές οι οποίες είναι δημόσια προσβάσιμες μέσω του διαδικτύου και σκοπεύουν στην εξυπηρέτηση μεγάλου αριθμού χρηστών. Είναι το μοντέλο που κατά κύριο λόγω επιτυγχάνει την ιδέα των οικονομιών κλίμακας. Βέβαια, προκύπτουν ζητήματα ασφάλειας αφενός λόγω της διασύνδεσής του στο διαδίκτυο που το καθιστά στόχο εξωτερικών επιθέσεων και αφετέρου λόγω της συνύπαρξης πολλών χρηστών μέσα στο «νέφος», που το καθιστά στόχο εσωτερικών επιθέσεων μέσω κακόβουλου κώδικα που θα προσπαθήσει να χειραγωγήσει την λειτουργικότητα

της υποδομής για να επιτύχει μη προβλεπόμενη συμπεριφορά (απάτη, υποκλοπή-αλλαγή-καταστροφή δεδομένων κ.α.)

-**Ιδιωτικό** «νέφος»: Στο μοντέλο αυτό ένας οργανισμός υλοποιεί ένα υπολογιστικό «νέφος», το οποίο του ανήκει και στο οποίο κάνει αποκλειστική χρήση, αν και δεν είναι σπάνιο η διαχείριση να έχει δοθεί σε «τρίτο» οργανισμό. Εδώ τα ρήγματα ασφάλειας είναι σαφώς λιγότερα μιας και το «νέφος» δεν είναι δημόσια προσβάσιμο, ο χρήστης είναι αποκλειστικά ο οργανισμός στον οποίο ανήκει και τα ευαίσθητα δεδομένα και λογισμικό του οργανισμού παραμένουν «εσωτερικά» (εν αντιθέσει με την «εξωτερίκευσή» τους σε ένα δημόσιο «νέφος»). Παρόλα αυτά, σε αντίθεση με το δημόσιο «νέφος», δεν επιτυγχάνει οικονομίες κλίμακας και αποτελεί ένα μεγάλου κόστους εγχείρημα.

-**Υβριδικό** «νέφος»: Το μοντέλο αυτό είναι ο συνδυασμός των ανωτέρω 2 μοντέλων. Συνήθως οργανισμοί που χρησιμοποιούν ιδιωτικό «νέφος», υλοποιούν μικρότερης κλίμακας υποδομές που εξυπηρετούν άμεσες ανάγκες και χρησιμοποιούν ένα δημόσιο «νέφος» αγοράζοντας υπολογιστική ισχύ ή/και αποθηκευτικό χώρο κατ'απαίτηση σε περιόδους υψηλής ζήτησης.

-**Κοινοτικό** «νέφος»: Το μοντέλο αυτό με την έννοια της κοινότητας αναφέρεται σε μια ομάδα από οργανισμούς με κοινό στόχο ή ενδιαφέρον, που υλοποιούν, κάνουν αποκλειστική μεταξύ τους χρήση και επωμίζονται το κόστος από κοινού ενός υπολογιστικού «νέφους». Και πάλι η διαχείριση μπορεί να βρίσκεται σε χέρια κάποιου «τρίτου» .

2.6 Παραδείγματα Χρήσης του «νέφους»

Σύμφωνα με πρόσφατη έρευνα της IBM σε συνεργασία με το τμήμα πληροφοριών του Economist το Φεβρουάριο του 2012 και δημοσιεύεται από το Forbes, σε 572 εταιρίες που συμμετείχαν, το 75% από αυτές είτε έχει δοκιμάσει είτε έχει υιοθετήσει ολοκληρωτικά ή μερικώς τεχνολογίες του υπολογιστικού νέφους ενώ το 90% από αυτές αναμένει ότι θα συμμετέχει σε 3 χρόνια. Μέσα στην έρευνα βρέθηκαν χαρακτηριστικά παραδείγματα επιτυχημένης χρήσης των υπηρεσιών του «νέφους».

Αναλυτικότερα:

-**Etsy**: Διαδικτυακή αγορά χειροποίητων προϊόντων, που φέρνει μαζί κατασκευαστές και αγοραστές και κάνει μελετημένες προτάσεις αγορών στους τελευταίους. Η ιδιότητα του «νέφους» να τιμολογεί τους υπολογιστικούς πόρους ευθέως ανάλογα με τη χρήση τους, έδωσε τη δυνατότητα στην εταιρία να επεξεργάζεται στοιχεία από περίπου 1 δισεκατομμύριο τελικούς χρήστες, χρησιμοποιώντας

υπολογιστική ισχύ, που με παραδοσιακές τεχνολογίες, θα αντιστοιχούσε σε εταιρίες πολύ μεγαλύτερου τζίρου.

-Netflix: Εταιρία που προσφέρει video, μέσω streaming, νέων ταινιών και τηλεοπτικών σειρών. Λόγω του μεγάλου αριθμού τελικών χρηστών, ο ιστότοπος έχει ανάγκη από πολύ υψηλή διαθεσιμότητα σε ώρες αιχμής. Όταν η υλική υποδομή της εταιρίας άρχισε να γίνεται ανεπαρκής, αυτή μετέφερε τη λειτουργία της υπηρεσίας της στο «νέφος», οπού μπορεί να εκμεταλλευτεί την ικανότητα του να κλιμακώνει/αποκλιμακώνει τη διάθεση των πόρων προς των ιστότοπο αυτόματα, έναντι της επιλογής να διαθέσει η ίδια το χρόνο και το κεφάλαιο για να δημιουργήσει μεγαλύτερης κλίμακας υλική υποδομή.

-Xerox: Η εγγενής ιδιότητα του «νέφους» να διαθέτει υπηρεσίες αποκρύπτοντας την υποκείμενη πολυπλοκότητα τους στον τελικό χρήστη ήταν ο κύριος λόγος που ώθησε τη Xerox στο να διαθέσει μέσω αυτού την υπηρεσία CloudPrint. Η υπηρεσία αυτή δίνει τη δυνατότητα σε εργαζομένους να εκτυπώνουν το επιθυμητό περιεχόμενο χρησιμοποιώντας το «νέφος» της Xerox από εκτυπωτές εκτός του οργανισμού στον οποίο εργάζονται. Ενώ η εκτύπωση μέσω του «νέφους» απαιτεί αρκετά λεπτομερή διαχείριση δεδομένων, όπως την αποθήκευση τεράστιου αριθμού εγγράφων, τη μετατροπή τους σε εκτυπώσιμη μορφή και το διαμοιρασμό τους σε ένα μεγάλο αριθμό από εκτυπωτές, αυτή η πολυπλοκότητα εξυπηρετείται με μέριμνα της Xerox και αποκρύπτεται τελείως από τον τελικό χρήστη.

-HealthHiway: Η εγγενής ιδιότητα του «νέφους» να ενθαρρύνει την συνδεσιμότητα, επέτρεψε στο HealthHiway να προσφέρει μέσω αυτού ένα δίκτυο πληροφοριών υγείας που επιτρέπει την ανταλλαγή πληροφοριών και συναλλαγές μεταξύ φορέων υγείας και ασφάλισης , εργαζομένων, επαγγελματιών και ασθενών, με πάνω από 1.100 νοσοκομεία και 10.000 γιατρούς να συμμετέχουν σε όλη την Ινδία, δίνοντας τη δυνατότητα για υψηλότερης ποιότητας υπηρεσίες υγείας σε χαμηλότερο κόστος.

2.7 Επιπλοκές

Σε έρευνα, που διεξήγε ο επενδυτικός όμιλος NorthBridgeVentureParteners σε συνεργασία με την εταιρία έρευνας στην αγορά πληροφοριών GigaOMResearch τον Ιούνιο του 2013 με 855 συμμετέχοντες (χρήστες στον τομέα των επιχειρήσεων, υπεύθυνων Πληροφοριακής Υποστήριξης , παρόχων υπηρεσιών «νέφους» κ.α.), ανάμεσα στα υπόλοιπα συμπεράσματα αναγνωρίστηκαν και οι βασικές δυσκολίες στην υιοθέτηση της τεχνολογίας.

Αναλυτικότερα:

-Κατά το 46% των συμμετεχόντων (έναντι 57% αντίστοιχα το 2012) η ασφάλεια αποτελεί κύριο ερωτηματικό πριν την υιοθέτηση της τεχνολογίας κυρίως διότι δεν είναι ακόμα δεδομένο το εύρος των κινδύνων από τη διατήρηση πιθανά κρίσιμων πληροφοριών σε ένα δημόσια προσβάσιμο και πολλές φορές διαχειριζόμενο από «τρίτο», περιβάλλον .

-Το ίδιο σημαντικό στοιχείο (46%) με την ασφάλεια αποτελεί η περεταίρω πολυπλοκότητα, σε ζητήματα διαχείρισης της τεχνολογίας, που εισάγεται στη πληροφοριακή υποστήριξη ενός οργανισμού που χρησιμοποιεί το «νέφος».

-Η εξάρτηση από των πάροχο (Vendorlock-in) (35%) και η διαλειτουργικότητα ανάμεσα στις υπηρεσίες διαφορετικών παρόχων (27%) αποτελούν σημαντικά ζητήματα επίσης, καθώς πολλοί οργανισμοί πρέπει να κάνουν επίπονη έρευνα αγοράς πριν επιλέξουν τους κατάλληλους πάροχους και υπηρεσίες.

