Σχεδιασμός και Υλοποίηση του Γραφικού Τμήματος του Λαβύρινθου

Χρήστος Γεωργής

Τμήμα Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

Περίληψη

Η εργασία αυτή αναφέρεται στα σημαντικότερα θέματα που προέκυψαν κατά την ανάλυση και υλοποίηση του γραφικού τμήματος του Λαβύρινθου: παρουσιάζονται οι επιλογές που έγιναν σε διάφορα σχεδιαστικά και αλγοριθμικά ζητήματα και αναπτύσσεται ο τρόπος με τον οποίο υλοποιήθηκαν οι αλγόριθμοι που επιλέχτηκαν.

Ο Λαβύρινθος είναι ένας πυρήνας γενικής χρήσης για την ανάπτυξη γραφικών εργαλείων και εργαλείων CAD. Παρέχει ένα σύνολο από μηχανισμούς σχεδίασης γραφικών αντικειμένων και διασύνδεσης αυτών σε ιεραρχικά δίκτυα κυττάρων. Το γραφικό τμήμα του είναι υπεύθυνο για τον χειρισμό των συσκευών εισόδου και την επικοινωνία του συστήματος με τον χρήστη, η οποία περιλαμβάνει και την απεικόνιση στην οθόνη των γραφικών αντικειμένων του.

Στην εργασία αυτή αναπτύσσονται οι αποφάσεις που πάρθηκαν ώστε το γραφικό τμήμα του Λαβύρινθου να έχει γρήγορους χρόνους απόκρισης στις εντολές που δέχεται από τους χρήστες του.

Παρουσιάζονται ακόμα οι μηχανισμοί επιλογής και επανασχεδίασης του συστήματος. Ο χρήστης μπορεί να επιλέγει κύτταρα στην οθόνη με σκοπό να μεταβάλλει τα χαρακτηριστικά τους. Μετά από κάθε μεταβολή ένα ή περισσότερα κύτταρα χρειάζεται να επανασχεδιαστούν. Οι δύο παραπάνω μηχανισμοί στηρίζονται στην ύπαρξη ορθογωνίων περιγραμμάτων (bounding boxes) για τα κύτταρα του συστήματος.

Ακόμα αναλύεται ο τρόπος που ο Λαβύρινθος υποστηρίζει πολλαπλά χρώματα, πολλαπλά είδη γραμμών, πολλαπλά είδη γραφής, και πολύπλοκες σχεδιαστικές οντότητες (widgets), καθώς και ο μηχανισμός εκτύπωσης μέσω της δημιουργίας αρχείων PostScript.

Τέλος, στην εργασία αυτή παρουσιάζεται το σύνολο των μεταβολών που χρειάζεται να γίνουν ώστε ο Λαβύρινθος να χειρίζεται επικαλυπτόμενα κύτταρα, δηλαδή να δίνει στον χρήστη τη ψευδαίσθηση ότι μερικά αντικείμενα βρίσκονται πάνω από άλλα. Αυτή η ψευδαίσθηση επηρεάζει τόσο την επιλογή όσο και την επανασχεδίαση των κυττάρων ενός σχεδίου. Κατά την επιλογή πρέπει να επιλέγεται το "πάνω" κύτταρο, όταν υπάρχουν περισσότερα από ένα υποψήφια για επιλογή κύτταρα.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου Μανόλη Κατεβαίνη, για τις πολύτιμες συμβουλές του, για την ενθάρρυνση και την συμπαράστασή του. Χωρίς την βοήθειά του αυτή η εργασία δεν θα είχε επιτευχθεί.

Ακόμα θα ήθελα να ευχαριστήσω τα μέλη της ερευνητικής ομάδας του Λαβύρινθου για τις προτάσεις και τις υποδείξεις τους, κατά τις τακτικές εβδομαδιαίες συναντήσεις μας. Ιδιαίτερα ευχαριστώ τον Τάσο Σορίλο που με μύησε στον "μαγικό κόσμο" του Λαβύρινθου.

Θα ήθελα ακόμα να ευχαριστήσω τους Μαρία Καραβασίλη, Κώστα Κούτσικο, και Frank Jansen, για την πολύτιμη συμβολή τους στην υλοποίηση μέρους της εργασίας αυτής. Ευχαριστώ ακόμα τους Γιώργο Δρεττάκη και Παναγιώτη Καλογεράκη που διάβασαν την εργασία αυτή και μου έδωσαν πολύτιμες συμβουλές για την διόρθωσή της.

Τέλος θα ήθελα να ευχαριστήσω το Ινστιτούτο Πληροφορικής του ΙΤΕ για την οικονομική ενίσχυση και το εργασιακό περιβάλλον που μου παρείχε κατά την διάρκεια των μεταπτυχιακών σπουδών μου.

Περιεχόμενα

Εισαγωγή

Η εργασία αυτή εκπονήθηκε ως μέρος του σχεδιαστικού συστήματος Λαβύρινθος, το οποίο αναπτύσσεται στα πλαίσια του ερευνητικού προγράμματος ITHACA στο Ινστιτούτο Πληροφορικής του ΙΤΕ. Συγκεκριμένα η εργασία αυτή ασχολείται με το γραφικό μέρος του Λαβύρινθου.

Ενα σχεδιαστικό σύστημα είναι μια εφαρμογή με την οποία ο χρήστης – σχεδιαστής μπορεί να δημιουργεί και να μεταβάλλει γραφικά σχέδια μέσω του υπολογιστή. Ο χρήστης ενός τέτοιου συστήματος, συχνά, δεν αρκείται στο να τοποθετεί γραφικά αντικείμενα σε συγκεκριμένες θέσεις για να συνθέσει το σχέδιό του. Πολλές φορές χρειάζεται να ορίζει σχέσεις, κατά το πλείστον γεωμετρικές, μεταξύ των γραφικών αντικειμένων του σχεδίου. Ως παράδειγμα μιας τέτοιας γεωμετρικής σχέσης μπορούμε να φανταστούμε ένα σχέδιο που αποτελείται από δύο ομόκεντρους κύκλους. Ο χρήστης γνωρίζοντας ότι οι κύκλοι είναι ομόκεντροι, τους σχεδιάζει ώστε να έχουν το κέντρο τους στο ίδιο σημείο. Πολλά σχεδιαστικά συστήματα, όπως το AutoCAD [1], το FrameMaker [4], και το xfig [17], δεν επιτρέπουν στον χρήστη να ορίζει τέτοιες σχέσεις, μεταξύ των γραφικών αντικειμένων του σχεδίου. Σε ένα τέτοιο σύστημα, αν ο χρήστης αλλάξει την θέση σε ένα από τους δύο κύκλους, ο άλλος δεν θα ακολουθήσει αυτόματα ώστε αυτοί να παραμείνουν ομόκεντροι.

Μια σημαντική βοήθεια που θέλουμε να παρέχει ένα σχεδιαστικό σύστημα είναι να επιτρέπει στον χρήστη να εκφράζει σχέσεις (ή περιορισμούς) μεταξύ των γραφικών αντικειμένων ενός σχεδίου, και αυτές να διατηρούνται αυτόματα από το σύστημα. Ετσι, ο χρήστης αποκτά την δυνατότητα να εισάγει στο σχέδιό του περισσότερη πληροφορία, χρήσιμη τόσο για την σχεδίαση όσο και για την ερμηνεία του σχεδίου. Στο παραπάνω παράδειγμά, ένα σύστημα που "γνωρίζει" την σχέση που συνδέει τα κέντρα των δύο κύκλων, θα μετακινούσε αυτόματα τον δεύτερο κύκλο, όταν μετακινηθεί ο πρώτος. Επίσης, εάν κάποιο πρόγραμμα επιθυμεί να ερμηνεύσει κάθε γραμμή και κύκλο π.χ. σαν πεζοδρόμιο ενός δρόμου, θα του είναι ευκολότερο να καταλάβει ότι οι δύο αυτοί ομόκεντροι κύκλοι είναι δύο πεζοδρόμια ενός κυκλικού δρόμου, αφού η αλληλοσυσχέτιση τους υπάρχει ρητά αναπαρησταμένη στο σύστημα.

Ο Λαβύρινθος είναι ένα σχεδιαστικό σύστημα στο οποίο ο χρήστης έχει την δυνατότητα να ορίζει σχέσεις (ή περιορισμούς) μεταξύ των γραφικών αντικειμένων (κυττάρων) ενός σχεδίου. Στην πραγματικότητα είναι ένας πυρήνας γενικής χρήσης για ανάπτυξη γραφικών εργαλείων και εργαλείων CAD (εργαλεία σχεδίασης με την βοήθεια υπολογιστή), ο οποίος παρέχει μηχανισμούς σχεδίασης γραφικών αντικειμένων και διασύνδεσης τους σε ιεραχικά δίκτυα παραμετρικών κυττάρων. Ανάλογα σχεδιαστικά συστήματα είναι το ThingLab [2] και το Juno [9]. Ο Λαβύρινθος φιλοδοξεί να υποκαταστήσει και να βελτιώσει τις υπηρεσίες που προσφέρουν υπάρχοντα σχεδιαστικά συστήματα. Κύριο πλεονέκτημα του απέναντι στα συστήματα αυτά, είναι το γενονός ότι είναι μικρότερος σε μέγεθος, γρηγορότερος και απλούστερος από αυτά.

Το Γραφικό Μέρος του Λαβύρινθου

Ο Λαβύρινθος παρέχει ένα γραφικό editor, με τον οποίο ο χρήστης μπορεί να χειρίζεται τα κύτταρα που απαρτίζουν τα σχέδιά του. Τα κύτταρα και οι σχέσεις μεταξύ τους αναπαριστώνται στις εσωτερικές δομές του Λαβύρινθου. Ο χρήστης, εκτός από το να δημιουργεί, μπορεί να μεταβάλλει κύτταρα είτε οριζοντας νέες σχέσεις μεταξύ των συνιστώντων κυττάρων τους, είτε μεταβάλλοντας τις ήδη υπάρχουσες, είτε προσθέτοντας νέα συνιστώντα κύτταρα.

Στην εργασία αυτή παρουσιάζουμε τα σημαντικότερα θέματα τα οποία προέκυψαν κατά την ανάλυση και υλοποίηση του γραφικού μέρους του Λαβύρινθου. Συγκεκριμένα παρουσιάζονται :

- το τμήμα της επικοινωνίας μεταξύ του χρήστη και του γραφικού editor του συστήματος
- οι μηχανισμοί επιλογής και επανασχεδίασης των κυττάρων ενός σχεδίου
- οι μηχανισμοί υποστήριξης πολλαπλών σχεδιαστικών χαρακτηριστικών (χρώμα, είδη γραμμών κ.α.) και πολύπλοκων σχεδιαστικών αντικειμένων (γραφικές οντότητες – widgets).
- ο μηχανισμός για την δημιουργία κώδικα PostScript

και αναπτύσσονται οι αλλαγές που χρειάζεται να γίνουν στους παραπάνω μηχανισμούς για να μπορεί ο Λαβύρινθος να χειρίζεται επικαλυπτόμενα κύτταρα.

Γραφική Επικοινωνία

Η γραφική επικοινωνία χρήστη – συστήματος είναι ένα από τα σημαντικότερα μέρη του γραφικού editor του Λαβύρινθου. Με το όρο γραφική επικοινωνία εννοούμε τον τρόπο με τον οποίο ένα σύστημα – όχι μόνο σχεδιαστικό – επιτρέπει στους χρήστες του να δρούν πάνω του μέσω των συσκευών εισόδου του υπολογιστή (πληκτρολόγιο, ποντίκι), και να δέχονται τις αντιδράσεις του μέσω γραφικών πράξεων στην συσκευή εξόδου (οθόνη). Ο χρήστης βλέπει το σχέδιό του σε μία ή περισσότερες περιοχές τις οθόνης, από μία ή περισσότερες όψεις, και ενεργεί πάνω του μέσω του ποντικιού (συσκευή ένδειξης) ή του πληκτρολογίου, δίνοντας τις κατάλληλες εντολές στο σύστημα.

Στην επικοινωνία αυτή έχει μεγάλη σημασία τόσο η ταχύτητα, με την οποία οι ενέργειες του χρήστη γίνονται αντιληπτές από το σύστημα, όσο και εκείνη με την οποία το σύστημα αντιδρά σχεδιάζοντας κάτι στην οθόνη. Η διατήρηση των χρόνων απόκρισης (το άθροισμα των παραπάνω δύο χρόνων) σε χαμηλά επίπεδα (από μερικά δέκατα του δευτερολέπτου ως και ένα δευτερόλεπτο), είναι ένα από τα σημαντικά ζητήματα που αντιμετωπίζει ο σχεδιαστής της επικοινωνίας χρήστη – συστήματος. Για τον λόγο αυτό χρειάζεται ιδιαίτερη προσοχή κατά τον σχεδιασμό της.

Μεγάλη σημασία, ακόμα, έχει η φιλικότητα με την οποία ένα σύστημα αντιμετωπίζει τον χρήστη. Το σύστημα πρέπει να επιτρέπει στον χρήστη να σχεδιάζει και να χειρίζεται με απλό και κατανοητό τρόπο τα γραφικά αντικείμενα του σχεδίου. Ακόμα πρέπει είναι ανεκτικό στα λάθη του χρήστη και να μην τον φέρνει προ εκλήξεων. Πολλά σχεδιαστικά συστήματα παρέχουν, για τον λόγο αυτό, παλέττες με γραφικά αντικείμενα, από τις οποίες ο χρήστης μπορεί εύκολα να διαλέξει αυτά που θα συνθέσουν το σχέδιό του. Ακόμα παρέχουν μενού εντολών, με τα οποία ο χρήστης μπορεί να επιλέγει εντολές για τον χειρισμό των παραπάνω γραφικών αντικείμενων, για την φύλαξη (και φόρτωση) του σχεδίου σε (και από) αρχείο, καθώς και για άλλες χρήσιμες λειτουργίες, ενώ επισημαίνουν σημαντικά λάθη του χρήστη με την εμφάνιση μιας επιγραφής που αναγράφει το είδος του λάθους.

Επιλογή, Επανασχεδίαση

Οι μηχανισμοί σχεδίασης και επιλογής των κυττάρων αποτελούν ένα σημαντικό μέρος της εργασίας αυτής. Ο χρήστης μπορεί να επιλέγει με το ποντίκι, το κύτταρο πάνω στο οποίο πρόκειται να επεμβεί και με κατάλληλες εντολές να το μεταβάλλει. Η επιλογή ενός κυττάρου είναι ένα δύσκολο πρόβλημα που απαιτεί από το σύστημα να γνωρίζει με ακρίβεια κάθε σημείο της γραφικής αναπαράστασης του κυττάρου και να το συγκρίνει με το σημείο επιλογής. Σε άλλη περίπτωση, το σύστημα χρειάζεται να γνωρίζει την απόσταση του κυττάρου από το σημείο επιλογής. Και στις δύο περιπτώσεις το υπολογιστικό κόστος της επιλογής μπορεί να είναι μεγάλο. Για την λύση του προβλήματος αυτού ο Λαβύρινθος χρησιμοποιεί ένα μηχανισμό εύρεσης και διατήρισης των ορθογώνιων περιγραμμάτων (bounding boxes) των κυττάρων ενός σχεδίου.

Η μεταβολή ενός κυττάρου έχει σαν αποτέλεσμα τον επανυπολογισμό των συστατικών του, ο οποίος μπορεί να προκαλέσει αλυσιδωτά τον επανυπολογισμό άλλων κυττάρων με τα οποία αυτό συνδέεται. Με την ολοκλήρωση αυτής της αλυσιδωτής διαδικασίας, τα κύτταρα που μεταβλήθηκαν πρέπει να επανασχεδιαστούν. Ο επανασχεδιασμός σε ένα γραφικό σύστημα μπορεί να αποβεί ιδιαίτερα δαπανηρός από άποψη χρόνου, ειδικά εάν το σχέδιο αποτελείται από πολλα κύτταρα με χρονοβόρες διαδικασίες σχεδίασης, όπως κύκλοι, τόξα, διακεκομμένα ευθύγραμμα τμήματα κ.α.

Βέβαια, οι σύγχρονοι σταθμοί εργασίας είναι συνήθως εφοδιασμένοι με γραφικούς επεξεργαστές και πρόσθετη μνήμη για γραφικά. Ομως όσο μεγαλώνουν οι δυνατότητες που μας προσφέρουν τα σύγχρονα συστήματα, τόσο μεγαλώνουν οι απαιτήσεις των χρηστών για περισσότερα και καλύτερα γραφικά, με συνέπεια να αυξάνεται η πολυπλοκότητα των σχεδίων.

Ακόμα, η χρήση γραφικών προγραμμάτων μέσω δικτύου, όπως συμβαίνει στο περιβάλλον γραφικών X που χρησιμοποιήσαμε στην εργασία αυτή, τονίζει αυτές τις καθυστερήσεις ακόμη περισσότερο, αφού μαζί με τον χρόνο για τον σχεδιασμό, προστίθεται και ο χρόνος μεταφοράς των εντολών σχεδίασης από το δίκτυο.

Για τους παραπάνω λόγους η διαδικασία επανασχεδίασης είναι σημαντική και χρειάζεται ιδιαίτερη προσοχή κατά την ανάλυση και την υλοποίηση ενός συστήματος ώστε να γίνονται οι λιγότερες δυνατές επανασχεδιάσεις και να αποφεύγονται οι περιττές σχεδιαστικές πράξεις.

Ομορφα Σχέδια, Εκτύπωση

Ο Λαβύρινθος παρέχει στον χρήστη διάφορα γραφικά αντικείμενα. Υπάρχουν κύτταρα, όπως το "κείμενο", όπου ο χρήστης μπορεί να καθορίσει το χρώμα, την οικογένεια γραφής και το μέγεθός του. Σε άλλα κύτταρα, όπως ο "κύκλος" και το "ευθύγραμμο τμήμα", μπορεί να καθορίσει το χρώμα, το πάχος και το είδος (συνεχής, διακεκομένη) των γραμμών τους. Ενώ η υλοποίησή τους, θα έπρεπε να είναι αρκετά απλή, στην πραγματικότητα παρουσιάζει αρκετές δυσκολίες που άπτονται του τρόπου σχεδίασης που παρέχει το γραφικό περιβάλον X σε συνδυασμό με τον μηχανισμό επανασχεδίασης του Λαβύρινθου.

Το γραφικό περιβάλλον X παρέχει, εκτός από απλές σχεδιαστικές λειτουργίες (σχεδίαση γραμμών, κύκλων κ.α.) και σύνθετες σχεδιαστικές οντότητες (widgets) των οποίων τόσο την γραφική αναπαράσταση, όσο και την λειτουργικότητα, ο Λαβύρινθος μπορεί να χρησιμοποιήσει. Στην εργασία αυτή παρουσιάζουμε τον τρόπο με τον οποίο το σύστημα θα μπορέσει να ενσωματώσει τέτοιες οντότητες ως κύτταρα.

