ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
http://www.cslab.ece.ntua.gr

Διπλωματικές Εργασίες
Ακαδημαϊκό έτος 2020-21

Ι. Παράλληλα Συστήματα

1 Πολλαπλασιασμός αραιού πίνακα με διάνυσμα (SpMV)

Ο υπολογιστικός πυρήνας του πολλαπλασιασμού αραιού πίνακα με διάνυσμα (SpMV) χρησιμοποιείται ευρέως σε παράλληλες εφαρμογές μεγάλης κλίμακας. Ωστόσο, λόγω της αλγοριθμικής του φύσης, δεν αξιοποιεί επαρκώς την υπολογιστική ισχύ των σύγχρονων επεξεργαστών. Οι παρακάτω εργασίας εστιάζουν στην βελτιστοποίηση του με σε διαφορετικές αρχιτεκτονικές με τη χρήση των κατάλληλων προγραμματιστικών μοντέλων.

1.1 Βελτιστοποίηση του υπολογιστικού πυρήνα πολλαπλασιασμού αραιού πίνακα με διάνυσμα (SpMV) σε FPGAs με τη χρήση του προγραμματιστικού μοντέλου OpenCL

Στην παρούσα διπλωματική εργασία, θα μελετηθεί η υλοποίηση και η βελτιστοποίηση του συγκεκριμένου υπολογιστικού πυρήνα σε επαναδιαμορφούμενες αρχιτεκτονικές (FPGAs), που επιτρέπουν στον προγραμματιστή τη δημιουργία υλικού εξειδικευμένου στην εφαρμογή (application-specific). Συγκεκριμένα, θα μελετηθεί η επίδοση βασικών υλοποιήσεων του SpMV για FPGAs με τη χρήση του προγραμματιστικού μοντέλου της OpenCL, εναλλακτικά σχήματα αποθήκευσης αραιών πινάκων και θα εφαρμοστούν τεχνικές βελτιστοποίησης του υπολογιστικού πυρήνα με στόχο την επίτευξη της μέγιστης δυνατής επίδοσης στις συγκεκριμένες αρχιτεκτονικές.

Σχετικά Μαθήματα: Συστήματα Παράλληλης Επεξεργασίας, Ψηφιακά Συστήματα VLSI

Επικοινωνία: Παναγιώτης Μπάκος, pmpakos@cslab.ece.ntua.gr
Νικέλα Παπαδοπούλου, nikela@cslab.ece.ntua.gr, 210-772-2279
Γεώργιος Γκούμας, goumas@cslab.ece.ntua.gr, 210-772-2402

1.2 Υλοποίηση βιβλιοθήκης για τον πολλαπλασιασμό αραιού πίνακα με διάνυσμα σε ετερογενή συστήματα με τη χρήση του προγραμματιστικού μοντέλου OmpSs

Το προγραμματιστικό μοντέλο OmpSs (`https://pm.bsc.es/ompss`) αποτελεί μια επέκταση του προγραμματιστικού μοντέλου OpenMP, που υποστηρίζει ετερογενείς αρχιτεκτονικές, μεταξύ των οποίων και επαναδιαμορφούμενες αρχιτεκτονικές (FPGAs). Η παρούσα διπλωματική θα στηριχθεί σε υπάρχουσες υλοποιήσεις του υπολογιστικού πυρήνα αραιού πίνακα με διάνυσμα (SpMV) για ετερογενείς αρχιτεκτονικές και τις ενσωματώσει σε μία βιβλιοθήκη λογισμικού, βασισμένη στο προγραμματιστικό μοντέλο OmpSs, που θα αξιοποιεί με βέλτιστο τρόπο τους διαθέσιμους υπολογιστικούς πόρους ετερογενών συστημάτων, επιλέγοντας στο χρόνο μεταγλώττισης ή/και εκτέλεσης τους καταλληλότερους υπολογιστικούς πυρήνες για την εκτέλεση εφαρμογών, με επίγνωση τόσο της αρχιτεκτονικής, όσο και των χαρακτηριστικών της εφαρμογής και του δεδομένου αραιού πίνακα.

Σχετικά Μαθήματα: Συστήματα Παράλληλης Επεξεργασίας

Επικοινωνία: Παναγιώτης Μπάκος, pmpakos@cslab.ece.ntua.gr
Νικέλα Παπαδοπούλου, nikela@cslab.ece.ntua.gr, 210-772-2279
Γεώργιος Γκούμας, goumas@cslab.ece.ntua.gr, 210-772-2402

2 Αποδοτική χρήση και προγραμματισμός GPGPU

Οι σύγχρονες μονάδες επεξεργασίας γραφικών ή κάρτες γραφικών (GPUs) έχουν εξελιχθεί από το να είναι χρήσιμες μόνο για συγκεκριμένες λειτουργίες, σε ισχυρά εργαλεία γενικής χρήσης (GPGPUs) ικανά να υποστηρίξουν μια πολύ μεγαλύτερη ποικιλία προβλημάτων, παρέχοντας μια ταχύτερη και πιο ενεργειακά αποδοτική εναλλακτική λύση σε σχέση με τους κανονικούς επεξεργαστές (CPUs).

2.1 Αξιολόγηση και σύγκριση εργαλείων για τον προγραμματισμό GPGPU.

Μαζί με την εξέλιξη των δυνατοτήτων των GPU ήρθε μια μεγάλη ποικιλία εργαλείων λογισμικού, μεταγλωττιστών και προτύπων για τον προγραμματισμό GPU, με σκοπό την αξιοποίηση των δυνατοτήτων τους από τους προγραμματιστές και τους μηχανικούς απόδοσης. Έτσι, ενώ 10 χρόνια πριν ο προγραμματισμός GPGPU ήταν σχεδόν ισοδύναμος με προγραμματισμό σε CUDA, σήμερα υπάρχει μια μεγάλη ποικιλία επιλογών (CUDA, OpenMP, OpenCl, OpenACC, oneAPI) και μια μετατόπιση του ενδιαφέροντος προς αυτές. Αυτά τα εργαλεία προσεγγίζουν τον προγραμματισμό GPGPU με διαφορετικούς τρόπους, θέτοντας διαφορετικούς στόχους όσον αφορα την ευκολία προγραμματισμού, το έυρος εφαρμογής και την απόδοση, θυσιάζοντας πάντα κάτι. Ο σκοπός της διπλωματικής αυτης είναι η εξοικίωση με ένα υποσύνολο αυτών των διαφορετικών εργαλείων και η αξιολογήση τους σε διάφορους αλγορίθμους, προκειμένου να διερευνηθούν, να εκτιμηθούν και να αξιολογηθούν οι δυνατότητες, οι αδυναμίες και τα περιθώρια του καθενός, καθώς και αποτελεσματικών μεθόδων για την έυρεση αυτων. Ορισμένα πιο συγκεκριμένα θέματα θα μπορούσαν να περιλαμβάνουν:

- Ανάπτυξη τεχνικών για τη μεταφορά κώδικα μεταξύ αυτών των εργαλείων.

- Εύρεση συγκεκριμένων σημείων συμφόρησης/αδυναμιών σε αυτά και πρόταση επιλογών για την αποφυγή τους.
- Μοντελοποίηση της απόδοσής τους.

Σχετικά Μαθήματα: Συστήματα Παράλληλης Επεξεργασίας **Σχετική Βιβλιογραφία**:

1. https://docs.nvidia.com/cuda/
2. https://www.openmp.org/
3. https://www.khronos.org/opencl/
4. https://www.openacc.org/
5. https://www.oneapi.com/

Επικοινωνία: Αναστασιάδης Πέτρος, panastas@cslab.ece.ntua.gr

2.2 Βελτιστοποίηση διαχείρισης μνήμης GPU

Οι περισσότερες GPU (και **όλες** οι GPU στοχευμένες για High performance computing (HPC) μέχρι σήμερα) χρησιμοποιούν ένα διαφορετικό σύστημα μνήμης από τη μνήμη των CPU. Η απόδοση των GPU εξαρτάται πολύ από το εύρος ζώνης (bandwidth) της μνήμης τους, το οποίο θέτει μια σοβαρή αντιστάθμιση μεταξύ της τιμής, του εύρους ζώνης και του μεγέθους της μνήμης που συνήθως οδηγεί στο οι GPUs να έχουν μικρές χωρητικότητες μνήμης. Επομένως, είναι σύνηθες η μνήμη της CPU να είναι πολύ μεγαλύτερη, ιδαίτερα στην περίπτωση των υπερυπολογιστικών HPC συστημάτων. Το εργαλείο προγραμματισμού CUDA, ο πιο ευρέως χρησιμοποιούμενος τρόπος για τον προγραμματισμό GPU της NVIDIA μέχρι σήμερα, προκειμένου να διευκολυνθεί η ανάπτυξη κώδικα εισήγαγε την έννοια της ενοποιημένης μνήμης (unified memory) στο CUDA 6.0. Το unified memory αποκρύπτει το πρόσθετο βήμα των μεταφορών δεδομένων από τον προγραμματιστή, προσφέροντας ενοποιημένους δείκτες για CPU και GPU (host-device pointers). Το unified memory back-end είναι υπεύθυνο για τη μεταφορά των δεδομένων μεταξύ CPU και GPU όποτε απαιτείται. Δυστυχώς, αυτο δημιουργεί δύο βασικά προβλήματα: 1) Η απόκρυψη της μεταφορας δεδομένων δεν αλλάζει το οτι αυτή πρέπει να γίνει και επομένως μπορεί να αργεί την εκτέλεση και 2) η ενοποιημένη μνήμη δεν λύνει το πρόβλημα της μικρής χωρητικότητας των GPU - εάν ένα πρόβλημα δεν χωράει στη μνήμη της GPU, ο προγραμματιστής πρέπει να τεμαχίσει τα δεδομένα ο ίδιος. Σκοπός αυτής της διπλωματικής είναι η διερεύνηση της χρήσης της ενοποιημένης μνήμης CUDA και των προβλημάτων της, προκειμένου να δοθεί μια λύση σχετικά με αυτά. Ορισμένα πιο συγκεκριμένα θέματα θα μπορούσαν να περιλαμβάνουν:

- Εξετάση/δοκιμή της χρήση έτοιμων σωρών μνήμης GPU με σκοπό την γρήγορη αξιοποίηση μνήμης.
- Εξερεύνηση ένος μηχανισμού 'smart-prefetch' με στόχο την εκτέλεση μεταφορών ενώ το μέσο επικοινωνίας CPU-GPU είναι κενό.
- Αυτοματοποίηση διοχέτευσης λογισμικού (software-pipelining) για μεγάλα προβλήματα που δεν χωράνε στη μνήμη GPU.

Σχετικά Μαθήματα: Συστήματα Παράλληλης Επεξεργασίας

Σχετική Βιβλιογραφία:

1. https://docs.nvidia.com/cuda/
2. https://developer.nvidia.com/blog/unified-memory-cuda-beginners/
3. https://developer.nvidia.com/blog/maximizing-unified-memory-performance-cuda/
4. "Where is the data? Why you cannot debate CPU vs. GPU performance without the answer"
5. "Evaluating characteristics of CUDA communication primitives on high-bandwidth interconnects"
6. "Software pipelining for graphic processing unit acceleration: Partition, scheduling and granularity"
7. "Performance Models for CPU-GPU Data Transfers"
8. "Benchmarking and evaluating unified memory for OpenMP GPU offloading"

Επικοινωνία: Αναστασιάδης Πέτρος, panastas@cslab.ece.ntua.gr

3 Δρομολόγηση Εφαρμογών και Διαχείριση Πόρων σε Υπολογιστικά Κέντρα

Οι υπολογιστικές υποδομές των Υπολογιστικών Κέντρων (Datacenters) χρησιμοποιούνται για την ταυτόχρονη εκτέλεση εφαρμογών. Η κατάλληλη χρονοδρομολόγηση και η διαχείριση των κοινόχρηστων πόρων του συστήματων αποτελούν καθοριστικούς παράγοντες για την αποτελεσματική χρήση των υπολογιστικών πόρων και την εξοικονόμηση χρόνου και ενέργειας.

3.1 Διαχείριση Πόρων σε Συστήματα Μεγάλης Κλίμακας

Καθώς η εκτέλεση πολλών τύπων υπηρεσιών μεταφέρεται σε συστήματα μεγάλης κλίμακας, η πρόκληση της διατήρησης υψηλής ποιότητας υπηρεσίας συνεχώς μεγαλώνει. Η απουσία αποδοτικών λύσεων διαμοιρασμού των κοινόχρηστων πόρων οδηγεί τους Cloud Service Providers στην απομόνωση ολόκληρων servers για την εκτέλεση εφαρμογών με αυστηρούς περιορισμούς για την επίδοση τους. Αυτό όμως οδηγεί στην υποχρησιμοποίηση αυτών των πόρων και την αύξηση του λειτουργικού κόστους. Για την αντιμετώπιση των ζητημάτων αυτών προτείνονται τεχνικές διαχείρισης των κοινόχρηστων πόρων (Last Level Cache - Intel CMT CAT, Memory Bandwidth, Core isolation), τεχνικές χαρακτηρισμού των εφαρμογών ως προς τους κρίσιμους πόρους με σκοπό τη συνεκτέλεση εφαρμογών με συμπληρωματικές απαιτήσεις για πόρους, και τεχνικές εντοπισμού μειωμένης επίδοσης κατά το χρόνο εκτέλεσης.