-Η αξιοπιστία (22.3%) και η πολυπλοκότητα (21%) ενός περιβάλλοντος αδιάκοπης λειτουργίας προβληματίζουν οργανισμούς με χαμηλότερα επίπεδα τεχνογνωσίας.

-Δευτερεύοντες προβληματισμούς αποτελούν ζητήματα νομικού πλαισίου (30%) και ιδιωτικότητας (26%) αλλά και του κόστους των υπηρεσιών (28%).

ΚΕΦΑΛΑΙΟ 3:Google Application Engine (GAE/AppEngine)

3.1 Εισαγωγή

Το GAE είναι μια εμπορική πλατφόρμα υπολογιστικού «νέφους» τύπου PaaS, υπό τη κυριαρχία της Google, του οποίου η λειτουργία ξεκίνησε το 2008. Προσφέρει ένα ευρύ πακέτο υπηρεσιών για την ανάπτυξη διαδικτυακών εφαρμογών και τη διάθεσή τους μέσω της πλατφόρμας σε τελικούς χρήστες. Η ποιότητα των υποδομών είναι αντίστοιχη με της προϋπάρχουσας κολοσσιαίας μηχανής αναζήτησης της Google. Υπόσχεται απεριόριστη κλιμάκωση των υπολογιστικών και αποθηκευτικών πόρων της εφαρμογής χωρίς τη μέριμνα του πελάτη, άριστη κατάσταση και μέγιστη αποδοτικότητα του υποκείμενου υλικού και υψηλά στάνταρ ασφάλειας, ιδιωτικότητας και προστασίας δεδομένων που υιοθετεί και η ίδια η εταιρία για της εφαρμογές της. Η κίνηση της Google να παραχωρεί μια λεπτή «ζώνη» πόρων, ικανών και αναγκαίων για την εκκίνηση μιας εφαρμογής, χωρίς χρέωση και περεταίρω συνδιαλλαγή μαζί της, κάνει το GAE ιδιαίτερα ελκυστικό για την δοκιμή της υπηρεσίας και την ανάπτυξη μικρής κλίμακας εφαρμογών, όπως η παρούσα εργασία (με ακαδημαϊκό και όχι εμπορικό ενδιαφέρον), δωρεάν.

3.2 Ανάπτυξη Λογισμικού & Υπηρεσίες

Η πλατφόρμα περιλαμβάνει περιβάλλον εκτέλεσης για τις γλώσσες Java, Python, Go και PHP (η τελευταία σε δοκιμαστικό στάδιο). Παρέχει πακέτο ανάπτυξης λογισμικού (SDK) για κάθε μια από αυτές το οποίο μπορεί να ενσωματωθεί σε δημοφιλή ολοκληρωμένα περιβάλλοντα ανάπτυξης λογισμικού (IDE) όπως το Eclipse και το Netbeans . Επίσης, περιλαμβάνει σε αυτό βιβλιοθήκες για μια μεγάλη γκάμα προγραμματιστικών διεπαφών (API) για τον εμπλουτισμό και καλύτερη διαχείριση μιας εφαρμογής. Αναλυτικότερα:

-DataStore API: Προσφέρει πρόσβαση σε μια «δυναμικού σχήματος» (NoSQL/Schemaless) βάση δεδομένων, με μια πλούσια διεπαφή μοντελοποίησης και γλώσσα επερωτήσεων όμοια με την SQL. (**Java, Python, Go**)

-Blobstore API: Επιτρέπει στην εφαρμογή να εξυπηρετήσει μεγάλα αντικείμενα δεδομένων, όπως εικόνες και video, που είναι πολύ μεγάλα για το DataStore (μεγαλύτερα από 1MB). (**Java, Python, Go**)

-Capabilities API: Παρέχει ανίχνευση διακοπών και προγραμματισμένων εργασιών συντήρησης των υπηρεσιών και διεπαφών του GAE, ώστε να είναι δυνατό για την εφαρμογή να τις προσπεράσει ή να ενημερώσει τους τελικούς χρήστες. (**Java, Python, Go**)

-Channel API: Παρέχει δυνατότητα για τη δημιουργία παραμένουσας σύνδεσης μεταξύ της εφαρμογής και των servers της Google για την αποστολή μηνυμάτων σε JavaScript clients σε πραγματικό χρόνο, χωρίς «polling». (**Java, Python, Go**)

-Images API: Παρέχει δυνατότητα για χειρισμό, συνδυασμό και εμπλουτισμό εικόνων, την μετατροπή τους ανάμεσα σε διαφορετικά πρότυπα αναπαράστασης(formats) καθώς επίσης και την αναζήτηση εικόνων με βάση μετα-δεδομένα, όπως ύψος και συχνότητα χρώματος. (**Java, Python, Go**(μερικώς),**PHP**(μερικώς))

-Logs API: Παρέχει πρόσβαση στα αρχεία καταγραφής συμβάντων της εφαρμογής και τον αιτήσεων ως προς αυτήν, μέσα από την εφαρμογή. (**Java, Python, Go,PHP**)

-Mail API: Ενσωμάτωση δυνατότητας αποστολής ηλεκτρονικού ταχυδρομείου εκ μέρους του διαχειριστή της εφαρμογής και τελικών χρηστών με ενεργό Λογαριασμό Χρήστη της Google καθώς και η λήψη από πολλαπλές τοποθεσίες. (**Java, Python, Go,PHP**)

-Memcache API: Διεπαφή για τη χρήση μιας διαμοιρασμένης, εντός-μνήμης RAM(in-memory), ενδιάμεσης αποθήκης δεδομένων που μπορεί να αυξήσει σημαντικά την αποδοτικότητα της εφαρμογής. (**Java, Python, Go,PHP**)

-Multitenancy API: Διεπαφή που διευκολύνει τον κατακερματισμό των δεδομένων ώστε να εξυπηρετούνται εξαρτώμενοι οργανισμοί από ένα μόνο στιγμιότυπο(instance) της εφαρμογής. (**Java, Python, Go**)

-Remote API: Παρέχει τη δυνατότητα για διαφανή πρόσβαση των υπηρεσιών του GAE από εξωτερικές εφαρμογές. Μπορεί π.χ. να χρησιμοποιηθεί για την πρόσβαση στο DataStore από μια εφαρμογή που τρέχει τοπικά σε περιφερειακό Η/Υ. (**Java, Python, Go**)

-TaskQueue API: Επιτρέπει την εκτέλεση εργασιών που αφορούν την εφαρμογή εκτός αυτών που αιτούνται οι τελικοί χρήστες. Οι εργασίες αυτές οργανώνονται σε μικρές διακριτές μονάδες που ονομάζονται «tasks» και εκτελούνται με προγραμματισμένη σειρά. (**Java, Python, Go,PHP**(μερικώς))

-URLFetch API: Παρέχει πρόσβαση στην δικτυακή υποδομή της Google με σκοπό την αποδοτική αποστολή αιτήσεων HTTP και HTTPS από την εφαρμογή σε οποιοδήποτε προσβάσιμο URL. (Java, Python, Go,PHP)

-Users API: Παρέχει τη δυνατότητα στις εφαρμογές να αυθεντικοποιούν χρήστες μέσω Λογαριασμών Χρήστη Google (Google Accounts) και της ενδιάμεσης εφαρμογής αυθεντικοποίησης Open ID καθώς επίσης και να τους απευθύνουν μοναδικά αναγνωριστικά. (Java, Python, Go,PHP)

-XMPP API: Διεπαφή που επιτρέπει στην εφαρμογή να στείλει και να λάβει μηνύματα chat από και προς οποιαδήποτε εξωτερική υπηρεσία που υιοθετεί το πρωτόκολλο XMPP. (Java, Python, Go)

Πέρα των παραπάνω υπάρχουν και άλλες υπηρεσίες σε δοκιμαστικό ή πειραματικό στάδιο, ενδεικτικά αναφέρουμε:

-Search API: Διεπαφή που δίνει τη δυνατότητα στην εφαρμογή να εκτελέσει «Google-like» αναζητήσεις σε «δομημένα» δεδομένα όπως plaintext, HTML, αριθμούς, ημερομηνίες και γεωγραφικές τοποθεσίες. (Java, Python)

-GoogleCloudSQL API: Παρέχει πρόσβαση σε μια πλήρως διαχειριζόμενη υπηρεσία που επιτρέπει τη δημιουργία, τη ρύθμιση και τη χρήση σχεσιακών βάσεων δεδομένων που «κατοικούν» στο «νέφος» της Google. (Java, Python, Go,PHP).

Επίσης, το GAE προσφέρει μια τοπική αναπαράσταση της λειτουργικότητας της πλατφόρμας ανεπτυγμένη σε γλώσσα Python (dev_appserver.py), για την διευκόλυνση της ανάπτυξης και ειδικότερα της αποσφαλμάτωσης της εφαρμογής. Η αναπαράσταση εκτελείται με εντολή κονσόλας η οποία λαμβάνει σαν όρισμα το φάκελο με τα αρχεία κώδικα της εφαρμογής και είναι προσβάσιμη από τον browser (στο localhost:8080).