Ο Λαβύρινθος για να επιτρέπει στον χρήστη να εκτυπώνει το σχέδιό του στον εκτυπωτή, δημιουργεί ένα αρχείο που περιέχει το σχέδιο σε κώδικα PostScript [6]. Με τον τρόπο αυτό το σχέδιο μπορεί να εκτυπωθεί σε ένα εκτυπωτή ή να συμπεριληφθεί ως εικόνα σε ένα κείμενο. Σχετικά με την παραγωγή κώδικα PostScript, το ενδιαφέρον βρίσκεται στο γεγονός ότι οι διαστάσεις και η ποιότητα ενός κειμένου είναι διαφορετικές στον εκτυπωτή απ’ ότι στην οθόνη.

Υποστήριξη Επικαλυπτόμενων Κυττάρων

Ο Λαβύρινθος πρέπει να υποστηρίζει την ύπαρξη και τον χειρισμό επικαλυπτόμενων κυττάρων. Επικαλυπόμενα κύτταρα έχουμε στην περίπτωση που, μέσα στο σχέδιο που δημιουργεί ο χρήστης, υπάρχουν κύτταρα των οποίων ένα ή περισσότερα μέρη – ή ακόμα και ολόκληρα κύτταρα, βρίσκονται "σκεπασμένα" από άλλα κύτταρα.

Η υποστήριξη (σχεδιασμός και επιλογή) των κυττάρων που επικαλύπτονται, από ένα σχεδιαστικό σύστημα, είναι ένα βασικό του στοιχείο, γιατί συμβάλει τόσο στην σωστή εμφάνιση του σχεδίου, όσο και στην κατανόηση της διάταξης – στον χώρο – των κυττάρων του αποτελούν το σχέδιο. Επιπλέον εισάγει ενδιαφέροντα θέματα για ανάλυση, όπως η σειρά επανασχεδίασης των κυττάρων. Ο μηχανισμός επιλογής πρέπει να λαμβάνει υπ’όψη του την πιθανή ύπαρξη κυττάρων που επικαλύπτονται ολόκληρα από άλλα και να προσφέρει την δυνατότητα για αυτά να επιλέγονται.

Η οργάνωση της εργασίας

Στο πρώτο κεφάλαιο παρουσιάζεται ο Λαβύρινθος και ειδικά ο κορμός γραφικής επικοινωνίας του, καθώς και το περιβάλον γραφικών X που χρησιμοποιήθηκε. Στο επόμενο κεφάλαιο παρουσιάζεται ο μηχανισμός των ορθογώνιων περιγραμμάτων (bounding boxes), πάνω στον οποίο είναι στηριγμένοι οι μηχανισμοί επιλογής και επανασχεδίασης του Λαβυρίνθου και περιγράφεται ο μηχανισμός επιλογής των κυττάρων του συστήματος.

Στο τρίτο κεφάλαιο γίνεται παρουσίαση του μηχανισμού σχεδίασης των γραφικών αντικειμένων (κυττάρων) του συστήματος τόσο στην οθόνη όσο και στον εκτυπωτή (δημιουργία κώδικα PostScript). Εδώ περιγράφεται η βελτίωση του τρόπου επανασχεδίασης στην οθόνη χρησιμοποιώντας τα ορθογώνια περιγράμματα. Ακόμα περιγράφονται οι αλλαγές που πρέπει να γίνουν στο σύστημα ώστε οι οντότητες – widgets να ενσωματωθούν σε αυτό ως κύτταρα. Στο επόμενο κεφάλαιο παρουσιάζεται η υποστήριξη χρωμάτων, πολλαπλών ειδών γραμμών (διακεκομμένες, σε διαφορα πλάτη), καθώς και πολλαπλών ειδών γραφής (απλή, πλάγια, έντονη).

Στο πέμπτο κεφάλαιο παρουσιάζεται ο μηχανισμός υποστήριξης (σχεδιασμός και επιλογή) για κύτταρα που επικαλύπτονται, και στο τελευταίο κεφάλαιο παρουσιάζονται σκέψεις σχετικά με την βελτίωση του γραφικού τμήματος του Λαβύρινθου.

1. Ο Λαβύρινθος και το Γραφικό Περιβάλλον X

1.1 Τι είναι ο Λαβύρινθος

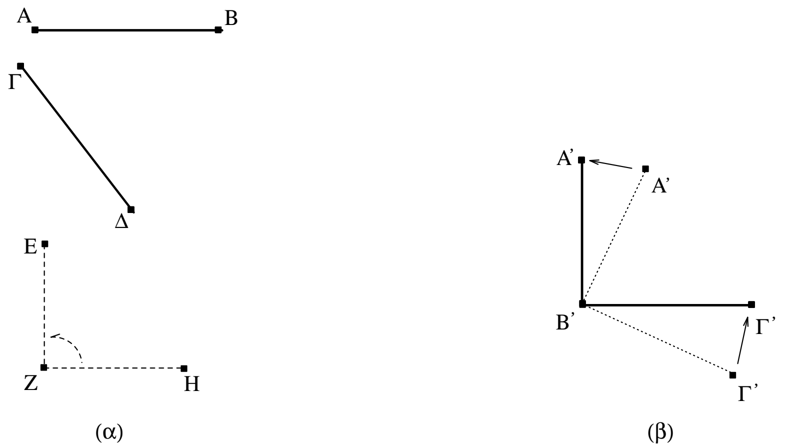
Οι γραφικές οντότητες στον Λαβύρινθο [7] ονομάζονται κύτταρα (cells) και έχουν παραμέτρους που ονομάζουμε πόρτες – επικοινωνίας – μέσω των οποίων μπορούν να καθορίζονται τα χαρακτηριστικά τους. Τα κύτταρα στον Λαβύρινθο είναι ιεραρχικά οργανωμένα σε πρωτογενή (primitives) και σύνθετα (composites): τα σύνθετα κύτταρα αποτελούνται από άλλα σύνθετα και πρωτογενή κύτταρα. Ακόμα, τα κύτταρα μπορεί να έχουν γραφική αναπαράσταση (ορατά) ή όχι (αόρατα).

Την ιεραρχική δομή που παρέχει ο Λαβύρινθος, δηλαδή την σύνθεση ενός κυττάρου από άλλα, την βρίσκουμε σε πολλές σχεδιαστικές εφορμογές (ηλεκτρονικά, μηχανολογικά, αρχιτεκτονικά σχέδια, κλπ). Παραδείγματος χάριν, για την σχεδίαση ψηφιακών κυκλωμάτων οι πύλες-NOR χρησιμοποιούνται για να συνθέσουν flip-flops, τα flip-flops για να συνθέσουν καταχωρητές, κ.ο.κ. Σαν ένα άλλο παράδειγμα, στις αρχιτεκτονικές εφαρμογές κομμάτια από ξύλο, γυαλί και μέταλλο χρησιμοποιούνται για να συνθέσουν παράθυρα και πόρτες, παράθυρα, πόρτες και τοίχοι χρησιμοποιούνται με την σειρά τους για την δημιουργία δωματίων, τα δωμάτια φτίαχνουν διαμερίσματα κ.ο.κ. Σήμερα, η ιεραρχική οργάνωση θεωρείται απαραίτητη για ένα εργαλείο CAD.

1.1.1 Παράδειγμα

Πριν μπούμε σε λεπτομέρειες, ας δούμε ένα παράδειγμα ορισμού ενός σύνθετου κυττάρου του οποίου τα επιμέρους συστατικά (συνιστώντα κύτταρα) συνδέονται μεταξύ τους με σχέσεις και περιορισμούς.

Εστω ότι θέλουμε να συνθέσουμε ένα κύτταρο από δύο "ευθύγραμμα τμήματα" (πρωτογενή ορατά κύτταρα) και μια "ορθή γωνία" (πρωτογενές αόρατο κύτταρο) και να το ονομάσουμε "ορατή ορθή γώνια". Στο σχήμα 1.1(α) φαίνονται τα "ευθύγραμμα τμήματα" AB και ΓΔ καθώς και η "ορθή γωνία" EZH που συμβολίζουμε με διακεκομμένη γραμμή για να είναι ορατή στο παράδειγμά μας. Οι πόρτες των παραπάνω κυττάρων είναι γεωμετρικά σημεία (αυτά που



Σχήμα 1.1: Παράδειγμα ορισμού σύνθετου κυττάρου

φαίνονται στο σχήμα).

Συνδέοντας τις πόρτες Α, Γ και Ζ μαζί, καθώς και την Β με την Ε και την Δ με την Η, ορίσαμε πλήρως τις σχέσεις μεταξύ των συστατικών του νέου κυττάρου. Ακόμα, ορίζουμε το νέο κύτταρο να έχει τρείς πόρτες, τις Α', Β', και Γ', τις οποίες συνδέουμε στις πόρτες Β, Α, και Δ, αντίστοιχα. Το νέο κύτταρο τώρα έχει στην πραγματικότητα δύο ευθύγραμμα τμήματα και μια σχέση μεταξύ τους, ότι αυτά πάντοτε θα σχηματίζουν μεταξύ τους ορθή γωνία και θα έχουν ένα ακραίο σημείο κοινό.

Το αποτέλεσμα όλης αυτής της διαδικασίας είναι ότι αν κάποιος θελήσει να μεταβάλλει την διεύθυνση μιάς πλευράς της "ορατής ορθής γώνιας", π.χ. αλλάζοντας την τιμή του σημείου Α', αυτό θα έχει ως συνέπεια να ακολουθήσει η αλλαγή της διεύθυνσης της άλλης πλευράς με τέτοιο τρόπο ώστε η περιεχόμενη γωνία να παραμείνει ορθή. Αυτό θα συμβεί όπως φαίνεται στο σχήμα 1.1(β) με την αλλαγή της θέσης του σημείου Γ'.

1.1.2 Πόρτες Κυττάρων

Οι πόρτες στα κύτταρα μοιάζουν με τα ορίσματα των διαδικασιών στις συμβατικές γλώσσες προγραμματισμού. Κάθε πόρτα ενός κυττάρου έχει μια τιμή, η οποία είτε έρχεται μέσα στο κύτταρο από το εξωτερικό περιβάλλον σαν είσοδος, είτε παράγεται μέσα στο κύτταρο ως έξοδος προς το εξωτερικό περιβάλλον, με σκοπό να χρησιμοποιηθεί από άλλα κύτταρα. Τις πόρτες αυτές τις ονομάζουμε πόρτες εισόδου και πόρτες εξόδου αντίστοιχα, ενώ υπάρχουν και πόρτες εισόδου–εξόδου, των οποίων οι τιμές άλλες φορές είναι είσοδοι και άλλοτε έξοδοι. Ο χρήστης μπορεί να μεταβάλλει τα χαρακτηριστικά ενός κυττάρου αλλάζοντας την τιμή των πορτών εισόδου του. Το κύτταρο με βάση τις εισόδους αυτές υπολογίζει τις τιμές εξόδου των πορτών του, ενώ ταυτόχρονα αλλάζει ενδεχομένως σχήμα, χρώμα, κατάσταση κλπ.

Οι πόρτες ενός κυττάρου, εκτός από γεωμετρικά σημεία, μπορεί να είναι και πραγματικοί αριθμοί, ή κείμενο. Στην έκδοση 3 (version 3) του συστήματος, ο χρήστης θα μπορεί να ορίζει, ο ίδιος, νέους τύπους τιμών για τις πόρτες των κυττάρων του συστήματος.

Οι πόρτες με τύπο γεωμετρικό σημείο παίρνουν ως τιμές ζευγάρια από αριθμούς (π.χ. <-10, 79.4>). Ο χρήστης δημιουργεί γραφικά αντικείμενα με γεωμετρικά σημεία εκφρασμένα σε συντεταγμένες κόσμου (world coordinates), ενώ το σύστημα τα σχεδιάζει στην οθόνη μετατρέποντας αυτές σε συντεταγμένες οθόνης (screen coordinates), όπως περιγράφεται από τους Foley και Van Dam [5].

1.1.3 Σύνθεση Κυττάρων

Για να ορίσουμε ένα σύνθετο κύτταρο πρέπει να φέρουμε μέσα σε αυτό και να ενώσουμε επιμέρους κύτταρα, ορίζοντας έτσι τις σχέσεις μεταξύ τους. Η "ένωση" επιμέρους κυττάρων σε ένα σύνθετο γίνεται συνδέοντας πόρτες του ιδίου τύπου. Η σύνδεση πορτών σημαίνει για τον Λαβύρινθο ταυτοποίηση των τιμών των πορτών αυτών. Η ταυτότητα στις τιμές των πορτών είναι η πλέον απλή σχέση που μπορούμε να ορίσουμε στον Λαβύρινθο ανάμεσα σε κύτταρα.

Περισσότερο σύνθετες σχέσεις ορίζουμε ανάμεσα σε κύτταρα, συνδέοντάς τα με άλλα, κατάλληλα κύτταρα. Στο παραπάνω παράδειγμα το πρωτογενές αόρατο κύτταρο "ορθή γωνία" εκφράζει μια γεωμετρική σχέση ανάμεσα σε τρία σημεία (πόρτες). Αντίστοιχα το σύνθετο κύτταρο "ορατή ορθή γώνια" είναι μια γεωμετρική σχέση ανάμεσα σε δύο ευθύγραμμα τμήματα. Στον Λαβύρινθο ο χρήστης μπορεί να ορίσει και άλλες σχέσεις εκτός από γεωμετρικές. Ενα παράδειγμα είναι ένα κύτταρο που έχει ως τιμές εισόδου δύο πραγματικούς αριθμούς και παράγει ως τιμή εξοδου τον μικρότερο από τους δύο. Στηριζόμενοι στο κύτταρο αυτό μπορούμε να τροφοδοτήσουμε σαν είσοδο σ’ ένα κύτταρο την μικρότερη τιμή από τις τιμές εξόδου ενός άλλου κυττάρου, δημιουργώντας έτσι μια σχέση μεταξύ τους.

Μια σχέση, που όπως αναφέραμε παραπάνω ορίζεται ως κύτταρο στον Λαβύρινθο, είναι στην ουσία σχέση μεταξύ των πορτών του κυττάρου που είναι είσοδοι και εκείνων που είναι έξοδοι κατά την διάρκεια του υπολογισμού του κυττάρου. Στο παραπάνω παράδειγμα δίνοντας μια νέα τιμή ως είσοδο στην πόρτα Α’ παρήχθηκε ως έξοδος η τιμή στην πόρτα Γ’, εκφράζοντας έτσι την σχέση ορθής γωνίας μεταξύ των πορτών (σημείων) του κυττάρου "ορατή ορθή γώνια".

Τα κύτταρα – πρωτογενή και σύνθετα – καθώς και οι πληροφορίες οι σχετικές με τις συνδέσεις μεταξύ τους αναπαρίστανται με δίκτυα, τα οποία συνιστούν τις εσωτερικές δομές του συστήματος. Ο χρήστης μπορεί να δίνει το δικό του νόημα στα δίκτυα αυτά, ορίζοντας διαδικασίες διάσχισής τους και να αξιοποιεί την σημασιολογία τους με τον δικό του τρόπο. Ετσι ο Λαβύρινθος ξεφεύγει από τα όρια ενός σχεδιαστικού συστήματος, παρέχοντας την δυνατότητα σημασιολογικής ερμηνείας των δικτύων που παράγει.

1.1.4 Μηχανισμός Επανυπολογισμού

Για να μπορεί ο Λαβύρινθος να διατηρεί συνεχώς όλες τις σχέσεις συνδέσεων που έχουν οριστεί, έχει ένα μηχανισμό επανυπολογισμού των κυττάρων.

Ο μηχανισμός αυτός στηρίζεται σε γεγονότα (event driven evaluator). Σε γενικές γραμμές λειτουργεί ως εξης: κάθε φορά που η τιμή μιάς πόρτας εξόδου ενός κυττάρου μεταβάλλεται, δημιουργείται ένα γεγονός στην πόρτα αυτή, καθώς και σε όλες τις πόρτες που είναι συνδεδεμένες με αυτήν. Για όλα τα κύτταρα που έχουν κάποιο γεγονός σε μία ή περισσότερες εισόδους, καλούμε αργότερα τη διαδικασία υπολογισμού τους.

Κάθε πρωτογενές κύτταρο έχει μια δική του διαδικασία υπολογισμού (evaluation function), η οποία με βάση τις τιμές των πορτών εισόδου παράγει τιμές εξόδου στις εξαρτώμενες πόρτες. Για τα σύνθετα κύτταρα, η διαδικασία υπολογισμού καλεί αναδρομικά τις διαδικασίες υπολογισμού των συνιστώντων κυττάρων που έχουν δεχτεί γεγονότα.

1.1.5 Αλλοι Μηχανισμοί

Αντίστοιχα με τον μηχανισμό υπολογισμού υπάρχουν και άλλοι μηχανισμοί, όπως επανασχεδίασης, δημιουργίας κώδικα Postscript κ.α.

Για την υποστήριξη των μηχανισμών αυτών κάθε πρωτογενές κύτταρο έχει διαδικασίες υπολογισμού, σχεδίασης, δημιουργίας κώδικα Postscript, εύρεσης της απόστασης του κυττάρου από δοθέν σημείο κ.α. γραμμένες σε πηγαίο κώδικα (source code), ενώ για τα σύνθετα οι μηχανισμοί λειτουργούν καλώντας αναδρομικά τις διαδικασίες των συνιστώντων κυττάρων.

1.1.6 Τρόποι Ορισμού Κυττάρων

Ο ορισμός των πρωτογενών κυττάρων γίνεται σε πηγαίο κώδικα (source code). Η διαδικασία αυτή μπορεί να είναι αρκετά επίπονη μιάς και ο χρήστης πρέπει να γνωρίζει αρκετές λεπτομέρειες του συστήματος. Ωστόσο, έχει γίνει προσπάθεια ο ορισμός των πρωτογενών κυττάρων να είναι τυποποιημένος, ώστε να απλοποιείται η διαδικασία αυτή. Ενας κατάλογος των πρωτογενών κυττάρων του συστήματος παρατίθεται στο παράρτημα Γ.

Ο ορισμός των σύνθετων κυττάρων – στην πραγματικότητα η σχεδίαση – γίνεται ορίζοντας συνδέσεις μεταξύ των συνιστώντων κυττάρων. Στον Λαβύρινθο η σχεδίαση μπορεί να γίνεται με τρείς τρόπους: διαλογικά με την βοήθεια του γραφικού editor, μέσω κλήσεων διαδικασιών (function calls) μέσα από άλλα προγράμματα και δημιουργώντας ή (και) μεταβάλλοντας τα αρχεία όπου φυλάσσονται οι ορισμοί των σύνθετων κυττάρων. Για την διαλογική επικοινωνία Λαβύρινθου – χρήστη θα μπούμε σε λεπτομέρειες στην παράγραφο 1.3.