3.1.1 Διαχείριση πόρων σε Kubernetes clusters με χρήση resource managers της Intel

Η χρήση containers και του Kubernetes ως πλατφόρμας διαχείρισης τους φαίνεται να επικρατεί στη βιομηχανία τα τελευταία χρόνια έναντι της χρήσης VMs, επομένως η μελέτη των επιλογών που προσφέρονται για την αποδοτική εκτέλεση πολλών containers σε ένα υπολογιστικό κόμβο είναι αναμφισβήτητα η επόμενη πρόκληση. Η Intel αναπτύσσει ήδη Resource Managers (Intel Telemetry Aware Scheduling, CRI Resource Manager, Workload Collocation Agent, Platform Resource Manager, CPU Manager for Kubernetes) προς αυτή την κατεύθυνση.
Σχετική Βιβλιογραφία:

1. Intel Resource Director Technology

2. Intel Telemetry Aware Scheduling

Επικοινωνία: Γιάννης Παπαδάκης, ypap@cslab.ece.ntua.gr
Κωστής Νίκας, knikas@cslab.ece.ntua.gr, 210-772-4159

3.1.2 Αποδοτική δρομολόγηση εφαρμογών υλοποιημένων με τη μορφή microservices

Αρκετές εφαρμογές μεγάλης κλίμακας αποτελούνται πλέον από μεγάλο αριθμό μικρότερων και απλούστερων εφαρμογών που επικοινωνούν μεταξύ τους ώστε να επιτευχθεί η επιθυμητή λειτουργικότητα. Τα πλεονεκτήματα αυτής της προσέγγισης περιλαμβάνουν την ευκολότερη συντήρηση και αποσφαλμάτωση της εφαρμογής και την εκμετάλλευση της ετερογένειας γλωσσών προγραμματισμού και προγραμματιστικών διεπαφών. Twitter, Netflix και eBay μεταξύ άλλων υιοθετούν αυτό το μοντέλο. Η διασφάλιση της διατήρησης της επίδοσης τέτοιων εφαρμογών σε συστήματα μεγάλης κλίμακας είναι μια πρόκληση η οποία προσφέρει πολλές ευκαιρίες έρευνας.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

1. Gan, Y., & Delimitrou, C. (2018). The Architectural Implications of Cloud Microservices. IEEE Computer Architecture Letters, 17(2), 155–158.

2. Microservices Benchmarks by Cornell Un.

Επικοινωνία: Γιάννης Παπαδάκης, ypap@cslab.ece.ntua.gr
Κωστής Νίκας, knikas@cslab.ece.ntua.gr, 210-772-4159

3.2 Βέλτιστη αξιοποίηση υπολογιστικών πόρων σε συστήματα μεγάλης κλίμακας

Σε συστήματα μεγάλης κλίμακας υψηλών υπολογιστικών επιδόσεων, οι αλγόριθμοι δρομολόγησης εργασιών, όπως ο Back-Filling, για να λάβουν αποφάσεις, αξιοποιούν την πληροφορία που τους παρέχουν οι χρήστες, οι οποίοι, καθώς υποβάλουν την εργασία τους, αιτούνται τους απαραίτητους υπολογιστικούς πόρους (κόμβους, πυρήνες, μνήμη, επιταχυντές) και παρέχουν μια εκτίμηση για το χρόνο ολοκλήρωσης των εργασιών τους. Όσο πιο ακριβής είναι αυτή η πληροφορία, τόσο καλύτερη είναι η αξιοποίηση του συστήματος (throughput) και η ικανοποίηση των χρηστών (χαμηλοί χρόνοι αναμονής). Ωστόσο, καθώς οι εφαρμογές που εκτελούνται συχνά περιλαμβάνουν χιλιάδες γραμμές κώδικα και χρησιμοποιούν αρκετές επιπλέον βιβλιοθήκες και υπολογιστικά πακέτα, οι χρήστες δεν είναι πάντα σε θέση να εκτιμήσουν σωστά την επίδοση της εφαρμογής τους και τον αναμενόμενο χρόνο εκτέλεσής της. Έτσι, υποβάλουν εκτιμήσεις που είναι ανακριβείς ως προς τους ζητούμενους πόρους και καταλήγουν σε σπατάλη πόρων (π.χ. ο χρήστης θα μπορούσε να είχε λάβει αντίστοιχο χρόνο εκτέλεσης με λιγότερους υπολογιστικούς πόρους). Σε σχέση με τους χρόνους εκτέλεσης, οι χρήστες κατά κανόνα υπερεκτιμούν τον αναμενόμενο χρόνο εκτέλεσης της εφαρμογής τους.

Στην παρούσα διπλωματική, θα μελετήσουμε την επίδραση των εκτιμήσεων των χρηστών στην επίδοση του συστήματος και θα επεκτείνουμε υπάρχοντες αλγορίθμους χρονοδρομολόγησης για συστήματα μεγάλης κλίμακας με δυνατότητες διάδρασης με το χρήστη, για την εκτίμηση και επιλογή των κατάλληλων πόρων σε σχέση με την εργασία του χρήστη, με στόχο τη βέλτιστη αξιοποίηση των πόρων του συστήματος και τη μεγιστοποίηση της απόδοσης του συστήματος.

Επικοινωνία: Νικέλα Παπαδοπούλου, nikela@cslab.ece.ntua.gr
Γεώργιος Γκούμας, goumas@cslab.ece.ntua.gr, 210-772-2402

3.3 Χρονοδρομολόγηση εφαρμογών σε υπολογιστικά συστήματα υψηλής επίδοσης

Τα υπολογιστικά συστήματα υψηλής επίδοσης (High Performance Computing clusters -HPC clusters) είναι ευρέως διαδεδομένα και συχνά χρησιμοποιούνται για την επίλυση πολύπλοκων προβλημάτων σε ποικίλες ερευνητικές περιοχές όπως η πρόγνωση και η μοντελοποίηση των καιρικών φαινομένων καθώς και η διερεύνηση της ακολουθίας του ανθρώπινου γονιδιώματος. Το βασικό λογισμικό που συνθέτει μια τέτοια υπολογιστική υποδομή, ονομάζεται διαχειριστής πόρων (resource manager) και περιλαμβάνει έναν χρονοδρομολογητή εργασιών (job scheduler). Ο διαχειριστής πόρων αναλαμβάνει να διαμοιράσει τους υπολογιστικούς πόρους στις αντίστοιχες εργασίες. Ο χρονοδρομολογητής εργασιών επικοινωνεί με τον διαχειριστή πόρων προκειμένου να πληροφορηθεί για τις ουρές (queues), τα φορτία των υπολογιστικών κόμβων (nodes) και την διαθεσιμότητα των πόρων, ώστε να πάρει αποφάσεις για τη χρονοδρομολόγηση εργασιών.

3.3.1 Ανάπτυξη ανοιχτού λογισμικού εξομοιωτή χρονοδρομολόγησης εφαρμογών

Σκοπός της διπλωματικής είναι η συγγραφή ανοιχτού λογισμικού που θα επιτρέπει τη δημιουργία ενός εικονικού (virtual) διαχειριστή πόρων SLURM. Το λογισμικό θα επιτρέπει να εκτελέσει ο χρήστης MPI εφαρμογές στον εικονικό διαχειριστή πόρων SLURM ο οποίος θα είναι ενσωματωμένος ως μια εργασία του πραγματικού (host) διαχειριστή πόρων. Ο κώδικας θα υποστηρίζει το SLURM ως εικονικό διαχειριστή πόρων, αλλά είτε το SLURM είτε το TORQUE ως τον πραγματικό διαχειριστή πόρων. Με αυτόν τον τρόπο, το πρόγραμμα θα μπορεί να μιμηθεί ένα πραγματικό HPC Cluster και ως συνέπεια οι ερευνητές/προγραμματιστές υπολογιστικών συστημάτων υψηλής επίδοσης θα μπορούν πιο εύκολα να πειραματιστούν και να βελτιώσουν υπάρχοντες αλγορίθμους χρονοδρομολόγησης αλλά και να τροποποιήσουν οποιοδήποτε υποσύστημα (module) του διαχειριστή πόρων.

Σχετικά Μαθήματα: Συστήματα Παράλληλης Επεξεργασίας

Σχετική Βιβλιογραφία:

1. `https://en.wikipedia.org/wiki/Supercomputer`
2. `https://en.wikipedia.org/wiki/Message_Passing_Interface`
3. `https://en.wikipedia.org/wiki/TORQUE`
4. `https://en.wikipedia.org/wiki/Slurm_Workload_Manager`
5. `https://github.com/nikosT/MPI-Job-Scheduler`

Επικοινωνία: Νικόλαος Τριανταφύλλης, ntriantafyl@cslab.ece.ntua.gr
Γεώργιος Γκούμας, goumas@cslab.ece.ntua.gr, 210-772-2402

3.3.2 Μελέτη και αξιολόγηση αλγορίθμων χρονοδρομολόγησης σε προσομοιωτή υπολογιστικού συστήματος υψηλής επίδοσης

Ο διαχειριστής πόρων (resource manager) αυτός αναλαμβάνει να εκτελέσει τις διάφορες εργασίες, διαμοιράζοντας τους υπολογιστικούς πόρους κατάλληλα. Ο job scheduler (υποτμήμα του resource manager) επικοινωνεί με τον resource manager προκειμένου να πληροφορηθεί για τις ουρές (queues), τα φορτία των υπολογιστικών κόμβων (nodes) και την διαθεσιμότητα των πόρων, ώστε να πάρει αποφάσεις για τη χρονοδρομολόγηση εργασιών. Υπάρχουν διάφοροι αλγόριθμοι στην κατηγορία των space-sharing αλγορίθμων χρονοδρομολόγησης, όπως ο First Come First Served (FCFS), ο Shortest Job

First (SJF), ο Longest Job First (LJF), ο Backfilling κ.α. Οι αλγόριθμοι αυτοί -στα υπολογιστικά σύστημα υψηλής επίδοσης- δεσμεύουν, συνήθως, πόρους στο επίπεδο του κόμβου. Εξ' ορισμού η επιλογή ανάθεσης πόρων στον επίπεδο κόμβου (δεδομένου ότι οι κόμβοι περιλαμβάνουν ολοένα περισσότερα και μεγαλύτερα εξαρτήματα υλικού πια) είναι αντιπαραγωγική όσον αφορά τη ρυθμαπόδοση (throughput) του συστήματος, την κατανάλωση ενέργειας και κόστους. Μελέτες δείχνουν πως το co-scheduling, δηλαδή η ανάθεση πόρων στο επίπεδο του πυρήνα (κι άρα η εκτέλεση διαφορετικών εφαρμογών ταυτόχρονα στον ίδιο κόμβο), οδηγεί σε αποτελεσματικότερη χρήση των υπολογιστικών πόρων. Από την άλλη πλευρά, η επιλογή των εφαρμογών που θα εκτελεστούν ταυτόχρονα λαμβάνει σημαντικό ρόλο για την επίδοση που θα πετύχουν λόγω των race conditions που θα αναπτυχθούν ανάλογα με τους πόρους που ζητά η εκάστοτε εφαρμογή. Σκοπό της διπλωματικής αποτελεί η πειραματική μελέτη και αξιολόγηση αλγορίθμων χρονοδρομολόγησης με χρήση υπαρχόντων benchmarks στο επιλεγμένο περιβάλλον προσομοίωσης. Συγκεκριμένα, θα μελετηθεί (i) η κλιμακωσιμότητα των benchmarks σ' ένα cluster, (ii) η επίδοση για διαφορετικού είδους αναθέσεις πόρων, (iii) τα race conditions που αναπτύσσονται σε διάφορα είδη εφαρμογών (π.χ. memory bounded, compute bounded), (iv) η αξιολόγηση και σύγκριση αλγορίθμων χρονοδρομολόγησης για τα συγκεκριμένα benchmarks με ή χωρίς τεχνικές co-scheduling.

Σχετικά Μαθήματα: Συστήματα Παράλληλης Επεξεργασίας

Σχετική Βιβλιογραφία:

1. `https://en.wikipedia.org/wiki/TORQUE`
2. `https://en.wikipedia.org/wiki/Slurm_Workload_Manager`
3. `https://en.wikipedia.org/wiki/Supercomputer`
4. `http://www.cslab.ntua.gr/~ntriantafyl/stuff/HPC_Job_Scheduling.pdf`

Επικοινωνία: Νικόλαος Τριανταφύλλης, ntriantafyl@cslab.ece.ntua.gr
Γεώργιος Γκούμας, goumas@cslab.ece.ntua.gr, 210-772-2402

II. Αρχιτεκτονική

4 Αρχιτεκτονική Υπολογιστών και Μηχανική Μάθηση

Αλγόριθμοι μηχανικής μάθησης ταξινόμησης (classification) και πρόβλεψης (prediction) εφαρμόζονται πλέον κατά κανόνα σε τομείς όπως η όραση υπολογιστών, η επεξεργασία φυσικής γλώσσας κ.α, πετυχαίνοντας εντυπωσιακά αποτελέσματα. Και ενώ συχνά σχεδιάζεται εξειδικευμένο hardware για την επιτάχυνση τους, λίγες είναι προς το παρόν οι περιπτώσεις εφαρμογής/χρήσης τους για τη βελτίωση της ίδιας της επίδοσης ενός υπολογιστικού συστήματος.

Οι παρακάτω εργασίες εστιάζουν τόσο στην χρήση τεχνικών μηχανικής μάθησης στην αρχιτεκτονική υπολογιστών όσο και στην αποδοτική υλοποίηση των ίδιων των αλγορίθμων.

4.1 Εφαρμογή αλγορίθμων μηχανικής μάθησης στην αρχιτεκτονική υπολογιστών

Οι σύγχρονες αρχιτεκτονικές συχνά εμπλέκουν ευριστικές μεθόδους, μεθόδους πρόβλεψης/υποθετικής εκτέλεσης για τη μεγιστοποίηση της επίδοσης ενός συστήματος. Παράδειγμα μπορεί να θεωρηθεί η χρήση προανάκλησης (prefetching), που χρησιμοποιείται για την αντιμετώπιση ενός σημαντικού σημείου συμφόρησης (bottleneck) επίδοσης των σύγχρονων αρχιτεκτονικών, του κόστους προσπέλασης της κύριας μνήμης. Σκοπός της συγκεκριμένης διπλωματικής είναι η διερεύνηση της δυνατότητας εφαρμογής αλγορίθμων μηχανικής μάθησης για τη βελτιστοποίηση της επίδοσης με στόχο βελτιστοποιήσεις στη χρήση των κρυφών μνημών (caches, prefetching), στο μηχανισμό εικονικής μνήμης (TLBs), στο μηχανισμό πρόβλεψης διακλαδώσεων (branch prediction) κ.α.