Ακόμα, σημαντική πτυχή για το στάδιο της ανάπτυξης της εφαρμογής αποτελεί η ύπαρξη του Sandbox. Προκειμένου να τηρηθούν τα στάνταρ απόδοσης και αξιοπιστίας της πλατφόρμας, όλες οι εφαρμογές λειτουργούν σε ένα περιχαρακωμένο περιβάλλον εκτέλεσης που ονομάζεται Sandbox. Το περιβάλλον αυτό αποτρέπει τις εφαρμογές από κάποιες ενέργειες. Αναλυτικότερα:

- απαγορεύεται η εγγραφή στο σύστημα αρχείων. Οι εφαρμογές θα πρέπει να χρησιμοποιούν το DataStore (ή το BlobStore) για την αποθήκευση δεδομένων. Επιτρέπεται η ανάγνωση από το σύστημα αρχείων καθώς επίσης είναι διαθέσιμα όλα τα αρχεία που φορτώθηκαν με την εφαρμογή.

-να δημιουργήσει διεπαφή δικτύου επικοινωνίας(socket) ή να επικοινωνήσει απευθείας με κάποιο εξωτερικό URL. Οι εφαρμογές οφείλουν να χρησιμοποιούν την υπηρεσία URLFetch.

-να αργήσει να αποκριθεί. Η εφαρμογή πρέπει να επιστρέψει απάντηση σε μια αίτηση μέσα σε 60 δευτερόλεπτα αλλιώς τερματίζεται αυτόματα.

-να κάνει οποιαδήποτε άλλη κλήση συστήματος.

3.3 Κλιμάκωση Πόρων-Απόδοση

Οι αιτήσεις των τελικών χρηστών εξυπηρετούνται από «στιγμιότυπα» της εφαρμογής (frontend instances). Με την έννοια «στιγμιότυπο» της εφαρμογής, τουλάχιστον όσον αφορά το GAE, περιγράφουμε μια αυτόνομη οντότητα λογισμικού που εκτελεί των κώδικα της εφαρμογής και κάνει αποκλειστική χρήση μνήμης και περιβάλλοντος εκτέλεσης. Εικονικές μηχανές εξυπηρετούν τα «στιγμιότυπα» της εφαρμογής με την έννοια της διάθεσης των απαραίτητων υπολογιστικών πόρων σε αυτά. Το GAE προσφέρει 4 κλάσεις εικονικών μηχανών, με διαφορετικό μέρισμα χρόνου στη κεντρική μονάδα επεξεργασίας (που αναπαριστάται με ονομαστική συχνότητα επεξεργασίας) και μέγεθος αποκλειστικής μνήμης τυχαίας προσπέλασης(dedicated RAM): **F1(προεπιλεγμένη)-(600MHz/128MB), F2-(1.2GHz/256MB), F4-(2.4GHz/512MB), F4_1G-(2.4GHz/1024MB).** Η ταχύτητα προσπέλασης των εικονικών «σκληρών» μέσων αποθήκευσης είναι ίδια για όλες τις κλάσεις μιας και εξαρτάται από διακριτές προτυποποιημένες υπηρεσίες του GAE όπως οι BlobStoreκαι DataStore.

Οι αιτήσεις προς την εφαρμογή μοιράζονται ανάμεσα σε ενεργά στιγμιότυπα, ενώ ένα «στιγμιότυπο» μπορεί να εξυπηρετεί μία ή παραπάνω αιτήσεις και έχει ανεξάρτητη ουρά αναμονής εισερχόμενων αιτήσεων. Ο αριθμός των «στιγμιότυπων» αυξάνεται και μειώνεται ανάλογα με τον όγκο των αιτήσεων προς την εφαρμογή, όταν οι ουρές των ενεργών στιγμιότυπων είναι μεγάλες και υπάρχει μεγάλη καθυστέρηση στην απόκριση της εφαρμογής προς τον τελικό χρήστη, τότε δημιουργείται ένα νέο στιγμιότυπο και διατηρείται «εν ζωή» μέχρι να μειωθεί ο ρυθμός των αφικνούμενων αιτήσεων προς την εφαρμογή. Υπάρχει και η δυνατότητα διατήρησης αποθήκης ανενεργών στιγμιότυπων, ώστε σε περίπτωση απότομης αύξησης των αιτήσεων να ενεργοποιηθούν άμεσα χωρίς να εισαχθεί περεταίρω καθυστέρηση.

Πέραν των στιγμιότυπων που εξυπηρετούν αφικνούμενες αιτήσεις προς την εφαρμογή, είναι δυνατή η δημιουργία στιγμιότυπων παρασκηνίου(backend instances) τα οποία εξυπηρετούν αιτήσεις που τους αποστέλλει η ίδια η εφαρμογή. Εξυπηρετούν εφαρμογές που έχουν ανάγκη υψηλής ταχύτητας, μεγάλων απαιτήσεων σε μνήμη, συνεχείς ή μακράς διάρκειας διεργασίες παρασκηνίου. Χρησιμοποιούν διαφορετικές κλάσεις εικονικών μηχανών**: B1-(600MHz/128MB),**

B2(προεπιλεγμένη)-(1.2GHz/256MB), B4-(2.4GHz/512MB), B4_1G-(2.4GHz/1024MB), B8-(4.8GHz/1024MB).

Το GAE προσφέρει 2 ειδών μνήμη ενδιάμεσης αποθήκευσης(memcache): κοινής χρήσης και αποκλειστική. Η μνήμη κοινής χρήσης είναι δωρεάν και προεπιλεγμένη για όλες τις εφαρμογές που λειτουργούν στη πλατφόρμα και λειτουργεί με best-effort λογική για να τις εξυπηρετήσει στο σύνολό τους χωρίς περεταίρω εγγυήσεις απόδοσης. Η αποκλειστική μνήμη είναι χρεώσιμη υπηρεσία που μπορεί να προσφέρει ως και 20GB γρήγορης μνήμης με απόδοση έως και 10 χιλιάδων πράξεων το δευτερόλεπτο ανά GB για αντικείμενα μικρότερα του ενός KB.

3.4 «Ζώνη» Δωρεάν Πόρων

Το GAE προσφέρει μια ζώνη δωρεάν πόρων ως προεπιλογή για την υποστήριξη της εφαρμογής του πελάτη. Αυτό κυρίως για να βοηθήσει στην ανάπτυξη , την αποσφαλμάτωση και εν γένει τη δοκιμή μιας νέας εφαρμογής πάνω στη πλατφόρμα. Αν και εξυπηρετεί τις παραπάνω προθέσεις η ζώνη αυτή είναι μικρή για να εξυπηρετήσει μια εφαρμογή που διατίθεται ευρέως προς τελικούς χρήστες, οπότε ο πελάτης θα πρέπει να προχωρήσει σε μια αυτοματοποιημένη οικονομική διευθέτηση με τον πάροχο προκειμένου να έχει ανοιχτή πρόσβαση σε πόρους.

Πιο συγκεκριμένα η ζώνη παρέχει ανά 24 ώρες ανά εφαρμογή (έως και 10 συνολικά):

- 28 ώρες συνολικής λειτουργίας frontend στιγμιότυπων κλάσης F1 (14 ώρες για F2 / 7 ώρες για F4 / 4,5ώρες για F4_1G)

- 9 ώρες συνολικής λειτουργίας backend στιγμιότυπων κλάσης B2 (18 ώρες για B1 / 4,5 ώρες για B4 / 3 ώρες για B4_1G / 2,25 ώρες για B8)

- αποθήκευση 1 GB δεδομένων στο DataStore

- 50K εγγραφές, 50K αναγνώσεις και 50K μικρολειτουργίες στο DataStore

- 0.49 GB όγκος δεδομένων από task στην ουρά

- αποθήκευση 5 GB δεδομένων στο BlobStore

- 1 GB χώρο για αρχεία κώδικα και στατικά αρχεία της εφαρμογής

- 100 παραδόσεις e-mail

- Δημιουργία 100 καναλιών

- 1 GB αποστολή δεδομένων συνολικά από την εφαρμογή προς τους τελικούς χρήστες(outgoing bandwidth)

- 1 GB χώρο για την αποθήκευση αρχείων καταγραφής συμβάντων

3.5 Διαχείριση της Εφαρμογής (Κονσόλα Διαχειριστή)

Το GAE παρέχει σαν κύριο εργαλείο διαχείρισης και ελέγχου της εφαρμογής τη διεπαφή (GUI) Κονσόλα Διαχειριστή (Administration Console). Οι διαχειριστές μπορούν να χρησιμοποιούν τη κονσόλα για να:

-ρυθμίσουν βασικά χαρακτηριστικά, όπως το τίτλο της εφαρμογής, τη διάρκεια των cookies, τις επιλογές αυθεντικοποίησης κ.α.

-ρυθμίσουν τις επιλογές υπολογιστικών επιδόσεων της εφαρμογής (π.χ. κλάση frontend) και να διαχειριστούν το κόστος.

-απενεργοποιήσουν ή να διαγράψουν την εφαρμογή.

-διαχειριστούν το Datastore επιτρέποντας τη δημιουργία αντιγράφων ασφαλείας, την ανάκτηση, την αντιγραφή και τη διαγραφή μιας βάσης δεδομένων ή να απενεργοποιήσουν την υπηρεσία.