Το σύστημα φυλάσσει τους ορισμούς των κυττάρων σε δομές που ονομάζουμε "cell definitions" και που περιέχουν όλες εκείνες τις πληροφορίες που είναι ανεξάρτητες από τις τιμές των πορτών του. Τέτοιες πληροφορίες είναι: οι τύποι και οι ονομασίες των πορτών του κυττάρου, από ποιά κύτταρα αποτελείται, πώς (δηλαδή με ποιές πόρτες) αυτά συνδέονται μεταξύ τους κ.α. Οι παραπάνω δομές περιγράφονται με λεπτομέρεια στο [15].

Κάθε φορά που ο χρήστης θέλει να χρησιμοποιήσει ένα ήδη ορισμένο κύτταρο, το σύστημα δημιουργεί ένα στιγμιότυπο του ("cell instance"). Η δομή που αναπαριστά το στιγμιότυπο ενός κυττάρου περιέχει τις πληροφορίες εκείνες που το διαφοροποιούν από τα άλλα στιγμιότυπα της ίδιας "κλάσης". Τέτοιες πληροφοριές είναι κυρίως οι τιμές που έχουν οι πόρτες του.

1.2 Περιβάλλον παραθύρων X

O Λαβύρινθος, όπως και κάθε γραφικό πρόγραμμα, πρέπει να μπορεί να κάνει χρήση των συσκευών εισόδου–εξόδου του υπολογιστή και να επικοινωνεί μέσω αυτών με τον χρήστη. Η επικοινωνία αυτή γίνεται συνήθως μέσα σε ένα ευρύτερο περιβάλλον παραθύρων (window environment) στο οποίο τρέχουν και άλλα γραφικά ή μη προγράμματα. Σε ένα περιβάλλον παραθύρων, ο χρήστης μπορεί να δημιουργήσει περιοχές στην οθόνη (παράθυρα) και να τις χρησιμοποιεί ως ανεξάρτητες οθόνες των οποίων τα χαρακτηριστικά μπορεί να μεταβάλλει κατά προτίμιση.

Συνήθως ένα τέτοιο περιβάλλον παρέχει μια βιβλιοθήκη διαδικασίων με τις οποίες ο προγραμματιστής μπορεί να δημιουργεί εφαρμογές που να συνεργάζονται με αυτό (library interface). Παλαιότερα η επικοινωνία του Λαβύρινθου με τον χρήστη στηρίζονταν στην βιβλιοθήκη γραφικών CGI και το περιβάλλον λειτουργίας του ήταν τα Suntools. Σήμερα, ο Λαβυρινθος χρησιμοποιεί την βιβλιοθήκη γραφικών Xlib με περιβαλλον λειτουργίας το X (X Window System) [14], ενώ ήδη χρησιμοποιείται επιπλέον και η βιβλιοθήκη των Motif Widgets [10, 11], για τις οποίες θα μιλήσουμε με περισσότερες λεπτομέρειες παρακάτω.

1.2.1 Γενικά

Το X είναι ένα περιβάλλον παραθύρων όπου μπορεί κανείς να τρέχει γραφικά προγράμματα από υπολογιστές που βρίσκονται διάσπαρτοι σε ένα δίκτυο, και των οποίων η γραφική έξοδος και η είσοδος από το πληκτρολόγιο μεταφέρονται μέσω του δικτύου αυτού.

Τα γραφικά προγράμματα που είναι γραμμένα για να συνεργάζονται με το X (X clients – εφαρμογές X) είναι ανεξάρτητα από τις συσκευές εισόδου – εξόδου που διαχειρίζονται (display device independent). Ετσι μπορούν να τρέχουν σε ένα υπολογιστή και να εμφανίζουν τα γραφικά τους στην οθόνη ενος άλλου, ενώ δέχονται την είσοδό τους από το αντίστοιχο πληκτρολόγιο ή ποντίκι.

Η ιδιότητα αυτή στηρίζεται στο γεγονός, ότι οι εφαρμογές X ζητούν την εξυπηρέτιση αιτήσεων για γραφικά από ένα πρόγραμμα "υπηρέτη" (X server) το οποίο είναι υπεύθυνο για τα γραφικά στην οθόνη ενός υπολογιστή. Το πρόγραμμα αυτό γνωρίζει τα χαρακτηριστικά του υλικού (hardware) του υπολογιστή στον οποίο τρέχει και αναλαμβάνει να ικανοποιεί τις αιτήσεις για γραφικά που έρχονται από τις διαφορετικές εφαρμογές που θέλουν να σχεδιάσουν στην οθόνη για την οποία είναι υπεύθυνο. Οι συσκευές για τις οποίες είναι υπεύθυνο είναι: το ποντίκι, το πληκτρολόγιο και η μία ή οι περισσότερες οθόνες που είναι συνδεδεμένες με τον υπολογιστή στον οποίο τρέχει.

Για την επικοινωνία αυτή ανάμεσα στον X server και τις γραφικές εφαρμογές X έχει οριστεί ένα πρωτόκολο επικοινωνίας (X protocol). Η επικοινωνία είναι ασύγχρονη και γίνεται προς τις δύο κατευθύνσεις. Συγκεκριμένα, ο X server πληροφορεί τις γραφικές εφαρμογές (X clients) για την θέση και την κατάσταση του ποντικιού (π.χ. πατημένο ή όχι), για το αν πατήθηκε κάποιο πλήκτρο στο πληκτρολόγιο, αλλά και για την κατάσταση της οθόνης, (δηλαδή αν άλλαξε η θέση ή το μέγεθος ενός παραθύρου, αν εμφανίστηκε κάποιο κρυμμένο μέρος του, κλπ). Παράλληλα, οι εφαρμογές X στέλνουν αιτήσεις για γραφικά στον X server (δηλαδή αιτήσεις εμφάνισης γραμμών, τόξων, κειμένου κλπ), και του ζητούν να τις ενημερώνει όποτε συμβαίνουν ορισμένα γεγονότα, όπως παραδείγματος χάριν η αλλαγή των διαστάσεων ενός παραθύρου.

1.2.2 Γιατί X;

Το κύριο πλεονέκτημα του περιβάλλοντος X, είναι ότι η παραπάνω επικοινωνία μπορεί να γίνεται και μέσω δικτύου. Για λόγους ταχύτητας, όταν οι εφαρμογές X και ο X server τρέχουν στον ίδιο υπολογιστή, μπορεί να επικοινωνούν μέσω κοινόχρηστης μνήμης, αντί μέσω Unix domain sockets. Το X πρωτόκολο σχεδιάστηκε να είναι ανεξάρτητο από το λειτουργικό σύστημα και την αρχιτεκτονική του υπολογιστή που τρέχει η εφαρμογή (ή ο server), αλλά και από την γλώσσα προγραμματισμού στην οποία γράφεται αυτή.

Ετσι μπορεί να υπάρχει μια οθόνη με την οποία να επικοινωνούν ταυτόχρονα πολλές εφαρμογές X που τρέχουν κάτω από διαφορετικά λειτουργικά συστήματα, σε υπολογιστές με διαφορετική αρχιτεκτονική και οι οποίες να είναι γραμμένες σε διαφορετικές γλώσσες προγραμματισμού. Το γεγονός αυτό δίνει στον Λαβύρινθο την δυνατότητα να χρησιμοποιεί την οθόνη και τις συσκευές εισόδου υπολογιστών διαφορετικής αρχιτεκτονικής από εκείνη στην οποία τρέχει, χωρίς να χρειάζεται να γίνει επαναμετάφραση (recompilation) του πηγαίου κώδικα.

Η δυνατότητα αυτή σε συνδυασμό με την ευρεία διάδοση του περιβάλλοντος X, δίνει στον Λαβύρινθο ένα ακόμα πλεονέκτημα απέναντι σε άλλα ανταγωνιστικά σχεδιαστικά συστήματα όπως το Juno [9] ή το ThingLab [2].

1.2.3 Βιβλιοθήκες Διαδικασιών

Ο προγραμματιστής που πρόκειται να γράψει μια εφαρμογή X δεν θέλει να σκέφτεται την επικοινωνία με τον X server σαν μηνύματα πρωτοκόλου. Γι' αυτό χρησιμοποιεί την βιβλιοθήκη που κρύβει τις λεπτομέρειες της κωδικοποίησης του πρωτοκόλου, αλλά και βελτιστοποιεί την επικοινωνία ελαχιστοποιώντας τα μηνύματα μεταξύ εφαρμογής και X server. Για τις γλώσσες προγραμματισμού C και C++ η βιβλιοθήκη αυτή ονομάζεται Xlib (Xlib – C language Interface): η Xlib είναι το χαμηλότερο επίπεδο επικοινωνίας μιάς εφαρμογής X με τον X server πριν φτάσουμε στο πρωτόκολο. Ο Λαβύρινθος είναι γραμμένος σε C++[1] [16] και η γραφική επικοινωνία του γίνεται με το περιβάλλον X μέσω της βιβλιοθήκης Xlib[2].

Οι διαδικασίες που παρέχει η βιβλιοθήκη Xlib μπορούν να κατηγοριοποιηθούν ανάλογα με τις υπηρεσίες που προσφέρουν. Μια κατηγορία είναι οι διαδικασίες σύνδεσης. Με αυτές μια εφαρμογή X μπορεί να συνδέεται με έναν ή περισσότερους X servers, ή να αποσυνδέεται από αυτούς. Ακόμα, σε αυτές περιλαμβάνονται διαδικασίες που δίνουν πληροφορίες για τα χαρακτηριστικά του X server, όπως π.χ. ποιές είναι οι διαστάσεις της οθόνης που διαχερίζεται, αν είναι έγχρωμη, αν διαχερίζεται μία ή περισσότερες οθόνες, κ.α.

Αλλη κατηγορία είναι οι διαδικασίες χειρισμού των παραθύρων. Ο προγραμματιστής μπορεί, χρησιμοποιώντας τις, να δημιουργεί ιεραρχίες από ορθογώνια παράθυρα, να τα μεταβάλλει, και να τα καταστρέφει. Με αυτές μπορεί να παίρνει πληροφορίες για την κατάστασή των παραθύρων, όπως η θέση και το μέγεθός τους, το αν είναι ορατά ή όχι, κ.α.

---

[1] Χρησιμοποιούμε τον GNU "g++" compiler, έκδοση 1.37.2.

[2] X έκδοση 11.4.

Μια ακόμα κατηγορία είναι οι διαδικασίες σχεδίασης. Αυτές είναι υπεύθυνες για την εμφάνιση γραφικών στα παράθυρα που χρησιμοποιεί η εφαρμογή. Ο προγραμματιστής μπορεί να σχεδιάζει με αυτές απλά ή σύνθετα σχέδια στον δισδιάστατο χώρο, χρησιμοποιώντας τόξα, γραμμές, χρώματα (αν έχει έγχρωμη οθόνη), και να γράφει κείμενο με πολλαπλά είδη γραφής (fonts).

Η τελευταία μεγάλη κατηγορία είναι οι διαδικασίες χειρισμού των γεγονότων X (X events), με τα οποία ο X server ειδοποιεί την εφαρμογή ότι κάτι έχει συμβεί, ή μεταφέρει δεδομένα από τις συσκευές εισόδου που την ενδιαφέρουν.

1.2.4 Δομή μιας Τυπικής Εφαρμογής X, Χειρισμός των Γεγονότων X

Η δομή που πρέπει να έχει μια τυπική X εφαρμογή, όπως ο Λαβύρινθος, είναι η εξής: αρχικά ανοίγει μια σύνδεση με ένα X server (δηλαδή συνδέεται με το πρόγραμμα που είναι υπεύθυνο για τις συσκευές που θέλει να χρησιμοποίησει). Μετά, δημιουργεί τα παράθυρα που χρειάζεται, ορίζοντας τα χαρακτηριστικά τους.

Σαν επόμενο βήμα, ενημερώνει τον X server για τα γεγονότα που την ενδιαφέρουν, όπως το άν το ποντίκι πατήθηκε και πού, άν πληκτρολογήθηκε κάτι και τί, κπλ. Τέλος, το πρόγραμμα μπαίνει σε έναν άπειρο βρόγχο, στον οποίο περιμένει για γεγονότα απο τον X server και για κάθε ένα από αυτά δρά ανάλογα. Ολη η επικοινωνία μεταξύ εφαρμογής και X server γίνεται μέσα στον βρόγχο αυτό. Παραδείγματος χάριν, ο X server ειδοποιεί την εφαρμογή ότι δέχτηκε το πάτημα του αριστερού πλήκτρου του ποντικιού και η εφαρμογή στέλνει σ’ αυτόν την αίτηση να εμφανίσει ένα κύκλο καθορισμένων διαστάσεων με κέντρο την θέση που πατήθηκε το πλήκτρο.

Τα γεγονότα για τα οποία μπορεί να ενδιαφέρεται μια εφαρμογή X είναι αυτά που δημιουργούνται όταν αλλάξει η κατάσταση μιάς συσκευής εισόδου, και εκείνα που οφείλονται στην αλλαγή της κατάστασης της συσκευής εξόδου. Τα πρώτα (ButtonPress, ButtonRelease, KeyPress, KeyRelease) δημιουργούνται όταν μεταβάλλεται η κατάσταση του ποντικιού ή του πληκτρολογίου αντίστοιχα. Τα άλλα γεγονότα, που ενημερώνουν την εφαρμογή για τις αλλαγές στην οθόνη, οφείλονται κατά κύριο λόγο σε άλλες εφαρμογές X, οι οποίες μπορούν να την επηρεάζουν μεταβάλλοντας τα παράθυρα που αυτή διαχειρίζεται. Μια τέτοια εφαρμογή είναι ο "διαχειριστής παραθύρων" (window manager). Με το πρόγραμμα αυτό ο χρήστης μπορεί να μετακινεί παράθυρα, να τους αλλάζει μέγεθος, να τα κρύβει κάνοντάς τα μικρές εικόνες (icons), δηλαδή μπορεί να τακτοποιεί τα παράθυρα στην οθόνη.

Ετσι, παραδείγματος χάριν, αν ο "διαχειριστής παραθύρων" αλλάξει τις διαστάσεις και την θέση του παραθύρου μιάς εφαρμογής X, τότε εκείνη ίσως να χρειάζεται να ειδοποιηθεί για την αλλαγή αυτή (Resize event). Μια τέτοια ειδοποίηση μπορεί να είναι αναγκαία π.χ. αν η εφαρμογή σχεδιάζει με κέντρο το μέσον του παραθύρου. Αυτές οι αλλαγές είναι πιθανό να έχουν παραμορφώσει το σχέδιο και κατά συνέπεια η επανασχεδίαση είναι απαραίτητη. Για τον λόγο αυτό, ο προγραμματιστής πρέπει να λαμβάνει υπ’ όψη του την επίδραση άλλων εφαρμογών X στο πρόγραμμά του ώστε αυτό να δρά κατάλληλα κάθε φορά.

1.2.5 Βιβλιοθήκες Οντοτήτων

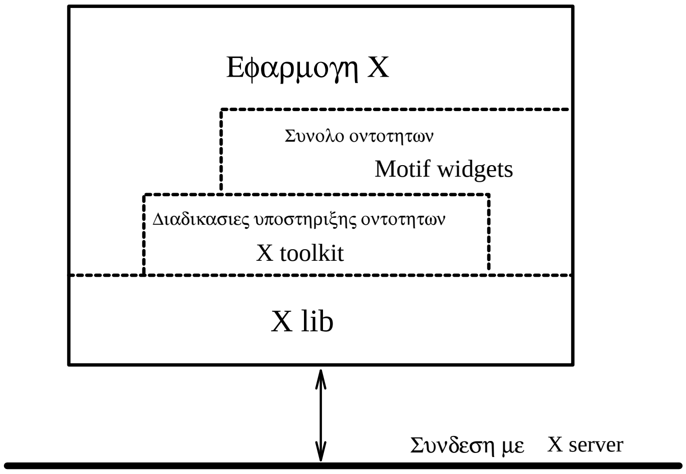
Ολα αυτά εισάγουν πολλές λεπτομέρειες στον προγραμματισμό που χρειάζεται να γίνει και κάνουν σχετικά πολύπλοκη την δημιουργία εφαρμογών X. Για τον λόγο αυτό, έχουν αναπτυχθεί πρόσθετες βιβλιοθήκες διαδικασιών, οι οποίες επιτρέπουν στον προγραμματιστή να εργάζεται σε υψηλότερο επίπεδο.

Οι βιβλιοθήκες αυτές (X widget sets) [11, 12] προσφέρουν αυτόνομες οντότητες με συγκεκριμένα χαρακτηριστικά, τις οποίες ο προγραμματιστής μπορεί να χρησιμοποιεί διαμορφώνοντάς τις κατάλληλα για να ταιριάζουν στις ανάγκες του προγράμματος. Ακόμα παρέχουν ένα σύνολο από διαδικασίες υποστήριξης της λειτουργικότητας αυτών των οντοτήτων (X toolkit) [8].

Μια τέτοια οντότητα ονομάζεται widget και είναι στην ουσία μια ιεραρχία από παράθυρα με δεδομένη λειτουργικότητα και διαμορφώσιμα χαρακτηριστικά. Ο προγραμματιστής επωφελείται αυτών, κάνοντας χρήση του ειδικού τρόπου επικοινωνίας που παρέχει το κάθε widget. Ο τρόπος αυτός συνίσταται στην παροχή μηχανισμών σύνδεσης συγκεκριμένων λειτουργιών του widget με διαδικασίες ορισμένες από τον προγραμματιστή ("hook's", callbacks), έτσι ώστε οι λειτουργίες αυτές να έχουν αντίκτυπο και στο πρόγραμμα.

Παράδειγμα ενός widget είναι το μενού (menu), στο οποίο ο προγραμματιστής μπορεί, μέσω των διαδικασιών της βιβλιοθήκης, να ορίσει το μέγεθος, τα πεδία του (items), την θέση που θα εμφανίζεται και πότε, καθώς και την διαδικασία (callback) που θα καλείται όποτε ο χρήστης επιλέγει το κάθε πεδίο του.

Ετσι, ο προγραμματιστής δεν χρειάζεται να προγραμματίσει το αντικείμενο αυτό λεπτομερώς. Το μειονέκτημα των widgets είναι ότι για να κατανοήσει κανείς την λειτουργία τους χρειάζεται να έχει γνώση των λεπτομερειών που αυτά προσπαθούν να κρύψουν, όπως τον χειρισμό των γεγονότων X.



Σχήμα 1.2: Τυπική εφαρμογή X (Λαβύρινθος)

Στο σχήμα 1.2 βλέπουμε την προγραμματιστική δομή μιάς τυπικής εφαρμογής X, δομή την οποία χρησιμοποιεί και ο Λαβύρινθος. Στο σχήμα βλέπουμε πώς η εφαρμογή δομημένη πάνω στην βιβλιοθήκη γραφικών Xlib, την βιβλιοθήκη υποστήριξης οντοτήτων (X toolkit) και την βιβλιοθήκη οντοτήτων Motif. Στο σχήμα οι διαδικασίες υποστήριξης των widgets φαίνονται χωριστά από το σύνολο των widgets που χρησιμοποιεί η εφαρμογή.