Στόχος είναι αρχικά να χρησιμοποιηθεί λογισμικό μηχανικής μάθησης (π.χ pytorch) με πραγματικά δεδομένα από σύγχρονα μηχανήματα για τη μελέτη διαφορετικών μοντέλων (π.χ LSTMs). Στη συνέχεια, ανάλογα με τα συμπεράσματα του πρώτου βήματος, θα επιχειρήσουμε να αξιολογήσουμε τη δυνατότητα εφαρμογής τους σε επίπεδο μικροαρχιτεκτονικής λαμβάνοντας υπόψη την πολυπλοκότητα, το χρόνο απόκρισης και την κατανάλωση χώρου και ενέργειας.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

1. Learning Memory Access Patterns
2. Dynamic Branch Prediction with Perceptrons
3. BranchNet: A Convolutional Neural Network to Predict Hard-to-Predict Branches
4. Virtual Address Translation via Learned Page Table Indexes
5. SmartChoices: Hybridizing Programming and Machine Learning
6. Applying Deep Learning to the Cache Replacement Problem

Επικοινωνία: Χλόη Αλβέρτη, xalverti@cslab.ece.ntua.gr, 210-772-2279
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133
Κωστής Νίκας, knikas@cslab.ece.ntua.gr, 210-772-4159

4.2 Απεικόνιση αλγορίθμων Βαθιάς Μάθησης στην πλατφόρμα Graphcore IPU

Τα IPUs της Graphcore (www.graphcore.ai) υπόσχονται επιδόσεις σημαντικά καλύτερες από GPUs στην επεξεργασία εφαρμογών βαθιάς μάθησης (που μπορούν να εκφραστούν ως γράφοι) με χρήση παραλληλισμού και πολλαπλών τοπικών μνημών. Τα IPUs της GraphCore προγραμματίζονται με το Poplar toolflow και υποστηρίζουν την απεικόνιση CNN (π.χ. με PyTorch). Ενδεικτικά, η εργασία αυτή θα ασχοληθεί με τα παρακάτω:

- Απεικόνιση ενός αλγορίθμου Histogram-of-Oriented-Gradients (HOG) με χρήση του προγραμματιστικού περιβάλλοντος Poplar και σύγκριση με multi-core CPUs και GPUs ως προς επιδόσεις, κλιμακωσιμότητα, κλπ.
- Αξιοποίηση πολλαπλών IPU για την επίλυση μεγαλύτερων προβλημάτων
- Αντικατάσταση του αλγορίθμου HOG με full-featured CNN και σύγκριση σε IPUs.

Σχετική Βιβλιογραφία:

1. Ενδεικτική εφαρμογή για απεικόνιση:`https://arxiv.org/abs/2006.00816`
2. Πληροφορίες για την πλατφόρμα:`https://www.graphcore.ai/hubfs/Lead%20gen%20assets/DSS8440%20IPU%20Server%20White%20Paper_2020.pdf`

Επικοινωνία:

Διονύσιος Πνευματικάτος, pnevmati@cslab.ece.ntua.gr, 6944763171,
Μηλιάδης Παναγιώτης, pmiliad@cslab.ece.ntua.gr

5 Κύρια Μνήμη Συστήματος

5.1 Αρχιτεκτονικές με ανομοιόμορφη πρόσβαση μνήμης (Non Uniform Memory Access - NUMA)

Η κύρια μνήμη στα σύγχρονα πολυεπεξεργαστικά υπολογιστικά συστήματα είναι συνήθως φυσικά κατανεμημένη σε πολλαπλούς κόμβους αλλα παραμένει λογικά ενιαία (ένας συνεχής χώρος φυσικών διευθύνσεων). Κάθε κόμβος αποτελείται από μια συστάδα επεξεργαστών συνδεδεμενων με μια τοπική μνήμη μέσω ένος κοινού διαύλου. Οι επεξεργαστές όλων των κόμβων μπορούν να προσπελάσουν τόσο την τοπική τους μνήμη όσο και όλες τις απομακρυσμένες (που ανήκουν στους υπόλοιπους κόμβους). Επομένως ο χρόνος πρόσβασης της κύριας μνήμης δεν είναι σταθερός και οι αρχιτεκτονικές αυτές ονομάζονται Ανομοιόμορφης Πρόσβασης Μνήμης ή αλλιώς Non Uniform Memory Access (NUMA).

5.1.1 Ανάλυση της επίδοσης εφαρμογών σε NUMA αρχιτεκτονικές και υλοποίηση αποτελεσματικής κατανομής και χρονοδρομολόγησης

Σε μία NUMA αρχιτεκτονική, ένας επεξεργαστής έχει γρηγορότερη πρόσβαση σε μία τοπική μνήμη από ότι σε μία απομακρυσμένη, η οποία όμως είναι τοπική για κάποιον άλλον επεξεργαστή. Στόχος της παρούσας διπλωματικής είναι η μελέτη και η ανάλυση της επίδοσης που προκαλεί η NUMA αρχιτεκτονική κατά την εκτέλεση σύγχρονων εφαρμογών, καθώς επίσης και η αναγνώριση προτύπων και συμπεριφορών και η κατηγοριοποίηση των εφαρμογών σε ευαίσθητες και μη-ευαίσθητες, ως

προς την συμπεριφορά τους λόγω της NUMA αρχιτεκτονικής. Επιπλέον, θα υλοποιηθεί ένας διαχειριστής πόρων και πολιτικές αποτελεσματικής χρονοδρομολόγησης πολλαπλών εφαρμογών σε συνέχεια προηγούμενης διπλωματικής εργασίας, τόσο μέσω της κατάλληλης κατανομής των κόμβων μνήμης (memory node allocation) και υπολογιστικών κόμβων (compute node allocation), όσο και μέσω κατάλληλης υποστήριξης σε επίπεδο λειτουργικου συστήματος.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών, Εργαστήριο Λειτουργικών Συστημάτων

Επικοινωνία: Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133
Νικέλα Παπαδοπούλου, nikela@cslab.ece.ntua.gr, 210-772-2495
Κωστής Νίκας, knikas@cslab.ece.ntua.gr, 210-772-4159

5.1.2 Τοποθέτηση εφαρμογών σε NUMA αρχιτεκτονικές λαμβάνοντας υπόψη τη συμφόρηση των διαύλων μνήμης

Σε τέτοια συστήματα η επίδοση των εφαρμογών αξαρτάται πολύ από την τοποθέτηση (placement) του αποτυπώματος μνήμης τους στους NUMA κόμβους. Το λειτουργικό σύστημα είναι υπεύθυνο για την δέσμευση φυσικής μνήμης μιας εφαρμογής και η αρχική πολιτική του είναι να χρησιμοποιεί την τοπική μνήμη των επεξεργαστών στους οποίους εκτελείται η εφαρμογή. Ωστόσο στα συστήματα μεγάλης κλίμακας συνήθως υπάρχει συνεκτέλεση (consolidation) πολλαπλών εφαρμογών ή εικονικών μηχανών (VM) για τη μέγιστη αξιοποίηση των πόρων του μηχανήματος. Σε τέτοιες συνθήκες παρατηρείται ότι η συνεκτέλεση πολλαπλών εφαρμογών στους επεξεργαστές ενός κόμβου δημιουργεί συμφόρηση στον κοινό δίαυλο προς την τοπική μνήμη και συνεπώς μεγάλες καθυστερήσεις. Σε τέτοιες περιπτώσεις η δέσμευση μνήμης σε απομακρυσμένο κόμβο μπορεί να οδηγήσει σε καλύτερες επιδόσεις λόγω μειωμένων καθυστερήσεων στο δίαυλο. Αυτό εξαρτάται τόσο από τις ανάγκες και τα χαρακτηριστικά των εφαρμογών όσο και από τις συνθήκες σε επίπεδο συστήματος.

Σκοπός της παρούσας διπλωματικής είναι ο σχεδιασμός και η υλοποίηση πολιτικών τοποθέτησης σε NUMA αρχιτεκτονικές που λαμβάνουν υπόψη εκτός από το χρόνο πρόσβασης της μνήμης και το βαθμό συμφόρησης των διαύλων.

Σχετικά Μαθήματα: Εργαστήριο Λειτουργικών Συστημάτων, Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Ενδεικτική Βιβλιογραφία:

1. Merlin: Application- and Platform-aware Resource Allocation in Consolidated Server Systems
2. Congestion-Aware Memory Management on NUMA Platforms: A VMware ESXi case study

Επικοινωνία: Χλόη Αλβέρτη, xalverti@cslab.ece.ntua.gr, 210-772-2279
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133

5.2 Αναλυση επίδοσης εφαρμογών και βελτιστοποίηση σε συστήματα με Non-Volatile Memories

Οι μνήμες είναι ενα από τα πιο ακριβά κομμάτια των υπολογιστικών συστημάτων, και επηρρεάζουν σημαντικά την ταχύτητα εκτέλεση σημαντικών εφαρμογών που κυριαρχούν στο υπολογιστικό νέφος (πχ. key-value stores, in-memory databases, graph analytics). Αλλά καθώς τα σύνολα δεδομένων των εφαρμογών συνεχίζουν να αυξάνονται, οι απαιτήσεις σε χωρητικότητα μνήμης αυξάνονται ακόμα περισσότερο. Ταυτόχρονα, η τεχνολογία της παραδοσιακής κύριας μνήμης (DRAM) έχει φτάσει σε ένα όριο κλιμάκωσης που περιορίζει την πυκνότητά της. Πρόσφατα έχουν κάνει την εμπορική εμφάνιση

τους αναδυόμενες μη πτητικές τεχνολογίες μνήμης (NVM) (δηλ. PCM, STTRAM κ.λπ.) οι οποίες φαίνονται να έχουν καλύτερη κλιμάκωση πυκνότητας με χαμηλότερο κόστος. Οι NVM μνήμες προσφέρουν byte-addressable πρόσβαση σε μόνιμη δεδομένα (διατηρημένα σε αστοχίες ισχύος), με καθυστέρηση πρόσβασης κοντά σε αυτή της DRAM. Επιπλέον, οι NVM μνήμες μπορούν να χρησιμοποιηθούν ως μη πτητική κύρια μνήμη (NVMM - Intel Optane DIMM), επιτρέποντας άμεση πρόσβαση μέσω εντολών μνήμης CPU (loads/stores). Τέτοιες μνήμες μπορούν να προσφέρουν χωρητικότητες της τάξης των TBs, και παρέχουν μια μοναδική ευκαιρία για την επαναπροσδιορισμό της αυστηρής διάκρισης μεταξύ μνήμης και αποθήκευσης στη στοίβα υπολογιστών. Στόχος της παρούσας διπλωματικής είναι η εξοικείωση με τα προγραμματιστικά περιβάλλοντα για NVM μνήμες, η ανάλυση επίδοσης εφαρμογών σε αυτά τα συστήματα, και η βελτιστοποίηση εκτέλεσης τους.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών, Εργαστήριο Λειτουργικών Συστημάτων

Σχετική Βιβλιογραφία:

1. NVDIMM
2. Basic Performance Measurements of the Intel Optane DC Persistent Memory Module
3. System Evaluation of the Intel Optane Byte-addressable NVM

Επικοινωνία: Χλόη Αλβέρτη, xalverti@cslab.ece.ntua.gr, 210-772-2279
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133

5.3 Επεξεργασία στη Μνήμη

Οι πρόσφατες εξελίξεις στην αρχιτεκτονική 3D-stack τεχνολογιών μνήμης (για παράδειγμα High-Bandwidth Memory, HBM) έχουν ανανεώσει το ενδιαφέρον για Επεξεργασία Κοντά στη Μνήμη, ή αλλιώς Near-Data-Processing (NDP). Οι NDP αρχιτεκτονικές έχουν σχεδιαστεί με στόχο να μειώσουν την κίνηση δεδομένων (data movement) μεταξύ του επεξεργαστή και της κύριας μνήμης, τοποθετώντας πυρήνες χαμηλής κατανάλωσης ενέργειας και μικρού κόστους κοντά στην κύρια μνήμη. Πρόσφατες εργασίες [1-3, 7-9, 11, 12] δείχνουν τα οφέλη των NDP αρχιτεκτονικών για παράλληλες εφαρμογές όπως graph-processing, neural networks, bioinformatics και databases. Ο στόχος αυτής της έρευνας είναι να μελετήσει εφαρμογές που επωφελούνται από NDP αρχιτεκτονικές και να αναπτύξει μηχανισμούς και προσομοιωτές για το σκοπό αυτό.

5.3.1 Υλοποίηση Προσομοιωτή (Simulator) για Near-Rank Processing χρησιμοποιώντας τους ZSim [5] και Ramulator [6].

5.3.2 Βελτιστοποίηση του Υπολογιστικού Πυρήνα Πολλαπλασιασμού Αραιού Πινακα με Διάνυσμα (SpMV) μέσω Near-Data Processing.

5.3.3 Σχεδίαση μιας NDP Αρχιτεκτονικής για την Επιτυχάνυση Εφαρμογών για Personalized Recommendation [10-12].

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών, Συστήματα Παράλληλης Επεξεργασίας

Σχετική Βιβλιογραφία:

1. Saugata Ghose et al., ”Processing-in-memory: A workload-driven perspective”, in IBM Journal of Research and Development 2019.