-χωρίσουν την «κίνηση» προς την εφαρμογή ανάμεσα σε 2 διαφορετικές εκδόσεις της(traffic split), ρυθμίζοντας το ποσοστό που επιθυμούν να σταλεί και να δοκιμαστεί σε μια νέα έκδοση της εφαρμογής.

-παρακολουθήσουν λεπτομέρειες για τα «στιγμιότυπα» της εφαρμογής, όπως μέγεθος ουράς, καθυστέρηση, κατανάλωση πόρων κ.α.

-εκκινήσουν μια καινούρια εφαρμογή στη πλατφόρμα.

-λάβουν πρόσβαση στα αρχεία καταγραφής ιστορικού των αιτήσεων και των σφαλμάτων.

-διαχειριστούν τις ουρές των «task», επιτρέποντας τη συνένωση, τη διαγραφή και την προσωρινή απενεργοποίηση τους καθώς και να διαχειριστούν τα «task» μεμονωμένα, επιτρέποντας τη παρακολούθηση, τη διαγραφή από την ουρά ή την άμεση εκτέλεση τους.

-παρακολουθήσουν λεπτομέρειες για την απόδοση της μνήμης προσωρινής αποθήκευσης, να την αδειάσουν, να δουν και να αποδώσουν διευθύνσεις.

-Να αναφέρουν άμεσα, με το πλήκτρο αναφοράς παραγωγικού ζητήματος, προβλήματα στην απόδοση της υπηρεσίας π.χ. της ανίχνευσης υψηλών καθυστερήσεων ή απροσδόκητα υψηλού αριθμού σφαλμάτων. (σε πειραματικό στάδιο)

3.6 Προστασία Δεδομένων

Όλη η υποκείμενη υποδομή που χρησιμοποιείται για την αποθήκευση και την επεξεργασία των δεδομένων των πελατών υπόκειται στα απαραίτητα πρότυπα ασφάλειας, όχι χαμηλότερης ποιότητας από αυτά που χρησιμοποιεί η Google και για τα δικά της δεδομένα τέτοιου τύπου. Υιοθετούνται βιομηχανικά πρότυπα συστημάτων και διαδικασιών για να εξασφαλιστεί η προστασία της εφαρμογής από αναμενόμενες επιθέσεις στην ασφάλεια και την ακεραιότητα των δεδομένων της, καθώς επίσης και η προστασία των δεδομένων του πελάτη από μη εξουσιοδοτημένη πρόσβαση.

Τα δεδομένα του πελάτη υπάρχει περίπτωση να αποθηκευτούν είτε στις Η.Π.Α, όπου βρίσκεται η βάση της εταιρίας, είτε σε οποιοδήποτε μέρος του κόσμου στο οποίο η εταιρία, ή κάποιος συνεργάτης της, διατηρεί υποδομές. Όσον αφορά όμως τα δεδομένα των τελικών χρηστών υπάρχει η δυνατότητα επιλογής ανάμεσα στη μόνιμη αποθήκευση τους στις Η.Π.Α ή εντός Ευρωπαϊκής Ένωσης ώστε να είναι δεδομένο το νομικό πλαίσιο που αφορά τη διαχείριση τους. Βέβαια, είναι δυνατό αυτά τα δεδομένα να βρεθούν σε οποιαδήποτε από τις πολλές εγκαταστάσεις που διαθέτει η Google ανά τον κόσμο όσο βρίσκονται σε επεξεργασία και μέχρι την τελική αποθήκευση.

Κεφάλαιο 4: Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ «Go»

4.1 Εισαγωγή

Η Go είναι μία γλώσσα που αναπτύχθηκε από τη Google και πιο συγκεκριμένα από τους Robert Griesemer, Rob Pike και Ken Thompson που ξεκίνησαν το σχεδιασμό της από τον Σεπτέμβριο του 2007. Είναι μια γλώσσα που προσανατολίζεται στη συγγραφή προγραμμάτων για συστήματα παράλληλης επεξεργασίας και δικτυακής λειτουργικότητας. Αν και όχι τόσο διαδεδομένη, όσο οι C(με την οποία συγκρίνεται πολύ συχνά κυρίως λόγω της ομοιότητας που έχουν στην εμφάνιση και τη σύνταξη), Java και Python, έχει αποκτήσει πολλούς υποστηρικτές, π.χ. BBCWorldNews, Open Knowledge Foundation, Moovweb, Second bit κ.α.

4.2 Στόχοι-Αρχές

Η Google αναγνώρισε τα εξής νέα χαρακτηριστικά στο τοπίο της πληροφορικής:

-Οι Η/Υ έχουν γίνει δραματικά ταχύτεροι ενώ η ανάπτυξη λογισμικού όχι.

-Οι διαχείριση της εξάρτησης ανάμεσα στα δεδομένα παίζει σημαντικό ρόλο πια στην ανάπτυξη λογισμικού αλλά οι γλώσσες που ακολουθούν την «παράδοση» της C δεν ενθαρρύνουν την καθαρή ανάλυση των εξαρτήσεων και τη γρήγορη μετάφραση(compiling) των προγραμμάτων.

-Υπάρχει αυξανόμενη δυσαρέσκεια για τα «κουραστικά» συστήματα τύπων όπως στις C και Java, που ωθεί όσους αναπτύσσουν λογισμικό να στρέφονται σε γλώσσες με πιο ευέλικτο σύστημα τύπων όπως οι Python και JavaScript.

-Θεμελιώδης έννοιες όπως της συλλογής απορριμμάτων (στη μνήμη) και της παράλληλης επεξεργασίας δεν υποστηρίζονται επαρκώς από τις δημοφιλείς γλώσσες προγραμματισμού συστημάτων.

-Η εμφάνιση των υπολογιστών πολλών πυρήνων έχει επιφέρει σύγχυση και προβληματισμό.

Έχοντας υπόψη τα παραπάνω η Google δημιούργησε μια νέα γλώσσα με την οποία:

-Είναι δυνατόν να μεταφραστεί ένα πολύ μεγάλο πρόγραμμα μέσα σε λίγα δευτερόλεπτα σε ένα μόνο Η/Υ.

-Προσφέρει ένα μοντέλο κατασκευής λογισμικού που κάνει την ανάλυση εξαρτήσεων εύκολη χωρίς περιττό φόρτο σε κώδικα.

-Το σύστημα τύπων της δεν είναι ιεραρχικό, ώστε να μη σπαταλείται κώδικας σε δηλώσεις σχέσεων μεταξύ τύπων. Επίσης, ενώ η Go περιλαμβάνει στατικούς τύπους, προσπαθεί να κάνει τους τύπους πιο ευέλικτους από των τυπικών αντικειμενοστραφών γλωσσών.

-Κάνει πλήρη συλλογή απορριμμάτων και προσφέρει εγγενώς υποστήριξη για ταυτόχρονη εκτέλεση και επικοινωνία.

-Από το σχεδιασμό της δίνει μια σαφή κατεύθυνση στη κατασκευή λογισμικού για συστήματα μηχανών πολλών πυρήνων.

4.3 Εκδόσεις

Η Go ανακοινώθηκε σαν εργασία ανοιχτού κώδικα(open-source) το Νοέμβριο του 2009. Μετά από περίπου 2 χρόνια ανάπτυξης ανακοινώθηκε η πρώτη σταθερή έκδοση της γλώσσας, Go1, το Μάρτιο του 2012. Η έκδοση περιέχει τις προδιαγραφές της γλώσσας, βιβλιοθήκες και εργαλεία παραμετροποίησης και προσφέρει μια σταθερή βάση για τη δημιουργία αξιόπιστων προϊόντων, εργασιών και δημοσιεύσεων. Εφόσον η σταθερότητα έχει εξασφαλιστεί η Google χρησιμοποιεί τη Go για την ανάπτυξη περεταίρω προϊόντων, προγραμμάτων και εργαλείων παρά κάνει κινήσεις για τροποποίηση της γλώσσας ή των βιβλιοθηκών της. Η έκδοση Go1 αναπτύχθηκε με γνώμονα την μακράς διάρκειας σταθερότητα και ενώ θα συνεχιστεί η έρευνα για βελτίωση στην απόδοση, την αξιοπιστία και τη φορητότητα, δεν αναμένεται να δημιουργηθούν μελλοντικές εκδόσεις ασύμβατες με τη προηγούμενη.

4.4 Σύστημα Τύπων & Δομές Ελέγχου

Ο τύπος καθορίζει ένα σύνολο τιμών και τις επιτρεπτές πράξεις επί αυτών. Η Go υποστηρίζει:

-**Σύνολο μεθόδων**. Ένας τύπος μπορεί να έχει ένα σύνολο μεθόδων (method set) συσχετισμένο με αυτόν. Είναι δηλαδή συναρτήσεις οι οποίες λειτουργούν μόνο πάνω σε τύπο ο οποίος αναφέρεται κατά τη δήλωση τους. Το σύνολο μεθόδων ενός τύπου καθορίζει ποιες διεπαφές υλοποιεί καθώς και την ομάδα από λειτουργίες που είναι διαθέσιμες με τη χρήση του τύπου.