Το γραφικό κομμάτι του Λαβύρινθου υλοποιήθηκε χωρίς να χρησιμοποιηθεί καμία βιβλιοθήκη οντοτήτων. Ο λόγος που χρησιμοποιήσαμε μόνο την Xlib είναι γιατί αρχικά την θεωρήσαμε ευκολότερη από την βιβλιοθήκη υποστήριξης X toolkit (Xt) και την βιβλιοθήκη οντοτήτων X Athena widgets (Xaw) [12]. Η Xlib είναι πολύ σαφής ως προς τις δυνατότητες και τις ευκολίες που προσφέρει σε αντιθεση με τις Xt και Xaw, οι οποίες ήταν ασαφές εάν προσέφεραν πραγματικά κάτι ανάλογα χρήσιμο σε σχέση με την αύξηση του μεγέθους του εκτελέσιμου κώδικα (binary code) που η χρήση τους συνεπάγεται.

Με την κοινοποίηση της βιβλιοθήκης Motif αποφασίστηκε να μετατρέψουμε το κομμάτι της γραφικής επικοινωνίας ώστε να χρησιμοποιεί τα Motif widgets[3]. Η βιβλιοθήκη Xm (Motif widgets) [10, 11] παρέχει ένα πλούσιο σύνολο από widgets, όπως μενού, ολισθαίνουσες λαβές (scrollbars), χειρισμό κειμένου (text widget), widgets εισόδου εντολών (command widget) κ.α.

Ο χρήστης του Λαβύρινθου έχει πλέον την δυνατότητα να μετακινεί το σχέδιό του στο γραφικό παράθυρο του συστήματος με την βοήθεια ολισθαινουσών λαβών. Μελλοντικά θα μπορεί να δίνει εντολές στο σύστημα χρησιμοποιώντας μενού εντολών και παλέττες πρωτογενών κυττάρων, ελαχιστοποιώντας έτσι την χρήση του πληκτρολογίου. Ακόμα, στο μέλλον ο Λαβύρινθος θα παρέχει widgets ως πρωτογενή κύτταρα.

1.2.6 Μειονεκτήματα από την χρήση της Βιβλιοθήκης Motif

Ενα σοβαρό μειονέκτημα της χρήσης των Motif widgets είναι ότι αυξάνουν σημαντικά το μέγεθος του εκτελέσιμου κώδικα του συστήματος. Συγκεκριμένα, ο κώδικας του Λαβύρινθου αυξήθηκε από 1.1 σε 1.8 MBytes για Sun-3, ή από 1.2 σε 5.8 MBytes για Sun-4, όταν προστέθηκαν τα Motif widgets στο αρχικό σύστημα που χρησιμοποιούσε μόνο Xlib.

Τα προβλήματα που παρουσιάζουν τα προγράμματα με μεγάλο κώδικα είναι η σπατάλη χώρου στο δίσκο, η σπατάλη κεντρικής μνήμης και swap space, η καθυστέρυση εκκίνησης του προγράμματος λόγω του μακρού χρόνου να φορτωθεί αυτό από τον δίσκο στην μνήμη, και ο μεγάλος χρόνος compiling που δυσκολεύει την ανάπτυξή τους. Το πρόβλημα επιδύνωσε για μας η αδυναμία του GNU g++ compiler να υποστηρίξει την δυναμική σύνδεση (dynamic linking) του προγράμματος με τις βιβλιοθήκες αυτές.

[3] Στην μεταγραφή αυτή βοήθησε σημαντικά ο Κώστας Κούτσικος, μέλος της ερευνητικής ομάδας του Λαβύρινθου.

1.3 Επικοινωνία με τον χρήστη

Ο Λαβύρινθος παρέχει, όπως αναφέραμε προηγουμένως, ένα γραφικό editor με τον οποίο ο χρήστης μπορεί να χειρίζεται τις εσωτερικές δομές αναπαράστασης σχεδίων. Ο χειρισμός αυτός συνίσταται στον ορισμό νέων κυττάρων, την προσθήκη σε αυτά ή αφαίρεση από αυτά κυττάρων–παιδιών, την πραγματοποίηση συνδέσεων και αποσυνδέσεων, και την αλλάγη τιμών των πορτών εισόδου.

1.3.1 Ο Γραφικός editor

Ο γραφικός editor του Λαβύρινθου είναι ένα μέρος του συστήματος το οποίο επιτρέπει στο χρήστη να χειρίζεται τα σχέδια του μέσω του πληκτρολογίου και του ποντικιού, ενώ παράλληλα μπορεί να βλέπει στην οθόνη τα αποτελέσματα των χειρισμών του.

Στην παρούσα κατάσταση του συστήματος, ο χρήστης μπορεί να ορίζει (ή να μεταβάλλει) ένα μόνο σύνθετο κύτταρο (σχέδιο) και αυτό πρέπει να είναι η ρίζα του δέντρου ιεραρχίας κυττάρων που υπάρχουν στις εσωτερικές δομές του συστήματος. Το κύτταρο αυτό λέγεται "κύτταρο εργασίας" (edit cell). Η γραφική αναπαράστασή του εμφανίζεται στην οθόνη. Ο χρήστης έχει την δυνατότητα να επιλέγει τα κύτταρά–παιδιά του "κυττάρου εργασίας", με σκοπό την αλλαγή των τιμών που έχουν οι πόρτες τους, ή την σύνδεση και την αποσύνδεσή τους. Η επιλογή γίνεται μέσω του ποντικίου, με το οποίο ο χρήστης διαλέγει ένα σημείο της οθόνης και το σύστημα, αυτόματα, προσπαθεί να ανακαλύψει ποιό κύτταρο είναι κοντά στο σημείο αυτό ώστε να το επιλέξει.

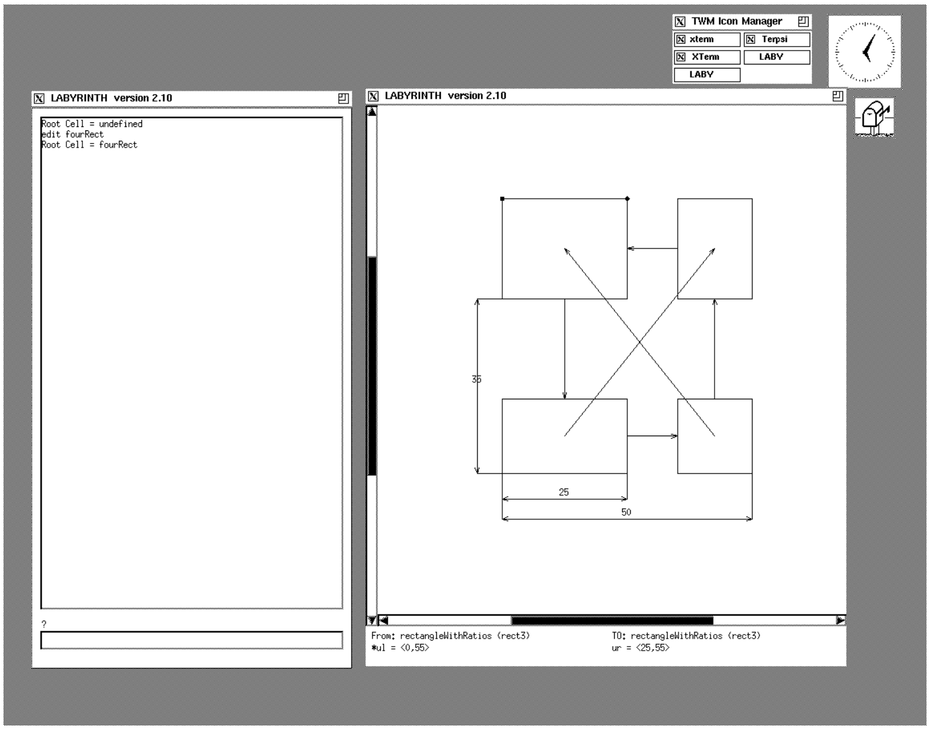
Οι συνδέσεις, οι αποσυνδέσεις, και η αλλαγή στις τιμές των πορτών των συνιστώντων κυττάρων γίνονται μέσω εντολών από το πληκτρολόγιο. Ο Λαβύρινθος παρέχει ακόμα στον χρήστη ένα πλήθος από εντολές με τις οποίες αυτός μπορεί να διασχίσει το δίκτυο των συνδέσεων και να επεμβαίνει σε κάθε ένα από τα κύτταρα του δικτύου ξεχωριστά.

Υπάρχουν, ακόμα, εντολές με τις οποίες ο χρήστης μπορεί να αποθηκεύσει τον ορισμό ενός κυττάρου σε ένα αρχείο, σε μορφή ASCII, ή να διαβάζει ένα τέτοιο ορισμό. Το σύστημα παρέχει επίσης εντολές διαχείρισης της γραφικής αναπαράστασης του "κυττάρου εργασίας" στην οθόνη, όπως η ανανέωση των περιεχομένων της οθόνης (refresh, redraw), η εμφάνιση ενός σχεδιαστικού πλέγματος (grid), κ.α.

Οι παραπάνω εντολές, καθώς και οι υπόλοιπες που δεν παρουσιάσαμε εδώ, περιγράφονται λεπτομερώς στο LABY User’s Manual [7].

1.3.2 Υλοποίηση με Motif widgets

Η επικοινωνία του γραφικού editor του Λαβύρινθου με τον χρήστη γίνεται μέσω δύο γραφικών παραθύρων (εικόνα 1.3). Το πρώτο (δεξιά) είναι το παράθυρο σχεδίασης. Εκεί εμφανίζεται η γραφική αναπαράσταση του "κυττάρου εργασίας", καθώς και διάφορες πληροφορίες σχετικές με τα κύτταρα που έχουν επιλεγεί, με σκοπό να υποστούν κάποια μεταβολή. Είναι υλοποιημένο σαν σύνθετο widget, που περιέχει μια ιεραρχία από άλλα widgets. Τη μεγαλύτερη περιοχή την καταλαμβάνει ένα DrawingArea widget, χωρίς ιδιαίτερη λειτουργικότητα εκτός από το ότι παρέχει ένα παράθυρο στο οποίο ο χρήστης μπορεί να σχεδιάζει. Στο παράθυρο αυτό στέλνονται οι εντολές δημιουργίας γραφικών από τις διαδικασίες σχεδίασης των πρωτογενών κυττάρων. Στο κάτω και αριστερό μέρος αυτής της περιοχής υπάρχουν δύο ολισθαίνουσες λαβές (ScrollBar widgets), μια οριζόντια και μια κατακόρυφη. Με αυτές, ο χρήστης μπορεί να μετακινεί το σχέδιο οριζόντια ή κατακόρυφα μέσα στην περιοχή σχεδίασης, προκειμένου να δει κομμάτια του που δεν χωρούσαν σ' αυτή. Τέλος, στο κάτω μέρος της περιοχής σχεδίασης υπάρχει μια άλλη περιοχή (Frame widget), όπου εμφανίζονται πληροφορίες σχετικές με τα επιλεγμένα κύτταρα.



Εικόνα 1.3: Ο Λαβύρινθος μέσα στο περιβάλλον X

Το δεύτερο παράθυρο (αριστερά) υπάρχει για να δέχεται εντολές πληκτρολογίου από τον χρήστη και να εμφανίζει μηνύματα λάθους (παράθυρο εντολών). Είναι υλοποιημένο ως ένα Command widget, το οποίο περιλαμβάνει μία περιοχή (κάτω) για να δέχεται κείμενο και μία (πάνω) όπου φαίνονται πιο πρόσφατες γραμμές του διαλόγου Λαβύρινθου–χρήστη. Οταν η περιοχή αυτή γεμίσει με μηνύματα, εμφανίζεται μια ολισθαίνουσα λαβή, με την οποία ο χρήστης μπορεί να ανατρέξει σε μηνύματα που δεν φαίνονται πλέον. Αντίστοιχα, αν πληκτρολογηθεί κείμενο με μήκος μεγαλύτερο από αυτό της περιοχής εισόδου, εμφανίζεται μια οριζόντια ολισθαίνουσα λαβή. Η λειτουργικότητα που περιγράφεται εδώ παρέχεται από το Command widget που χρησιμοποιήσαμε [10, 11].

1.3.3 Επίδραση από άλλες Εφαρμογές X

Ενα από τα ενδιαφέροντα θέματα που αντιμετωπίστηκαν κατά την υλοποίηση μας, είναι η επίδραση πάνω στα παράθυρα του Λαβύρινθου από άλλες εφαρμογές, όπως ο "διαχειριστής παραθύρων". Ο "διαχειριστής παραθύρων" μπορεί να επεμβαίνει σε οποιοδήποτε από τα δύο παράθυρα, μεταβάλλοντας το μέγεθός του, αλλάζοντάς του θέση, κάνοντάς το μικρή εικόνα (icon – εικονίδιο), ή μεταβάλλοντας την σειρά του στην στοίβα επικαλυπτόμενων παραθύρων (stack of overlapping windows). Για όλες αυτές τις μεταβολές ο X server ειδοποιεί την εφαρμογή με ένα ή περισσότερα γεγονότα X για κάθε επηρεαζόμενο παράθυρο. Ο Λαβύρινθος έχει δηλώσει εξ' αρχής στον X server ότι ενδιαφέρεται για τα γεγονότα αυτά, και έχει ορίσει για κάθε ένα από αυτά μία διαδικασία ως χειριστή γεγονότων (event handler). Συγκεκριμένα, για την παραπάνω επίδραση, μας ενδιαφέρουν τα γεγονότα: ConfigureNotify (αλλαγή μεγέθους ή θέσης), και Expose (εμφάνιση μέρους ή ολόκληρου του παραθύρου).

Η διαδικασία που καλείται όταν έρθει το γεγονός ConfigureNotify από το παράθυρο εντολών, διαβάζει τις νέες διαστάσεις του παραθύρου και με βάση αυτές μεταβάλλει (αυξάνει ή μειώνει) το πλήθος των γραμμών της λίστας της ιστορίας των εντολών. Οταν συμβαίνει το ίδιο στο παράθυρο σχεδίασης, όλα τα επιμέρους widgets θα ακολουθήσουν την αλλαγή αυτή, αλλά υπεύθυνη για την διάδοση της αλλαγής είναι η βιβλιοθήκη Xm. Ωστόσο, ορίσαμε μια διαδικασία, η οποία διαβάζει τις νέες διαστάσεις των επιμέρους παραθύρων του παραθύρου σχεδίασης και ενημερώνει ορισμένες μεταβλητές. Τις μεταβλητές αυτές θα χρησιμοποιήσουμε αργότερα για να ευθυγραμμίσουμε το σχέδιο στο κέντρο της οθόνης, για να γράφουμε σε συγκεκριμένες θέσεις τις πληροφορίες για τα κύτταρα, και για να μετατρέπουμε σωστά τις συντεταγμένες που χρησιμοποιεί ο Λαβύρινθος (συντεταγμένες κόσμου) σε συντεταγμένες παραθύρου σχεδίασης (συντεταγμένες οθόνης) και αντίστροφα.

Στην περίπτωση που ένα από τα δύο παράθυρα ήταν κρυμμένο (ολόκληρο ή ένα μέρος του), πίσω από κάποιο άλλο και ο "διαχειριστής παραθύρων" το έφερε μπροστά, καθώς και όταν ένα παράθυρο επανέρχεται από εικονίδιο στην αρχική του κατάσταση, ο Λαβύρινθος πρέπει να επανεμφανίσει τα χαμένα (εξαφανισμένα) περιεχόμενα του. Στις παραπάνω περιπτώσεις έχουμε ένα γεγονός Expose. Ορίσαμε μια διαδικασία χειρισμού των γεγονότων Expose του σχεδιαστικού παραθύρου, η οποία καλεί τον μηχανισμό επανασχεδίασης του "κυττάρου εργασίας" που εμφανίζεται στο σχεδιαστικό παράθυρο, καθώς και την διαδικασία ανανέωσης των περιεχομένων του παραθύρου πληροφοριών. Τα γεγονότα Expose του παραθύρου εντολών τα χειρίζεται κατευθείαν το Command widget.

1.3.4 Χειρισμός του Πληκτρολογίου

Αλλο ενδιαφέρον θέμα που αντιμετωπίσαμε είναι ο χειρισμός του πληκτρολογίου (γεγονότα KeyPress). Στο περιβάλλον X τα γεγονότα αυτά κατευθύνονται στο παράθυρο μέσα στο οποίο βρίσκεται ανά πάσα στιγμή το ποντίκι. Εμείς όμως θέλαμε να δέχεται ο Λαβύρινθος εντολές μέσω του πληκτρολογίου, ακόμα και όταν το ποντίκι βρίσκεται στο παράθυρο σχεδίασης, και οι εντολές αυτές να κατευθύνονται στο παράθυρο εντολών με τον ίδιο τρόπο όπως και όταν το ποντίκι βρίσκεται εκεί. Παραδείγματος χάριν, κατά την διάρκεια της σχεδίασης, ο χρήστης χρησιμοποιεί το ποντίκι για να επιλέγει τις πόρτες των κυττάρων που θα συνδέσει, ενώ παράλληλα δίνει εντολές μέσω του πλήκτρολογίου, χωρίς προφανώς να αλλάζει παράθυρο. Κατά την ανάπτυξη του συστήματος δώθηκαν δύο λύσεις στο πρόβλημα αυτό, ανάλογα με την βιβλιοθήκη που χρησιμοποιήσαμε και τις ευκολίες που αυτή παρείχε.

Οταν ο Λαβύρινθος συνεργάζονταν μόνο με την βιβλιοθήκη Xlib (εκδόσεις 2.0 έως και 2.8), χρησιμοποιούσαμε έναν προσομοιωτή τερματικού σαν παράθυρο εντολών, και συγκεκριμένα το xterm – την πλέον γνωστή εφαρμογή X (DEC VT100 terminal emulator). Επειδή το xterm είναι μια ανεξάρτητη εφαρμογή X και το χρειαζόμασταν μόνο για να τυπώνει χαρακτήρες με όμορφο τρόπο, τοποθετήσαμε πάνω του ένα αόρατο παράθυρο το οποίο στην ουσία "έκλεβε" τα γεγονότα

KeyPress από το xterm και τα έστελνε στον Λαβύρινθο, ενώ άφηνε τον χρήστη να βλέπει τα περιεχόμενα του παραθύρου του xterm. Με το τέχνασμα αυτό αδρανοποιούσαμε την είσοδο (stdin) της εφαρμογής και χρησιμοποιούσαμε μόνο την έξοδό της (stdout), χωρίς να χάνεται όμως η υπόλοιπη λειτουργικότητά της (δυνατότητα προσομοίωσης). Κάθε φορά που το αόρατο ή το σχεδιαστικό παράθυρο δεχόταν ένα γεγονός KeyPress, αποθηκεύαμε τον εισερχόμενο χαρακτήρα σε έναν string buffer, και ταυτόχρονα τον τυπώναμε στο παράθυρο του xterm.