2. Benjamin Y. Cho et al., "Near Data Acceleration with Concurrent Host Access", in ISCA 2020.

3. Elliot Lockerman et al., "Livia: Data-Centric Computing Throughout the Memory Hierarchy", in ASPLOS 2020.

4. Po-An Tsai et al., "Adaptive Scheduling for Systems with Asymmetric Memory Hierarchies", in MICRO 2018.

5. Daniel Sanchez et al., "ZSim: Fast and Accurate Microarchitectural Simulation of Thousand-Core Systems", in ISCA 2013.

6. Yoongu Kim et al., "Ramulator: A Fast and Extensible DRAM Simulator", in IEEE CAL 2016.

7. Junwhan Ahn et al., "A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing", in ISCA 2015.

8. Youwei Zhuo et al., "GraphQ: Scalable PIM-Based Graph Processing", in MICRO 2019.

9. Lifeng Nai et al., "GraphPIM: Enabling Instruction-Level PIM Offloading in Graph Computing Frameworks", in HPCA 2017.

10. Udit Gupta et al., "The Architectural Implications of Facebook's DNN-based Personalized Recommendation", in HPCA 2020.

11. Youngeun Kwon et al., "TensorDIMM: A Practical Near-Memory Processing Architecture for Embeddings and Tensor Operations in Deep Learning", in MICRO 2019.

12. Liu Ke et al., "RecNMP: Accelerating Personalized Recommendation with Near-Memory Processing", in ISCA 2020.

Επικοινωνία: Χριστίνα Γιαννούλα, cgiannoula@cslab.ece.ntua.gr

6 Μελέτη οργανώσεων πινάκων σελίδων (page tables)

Το λειτουργικό σύστημα είναι υπεύθυνο για την δέσμευση φυσικής μνήμης και για την αποθήκευση και συντήρηση των αντιστοιχίσεων των εικονικών διευθύνσεων των εφαρμογών σε φυσικές διευθύνσεις. Τη πληροφορία αυτή τη συντηρεί το λειτουργικό σε ειδικές δομές ανά εφαρμογή, τους πίνακες σελίδων (page tables). Οι δομές αυτές είναι παραδοσιακά δενδρικές (radix trees) τις οποίες προσπελαύνει ειδικός μηχανισμός υλικού (hardware page tables walkers) για να βρει τις φυσικές μεταφράσεις εικονικών διευθύνσεων των εφαρμογών κατά την εκτέλεση τους. Το βάθος του δέντρου εξαρτάταται από το μέγεθος του χώρου διευθύνσεων (address space) και ως σήμερα τα δέντρα ήταν τεσσάρων επιπέδων. Με την ολοένα αυξανόμενη χωρητικότητα των κύριων μνημών, το βάθος των δέντρων επίκειται να μεγαλώσει και να γίνει 5 επιπέδων. Το βάθος επηρρεάζει το χρόνο που απαιτείται για την ανάκτηση μιας μετάφρασης και συνεπώς την επίδοση των εφαρμογών. Η επίδρασή πολλαπλασιάζεται όταν οι εφαρμογές εκτελούνται εντός εικονικών μηχανών (virtual machines) όπου χρειάζεται η εμφολευμένη προσπέλαση (nested paging) των πινάκων σελίδων τόσο του guest οσο και του host machine. Σύγχρονες ερευνητικές δουλειές προτείνουν εναλλακτικές οργανώσεις των πινάκων σελίδων, π.χ πίνακες κατακερματισμού αντί για δέντρα, για τη μείωση του χρόνο του page table walk. Σκοπός της παρούσας διπλωματικής είναι να προσομοιώσει και να αξιολογήσει διαφορετικές οργανώσεις page tables.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών, Λειτουργικά Συστήματα

Σχετική Βιβλιογραφία:

1. Hash don't cache the page table

2. Elastic Cuckoo Page Tables: Rethinking Virtual Memory Translation for Parallelism

Επικοινωνία: Χλόη Αλβέρτη, xalverti@cslab.ece.ntua.gr, 210-772-2279
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133

7 Επεκτάσεις Αρχιτεκτονικής

7.1 Αποδοτικός συγχρονισμός σε πολυεπεξεργαστικά συστήματα

Ένα σχήμα συγχρονισμού, π.χ. ένα κλειδώμα (lock), θεωρείται αποδοτικό όταν: (i) προσφέρει υψηλή κλιμακωσιμότητα σε highly-contended σενάρια, και (ii) δεν μειώνει την απόδοση του συστήματος σε low-contention σενάρια. Από τη μία πλευρά, τα locks που υπολοποιούνται μέσω hardware cache coherence αποτελούν συχνά το κύριο bottleneck σε highly-contended σενάρια. Σε μια πρόσφατη μελέτη [1] αναφέρεται ότι τα locks που υπολοποιούνται μέσω hardware message-passing μειώνουν τη συμφόρηση (contention) και προσφέρουν καλύτερη απόδοση όταν ένας μεγάλος αριθμός νημάτων συμετέχει στη διαδικασία συγχρονισμού. Από την άλλη πλευρά, αρκετά συστήματα χρησιμοποιούν fine-grain locking στρατηγικές, όπου ένας μεγάλος αριθμός locks χρησιμοποιείται στην εφαρμογή το καθένα από αυτά προστατεύει ένα μικρό κομμάτι κοινών δεδομένων, με στόχο να μειώθεί η συμφόρηση. Ως αποτέλεσμα, σε low-contention σενάρια τα locks που υπολοποιούνται μέσω hardware cache coherence μπορούν να προσφέρουν υψηλή απόδοση. Ο στόχος αυτής της διπλωματικής είναι να σχεδιάσει ένα hardware μηχανισμό συγχρονισμού που επιτρέπει την υψηλή απόδοση του συστήματος και στα δύο αυτά σενάρια προσαρμόζοντας δυναμικά το lock στο επίπεδο συμφόρησης (contention level). Παρόλο που δύο πρόσφατες εργασίες [2,3] έχουν ήδη προτείνει δυναμικά σχήματα συγχρονισμού σε software level, δεν υπάρχει προηγούμενη ερευνητική προσπάθεια σχετικά τη μελέτη *δυναμικών* προσεγγίσεων συγχρονισμού σε hardware level.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών, Συστήματα Παράλληλης Επεξεργασίας

Σχετική Βιβλιογραφία:

1. Tudor David et al., "Everything You Always Wanted to Know About Synchronization but Were Afraid to Ask", in SOSP 2013.

2. Jelena Antic et al., "Locking Made Easy", in ACM Middleware 2016.

3. Foteini Strati et al., "An adaptive concurrent priority queue for NUMA architectures", in ACM Computing Frontiers 2019.

4. Ching-Kai Liang et al., "MiSAR: Minimalistic Synchronization Accelerator with Resource Overflow Management", in ISCA 2015.

5. Enrique Vallejo et al., "Architectural Support for Fair Reader-Writer Locking", in MICRO 2010.

6. Anurag Mukkara et al., "PHI: Architectural Support for Synchronization-and Bandwidth-Efficient Commutative Scatter Updates", in MICRO 2019.

7. Guowei Zhang et al., "Exploiting semantic commutativity in hardware speculation", in MICRO 2016.

8. Guowei Zhang et al., ”Exploiting Commutativity to Reduce the Cost of Updates to Shared Data in Cache-Coherent Systems”, in MICRO 2015.

Επικοινωνία: Χριστίνα Γιαννούλα, cgiannoula@cslab.ece.ntua.gr

7.2 Αρχιτεκτονική υποστήριξη για βελτίωση του χρόνου εκκίνησης και εκτέλεσης containers

Πολλοί πάροχοι υποδομών υπολογιστικού νέφους (cloud computing) παρέχουν την δυνατότητα εκτέλεσης εφαρμογών χρησιμοποιώντας το περιβάλλον των containers. Ωστόσο, σε αυτό το περιβάλλον εκτέλεσης υπάρχουν δύο αιτίες που επηρρεάζουν την απόδοση των εφαρμογών: (α) η αργή αρχικοποίηση (boot time due to cold starts) των containers (για παράδειγμα, serverless functions [2,3,4,5,1]), και (β) η αργή εκτέλεση των κλήσεων συστήματος (system calls) λόγω των επιπλέον ελέγχων που γίνονται [6] (για παράδειγμα, I/O intensive applications).

Σε αυτή τη διπλωματική εργασία, θα αναλύσουμε την εκτέλεση υπάρχοντων περιβάλλοντων εκτέλεσης containers (για παράδειγμα, Docker, gVisor, Firecracker) για να καταλάβουμε καλύτερα τις συνέπειες της εκτέλεσης εφρμογών σε docker, και θα αναγωνρίσουμε λειτουργίες που έχουν την δυνατότητα να επιταχυνθούν μέσω ειδικής υποστήριξης στο υλικό. Πιο συγκεκριμένα, θα επικεντρωθούμε σε εκείνα τα επίπεδα εικονικοποίησησς που επιτρέπουν την απομόνωση εφαρμογών (e.g. SecComp [6], Namespaces). Αυτά τα επίπεδα εικονικοποίησης χρησιμοποιουν διάφορους πίνακες που χρειάζονται να δημιουργούνται, να ενημερώνονται, και να χρησιμοποιούνται από τον πυρήνα του λειτουργικού συστήματος, για να παρέχεται απομόνωση των εφσρμογών. Μετά την ανάλυση, θα επικεντρωθούμε στην ανάπτυξη ειδικής υποστήριξης σε επίπεδο υλικού και αρχιτεκτονικής με σκοπό να μειώσουμε τον χρόνο αρχικοποίησης των containers και το κόστος εκτέλεσης των system calls.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών, Εργαστήριο Λειτουργικών Συστημάτων

Σχετική Βιβλιογραφία:

1. Architectural Implications of Function-as-a-Service Computing, MICRO 2019
2. Catalyzer: Sub-millisecond Startup for Serverless Computing with Initialization-less Booting, ASPLOS 2020
3. SOCK: Rapid Task Provisioning with Serverless-Optimized Containers
4. SAND: Towards High-Performance Serverless Computing
5. Σerverless in the Wild: Characterizing and Optimizing the Serverless Workload at a Large Cloud Provider, ATC 2020
6. Draco: Architectural and Operating System Support for System Call, Security MICRO 2020
7. BabelFish: Fusing Address Translations for Containers, ISCA 2020

Επικοινωνία: Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133
Κωστής Νίκας, knikas@cslab.ece.ntua.gr, 210-772-4159

8 Αρχιτεκτονική RISC-V

Η RISC-V αρχιτεκτονική είναι μια ανοικτή και επεκτάσιμη αρχιτεκτονική συνόλου εντολών που ξεκίνησε να αναπτύσσεται στο Πανεπιστήμιο του Berkeley το 2010 και από το 2016 λαμβάνει διεθνή προσοχή τόσο από τον ακαδημαϊκό χώρο όσο και από τον χώρο της βιομηχανίας, με κατάλληλη υποστήριξη σε όλα τα επίπεδα της υπολογιστικής στοίβας (υλικό, λειτουργικό σύστημα, βιβλιοθήκες, μεταγλωττιστές, κτλ). Ως ανοικτή και επεκτάσιμη αρχιτεκτονική προσφέρεται για την έρευνα σε λειτουργικές επεκτάσεις, ενώ πολλές υλοποιήσεις ανοικτού κώδικα είναι άμεσα διαθέσιμες, άλλες απλούστερες με γραμμική in-order pipeline και άλλες μεγαλύτερων επιδόσεων με πυρήνα εκτέλεσης εντολών εκτός σειράς (out-of-order).

8.1 Μελέτη περιβάλλοντος ανάπτυξης και υλοποίηση επιταχυντών σε RISC-V αρχιτεκτονική

Στόχος της παρούσας διπλωματικής εργασίας είναι η μελέτη του περιβάλλοντος ανάπτυξης υλικού του Rocket Chip Generator που αναπτύσσεται από το Πανεπιστήμιο του Berkeley και θα εστιάσουμε στην ανάπτυξη επιταχυντών σε RISC-V αρχιτεκτονικές χρησιμοποιώντας το framework του Rocket Custom Coprocessor (RoCC).

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

1. A Hardware Accelerator for Tracing Garbage Collection
2. `https://en.wikipedia.org/wiki/RISC-V`
3. `https://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-17.pdf`

Επικοινωνία: Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133
Κωστής Νίκας, knikas@cslab.ece.ntua.gr, 210-772-4159

8.2 Σχεδιασμός, υλοποίηση, και αξιολόγηση επίδοσης προηγμένων σχημάτων κρυφής μνήμης του συστήματος διαχείρισης εικονικής μνήμης του Rocket Chip Generator

Ο Rocket Chip Generator [2] είναι ένας ανοιχτού κώδικα System-on-Chip (SoC) Generator που παράγει παραμετροποιήσιμα RISC-V SoCs. Είναι υλοποιημένος στην γλώσσα Chisel [3], η οποία καθιστά εύκολη την περιγραφή πολύπλοκων και παραμετροποιήσιμων γεννητριών για επεξεργαστικούς πυρήνες, κρυφές μνήμες και δικτύων διασύνδεσης εντός του SoC.