-**Δυαδικό** Τύπο bool

-Αριθμητικούς Τύπους:

-φυσικούς ακέραιους (χωρίς πρόσημο) (uint) ακρίβειας 8/16/32/64 bit

-πραγματικούς ακέραιους (με πρόσημο) (int) ακρίβειας 8/16/32/64 bit

-αριθμούς κινητής υποδιαστολής (IEEE-754) (float) ακρίβειας 32/64 bit

-μιγαδικούς αριθμούς με πραγματικό και φανταστικό μέρος(complex), αριθμό κινητής υποδιαστολής με ακρίβεια 32/64 bit (64/128)

- τον τύπο byte σαν παραλλαγή του τύπου uint8

-τον τύπο rune σαν παραλλαγή του τύπου int32

-Αλφαριθμητικό Τύπο (String). Η σειρά των byte που απαρτίζει ένα αλφαριθμητικό δεν μπορεί να αλλαχθεί κατόπιν της δημιουργίας του. Είναι επίσης συμβατός σαν όρισμα με την εγγενή συνάρτηση len(), που επιστέφει το μήκος του σε αριθμό από bytes.

- Τύπο Πίνακα. Οι διαστάσεις του πίνακα είναι μέρος του τύπου του, δηλαδή τα [2] int32, [7] int32 και [] int32 αποτελούν διαφορετικούς τύπους. Επίσης συμβατός με τη len().

-Τύπο Slice. Πρωτοεμφανιζόμενος τύπος, που λειτουργεί σαν ευρετήριο πάνω σε ένα συναφές κομμάτι ενός πίνακα επιτρέποντας την πρόσβαση σε μια αριθμημένη ακολουθία των στοιχείων του. Μόλις αρχικοποιηθεί ένα slice είναι πάντα συσχετισμένο με τον υποκείμενο πίνακα που περιέχει τα στοιχεία στα οποία αναφέρεται, οπότε το slice μοιράζεται τα στοιχεία του και με τον πίνακα και με άλλα slices που αναφέρονται σε αυτόν, εν αντιθέσει όμως με ένα πίνακα το μήκος του μπορεί να αλλάξει κατά την εκτέλεση.

-Τύπο Struct. Ο τύπος ορίζει μία ομάδα πεδίων (όνομα-τύπος), τα οποία έχουν ονόματα μοναδικά μεταξύ τους. Χρησιμοποιούνται κυρίως για να ομαδοποιήσουν μεταβλητές και συναρτήσεις επί αυτών δημιουργώντας μια μορφή «αντικειμένου».

-Τύπο Δείκτη. Συμβολίζεται με * πάνω σε οποιοδήποτε τύπο.

-Τύπο Συνάρτηση. Ουσιαστικά ο ορισμός συνάρτησης.

-Τύπο Διεπαφή(Interface). Ο τύπος αυτός ορίζει ένα σύνολο μεθόδων και λειτουργεί σαν διεπαφή για αυτό. Μία μεταβλητή ενός τύπου interface μπορεί να λάβει σαν τιμή οποιοδήποτε τύπο του οποίου το σύνολο μεθόδων είναι υπερσύνολο των μεθόδων του interface. Ένας τέτοιος τύπος λέγεται ότι υλοποιεί την διεπαφή.

-Τύπο **Χάρτη(Map)**. Ο χάρτης είναι μία άτακτη συλλογή στοιχείων ενός συγκεκριμένου τύπου ευρετηριασμένων με στοιχεία άλλου τύπου. Δηλαδή ο χάρτης map [string] int32, περιέχει σαν στοιχεία ακέραιους στους οποίους η αναφορά γίνεται με ένα μοναδικό αναγνωριστικό τύπου string.

-Τύπο **Κανάλι**. Ένα κανάλι παρέχει μηχανισμό για το συγχρονισμό της εκτέλεσης και της επικοινωνίας συναρτήσεων που εκτελούνται παράλληλα.

Όσον αφορά τις δομές ελέγχου υποστηρίζονται οι **«if», «for», «switch»** και **«select»** που έχουν τη συνηθισμένη χρήση. Λείπει η «while» η οποία όμως μπορεί εύκολα να αντικατασταθεί με μια «for» με κατάλληλες συνθήκες.

4.5 Block, Δηλώσεις & Εμβέλεια

Το «**block**» κώδικα είναι μία, πιθανόν και κενή, ακολουθία δηλώσεων και εκφράσεων που περικλείεται από αγκύλες. Υπονοούμενα «block» κώδικα είναι:

-**Universal block**, που εμπεριέχει όλο τον κώδικα της Go.

-**Package block**, που περικλείει όλο των κώδικα ενός πακέτου της γλώσσας.

-**File block**, που περικλείει όλο τον κώδικα που βρίσκεται σε ένα αρχείο της γλώσσας.

-Κάθε έκφραση «if», «for» και «switch» θεωρείται ξεχωριστό block.

-Κάθε πεδίο, έκφρασης «switch» και «select», θεωρείται ξεχωριστό block.

Τα block εμφωλιάζουν και επηρεάζουν την εμβέλεια.

Μία **δήλωση** συνδέει ένα αναγνωριστικό με μία σταθερά, τύπο, μεταβλητή, συνάρτηση ή πακέτο. Κάθε αναγνωριστικό του προγράμματος πρέπει να έχει δηλωθεί. Κανένα αναγνωριστικό δε πρέπει να δηλωθεί 2 φορές στο ίδιο block κώδικα καθώς και να έχει δηλωθεί και στο αρχείο και σε πακέτο. Εδώ πρέπει να αναφέρουμε και την ύπαρξη του τελεστή σύντομης δήλωσης μεταβλητής **:=** ,ο οποίος επιτρέπει την άμεση δήλωση και αρχικοποίηση με τύπο της μεταβλητής, στο αριστερό μέλος, αντίστοιχου με την τιμή που παρουσιάζεται στο δεξιό.

Όσον αφορά την **εμβέλεια**, αυτή διευθετείται με την εκμετάλλευση των block. Πιο συγκεκριμένα:

-Η εμβέλεια ενός δεσμευμένου από τη γλώσσα αναγνωριστικού είναι το Universal block.

-Η εμβέλεια ενός αναγνωριστικού που αντιπροσωπεύει σταθερά, τύπο, μεταβλητή ή συνάρτηση (όχι όμως μέθοδο) που έχει δηλωθεί στην αρχή του αρχείου (όχι μέσα σε συνάρτηση) είναι το Package block του πακέτου που έχει ανήκει το αρχείο.

-Η εμβέλεια ενός αναγνωριστικού που αντιπροσωπεύει των ιδιοκτήτη μεθόδου, παράμετρο συνάρτησης ή αποτέλεσμα είναι το σώμα της συνάρτησης.

-Η εμβέλεια ενός αναγνωριστικού σταθεράς, μεταβλητής ή τύπου μέσα σε μια συνάρτηση ξεκινά από τη δήλωσή του ως το τέλος του εσωτερικότερου block που το περιέχει. (συμπεριλαμβανομένων των block των «if», «for» και «switch»).

4.6 Συγχρονισμός

Ο ταυτόχρονος προγραμματισμός είναι δύσκολος σε πολλά περιβάλλοντα λόγω των λεπτομερειών που απαιτούνται για να υλοποιηθεί ορθή πρόσβαση στα κοινόχρηστα δεδομένα. Χρησιμοποιώντας δυο στοιχεία, τα κανάλια και τις **goroutines**,η Go ωθεί σε μια νέα νοοτροπία: τη κοινή χρήση μνήμης μέσω επικοινωνίας, παρά την επικοινωνία μέσω της κοινής χρήσης μνήμης. Οι κοινόχρηστες τιμές περνάνε δια μέσου των καναλιών και στη πραγματικότητα ποτέ δε γίνεται κοινή χρήση του χώρου τους από ξεχωριστά νήματα εκτέλεσης, ενώ πάντα μόνο μια goroutine έχει πρόσβαση σε αυτές ανά πάσα στιγμή.

Πιο αναλυτικά, οι goroutines ακολουθούν ένα απλό μοντέλο, είναι συναρτήσεις που εκτελούντα ταυτόχρονα με άλλες goroutines στον ίδιο χώρο διευθύνσεων. Είναι πολυπλεγμένες με ένα αριθμό από νήματα του λειτουργικού συστήματος έτσι ώστε όταν ένα από αυτά υποχρεούται να σταματήσει προσωρινά την εκτέλεση του, π.χ. ενώ περιμένει για κάποια I/O πράξη, τα υπόλοιπα συνεχίζουν. Προσθέτοντας το πρόθεμα go σε μια κλήση συνάρτησης ή μεθόδου, αυτή τρέχει σαν goroutine και όταν η κλήση της συνάρτησης επιστρέψει αυτή τερματίζει «σιωπηλά». Το σχέδιο τους εν γένει αποκρύπτει πολλές λεπτομέρειες για τη δημιουργία και τη διαχείριση νημάτων.

Επιπλέον, η λειτουργικότητα των **καναλιών** παίζει πολύ σημαντικό ρόλο. Τα κανάλια είναι δυνατόν να δημιουργηθούν είτε μαζί με μία υποκείμενη ενδιάμεση αποθήκη (**buffered channel**) είτε χωρίς (**unbuffered channel**). Τα κανάλια χωρίς αποθήκη συνδυάζουν την επικοινωνία- την ανταλλαγή μιας τιμής- με τον συγχρονισμό, εφόσον μπορούν να εγγυηθούν ότι δύο goroutines θα βρίσκονται σε γνωστή κατάσταση εκτέλεσης. Ο παραλήπτης σταματά προσωρινά την εκτέλεση του μέχρι να υπάρχουν δεδομένα προς παραλαβή και ο αποστολέας μέχρι ο παραλήπτης να λάβει τα δεδομένα. Αν το κανάλι διαθέτει ενδιάμεση αποθήκη τότε ο αποστολέας σταματά προσωρινά την

εκτέλεση του μέχρι η τιμή να εγγραφεί στην αποθήκη και αν αυτή είναι γεμάτη μέχρι να λάβει δεδομένα κάποιος παραλήπτης, ιδιότητα που επιτρέπει στα κανάλια αυτά να λειτουργήσουν και σαν σημαφόροι.

