ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ & ΣΥΣΤΗΜΑΤΩΝ
ΠΛΗΡΟΦΟΡΙΚΗΣ

Τεχνολογίες Υπολογιστικού Νέφους με έμφαση στη δυναμική αξιολόγηση των παρεχόμενων υπηρεσιών με βάση την ανάλυση της απόδοσης των εφαρμογών και της συγκριτικής αξιολόγησης

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Αθανασία Δ. Ευαγγελινού

Αθήνα, Μάρτιος 2017

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ & ΣΥΣΤΗΜΑΤΩΝ
ΠΛΗΡΟΦΟΡΙΚΗΣ

Τεχνολογίες Υπολογιστικού Νέφους με έμφαση στη δυναμική αξιολόγηση των παρεχόμενων υπηρεσιών με βάση την ανάλυση της απόδοσης των εφαρμογών και της συγκριτικής αξιολόγησης

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Αθανασία Δ. Ευαγγελινού

Συμβουλευτική Επιτροπή : Θεοδώρα Βαρβαρίγου

Εμμανουήλ Βαρβαρίγος

Δημήτριος Ασκούνης

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 31η Μαρτίου 2017.

.....	.....	.....
Θ. Βαρβαρίγου	Ε. Βαρβαρίγος	Δ. Ασκούνης
Καθηγήτρια Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π
.....	.....	.....
Β. Λούμος	Α. Δουλάμης	Α. Σταφυλοπάτης
Καθηγητής Ε.Μ.Π	Επίκουρος Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π
	.....	
	Α. Πρέντζα	
	Αναπληρώτρια Καθηγήτρια	
	Πανεπιστημίου Πειραιώς	

.....

Αθανασία Δ. Ευαγγελινού

Διδάκτωρ Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

ΠΡΟΛΟΓΟΣ

Η διδακτορική διατριβή που παρουσιάζεται στις επόμενες σελίδες εκπονήθηκε από το Νοέμβριο του 2012 μέχρι το Μάρτιο του 2017, στο εργαστήριο Τηλεπικοινωνιών του τομέα Επικοινωνιών, Ηλεκτρονικής και Συστημάτων Πληροφορικής, στη Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου.

Κατά την διάρκεια της εκπόνησης αυτής της διατριβής, είχα την ευκαιρία να ασχοληθώ με αρκετά ενδιαφέροντα επιστημονικά θέματα που αφορούν κυρίως στους τομείς της προδιαγραφής, του σχεδιασμού, της υλοποίησης και του ελέγχου υποδομών πλέγματος και να αποκτήσω πολύτιμη εμπειρία και γνώσεις.

Θα ήθελα να ευχαριστήσω από τα βάθη της καρδιάς μου, την καθηγήτρια μου Θεοδώρα Βαρβαρίγου για το ενδιαφέρον που έδειξε, για τις πολύτιμες συμβουλές της και για την ιδιαίτερη στήριξη που μου παρείχε κατά την διάρκεια αυτής της πορείας μου, καθώς επίσης τους καθηγητές της τριμελούς συμβουλευτικής επιτροπής Εμμανουήλ Βαρβαρίγο και Δημήτριο Ασκούνη.

Ιδιαίτερα θερμές είναι οι ευχαριστίες μου και στην αναπληρώτρια καθηγήτρια Ανδριάνα Πρέντζα, η παρουσία της οποίας υπήρξε ηθική και επιστημονική έμπνευση για μένα, από τον πρώτο μόλις καιρό της φοίτησής μου στο προπτυχιακό πρόγραμμα σπουδών της Σχολής ΗΜΜΥ.

Ιδιαίτερη αναφορά θα ήθελα επίσης να κάνω και στον επίκουρο καθηγητή Δημοσθένη Κυριαζή, για την ενθάρρυνσή του και την αξιοθαύμαστη ικανότητά του να εμπνέει αυτοπεποίθηση σε δύσκολες στιγμές και να κάνει τα πράγματα να φαίνονται απλούστερα.

Επίσης, θα ήθελα να ευχαριστήσω όλους τους συναδέλφους μου στην ερευνητική ομάδα με τους οποίους συνεργάστηκα άψογα και επιτυχώς όλα αυτά τα χρόνια.. Ιδιαίτερες ευχαριστίες ωστόσο θα ήθελα να απευθύνω στους στενούς μου συνεργάτες και κυρίως στον Γιώργο Κουσιουρή,

Χριστίνα Σαντζαρίδου, Αλίκη Κοπανέλη, Βρεττό Μουλό, Παύλο Κρανά, Ανδρέα Μενύχτα και Φώτη Αίσωπο με τους οποίους μοιραστήκαμε τις πάρα πολλές ώρες της ερευνητικής εργασίας.

Ακόμη θα ήθελα να ευχαριστήσω τους κοντινούς μου ανθρώπους, τη νονά μου Μαρία Τσακάλη και τους πολύ καλούς μου φίλους Παναγιώτη Νικητόπουλο και Κατερίνα Σκούτα που στάθηκαν δίπλα μου σε αυτό το δύσκολο βήμα. Τους ευχαριστώ θερμά και ελπίζω να μπορώ να τους το ανταποδώσω έστω και στο ελάχιστο.

Τέλος, θα ήθελα να ευχαριστήσω μέσα από την καρδιά μου, τους γονείς μου και την αδερφή μου που πίστεψαν σε εμένα και στήριξαν τις επιλογές μου.

Αθανασία Δ. Ευαγγελινού

Μάρτιος 2017

Πίνακας Περιεχομένων

Σχήματα

Πίνακες

Περίληψη

Οι υπηρεσίες του Νέφους έχουν αναδειχθεί ως ένα καινοτόμο μοντέλο στην Πληροφορία της Τεχνολογίας τα τελευταία χρόνια. Ωστόσο, μετά την εκτεταμένη χρήση τους σοβαρά ζητήματα έχουν προκύψει σε σχέση με τη διακύμανση της επίδοσής τους λόγω της πολλαπλής μίσθωσης (multi-tenancy) και της κατανομής των πόρων. Τα ζητήματα αυτά, καθιστούν πολύ δύσκολη την παροχή κάθε είδους εκτίμησης των επιδόσεων κατά τη διάρκεια του σχεδιασμού ή της ανάπτυξης της εφαρμογής. Σε αυτό το πλαίσιο και λαμβάνοντας υπόψη τις ετερογενείς προσφορές και τα μοντέλα τιμολόγησης που διατίθενται σήμερα στην αγορά του Νέφους, παρουσιάζουμε ένα καινοτόμο μηχανισμό για την εύρεση της κατάλληλης υπηρεσίας που ταιριάζει στις απαιτήσεις της εφαρμογής και παρέχει την καλύτερη ποιότητα υπηρεσιών (QoS) σε συνδυασμό με το κόστος. Βάσει αυτού παρέχεται ένας πλήρης ορισμός της απόδοσης στο Υπολογιστικό Νέφος και αναλύονται σχετικά θέματα, όπως η εφαρμογή της μεθόδου της συγκριτικής αξιολόγησης στους παρόχους του νέφους αλλά και η υπολογιστική ανάλυση των εφαρμογών.

Αυτή η διατριβή εστιάζει στην μελέτη της απόδοσης των υπηρεσιών του Νέφους και επιδιώκει να εισάγει και να αναλύσει καινοτόμους μηχανισμούς για την αποτελεσματική επιλογή ενός παρόχου Νέφους που προσφέρει το καλύτερο περιβάλλον όσον αφορά στην απόδοση και στο κόστος για να φιλοξενηθεί μια αυθαίρετη εφαρμογή.

Από τη μία πλευρά, ο τρόπος με τον οποίο μία εφαρμογή χρησιμοποιεί τους διαθέσιμους υπολογιστικούς πόρους ίσως να μην είναι γνωστός, γεγονός που καθιστά δύσκολη την επιλογή του κατάλληλου παρόχου. Από την άλλη πλευρά, είναι πολύ πιθανό οι πάροχοι του Νέφους να ενδιαφέρονται να γνωρίζουν τους τύπους των εφαρμογών που φιλοξενούνται στα data centers τους, προκειμένου να αποφεύγονται οι ανεπιθύμητες παρεμβολές στην απόδοση εξαιτίας της ταυτόχρονης εκτέλεσης των εικονικών μηχανών σε ένα φυσικό κόμβο. Η

παρούσα διατριβή πραγματεύεται τα θέματα αυτά με την παρουσίαση ενός μηχανισμού και των κατάλληλων μεθόδων που προσδιορίζουν το υπολογιστικό προφίλ μιας αυθαίρετης εφαρμογής, το κατατάσσουν σύμφωνα με ένα περιορισμένο αριθμό γνωστών benchmark εφαρμογών σε μία τυπική κατηγορία εφαρμογών και παρέχουν την καλύτερη αντιστοίχιση του με μία υπηρεσία Νέφους ως προς την επίδοση και το κόστος.

Τα πειραματικά αποτελέσματα ήταν ενθαρρυντικά και ως εκ τούτου η απόδοση του μηχανισμού θεωρείται ότι είναι καλά εδραιωμένη επιτρέποντας την υιοθέτηση της ως υπηρεσία του Νέφους η οποία επιδιώκει την εισαγωγή της γνώσης αναφορικά με τη σύγκριση της απόδοσης και της ποιότητα των παρεχόμενων υπηρεσιών.

Επιπλέον, η διατριβή παρουσιάζει μια συνδυαστική μεθοδολογία, η οποία παρέχει μια ολοκληρωμένη λύση για το σχεδιασμό και τη μεταφορά των εφαρμογών στο Νέφος, αποδεικνύοντας ότι ο καινοτόμος μηχανισμός που αναπτύχθηκε στην παρούσα διατριβή μπορεί να λειτουργήσει συμπληρωματικά με την αναλυτική μοντελοποίηση και τις προσεγγίσεις για την εξερεύνηση του χρόνου σχεδίασης.

Επιπρόσθετα, η διατριβή αυτή αναλύει την αλληλεπίδραση των εφαρμογών στα υπολογιστικά Νέφη. Επιλέγεται το ίδιο σύνολο των Benchmark εφαρμογών που χρησιμοποιήθηκε για την συγκριτική αξιολόγηση των παρόχων του Νέφους και ανιχνεύεται η μείωση της απόδοσής τους λόγω του διαμοιρασμού των φυσικών πόρων. Με βάση τα πειραματικά δεδομένα, ο πάροχος του Νέφους μπορεί να γνωρίζει εκ των προτέρων την επίδοση ενός συγκεκριμένου συνδυασμού εργασιών που ανατίθενται σε ένα φυσικό κόμβο.

Λέξεις κλειδιά: Υπολογιστικό Νέφος, Βελτιστοποίηση Απόδοσης, Υπηρεσίες Νέφους, Συγκριτική Αξιολόγηση, Κατηγοριοποίηση ,Ανάλυση Εφαρμογής

Abstract

Cloud services have emerged as an innovative IT provisioning model in the recent years. However, after their usage severe considerations have emerged with regard to their varying performance due to multitenancy and resource sharing issues. These issues make it very difficult to provide any kind of performance estimation during application design or deployment time. In that frame and considering the heterogeneous technology offer and the pricing models currently available in the cloud market a novel mechanism for finding the deployment that fits the application requirements and provides the best Quality of Service (QoS) and cost trade-offs is presented. Based on this a complete definition of cloud performance is presented and various relative subjects are analysed such as the benchmarking of Cloud Providers and the application computational analysis.

This thesis focuses on the performance of Cloud services and seeks to introduce and analyze innovative mechanisms for the effective selection of a cloud provider that offers the best environment to host an arbitrary application in terms of performance and cost. On the one hand, the way in which a migrated application component uses the available computing resources may not be known and it might be difficult to choose the best Cloud provider for the component. On the other hand, are Cloud providers interested in knowing the application types that run in their data center in order to avoid interference effects of concurrently running VMs on performance. So, the current thesis addresses these issues by presenting the mechanisms and methods which identify the computational profile of an arbitrary application component, classify it according to a known limited number of application categories, each represented by a relevant benchmark and provide the best matching with a cloud service solution in terms of performance and cost.

Consequently, the experimental results are provided that demonstrate and evaluate the performance and effectiveness of the proposed mechanism for two real application scenarios. Given the experimental results, it appears that the mechanism maps the application profile to a predefined benchmark category and as a result defines the concrete cloud service taking into account the user preferences in terms of performance and cost. The experiments showed promising results and therefore the performance of the mechanism is considered to be well established allowing the adoption of it as a cloud service that seeks to bring QoS knowledge.

In addition, the thesis presents a combined methodology which provides an integrated solution for the design and the migration of enterprise applications to Cloud, proving that our innovative mechanism can work complementary with analytical modeling and design space exploration approaches.

Furthermore, the thesis analyzes the performance interference between concurrently running virtual resources and applications in Clouds environments. The same set of application benchmarks as used for Cloud Providers' benchmarking is chosen that depicts characteristic usage of the hardware resources and the degradation of their performance is measured due to interference effects of concurrently running VM in the same physical host. The outcomes of this experimental process can be used by Cloud Providers as a priori knowledge of the overhead inserted by the execution of a specific task combination on a physical host.

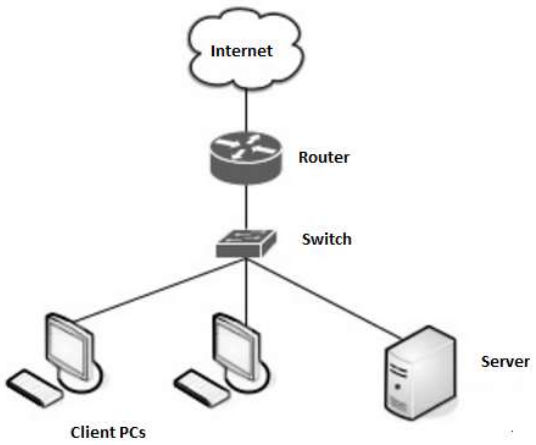
Key words: Cloud Computing, Performance Optimization, Cloud Services, Benchmarking, Classification, Application Analysis

1 *Εισαγωγή*

Το τρέχον κεφάλαιο στοχεύει στην εισαγωγή του αναγνώστη στην έννοια του Υπολογιστικού Νέφους (Cloud Computing). Αρχικά λοιπόν, παρατίθεται μια ενότητα που παρουσιάζει τους διάφορους ορισμούς που υπάρχουν στη βιβλιογραφία, εστιάζοντας και αναλύοντας τον επικρατέστερο σύμφωνα με το Εθνικό Ινστιτούτο Επιστήμης και Τεχνολογίας των Ηνωμένων Πολιτειών/US National Institute of Science and Technology (NIST). Στη συνέχεια αναλύεται η οργάνωση της παρούσης Διατριβής και περιγράφεται συνοπτικά το περιεχόμενο των κεφαλαίων.

1.1 Ορισμοί

Το Υπολογιστικό Νέφος (Cloud Computing) πήρε το όνομά του από μία παρομοίωση του διαδικτύου. Γενικά το διαδίκτυο αντιπροσωπεύεται στα διαγράμματα δικτύων ως ένα Νέφος (Cloud) όπως φαίνεται στο Σχήμα 1.



Σχήμα 1: ***Απεικόνιση του διαδικτύου με τη χρήση ενός Νέφους σε διαγράμματα δικτύων***

Αυτή η σελίδα είναι σκόπιμα λευκή

Με αυτόν τον τρόπο συνήθως προσπαθούμε να περιγράψουμε απομακρυσμένο σύνολο αξιόπιστων υπηρεσιών στον οποίο και στηριζόμαστε, χωρίς όμως να μας ενδιαφέρει το πώς λειτουργεί εσωτερικά. Το εικονίδιο του Νέφους αντιπροσωπεύει "όλα τα άλλα πράγματα" που χρειάζονται ώστε το δίκτυο να δουλεύει [1].

Η ιδέα του Υπολογιστικού Νέφους δεν είναι καινούρια. Παλιότερα ήταν γνωστή ως συσκεύαση υπολογιστικών και αποθηκευτικών πόρων (Utility Computing), Grid Computing (Υπολογιστικό Πλέγμα) κλπ. Ωστόσο, αυτό που θεωρείται καινούριο είναι η ανάπτυξη και η ωρίμανση των μεθόδων του Υπολογιστικού Νέφους καθώς και οι στρατηγικές που ενεργοποιούν τους στόχους της επιχειρηματικής ευελιξίας. Τα τελευταία χρόνια οι επαγγελματίες στο χώρο της τεχνολογίας στην πληροφορία, οι διευθυντές των επιχειρήσεων αλλά και οι ερευνητές ορίζουν το Υπολογιστικό Νέφος με διαφορετικό τρόπο ανάλογα με το τί προσφέρει σε κάθε έναν από τους παραπάνω τομείς.

Το Υπολογιστικό Νέφος είναι η διαθεσιμότητα υπολογιστικών πόρων και υπηρεσιών της πληροφορικής, κατόπιν απαίτησης, και βασίζεται στη διανομή διαμοιραζόμενων πόρων ώστε να επιτύχει συνέπεια και κλιμάκωση σαν μια δημόσια υπηρεσία. Η διάθεση των παραπάνω γίνεται με τη βοήθεια του Διαδικτύου με κόστος που αναλογεί μόνο για τη διάρκεια της χρήσης τους, παρέχοντας υψηλή ευελιξία και παραμετροποίηση, ελάχιστη προσπάθεια από την πλευρά του χρήστη και υψηλή αυτοματοποίηση στοχεύοντας παράλληλα στη βέλτιστη χρήση των διαθέσιμων υπολογιστικών πόρων και ενεργειών. Ο χρήστης είναι ελεύθερος να κάνει αβίαστη χρήση των υπηρεσιών, όταν και όποτε θέλει, χωρίς κανένα περιορισμό.

Επίσης, παρέχει τρεις βασικούς τύπους μοντέλων υπηρεσιών που έχουν αναδειχθεί ως ένα καινοτόμο μοντέλο στην τεχνολογία της πληροφορίας τα τελευταία χρόνια και τα οποία θα αναλυθούν εκτενώς στο Κεφάλαιο 2: Λογισμικό ως Υπηρεσία (Software-as-a-Service/SaaS), Πλατφόρμα ως Υπηρεσία (Platform as a Service/PaaS) και Υποδομές ως υπηρεσία

(Infrastructure as a Service/IaaS). Με αυτή την έννοια το Υπολογιστικό Νέφος έχει τη δυνατότητα να αλλάξει ριζικά τον τρόπο λειτουργίας των υπολογιστικών πόρων καθώς και την ανάπτυξη των εφαρμογών αφήνοντας χώρο για νέα επιχειρηματικά μοντέλα. Ουσιαστικά αποτελεί μια σύγκλιση δύο μεγάλων τάσεων στην πληροφορία της τεχνολογίας [2]. Το πρώτο είναι η αποτελεσματικότητα της τεχνολογίας της πληροφορίας που σχετίζεται με τη δύναμη των σύγχρονων υπολογιστών που χρησιμοποιείται πιο αποτελεσματικά μέσα από κλιμακούμενος υλικούς και λογισμικούς πόρους. Η άλλη τάση είναι η επιχειρηματική ευελιξία, σύμφωνα με την οποία η πληροφορία της τεχνολογίας μπορεί να χρησιμοποιηθεί ως ανταγωνιστικό εργαλείο μέσα από την ταχεία ανάπτυξη, την παράλληλη επεξεργασία, τη χρήση της ανάλυσης των επιχειρήσεων και τις κινητές διαδραστικές εφαρμογές που ανταποκρίνονται σε πραγματικό χρόνο στις απαιτήσεις του χρήστη. Στην συνέχεια παρατίθεται ένα σύνολο με ακριβείς ορισμούς του Υπολογιστικού Νέφους όπως έχουν οριστεί από επιστημονικές αναφορές στην βιβλιογραφία.

Το Υπολογιστικό Νέφος [3] είναι ένα σύνολο υπηρεσιών που διατίθενται μέσω δικτύου οι οποίες παρέχουν επεκτάσιμη και εγγυημένη ποιότητα της υπηρεσίας (QoS), συνήθως εξατομικευμένες, με ανέξοδες υπολογιστικές υποδομές αν και όταν ζητηθούν και στις οποίες οι χρήστες έχουν πρόσβαση με έναν απλό τρόπο.

Σύμφωνα με το [4] το Υπολογιστικό Νέφος είναι μια τεχνολογία που χρησιμοποιεί το διαδίκτυο και κεντρικούς απομακρυσμένους εξυπηρετητές ώστε να φιλοξενεί τα δεδομένα και τις εφαρμογές. Επιτρέπει στους καταναλωτές και στις επιχειρήσεις να χρησιμοποιούν εφαρμογές χωρίς την εγκατάσταση και να έχουν πρόσβαση στα προσωπικά τους αρχεία σε οποιονδήποτε υπολογιστή μέσω του διαδικτύου, συγκεντρώνοντας την αποθήκευση, τη μνήμη, την επεξεργασία και το εύρος ζώνης.

Σύμφωνα με την αναφορά [5] ο επίσημος ορισμός του Υπολογιστικού Νέφους έχει ως εξής: «Είναι ένα υπηρεσιακό μοντέλο παροχή υπηρεσιών της πληροφορίας της τεχνολογίας όπου οι υπολογιστικές υπηρεσίες (τόσο σε υλικό όσο και σε και λογισμικό επίπεδο) παραδίδονται κατ' απαίτηση σε πελάτες μέσω ενός δικτύου σε ένα «self-service» μοντέλο, ανεξάρτητα από τη συσκευή και την τοποθεσία. Οι πόροι που απαιτούνται για να παρέχουν την απαιτούμενη ποιότητα των παρεχόμενων υπηρεσιών, μοιράζονται, είναι δυναμικά κλιμακούμενοι, παρέχονται άμεσα, είναι εικονικοί και διανέμονται με τη λιγότερο δυνατή αλληλεπίδραση με το χρήστη. Οι χρήστες πληρώνουν για την υπηρεσία τα λειτουργικά έξοδα χωρίς να υποστούν οποιοδήποτε αρχικό κεφάλαιο των δαπανών, με τις υπηρεσίες του Νέφους να χρησιμοποιούν ένα σύστημα μέτρησης που χωρίζει τους υπολογιστικούς πόρους σε κατάλληλα μπλοκ.

Τα υπολογιστικά Νέφη [6] θεωρούνται ως ένα σύνολο χρήσιμων και προσιτών εικονικών πόρων που παρέχουν υπολογιστικό εξοπλισμό, πλατφόρμες και υπηρεσίες. Αυτοί οι πόροι αναδιαμορφώνονται δυναμικά ώστε να προσαρμοστούν σε ένα μεταβλητό φορτίο που επιτρέπει τη βέλτιστη αξιοποίηση τους. Οι πόροι αυτοί χρησιμοποιούνται από ένα κατ' απαίτηση μοντέλο σύμφωνα με το οποίο παρέχονται εγγυήσεις στους χρήστες-πελάτες από τους παρόχους των υποδομών μέσω εξατομικευμένων Συμβολαίων Διασφάλισης Επιπέδου Ποιότητας (SLAs).

Ο κύριος λόγος των διαφορετικών αντιλήψεων του Υπολογιστικού Νέφους είναι το γεγονός ότι στην πραγματικότητα δεν είναι μια νέα τεχνολογία, αλλά ένα νέο μοντέλο λειτουργιών που εννοποιεί ένα σύνολο από υπάρχουσες τεχνολογίες που συμβάλλουν στο να «τρέξει» η επιχείρηση με διαφορετικό τρόπο. Το Υπολογιστικό Νέφος αξιοποιεί το σύνολο των τεχνολογιών που ήδη, υπάρχουν ώστε να ανταποκριθεί σε τεχνολογικές και οικονομικές απαιτήσεις που έχουν στις μέρες μας ζήτηση στον τομέα της πληροφορίας της Τεχνολογίας [7].

Σύμφωνα με το US NIST (United States – National Institute of Standards and Technology) το Υπολογιστικό Νέφος είναι ένα μοντέλο που επιτρέπει την εύκολη και κατ' απαίτηση πρόσβαση μέσω δικτύου σε ένα κοινό σύνολο από παραμετροποιήσιμους υπολογιστικούς πόρους (π.χ. Δίκτυα, servers, αποθηκευτικό χώρο, εφαρμογές και υπηρεσίες) οι οποίοι μπορούν πολύ εύκολα να παρακολουθηθούν και να αποδοθούν με πολύ μικρή παρέμβαση της διαχείρισης, ή αλληλεπίδρασης από τον πάροχο των υπηρεσιών. Αυτό το μοντέλο προάγει τη διαθεσιμότητα και απαρτίζεται από πέντε βασικά χαρακτηριστικά, τρία μοντέλα παροχής-παράδοσης της υπηρεσίας και τέσσερα μοντέλα υλοποίησης του [8]. Ο ορισμός αυτός προορίζεται ώστε να χρησιμεύσει ως μέσο για την ευρεία σύγκριση των υπηρεσιών του Νέφους και των στρατηγικών ανάπτυξης και να παράξει μία βάση για συζήτηση από το τί είναι το Υπολογιστικό Νέφος εως το πώς θα μπορούσε να γίνει καλύτερη η χρήση του.

1.2 Καινοτομία - συνεισφορά

Όπως προαναφέρθηκε η παρούσα διατριβή εστιάζει στην μελέτη της απόδοσης των υπηρεσιών του Νέφους και στην κατάταξή τους βάσει της ανάλυσης της απόδοσης της εφαρμογής αλλά και εφαρμόζοντας τη μέθοδο της συγκριτικής αξιολόγησης στους παρόχους του Νέφους.

Η βασική συνεισφορά της παρούσας διατριβής επικεντρώνεται στη μελέτη των Υπηρεσιών του Νέφους στοχεύοντας στην αξιολόγησή τους βάσει της απόδοσης και του κόστους, παρέχοντας στον ιδιοκτήτη μιας εφαρμογής τη βέλτιστη λύση κατά τη μεταφορά της από τα συστήματα της εταιρείας στο Νέφος. Στην επίτευξη αυτού του στόχου εστιάζεται ένα σύνολο από χαρακτηριστικά της επίδοσης του παρόχου που είναι απαραίτητα για την ποιότητα της υπηρεσίας (QoS).

Επιπλέον, σε ένα τέτοιο πλαίσιο σχεδιάζεται και υλοποιείται μια καινοτόμα μεθοδολογία ταξινόμησης των Υπηρεσιών του Νέφους που μπορεί να χρησιμοποιηθεί σε αυτά τα

περιβάλλοντα, η οποία βασίζεται στις τεχνολογίες της συγκριτικής αξιολόγησης και στην ανάλυση της απόδοσης των εφαρμογών. Κίνητρο για τη μεθοδολογία αυτή είναι το γεγονός ότι υπάρχουν πολλές περιπτώσεις όπου ο ιδιοκτήτης μιας εφαρμογής δεν γνωρίζει πώς τα δομικά στοιχεία της εφαρμογής συμπεριφέρονται κατά τη διάρκεια εκτέλεσης της, είτε δεν είναι γνωστός ο τρόπος χρήσης των υπολογιστικών πόρων. Από την άλλη πλευρά, οι πάροχοι του Υπολογιστικού Νέφους, γνωρίζοντας τους τύπους των εφαρμογών που φιλοξενούνται στις υποδομές τους, μέσω της προτεινόμενης μεθοδολογίας μπορούν να αποφύγουν τις ανεπιθύμητες παρεμβολές, που μπορεί να οφείλονται στην ταυτόχρονη εκτέλεση των εικονικών μηχανών (VMs), που υποβαθμίζουν σημαντικά την απόδοση των εφαρμογών. Σύμφωνα με τα παραπάνω ο υλοποιημένος μηχανισμός είναι σε θέση να εντοπίσει την υπολογιστική φύση ενός λογισμικού δομικού στοιχείου μιας εφαρμογής αλλά και να την ταξινομήσει, με τη χρήση ενός αλγορίθμου κατηγοριοποίησης, σε μια λίστα γνωστών υπολογιστικών προτύπων. Τα πρότυπα αυτά ορίζονται μέσα από τα εντοπισμένα στερεότυπα της απόδοσης τα οποία έχουν σα σκοπό να απλοποιήσουν τη διαδικασία της αξιολόγησης της επίδοσης των Υπολογιστικών Νεφών ώστε να υποστηρίξουν το χρήστη στη διαδικασία λήψης αποφάσεων. Η παραγόμενη μεθοδολογία εξετάζεται πειραματικά με χρήση πραγματικών εφαρμογών και επικυρώνεται η ακρίβεια της μεθόδου.

Τέλος η διατριβή επικεντρώνεται στο επίπεδο της κοινής χρήσης πόρων από διαφορετικές υπηρεσίες και διερευνά τις αλληλεπιδράσεις απόδοσης μεταξύ αυτών λόγω της ταυτόχρονης εκτέλεσης στη hardware υποδομή. Διαφορετικοί συνδυασμοί μελετώνται με βάση προκαθορισμένους τύπους εφαρμογών και με βάση τα χαρακτηριστικά χρήσης των υποκείμενων πόρων. Από την μελέτη αυτή προκύπτει ότι ο κατάλληλος συνδυασμός των εφαρμογών για εκτέλεση στον ίδιο φυσικό πόρο μπορεί να ελαχιστοποιήσει ή ακόμα και να εκμηδενίσει τα φαινόμενα υποβάθμισης της υπολογιστικής ικανότητας των εικονικών πόρων

λόγω της ταυτόχρονης χρήσης του hardware.

1.3 Οργάνωση της Διατριβής

Το παρόν έγγραφο αποτελείται από οκτώ (8) κεφάλαια. Στις ενότητες των κεφαλαίων αυτών παρουσιάζεται ουσιαστικά και με αναλυτικό τρόπο το αντικείμενο της διδακτορικής διατριβής.

Το δεύτερο κεφάλαιο αποτελεί μία εισαγωγή στην έννοια του Υπολογιστικού Νέφους (Cloud) καθώς και στα διαφορετικά μοντέλα υπηρεσίας και ανάπτυξης μέσα σε αυτό. Επιπλέον, περιλαμβάνει τις επικρατέστερες ταξινομήσεις των διαθέσιμων αυτών υπηρεσιών που διαδραματίζουν καθοριστικό ρόλο στην απόδοση των εφαρμογών στο Νέφος αλλά και μία σύντομη παρουσίαση των δημοφιλέστερων παρόχων.

Στο τρίτο κεφάλαιο παρουσιάζεται ένα από τα κύρια χαρακτηριστικά του Υπολογιστικού Νέφους: (α) η απόδοση (που έχει κερδίσει την προσοχή τα τελευταία χρόνια, ειδικά τώρα που σχετίζεται με πιο εκτεταμένες έννοιες όπως είναι η διαθεσιμότητα πόρων), (β) η επάρκεια και (γ) η αξιοπιστία. Επίσης γίνεται αναφορά στις γενικές μετρήσεις της απόδοσης καθώς και στους παράγοντες που την επηρεάζουν.

Στο τέταρτο κεφάλαιο παρουσιάζεται αναλυτικά η μέθοδος της συγκριτικής αξιολόγησης στο Υπολογιστικό Νέφος ενώ γίνεται αναφορά στα σημαντικότερα πλαίσια, εργαλεία και εφαρμογές τα οποία έχουν προταθεί για τη μέτρηση και την κατάταξη των υπηρεσιών του Υπολογιστικού Νέφους, εστιάζοντας στη συγκριτική αξιολόγηση σε επίπεδο εφαρμογών. Επιπλέον παρουσιάζονται οι βασικές αρχές της συγκριτικής αξιολόγησης που υιοθετήθηκαν στην παρούσα διατριβή.

Στο πέμπτο κεφάλαιο γίνεται αναφορά στην εξαγωγή των χαρακτηριστικών της απόδοσης και υπολογίζεται η απόδοση των υπηρεσιών του Νέφους με την μέθοδο της συγκριτικής

αξιολόγησης. Επιπλέον, παρουσιάζεται η βασική μεθοδολογία και ο μηχανισμός που αναπτύχθηκε στην παρούσα διατριβή για τη αξιολόγηση των παρόχων του Υπολογιστικού Νέφους βάσει του προφίλ των εφαρμογών εφαρμόζοντας τον αλγόριθμο αλλά και ένα δείκτη αποδοτικότητας που συνδυάζει την απόδοση με το κόστος.

Στο έκτο κεφάλαιο γίνεται αξιολόγηση της λειτουργίας και της απόδοσης του μηχανισμού Δυναμικής Αξιολόγησης των Υπηρεσιών του Νέφους βάσει της ανάλυσης της απόδοσης της εφαρμογής καθώς και της αποτελεσματικότητάς του με χρήση δύο πραγματικών εφαρμογών. Επίσης, παρουσιάζεται η συνδιαστική εφαρμογή του μηχανισμού που παρουσιάζεται στη διατριβή αυτή με ένα εργαλείο για την εξερεύνηση του χώρου σχεδίασης με βέλτιστο τρόπο, το οποίο μπορεί να εντοπίσει αποτελεσματικά τη λύση με το ελάχιστο κόστος, λαμβάνοντας υπ' όψη τις αλλαγές του φορτίου εργασίας παρέχοντας QoS εγγυήσεις.

Στο έβδομο κεφάλαιο παρουσιάζονται τα πειράματα για τη μελέτη της αλληλεπίδρασης των ταυτόχρονα εκτελούμενων εικονικών μηχανών στον ίδιο φυσικό πόρο και η δυνατότητα για την εκ των προτέρων πρόβλεψη αυτής της αλληλεπίδρασης.

Τέλος, στο Κεφάλαιο 8 περιλαμβάνεται η σύνοψη της διατριβής, τα συμπεράσματα που εξήχθησαν κατά την εκπόνησή της, η συνεισφορά και η καινοτομία που επιδεικνύει στον αντίστοιχο ερευνητικό χώρο, ενώ συζητούνται θέματα μελλοντικής εργασίας και επέκτασης των ερευνητικών αποτελεσμάτων.

Αυτή η σελίδα είναι σκόπιμα λευκή

2 Γενικά για το Υπολογιστικό Νέφος

Στο κεφάλαιο αυτό περιγράφονται αναλυτικά τα χαρακτηριστικά του Υπολογιστικού Νέφους, οι υπηρεσίες, καθώς και τα μοντέλα ανάπτυξης που προσφέρονται στους πελάτες. Τέλος αναφέρονται διάφορες ταξινομήσεις των διαθέσιμων Υπηρεσιών του Υπολογιστικού Νέφους, ώστε να υπάρχει μία κοινή ορολογία, καθώς γίνεται και αναφορά στους σημαντικότερους παρόχους και στις υπηρεσίες τους.

2.1 Χαρακτηριστικά

Σε μια προσπάθεια να περιγραφεί ακόμα πιο συγκεκριμένα το Υπολογιστικό Νέφος, παρουσιάζονται σε αυτή την ενότητα τα χαρακτηριστικά του.

2.1.1 Βασικά Λειτουργικά Χαρακτηριστικά κατά NIST

• **Αυτοεξυπηρέτηση κατ' απαίτηση (On-demand self-service).** Ο χρήστης μπορεί να ζητήσει οποιαδήποτε στιγμή μια υπηρεσία ή να δεσμεύσει υπολογιστικούς πόρους μέσω δικτύου αυτόματα χωρίς να απαιτείται καμία ανθρώπινη αλληλεπίδραση με τον πάροχο της εκάστοτε υπηρεσίας.

• **Ευρεία πρόσβαση στο δίκτυο – άμεση ανταπόκριση**. Οι δυνατότητες αυτές είναι προσπελάσιμες από παντού μέσω δικτύου και η προσπέλαση πραγματοποιείται μέσω τυποποιημένων μηχανισμών από συσκευές-πελάτες (π.χ. Κινητά τηλέφωνα, ταμπλέτες, φορητούς υπολογιστές).

• **Αφθονία, διαθεσιμότητα πόρων.** Οι υπολογιστικοί πόροι του παρόχου αποτελούν ένα κοινό σύνολο χρησιμοποιώντας ένα πολύ-πελατειακό μοντέλο, με διαφορετικούς φυσικούς και εικονικούς πόρους οι οποίοι αποδίδονται πολλές φορές δυναμικά και αναδιανέμονται ανάλογα με τη ζήτηση των πελατών. Ο χρήστης γενικά δεν έχει κανένα έλεγχο και γνώση για την ακριβή τοποθέτηση του παρεχόμενου πόρου, αλλά δύναται να προσδιορίσει σε ένα πιο αφηρημένο επίπεδο την τοποθεσία (π.χ. τη χώρα ή την πόλη ή το συγκεκριμένο data-center). Παραδείγματα τέτοιων πόρων μπορεί να είναι η αποθήκευση, η επεξεργασία, η μνήμη και το εύρος ζώνης του δικτύου.

• **Ταχεία ελαστικότητα (Rapid Elasticity).** Οι υπολογιστικοί πόροι μπορούν να δεσμεύονται και να αποδεσμεύονται ελαστικά και σε ορισμένες περιπτώσεις, αυτόματα, ώστε το υπολογιστικό νέφος να κλιμακώνεται ανάλογα με τη ζήτηση. Ο χρήστης έχει την αίσθηση ότι οι υπολογιστικοί πόροι είναι απεριόριστοι και μπορούν να διατεθούν σε οποιαδήποτε ποσότητα κι ανά πάσα στιγμή.

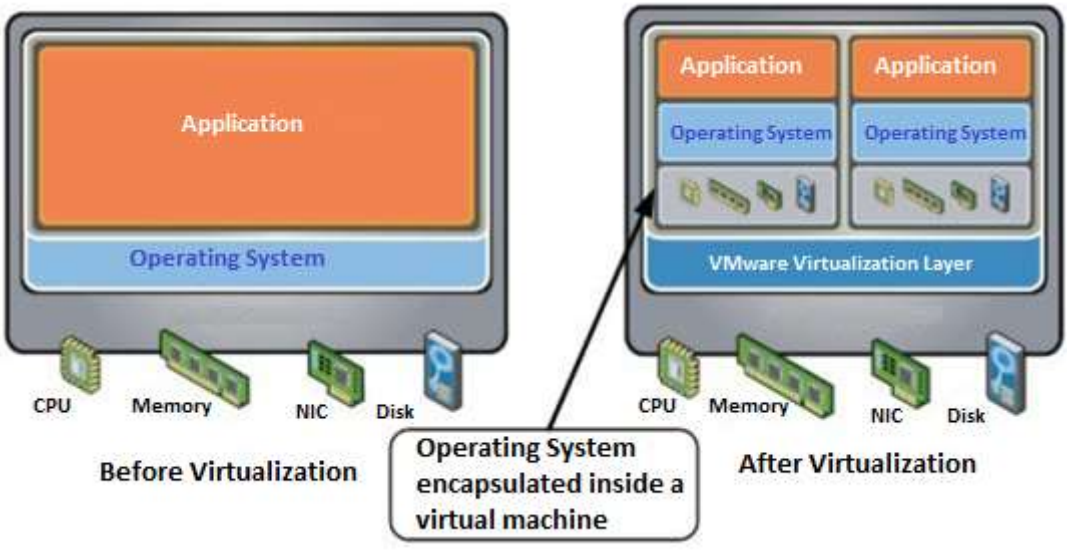
• **Βαθμονομημένη υπηρεσία (Measured Service).** Τα Υπολογιστικά συστήματα ελέγχουν αυτόματα και βελτιστοποιούν τη χρήση των υπολογιστικών πόρων χρησιμοποιώντας ένα σύστημα μέτρησης σε κάποιο από τα επίπεδα της αφαίρεσης που εισάγουν, κατάλληλο για την συγκεκριμένη παρεχόμενη υπηρεσία (συνήθως pay-per-use, charge-per-use). Η χρήση των πόρων μπορεί να παρακολουθηθεί, να ελεγχθεί και να αναφερθεί παρέχοντας διαφάνεια και στις δύο πλευρές χρήστη-παρόχου για την υπηρεσία που χρησιμοποιείται. Ακόμη υπηρεσίες όπως η διαχείριση της ασφάλειας του δικτύου, η φιλοξενία των δεδομένων σε υπολογιστικά κέντρα ή και η τιμολόγηση μπορούν εύκολα να διατεθούν.

2.1.2 Επιπρόσθετα Χαρακτηριστικά

• **Κλιμάκωση**. Στο Νέφος είναι δυνατή η προσθήκη και αφαίρεση κόμβων επεξεργασίας, εκτέλεσης εργασιών ή αποθήκευσης, ανάλογα με την αυξομείωση των απαιτήσεων, χωρίς να

αλλοιώνεται η υπηρεσία που παρέχεται στο χρήστη. Υπάρχουν δύο προσεγγίσεις για κλιμάκωση: η πρώτη scale up επιτυγχάνεται με την αύξηση των υπολογιστικών πόρων (περισσότερη μνήμη RAM, δίσκο, εικονική CPU κτλ.) ενώ η δεύτερη scale out προσθέτοντας περισσότερες μηχανές ή συσκευές ώστε να χειριστούν τη μεγάλη ζήτηση.

Η scale up κλιμάκωση μπορεί να χειριστεί ξαφνικές και προσωρινές κορυφώσεις ζήτησης εφαρμογών δεδομένου ότι δεν περιλαμβάνουν εντατικές CPU διεργασίες. Ο μόνος περιορισμός σε αυτή την περίπτωση σχετίζεται με το υλικό (hardware): Πόση μνήμη, δίσκος και επεξεργαστής μπορούν να υποστηριχθούν από ένα διακομιστή. Ενώ η scale out κλιμάκωση αναπαράγει ή αφαιρεί εικονικές μηχανές (virtual machines/VMs) για την εξισορρόπηση του φορτίου. Συνήθως ακόμη απαιτεί την προσθήκη ενός άλλου συστατικού που έχει το ρόλο του εξισορροπητή φορτίου (Load Balancer). Η scale out κλιμάκωση δεν είναι αυτοματοποιημένη και πρέπει να σχεδιαστεί μέσα στο σύστημα είναι δηλαδή ένα χαρακτηριστικό της αρχιτεκτονικής του συστήματος. Τόσο το scale out, όσο και ο Load Balancer απαιτούνται για τις σημαντικές αυξήσεις ζήτησης προκειμένου να αποκατασταθεί και να διατηρηθεί η μέγιστη απόδοση [10].



Σχήμα 2: Εικονικοποίηση σε πολυεπίπεδες αρχιτεκτονικές [13]

Αυτή η σελίδα είναι σκόπιμα λευκή

Ελαστικότητα. Σύμφωνα με το [11] η ελαστικότητα είναι ο βαθμός στον οποίο ένα σύστημα είναι σε θέση να προσαρμοστεί στις αλλαγές φόρτο εργασίας προσθέτοντας η αφαιρώντας υπολογιστικούς πόρους με αυτόνομο τρόπο, έτσι ώστε σε κάθε χρονική στιγμή οι διαθέσιμοι πόροι να ταιριάζουν με την τρέχουσα ζήτηση όσο το δυνατόν περισσότερο. Τα συστήματα θα πρέπει να εκτελούν αυτόνομα προκαθορισμένες ενέργειες κλιμάκωσης και να πληρούν τις προσυμφωνημένες απαιτήσεις απόδοσης με την ελάχιστη απαίτηση των πόρων [12]. Οι μηχανισμοί και οι ροές εργασίας που χρησιμοποιούνται από το σύστημα για την εκπλήρωση της ελαστικότητας, καθώς και τα κριτήρια αξιολόγησης και η διαδικασία λήψης αποφάσεων ποικίλλουν από το ένα σύστημα στο άλλο ή από τη μία εφαρμογή στην άλλη,

ακόμη και μέσα στο ίδιο σύστημα.

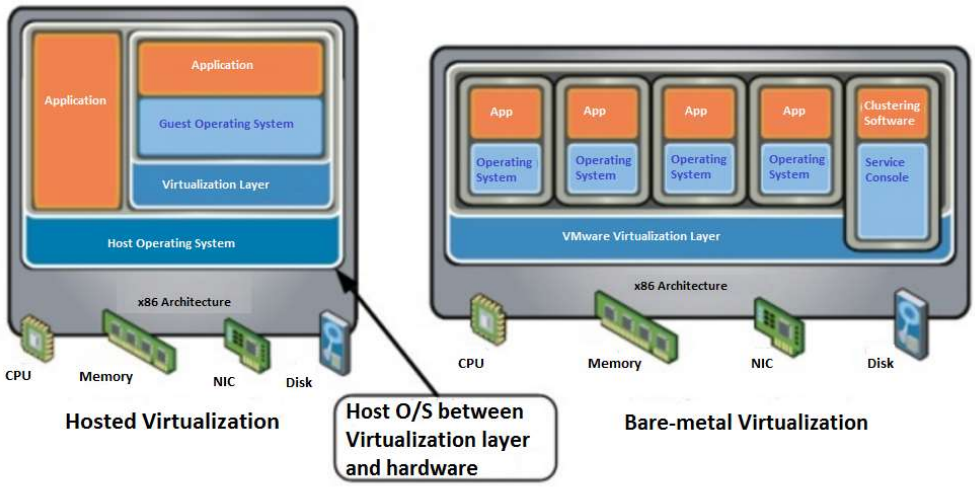
• **Εικονικοποίηση**. Δεν είναι μια νέα ιδέα [20], ουσιαστικά είναι η κύρια τεχνολογία που χρησιμοποιείται στο Υπολογιστικό Νέφος, το οποίο χρησιμοποιεί ένα φυσικό πόρο, πχ. ένα διακομιστή (server) τον οποίο διαιρεί σε εικονικούς πόρους που ονομάζονται VMs. Υπάρχουν έξι μεγάλες κατηγορίες εικονικοποίησης: υλικού, λογισμικού, μνήμης, αποθήκευσης, δεδομένων και δικτύου. Η εικονικοποίηση είναι το κλειδί για το Υπολογιστικό Νέφος, δεδομένου ότι είναι ευρείας διάδοσης τεχνολογία που επιτρέπει τη δημιουργία ενός έξυπνου επιπέδου αφαίρεσης που κρύβει την πολυπλοκότητα του υλικού ή του λογισμικού που βρίσκεται από κάτω.

Η **εικονικοποίηση του διακομιστή** σχετίζεται με τη μετακίνηση των υφιστάμενων φυσικών διακομιστών σε ένα εικονικό περιβάλλον το οποίο στη συνέχεια φιλοξενείται σε ένα φυσικό διακομιστή. Σε αυτόν τον τύπο της εικονικοποίησης εστιάζεται σήμερα το μεγαλύτερο μέρος της προσοχής καθώς και οι περισσότερες εταιρείες αρχίζουν να εφαρμόζουν αυτή την τεχνολογία. Πολλοί σύγχρονοι διακομιστές είναι σε θέση να φιλοξενήσουν περισσότερους

από έναν διακομιστές ταυτόχρονα, το οποίο επιτρέπει να μειωθεί ο αριθμός των διακομιστών που υπάρχουν στις εταιρείες, μειώνοντας έτσι τις δαπάνες. Ορισμένοι διακομιστές μπορούν επίσης εικονικοποιηθούν και να αποθηκευτούν εκτός των εγκαταστάσεών τους από άλλες εταιρείες.

Η **εικονικοποίηση λογισμικού** (hypervisor) αφαιρεί τη διαδικασία εγκατάστασης του λογισμικού και δημιουργεί εικονικές εγκαταστάσεις λογισμικού [14]. Μιμείται τη λειτουργία του υπολογιστή και επιτρέπει σε διαφορετικά λειτουργικά συστήματα να τρέξουν σε ένα ενιαίο φυσικό υπολογιστή. Κάθε ένα από τα φιλοξενούμενα λειτουργικά σύστημα φαίνεται να έχει τον επεξεργαστή και τη μνήμη του κεντρικού υπολογιστή. Ο hypervisor, ωστόσο, στην πραγματικότητα ελέγχει τον κεντρικό επεξεργαστή και τους υπολογιστικούς πόρους και προσφέρει ό,τι απαιτείται σε κάθε λειτουργικό σύστημα, εξασφαλίζοντας ότι οι VMs δεν μπορούν να διαταράξουν η μία την άλλη. Υπάρχουν δύο τύποι hypervisor οι hosted hypervisors και οι bare metal hypervisors. Οι πρώτοι λειτουργούν σαν ένα λογισμικό που χρησιμοποιεί ένα λειτουργικό σύστημα, ενώ οι bare metal χρησιμοποιούν το υλικό του κεντρικού υπολογιστή προκειμένου να ελέγχουν και να διαχειρίζονται τα φιλοξενούμενα λειτουργικά συστήματα.

Η **εικονικοποίηση του υλικού** [15] επιτυγχάνεται αφαιρώντας το φυσικό υλικό στρώμα, χρησιμοποιώντας ένα hypervisor, ο οποίος χειρίζεται τον τρόπο με τον οποίο μοιράζονται οι υπολογιστικοί πόροι ανάμεσα στα φιλοξενούμενα λειτουργικά συστήματα που εκτελούνται στον ξενιστή. Το σημαντικότερο που παρέχει αυτό το είδος της εικονικοποίησης είναι πως τα εικονικά λειτουργικά συστήματα είναι σε θέση να συνδέονται σε αυτούς τους υπολογιστικούς πόρους σαν να είναι φυσικά λειτουργικά συστήματα.



Σχήμα 3: Bare-metal και hosted τύποι εικονικοποίησης [13]

Εικονικοποίηση αποθήκευσης. Με τον όρο αυτό εννοείται η λογική παρουσίαση φυσικών πολλαπλών αποθηκευτικών συσκευών. Τα λειτουργικά συστήματα και οι εφαρμογές με raw device access προτιμούν να γράφουν κατευθείαν στο δίσκο. Οι ελεγκτές αποθήκευσης ρυθμίζουν την τοπική αποθήκευση σε ομάδες χρησιμοποιώντας την τεχνολογία RAID και παρουσιάζουν την αποθήκευση του λειτουργικού συστήματος σαν ένα υπολογιστικό όγκο (ή σαν πολλαπλούς υπολογιστικούς όγκους ανάλογα με την παραμετροποίηση). Το λειτουργικό σύστημα αποθηκεύει εντολές στους υπολογιστικούς όγκους θεωρώντας ότι γράφει κατευθείαν στο δίσκο. Ωστόσο, η αποθήκευση έχει αφαιρετικό χαρακτήρα και ο ελεγκτής καθορίζει πλέον το πώς να αποθηκεύσει ή να ανακτήσει τα δεδομένα τα οποία έχουν ζητηθεί για το λειτουργικό σύστημα.

Η **εικονικοποίηση μνήμης** [16] θεωρείται σαν εικονική μνήμη, ή σαν swap (ανταλλαγή) στους διακομιστές και στους σταθμούς εργασίας. Θεωρητικά, η διαδικασία της ανταλλαγής (swap) χρησιμοποιείται όταν η φυσική μνήμη είναι πλήρης. Ο ξενιστής βλέπει σαν πρόσθετη θέση μνήμης και δεν οριοθετεί μεταξύ RAM και swap. Όπως και στην περίπτωση του swap η εικονικοποίηση της μνήμης επιτρέπει στους διακομιστές του δικτύου να μοιράζονται το

Αυτή η σελίδα είναι σκόπιμα λευκή

σύνολο της μνήμης ώστε να ξεπεραστούν οι περιορισμοί που προκύπτουν από τη φυσική μνήμη.

Η **εικονικοποίηση των δεδομένων** [17] σχετίζεται με οποιαδήποτε προσέγγιση που αφορά στη διαχείριση των δεδομένων όπου επιτρέπει σε μια εφαρμογή να ανακτήσει και να διαχειριστεί τα δεδομένα χωρίς να απαιτούνται τεχνικές λεπτομέρειες για τα δεδομένα, όπως το πώς είναι διαμορφωμένα ή ποια είναι η φυσική τους τοποθεσία. Η διαχείριση της τοποθεσίας των δεδομένων και η διαθεσιμότητα τους μπορεί να είναι μια δύσκολη διαδικασία όταν προσπαθεί κανείς να ανασύρει και να αναλύσει δεδομένα από πολλές πηγές.

Σχετικά με την **εικονικοποίηση του δικτύου** [18], είναι πιθανό να υπάρχει, ωστόσο δεν είναι τόσο ξεκάθαρη όπως η εικονικοποίηση του διακομιστή. Οι συσκευές δικτύωσης χρησιμοποιούν τόσο την μερική εικονικοποίηση (paravirtualization), όσο και τις τεχνικές του επόπτη/hypervisor.

Το πρώτο βασίζεται στην ιδέα της μερικής εικονικοποίησης, όπου το λογισμικό που βρίσκεται από κάτω δημιουργεί ένα ξεχωριστό πίνακα προώθησης για κάθε εικονικό δίκτυο, όπως γίνεται με τα MPLS δίκτυα για κάθε VRF. Στο MPLS, το λειτουργικό σύστημα δημιουργεί μια ενιαία δρομολόγηση και προώθηση της βάσης δεδομένων για κάθε VRF, αλλά σηματοδοτεί κάθε καταχώρηση στη βάση δεδομένων με μία ετικέτα ιδιοκτησίας. Το Border Gateway Protocol (BGP) χρησιμοποιείται για την ενημέρωση της βάσης δεδομένων, και μοιράζεται τις διαδρομές και τις ετικέτες ώστε να διανείμει τα δεδομένα σε ολόκληρη το δίκτυο.

Στο δεύτερο τύπο του hypervisor, το λειτουργικό σύστημα της συσκευής δικτύου δημιουργεί πολλαπλά στιγμιότυπα του λειτουργικού συστήματος. Ίσως το πιο κοινό παράδειγμα αυτής της περίπτωσης μπορεί να είναι τα Cisco ASA firewalls, με τη χρήση των Virtual Contexts. Κάθε virtual context εμφανίζεται ως μια εντελώς ξεχωριστό ASA στιγμιότυπο και μοιράζεται

την πρόσβαση στις φυσικές διασυνδέσεις. Η επικοινωνία μεταξύ των Virtual Contexts δεν είναι δυνατή μέσα το λειτουργικό σύστημα ASA και όλη η κίνηση πρέπει να περάσει από τις φυσικές διασυνδέσεις (interfaces).

• Αξιοπιστία (Reliability): σχετίζεται με την διαβεβαίωση ότι ένα σύστημα θα εκτελέσει την προβλεπόμενη λειτουργία του για την απαιτούμενη διάρκεια σε ένα δεδομένο περιβάλλον, συμπεριλαμβανομένης της δυνατότητας να δοκιμάσει και να υποστηρίξει το σύστημα μέσω του συνολικού κύκλου ζωής του. Για το λογισμικό, η αξιοπιστία ορίζεται ως «η πιθανότητα της να μην αποτύχει η λειτουργία του λογισμικού για ορισμένο χρονικό διάστημα σε ένα καθορισμένο περιβάλλον. Οι χρήστες θεωρούν πως το Νέφος είναι μια αξιόπιστη πηγή, ειδικά αν ο πάροχος αναλαμβάνει να φιλοξενήσει και να εκτελέσει στην υποδομή του «κρίσιμες» εφαρμογές και γι' αυτό το λόγο αναμένουν σαφή οριοθέτηση της ευθύνης, αν προκύψουν σοβαρά προβλήματα [19].