Στα θέματα των υποενοτήτων που ακολουθούν θα ασχοληθούμε με το σύστημα διαχείρισης εικονικής μνήμης του Rocket Chip Generator. Το σύστημα διαχείρισης εικονικής μνήμης (Memory Management Unit - MMU) παίζει διττό ρόλο στα σύγχρονα υπολογιστικά συστήματα, (i) διασφαλίζει την μνήμη του συστήματος μέσω της απομόνωσης των διεργασιών και (ii) ενισχύει την παραγωγικότητα του προγραμματιστή. Τα παραπάνω επιτυγχάνονται με την χρήση εικονικών διευθύνσεων πρόσβασης στην μνήμη αντί για φυσικών, και με τον διαχωρισμό της μνήμης σε σελίδες (συνήθως των 4KB). Με την χρήση εικονικών διευθύνσεων κάθε εφαρμογή ”νομίζει” ότι δουλεύει πάνω σε συνεχόμενες σελίδες μνήμης, ενώ στην πραγματικότητα οι σελίδες μπορεί να είναι διάσπαρτες στην φυσική μνήμη. Το σύστημα διαχείρισης εικονικής μνήμης γνωστών αρχιτεκτονικών (x86, ARM, RISC-V, κλπ) αποτελείται από την μονάδα του Page Table Walker ο οποίος είναι υπεύθυνος για την μετάφραση των διευθύνσεων

από εικονικές σε φυσικές, καθώς και από τον Translation Lookaside Buffer (TLB), μία κρυφή μνήμη, στην οποία κρατούνται οι πρόσφατες εικονικές-σε-φυσικές μεταφράσεις διευθύνσεων. Στα παρακάτω θέματα θα χρησιμοποιήσουμε εργαλεία ανοικτού κώδικα για την ανάπτυξη υλικού, επιβεβαίωσης ορθής λειτουργίας του, καθώς και FPGAs (Field Programmable Gate Arrays) για την μελέτη επίδοσης του υλικού που σχεδιάσαμε.

Σχετική Βιβλιογραφία:

[1] RISC-V Technical Specifications,
`https://riscv.org/technical/specifications/`

[2] Rocket Chip Generator,
`https://github.com/chipsalliance/rocket-chip/`

[3] The Chisel Language,
`https://www.chisel-lang.org`

8.2.1 Advanced MMU Caching Techniques for the Rocket Chip Generator

Σε πολλά σύγχρονα benchmarks/workloads, η μετάφραση των εικονικών διευθύνσεων μπορεί να επιβαρύνει αισθητά την επίδοση και την ενεργειακή απόδοση του υπολογιστικού συστήματος, λόγω των αστοχιών TLB. Αναλόγως της αρχιτεκτονικής του πίνακα σελίδων, απαιτούνται 3-4 προσβάσεις στην μνήμη για την μετάφραση της εικονικής-σε-φυσική διεύθυνση. Συγκεκριμένα, σε περιβάλλοντα εικονοποίησης ο αριθμός προσβάσεων στην μνήμη μπορεί να φτάσει τις 24. Μία πρόταση στο παραπάνω πρόβλημα είναι η χρήση μεγαλύτερης χωρητικότητας -ή μεγαλύτερης συσχετιστικότητας- TLB. Όμως, το TLB βρίσκεται στο critical path του επεξεργαστή με αποτέλεσμα να επηρεάζει τον χρονισμό του: δημιουργείται ένα trade-off μεταξύ μεγέθους TLB (χαμηλότερος χρονισμός CPU) και αστοχιών TLB (χαμηλότερη επίδοση). Στην σύγχρονη βιβλιογραφία προτείνονται διάφορες αρχιτεκτονικές για caching εικονικής μνήμης, όπως τα Coalesced TLBs [1], Clustered TLBs [2], Hybrid TLB Coalescing [3], Direct Segments [4], Redundant Memory Mappings [5] και άλλα. Η παρούσα διπλωματική στοχεύει σε συνέχεια προηγούμενης διπλωματικής εργασίας στην υλοποίηση κάποιου -ή κάποιων- από τα παραπάνω σχήματα στον Rocket Chip Generator (RCG), και να εξεταστεί η επίδοση τους έναντι του vanilla TLB του RCG.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

[1] CoLT: Coalesced Large-Reach TLBs,
`https://ieeexplore.ieee.org/document/6493625`

[2] Increasing TLB Reach by Exploiting Clustering in Page Translations,
`http://www.cs.yale.edu/homes/abhishek/binhpham-hpca14.pdf`

[3] Hybrid TLB Coalescing: Improving TLB Translation Coverage under Diverse Fragmented Memory Allocations,
`https://iamchanghyunpark.github.io/papers/htc-isca2017.pdf`

[4] Efficient Virtual Memory for Big Memory Servers,
`https://research.cs.wisc.edu/multifacet/papers/isca13_direct_segment.pdf`

[5] Redundant Memory Mappings for Fast Access to Large Memories,
`http://www.cslab.ece.ntua.gr/~vkarakos/papers/isca15_redundant_memory_mappings.pdf`

8.2.2 Enabling TLB Prefetching for the Rocket Chip Generator

Μία άλλη τεχνική για την βελτίωση της επίδοσης του συστήματος εικονικής μνήμης είναι το prefetching [1]. Η τεχνική αυτή βασίζεται στην εικασία ότι φέρνοντας δεδομένα από την κύρια μνήμη στην κρυφή μνήμη πριν χρειαστούν, θα μηδενιστεί η αναμονή σε μελλοντική πρόσβαση στα δεδομένα αυτά. Στην διπλωματική αυτή, θα μελετηθεί η ανάπτυξη prefetcher για το TLB [2, 3, 4] του Rocket Chip Generator, και θα εξεταστεί η επίδοση του χρησιμοποιώντας benchmarking suites όπως το SPEC2006/SPEC2017.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

[1] Cache Prefetching (Wikipedia Page)
`https://en.wikipedia.org/wiki/Cache_prefetching`

[2] Recency-Based TLB Preloading,
`https://courses.cs.washington.edu/courses/cse590g/00au/p117-saulsbury.pdf`

[3] Going the Distance for TLB Prefetching: An Application-driven Study,
`http://www.cse.psu.edu/~axs53/csl/papers/isca02.pdf`

[4] Inter-core cooperative TLB for chip multiprocessors,
`https://dl.acm.org/doi/abs/10.1145/1735970.1736060`

8.2.3 Enabling Configurable Page Table Walk Caches for the Rocket Chip Generator

Η μονάδα που αναλαμβάνει την μετάφραση μιας εικονικής διεύθυνσης σε φυσική ονομάζεται Page Table Walker (PTW) και είναι συνήθως υλοποιημένη στο υλικό για την ταχύτερη μετάφραση των εικονικών διευθύνσεων. Η δομή δεδομένων που χρησιμοποιείται για το mapping των εικονικών σε φυσικές διευθύνσεις ονομάζεται πίνακας σελίδων (page table) [1] και αναλόγως την αρχιτεκτονική, αποτελείται από 3 ή 4 επίπεδα. Στον Rocket Chip Generator (RCG) ο πίνακας σελίδων αποτελείται από 3 επιπεδα για το σχήμα εικονικής μνήμης Sv39 [2]. Σε περίπτωση αστοχίας TLB, η μονάδα PTW πρέπει να κάνει επομένως 3 προσβάσεις στον πίνακα σελίδων ώστε να μεταφράσει την ζητούμενη εικονική διεύθυνση σε φυσική. Για να αποφευχθούν οι κοστοβόρες προσβάσεις στην μνήμη, στον RCG έχει υλοποιηθεί μία μικρή PTW Cache [3] η οποία αποθηκεύει το mapping των πρώτων 2 επιπέδων (το mapping του 3ου επιπέδου αποθηκεύεται στο TLB). Αντικείμενο της διπλωματικής αυτής είναι η παραμετροποίηση της PTW Cache του RCG, η μελέτη υλοποίησης πιο προηγμένων σχημάτων PTW Cache [3], και η εξέταση της επίδοσης τους χρησιμοποιώντας benchmarking suites όπως το SPEC2006/SPEC2017.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

[1] Page Table (Wikipedia Page),
`https://en.wikipedia.org/wiki/Page_table`

[2] RISC-V Page-Based 39-bit Virtual-Memory System, pages 62-64,
`https://riscv.org//wp-content/uploads/2017/05/riscv-privileged-v1.10.pdf`

[3] Translation caching: skip, don't walk (the page table),
`https://dl.acm.org/doi/10.1145/1816038.1815970`

Επικοινωνία: Νίκος Χ. Παπαδόπουλος, ncpapad@cslab.ece.ntua.gr
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133

8.3 Επεκτάσεις του RISC-V για near/in memory accelerators

Η εργασία αυτή αφορά αφενός την δημιουργία ενός μικρού πυρήνα RISC-V ο οποίος θα είναι ο δομικός λίθος για επεξεργασία κοντά στην μνήμη για επεξεργασία μεγάλων δεδομένων (η αρχιτεκτονική αναφοράς είναι το Modrian Data Engine). Έτσι οι βασικές συναρτήσεις θα είναι ερωτήσεις βάσεων δεδομένων στις οποίες τα δεδομένα βρίσκονται στην μνήμη. Το σύστημα μνήμης θα αποτελείται από ένα (η περισσότερα) HMC modules. Συγκεκριμένα, τα βήματα της εργασίας είναι (α) η επιλογή και επέκταση ενός υπάρχοντος πυρήνα RISC-V με εντολές vector (β) ο προγραμματισμός των βασικών λειτουργιών και η επιβεβαίωση ορθής λειτουργίας με προσομοιώσεις, (γ) η ολοκλήρωσή του επεξεργαστή στο περιβάλλον FPGA+HMC της Micron και (δ) η αξιολόγηση του συνολικού συστήματος.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

1. `https://en.wikipedia.org/wiki/RISC-V`
2. `https://en.wikipedia.org/wiki/Vector_processor`
3. `https://en.wikipedia.org/wiki/Hybrid_Memory_Cube`
4. `https://pure.tue.nl/ws/files/100178113/gagan2018dsd.pdf`
5. `https://arxiv.org/pdf/1908.02640.pdf`
6. The Mondrian Data Engine: `https://dl.acm.org/citation.cfm?id=3080233`
7. `https://www.sigarch.org/simd-instructions-considered-harmful/`
8. `https://www.youtube.com/watch?v=GzZ-8bHsD5s`

Επικοινωνία: Διονύσιος Πνευματικάτος, pnevmati@cslab.ece.ntua.gr, 6944763171

9 Αποδοτική απεικόνιση σε FPGAs

Στις μέρες μας, η χρήση των επαναπρογραμματιζόμενων αρχιτεκτονικών (FPGAs) αποτελεί σημαντική εναλλακτική πρόταση, καθώς προσφέρει τη σχεδίαση υλικού για την εκτέλεση συγκεκριμένων εφαρμογών (application-specific), με σκοπό τη βελτιστοποίηση και την επιτάχυνση του χρόνου εκτέλεσης. Αν και μπορούν να απεικονίσουν όμως οποιαδήποτε σχεδίαση υλικού, οι FPGAs έχουν ιδιαιτερότητες και «προτιμήσεις».

9.1 Αποδοτική απεικόνιση επεξεργαστών RISC-V σε FPGA

Η εργασία αυτή αφορά αφενός την συγκριτική μελέτη βασικών υπαρχόντων υλοποιήσεων RISC-V ως προς το κόστος και τις επιδόσεις τους όταν υλοποιούνται με διαφορετικές FPGA, και αφετέρου την παραμετροποίηση των εσωτερικών δομών (η την αντικατάστασή τους με άλλες ισοδύναμες) ώστε η συνολική σχεδίαση να είναι περισσότεραο «φιλική» προς τις FPGA. Η εργασία συνδυάζει προσομοιώσεις για την μέτρηση επιδόσεων σε αρχιτεκτονικό επίπεδο και απεικόνιση των αρχιτεκτονικών σε FPGA με εργαλεία CAD.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

1. https://en.wikipedia.org/wiki/RISC-V
2. https://github.com/pulp-platform/riscv
3. https://tspace.library.utoronto.ca/handle/1807/80713
4. https://github.com/riscv-boom/riscv-boom

Επικοινωνία: Διονύσιος Πνευματικάτος, pnevmati@cslab.ece.ntua.gr, 6944763171

9.2 Υλοποίηση υβριδικού επιταχυντή Convolutional-Recurrent Neural Network (CRNN) με τη χρήση του προγραμματιστικού μοντέλου HLS/OpenCL για FPGAs.

Τα τελευταία χρόνια, η χρήση των Convolutional Neural Networks (CNN) και των Recurrent Neural Networks (RNN) έχουν επιδείξει μεγάλη επιτυχία σε διάφορα σενάρια βαθιάς μηχανικής μάθησης. Οι απαιτήσεις και η πολυπλοκότητα των νέων εφαρμογών μηχανικής μάθησης, έχουν οδηγήσει στην ανάπτυξη υβριδικών λύσεων (CRNN) ή στη χρήση πολλαπλών μοντέλων(LSTM, YoloV3, VGG16) προκειμένου να εκμεταλλευτούν τα χαρακτηριστικά και των δύο τύπων νευρωνικών δικτύων. Διάφοροι επιταχυντές έχουν αναπτυχθεί σε FPGA για κάθε έναν από τους δύο τύπους δικτύων ξεχωριστά. Ωστόσο, πολύ λίγες υλοποιήσεις μπορούν να συνδυάσουν αποδοτικά την επιτάχυνση των CNN και RNN μαζί, παρόλο που μοιράζονται αρκετά χαρακτηριστικά μεταξύ τους. Σκοπός της διπλωματικής εργασίας είναι η ανάπτυξη ενός υβριδικού επιταχυντή CNN-RNN για εφαρμογές βαθιάς μηχανικής μάθησης, με τη χρήση του προγραμματιστικού μοντέλου HLS/OpenCL, που επιτρέπει την σχεδίαση υλικού σε μια FPGA σε γλώσσα υψηλού επιπέδου.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

1. A. X. M. Chang and E. Culurciello, "Hardware accelerators for recurrent neural networks on FPGA," 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD, 2017, pp. 1-4, doi: 10.1109/ISCAS.2017.8050816.
2. C. Gao and F. Zhang, "FPGA-based Accelerator for Independently Recurrent Neural Network," 2018 IEEE 4th International Conference on Computer and Communications (ICCC), Chengdu, China, 2018, pp. 2075-2080, doi: 10.1109/CompComm.2018.8780644.
3. Y. Sun, B. Liu and X. Xu, "An OpenCL-Based Hybrid CNN-RNN Inference Accelerator On FPGA," 2019 International Conference on Field-Programmable Technology (ICFPT), Tianjin, China, 2019, pp. 283-286, doi: 10.1109/ICFPT47387.2019.00048
4. https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html
5. https://www.khronos.org/opencl/

Επικοινωνία: Μηλιάδης Παναγιώτης, pmiliad@cslab.ece.ntua.gr
Διονύσιος Πνευματικάτος, pnevmati@cslab.ece.ntua.gr, 6944763171

9.3 Υλοποίηση επιταχυντή για Collaborative - Filtering με τη χρήση του προγραμματιστικού μοντέλου HLS/OpenCL για FPGAs.