ΚΕΦΑΛΑΙΟ 5: ΗΕΦΑΡΜΟΓΗ “GO linear algebra calculator”

5.1 Εισαγωγή

Το **“GO linear algebra calculator”** (URL: https://**linear-algebra-calculator.appspot.com**) είναι μια μαθηματική εφαρμογή, ανεπτυγμένη και διαθέσιμη μέσω της Paas πλατφόρμας υπολογιστικού «νέφους» Google Application Engine και υλοποιημένη σε γλώσσα Go. Δίνει τη δυνατότητα εφαρμογής δημοφιλών επαναληπτικών αλγορίθμων επίλυσης συστημάτων γραμμικών εξισώσεων, πάνω σε πίνακες που παρέχει σαν είσοδο ο χρήστης. Οι αλγόριθμοι **Jacobi**, **Gauss-Seidel**, **SOR** (successive over-relaxation), **Steepest Gradient Descent** και **Conjugate Gradient** μπορούν να εφαρμοστούν σε τυπικά γραμμικά συστήματα Ax=b (όπου A δισδιάστατος πίνακας, x η λύση και b το διάνυσμα δεξιού μέλους). Ακόμα γίνεται διαθέσιμος βοηθητικά ο αλγόριθμος **Power Iteration** για τον υπολογισμό της φασματικής ακτίνας (spectral radius) πίνακα καθώς και ένας αναδρομικός αλγόριθμος ανεπτυγμένος στα πλαίσια της εφαρμογής για τον υπολογισμό ορίζουσας πίνακα. Η ανάπτυξη της και η διάθεση της στο GAE έχει μηδενικό κόστος αφενός διότι το πακέτο ανάπτυξης λογισμικού (SDK) καθώς και η αναπαράσταση της πλατφόρμας (dev_appserver.py) διατίθενται ελεύθερα από τη Google και αφετέρου λόγω της «ζώνης» δωρεάν πόρων στους οποίους στηρίζεται η λειτουργία της εφαρμογής. Είναι προφανές ότι ο περιορισμός στη χρήση πόρων εντός της «ζώνης» επιφέρει περιορισμούς και στο εύρος των δεδομένων που μπορεί να επεξεργαστεί επιτυχημένα η εφαρμογή. Αυτό εκφράζεται κυρίως με περιορισμούς στις **διαστάσεις του πίνακα εισόδου οι οποίες πρέπει να περιορίζονται σε 1500~1800 (NxN) για αραιούς πίνακες και ~700 για πυκνούς**.

5.2 Οργάνωση Κώδικα

Ο κώδικας της εφαρμογής έχει χτιστεί σε 4 αυτόνομα πακέτα υλοποιημένα σε γλώσσα Go:

-forms package: Στο πακέτο αυτό βρίσκονται οι φόρμες HTML που χρησιμοποιεί η εφαρμογή, αποθηκευμένες σε μορφή σταθεράς (constant).

-primary package: Στο πακέτο αυτό βρίσκονται η συναρτήσεις μαθηματικής επεξεργασίας των δεδομένων, οπού και υλοποιούνται οι προσφερόμενοι αλγόριθμοι.

-secondary package: Στο πακέτο αυτό βρίσκονται οι συναρτήσεις μορφοποίησης που μετατρέπουν τα δεδομένα από την αναπαράσταση που έχουν στην είσοδο σε επεξεργάσιμη μορφή.

-submain package:Το πακέτο αυτό αποτελεί την άτυπη «main» του προγράμματος καθώς παρέχει ουσιαστικά τη διεπαφή του κώδικα επεξεργασίας με τη λειτουργικότητα της πλατφόρμας του GAE και ελέγχει τη ροή του προγράμματος .

5.3 Είσοδος-Εξοδος

Η εφαρμογή βασίζεται σε μια ομάδα από φόρμες HTML για την διεπαφή με τον χρήστη. Στην σελίδα υποδοχής παρατίθενται σε λίστα οι διαθέσιμοι αλγόριθμοι. Εφόσον ο χρήστης επιλέξει τον αλγόριθμο μεταφέρεται σε μια φόρμα οπού δίνονται διάφορες επιλογές για την είσοδο και την επεξεργασία του πίνακα (και του διανύσματος δεξιού μέλους αντίστοιχα).Ο χρήστης έχει την δυνατότητα να επιλέξει μεταξύ 4 «αναπαραστάσεων» του πίνακα:

-**Manual input**: Εισαγωγή του πίνακα με «προτυποποιημένο» τρόπο καθώς και του αριθμού στηλών/γραμμών (τετραγωνικοί πίνακες) σε πεδίο κειμένου (textfield) της φόρμας. Τα στοιχεία καταχωρούνται χωρισμένα με κόμμα ανά σειρά από πάνω προς τα κάτω. Π.χ. ένας 2x2 μοναδιαίος πίνακας: 1,0,0,1

-**.csv files input**: Εισαγωγή του πίνακα μέσω μεταφόρτωσης (uploading) αρχείου .csv (ο πίνακας οφείλει να είναι τυποποιημένος όπως και πάνω)

-**.mmx-sparse/dense files input**: Εισαγωγή του πίνακα μέσω μεταφόρτωσης αρχείου τυποποίησης **MatrixMarketExchange format** (coordinate/array).

Το **MatrixMarket** είναι μια αποθήκη δοκιμαστικών δεδομένων που χρησιμοποιούνται για συγκριτικές μελέτες αλγορίθμων αριθμητικής γραμμικής άλγεβρας υπό την αιγίδα του Εθνικού Ινστιτούτου Τεχνολογίας και Προτύπων των Η.Π.Α (NIST). Περιέχει μια ευρεία συλλογή από πίνακες που έχουν προκύψει από πραγματικά προβλήματα γραμμικής άλγεβρας και παράλληλα δίνει πρόσβαση σε διάφορες εφαρμογές «γεννήτορες» πινάκων. Μέσω αυτού του προγράμματος προέκυψε και το προαναφερθέν πρότυπο σε δυο μορφές: **coordinate**, για την αναπαράσταση αραιών (sparse) πινάκων (τυπικά πίνακες που περιέχουν πάνω από 60% μηδενικά στοιχεία) και **array** για πυκνούς (dense) πίνακες.

Περεταίρω επιλογές στην **είσοδο** της εφαρμογής αποτελούν :

-**Select error margin:**το επιτρεπτό περιθώριο σφάλματος (ή η ελάχιστη απόσταση μεταξύ λύσεων για τον Power Iteration), μπορεί να είναι ο οποιοσδήποτε έγκυρος αριθμός κινητής υποδιαστολής διπλής ακρίβειας ή -1 που υποδεικνύει περιθώριο όσο 0,1% του $||b||_2$

-**i-th column as right hand side**: η εξαγωγή δεξιού μέλους από την ι-στη στήλη του πίνακα

-b=Ax, x=[1...1] as right hand side: για την εκτέλεση των αλγορίθμων, έχοντας υπόψιν αναμενόμενη λύση.

-Show Solutions for each step: η εμφάνιση των ενδιάμεσων προσεγγίσεων λύσης στην έξοδο

Κατά την **έξοδο** η εφαρμογή τυπώνει:

Step: Αριθμός βήματος
Distance from former solution: Ευκλείδεια απόσταση της προσέγγισης που υπολογίστηκε σε αυτό το βήμα από την προηγούμενη.
Residual: το υπόλοιπο $r=||b-Ax^{temp}||_2$

Current Solution: Η προσέγγιση λύσης που υπολογίστηκε στο βήμα (αν έχει επιλεγεί από τον χρήστη)

Τέλος τυπώνει την τελική προσέγγιση λύσης σε μορφή MatrixMarketExchange array format.

5.4 Επεξεργασία Δεδομένων

Η επεξεργασία των δεδομένων γίνεται σε 2 στάδια μέσω των δύο ανεξάρτητων μεταξύ τους πακέτων secondary και primary. Στο πρώτο στάδιο είναι οι συναρτήσεις μορφοποίησης του πακέτου secondary που ανιχνεύουν τις επιλογές του χρήστη και διαβάζουν τον πίνακα και το διάνυσμα δεξιού μέλους (αν δίνεται) από τη φόρμα εισόδου. Αναλόγως της αναπαράστασης του πίνακα ενεργοποιούνται οι κατάλληλες συναρτήσεις που τον μετατρέπουν, είτε από αλφαριθμητικό (Manual input) είτε από αρχείο μορφής .csv/.mmx(sparse/dense), σε δισδιάστατο **slice** πραγματικών αριθμών κινητής υποδιαστολής ακρίβειας 64 bit (float 64) στη μνήμη τυχαίας προσπέλασης της εφαρμογής.