• Πολλαπλή-μίσθωση: είναι μια αρχιτεκτονική στην οποία πολλοί χρήστες παρ' όλο που μοιράζονται ή δεν βλέπουν τα δεδομένα των υπολοίπων, είναι δυνατό να μοιράζονται τις ίδιες εφαρμογές, οι οποίες εκτελούνται στο ίδιο λειτουργικό σύστημα, χρησιμοποιώντας το ίδιο υλικό (hardware) και τον ίδιο μηχανισμό αποθήκευσης δεδομένων. Κάθε χρήστης μιας υπηρεσίας υπολογιστικού νέφους, δεν χρειάζεται να έχει δικό του αντίγραφο της εφαρμογής. Αρκεί ένα μοναδικό στιγμιότυπο (instance) το οποίο μπορεί να προσαρμοστεί στις ανάγκες του κάθε χρήστη. Αυτό έχει ως αποτέλεσμα την εξοικονόμηση πόρων στο νέφος και την ευκολότερη συντήρηση της εφαρμογής. Οι πληροφορίες των χρηστών αποθηκεύονται σε ένα κεντρικό σημείο και έτσι διευκολύνεται η εξαγωγή στατιστικών στοιχείων σχετικά με την εφαρμογή που βοηθούν τον πάροχο να την βελτιώσει και να την προσαρμόσει ακόμα καλύτερα στις ανάγκες των χρηστών. Στο Νέφος, η έννοια της αρχιτεκτονικής πολλαπλής μίσθωσης

έχει διευρυνθεί λόγω των νέων μοντέλων παροχής υπηρεσιών που εκμεταλλεύονται την εικονικοποίηση και την απομακρυσμένη πρόσβαση [20].

2.2 Μοντέλα Υπηρεσίας

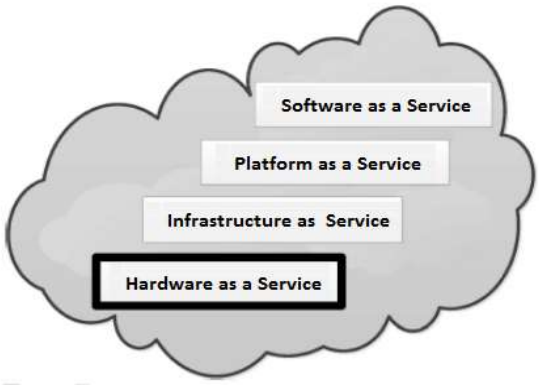
Οι υπηρεσίες του Υπολογιστικού Νέφους, σε γενικές γραμμές, χωρίζονται σε τρεις κατηγορίες βάσει του αφαιρετικού επιπέδου ελέγχου των χρηστών στους υπολογιστικούς πόρους που τους παρέχονται. Κάθε κατηγορία εξυπηρετεί έναν διαφορετικό σκοπό και παρέχει διαφορετικές προσφορές για τόσο για τις επιχειρήσεις όσο και για τους ιδιώτες. Τα πιο σημαντικά μοντέλα, σε φθίνουσα σειρά με βάση το πόσο έλεγχο επιτρέπουν στον χρήστη, είναι η υποδομή ως υπηρεσία (IaaS), η πλατφόρμα ως υπηρεσία (PaaS) και το λογισμικό ως υπηρεσία (SaaS) ενώ υπάρχουν και άλλα μοντέλα που δεν εντάσσονται στην παραπάνω κατηγοριοποίηση όπως το "οτιδήποτε" ως υπηρεσία (XaaS) και το δίκτυο ως υπηρεσία (NaaS).

- **Υποδομή ως Υπηρεσία (IaaS).** Είναι το πρώτο στρώμα του Υπολογιστικού Νέφους και μέσω αυτού παρέχεται πρόσβαση σε ουσιώδεις υπολογιστικούς πόρους [21]. Οι φυσικοί πόροι εικονικοποιούνται, το οποίο σημαίνει ότι μπορούν να μοιραστούν από διαφορετικά λειτουργικά συστήματα και περιβάλλοντα χρηστών - ιδανικά - χωρίς αμοιβαίες παρεμβολές. Χρησιμοποιώντας αυτό το μοντέλο υπηρεσίας, ο χρήστης μπορεί αναπτύξει και να διαχειριστεί οποιαδήποτε εφαρμογή. Ωστόσο, ο χρήστης δεν έχει τον έλεγχο της υποδομής αλλά έχει τη δυνατότητα να ελέγξει τα λειτουργικά συστήματα, την αποθήκευση, την ανάπτυξη των εφαρμογών καθώς και ένα μικρό έλεγχο του δικτύου. Αυτό επιτρέπει στον χρήστη να αποφύγει δαπάνες υλικού, καθώς σε αυτό το μοντέλο η χρέωση αφορά μόνο τους πόρους που χρησιμοποιούνται, και να βελτιστοποιήσει αλλά και να αυτοματοποιήσει την κλιμάκωση. Τα πλεονεκτήματα της συγκεκριμένης υπηρεσίας είναι ότι οι χρήστες πληρώνουν για όσο χρησιμοποιούν την υπηρεσία (pay-as-you-go), η άμεση κλιμάκωση, η ασφάλεια, η αξιοπιστία και η

παροχή Διεπαφών Προγραμματισμού Εφαρμογών (APIs). Κάποιοι γνωστοί πάροχοι IaaS είναι οι: Amazon, Microsoft, VMWare, Rackspace και Red Hat.

- **Πλατφόρμα ως Υπηρεσία (PaaS).** Αυτό το μοντέλο θα μπορούσε να θεωρηθεί το δεύτερο στρώμα και αποτελεί το περιβάλλον στο οποίο οι χρήστες μπορούν να αναπτύξουν τις δικές τους εφαρμογές ή να αποκτήσουν εφαρμογές χρησιμοποιώντας τις γλώσσες προγραμματισμού, τις βιβλιοθήκες και τα εργαλεία που υποστηρίζονται από τον πάροχο. Ο χρήστης διαχειρίζεται τις εφαρμογές και τα δεδομένα και ο πάροχος αναλαμβάνει και επιβλέπει όλα τα υπόλοιπα (π.χ. διανομή της εφαρμογής στην υποκείμενη υποδομή). Επιπλέον, ο πάροχος υποστηρίζει το χρήστη με ένα σύνολο βασικών υπηρεσιών για να βοηθήσει την επικοινωνία, την παρακολούθηση και τη χρέωση καθώς και διάφορα άλλα κομμάτια για την διευκόλυνση εκκίνησης μιας εφαρμογής, την εξασφάλιση της κλιμάκωσης και/ή ελαστικότητάς της. Οι υπηρεσίες αυτού του μοντέλου προσφέρουν έναν συμβιβασμό μεταξύ πολυπλοκότητας και ευελιξίας που επιτρέπει την γρήγορη υλοποίηση και διάθεση των εφαρμογών όμως εισάγουν περιορισμούς σχετικά με τις γλώσσες και τα μοντέλα προγραμματισμού που υποστηρίζονται και την πρόσβαση σε πόρους. Μια δημοφιλής PaaS υπηρεσία είναι η Google App Engine. Πλατφόρμα ως υπηρεσία (PaaS) [31]. Η ικανότητα που παρέχεται στον καταναλωτή είναι να αναπτύξει πάνω στο Νέφος υποδομή καταναλωτή που δημιουργούνται ή αποκτώνται εφαρμογές δημιουργήθηκε χρησιμοποιώντας γλώσσες προγραμματισμού, τις βιβλιοθήκες, τις υπηρεσίες και τα εργαλεία που υποστηρίζεται από τον πάροχο. Ο καταναλωτής δεν διαχειρίζεται ή ελέγχει την υποκείμενη υποδομή του Νέφους, συμπεριλαμβανομένων δικτύων, servers, λειτουργικά συστήματα, ή την αποθήκευση, αλλά έχει τον έλεγχο των

αναπτυσσόμενων εφαρμογών και, ενδεχομένως, τις ρυθμίσεις διαμόρφωσης του περιβάλλοντος εφαρμογής-hosting. Εκτός από τις προαναφερθείσες κατηγορίες στο [34], παρουσιάζεται μια προσέγγιση ενός νέου στρώματος Νέφους που ονομάζεται Hardware-as-a-Service (Haas). Το Haas επικεντρώνεται στη διαφανή ενσωμάτωση του απομακρυσμένου υλικού που διανέμεται μέσω του λειτουργικού συστήματος σε πολλαπλές γεωγραφικές τοποθεσίες. Μέσω αυτής της διαδικασίας το τοπικό σύστημα εμφανίζεται σαν όλες οι συσκευές υλικού να συνδέονται σε τοπικό επίπεδο. Το μοντέλο αυτό ευνοεί τους χρήστες των επιχειρήσεων, δεδομένου ότι δεν χρειάζεται να επενδύσουν τόσο σε κτίριο όσο και στη διαχείριση των κέντρων δεδομένων.



Σχήμα 4: Επέκταση του παραδοσιακού μοντέλου με την προσθήκη του HaaS επιπέδου [185]

- **Software as a Service (SaaS).** Η πιο διαδεδομένη μορφή παροχής υπηρεσιών Υπολογιστικού Νέφους είναι η SaaS. Είναι ένα μοντέλο λογισμικού στο οποίο οι εφαρμογές φιλοξενούνται από τον Πάροχο του Νέφους και είναι προσβάσιμες από διάφορες συσκευές client μέσω web browser ή μέσω διεπαφής. Αυτό είναι ένα "pay-as-you-go" μοντέλο και κάποια από τα πλεονέκτηματά του είναι ότι δεν χρειάζεται συγκεκριμένο υλικό για να «τρέξει» το λογισμικό, οι πελάτες πληρώνουν ανά χρήση,

Αυτή η σελίδα είναι σκόπιμα λευκή

παρέχει άμεση επεκτασιμότητα, ασφάλεια και αξιοπιστία. Επίσης, ο χρήστης δεν είναι υπεύθυνος για τη διαχείριση και τον έλεγχο της υποκείμενης υποδομής Νέφους συμπεριλαμβανομένου του δικτύου, των servers, των λειτουργικών συστημάτων, της αποθήκευσης, ή ακόμη και μεμονωμένων δυνατοτήτων της εφαρμογής. Πάροχοι τέτοιων υπηρεσιών είναι η SAP (CRM application), Microsoft Office 365, Salesforce.com, Google Apps κτλ. [30].

Στη συνέχεια παρουσιάζονται δύο πίνακες που συγκρίνουν διαφορετικές πλευρές τόσο της χρήσης όσο και της υλοποίησης των μοντέλων του Νέφους.

Μοντέλα του Νέφους	**Επίπεδο Ελέγχου που παρέχεται στο χρήστη**	**Τυπικές λειτουργίες διαθέσιμες στο χρήστη**
SaaS	Χρήση και δυνατότητα ρύθμισης των παραμέτρων της εφαρμογής	Πρόσβαση στη διεπαφή χρήστη
PaaS	Περιορισμένη διαχείριση	Μέτριο επίπεδο ελέγχου διαχείρισης των πόρων που σχετίζονται με την πλατφόρμα χρήσης των καταναλωτών
IaaS	Πλήρη διαχείριση	Πλήρη πρόσβαση σε εικονικούς πόρους υποδομών και πιθανώς σε υποκείμενους φυσικούς πόρους

Πίνακας 1: Μία σύγκριση των επιπέδων ελέγχου των τυπικών μοντέλων του Νέφους

Μοντέλα του Νέφους	Κοινές δραστηριότητες του χρήστη	Κοινές δραστηριότητες του παρόχου του Νέφους
SaaS	Χρησιμοποιεί και ρυθμίζει τις παραμέτρους της υπηρεσίας του Νέφους	• Υλοποιεί, διαχειρίζεται και συντηρεί την υπηρεσία του Νέφους • Παρακολουθεί τη χρήση του καταναλωτή
PaaS	Αναπτύσσει, δοκιμάζει και διαχειρίζεται τις υπηρεσίες και τις λύσεις του Νέφους	• Προρυθμίζει την πλατφόρμα και παρέχει την υποκείμενη υποδομή, τα ενδιάμεσα συστήματα(middleware) και άλλους πόρους που χρειάζονται • Παρακολουθεί τη χρήση του καταναλωτή
IaaS	Δημιουργεί και ρυθμίζει τις παραμέτρους της υποδομής, εγκαθιστά, διαχειρίζεται και παρακολουθεί το λογισμικό	• Παρέχει και διαχειρίζεται τη φυσική επεξεργασία, αποθήκευση, δικτύωση και την φιλοξενία που απαιτείται • Παρακολουθεί τη χρήση του καταναλωτή

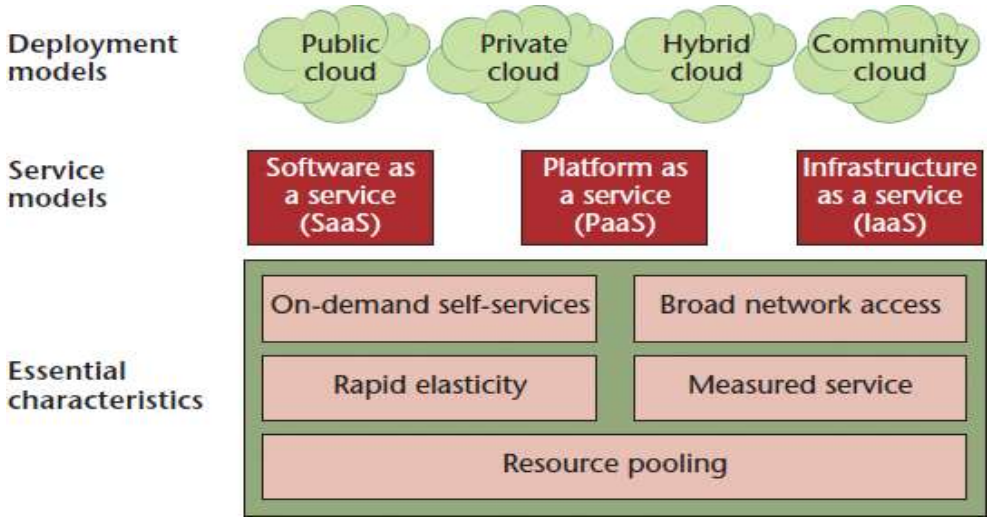
Πίνακας 2: Οι τυπικές δραστηριότητες που διεξάγονται από τους χρήστες και τους παρόχους σε σχέση με τα μοντέλα παροχής του Νέφους

2.3 Μοντέλα Ανάπτυξης του Νέφους

Λαμβάνοντας υπ΄ όψιν πώς λειτουργεί η υποδομή στο υπολογιστικό Νέφος, αλλά και πώς αυτή γίνεται διαθέσιμη στους πελάτες, δημιουργούνται τα εξής Μοντέλα Ανάπτυξης:

- Δημόσιο Νέφος (Public Cloud)
- Ιδιωτικό Νέφος (Private Cloud)
- Κοινωτικό Νέφος (Community Cloud)
- Υβριδικό Νέφος (Hybrid Cloud)

Τα διαφορετικά Μοντέλα Ανάπτυξης των υποδομών διακρίνονται από την αρχιτεκτονική τους, την τοποθεσία που έχουν εγκατεστημένο το Κέντρο Φιλοξενίας Δεδομένων που υποστηρίζει το Υπολογιστικό Νέφος και τις ανάγκες του τελικού χρήστη [22]. Στο σχήμα 5 παρουσιάζονται συνολικά τα διαφορετικά Μοντέλα Ανάπτυξης, τα μοντέλα Υπηρεσίας και τα σημαντικά χαρακτηριστικά του Υπολογιστικού Νέφους όπως ορίζονται από το NIST [173].



Σχήμα 5: Τα μοντέλα Ανάπτυξης, Υπηρεσίας και τα χαρακτηριστικά του Υπολογιστικού Νέφους όπως ορίζονται από το NIST

Αυτή η σελίδα είναι σκόπιμα λευκή

2.3.1 Δημόσιο Νέφος (Public Cloud)

Σύμφωνα με αυτό το Μοντέλο Ανάπτυξης οι Πάροχοι Υπηρεσιών Υπολογιστικού Νέφους διαθέτουν τους πόρους τους ως υπηρεσίες στους χρήστες, προσφέροντας πρόσβαση μέσω διαδικτύου, σύμφωνα με το μοντέλο «πληρωμή ανά χρήση» (Pay Per User). Στα δημόσια νέφη, όλοι οι χρήστες μοιράζονται τους ίδιους υπολογιστικούς πόρους και έχουν περιορισμένη παραμετροποίηση και ασφάλεια. Στα δημόσια νέφη, οι χρήστες ωφελούνται όσον αφορά το κόστος γιατί οι δαπάνες συντήρησης των υποδομών μοιράζονται επομένως σε κάθε πελάτη αντιστοιχεί ένα μοντέλο χαμηλού κόστους pay as you go. Οι χρήστες δηλαδή δεν χρεώνονται από πριν για δεδομένη ποσότητα υπολογιστικών πόρων, αλλά χρεώνονται ανάλογα με το βαθμό χρησιμοποίησής τους χωρίς να ανησυχούν για την κλιμάκωση της εφαρμογής τους, αφού έχουν στη διάθεσή τους εικονικά άπειρους πόρους.

2.3.2 Ιδιωτικό Νέφος (Private Cloud)

Η συγκεκριμένη υποδομή προορίζεται για την αποκλειστική χρήση από έναν ενιαίο οργανισμό ο οποίος περιλαμβάνει πολλούς καταναλωτές (π.χ., επιχειρηματικές μονάδες). Ένα Ιδιωτικό Νέφος μπορεί να αποτελεί ιδιοκτησία του πελάτη. Η εγκατάσταση, η λειτουργία και η συντήρηση του, μπορεί να ρυθμιστεί είτε από τον ίδιο είτε από κάποιον τρίτο [24]. Οι φυσικές υποδομές(εξυπηρετητές) είναι δυνατόν να βρίσκονται είτε στο χώρο του πελάτη είτε στις εγκαταστάσεις του παρόχου της υπηρεσίας [25]. Ανάλογα με το πού μπορεί να βρίσκονται οι υποδομές, τα Ιδιωτικά Νέφη χωρίζονται στις εξής κατηγορίες.

- On-Premise Private Cloud: Αυτός ο τύπος Υπολογιστικού Νέφους μπορεί να φιλοξενηθεί στις εγκαταστάσεις του ίδιου του οργανισμού και χρησιμοποιείται για την εκτέλεση εφαρμογών στις οποίες απαιτείται πλήρης έλεγχος και παραμετροποίηση της υποδομής καθώς επίσης και ασφάλεια.

- Off-Premise Private Cloud: Στην προκειμένη περίπτωση σε αντίθεση με το On-Premise Ιδιωτικό Νέφος μπορεί να φιλοξενηθεί εκτός των εγκαταστάσεων της επιχείρησης, συνήθως από κάποιον τρίτο που ειδικεύεται στην υποδομή Νέφους. Ο πάροχος δημιουργεί ένα αποκλειστικό περιβάλλον Νέφους και εγγυάται πλήρως για την ιδιωτικότητά του. Όπως γίνεται κατανοητό μέσα από την περιγραφή του συγκεκριμένου Μοντέλου Ανάπτυξης το Ιδιωτικό Νέφος είναι και το πιο ακριβό σε σχέση με τα υπόλοιπα Μοντέλα Ανάπτυξης που υπάρχουν, αφού στην ουσία οι υποδομές/πόροι δεσμεύονται αποκλειστικά για το συγκεκριμένο πελάτη. Τα ιδιωτικά νέφη επιτρέπουν σε επιχειρήσεις να φιλοξενούν εφαρμογές στο Νέφος και παράλληλα να διευθετούν θέματα που αφορούν στην ασφάλεια και στον έλεγχο δεδομένων, για τα οποία δε θέλει ο ίδιος ο πελάτης να διακινδυνεύσει τη διαρροή τους προς το υπόλοιπο δίκτυο.

2.3.3 Νέφος Κοινότητας (Community Cloud)

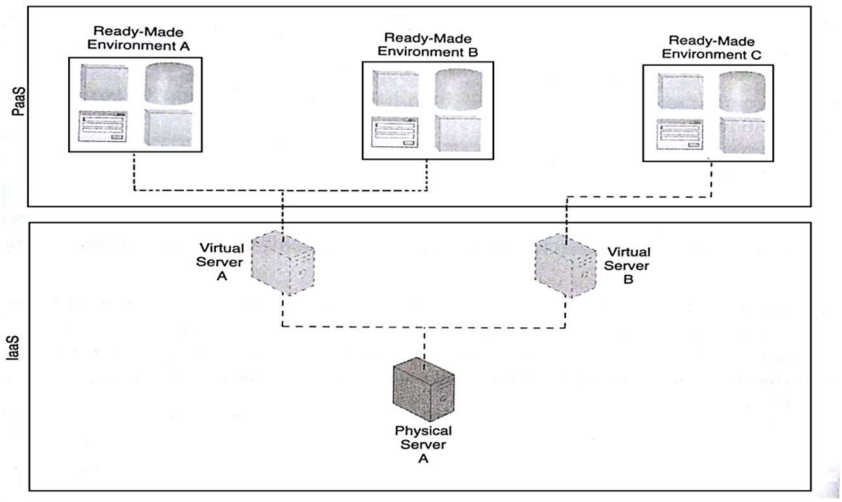
Το Νέφος Κοινότητας είναι ένα πολυ-πελατιακό μοντέλο υπηρεσιών το οποίο διαμοιράζεται από καταναλωτές που ανήκουν σε συγκεκριμένους οργανισμούς που χειρίζονται παρόμοιες υποθέσεις (π.χ. προϋποθέσεις ασφάλειας, πολιτική χρήσης, παράγοντες συμμόρφωσης κτλ.). Αυτές οι κοινότητες έχουν παρόμοιες απαιτήσεις ως προς το Νέφος και ο απώτερος σκοπός τους είναι να επιτευχθούν οι επιμέρους στόχοι του καθενός μέσα από τη συνεργασία. Επιδίωξη των Νεφών Κοινότητας είναι να μπορέσουν οι συμμετέχοντες οργανισμοί να αξιοποιήσουν τα οφέλη ενός δημόσιου νέφους σε συνδυασμό με ένα επίπεδο ιδιωτικότητας, ασφάλειας και συμμόρφωσης που συνήθως προσφέρονται από τα ιδιωτικά νέφη. Παραδείγματα οργανισμών που μπορούν να χρησιμοποιήσουν το Community Cloud ως μοντέλο ανάπτυξης, αποτελούν η βιομηχανία των μέσων μαζικής ενημέρωσης, οι τράπεζες και τα πανεπιστήμια.

2.3.4 Υβριδικό Νέφος (Hybrid Cloud)

Τα υβριδικά νέφη αποτελούν σύνθεση από δύο ή περισσότερα μοντέλα Ανάπτυξης (ιδιωτικών, δημόσιων ή κοινοτικών) τα οποία παραμένουν αυτόνομες οντότητες αλλά συνδέονται μεταξύ τους με μία τυποποιημένη ή συγκεκριμένη τεχνολογία η οποία επιτρέπει τη φορητότητα των δεδομένων και της εφαρμογής, διαμέσου των Μοντέλων Ανάπτυξης του Υπολογιστικού Νέφους (π.χ Cloud Bursting) [4] ώστε να προσφέρουν τα πλεονεκτήματα που έχει το κάθε είδος. Το μειονέκτημα αυτού του είδους είναι ότι πρέπει να παρακολουθούνται πολλαπλές πλατφόρμες ασφάλειας Νέφους καθώς και το ότι πρέπει να εξασφαλίζεται ότι τα τμήματα που βρίσκονται σε διαφορετικά Νέφη μπορούν να επικοινωνούν μεταξύ τους.

2.3.5 Συνδυασμός των μοντέλων ανάπτυξης του Νέφους

Τα τρία μοντέλα ανάπτυξης περιλαμβάνουν μία φυσική ιεραρχία παροχής, επιτρέποντας ευκαιρίες για μία συνδυασμένη εφαρμογή των μοντέλων ανάπτυξης. Ο συνδυασμός αυτός εξαρτάται από το πώς οι χρήστες και οι πάροχοι επιλέγουν να αξιοποιήσουν τη φυσική ιεραρχία που συστάθηκε από τα μοντέλα αυτά. Σε αυτή την ενότητα επισημαίνονται ζητήματα που αφορούν δύο κοινούς συνδυασμούς.



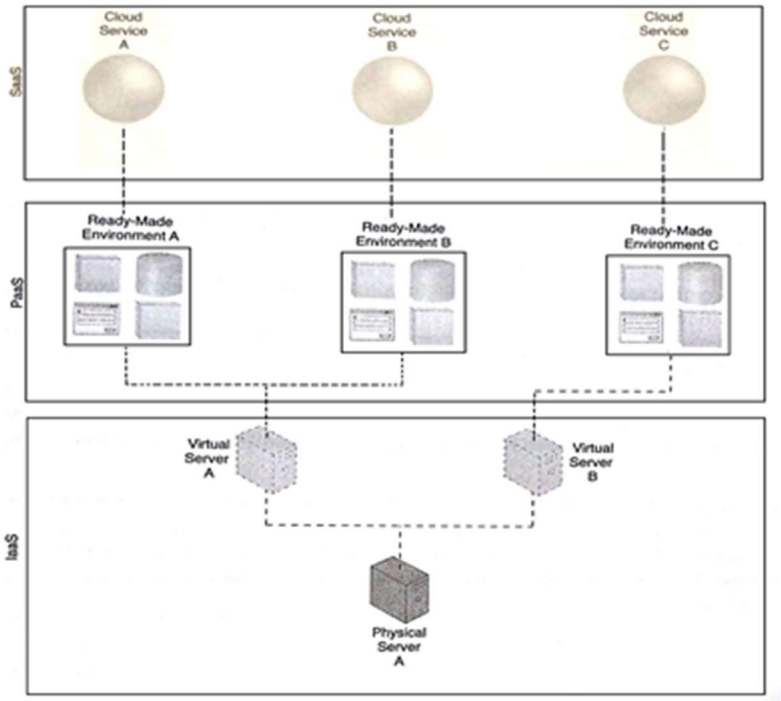
Σχήμα 6: Ένα PaaS περιβάλλον στηριζόμενο σε πόρους που παρέχονται από ένα υποκείμενο IaaS περιβάλλον

Αυτή η σελίδα είναι σκόπιμα λευκή

Iaas και PaaS

Ένα PaaS περιβάλλον μπορεί να κατασκευαστεί πάνω στην υποκείμενη υποδομή που παρέχει τους φυσικούς και τους εικονικούς διακομιστές και άλλους πόρους που παρέχονται από ένα IaaS περιβάλλον. Η εικόνα δείχνει πώς αυτά τα δύο μοντέλα μπορούν να συνδυαστούν εννοιολογικά σε μια απλή πολυεπίπεδη αρχιτεκτονική.

Το κίνητρο για μια τέτοια ρύθμιση μπορεί να επηρεαστεί από οικονομικούς λόγους ή ίσως επειδή ο πρώτος πάροχος είναι κοντά στο να υπερβεί την υπάρχουσα χωρητικότητα του υπηρετώντας τους άλλους χρήστες του Νέφους. Ή ίσως ένας συγκεκριμένος χρήστης επιβάλλει μια νομική απαίτηση για κάποια δεδομένα τα οποία θα πρέπει να είναι αποθηκευμένα σε μία συγκεκριμένη περιοχή διοαφορετική από εκείνη στην οποία βρίσκεται ο πρώτος πάροχος.



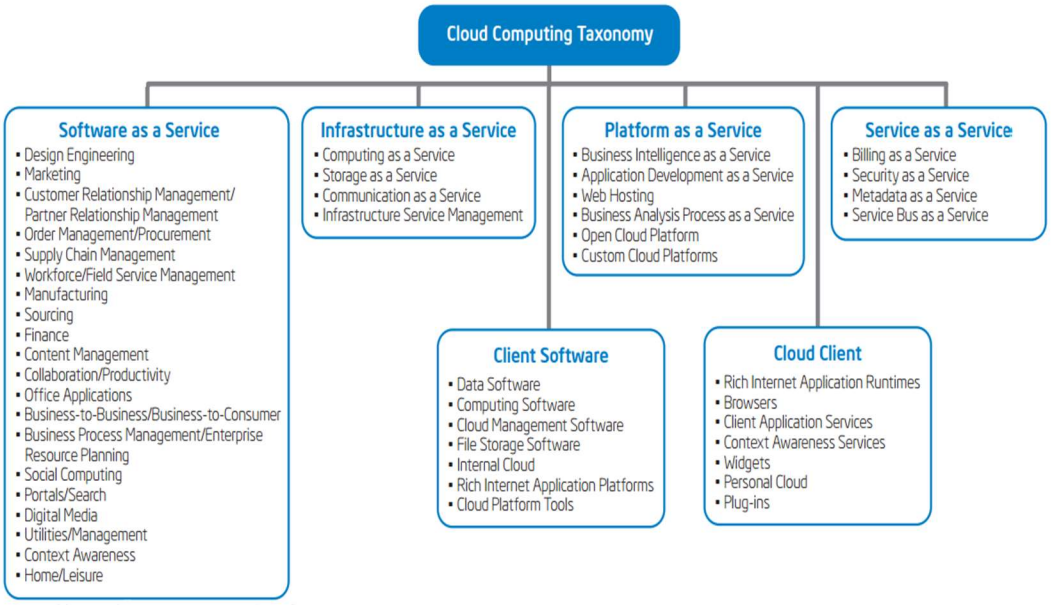
Σχήμα 7: Ένα PaaS περιβάλλον στηριζόμενο σε πόρους που παρέχονται από ένα υποκείμενο IaaS περιβάλλον

Αυτή η σελίδα είναι σκόπιμα λευκή

IaaS και PaaS και SaaS

Και τα τρία μοντέλα ανάπτυξης μπορούν να συνδυαστούν για τη δημιουργία στρωμάτων με πόρους που χτίζονται το ένα πάνω στο άλλο. Για παράδειγμα με την προσθήκη στην προηγούμενη αρχιτεκτονική όπως φαίνεται στο Σχήμα 7 το έτοιμο περιβάλλον το οποίο παρέχεται από το PaaS περιβάλλον μπορεί να χρησιμοποιηθεί από τον οργανισμό του χρήστη ώστε να αναπτύξει τις δικές του υπηρεσίες SaaS, τις οποίες μπορεί να διαθέσει ως εμπορικά προϊόντα.

2.3.6 Ταξινόμηση των διαθέσιμων Υπηρεσιών του Υπολογιστικού Νέφους



Σχήμα 8: Η ταξινόμηση της Intel για την κατάταξη του εύρους των υφιστάμενων τεχνολογιών του Νέφους

Όσον αφορά τις τρεις κύριες κατηγορίες των υπηρεσιών του Υπολογιστικού Νέφους έχουν πραγματοποιηθεί πολλές ερευνητικές προσπάθειες ώστε να καθοριστεί μια σαφής ταξινόμηση. Ένα από σημαντικότερο πλεονέκτημα της ταξινομήσεων είναι ότι παρέχουν μία κοινή ορολογία για να διευκολύνουν την κατανόηση και την επικοινωνία. Η Intel έχει δημιουργήσει το Cloud Computing Services Taxonomy [5]. Η ταξινόμηση αυτή αποτελείται από ορισμένες

Αυτή η σελίδα είναι σκόπιμα λευκή

κύριες κατηγορίες των υπηρεσιών του Υπολογιστικού Νέφους. Κάθε μία από αυτές τις κατηγορίες διαιρείται περαιτέρω σε διάφορες υποκατηγορίες. Οι συγγραφείς και στις δύο αναφορές [6] [7] προτείνουν μια οντολογία που περιγράφει το γνωστικό πεδίο των υπηρεσιών του Υπολογιστικού Νέφους. Μια άλλη ταξινόμηση για το Υπολογιστικό Νέφος έχει δημιουργηθεί [8] αυτή τη φορά από την πλευρά της επιχείρησης και των καταναλωτών αντί των πωλητών. Η ταξινόμηση που δημιουργήθηκε στο [9] έχει στηριχτεί στα διαφορετικά κοινά χαρακτηριστικά που μπορεί κανείς να βρει στο πλαίσιο των υπηρεσιών. Η σύνοψη που παρουσιάζεται στο [10] αποτελεί επίσης μια καλή ένδειξη των διαφόρων διαθέσιμων υπηρεσιών. Ωστόσο, η ταξινόμηση όπως προβλέπεται από την Intel Σχήμα 5 μπορεί να χρησιμοποιηθεί ως ένα εργαλείο καθοδήγησης, δεδομένου ότι καλύπτει εκτενώς το εύρος όσο και το βάθος των υφιστάμενων υπηρεσιών που προσφέρονται από το Υπολογιστικό Νέφος. Σύμφωνα με την Intel η ταξινόμηση περιλαμβάνει τέσσερις κύριες κατηγορίες υπηρεσιών οι οποίες περιλαμβάνουν όλες τις υπάρχουσες τεχνολογίες. Επιπλέον περιλαμβάνει κοινή ορολογία και βασικές πληροφορίες που μπορεί να εφαρμοστούν ώστε να βοηθήσουν στην ανάπτυξη στρατηγικών στο Νέφος αλλά και να προσδιορίσει καινοτόμες λύσεις που θα μπορέσουν να λειτουργήσουν αποδοτικότερα.

Δημοφιλείς Πάροχοι Νέφους και Υπηρεσίες

Στην ενότητα αυτή γίνεται εκτενής αναφορά σε τρεις δημοφιλείς εμπορικούς παρόχους του Νέφους και στις υπηρεσίες που προσφέρουν.

2.3.7 Amazon EC2

Η Amazon Elastic Compute Cloud (Amazon EC2) [33] παρέχει κλιμακούμενη υπολογιστική δυνατότητα μέσω των υπηρεσιών Νέφους Amazon Web Services (AWS). Η Amazon EC2, η οποία αποτελεί την κυριότερη προσφορά της Amazon, παρέχει ένα εικονικό υπολογιστικό περιβάλλον για τη διαμόρφωση, τη φόρτωση, την παρακολούθηση και τη διαχείριση

στιγμιότυπων που προέρχονται από προ-ρυθμισμένα, πρότυπα εικονικών μηχανών (AMIs) της Amazon, ή από εικόνες (images) που έχουν δημιουργηθεί και ρυθμιστεί από το χρήστη. Λειτουργεί σε συνδυασμό με άλλες υπηρεσίες της Amazon (όπως την αυτόματη κλιμάκωση (auto-scaling), την εξισορρόπηση φορτίου (load-balancing), την αποθήκευση (storage), τις βάσεις δεδομένων (database) και τις ουρές (queuing) οι οποίες προσφέρουν υψηλή επεκτασιμότητα, ανοχή σε σφάλματα και ασφάλεια. Δύο πολύ σημαντικές υπηρεσίες της Amazon, που επιτρέπουν την κλιμάκωση και την εξισορρόπηση φορτίου σε EC2 στιγμιότυπα είναι η **αυτόματη κλιμάκωση** (auto-scaling) και η **ελαστική εξισορρόπηση φορτίου** (elastic load balancing). Και οι δύο αυτές υπηρεσίες είναι πολύ σημαντικές, μιας και συμβάλλουν στην αύξηση της απόδοσης των εφαρμογών που φιλοξενούνται στην πλατφόρμα του Νέφους, καθώς και στη μείωση του κόστους.

Η **αυτόματη κλιμάκωση** είναι η διαδικτυακή υπηρεσία που εφαρμόζει κλιμακωτή δράση για scale-up ή scale-down για τα EC2 στιγμιότυπα που έχουν επιλεγεί, όταν έχουν εκπληρωθεί οι προϋποθέσεις. Κάθε μία από αυτές τις τρεις μεταβλητές (η πολιτική κλιμάκωσης, τα στιγμιότυπα στα οποία πρέπει να εφαρμοστεί η κλιμάκωση και οι ανησυχητικές συνθήκες (alarming conditions) ορίζονται από το χρήστη. Οι συνθήκες βάσει των οποίων εφαρμόζεται η κλιμάκωση ρυθμίζονται για κάθε μετρική μέσω της υπηρεσίας CloudWatch (εργαλείο παρακολούθησης) της Amazon. Οι χρήστες επιλέγουν τα μηχανήματα που θέλουν να παρακολουθήσουν, και λαμβάνουν έτσι πληροφορίες και μετρικές μπορούν να χρησιμοποιήσουν είτε για τη συλλογή στατιστικών αποτελεσμάτων είτε για την αυτόματη κλιμάκωση των πόρων τους. Οι μετρικές αυτές έχουν να κάνουν με τη χρήση της CPU, τα read/writes στο δίσκο, την κίνηση του δικτύου κ.α.

Σε περίπτωση που υπάρχει ζήτηση για τη σταθεροποίηση της κατάστασης πριν από την ενεργοποίηση της κλιμάκωσης υπάρχει μια μεταβλητή που ορίζει ένα χρόνο αναμονής

(cooldown). Με αυτόν τον τρόπο εξασφαλίζεται ότι η αυτόματη κλιμάκωση δεν ξεκινά ή δεν τερματίζει πρόσθετα στιγμιότυπα, πριν δράσει η προηγούμενη δραστηριότητα κλιμάκωσης.

Η **Ελαστική Εξισορρόπηση Φόρτου** είναι μια υπηρεσία που είναι υπεύθυνη για την αυτόματη διανομή εισερχόμενης κίνηση σε πολλά EC2 στιγμιότυπα. Κάθε load balancer έχει τη δυνατότητα να αναβαθμίσει την ικανότητα χειρισμού του αιτήματός του, σύμφωνα με την αύξηση της κυκλοφορίας. Εκτός από το χειρισμό των εισερχόμενων αιτημάτων, οι load balancers συμβάλλουν στην αύξηση της ανοχής σε σφάλματα της εφαρμογής με το να ανιχνεύουν μη υγιή στιγμιότυπα. Όταν ένα τέτοιο στιγμιότυπο εντοπιστεί, η υπηρεσία δεν στέλνει κυκλοφορία σε αυτό το στόχο. Ενδιαφέροντα χαρακτηριστικά της υπηρεσίας **Load Balancing** είναι:

- Η ικανότητα να τοποθετεί και να διευθύνει τα στιγμιότυπα πίσω από το Load Balancer τοπικά, χρησιμοποιώντας ως εξωτερικό σημείο μόνο τη δημόσια IP του.
- Ο συνδυασμός της εξισορρόπησης φόρτου και της αυτόματης κλιμάκωσης για διαχειριστικούς σκοπούς.
- Η ανακατεύθυνση της κυκλοφορίας σε άλλο προορισμό, εάν ο Load Balancer ή τα στιγμιότυπα της εφαρμογής φαίνεται να μην είναι διαθέσιμα (Amazon Route 53 Domain Name System υπηρεσία web).
- Ο AWS Load balancer παρέχει τη δυνατότητα για sticky load balancing. Από προεπιλογή, ένας load balancer δρομολογεί κάθε αίτημα στο στιγμιότυπο με το μικρότερο φορτίο. Ωστόσο, ο χρήστης μπορεί να χρησιμοποιήσει τη λειτουργία sticky session, η οποία επιτρέπει στο load balancer για να δεσμεύσει ένα σύνολο αιτημάτων ενός χρήστη σε ένα συγκεκριμένο παράδειγμα. Αυτό εξασφαλίζει ότι όλα τα αιτήματα από το χρήστη κατά τη διάρκεια της συνόδου αποστέλλονται στον ίδιο βαθμό.

2.3.8 Microsoft Azure

Η Microsoft Azure [34] είναι ένας πάροχος Υπολογιστικού Νέφους που διαθέτει υπηρεσίες υποδομής και πλατφόρμας. Δημιουργήθηκε από τη Microsoft, για την κατασκευή, ανάπτυξη και διαχείριση εφαρμογών και υπηρεσιών μέσω ενός παγκόσμιου δικτύου κέντρων διαχείρισης δεδομένων της Microsoft. Παρέχει PaaS και IaaS υπηρεσίες και υποστηρίζει πολλές διαφορετικές γλώσσες προγραμματισμού, εργαλεία και πλαίσια, συμπεριλαμβανομένων τόσο της Microsoft ειδικά και με άλλους κατασκευαστές λογισμικού και συστημάτων.

Σχετικά με την υποδομή ως υπηρεσία, η Windows Azure παρέχει τη δυνατότητα να φιλοξενήσει στοιχεία της εφαρμογής σε στιγμιότυπα εικονικών μηχανών. Μερικές σημαντικές πληροφορίες σχετικά με αυτές τις VMs είναι:

• Δημιουργία: Υπάρχουν δύο τρόποι δημιουργίας εικονικών μηχανών. Είτε να δημιουργηθεί άμεσα μια εικονική μηχανή επιλέγοντας μια εικόνα (image) από τις εικόνες που παρέχονται από την επίσημη ιστοσελίδα της εταιρείας Windows Azure είτε ο χρήστης να δημιουργήσει μία δικιά του εικόνα. Η ιστοσελίδα της Windows Azure περιέχει μια ποικιλία από εικόνες με διάφορα λειτουργικά συστήματα (αρκετές διανομές Linux, Windows και Mac).

• Κλιμάκωση: Οι VMs μπορούν να κλιμακωθούν αυτόματα αυξάνοντας ή μειώνοντας τον αριθμό τα στιγμιότυπα των εικονικών μηχανών που χρησιμοποιούνται από την εφαρμογή. Η κλιμάκωση μπορεί να διαμορφωθεί με βάση τη μέση χρήση της CPU και τον αριθμό των μηνυμάτων στην ουρά.

• Τα στιγμιότυπα των εικονικών μηχανών με τη μέθοδο Εξισορρόπησης Φόρτου μπορεί να υλοποιηθεί με δύο τρόπους. Ο πρώτος είναι με τη δημιουργία ενός εξισορροπημένου φορτίου τελικό σημείο (TCP ή UDP τελικό σημείο) που χρησιμοποιείται από όλα τα στιγμιότυπα των εικονικών μηχανών που περιέχονται σε μια υπηρεσία Νέφους. Το τελικό σημείο μπορεί να

παρέχει εξισορρόπηση φόρτου χρησιμοποιώντας τον αλγόριθμο Round-Robin (RR). Ο δεύτερος είναι, με τη χρήση ενός Διαχειριστή Κυκλοφορίας (Traffic Manager) για την εκτέλεση εξισορρόπησης φόρτου εφαρμόζοντας μία από τις τρεις διαθέσιμες μεθόδους πολιτικής:

• Πολιτική της απόδοσης (latency): Στην περίπτωση που υπάρχουν τελικά σημεία σε διαφορετικές γεωγραφικές περιοχές η κίνηση κατευθύνεται και φιλοξενείται από την κοντινότερη γεωγραφικά υπηρεσία.

• Πολιτική της ανακατεύθυνσης: Εφαρμόζεται όταν ο χρήστης θέλει να χρησιμοποιήσει ένα πρωτεύον καταληκτικό σημείο για όλη την κυκλοφορία αλλά σε στην περίπτωση που το σημείο αυτό δεν είναι διαθέσιμο τότε ένα δευτερεύον τελικό σημείο παρέχεται.

• Round Robin πολιτική. Η πολιτική αυτή μπορεί να εφαρμοστεί όταν ο χρήστης θέλει να κατανείμει το φορτίο σε όλη την έκταση ενός συνόλου υπηρεσιών στο ίδιο ή σε διαφορετικά κέντρα δεδομένων.

Όσον αφορά την υπηρεσία ως πλατφόρμα, η Windows Azure παρέχει την αντίστοιχη υπηρεσία Νέφους. Η συγκεκριμένη υπηρεσία αποτελείται από τον κώδικα και τη διαμόρφωση τη εφαρμογής που φιλοξενείται στην πλατφόρμα της Azure. Η υποδομή, τα στιγμιότυπα και τα λειτουργικά συστήματα συντηρούνται από την Azure, ενώ οι αναβαθμίσεις πραγματοποιούνται χωρίς καμία διακοπή της υπηρεσίας. Ακόμη προσδιορίζονται δύο τύποι ρόλων:

- Ο **web ρόλος** είναι ένα ειδικός web-server που χρησιμοποιείται για τη φιλοξενία των frontend εφαρμογών.
- Ο **worker ρόλος** χρησιμοποιείται για την εξυπηρέτηση αιτημάτων στο παρασκήνιο, ανεξάρτητα από την αλληλεπίδραση του χρήστη ή την είσοδο, και έτσι επιτρέπει την ασύγχρονη καθώς και τη μακρόχρονη ή τη διαρκή εκτέλεση μιας εργασίας.

Η λειτουργία των υπηρεσιών Νέφους στηρίζεται στα στιγμιότυπα. Ένα ή περισσότερα στιγμιότυπα μπορούν να χρησιμοποιηθούν για ένα συγκεκριμένο ρόλο. Ο αριθμός των στιγμιότυπων μπορεί να αυξηθεί ή να μειωθεί ανάλογα με τις απαιτήσεις της κλιμάκωσης.

2.3.9 Google App Engine (GAE)

Η Google App Engine(GAE) [35] είναι μια υπηρεσία για τη φιλοξενία διαδικτυακών εφαρμογών (εφαρμογών του ιστού), επιτρέποντας την ανάπτυξη των εφαρμογών μέσα σε ένα προκαθορισμένο περιβάλλον χρόνου εκτέλεσης. Σε αντίθεση με άλλες αντίστοιχες υπηρεσίες του Νέφους, όπως είναι η Amazon Web Services, που λειτουργεί σε επίπεδο IaaS, η GAE παρέχει ήδη μια εφαρμογή υποδομής στο επίπεδο PaaS. Αυτό σημαίνει ότι η GAE λειτουργεί αφαιρετικά από το υποκείμενο υλικό και το λειτουργικό σύστημα, παρέχοντας στην φιλοξενούμενη εφαρμογή ένα σύνολο υπηρεσιών με γνώμονα την εφαρμογή. Η προσέγγιση αυτή είναι πολύ βολική για τους προγραμματιστές των εφαρμογών αυτών, ενώ η λογική πίσω από τη GAE είναι η επικέντρωσή της στην επεκτασιμότητα και τη χρήση της υποδομής.

2.3.10 Flexiant Cloud Provider

Η Flexiant [36] είναι μια εταιρεία/πάροχος με έδρα το Ηνωμένο Βασίλειο που παρέχει μία πλήρη IaaS υποδομή. Δύο σημαντικές υπηρεσίες που προσφέρει είναι η Flexiant Cloud Orchestrator και η Flexiant Concerto. Η πρώτη είναι μια αξιόπιστη πλατφόρμα με πολλά χαρακτηριστικά που συμβάλλουν στην καλύτερη διαχείριση του Νέφους. Ένα από τα χαρακτηριστικά που ξεχωρίζουν είναι η μείωση του κίνδυνου που εγκυμονεί κατά τη μεταφορά μιας εφαρμογής στο Νέφος, παρέχοντας στους χρήστες τη δυνατότητα να επιλέξουν ποιον hypervisor θα χρησιμοποιήσουν για το δικό τους φορτίο, υποστηρίζοντας όλους τους hypervisors εμπορικούς (VMware vSphere, Microsoft Hyper-V and Odin Virtuozzo) αλλά και ανοιχτού κώδικα (KVM, Xen4). Ακόμη παρέχει ένα δυναμικό αλγόριθμο τοποθέτησης του

φορτίου για την επιλογή της πιο λογικής λύσης για το σημείο από το οποίο θα πρέπει να ξεκινήσει μία εικονική μηχανή καθώς και την απευθείας μεταφορά και αποκατάσταση στην περίπτωση που αποτύχει κάποιος κόμβος. Η δεύτερη υπηρεσία προσφέρει στους χρήστες έναν γρήγορο τρόπο ώστε να αναπτύξουν και να αυτοματοποιήσουν τις εφαρμογές τους σε διαφορετικούς παρόχους του Νέφους αποφεύγοντας το κλείδωμα του πελάτη (vendor lock-in).

Αυτή η σελίδα είναι σκόπιμα λευκή

3 *Η απόδοση στο Υπολογιστικό Νέφος*

Η απόδοση είναι ένα χαρακτηριστικό του Υπολογιστικού Νέφους που έχει κερδίσει την προσοχή της ερευνητικής κοινότητας τα τελευταία χρόνια, ειδικά τώρα που σχετίζεται με πιο εκτεταμένες έννοιες όπως είναι η διαθεσιμότητα πόρων, η επάρκεια και η αξιοπιστία [76]. Γενικά, η απόδοση [37] είναι συνδεδεμένη με δυνατότητες μιας εφαρμογής μέσα στην ίδια υποδομή του Νέφους. Ο περιορισμός στο εύρος ζώνης, στο χώρο του δίσκου, της μνήμης, της CPU, και στις δικτυακές συνδέσεις μπορούν να προκαλέσουν μείωση της απόδοσης. Σε ορισμένες περιπτώσεις, η κακή απόδοση αποτελεί συνδυασμό της έλλειψης πόρων, ενώ σε κάποιες άλλες οφείλεται στην αρχιτεκτονική της εφαρμογής η οποία δε διανέμει σωστά τις διαδικασίες της σε όλους τους διαθέσιμους πόρους στο Νέφος. Οι ενδιαφερόμενοι φορείς του Νέφους, όπως οι πάροχοι των υποδομών, του λογισμικού και οι τελικοί χρήστες έχουν διαφορετικές ανησυχίες που σχετίζονται με την απόδοση. Οι πάροχοι των υποδομών, δίνουν έμφαση στην αξιοποίηση των πόρων που σημαίνει ότι ενδιαφέρονται για την έγκαιρη απελευθέρωση πόρων, έτσι ώστε το σύστημα να μπορεί να τους ανακατανείμει σε άλλες εφαρμογές και πελάτες. Υπό το πρίσμα των παρόχων υπηρεσίας θα πρέπει να υπάρξει ισορροπία μεταξύ της απόδοσης του συστήματος και του κόστους της κράτησης πόρων. Αν οι χρήστες κρατούν τους πόρους περισσότερο από ό, τι χρειάζεται, το κόστος θα είναι πολύ μεγαλύτερο και θα θεωρηθεί ως σπατάλη πόρων. Ωστόσο, εάν οι πόροι ενοικιαστούν για λιγότερο χρόνο από ότι χρειάζεται, οι πάροχοι δεν μπορούν να εγγυηθούν διαθεσιμότητα της υπηρεσίας και τον απαιτούμενο χρόνο απόκρισης.

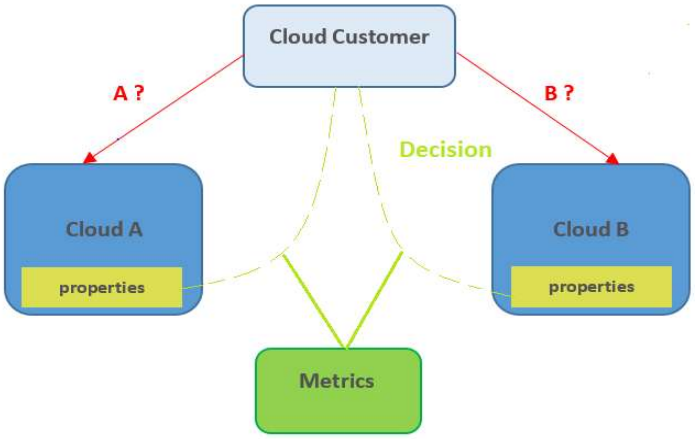
Τα προβλήματα που σχετίζονται με την αστάθεια στην επίδοση των περιβάλλοντων του Νέφους [77] έκαναν αισθητή την παρουσία τους μετά τις υποσχέσεις των παρόχων για απεριόριστους πόρους και την κατ' απαίτηση κλιμάκωση. Επομένως, για να είναι επιτυχημένη η διαδικασία της μετάβασης (migration) μιας εφαρμογής στο Νέφος, είναι πολύ σημαντικό να ληφθεί υπόψη το ζήτημα της απόδοσης του εκάστοτε παρόχου, προκειμένου οι ιδιοκτήτες των εφαρμογών να εξοικονομήσουν χρήματα, αλλά και εγγυήσεις όσον αφορά στην σταθερότητα (όσο το δυνατόν) μετά την εγκατάσταση της εφαρμογής που περιλαμβάνονται στο Συμβόλαιο Διασφάλισης Επιπέδου Ποιότητας (SLA). Στο [78], παρουσιάζονται τόσο το ευρύ φάσμα των εφαρμογών που έχουν αναπτυχθεί σε περιβάλλοντα Νέφους, όσο και η μεταβλητότητα των υπηρεσιών του Νέφους.

3.1 Γενικές μετρήσεις της απόδοσης

Σε αυτή την ενότητα παρουσιάζεται ένα σύνολο μετρήσεων της απόδοσης του Νέφους οι οποίες σχετίζονται με τυπικές εφαρμογές που χρησιμοποιεί ο πελάτης και καλύπτουν τις βασικές υπηρεσίες του Νέφους.

Σήμερα, υπάρχουν πολλές εταιρείες που μεταφέρουν ολόκληρες τις εφαρμογές τους ή ένα μέρος αυτών στο Νέφος. Ως συνέπεια, οι μετρήσεις που υπάρχουν για την απόδοση (π.χ. το ποσοστό της CPU που διατίθενται για τις VMs, οι I/O λειτουργίες του δίσκου, η κοινή χρήση της κρυφής μνήμης κτλ.) θα πρέπει να ληφθούν υπόψη, δεδομένου ότι διαφορετικοί τύποι εφαρμογών έχουν διαφορετικές παρεμβολικές επιπτώσεις σε κατανεμημένα και εικονικά περιβάλλοντα. Ως εκ τούτου, οι χρήστες στηριζόμενοι στις υπάρχουσες μετρήσεις της απόδοσης καλούνται να επιλέξουν εκείνη την υπηρεσία του Νέφους, που θα παρέχει καλή απόδοση προκειμένου να αποφευχθούν προβλήματα που σχετίζονται με την επεκτασιμότητα

και τη διαχείριση των δεδομένων. Ωστόσο η απρόβλεπτη απόδοση του Νέφους αποτελεί το σημαντικότερο εμπόδιο στη δημιουργία αξιόπιστων υπηρεσίων.



Σχήμα 9: Επιλογή κατάλληλης υπηρεσίας του Νέφους βάσει των μετρήσεων της απόδοσης

Στο [38] προσδιορίζονται ορισμένες ζωτικής σημασίας μετρήσεις της απόδοσης όπως:

- Η μέτρηση του χρόνου τερματισμού με τη χρήση της συγκριτικής αξιολόγησης (benchmarking). Αυτή η μέτρηση μέτρα το χρόνο που διαρκεί η ολοκλήρωση των εργασιών των tests της σιγκριτικής αξιολόγησης τα οποία στρεσάρουν τους κύριους υπολογιστικούς πόρους (CPU, τη μνήμη, και το δίσκο I/O).
- Κλιμάκωση λανθάνοντα χρόνου/συνολικής καθυστέρησης (latency): Είναι ο χρόνος που χρειάζεται ένας πάροχος να διαθέσει ένα νέο στιγμιότυπο όταν ζητηθεί από τον πελάτη. Η κλιμάκωση λανθάνοντα χρόνου μπορεί να επηρεάσει την απόδοση και το κόστος της λειτουργίας μιας εφαρμογής.
- Μόνιμη Αποθήκευση: Υπάρχουν τρεις κοινοί τύποι υπηρεσιών αποθήκευσης που προσφέρουν οι πάροχοι για την εφαρμογή και τα δεδομένα: ο πίνακας (table), το BLOB (Binary Large Object) και η ουρά (queue). Η αποθήκευση στο Νέφος έχει δύο πλεονεκτήματα: την επεκτασιμότητα και τη διαθεσιμότητα. Χρησιμοποιούνται τρεις

Αυτή η σελίδα είναι σκόπιμα λευκή

μετρήσεις για να συγκριθούν η απόδοση και το κόστος των υπηρεσιών αποθήκευσης: ο χρόνος λειτουργίας, ο χρόνος για επιστροφή έγκυρων αποτελεσμάτων (με συνοχή) και ο χρόνος για κάθε διεργασία.