Στη σύγχρονη εποχή, τα συστήματα δημιουργίας προτάσεων (recommendation systems) βρίσκουν ευρεία εφαρμογή σε πληθώρα εμπορικών πλατφορμών και υπηρεσιών, όπως είναι το Netflix για ταινίες, Spotify για μουσική κ.α., για την δημιουργία προτάσεων, όσον αφορά το περιεχόμενο, ανάλογα με τις προσωπικές προτιμήσεις και το ιστορικό του χρήστη. Ο βασικός αλγόριθμος που χρησιμοποιείται ευρέως για την ανάπτυξη τέτοιων εφαρμογών καθώς και άλλων αναδυόμενων υπηρεσιών που βασίζονται στη μηχανική μάθηση είναι το "συνεργατικό φιλτράρισμα" (Collaborative Filtering). Το Collaborative Filtering είναι μια μέθοδος δημιουργίας αυτόματων προβλέψεων/προτάσεων (filtering) σχετικά με τις προτιμήσεις του χρήστη συλλέγοντας πληροφορίες από άλλους χρήστες (Collaborative). Στην εποχή των μεγάλων δεδομένων (big data), η συνεχής αύξηση της ποσότητας δεδομένων δημιουργεί σημαντικές προκλήσεις στους αλγορίθμους που βασίζονται στην τεχνική Collaborative - Filtering, καθώς τους καθιστά πολύ χρονοβόρους και, τα συστήματα που τους ενσωματώνουν, ενεργειακά σπάταλους. Αν και η βασική πλατφόρμα επιτάχυνσης του συγκεκριμένου αλγορίθμου είναι οι κάρτες γραφικών (GPUs), η χρήση της έρχεται με αρκετά μειονεκτήματα όπως είναι η χαμηλή υπολογιστική αποδοτικότητα και η υψηλή ισχύς. Στις μέρες μας, η χρήση των επαναπρογραμματιζόμενων αρχιτεκτονικών (FPGAs) αποτελεί σημαντική εναλλακτική πρόταση. Σκοπός της διπλωματικής εργασίας είναι η ανάπτυξη ενός επιταχυντή ο οποίος θα βασίζεται στην τεχνική Collaborative-Filtering, για εφαρμογές βαθιάς μηχανικής μάθησης και recommendation systems. Η δημιουργία του επιταχυντή θα γίνει μέσω της χρήση του προγραμματιστικού μοντέλου HLS/OpenCL, που επιτρέπει την σχεδίαση υλικού σε μια FPGA σε γλώσσα υψηλού επιπέδου.

Σχετικά Μαθήματα: Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

1. C. Wang, L. Gong, X. Ma, X. Li and X. Zhou, "WooKong: A Ubiquitous Accelerator for Recommendation Algorithms With Custom Instruction Sets on FPGA," in IEEE Transactions on Computers, vol. 69, no. 7, pp. 1071-1082, 1 July 2020, doi: 10.1109/TC.2020.2988209

2. X. Ma, C. Wang, Q. Yu, X. Li and X. Zhou, "An FPGA-Based Accelerator for Neighborhood-Based Collaborative Filtering Recommendation Algorithms," 2015 IEEE International Conference on Cluster Computing, Chicago, IL, 2015, pp. 494-495, doi: 10.1109/CLUSTER.2015.79.

3. `https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html`

4. `https://www.khronos.org/opencl/`

Επικοινωνία: Μηλιάδης Παναγιώτης, pmiliad@cslab.ece.ntua.gr
Διονύσιος Πνευματικάτος, pnevmati@cslab.ece.ntua.gr, 6944763171

III. Λειτουργικά Συστήματα - Εικονικές Μηχανές

10 Εικονική Μνήμη - TLBs

Η εικονική μνήμη αποτελεί θεμελιώδη "αφαίρεση" ενός υπολογιστικού συστήματος. Κάθε διεργασία χρησιμοποιεί το δικό της συνεχή χώρο εικονικών διευθύνσεων για να προσπελάσει τη μνήμη, ενώ το λειτουργικό σύστημα είναι υπεύθυνο για να αντιστοιχίσει τις εικονικές διευθύνσεις σε φυσικές διευθύνσεις μηχανήματος και να αποθηκεύσει/διατηρήσει/διαχειριστεί αυτές τις αντιστοιχίσεις σε πίνακες σελίδων (page tables) για κάθε διεργασία. Για να εξυπηρετήσει μια εντολή προσπέλασης μνήμης ο επεξεργαστής πρέπει να μεταφράσει την εικονική διεύθυνση (virtual address) σε φυσική (physical address). Για λόγους επίδοσης, οι επεξεργαστες χρησιμοποιούν εξειδικευμένες κρυφές μνήμες (Translation Lookaside Buffers, TLBs) για την αποθήκευση των πιο πρόσφατων μεταφράσεων.

10.1 Βελτιστοποίηση μεθόδων διαχείρισης φυσικής μνήμης για την υποστήριξη σύγχρονων ερευνητικών τεχνολογιών μετάφρασης διευθύνσεων (state-of-the-art address translation schemes)

Στους σύγχρονους υπολογιστές τα TLBs αποθηκεύουν μεταφράσεις σε επίπεδο σελίδων, μικρών (4K) ή μεγαλύτερων (2M, 1G). Ωστόσο οι όλο αυξανόμενες απαιτήσεις των εφαρμογών σε μνήμη περιορίζουν την αποδοτικότητα των TLBs, καθώς αποτυγχάνουν να καλύψουν την ενεργά χρησιμοποιούμενη μνήμη (working set) των εφαρμογων. Στη σύγχρονη ερευνητική βιβλιογραφία έχουν προταθεί σχεδιασμοί TLBs που επιτρέπουν την αποθήκευση πολλαπλών μεταφράσεων σελίδων σε μια θέση του TLB για να αυξηθεί το reach του, δηλαδή το εύρος των εικονικών διευθύνσεων που μπορέι να μεταφράσει, χωρις να μεγαλώσει το μέγεθος του. Προυπόθεση είναι η ύπαρξη μεγάλων συνεχών αντιστοιχίσεων σελίδας (contiguous page mappings), δηλαδή πολλές ακολουθητικές εικονικές σελίδες να αντιστοιχούνται σε ακολουθητικές φυσικές σελίδες. Ο διαχειριστής μνήμης των λειτουργικών συστημάτων δε διαθέτει μηχανισμούς για τη δημιουργία τέτοιων αντιστοιχίσεων με εύκολο τρόπο. Πρόσφατες ερευνητικές προτάσεις εξετάζουν μηχανισμούς είτε επεκτείνοντας τον κατανεμητη μνήμης τους λειτουργικού (allocator) είτε δημιουργώντας αυτόνομα νήματα πυρήνα (kernel threads) που μετακινούν σελίδες (page migrations). Η παρούσα διπλωματική έχει ως σκοπό να συνδυάσει τις δυο προτάσεις.

Σχετικά Μαθήματα: Εργαστήριο Λειουργικών Συστημάτων

Σχετική Βιβλιογραφία:

1. Translation Ranger: Operating System Support for Contiguity-Aware TLBs
2. Enhancing and Exploiting Contiguity for Fast Memory Virtualization

Επικοινωνία: Χλόη Αλβέρτη, xalverti@cslab.ece.ntua.gr, 210-772-2279
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133

10.2 Μελέτη διαφορετικών τεχνικών ακύρωσης TLB καταχωρήσεων (TLB shootdowns) για την επίτευξη συνάφειας

Η ιεραρχία TLBs των σύγχρονων επεξεργαστών δεν είναι διαμοιραζόμενη. Σε ενα πολυεπεξεργαστικό μηχάνημα κάθε επεξεργαστής διαθέτει τη δική του ιδιωτική ιεραρχία TLBs. Σε αντίθεση με τις κρυφές μνήμες δεδομένων (data caches), η συνάφεια των TLBs (TLB coherence) δε διασφαλίζεται από το υλικό

(hardware) αλλά από το λογισμικό (software). Το λειτουργικό σύστημα είναι υπεύθυνο οι καταχωρήσεις των TLBs να είναι πάντα ενημερωμένες με βάση τις καταχωρήσεις μεταφράσεων στους πίνακες σελίδων (page tables). Για να το πετύχει αυτό χρησιμοποιεί ειδικές εντολές ακύρωσης TLB καταχωρήσεων (TLB invalidation). Όταν η εφαρμογή που τρέχει είναι σειριακή (ένα νήμα), η διαδικασία ακύρωσης είναι σχετικά φθηνή γιατί αρκεί η "εκκαθάριση" της τοπικής ιεραρχίας TLBs ενός επεξεργαστή (local TLB flush). Όταν η εφαρμογή είναι παράλληλη (πολλαπλά νήματα), τότε η ακύρωση πρέπει να γίνει και σε απομακρυσμένες CPU (remote TLB shootdown). Τα shootdowns είναι ακριβές λειτουργίες γιατί περιλαμβάνουν σύγχρονες διακοπές μεταξύ των επεξεργαστών (inter-processors interrupts (IPI)) για την σύγχρονη ακύρωση TLB καταχωρήσεων. Τα IPIs μπορούν να επηρρεάσουν σημαντικά την επίδοση μιας εφαρμογής και να μειώσουν την κλιμακωσιμότητα της (scalability). Για το λόγο αυτό, στη σύγχρονη βιβλιογραφία υπάρχουν προτάσεις για την υλοποίηση μηχανισμών shootdown που αποφεύγουν τα συχνά IPIs ή και τα αντικαθιστούν πλήρως με άλλους μηχανισμούς. Σκοπός της διπλωματικής είναι η μελέτη και υλοποίηση διαφορετικών τεχνικών TLB shootdown.

Σχετικά Μαθήματα: Εργαστήριο Λειτουργικών Συστημάτων

Σχετική Βιβλιογραφία:

1. Don't shoot down TLB shootdowns!
2. Latr: Lazy Translation Coherence

Επικοινωνία: Χλόη Αλβέρτη, xalverti@cslab.ece.ntua.gr, 210-772-2279
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133

10.3 Μελέτη, σχεδιασμός, και υλοποίηση επεκτάσεων των μηχανισμών εικονικής μνήμης για μεγάλες 2MB σελίδες

Ένας τρόπος μείωσης του κόστους της εικονικής μνήμης είναι η αύξηση του TLB reach, δηλαδή του εύρους των εικονικών διευθύνσεων που το TLB μπορεί να μεταφράσει. Ένας τρόπος αύξησης του TLB reach είναι η χρήση μεγάλων σελίδων μεγέθους 2MB. Ωστόσο η δημιουργία και η διαχείριση μεγάλων σελίδων, επιφέρει δυσκολίες/προκλήσεις στη διαχείριση μνήμης (π.χ fragmentation, memory bloat, unfairness, page sharing). Σκοπός της παρούσας εργασίας είναι η μελέτη αυτών των προβλημάτων και η διερεύνηση πολιτικών διαχείρισης με χρήση των huge pages σε συνέχεια προηγούμενης διπλωματικής εργασίας.

Σχετικά Μαθήματα: Εργαστήριο Λειτουργικών Συστημάτων, Προηγμένα Θέματα Αρχιτεκτονικής Υπολογιστών

Σχετική Βιβλιογραφία:

1. Ingens: Huge Page Support for the OS and Hypervisor
2. HawkEye: Efficient Fine-grained OS Support for Huge Pages
3. Nimble Page Management for Tiered Memory Systems
4. MEGA: overcoming traditional problems with OS huge page management
5. A Comprehensive Analysis of Superpage Management Mechanisms and Policies

Επικοινωνία: Χλόη Αλβέρτη, xalverti@cslab.ece.ntua.gr, 210-772-2279
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133

11 Ανάλυση και βελτιστοποίηση του μηχανισμού και των πολιτικών διαχείρισης μνήμης Automatic NUMA balancing για αρχιτεκτονικές με ανομοιόμορφη πρόσβαση μνήμης

Ένας από τους τρόπους αύξησης της κλιμακωσίμοτητας των υπολογιστικών συστημάτων που έχουν πολλαπλούς επεξεργαστές στο ίδιο σύστημα είναι μέσω των Αρχιτεκτονικών με Ανομοιόμορφη Πρόσβαση Μνήμης ή αλλιώς Non Uniform Memory Access (NUMA) systems. Η NUMA αρχιτεκτονική αποτελεί έναν σχεδιασμό συστήματος μνήμης πολυεπεξεργαστικών υπολογιστικών συστημάτων, στα οποία ο χρόνος πρόσβασης της κύριας μνήμης (RAM) εξαρτάται από την απόσταση της θέσης μνήμης που προσπελαύνει ο επεξεργαστής. Σε μία NUMA αρχιτεκτονική, ένας επεξεργαστής έχει γρηγορότερη πρόσβαση σε μία τοπική μνήμη από ότι σε μία απομακρυσμένη, η οποία όμως είναι τοπική για κάποιον άλλον επεξεργαστή. Το λειτουργικό σύστημα Linux παρέχει τον μηχανισμό Automatic NUMA Balancing ο οποίος μεταφέρει δυναμικά δεδομένα ανάμεσα σε κόμβους μνήμης για να μειώσει τον χρόνο πρόσβασης στην μνήμη. Στόχος της παρούσας διπλωματικής είναι η μελέτη, η ανάλυση, και η βελτιστοποίηση του μηχανισμού Automatic NUMA balancing του Linux, καθώς επίσης και η υλοποίηση αποδοτικών πολιτικών χρήσης του.