Η εμπειρία αυτή φανηκε ιδιαίτερα χρήσιμη στη δημιουργία του νέου περιβάλλοντος επικοινωνίας (User Interface) με την χρήση της βιβλιοθήκης των Motif widgets (εκδόσεις 2.9 έως 2.11 – τρέχουσα). Αντικαταστήσαμε το xterm με ένα Motif "Command" widget κυρίως λόγω της αυξημένης ευελιξίας που προσφέρει το τελευταίο σε editing της εντολής πριν αυτή δοθεί στο σύστημα, και στην δυνατότητα ανάκλησης εντολών από ένα "history list" και επαναχρησιμοποίησής τους. Ωστόσο ένα νέο πρόβλημα παρουσιάστηκε: πώς το Command widget θα μπορούσε να δέχεται, εκτός από τα γεγονότα KeyPress που συνέβαιναν στο δικό του παράθυρο, και εκείνα που συνέβαιναν στο σχεδιαστικό παράθυρο; Το πρόβλημα δεν είναι απλό, διότι ο ρόλος μιας βιβλιοθήκης, όπως τα Motif widgets, είναι να κρύψει από τον προγραμματιστή τέτοιες λεπτομέρειες. Ετσι, δεν μπορούσαμε να ορίσουμε, όπως κάναμε πριν, μια διαδικασία που να δέχεται τα γεγονότα KeyPress και από τις δύο περιοχές, γιατί το Command widget είχε ήδη ορισμένες τέτοιες διαδικασίες για τον χειρισμό των γεγονοτων που συνέβαιναν στο παράθυρό του και δεν είχαμε την δυνατότητα να τις αλλάξουμε. Τελικά, η λύση που βρήκαμε ήταν να διαβιβάζουμε τα γεγονότα KeyPress του παραθύρου σχεδίασης, στο παράθυρο του Command widget κάνοντας χρήση της διαδικασίας XSetInputFocus της βιβλιοθήκης Xlib. Με τον τρόπο αυτό η διαδικασία χειρισμού των KeyPress του Command widget αντιλαμβανόταν τα γεγονότα αυτά ως γεγονότα που συνέβησαν στο παράθυρο εντολών. Ωστόσο αυτή η λύση αύξησε την πολυπλοκότητα του κώδικα, γιατί έπρεπε να χειριζόμαστε τις διάφορες ειδικές περιπτώσεις που προέκυψαν, όπως παραδείγματος χάριν τί νόημα θα έχει η διαβίβαση αυτή, όταν το παράθυρο εντολών έχει εξαφανιστεί (έχει γίνει εικονίδιο) κ.α.

1.3.5 Χειρισμός του Ποντικιού

Για την διαχείριση των γεγονότων από το ποντίκι (Button Press) στο σχεδιαστικό παράθυρο, ορίσαμε μια διαδικασία χειρισμού τους, η οποία, όταν πατηθεί ένα κουμπί (αριστερό, μεσαίο, δεξιό), καλεί τον μηχανισμό επιλογής του Λαβύρινθου, τον οποίο θα περιγράψουμε παρακάτω, για να επιλέξει το κοντινότερο κύτταρο στο σημείο εκείνο.

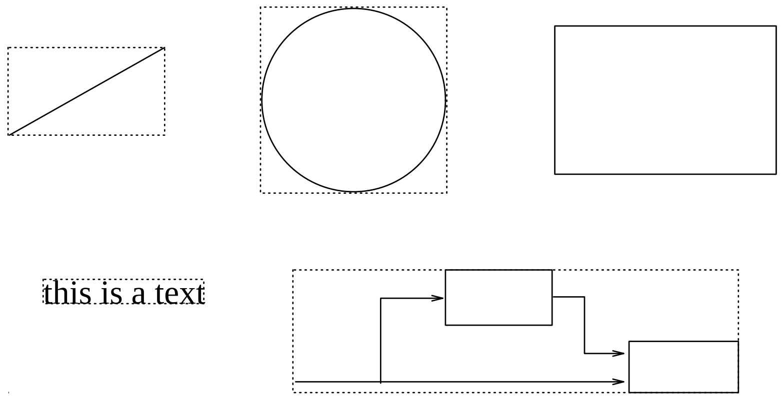
1.3.6 Συνεργασία με άλλες Εφαρμογές

Εκτός από ανθρώπινους χρήστες, ο γραφικός editor του Λαβύρινθου μπορεί να επικοινωνεί και με άλλες εφαρμογές X. Με τον τρόπο αυτό, σήμερα, το σύστημα επικοινωνεί με την εφαρμογή SelectionTool – έναν browser για την Software Information Base (SIB) που χρησιμοποιεί τον Λαβύρινθο για να αναπαριστά γραφικά τα περιεχόμενά της. Η εφαρμογή αυτή αναπτύσσεται στα πλαίσια του ερευνητικού προγράμματος ITHACA στο Ινστιτούτο Πληροφορικής του ΙΤΕ [3]. Ο Λαβύρινθος δέχεται από το SelectionTool εντολές εμφάνισης σχεδίων στην οθόνη, και παράλληλα την ενημερώνει για τις επιλογές που κάνει ο χρήστης. Για να γίνει η επικοινωνία αυτή, ορίσαμε έναν διαχειριστή γεγονότων ClientMessage (με τα οποία οι εφαρμογές X επικοινωνουν μεταξύ τους), που για κάθε τέτοιο γεγονός αποθηκεύει τα δεδομένα που αυτό μεταφέρει, και όταν η αποστολή της εντολής ολοκληρωθεί την τροφοδοτεί στο σύστημα πρός εκτέλεση.

Ανάλογα ορίσαμε μια διαδικασία με την οποία ο Λαβύρινθος παράγει τέτοια μηνύματα και μεταφέρει δεδομένα προς άλλες εφαρμογές X. Ωστόσο ο χρήστης δεν μπορεί ακόμα να ορίζει δυναμικά πότε και ποιά δεδομένα θα μεταδίδει το σύστημα και είναι αναγκασμένος να δώσει τις πληροφορίες αυτές σε πηγαίο κώδικα.

2. Ορθογώνια περιγράμματα κυττάρων (bounding boxes) και μηχανισμός επιλογής

Τα κύτταρα στον Λαβύρινθο έχουν ορθογώνια περιγράμματα (bounding boxes). Ως ορθογώνιο περίγραμμα ενός κυττάρου ορίζουμε το ελάχιστο ορθογώνιο παραλληλόγραμμο με πλευρές παράλληλες στους άξονες των X και Y που το περιβάλλει πλήρως. Ο ορισμός αυτός ισχύει για κύτταρα που έχουν ένα σχήμα ορατό στην οθόνη. Τέτοια γεωμετρικά κύτταρα, καθώς και τα ορθογώνια περιγράμματα τους (με διακεκομμένες γραμμές), βλεπουμε στο σχήμα 2.1.



Σχήμα 2.1: Κύτταρα με τα ορθογώνια περιγράμματά τους

2.1 Σκοπός και Αλγόριθμοι Ορθογώνιων Περιγραμμάτων

Ο λόγος που χρειάζονται τα περιγράμματα αυτά σε ένα σχεδιαστικό πρόγραμμα είναι, ότι αποτελούν μία εύκολη προσέγγιση της επιφάνειας που καλύπτει το κύτταρο στην οθόνη. Μια γραφική εφαρμογή, πολλές φορές, χρειάζεται να γνωρίζει την επιφάνεια που καταλαμβάνει ένα κύτταρο, έτσι ώστε να μπορεί να το εντοπίσει μέσα στο ευρύτερο σχέδιο.

Ο μηχανισμός επιλογής κυττάρων, στον Λαβύρινθο, στηρίζεται στην ύπαρξη των περιγραμμάτων αυτών. Για να επιλέξει ένα κύτταρο, ο χρήστης πρέπει να πατήσει το πλήκτρο του ποντικιού, όταν αυτό βρίσκεται "πάνω" στο κύτταρο. Αυτό το σημείο βρίσκεται οπωσδήποτε μέσα στην επιφάνεια που ορίζει το περίγραμμα του κυττάρου αυτού. Αρα, όλα τα κύτταρα των οποίων τα περιγράμματα δεν περιλαμβάνουν το επιλεγέν σημείο μπορούν να απορριφθούν άμεσα από υποψήφια για επιλογή.

Ενας ακόμα λόγος που μπορεί ένα σύστημα να χρειάζεται τα περιγράμματα είναι για να βελτιώσει την διαδικασία επανασχεδίασης. Συγκεκριμένα, ο Λαβύρινθος χρειάζεται να επανασχεδιάζει ορισμένα κύτταρα, στην περίπτωση που ένα μέρος του παραθύρου σχεδίασης ήταν κρυμμένο από "κάτι" το οποίο τώρα μετακινήθηκε. Καθώς αυτό το μέρος έρχεται πάλι μπροστά, χρειάζεται η επαναφορά στη οθόνη του κομματιού του σχεδίου που έχει σβηστεί. Τα κύτταρα που πρέπει να επανασχεδιαστούν είναι αυτά των οποίων το περίγραμμα βρίσκεται ολόκληρο, ή μερικώς, μέσα στην σβησμένη περιοχή.

2.1.1 Ποιά Κύτταρα έχουν Ορθογώνια Περιγράμματα

Στον Λαβύρινθο ορθογώνια περιγράμματα έχουν μόνο όσα κύτταρα είναι ορατά. Την συνθήκη αυτή μπορούν να ικανοποιήσουν τόσο πρωτογενή όσο και σύνθετα κύτταρα. Ενα σύνθετο κύτταρο είναι ορατό, αν ένα τουλάχιστον από τα επιμερους κύτταρα που το συνθέτουν είναι ορατό.

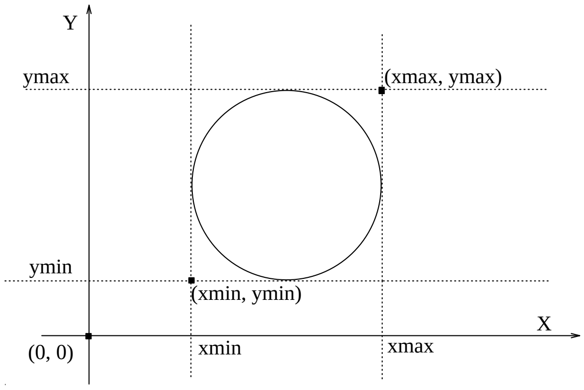
Ο λόγος που τα αόρατα κύτταρα δεν έχουν περίγραμμα είναι, διότι δέν έχουν γεωμετρική αναπαράσταση (δεν έχουν διαδικασίες σχεδίασης), και – προς το παρόν, τουλάχιστον – δεν επιλέγονται. Στο μέλλον, ίσως, αυτό να αλλάξει, ώστε ο χρήστης να μπορεί να τα επιλέγει και να τα μεταβάλλει με πιο άμεσο τρόπο απ' ότι συμβαίνει τώρα, όπου ο μόνος τρόπος για να τα επιλέξει είναι έμμεσα, διατρέχοντας τις συνδέσεις των κυττάρων.

2.1.2 Μηχανισμός Υποστήριξης των Ορθογωνίων Περιγραμμάτων

Κάθε πρωτογενές κύτταρο έχει μια διαδικασία προσδιορισμού του ορθογωνίου περιγράμματός του. Η διαδικασία αυτή είναι ορισμένη σε πηγαίο κώδικα, όπως και όλες οι διαδικασίες των πρωτογενών κυττάρων. Ετσι ο σχεδιαστής των πρωτογενών κυττάρων είναι υπεύθυνος να μάθει στο σύστημα πώς υπολογίζεται το περίγραμμα κάθε κυττάρου.

Για ένα σύνθετο κύτταρο, το ορθογώνιο περίγραμμα ορίζεται σαν το ελάχιστο εκείνο ορθογώνιο που περικλύει τα περιγράμματα όλων των συνιστώντων κυττάρων του. Χρησιμοποιώντας τους μηχανισμούς "κληρονομικότητας" της C++, δοθέντος ενός κυττάρου, το σύστημα είτε καλεί την διαδικασία προσδιορισμού του ορθογωνίου περιγράμματός του, εάν αυτό είναι πρωτογενές, αλλιώς βρίσκει το ελάχιστο περίγραμμα που περικλύει τα περιγράμματα των συνιστώντων κυττάρων του.

Οσο ένα κύτταρο δεν μεταβάλλεται, δεν υπάρχει λόγος να υπολογίζουμε το ορθογώνιο περίγραμμά του κάθε φορά που το χρειαζόμαστε. Μόνο όταν ένα κύτταρο μεταβληθεί χρειάζεται να υπολογίσουμε το νέο ορθογώνιο περίγραμμά του, αφού αυτό μπορεί να έχει αλλάξει. Το σύστημα διατηρεί το ορθογώνιο περίγραμμα κάθε κυττάρου σε μία δομή ονόματι class BBOX, που περιέχει τις συντεταγμένες κόσμου *xmin*,*ymin*,*xmax*,*ymax* (σχήμα 2.2).



Σχήμα 2.2: Τιμές και σημεία που ορίζουν ένα ορθογώνιο περίγραμμα

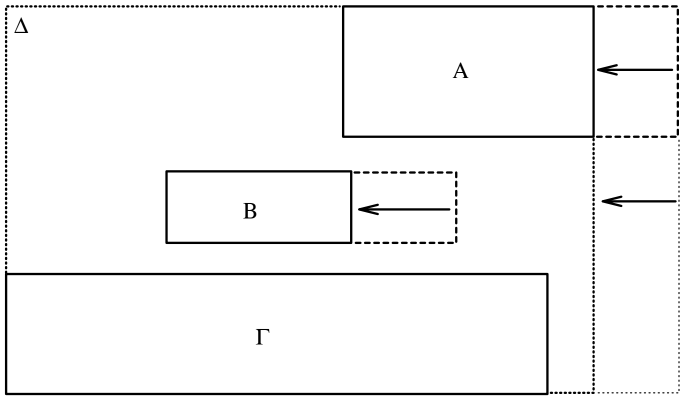
Ενας απλοϊκός τρόπος υπολογισμού του περιγράμματος ενός σύνθετου κύτταρου θα ήταν ο εξής. Κάθε φορά που αλλάξει το περίγραμμα ενός ή περισσοτέρων κυττάρων–παιδιών, διατρέχουμε όλα τα παιδιά του σύνθετου κυττάρου βρίσκοντας τα minimum των *xmin*,*ymin* τους και τα maximum των *xmax*,*ymax* τους. Στην πραγματικότητα όμως, εάν ξέρουμε ποιά περιγράμματα παιδιών αλλάζουν και ποιά ήταν η προηγούμενη τιμή τους, μπορούμε συχνά να κάνουμε τον παραπάνω επανυπολογισμό πολύ ταχύτερα, όπως εξηγεί η επόμενη παράγραφος.

2.1.3 Αποδοτικότερος υπολογισμός του ορθογώνιου περιγράμματος ενός σύνθετου κυττάρου

Ας δούμε πρώτα ένα παράδειγμα. Εστω ότι το σχέδιό μας είναι ένα σύνθετο κύτταρο, το Δ, αποτελούμενο από τρία ορθογώνια παραλληλόγραμμα, Α, Β, Γ, και ορισμένα κύτταρα που να τα συνδέουν και να διατηρούν μια σχέση μεταξύ αυτών (σχήμα 2.3). Το ορθογώνιο περίγραμμα του σύνθετου κύτταρου Δ θα είναι το ελάχιστο εκείνο ορθογώνιο περίγραμμα που περικλύει τα περιγράμματα των τριών ορατών κυττάρων Α, Β και Γ.

Σε περίπτωση που αλλάξει το μέγεθος ή (και) η θέση κυττάρου Β, όπως φαινεται στο σχήμα, το περίγραμμα του σύνθετου κυττάρου Δ δεν μεταβάλλεται. Αν όμως κατά κάποιο τρόπο, αλλάξει το μέγεθος ή (και) η θέση του κυττάρου Α, όπως φαίνεται στό σχήμα, τότε θα χρειαστεί επανυπολογισμός του ορθογωνίου περιγράμματος του σύνθετου κυττάρου. Την μεταβολή που πρέπει να υποστεί το Δ, την βλέπουμε στο σχήμα με διακεκομμένη γραμμή.

Με το παράδειγμα αυτό βλέπει κανείς ότι η μεταβολή του ορθογωνίου περιγράμματος ενός κυττάρου χαμηλά στην ιεραρχία μπορεί να συνεπάγεται, ή να μην συνεπάγεται, την ανάγκη ενημέρωσης του πατέρα του. Ο μηχανισμός μας πρέπει να λαμβάνει υπ’ όψη του τις μεταβολές στα περιγράμματα των συνιστώντων κυττάρων (παιδιά), όπου αυτές υπάρχουν, και να ενημερώνει μόνο εάν χρειάζεται το περίγραμμα του πατέρα. Τα παραπάνω οδηγούν στο



Σχήμα 2.3: Παράδειγμα ενημέρωσης του περιγράματος σύνθετου κυττάρου

συμπέρασμα ότι ο υπολογισμός του ορθογωνίου περιγράμματος ενός σύνθετου κυττάρου πρέπει να γίνεται σταδιακά, καθώς υπολογίζονται τα περιγράμματα των συνιστώντων κυττάρων.

Με βάση το παραπάνω παράδειγμα μπορούμε να δούμε τις επιμέρους περιπτώσεις που υπάρχουν. Οταν μεγαλώνει από μία πλευρά το περίγραμμα ενός παιδιού, αρκεί να ενημερώνουμε το περίγραμμα του πατέρα, κρατώντας την "απόλυτα μεγαλύτερη"[4] τιμή μεταξύ της νέας τιμής αυτής της πλευράς του περιγράμματος του παιδιού και της παλαιάς τιμής της αντίστοιχης πλευράς του πατέρα.

Οταν μικραίνει από μία πλευρά το περίγραμμα του παιδιού, τότε πρέπει να ενημερώσουμε τον πατέρα μόνο αν αυτή η πλευρά του παιδιού ήταν η ακραία εκείνη πλευρά η οποία καθόριζε προηγουμένως την αντίστοιχη πλευρά του πατέρα. Σ’ αυτή την περίπτωση, η ενημέρωση του πατέρα πρέπει να γίνει με "πλήρη επανυπολογισμό". Οπως φαίνεται στο παράδειγμα, η πλευρά που μετακινείται (στο κύτταρο Α) "παρασύρει" μαζί της και την πλευρά του πατέρα (Δ). Αν το περίγραμμα του παιδιού μικρύνει αρκετά, μπορεί το περίγραμμα του πατέρα να μην ακολουθήσει την αλλαγή, γιατί το περίγραμμα ενός άλλου παιδιού (Γ) το εμποδίζει. Για την περίπτωση αυτή, είμαστε υποχρεωμένοι να περιμένουμε να τελειώσει ο υπολογισμός των συνιστώντων κυττάρων (άρα και των ορθογωνίων περιγραμμάτων τους), και μετά να υπολογίσουμε το περίγραμμα του σύνθετου κύτταρου.