- Χρόνος απόκρισης της λειτουργίας: Αυτή η μέτρηση μετρά το χρόνο που χρειάζεται για να ολοκληρωθεί μια διεργασία αποθήκευσης.
- Intra-Cloud Δίκτυο: Το δίκτυο αυτό συνδέει τα στιγμιότυπα ενός πελάτη μεταξύ τους και με τις κοινές υπηρεσίες που προσφέρονται από το Νέφος. Για να συγκριθεί η απόδοση των δικτύων κάποιες κοινές μετρήσεις είναι το path capacity και ο λανθάνων χρόνος.
- Δίκτυο ευρείας περιοχής (Wide-area Network): περιλαμβάνει τη συλλογή των διαδρομών του δικτύου μεταξύ ενός κέντρου δεδομένων στο Νέφους και των εξωτερικών host στο Διαδίκτυο. Για τη φιλοξενία των εφαρμογών των πελατών υπάρχουν πολλές τοποθεσίες, έτσι ώστε αιτήματα από έναν τελικό χρήστη να μπορούν να εξυπηρετηθούν από ένα στιγμιότυπο που βρίσκεται σε κοντινή περιοχή σε αυτόν το χρήστη, ώστε να μειωθεί ο χρόνος καθυστέρησης.

Σύμφωνα με το [39] μερικές από τις πιο ενδιαφέρουσες μετρήσεις της απόδοσης περιλαμβάνονται στον *Πίνακας 3*:

Performance metric	Description
Scalability based metrics	
CPU capacity	CPU's speed in flops
Memory size	In general, cache memory size for a VM
Scale up	Maximum number of VMs allocated for a user

Scale down	Minimum number of VMs allocated for a user
Boot time	Booting time for a VM to get ready for usage
Storage capacity	Storage size of data
Scale uptime	Time taken for increasing a specific number of VMs
Scale downtime	Time taken for decreasing a specific number of VMs
Autoscale	Boolean value for autoscaling feature
Response time	Time required to complete and receive a process
Architecture specific metrics	
Pipeline stalls	Processor specific pipelines stalls e.g. IA64
Cache misses	L2 or L3 cache misses
Frequent voltage switches	Voltage variations caused due to applications in processors such as Νεγalem

Πίνακας 3: Οι πιο σημαντικές μετρήσεις τις απόδοσης

Άλλες ενδιαφέρουσες μετρήσεις που έχουν εντοπιστεί στο [40] είναι i) η ταχύτητα της μνήμης που είναι σημαντική κυρίως για τις data-intensive εφαρμογές, όπως τις DBMSs ή το MapReduce ii) οι I/O του δίσκου (διαδοχικές και τυχαίες): πολλές εφαρμογές του Νέφους απαιτούν στιγμιότυπα για να αποθηκεύσουν τα ενδιάμεσα αποτελέσματα σε τοπικούς δίσκους αν τα δεδομένα εισόδου δεν μπορούν να υποβληθούν σε επεξεργασία στην κύρια μνήμη iii) το εύρος ζώνης δικτύου (network bandwidth) και το GPU φορτίο μεταξύ των στιγμιότυπων,

διότι η εφαρμογή ανταλλάσσει μέσω των δικτύων μεγάλες ποσότητες δεδομένων και iv) τα δεδομένα που υποβάλλονται σε επεξεργασία ανά δευτερόλεπτο και δεδομένα που υφίστανται επεξεργασία ανά Joule, προκειμένου να αξιολογηθεί ολόκληρο το σύστημα του Νέφους για μεγάλες εφαρμογές δεδομένων.

Όσον αφορά στις μετρήσεις για την απόκλιση σύμφωνα με τα αποτελέσματα μιας έρευνας [41] τόσο το μικρό στιγμιότυπο (που αντιστοιχεί σε 1.76B κύριας μνήμης 1 ECU), όσο και το μεγάλο (που αντιστοιχούν σε 7,4GB κύριας μνήμης, 4ECU) παρουσιάζουν μια μεγάλη διακύμανση στην απόδοση που είναι συνέπεια των διαφορετικών τύπων συστήματος που χρησιμοποιούνται από τους εικονικούς κόμβους. Επίσης, οι μετρήσεις που αναφέρονται στο χρόνο εκτελέσεις (runtime) στο Νέφος διακρίνονται από υψηλή διακύμανση. Δεδομένου ότι οι χρήστες του Νέφους χρειάζονται σταθερότητα όταν χρησιμοποιούν τους πόρους που νοικιάζουν (π.χ. εφαρμογές που απαιτούν σημαντικό χρόνο εκτέλεσης ώστε να ολοκληρωθούν ή οι επιχειρήσεις να διαθέσουν το κατάλληλο ποσό των πόρων για τις εφαρμογές τους εγγυώντας στους πελάτες τους QoS), είναι απαραίτητο να λαμβάνονται υπόψη κάποιες μετρήσεις για τη σταθερότητα της κάθε παρατηρούμενης μέτρησης όπως:

- Η επαναλαμβανόμενη διαδικασία μέτρησης κατά την πάροδο του χρόνου ώστε να παρατηρηθούν οι διακυμάνσεις των προσφερόμενων υπηρεσιών του Νέφους.
- Τυπική απόκλιση της μετρούμενης μέτρησης μιας υπηρεσίας ενός παρόχου του Νέφους.
- Πιθανοτική πληροφορία της διανομής (π.χ. διαστήματα εμπιστοσύνης, τύπος διανομής).

3.2 Στερεότυπα και εξαγωγή των χαρακτηριστικών απόδοσης

Ο κύριος στόχος των στερεότυπων απόδοσης [42] είναι να εξάγουν μια σειρά από χαρακτηριστικά των επιδόσεων του παρόχου που είναι απαραίτητα ώστε να καλύψουν τις απαιτήσεις για την ποιότητα της υπηρεσίας QoS των εφαρμογών που μεταφέρονται στο Νέφος. Προκειμένου να επιτευχθεί αυτό, πρέπει να οριστεί ένα συγκεκριμένο σύνολο από αυτά τα χαρακτηριστικά, έτσι ώστε να μπορούν να μετρηθούν. Η προέλευση αυτών των χαρακτηριστικών πηγάζει από τα εξής πεδία:

- Από τους γενικούς τύπους των εφαρμογών που υπάρχουν στη σύγχρονη δημιουργία λογισμικού (όπως οι διακομιστές εφαρμογών, οι DB servers, κτλ.). Αυτοί οι τύποι εφαρμογών δείχνουν διαφορετικά πρότυπα χρήσης των υποκείμενων φυσικών πόρων (κυρίως CPU, αποθήκευση και δίκτυο). Κάποιες από τις εφαρμογές αυτές μπορεί να αναφέρονται σε τυχαίες λειτουργίες ανάγνωσης ή σε διαδοχικές λειτουργίες εγγραφής, σε κυμαινόμενους υπολογισμούς ή σε πολλαπλασιασμός πινάκων, σε κανονικές ή σε ακανόνιστες προσβάσεις στη μνήμη κλπ). Οι εν λόγω πληροφορίες σχετικά με αυτές τις πιθανές κατηγορίες εφαρμογών θα μπορούσε να προέλθει από τη χρήση της συγκριτικής αξιολόγησης που στοχεύει σε συγκεκριμένους τύπους εφαρμογών. Το βασικό αποτέλεσμα αυτής της διαδικασίας είναι να εντοπιστούν συγκεκριμένες κατηγορίες υπολογιστικών προφίλ που ταιριάζουν σε κοινούς τύπους εφαρμογών.
- Από τα χαρακτηριστικά που διακρίνουν το κάθε υπολογιστικό Νέφος, τα οποία θα μπορούσαν να επηρεάσουν την απόδοση και τη λειτουργία της εφαρμογής. Αυτά μπορεί να προέρχονται από τη φύση του Υπολογιστικού Νέφους και την επεκτασιμότητα των πόρων. Κάποια παραδείγματα αυτών των χαρακτηριστικών είναι οι δυνατότητες επεκτασιμότητας, οι καθυστερήσεις ελαστικότητας (π.χ. το πόσο γρήγορα μπορεί να αρχίσει τη λειτουργία της μια εικονική μηχανή και να

συμπεριληφθεί στην υπηρεσία) και οι δυνατότητες ασφάλειας (π.χ. ανθεκτικότητα σε διάφορους τύπους επιθέσεων DoS). Σχετικά με τα χαρακτηριστικά αυτά θα πρέπει να ληφθεί υπόψη ότι οι πάροχοι δεν μοιράζονται τις πληροφορίες σχετικά με τον τρόπο που διαχειρίζονται και διαμορφώνουν το Νέφος, επομένως ο μόνος τρόπος για να καθοριστούν οι ικανότητές τους είναι μέσω μιας μακροσκοπικής παρατήρησης (π.χ. με την έναρξη μιας επίθεσης DoS και τη χρήση μετρήσεων όπως τη server response degradation/υποβάθμιση της ανταπόκρισης του διακομιστή κλπ.).

- Από τις συγκεκριμένες υπηρεσίες του Νέφους που μελετήθηκαν στο κεφάλαιο 2. Αυτές οι υπηρεσίες μπορεί να έχουν συγκεκριμένες μετρήσεις που πρέπει να μελετηθούν και να εξεταστούν ανά περίπτωση. Για παράδειγμα, κάποιες εξειδικευμένες υπηρεσίες, όπως τα MapReduce clusters διαθέτουν τα δικά τους tests επίδοσης (π.χ. Terasort). Το ίδιο ισχύει και για κάποιες υπηρεσίες συγκεκριμένου σκοπού, όπως η παρακολούθηση (monitoring), η τιμολόγηση (billing) κ.λπ. Η απόδοση των υπηρεσιών αυτών μπορούν να επηρεάσουν άμεσα την απόδοση των εφαρμογών (π.χ. εάν η ελαστικότητα της εφαρμογής βασίζεται στην πληροφορία της τιμολόγησης ή της παρακολούθησης όσον αφορά τους πόρους επομένως οποιαδήποτε καθυστέρηση αυτών των υπηρεσιών να είναι περιοριστική).

3.3 Παράγοντες που επηρεάζουν την απόδοση του Υπολογιστικού Νέφους

Οι κύριοι παράγοντες που επηρεάζουν την απόδοση του Υπολογιστικού Νέφους μπορούν να συνοψιστούν ως εξής:

- Ετερογενείς και άγνωστοι πόροι υλικού: οι υπολογιστικοί πόροι που προσφέρονται από τους παρόχους του Νέφους είναι άγνωστοι στους εξωτερικούς χρήστες. Οι

διαθέσιμες πληροφορίες μπορεί να περιορίζονται στον αριθμό των πυρήνων, στο μέγεθος της μνήμης ή στην ποσόστωση του δίσκου. Ωστόσο, αυτό το επίπεδο πληροφορίας δεν είναι επαρκές για να χαρακτηρίσει τις δυνατότητες του υλικού του παρόχου διότι μπορεί επίσης να εξαρτάται, από την αρχιτεκτονική, τη διασύνδεση, την ταχύτητα της RAM κλπ. Σύμφωνα με μια μελέτη για την πλατφόρμα της Amazon πραγματοποιήθηκε από το Πανεπιστήμιο Aalto [43] η διακύμανση μεταξύ των γρήγορων και των αργών στιγμιότυπων μπορεί να φτάσει το 40%. Σε ορισμένες εφαρμογές, η διακύμανση μπορεί να προσεγγίσει ακόμη και το 60%.

- Διαφορετικές ρυθμίσεις παραμέτρων: ακόμη και στην περίπτωση που υπάρχει το ίδιο υλικό, σημαντικό ρόλο στην απόδοση διαδραματίζει ο τρόπος με τον οποίο έχουν ρυθμιστεί οι παράμετροι. Το ίδιο ισχύει και για τη ρύθμιση των παραμέτρων του λογισμικού (π.χ. ένα στιγμιότυπο βάσης πάνω από ένα εικονικό cluster) ή παραλλαγές στην ανάπτυξη του λογισμικού. Για παράδειγμα, για το μοντέλο ζευγών κλειδιού-τιμής για τη βάση δεδομένων Κασσάνδρα (Cassandra) μπορεί να υπάρξει διαφορά όταν δοκιμάζεται με μια συγκεκριμένη ρύθμιση παραμέτρων σε ένα ιδιωτικό Νέφος για μια ήδη υπάρχουσα εφαρμογή πληροφορικής και όταν μεταφερθεί στο δημόσιο Νέφος με την χρήση της βάσης δεδομένων Κασσάνδρα ως λογισμικό ως υπηρεσία (SaaS). Στη δεύτερη περίπτωση μπορεί να παρατηρηθούν διακυμάνσεις στην απόδοση που βασίζονται στη ρύθμιση των παραμέτρων του λογισμικού ως υπηρεσία.
- Πολλαπλή-μίσθωση και μη γνωστός τρόπος διαχείρισης από τους παρόχους: ένα από τα κύρια ζητήματα επιδόσεων σε υποδομές Νέφους είναι ότι ασχολούνται με πολλούς διαφορετικούς χρήστες που μπορεί να ξεκινήσουν τους εικονικούς τους πόρους στην ίδια φυσική υποδοχή ανά πάσα στιγμή. Ωστόσο, για παράδειγμα η επίδραση των εικονικών μηχανών που λειτουργούν ταυτόχρονα **Error! Reference source not**

found. υποβαθμίζει σημαντικά την πραγματική απόδοση των εφαρμογών. Αυτό επηρεάζεται ακόμα περισσότερο από τη χρήση των πόρων αυτών από τους ιδιοκτήτες ή τους πελάτες τους. Τέλος η απόδοση μπορεί να επηρεαστεί από τις αποφάσεις των παρόχων, οι οποίοι μπορεί ανά πάσα στιγμή να ομαδοποιήσουν οποιαδήποτε στιγμή τους εικονικούς πόρους ώστε να αντιστοιχούν στον ίδιο φυσικό κόμβο σε οποιαδήποτε δεδομένη στιγμή χωρίς να ενημερώσουν τους χρήστες.

- Επίδραση από τις παρεμβολές των εικονικών μηχανών. Στο [44] μια ενδιαφέρουσα έρευνα μελετά κατά πόσο επηρεάζεται η απόδοση ενός αριθμού εφαρμογών σε πειραματικά εικονικά περιβάλλοντα που επιλέχθηκαν για να ταξινομηθούν χρησιμοποιώντας διαφορετικές μετρήσεις. Επιπλέον, αναπτύχθηκε ένας μηχανισμός ο οποίος προβλέπει τις επιδόσεις των εφαρμογών για κάθε μία από τις οποίες εκτελούνται διαφορετικοί τύποι φορτίων. Ο μηχανισμός είναι σε θέση να προβλέψει το αποτέλεσμα με μέσο όρο σφάλματος περίπου 5%. Το περιβάλλον το οποίο αναπτύχθηκε ήταν ένα Virtual Resource Allocation περιβάλλον (VRA) και οι εφαρμογές που εξετάστηκαν στις δύο VMs ήταν η συμπίεση των δεδομένων, η σύνταξη του πηγαίου κώδικα και η απόδοση του πλαισίου. Το αποτέλεσμα από την έρευνα δείχνει ότι η συνδυασμένη απόδοση διαφέρει σημαντικά με διαφορετικούς συνδυασμούς εφαρμογών. Οι εφαρμογές που σπανίως αλληλοεπιδρούν μεταξύ τους επιτυγχάνουν την ίδια απόδοση με εκείνη που θα είχαν εάν εκτελούνταν αυτόνομα. Ωστόσο, ορισμένοι συνδυασμοί αλληλοεπιδρούν μεταξύ τους με δυσμενή τρόπο. Επιπλέον τα αποτελέσματα των εφαρμογών επηρεάζονται από τα διαφορετικά φορτία αλλά και από τις εφαρμογές που εκτελούνται στο παρασκήνιο. Τέλος, από τις τα αποτελέσματα σχετικά με τις παρεμβολές στην απόδοση αλλά και τα χαρακτηριστικά των φορτίων που χρησιμοποιούνται δημιουργήθηκε μια εφαρμογή ομαδοποίησης. Οι

ερευνητές έτρεξαν έναν ιεραρχικό αλγόριθμο ομαδοποίησης ο οποίος χρησιμοποιεί ένα διάνυσμα με τη βαθμολογία της απόδοσης της κάθε εφαρμογής, το οποίο αποτελείται συγκεκριμένα από την κανονικοποιημένη βαθμολογία των επιδόσεων μιας εφαρμογής έναντι όλων εφαρμογών. Οι εφαρμογές συσταδοποίησης είναι χρήσιμες για την πρόβλεψη της απόδοσης της νέας εφαρμογής.

- Η εικονικοποίηση είναι μια τεχνολογία που χρησιμοποιείται σε όλα τα κέντρα δεδομένων του Νέφους για να εξασφαλιστεί η υψηλή αξιοποίηση των πόρων του υλικού καθώς και η καλύτερη διαχείριση των εικονικών μηχανών. Σύμφωνα με τη μελέτη [45], παρά τα πλεονεκτήματα που παρέχονται από την εικονικοποίηση, δεν παρέχουν αποτελεσματική απομόνωση της απόδοσης. Παρ' όλο που ο hypervisor (γνωστός και ως την εικονική οθόνη της μηχανής) χωρίζει τους πόρους και τους διαθέτει σε διάφορες VMs, η συμπεριφορά μιας εικονικής μηχανής μπορεί να εξακολουθεί να επηρεάζει την απόδοση μιας άλλης εξαιτίας της κοινής χρήσης των πόρων του συστήματος. Επιπλέον, η απομόνωση που προκαλείται από την εικονικοποίηση μπορεί να περιορίζει την εφαρμογή η οποία εκτελείται σε ένα εικονικό περιβάλλον VM λόγω της διακύμανσης στην απόδοση. Συγκεκριμένα, εάν ένας χρήστης εκτελέσει την ίδια εικονική μηχανή στο ίδιο υλικό σε διαφορετικές χρονικές στιγμές, θα παρατηρήσει μεγάλη διαφορά στην απόδοση εξαιτίας των άλλων εικονικών μηχανών που εκτελούνται την εκάστοτε στιγμή σε εκείνο το φυσικό ξενιστή (host).

4 *Συγκριτική Αξιολόγηση του Υπολογιστικού Νέφους*

Στο συγκεκριμένο κεφάλαιο παρουσιάζεται η έννοια της συγκριτικής αξιολόγησης. Αρχικά γίνεται αναφορά στον ορισμό της, ενώ στη συνέχεια περιγράφονται οι φορείς που καθορίζουν τα πρότυπα της συγκριτικής αξιολόγησης και αφορούν στις μετρήσεις των τιμών και στην παρουσίαση των αποτελεσμάτων. Στη συνέχεια γίνεται αναφορά σε ένα σύνολο από τα σημαντικότερα πλαίσια, εργαλεία, εφαρμογές και σουίτες τα οποία έχουν προταθεί για τη μέτρηση και την κατάταξη των υπηρεσιών του Υπολογιστικού Νέφους. Το κεφάλαιο καταλήγει αναφέροντας τις βασικές αρχές που εφαρμόστηκαν στην παρούσα διατριβή κατά τη διαδικασία της συγκριτικής αξιολόγησης.

4.1 Ορισμός και απαιτήσεις της συγκριτικής αξιολόγησης

Η ευρεία υιοθέτηση της τεχνολογίας του Υπολογιστικού Νέφους δημιούργησε την ανάγκη για την εφαρμογή μιας πιο αφαιρετικής διαδικασίας που ονομάζεται συγκριτική αξιολόγηση του Νέφους (Cloud Benchmarking). Μέσω της διαδικασίας αυτής αξιολογείται η απόδοση των υποδομών και του λογισμικού του Νέφους και διευκολύνεται η λήψη αποφάσεων των χρηστών ώστε να επιλέξουν εκείνη που ταιριάζει καλύτερα στον τύπο της εφαρμογής τους. Η συγκριτική αξιολόγηση είναι η διαδικασία της αξιολόγησης των επιδόσεων ενός συστήματος χρησιμοποιώντας τυπικές, επαναλαμβανόμενες δοκιμές. Συνήθως η διαδικασία της συγκριτικής αξιολόγησης έχει ως στόχο να παρέχει συγκριτικά αποτελέσματα προκειμένου οι κάτοχοι μιας εφαρμογής να προβούν σε αποφάσεις σχετικά με το ποιο είναι το αποδοτικότερο

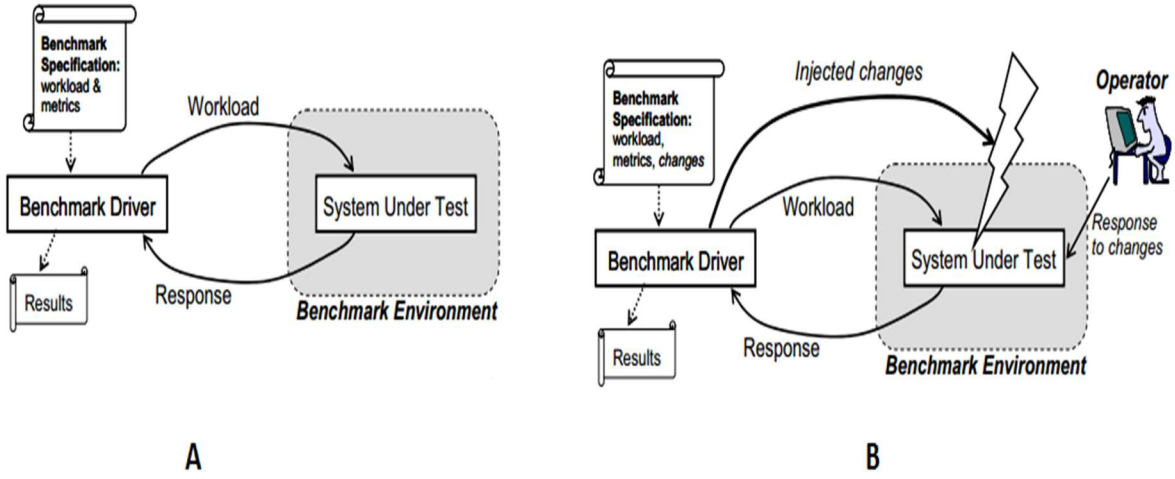
σύστημα βάσει των μετρήσεων.

Η συγκριτική αξιολόγηση είναι εξαιρετικά ενδιαφέρουσα για το Υπολογιστικό Νέφος επειδή πολλές εταιρείες επιθυμούν να μεταφέρουν τις εφαρμογές τους σε αυτό. Ωστόσο, εξαιτίας της αυξανόμενης προσφοράς των παρόχων η επιλογή της καλύτερης υπηρεσίας δεν είναι εύκολη υπόθεση. Γι' αυτό το λόγο η εφαρμογή της συγκριτικής αξιολόγησης στις διάφορες επιλογές του νέφους μπορεί να βοηθήσει στη σύγκριση διαφορετικών υπηρεσιών με ακρίβεια.

Ωστόσο, όταν η διαδικασία συγκριτικής αξιολόγησης εφαρμόζεται στο νέφος, τα κλασικά εργαλεία της συγκριτικής αξιολόγησης και μετρήσεις πρέπει να αναθεωρηθούν. Οι υπολογιστικοί πόροι του νέφους έχουν δύο αποκλειστικά χαρακτηριστικά - την ελαστικότητα και το μοντέλο pay-as-you-go τα οποία όταν συνδυαστούν δίνουν στον χρήστη την ψευδαίσθηση της διαθεσιμότητας άπειρων πόρων προς χρήση. Στο πλαίσιο αυτό, οι κλασικές μετρήσεις της συγκριτικής αξιολόγησης θεωρούνται ξεπερασμένες. Για παράδειγμα, ο αριθμός των συναλλαγών (transactions) που ολοκληρώνεται σε μια δεδομένη χρονική στιγμή (throughput) χάνει την αξία του εάν ένας πόρος στο νέφος μπορεί μέσω της ελαστικής κλιμάκωσης (scale up) να εκπληρώσει οποιοδήποτε αριθμό των συναλλαγών. Από την άλλη πλευρά, οι νέες μετρήσεις, όπως η ικανότητα του νέφους να προσαρμόζει αλλαγές στο φόρτο εργασίας (η ακρίβεια της ελαστικότητας) και το κόστος ανα συναλλαγή σχετίζονται. Η προσαρμογή νέων εργαλείων και μετρήσεων θεωρείται ως απαραίτητο βήμα το οποίο πρέπει να ακολουθηθεί για τη συγκριτική αξιολόγηση του νέφους [78] [85] [86] [87] [88].

Οι παραδοσιακές μέθοδοι της συγκριτικής αξιολόγησης [46] χρησιμοποιούν εργαλεία που παρέχουν μια μέθοδο σύγκρισης των διαφόρων υποσυστημάτων με διαφορετικές αρχιτεκτονικές απαντώντας στο ερώτημα για το ποιο είναι το καλύτερο σε ένα δεδομένο τομέα. Τα περισσότερα από αυτά τα tests απαιτούν το υπό δοκιμή σύστημα να έχει αναπτυχθεί σε ένα διαχειριζόμενο περιβάλλον, χρησιμοποιώντας μια σταθερή ρύθμιση παραμέτρων. Αυτό

σημαίνει ότι τα αποτελέσματα που προκύπτουν από τα tests της συγκριτικής αξιολόγησης πρέπει να αντικατοπτρίζουν τη μέση απόδοση ενός στατικού μη μεταβαλλόμενου συστήματος. Επιπλέον, κάθε ένα από αυτά τα tests εφαρμόζει ένα αντιπροσωπευτικό σενάριο για το συγκεκριμένο τομέα λαμβάνοντας υπόψη τις ιδιότητες και τους περιορισμούς του συστήματος που θα εξεταστεί.



Σχήμα 10: Στο Α απεικονίζεται η παραδοσιακή συγκριτική αξιολόγηση της απόδοσης ενώ στο Β η συγκριτική αξιολόγηση για αυτόνομες δυνατότητες

Ένα κρίσιμο σημείο σχετικά με τη μέθοδο της συγκριτικής αξιολόγησης ενός ή περισσότερων στοιχείων (components) μιας πολυ-επίπεδης εφαρμογή είναι το Σύστημα υπό Δοκιμή (SUT) [47], το οποίο περιλαμβάνεται στον ορισμό της συγκριτικής αξιολόγησης. Το SUT περιλαμβάνει components των οποίων η επίδοση πρέπει να μετρηθεί αποκλείοντας εξωτερικά συστήματα από τα οποία η εφαρμογή εξαρτάται και τα οποία δεν αποτελούν μέρος της αξιολόγησης της επίδοσης. Ωστόσο, τα tests μετρούν την συνολική απόδοση για όλα τα components που εμπλέκονται.

Αυτή η σελίδα είναι σκόπιμα λευκή

Μέσω της συγκριτικής αξιολόγησης στο Υπολογιστικό Νέφος, εννοούμε τη δοκιμαστική διαδικασία των υπηρεσιών του Νέφους που παρέχονται από διαφορετικούς παρόχους και κατά την οποία το σύστημα υπό δοκιμή (SUT) περιλαμβάνει μια υπηρεσία Νέφους ως component. Η κύρια διαφορά μεταξύ του παραδοσιακού και του τρόπου συγκριτικής αξιολόγησης στο Νέφος είναι ότι για το τελευταίο χρειαζόμαστε διαφορετικούς τρόπους για τη μέτρηση της απόδοσης και του κόστους λόγω της επεκτασιμότητας των συστημάτων καθώς οι πόροι έρχονται και παρέρχονται. Επιπλέον, η συγκριτική αξιολόγηση στο Νέφος θα πρέπει να ελέγχει τα ειδικά χαρακτηριστικά του Νέφους (επεκτασιμότητα, pay-per-use χρήση και ανοχή σε σφάλματα) και να παρέχει κατάλληλες μετρήσεις για αυτούς. Η βασική πρόκληση [48] των νέων τρόπων συγκριτικής αξιολόγησης είναι να κάνουν τα αποτελέσματα των δοκιμών συγκρίσιμα, μιας και διαφορετικοί πάροχοι προσφέρουν διαφορετικές υπηρεσίες με διαφορετικές δυνατότητες και εγγυήσεις των υπηρεσιών αυτών.

Εξαιτίας της δυναμικότητας στη διαχείριση των πόρων, η διαδικασία της συγκριτικής αξιολόγησης πρέπει να επαναλαμβάνεται με την πάροδο του χρόνου, έτσι ώστε να μπορούμε να εξασφαλίσουμε όσο είναι δυνατόν ότι το διαφορετικό υλικό, και οι διαφορετικές αποφάσεις στη διαχείριση (όπως π.χ. ενημέρωση / αναδιάρθρωση / βελτίωση των υποδομών) περιλαμβάνονται στις ανανεωμένες μετρικές τιμές, αλλά και χαρακτηριστικά όπως η διακύμανση των επιδόσεων, η τυπική απόκλιση κλπ.

4.2 Φορείς που καθορίζουν τα πρότυπα της συγκριτικής αξιολόγησης

Cloud Commons

Το Cloud Service Measurement Index Consortium (CSMIC) [49] έχει προσδιορίσει μετρικές που συνδυάζονται σε μια μορφή ενός Δείκτη Μέτρησης Υπηρεσίας (SMI), προσφέροντας τη

συγκριτική αξιολόγηση των υπηρεσιών του Νέφους. Αυτοί οι δείκτες μέτρησης μπορεί να χρησιμοποιηθούν από τους πελάτες ώστε να συγκρίνουν τις διάφορες υπηρεσιών του Νέφους.

SPEC (System Performance Evaluation Cooperative)

Η SPEC [50] είναι μία μη κερδοσκοπική εταιρεία που δημιουργήθηκε για να καθιερώσει, να διατηρήσει και να εγκρίνει ένα τυποποιημένο σύνολο από tests τα οποία θα χρησιμοποιηθούν στη μέθοδο της συγκριτικής αξιολόγησης και τα οποία μπορούν να εφαρμοστούν στη νεότερη γενιά υπολογιστών υψηλής απόδοσης. Το Open Systems Group (OSG) της SPEC έχει σχηματίσει μία νέα ομάδα, η οποία ονομάζεται Cloud Group, σκοπός της οποίας είναι να συνεργαστεί με άλλες ομάδες της SPEC για να ορίσουν μεθοδολογίες της συγκριτικής αξιολόγησης, να καθορίσουν και να προτείνουν αντιπροσωπευτικά φορτία εφαρμογών, να προσδιορίσουν νέες μετρήσεις για το Νέφος για τα υφιστάμενα SPEC tests αλλά και να αναπτύξουν νέα tests συγκριτικής αξιολόγησης αναπτυχθούν νέα πρότυπα σύννεφο. Η ομάδα εργασίας OSG Cloud ιδρύθηκε με κύριο στόχο να ερευνήσει και να συστήσει φορτία εργασίας για το Υπολογιστικό Νέφος. Ο κύριος στόχος της είναι να παρέχει αντιπροσωπευτικά σενάρια για τις εφαρμογές, που να ορίζονται σε ένα υψηλότερο επίπεδο αφαίρεσης. Αυτό μπορεί να χρησιμοποιηθεί ως βάση για την αξιολόγηση προτύπων και ερευνητικών αποτελεσμάτων καθώς και στην πλήρη άνθηση διαφορετικών πλατφόρμων Νέφους.

Το Working Group έχει εντοπίσει τρεις κατηγορίες που σχετίζονται με τα αποτελέσματα της συγκριτικής αξιολόγησης στο Νέφος. Οι κατηγορίες αυτές είναι: Hardware / Software-προμηθευτές, πάροχοι του Νέφους και οι τελικοί χρήστες. Αυτές οι τρεις κατηγορίες σχηματίζουν δύο διακριτές σχέσεις που καθορίζουν δύο τύπους tests συγκριτικής αξιολόγησης: τον Black Box και τον White Box.

The Perfect Club

Μια κοινοπραξία από προμηθευτές και πανεπιστημιακούς οι οποίοι καθορίζουν τα tests συγκριτικής αξιολόγησης για τον επιστημονικό τομέα, με ιδιαίτερη έμφαση στις παράλληλες υπολογιστικές αρχιτεκτονικές.

TPC (Transaction Processing Performance Council)

Μια κοινοπραξία προμηθευτών για τον καθορισμό tests συγκριτικής αξιολόγησης αναφοράς για τους τομείς των βάσεων δεδομένων και την εκτέλεση συναλλαγών.

4.3 Πλαίσια Συγκριτικής Αξιολόγησης των παρόχων του Νέφους

Υπάρχουν διάφορα πλαίσια τα οποία έχουν προταθεί για τη μέτρηση της απόδοσης των Νεφών και την κατάταξη των υπηρεσιών Νέφους. Το YCSB (Yahoo! Cloud Serving Benchmark) [72] και το AppScale [52] είναι πλαίσια που εστιάζουν στις συγκρίσεις των επιδόσεων των κατανεμημένων υπηρεσιών για την αποθήκευση δεδομένων, όπως η Cassandra, η MySQL, η MongoDB κλπ.

Πλαίσιο YCSB

Στόχος του συγκεκριμένου πλαισίου αξιολόγησης είναι να βοηθήσει στην αξιολόγηση διαφορετικών συστημάτων του Νέφους. Κυρίως επικεντρώνεται σε συστήματα που παρέχουν απευθείας read/write πρόσβαση στα δεδομένα, όπως αυτό συμβαίνει όταν ένας χρήστης του ιστού είναι σε αναμονή για την εμφάνιση μιας ιστοσελίδας όπου διεξάγονται reads και writes στη βάση δεδομένων ώστε να δημιουργηθεί και να παραδοθεί στο χρήστη η συγκεκριμένη ιστοσελίδα. Το YCSB παρέχει ένα στρώμα διασύνδεσης βάσης δεδομένων που μεταφράζει τις απλές αιτήσεις σε κλήσεις της βάσης δεδομένων (όπως τις thrift κλήσεις στη Cassandra ή rest αιτήσεις στο PNUTS). Εστιάζει στο επίπεδο πρόσβασης στις συναλλαγές και εκτός από τις

μετρήσεις όπως είναι ο ρυθμός απόδοσης και ο χρόνος καθυστέρησης για την ανάγνωση από τη βάση δεδομένων μετρά την επεκτασιμότητα και ελαστική επιτάχυνση.

Το συγκεκριμένο πλαίσιο δημιουργήθηκε από τη Yahoo! και αποτελείται από δύο τμήματα το YCSB Client και το σύνολο των βασικών φορτίων. Το YCSB Client είναι υλοποιημένο σε Java και χρησιμοποιείται για τη δημιουργία των φορτίων. Η αρχιτεκτονική του Client παρουσιάζεται στην εικόνα. Η βασική λειτουργία είναι ότι ο εκτελεστής φόρτου μπορεί να οδηγεί πολλαπλά νήματα πελάτη. Κάθε νήμα με τη σειρά του εκτελεί μία διαδοχική σειρά ενεργειών, καλώντας τη διεπαφή της βάσης δεδομένων τόσο για να «φορτώσει» τη βάση δεδομένων (φάση φόρτου) όσο και για να εκτελέσει το φορτίο εργασίας (φάση συναλλαγής). Τα νήματα περιορίζουν το ρυθμό δημιουργίας των αιτημάτων, ώστε να υπάρχει άμεσος έλεγχος του φόρτου εναντίον της βάσης. Η καθυστέρηση και η απόδοση των διεργασιών των νημάτων, παρατείθενται στις μετρήσεις στην ενότητα των στατιστικών στοιχείων, ενώ στο τέλος του πειράματος, προστείθενται ο μέσος όρος, το 95ο και 99ο εκατοστημόριο, είτε ένα ιστόγραμμα ή χρονοσειρές των χρόνων καθυστέρησης. Τα βασικά φορτία του YCSB είναι τα εξής:

• Workload A-Update heavy workload: Το φορτίο αυτό είναι μία μίξη με 50/50 αιτήματα εγγραφής και ανάγνωσης. Ένα παράδειγμα εφαρμογής είναι η αποθήκευση πρόσφατων ενεργειών ηχογράφησης.

• Workload B-Read mostly workload: Το φορτίο αυτό περιέχει μία μίξη από 95/5 αιτήματα ανάγνωσης και εγγραφής.

• Workload C-Read only: Το φορτίο αυτό αποτελείται από 100% αιτήματα ανάγνωσης. Ένα παράδειγμα εφαρμογής είναι το προφίλ χρήστη της κρυφής μνήμης όπου τα προφίλ κατασκευάζονται αλλού(π.χ Hadoop).

• Workload D-Read latest workload: Σε αυτό το φορτίο, εισάγονται νέες εγγραφές όπου οι πιο πρόσφατες είναι και οι πιο δημοφιλείς.

• Workload E-Short ranges: Σε αυτό το φορτίο υποβάλλονται ερωτήματα σε μικρό εύρος εγγράφων αντί μεμονωμένων εγγράφων. Ένα παράδειγμα εφαρμογής είναι οι συζητήσεις που χρησιμοποιούν νήματα.

• Workload F-Read-modify-write: Σε αυτό το φορτίο ο χρήστης διαβάζει ένα αρχείο, το τροποποιεί και ξαναγράφει πίσω τις αλλαγές. Μια τέτοια εφαρμογή είναι η βάση δεδομένων όπου ο χρήστης διαβάζει και τροποποιεί ο ίδιος τα αρχεία του.

AppScale

Το AppScale χρησιμοποιεί τη DataStore διεπαφή από το Google AppEngine ως μια καθολική διασύνδεση διαφορετικών κατανεμημένων τεχνολογιών ανοιχτού κώδικα των βάσεων δεδομένων, επιτρέποντας έτσι διάφορες εφαρμογές που έχουν υλοποιηθεί για το Google AppEngine να ελεγχθούν με αυτό το πλαίσιο, χωρίς να υπάρξει οποιαδήποτε τροποποίηση. Εστιάζει στην καταγραφή του χρόνου απόδοσης κυρίως εφαρμογών του ιστού και σε αντίθεση με το YCSB, που επικεντρώνεται στο επίπεδο των queries, αυτοματοποιεί τη ρύθμιση των παραμέτρων και την ανάπτυξη των βάσεων δεδομένων.

Εργαλεία μέτρησης της απόδοσης

Τα OpenBenchmarking.org [53], CloudHarmony [53] και CloudSleuth [54] είναι εργαλεία για τη μέτρηση της απόδοσης που αρχειοθετούν τα αποτελέσματα των tests και τα διαθέτουν μέσω του διαδικτύου. Το OpenBenchmarking.org είναι μια ολοκληρωμένη πλατφόρμα με test συγκριτικής αξιολόγησης. Παρέχει ένα μεγάλο σύνολο από test και αρχειοθετημένα αποτελέσματα των tests αυτών που έχουν εφαρμοστεί σε διάφορα hardware. Η απόδοση των πόρων του υλικού για διάφορα hardware μπορεί να ζητηθεί σε απευθείας σύνδεση από το αρχείο. Το CloudHarmony παρέχει μια παρόμοια λύση συγκριτικής αξιολόγησης, αλλά

επικεντρώνεται κυρίως στο Νέφος, και παρέχει διάφορες μετρήσεις απόδοσης με έμφαση στην εφαρμογή, τη CPU, τα I/O στο δίσκο κλπ. για διάφορους παρόχους του Νέφους σε απευθείας σύνδεση.

Τα αποτελέσματα από διάφορες εκτελέσεις είναι αρχειοθετημένα και είναι διαθέσιμα για πρόσβαση μέσω του ιστού. Το CloudSleuth παρέχει μετρήσεις όπως η διαθεσιμότητα , ο χρόνος απόκρισης των διαφόρων παρόχων του Νέφους σε απευθείας σύνδεση με τη συνεχή παρακολούθηση μιας εφαρμογής που εκτελείται στους παρόχους του Υπολογιστικού Νέφους.

CloudStone

Το CloudStone ορίζει μία μέθοδος συγκριτικής αξιολόγησης που μετρά την απόδοση των Web 2.0 εφαρμογών στο Νέφος. Αποτελείται από 3 components – το Olio, που είναι εφαρμογή ημερολογίου, το Faban το οποίο παράγει το φορτίο εργασίας και τα εργαλεία για τη διαχείριση και τη μέτρηση. Το Olio είναι μια εφαρμογή ημερολογίου με κοινωνικές εκδηλώσεις, η οποία μπορεί να αναπτυχθεί στο σύστημα που εκτελείται στο Νέφος και επρόκειτο να περάσει τη διαδικασία της συγκριτικής αξιολόγησης. Το Faban είναι μία γεννήτρια παραγωγής φόρτου εργασίας που εκτελείται στους πελάτες και προσομοιώνει μεγάλο αριθμό χρηστών που ταυτόχρονα χρησιμοποιούν το Olio. Τέλος τα εργαλεία διαχείρισης, χρησιμοποιούνται για την ανάπτυξη του Olio και για τη μέτρηση της απόδοση του συστήματος στο Νέφος. Το CloudStone θα μπορούσε να χρησιμοποιηθεί στο να βρει τον καλύτερο σχεδιασμό συστήματος για τη βελτιστοποίηση της αρχιτεκτονικής μια συγκεκριμένης εφαρμογής.

Το CloudCmp [55] παρέχει μια μεθοδολογία που έχει ως στόχο την εκτίμηση της απόδοσης και του κόστους μιας υπάρχουσας εφαρμογής η οποία αναπτύσσεται σε έναν πάροχο στο Νέφος. Ένας πιθανός πελάτης μπορεί να χρησιμοποιήσει τα αποτελέσματα για να συγκρίνει διάφορους παρόχους και να αποφασίσει αν θα πρέπει να μετακινήσει την εφαρμογή του στο Νέφος καθώς και ποιος πάροχος είναι ο καταλληλότερος για να φιλοξενήσει την εφαρμογή

του. Το CloudCmp εντοπίζει κοινές υπηρεσίες για διάφορους παρόχους Νέφους, και στη συνέχεια, για κάθε υπηρεσία προσδιορίζει ένα σύνολο μετρήσεων που έχουν σχέση με την απόδοση και το κόστος των εφαρμογών και αναπτύσσει ένα test συγκριτικής αξιολόγησης για κάθε μέτρηση το οποίο εκτελείται σε διαφορετικούς παρόχους και στη συνέχεια συγκρίνονται τα αποτελέσματα των εκτελέσεων αυτών.

Πλαίσιο Skymark

Το Skymark είναι ένα πλαίσιο το οποίο έχει σχεδιαστεί για να αναλύσει την απόδοση των υποδομών ως υπηρεσία (IaaS). Το πλαίσιο αποτελείται από δύο components - το Grenchmark και το C-Meter. Το Grenchmark είναι υπεύθυνο για την παραγωγή του φόρτου εργασίας, ενώ το C-Meter αποτελείται από ένα χρονοδιάγραμμα εργασίας και υποβάλλει την εργασία σε έναν διαχειριστή Νέφους ο οποίος διαχειρίζεται τις υποδομές ως υπηρεσία με μία συνδεόμενη (pluggable) αρχιτεκτονική. Το Skymark [56] εστιάζει στις παραμέτρους απόδοσης χαμηλού επιπέδου των υπηρεσιών του Νέφους όπως η CPU, μνήμη, κλπ, χωρίς να περιλαμβάνει τις μετρήσεις απόδοσης άλλων μοντέλων του Νέφους όπως το SaaS και το PaaS.

Πλαίσιο SMICloud

Το πλαίσιο SMICloud [57] παρέχει ένα μηχανισμό που μετρά την ποιότητα και την ιεράρχηση των υπηρεσιών του Νέφους. Ορίζει ένα πλαίσιο το οποίο αποτελείται από τρία στοιχεία - την Υπηρεσία Καταλόγου, Παρακολούθησης, και τον Service Measurement Index (SMI) διαμεσολαβητή του Νέφους. Το πρώτο στοιχείο αποθηκεύει τις υπηρεσίες και τα χαρακτηριστικά όπως διαφημίζονται από τους παρόχους του Νέφους. Η υπηρεσία παρακολούθησης ανακαλύπτει τις υπηρεσίες του Νέφους και παρακολουθεί την απόδοση τους, εποπτεύοντας πώς οι SLA απαιτήσεις των πελατών ικανοποιούνται από τον πάροχο. Ο SMI διαμεσολαβητής συλλέγει τις απαιτήσεις της εφαρμογής του πελάτη και κατατάσσει τις κατάλληλες υπηρεσίες. Το παραπάνω πλαίσιο παρουσιάζει ένα μοντέλο ποιότητας της

υπηρεσίας (QoS) για τους παρόχους IaaS που μπορεί να επεκταθεί σε SaaS και PaaS, προκειμένου να παρέχει ένα ολοκληρωμένο πλαίσιο για τις μετρήσεις των υπηρεσιών του Νέφους.

Cloud Rank D

Η Cloud Rank D [69], είναι η πρώτη σουίτα συγκριτικής αξιολόγησης για την μέτρηση της απόδοσης του Νέφους σε επίπεδο ολόκληρου του συστήματος, για τις μεγάλες εφαρμογές δεδομένων. Υπάρχουν τρεις τρόποι που χρησιμοποιείται η συγκεκριμένη σουίτα. Ο πρώτος είναι ότι ο χρήστης μπορεί να πραγματοποιήσει ποσοτική μέτρηση διαφορετικών συστημάτων του Νέφους και συγκεκριμένα να μετρήσει πόσο ένα σύστημα υπερτερεί σε σχέση με ένα άλλο. Ο δεύτερος είναι ότι το Cloud Rank-D μπορεί να καθοδηγήσει τη βελτιστοποίηση του συστήματος υπό δοκιμή και ο τρίτος είναι να μπορούμε να κατατάξουμε διαφορετικά συστήματα ανάλογα με τις λειτουργικές μετρήσεις που προέρχονται από το Cloud Rank-D. Το Cloud Rank-D περιλαμβάνει μια σειρά από δεκατρία αντιπροσωπευτικά εργαλεία ανάλυσης δεδομένων. Επομένως, οι χρήστες, σύμφωνα με τα επιχειρησιακά τους προαπαιτούμενα, μπορούν να επιλέξουν μία από τις τέσσερις βασικές κατηγορίες συγκριτικής αξιολόγησης από το Cloud Rank D: του μετασχηματισμού, της συνάθροιση, της σύνοψης, της επέκτασης και του υβριδίου. Για τη δημιουργία μιας εφαρμογής συγκριτικής αξιολόγησης, χρησιμοποιήθηκε η μέθοδος της καθοδικής προσέγγισης. Η σουίτα περιλαμβάνει βασικές λειτουργίες για την ανάλυση δεδομένων, την ταξινόμηση, την ομαδοποίηση, τη συσταδοποίηση κτλ. Καθώς και μια πραγματική εφαρμογή που ονομάζεται ProfSearch. Τέλος περιλαμβάνει τους πιο δημοφιλείς αλγορίθμους εξόρυξης δεδομένων όπως naive Bayes SVM και ο k-means.

4.3.1 Συγκριτική Αξιολόγηση σε επίπεδο εφαρμογών (Application Benchmark)

Η **PARSEC** (Princeton Application Repository for Shared-Memory Computers) είναι μία σουίτα συγκριτικής αξιολόγησης για πολλαπλούς πυρήνες και τσιπ πολυεπεξεργαστών, που κυκλοφόρησε στις αρχές του 2008 και είναι υλοποιημένο σε C / C ++. Αποτελείται από εννιά εφαρμογές και τρεις πυρήνες που επιλέχθηκαν από ένα ευρύ φάσμα τομέων εφαρμογών. Συγκεκριμένα περιλαμβάνει αναδυόμενες εφαρμογές αναγνώρισης, εξόρυξης και σύνθεσης (RMS), καθώς και συστήματα εφαρμογών οι οποίες μιμούνται μεγάλης κλίμακας εμπορικά προγράμματα πολλαπλών νημάτων αλλά και πολλές εφαρμογές αιχμής από το Πανεπιστήμιο του Princeton και του Stanford. Το PARSEC μπορεί να διαχειριστεί τις πολυνηματικές εφαρμογές, τον ξαφνικό φόρτο εργασίας που μπορεί να προκύψει καθώς και την ποικιλομορφία των εφαρμογών [58].

Η **Rodinia** [59] είναι μια πολυ-πλατφόρμα συγκριτικής αξιολόγησης για την ετερογενή υπολογιστική και βασίζεται στην ταξινόμηση των Berkeley dwarfs. Η σουίτα αποτελείται από τέσσερις εφαρμογές και πέντε πυρήνες που στοχεύουν σε πλατφόρμες CPU και GPU πολλαπλών πυρήνων καθώς επίσης και σε θέματα που σχετίζονται με παράλληλες μορφές επικοινωνίας και τεχνικών συγχρονισμού. Προκειμένου να βοηθήσει τους ερευνητές στην επιστήμη των υπολογιστών να έχουν μια εικόνα σε αναδυόμενες πλατφόρμες hardware, υλοποιήθηκε τόσο για CPU όσο και για GPU πολυπύρηνους επεξεργαστές χρησιμοποιώντας τρία διαφορετικά μοντέλα παράλληλου προγραμματισμού το OpenMP, το CUDA και το OpenCL. Το OpenMP είναι ένα απλό, παραδοσιακό μοντέλο παράλληλου προγραμματισμού κοινής μνήμης, με ένα στερεό υπόβαθρο στην HPC κοινότητα το οποίο διαθέτει μια διεπαφή που υποστηρίζει τον προγραμματισμό σε C, C ++, και Fortran σε πλατφόρμες πολυεπεξεργασίας κοινής μνήμης. Αποτελείται από ένα σύνολο οδηγιών (Compiler Directives) προς το μεταγλωττιστή που υποδεικνύουν τις περιοχές παράλληλης εκτέλεσης,

από συναρτήσεις Βιβλιοθήκης με την χρήση των οποίων διαμορφώνεται ανάλογα η παράλληλη εκτέλεση ή επιστρέφονται πληροφορίες σχετικές με τον τρόπο εκτέλεσης που έχει επιλεγεί και από τις μεταβλητές περιβάλλοντος που επηρεάζουν τη συμπεριφορά του χρόνου εκτέλεσης.

Η αρχιτεκτονική CUDA είναι ένα προγραμματιστικό μοντέλο το οποίο υποστηρίζει την εκτέλεση παράλληλων προγραμμάτων χρησιμοποιώντας τους πυρήνες μιας ή περισσότερων GPU. Κάθε ένας αναλαμβάνει να εκτελεί ακολουθίες εντολών που εκτελούνται πολλές φορές κατά τη διάρκεια ενός προγράμματος, με διαφορετικές ομάδες δεδομένων και, κατ' επέκταση, διαφορετικές ομάδες αποτελεσμάτων. Κάθε τέτοια ακολουθία εντολών αναφέρεται ως νήμα ή thread και ορίζεται ως η βασική μονάδα επεξεργασίας. Με τον τρόπο αυτό μπορούν να υλοποιηθούν αλγόριθμοι μαζικής παράλληλης επεξεργασίας. Τέλος το OpenCL είναι ένα άλλο δημοφιλές μοντέλο προγραμματισμού της εταιρείας Khronos, για ετερογενή παράλληλη υπολογιστική. Παρέχει ένα ανώτερο αφαιρετικό επίπεδο για ρουτίνες σε χαμηλό hardware επίπεδο, καθώς και μοντέλα για μαζική παράλληλη εκτέλεση κώδικα το οποίο αποτελείται από CPUs, GPUs και άλλους επεξεργαστές [60].

HPC εφαρμογές και συγκριτική αξιολόγηση

Το Υπολογιστικό Νέφος με την τεχνική της εικονικοποίησης λύνει θέματα που αφορούν στην εκτέλεση των High-performance computing εφαρμογών [60]. Αυτά τα είδη των εφαρμογών, συχνά παρουσιάζουν προβλήματα που σχετίζονται με την αύξηση της ανισορροπίας του φορτίου, την επεκτασιμότητα, τη διαχείριση των δεδομένων και την ασφάλεια.

Μερικά παραδείγματα από τις υπάρχουσες HPC εφαρμογές στο Νέφος σχετίζονται με τους εξής τομείς:

- Τομέας Υψηλών Τάσεων: οι περισσότερες από αυτές τις εφαρμογές είναι data-intensive (απαιτητικές από άποψη έντασης δεδομένων) όπως είναι οι εφαρμογές BaBar [66]

που κατέγραψε συγκρούσεις ηλεκτρονίων-ποζιτρονίων στο εργαστήριο SLAC National Accelerator Laboratory την περίοδο 2008-2009 καθώς και η DZero1 η οποία δημιουργεί πάνω από ένα TeraByte δεδομένων την ημέρα.

• Γεωγραφικός και Σεισμικός Τομέας: Οι εφαρμογές αυτές έχουν στόχο την αντιμετώπιση προβλημάτων που σχετίζονται με την απόδοση του Νέφους όπως είναι η διαχείριση μεγάλων συνόλων δεδομένων, η προσβασιμότητα και η επεκτασιμότητα. Οι περισσότερες από αυτές τις εφαρμογές είναι data-intensive και ευαίσθητες στη διαχείριση μνήμης.

• Τομέας Ηλεκτρονικών: Μία εφαρμογή του συγκεκριμένου τομέα είναι η Στατική Ανάλυση χρονισμού η οποία υπολογίζεται να χρησιμοποιήσει τις πηγές του Νέφους ώστε να ελαττώσει το κόστος και να επιταχύνει την παραγωγή.

• Μεγάλης κλίμακας μελέτες μηχανικής προσομοίωσης. Πολλές βιομηχανίες αυτοκινήτων και αεροσκαφών έχουν ήδη αρχίσει να χρησιμοποιούν το Νέφος. Πολλές μελέτες προσομοίωσης και προγνωστικής μοντελοποίησης πριν την παραγωγή επιλύονται με ένα κατανεμημένο τρόπο χρησιμοποιώντας το Νέφος όπως στην περίπτωση της IBM Engineering Solutions for Cloud.