Σχετικά Μαθήματα: Εργαστήριο Λειτουργικών Συστημάτων

Σχετική Βιβλιογραφία:

1. Automatic Non-Uniform Memory Access (NUMA) Balancing, SUSE
2. Automatic NUMA Balancing, Red Hat
3. NUMA Memory Architectures and the Linux Memory System, Red Hat

Επικοινωνία: Χλόη Αλβέρτη, xalverti@cslab.ece.ntua.gr, 210-772-2279
Βασίλης Καρακώστας, vkarakos@cslab.ece.ntua.gr, 210-772-4133

12 Υλοποίηση utmem (Userspace Transcendent Memory) σε ARM πλατφόρμες

Η transcendent μνήμη (transcendent memory (tmem)) αποτελεί μία προσέγγιση για καλύτερη χρήση της υποχρησιμοποιούμενης μνήμης σε ένα εικονικοποιημένο περιβάλλον. Σε τέτοιες περιπτώσεις ο host μπορεί με δυναμικό τρόπο να διαχειρίζεται αποδοτικότερα ένα tmem pool μεταξύ των εικονικών μηχανών (VMs). Ένας τέτοιος μηχανισμός φαντάζει ακόμα περισσότερο χρήσιμος σε πλατφόρμες με μειωμένους πόρους σε μνήμη, όπως πχ ένα ενσωματωμένο board. Σκοπός αυτής της εργασίας είναι η μελέτη του μηχανισμού utmem (Userspace Transcendent Memory) προηγούμενης διπλωματικής εργασίας, η τροποποίηση (porting) του για ARM πλατφόρμες και η πειραματική του αποτίμηση.

Σχετικά Μαθήματα: Λειτουργικά Συστήματα, Εργαστήριο Λειτουργικών Συστημάτων

Επικοινωνία: Στράτος Ψωμαδάκης, psomas@cslab.ece.ntua.gr
Ορέστης Λάγκας-Νικολός, olagkas@cslab.ece.ntua.gr
Κωνσταντίνος Παπαζαφειρόπουλος, kpapazaf@cslab.ece.ntua.gr

13 Επιτάχυνση υλικού για αποδοτική εκτέλεση εφαρμογών ως unikernels

Μια ενδιαφέρουσα προσέγγιση στη μείωση του θορύβου του λειτουργικού συστήματος και περιττών εξαρτήσεων στο περιβάλλον εκτέλεσης μιας εφαρμογής είναι η δημιουργία ενός λεπτού στρώματος εξαρτήσεων (βιβλιοθήκες, λειτουργικό σύστημα) και η σύνθεση ενός ενιαίου εκτελέσιμου αρχείου της εφαρμογής (unikernel), που θα μπορεί να εκτελεστεί αυτόνομα, όπως σε ένα κοινό λειτουργικό σύστημα. Ταυτόχρονα, η διεύρυνση της χρήσης επιταχυντών υλικού για υπολογιστικά απαιτητικά κομμάτια εφαρμογών καθιστά το υλικό περισσότερο προσβάσιμο, και άρα διαθέσιμο σε περιβάλλοντα cloud (Amazon AWS, Azure, κλπ). Στόχος της παρούσας εργασίας είναι η σχεδίαση και υλοποίηση ενός συστήματος που θα συνδυάζει την απάλειψη περιττών εξαρτήσεων της εφαρμογής από το περιβάλλον εκτέλεσης (unikernel) καθώς και την ένταξη επιτάχυνσης συγκεκριμένων υπολογιστικά απαιτητικών κομματιών της εφαρμογής. Συγκεκριμένα, η εργασία περιλαμβάνει: (α) μελέτη των διαθέσιμων frameworks για unikernels, (β) αποδελτίωση εφαρμογών που αξιοποιούν την επιτάχυνση υλικού σε GPUs/FPGAs, (γ) υλοποίηση του συστήματος που παράγει unikernels με αυτή την υποστήριξη. και (δ) πειραματική αποτίμηση του συστήματος.

Σχετικά Μαθήματα: Λειτουργικά Συστήματα, Εργαστήριο Λειτουργικών Συστημάτων

Σχετική Βιβλιογραφία:

- Unikernel frameworks:
    1. `https://github.com/cloudius-systems/osv`
    2. `http://rumpkernel.org/`
    3. `https://github.com/libos-nuse/lkl-linux`
    4. `http://cnp.neclab.eu/clickos/`
    5. `https://wiki.xenproject.org/wiki/Mini-OS`
- Acceleration:
    1. `https://www.khronos.org/opencl/`
    2. `https://www.xilinx.com/products/design-tools/software-zone/sdaccel.html`

Επικοινωνία: Κωνσταντίνος Παπαζαφειρόπουλος, kpapazaf@cslab.ece.ntua.gr
Στράτος Ψωμαδάκης, psomas@cslab.ece.ntua.gr
Ορέστης Λάγκας-Νικολός, olagkas@cslab.ece.ntua.gr

IV. Κατανεμημένα Συστήματα - Προχωρημένα θέματα βάσεων δεδομένων

14 Αυτόματη επιλογή και ταξινόμηση δεδομένων εισόδου βάσει χρησιμότητας για αναλυτικές εργασίες μεγάλου όγκου δεδομένων

Ενώ η βελτιστοποίηση εργασιών στον τομέα των Big Data συνήθως υλοποιείται με την αύξηση της παραλληλοποίησης και τη χρήση πλατφορμών κατανεμημένης επεξεργασίας, λίγα έχουν γίνει σχετικά με την επιλογή των πιο κατάλληλων δεδομένων από μια μεγάλη συλλογή διαθέσιμων. Πολλές αναλυτικές εργασίες είναι ιδιαίτερα ευαίσθητες στο περιεχόμενο των δεδομένων και όχι τόσο στο μέγεθός τους (π.χ., content based advertising, social network analytics). Στις εργασίες [1, 2] παρουσιάσαμε μια γενική μεθοδολογία για γρήγορη σύγκριση και ταξινόμηση πολλαπλών διαθέσιμων δεδομένων εισόδου (datasets) με βάση το αποτέλεσμα που προκαλούν όταν εφαρμόζονται σε τελεστές αναλυτικής επεξεργασίας. Στη συγκεκριμένη διπλωματική, καλείστε να υλοποιήσετε και να βελτιστοποιήσετε την επέκταση του συστήματος σε μια από τις ακόλουθες κατευθύνσεις:

- Σε δεδομένα κειμένου. Συγκεκριμένα, για τελεστές που παίρνουν σαν είσοδο 1 αρχείο κειμένου το σύστημα πρέπει να μοντελοποιεί τις διάφορες ανάμεσα σε πολλά κείμενα καθώς και να προβλέπει την έξοδο του τελεστή για οποιοδήποτε κείμενο με το ελάχιστο σφάλμα.
- Σε τελεστές που δέχονται περισσότερες της μιας εισόδους (π.χ. Join operator). Το σύστημα θα πρέπει να τροποποιηθεί ώστε να μοντελοποιεί την επίδραση που έχουν 2 dataset εισόδου καθώς και οι ομοιότητές τους στην πρόβλεψη του αποτελέσματος του τελεστή.

Επιθυμητή και είναι η υποβολή δημοσίευσης από την παραπάνω εργασία σε σχετικό workshop.

Σχετικά Μαθήματα: Προχωρημένα θέματα βάσεων δεδομένων, Κατανεμημένα Συστήματα

Σχετική Βιβλιογραφία:

1. T. Bakogiannis, I. Giannakopoulos, D. Tsoumakos and N. Koziris: Apollo: A Dataset Profiling and Operator Modeling System. In Proceedings of the 2019 ACM SIGMOD/PODS.
2. I. Giannakopoulos, D. Tsoumakos and N. Koziris: A Content-Based Approach for Modeling Analytics Operators. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management.

Επικοινωνία: Δημήτριος Τσουμάκος, dtsouma@cslab.ece.ntua.gr

15 Αποδοτικός συνεχής υπολογισμός ερωτημάτων σε δυναμικά δεδομένα γράφων με τη χρήση του Timely Dataflow

Η ανάλυση σε μεγάλα δεδομένα γράφων τόσο σε στατικό όσο και δυναμικό (δλδ, ο γράφος διαρκώς μεταβάλλεται με εισαγωγές & διαγραφές κόμβων και ακμών) επίπεδο έχει εξελιχθεί σε πολύ σημαντικό πεδίο έρευνας και ανάπτυξης. Ένα από τα πιο σύγχρονα εργαλεία που επιτρέπουν τέτοιους υπολογισμούς είναι το Naiad [1], που υλοποιεί το Timely-Dataflow υπολογιστικό μοντέλο.

Το μοντέλο υποστηρίζει επαναληπτικούς και συνεχείς υπολογισμούς. Δίνει τη δυνατότητα επεξεργασίας ροών και εργασιών δέσμης με ταχύτητα, χρησιμοποιώντας μια νέα προσέγγιση συντονισμού που συνδυάζει ασύγχρονη και σύγχρονη εκτέλεση. Το Differential dataflow [2] εκτελεί επαναληπτικό υπολογισμό σε ροές δεδομένων με τον υπολογισμό να υφίσταται μόνο σε απόκριση προς την αλλαγή των δεδομένων. Στη συγκεκριμένη διπλωματική, καλείστε να χρησιμοποιήσετε την open-source έκδοση σε Rust (`https://github.com/frankmcsherry/timely-dataflow` & `https://github.com/frankmcsherry/differential-dataflow`) και, αφού υλοποιήσετε γνωστούς αλγορίθμους γράφων (π.χ. Triangle counting, pagerank, centralities) να συγκρίνετε τη streaming εκδοχή τους στο Timely-Dataflow με το GraphX (Spark) [3].

Σχετικά Μαθήματα: Προχωρημένα θέματα βάσεων δεδομένων, Κατανεμημένα Συστήματα

Σχετική Βιβλιογραφία:

1. Derek G. Murray, Frank McSherry, Rebecca Isaacs, Michael Isard, Paul Barham, and Martín Abadi. Naiad: A timely dataflow system. In SOSP, pages 439–455, 2013.

2. Frank McSherry, Derek Gordon Murray, Rebecca Isaacs, and Michael Isard. 2013. Differential Dataflow. In Proc. Conf. on Innovative Data Systems Research (CIDR).

3. Joseph E. Gonzalez, Reynold S. Xin, Ankur Dave, Daniel Crankshaw, Michael J. Franklin, Ion Stoica. GraphX: Graph Processing in a Distributed Dataflow Framework. In USENIX Symposium on Operating Systems Design and Implementation (OSDI 14).

Επικοινωνία: Δημήτριος Τσουμάκος, dtsouma@cslab.ece.ntua.gr

16 Μελέτη και σύγκριση πολυ-συστημάτων (polystores) εκτέλεσης αναλυτικών SQL ερωτημάτων

Το Presto [1,2] είναι μια κατανεμημένη μηχανή εκτέλεσης SQL ερωτημάτων ανοιχτού κώδικα για τη διεξαγωγή διαδραστικών αναλυτικών ερωτημάτων σε πηγές δεδομένων όλων των μεγεθών που κυμαίνονται από gigabytes έως petabytes. Το Presto επιτρέπει την αναζήτηση δεδομένων σε πολλαπλές βάσεις όπως Hive, Cassandra, MySQL, MongoDB, ElasticSearch, κλπ.. Ένα ερώτημα Presto μπορεί να συνδυάσει δεδομένα από πολλαπλές πηγές, ανήκοντας στην κατηγορία των "polystores". Στη συγκεκριμένη διπλωματική, καλείστε να μελετήσετε τις δυνατότητες του Presto και να το συγκρίνετε με τα "συγγενικά" MuSQLE [3], Impala[4] και SparkSQL [5] σχετικά με:

- βελτιστοποίηση ερωτημάτων,
- κλιμακωσιμότητα,
- απόδοση.

Σχετικά Μαθήματα: Προχωρημένα θέματα βάσεων δεδομένων, Κατανεμημένα Συστήματα

Σχετική Βιβλιογραφία:

1. R. Sethi et al., "Presto: SQL on Everything," 2019 IEEE 35th International Conference on Data Engineering (ICDE).

2. `https://github.com/prestosql/presto`

3. V. Giannakouris, N. Papailiou, D. Tsoumakos and N. Koziris: MuSQLE: Distributed SQL Query Execution Over Multiple Engine Environments. In Proceedings of the 2016 IEEE International Conference on Big Data (BigData 2016).

4. `https://impala.apache.org/`

5. `https://spark.apache.org/sql/`

Επικοινωνία: Δημήτριος Τσουμάκος, dtsouma@cslab.ece.ntua.gr

17 Benchmarking Αλγορίθμων Consensus σε Ethereum/Hyperledger

Η τεχνολογία blockchain, που αρχικά δημιουργήθηκε για να αποτελέσει τη βάση λειτουργίας του δικτύου Bitcoin, λειτουργεί ως ένα κοινόχρηστο δημόσιο λογιστικό βιβλίο στο οποίο εγγράφονται όλες οι επιβεβαιωμένες συναλλαγές – ένα σύνολο συναλλαγών αποτελούν ένα block και το κάθε block αναφέρεται στο προηγούμενο του δημιουργώντας μια αλυσίδα [1]. Η επιβεβαίωση των συναλλαγών και η συμφωνία για τη σειρά εκτέλεσής τους γίνεται με κατανεμημένο τρόπο με χρήση αλγορίθμων consensus. Οι δύο βασικές κατηγορίες τέτοιων αλγορίθμων είναι οι lottery-based (π.χ., ο Proof-of-Work του Bitcoin) και οι voting-based (π.χ., ο Byzantine Fault Tolerance του Hyperledger) [2, 3]. Καθένας από του αλγορίθμους αυτού έχουν πλεονεκτήματα και μειονεκτήματα σε σχέση με το transaction throughput που επιτυγχάνουν, την κλιμακωσιμότητά τους, την ασφάλεια που παρέχουν, ακόμα και την ενέργεια που σπαταλούν. Στόχος της διπλωματικής είναι η μελέτη διαφορετικών αλγορίθμων consensus που χρησιμοποιούνται στις πιο δημοφιλείς υλοποιήσεις Blockchain (π.χ., Ethereum [4], Hyperledger [5]) με τη χρήση benchmarks (Blockbench [6] ή Hyperledger Caliper [7] αντίστοιχα) ως προς τις παραπάνω ιδιότητες.

