ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΑΣΚΗΣΗ ΜΕΤΑΦΡΑΣΤΕΣ 2024

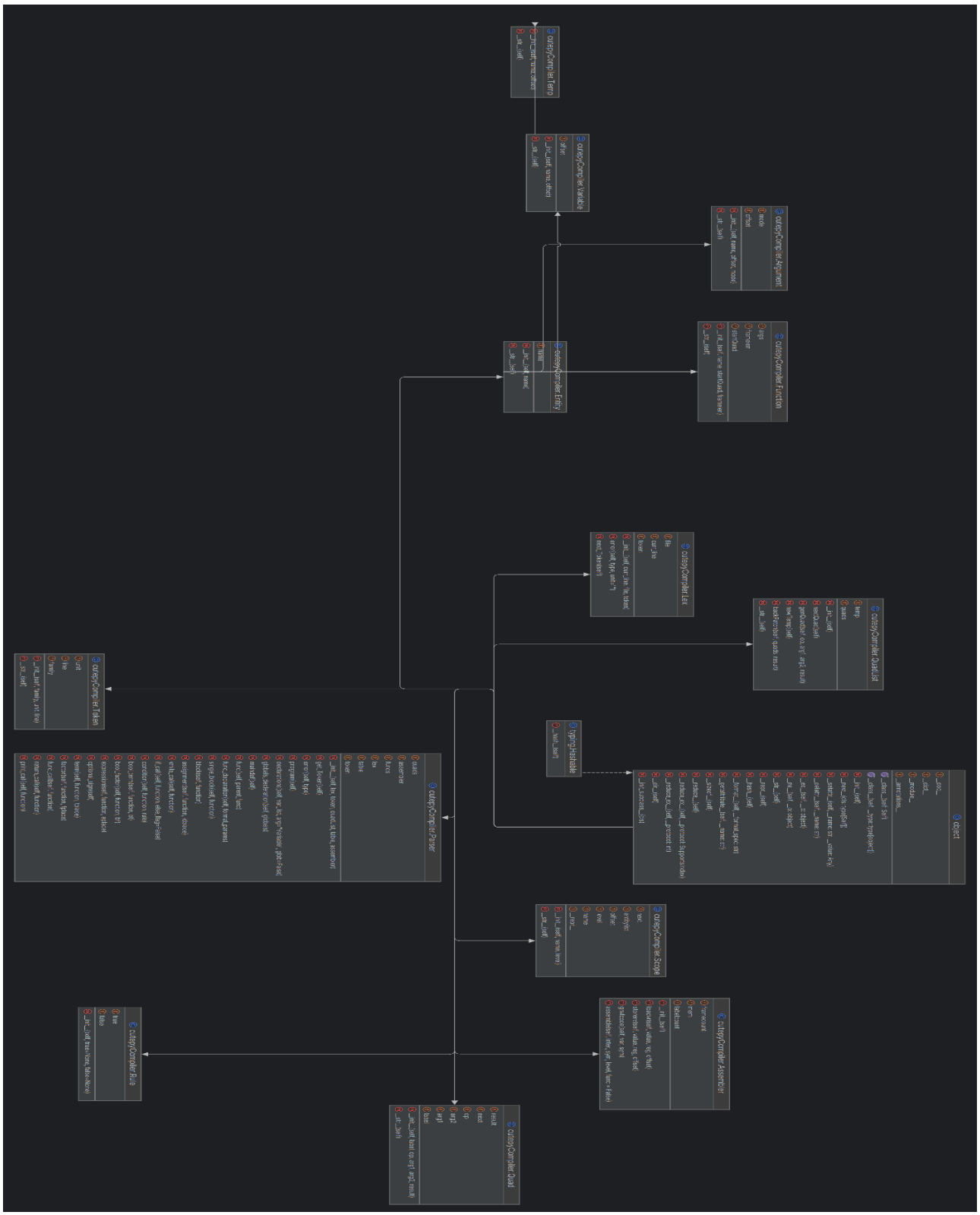
ΠΑΝΑΓΙΩΤΗΣ ΟΔΥΣΣΕΑΣ ΒΑΡΕΛΗΣ,(ΑΜ: 3388)

ΖΗΣΗΣ ΨΑΛΙΔΑΣ, (ΑΜ: 3369)