Μέθοδος συγκριτικής αξιολόγησης για HPL εφαρμογές

Εκτός από τις πραγματικές HPL εφαρμογές που μπορούν να εκτελεστούν στο Νέφος κάποιοι ερευνητές μελέτησαν ζητήματα που αφορούν στην απόδοση του Νέφους χρησιμοποιώντας κάποια πρότυπα συγκριτικής αξιολόγησης. Κάποια από αυτά τα εργαλεία είναι το HPL (High Performance Linpack) [68], το Nas Parallel [61] και το NERSC. Το πρώτο είναι μια φορητή υλοποίηση για υπολογιστές κατανεμημένης μνήμης. Το Linpack είναι μία συλλογή από Fortran υπορουτίνες που αναλύουν και επιλύουν γραμμικές εξισώσεις και γραμμικά προβλήματα ελάχιστων τετραγώνων. Το Nas Parallel χρησιμοποιείται για να αξιολογηθεί η

απόδοση των παράλληλων υπολογιστών που προέρχονται από εφαρμογές υπολογιστικής ρευστοδυναμικής (CFD) και αποτελείται από πέντε πυρήνες και τρεις ψευδο-εφαρμογές. Η συγκεκριμένη σουίτα έχει επεκταθεί προκειμένου να παραχθούν νέα tests για αδόμητα adaptive mesh μοντέλα, για παράλληλες ενέργειες Εισόδου/Εξόδου, για εφαρμογές πολλαπλών ζωνών και υπολογιστικών πλεγμάτων. Τέλος το NERSC [69] είναι ένα πλαίσιο το οποίο περιέχει πραγματικές επιστημονικές εφαρμογές. Περιλαμβάνει ένα ευρύ φάσμα αριθμητικών μεθόδων και αναπαραστάσεις δομών δεδομένων στην περιοχή της επιστήμης των υλικών, της σύντηξης, της αστροφυσικής κτλ.

MATLAB σουίτα αξιολόγησης επιδόσεων

Η συγκεκριμένη σουίτα [62] αποτελείται από έξι tests που χρησιμοποιούνται τόσο για τον καθορισμό της υπολογιστικής ικανότητας του hardware (βαθμολογία του test) όσο και για τον χαρακτηρισμό των τύπων του φόρτου εργασίας (αριθμός test). Τα tests περιλαμβάνουν floating-point με κανονικές ή μη κανονικές προσβάσεις στη μνήμη, δομές δεδομένων, μικτούς ακέραιους και πράξεις κινητής υποδιαστολής καθώς και δισδιάστατα και τρισδιάστατα γραφικά. Μια έρευνα, η οποία βασίζεται στην σουίτα του MATLAB, παρουσιάζεται στο [41]. Η ανάλυση σχετίζεται με ένα σημαντικό αριθμό κρίσιμων παραμέτρων οι οποίες επιδρούν στην απόδοση των εικονικών μηχανών όπως τα ποσοστά κατανομής της CPU, ο προγραμματισμός των αποφάσεων σε πραγματικό χρόνο και όταν τρέχουν εφαρμογές στο ίδιο φυσικό κόμβο, και μοιράζονται τις υποδομές. Επιπλέον, παρουσιάζεται μία μέθοδος (black-box) που βασίζεται σε γενετικά βελτιστοποιημένα μοντέλα Τεχνητών Νευρωνικών Δικτύων (ΤΝΔ) τα οποία διαμορφώνουν και προβλέπουν την απόδοση μιας εφαρμογής.

Σουίτα Berkeley Dwarfs

Ένας dwarf [63] είναι μια αλγοριθμική μέθοδος που καταγράφει ένα υπολογιστικό και επικοινωνιακό πρότυπο. Οι πρώτοι επτά dwarfs που χρησιμοποιούνται για υπολογιστική

υψηλής απόδοσης (High Performance Computing) εμπνεύστηκαν από τον Phil Colellao ο οποίος εντόπισε επτά σημαντικές αριθμητικές μεθόδους για την επιστήμη και την τεχνολογία. Σε αντίθεση με τις παραδοσιακές αξιολογήσεις επιδόσεων οι dwarfs χρησιμοποιούνται για τον παράλληλο σχεδιασμό προγραμματισμού μοντέλων και αρχιτεκτονικών. Μερικά παραδείγματα των εξεταζόμενων εφαρμογών είναι τα πυκνά πλέγματα ή διανύσματα, εφαρμογές γραμμικής άλγεβρας, εξόρυξη δεδομένων και ομαδοποίησης, αραιής γραμμικής άλγεβρας (ανάλυση πεπερασμένων στοιχείων και μερικών διαφορικών εξισώσεων), φασματικές μέθοδοι (δυναμική ρευστών, η κβαντική μηχανική και την πρόβλεψη του καιρού), μεθόδους N-σώματος (μοριακή μοντελοποίηση, μοριακή δυναμική και την κοσμολογία), δομημένα δίκτυα με υψηλή τοποθεσίας (επεξεργασία χωρική εικόνα, όπως SRAD και προσομοιώσεις φυσικής, όπως Hotspot), μη κανονικά πλέγματα (πολλαπλασιασμό πίστη και δυναμική Computational Fluid), MapReduce (κατανεμημένη αναζήτηση, ακολουθιακή ευθυγράμμιση και παράλληλες προσομοιώσεις Monte Carlo) κτλ. Το κύριο πλεονέκτημα των dwarfs είναι ότι καλύπτουν ένα πολύ μεγάλο φάσμα κατηγοριών εφαρμογών και να δημιουργήσουν τα υπολογιστικά τους μοντέλα.

Filebench

Το Filebench [71] είναι ένα από τα πιο γνωστά και ευέλικτα εργαλεία που χρησιμοποιείται στο σύστημα αρχείων και στη συγκριτική αξιολόγηση της αποθήκευσης (storage). Είναι ένα πλαίσιο ανοιχτού κώδικα υλοποιημένο σε C που φιλοξενείται στο sourceforge.net [65] και χρησιμοποιεί ως φόρτο εργασίας προσωπικότητες ώστε να επιτρέπει την εύκολη εξομοίωση των σύνθετων εφαρμογών. Η δημοτικότητα του Filebench οφείλεται στο γεγονός ότι είναι διαθέσιμο με πολλές προκαθορισμένες μακρο-εργασίες φόρτου όπως ο Web-server, ο Mail-server, και ο File-Server. Αυτό, επιτρέπει στους χρήστες να ελέγχουν επαρκώς τα συστήματα αρχείων τους χρησιμοποιώντας πολλά και διαφορετικά φορτία εργασίας με ένα μόνο εργαλείο.

Ακόμη, επιτρέπει στους χρήστες να υλοποιήσουν νέα φορτία χρησιμοποιώντας τη γλώσσα Workload Model (WML). Τα σημαντικότερα workloads του Filebench παρουσιάζονται στον

Filebench	**Περιγραφή φορτίου εργασίας**
fileserver	Είναι ένα φορτίο εργασίας γα το σύστημα αρχείων παρόμοιο με το SPECsfs. Αυτό το φορτίο εργασίας εκτελεί μια ακολουθία από λειτουργίες δημιουργίας, διαγράφης, προσθήκης, διαβάσματος στο σύστημα αρχείων. Ένας διαμορφώσιμος και με ιεραρχική δομή κατάλογος χρησιμοποιείται για το σύνολο του αρχείου.
Varmail	Ένας / var / mail NFS διακομιστής αλληλογραφίας, όπως το Postmark αλλά με τη χρήση πολλών νημάτων (multi-threaded). Το φορτίο εργασίας αποτελείται από ένα multi-threaded σύνολο άνοιγμα/διάβασμα/κλείσιμο, άνοιγμα/προσαρτήση/κλείσιμο και διαγραφές σε ένα ενιαίο κατάλογο.
Videoserver	Μία μείξη από λειτουργίες άνοιγμα/διάβασμα/κλείσιμο σε πολλαπλά αρχεία σε ένα δέντρο καταλόγου καθώς και ένα αρχείο προσάρτησης (για την προσομοίωση των web logs). Χρησιμοποιούνται 100 νήματα καθώς 16k προστίθενται στο weblog κάθε 10 αναγνώσεις (reads).
Webproxy	Μία μείξη από λειτουργίες δημιουργία/εγγραφή/κλείσιμο, άνοιγμα/ανάγνωση/κλείσιμο, διαγραφή πολλαπλών αρχείων σε ένα δέντρο, και επιπλέον ένα αρχείο προσάρτησης (για την προσομείωση των proxy logs). Χρησιμοποιούνται 100 νήματα καθώς 16k προστίθενται στο proxy log για κάθε 10 αναγνώσεις/εγγραφές.

Πίνακας 4: Περιγραφή του φόρτου εργασίας του Filebench Benchmark

Στην περίπτωση ενός Web-server από την πλευρά του συστήματος αρχείων για κάθε HTTP αίτημα, ο Web-Server ανοίγει ένα ή περισσότερα αρχεία HTML, τα διαβάζει και επιστρέφει το περιεχόμενό τους στον πελάτη. Κάποιες φορές καταγράφει τα αρχεία πρόσβασης του πελάτη σε ένα αρχείο καταγραφής (log). Στην περίπτωση του Web-server του Filebench, το συγκεκριμένο φόρτο δημιουργήθηκε στηριζόμενο στην παραπάνω υπόθεση. Το κάθε νήμα

ανοίγει ένα αρχείο, το διαβάζει σε μια κλήση, και στη συνέχεια το αρχείο κλείνει. Κάθε δέκατη ανάγνωση, ο Web-Server του Filebench προσθέτει μια μικρή ποσότητα δεδομένων σε ένα αρχείο καταγραφής (log). Τα μεγέθη των αρχείων ακολουθούν την κατανομή γάμα, με μέσο μέγεθος αρχείου 16 KB. Από προεπιλογή, ο φόρτος εργασίας του Web-Server έχει ρυθμιστεί με 100 νήματα και 1.000 αρχεία.

Στην περίπτωση του File-Server από το Filebench το φορτίο είναι ναι μεν σχεδιασμένο απλά, αλλά έτσι ώστε να προσομειώνει πραγματικά ό,τι παράγει ο File-Server σε ένα σύστημα αρχείου. Σε αυτό το φόρτο πραγματοποιούνται πενήντα διαδικασίες οι οποίες αντιπροσωπεύουν 50 χρήστες. Ο καθένας από τους χρήστες αυτούς μπορεί να δημιουργεί και να γράφει σε ένα αρχείο, να ανοίγει ένα υπάρχον αρχείο και να προσθέτει σε αυτό. Ακόμη μπορεί να ανοίγει ένα άλλο αρχείο και να το διαβάζει αλλά και να διαγράφει ένα αρχείο και επικαλώντας μια stat λειτουργία. Τέτοιες μεικτές λειτουργίες είναι οι πιο συνηθισμένες λειτουργίες που αναμένει κανείς από ένα πραγματικό Fileserver. Το συγκεκριμένο φορτίο από προεπιλογή αποτελείται από 10.000 αρχεία μεγέθους 128 KB.

Το φορτίο Mail-server που ονομάζεται varmail.f αντιπροσωπεύει ένα φόρτο εργασίας που υφίσταται σε ένα /var/mail κατάλογο σε ένα παραδοσιακό σύστημα UNIX που χρησιμοποιεί τη μορφή Maildir (ένα μήνυμα ανά αρχείο). Όταν ο χρήστης λάβει ένα μήνυμα ηλεκτρονικού ταχυδρομείου, ένα αρχείο δημιουργείται, γράφεται και συγχρονίζεται. Στην περίπτωση που ο χρήστης διαβάζει ένα μήνυμα ηλεκτρονικού ταχυδρομείου, ένα άλλο αρχείο ανοίγει, το οποίο διαβάζεται και συγχρονίζεται. Μερικές φορές, οι χρήστες ξαναδιαβάζουν ηλεκτρονικά μηνύματα τα οποία έχουν ήδη διαβάσει. Το μέσο μέγεθος ενός ηλεκτρονικού μηνύματος ορίζεται ως 16 KB.

Το Filebench περιλαμβάνει αρκετά χαρακτηριστικά που διευκολύνουν την συγκριτική αξιολόγηση ενός συστήματος αρχείων κάποια από τα οποία είναι τα εξής:

• Χρησιμοποιεί προσωπικότητες πολλαπλών φορτίων (loadable workload personalities) που επιτρέπουν την εύκολη προσομείωση σύνθετων εφαρμογών.

• Παρέχει μία βιβλιοθήκη με περισσότερες από 40 προκαθορισμένες προσωπικότητες, μεταξύ των οποίων εκείνες που περιγράφουν την συμπεριφορά των mail, web, file και βάσης δεδομένων διακομιστών. Οι προσωπικότητες φορτίων καθορίζουν το φόρτο για την εφαρμογή του συστήματος και περιλαμβάνουν τις μεταβλητές για την κλιμάκωση του φόρτου σε συγκεκριμένα συστήματα.

• Παρέχει ευκολία στην πρόσθεση νέων προσωπικότητων χρησιμοποιώντας την Workload Model Language (WML) [72].

• Υποστηρίζει φορτία πολλαπλών διεργασιών και πολλαπλών νημάτων.

• Υποστηρίζει τις ασύγχρονες I/O ενέργειες και τους κανόνες συγχρονισμού των διαδικασιών.

• Παρέχει στατιστικά για την απόδοση, το συνολικό χρόνο καθυστέρησης, και τον κύκλο της CPU ανά κλήση συστήματος.

• Έχει δοκιμαστεί σε πλατφόρμες Linux, FreeBSD και Solaris.

DaCapo μέθοδος συγκριτικής αξιολόγησης

Η σουίτα DaCapo [76] έχει σχεδιαστεί για να διευκολύνει την ανάλυση των επιδόσεων των Java εικονικών μηχανών, των μεταγλωτιστών και τη διαχείριση μνήμης. Αποτελείται από ένα σύνολο client-side πραγματικών εφαρμογών ανοικτού κώδικα με μη-τετριμμένα φορτία μνήμης. Τα tests της συγκεκριμένης σουίτας είναι πιο σύνθετα από πλευράς στατικής και δυναμικής των μετρήσεων συγκριτικά με της SPEC [75]. Για παράδειγμα παρουσιάζουν πλουσιότερη πολυπλοκότητα στον κώδικα, στη δομή των κλάσεων αλλά και στην ιεραρχία των κλάσεων σε σχέση με τη SPEC όπως παρουσιάζεται στο [74]. Οι εφαρμογές της

συγκριτικής αξιολόγησης της DaCapo σουίτας παρουσιάζονται στον πίνακα: Οι εφαρμογές της συγκριτικής αξιολόγησης της DaCapo σουίτας παρουσιάζονται στον πίνακα:

DaCapo Applications	Description
xalan	Μετατρέπει XML έγγραφα σε HTML.
tomcat	Εκτελεί μία σειρά από queries σε έναν tomcat εξυπηρετητή ανακτώντας και επαληθεύοντας τις προκύπτουσες ιστοσελίδες.
pmd	Αναλύει ένα σύνολο Java κλάσεων για ένα εύρος πρβλημάτων πηγαίου κώδικα.
jython	Χρησιμοποιείται ως διερμηνέας του pybench Python
h2	Εκτελεί ένα JDBC test συγκριτικής αξιολόγησης χρησιμοποιώντας έναν αριθμό συναλλαγών έναντι ενός μοντέλου τραπεζικής εφαρμογής.
fop	Αναλύει και μορφοποιεί XSL-FO αρχεία και παράγει ένα pdf αρχείο.
eclipse	Εκτελεί jdt test επιδόσεων για το Eclipse IDE.
avrora	Προσομοιώνει μια σειρά από προγράμματα που εκτελούνται σε ένα πλέγμα από AVR micro-controllers.

Πίνακας 5: Οι benchmark εφαρμογές (φορτία εργασίας) της DaCapo Suite

Στον *Πίνακας 6* παρουσιάζονται συγκεντρωτικά οι σημαντικότερες μέθοδοι συγκριτικής αξιολόγησης και τα χαρακτηριστικά τους.

Μέθοδος Συγκριτικής Αξιολόγησης	Τύπος Εφαρμογής	Πόροι που εξετάζονται	Υλοποίηση	Τύπος Άδειας Χρήσης	Μετρικές

YCSB	OLTP(online transaction processing) εφαρμογές Νέφους	Συστήματα που παρέχουν απευθείας read/write πρόσβαση στα δεδομένα (χρόνος καθυστέρησης, επεκτασιμότητα)	Java	Ανοικτού κώδικα, επεκτάσιμο, εύκολος καθορισμός νέων φορτίων, ευκολία στην συγκριτική αξιολόγηση νέων συστημάτων	Online read/write πρόσβαση στα δεδομένα, καθυστέρηση των requests όταν η βάση δεδομένων παρουσιάζει υψηλό φόρτο εργασίας, επεκτασιμότητα-ελαστικότητα
AppScale	GAE εφαρμογές (Κατανεμημένες τεχνολογίες βάσεων δεδομένων, εφαρμογές ιστού)	Virtualized cluster resoures	Java	Ανοικτού κώδικα	Κλιμάκωση, fault tolerance
PARSEC	Μηχανική όραση, τεχνική φυσικού μοντέλου, future media, content based search, κατάργηση διπλότυπων δεδομένων, οικονομικά-πολυμέσα	Πολυπύρηνα Συστήματα	C/C++	Ανοικτού κώδικα, επεκτάσιμο	miss rates της κρυφής μνήμης κατά τη διάρκεια του φόρτου εργασίας και της αποθήκευσης

Rodinia	Dwarfs-επιστημονική-μηχανική εξόρυξη δεδομένων	Πολυπύρηνα συστήματα, GPU, εύρος ζώνης της μνήμης	OpenMP, OpenCL&CUBA	Ανοικτού κώδικα-επέκταση της Rodinia στο μέλλον να καλύψει και τα υπόλοιπα dwarfs	Παράλληλη επικοινωνία & πρότυπα πρόσβασης σε δεδομένα, χαρακτηριστικά δεδομένων κοινοχρησίας, κατανάλωση της ενέργειας
HPL	Η τελευταία έκδοση χρησιμοποιείται για να κατατάξει τους πιο ισχυρούς υπερυπολογιστές	Mflop/s (millions of floating point operations per second)	C	Ανοικτού κώδικα	Τη συχνότητα του μηχανήματος σε κύκλους/ δευτερόλεπτο, τον αριθμό των διεργασιών ανά κύκλο
NAS Parallel	Εφαρμογές πολλαπλών ζωνών, υπολογιστικά πλέγματα, παράλληλοι υπερ-υπολογιστές	CPU, GPU	MPI, OpenMP, Java	Ανοικτού κώδικα , επέκταση του Linux για να συμπεριλάβει νέες συγκριτικές μεθόδους αξιολόγησης	MOPS(Millions of Operations Per Second)- μέτρηση των kernel διεργασιών που διαφέρουν από τις CPU διεργασίες
Cloud Rank D	Εφαρμογές με πολλά δεδομένα	CPU	Hadoop (έκδοση 0.20.2), Hive(έκδοση 0.6.0) και Mahoot	Δεν είναι διαθέσιμο	Συνολική απόδοση του συστήματος με τη χρήστη δύο συμπληρωματικών μετρήσεων: δεδομένα που

			(έκδοση 0.6)		επεξεργάζονται ανά sec και ανά Joule (μνήμης, δίσκου και I/O δικτύου)
Berkeley Dwarfs	Αριθμητικές,2 D γραφικές, 3D απεικονιστικές εφαρμογές, HPDC (High Performance Distributed Computing) εφαρμογές	ταχύτητα CPU, μέγεθος της κρυφής μνήμης ή της RAM	Αλγοριθμικές μέθοδοι, 13 kernels υλοποιημένοι σε C++ ή C# και μία γεννήτρια δεδομένων (F#)	Ανοικτού κώδικα	Απόδοση του εικονικού hardware, συλλαμβάνει ένα πρότυπο υπολογισμού και επικοινωνίας
MATLAB	Πολυμέσα, επιστημονικές εφαρμογές	CPU, RAM	Matlab scripting	Ανοικτού κώδικα	Ποσοστά κατανομής της CPU, προγραμματισμός αποφάσεων πραγματικού χρόνου
CloudStone	Web 2.0 εφαρμογές	Web 2.0 applications	Java, php, ruby	Apache 2.0	Χρόνος απόκρισης(μέσος όρος, μέγιστος, 90 εκατοστιαία τιμή), χρήση δεδομένων

Πίνακας 6: Οι σημαντικότερες μέθοδοι της συγκριτικής αξιολόγησης και τα χαρακτηριστικά τους

4.4 Οι βασικές αρχές που υιοθετήθηκαν στη διαδικασία της Συγκριτικής Αξιολόγησης

Στο κεφάλαιο αυτό πραγματοποιήθηκε εκτενής αναφορά στα διαθέσιμα εργαλεία της συγκριτικής αξιολόγησης τα οποία μπορούν να εφαρμοστούν για τη μέτρηση της απόδοσης των υπηρεσιών του Νέφους. Ωστόσο, σε αυτό το σημείο θα πρέπει να αναφερθεί πως εκτός από την επιλογή των κατάλληλων benchmarks κρίσιμο συστατικό στοιχείο αποτελεί και ο τρόπος σύμφωνα με τον οποίο εφαρμόζεται η συγκεκριμένη διαδικασία.

Μία από τις βασικές πτυχές είναι ότι λόγω της δυναμικότητας στη διαχείριση των πόρων η διαδικασία της συγκριτικής αξιολόγησης να πρέπει να επαναλαμβάνεται στο χρόνο ώστε να μπορούμε να διασφαλίσουμε την καταγραφή όσο το δυνατόν του διαφορετικό hardware, των αποφάσεων της διοίκησης (π.χ., ενημέρωση / αναδιάρθρωση / βελτίωση των υποδομών) οι οποίες θα αποτυπώνονται στις ανανεωμένες τιμές που θα προκύπτουν αλλά και να μελετηθούν σημαντικές χαρακτηριστικές μετρήσεις όπως η διακύμανση της επίδοσης κτλ.

Ακόμη οποιαδήποτε διαδικασία μέτρησης πρέπει να βασίζεται σε ένα μινιμαλιστικό σύνολο γενικών εφαρμογών συγκριτικής αξιολόγησης τα οποία είναι ενδεικτικά και αντικατοπτρίζουν εφαρμογές του πραγματικού κόσμου. Με αυτό τον τρόπο επιτυγχάνεται η αφαιρετικότητα των αποτελεσμάτων της συγκριτικής αξιολόγησης σε ένα υψηλότερο επίπεδο γνώσης (σε επίπεδο εφαρμογής) και όχι βάσει των τιμών των μετρήσεων σε χαμηλό επίπεδο (π.χ. MB / sec), με αποτέλεσμα να γίνεται ευρύτερα κατανοητή από τον ιδιοκτήτη της εφαρμογής.

Γι’ αυτό λοιπόν στην προσέγγισή μας, η διαδικασία συγκριτικής αξιολόγησής περιλαμβάνει διάφορες κοινές εφαρμογές για τις οποίες έχουν εντοπιστεί αντιπροσωπευτικά σενάρια benchmark εφαρμογών τα οποία μπορούν να αποτυπώσουν τις δυνατότητες των υπηρεσιών. Η κύρια πτυχή του ενδιαφέροντος για την επιλογή των σημείων αναφοράς είναι η δυνατότητα

να έχουν φόρτο εργασίας σε επίπεδο εφαρμογής. Ο βασικός στόχος αυτής της διαδικασίας είναι η αφαιρετικότητα της απόδοσης των υπηρεσιών του Νέφους σε κατάλληλο βαθμό που να μπορεί να είναι κατανοητή και να χρησιμοποιείται από την πλειοψηφία των ατόμων που δεν έχουν εξιδεικευμένη γνώση της απόδοσης.

Σε αυτό το σημείο σημαντική κρίνεται εκτός από τη διακύμανση της απόδοσης και η ενσωμάτωση του κόστους της υπηρεσίας ώστε να μπορούν να απαντηθούν ερωτήματα όπως «ποιος είναι ο καλύτερος τύπος υπηρεσίας / πάροχος που θα μπορούσε να φιλοξενήσει μία εφαρμογή βάσης δεδομένων στην περίπτωση που ο ιδιοκτήτης ενδιαφέρεται περισσότερο για μια φθηνή λύση».

Τέλος η διαδικασία της συγκριτικής αξιολόγησης θα πρέπει να περιλαμβάνει την εξέταση ενός σημαντικού αριθμού παρόχων και τύπων υπηρεσιών όπου οι μετρήσεις να πραγματοποιούνται μέσω ενός αυτοματοποιημένου μηχανισμού ο οποίος θα περιλαμβάνει, την εγκατάσταση την εκτέλεση και την αποθήκευση των αποτελεσμάτων.

5 *Δυναμική Αξιολόγηση των Υπηρεσιών του Νέφους βάσει της ανάλυσης της απόδοσης της εφαρμογής*

Η συνδυαστική μεθοδολογία αξιολόγησης που παρουσιάζεται στο παρόν κεφάλαιο εκμεταλλεύεται τη συγκριτική αξιολόγηση καθώς και τον τρόπο που οι εφαρμογές χρησιμοποιούν τους υποκείμενους υπολογιστικούς πόρους ώστε να προταθεί στο χρήστη της εφαρμογής η βέλτιστη λύση Υπηρεσίας Νέφους, όσον αφορά την απόδοση και την τιμολόγηση, κατά τη μεταφορά της εφαρμογής στο Νέφος. Κίνητρο για τη μεθοδολογία αυτή είναι είτε το γεγονός ότι υπάρχουν πολλές περιπτώσεις όπου ο ιδιοκτήτης μιας εφαρμογής δεν γνωρίζει πώς τα components της εφαρμογής συμπεριφέρονται κατά τη διάρκεια εκτέλεσης τους, είτε δεν είναι γνωστός ο τρόπος χρήσης των πόρων. Ακόμη, σε κάποιες περιπτώσεις εξαιτίας των δομικών αλλαγών κατά τη διαδικασία της μετανάστευσης του λογισμικού στο Νέφος, ίσως λανθασμένα να θεωρείται ή να έχει τροποποιηθεί ο τρόπος που αυτό χρησιμοποιεί τους πόρους. Από την άλλη πλευρά, οι πάροχοι του υπολογιστικού νέφους, μπορεί επίσης να ενδιαφέρονται να μάθουν τους τύπους των εφαρμογών που φιλοξενούνται στις υποδομές τους, ώστε να αποφεύγονται οι ανεπιθύμητες παρεμβολές που μπορεί να οφείλονται στην ταυτόχρονη εκτέλεση των εικονικών μηχανών (VMs), που υποβαθμίζουν σημαντικά την απόδοση των εφαρμογών.

Η αξιολόγηση ενός συγκεκριμένου και αυθαίρετου δομικού στοιχείου μιας εφαρμογής σε όλο το εύρος των προσφορών θεωρείται δύσκολο έργο, ειδικά όταν η ανάπτυξη της εφαρμογής μπορεί να εξαρτάται από ειδικές ενέργειες του παρόχου ή από συγκεκριμένες ενέργειες του δομικού στοιχείου της εφαρμογής ή ακόμη και από τον ίδιο τον κώδικα. Ωστόσο, η εξεύρεση ενός πιο αφηρημένου και κοινού τρόπου για τον εντοπισμό τόσο του προφίλ της εφαρμογής (ο τρόπος που χρησιμοποιεί τους υποκείμενους υπολογιστικούς πόρους) όσο και των χαρακτηριστικών της απόδοσης των Υπολογιστικών Νεφών μπορεί να μειώσει σημαντικά την προσπάθεια που απαιτεί αυτή η διαδικασία.

Επομένως, βάσει των παραπάνω είναι αναγκαία η δημιουργία μιας μεθοδολογίας και η υλοποίηση ενός μηχανισμού για τον εντοπισμό της υπολογιστικής φύσης ενός λογισμικού δομικού στοιχείου μιας εφαρμογής αλλά και η ταξινόμησή του σε μια λίστα γνωστών υπολογιστικών προτύπων που ορίζονται μέσα από τα εντοπισμένα στερεότυπα της απόδοσης τα οποία έχουν σα σκοπό να απλοποιήσουν τη διαδικασία της αξιολόγησης της επίδοσης των Υπολογιστικών Νεφών ώστε να υποστηρίξουν το χρήστη στη διαδικασία λήψης αποφάσεων.

5.1 Σχετικές Εργασίες

Το Υπολογιστικό Νέφος έχει επιστήσει ιδιαίτερη προσοχή από τους ερευνητές τα τελευταία χρόνια. Στο [22] παρέχεται μια περιεκτική επισκόπηση του Υπολογιστικού Νέφους, συμπεριλαμβανομένων των νέων ευκαιριών και των δυνατότητων που παρέχει, τα πιθανά εμπόδια, καθώς και μια ταξινόμηση των παρόχων του Νέφους. Η συγκεκριμένη έρευνα αποτελεί κίνητρο για τη μελέτη και τη σύγκριση των παρόχων του Νέφους στη σημερινή αγορά. Ο Wang και Ng δείχνουν ότι η τεχνική της εικονικοποίησης στην Amazon EC2 μπορεί να οδηγήσει σε δραματική αστάθεια στην απόδοση του δικτύου και στην καθυστέρηση, ακόμα και όταν το κέντρο δεδομένων του δικτύου είναι ελαφρά φορτωμένο.

Αυτό προκαλείται κυρίως από την κοινή χρήση της CPU μεταξύ των μικρών εικονικών μηχανών. Για την αποφυγή τέτοιων επιπτώσεων, θέλει προσεκτικό σχεδιασμό των μετρήσεων του δικτύου ώστε να χρησιμοποιεί VMs που χρησιμοποιούν τουλάχιστον πλήρως ένα CPU πυρήνα. Στο [24] μελετάται η απόδοση της υπηρεσίας Amazon Simple Storage Service και μελετάται η εμπειρία της μετανάστευσης μιας εφαρμογής στο Νέφος.

Η μελέτη σχετίζεται κυρίως με τρεις τομείς έρευνας: τη συγκριτική αξιολόγηση του Υπολογιστικού Νέφους, την αξιολόγηση της απόδοσης των εφαρμογών του Νέφους καθώς και την πρόβλεψη της απόδοσης των εφαρμογών κατά τη μετάβασή τους σε αυτό. Σε σχέση με τη συγκριτική αξιολόγηση των υπηρεσιών του Νέφους, το CloudHarmony [22] και το CloudSleuth [23] είναι εργαλεία μέτρησης της απόδοσης που αποθηκεύουν τα αποτελέσματα των tests δίνοντας πρόσβαση μέσω ενός web API. Το πρώτο προσφέρει ένα πολύ μεγάλο αριθμό benchmarks και μετρήσεων της απόδοσης με έμφαση στη CPU, το δίσκο (I/O), τη μνήμη (I/O) κλπ. για διάφορους παρόχους του Νέφους σε απευθείας σύνδεση. Ωστόσο, καθώς περιλαμβάνεται ένας μεγάλος αριθμός benchmarks, θα ήταν επιθυμητό να περιοριστεί το πεδίο των tests και να στραφεί σε ενδιαφέρουσες μετρήσεις. Το CloudSleuth μπορεί να δημιουργήσει μία benchmark εφαρμογή η οποία παρέχει τη διαθεσιμότητα (availability) και το χρόνο απόκρισης των διαφόρων παρόχων Νέφους σε απευθείας σύνδεση, με τη συνεχή παρακολούθηση μιας εφαρμογής δείγματος η οποία εκτελείται στους παρόχους του Υπολογιστικού Νέφους. Ωστόσο, εστιάζει μόνο σε web-based εφαρμογές.

Όσον αφορά τα πλαίσια της απόδοσης, το PerfKit Benchmarker [24] είναι ένα εργαλείο ανοικτού κώδικα για τη συγκριτική αξιολόγηση του Νέφους, δίνοντας πληροφορία στους προγραμματιστές σχετικά με το throughput, το latency τη διακύμανση και το overhead. Το πλαίσιο αυτό περιλαμβάνει δημοφιλή benchmark φορτία εργασίας που μπορούν να εκτελεστούν σε πολλαπλούς παρόχους Νέφους. Ωστόσο, το PerfKit υποστηρίζει μόνο τις

εταιρείες Amazon AWS, τη Microsoft Azure και τη Gogle Compute Google. Τέλος, το Skymark [25] είναι ένα επεκτάσιμο και φορητό πλαίσιο ανάλυσης της απόδοσης για IaaS παρόχους. Επιτρέπει τη δημιουργία πραγματικών ή συνθετικών πολύπλοκων φορίτων εργασίας σε όλα τα IaaS περιβάλλοντα Νέφους και αναλύει τον αντίκτυπο των επιμέρους διατάξεων και της πολιτικής που καθορίζεται από το χρήστη, πριν από την έναρξη του πειράματος. Μέσα από τη συσσώρευση των στατιστικών πληροφοριών σχετικά με την εκτέλεση του φόρτου εργασίας, το πλαίσιο είναι σε θέση να προβεί σε ανάλυση των επιδόσεων του υποκείμενου συστήματα IaaS. Τέλος το CloudStatus καταγράφει συνεχώς τους δύο παρόχους Amazon AWS και Google AppEngine [5] και κυρίως συγκεκριμένες μετρήσεις απόδοσης που αφορούν ειδικά τους δύο αυτούς παρόχους.

Όσον αφορά την πρόβλεψη της απόδοσης της εφαρμογής κατά τη μετανάστευσή της σε υποδομές Νέφους, υπάρχουν δύο βασικές προσεγγίσεις που χρησιμοποιούνται από τους μηχανισμούς πρόβλεψης της απόδοσης. Το αποτέλεσμα της πρόβλεψης είτε μπορεί να υποδείξει στον πελάτη άμεσα τον καλύτερο πάροχο υπηρεσίας, είτε να περιορίσει το πεδίο των πιθανών λύσεων και των δοκιμών. Η πρώτη προσέγγιση είναι η μέθοδος της συγκριτικής αξιολόγησης κατά την οποία χρησιμοποιούνται τα benchmarks [1,3] παρέχουν μία κοινή βάση για τη σύγκριση της απόδοσης των διαφορετικών παρόχων. Ωστόσο, η επιλογή των κατάλληλων benchmarks απαιτεί ιδιαίτερη προσοχή και μελέτη.

Η δεύτερη προσέγγιση είναι η Μοντελοποίηση και χρησιμοποιείται ευρέως για την πρόβλεψη της απόδοσης των εφαρμογών [5]. Ωστόσο, θεωρείται πρόκληση η περιγραφή της πολυπλοκότητας του φόρτου εργασίας μιας εφαρμογής χρησιμοποιώντας τα συνοπτικά χαρακτηριστικά ενός μοντέλου.

Οι δύο μελέτες που εντοπίζονται στη βιβλιογραφία και είναι πιο κοντά στην παρούσα διατριβή είναι οι μηχανισμοί CloudProphet και CloudCMP. Η πρώτη περίπτωση αποτελεί ένα εργαλείο,

στόχος του οποίου είναι να παρέχει ακριβείς προβλέψεις για τις εφαρμογές. Χρησιμοποιεί την προσέγγιση trace-and-replay [4]. Κατά τη διάρκεια της ανίχνευσης της εφαρμογής το CloudProphet καταγράφει τις λεπτομερείς πληροφορίες του φόρτου και τις εσωτερικές εξαρτήσεις από μία αντιπροσωπευτική εκτέλεση της εφαρμογής και επαναλαμβάνει το ίδιο φόρτου εργασίας στο Νέφος για την πρόβλεψη της απόδοσης και του κόστους.

Ωστόσο, ένας σημαντικά πρακτικός περιορισμός είναι ότι απαιτεί πολλαπλές εκτελέσεις της εφαρμογής για την απόκτηση του κατάλληλου φόρτου εργασίας το οποίο θα εκτελεστεί στο Νέφος, και αυτό γενικά μπορεί να είναι απαγορευτικό, αν η εφαρμογή έχει πολλά events συγχρονισμού.

Επιπλέον, το CloudProphet στοχεύει μόνο σε εφαρμογές web, ενώ η προσέγγισή μας καλύπτει όσο το δυνατόν περισσότερους τύπους εφαρμογών. Το CloudCmp [3] παρουσιάζει πολλά κοινά σημεία με τη δική μας προσέγγιση. Το CloudCmp παρέχει μια μεθοδολογία που έχει ως στόχο την εκτίμηση της απόδοσης και του κόστους των εφαρμογών όταν μεταφερθούν σε περιβάλλοντα Νέφους. Ένας πιθανός πελάτης μπορεί να χρησιμοποιήσει τα αποτελέσματα για να συγκρίνει τους διαφορετικούς παρόχους και να αποφασίσει αν θα πρέπει να μεταφέρει την εφαρμογή του στο Νέφος, αλλά και ποιος πάροχος είναι ο καταλληλότερος να τη φιλοξενήσει. Το CloudCmp προσδιορίζει τις κοινές υπηρεσίες για διάφορους παρόχους Νέφους, και στη συνέχεια, για κάθε υπηρεσία προσδιορίζει ένα σύνολο μετρήσεων σχετικών με την απόδοση και το κόστος της εφαρμογής. Στη συνέχεια εκτελεί μια benchmark διεργασία για κάθε μετρική στους παρόχους του Νέφους. Παρ' όλο που το CloudCmp ακολουθεί παρόμοια προσέγγιση με εμάς, ωστόσο, δεν καθορίζει ένα κοινό πλαίσιο για όλα τις benchmark διεργασίες, καθώς επίσης δεν λαμβάνει υπ' όψη το υπολογιστικό προφίλ της εφαρμογής.

5.2 Εξαγωγή χαρακτηριστικών απόδοσης

Όπως ήδη έχει αναφερθεί η απόδοση των υπηρεσιών του Νέφους αποτελεί κρίσιμο σημείο για την εξαγωγή δεδομένων σχετικά με την απόδοση τα οποία μπορούν να βοηθήσουν στη φάση της μετανάστευσης των εφαρμογών στο Νέφος. Σε αυτή την προσπάθεια εφαρμόστηκε η μέθοδος της συγκριτικής αξιολόγησης.

Το πρώτο βήμα της διαδικασίας αυτής είναι να καθοριστεί μια σειρά από στερεότυπα απόδοσης τα οποία στηρίζονται σε διαφορετικές κατηγορίες εφαρμογών. Ο κύριος στόχος αυτών των στερεοτύπων είναι να εξάγουν μια σειρά από χαρακτηριστικά απόδοσης του παρόχου που είναι απαραίτητα για την ικανοποίηση των απαιτήσεων της ποιότητας της υπηρεσίας (QoS) των εφαρμογών που μεταναστεύουν στο Νέφος. Η πηγή αυτών των χαρακτηριστικών είναι κοινοί τύποι εφαρμογών που αντιστοιχούν σε διάφορες δημοφιλείς εφαρμογές και έχουν συνδεθεί με τα αντίστοιχα tests της συγκριτικής αξιολόγησης που μπορεί να χρησιμοποιηθούν για να υποδείξουν μια συγκεκριμένη δυνατότητα της υπηρεσίας για την επίλυση πραγματικών υπολογιστικών προβλημάτων. Έτσι, έχουν εντοπιστεί tests τα οποία έχουν ταυτιστεί με συγκεκριμένα μοτίβα φορτίων εργασίας που μπορεί να αντιστοιχιστούν σε συγκεκριμένες εφαρμογές του πραγματικού κόσμου. Το κύριο όφελος από μια τέτοια κατηγοριοποίηση, είναι η δυνατότητα να περιγραφούν οι δυνατότητες των υπηρεσιών που σχετίζονται με την απόδοση με ένα πιο αφαιρετικό τρόπο σε επίπεδο εφαρμογής. Με αυτό τον τρόπο οι υπηρεσίες κατατάσσονται ανάλογα με τα ενδιαφέροντα του χρήστη για μια συγκεκριμένη κατηγορία.

5.1 Μηχανισμός μέτρησης της απόδοσης του Υπολογιστικού Νέφους

Για τη μέτρηση της απόδοσης των παρόχων του Νέφους χρησιμοποιήθηκε η Benchmarking Suite [109], μία σουίτα λογισμικού, που περιλαμβάνει εργαλεία εγκατάστασης, ρύθμισης παραμέτρων και εκτέλεσης, ενώ ενσωματώνει ένα σύνολο εργαλείων συγκριτικής αξιολόγησης τα οποία έχουν επιλεχθεί ειδικά για την αποτελεσματικότητά τους όσον αφορά στην αξιολόγηση των πόρων του Νέφους και τα οποία έχουν παρουσιαστεί εκτενώς στο κεφάλαιο. Αυτό το πλαίσιο είναι μια πλήρως αυτοματοποιημένη λύση για τη διαχείριση της διαδικασίας συγκριτικής αξιολόγησης, η οποία βασίζεται σε ένα σύνολο εργαλείων αξιολόγησης που καλύπτουν όσο το δυνατόν περισσότερο το σύνολο του πεδίου εφαρμογής.

Σε σύγκριση με άλλες διαθέσιμες λύσεις συγκριτικής αξιολόγησης του Νέφους, η Benchmarking Suite μειώνει δραστικά την χειροκίνητη παρέμβαση στη διαδικασία συγκριτικής αξιολόγησης επιτρέποντας tests μαζικής και μεγάλης κλίμακας.

Μία τέτοια σουίτα επιτρέπει τη διεξαγωγή των tests και την απόκτηση των μετρήσεων απόδοσης με έναν ομοιογενή και ανεξάρτητο τρόπο, χωρίς να επηρεάζεται και να λαμβάνει υπόψη τις αποδόσεις που δημοσιεύονται από τους παρόχους. Στη συνέχεια, οι μετρήσεις μπορούν να χρησιμοποιηθούν για τη συλλογή ποσοτικής πληροφορίας που σχετίζεται με τις προσφερόμενες αποδόσεις και αποτελούν τη βάση για την επιλογή των υπηρεσιών του Νέφους.

Η αρχιτεκτονική της Benchmarking Suite εμφανίζεται στο *Σχήμα 11* . Ο χρήστης μέσω της γραφικής διασύνδεσης (GUI), μπορεί να ρυθμίσει τις συνθήκες των tests, επιλέγοντας σχετικά tests της συγκριτικής αξιολόγησης, τις συνθήκες του φόρτου εργασίας, τον πάροχο και την προσφερόμενη υπηρεσία. Αυτή η πληροφορία μεταβιβάζεται στον Benchmarking Controller

που είναι υπεύθυνος τόσο για την εκκίνηση των εικονικών πόρων στον πάροχο-στόχο, όσο για την εκτέλεση των tests. Ο Benchmarking Controller βασίζεται στην ενσωμάτωση του Apache LibCloud [108], προκειμένου να υποστηρίξει πολλαπλά πλαίσια παρόχων χρησιμοποιώντας συνδέσμους (connectors). Αρχικά, εγκαθίσταται τα tests τα οποία βρίσκονται σε έναν εξωτερικό Linux χώρο αποθήκευσης. Μόλις τα tests εγκατασταθούν τα scripts που ρυθμίζουν το φόρτο εργασίας μεταφέρονται στα εικονικά μηχανήματα στόχο ώστε να αρχίσει η εκτέλεση. Τα αποτελέσματα μεταφέρονται πίσω και αποθηκεύονται σε μία βάση ώστε στη συνέχεια να αξιοποιηθούν ανάλογα.

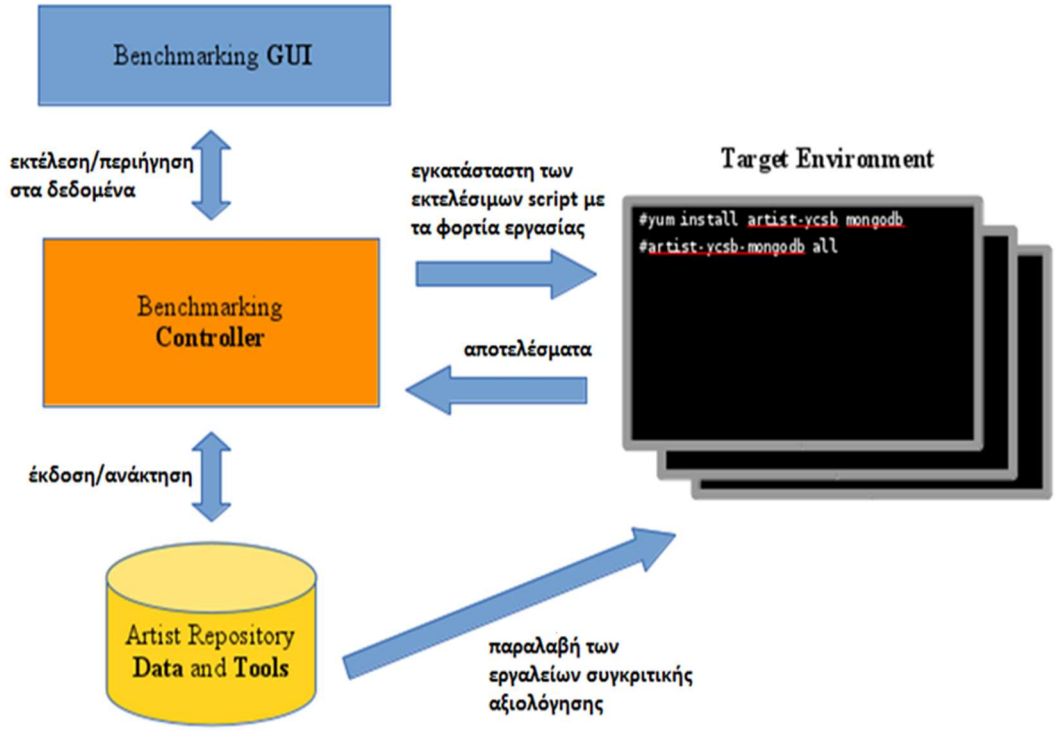
Στη συνέχεια παρουσιάζονται αναλυτικότερα τα τμήματα από τα οποία αποτελείται η Benchmarking Suite.

• Ο αποθηκευτικός χώρος περιέχει τα εργαλεία της συγκριτικής αξιολόγησης που χρησιμοποιούνται για τη μέτρηση των επιδόσεων των υπηρεσιών του Νέφους. Περιέχει επίσης τα αποτελέσματα των tests σε κατάλληλη μορφή που μπορεί να χρησιμοποιηθεί και από άλλα δομικά στοιχεία.

• Περιβάλλον στο οποίο θα εκτελεστεί το υπό δοκιμή σύστημα. Στην προσφορές IaaS, το περιβάλλον είναι μια εικονική μηχανή η οποία λειτουργεί με ένα συγκεκριμένο λειτουργικό σύστημα ενώ στις PaaS προσφορές θεωρούνται όλες υπηρεσίες που εκτελούνται στην υποδομή του παρόχου του Νέφους και μπορούν να χρησιμοποιηθούν από εφαρμογές που έχουν αναπτυχθεί.

• Ο Benchmarking Controller είναι το κύριο δομικό στοιχείο της αρχιτεκτονικής αυτής το οποίο αυτοματοποιεί την εκτέλεση των tests της συγκριτικής αξιολόγησης. Ο κύριος στόχος του είναι να αυτοματοποιήσει την συνήθη ροή εργασίας της εκτέλεσης της συγκριτικής αξιολόγησης που μέχρι τώρα έπρεπε να γίνει χειροκίνητα και περιλάμβανε τα εξής βήματα: 1) τη δημιουργία του περιβάλλοντος στόχου, 2) την εγκατάσταση των εργαλείων συγκριτικής

αξιολόγησης , 3) την εκτέλεση των tests της συγκριτικής αξιολόγησης 4) την ανάκτηση των αποτελεσμάτων και 5) την αποθήκευση δεδομένων.

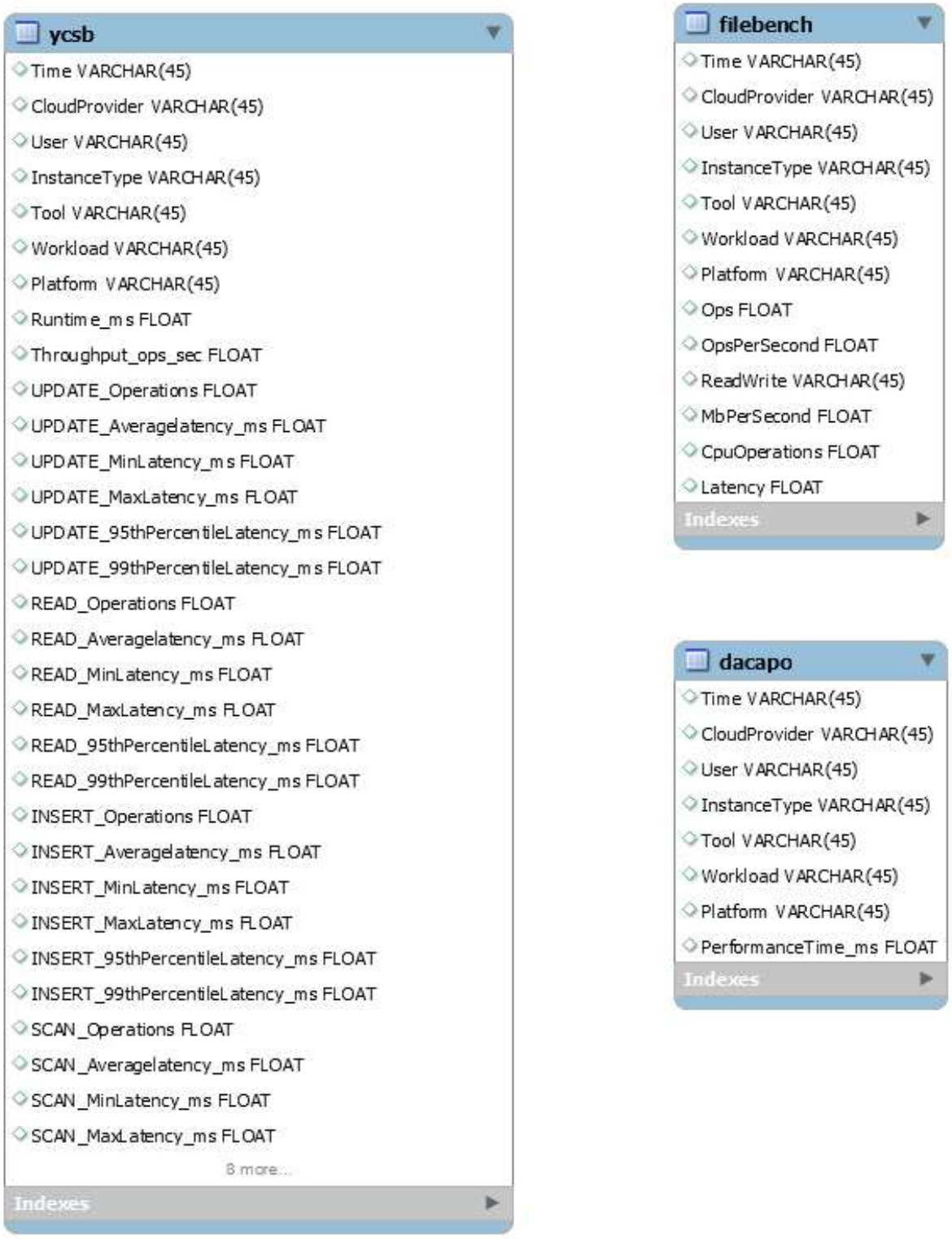


Σχήμα 11: Αρχιτεκτονική του συστήματος Benchmarking Suite

Στην παρούσα διατριβή για την εκτέλεση των tests της συγκριτικής αξιολόγησης χρησιμοποιήθηκε μόνο ο Benchmarking Controller, ο οποίος μπορεί να λειτουργήσει αυτόνομα για την εκτέλεση των μετρήσεων για τους διάφορους τύπους των εφαρμογών.

Η πλήρης αυτοματοποίηση της ροής εργασίας, είναι δυνατή μόνο για ορισμένο συνδυασμό παρόχων του Νέφους και εργαλείων συγκριτικής αξιολόγησης . Σε όλες τις άλλες περιπτώσεις, μερικά από τα βήματα εξακολουθούν να απαιτούν χειροκίνητη παρέμβαση του χρήστη. Για παράδειγμα, στην περίπτωση που ένας IaaS πάροχος δεν παρέχει ένα API για τη δημιουργία ή την καταστροφή των εικονικών μηχανών, τότε θα πρέπει ο χρήστης να δημιουργήσει χειροκίνητα το περιβάλλον και στη συνέχεια να επισημάνει το τέλος της λειτουργίας των εικονικών μηχανών στον ελεγκτή συγκριτικής αξιολόγησης που θα συνεχίσει το έργο της ροής.

Αυτή η σελίδα είναι σκόπιμα λευκή



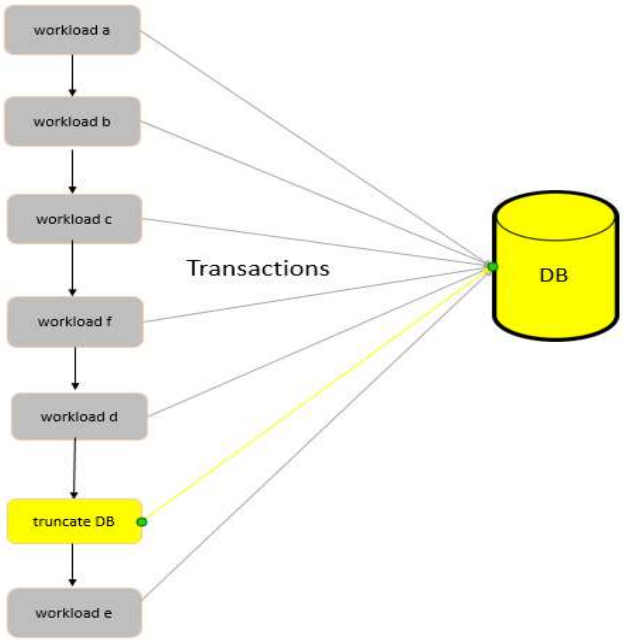
Σχήμα 12: EER διάγραμμα βάσης

Αυτή η σελίδα είναι σκόπιμα λευκή

• Η γραφική διασύνδεση (Benchmarking GUI) η οποία προσφέρει δύο κύριες λειτουργίες στον χρήστη οι οποίες είναι η υποβολή εκτέλεσης νέων tests συγκριτικής αξιολόγησης και η δυνατότητα περιήγησης στα δεδομένα απόδοσης που συλλέγονται.

• Η Backend βάση δεδομένων για την αποθήκευση των αποτελεσμάτων από τις εκτελέσεις των tests τα οποία μπορούν να ανακτηθούν εκτελώντας queries στη βάση. Επιπλέον, έχει δημιουργηθεί ένα mysql σχήμα βάσης δεδομένων και παρέχεται στον τελικό χρήστη σε περίπτωση αυτός επιθυμεί τα αποτελέσματα να αποθηκεύονται τοπικά.