Ετσι καταλήξαμε στον παρακάτω αργόριθμο επανυπολογισμού των κυττάρων και ενημέρωσης των ορθογωνίων περιγραμμάτων τους.

Για κάθε κύτταρο στην λίστα επανυπολογισμού:

Εάν το κύτταρο είναι πρωτογενές **τότε**

α. Κάλεσε την διαδικασία υπολογισμού του (επανυπολόγισέ το)

β. Κάλεσε την διαδικασία υπολογισμού του νέου ορθογωνίου περιγράμματός του (υπολόγισε το περίγραμμά του), **μόνο εάν** το κύτταρο είναι ορατό

[4] Ονομάζω "απόλυτα μεγαλύτερη" το minimum προκειμένου για *xmin* ή *ymin*, ή το maximum προκειμένου για *xmax* ή *ymax*.

Εάν το κύτταρο είναι σύνθετο **τότε**

α. **Για κάθε** συνιστών κύτταρο (παιδί)

1. Κάλεσε την διαδικασία υπολογισμού του (επανυπολόγισέ το κύτταρο και το περίγραμμά του)
2. Ενημέρωσε το περίγραμμα του πατέρα, αν χρειάζεται, ως εξής:

Για κάθε πλευρά του παιδιού

i. **Εάν** η πλευρά του νέου περιγράμματος του παιδιού είναι "απόλυτα μεγαλύτερη" εκείνης του προηγούμενου περιγράμματός του (το περίγραμμα του παιδιού μεγαλώνει προς την πλευρά αυτή), **τότε** μεγάλωσε το περίγραμμα του πατέρα **μόνο εάν** η πλευρά του νέου περιγράμματος του παιδιού είναι "απόλυτα μεγαλύτερη" εκείνης του περιγράμματος του πατέρα.

ii. **Αλλιώς εάν** η πλευρά του νέου περιγράμματος του παιδιού είναι "απόλυτα μικρότερη" εκείνης του προηγούμενου περιγράμματός του **και** η τελευταία ήταν ίση με την αντίστοιχη πλευρά του περιγράμματος του πατέρα (το περίγραμμα του παιδιού μικραίνει προς την πλευρά που καθορίζει την πλευρά του περιγράμματος του πατέρα), **τότε** σταμάτησε να ενημερώνεις σταδιακά τον πατέρα και μην ξανακάνεις το βήμα (α.2) μέχρι να ολοκληρωθεί η διαδικασία επανυπολογισμού όλων των παιδιών του πατέρα.

β. **Εαν** για κάποιο από τα παιδιά το βήμα (α.2.ii) ήταν αληθές **τότε**

1. Μηδένισε το περίγραμμα του πατέρα
2. **Για κάθε** συνιστών κύτταρο (παιδί)

Πρόσθεσε το περίγραμμά του στο περίγραμμα του πατέρα.

Ο αλγόριθμος για την ενημέρωση των ορθωγονίων περιγραμμάτων των κυττάρων, διασχίζει αποδοτικά την ιεραρχία των κυττάρων του σχεδίου. Επειδή γίνεται παράλληλα με την διαδικασία επανυπολογισμού των κυττάρων, εξασφαλίζει ότι μόνο τα κύτταρα που επανυπολογίστηκαν θα ανανεώσουν το ορθογώνιο περίγραμμά τους. Επίσης, ενημερώνει τα περιγράμματα μόνο όταν αυτό χρειάζεται να γίνει (βήμα (α.2)).

2.2 Μηχανισμός Επιλογής

Ο μηχανισμός επιλογής των κυττάρων στον Λαβύρινθο χρησιμοποιεί άμεσα τα ορθογώνια περιγράμματα των κυττάρων. Χωρίς περιγράμματα, η επιλογή ενός κυττάρου θα απαιτούσε να διασχίσει κανείς όλη την ιεραρχία των κυττάρων και για καθένα να καλεί μια διαδικασία η οποία δοθέντος σημείου, βρίσκει αν το σημείο αυτό αποτελεί μέρος του κυττάρου ή βρίσκεται πολύ κοντά του.

2.2.1 Χρησιμότητα των Ορθογωνίων Περιγραμμάτων

Την έννοια της απόστασης δεν είναι άσχετη με το ορθογώνιο περίγραμμα. Ωστόσο, είναι φανερό ότι το ορθογώνιο περίγραμμα δεν μπορεί να περιγράψει αποστάσεις, αλλά μόνο επιφάνειες. Ακόμα, είναι σαφές ότι δεν παρέχει ακρίβεια ούτε σε αυτή την έννοια της επιφάνειας του κυττάρου. Στο σχήμα 2.1 βλέπουμε το ορθογώνιο περίγραμμα του κυττάρου "ευθύγραμμο τμήμα". Η επιφάνεια που καταλαμβάνει το κύτταρο, και αυτή που περιγράφεται από το περίγραμμά του είναι πολύ διαφορετικές.

Ωστόσο, αυτό δεν είναι μειονέκτημα στην περίπτωση του μηχανισμού επιλογής, διότι ο λόγος που χρειαζόμαστε τα περιγράμματα είναι για να μπορούμε να απορρίπτουμε εύκολα τα κύτταρα που δεν είναι υποψήφια για επιλογή. Αυτό είναι απαραίτητο να γίνεται, γιατί, όπως είπαμε και στην προηγούμενη παράγραφο, το σχέδιο του χρήστη μπορεί να έχει αρκετές χιλίαδες κύτταρα.

Το "ξάκρισμα" αυτό μπορεί να ξεκινά από τα κύτταρα που δεν εμφανίζονται στην οθόνη. Αυτή είναι η περίπτωση όπου το σχέδιο δεν χωράει ολόκληρο στην οθόνη και έχει "ψαλιδιστεί" (clipped). Προφανώς τα κύτταρα εκείνα δεν μπορούν να επιλεγούν, αφού ο χρήστης δεν τα βλέπει. Αλλα κύτταρα που απορρίπτονται είναι εκείνα, που το ορθογώνιο περίγραμμά τους δεν τέμνεται με την περιοχή που ο μηχανισμός επιλογής ορίζει γύρω από το σημείο που επέλεξε ο χρήστης με το ποντίκι.

Η περιοχή αυτή χρησιμοποιείται για να επιτρέψει μια "ασάφεια" ή ένα σφάλμα στο σημείο που το ποντίκι υποδεικνύει, δεδομένου ότι το ανθρώπινο χέρι εύκολα κάνει σφάλμα μερικών pixels[5] όταν χειρίζεται το ποντίκι. Η περιοχή αυτή ορίσαμε να είναι τέσσερα pixels γύρω από το σημείο επιλογής και για απλότητα περιγράφεται από ένα ορθογώνιο περίγραμμα. Χρησιμοποιώντας το σύστημα διαπιστώσαμε ότι ο αριθμός αυτός (4 pixels) δεν είναι ούτε πολύ μικρός ώστε να είναι δύσκολο να "πετύχει" κανείς ένα αντικείμενο, ούτε πολύ μεγάλος ώστε να αυξάνεται υπέρμετρα το πλήθος των αντικειμένων που βρίσκονται τόσο κοντά. Επειδή τα περιγράμματα ορίζονται σε συντεταγμένες κόσμου, μετατρέπουμε τα σημεία που ορίζουν την επιφάνεια επιλογής σε συντεταγμένες κόσμου και δημιουργούμε μια δομή BBOX που περιγράφει την επιφάνεια αυτή.

2.2.2 Τρόπος Επιλογής

Ο χρήστης μπορεί να υποδείξει ένα σημείο που να βρίσκεται "κοντά" σε περισσότερα από ένα κύτταρα. Απορρίπτοντας τα κύτταρα που απέχουν από το σημείο επιλογής, περισσότερο από το προκαθορισμένο "όριο ανοχής", που αποτελεί ένα φράγμα για την επιλογή, τα υποψήφια κύτταρα μπορεί να είναι και πάλι περισσότερα από ένα. Από αυτά πρέπει να διαλεξουμε το κύτταρο εκείνο που απέχει την μικρότερη απόσταση από το δοθέν σημείο. Ωστόσο πρέπει να παρέχουμε στον χρήστη την δυνατότητα να επιλέγει κατά προσέγγιση, και για το λόγο αυτό, πρέπει να κρατάμε τα υπολοιπα υποψήφια κύτταρα και να παρέχουμε την ευκολία στον χρήστη να τα βλέπει.

---

[5] Κουκίδες της οθόνης, η ελάχιστη μονάδα χειρισμού γραφικών, μονάδα μέτρησης των συντεταγμένων στο παράθυρο ή την οθόνη. (pixel = picture element).

Πράγματι, ο μηχανισμός επιλογής του Λαβύρινθου δίνει την δυνατότητα επιλογής μέχρι 64 κυττάρων που είναι όλα "πολύ κοντά" στο σημείο επιλογής. Πιστεύουμε ότι αν στο σχέδιο του χρήστη έχει περισσότερα από 64 κύτταρα κοντά στο σημείο επιλογής, τότε είτε το σημείο επιλογής είναι λανθασμένο, οπότε ο χρήστης καλείται να επιλέξει από άλλο σημείο το κύτταρο που θέλει, είτε το σχέδιο είναι πολυ πυκνό, οπότε ο χρήστης πρέπει να δώσει εντολή στο σύστημα να μεγενθύνει το σχέδιο στην οθόνη (zoom), ώστε οι σχετικές αποστάσεις μεταξύ των κυττάρων να μεγαλώσουν. Στην πράξη, φαίνεται ότι τα επιλεγμενα κύτταρα σπάνια ξεπερνούν τα 5 με 10. Ωστόσο, στην περίπτωση που τα επιλεγμένα κύτταρα είναι περισσότερα από 64, το σύστημα ενημερώνει τον χρήστη με ένα μήνυμα ότι ορισμένα κύτταρα δεν επιλέκτηκαν και τον συμβουλεύει να μεγενθύνει το σχέδιό του στην οθόνη και να ξαναπροσπαθήσει. Αποφασίσαμε να χρησιμοποιήσουμε ένα πίνακα για να κρατάμε τις ταυτότητες των επιλεγμένων κυττάρων. Οπως είναι φυσικό το σύστημα ξεχωρίζει τα κύτταρα μεταξύ τους δίνοντάς τους μοναδικές ταυτότητες (cell–ID’s). Οι ταυτότητες αυτές είναι ακέραιοι αριθμοί και συνεπώς ορίσαμε ένα πίνακα ακεραίων 64 θέσεων.

Η διαδικασία επιλογής κυττάρων δοθέντος ενός σημείου λειτουργεί ως εξής:

Δοθέντος του σημείου επιλογής

α. Δημιούργησε ένα BBOX γύρω από το σημείο αυτό (με διαστάσεις 2*pointingTolerance* × 2*pointingTolerance*)[6]

β. Ακύρωσε τα προηγούμενα περιεχόμενα του πίνακα επιλογών

γ. **Για κάθε** κύτταρο (παιδί) του "κυττάρου εργασίας"

   Εάν το BBOX του κυττάρου τέμνει το BBOX επιλογής **τότε**

   1. **Για κάθε** γεωμετρική πόρτα του κυττάρου

      Εάν η απόσταση της πόρτας από το σημείο επιλογής είναι μικρότερη ή ίση από *pointingTolerance* **τότε**

      i. Πρόσθεσε τη ταυτότητα του κυττάρου στο πίνακα επιλεγμένων

      ii. Πήγαινε στο βήμα (γ) και έλεγξε το επόμενο κύτταρο

   2. **Εάν** το βήμα (1) απέτυχε **τότε** κάλεσε την διαδικασία απόστασης του κυττάρου από το σημείο επιλογής

      Εάν η απόσταση αυτή είναι μικρότερη, ή ίση, από *pointingTolerance* **τότε**

      i. Πρόσθεσε τη ταυτότητα του κυττάρου στο πίνακα επιλεγμένων

      ii. Πήγαινε στο βήμα (γ) και έλεγξε το επόμενο κύτταρο.

Οπως βλέπουμε, ο μηχανισμός επιλογής επιτρέπει στον χρήστη να επιλέξει μόνο τα κύτταρα εκείνα που είναι άμεσα παιδιά του ψηλότερου στην ιεραρχία κυττάρου, δηλαδή του "κυττάρου εργασίας". Ο λόγος που γίνεται αυτό είναι, ότι απαγορεύουμε στο χρήστη να κάνει

[6] *pointingTolerance* είναι μία σταθερά ορισμένη σε πραγματικές συντεταγμένες – ίση με απόσταση 4 pixels, όταν μετατραπεί σε μονάδες συστήματος (απόσταση σε συντεταγμένες κόσμου).

συνδέσεις στις πόρτες άλλων κυττάρων εκτός αυτών που είναι άμεσα παιδιά του "κυττάρου εργασίας" που επεξεργάζεται. Ακόμα, ο αλγόριθμος ελέγχει τα παραπάνω κύτταρα, μόνο αν το ορθογώνιο περίγραμμά τους τέμνει το BBOX της περιοχής επιλογής.

Ετσι, φαίνεται καθαρά αυτό που αναφέραμε στην αρχή της παραγράφου, ότι δηλαδή ο μηχανισμός αυτός είναι αποδοτικός, εφόσον, με τον τρόπο αυτό, ολόκληρα υποδέντρα της ιεραρχίας αποκλείονται από την υπόλοιπη διαδικασία που είναι και η περισσότερο δαπανηρή. Η διαδικασία εύρεσης της απόστασης κυττάρου από δοθέν σημείο διασχίζει αναδρομικά όλα τα συνιστώντα κύτταρα μέχρι να φτάσει στα πρωτογενή κύτταρα που συνθέτουν το συγκεκριμένο κύτταρο και για κάθε ένα από αυτά καλεί την διαδικασία εύρεσης αποστασής από το σημείο αυτό. Συνήθως οι διαδικασίες αυτές είναι πολύπλοκες, και ως εκ τούτου χρονοβώρες, και γι' αυτό προσπαθούμε να αποφεύγουμε την κλήση τους. Για το λόγο αυτό, στο αλγόριθμο, κάνουμε πρώτα το βήμα (γ.1), με το οποίο ελέγχουμε αν οι γεωμετρικές πόρτες του προς επιλογή κυττάρου είναι κοντά στο σημείο επιλογής. Ο έλεγχος αυτός γίνεται με απλές διαδικασίες σύγκρισης των τιμών των συντεταγμένων, ώστε να είναι, κατά το δυνατό, γρηγορότερος.

2.3 Υλοποίηση

Η γλώσσα προγραμματισμού C++ [16], στην οποία είναι γραμμένος ο Λαβύρινθος, είναι οντοκεντρική (Object Oriented). Κάθε οντότητα (κλάση) έχει ως πεδία τόσο μεταβλητές (private and public data), όσο και διαδικασίες που τις χειρίζονται (member functions).

Τα ορθογώνια περιγράμματα των κυττάρων τα έχουμε ορίσει ως στιγμιότυπα της κλάσης BBOX. Η κλάση αυτή μαζί με τις δηλώσεις των μεταβλητών και των διαδικασιών που την ορίζουν παρουσιάζεται παρακάτω.

```
class BBOX {
    private:
        Wcoor xmin;
        Wcoor ymin;
        Wcoor xmax;
        Wcoor ymax;
    public:
        BBOX();
        void set();
        void set(Wcoor xMin, Wcoor yMin, Wcoor xMax, Wcoor yMax);
        void get(Wcoor *xMin, Wcoor *yMin, Wcoor *xMax, Wcoor *yMax);
        void expand_by(BBOX* bbox);
        BOOLEAN intersects(BBOX* bbox);
        BOOLEAN update_parent_bbox(BBOX* child_old_bbox, BBOX* child_new_bbox);
};
```

Ο τύπος *Wcoor* δηλώνει ότι οι πεδία *xmin, xmax, ymin, ymax* είναι συντεταγμένες κόσμου. Οι διαδικασίες–μέλη (member functions) αυτής της κλάσης είναι η εξής:

BBOX()

Η διαδικασία *BBOX()* είναι ο "κατασκευαστής" (constructor) στιγμιοτύπων της κλάσης

BBOX. Εκτός από το να δεσμεύει μνήμη για τα πεδία του στιγμιότυπου, δίνει και αρχικές τιμές σε αυτά. Οι αρχικές τιμές που παίρνει ένα στιγμιότυπο είναι "παράνομες" (illegal values), και γίνονται "νόμιμες" με την πρώτη κλήση της διαδικασίας *expand_by(...)*.

void set(), void set(Wcoor xMin, Wcoor yMin, Wcoor xMax, Wcoor yMax)

Η διαδικασία *void set(...)* δέχεται ως ορίσματα τις τιμές που θέλουμε να δώσουμε σε ένα BBOX (στιγμιότυπο που την καλεί) και θέτει τα κατάλληλα πεδία του. Η διαδικασία *void set()* (χωρίς ορίσματα) δίνει "παράνομες" αρχικές τιμές στα πεδία του στιγμιότυπου BBOX που την καλεί. Διαφέρει από τον "κατασκευαστή" στο ότι δεν δεσμεύει μνήμη.

*void get(Wcoor *xMin, Wcoor *yMin, Wcoor *xMax, Wcoor *yMax)*

Η διαδικασία αυτή επιστρέφει στα ορίσματά τις, τις τιμές που έχουν τα πεδία του στιγμιότυπου BBOX που την κάλεσε.

void expand_by(BBOX bbox)*

Η διαδικασία αυτή βρίσκει το ελάχιστο BBOX που περικλύει το BBOX που την καλεί και το BBOX που περνίεται ως παράμετρος, και ενημερώνει ανάλογα το πρώτο. Αυτή η διαδικασία διογκώνει ένα BBOX προσθέτοντάς του κάθε φορά και ένα άλλο.

BOOLEAN intersects(BBOX bbox)*

Η διαδικασία αυτή ελέγχει αν το BBOX που την καλεί τέμνεται με το BBOX που περνίεται ως παράμετρος, και επιστρέφει TRUE ή FALSE ανάλογα.