Σχετικά Μαθήματα: Κατανεμημένα Συστήματα

Σχετική Βιβλιογραφία:

1. Nakamoto, Satoshi. ”Bitcoin: A peer-to-peer electronic cash system.” (2008): 28.

2. Bano, S., Sonnino, A., Al-Bassam, M., Azouvi, S., McCorry, P., Meiklejohn, S., Danezis, G. (2017). Consensus in the age of blockchains. arXiv preprint arXiv:1711.03936.

3. Hyperledger Architecture, Volume 1

4. Ethereum

5. Hyperledger

6. Dinh, Tien Tuan Anh, et al. ”Blockbench: A framework for analyzing private blockchains.” Proceedings of the 2017 ACM International Conference on Management of Data. ACM, 2017.

7. Hyperledger Caliper

Επικοινωνία: Κατερίνα Δόκα, katerina@cslab.ece.ntua.gr, 210-772-1175

18 Μελέτη και βελτιστοποίηση του επιπέδου αποθήκευσης των Blockchain clients

Οι κόμβοι του Blockchain (κυρίως οι full nodes) αποθηκεύουν μεγάλο όγκο δεδομένων που σχετίζονται με το state του, τα transactions που έχουν γίνει, τα δεδομένα των έξυπνων συμβολαίων [1]. Συνήθως τα δεδομένα αυτά φυλάσσονται σε δενδρικές δομές αποθήκευσης (tries) που προσφέρουν γρήγορη αναζήτηση και που υλοποιούνται με τη βοήθεια κάποιου key-value store (leveldb [2], rocksdb [3]). Ωστόσο το storage layer μπορεί σε κάποιες λειτουργίες του πρωτοκόλου να αποτελεί performance bottleneck, όπως για παράδειγμα στο αρχικό sync ενός κόμβου που μόλις πρωτοεισέρχεται στο blockchain [4].

Στόχος της διπλωματικής είναι η μελέτη του workload που καλείται να εξυπηρετήσει το storage layer ενός blockchain client κατά τις διάφορες φάσεις της λειτουργίας του και η βελτιστοποίηση της απόδοσής του με χρήση διαφορετικών δομών αποθήκευσης ή/και τεχνικών caching [5].

Σχετικά Μαθήματα: Κατανεμημένα Συστήματα

Σχετική Βιβλιογραφία:

1. Getting Deep Into Ethereum: How Data Is Stored In Ethereum?
2. LevelDB
3. RocksDB
4. Why Syncing Ethereum Node Is Slow
5. Gorenflo, Christian, et al. ”Fastfabric: Scaling hyperledger fabric to 20,000 transactions per second.” 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). IEEE, 2019.

Επικοινωνία: Κατερίνα Δόκα, katerina@cslab.ece.ntua.gr, 210-772-1175

19 Επαληθεύσιμη κατανεμημένη επεξεργασία με χρήση Blockchain

Η διαφάνεια συναλλαγών και η έλλειψη κεντρικού ελέγχου που προσφέρει η τεχνολογία blockchain έχουν οδηγήσει στην υιοθέτησή της σε πληθώρα εφαρμογών πέρα από το νόμισμα (διαχείριση ψηφιακών περιουσιακών στοιχείων, από τίτλους ιδιοκτησίας και μετοχές μέχρι ταξιδιωτικά μίλια, δημιουργία ψηφιακών ταυτοτήτων που χρησιμοποιούνται σε ψηφιακές υπογραφές, δημιουργία επαληθεύσιμης καταγραφής για κάθε είδους δεδομένο, αρχείο ή διεργασία). Η δυνατότητα που προσφέρουν κάποια blockchains για εκτέλεση έξυπνων συμβολαίων, κώδικα δηλαδή που εκτελείται αυτόματα όταν ικανοποιηθούν ορισμένες συνθήκες, έχει δώσει ακόμα μεγαλύτερη ώθηση στη χρήση τους. Παράδειγμα εφαρμογής που έχει αναπτυχθεί στο εργαστήριο είναι μια πλατφόρμα για δημιουργία πραγματικά αποκεντρωμένης υποδομής Cloud [1,2].

Σκοπός της διπλωματικής είναι να εκμεταλλευτούμε τα πλεονεκτήματα των blockchains για υποστηρίξουμε την επαληθεύσιμη εκτέλεση κατανεμημένων προγραμμάτων, ακόμα και σε συνθήκες όπου δεν εμπιστευόμαστε τους συμμετέχοντες κόμβους. Πρακτικά, θα ενισχύσουμε υπάρχοντα δημοφιλή κατανεμημένα frameworks (Hadoop MapReduce [3], Spark [4]) με μηχανισμούς επαλήθευσης που θα στηρίζονται στην τεχνολογία blockchain (π.χ. truebit [5]). Έτσι θα διασφαλίσουμε ότι κανένας κακόβουλος κόμβος δε θα μπορέσει να αλλάξει το αποτέλεσμα ενός κατανεμημένου υπολογισμού.

Σχετικά Μαθήματα: Κατανεμημένα Συστήματα

Σχετική Βιβλιογραφία:

1. Katerina Doka, Tasos Bakogiannis, Ioannis Mytilinis and Georgios Goumas: CloudAgora: Democratizing the Cloud. In Proceedings of the 2nd International Conference on Blockchain (ICBC), 2019, San Diego, CA, USA.

2. Tasos Bakogiannis, Ioannis Mytilinis, Katerina Doka and Georgios Goumas: Building Ad-Hoc Clouds with CloudAgora. Accepted for publication at the 38th International Symposium on Reliable Distributed Systems (SRDS 2019), October 1-4, Lyon, France.

3. Apache Hadoop MapReduce

4. Apache Spark

5. `https://truebit.io/`

Επικοινωνία: Κατερίνα Δόκα, katerina@cslab.ece.ntua.gr, 210-772-1175

20 Εφαρμογές τεχνικών mechanism design (υποκλάδος της αλγοριθμικής θεωρίας παιγνίων) και deep learning για αποδοτικές online δημοπρασίες πόρων μεταξύ χρηστών σε περιβάλλοντα cloud

Στα σύγχρονα μεγάλα υπολογιστικά νέφη η δυναμική εκχώρηση πόρων σε χρήστες ή tasks χρηστών ανάλογα με την ανάγκη του χρήστη και τη διαθεσιμότητα πόρων στο δίκτυο μια δεδομένη στιγμή εφάπτεται στο πρόβλημα του αποδοτικού resource allocation. Με την άνοδο του machine learning και την ανάπτυξη περισσότερων non-critical ή easy scalable jobs που μπορεί να τρέξει ο χρήστης σε μια cloud υπηρεσία και για να αντιμετωπιστεί το πρόβλημα των πόρων που μένουν αχρησιμοποίητοι στο AWS, η Amazon δημιούργησε την υπηρεσία Spot Instances όπου τιμές των Vms διαμορφώνονται κατά το δοκούν περιοδικά, ανάλογα με τη ζήτηση. Αυτό εμπεριέχει τον κίνδυνο κάποιος χρήστης να απωλέσει μεγάλο αριθμό των πόρων του χωρίς προειδοποίηση. Στο εργαστήριο αναπτύξαμε έναν decision component, ονόματει Game Master, που εφαρμόζεται στο Spot Instances και διενεργεί online δημοπρασίες περιοδικά με στόχο οι πόροι ενός χρήστη να αυξομειώνονται ελαστικά. Για να διασφαλίσουμε την εγκυρότητα, την τιμιότητα και τη φιλαλήθεια των παραπάνω δημοπρασιών χρησιμοποιούμε τεχνικές δανεισμένες από το mechanism design- ένας υποκλάδος της αλγοριθμικής θεωρίας παιγνίων που στόχο έχει την κατασκευή μηχανισμών που οδηγούν τους παίκτες ενός παιγνίου να παρουσιάζουν φιλαλήθη συμπεριφορά κατά τη συμμετοχή τους στο παίγνιο. Επίσης χρησιμοποιούμε μια προσέγγιση ενός deep learning min max νευρωνικού δικτύου. Καταφέρνουμε να πετύχουμε οφέλη για σωρεία διαφορετικών περιπτώσεων όπως το να πετύχουμε μεγάλα κέρδη για τον cloud vendor ή μέγιστη κοινωνική ωφέλεια για τους χρήστες (οι χρήστες μένουν στην πλειονότητά τους ευχαριστημένοι) Οι προσομοιώσεις πάνω στις οποίες τεστάραμε τον component αφορούν στατιστικά στοιχεία από το Google Trace. Στόχος μας είναι να η δημιουργία ενός actual cluster με διαφορετικούς χρήστες, με διαφορετικά non-critical applications που συμμετέχουν στις άνωθι δημοπρασίες του Game Master περιοδικά, προσθαφαιρούνται πόροι τους και να δείξουμε πως τα κριτήρια που εμφανώς πληρούνται στις προσομοιώσεις, πληρούνται και σε actual executions environemtns. Παράλληλα η χρησιμοποίηση του Game Master σε περιβάλλοντα ετερογενών αρχιτεκτονικών θα μπορούσε να εξεταστεί σαν υποψήφιο θέμα.
Σχετική Βιβλιογραφία:

1. https://aws.amazon.com/ec2/spot/

2. L. Zhang, Z. Li and C. Wu, "Dynamic resource provisioning in cloud computing: A randomized auction approach," IEEE INFOCOM 2014 - IEEE Conference on Computer Communications, Toronto, ON, 2014, pp. 433-441.

3. W. Shi, L. Zhang, C. Wu, Z. Li and F. C. M. Lau, "An Online Auction Framework for Dynamic Resource Provisioning in Cloud Computing," in IEEE/ACM Transactions on Networking, vol. 24, no. 4, pp. 2060-2073, Aug. 2016.

4. Dütting, Paul, Zhe Feng, Harikrishna Narasimhan, David C. Parkes and Sai Srivatsa Ravindranath. "Optimal Auctions through Deep Learning." ICML (2017).

Επικοινωνία: Κωνσταντίνος Μπιτσάκος, kbitsak@cslab.ece.ntua.gr
Ιωάννης Κωνσταντίνου, ikons@cslab.ece.ntua.gr

21 Αποτίμηση λειτουργίας συστημάτων επεξεργασίας μεγάλου όγκου δεδομένων πάνω σε skewed σύνολα δεδομένων.

Η ανάλυση μεγάλου όγκου δεδομένων (Big Data) με κατανεμημένα συστήματα επεξεργασίας είναι ένας από τους πιο δημοφιλείς τομείς ανάπτυξης τα τελευταία χρόνια. Τα περισσότερα κατανεμημένα συστήματα επεξεργασίας όμως, δε μπορούν να επεξεργαστούν αποδοτικά δεδομένα του παρουσιάζουν ανομοιομορφίες (skew). Συγκεκριμένα, η ανάλυση δεδομένων με χρήση συνενώσεων (joins) και συνυπολογισμών (aggregations) επηρεάζεται σημαντικά από το skew των δεδομένων, και τα εν λόγω συστήματα επεξεργασίας παρουσιάζουν συνήθως χαμηλή απόδοση σε τέτοιου τύπου ανάλυση πάνω σε skewed δεδομένα. Παρόλο που έχουν γίνει ερευνητικές προσπάθειες αντιμετώπισης του φαινομένου του data skew σε αλγοριθμικό επίπεδο και πάνω σε ειδικά συστήματα [1,2,3], τα περισσότερα από τα υπάρχοντα κατανεμημένα συστήματα επεξεργασίας δεν έχουν υλοποιήσει σχετικούς μηχανισμούς για την αντιμετώπισή του. Το Apache Spark είναι ένα από τα πιο δημοφιλή συστήματα επεξεργασίας μεγάλου όγκου δεδομένων, και παρέχει το SparkSQL[4] module για την ανάλυση δεδομένων με χρήση SQL ερωτημάτων. Σε μια προσπάθεια αντιμετώπισης του προβλήματος αυτού, ανέπτυξε πρόσφατα το adaptive execution framework [5]. Στη συγκεκριμένη διπλωματική, καλείστε να μελετήσετε το adaptive execution framework του Spark SQL, και να αξιολογήσετε την απόδοση του συστήματος με διαφορετικά επίπεδα data skew και με διαφορετικές ρυθμίσεις του συστήματος χρησιμοποιώντας το TPC-DS benchmark.

Σχετικά Μαθήματα: Προχωρημένα θέματα βάσεων δεδομένων, Κατανεμημένα Συστήματα

Σχετική Βιβλιογραφία:

1. Y. Kwon, M. Balazinska, B. Howe, and J. Rolia. SkewTune: mitigating skew in mapreduce applications. In Proceedings of the 2012 ACM SIGMOD.

2. R. Li, M. Riedewald, and X. Deng. Submodularity of Distributed Join Computation. In Proceedings of the 2018 ACM SIGMOD.

3. W. Rödiger, S. Idicula, A. Kemper and T. Neumann. Flow-Join: Adaptive skew handling for distributed joins over high-speed networks. In Proccedings of the 2016 IEEE ICDE.

4. https://spark.apache.org/sql/

5. https://databricks.com/blog/2020/05/29/adaptive-query-execution-speeding-up-spark-sql-at-runtim
   html

Επικοινωνία: Ευδοκία Κασσέλα, evie@cslab.ece.ntua.gr
Ιωάννης Κωνσταντίνου, ikons@cslab.ece.ntua.gr