5.1.1 Εργαλεία Συγκριτικής Αξιολόγησης



Σχήμα 13: Μεθοδολογία εκτέλεσης του YCSB benchmark

Για την μέτρηση της απόδοσης των παρόχων του Νέφους επιλέχθηκαν κάποια από τα benchmark εργαλεία που έχουν περιγραφεί εκτενώς στο κεφάλαιο 4. Τα εργαλεία αυτά είναι το DaCapo, το YCSB και το Filebench. Τα προαναφερθέντα εργαλεία επιλέχθηκαν επειδή α) έχουν αποδειχθεί πως λειτουργούν σωστά και δίνουν ακριβή αποτελέσματα, β) υποστηρίζονται από μια μεγάλη κοινότητα ειδικών γ) υπάρχει πολλή πληροφορία και οδηγίες

Αυτή η σελίδα είναι σκόπιμα λευκή

καθώς και tests που ήδη έχουν πραγματοποιηθεί και είναι διαθέσιμα. Στόχος των εργαλείων της αξιολόγησης είναι να εκτελεστούν πολλές φορές ώστε να συλλάβουν και να αποτυπώσουν τη διακύμανση των τιμών της απόδοσης. Δεδομένου ότι οι χρήστες του Νέφους χρειάζονται σταθερότητα στην απόδοση, η συγκριτική αξιολόγηση των πόρων πρέπει να είναι μια επαναλαμβανόμενη διαδικασία μέτρησης με την πάροδο του χρόνου ώστε να παρατηρηθούν οι διακυμάνσεις των προσφερόμενων υπηρεσιών. Προκειμένου να επιτευχθεί η επανάληψη της συγκριτικής αξιολόγησης απαραίτητη προϋπόθεση ήταν η εκτενής μελέτη των φορτίων εργασίας της συγκριτικής αξιολόγησης ώστε να κατανοηθεί πλήρως ο τρόπος λειτουργίας τους. Σε κάποιες περιπτώσεις απαραίτητη ήταν η ενσωμάτωση και η ρύθμιση ενός αριθμού παραμέτρων ώστε το κάθε test να επαναλαμβάνεται με τέτοιο τρόπο, ικανό να συμπεριλάβει τις επιθυμητές διακυμάνσεις της απόδοσης. Τέλος, υλοποιήθηκαν script bash και χρησιμοποιήθηκαν με τον Benchmarking Controller ώστε να ρυθμίσουν τη σωστή σειρά για την εκτέλεση των tests αλλά και για να διαγράψουν σε ορισμένες περιπτώσεις δεδομένα στη βάση δεδομένων. Για παράδειγμα, για το YCSB, προκειμένου να διατηρηθεί το μέγεθος της βάσης δεδομένων σε αρμονία τα φορτία εργασίας πρέπει να εκτελεστούν με μία συγκεκριμένη σειρά και πριν την εκτέλεση του τελευταίου φόρτου εργασίας να αδειάσει η βάση.

5.2 Εφαρμογή της Συγκριτικής Αξιολόγησης σε παρόχους του Υπολογιστικού Νέφους

Σε αυτή την ενότητα παρουσιάζεται μια λεπτομερής ανάλυση για τη διαδικασία της συγκριτικής αξιολόγησης σε τρεις μεγάλους εμπορικούς παρόχους του Νέφους: την Amazon EC2, τη Microsoft Azure και τη Flexiant. Τα αποτελέσματα των μετρήσεων αποθηκεύονται στη βάση δεδομένων που προαναφέρθηκε, προκειμένου να χρησιμοποιηθούν στον κύριο

πλαίσιο που έχει υλοποιηθεί στην παρούσα διατριβή και συμβάλλει καταλυτικά στην επιλογή του χρήστη για την καλύτερη εύρεση παρόχου στο Νέφος για τη φιλοξενία της εφαρμογής του.

5.2.1 Εφαρμογή της Συγκριτικής Αξιολόγησης

Για να τον αρχικό πειραματισμό με τις προσδιορισμένες μετρήσεις και για τη διερεύνηση της απόδοσης διαφόρων εικονικών μηχανών, χρησιμοποιήθηκαν όπως ήδη έχει αναφερθεί φορτία εργασίας από τις σουίτες συγκριτικής αξιολόγησης DaCapo, YCSB και Filebench. Αρχικά ο χρήστης εγκαθιστά τοπικά τον Benchmarking Controller, τα εργαλεία της συγκριτικής αξιολόγησης και τα αντίστοιχα script bash για την παραμετροποίηση των tests. Κατά τη διάρκεια της διαδικασίας εκτέλεσης ο χρήστης τρέχει τοπικά τον Benchmarking Controller προσδιορίζοντας το περιβάλλον στόχο όπως π.χ. την Amazon καθώς και τον τύπο της εικονικής μηχανής ο οποίος θα δημιουργηθεί και μετά θα καταστραφεί (π.χ. OS και μέγεθος). Επίσης, ο χρήστης επιλέγει το εργαλείο συγκριτικής αξιολόγησης το οποίο θα εκτελεστεί στην υπηρεσία του παρόχου του Νέφους (π.χ. DaCapo, YCSB, Filebench). Τα αποτελέσματα της εκτέλεσης μεταφέρονται πίσω σε τοπικό επίπεδο, αναλύονται και τελικά αποθηκεύονται στην τοπική βάση δεδομένων. Στον *Πίνακας 7* παρουσιάζονται οι τύποι των εφαρμογών που χρησιμοποιήθηκαν στη διαδικασία της συγκριτικής αξιολόγησης στην παρούσα διατριβή.

Benchmark Test	**Application Type**
YCSB	Databases
Filebench	File system & storage
DaCapo	JVM applications

Πίνακας 7: Οι μέθοδοι της συγκριτικής αξιολόγησης και οι τύποι των εφαρμογών που χρησιμοποιήθηκαν στη μεθοδολογία

Τα επιλεγμένα φορτία εργασίας από κάθε test εκτελέστηκαν σε τρία διαφορετικά περιβάλλοντα Νέφους: στην Amazon EC2, στη Microsoft Azure και στη Flexiant. Όσον

αφορά στη μέτρηση της απόδοσης, από κάθε πάροχο επιλέχθηκαν διάφοροι τύποι εικονικών μηχανών στις οποίες εκτελέστηκαν τα tests της συγκριτικής αξιολόγησης.

Η εκτέλεση των tests πραγματοποιήθηκε σε συγκεκριμένες ώρες (σε διαφορετικά χρονικά διαστήματα) κατά τη διάρκεια μιας περιόδου οκτώ μηνών (Ιούλιος 2014 - Φεβρουάριος 2015) και στη συνέχεια για κάθε περίπτωση υπολογίστηκε η μέση τιμή της απόδοσης για το κάθε test. Επιπλέον, λήφθηκε υπόψη η διαφορετική ώρα ζώνης της τοποθεσίας του κάθε παρόχου έτσι ώστε οι ώρες αιχμής να είναι ίδιες σε κάθε ζώνη. Στην περίπτωση της Amazon EC2, οι VMs που χρησιμοποιήθηκαν «έτρεχαν» σε ένα datacenter στη Βόρεια Βιρτζίνια, ενώ στην περίπτωση της Microsoft Azure και της Flexiant τα datacenters βρίσκονται στην Ιρλανδία και το Ηνωμένο Βασίλειο, αντίστοιχα. Πληροφορίες σχετικά με τα χαρακτηριστικά των εικονικών μηχανών που χρησιμοποιήθηκαν παρουσιάζονται στον *Πίνακας 8*.

Cloud Provider	VM instance	Region
Amazon EC2	t1.micro	N.Virginia
	m1.small	
	m1.medium	
	m1.large	
Microsoft Azure	A1 small Standard	Ireland
	A2 medium Standard	
Flexiant	1GB RAM- 1CPU	UK
	2GB RAM- 2CPU	
	4GB RAM- 3CPU	
	4GB RAM- 4CPU	

Πίνακας 8: Οι πάροχοι και οι τύποι των εικονικών μηχανών αλλά και η τοποθεσία τους που χρησιμοποιήθηκαν στην μεθοδολογία

Μετά την ολοκλήρωση της διαδικασίας της συγκριτικής αξιολόγησης τα αποτελέσματα ανακτήθηκαν από την τοπική βάση δεδομένων, επεξεργάστηκαν και δημιουργήθηκαν οι αντίστοιχες γραφικές παραστάσεις που παρουσιάζονται στη συνέχεια ώστε να διεξαχθούν συμπεράσματα σχετικά με την απόδοση των υπηρεσιών.

5.2.2 Δείκτης Αποδοτικότητας Service Efficiency

Είναι σημαντικό να αναφερθεί σε αυτό το σημείο πως η ταξινόμηση των υπηρεσιών του Νέφους μπορεί να χαρακτηριστεί είτε μόνο με βάση την απόδοση είτε με ένα συνδυασμένο δείκτη αποδοτικότητας που περιλαμβάνει την απόδοση και το κόστος και εκφράζεται μέσω της εξίσωσης:

$$\mathrm{SE} = \frac{1}{w1 * Delay + w2 * Cost}$$

Η υλοποίηση της παραπάνω εξίσωσης προέκυψε στην προσπάθεια να δημιουργηθεί μία μετρική που θα εστιάζει στην απόδοση της υπηρεσίας και θα είναι απλή, εύκολα υπολογίσιμη και κατανοητή από τους χρήστες καθώς και θα πληρεί τις ακόλουθες προϋποθέσεις:

- Να περιλαμβάνει πληροφορία για το κόστος της επιλεγμένης υπηρεσίας
- Να λαμβάνει υπόψη την απόδοση ενός δεδομένου φόρτου εργασίας για ένα δεδομένο τύπο εφαρμογής
- Να δίνει τη δυνατότητα για διαφορετικές κατατάξεις με βάση τα ενδιαφέροντα των χρηστών χρησιμοποιώντας βάρη
- Διαισθητικά οι υψηλότερες τιμές θεωρούνται σαν καλύτερο αποτέλεσμα

Έχοντας σαν οδηγό τις παραπάνω προϋποθέσεις η εξίσωση περιλαμβάνει τους θετικούς παράγοντες (π.χ. το φόρτο εργασίας) στον αριθμητή και τους αρνητικούς στον παρονομαστή όπως είναι το κόστος και οι βασικοί δείκτες απόδοσης που ακολουθούν την προσέγγιση «το υψηλότερο είναι και χειρότερο». Ακόμη, για να υπάρξει μια γενική και ενιαία βαθμολογία

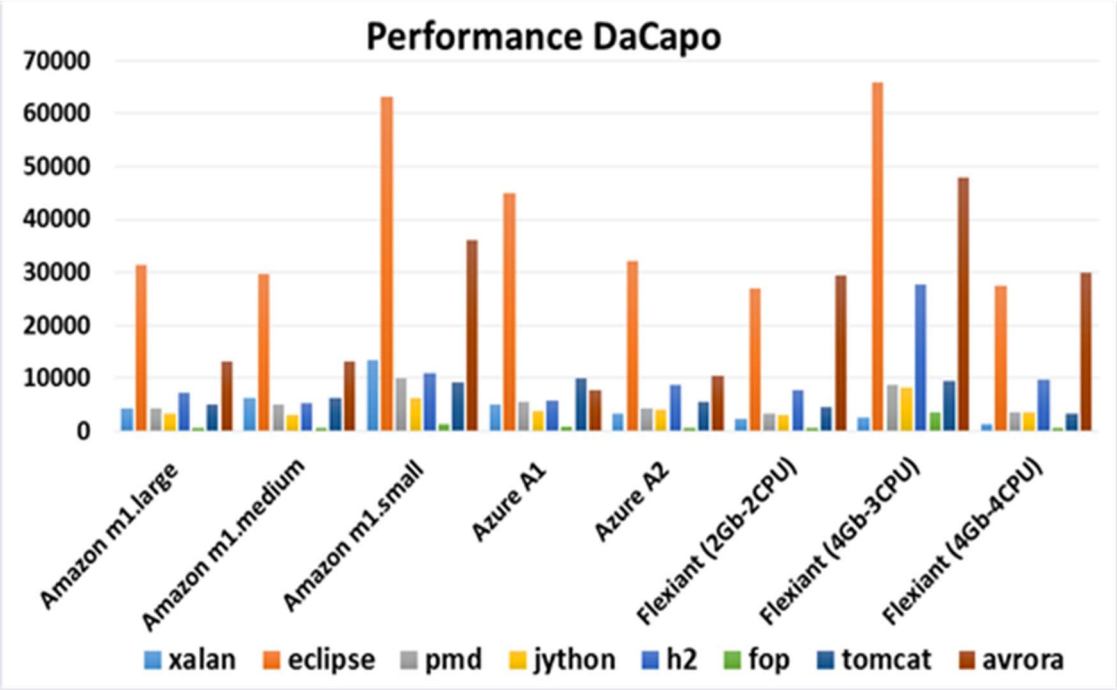
είναι αναγκαία η χρήση της κανονικοποίησης. Θέλοντας να συμπεριληφθεί ο παράγοντας του βάρους στην εξίσωση, είναι απαραίτητη η εμφάνιση αθροίσματος στον παρονομαστή το οποίο προσφέρει αυτή τη δυνατότητα. Για να γίνει απόλυτα κατανοητό, η μετρική που αναπτύχθηκε ουσιαστικά μπορεί να απαντήσει σε πιθανές ερωτήσεις όπως: "ποια είναι η καλύτερη υπηρεσία για να τρέξει μία web streaming εφαρμογή, όταν ο χρήστης ενδιαφέρεται να χρησιμοποιήσει μια φθηνή υπηρεσία;". Η προκύπτουσα μετρική μπορεί να συγκριθεί μεταξύ διαφορετικών υπηρεσιών που προσφέρονται από τους διαφορετικούς παρόχους, αλλά χρησιμοποιώντας τον ίδιο φόρτο εργασίας. Η ενσωμάτωση του φόρτου εργασίας είναι απαραίτητη, δεδομένου ότι επηρεάζει την απόδοση και κατά συνέπεια την κατάταξη των υπηρεσιών.

5.2.3 Γραφική Απεικόνιση των αποτελεσμάτων

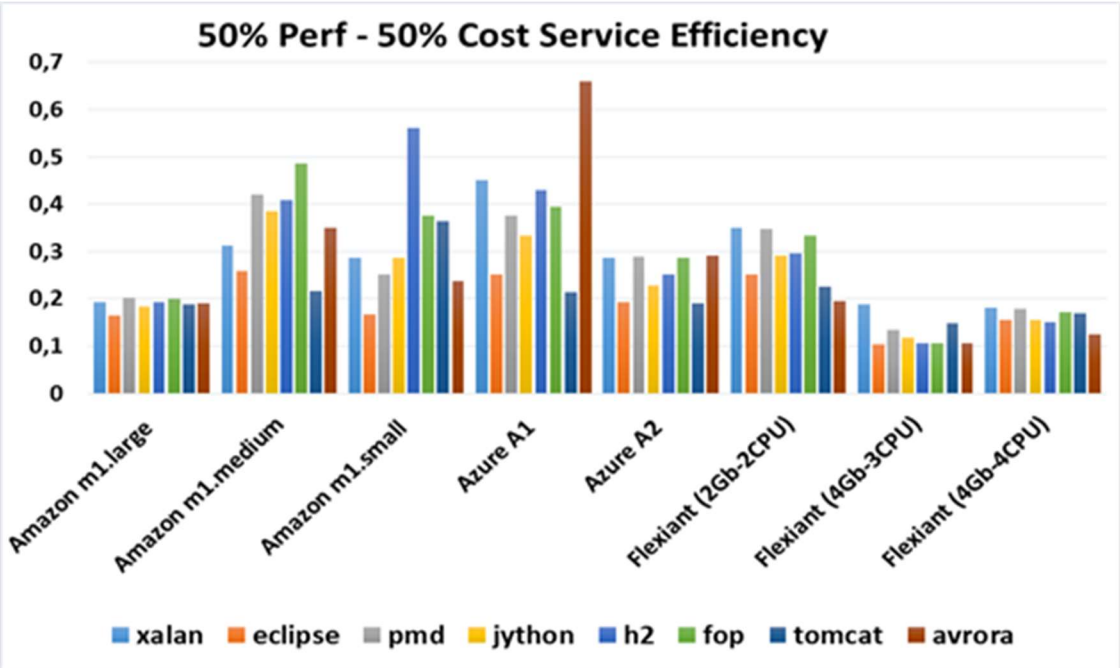
Στη συνέχεια παρουσιάζονται οι γραφικές παραστάσεις που δημιουργήθηκαν με βάση τα δεδομένα που συλλέχθηκαν από τη διαδικασία της συγκριτικής αξιολόγησης. Συγκεκριμένα για να διεξαχθούν συμπεράσματα θα πρέπει κανείς να συγκρίνει τον ίδιο φόρτο εργασίας (υποδεικνύεται με το ίδιο χρώμα στο ραβδόγραμμα) το οποίο έχει εκτελεστεί σε διάφορους τύπους εικονικών μηχανών και στους τρεις παρόχους του Νέφους.

Από τις γραφικές παραστάσεις είναι εμφανές ότι η απόδοση για ένα συγκεκριμένο φόρτο εργασίας ποικίλει και εξαρτάται τόσο από τον τύπο του φόρτου εργασίας όσο και από το μέγεθος της εικονικής μηχανής. Για παράδειγμα, για την DaCapo σουίτα η απόδοση των φορτίων εργασίας (*Σχήμα 14*) παρουσιάζει σχεδόν παρόμοια συμπεριφορά σε τρεις τύπους εικονικών μηχανών: στην A2 Standard εικονική μηχανή της Azure και στις m1.large και m1.medium της Amazon EC2. Ωστόσο, σε ορισμένες περιπτώσεις υπάρχουν κάποιες εξαιρέσεις όπου η Amazon παρέχει καλύτερα αποτελέσματα, όπως για το avrora φόρτο εργασίας, ενώ η Azure φαίνεται πως δίνει καλύτερα αποτελέσματα για το h2 φόρτο εργασίας.

Παρόμοια αποτελέσματα έχουν ληφθεί για τα YCSB και Filebench τα οποία θα συμπεριληφθούν στην τελική διατριβή. Αυτό που φαίνεται επίσης από το *Σχήμα 14* και είναι πιο εμφανές μέσω του αποτελέσματος που παράγεται από την εφαρμογή της Service Efficiency μετρικής στο *Σχήμα 15*, είναι το γεγονός ότι σε πολλές περιπτώσεις VMs χαμηλότερης δυνατότητας, έχουν αποδειχθεί ότι είναι σημαντικά πιο αποτελεσματικές. Αυτό μπορεί να οφείλεται στο γεγονός ότι τα φορτία εργασίας των tests δεν οδηγούν τους πόρους των εικονικών μηχανών στα όριά τους, επομένως όταν το κόστος της χρήσης της εικονικής μηχανής συμπεριλαμβάνεται στον υπολογισμό της βέλτιστης λύσης, είναι προφανές ότι δεν είναι αναγκαία η χρήση εικονικών μηχανών υψηλότερης δυνατότητας για τη συγκεκριμένη περίπτωση. Επιπλέον, από τις απόλυτες βαθμολογίες της απόδοσης στο σχήμα 14, τα αποτελέσματα ακολουθούν τη λογική πως οι VMs με υψηλότερη ικανότητα παράγουν καλύτερα αποτελέσματα. Ωστόσο, μπορεί να ληφθεί ένας αριθμός μη διαισθητικών περιπτώσεων, για παράδειγμα στην περίπτωση του avrora όταν εκτελείται στις VMs της Flexiscale 2Gb-2CPU και 4GB-3CPU αντίστοιχα. Σε αυτή την περίπτωση, η χρήση μιας μικρότερης VM δεν είναι μόνο πιο αποτελεσματική, αλλά παρουσιάζει επίσης ενισχυμένη απόλυτη βαθμολογία. Αυτό μπορεί να αποδοθεί σε μια τυχαία διαδικασία έναρξης στο guest OS (το οποίο ωστόσο μειώνεται από την επανάληψη της διαδικασίας της μέτρησης).



Σχήμα 14: Η μέτρηση σε ms της απόδοσης των φορτίων εργασίας του DaCapo Benchmark



Σχήμα 15: Η SE μετρική για τα φορτία εργασίας του DaCapo Benchmark με βάρη 50% για την απόδοση και το κόστος αντίστοιχα

Αυτή η σελίδα είναι σκόπιμα λευκή

5.3 Μέθοδοι και εργαλεία για την αξιολόγηση των παρόχων του Υπολογιστικού Νέφους βάσει του προφίλ των εφαρμογών

5.3.1 Περιγραφή της συνολικής μεθοδολογίας

Για την κατηγοριοποίηση μιας εφαρμογής ή ενός δομικού της στοιχείου που «τρέχει» σε έναν εικονικό πόρο, το πρώτο βήμα της μεθοδολογίας που παρουσιάζεται στην παρούσα διατριβή είναι ο υπολογισμός του αποτυπώματος (προφίλ/profile) της απόδοσης συγκεκριμένων γνωστών εφαρμογών που έχουν επιλεχθεί και εκπροσωπούν τυπικές κατηγορίες εφαρμογών, κάθε μία από τις οποίες αντιπροσωπεύεται από ένα benchmark test. Για τη μέτρηση του προφίλ αυτού, έχει υλοποιηθεί το Profiling Tool το οποίο διαθέτει δύο καταστάσεις λειτουργίας ώστε να υπολογίζει το προφίλ της απόδοσης των benchmark tests αλλά και μιας μη γνωστής αυθαίρετης εφαρμογής.

Στην πρώτη φάση της μεθοδολογίας το Profiling Tool βρίσκεται στην κατάσταση λειτουργίας benchmarking σύμφωνα με την οποία ο χρήστης φροντίζει να έχει εγκαταστήσει τοπικά σε μία εικονική μηχανή τα tests της συγκριτικής αξιολόγησης χρησιμοποιώντας την Benchmarking Suite (η οποία παρουσιάστηκε στην ενότητα 7.1) και στη συνέχεια, εκτελεί τα tests ενώ ταυτόχρονα ξεκινά αυτόματα την παρακολούθηση της εκτέλεσης τους μέσω των εργαλείων παρακολούθησης Pidstat και Tsark [109], [110] χρησιμοποιώντας το Profiling Tool το οποίο είναι εγκατεστημένο στο physical host, όπου βρίσκεται η εικονική μηχανή με τα tests της συγκριτικής αξιολόγησης.

Στη συνέχεια το επόμενο βήμα της μεθοδολογίας είναι η μέτρηση του αποτυπώματος του δομικού στοιχείου (component) της εφαρμογής το οποίο είναι εγκατεστημένο ακριβώς στο ίδιο περιβάλλον όπως ήταν και τα benchmark tests. Και σε αυτή την περίπτωση χρησιμοποιείται το Profiling Tool, αλλά αυτή τη φορά σε κατάσταση λειτουργίας «application

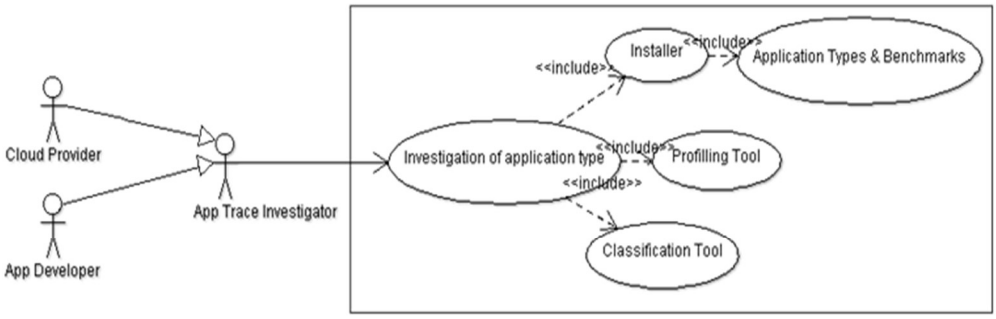
mode». Τα αποτυπώματα της απόδοσης που λαμβάνονται στο πρώτο στάδιο, χρησιμοποιούνται σε ένα τρίτο στάδιο για να την εκπαίδευση του Classification Tool. Η υλοποίηση του συγκεκριμένου εργαλείου έχει σαν στόχο την καλύτερη δυνατή αντιστοίχιση της εφαρμογής με τα στερεότυπα απόδοσης, χρησιμοποιώντας την αλγοριθμική προσέγγιση των k-κοντινών γειτόνων (*knn*). Η αντιστοίχιση πραγματοποιείται με τη σύγκριση μεταξύ των αποτυπωμάτων απόδοσης των tests συγκριτικής αξιολόγησης και αυτών της αυθαίρετης εφαρμογής που προέρχονται από το πρώτο και το δεύτερο βήμα της μεθοδολογίας. Στο τελικό στάδιο, έπειτα από την αντιστοίχιση της εφαρμογής σε ένα από τα tests της συγκριτικής αξιολόγησης που πραγματοποιείται μέσω του Classification Tool εντοπίζεται ο καταλληλότερος πάροχος του Νέφους όσον αφορά στην απόδοση και στο κόστος ο οποίος θα μπορούσε να φιλοξενήσει την συγκεκριμένη εφαρμογή. Η διαδικασία της εύρεσης της καλύτερης προσφοράς στηρίζεται στα δεδομένα που έχουν συλλεχθεί από την εκτέλεση των tests της συγκριτικής αξιολόγησης σε διαφορετικούς τύπους εικονικών μηχανών σε διάφορους παρόχους του Νέφους σε συνδυασμό με την εφαρμογή της Service Efficiency μετρικής.

5.3.2 Διάγραμμα περιπτώσεων χρήσης της μεθοδολογίας

Στο *Σχήμα 16* αποτυπώνεται το διάγραμμα περιπτώσεων χρήσης (Use Case) για τη μεθοδολογία που έχει αναπτυχθεί στην παρούσα διατριβή και παρουσιάζεται η σχέση ανάμεσα στους actors και τις περιπτώσεις χρήσης του συστήματος, μαζί με τις μεταξύ τους σχέσεις.

Actor: ως actor ορίζεται ο App Trace Investigator, μια οντότητα που ερευνά πώς ένα συγκεκριμένο δομικό στοιχείο της εφαρμογής χρησιμοποιεί τους πόρους του συστήματος. Η σχέση που υπάρχει ανάμεσα σε όλες τις περιπτώσεις χρήσης είναι η συμπερίληψη. Αυτό σημαίνει ότι για παράδειγμα μια σχέσης συμπερίληψης από την περίπτωση χρήσης «Investigation of application type» στην περίπτωση χρήσης «Installer» δηλώνει ότι ένα στιγμιότυπο της περίπτωσης χρήσης «Investigation of application type» θα περιέχει επίσης τη

συμπεριφορά που καθορίζει η «Installer». Η συμπεριφορά περιλαμβάνεται στη θέση που καθορίζει η «Investigation of application type».



Σχήμα 16: Use Case διάγραμμα για τη μεθοδολογία της παρούσας διατριβής

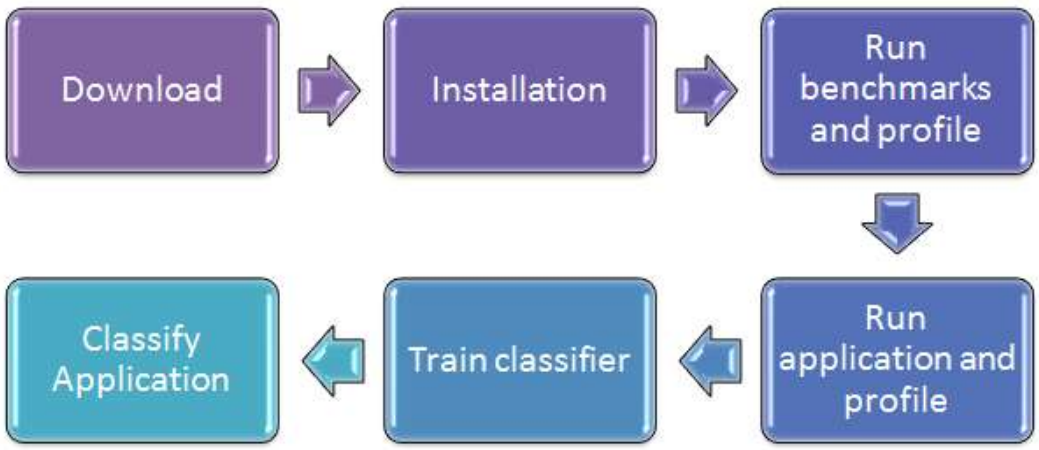
Όσον αφορά τους actors η οντότητα αυτή μπορεί περαιτέρω να αναλυθεί σε δύο συγκεκριμένες οντότητες, σε εκείνη του προγραμματιστή της εφαρμογής και σε εκείνη του παρόχου του Νέφους. Ο πρώτος ενδιαφέρεται κυρίως προκειμένου να ελέγξει τα σημεία συμφόρησης(bottleneck) της εφαρμογής ή να αποφασίσει για το ποια είναι η καλύτερη υπηρεσία του Νέφους για την εφαρμογή του, δεδομένου ότι οι υπηρεσίες του Νέφους συγκρίνονται με τις ίδιες κατηγορίες εφαρμογών που χρησιμοποιούνται για την ταξινόμηση της εφαρμογής. Το όφελος των παρόχων του Νέφους από τη μεθοδολογία αυτή, είναι να ελαχιστοποιήσουν την απόκλιση που παρατηρείται στις υποδομές τους λόγω της επίδρασης του θορυβώδους γείτονα (noisy neighbor effect) [41]. Μία προηγούμενη έρευνα [3] έδειξε ότι το είδος των υπολογιστικών φορτίων εργασίας που είναι συν-προγραμματισμένα ναεκτελεστούν σε ένα φυσικό server παρουσιάζει σημαντική διαφορά ως προς την ποιότητα της υπηρεσίας που προσφέρουν οι εικονικοί πόροι που «τρέχουν» πάνω στους φυσικούς κόμβους. Επομένως, είναι ως προς το συμφέρον του παρόχου να προσδιορίσει τους τύπους

Αυτή η σελίδα είναι σκόπιμα λευκή

των εφαρμογών που εκτελούνται στο στις VMs των πελατών τους, έτσι ώστε να μπορούν ομάδας τους με τον βέλτιστο τρόπο από ένα σημείο παρεμβολής απόδοση του άποψη.

5.3.3 Τεχνική Ανάλυση της μεθοδολογίας

Στην εικόνα παρουσιάζεται η περιγραφή της μεθοδολογίας που αναπτύχθηκε στα πλαίσια της παρούσας διατριβής. Απαραίτητη προϋπόθεση είναι να εγκατασταθούν στον υπολογιστή του χρήστη τα κατάλληλα εργαλεία (Benchmark controller, Benchmark tests, Profiling tool και Classification tool) και να εκτελεστούν τοπικά.



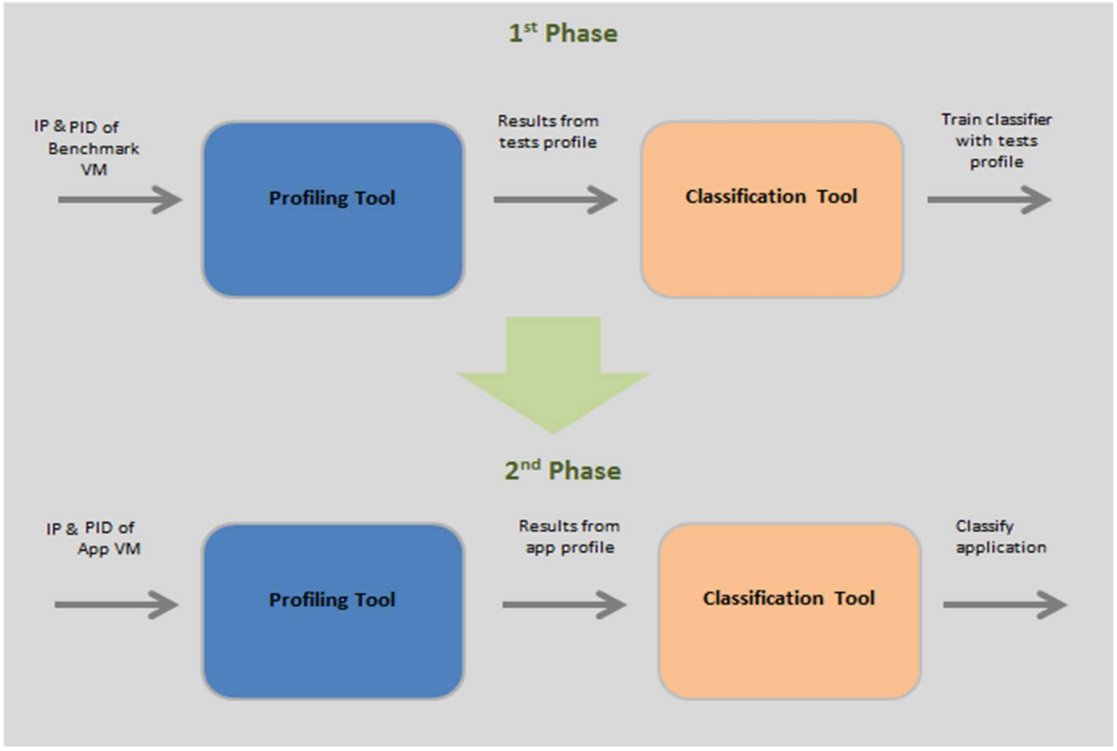
Σχήμα 17: Περιγραφή των διαδικασιών του προφίλ και της κατάταξης μιας εφαρμογής

Σαν πρώτο βήμα, ο προγραμματιστής της εφαρμογής αφού εγκαταστήσει το Profiling και το Benchmarking Tool θα πρέπει να δημιουργήσει τοπικά μια εικονική μηχανή που περιλαμβάνει τα tests της συγκριτικής αξιολόγησης τα οποία έχουν εγκατασταθεί και εκτελούνται αυτοματοποιημένα μέσω του Benchmarking Controller. Καθώς εκτελούνται τα tests, το Profiling Tool παρακολουθεί και καταγράφει αυτοματοποιημένα την εκτέλεση των tests μέσω των εργαλείων Pidstat και Tshark για τα οποία θα γίνει εκτενής αναφορά στη συνέχεια.

Αυτή η σελίδα είναι σκόπιμα λευκή

Τόσο το Pidstat όσο και το Tshark είναι εγκατεστημένα στον ίδιο physical host με την εικονική μηχανή και χρησιμοποιούνται ώστε να καταγράψουν τον τρόπο με τον οποίο τα tests χρησιμοποιούν τους υπολογιστικούς πόρους και το δίκτυο.

Το επόμενο βήμα είναι η δημιουργία μιας δεύτερης εικονικής μηχανής τοπικά στον ίδιο physical host η οποία φιλοξενεί την εφαρμογή που είναι υπό εξέταση. Αυτή τη φορά το Profiling Tool εκτελείται προκειμένου να δημιουργηθεί το υπολογιστικό προφίλ της εφαρμογής, ακολουθώντας την ίδια διαδικασία που περιγράφηκε προηγουμένως (με τη χρήση του Pidstat και του Tskark). Τα δεδομένα που προκύπτουν από τις συγκεκριμένες διαδικασίες χρησιμοποιούνται σαν είσοδο στο Classification Tool που αποτελεί το κύριο εργαλείο του μηχανισμού.

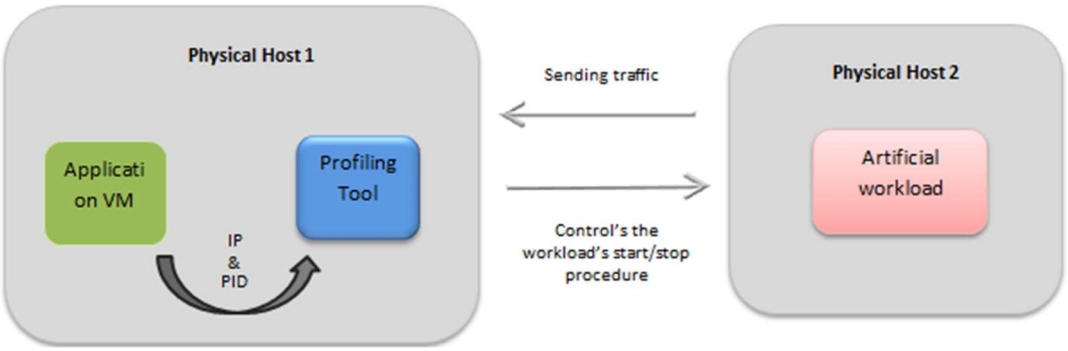


Σχήμα 18: Περιγραφή των δύο φάσεων του μηχανισμού

Αυτή η σελίδα είναι σκόπιμα λευκή

Έχοντας ο χρήστης τα συνολικά αποτελέσματα που παράγονται κατά τη διάρκεια των προηγούμενων σταδίων, μπορεί να προχωρήσει με την κατηγοριοποίηση της εφαρμογής του. Μέσω της γραφικής διεπαφής ο χρήστης επιλέγει τα αρχεία εισόδου. Έπειτα ο controller του Classification Tool στέλνει την πληροφορία στον Classifier που είναι υπεύθυνος για την αντιστοίχιση της εφαρμογής σε μια προκαθορισμένη κατηγορία εφαρμογής από την συγκριτική αξιολόγηση. Έχοντας τον ακριβή τύπο της εφαρμογής, ο controller είναι τώρα υπεύθυνος για να εντοπίσει την καταλληλότερη υπηρεσία του Νέφους ως προς την απόδοση και το κόστος με βάση τη χρήση της Service Efficiecny μετρικής. Ο κάθε χρήστης έχει τη δυνατότητα να διαμορφώσει τις τιμές στα βάρη της απόδοσης και του κόστους ανάλογα με την εκάστοτε επιθυμία του.

5.3.4 Αρχιτεκτονική του Profiling Tool



Σχήμα 19: Η επικοινωνία του Profiling Tool στο φυσικό επίπεδο

Το Profiling Tool, παρέχει ένα αυτοματοποιημένο τρόπο για την σκιαγράφηση του προφίλ τόσο της υπό εξέταση εφαρμογής όσο και των tests της συγκριτικής αξιολόγησης. Όπως έχει αναφερθεί προηγουμένως, παρουσιάζει δύο καταστάσεις λειτουργίας (application και benchmark mode) οι οποίες περιγράφονται σε φυσικό επίπεδο στο σχήμα 19. Όσον αφορά στην κατάσταση λειτουργίας «application», εάν η εφαρμογή χρησιμοποιεί κάποιο τεχνητό σύστημα παραγωγής φόρτου τότε το σύστημα αυτό θα πρέπει να εγκατασταθεί σε διαφορετικό περιβάλλον από εκείνο που φιλοξενείται το Profiling Tool και η εικονική μηχανή.

Αυτή η σελίδα είναι σκόπιμα λευκή

Αυτό είναι αναγκαίο για να εξασφαλιστεί ότι το τεχνητό σύστημα παραγωγής δεν προκαλεί παρεμβολές στα αποτελέσματα που λαμβάνονται από το Pidstat και το Tshark. Αυτός είναι ο λόγος για τον οποίο ο φόρτος κίνησης στην εφαρμογή αποστέλλεται (*Σχήμα 19*) από άλλο φυσικό περιβάλλον (Physical Host 2). Μόλις ολοκληρωθεί η διαδικασία της δημιουργίας των προφίλ, τα αποτελέσματα των εργαλείων παρακολούθησης Tshark και Pidstat μετσχηματίζονται σε κατάλληλη μορφή, προκειμένου να χρησιμοποιηθούν ως είσοδο στην επόμενη φάση, η οποία είναι η διαδικασία ταξινόμησης.

Όσον αφορά στην αρχιτεκτονική του Profiling Tool, το εργαλείο αυτό, αποτελείται από το User Interface (UI) δομικό στοιχείο, το οποίο απλοποιεί τη διαχείριση της σκιαγράφησης των προφίλ τόσο της εφαρμογής όσο και των tests της συγκριτικής αξιολόγησης. Το Profiling process controller είναι το κύριο δομικό στοιχείο του εργαλείου και είναι υπεύθυνο για το χειρισμό και το συγχρονισμό όλων των ενεργειών που εκτελούνται ταυτόχρονα κατά τη διαδικασία της σκιαγράφησης του προφίλ. Τα εργαλεία Pidstat και Tshark τα οποία θεωρούνται σαν εξωτερικά εργαλεία, χρησιμοποιούνται για την ανάλυση της απόδοσης μέσω της παρακολούθησης. Συγκεκριμένα το Pidstat χρησιμοποιείται για την παρακολούθηση των επιμέρους εργασιών μέσω του Process ID (PID) και επικεντρώνεται στη χρήση της CPU και στην αποθήκευση, ενώ το Tshark είναι υπεύθυνη για την παρακολούθηση του δικτύου μέσω της IP.

Στη συνέχεια παρουσιάζεται αναλυτική περιγραφή των βασικών δομικών στοιχείων του Profiling Tool καθώς και το διάγραμμα της αρχιτεκτονικής στην οποία βασίστηκε η υλοποίησή του.

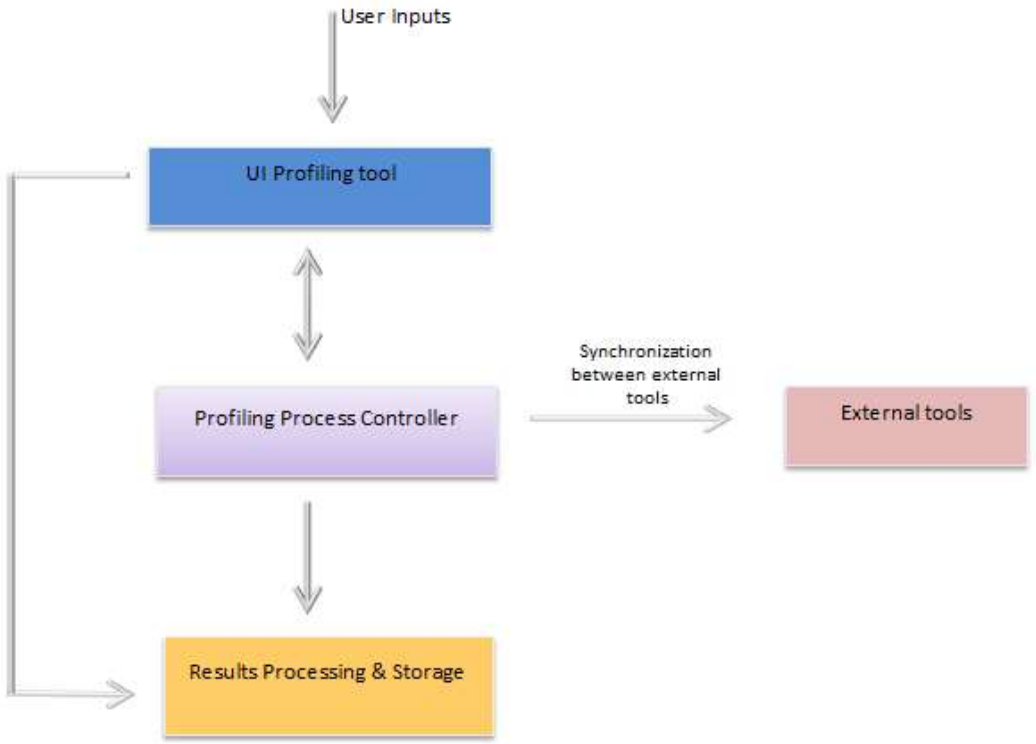
Το **User Interface (UI)** διευκολύνει την επικοινωνία μεταξύ του χρήστη και του εργαλείου και καθιστά την αλληλεπίδρασή τους γρήγορη και άμεση. Παρέχει ένα περιβάλλον εργασίας στον προγραμματιστή, προκειμένου να διαχειριστεί την όλη διαδικασία της σκιαγράφισης των

προφίλ της εφαρμογής και των tests της συγκριτικής αξιολόγησης. Κατά την εκτέλεση του Profiling Tool ζητείται από το χρήστη να συμπληρώσει μία σειρά από παραμέτρους όπως τον αριθμό του αναγνωριστικού της διαδικασίας της υπό εξέτασης εικονικής μηχανής, τα διαπιστευτήρια του χρήστη κτλ.

Ο Profiling process controller παρέχει το βασικό μηχανισμό για τον έλεγχο και το συγχρονισμό όλων των εργασιών που εκτελούνται ταυτόχρονα κατά τη διάρκεια της διαδικασίας της σκιαγράφισης των προφίλ. Πιο συγκεκριμένα, για την απόκτηση των επιθυμητών μετρήσεων, είναι σημαντικό η εικονική μηχανή εκτέλεσης, οι διαδικασίες Pidstat και Tshark και ο συλλέκτης αποτελεσμάτων να τρέχουν ταυτόχρονα με τον καλύτερο δυνατό τρόπο. Με αυτόν τον τρόπο, ο controller ελέγχει όλα τα εμπλεκόμενα μέρη και εποπτεύει αυτόματα όλη την εκτέλεση του πειράματος.

Σχετικά με τη λειτουργία του Profiling Tool, όπως ήδη έχει αναφερθεί υπάρχουν δύο διαφορετικοί τρόποι χρήσης (profiling και benchmark mode). Οι δύο αυτές καταστάσεις λειτουργίας παρουσιάζουν σημαντική διαφορά στο επίπεδο του αυτοματισμού μεταξύ τους. Ο λόγος είναι ότι στην περίπτωση της profiling κατάσταση λειτουργίας, η πληροφορία που δίνονται στο εργαλείο είναι πολύ περιορισμένη. Για παράδειγμα, πριν από την εκτέλεση, δεν υπάρχουν πληροφορίες σχετικά με τον τύπο εφαρμογής, το λειτουργικό σύστημα της εικονικής μηχανής, ο φόρτος εργασίας που χρησιμοποιείται για το πείραμα κ.λπ. Ως αποτέλεσμα, είναι ο χρήστης να συμβάλλει σε μεγάλο βαθμό στον έλεγχο και στον συγχρονισμό κατά τη διαδικασίας της σκιαγράφησης του προφίλ της εφαρμογής. Ωστόσο, η όλη διαδικασία απλοποιείται σημαντικά από το να εκτελούνταν χειροκίνητα, καθώς οι εντολές παρακολούθησης εξακολουθούν να διαχειρίζονται από το Profiling Tool. Στην περίπτωση της benchmarking λειτουργίας, η αυτοματοποίηση φτάνει σε υψηλότερο επίπεδο μιας και είναι

γνωστός ο τρόπος εκτέλεσης και λειτουργίας των benchmark tests και το μόνο που απαιτείται από το χρήστη είναι να τροφοδοτήσει το Profiling Tool με τις βασικές πληροφορίες.



Σχήμα 20: Τα υλοποιημένα components του Profiling Tool

Commands executor: Αυτό το δομικό στοιχείο χρησιμεύει ως μια βιβλιοθήκη η οποία αξιοποιείται πλήρως από τον controller για να εκτελέσει όλες τις απαραίτητες ενέργειες που σχετίζονται με τις διεργασίες ssh και UNIX.

Results processing and storage: Το δομικό αυτό στοιχείο έχει σχέση με το μετασχηματισμό των αποτελεσμάτων στην κατάλληλη μορφή. Μετά την εκτέλεση της σκιαγράφισης των προφίλ, τα αποτελέσματα των εργαλείων παρακολούθησης που χρησιμοποιούνται, δεν έχουν την κατάλληλη μορφοποίηση για την αυτοματοποιημένη επεξεργασία ή την αποθήκευση τους, η οποία τελικά παρέχεται από το συγκεκριμένο δομικό στοιχείο.

Αυτή η σελίδα είναι σκόπιμα λευκή

External Tools: Για τη διαδικασία της παρακολούθησης χρησιμοποιήθηκαν δύο εργαλεία το Pidstat από την σουίτα Sysstat και το Tshark που είναι ένα εργαλείο γραμμής εντολών για την παρακολούθηση του δικτύου. Αναλυτικά η Sysstat σουίτα είναι μια ομάδα από Linux εργαλεία γραμμής εντολών για την ανάλυση της απόδοσης και την παρακολούθηση.

Επιπλέον, τα εργαλεία αυτά είναι υπεύθυνα για τη συλλογή πληροφοριών του συστήματος, για την αποθήκευσή τους για κάποιο χρονικό διάστημα και για τον υπολογισμό των μέσων τιμών. Στην παρούσα διατριβή χρησιμοποιήθηκε η έκδοση 10.0.5.

Σχετικά με το εργαλείο Pidstat χρησιμοποιείται για την παρακολούθηση των επιμέρους εργασιών μέσω του αναγνωριστικού διεργασίας (Process Identifier). Αυτές οι διεργασίες διαχειρίζονται από το Linux Kernel . Ο ρόλος του είναι να καταγράψει τη δραστηριότητα για κάθε επιλεγμένη εργασία. Το Pidstat εκτελείται σε root περιβάλλον επομένως είναι αναγκαία τα διαπιστευτήρια του χρήστη τα οποία παρέχονται μέσω του User Interface όπως και κάθε άλλη απαραίτητη παράμετρος. Η εντολή και η περιγραφή των διεργασιών που επιλέχθηκαν να καταγραφούν παρουσιάζονται στους επόμενους πίνακες.

Εντολή Pidstat	
sudo Pidstat –urdw –p <INSERT_PID_OF_VM_PROCESSSES> <TIME_OF_ITERATIONS> <DURATION> >> <INSERT_OUUTPUT_FILE_NAME>	
%user	Ποσοστό χρήσης της CPU που σημειώθηκε κατά την εκτέλεση σε επίπεδο χρήστη (εφαρμογής). Περιλαμβάνει το χρόνο που δαπανάται για τη λειτουργία των εικονικών επεξεργαστών
%system	Ποσοστό χρήση της CPU που σημειώθηκε κατά την εκτέλεση σε επίπεδο συστήματος (πυρήνα). Περιλαμβάνει το χρόνο που δαπανάται για την εξυπηρέτηση των διακοπών του υλικού και του λογισμικού.

%guest	Ποσοστό του χρόνου που δαπανάται από την CPU ή τους επεξεργαστές για να τρέξει έναν εικονικό επεξεργαστή.
%CPU	Μέσος όρος της CPU για το χρόνο που ορίζεται από το διάστημα παραμέτρου (interval parameter)
CPU	Ο αριθμός του επεξεργαστή στο οποίο προσαρτάται η διεργασία.
kB_rd/s	Ο αριθμός των kilobytes που καταναλώθηκαν για να διαβαστεί η διεργασία από το δίσκο ανά δευτερόλεπτο.
kB_wr/s	Ο αριθμός των kilobytes το έργο έχει προκαλέσει, ή θα πρέπει να φροντίζει να γραφτεί στο δίσκο ανά δευτερόλεπτο.
kB_ccwr/s	Ο αριθμός των kilobytes των οποίων η καταγραφή στο δίσκο έχει ακυρωθεί από την διεργασία.
minflt/s	Ο συνολικός αριθμός των minor faults ανά δευτερόλεπτο που έχει κάνει η διεργασία, και δεν έχουν απαιτηθεί για τη φόρτωση μια σελίδας μνήμης από το δίσκο.
majflt/s	Ο συνολικός αριθμός των major faults ανά δευτερόλεπτο που έχει κάνει η διεργασία, και έχουν απαιτηθεί για τη φόρτωση μια σελίδας μνήμης από το δίσκο.
VSZ Virtual Size	Η χρήση της εικονικής μνήμης για ολόκληρη την διεργασία σε kilobyte.
RSS	Resident Set Size: Η non-swapped φυσική μνήμη που χρησιμοποιείται από τη διεργασία σε kilobytes.
%MEM	Το ποσοστό της φυσικής μνήμης που χρησιμοποιείται από τις τρέχουσες διεργασίες.
cswch/s	Ο συνολικός αριθμός των voluntary context switches ανά δευτερόλεπτο.
nvcswch/s	Ο συνολικός αριθμός των voluntary context switches ανά δευτερόλεπτο.

Πίνακας 9: Οι εντολές και η περιγραφή των διεργασιών που χρησιμοποιούνται από το Pidstat

Εντολές Tshark	
sudo -S Tshark -f "ip and (dst net <INSERT THE IP>)" -i <ETHERNET CARD> -w /tmp/TsharkDst.cap	
• sudo -S Tshark -f "ip and (src net <INSERT THE IP>)" -i <ETHERNET CARD> -w /tmp/TsharkSrc.cap	
sudo capinfos /tmp/<filename.cap	Το Tshark καταγράφει πληροφορίες όσον αφορά στον αριθμό των πακέτων, το μέσο μέγεθος του πακέτου και το ρυθμό.