Τα δεδομένα μεταλαμπαδεύονται μέσω του πακέτου submain στις συναρτήσεις μαθηματικής επεξεργασίας του πακέτου primary. Εκεί υλοποιούνται όλοι οι προαναφερθέντες αλγόριθμοι σε

ξεχωριστές συναρτήσεις έκαστος ενώ χρησιμοποιούν μια μικρής έκτασης βιβλιοθήκη συναρτήσεων μικρότερης πολυπλοκότητας, αλλά συχνά απαιτούμενης λειτουργικότητας, που βρίσκεται στο ίδιο πακέτο. Οι συναρτήσεις εκτελούν τους αλγόριθμους πάνω στα δεδομένα τυπώνοντας παράλληλα τα αποτελέσματα κάθε βήματος στην απάντηση (response) της εφαρμογής προς τον χρήστη.

5.5 Περιορισμοί

Οι βασικοί περιορισμοί που διέπουν την εφαρμογή είναι οι εξής:

- ο περιορισμός που τίθεται από το Sandbox για τη διάρκεια ζωής μιας αίτησης στα 60 δευτερόλεπτα. Ο περιορισμός αυτός υποχρεώνει την εφαρμογή να έχει ολοκληρώσει τους υπολογισμούς μέσα σε 60 δευτερόλεπτα και άρα πίνακες που έχουν μεγάλο μέγεθος δε θα έχουν επιτυχή επεξεργασία.

-Η μνήμη της εικονικής μηχανής. Η εφαρμογή χρησιμοποιεί την κλάση F2 η οποία διαθέτει μνήμη 256 MB και εφόσον μια αίτηση δεν μπορεί να εξυπηρετηθεί από παραπάνω από ένα στιγμιότυπο frontend, το εύρος των δεδομένων του πίνακα, συν τον περεταίρω χώρο που απαιτείται για την εκτέλεση του επιλεγμένου αλγορίθμου, δεν μπορεί να ξεπερνά αυτό το μέγεθος.

5.6 Αλγόριθμοι

Jacobi

Η βασική ιδέα ξεκινά με τη διάσπαση του πίνακα **A=D-L-U** (όπου D πίνακας ίσων διαστάσεων με τον A που περιέχει μόνο τα στοιχεία της κύριας διαγωνίου του, L πίνακας ίσων διαστάσεων με τον A που περιέχει μόνο τα στοιχεία που βρίσκονται κάτω από την κύρια διαγώνιο με ανεστραμμένα πρόσημα και U πίνακας ίσων διαστάσεων με τον **A** που περιέχει μόνο τα στοιχεία πάνω από την κύρια διαγώνιο του με ανεστραμμένα πρόσημα) και τη μετατροπή **$Ax=b \Leftrightarrow Dx=(L+U)x+b \Leftrightarrow x=D^{-1}(L+U)x+D^{-1}b$** (συνάρτηση σταθερού σημείου).

Αλγόριθμος:

$x^{(k+1)} = D^{-1}(L+U)\, x^{(k)} + D^{-1}b$, k=0,1,2,...

όπου

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right), \quad i = 1, 2, \ldots, n.$$

Ο αλγόριθμος τυπικά συγκλίνει αν η φασματική ακτίνα (μέγιστη κατ'απόλυτη τιμή ιδιοτιμή) του πίνακα $D^{-1}(L+U)$ είναι μικρότερη του ένα ή με άλλα λόγια εάν ο **A** έχει **αυστηρά κυρίαρχη διαγώνιο** (δηλαδή τα στοιχεία της διαγωνίου έχουν μεγαλύτερη απόλυτη τιμή από το άθροισμα των απόλυτων τιμών των υπόλοιπων στοιχείων της σειράς στη οποία ανήκουν).

Gauss-Seidel

Διάσπαση-Μετατροπή$Ax=b \Leftrightarrow (D-L)x=Ux+b \Leftrightarrow x=(D-L)^{-1}Ux+(D-L)^{-1}b$

Αλγόριθμος:

$x^{(k+1)}=(D-L)^{-1}Ux^{(k)}+(D-L)^{-1}b$

όπου

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} \right), \quad i,j = 1,2,\ldots,n.$$

Ο αλγόριθμος τυπικά συγκλίνει αν η φασματική ακτίνα του $(D-L)^{-1}U$είναι μικρότερη του 1. Με άλλα λόγια αν ο **A** είναι **συμμετρικός** και **θετικά ορισμένος** ή αν έχει **αυστηρά κυρίαρχη διαγώνιο**.

SOR

Αν $x^{(k)}$ γνωστή προσέγγιση, $z^{(k)}$ η προσέγγιση Gauss-Seidel από την $x^{(k)}$ τότε $x^{(k+1)} = \omega z^{(k)} + (1-\omega) x^{(k)}$. Αν το ω=1 τότε η μέθοδος SOR είναι ταυτόσημη με τη Gauss-Seidel.

Αλγόριθμος:

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} \right), \quad i = 1,2,\ldots,n.$$

Ο αλγόριθμος εγγυάται σύγκλιση εάν ο **A** είναι **συμμετρικός, θετικά ορισμένος** και **0<ω<2**

Steepest Gradient Descent

Για τους επαναληπτικούς αλγόριθμους ελαχιστοποίησης όπως οι Steepest Descent και Conjugate Gradient η βασική ιδέα είναι ο ορισμός μιας συνάρτησης **f** της οποίας το **τοπικό ελάχιστο** είναι η **λύση του** αρχικού γραμμικού συστήματος **Ax=b** , αν ο **A** είναι **συμμετρικός** και **θετικά ορισμένος**. Κατόπιν, ορίζουμε συνάρτηση g τ.ω. $g(a)= f(x+au) = f(x) +a(\nabla f(x),u) +O(a^2)$ και

$$g(0)= f(x), \quad g'(0)= (\nabla f(x),u)$$

Hg(a) είναι γνησίως αύξουσα κοντά στο 0 αν $(\nabla f(x),u)>0$ και η ταχύτητα ελάττωσης της f μέγιστη αν $(\nabla f(x),u) = |\nabla f(x)||u|\cos(\theta)$ δηλαδή για $u= -\nabla f(x)$. Οπότε ξεκινώντας από x=0,..,0 και κινούμενοι με κατεύθυνση $-\nabla f(x)$, προσεγγίζουμε το ελάχιστο της f και άρα τη λύση του αρχικού συστήματος.

Αλγόριθμος:

Ορίζουμε $f(x)= {}^1/_2 (Ax,x) –(b,x)$

1. $x^{(0)} = 0$

2. για k = 1, 2,...

3. $r^{(k-1)} =b - Ax^{(k-1)}$

4. αν $r^{(k-1)} =0$ τότε $x = x^{(k-1)}$, ΤΕΛΟΣ

5. αλλιώς,

6. $\alpha_k = (r^{(k-1)} , r^{(k-1)}) / (A r^{(k-1)} , r^{(k-1)})$

7. $x^{(k)} = x^{(k-1)} + \alpha_k r^{(k-1)}$

8. επιστροφή στο βήμα 3.

Conjugate Gradient

Και πάλι ορίζουμε $f(x)= {}^1/_2 (Ax,x) –(b,x)$

Αλγόριθμος:

1. $x^{(0)} = 0$

2. για k = 1, 2,...

3. $r^{(k-1)} = b - A x^{(k-1)} = r^{(k-2)} – \alpha_{k-1} A p^{(k-1)}$

4. αν $r^{(k-1)} = 0$ τότε x = x(k-1) , ΤΕΛΟΣ

5. αλλιώς,

    Για k=1 -> $p^{(1)} = r^{(0)}$

    Για k>1 -> $\beta_k = (r^{(k-1)} , r^{(k-1)}) / (r^{(k-2)} , r^{(k-2)})$

    $p^{(k)} = r^{(k-1)} - \beta_k p^{(k-1)}$

    $\alpha_k = (r^{(k-1)} , r^{(k-1)}) / (A p^{(k)} , p^{(k)})$

    $x^{(k)} = x^{(k-1)} + \alpha_k p^{(k)}$

    επιστροφή στο βήμα 3.

ΚΕΦΑΛΑΙΟ 6: ΔΟΚΙΜΑΣΤΙΚΟΙ ΠΙΝΑΚΕΣ

6.1 Εισαγωγή

Στο κεφάλαιο αυτό δοκιμάζονται 6 πίνακες με διαφορετικές ιδιότητες για επισκόπηση των δυνατοτήτων της εφαρμογής. Εξετάζεται η απόδοση των αλγορίθμων Jacobi, Gauss-Seidel, Conjugate Gradient και Steepest Gradient Descent σε εξισώσεις Ax=b με δεξιό μέλος b=Ax για x=[1…1] και άρα αναμενόμενη λύση. Οι αλγόριθμοι εξετάζονται με μηδενική ανοχή σφάλματος (συμβολίζεται με 0 μετά το όνομα του αλγορίθμου) και με ανοχή σφάλματος 0,1% του αρχικού σφάλματος $r^{(0)}=||b-Ax^{(0)}||_2$ (όπου $x^{(0)}=[0...0]$) $=>r^{(0)}$ = b επιλογή που ενεργοποιεί η εφαρμογή με την τιμή -1 (συμβολίζεται με -1 μετά το όνομα του αλγορίθμου). Οι επαναλήψεις των αλγορίθμων περιορίζονται και είναι όσες το πλήθος των γραμμών του πίνακα εισόδου.