BOOLEAN update_parent_bbox(BBOX child_old_bbox, BBOX* child_new_bbox)*

Η διαδικασία αυτή, με παραμέτρους το παλαιό και το νέο BBOX ενός παιδιού, ελέγχει αν το BBOX του πατέρα (αυτός που την καλεί) πρόκειται να μεταβληθεί. Ενημερώνει το BBOX του πατέρα και επιστρέφει TRUE αν η ενημέρωση έχει επιτύχει. Αν η ενημέρωση αποτύχει σημαίνει ότι πρέπει να εκτελεστεί το βήμα (β) του αλγορίθμου που περιγράφεται στη παράγραφο 2.1.4.

Ο πλήρης κώδικας για την κλάση αυτή παραθέτεται στο παράρτημα Α.

3. Επανασχεδίαση

Στον Λαβύρινθο ένα κύτταρο επικοινωνεί με άλλα κύτταρα μέσω των πορτών του, στις οποίες ο χρήστης δίνει τιμές, ή τις οποίες "ενώνει" με πόρτες άλλων κυττάρων. Ο χρήστης, για να μεταβάλλει ένα κύτταρο, πρέπει να επιλέξει μια πόρτα του. Ετσι μπορεί να αλλάξει τα χαρακτηριστικά του κυττάρου που εξαρτώνται από αυτή, ή τον τρόπο με τον οποίο αυτό συνδέεται με άλλα κυτταρα. Οποιαδήποτε μεταβολή στις τιμές των πορτών ενος κυττάρου συνεπάγεται τον επανυπολογισμό του, ο οποίος μπορεί να προκαλέσει αλυσιδωτά τον επανυπολογισμό άλλων κυττάρων με τα οποία αυτό συνδέεται. Ετσι τελικά, τα κύτταρα του Λαβύρινθου μεταβάλλονται είτε άμεσα, όταν ο χρήστης αλλάξει διαλογικά την τιμή μιας πόρτας τους, είτε έμμεσα, μέσω του επανυπολογισμού μιας "αλυσίδας" κυττάρων.

3.1 Λίστες Γεγονότων και Επανασχεδίαση

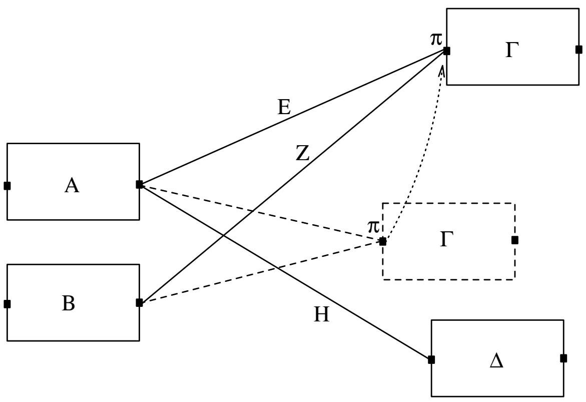
Οπως αναφέραμε σε προηγούμενη παράγραφο, για τα κύτταρα των οποίων οι πόρτες έχουν αλλάξει τιμή λέμε ότι έχουν δεχτεί ένα γεγονός, και τα γεγονότα αυτά τα κρατάμε σε μιά λίστα γεγονότων (λίστα επανυπολογισμού). Μέσω της λίστας αυτής το σύστημα επανυπολογίζει μόνο τα κύτταρα εκείνα που έχουν δεχτεί νέες τιμές (γεγονότα) σε μία ή περισσότερες πόρτες τους. Ο μηχανισμός επανυπολογισμού των κυττάρων παρουσιάστηκε στην παράγραφο 1.1.4. Στην παράγραφο αυτή μας ενδιαφέρει το πώς ο επανυπολογισμός των κυττάρων επηρεάζει την γραφική τους αναπαράσταση στην οθόνη.

3.1.1 Λίστα Επανασχεδίασης

Ο επανυπολογισμός ενός κυττάρου, συνήθως, συνεπάγεται την αλλαγή της γραφικής του αναπαράστασης. Παραδείγματος χάριν, η αλλαγή της τιμής μιας πόρτας τύπου σημείου συνεπάγεται αυτόματα την αλλαγή της αναπαράστασης του κυττάρου στην οθόνη, εφόσον ένα σημείο του άλλαξε. Ανάλογα, η αλλαγή της τιμής μιας πόρτας, που αναπαριστά το χρώμα σε ένα κύτταρο, συνεπάγεται ότι το κύτταρο πρέπει να αλλάξει χρώμα. Αρα, είναι φανερό ότι ορισμένα από τα κύτταρα που μεταβλήθηκαν (επανυπολογίστηκαν) πρέπει να επανασχεδιαστουν (λίστα επανασχεδιασμού). Γι’ αυτό, παράλληλα με την λίστα επανυπολογισμού, πρέπει να κρατάμε και μία λίστα για τα κύτταρα που πρόκειται να επανασχεδιαστούν.

Οι λίστες αυτές είναι διαφορετικές, για λόγους ταχύτητας. Στην λίστα επανυπολογισμού μπορεί να υπάρχουν και αόρατα κύτταρα, τα οποία υπολογίζονται αλλά δεν έχουν γραφική αναπαράσταση, και έτσι δεν χρειάζεται να βρίσκονται στην λίστα επανασχεδίασης. Ενας άλλος λόγος είναι ότι από τον επανασχεδιασμό των κυττάρων που επανυπολογίστηκαν επηρεάζονται γραφικά και άλλα κύτταρα, όπως θα δείξουμε παρακάτω, των οποίων οι πόρτες τους δεν έχουν αλλάξει τιμή, άρα προφανώς δεν ανήκουν στην λίστα επανυπολογισμού.

Η λίστα επανασχεδίασης δημιουργείται σταδιακά κατά την διάρκεια της διαδικασίας επανυπολογισμού των κυττάρων. Κάθε ορατό κύτταρο, που η διαδικασία υπολογισμού του επιστρέφει με επιτυχία (άρα ο υπολογισμός του έγινε σωστά), μπαίνει στην λίστα επανασχλδίασης, αν δεν βρίσκεται ήδη σ' αυτήν. Με το τέλος της διαδικασίας επανυπολογισμού των κυττάρων, ο μηχανισμός επανασχεδίασης πρέπει να διατρέξει την λίστα αυτή, και για κάθε ένα από τα στοιχεία της, να καλέσει την διαδικασία σχεδίασής του.



Σχήμα 3.1: Επανυπολογισμός και επανασχεδίαση κυττάρων

3.1.2 Παράδειγμα

Στο σχήμα 3.1 έχουμε τέσσερα σύνθετα κύτταρα (Α, Β, Γ, Δ), με πόρτες τα σημεία που φαίνονται τονισμένα στο σχέδιο, και τα "ευθύγραμμα τμήματα" Ε, Ζ και Η. Κάθε ένα από τα Α, Β, Γ και Δ αποτελείται από τέσσερα "ευθύγραμμα τμήματα" και ορισμένα αόρατα κύτταρα που ορίζουν ότι αυτά δημιουργούν ένα ορθογώνιο παραλληλόγραμμο, καθώς και ότι οι πόρτες του κυττάρου βρίσκονται στο μέσο των κάθετων πλευρών.

Ο χρήστης μπορεί να δημιουργήσει ένα γεγονός μετακινώντας την πόρτα π του κυττάρου Γ σε νέα θέση. Κατά την διάρκεια της διαδικασίας επανυπολογισμού του Γ, το γεγονός θα μεταδωθεί στις πόρτες των Ε και Ζ που είναι συνδεδεμένες με την πόρτα π του Γ. Ετσι τα Ε και Ζ θα μπούνε στην λίστα επανυπολογισμού. Μετά το τέλος της διαδικασίας επανυπολογισμού τα κύτταρα Γ, Ε και Ζ βρίσκονται στην λίστα επανασχεδίασης και πρέπει να επανασχεδιαστούν στις νέες τους θέσεις.

3.1.3 Απλοϊκή Επανασχεδίαση

Προφανώς, πριν σχεδιάσουμε τα κύτταρα που μεταβλήθηκαν στις νέες τους θέσεις, πρέπει να τα σβήσουμε από τις παλαιές. Ενας τρόπος για να το πετύχουμε είναι να επανασχεδιάσουμε τα κύτταρα αυτά στις παλαιές θέσεις τους, με "άσπρο" χρώμα. Αυτό απαιτεί να κρατάμε, για κάθε κύτταρο στην λίστα επανασχεδίασης, μαζί με την νέα θέση του, και την παλαιά. Ετσι, διατρέχουμε την λίστα επανασχεδίασης μία φορά για να σβήσουμε τα κύτταρα από τις παλαιές θέσεις τους και μία δεύτερη για να τα σχεδιάσουμε στις νέες.

Ο παραπάνω τρόπος έχει τρία ελλατώματα. Πρώτον, διατρέχουμε την λίστα δύο φορές, στέλνοντας στον X server δύο μηνύματα σχεδίασης για κάθε κύτταρο, διπλασιάζοντας έτσι τα μηνύματα σχεδίασης της εφαρμογής μας. Δεύτερον, σπαταλάμε χώρο για να θυμόμαστε τις παλαιές συντεταγμένες κάθε κυττάρου. Τρίτον υπάρχει η πιθανότητα να αφήνουμε "άσπρα" ίχνη πάνω σε άλλα κύτταρα του σχεδίου που δεν άλλαξαν θέση.

Στο παράδειγμα του σχήματος 3.1 σβήνοντας με "άσπρο" χρώμα το κύτταρο Ζ από την παλαιά του θέση αφήνουμε ένα "άσπρο" ίχνος πάνω στο κύτταρο Η, στο σημείο που παλαιότερα τέμνονταν τα δύο κύτταρα. Αν για να αποφύγουμε την διπλή διασχιση της λίστας επανασχεδίασης, κάθε κύτταρο στην λίστα πρώτα το σβήνουμε και μετά το σχεδιάζουμε στην νέα θέση, η πιθανότητα δημιουργίας "άσπρων" ιχνών αυξάνεται. Τότε, είναι πιθανό να αφήσουμε "άσπρα" ίχνη, κατα το σβήσιμο, όχι μόνο στα κύτταρα που δεν έχουν αλλάξει θέση, αλλά και σε όσα προστίθενται σταδιακά στο σχέδιο (στα κύτταρα της λίστας που μόλις επανασχεδιάστηκαν).

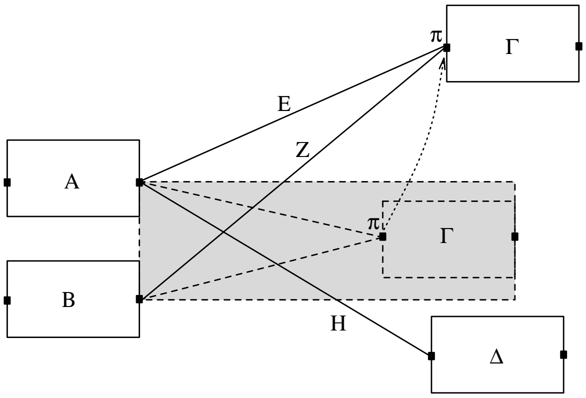
Για να διορθωθεί το πρόβλημα των "άσπρων" ιχνών, το οποίο είναι σοβαρό σε περιπτώσεις που πρέπει να σβηστούν κύτταρα των οποίων η γραφική αναπαράσταση καλύπτει ολόκληρη επιφάνεια όπως η "ορθωγώνια παραλληλόγραμη επιφανεια" (βλέπε παράρτημα Β), πρέπει να σβήνουμε τα "άσπρα" ίχνη, επανασχεδιάζοντας τα κύτταρα που "χάλασαν". Αυτό προϋποθέτει ότι υπάρχει ένας μηχανισμός με τον οποίο το σύστημα εντοπίζει τα κύτταρα αυτά. Στη παρακάτω παράγραφο παρουσιάζεται ο μηχανισμός αυτός καθώς και ο τρόπος με τον οποίο κάνουμε την επανασχεδίαση περισσότερο αποδοτική.

3.2 Αποδοτικότερη Επανασχεδίαση

3.2.1 Ορθογώνια Περιγράμματα και Επανασχεδίαση

Για να βρίσκουμε ποιά από τα κύτταρα του σχεδίου επηρεάζονται από τον μηχανισμό επανασχεδίασης, και συγκεκριμένα από τη διαδικασία σβησίματος των κυττάρων από την προηγούμενη θέση τους, θα πρέπει να μπορούμε να προσδιορίσουμε την περιοχή του σχεδίου την οποία καταλαμβάνουν τα κύτταρα που πρόκειται να αλλάξουν θέσεις. Ο μηχανισμός των ορθογωνίων περιγραμμάτων προσφέρεται για τον σκοπό αυτό: την περιοχή αυτή μπορούμε να την προσδιορίσουμε κατά προσέγγιση βρίσκοντας το ελάχιστο ορθογώνιο περίγραμμα το οποίο περικλύει όλα τα κύτταρα που πρόκειται να μεταβληθούν.

Το περίγραμμα αυτό προκύπτει "προσθέτοντας" τα ορθογώνια περιγράμματα όλων των πρωτογενών κυττάρων της λίστας επανυπολογισμού. Από αυτό έχουμε μια πρώτη προσέγγιση της περιοχής που επηρεάζεται από το σβήσιμο των κυττάρων που άλλαξαν θέσεις. Τα κύτταρα, που τυχαίνει να βρίσκονται ολόκληρα, ή μέρη αυτών, στην περιοχή αυτή, θα πρέπει να επανασχεδιαστούν, γιατί είναι πιθανό να έχουν "άσπρα" ίχνη. Την περιοχή αυτή την ονομάζουμε "περιοχή σβησίματος" (erase area). Στο σχήμα 3.2 βλέπουμε την "περιοχή σβησίματος" για το παράδειγμα του σχήματος 3.1.



Σχήμα 3.2: Περιοχή που επηρεάζεται από τον μηχανισμό επανασχεδίασης

Ζητώντας από τον X server να σβήσει την "περιοχή σβησίματος" με την εντολή: σβήσε ένα ορθογώνιο παραλληλόγραμμο στην οθόνη, πετυχαίνουμε με μια μόνο εντολή να σβήσουμε τα κύτταρα που πρόκειται να αλλάξουν θέση, οπότε μειώνουμε σημαντικά τον αριθμό τον εντολών σχεδίασης προς τον X server, και αποφεύγουμε την διπλή διάσχιση της λίστας επανασχεδίασης. Παράλληλα βέβαια, σβήνουμε και άλλα κύτταρα που δεν άλλαξαν θέση, αλλά βρισκόνταν στην περιοχή αυτή. Ομως τα κύτταρα αυτά επρόκειτο να επανασχεδιαστούν λόγω του προβλήματος των "άσπρων" ιχνών, και έτσι δεν κάνουμε τίποτα το περιττό.

3.2.2 Μηχανισμός επανασχεδίασης

Ετσι συμπληρώσαμε τον αλγόριθμο της παραγράφου 2.1.3, ώστε να δημιουργεί την "περιοχή σβησίματος", καθώς και την λίστα επανασχεδίασης για την οποία μιλήσαμε στην προηγούμενη παράγραφο, ως εξής:

Για κάθε κύτταρο στην λίστα επανυπολογισμού:

Εάν το κύτταρο είναι πρωτογενές **τότε**

α. Κάλεσε την διαδικασία υπολογισμού του (επανυπολόγισέ το) και πρόσθεσέ το στην λίστα επανασχεδίασης

β. "Πρόσθεσε" το ορθογώνιο περίγραμμά του στην "περιοχή σβησίματος"

γ. **Εάν** το κύτταρο είναι ορατό **τότε** κάλεσε την διαδικασία υπολογισμού του νέου ορθογωνίου περιγράμματός του (υπολόγισε το περίγραμμά του)

Εάν το κύτταρο είναι σύνθετο **τότε**

(όμοια με την παράγραφο 2.1.3) ...

Αντίστοιχα ο μηχανισμός επανυπολογισμού και επανασχεδίασης περιγράφεται από τον παρακάτω αλγόριθμο:

α. Δώσε αρχική τιμή στην "περιοχή σβησίματος"

β. Επανυπολόγισε τα κύτταρα στην λίστα επανυπολογισμού (αλγόριθμος της παραγράφου 2.1.3)

γ. Σβήσε τα περιεχόμενα της "περιοχής σβησίματος" από την οθόνη

δ. Ορισε την "περιοχή σβησίματος" ως "περιοχή ψαλιδίσματος" (clipping area)

ε. **Για κάθε** κύτταρο στη ιεραρχία των κυτταρων του σχεδίου

  Εάν το ορθογώνιο περίγραμμά του τέμνει την "περιοχή σβησίματος" **και** δεν ανήκει στην λίστα επανασχεδίασης **τότε**

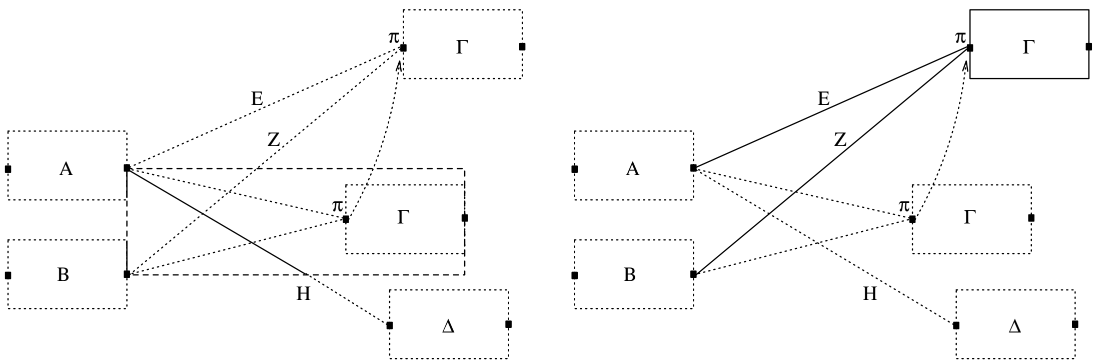
    Κάλεσε την διαδικασία επανασχεδίασης του

στ. Ακύρωσε την "περιοχή ψαλιδίσματος" (ορισέ να είναι ολόκληρο το σχεδιαστικό παράθυρο του συστήματος)

ζ. **Για κάθε** κύτταρο την λίστα επανασχεδίασης

  Κάλεσε την διαδικασία επανασχεδίασης του.

Στον παραπάνω αλγόριθμο η "περιοχή σβησίματος" χρησίμευει τόσο στο βήμα (γ), όσο και στο βήμα (ε). Στο βήμα (δ) ορίζουμε τη περιοχή αυτή να είναι η "περιοχή ψαλιδίσματος" (clipping area) του συστήματος. Με την εντολή αυτή ενημερώνουμε τον X server να σχεδιάζει μόνο μέσα στην περιοχή αυτή, "ψαλιδίζοντας" οτιδήποτε πρόκειται να σχεδιαστεί έξω αυτή. Ετσι, αν τα κύτταρα που πρόκειται να σχεδιαστούν στο βήμα (ε) έχουν κάποιο μέρος τους μέσα στην "περιοχή ψαλιδίσματος" και κάποιο έξω από αυτήν, επανασχεδιάζεται μόνο το μέρος εκείνο που βρίσκεται μέσα στην περιοχή. Ετσι, όπως φαίνεται και στο σχήμα 3.3 (αριστερά), στο βήμα (ε) επανασχεδιάζονται μέρη μόνο από τα κύτταρα Α, Β και Η. Δεξιά φαίνεται το βήμα (ζ), όπου επανασχεδιάζονται τα κύτταρα Γ, Ε και Ζ (στοιχεία της λίστας επανασχεδίασης). Με τον τρόπο αυτό, επιτυγχάνουμε ο X server να κάνει όσο το δυνατό λιγότερες πράξεις σχεδίασης, βελτιώνοντας έτσι τον χρόνο απόκρισης του συστήματος.