-S	Separator
-f	capture filter
-i	capture interface
-w	Outfile
Dst	Destination
Src	Source

Πίνακας 10: Οι εντολές και η περιγραφή των διεργασιών που χρησιμοποιούνται από το Tshark

5.3.5 Αρχιτεκτονική του Classification Tool

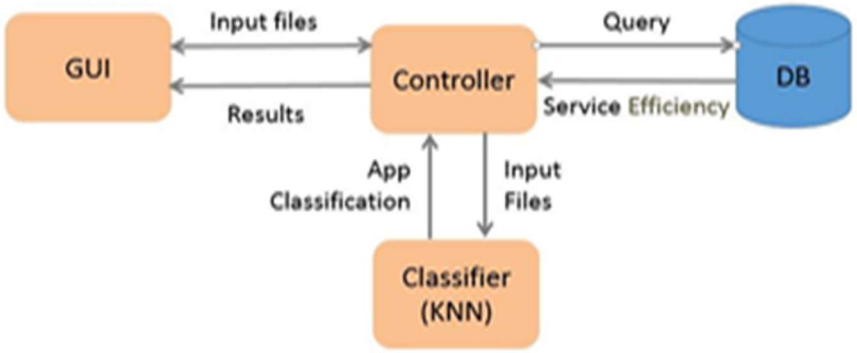
Το Classicifation Tool ταξινομεί ένα αυθαίρετο δομικό στοιχείο της εφαρμογή σε μια προκαθορισμένη και γνωστή κατηγορία εφαρμογών συγκριτικής ταξινόμησης και στη συνέχεια το αντιστοιχεί με μία υπηρεσία του Νέφους (με έναν τύπο στιγμιότυπου εικονικής μηχανής). Η συσχέτιση πραγματοποιείται με την επιλογή του τύπου στιγμιότυπου εικονικής μηχανής η οποία παρέχει την καλύτερη βαθμολογία επίδοσης για την εφαρμογή συγκριτικής αξιολόγησης που έχει ανιχνευθεί, σε συνδυασμό με το αντίστοιχο κόστος της υπηρεσίας που

προσφέρεται. Ο στόχος αυτού του εργαλείου είναι να προτείνει στον Προγραμματιστή Εφαρμογών την πιο κατάλληλη λύση με βάση συγκεκριμένα ενδιαφέροντα των χρηστών (όπως είναι η απόδοση και το κόστος) κατά τη διάρκεια της μετανάστευσης της εφαρμογής στο Νέφος. Το Classification Tool παρέχει τις ακόλουθες λειτουργίες:

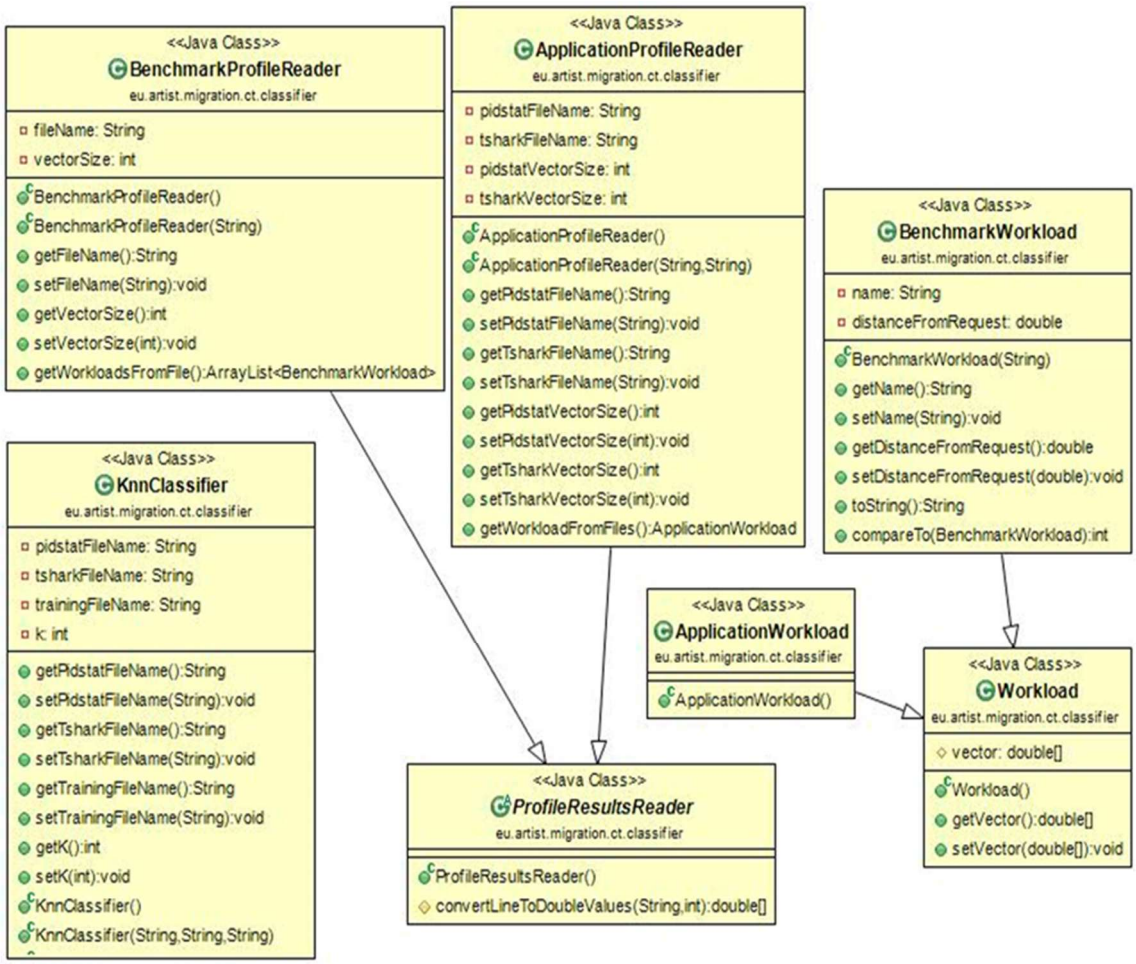
Ο *knn* ταξινομητής: κατατάσσει ένα δομικό στοιχείο της εφαρμογής υπολογίζοντας την Ευκλείδεια απόσταση μεταξύ των προφίλ της εφαρμογής και των εφαρμογών της συγκριτικής αξιολόγησης που παρέχονται μέσω του Profiling Tool. Η αντιστοίχιση του προφίλ της εφαρμογής σε ένα από τα προφίλ των εφαρμογών της συγκριτικής ταξινόμησης επιτυγχάνεται με την επιλογή της ελάχιστης απόστασης που ανιχνεύεται από τον αλγόριθμο *knn* (αλγόριθμος k-πλησιέστερων γειτόνων). Ο σκοπός αυτού του αλγορίθμου είναι να χρησιμοποιήσετε ένα σύνολο δεδομένων (τα προφίλ εφαρμογών συγκριτικής ταξινόμησης) στην οποία τα δεδομένα χωρίζονται σε αρκετές ξεχωριστές κατηγορίες σε μία από τις οποίες θα κατηγοριοποιηθεί ένα νέο δείγμα που στην προκειμένη περίπτωση είναι το προφίλ της εφαρμογής.

Έχοντας την ταξινόμηση ενός αυθαίρετου στοιχείου της εφαρμογής το εργαλείο μέσω του Classification Tool εντοπίζει την καλύτερη προσφορά εικονικής μηχανής με βάση το αποτέλεσμα της Service Efficiency μετρικής. Αυτή η μέτρηση παρέχει έναν τρόπο για τη μέτρηση των επιδόσεων των υπηρεσιών και κατάταξη των υπηρεσιών σύμφωνα με ένα σταθμισμένο συνδυασμό κόστους, απόδοσης και φόρτου εργασίας όπως ήδη έχει αναφερθεί. Προκειμένου να υπολογιστεί το αποτέλεσμα της Service Efficiency για κάθε προσφορά του Νέφους, είναι απαραίτητη η αποθήκευση των αποτελεσμάτων σε μία βάση δεδομένων. Αφού πραγματοποιηθούν και ολοκληρωθούν οι διαδικασίες της εύρεσης του προφίλ τόσο του αυθαίρετου component μιας εφαρμογής, όσο και των benchmark εφαρμογών, ο ιδιοκτήτης της εφαρμογής χρησιμοποιεί το Classification Tool, προκειμένου να αντιστοιχίσει το component της εφαρμογής σε μια προκαθορισμένη κατηγορία benchmark εφαρμογής και να εντοπίσει την

κατάλληλη υπηρεσία Νέφους χρησιμοποιώντας τη Service Efficiency μετρική όπως αναφέρθηκε νωρίτερα.



Πίνακας 11: Η αρχιτεκτονική των components του Classification Tool



Σχήμα 22:* Class *Diagram του Classifier.

Αυτή η σελίδα είναι σκόπιμα λευκή

Το Classification Tool αποτελείται από τρία components: το GUI το οποίο διευκολύνει τη διαχείριση της διαδικασίας ταξινόμησης, τον Classifier που υλοποιεί τον αλγόριθμο ταξινόμησης και τον Controller, ο οποίος είναι ο γενικός επόπτης της διαδικασίας και υπεύθυνος για την επικοινωνία με το σύστημα της βάσης δεδομένων για τον εντοπισμό της καλύτερης VM υπολογίζοντας τη SE μετρική. Μετά τη συγκέντρωση των αποτελεσμάτων που παράγονται κατά τη διάρκεια των προηγούμενων σταδίων (Benchmarking και Profiling διαδικασίες), ο χρήστης μπορεί να προχωρήσει με την κατάταξη μιας αυθαίρετης εφαρμογής (ή ενός component). Κατά την έναρξη της κατάταξης, ο Controller του Classification Tool περνάει τις σχετικές πληροφορίες στον Classifier ο οποίος είναι υπεύθυνος για την αντιστοίχιση της εφαρμογής σε μια προκαθορισμένη κατηγορία benchmark εφαρμογής. Έχοντας τον ακριβή τύπο της εφαρμογής, ο Controller αλληλεπιδρά με το σύστημα βάσης, μετά την απόκτηση του αποτελέσματος από τον Classifier, προκειμένου να μετρηθεί η βαθμολογία της SE η οποία βασίζεται στις κανονικοποιημένες τιμές τόσο του κόστους όσο και των μετρήσεων της απόδοσης. Τέλος, ο Controller επιστρέφει την κατάταξη με τις VMs που παρουσίασαν την υψηλότερη μέτρηση της SE μετρικής.

Αυτή η σελίδα είναι σκόπιμα λευκή

6 *Αξιολόγηση του μηχανισμού*

Στο συγκεκριμένο κεφάλαιο γίνεται αξιολόγηση της λειτουργίας και της απόδοσης του μηχανισμού εύρεσης του προφίλ και κατηγοριοποίησης μιας εφαρμογής σε μία γνωστή κατηγορία εφαρμογών, καθώς και της αποτελεσματικότητάς του με τη χρήση δύο εφαρμογών πραγματικού περιβάλλοντος.

6.1 Βήματα αξιολόγησης

Προκειμένου να αξιολογηθεί ο μηχανισμός που παρουσιάστηκε στο προηγούμενο κεφάλαιο, χρησιμοποιούνται τα αποτελέσματα που προέκυψαν από την αυτοματοποιημένη διαδικασία της συγκριτικής αξιολόγησης που παρουσιάστηκε στο κεφάλαιο 4. Τα αποτελέσματα για την απόδοση των τριών παρόχων αποθηκεύονται σε μία Raw Data ώστε να χρησιμοποιηθούν αργότερα κατά τη διάρκεια της κατηγοριοποίησης ενός αυθαίρετου component μιας εφαρμογής.

Στη συνέχεια επιλέγονται οι αυθαίρετες εφαρμογές οι οποίες θα χρησιμοποιηθούν για τη δοκιμή του μηχανισμού. Αναλυτική παρουσίαση των εφαρμογών αυτών πραγματοποιείται σε επόμενη ενότητα. Αρχικά με τη χρήση του Profiling Tool στο application mode πραγματοποιείται η σκιαγράφση του προφίλ της εφαρμογής ώστε να μελετηθεί ο τρόπος με τον οποίο η εφαρμογή χρησιμοποιεί τους υπολογιστικούς πόρους και το δίκτυο. Συγκεκριμένα η εφαρμογή έχει εγκατασταθεί μέσα σε μία εικονική μηχανή τοπικά σε έναν υπολογιστή. Στη συνέχεια, χρησιμοποιώντας και πάλι το Profiling Tool αλλά στο Benchmark mode αυτή τη

φορά, πραγματοποιούμε τη σκιαγράφηση των benchmarks που είναι τα ίδια με εκείνα τα οποία έχουν εφαρμοστεί για να μετρηθεί η απόδοση των παρόχων του Νέφους. Σημαντικό είναι να αναφερθεί πως τα benchmarks είναι εγκατεστημένα σε μία VM στο ίδιο physical host όπως και η εφαρμογή. Έχοντας τις παραπάνω πληροφορίες, ο ιδιοκτήτης της εφαρμογής μπορεί εφαρμόζοντας το Classification Tool να αρχίσει τη διαδικασία κατηγοριοποιήσης της. Τέλος, γνωρίζοντας σε ποια κατηγορία ανήκει η εφαρμογή το Classification Tool υπολογίζει τη καλύτερη προσφορά στο Νέφος ανάλογα με τις προτιμήσεις του χρήστη ως προς την απόδοση και το κόστος.

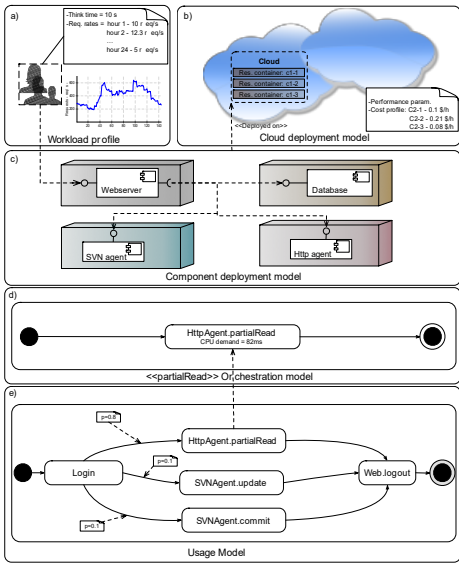
6.2 Μελέτη περίπτωσης: HTTPAgent εφαρμογή

Η πρώτη εφαρμογή που χρησιμοποιήθηκε πειραματικά για τη εφαρμογή της προτεινόμενης προσέγγισης είναι ένα component λογισμικού το οποίο ονομάζεται HTTPAgent και ανήκει στην web-based SaaS εφαρμογή μοντελοποίησης Modelio Constallation η οποία αναπτύχθηκε από τη Softeam [38].

6.2.1 Περιγραφή της Modelio Constallation εφαρμογής

Τα εκτεταμένα PCM μοντέλα (τμήμα της MODACloudsML) επιτρέπουν την αναπαράσταση της εφαρμογής από διαφορετική οπτική γωνία. Για παράδειγμα, το Σχήμα 21b παρουσιάζει ένα μοντέλο ανάπτυξης για την εφαρμογή. Η απλουστευμένη έκδοση της Constellation αποτελείται από 4 components. Συγκεκριμένα υπάρχει ένας Administrator Server που παρέχει μια WebService διεπαφή στους χρήστες μέσω ενός GUI ώστε να έχουν τη δυνατότητα να ανακτήσουν, να τροποποιήσουν και να ενημερώσουν τα διαθέσιμα έργα καθώς και να διαβάσουν τη ρύθμιση των παραμέτρων. Αυτό το component χρησιμοποιεί την Administration βάση δεδομένων για την αποθήκευση της πρόσβασης πολιτικών αδειών. Το SVN Agent

component χρησιμοποιεί το SVN ώστε πολλοί χρήστες να μπορούν να εργάζονται ταυτόχρονα για το ίδιο έργο.



Σχήμα 21: Constellation platform: Εκτεταμένο PCM στιγμιότυπο της εφαρμογής

Ωστόσο, για να μοιραστεί το προηγούμενο component ένα τμήμα του φόρτου, δημιουργήθηκε η HTTPAgent εφαρμογή η οποία παρέχει πρόσβαση στα μοντέλα μόνο για ανάγνωση. Η Constallation εφαρμογή μπορεί

να αναπτυχθεί τόσο σε δημόσια όσο και σε ιδιωτικά Νέφη και έχει σχεδιαστεί ώστε να μπορεί να δέχεται μεταβλητούς φόρτους εργασίας *Σχήμα 21b* και *30c*.

Για να γίνει καλύτερα κατανοητή η λειτουργία του σύστηματος χρησιμοποιήσαμε το Usage model το οποίο φαίνεται στο *Σχήμα 21.* Αυτό το μοντέλο αναπαριστά μια τυπική αλληλεπίδραση μεταξύ ενός χρήστη και του συστήματος στην περίπτωση όπου έχει ήδη ανακτήσει ένα αντίγραφο του μοντέλου πάνω στο οποίο θα εργαστεί. Στην εικόνα απεικονίζονται τμηματικά τρεις απλές πιθανές αλληλεπιδράσεις. Στο ανώτερο τμήμα, μετά την είσοδο του στο σύστημα, ο χρήστης αλληλεπιδρά με το σύστημα απαιτώντας μερική ανάγνωση του έργου. Αυτή η δραστηριότητα είναι μακράν η πιο κοινή και πραγματοποιείται στο 80% του χρόνου.

Αυτή η σελίδα είναι σκόπιμα λευκή

Πιο σπάνια ο χρήστης ενημερώνει το μοντέλο από το SVN (10%) ή εκτελεί κάποιες αλλαγές (10%). Τα σενάρια αλληλεπίδρασης και οι σχετικές πιθανότητες φαίνονται στο Σχήμα 21 έχουν συγκεντρωθεί από την τρέχουσα έκδοση του Modelio. Τέλος, το Orchestration model αναφέρει για κάθε λειτουργεία τη ζήτηση της CPU και (όπου απαιτείται) ένα γράφημα των κλήσεων ως προς άλλες λειτουργίες.

Στην συγκεκριμένη περίπτωση που εξετάζουμε, το HttpAgent.partialRead παρουσιάζει μια χρονική ζήτηση της τάξης των 82 ms. Αυτή η ζήτηση υπολογίζεται για την Amazon m1.large VM. Η μελέτη μας λαμβάνει υπόψη μόνο το HTTPAgent component καθώς, σύμφωνα με το μοντέλο χρήσης, το component αυτό θα πρέπει να χειριστεί το μεγαλύτερο μέρος του φόρτου εργασίας. Ως εκ τούτου, σημαντική είναι η επιλογή του καταλληλότερου είδους VM και συναφή αριθμό των αντιγράφων ώστε να ελαχιστοποιηθεί το κόστος εκτέλεσης.

6.2.2 Διαδικασία εύρεσης του προφίλ της HTTPAgent εφαρμογής

Πριν από την εκτέλεση της διαδικασίας εύρεσης του προφίλ, τόσο το Profiling Tool όσο και τα εξωτερικά εργαλεία που χρησιμοποιούνται (PIDstat και Tshark) εγκαθίστανται στο physical host. Η διαδικασία εύρεσης του προφίλ διεξήχθη σε τρία στάδια. Το πρώτο βήμα περιλαμβάνει την εγκατάσταση του HTTPAgent component σε μία VM. Σαν περιβάλλον εικονικοποίησης χρησιμοποιήθηκε το εργαλείο VMware Workstation 12 Player (Ubuntu (64-bit), 4Gb RAM και 40Gb storage Disk) το οποίο φιλοξενήθηκε τοπικά σε Linux περιβάλλον (Ubuntu 14.04), μ το Εργαλείο Profiling έχει σχεδιαστεί για Linux λειτουργικά συστήματα. Στο δεύτερο στάδιο, χρησιμοποιήθηκε το Apache Jmeter για τη τεχνητή δημιουργία φόρτου εργασίας. Το Apache Jmeter εγκαταστάθηκε σε μία VM, χρησιμοποιώντας την ίδια ρύθμιση παραμέτρων όπως και η εφαρμογή. Ωστόσο, το Jmeter εγκαταστάθηκε σε μία ξεχωριστή VM και φιλοξενήθηκε σε διαφορετικό φυσικό περιβάλλον, προκειμένου να αποφευχθεί

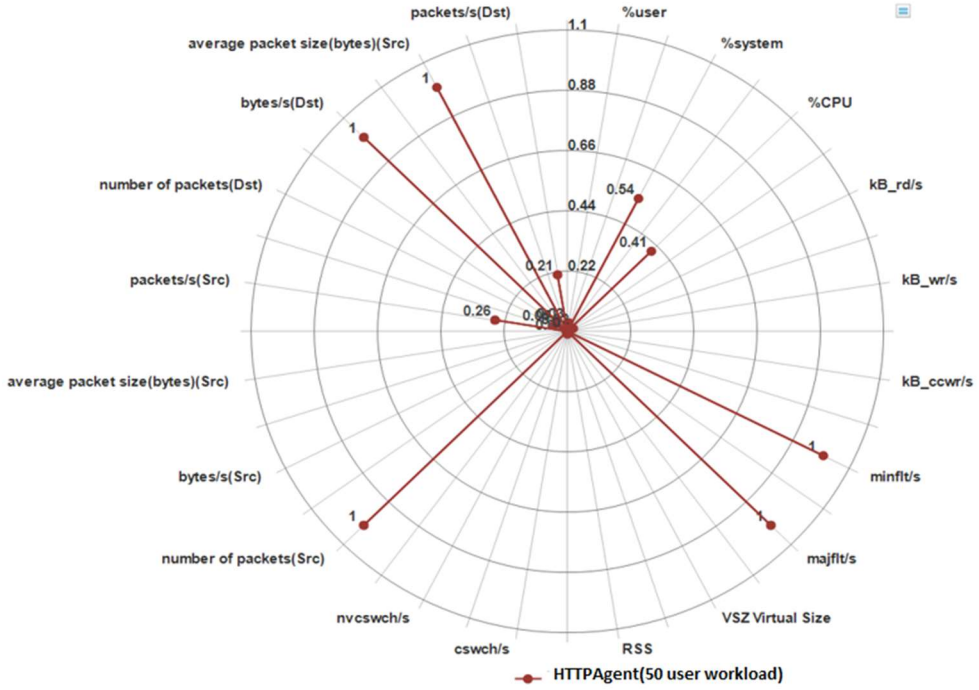
οποιαδήποτε παρεμβολή με τα αποτελέσματα που προκύπτουν από τα εργαλεία Pidstat και Tshark.

Στο τρίτο βήμα, το Jmeter μαζί με τα εργαλεία Pidstat και Tshark συγχρονίστηκαν μέσω του Profiling Tool, προκειμένου να δημιουργηθεί κίνηση προς την εφαρμογή αλλά και να καταγραφεί η απόδοση καθώς και ο τρόπος που η τελευταία χρησιμοποιεί τους πόρους.

Τέλος, τα αποτελέσματα που παρήχθησαν κατά τη διάρκεια του τρίτου βήματος υποβλήθηκαν σε επεξεργασία και, στη συνέχεια, αποθηκεύτηκαν με σκοπό να συγκριθούν με τα αποτελέσματα που προέκυψαν από τη φάση της συγκριτικής αξιολόγηση. Όσον αφορά τον τύπο των φορτίων που εξετάστηκαν στη διαδικασία εύρεσης του προφίλ και της κατηγοριοποίησης, πρόκειται για τυπικά σενάρια και όχι για τη χειρότερη περίπτωση χρόνου εκτέλεσης (Worst-case execution time-WCET) που θα ήταν υπερβολικά δαπανηρή. Ωστόσο, αν υπάρχουν συγκεκριμένα φορτία εργασίας τα οποία αντιστοιχούν σε μία εφαρμογή, προτείνεται να δημιουργούν από τον προγραμματιστή της εφαρμογής τεχνητά φορτία (π.χ. χρησιμοποιώντας το εργαλείο Apache Jmeter) βασιζόμενο σε μια κανονική αλληλουχία ενεργειών του χρήστη. Τέλος, τα αποτελέσματα που παράγονται κατά τη διάρκεια του τρίτου βήματος υποβάλλονται σε επεξεργασία και, στη συνέχεια, αποθηκεύονται με σκοπό να συγκριθούν με τα εξαγόμενα αποτελέσματα από τη συγκριτική αξιολόγηση φάση.

Όσον αφορά τον τύπο του workload που εκτελείται, το προφίλ και η ταξινόμηση γίνεται με βάση το τυπικό σενάριο και όχι τη χειρότερη περίπτωση χρόνου εκτέλεσης (WCET) που θα ήταν υπερβολικά δαπανηρή. Ωστόσο, αν υπάρχουν συγκεκριμένα workloads, συνιστούμε να δημιουργηθούν από τον προγραμματιστή της εφαρμογής τεχνητά workloads (π.χ. χρησιμοποιώντας το Apache Jmeter) ακολουθώντας μια κανονική αλληλουχία δράσεων από το χρήστη. Συγκεκριμένα, για την εφαρμογή HHTPAgent, δημιουργήθηκαν τα προφίλ από

τρεις διακριτούς φόρτους εργασίας, που δείχνουν την πραγματική χρήση από τους πελάτες κατά τη διάρκεια της κανονικής λειτουργίας.



Σχήμα 22: Διανυσματικό προφίλ της HTTPAgent εφαρμογής

Στη συνέχεια, κάθε ένα από τα προηγούμενα προφίλ χρησιμοποιήθηκε ως είσοδος στο Classification Tool για τη φάση της ταξινόμησης. Όσον αφορά το πείραμα για την HTTPAgent εφαρμογή, χρησημοποιήθηκαν τρία ρεαλιστικά σενάρια χρήσης με 50, 200 και 400 χρήστες στέλνοντας στον εξυπηρετητή HTTP αιτήματα, προκειμένου να ελεγχθεί η συμπεριφορά της εφαρμογής. Στο *Σχήμα 22* αναπαρίσταται ως διάνυσμα το προφίλ που δημιουργήθηκε για την HTTPAgent εφαρμογή με φόρτο αποτελλούμενο από 50 χρήστες αναπαρίσταται ως διάνυσμα.

6.2.1 Το διανυσματικό προφίλ των benchmark εφαρμογών

Όσον αφορά τη διαδικασία εύρεσης των προφίλ των benchmarks, οι εφαρμογές εγκαταστάθηκαν σε μία Centos 6.5 (x64) VM σε VMware 12 player. Για την αυτοματοποίηση της διαδικασίας της εγκατάστασης των benchmarks και την εκτέλεση τους, χρησιμοποιήθηκε η Benchmarking Suite καθώς και τα εργαλεία (Filebench, DaCapo και YCSB).

Ο συνολικός αριθμός των φορτίων εργασίας των benchmark εφαρμογών που συμμετείχαν στη διαδικασία εύρεσης του προφίλ ήταν δεκαοχτώ. Ωστόσο για να αυξηθεί η ακρίβεια και η αξιοπιστία της διαδικασίας κατηγοριοποίησης, για κάθε ένα από τα benchmark δημιουργήθηκε το προφίλ για περισσότερες από μία φορές. Σε κάθε επανάληψη παρήχθη ένα διάνυσμα με τα αποτελέσματα αποτελούμενο από τα υπολογιστικά ίχνη που δημιουργήθηκαν από τα εργαλεία Pidstat και Tshark. Η προαναφερθείσα διαδικασία είχε ως αποτέλεσμα τη δημιουργία ενός συνόλου 180 διανυσμάτων, που χρησιμοποιούνται ως είσοδοι για το Classification Tool. Μετά την εκτέλεση του Profiling Tool (benchmark mode), τα αποτελέσματα από τα background εργαλεία που χρησιμοποιούνται για την παρακολούθηση, αποθηκεύονται στο workspace του Profiling Tool.

6.2.2 Η ταξινόμηση της HTTPAgent εφαρμογής και η επιλογή παρόχου

Σχετικά με την ταξινόμηση της HTTPAgent εφαρμογής, τα αποτελέσματα που προέκυψαν από τη διαδικασία της Συγκριτικής Αξιολόγησης και της εύρεσης του προφίλ χρησιμοποιούνται ως είσοδοι στο Classification Tool. Μέσω του GUI ο χρήστης επιλέγει τα απαραίτητα αρχεία ώστε να αρχίσει η διαδικασία της ταξινόμησης. Ο ταξινομητής που χρησιμοποιείται είναι ο *knn*, ο οποίος ταξινομεί την HTTPAgent εφαρμογή υπολογίζοντας την απόσταση συνημιτόνου μεταξύ της εφαρμογής και των benchmark προφίλ. Προκειμένου να αντιμετωπιστεί το πρόβλημα της διαστατικότητας και να μειωθεί ο αριθμός των χαρακτηριστικών, ώστε να παραχθεί μια αξιόπιστη πρόβλεψη, χρησιμοποιήθηκε η συσχέτιση

Pearson [14] για τη μείωση των χαρακτηριστικών μιας και κάθε ένα προφίλ διανύσματος περιγράφεται από 23 χαρακτηριστικά.

Η αντιστοίχηση του προφίλ εφαρμογής σε ένα benchmark προφίλ επιτυγχάνεται με την επιλογή της ελάχιστης απόστασης που ανιχνεύεται από τον αλγόριθμο.

Σκοπός αυτού του αλγόριθμου είναι η χρήση ενός συνόλου δεδομένων (benchmark προφίλ) το οποίο αποτελείται από ένα σύνολο γνωστών κατηγοριών. Οι κατηγορίες αυτές χρησιμοποιούνται για την πρόβλεψη και την κατάταξη ενός νέου “σημείου” που στην συγκεκριμένη περίπτωση είναι το προφίλ της άγνωστης εφαρμογής. Στην περίπτωσή που μελετήσαμε, το προφίλ της HTTPgent εφαρμογής χαρακτηρίζεται μέσω της πλειοψηφίας των γειτόνων της και αντιστοιχίζεται με το benchmark προφίλ που παρουσιάζει τη μεγαλύτερη συχνότητα μεταξύ των *k* κοντινότερων γειτόνων. Στην περίπτωση που ένα ή περισσότερα benchmark προφίλ εμφανίζονται με την ίδια υψηλότερη συχνότητα, η επιλογή που "κερδίζει" προκύπτει είναι εκείνη παρουσιάζει την ελάχιστη μέση τιμή της ομοιότητας συνημίτονου μεταξύ του προφίλ εφαρμογής και του benchmark προφίλ.

Σύμφωνα με το Classification Tool, το υπολογιστικό προφίλ της HTTPAgent εφαρμογής στην περίπτωση ενός μικρού φόρτου εργασίας (50 χρήστες) συμπεριφέρεται παρόμοια με την tomcat εφαρμογή από τη DaCapo Suite, ενώ στην περίπτωση μεγαλύτερων φορτίων εργασίας δηλαδή για 200 και 400 χρήστες, η HTTPAgent συμπεριφέρεται ομοίως με την eclipse εφαρμογή επίσης από την DaCapo Suite. Σε αυτό το σημείο, είναι ενδιαφέρον να αναφερθεί η διπλή συμπεριφορά της εφαρμογής HTTPAgent ανάλογα με το είδος του φόρτου.

Στην προσέγγισή μας για να είναι εφικτός ο χειρισμός πολλαπλών ταξινομήσεων, ο ιδιοκτήτης της εφαρμογής θα πρέπει να γνωρίζει εκ των προτέρων την πραγματική χρήση της εφαρμογής και να παρέχει τα αντίστοιχα φορτία κατά τη φάση της διαδικασίας εύρεσης του προφίλ. Εφαρμόζοντας την SE μετρική με ποσοστό 50% -50% μεταξύ της απόδοσης και του κόστους,

καταλήξαμε στο συμπέρασμα ότι ο βέλτιστος τύπος VM για 50 χρήστες είναι η m1.small από την Amazon EC2, ενώ για βαρύτερα φορτία εργασίας είναι ο τύπος m1.medium παράδειγμα από τον ίδιο πάροχο. Όσον αφορά το κόστος υπολογισμού για τη φάση της κατηγοριοποίησης (χρησιμοποιώντας το Classification Tool), προτείνεται να προβλεφθεί η διαδικασία αυτή ως υπηρεσία από ένα Benchmark πάροχο, όπου ο χρόνος που χρειάζεται, εξαρτάται από το μέγεθος της βάσης δεδομένων που περιλαμβάνει τα αποθηκευμένα αποτελέσματα των benchmarks και το κόστος των VMs. Εκτενέστερη αναφορά για τη δημιουργία της προτεινόμενης υπηρεσίας από ένα Benchmark Provider περιλαμβάνεται στο κεφάλαιο 8.

6.2.3 Αξιολόγηση της μεθοδολογίας και του μηχανισμού με τη χρήση της HTTPAgent εφαρμογής

Σκοπός της διαδικασίας επικύρωσης ήταν να αποδειχθεί η αποτελεσματικότητα και η ακρίβεια της μεθοδολογίας μέσω του μηχανισμού που αναπτύχθηκε στην παρούσα διατριβή.

Όπως έχει ήδη αναφερθεί, χρησιμοποιώντας το Classification Tool οι προσφορές του Νέφους ταξινομήθηκαν σύμφωνα με τη SE μετρική, η οποία χρησιμοποιεί τα αποτελέσματα για την απόδοση όπως προκύπτουν από τη συγκριτική αξιολόγηση. Ειδικότερα, γνωρίζοντας ότι το υπολογιστικό προφίλ της HTTPAgent εφαρμογής είναι παρόμοιο με τις tomcat και eclipse benchmark εφαρμογές (τμήμα του διαδικασία ταξινόμησης), διαπιστώθηκε ότι η καλύτερη τιμή της SE μετρικής για αυτά τα benchmarks παρέχεται από την m1.small και m1.medium VMs αντίστοιχα. Οι *Πίνακας 12Πίνακας 13* περιλαμβάνουν όλες τις μετρούμενες τιμές της SE για τις tomcat και eclipse εφαρμογές που εκτελέστηκαν σε διάφορους τύπους VMs στην Amazon EC2, τη Microsoft Azure και τη Flexiant, επαληθεύοντας ότι η Amazon EC2 παρέχει τα καλύτερα

Benchmark application	VM Instance Type	Normalized Service Efficiency
tomcat DaCapo	amazon m1.small	0,3644
	Flexiant 2Gb-2CPU	0,226
	Amazon m1.medium	0,2169
	Azure A1	0,2134
	Azure A2	0,1913
	Amazon m1.large	0,1732
	Flexiant 1Gb-1CPU	0,1882
	Flexiant 4Gb-4CPU	0,1722
	Flexiant 4Gb-3CPU	0,1699

Πίνακας 12: Τα αποτελέσματα της SE για την tomcat εφαρμογή κατά την εκτέλεσή της σε όλους τους VM τύπους

αποτελέσματα για τη SE μετρική. Για να επικύρωση των αποτελεσμάτων του μηχανισμού, η πιο αποτελεσματική μέθοδος ήταν να εκτελεστεί η HTTPAgent σε διάφορους τύπους VMs της Amazon EC2. Όσον αφορά το Apache Jmeter, εγκαταστάθηκε επίσης στην Amazon EC2 σε m1.large VM, προκειμένου να αποφευχθεί η συμφόρηση και να διασφαλιστεί ότι οι απαιτούμενοι πόροι μπορούν να διατεθούν ώστε να εξυπηρετηθούν όλα τα νήματα που δημιουργούνται. Όταν η ανάπτυξη ολοκληρώθηκε, η εφαρμογή εξετάστηκε με τα σενάρια χρήστη που περιλάμβαναν τα ίδια φορτία με αυτά που χρησιμοποιήθηκαν στη διαδικασία δημιουργίας του προφίλ της HTTPAgent. Τα αποτελέσματα της επίδοσης από τις προαναφερθείσες διαδικασίες χρησιμοποιήθηκαν για τον υπολογισμό της κανονικοποιημένης SE για κάθε προσφορά του Νέφους. Αποφύγαμε να χρησιμοποιήσουμε στο διάστημα που χρησιμοποιήθηκε για την κανονικοποίηση την τιμή 0, καθώς αυτό σε ορισμένες περιπτώσεις μπορεί να οδηγήσει σε άπειρες τιμές. Τέλος, στον τύπο της SE μετρικής χρησιμοποιήθηκαν οι

ωριαίες τιμές για τους τύπους των VMs που ανακτήθηκαν από την επίσημη ιστοσελίδα της Amazon EC2.

Benchmark application	VM Instance Type	Normalized Service Efficiency
eclipse DaCapo	Amazon m1.medium	0,2586
	Flexiant 1Gb-1CPU	0,2451
	Azure A1	0,2401
	Flexiant 2Gb-2CPU	0,2387
	Amazon m1.small	0,2059
	Azure A2	0,1987
	Amazon m1.large	0,1854
	Flexiant 4Gb-4CPU	0,1623
	Flexiant 4Gb-3CPU	0,1306

Πίνακας 13: Τα αποτελέσματα της SE για την eclipse εφαρμογή κατά την εκτέλεσή της σε όλους τους VM τύπους

Ωστόσο, προκειμένου η επικύρωση να είναι ακριβής και πλήρης, υπολογίστηκε η SE μετρική για όλες τις benchmark εφαρμογές (*Πίνακας 14*) που χρησιμοποιήθηκαν στη διαδικασία της εύρεσης του προφίλ και της ταξινόμησης και επιλέχθηκαν εκείνες που παρουσίασαν την υψηλότερη βαθμολογία της SE για την m1.small VM για την περίπτωση 50 χρηστών και τη m1.medium VM για 200 και 400 χρήστες (*Πίνακας 15*). Στη συνέχεια ακολούθησε η σύγκριση τους με την SE μετρική για κάθε σενάριο χρήστη της HTTPAgent εφαρμογής. Από τη διαδικασία ταξινόμησης καταλήξαμε στο συμπέρασμα ότι για μικρά φορτία (50 χρήστες) η εφαρμογή μοιάζει με την εφαρμογή tomcat, ενώ για τα μεσαία και μεγάλα φόρτια (200 και 400 χρήστες) με την eclipse εφαρμογή. Από τον *Πίνακα 5,* οδηγούμαστε στο συμπέρασμα ότι

για ένα φορτίο 50 χρηστών θα πρέπει να χρησιμοποιηθεί μία m1.small VM ενώ για τις άλλες δύο περιπτώσεις μία m1.medium.

Benchmark application	VM Instance Type	Normalized Service Efficiency
tomcat DaCapo	m1.small	0,3644
fop DaCapo	m1.small	0,3759
h2 DaCapo	m1.small	0,4067
fileserver Filebench	m1.small	0,6704
Varmail Filebench	m1.small	0,6934
Videoserver Filebench	m1.small	0,8419
c YCSB	m1.medium	0,1919
f YCSB	m1.medium	0,1961
b YCSB	m1.medium	0,2041
d YCSB	m1.medium	0,2098
a YCSB	m1.medium	0,2351
eclipse DaCapo	m1.medium	0,2586
pmd DaCapo	m1.medium	0,2908
xalan DaCapo	m1.medium	0,2914
avrora DaCapo	m1.medium	0,3024
e YCSB	m1.medium	0,3024
jython DaCapo	m1.medium	0,3095
webproxy Filebench	m1.medium	0,3622

Πίνακας 14: Υψηλότερη βαθμολογία της SE μετρικής για τους m1.small και m1.medium τύπους VM

Προκειμένου να επικυρώσουμε την προσέγγιση μας, τα αντίστοιχα φορτία εκτελέστηκαν στους τρεις τύπους VMs ανά περίπτωση. Με τη μέτρηση της SE μετρικής από τον *Πίνακας 15* μπορούμε να συμπεράνουμε ότι η διαδικασία επικύρωσης ήταν επιτυχής, δεδομένου ότι οι

αρχικά επιλεγμένοι τύποι VMs παρουσίασαν τη βέλτιστη SE ανά φορτίο. Ένα άλλο συμπέρασμα που προκύπτει από τη σύγκριση των δεικτών της SE για την HTTPAgent εφαρμογή το ιδανικό βέλτιστο VM μέγεθος και workload είναι 50 χρήστες ανά small VM.

HTPAgent Workload	**VM Instance Type**	**Normalized Service Efficiency**	**Predicted Optimal VM from Profiling and Classification**
50 users	m1.large	0,2005	m1.small
	m1.medium	0,2818	
	m1.small	**0,2961**	
200 users	m1.large	0,1875	m1.medium
	m1.medium	**0,2751**	
	m1.small	0,2026	
400 users	m1.large	0,2019	m1.medium
	m1.medium	**0,2818**	
	m1.small	0,2005	

Πίνακας 15: Κανονικοποιημένη SE και πρόβλεψη βέλτιστης VM από την προτεινόμενη μεθοδολογία

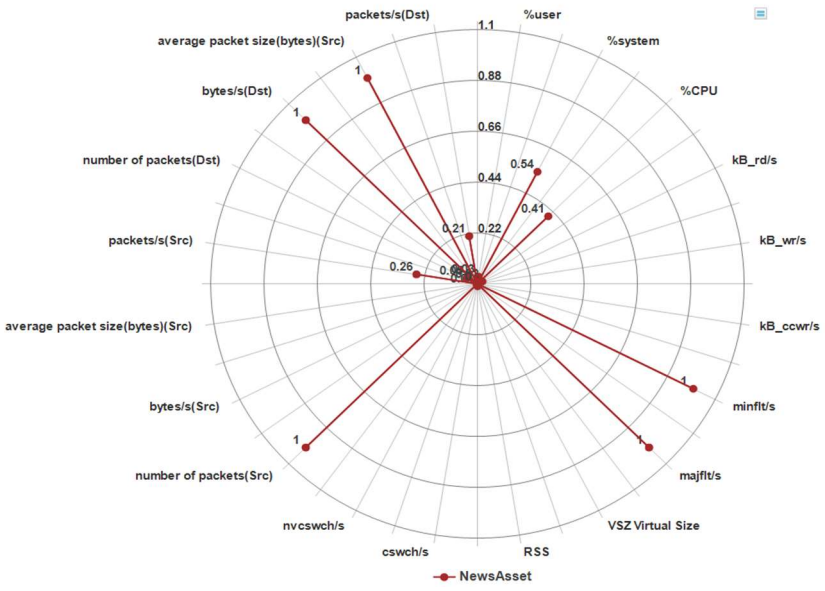
6.3 Μελέτη περίπτωσης: NewsAsset εφαρμογή

Για τη διαδικασία της αξιολόγησης του μηχανισμού, η δεύτερη εφαρμογή που χρησιμοποιείται ονομάζεται NewsAsset και είναι μία end-to-end multimedia διακαναλική λύση για τα Πρακτορεία Ειδήσεων, τους ραδιοτηλεοπτικούς φορείς και τους εκδότες. Πρόκειται για ένα εμπορικό προϊόν που διατίθεται στο εμπόριο από την ATC και χρησιμοποιείται από τα εθνικά Πρακτορεία Ειδήσεων της Ελλάδας αλλά και του εξωτερικού. Πρόκειται για μία σπονδυλωτή, παραμετροποιήσιμη, multimedia λύση για Πρακτορεία Ειδήσεων. Έχει κατασκευαστεί για να καλύψει τις ανάγκες ενός εξελισσόμενου Πρακτορείου Ειδήσεων για να βοηθήσει το

σχεδιασμό, τη δημιουργία, τη διαχείριση και τη διανομή των έκτακων είδησεων γρήγορα και αποτελεσματικά σε ένα ευρύ φάσμα πελατών μέσω πολλαπλών καναλιών διανομής. Παρέχει τη βασική λειτουργικότητα που απαιτείται για τη διαχείριση των γεγονότων και το σχεδιασμό σύνταξης, τη δημιουργία περιεχομένου, τη συγκέντρωση, την παραγωγή, την αρχειοθέτηση, τη διανομή και τη δημοσίευση μέσω πολλαπλών καναλιών. Το NewsAsset υποστηρίζει το σύνολο του κύκλου ζωής της είδησης από τον προγραμματισμό, μέσω της δημιουργίας, της συλλογής και της επεξεργασίας, εως την παραγωγή, τη διανομή και την αρχειοθέτηση. Περισσότερες πληροφορίες για τη συγκεκριμένη εφαρμογή μπορούν να βρεθούν στο [174].

6.3.1 Διαδικασία εύρεσης του προφίλ της NewsAsset εφαρμογής

Η σκιαγράφηση του προφίλ της NewsAsset εφαρμογής δειξάγεται σε τρία στάδια. Το πρώτο βήμα είναι η εγκατάσταση της σε μια εικονική μηχανή (VMware περιβάλλον εικονικοποίησης) που φιλοξενείται σε Linux περιβάλλον (Ubuntu 14.04), μιας και το Profiling Tool έχει σχεδιαστεί για λειτουργικά συστήματα Linux. Ωστόσο, κάθε χρήστης είναι υπεύθυνος να παρέχει την εφαρμογή με την κατάλληλη ρύθμιση παραμέτρων.



Σχήμα 23: Το διανυσματικό προφίλ της NewsAsset εφαρμογής

Αυτή η σελίδα είναι σκόπιμα λευκή

Στο δεύτερο βήμα, χρησιμοποιείται ένα εργαλείο για την παραγωγή τεχνητού φόρτου εργασίας που αντιστοιχεί σε ένα ρεαλιστικό σενάριο. Το εργαλείο φόρτου εργασίας εγκαταθίσταται σε μια VM με τις ίδιες ρυθμίσεις παραμέτρων όπως αυτή που χρησιμοποιήθηκε και στην εφαρμογή. Επιπλέον, η VM με τη γεννήτρια παραγωγής φόρτου εργασίας φιλοξενείται σε ένα διαφορετικό φυσικό περιβάλλον, προκειμένου να διασφαλιστεί ότι δεν θα παρεμβαίνει με τα αποτελέσματα του Pidstat και Tshark. (Και τα δύο αυτά βοηθητικά εργαλεία που χρησιμοποιούνται για την ανάλυση και την παρακολούθηση των επιδόσεων ελέγχονται από το Profiling Tool). Τέλος, τα αποτελέσματα που προέρχονται από το δεύτερο στάδιο μετά από επεξεργασία στη συνέχεια αποθηκεύονται προκειμένου να συγκριθεί με τα εξαγόμενα αποτελέσματα από τη διαδικασία της σκιαγράφησης του προφίλ των benchmarks. Στην NewsAsset εφαρμόστηκαν δύο σενάρια χρήστη με 40 και 100 χρήστες αντίστοιχα προκειμένου να ελεγχθεί η συμπεριφορά της εφαρμογής. Τα σενάρια του φόρτου εργασίας αντανακλούν όλες τις πιθανές ενέργειες που είναι διαθέσιμες για 40 χρήστες (δημοσιογράφοι) που εργάζονται ταυτόχρονα (ενημέρωση, δημιουργία, διαγράφη, πρόσθεση συνημμένων). Ένα παράδειγμα από το NewsAsset προφίλ που δημιουργήθηκε με τη βοήθεια του Profiling Tool παρουσιάζεται στο Σχήμα 23 και αναπαρίσταται σαν ένα διάνυσμα. Συγκεκριμένα το προφίλ αυτό αναφέρεται σε 40 χρήστες οι οποίοι εργάζονται ταυτόχρονα και εκτελούν μια σειρά από εργασίες (updates, creates, deletes, add attachements, get attachements) για 20 επαναλήψεις (loops).

6.3.2 Η ταξινόμηση της NewsAsset εφαρμογής και η επιλογή παρόχου

Σχετικά με την ταξινόμηση της NewsAsset εφαρμογής, ακολουθείται ακριβώς η ίδια διαδικασία που εφαρμόστηκε και στην περίπτωση της HTTPAgent εφαρμογής. Σύμφωνα με αυτή, τα αποτελέσματα που προέκυψαν από τη διαδικασία της συγκριτικής αξιολόγησης και

της εύρεσης του προφίλ χρησιμοποιούνται ως είσοδοι στο Classification Tool. Στη συνέχεια με τη χρήση του *knn αλγορίθμου*, η εφαρμογή αντιστοιχίζεται σε μία benchmark εφαρμογή υπολογίζοντας την απόσταση συνημιτόνου μεταξύ της εφαρμογής και των benchmark προφίλ. Η αντιστοίχιση του προφίλ εφαρμογής σε ένα benchmark προφίλ επιτυγχάνεται με την επιλογή της ελάχιστης απόστασης που ανιχνεύεται από τον αλγόριθμο.

Σύμφωνα με το Classification Tool, το υπολογιστικό προφίλ της NewsAsset εφαρμογής στην περίπτωση τόσο στην περίπτωση ενός μικρού φορτίου εργασίας (40 χρήστες) όσο και στην στην περίπτωση ενός μεγαλύτερου συμπεριφέρεται παρόμοια με την webproxy benchmark εφαρμογή από τη Filebench Suite. Εφαρμόζοντας την SE μετρική με ποσοστό 50% -50% μεταξύ της απόδοσης και του κόστους, καταλήξαμε στο συμπέρασμα ότι ο βέλτιστος τύπος VM και στις δύο περιπτώσεις είναι η m1.medium από την Amazon EC2.

6.3.3 NewsAsset αποτελέσματα για την επικύρωση της μεθοδολογίας και του μηχανισμού

Σκοπός της διαδικασίας επικύρωσης ήταν να αποδειχθεί η αποτελεσματικότητα και η ακρίβεια της μεθοδολογίας μέσω του μηχανισμού που αναπτύχθηκε στην παρούσα διατριβή.

Όπως και στην περίπτωση της HTTPAgent, επιλέχθηκε ο Amazon EC2 πάροχος για το deployment της NewsAsset εφαρμογής. Αρχικά η εφαρμογή εγκαταστάθηκε σε ένα m1.large VM (90GiB) όπως συνέστησε ο προγραμματιστής λογισμικού. Σχετικά με την γεννήτρια φορτίου Workload Generator εγκαταστάθηκε σε μία m1.large VM (60 GiB) ώστε να αποφευχθεί η συμφόρηση και να διασφαλιστεί ότι οι απαιτούμενοι πόροι μπορούν να διατεθούν για όλα τα νήματα που δημιουργούνται από το εργαλείο.

Όταν η διαδικασία της εγκατάστασης ολοκληρώθηκε, η εφαρμογή δοκιμάστηκε με τα ίδια φορτία (σενάρια χρήστη) τα οποία χρησιμοποιήθηκαν στη διαδικασία της δημιουργίας του

προφίλ της NewsAsset εφαρμογής. Όταν η εκτέλεση ολοκληρώθηκε η εφαρμογή εγκαταστάθηκε σε άλλες δύο VMs (m1.medium και m1.small) προκειμένου να συγκριθεί η απόδοση της εφαρμογής σε δάφορους τύπους. Τα αποτελέσματα για την απόδοση από τις παραπάνω διαδικασίες χρησιμοποιήθηκαν για να υπολογιστεί η κανονικοποιημένη SE μετρική. Η τιμή του συντελεστή βάρους που δόθηκε στη απόδοση και στο κόστος ήταν 0.5 και 0.5 αντίστοιχα ως η πιο λογική εκδοχή και ως διάστημα κανονικοποίησης ορίστηκε το (1-10). . Ειδικότερα, γνωρίζοντας ότι το υπολογιστικό προφίλ της NewsAsset εφαρμογής είναι παρόμοιο με την webproxy εφαρμογή (τμήμα του διαδικασία ταξινόμησης), διαπιστώθηκε ότι η καλύτερη τιμή της SE μετρικής για αυτό το benchmark παρέχεται από την m1.medium VM. Ο *Πίνακας 16* περιλαμβάνει όλες τις μετρούμενες τιμές της SE για την webproxy εφαρμογή που εκτελέστηκε σε διάφορους τύπους VMs στην Amazon EC2, τη Microsoft Azure και τη Flexiant, επαληθεύοντας ότι η Amazon EC2 παρέχει τα καλύτερα αποτελέσματα για τη SE μετρική.

Benchmark application	**VM Instance Type**	**Normalized Service Efficiency**
webproxy Filebench	Amazon m1.medium	0,3622
	amazon m1.small	0,2341
	Azure A1	0,3206
	Azure A2	0,2574
	Flexiant 2Gb-2CPU	0,2523
	Flexiant 4Gb-4CPU	0,2066
	Amazon m1.large	0,1750
	Flexiant 4Gb-3CPU	0,1723
	Flexiant 1Gb-1CPU	0,16403

Πίνακας 16: Μετρούμενες τιμές της SE για την webproxy εφαρμογή που εκτελέστηκε σε διάφορους τύπους VMs σε διαφορετικούς παρόχους

Επιπλέον, στην περίπτωση της NewsAsset εφαρμογής στα *Σχήμα 24Σχήμα 25* παρουσιάζονται επίσης τα αποτελέσματα της SE στην περίπτωση που οι συντελεστές βάρους είναι 0.9 για το κόστος και 0.1 για την απόδοση, θέλοντας να εξεταστεί και αυτή η πιθανή επιλογή από την πλευρά του χρήστη. Αποφύγαμε όπως και στην περίπτωση της HTTPAgent εφαρμογής να χρησιμοποιήσουμε διάστημα κανονικοποίησης την τιμή 0, καθώς αυτό σε μερικές περιπτώσεις ίσως να οδηγήσει σε άπειρες τιμές. Επιπλέον, στον πίνακα συμπεριλαμβάνονται και τα αντίστοιχα κόστη ανα χρήστη. Μία μελλοντική αξιοποίηση της πληροφορίας του κόστους θα μπορούσε να είναι η εξής: Αφού εντοπιστούν τα βέλτιστα μεγέθη ανα αριθμό χρηστών για διαφορετικά επίπεδα χρόνου απόκρισης, θα μπορούσαν στη συνέχεια να αντιστοιχίζονται με διαφορετικού τύπου SLA (gold, silver, bronze) τα οποία θα διατίθενται στους χρήστες της υπηρεσίας. Από το *Σχήμα 24* οδηγούμαστε στο συμπέρασμα πως στην περίπτωση που επιλεχθούν συντελεστές βάρους 0.9 και 0.1 για το κόστος και την απόδοση αντίστοιχα, προκύπτει όπως ήταν αναμενόμενο ότι το καλύτερο αποτέλεσμα για τη SE για 40 χρήστες δίνεται από τη VM m1.small. Όσον αφορά το κόστος σε συνδυασμό με την απόδοση η επιλογή μιας m1.medium VM συγκριτικά με μία m1.small παρέχει μια σημαντική αύξηση της απόδοσης της τάξης του 15.1%, ενώ το κόστος σχεδόν διπλασιάζεται. Ωστόσο, μια ιδιαίτερα σημαντική παρατήρηση είναι ένα μη αναμενόμενο αποτέλεσμα για την απόδοση της m1.large VM. Η επιλογή μιας m1.large σε σχέση με μία m1.medium VM παρουσιάζει μείωση της απόδοσης (-1.4%), ενώ το αντίστοιχο κόστος διπλασιάζεται (50.8%). Αυτό πιθανότατα να οφείλεται στο μικρό μέγεθος του φορτίου αλλά και στη διακύμανση της απόδοσης της υποδομής Νέφους.

NewsAsset workload	VM Instance Type	Normalized Service Efficiency(50%cost-50%performance)	Normalized Service Efficiency(90%cost-10%performance)	Response Time	Cost per user session(s)
40 threads	m1.small	0,1702127	0,3883495	1546	0,00080520
				15,1%	40,1%
40 threads	m1.medium	0,3623693	0,239963	1300	0,00134513
				-1,4%	50,8%
40 threads	m1.large	0,1647133	0,108527	1319	0,00273875

Σχήμα 24: Αποτελέσματα αξιολόγησης από τη διαδικασία της ταξινόμησης για τη βέλτιστη επιλογή VM για την εφαρμογή News Asset 40 χρηστών

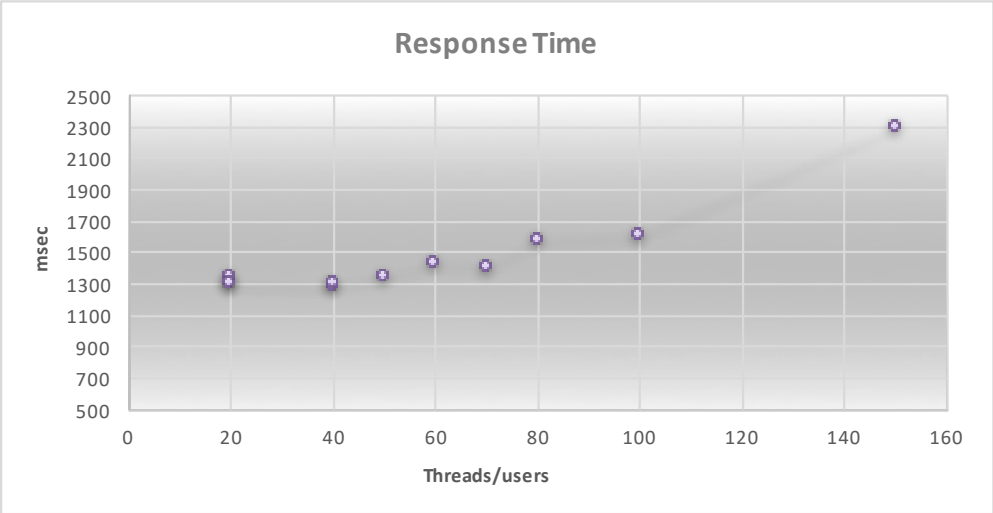
Στην περίπτωση των 100 χρηστών παρατηρούμε πως όπως προέκυψε και από τη διαδικασία της κατηγοριοποίησης το καλύτερο αποτέλεσμα δίνεται από τη m1.medium VM στην περίπτωση που επιλεχθεί ο συνδυασμός βαρών 0.5 κόστος και 0.5 απόδοση, ενώ η m1.small VM όταν τα βάρη αλλάξουν σε 0.9 κόστος και 0.1 απόδοση. Όσον αφορά στη μεταβολή της απόδοσης και του κόστους ανάλογα με την επιλογή VM είναι φανερό ότι η απόδοση διπλασιάζεται στην περίπτωση που επιλεγεί μία m1.medium, ενώ το κόστος αυξάνεται ελάχιστα. Αντίθετα η επιλογή μιας m1.large VM οδηγεί σε μία αύξηση της απόδοσης της τάξης του 14% ενώ το κόστος σχεδόν διπλασιάζεται.