Στους παρακάτω πίνακες παρατίθενται σε στήλες : **α)** ο αριθμός του βήματος στο οποίο είτε βρέθηκε πετυχημένα τελική προσέγγιση λύσης είτε το χαμηλότερο σφάλμα αν ο αλγόριθμος δεν συνέκλινε, **β)** το απόλυτο σφάλμα της προσέγγισης λύσης, **γ)** ο χρόνος που καταναλώθηκε για την εκτέλεση του αλγορίθμου, **δ)** το μέγεθος της μνήμης που χρειάστηκε να δεσμευτεί για την εκτέλεση του αλγορίθμου, **ε)** το χρηματικό κόστος της εκτέλεσης του αλγορίθμου και **στ)** χαρακτηρισμός για την ταχύτητα σύγκλισης του αλγορίθμου πάνω στον πίνακα (η σύγκλιση με περιθώριο σφάλματος χαρακτηρίζεται πάντα ως μερική).

6.2 SHERMAN2: Oil reservoir simulation

Πίνακας του MatrixMarket που προέρχεται από πρόβλημα προσομοίωσης δεξαμενής πετρελαίου. Πρόκειται για έναν**1080x1080**, **αραιό**, **μη συμμετρικό** πίνακα**, χωρίς ισχυρή διαγώνιο**, ο οποίος θεωρητικά δεν πληροί τις προϋποθέσεις για την εφαρμογή οποιουδήποτε από τους αλγορίθμους.

Αλγόριθμος	#Βήματος (Τελικής ή πλησιέστερης προσέγγισης)	Υπόλοιπο	Χρόνος στη CPU (seconds)	Μνήμη	Κόστος $	Σύγκλιση
Jacobi(0)	1	6.145e+14	21.904	112KB	0.0125	Όχι

Jacobi(-1)	1	6.145e+14	23.161	112KB	0.0125	Όχι
GS(0)	1	1.019e+56	22.576	112KB	0.0126	Όχι
GS(-1)	1	1.019e+56	22.142	112KB	0.0126	Όχι
Conjug(0)	36	1.1258e+09	22.792	150KB	0.0168	Όχι
Conjug(-1)	36	1.1258e+09	22.142	150KB	0.0168	Όχι
Steep(0)	1079	5.574e+08	17.484	146KB	0.0164	Όχι
Steep(-1)	1079	5.574e+08	19.434	146KB	0.0164	Όχι

6.3 SHERMAN4: Oil reservoir simulation

Πίνακας του MatrixMarket που προέρχεται από πρόβλημα προσομοίωσης δεξαμενής πετρελαίου. Πρόκειται για έναν **1104x1104, αραιό**, **μη συμμετρικό** πίνακα **με ισχυρή διαγώνιο**.

Αλγόριθμος	#Βήματος (Λύσης ή Βέλτιστης προσέγγισης)	Υπόλοιπο	Χρόνος στη CPU (seconds)	Μνήμη	Κόστος $	Σύγκλιση
Jacobi(0)	1104	0.122	24.808	139KB	0.0156	Αργή
Jacobi(-1)	1104	0.122	22.662	139KB	0.0156	Αργή
GS(0)	1104	0.016	24.548	139KB	0.0156	Αργή
GS(-1)	993	0.024	21.991	124KB	0.0141	Μερική
Conjug(0)	1086	5.326e-15	25.128	150KB	0.0168	Αργή
Conjug(-1)	176	0.023	4.289	44KB	0.0050	Μερική
Steep(0)	1104	0.333	24.071	147KB	0.0165	Πολύ αργή
Steep(-1)	1104	0.333	23.529	147KB	0.0165	Πολύ αργή

6.4 BCSSTK27: BCS Structural Engineering

Πίνακας του MatrixMarket που προέρχεται από πρόβλημα προσομοίωσης εισόδου τουρμπίνας Boeing. Ο πίνακας διαστάσεων**1224x1224** είναι **αραιός**, **συμμετρικός**, **θετικά ορισμένος**, **χωρίς κυρίαρχη διαγώνιο**.

Αλγόριθμος	#Βήματος (Λύσης ή Βέλτιστης προσέγγισης)	Υπόλοιπο	Χρόνος στη CPU (seconds)	Μνήμη	Κόστος $	Σύγκλιση
Jacobi(0)	1	2.316e+06	31.394	138KB	0.0155	Όχι
Jacobi(-1)	1	2.316e+06	30.094	138KB	0.0155	Όχι
GS(0)	1224	37.1	23.812	163KB	0.0182	Αργή
GS(-1)	282	3844.7	7.778	59KB	0.0066	Μερική
Conjug(0)	1223	0.0001	36.161	165KB	0.0185	Λιγότερο αργή
Conjug(-1)	176	3692.9	5.546	48KB	0.0054	Μερική
Steep(0)	1224	8370	29.509	163KB	0.0182	Πολύ αργή
Steep(-1)	1224	8370	30.744	163KB	0.0182	Πολύ αργή

6.5 BCSSTM19: BCS Structural Engineering

Πίνακας του MatrixMarket που προέρχεται από πρόβλημα προσομοίωσης μέρους κρεμαστής γέφυρας. Ο πίνακας διαστάσεων **817x817** είναι **αραιός**, **συμμετρικός**, **θετικά ορισμένος**, **με κυρίαρχη διαγώνιο**.

Αλγόριθμος	#Βήματος (Λύσης ή Βέλτιστης προσέγγισης)	Υπόλοιπο	Χρόνος στη CPU (seconds)	Μνήμη	Κόστος $	Σύγκλιση
Jacobi(0)	1	0	0.042	5KB	0.0007	Ναι
Jacobi(-1)	-	-	-	-	-	Ναι
GS(0)	1	0	0.042	5KB	0.0007	Ναι
GS(-1)	-	-	-	-	-	Ναι
Conjug(0)	801	5.487e-08	10.421	110KB	0.0123	Αργή
Conjug(-1)	26	99663.6	0.368	23KB	0.0026	Μερική
Steep(0)	817	79.714	8.297	109KB	0.0123	Πολύ αργή
Steep(-1)	669	1.075e+06	7.084	93KB	0.0104	Πολύ αργή

6.6 Hilbert 500

Τυχαίος πίνακας Hilbert από τους «γεννήτορες» του MatrixMarket Deli, διαστάσεων **500x500**, **πυκνός**, **συμμετρικός**, **θετικά ορισμένος** με πολύ κακή «κατάσταση» (ill-conditioned).

Αλγόριθμος	#Βήματος (Λύσης ή Βέλτιστης προσέγγισης)	Υπόλοιπο	Χρόνος στη CPU (seconds)	Μνήμη	Κόστος $	Σύγκλιση
Jacobi(0)	1	1.354e+05	2.122	54KB	0.0060	Όχι
Jacobi(-1)	1	1.354e+05	2.296	54KB	0.0060	Όχι
GS(0)	500	88.399e-05	2.122	67KB	0.0075	Αργή
GS(-1)	56	0.036	0.454	18KB	0.0020	Μερική

Conjug(0)	105	4.48e-14	2.491	69KB	0.0077	Λιγότερο αργή
Conjug(-1)	5	0.011	0.238	12KB	0.0014	Μερική
Steep(0)	499	82.527e-05	2.144	68KB	0.0076	Αργή
Steep(-1)	43	0.035	0.346	16KB	0.0018	Μερική

6.7 Wilkinson 500

Τυχαίος πίνακας Wilkinson από τους «γεννήτορες» του MatrixMarket Deli, διαστάσεων **500x500**, **πυκνός**, **μη συμμετρικός**, **χωρίς ισχυρή διαγώνιο**.

Αλγόριθμος	#Βήματος (Λύσης ή Βέλτιστης προσέγγισης)	Υπόλοιπο	Χρόνος στη CPU (seconds)	Μνήμη	Κόστος $	Σύγκλιση
Jacobi(0)	1	300631	2.361	48KB	0.0054	Όχι
Jacobi(-1)	1	300631	2.318	48KB	0.0054	Όχι
GS(0)	1	1.634e+89	2.512	52KB	0.0059	Όχι
GS(-1)	1	1.634e+89	2.448	52KB	0.0059	Όχι
Conjug(0)	1	870.8	2.859	67KB	0.0075	Όχι
Conjug(-1)	1	870.8	2.794	67KB	0.0075	Όχι
Steep(0)	1	870.8	2.642	56KB	0.0063	Όχι
Steep(-1)	1	870.8	2.599	56KB	0.0063	Όχι

Βιβλιογραφία

1. NIST Draft Special Publication 800-144 “Guidelines on Security and Privacy in Public Cloud Computing”

2. developers.google.com/appengine/

3. appengine.google.com

4.golang.org

5. www.forbes.com, “Shining Examples of Cloud Computing in Action”

6. northbridge.com, “2013 Future of Cloud Computing Survey Reveals Business Driving Cloud Adoption in Everything as a Service Era”

7. en.wikipedia.org, Cloud Computing/Jacobi Method/Gauss-Seidel Method/Successive over-relaxation (SOR) Method/ Gradient Descent/ Conjugate Gradient

8.Σημειώσεις του μαθήματος «Επιστημονικός Υπολογισμός»