ΣΥΝΟΠΤΙΚΕΣ ΠΛΗΡΟΦΟΡΙΕΣ ΥΛΟΠΟΙΗΣΗΣ

- Η εργασία υλοποιήθηκε με Python 3.12 σε περιβάλλον Windows 10, σε συνδυασμό με τη χρήση εργαλείων για version control (Github)
- Ακολουθήθηκε αντικειμενοστραφής και αναδρομικός σχεδιασμός. Συγκεκριμένα ο Συντακτικός αναλυτής παίρνει αντικείμενα τύπου Token από τον Λεκτικό Αναλυτή σε πρώτη φάση ενώ παράλληλα σε σημεία που απαιτείται δημιουργούνται αντικείμενα τύπου Quad τα οποία αποθηκεύονται στο λεξικό τύπου QuadList για την υλοποίηση του ενδιάμεσου κώδικα και αντικείμενα τύπου Entity(Variable, Function etc..) τα οποία αποθηκεύονται σε αντικείμενα τύπου Scope για την υλοποίηση του πίνακα συμβόλων. Τέλος, καλείται ο assembler τύπου Assemble, ο οποίος αναλαμβάνει να παράξει τον Τελικό κώδικα πριν το κλείσιμο των Scopes.
- Όλοι οι αναλυτές έχουν δοκιμαστεί με το αρχείο cpy test.cpy και έχει επιβεβαιωθεί η λειτουργία τους, πέρα από την παραγωγή τελικού κώδικα ή οποία δεν γίνεται 100% σωστά καθώς δεν υλοποιήθηκαν η μετάφραση σε τελικό κώδικα των εντολών return καθώς και κάποιες από τις άλλες λειτουργίες είναι ημιτελής.

ΔΙΑΓΡΑΜΜΑΤΑ ΚΛΑΣΕΩΝ



ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ

Λεκτικός Αναλυτής

Ο λεκτικός αναλυτής υλοποιήθηκε στην κλάση Lex και λειτουργεί ως εξής:

- Διαβάζει το αρχείο byte ανά byte
- Αν διαβάζει λεκτική μονάδα η οποία είναι αποδεκτή από την γλώσσα cutePy την επιστρέφει ως αντικείμενο Token, με συμπληρωμένα πεδία την κατηγορία της λεκτικής μονάδας, την ίδια την λεκτική μονάδα και την γραμμή στην οποία βρέθηκε.
- Σε αυτήν την φάση γίνεται έλεγχος για λεκτικά σφάλματα, όπως το μέγεθος ενός αριθμού κ.α.

Συντακτικός Αναλυτής

Ο συντακτικός αναλυτής υλοποιήθηκε στην κλάση Parser και λειτουργεί ως εξής:

- Χρησιμοποιεί παίρνει τα αντικείμενα Tokens από τον Λεκτικό με την μέθοδο get_Token και πραγματοποιεί την ανάλυση του χρησιμοποιώντας αναδρομική κατάβαση.
- Ξεκινώντας από την μέθοδο program, αρχικά βλέπει για δηλώσεις καθολικών μεταβλητών, συναρτήσεων και δήλωση κύριας μεθόδου και την σειρά στην οποία δηλώθηκαν. Όσο βλέπει δήλωση καθολικής μεταβλητής καλείται η declarations, ενώ όσο βλέπει συναρτήσεις καλείται η func(αντιστοίχως η maindef για την κύρια συνάρτηση).
- Με τη σειρά της καλεί πάλι τις μεθόδους για δηλώσεις μεταβλητών, μεθόδων κ.α. και στην συνέχεια μεταβαίνει στην block, η οποία είναι η μέθοδος στην οποία ελέγχονται τα διάφορα statements σε ένα μπλοκ και σε αυτήν θα επιστρέφουμε αναδρομικά για όσα εμφωλευμένα μπλοκ έχουμε.
- Συνοπτικά, με αναδρομική κατάβαση σχεδιάστηκαν και οι μεθόδοι που ελέγχουμε για expressios, conditions, statements, κ.α.
- Επιπλέον, σε αυτήν την φάση γίνεται έλεγχος για λάθος στην σύνταξη ενός προγράμματος, όπως μη αναμενόμενων λεκτικών μονάδων σε διάφορα σημεία κ.α.

ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ

Ενδιάμεσος Κώδικας

Η μετατροπή του κώδικα σε ενδιάμεσο η παραγωγή των τετράδων υλοποιήθηκε ως εξής:

- Υλοποιήθηκαν μεθόδοι στην κλάση QuadList, η οποία παράγει αντικείμενα τύπου Quad, στα οποία αποθηκεύεται ο τελεστής/πράξη, το πρώτο τελούμενο, το δεύτερο και το αποτέλεσμα.
- Με την μέθοδο genQuad παράγουμε σε διάφορα σημεία του συντακτικού αναλυτή κατά την συντακτική ανάλυση τα Quad, στα οποία συμπληρώνουμε και τον αριθμό της τετράδας(label). Κατά την παραγωγή τους, αποθηκεύονται στο λεξικό quads του αντικειμένου QuadList.
- Με την μέθοδο newTemp παράγουμε προσωρινά τελούμενα, τα οποία τα χρησιμοποιούμε κατά την παραγωγή τετράδων σε πράξεις, αναθέσεις ή conditions και όπου αλλού χρειάζεται η «αποθήκευση» ενδιάμεσου αποτελέσματος.
- Με την backpatch συμπληρώνουμε τις τετράδες όπου δεν έχει συμπληρωθεί το label κατά την παραγωγή τετράδων σε conditional statements.
- Με τα αντικείμενα Rule «κρατάμε» τις τετράδες που θα παράγονται για false labels, και αυτές που θα είναι για true labels, τις οποίες τετράδες στην συνέχεια τις συμπληρώνουμε με τα κατάλληλα labels με την χρήση της backpatch

ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ

Πίνακας Συμβόλων

Για την παραγωγή του πίνακα συμβόλων ακολουθήθηκαν τα εξής βήματα:

- Δημιουργία αντικειμένων τύπου Entity(Variable, Function, Argument, Temp) τα οποία θα αντιπροσωπεύουν τα σύμβολα μας και αποθηκεύουμε τις απαραίτητες πληροφορίες για το κάθε Entity, όπως όνομα, offset κ.α.
- Δημιουργία αντικειμένων τύπου Scope, τα οποία αντιπροσωπεύουν το επίπεδο το οποίο βλέπει το κάθε function και τα σύμβολα που περιέχει. Σε κάθε Scope αποθηκεύουμε τα Entities που περιέχει στο λεξικο entitylist, το επίπεδο στο πρόγραμμα, το offset του και το επόμενο.
- Κατά την διάρκεια της συντακτικής ανάλυσης, δημιουργούνται τα ανάλογα Entities όταν δηλώνονται μέθοδοι και μεταβλητές, τα οποία χάριν της αναδρομικής κατάβασης δημιουργούνται στα κατάλληλα Scope και σε διαφορετικά επίπεδα αν έχουμε εμφωλευμένες συναρτήσεις, καθώς το κάθε Scope αποθηκέυεται σε έναν πίνακα table.
- Τέλος, αφού επιστρέψουμε στο σημείο που δημιουργήθηκε το Scope και αφού σίγουρα θα έχουμε τελειώσει με τυχόν δηλώσεις συμβόλων μέσα σε αυτό, συμπληρώνουμε πληροφορίες για την αρχική εκτελούμενη τετράδα και το «μήκος»(offset) του Scope στο αντίστοιχο Entity στο προηγούμενο επίπεδο και το κλείνουμε.

ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ

Τελικός Κώδικας

Για την παραγωγή του τελικού κώδικα:

- Χρησιμοποιούμε τις μεθόδους της κλάσης Assembler, και συγκεκριμένα την μέθοδο assemble πριν το κλείσιμο ενός Scope για να παράξουμε τελικό κώδικα για το Scope αυτό.
- Στην μέθοδο assemble βλέπουμε στις αποθηκευμένες τετράδες τις εντολές και αναλόγως παράγουμε την αντίστοιχη εντολή σε κώδικα assembly.
- Για αριθμητικές και λογικές παραστάσεις, χρησιμοποιώντας την μέθοδο gnvlcode, βλέπουμε αν το τελούμενο είναι μέρος του τωρινού Scope, του γονέα στον πίνακα συμβόλων ή καθολική μεταβλητή. Ανάλογα την περίπτωση, καλείται με τις κατάλληλες παραμέτρους η μέθοδος loadvr, η οποία παράγει εντολή η οποία φορτώνει σε καταχωρητή το δοσμένο δεδομένο. Τέλος, παράγουμε την κατάλληλη εντολή σε assembly, ανάλογα με την αριθμητική ή λογική παράσταση.
- Στις λογικές παραστάσεις, πέρα από τις εντολές παράγουμε και τα labels στα οποία θα γίνουν jump με τις εντολές διακλαδώσεων.
- Για την παραγωγή εντολών αλμάτων αρκεί να παράξουμε την εντολή στο label και αν το label είναι από εντολή λογικής διακλάδωσης τότε να παράξουμε το label.
- Για την αρχή και τέλος των block, παράγουμε τις εντολές όπου φορτώνουμε και αποθηκεύουμε την διεύθυνση επιστροφής. Συγκεκριμένα στο begin block λείπουν οι παραγωγές εντολών για την ανάθεση χώρου στην στοίβα για τις παραμέτρους.
- Για τις εντολές εισόδου και εξόδου παράγουμε τις εντολές για την φόρτωση στους καταχωρητές a1 και a7 τις κατάλληλες παραμέτρους, αναλόγως την περίπτωση, καθως και του ecall.
- Για κλήσεις συναρτήσεων παράγουμε τις εντολές για τις οποίες μετακινούμε τον δικό μας frame pointer(s7) κατά framelen, ανάλογα με το offset του κάθε Scope, και την εντολή διακλάδωσης jal με το label.
- Τέλος, για παραμέτρους χρησιμοποιούμε την gnvlcode για να εντοπίσουμε και να ταυτοποιήσουμε τον τύπο της, και να τις φορτώσουμε σε καταχωρητές με την loadvr.