NewsAsset workload	VM Instance Type	Normalized Service Efficiency(50%cost-50%performance)	Normalized Service Efficiency(90%cost-10%performance)	Response Time	Cost per user session(s)
100 threads and 20 loops	m1.small	0,1702127	0,38834951	3235	0,00067396
				50,5%	1,6%
100 threads and 20 loops	m1.medium	0,3567221	0,23380074	1602	0,00066305
				14%	41,9%
100 threads and 20 loops	m1.large	0,1818181	0,10989011	1375	0,001142014

Σχήμα 25: Αποτελέσματα αξιολόγησης από τη διαδικασία της ταξινόμησης για τη βέλτιστη επιλογή VM για την εφαρμογή News Asset 100 χρηστών

Αυτή η σελίδα είναι σκόπιμα λευκή

Στο *Σχήμα 26* παρουσιάζονται τα αποτελέσματα από την εκτέλεση της NewsAsset εφαρμογής σε μία medium VM χρησιμοποιώντας διαφορετικό αριθμό χρηστών ακολουθώντας το ίδιο σενάριο διεργασιών. Παρατηρούμε λοιπόν πως ο χρόνος απόκρισης μέχρι 100 χρήστες παρουσιάζει μικρή αύξηση, ενώ για 150 χρήστες παρουσιάζει έντονη μεταβολή. Μία τέτοια διαπίστωση θα μπορούσε να συμβάλει στον εντοπισμό και στη διαμόρφωση της κοστολόγησης από την πλευρά του παρόχου.



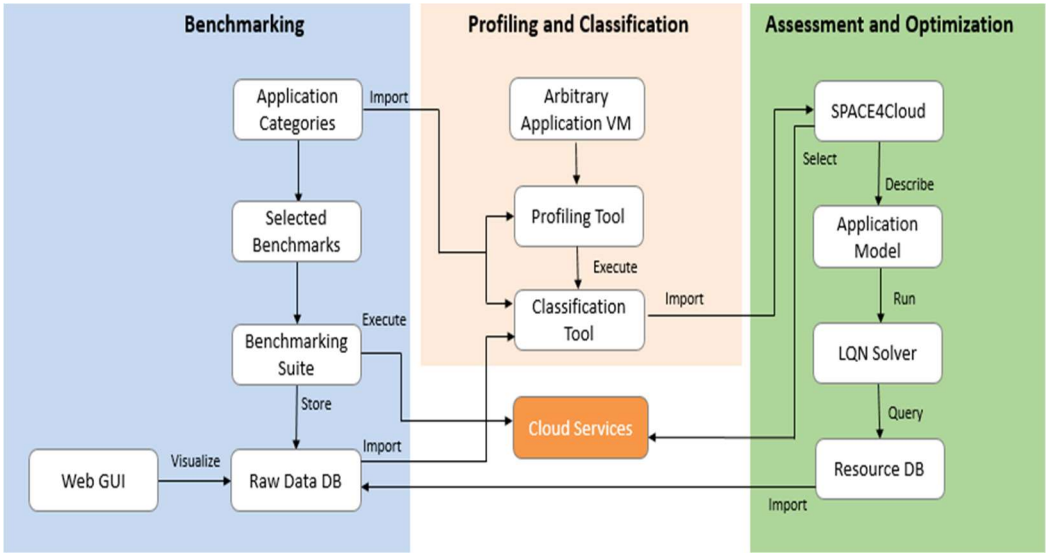
Σχήμα 26: Εκτέλεση της NewsAsset εφαρμογής σε μία medium VM για διαφορετικό αριθμό χρηστών

6.1 Εφαρμογή της προτεινόμενης μεθοδολογίας σε συνδυαστικό μηχανισμό για την εύρεση της βέλτιστης υπηρεσίας Νέφους

Στη συγκεκριμένη ενότητα αναλύεται η δυνατότητα εφαρμογής της μεθοδολογίας που αναπτύχθηκε στην παρούσα διατριβή συνδυαστικά με ένα εργαλείο για την εξερεύνηση του χώρου σχεδίασης με βέλτιστο τρόπο, το οποίο μπορεί να εντοπίσει αποτελεσματικά τη λύση με το ελάχιστο κόστος, λαμβάνοντας υπ’ όψη τις αλλαγές του φορτίου εργασίας παρέχοντας QoS εγγυήσεις. Δεδομένης της πολυπλοκότητας των συστημάτων λογισμικού αλλά και της ποικιλίας των υπηρεσιών του Νέφους σε συνδυασμό με τα μοντέλα τιμολόγησης που

Αυτή η σελίδα είναι σκόπιμα λευκή

διατίθενται σήμερα στην αγορά, είναι εξαιρετικά περίπλοκο βρεθεί η βέλτιστη λύση που ικανοποιεί τις απαιτήσεις της εφαρμογής και προσφέρει τον βέλτιστο συνδυασμό κόστους και QoS, μιας και ο σχεδιαστής της εφαρμογής θα πρέπει να αξιολογήσει ένα συνδυαστικά αυξανόμενο αριθμό εναλλακτικών λύσεων σχεδιασμού [175].



Σχήμα 27: Η συνδυαστική μεθοδολογία του μηχανισμού

Η προτεινόμενη μεθοδολογία και τα εργαλεία από τα οποία αποτελείται παρουσιάζονται στο Σχήμα 26. Η προσέγγισή μας μαζί με τα εργαλεία υποστήριξης, επιτρέπει στις εφαρμογές να εντοπίσουν τα οφέλη των υπηρεσιών του Νέφους από άποψη απόδοσης και κόστους λαμβάνοντας υπ' όψη την υπολογιστική τους συμπεριφορά μέσω της διαδικασίας εύρεσης του προφίλ. Η συνδυαστική μεθοδολογία περιλαμβάνει τρεις φάσεις: (i) Τη συγκριτική αξιολόγηση (ii) την εύρεση του προφίλ και την κατηγοριοποίση και τέλος (iii) την αξιολόγηση και τη φάση βελτιστοποίησης. Αρχικά όπως ήδη έχει αναφερθεί και στη μεθοδολογία που προτείνεται στην παρούσα διατριβή επιλέγονται οι benchmark εφαρμογές, ενώ στη συνέχεια πραγματοποιείται μία σειρά από πειράματα για την μέτρηση της απόδοσης των παρόχων του Νέφους, με αυτοματοποιημένο τρόπο μέσω της χρήσης της Benchmarking Suite και τα αποτελέσματά τους χρησιμοποιούνται αποθηκεύονται σε μία βάση ώστε να είναι διαθέσιμα αργότερα για επεξεργασία. Στη συνέχεια δημιουργούνται τα προφίλ των VMs με την

Αυτή η σελίδα είναι σκόπιμα λευκή

εφαρμογή και τα benchmarks με τη χρήση του Profiling Tool τα οποία στη συνέχεια χρησιμοποιούνται ως είσοδος στο Classification Tool. Το τελευταίο εντοπίζει τη βέλτιστη λύση της υπηρεσίας του Νέφους για τη συγκεκριμένη εφαρμογή ως προς την απόδοση και το κόστος περιορίζοντας τις εναλλακτικές λύσεις που πρόκειται να χρησιμοποιηθούν στη φάση της Βελτιστοποίησης.

Στη συνέχεια τα αποτελέσματα των benchmark μετρήσεων που βρίσκονται αποθηκευμένα στη βάση μαζί με τους υποψήφιους τύπους των υπηρεσιών Νέφους που προέκυψαν από τη διαδικασία της κατηγοριποίησης, χρησιμοποιούνται ως είσοδοι στο SPACE4Cloud μηχανισμό ώστε να υπολογιστεί και να αξιολογηθεί το πώς μεταβάλλονται οι μετρήσεις της απόδοσης της εφαρμογής χρησιμοποιώντας διαφορετικούς τύπους και μεγέθη πόρων για το υποσύνολο των παρόχων του Νέφους που θεωρούνται ως τελικοί στόχοι για την ανάπτυξη της εφαρμογής.

6.1.1 Design-time exploration μέσω του εργαλείου SPACE4Cloud

Η εξερεύνηση του χρόνου σχεδίασης υποστηρίζεται από το εργαλείο SPACE4Cloud (System Performance and Cost Evaluation on Cloud), το οποίο πρόκειται για μία εφαρμογή ανοικτού κώδικα για την αξιολόγηση και τη βελτιστοποίηση των QoS ιδιοτήτων των εφαρμογών του Νέφους. Ειδικότερα, το εργαλείο αυτό επιτρέπει στους αρχιτέκτονες λογισμικού για να περιγράψουν, να αναλύσουν και να βελτιστοποιήσουν τις εφαρμογές του Νέφους ακολουθώντας την οδηγούμενη από μοντέλα προσέγγιση (Model-Driven Approach). Η γλώσσα μοντελοποίησης που υποστηρίζεται από το SPACE4Cloud είναι η MODACloudsML, η οποία έχει επινοηθεί για να περιγράψει την αρχιτεκτονική του Νέφους εκφράζοντας συγκεκριμένα χαρακτηριστικά του. Ανάμεσα σε άλλα, η MODACloudsML [176] γλώσσα περιλαμβάνει αρχιτεκτονικούς και QoS περιορισμούς (όπως τη χρήση των VMs ή τη μέση τιμή της απόκρισης της εφαρμογής βάσει ενός κατώτατου ορίου) καθώς και ένα φορτίο

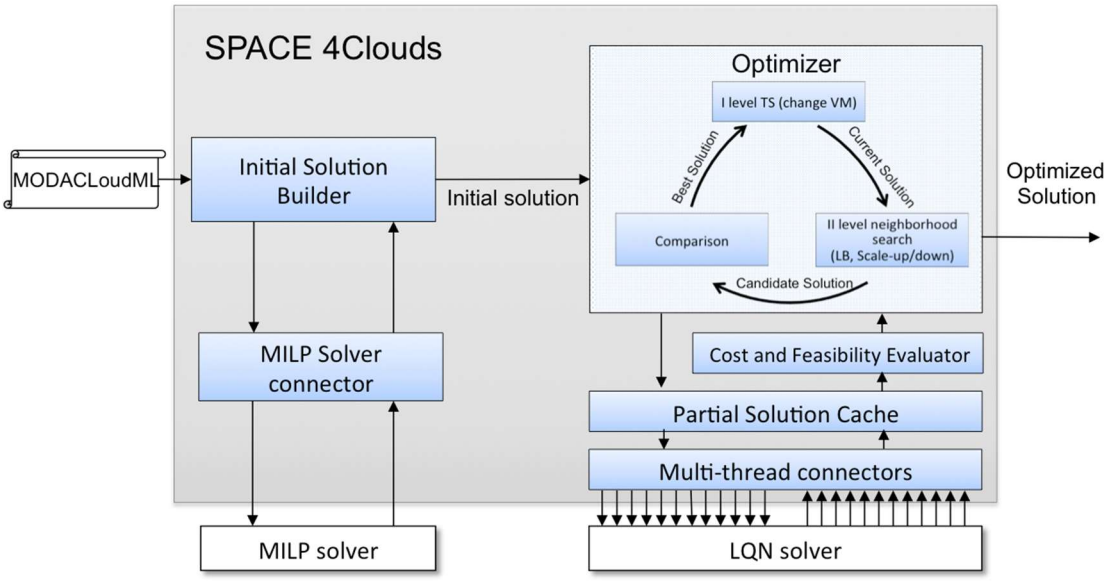
εργασίας [177], απαραίτητο για τη αξιολόγηση της απόδοσης και του κόστους της εφαρμογής κάτω από διαφορετικές συνθήκες φορτίου. Το φορτίο εργασίας ορίζεται σε ημερήσια αναφορά που αποτελείται από 24 χρονοθυρίδες. Η επιλογή αυτή είναι σύμφωνη με τα πιο συνηθισμένα μοντέλα τιμολόγησης που επιτρέπουν τη μίσθωση εικονικών πόρων σε ωριαία βάση.

Προκειμένου να αξιολογηθεί η απόδοση της υπό ανάπτυξη εφαρμογής, το SPACE4Cloud μεταφράζει τα μοντέλα σχεδιασμού που αποτελούν επέκταση του Palladio Component Model συνόλου (PCM) [179], σε πολυεπίπεδα δίκτυα αναμονής (LQNs) [180] τα οποία αποτελούν μια συγκεκριμένη ομάδα μοντέλων απόδοσης τα οποία λύνονται με τα κατάλληλα εργαλεία (κυρίως το LINE [178] ή το LQNS[181]).

Το *Σχήμα 28* παρουσιάζει τα κύρια στοιχεία του SPACE4Cloud εργαλείου καθώς και τις κύριες εξαρτήσεις των third-party components.

Το στοιχείο Initial Solution Builder είναι υπεύθυνο για την δημιουργία μιας αρχικής λύσης, λύνοντας ένα πρόβλημα μεικτού ακέραιου γραμμικού προγραμματισμού/MILP (Mixed Integer Linear Program). Όλα τα προβλήματα διεξήχθησαν χρησιμοποιώντας CPLEX [181]. Η προκύπτουσα λύση χρησιμοποιείται έπειτα από το Optimizer στοιχείο του οποίου ο πυρήνας είναι η γρήγορη και αποτελεσματική εξερεύνηση τοπικής αναζήτησης που συνδυάζει στοιχεία από την Tabu [182] και την επαναλαμβανόμενη τοπική αναζήτηση [183]. Η βασική σκέψη είναι η βελτίωση μέσω επανάληψης μιας τρέχουσας λύσης με τη βοήθεια τοπικών κινήσεων, ξεκινώντας από μία λύση η οποία θα οδηγήσει σε μία πιθανότατα καλύτερη. Συγκεκριμένα, μιας και το εργαλείο αυτό βρίσκει μία πιθανή παραμετροποίηση ως προς τον τύπο των VMs και τον αριθμό τους ανά επίπεδο εφαρμογής οι κινήσεις του SPACE4Cloud είναι δύο ειδών. Από τη μία πλευρά, έχει επινοηθεί μια στρατηγική που βασίζεται στην τοπική αναζήτηση Tabu που εφαρμόζεται στο επίπεδο ολόκληρου του χρονικού ορίζοντα (24 ώρες) και αλλάζει τον τύπο του VM χρησιμοποιώντας εκείνους που χρονικά έχουν ληφθεί λιγότερο υπ’ όψιν. Από

την άλλη πλευρά, γνωρίζοντας ότι η αλλαγή του τύπου τω VMs μπορεί να δημιουργήσει αρκετά προβλήματα, εφαρμόζεται μία γρήγορη επαναληπτική μέθοδος η οποία έχει αναπτυχθεί ώστε να υπολογίσει εκ νέου το βέλτιστο αριθμό των VMs. Η νέα λύση που προκύπτει αξιολογείται όσον αφορά στο κόστος και στην απόδοση. Τμήμα της διαδικασίας αξιολόγησης στηρίζεται στη δημιουργία μοντέλων απόδοσης LQN τα οποία αναλύονται από ένα εξωτερικό εργαλείο. Η βάση δεδομένων παρέχει τη σχετική πληροφορία τόσο για την τιμολόγηση των υπηρεσιών του Νέφους όσο και τις μετρήσεις απόδοσης οι οποίες δίνονται από τους παρόχους και αποτελούν απαραίτητα στοιχεία για την δημιουργία των LQN μοντέλων. Σε μία άλλη βάση δεδομένων η οποία συνδέεται με το Classification Tool είναι αποθηκευμένες οι μετρήσεις που προκύπτουν από Classification Tool και σχετίζονται με την απόδοση των διαφορετικών τύπων υπηρεσίας.



Σχήμα 28: Αρχιτεκτονική του SPACE4Cloud εργαλείου

Τα LQN μοντέλα προτιμώνται σε σχέση με άλλα μοντέλα υψηλών επιδόσεων, δεδομένου ότι μπορούν να χρησιμοποιηθούν για να εκπροσωπήσουν πολύπλοκά συστήματα (π.χ multi-tier εφαρμογές) και τον ανταγωνισμό μεταξύ των αιτημάτων των εφαρμογών στο επίπεδο του

Αυτή η σελίδα είναι σκόπιμα λευκή

λογισμικού. Σε αυτή την εργασία υιοθετήθηκε ο LINE solver [178] σε όλα τα πειράματα μιας και είναι η καλύτερη προσέγγιση που είναι σε θέση να λάβει υπ' όψη τη μεταβλητότητα των επιδόσεων του Νέφους μέσω τυχαίων περιβάλλοντων [34]. Σε αυτό το σημείο είναι σημαντικό να αναφερθεί ότι η αξιολόγηση μιας υποψήφιας λύσης που προκύπτει από το Optimizer είναι μια χρονοβόρα διαδικασία η οποία δυσχεραίνει την όλη διαδικασία της βελτιστοποίησης. Αυτό συμβαίνει επειδή μία λύση περιλαμβάνει 24 διαφορετικές παραμετροποιήσεις κατά τη διάρκεια της μέρας που οδηγούν σε διαφορετικά LQN μοντέλα τα οποία πρέπει να αξιολογηθούν ως προς το κόστος και το κατά πόσο μπορεί να εφαρμοστεί. Γι 'αυτό το λόγο, προκειμένου να επιταχυνθεί η διαδικασία της αξιολόγησης, επινοήθηκε μία λύση πολλαπλών νημάτων παράλληλης αξιολόγησης των 24 LQN μοντέλων μιας ενιαίας λύσης και ενός cache-based proxy για την αποθήκευση και την ανάκτηση της αξιολόγησης των προηγούμενων λύσεων για κάθε ώρα στο χρονικό διάστημα των 24 ωρών.

6.1.2 Αξιολόγηση της φάσης Βελτιστοποίησης

Η ενότητα αυτή έχει σαν στόχο να αξιολογήσει την περίπτωση όπου λαμβάνεται υπ΄ όψη μια πιο ακριβής πληροφορία σχετικά με την απόδοση των υπολογιστικών πόρων του Νέφους κατά την αξιολόγηση του χρόνου σχεδίασης και τη μεθοδολογία βελτιστοποίησης. Γι' αυτό το λόγο εκτελέστηκε ένα πείραμα δύο φάσεων. Στην πρώτη φάση χρησιμοποιείται το SPACE4Cloud ώστε να βρεθεί η βέλτιστη παραμετροποίηση (τύπος και αριθμός VMs) χρησιμοποιώντας διαφορετικά workloads για την HTTPAgent εφαρμογή χρησιμοποιώντας ως πάροχο την Amazon EC2. Κατά τη διάρκεια αυτής της διαδικασίας λήφθηκαν υπ' όψιν μόνο οι τιμές της ονομαστικής απόδοσης που είναι διαθέσιμες από τους παρόχους για τις m1.small και m1.medium VMs.

Σχετικά με τη δεύτερη φάση του πειράματος, στο SPACE4Cloud χρησιμοποιήθηκαν οι τιμές της απόδοσης οι οποίες αποκτήθηκαν μέσω της εφαρμογής της συγκριτικής αξιολόγησης και

εκτελέστηκε η ίδια ακριβώς διαδικασία όπως στην πρώτη περίπτωση. Όπως έχει ήδη αναφερθεί, κατά τη διάρκεια του πειράματος εξετάστηκαν διαφορετικά φορτία και συγκεκριμένα θεωρήσαμε 3 σταθερά φορτία όπου ο αριθμός των χρηστών ορίστηκε σε 50, 200 και 400 χρήστες και 3 μεταβλητά με τον αριθμό των χρηστών να φτάνει τους 50, 200 και 400 χρήστες κατά τις ώρες αιχμής στη διάρκεια της ημέρας (*Σχήμα 29*).

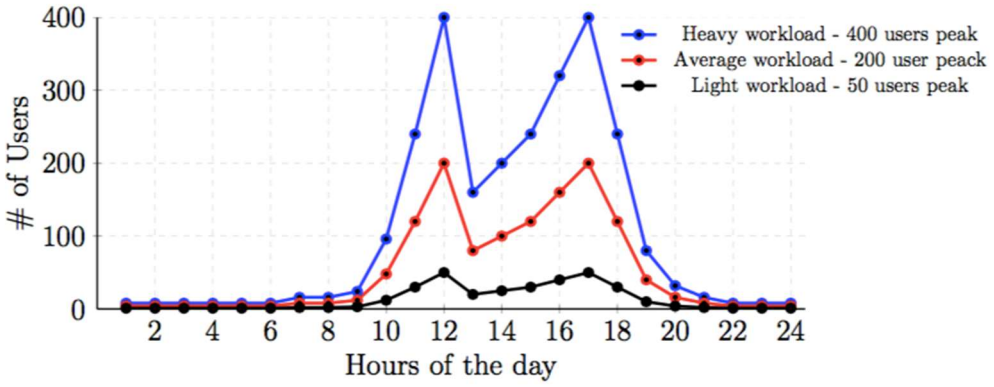
Όλα τα φορτία χαρακτηρίζονται από ένα χρόνο "σκέψης" 10 sec. Για το πείραμα, εφαρμόστηκε ένα εκτεταμένο PCM μοντέλο της HTTPAgent εφαρμογής και προσδιορίστηκαν οι απαιτούμενοι πόροι λειτουργίας. Το μοντέλο είναι διαθέσιμο στο [184]. Επιπλέον, το πείραμα πραγματοποιήθηκε λαμβάνοντας υπ' όψη δύο QoS περιορισμούς. Ο πρώτος (C1) περιορίζει το μέσο χρόνο απόκρισης της partialRead λειτουργίας στα 200 ms ενώ ο δεύτερος (C2) ορίζει στα 300 ms το 90ό εκατοστημόριο του ίδιου χρόνου απόκρισης. Ο χρόνος που απαιτείται για τη βελτιστοποίηση και την ανάλυση των δύο σεναρίων που αναφέρθηκαν παραπάνω διήρκησε μεταξύ 5 εως 10 λεπτών. Τα αποτελέσματα αυτής της ανάλυσης για το μεταβλητό φόρτο εργασίας και το C1 περιορισμό απεικονίζονται στο *Σχήμα 30*.

Μπορούμε να παρατηρήσουμε όπως ήταν αναμενόμενο ότι όλα τα ίχνη ακολουθούν όπως ορίζεται από το φόρτο εργασίας. Επιπλέον στην ανάλυση που πραγματοποιήθηκε με τη χρήση των τιμών από τα benchmarks, κατά μέσο όρο χρειάζεται ένας μεγαλύτερος αριθμός εικονικών μηχανών ώστε να καλυφθεί η C1 απαίτηση ποιότητας σε σχέση με τα αποτελέσματα από την χρήση των ονομαστικών τιμών που δίνονται από τον πάροχο.

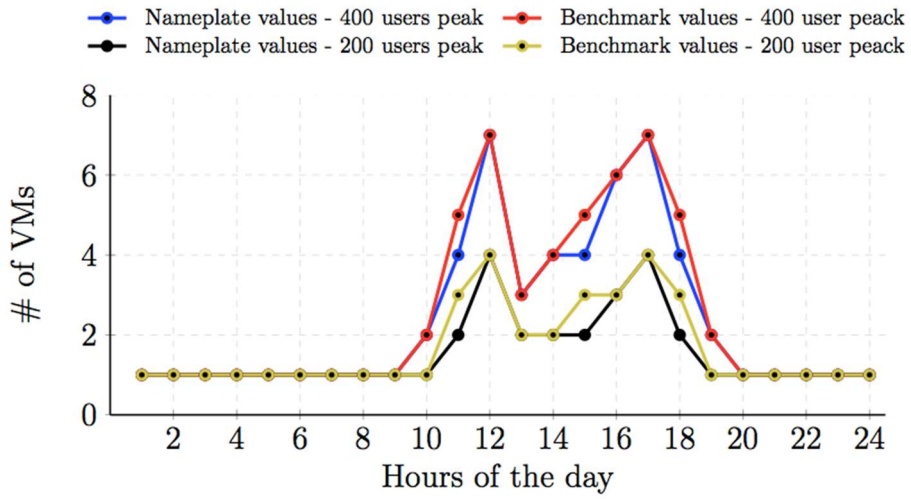
Σαν αποτέλεσμα προκύπτει η αύξηση του κόστους η οποία κυμαίνεται από 5% εως 8% περίπου. Αυτή η διαφορά προκύπτει εξαιτίας της διαφοράς που υπάρχει μεταξύ της ονομαστικής και της πραγματικής απόδοσης των m1.small VMs. Από την εικόνα έχει αφαιρεθεί το αποτέλεσμα για μέγιστο αριθμό 50 χρηστών σε ώρες αιχμής, μιας και απαιτείται σταθερά μόνο ένα VM την ώρα. Οι *Πίνακας 17*

Περιορισμός C1						
	Ονομαστικές Τιμές			**Benchmark Τιμές**		
	Ελαφρύ	Μέτριο	Βαρύ	Ελαφρύ	Μέτριο	Βαρύ
Τύπος VM	m1.small	m1.small	m1.small	m1.small	m1.medium	m1.small
Μέσο πλήθος VMs	4.00	7.00	1.00	1.00	2.00	7.00
Κόστος ($)	55.44	97.2	13.92	13.92	55.44	97.2

Πίνακας 18 συνοψίζουν τα αποτελέσματα των πειραμάτων.



Σχήμα 29: Το φορτίο εργασίας που υιοθετήθηκε στην πειραματική διαδικασία



Σχήμα 30: Αποτελέσματα λαμβάνοντας υπ' όψη μεταβλητό workload και C1 περιορισμό

Αυτή η σελίδα είναι σκόπιμα λευκή

Περιορισμοί C1 & C2						
	Ονομαστικές Τιμές			**Benchmark Τιμές**		
	Ελαφρύ	Μέτριο	Βαρύ	Ελαφρύ	Μέτριο	Βαρύ
Τύπος VM	m1.medium	m1.medium	m1.medium	m1.medium	m1.medium	m1.medium
Μέσο πλήθος VMs	1.00	1.46	2.21	1.00	1.46	2.21
Κόστος ($)	27.84	40.49	61.27	27.84	40.49	61.27

Περιορισμός C1						
	Ονομαστικές Τιμές			**Benchmark Τιμές**		
	Ελαφρύ	Μέτριο	Βαρύ	Ελαφρύ	Μέτριο	Βαρύ
Τύπος VM	m1.small	m1.small	m1.small	m1.small	m1.small	m1.small
Μέσο πλήθος VMs	1.00	1.54	2.38	1.00	1.67	2.50
Κόστος ($)	13.92	21.43	32.98	13.92	23.14	34.72

Πίνακας 17: Αποτελέσματα για μεταβλητό φόρτο εργασίας

Για κάθε πιθανή ρύθμιση (τύπος φορτίου x περιορισμός x ονομαστικές vs benchmark τιμές της απόδοσης) παρέχεται η εξής πληροφορία: ο επιλεγμένος τύπος VM, ο μέσος όρος των μηχανημάτων ανά ώρα και το καθημερινό κόστος. Αξίζει να μελετηθεί λίγο περισσότερο το αποτέλεσμα της επιδρασης του C2 περιορισμού. Επιβάλλοντας το 90ό εκατοστημόριο του

χρόνου απόκρισης να παραμείνει κάτω από 300 ms έχει σαν αποτέλεσμα την ανάγκη για την επιλογή ενός πιο ισχυρού μηχανήματος ακόμη και εαν εκτελείται ένα μικρό και σταθερό φορτίο. Αυτή η επιλογή όπως είναι αναμενόμενο να αντικατοπτρίζεται στο καθημερινό κόστος ακόμη και εαν ο μέσος αριθμός των VMs που απαιτούνται ελαττωθεί. Επιπλέον, η χρήση μεγαλύτερων VMs μπορεί να οδηγήσει σε μια πιθανή υποχρησιμοποίηση των πόρων όταν εξυπηρετείται ένα συγκεκριμένο φορτίο και σε αυτή την περίπτωση το SPACE4Cloud παράγει ίδια αποτελέσματα όταν χρησιμοποιούνται ονομαστικές και benchmark τιμές της απόδοσης. Αυτό ισχύει ιδιαίτερα όταν ο φόρτος εργασίας είναι σχετικά μικρός με αποτέλεσμα να είναι μικρή και η διαφορά μεταξύ των ονομαστικών και των benchmark τιμών.

	Περιορισμοί C1 & C2					
	Ονομαστικές Τιμές			**Benchmark Τιμές**		
	Ελαφρύ	Μέτριο	Βαρύ	Ελαφρύ	Μέτριο	Βαρύ
Τύπος VM	m1.medium	m1.medium	m1.medium	m1.medium	m1.medium	m1.medium
Μέσο πλήθος VMs	1.00	2.00	4.00	1.00	2.00	4.00
Κόστος ($)	27.84	55.44	110.88	27.84	55.44	110.88

	Περιορισμός C1					
	Ονομαστικές Τιμές			**Benchmark Τιμές**		
	Ελαφρύ	Μέτριο	Βαρύ	Ελαφρύ	Μέτριο	Βαρύ
Τύπος VM	m1.small	m1.small	m1.small	m1.small	m1.medium	m1.small
Μέσο πλήθος VMs	4.00	7.00	1.00	1.00	2.00	7.00
Κόστος ($)	55.44	97.2	13.92	13.92	55.44	97.2

Πίνακας 18: Αποτελέσματα για σταθερό φόρτο εργασίας

6.1.3 Συζήτηση

Η εύρεση υπηρεσιών Νέφους που να ικανοποιούν τα χαρακτηριστικά των εφαρμογών παρέχοντας ταυτόχρονα QoS εγγυήσεις είναι μια δύσκολη διαδικασία. Τα αποτελέσματα που προέκυψαν από τη μέχρι τώρα έρευνα αποδεικνύουν ότι η πληροφορία σχετικά με την απόδοση που δίνεται από τους παρόχους θα πρέπει να χρησιμοποιείται με προσοχή και κυρίως ως μια κατευθυντήρια γραμμή για την επιλογή τύπου VM, ενώ σημαντικό είναι να εκτιμηθεί η απόδοση των εφαρμογών του Νέφους βασιζόμενη στην διαδικασία της συγκριτικής αξιολόγησης. Τα αποτελέσματα μας δείχνουν ωστόσο ότι η βέλτιστη λύση εξαρτάται από πολλούς παράγοντες, συμπεριλαμβανομένων των χαρακτηριστικών της εφαρμογής, το φορτίο εργασίας και των QoS περιορισμών που θα πρέπει να πληρούνται. Το αρχικό βήμα της συγκριτικής αξιολόγησης και της κατηγοριοποίησης βοηθά ως προς δύο κατευθύνσεις. i) Αξιολογεί με ακρίβεια το πώς μια υπηρεσία του Νέφους μεταβάλλεται ανάλογα με τον τύπο του VM αλλά και τον πάροχο. ii) Φιλτράρει το σύνολο των παρόχων και των VM τύπων τα οποία στη συνέχεια θα θεωρηθούν υποψήφια για το στάδιο της βελτιστοποίησης ώστε τελικά να βρεθεί η βέλτιστη λύση.

Αυτή η σελίδα είναι σκόπιμα λευκή

7 Επίδραση παραμέτρων στην απόδοση των εφαρμογών

Στο παρόν κεφάλαιο περιγράφεται το πρόβλημα της επιδείνωσης στην απόδοση μιας εφαρμογής έπειτα από την συντοποθέτηση άλλων VMs στον ίδιο κόμβο της υποδομής που διαθέτει ένας πάροχος Υπολογιστικού Νέφους στους πελάτες του, καθώς και η δυνατότητα πρόβλεψης της επιβάρυνσης αυτής ώστε να επιλεγεί ο πλέον κατάλληλος τρόπος ανάθεσης VMs ανά πυρήνα που θα προκαλεί την μικρότερη επιβάρυνση.

7.1 Ορισμός του Προβλήματος

Η χρησιμοποίηση τεχνικών virtualization σε πρότυπα υπολογιστών όπως το Cloud Computing ώστε να εξασφαλιστεί η υψηλή χρησιμοποίηση των υπολογιστικών πόρων και η καλύτερη διαχειρισιμότητα των VMs επιτρέπει την εκτέλεση εφαρμογών με διαφορετικά χαρακτηριστικά και απαιτήσεις μέσα σε VMs στον ίδιο φυσικό πόρο. Ωστόσο, παρά τα πλεονεκτήματα που παρέχονται από την τεχνική αυτή, δεν εξασφαλίζεται αποτελεσματική απομόνωση και παρατηρείται μείωση της απόδοσης της εφαρμογής. Σε αυτό, έρχεται να προστεθεί μια σειρά άλλων παραμέτρων που μπορούν να επηρεάσουν την απόδοση των εφαρμογών όταν περισσότερες από μία VMs συνδρομολογούνται σε ένα φυσικό πόρο. Ένας από αυτούς τους παράγοντες είναι ο τύπος των εφαρμογών που εκτελούνται ταυτόχρονα στον ίδιο φυσικό πόρο, καθώς τα εσωτερικά χαρακτηριστικά του κάθε τύπου έχουν διαφορετική επίπτωση στην απόδοση των υπόλοιπων εφαρμογών που μοιράζονται τον ίδιο φυσικό πόρο.

Γνωρίζοντας ο πάροχος του Υπολογιστικού Νέφους την υποβάθμιση της απόδοσης των εφαρμογών σε αυτή την περίπτωση θα είναι σε θέση να κατανείμει αποτελεσματικότερα τους υπολογιστικούς πόρους της υποδομής του, αλλά και να βελτιστοποιήσει την κατανομή των VMs ελαχιστοποιώντας την παρεμβολή που προκαλεί ένα φορτίο εργασίας μιας εφαρμογής στα υπόλοιπα φορτία που εκτελούνται στον ίδιο πόρο. Με αυτόν τον τρόπο ο πάροχος του Υπολογιστικού Νέφους θα εξασφαλίσει το επιθυμητό επίπεδο ποιότητας υπηρεσίας στην εφαρμογή βελτιώνοντας τη φήμη του και ελαχιστοποιώντας την ανάγκη για περισσότερη διάθεση υπολογιστικής ισχύος.

Ο σκοπός του παρόντος κεφαλαίου είναι να εξεταστεί η απόδοση των εφαρμογών όταν εκτελούνται σε εικονικές υποδομές μοιραζόμενες τον ίδιο φυσικό κόμβο.

7.2 Δοκιμές και διαδικασία μετρήσεων

Ως ενδεικτικές εφαρμογές θεωρούμε τα 18 workloads από τα 3 benchmarks τα οποία παρουσιάστηκαν στο κεφάλαιο, εξαιτίας του γεγονότος ότι πρόκειται για benchmarks σε επίπεδο εφαρμογής τα οποία αντιπροσωπεύουν το μεγαλύτερο τμήμα εφαρμογών που υπάρχουν στο πραγματικό περιβάλλον, και είναι εύκολο να επαναχρησιμοποιηθούν από άλλους ερευνητές. Κατά τη διάρκεια της πειραματικής διαδικασίας εκτελέστηκαν διαφορετικοί συνδυασμοί benchmark tests ώστε να εντοπίσουμε εκείνα που προκαλούν λιγότερες παρεμβολές όταν εκτελούνται ταυτόχρονα στον ίδιο κόμβο. Ακόμη διαφορετικές αποφάσεις τοποθέτησης λήφθηκαν υπόψη υπολογίζοντας πώς αυτές επηρεάζουν την επιβάρυνση για τον ίδιο συνδυασμό benchmarks. Γι’ αυτό το σκοπό τοποθετήσαμε τα tests σε γειτονικούς ή μη γειτονικούς πυρήνες σε ένα φυσικό κόμβο, σε σύγκριση με την αυτόνομη εκτέλεση. Ως γειτονικούς πυρήνες θεωρούμε αυτούς που μοιράζονται μνήμη L2 cache.

Ο υπολογιστής που χρησιμοποιήθηκε για την εκτέλεση των πειραμάτων είναι μια εξαπλού πυρήνα (3,5 Ghz) CPU, με 16 GB μνήμης RAM, 6MB της L2 cache (2MB ανά ζευγάρι πυρήνα) και 16MB L3 cache (CPU 0 και CPU 1 μοιράζονται 8 MB ενώ οι CPU 2,3,4,5 μοιράζονται τα υπόλοιπα 8 MB). Το λειτουργικό σύστημα του φυσικού κόμβου ήταν Ubuntu Linux 14.04 ενώ το λειτουργικό σύστημα των VMs ήταν CentOS 6.5 (με OpenSSH) και το hypervisor της εικονικοποίησης είναι το VMware. Στην κάθε VM έχουν εγκατασταθεί τα 18 εκτελέσιμα workloads. Τα ορίσματα που έχουν χρησιμοποιηθεί στο πείραμα είναι ο αριθμός του test προς εκτέλεση και ένα αναγνωριστικό. Επειδή τα workloads που επιλέχθηκαν έχουν διαφορετικό χρόνο εκτέλεσης, όλα τα tests να εκτελέστηκαν για συγκεκριμένο χρονικό διάστημα και όχι για συγκεκριμένο αριθμό εκτελέσεων. Επιπλέον, κάθε test εκτελέστηκε αρκετές φορές για την συγκέντρωση αξιόπιστου στατιστικού δείγματος, έτσι ώστε να συγκεντρωθούν δεδομένα για εκατοντάδες εκτελέσεις ενός test ανά configuration και στο τέλος χρησιμοποιήθηκε η μέση τιμή των χρόνων εκτέλεσης για κάθε περίπτωση.

Τα πειράματα συντονίζονται από ένα Java Coordinator που τοποθετήθηκε στη CPU 0. Οι VMs τοποθετήθηκαν είτε σε γειτονικούς πυρήνες, η μία στη CPU 2 και η άλλη στη CPU 3 είτε σε μη γειτονικούς πυρήνες (η μία στη CPU 3 και η άλλη στη CPU 4), ανάλογα με το πείραμα.

Ο BCoordinator είναι μια custom Java εφαρμογή που εκτελείται στο φυσικό κόμβο και σκοπό έχει το συντονισμό της εκτέλεσης των workloads στις VMs. Η επίβλεψη της εκτέλεσης των benchmarks στις VMs πραγματοποιείται από τη Java εφαρμογή BClient. Οι διάφορες εφαρμογές BClient που εκτελούνται στις VMs, επικοινωνούν με τον BCoordinator μέσω TCP sockets, ώστε να συγχρονίζεται απόλυτα η εκτέλεσή τους. Ο BCoordinator δίνει εντολή στους BClient να ξεκινήσουν την εκτέλεση ενός benchmark σε μια ορισμένη χρονική στιγμή. Έχει επίσης τη δυνατότητα να εκτελεί σειριακά πολλαπλά benchmarks, τα οποία ορίζονται σε ένα αρχείο παραμετροποίησης που δίνεται ως είσοδο κατά την εκτέλεσή του. Έτσι, ο συνδυασμός

των BCoordinator, BClient και των scripts για την εκτέλεση των 18 workloads, μπορούν να θεωρηθούν ως μια σουίτα εργαλείων για πραγματοποίηση ταυτόχρονων συγκριτικών αξιολογήσεων. Τα εργαλεία έχουν σχεδιαστεί σύμφωνα με την client–server αρχιτεκτονική και ακολουθούν τις βέλτιστες πρακτικές ανάπτυξης εφαρμογών.

Κατά την ανάπτυξη των BCoordinator και BClient, λήφθηκε υπόψη το γεγονός ότι πολλά benchmarks χρειάζονται μια φάση αρχικοποίησης (initialization phase) ή/και φάση εκκαθάρισης (finalization phase) πριν και μετά την εκτέλεση ενός benchmark αντίστοιχα. Ο χρόνος των φάσεων αυτών, δε συμπεριλαμβάνεται στην τελική καταμέτρηση του χρόνου εκτέλεσης των benchmarks και δεν επηρεάζει το συγχρονισμό της εκτέλεσης αυτών. Για το σκοπό αυτό, ένας BClient μπορεί να περιμένει την ολοκλήρωση της φάσης αρχικοποίησης ενός benchmark που θα εκτελεστεί σε διαφορετικό BClient. Υπεύθυνος για την πραγματοποίηση του συγχρονισμού αυτού, είναι ο BCoordinator.

Τα βήματα για την εκτέλεση ενός πειράματος συγκριτικής αξιολόγησης, είναι τα εξής:

1. Κατά την εκκίνηση, ο BCoordinator περιμένει την υποδοχή συνδέσεων από τους BClients.

2. Οι BClients συνδέονται στον BCoordinator μέσω TCP sockets και περιμένουν την ειδοποίηση για την εκκίνηση της εκτέλεσης ενός benchmark.

3. Όταν ολοκληρωθεί η σύνδεση όλων τον BClients στον BCoordinator, ο τελευταίος διαβάζει το αρχείο παραμετροποίησης, το οποίο περιέχει τη λίστα με τα benchmarks που πρέπει να εκτελέσει ο κάθε BClient. Στο αρχείο αυτό, περιγράφεται η ακριβής σειρά εκτέλεσης των benchmarks, καθώς και σε ποιον BClient πρέπει το καθένα να εκτελεστεί.

4. Ο BCoordinator ειδοποιεί όλους τους BClients για το workload που πρέπει να εκτελέσουν, σύμφωνα με τη λίστα του αρχείου παραμετροποίησης. Έτσι, ο κάθε BClient εκτελεί τη φάση αρχικοποίησης για το workload που του αντιστοιχεί και περιμένει νέα εντολή από τον BCoordinator.

5. Όταν ολοκληρωθούν όλες οι φάσεις αρχικοποίησης, ο BCoordinator στέλνει, ταυτόχρονα σε όλους τους BClients, εντολές να εκκινήσουν την διαδικασία εκτέλεσης των workloads. Με την ολοκλήρωση της εκτέλεσης αυτών, αποστέλλονται στον BCoordinator τα αποτελέσματα του κάθε BClient. Ο BCoordinator αποθηκεύει τα αποτελέσματα σε αρχεία, τα οποία θα χρησιμοποιηθούν αργότερα ως είσοδο στο Matlab.

6. Μετά την αποθήκευση όλων των αποτελεσμάτων, ο BCoordinator ενημερώνει τους BClients να πραγματοποιήσουν την διαδικασία εκκαθάρισης.

7. Αν υπάρχουν επιπλέον workloads στη λίστα που πρέπει να εκτελεστούν, ο BCoordinator μεταφέρεται στο βήμα 4.

Ο BCoordinator έχει υλοποιηθεί ακολουθώντας το πολυνηματικό (multithreading) μοντέλο παράλληλης επεξεργασίας για την επικοινωνία με τους BClients. Πιο συγκεκριμένα, στον BCoordinator εκκινείται ένα ξεχωριστό νήμα για κάθε νέο αίτημα σύνδεσης από κάποιο BClient. Το νήμα αυτό, αναλαμβάνει τη διαχείριση της επικοινωνίας με το συγκεκριμένο BClient για όλη τη διάρκεια της εκτέλεσης του πειράματος. Ο συγχρονισμός με τα υπόλοιπα νήματα (ώστε να εκτελούνται ταυτόχρονα τα workloads), γίνεται χρησιμοποιώντας την τεχνική CountDownLatch της Java. Η τεχνική αυτή, επιτρέπει την ενημέρωση ενός νήματος, ότι όλα τα υπόλοιπα έχουν ολοκληρώσει την εκτέλεση μιας διαδικασίας. Έτσι δίνεται η δυνατότητα στο πρώτο νήμα να περιμένει την εκτέλεση όλων των ενεργειών από τα υπόλοιπα νήματα, πριν προχωρήσει στο επόμενο βήμα. Με τον τρόπο αυτό, οι τυχαίες καθυστερήσεις που μπορεί να υπάρξουν κατά τη διάρκεια της εκτέλεσης του πειράματος, δεν επηρεάζουν καθόλου το συγχρονισμό μεταξύ των BClients.

Αξίζει ακόμη να αναφερθεί ότι λόγω του πολύ μεγάλου συνολικού χρόνου εκτέλεσης των πειραμάτων, παρατηρήθηκε ότι ήταν πιθανό ορισμένα workloads να αποτυγχάνουν την εκτέλεσή τους (πχ λόγω δικτύου). Για το λόγο αυτό, έχει ληφθεί ειδική μέριμνα κατά την

ανάπτυξη των εφαρμογών, ώστε να μην επηρεάζονται από τέτοιου είδους σφάλματα. Τα αποτυχημένα workloads δεν παράγουν αρχεία αποτελεσμάτων και δεν απορρυθμίζουν τη συγχρονισμένη εκτέλεσή τους. Έτσι μια χρήστης έχει τη δυνατότητα αυτοματοποιημένης εκτέλεσης πολλαπλών συγκριτικών αξιολογήσεων, χωρίς να απαιτείται η αλληλεπίδρασή της με την εφαρμογή.

Host OS	**Ubuntu 14.04**
Guest OS	Centos 6.5
Hypervisor	VMware
Benchmarks	DaCapo Suite Filebench YCSB

Πίνακας 13: Λεπτομέρειες μετρητικής διάταξης

7.3 Αναλυτικά αποτελέσματα

Σε αυτή την ενότητα παρουσιάζονται τα αποτελέσματα των μετρήσεων από την εκτέλεση των VMs στους πυρήνες. Το σύστημα που χρησιμοποιήθηκε χρειάστηκε ένα χρόνο προθέρμανσης, προκειμένου επιτευχθεί μία σταθερή συμπεριφορά των VMs. Στα αποτελέσματα χρησιμοποιήθηκε η μέση τιμή των επιμέρους βαθμολογιών των tests για όλες τις εκτελέσεις μέσα στο διάστημα των 300 δευτερολέπτων. Στο διάστημα αυτό κάθε test εκτελέστηκε αρκετές φορές μέχρι να παρέλθουν τα 300 δευτερόλεπτα.

Τα πειράματα που εκτελέστηκαν για τη διερεύνηση στην απόδοση ήταν τρία. Στην πρώτη διαμόρφωση κάθε VM εκτελείται ως αυτόνομη σε ένα πυρήνα. Καμία άλλη VM δεν είναι σε λειτουργία. Τα αποτελέσματα από το πρώτο πείραμα χρησιμοποιήθηκαν ως μέτρο σύγκρισης

με τα σενάρια συνεκτέλεσης των VMs στον ίδιο φυσικό πόρο αποκτώντας ένα βασικό σκορ εκτέλεσης (standalone baseline test score) για σύγκριση με τα σενάρια συνεκτέλεσης των VMs στον ίδιο φυσικό πόρο.

Στο δεύτερο πείραμα συνεκτελέστηκαν οι VMs στον ίδιο φυσικό πυρήνα αλλά σε διαφορετικούς εικονικούς πυρήνες με άμεση γειτνίαση στην ίδια CPU. Σε αυτή την περίπτωση οι πυρήνες έχουν κοινές μνήμες cache L1, L2 και L3. Το τρίτο πείραμα έχει ως στόχο τη διερεύνηση της επίδρασης στην απόδοση των VMs όταν εκτελούνται σε διαφορετικούς φυσικούς και εικονικούς πυρήνες χωρίς άμεση γειτνίαση. Η μόνη μνήμη cache που μοιράζεται σε αυτή την περίπτωση είναι η L3. Τα παραπάνω πειράματα έχουν σα στόχο να προσδιοριστεί το αποτέλεσμα της παρεμβολής μνήμης L1, L2 και L3 cache.

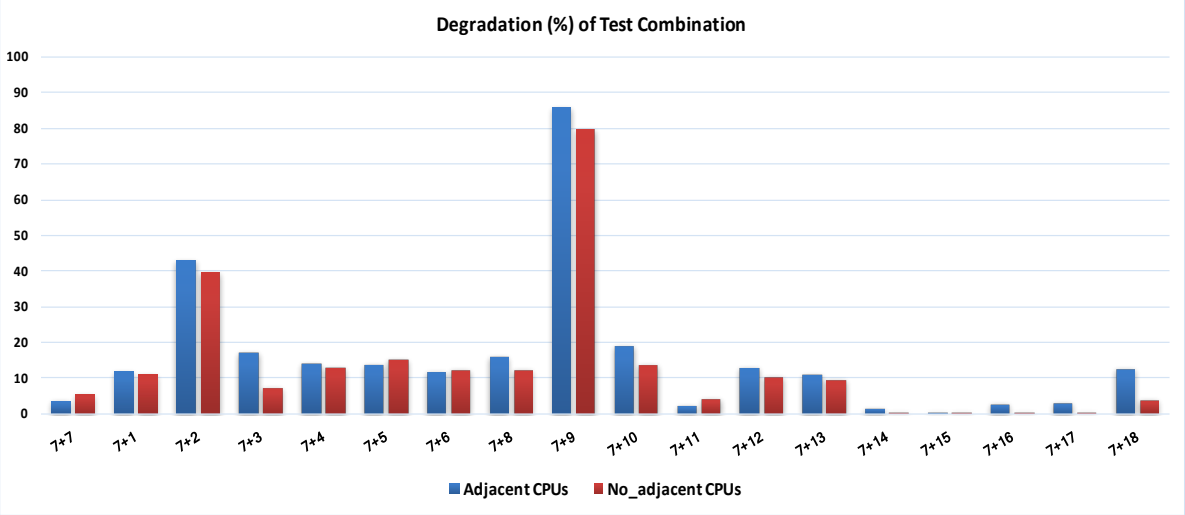
σύστημα είχε τη μικρότερη επιβάρυνση. Για την απεικόνιση της υποβάθμισης της απόδοσης ανάλογα με τους συνδυασμούς των tests που συνεκτελούνται, υπολογίστηκε το % ποσοστό υποβάθμισης ανά συνδυασμό tests και τα αποτελέσματα παρουσιάζονται στην εικόνα

Για τον υπολογισμό της υποβάθμισης της απόδοσης χρησιμοποιήθηκε η εξίσωση Σχήμα 31. Στον τύπο αυτό, A και B είναι οι αριθμοί tests ανά συνδυασμό, I είναι το σενάριο ανάπτυξης (πυρήνες με άμεση γειτνίαση, πυρήνες με μη άμεση) και BASE είναι το σκορ όταν τα tests εκτελούνται ως standalone, χωρίς οποιοδήποτε άλλο test να εκτελείται στο φυσικό κόμβο.

$$Degradation = 100 * \left(\frac{T_{A_I} - T_{A_{BASE}}}{T_{A_{BASE}}} + \frac{T_{B_I} - T_{B_{BASE}}}{T_{B_{BASE}}}\right)$$

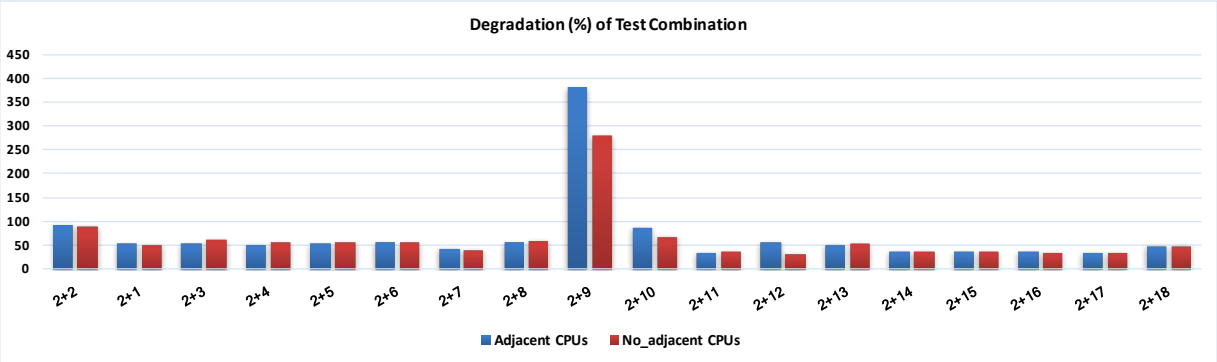
Σχήμα 31: Ορισμός ποσοστιαίας υποβάθμισης απόδοσης

Αυτή η σελίδα είναι σκόπιμα λευκή



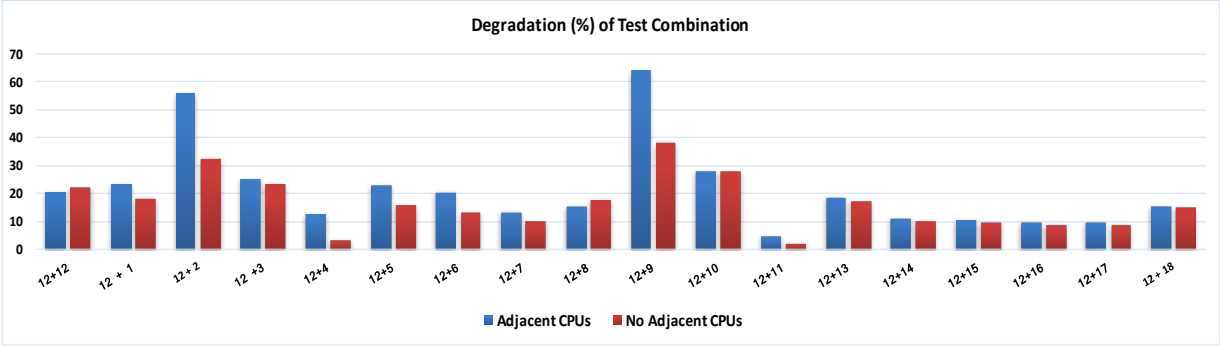
Σχήμα 32: Ποσοστό της επιβάρυνσης της βαθμολογίας για το συνδυασμό του tomcat test με όλα τα υπόλοιπα φορτία προκειμένου να καταδειχθεί το εύρος της υποβάθμισης και ο προσδιορισμός των βέλτιστων συνδυασμών.

Σε αυτό το σχήμα αποτυπώνεται το ποσοστό επιβάρυνσης της απόδοσης του tomcat benchmark σε συνδυασμό με τα υπόλοιπα εξεταζόμενα φορτία για τα δύο διαφορετικά σενάρια.



Σχήμα 33: Ποσοστό της επιβάρυνσης της βαθμολογίας για το συνδυασμό του eclipse test με όλα τα υπόλοιπα φορτία προκειμένου να καταδειχθεί το εύρος της υποβάθμισης και ο προσδιορισμός των βέλτιστων συνδυασμών

Αυτή η σελίδα είναι σκόπιμα λευκή



Σχήμα 34: Ποσοστό της επιβάρυνσης της βαθμολογίας για το συνδυασμό του webproxy test με όλα τα υπόλοιπα φορτία προκειμένου να καταδειχθεί το εύρος της υποβάθμισης και ο προσδιορισμός των βέλτιστων συνδυασμών

Στα παραπάνω σχήματα αποτυπώνονται τα ποσοστό επιβάρυνσης της απόδοσης των tomcat, eclipse και webproxy benchmarks σε συνδυασμό με τα υπόλοιπα εξεταζόμενα φορτία για τα δύο διαφορετικά σενάρια. Επειδή ο αριθμός των φορτίων είναι πολύ μεγάλος μία γραφική παράσταση που θα περιλάμβανε τη σύγκριση όλων των δυνατών συνδυασμών θα ήταν δυσανάγνωστη. Γι' αυτό το λόγο επιλέχθηκαν μόνο τα τρία benchmarks τα οποία μελετήθηκαν στη διαδικασία της κατηγοριοποίησης και η διακύμανση της απόδοσης τους παρουσιάζει το μεγαλύτερο ενδιαφέρον μιας και παρουσιάζουν παρόμοια συμπεριφορά με τις εφαρμογές (NewsAsset και HTTPAgent) που μελετήθηκαν κατά τη διάρκεια της αξιολόγησης της μεθοδολογίας που παρουσιάστηκε σε αυτή τη διατριβή.

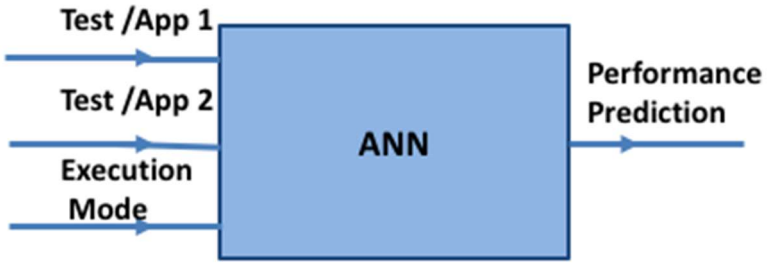
Από αυτά φαίνεται ότι η υπολογιστική επιβάρυνση μιας VM μπορεί να κυμαίνεται από σχεδόν 0 και μέχρι 360%, ανάλογα με τα συνεκτελούμενα tests και το σενάριο ανάπτυξης. Από τα γραφήματα είναι επίσης προφανές ότι και για τα τρία benchmarks η χειρότερη επιβάρυνση προκύπτει όταν συνεκτελούνται με το fileserver benchmark, ενώ για τα tomcat και webproxy η επιβάρυνση είναι επίσης αυξημένη όταν εκτελούνται με το eclipse benchmark. Επομένως τα αποτελέσματα αυτά αποδεικνύουν τη σημαντικότητα της έρευνας για την επιβάρυνση που προκαλείται μεταξύ VMs καθώς η ταυτόχρονη εκτέλεση τους στον ίδιο φυσικό πόρο οδηγήσει σε σημαντική υποβάθμιση των χαρακτηριστικών QoS των εφαρμογών. Επιπλέον, στις

Αυτή η σελίδα είναι σκόπιμα λευκή

περισσότερες περιπτώσεις η χρήση παρακείμενων πυρήνων έχει τη χειρότερη επίδοση, λόγω παρεμβολών μνήμης L2 cache.

Σχετικά με την ικανότητα πρόβλεψης εκ των προτέρων αυτής της επιβάρυνσης, βασιστήκαμε στο μοντέλο που αναπτύχθηκε στο το οποίο και τροποποιήσαμε ώστε να είναι δυνατό να εφαρμοστεί και να υιοθετηθεί στην πειραματική διαδικασία που αναπτύχθηκε στην παρούσα διατριβή. Το μοντέλο αυτό σχετίζεται με την δημιουργία Τεχνητών Νευρωνικών Δικτύων (ΤΝΔ) τα οποία δημιουργούνται μέσω ενός εξελικτικού (GAbased) αλγόριθμου. Σύμφωνα με τη μελέτη αυτή δημιουργείται ένα μοντέλο το οποίο είναι σε θέση να προβλέψει την απόδοση των εφαρμογών ανάλογα με τους τρόπους δρομολόγησης. Ως είσοδο παίρνει τους όρους της εφαρμογής και παράγει το αναμενώμενο επίπεδο εκτέλεσης το οποίο είναι το αποτέλεσμα από την εκτέλεση του test. Τα ΤΝΔ εκπροσωπεύουν τη μέθοδο μαύρου κουτιού και μπορούν να προβλέψουν τη συμπεριφορά ενός συστήματος γρήγορα και αποτελεσματικά, βασιζόμενα μόνο σε ένα σύνολο δεδομένων το οποίο χρησιμοποιούν για να εκπαιδευτούν το οποίο περιλαμβάνει εισόδους και τις αντίστοιχες εξόδους του συστήματος.

Οι καθορισμένες είσοδοι και έξοδοι του μοντέλου ΤΝΔ απεικονίζονται στο **Error! Reference source not found.**. Ο Πίνακας 19 περιέχει την κλίμακα των εξεταζόμενων τιμών.



Σχήμα 35: Είσοδοι και έξοδοι του μοντέλου ΤΝΔ

Αριθμός test 1	Από 1 έως 18 (Συνολικός Αριθμός DaCapo, Filebench και YCSB Benchmarks)
Αριθμός test 2 (συνεκτελούμενη εργασία)	Από 1 έως 18 (Συνολικός Αριθμός DaCapo, Filebench και YCSB Benchmarks)
Είδος εκτέλεσης (Execution Mode)	Παρακείμενοι Πυρήνες, Μη παρακείμενοι πυρήνες

Πίνακας 19: .Δυνατές τιμές Εισόδων Μοντέλου Υποδομής ΤΝΔ

Όλες οι είσοδοι και οι έξοδοι κανονικοποιήθηκαν στο (-1,1) αριθμητικό διάστημα, ενώ στην περίπτωση των μη αριθμητικών εισόδων, ορίστηκαν διαφορετικές τιμές για κάθε κατάσταση. Η κανονικοποίηση είναι απαραίτητη για την εσωτερική αντιπροσώπευση στο ΤΝΔ και την

7.4 Βελτιστοποίηση της δομής του ΤΝ

Για την βελτιστοποίηση της δομής του ΤΝ χρησιμοποιήθηκε ο αλγόριθμος που περιγράφεται στο [41] ακολουθώντας την ίδια διαδικασία. Σύμφωνα με την προσέγγιση αυτή, εξετάστηκαν 13 διαφορετικές συναρτήσεις μεταφοράς σύμφωνα με το [44] όπου το μέγιστο όριο των στρωμάτων ήταν δέκα (10) και των νευρώνων ανά στρώμα τριάντα (30). Πραγματοποιήθηκαν 2 ανεξάρτητες εκτελέσεις με 20 και 30 γενεές αντίστοιχα. Η πρώτη γενεά ερεύνησε 20 πιθανές λύσεις (μέγεθος πληθυσμού) ενώ η δεύτερη 40. Ο Πίνακας 16 περιέχει την δομή *ANN* και τη γενική ακρίβεια πρόβλεψης των συγκεκριμένων εκτελέσεων. Ακόμη μόνο το καλύτερο μοντέλο από κάθε εκτέλεση απεικονίζεται. Τα μοντέλα που σχηματίστηκαν κατά την πρώτη εκτέλεση ήταν 307 ενώ στη δεύτερη περίπτωση 165. Για τον υπολογισμό του μέσου απόλυτου σφάλματος μόνο οι τιμές στο ανεξάρτητο σύνολο επικύρωσης χρησιμοποιήθηκαν τα οποία δεν εφαρμόστηκαν κατά τη διάρκεια της εκπαίδευσης.

Με την αύξηση των γενεών παρατηρείται βελτίωση της απόδοσης. Συγκεκριμένα η καταλληλότερη λύση βρίσκεται με τις 30 γενεές μιας και το μέσο λάθος είναι καλύτερο. Τα πειραματικά δεδομένα που παρουσιάστηκαν στην ενότητα αυτή χρησιμοποιήθηκαν για την

εκπαίδευση (50%), την ενδιάμεση επικύρωση κατά τη διάρκεια της εκπαίδευσης (20%) και την ανεξάρτητη τελική επικύρωση (30%) του ΤΝΔ.

Αριθμός Στρωμάτων/ Γενεές ΓΑ	**Νευρώνες ανά στρώμα**	**Συναρτήσεις Μεταφοράς ανά στρώμα**	**Μέσο Απόλυτο Σφάλμα (%)**	**Αφαίρεση top 2 σφαλμάτων ΜΑΣ (%)**	**Αφαίρεση top 4 σφαλμάτων ΜΑΣ (%)**	**Αφαίρεση 10 σφαμλμάτων ΜΑΣ (%)**
3/20	3-19-1	Purelin-tansig-tansig	15,8801	9,9249	8,9807	7,8494
4/30	3-26-22-1	Satlin-tribas-purelin-purelin	18,6304	9,7512	8,1285	5,7815

Ένα άλλο ενδιαφέρον συμπέρασμα είναι ότι αφαιρώντας κάποιες από τις τιμές οι οποίες από τη γραφική στο σχήμα φαίνεται ότι παρεκλίνουν σε μεγάλο βαθμό από τις υπόλοιπες συμπαραίνουμε ότι το απόλυτο μέσο σφάλμα μειώνεται σημαντικά. Σε αυτή την περίπτωση μία πιθανή αιτία είναι το σύνολο δεδομένων που χρησιμοποιήθηκε να περιέχει κάποια αποτελέσματα τα οποία ίσως να έχουν επηρεαστεί από κάποιο απρόβλεπτο παράγοντα. Επειδή δεν έχουμε real time scheduling process, είναι πολύ πιθανό ανά πάσα στιγμή να ξεκινήσει κάποια διαδικασία, όπως κάποιο update κατά τη διάρκεια εκτέλεσης του πειράματος που τελικά να επηρεάσει τα αποτελέσματα των μετρήσεων.

Σχετικά με τις δύο εκτελέσεις αρχικά φαίνεται πως η πρώτη δίνει καλύτερα αποτελέσματα, ωστόσο αφαιρώντας 2 και στη συνέχεια 4 σφάλματα συμπεραίνουμε τελικά ότι η καλύτερη λύση είναι η δεύτερη.

7.5 Σύγκριση με γραμμική παλινδρόμηση πολλαπλών μεταβλητών

Προκειμένου να συγκριθεί η προσέγγισή μας, η μέθοδος γραμμικής παλινδρόμησης πολλών μεταβλητών επιλέχτηκε. Η λειτουργία MATLAB mvregress χρησιμοποιήθηκε, στο ίδιο κανονικοποιημένο σύνολο δεδομένων. Για την επικύρωση των αποτελεσμάτων, το ίδιο 30% ανεξάρτητο σύνολο δεδομένων επικύρωσης χρησιμοποιήθηκε, για λόγους αντικειμενικότητας. Ο Πίνακας 17 περιέχει τις λεπτομέρειες της συνάρτησης προσέγγισης. Το κελί συντελεστών περιέχει στην αντίστοιχη σειρά τους πολλαπλασιαστές για καθεμία από τις 5 εισόδους που απεικονίζονται στο Σχήμα 59, συν το σταθερό παράγοντα. Οι τιμές τους είναι χαμηλές επειδή έχουν παραχθεί για τις κανονικοποιημένες τιμές του συνόλου δεδομένων στο [-1.1] διάστημα.

Συντελεστές	**Μέσο Απόλυτο Σφάλμα (%)**	**Αφαίρεση top 2 σφαλμάτων ΜΑΣ (%)**	**Αφαίρεση top 4 σφαλμάτων ΜΑΣ (%)**	**Αφαίρεση top 10 σφαλμάτων ΜΑΣ (%)**
-0,0442 -0,0628 -0,0130 -0,6960	53,68	12,77	10,6905	8,5732

Από τη σύγκριση των δύο προσεγγίσεων μπορεί να εξαχθεί το συμπέρασμα ότι τα βελτιστοποιημένα ΤΝ έχουν καλύτερο μέσο απόλυτο σφάλμα απόδοσης.

8 Συμπεράσματα και Μελλοντική εργασία

8.1.1 Σύνοψη

Η κατανόηση του πώς μια αυθαίρετη εφαρμογή χρησιμοποιεί τους υπολογιστικούς πόρους είναι ζωτικής σημασίας για τη μετάβαση της στο Νέφος. Επιπλέον, η δυνατότητα για τη μέτρηση των Υπηρεσιών Νέφους χρησιμοποιώντας benchmark εφαρμογές δίνει τη δυνατότητα να εφαρμοστεί μια πιο αφαιρετική διαδικασία για τη μέτρηση τους, ώστε να βρεθεί η κατάλληλη λύση που θα φιλοξενήσει την εφαρμογή. Στα πλαίσια της διδακτορικής διατριβής αναπτύχθηκε μια μεθοδολογία και μια αλυσίδα από εργαλεία που αποτελείται από το Profiling και το Classification Tool. Χρησιμοποιώντας τα εργαλεία αυτά και σε συνδυασμό με τα αποτελέσματα από τη διαδικασία της συγκριτικής αξιολόγησης προσδιορίζεται το προφίλ μιας εφαρμογής η οποία κατατάσσεται σε μία τυπική κατηγορία εφαρμογών, ώστε στη συνέχεια εντοπιστεί η καλύτερη υπηρεσία Νέφους λαμβάνοντας υπ'όψιν την απόδοση παρέχοντας την καλύτερη Ποιότητα Υπηρεσιών (QoS).

8.1.2 Προσθήκη νέων εφαρμογών συγκριτικής αξιολόγησης και παρόχων του Νέφους

Η μελέτη που εκπονήθηκε στην παρούσα διατριβή επιτρέπει τη συζήτηση για ζητήματα μελλοντικής έρευνας στην περιοχή της επιλογής της βέλτιστης υπηρεσίας του Νέφους κατά τη «μετανάστευση» μιας εφαρμογής σε αυτό. Μια ενδιαφέρουσα πτυχή είναι να επεκταθεί η συγκεκριμένη μελέτη ώστε να συμπεριληφθεί ένας μεγαλύτερος αριθμός εφαρμογών

συγκριτικής αξιολόγησης, όπως για παράδειγμα την ενσωμάτωση των CFD και security benchmarks κλπ. Ειδικότερα για τα security benchmarks, παρά το γεγονός ότι οι υπηρεσίες του Νέφους έχουν αποδείξει τα σημαντικά οφέλη για τους χρήστες, ωστόσο οι διάφορες επιχειρήσεις είναι διστακτικές ως προς το να υιοθετήσουν αυτές τις λύσεις μιας και είναι έντονη η ανησυχία για θέματα ασφάλειας των περιβάλλοντων του Νέφους. Για το σκοπό αυτό η ενσωμάτωση αντίστοιχων benchmarks για τη μέτρηση και την αξιολόγηση των παρόχων και από αυτό το πεδίο θα ήταν σημαντική.

Ωστόσο κρίσιμη θεωρείται η τήρηση των βασικών αρχών που αναπτύχθηκαν στην παρούσα διατριβή, οι οποίες θα εφαρμοστούν στη διαδικασία της συγκριτικής αξιολόγησης όπως για παράδειγμα το να μελετηθούν και να συμπεριληφθούν benchmarks τα οποία περιλαμβάνουν workloads σε επίπεδο εφαρμογής ή το να εκτελούνται περιοδικά τα benchmarks και να καταγράφονται οι αντίστοιχες μετρήσεις.

8.2 Μηχανισμός εκτίμησης της ποιότητας της υπηρεσίας

Όσων αφορά στον τρόπο εφαρμογής της συγκριτικής αξιολόγησης όπως ήδη έχει αναφερθεί η διαδικασία της εκτέλεσης των benchmarks θα πρέπει να επαναλαμβάνεται σε τακτά χρονικά διαστήματα. Γι ‘αυτό το λόγο μελλοντκά η ανάπτυξη μιας benchmark υπηρεσίας (3rd party service) που θα είναι υπεύθυνη για τη χρήση των benchmarking tools σε συστηματική βάση είναι σημαντική. Βασική αρμοδιότητα της υπηρεσίας αυτής θα είναι εκτός από την εκτέλεση των tests να διατηρεί λεπτομερή στοιχεία σχετικά με τα αποτελέσματα του κάθε παρόχου για την κάθε κατηγορία των benchmark εφαρμογών. Επιπλέον, θα μπορεί να καθορίσει μετρήσεις που βοηθούν στην άμεση σύγκριση των παρόχων του Νέφους λειτουργώντας αφαιρετικά και παρέχοντας μια ευρεία αξιολόγηση των υπηρεσιών.

8.3 Μηχανισμός για την διατήρηση της αξιοπιστίας του παρόχου

Προκειμένου να διατηρηθεί η αξιοπιστία του παρόχου του Νέφους και να διευκολυνθεί ο ρόλος του κρατώντας σταθερές τις παραμέτρους της ποιότητας υπηρεσιών, ένας συνδυαστικός μηχανισμός των εργαλείων που αναπτύχθηκαν στην παρούσα διατριβή θα μπορούσε να υλοποιηθεί. Ο μηχανισμός αυτός θα επικοινωνεί με το επίπεδο στο Νέφος που είναι υπεύθυνο για τη δημιουργία και το χειρισμό των εικονικών πόρων που τρέχουν πάνω από τους φυσικούς κόμβους. Από το επίπεδο αυτό, ο μηχανισμός θα αντλεί πληροφορίες σχετικά με την ταυτότητα των εικονικών μηχανών την τρέχουσα κατάστασή τους, τον κόμβο στον οποίο τρέχουν κτλ. προκειμένου να χρησιμοποιήσουν αυτή την πληροφορία ως αναφορά κατά τη διάρκεια της ανάλυσης. Η διαδικασία της ανάλυσης θα περιλαμβάνει τη χρήση των εικονικών μηχανών με τα προκαθορισμένα benchmarks τα οποία θα εκτελούνται σε διαφορετικούς συνδυασμούς, ώστε να προσδιορίσει το overhead στο κάθε κόμβο από την ταυτόχρονη εκτέλεση για τον κάθε συνδυασμό. Τα αποτελέσματα από την προηγούμενη εκτέλεση μπορούν να χρησιμοποιηθούν στη μεθοδολογία που αναπτύχθηκε στο κεφάλαιο 7 και να χρησιμοποιηθούν ως σύνολο εκπαίδευσης, για τον καθορισμό της βέλτιστης κατανομής των εικονικών μηχανών.

Με τη δημιουργία τέτοιων μοντέλων, ο πάροχος θα είναι σε θέση να προσδιορίσει από πριν την επίδραση της ποιότητας της υπηρεσίας QoS μέσα στις VMs. Τα Profiling και Classification Tools θα χρησιμοποιηθούν σε αυτό το επίπεδο, ώστε να i) να εντοπίσουν στο συγκεκριμένο φυσικό κόμβο το ίχνος των VMs που περιλαμβάνουν τα benchmarks ii) να συγκρίνουν το ίχνος μιας αυθαίρετης εφαρμογής ενός πελάτη με τα ίχνη από τα benchmarks και να υπολογίσουν το προφίλ της βάσει των benchmarks. Έτσι, μέσα από αυτή τη διαδικασία ο πάροχος θα είναι σε θέση να γνωρίζει τις υπολογιστικές ανάγκες των εφαρμογών (χωρίς να έχει γνώση της εφαρμογής που τρέχει μέσα στο VM) καθώς και ποιες θα είναι οι επιπτώσεις

στην απόδοσή τους σε περίπτωση που διαφορετικά είδη εφαρμογών μοιράζονται τον ίδιο φυσικό κόμβο χρησιμοποιώντας τα αποτελέσματα από τα μοντέλα πρόβλεψης.

Βιβλιογραφικές Αναφορές

[1] Antony T.Velte, Toby J.Velte, Robert Elsenpeter, 2010, Cloud computing: a practical approach, The McGraw-Hill Companies)I. Foster, C. Kesselman, S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations" International Journal Supercomputer Applications, Vol. 15, No. 3, 2001.

[2] M.G. Avram. ―Advantages and Challenges of Adopting Cloud Computing from an Enterprise Perspective‖,Procedia Technology, 12, 2014, pp. 529-534.

[3] Wang, Lizhe, et al. "Cloud computing: a perspective study." New Generation Computing 28.2 (2010): 137-146.

[4] Zhang Mian, Zhang Nong; "The Study of Multimedia Data Model Technology Based on Cloud Computing"; 2010 2nd International Conference on Signal Processing Systems (ICSPS).

[5] Marston, Sean, et al. "Cloud computing — The business perspective." Decision Support Systems 51.1 (2011): 176-189.

[6] Vaquero Luis M., et al. "A break in the clouds: towards a cloud definition."*ACM SIGCOMM Computer Communication Review* 39.1 (2008): 50-55.,

[7] Zhang, Qi, Lu Cheng, and Raouf Boutaba. "Cloud computing: state-of-the-art and research challeonges." *Journal of internet services and applications* 1.1 (2010): 7-18.

[8] Mell, Peter, and Tim Grance. "The NIST definition of cloud computing."

(2011).

[9] Hashemi, Seyyed Mohsen, and Amid Khatibi Bardsiri. "Cloud computing Vs. grid computing." ARPN J. Syst. Softw 2.5 (2012): 188-194

[10] Gong, Chunye, et al. "The characteristics of cloud computing." *Parallel Processing Workshops (ICPPW), 2010 39th International Conference on*. IEEE, 2010.

[11] N. R. Herbst, S. Kounev, and R. Reussner. "Elasticity in Cloud Computing:What It Is, and What It Is Not". In: ICAC. 2013.

[12] "Database scalability, elasticity, and autonomy in the cloud." Database Systems for Advanced Applications. Springer Berlin Heidelberg, 2011.

[13] Virtualization definition available at: http://yoyoclouds.wordpress.com/2012/04/24/what-is-virtualization/

[14] Gurav, U., and R. Shaikh. "Virtualization: a key feature of cloud computing."*Proceedings of the International Conference and Workshop on Emerging Trends in Technology*. ACM, 2010.

[15] Macias, Guillermo. "Virtualization and Cloud Computing." (2013).

[16] Chung, JaeWoong, et al. "Tradeoffs in transactional memory virtualization."ACM SIGARCH Computer Architecture News. Vol. 34. No.5. ACM, 2006

[17] Data Virtualization online available at: http://www.compositesw.com/data-virtualization/

[18] Vmware: The Software-Defined Data Center online available at: http://www.vmware.com/software-defined-datacenter/networkingsecurity

[19] Vishwanath, Kashi Venkatesh, and Nachiappan Nagappan. "Characterizing cloud computing hardware reliability." *Proceedings of the 1st ACM symposium on Cloud computing*. ACM, 2010.

[20] Shen, Zhiming, et al. "Cloudscale: elastic resource scaling for multi-tenant cloud systems." *Proceedings of the 2nd ACM Symposium on Cloud Computing*. ACM, 2011.

[21] Hay, Brian, Kara Nance, and Matt Bishop. "Storm clouds rising: security challenges for IaaS cloud computing." *System Sciences (HICSS), 2011 44th Hawaii International Conference on*. IEEE, 2011.

[22] Youseff, Lamia, Maria Butrico, and Dilma Da Silva. "Toward a unified ontology of cloud computing." *Grid Computing Environments Workshop, 2008. GCE'08*. IEEE, 2008.

[23] F. Leymann, “Cloud Computing: The Next Revolution in IT,” in Proceedings of the 52th Photogrammetric Week, 2009, pp.3–12.

[24] NIST, NIST Definition of cloud computing v15, NIST, Editor. 2009, National Institute of Standards and Technology: Gaithersburg, MD (2009).

[25] Zhang, Qi, Lu Cheng, and Raouf Boutaba. "Cloud computing: state-of-the-art and research challenges." Journal of internet services and applications 1.1 (2010): 7-18.

[26] Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L., Leaf, D. (2011) 'NIST Cloud Computing Reference Architecture' National Institute of Standards and Technology, Special Publication 500-292

[27] OpenCrowd, http://cloudtaxonomy.opencrowd.com/, accessed December 7, 2012.

[28] Han, T., Sim, K.W. (2010) An Ontology-enhanced Cloud Service Discovery System, in Proceedings of the International MultiConference of Engineers and Computer Scientists 2010 (IMECS 2010).

[29] Rimal, B.P., Eunmi, C., Lumb, I. (2010) A Taxonomy, Survey, and Issues of Cloud Computing EcoSystems, in Journal of Computer Commications and Networks, 0(0) pp. 21-46. Springer-Verlag.

[30] Höfer, C.N. and Karagiannis, G. (2011) Cloud computing services: taxonomy and comparison. Journal of Internet Services and Applications, 2 (2). pp. 81-94

[31] Li H, Spence C, Armstrong R, Godfrey R, Schneider R, Smith J, White R (2010) Intel cloud computing taxonomy and ecosystem analysis. IT-Intel Brief (Cloud Computing).

[32] Voorsluys, William; Broberg, James; Buyya, Rajkumar (February 2011). "Introduction to Cloud Computing". In R. Buyya, J. Broberg, A.Goscinski. Cloud Computing: Principles and Paradigms. New York, USA: Wiley Press.pp. 1–44.

[33] Amazon EC2online available at: http://aws.amazon.com/ec2/

[34] Microsoft Azure online available at: http://azure.microsoft.com/el-gr/

[35] Google App Engine Cloud Platform online available at: https://cloud.google.com/appengine/

[36] Flexiant Cloud Provider online available at: http://www.flexiant.com/

[37] Armbrust, Michael, et al. "A view of cloud computing." *Communications of the ACM* 53.4 (2010): 50-58.

[38] Li, Ang, et al. "CloudCmp: comparing public cloud providers." *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 2010.

[39] BENEDICT, S., 2012. Performance issues and performance analysis tools for HPC cloud applications: a survey. Vienna: Springer.

[40] Memory speed more important for data-intensive applications such DBMSs or MapReduce.

[41] G. Kousiouris, T. Cucinotta, T. Varvarigou, The effects of scheduling, wor-kload type and consolidation scenarios on virtual machine performance and their prediction through optimized artificial neural networks, The Journal of Systems and Software, vol. 84, 2011, pp. 1270-1291.

[42] PaaS/IaaS Metamodeling Requirements and SOTA online available at: www.neotextus.net/papers/ispass07/

[43] https://www.usenix.org/sites/default/files/conference/protected-files/ou_hotcloud12_slides.pdf

[44] Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu. An analysis of performance interference effects in virtual environments. In IEEE In-ternational Symposium on Performance Analysis of Systems and Soft-ware (ISPASS), pages 200–209, April 2007.

[45] PaaS/IaaS Metamodeling Requirements and SOTA online available at: www.neotextus.net/papers/ispass07/

[46] Dixit, Kaivalya M. "Overview of the SPEC Benchmarks." (1993): 489-521.

[47] Brown, Aaron B., et al. "Benchmarking autonomic capabilities: Promises and pitfalls." *Autonomic Computing, 2004. Proceedings. International Conference on*. IEEE, 2004.

[48] Carsten Binnig, Donald Kossmann, Tim Kraska, and Simon Loesing. How is the Weather tomorrow? Towards a Benchmark for the Cloud. In Proceedings of the 2nd International Workshop on Testing Database Systems (DBtEST ,09),Providence, Rhode Island, June 2009.

[49] Cloud Service Measurement Index Consortium (CSMIC), SMI Framework, http://www.cloudcommons.com/web/cc/SMIintro.

[50] Standard Performance Evaluation Corporation online available at: http://www.spec.org/

[51] European Telecommunications Standards online available at: http://www.etsi.org/

[52] Chohan, N., Bunch, C., Pang, S., Krintz, C., Mostafa, N., Soman, S., & Wolski, R. (2009). AppScale design and implementation.

[53] Open benchmarking online available at: openbenchmarking.org

[54] https://cloudsleuth.net/

[55] Iosup, Alexandru, et al. "Performance analysis of cloud computing ser-vices for many-tasks scientific computing." *Parallel and Distributed Sys-tems, IEEE Transactions on* 22.6 (2011): 931- 945.conferences.sigcomm.org/imc/2010/papers/p1.pdf

[56] Iosup, Alexandru, Radu Prodan, and Dick Epema. "IaaS cloud benchmarking:

approaches, challenges, and experience." HotTopiCS. 2013.

[57] Ranking of Cloud Computing Services," Future Generation Computer Systems, Vol. 29, No. 4, pp. 1012-1023, 2013.

[58] Bienia, S. Kumar, J. P. Singh, and K. Li. The PARSEC benchmark suite: Characterization and architectural implications. In Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques, Oct 2008.

[59] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer,S.-H. Lee, and K. Skad-ron. Rodinia: A benchmarksuite for heterogeneous computing. In Pro-ceedings of the 2009 IEEE International Symposium on Workload Characterization (IISWC '09), pages 44–54, Austin, TX, USA, 2009. IEEE.

[60]] J. Thompson and K. Schlachte, "An Introduction to the OpenCL Programming Model", Person Education,

(2012).

[61] BENEDICT, S., 2012. Performance issues and performance analysis tools for HPC cloud applications: a survey. Vienna: Springer.

[62] Nasa Advanced Supercomputing Division available online at: http://www.nas.nasa.gov/publications/npb.html

[63] MathWorks available online at: http://www.mathworks.com/help/matlab/ref/bench.html

[64] The Landscape of Parallel Computing Research: A View from Berkeley available at: http://eugen.leitl.org/comp/EECS-2006-183.pdf

[65] Filebench: http://filebench.sf.net.

[66] Sobie RJ, Agarwal A, Anderson M, Armstrong P, Fransham K, Gable I, Harris D, Leavett-Brown C, Paterson M, Penfold-Brown D, Vliet M, Charbonneau A, Impey R, Podaima W (2011) Data intensive high energy physics analysis in a distributed cloud. http://arxiv.org/abs/1101.0357. Accessed 31 Aug 2012.

[67] Sakr Sherif, Liu Anna, Batista Daniel M, Alomari Mohammad (2011) A survey of large scale data management approaches in cloud environments. IEEE Commun Surv Tutor 13(3):311–335.

[68] Davies, Teresa, et al. "High performance linpack benchmark: a fault tolerant implementation without checkpointing." *Proceedings of the international conference on Supercomputing*. ACM, 2011.

[69] Jackson, Keith R., et al. "Performance analysis of high performance computing applications on the amazon web services cloud." Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on. IEEE, 2010.

[70] Filebench Home Page - http://sourceforge.net/apps/mediawiki/filebench/index.php? title=Filebench

[71] Filebench WML - http://sourceforge.net/apps/mediawiki/filebench/index.php? title=Workload_Model_Language.

[72] YCSB benchmark suite online available at: http://research.yahoo.com/Web_Information_Management/YCSB

[73] S. R. Chidamber and C. F. Kemerer. A metrics suite for object-oriented design. IEEE Transactions on Software Engineering, 20(6):476–493, June 1994.

[74] Blackburn, Stephen M., et al. "The DaCapo benchmarks: Java benchmarking development and analysis." *ACM Sigplan Notices*. Vol. 41. No. 10. ACM, 2006.

[75] DaCappo Benchmarking Suite, Available at: http://www.dacapobench.org/

[76] Hauck, M., Huber, M., Klems, M., Kounev, S., Muller-Quade, J., Pretschner, A., Reussner, R., and Tai, S. Challenges and opportunities of Cloud computing. Karlsruhe

Reports in Informatics 19, Karlsruhe Institute of Technology - Faculty of Informatics, 2010.

[77] George Kousiouris, Dimosthenis Kyriazis, Andreas Menychtas and Theodora Varvarigou, "Legacy Applications on the Cloud: Challenges and enablers focusing on application performance analysis and providers characteristics", in Proceedings of the 2012 2nd IEEE International Conference on Cloud Computing and Intelligence Systems (IEEE CCIS 2012), Oct. 30th ~ Nov. 1st,Hangzhou, China.

[78] Aleksandar Milenkoski, Alexandru Iosup, Samuel Kounev, Kai Sachs, Piotr Rygielski, Jason Ding, Walfredo Cirne, and Florian Rosenberg. Cloud Usage Patterns: A Formalism for Description of Cloud Usage Scenarios. Technical Report SPEC-RG-2013-001 v.1.0.1, SPEC Research Group - Cloud Working Group, Standard Performance Evaluation Corporation (SPEC), April 2013.

[79] ARTIST Consortium (2013), Deliverable D7.2 v1.0- PaaS/IaaS Metamodelling Requirements and SOTA, Available at: http://www.artistproject.eu/sites/default/files/D7.2%20PaaS%20IaaS%20metamodeling%20requi rements%20and%20SOTA_M4_31012013.pdf.

[80] Folkerts, E., Alexandrov, A., Sachs, K., Iosup, A., Markl, V., & Tosun, C. (2013). Benchmarking in the cloud: What it should, can, and cannot be. InSelected Topics in Performance Evaluation and Benchmarking (pp. 173- 188). Springer Berlin Heidelberg.

[81] REMICS project-available online at: http://www.remics.eu/system/files/REMICS_D6.6.lowres.pdf.

[82] Fuentes, L. and Vallecillo, A. (2004): An Introduction to UML Profiles. UPGRADE 2(2): 6-13.

[83] Eirik Brandzæg, Parastoo Mohagheghi, and Sébastien Mosser. Towards a Domain-Specific Language to Deploy Applications in the Cloud. In Proc. Intl. Conf. on Cloud Computing, GRIDs, and Virtualization (CLOUD COMPUTING), pages 213–218, 2012.

[84] REMICS EU project available at: http://www.remics.eu/

[85] Object Management Group online available at: http://www.omg.org/

[86] UML 2.3 online available at: http://www.omg.org/spec/UML/2.3/

[87] Amazon Cloud Formation online available at: https://aws.amazon.com/cloudformation/

[88] REMICS Consortium (2012), Deliverable D4.1 v1.0- REuse and Migration of legacy applications to Interoperable Cloud Services, Available at: http://www.remics.eu/system/files/REMICS_D4.1_V2.0_LowResolution.pdf.

[89] The Fast Guide to Model Driven Architecture(OMG) online available at: http://www.omg.org/mda/mda_files/Cephas_MDA_Fast_Guide.pdf

[90] Glauco Gonçalves, Patricia Endo, Marcelos Santos, Djamel Sadok, Judith Kelner, Bob Merlander, and Jan-Erik Mångs. CloudML: An Integrated Language for Resource, Service and Request Description for D-Clouds. In Proc. Intl. Conf. on Cloud Computing Technologies and Science (CloudCom), pages 399–406, 2011.

[91] http://www.omg.org/spec/SysML/1.3/

[92] Sanford Friedenthal, Alan Moore, Rick Steiner: OMG Systems Modeling Language (OMG SysML) Tutorial, 2008.

[93] http://www.w3.org/Math/

[94] Object Constraint Language available at: http://www.omg.org/spec/OCL/2.0/

[95] FUML language description available at: http://www.omg.org/spec/FUML/1.0/, 2011.

[96] 103] B. Selic. The less well known UML. In Formal Methods for MDE, volume 7320 of LNCS, pages 1-20. Springer Berlin / Heidelberg, 2012.

[97] moLiz project available online at: http://www.modelexecution.org

[98] Mayerhofer, T., Langer, P., and Wimmer, M. Towards xMOF: Executable DSMLs based on fUML. In Proceedings of the 12th Workshop on Domain-Specific Modeling (DSM'12) at SPLASH 2012.

[99] Filippo Bosi et al, Cloud4SOA Semantic Layer, Cloud4SOA deliverable, 2011.

[100] cloud4SOA EU project online available at: http://www.cloud4soa.com/

[101] O. Corcho, M. Fernández-lópez, A. Gómez-pérez, and A. López, "Building legal ontologies with METHONTOLOGY and WebODE," in Law and the Semantic Web, number 3369 in LNAI: Springer-Verlag, 2005, pp. 142--157.

[102] The Open Group - SOA WG Open SOA Ontology TC – "Service- Oriented Architecture Ontology" – TS C104, October 2010.

[103] Essential Meta-Model – an ontology for the domain of enterprise architecture – http://www.enterprise-architecture.org/about/35- essential-meta-model.

[104] Togaf 9 Core Content Metamodel – An OWL Ontology for the TOGAF 9 Core ContentMetamodel - http://sites.google.com/site/ontologyprojects/home/togaf-core-content-metamodel.

[105] Nguyen, D.K., Lelli, F., Taher, Y., Parkin, M., Papazoglou, M.P., van den Heuvel, W.-J.: Blueprint Template Support for Engineering Cloud- Based Services. In: Abramowicz, W., Llorente, I.M., Surridge, M., Zisman, A., Vayssière, J. (eds.) ServiceWave 2011. LNCS, vol. 6994, pp. 26–37. Springer, Heidelberg (2011).

[106] Cloud Services and Performance Analysis Framewokr online available at: http://www.artist-project.eu/sites/default/files/D7.2.1%20Cloud%20services%20modeling%20and%20performance%20analysis%20framework_M12_30092 013.pdf

[107] Apache Libcloud online available at: https://libcloud.apache.org/

[108] Kousiouris G., Giammatteo G., Evangelinou A., Galante N., Kevani E., Stampoltas C., Menychtas A., Kopaneli A., Ramasamy Balraj K., Kyriazis D., Varvarigou T., Stuer P., Orue-Echevarria Arrieta L. A Multi-Cloud Framework for Measuring and Describing Performance Aspects of Cloud Services Across Different Application Types, In Proc. of MultiCloud 2014.

[109] Pidstat tool online available at: http://sebastien.godard.pagespersoorange.fr/man_Pidstat.html.

[110] Tshark tool online available at: https://www.wireshark.org/docs/manpages/Tshark.html.

[111] Wireshark online available at: https://www.wireshark.org/

[112] I. Altintas, A. Birnbaum, K. Baldridge, W. Sudholt, M. Miller, C. Amoreira, Y. Potier, and B. Ludaescher, "A Framework for the Design and Reuse of Grid Workflows", International Workshop on Scientific Applications on Grid Computing (SAG'04), LNCS 3458, Springer, 2005.

[113] E. Deelman, J. Blythe, Y. Gil, and C. Kesselman, "Workflow Management in GriPhyN", The Grid Resource Management, Kluwer, Netherlands, 2003.

[114] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, S. Patil, M. H. Su, K. Vahi, M. Livny, "Pegasus: Mapping Scientific Workflow onto the Grid", Across Grids Conference 2004, Nicosia, Cyprus, 2004.

[115] B. Ludäscher, I. Altintas, and A. Gupta, "Compiling Abstract Scientific Workflows into Web Service Workflows", 15th International Conference on Scientific and Statistical Database Management, Cambridge, Massachusetts, USA., IEEE CS Press, pp. 241-244, Los Alamitos, CA, USA., July 09-11, 2003

[116] Jia Yu, Rajkumar Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing", Journal of Grid Computing, Volume 3, Issue 3 - 4, pp. 171-200, Springer, 2005

[117] M. Surridge, S. Taylor, D. De Roure, and E. Zaluska, "Experiences with GRIA-Industrial Applications on a Web Services Grid", in Proceedings of the First International Conference on e-Science and Grid Computing, pp. 98-105. IEEE Press, 2005

[118] GRIA, Grid Resources for Industrial Applications, www.gria.org

[119] G. Bochmann and A. Hafid. "Some Principles for Quality of Service Management", Technical report, Universite de Montreal, 1996.

[120] R. J. Al-Ali, K. Amin, G. von Laszewski, O. F. Rana, D. W. Walker, M. Hategan, N. J. Zaluzec: "Analysis and Provision of QoS for Distributed Grid Applications", pp. 163-182, Journal of Grid Computing, 2004

[121] Padgett, J., K. Djemame, and P. Dew, "Grid-based SLA Management", Lecture Notes in Computer Science, pp. 1282-1291, 2005

[122] I. Foster, C. Kesselman, C. Lee, B Lindell, K. Nahrstedt, A. Roy, "A Distributed Resource Management Architecture that Supports Advance Reservation and Co-Allocation", Proceedings of the International Workshop on QoS, pp.27-36, 1999

[123] Min-You Wu, Wei Shu, H. Zhang, "Segmented min-min: a static mapping algorithm for meta-tasks on heterogeneous computing systems", Heterogeneous Computing Workshop, 2000

[124] Shanshan Song, Yu-Kwong Kwok, Kai Hwang, "Security- Driven Heuristics and A Fast Genetic Algorithm for Trusted Grid Job Scheduling Parallel and Distributed Processing Symposium", 19th IEEE International 04-08 April 2005

[125] R. Buyya, M. Murshed, D. Abramson, "A Deadline and Budget Constrained Cost-Time Optimization Algorithm for Scheduling Task Farming Applications on Global Grids", Proceedings of the 2002 International Conference on Parallel and Distributed Processing Techniques and Applications(PDPTA702), 2002

[126] R. Buyya, D. Abramson, S. Venugopal, "The Grid Economy", Proceedings of the IEEE Volume 93, Issue 3, pp. 698-714, Mar 2005

[127] Li Kenli, Tang Xiaoyong, Zhaohuan, "Grid Classified Optimization Scheduling Algorithm under the Limitation of Cost and Time", Proceedings of the Second International Conference on Embedded Software and Systems (ICESS'05), 0-7695-2512-1/05 IEEE

[128] Jorge Cardoso, Amit Sheth and John Miller, "Workflow Quality of Service", Proceedings of the International Conference on Enterprise Integration and Modeling Technology and International Enterprise Modeling Conference (ICEIMT/IEMC'02), Kluwer Publishers, April 2002

[129] J. Cardoso, J. Miller, A. Sheth, J. Arnold, "Modeling Quality of Service for Workflows and Web Service Processes", Technical Report, LSDIS Lab, Department of Computer Science University of Georgia, 2002

[130] D. Angulo, I. Foster, C. Liu, and L. Yang, "Design and Evaluation of a Resource Selection Framework for Grid Applications", Proceedings of IEEE International Symposium on High Performance Distributed Computing (HPDC-11), Edinburgh, Scotland, 2002

[131] R. Buyya, D. Abramson, J. Giddy, "Economy Driven Resource Management Architecture for Computational Power Grids", International Conference on Parallel and Distributed Processing Techniques and Applications, Las Vegas, USA., 2000

[132] R. Buyya, D. Abramson, J. Giddy, "High Nimrod-G: an architecture for a resource management and scheduling system in a global computational grid", Performance Computing in the Asia- Pacific Region, Proceedings of the Fourth International Conference Exhibition on Volume 1, pp. 283-289, 14-17 May 2000

[133] DAGMan, http://www.cs.wisc.edu/condor/dagman/

[134] C. Weng, X. Lu, "Heuristic scheduling for bag-of-tasks applications in combination with QoS in computational grid", Future Generation Computer Systems, pp. 271-280, 2005

[135] V. Subramani, R. Ketiimuthu, S. Srinivasan, P. Sadayappan, "Distributed job scheduling on computational grids using multiple simultaneous requests", Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing, pp. 359-367, Edinburgh, Scotland, 2002

[136] Open Grid Services Architecture (OGSA), www.ggf.org/documents/GFD.30.pdf

[137] WSRF, The Web Services Resource Framework (WSRF) v1.2, http://www.oasis-open.org/committees/download.php/17833/wsrf-1.2-os.zip

[138] Air Renderer: http://www.sitexgraphics.com/html/air.html

[139] RenderMan Interface Specification v3.2, https://renderman.pixar.com/products/rispec/rispec_pdf/RISpec3_2.pdf

[140] K. Dolkas, D. Kyriazis, A. Menychtas, T. Varvarigou, "e-Business Applications on the Grid: a toolkit for centralized workload prediction and access", Concurrency and Computation: Practice and Experience, Wiley Interscience, 2006

[141] C. Skianis, G. Xilouris, G. Kormentzas, A. Kourtis, "A Testbed Environment for Validation of End-to-End QoS Provision for the Content Delivery Chain over Heterogeneous Systems", Proceedings of the 2nd International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs'04), Ilkley, pp. 89/1-89/8, 2004

[142] Pawel Plaszczak and Richard Wellner,Jr., Grid Computing – The Savvy's Manager Guide, Morgan Kaufmann Publishers, 2006, ISBN-13:978-0-1274-2503-0

[143] Dimosthenis Kyriazis, Konstantinos Dolkas, Andreas Menychtas and Theodora Varvarigou, "A new Workflow Mapping Mechanism for Grids", IST Mobile & Wireless Communication Summit 2006, Myconos - Greece, 04-08 June 2006.

[144] Fran Berman, Anthony J. G. Hey, Geoffrey C. Fox, Grid Computing – Making the Global Infrastructure A Reality, John Wiley and Sons Ltd, 2003, ISBN 0-470-85319-0

[145] A. Sahai and S. Graupner and V. Machiraju and A. Moorsel. (Specifying and Monitoring) Guarantees in Commercial Grids through SLA. Proceedings of the 3rd IEEE/ACM CCGrid2003, 2003.

[146] K. Czajkowski and I. Foster and C. Kesselman and V. Sander and S. Tuecke SNAP: A Protocol for Negotiating Service Level Agreements and Coordinating Resource Management in Distributed Systems. Proceedings of the 8th Workshop on Job Scheduling Strategies for Parallel Processing, 2002.

[147] The Business Grid: Providing Transactional Business Processes via Grid Services Frank Leymannand, Kai Güntzel.

[148] Open Grid Forum, www.ogf.org

[149] The World Wide Web Consortium, www.w3.org

[150] Grimoires model, http://twiki.grimoires.org/bin/view/Grimoires/

[151] RDQL - A Query Language for RDF, http://www.w3.org/Submission/RDQL/

[152] Peer Hasselmeyer, Changtao Qu, Lutz Schubert, Bastian Koller, and Philipp Wieder. Towards Autonomous Brokered SLA Negotiation. In: Proceedings of the eChallenges e-2006 Conference, Barcelona, Spain, October 2006.

[153] Konstantinos Tserpes, Dimosthenis Kyriazis, Andreas Menychtas, Theodora Varvarigou, Fabrizio Silvestri, Domenico Laforenza, "Business Environments and QoS: Correlating Customer Requirements with Provider Capabilities", CoreGRID'07 Symposium, 2007.

[154] Duan-Shin Lee, Chun-Min Chen, Weighted fair queueing and compensation techniques for wireless packet switched networks. The 5th International Symposium on

Wireless Personal Multimedia Communications, 2002. Volume: 3, 27-30 Oct. 2002 Page(s): 887 -891.

[155] Bennett, J.C.R., Hui Zhang, WF2Q: worst-case fair weighted fair queueing INFOCOM '96. Proceedings IEEE on Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Volume: 1, 24-28 March 1996 Page(s): 120 - 128 vol.1.

[156] Dimosthenis Kyriazis, Konstantinos Tserpes, Andreas Menychtas, Antonios Litke and Theodora Varvarigou, “An innovative Workflow Mapping Mechanism for Grids in the frame of Quality of Service”, Future Generation Computer Systems (The International Journal of Grid Computing), Elsevier, 2007.

[157] Peha, J.M., Tobagi, F.A., Evaluating scheduling algorithms for traffic with heterogeneous performance objectives. Global Telecommunications Conference, 1990, and Exhibition. 'Communications: Connecting the Future', GLOBECOM '90, IEEE , 2-5 Dec. 1990 Page(s): 21 -27 vol.1

[158] Andreas Menychtas, Konstantinos Dolkas, Dimosthenis Kyriazis and Theodora Varvarigou, “Building Portals to access Grid Middleware”, GGF 14 Conference Chicago, USA, 27-30 June 2005.

[159] Spuri M, Buttazzo G.C., Efficient aperiodic service under earliest deadline scheduling. Real-Time Systems Symposium, 1994, Proceedings. , 7-9 Dec. 1994 Page(s): 2 –11.

[160] Sung-Heun Oh, Seung-Min Yang, A Modified Least-Laxity-First scheduling algorithm for real-time tasks. Proceedings. Fifth International Conference on Real-Time Computing Systems and Applications, 1998., 27-29 Oct. 1998 Page(s): 31 –36.

[161] Kreuzinger J., Schulz A., Pfeffer M., Ungerer T., Brinkschulte U., Krakowski C., “Real-time scheduling on multithreaded processors”. Seventh International Conference on Real-Time Computing Systems and Applications, 2000., 12-14 Dec. 2000 Page(s): 155 – 159.

[162] Dimosthenis Kyriazis, Konstantinos Tserpes, Andreas Menychtas, Ioannis Sarantidis and Theodora Varvarigou, “Service Selection and Workflow Mapping for Grids: An approach exploiting Quality of Service Information”, Concurrency and Computation: Practice and Experience, Willey Interscience, 2007.

[163] Al-Mouhamed M.A., Lower bound on the number of processors and time for scheduling precedence graphs with communication costs. IEEE Transactions on Software Engineering, Volume: 16 Issue: 12, Dec. 1990 Page(s): 1390 –1401.

[164] NextGRID, www.nextgrid.org

[165] Konstantinos Tserpes, Dimosthenis Kyriazis, Andreas Menychtas and Theodora Varvarigou, "A Novel Mechanism for Provisioning of High-Level Quality of Service Information in Grid Environments", Special Issue on "Performance Evaluation of QoS-aware Heterogeneous Systems, European Journal of Operational Research, 2007.

[166] Haddad E., Optimal load sharing in dynamically heterogeneous systems. Proceedings. Seventh IEEE Symposium on Parallel and Distributed Processing, 1995. , 25-28 Oct. 1995 Page(s): 346 –353.

[167] Antonios Litke, Dimitrios Halkos, Konstantinos Tserpes, Dimosthenis Kyriazis and Theodora Varvarigou, "Fault Tolerant and Prioritized Scheduling in OGSA-based Mobile Grids", Concurrency and Computation: Practice and Experience, Willey Interscience, 2007.

[168] Tei-Wei Ku, Wang-Ru Yang, Kwei-Jay Lin, A class of rate-based real-time scheduling algorithms. IEEE Transactions on Computers, Volume: 51 Issue: 6, June 2002 Page(s): 708 –720.

[169] Jeffay K., Stanat D.F., Martel C.U., On non-preemptive scheduling of period and sporadic tasks. Real-Time Systems Symposium,1991.Proceedings.Twelfth, 4-6 Dec.1991 Page(s): 129-139.

[170] A. Menychtas, D. Apostolopoulos, D. Kyriazis, K. Christodoulopoulos, H. Avramopoulos and Theodora Varvarigou, "Enabling a Network Simulation Application on Grid Infrastructure", 11th Panhellenic Conference on Informatics (PCI 2007), Patras, Greece, 2007.

[171] Saez S., Vila J., Crespo A., A dynamic real-time scheduler for shared memory multiprocessors. Proceedings of the Eighth Euromicro Workshop on Real-Time Systems, 1996. Page(s): 158 –163.

[172] http://www.w3.org/Submission/OWL-S/

[173] Kirsten Ferguson-Boucher, "Cloud Computing: A Records and Information Management Perspective", *IEEE Security & Privacy*, vol.9, no. 6, pp. 63-66, November/December 2011, doi:10.1109/MSP.2011.159

[174] ARTIST Deliverable D.12.1: Use cases definition and migration architecture (Appendixes)

[175] Koziolek A., Koziolek H., Reussner R. PerOpteryx: Automated Application of Tactics in Multi-objective Software Architecture Optimization. In QoSA 2011 Proc., QoSA-ISARCS ’11, pages 33–42, New York, NY, USA, 2011. ACM

[176] Ferry N., Rossini A., Chauvel F., Morin B., Solberg A. Towards model-driven provisioning, deployment, monitoring, and adaptation of multi-cloud systems. In IEEE CLOUD 2013 Proc., pages 887–894. IEEE Computer Society, 2013.

[177] Franceschelli D., Ardagna D., Ciavotta M., Di Nitto E. Space4cloud: a tool for system performance and cost evaluation of cloud systems. In Proc. Multi-Cloud ’13, pages 27–34, New York, NY, USA, 2013. ACM.

[178] Perez J., Casale G. Assessing SLA Compliance from Palladio Component Models. In SYNASC 2013 Proc., pages 409–416, Sept 2013.

[179] Becker S., Koziolek H., Reussner R. The palladio component model for model-driven performance prediction. Journal of Systems and Software, 82(1):3–22, 2009.

[180] Rolia J., Sevcik K. The method of layers. Software Engineering, IEEE Transactions on, 21(8):689–700, 1995.

[181] Franks G., Al-Omari T., Woodside M., Das O., Derisavi S. Enhanced modeling and solution of layered queueing networks. Software Engineering, IEEE Transactions on, 35(2):148–161, March 2009.

[182] Ardagna D., Gibilisco G., Ciavotta M., Lavrentev A. A multi-model optimization framework for the model driven design of cloud applications. In SBSE 2014 Proc. 2014.

[183] Glover F. Tabu search: part i. ORSA Journal on computing, 1(3):190–206, 1989.

[184] Casale G., Tribastone M., Harrison P.G. Blending randomness in closed queueing network models. Performance. Evaluation. 82: 15-38 2014.

[185] A. Stanik, M. Hovestadt, and Odej Kao. Hardware as a service (haas): The completion of the cloud stack. In Computing Technology and Information Management (ICCM), 2012 8th International Conference on,volume 2, pages 830–835, 2012.

Η δρ. Αθανασία-Χαραλαμπία Δ. Ευαγγελινού γεννήθηκε την 22η *Μαΐου* 1982 στην Αθήνα. Το 2012 αποφοίτησε από τη Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου («Καλώς») και το θέμα της διπλωματικής εργασίας της ήταν «Αξιολόγηση μεθόδων κατηγοριοποίησης δεδομένων γονιδιακής έκφρασης cDNA μικροσυστοιχιών». Η εργασία ολοκληρώθηκε υπό την επίβλεψη του Καθηγητή κ. Δημήτριου Κουτσούρη και βαθμολογήθηκε με γενικό χαρακτηρισμό «Άριστα» (10).

Τον Οκτώβριο του 2012, έγινε δεκτή για μεταπτυχιακές σπουδές που οδήγησαν στην απόκτηση Μεταπτυχιακού Διπλώματος το Νοέμβριο του 2014 με βαθμό «Άριστα» (8.71), από το Τμήμα Ψηφιακών Συστημάτων με κατεύθυνση Δικτυοκεντρικά Πληροφοριακά Συστήματα του Πανεπιστημίου Πειραιώς.

Το Νοέμβριο του 2012, έγινε δεκτή για μεταπτυχιακές σπουδές που οδηγούν στην απόκτηση Διδακτορικού Διπλώματος στη Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου.

Κατά το παρελθόν εργάστηκε στον ιδιωτικό τομέα (ΟΤΕ) καθώς επίσης από το 2007 έως το 2012 της δόθηκε η ευκαιρία να ασχοληθεί επαγγελματικά με μια σειρά Ευρωπαϊκών Προγραμμάτων στον τομέα των υποστηρικτικών τεχνολογιών επικοινωνιών και πληροφορίας (ICT-Assistive Technologies). Κατά τη διάρκεια της εκπόνησης της Διδακτορικής της Διατριβής η Αθανασία-Χαραλαμπία Δ. Ευαγγελινού εργάστηκε σε Ελληνικά και Ευρωπαϊκά Προγράμματα, στα οποία της δόθηκε η δυνατότητα να εμβαθύνει σε ερευνητικά θέματα, άμεσα συνδεδεμένα με την περιοχή του διδακτορικού της. Έλαβε μέρος σε πολλές επιστημονικές συναντήσεις στην Ελλάδα και στο εξωτερικό και συνεργάστηκε με μηχανικούς και ανώτερα στελέχη διαφόρων οργανισμών, εταιριών υπολογιστικών συστημάτων και ερευνητικών πανεπιστημιακών ομάδων. Έχει εκτελέσει σειρά από παρουσιάσεις και σεμινάρια για ζητήματα πρόβλεψης απόδοσης σε περιβάλλοντα Υπολογιστικών Νεφών. Τα αποτελέσματα της ερευνητικής της εργασίας παρουσιάστηκαν σε διεθνή συνέδρια και δημοσιεύθηκαν στον επιστημονικό Τύπο, σε περιοδικά και βιβλία.

Η Α. Ευαγγελινού είναι μέλος του Τεχνικού Επιμελητηρίου της Ελλάδος (ΤΕΕ) από το 2012.