Σχήμα 3.3: Διαδοχικά στάδια του μηχανισμού επανασχεδίασης

Πολλές φορές συμβαίνει τα κύτταρα που πρέπει να επανασχεδιαστούν λόγω επανυπολογισμού να βρίσκονται διάσπαρτα στην οθόνη (μακριά το ένα από το άλλο). Αυτό έχει σαν συνέπεια η "περιοχή σβησίματος" να είναι κατά πολύ μεγαλύτερη από αυτή που πραγματικά χρειάζεται να σβηστεί. Μια λύση για το παραπάνω πρόβλημα θα ήταν να προσεγγίσουμε την περιοχή αυτή με πολλές "περιοχές σβησίματος". Κάθε τέτοια περιοχή θα την αποτελούσαν κύτταρα που βρίσκονταν κοντά το ένα με το άλλο. Η λύση αυτή δεδομένου του ότι τοπολογικά τα κύτταρα βρίσκονται σε θέσεις όπου μπορούν ομαδοποιηθούν ανά περιοχές, είναι περισσότερο ακριβής από αυτή που παρουσιάσαμε παραπάνω. Το πρόβλημα της όμως είναι οτι ο μηχανισμός επανυπολογισμού θα χρειαζόταν να σπαταλήσει χρόνο προκειμένου να ομαδοποιήσει κατά ένα αξιόπιστο τρόπο τα κύτταρα που πρόκειται να σβηστούν. Για τον λόγο αυτό και επιπλέον γιατί θεωρούμε οτι το παραπάνω πρόβλημα δε θα εμφανίζεται συχνά, χρησιμοποιούμε μία "περιοχή σβησίματος" στο Λαβύρινθο.

3.3 Μηχανισμός Δημιουργίας κώδικα PostScript

Ο Λαβύρινθος, όπως κάθε σχεδιαστικό σύστημα, πρέπει να δίνει στον χρήστη τη δυνατότητα να εκτυπώνει τα σχέδιά του στον εκτυπωτή. Η συνήθης διαδικασία για κάτι τέτοιο είναι η δημιουργία ενός αρχείου που περιέχει το σχέδιο σε κώδικα PostScript[7] [6]. Ετσι ο χρήστης μπορεί να εκτυπώσει το σχέδιό του σε ένα εκτυπώτη ή να το συμπεριλάβει ως εικόνα σε ένα κείμενό του.

Ο Λαβύρινθος έχει έναν τέτοιο μηχανισμό για την αναπαράσταση του σχεδίου σε κώδικα PostScript: κάθε πρωτογενές κύτταρο, εκτός από τις διαδικασίες υπολογισμού και σχεδίασής του, έχει και μία διαδικασία, η οποία αναλαμβάνει να εισάγει σε ένα αρχείο εντολές PostScript που περιγράφουν το πρωτογενές κύτταρο. Για τα σύνθετα κύτταρα, ο μηχανισμός αυτός διατρέχει την ιεραρχία των συνιστώντων κυττάρων, καλώντας κάθε φορά την διαδικασία δημιουργίας του κώδικα. Ετσι, κάθε σύνθετο κύτταρο περιγράφεται στο αρχείο αυτό από τις περιγραφές (εντολές PostScript) των κυττάρων που το συνθέτουν. Οι συντεταγμένες που το σύστημα χρησιμοποιεί για την περιγραφή αυτή είναι οι συντεταγμένες κόσμου των κυττάρων.

3.3.1 Η ειδική περίπτωση του κυττάρου TextSize

Τη διαδικασία δημιουργίας κώδικα PostScript δυσκολεύει η ύπαρξη κειμένου μέσα στο σχέδιο. Συγκεκριμένα, για να τυπώσουμε ένα κείμενο στην οθόνη χρησιμοποιούμε ορισμένες οικογένειες γραφής σε συγκεκριμένα μεγέθη (Times-Roman, Times-Italic, με point-size 8, 10, 12) που παρέχονται από το περιβάλλον X. Αυτές οι οικογένειες γραφής είναι τυποποιημένες (standard), υπάρχουν μόνο σε ορισμένα μεγέθη, και είναι φτιαγμένες μόνο για την οθόνη (screen fonts). Αντίστοιχα, υπάρχουν τυποποιημένες οικογένεις γραφής για εκτυπωτές (PostScript fonts). Σε αυτές δεν υπάρχει περιορισμός ως προς τα μεγέθη. Δυστυχώς, οι διαστάσεις στην οθόνη ενός κειμένου συγκεκριμένης οικογένειας γραφής και μεγέθους (π.χ. Times-Roman 10), δεν είναι συμβατές με τις διαστάσεις πού έχει το ίδιο κείμενο στην εκτύπωση.

[7] Γλώσσα περιγραφής εικόνων και σχεδίων που υποστηρίζεται από πολλούς σύγχρονους εκτυπωτές.

Ο Λαβύρινθος χρησιμοποιεί τα πρωτογενή κύτταρα "κείμενο" και "μέγεθος κειμένου"[8] (text και textSize), για να σχεδιάζει ένα κείμενο και να υπολογίζει το μέγεθός του στην οθόνη. Ο χρήστης μπορεί να συνδέσει ένα κύτταρο "μέγεθος κειμένου" με άλλα κύτταρα με τέτοιο τρόπο ώστε οι θέσεις τους να εξαρτώνται από το μέγεθος του κειμένου. Ειναι προφανές ότι στην εκτύπωση οι συντεταγμένες των κυττάρων αυτών θα είναι λανθασμένες: δεν θα συμπίπτουν με εκείνες του "μεγέθους κειμένου", αφού το κείμενο έχει άλλες διαστάσεις στην οθόνη και άλλες στην εκτύπωση.

Γι’ αυτό, ο μηχανισμός δημιουργίας αρχείου PostScript γνωρίζοντας ότι ανά πάσα στιγμή οι συντεταγμένες των κυττάρων είναι σωστές για την γραφική αναπαράσταση του σχεδίου μόνο στην οθόνη, πρέπει πρώτα να επανυπολογίσει τα "μεγέθη κειμένων" για τον εκτυπωτή. Αυτό θα προκαλέσει τον αλυσιδωτό επανυπολογισμό των κυττάρων του σχεδίου που συνδέονται με τα κύτταρα αυτά, και μόνο αφού αυτός ολοκληρωθεί το σύστημα μπορεί να δημιουργήσει το αρχείο. Μετά, τα κύτταρα αυτά πρέπει να επανυπολογιστούν, για να επανέλθουν οι παλαιές τιμές τους, για την οθόνη. Ετσι, ορίσαμε η διαδικασία υπολογισμού του "μεγέθους κειμένου" να αποτελείται από δύο μέρη: ένα που υπολογίζει τις διαστάσεις του κειμένου για την οθόνη και ένα που υπολογίζει τις διαστάσεις για την εκτύπωση. Η διαδικασία αυτή ελέγχει πρώτα μια μεταβλητή (device) και υπολογίζει ανάλογα τις διαστάσεις του κειμένου. Ετσι καταλήξαμε στο παρακάτω αλγόριθμο για τη δημιουργία κώδικα PostScript:

α. Θέσε την μεταβλητή device ίση με ΕΚΤΥΠΩΤΗΣ

β. Δημιούργησε γεγονότα σε όλα τα "μεγέθη κειμένων" του σχεδίου

γ. Κάλεσε την διαδικασία επανυπολογισμού (παράγραφος 2.1.4)

δ. **Για κάθε** ορατό κύτταρο στην ιεραρχία των κυττάρων του σχεδίου

Κάλεσε την διαδικασία δημιουργίας κώδικα PostScript

ε. Θέσε την μεταβλητή device ίση με ΟΘΟΝΗ

στ. Δημιούργησε γεγονότα σε όλα τα "μεγέθη κειμένων" του σχεδίου

ζ. Κάλεσε την διαδικασία επανυπολογισμού (παράγραφος 2.1.4).

Στον αλγόριθμο αυτό επανυπολογίζονται μόνο εκείνα τα κύτταρα που επηρεάζονται άμεσα ή έμμεσα από τα "μεγέθη κειμένων", ελαχιστοποιώντας έτσι τους επανυπολογισμούς που πρέπει να κάνει το σύστημα. Στο βήμα (γ) αλλάζουν οι συντεταγμένες στις εσωτερικές δομές του συστήματος, ώστε αυτές να είναι οι κατάλληλες για την εκτύπωση, ενώ στο βήμα (ζ) επαναφέρονται οι συντεταγμένες αυτές στις αρχικές τους τιμές.

[8] Πρωτογενή κύτταρα για χειρισμό κειμένου. Το κύτταρο "κείμενο" τυπώνει ένα κείμενο αρχίζοντας από δοθέν σημείο. Το κύτταρο "μέγεθος κειμένου" δεν τυπώνει τίποτα, αλλά υπολογίζει το μέγεθος που θα έχει το δοθέν κείμενο όταν τυπωθεί. Ακόμα υπολογίζει που πρέπει να αρχίζει η γραμμή βάσης (base-line) του κειμένου, όταν αυτό είναι κεντραρισμένο ή στοιχισμένο αριστερά (κλπ) σε σχέση με δοθέν σημείο. Ο λόγος που έχουμε το "μέγεθος κειμένου" ως διαφορετικό κύτταρο είναι γιατι μπορεί να είναι απαραίτητο να υπολογίστει το μέγεθος ενός κειμένου χωρίς αυτό να τυπωθεί, π.χ. όταν θέλουμε να υπολογίσουμε το μεγαλύτερο από κάποια κείμενα και μετά να αποφασίσουμε που θα τυπωθούν αυτά.

3.4 Widgets ως Κύτταρα

Ενα από τα σημαντικά θέματα που αντιμετωπίσαμε κατά τον σχεδιασμό του Λαβύρινθου, ήταν το τι πρωτογενή κύτταρα πρέπει να παρέχει το σύστημα στο χρήστη, υπό την μορφή "βιβλιοθήκης". Τα πρώτογενή κύτταρα που είναι διαθέσιμα αυτή την στιγμή είναι "ευθύγραμμα τμήματα", "τόξα", "κύκλοι", "κείμενο", "μέγεθος κειμένου", διάφορα αόρατα κύτταρα που αναπαραριστούν γεωμετρικές σχέσεις, κ.α. (βλέπε παράρτημα Β). Επιπλέον αυτών, ο Λαβύρινθος πρέπει να επιτρέπει την δημιουργία πρωτογενών κυττάρων με σύνθετο σχεδιαστικό περιεχόμενο και σύνθετη λειτουργικότητα, όπως τα widgets. Στην παράγραφο αυτή αναπτύσσονται διάφορες σκέψεις σχετικά με το τρόπο που πρέπει ο μελλοντικός σχεδιαστής πρωτογενών κυττάρων να σχεδίαζει τα κύτταρα–widgets και τι πρέπει να προσέξει κατά την διαδικασία αυτή, καθώς και σκέψεις πάνω στους μηχανισμούς που ο Λαβύρινθος πρέπει να παρέχει για την υποστήριξη των κυττάρων–widgets.

3.4.1 Μελέτη της Λειτουργικότητας των Widgets

Τα widgets είναι αυτόνομες οντότητες με συγκεκριμένη λειτουργικότητα, τις οποίες ο προγραμματιστής μπορεί να χρησιμοποιεί, διαμορφώνοντάς τις κατάλληλα για να εξυπηρετούν τις ανάγκες ενός προγράμματος. Για τον σκοπό αυτό, χρείαζεται να γραφτεί πηγαίος κώδικας που να διαμορφώνει κατάλληλα την συμπεριφορά τους.

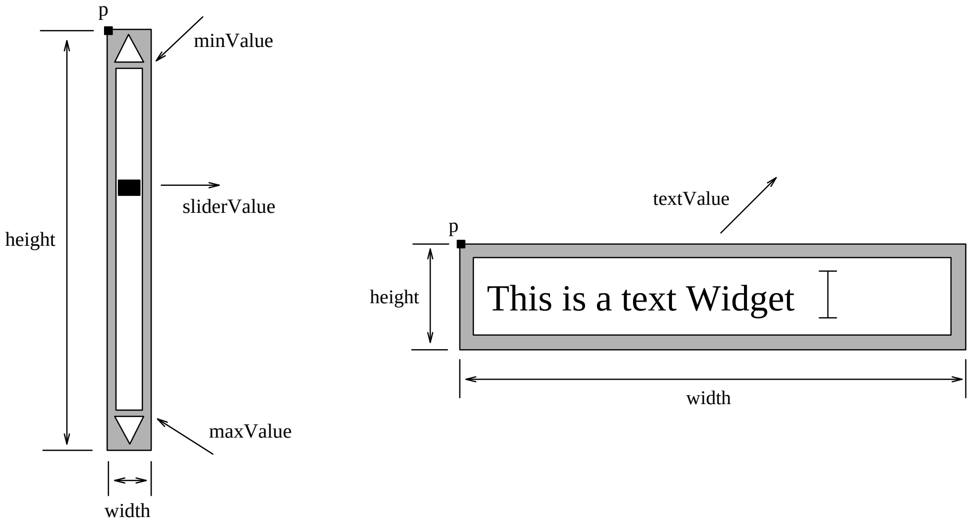
Για να ορίσει ο σχεδιαστής των πρωτογενών κυττάρων ένα widget ως κύτταρο, θα πρέπει να γνωρίζει καλά την λειτουργικότητά του. Εχοντας αυτή ως δοδεμένη, πρέπει να ορίσει τον τρόπο επικοινωνίας του νέου κυττάρου–widget με άλλα κύτταρα, καθώς και τον τρόπο με τον οποίο ο χρήστης που δεν γνωρίζει λεπτομέρειες υλοποίησης μπορεί να αλλάζει τα χαρακτηριστικά του. Δηλαδή, ο σχεδιαστής πρέπει να ορίσει ποιές θα είναι οι πόρτες του κυττάρου–widget και σε ποιά χαρακτηριστικά του widget αυτές αντιστοιχούν.

Δεδομένου ότι τα χαρακτηριστικά ενός widget είναι πολλά και διαμορφώνονται μέσω συγκεκριμένων διαδικασιών της βιβλιοθήκης Xtoolkit [8], και Xm [10], ο σχεδιαστής πρέπει να δώσει σταθερές τιμές σε κάποια από αυτά προκειμένου να αποφύγει να ορίσει μία πόρτα του κυττάρου–widget για κάθε ένα χαρακτηριστικό του. Ακόμα, επειδή η λειτουργικότητα ενός widget είναι προσβάσιμη μόνο μέσω διαδικασιών–callbacks και χειριστών γεγονότων (event handlers), ο σχεδιαστής θα πρέπει να ορίσει τις διαδικασίες και τους χειριστές γεγονότων που είναι απαραίτητες για την καλή λειτουργία του κυττάρου–widget.

3.4.2 Παράδειγμα

Παρακάτω θα δούμε ένα παράδειγμα ορισμού ενός νέου κυττάρου–widget και θα παρουσιάσουμε σταδιακά τα διάφορα προβλήματα που προκύπτουν.

Στο σχήμα 3.4 βλέπουμε δύο Motif widgets, το ScrollBar (αριστερά) και το Text (δεξιά). Στο πρώτο ο προγραμματιστής αρκεί να δώσει τις διαστάσεις (height, width) και την θέση (σημείο p) που αυτό θα τοποθετηθεί, καθώς και μια μέγιστη (maxValue) και μια ελάχιστη τιμή (minValue). Η οντότητα αυτή είναι έτσι ορισμένη, ώστε καθώς ο χρήστης μετακινεί με το ποντίκι την λαβή (slider) να παράγεται μια νέα τιμή (sliderValue) ανάμεσα στις τιμές: maxValue και minValue, κατ' αντιστοιχία προς την θέση της λαβής ανάμεσα στις δύο άκρες. Αντιστοιχη λειτουργία έχουμε,



Σχήμα 3.4: Ορισμός οντοτήτων widgets ως πρωτογενή κύτταρα (κύτταρα–widgets)

όταν ο χρήστης πατήσει το ποντίκι πάνω σε ένα από τα δύο βέλη: η λαβή μετακινείται προς την μεριά που δείχνει το βέλος. Με οποιαδήποτε μετακίνηση της λαβής καλούνται διαδικασίες–callbacks που ορίζονται από τον προγραματιστή και ενημερώνουν το πρόγραμμα για την νέα τιμή της.

Ανάλογη λειτουργία έχει και το Text widget. Εδώ, ο προγραμματιστής πρέπει να ορίσει την θέση και τις διαστάσεις του widget καθώς και διαδικασίες–callbacks που θα καλούνται όταν ο χρήστης πληκτρολογεί έχοντας το ποντίκι πάνω στο widget αυτό, και με τις οποίες το πρόγραμμα ενημερώνεται για το κείμενο που πληκτρολογήθηκε (textValue).

3.4.3 Οι Πόρτες του Κυττάρου–widget

Ο σχεδιαστης κυττάρων–widgets αρκεί να ορίσει ως πόρτες αυτές που φαίνονται στο σχήμα. Ακόμα πρέπει να ορίσει τις διαδικασίες–callbacks, που ενημερώνονται για τις τιμές της λαβής και του κειμένου αντίστοιχα στα δύο παραπάνω παραδείγματα, ώστε να δίνουν νέες τιμές στις αντίστοιχες πόρτες των κυττάρων–widgets και να δημιουργούν γεγονότα σ’ αυτές. Ετσι το σύστημα θα μπορεί να επανυπολογίσει τα κύτταρα που συνδέονται με τις πόρτες αυτές, ενεργοποιώντας το μηχανισμό επανυπολογισμού. Ακόμα ο σχεδιαστής μπορεί να ορίσει και άλλες πόρτες για το κύτταρο–widget, όπως διάφορα σημεία του κυττάρου–widget, οι τιμές των οποίων θα παράγονται ως έξοδοι μετά από κάθε επανυπολογισμό του κυττάρου.

Μέχρι τώρα για να μεταβάλλει, ο χρήστης, την πόρτα ενός κυττάρου έπρεπε να την επιλέξει και με μια κατάλληλη εντολή να της δώσει νέα τιμή, ενεργοποιώντας έτσι το μηχανισμό επανυπολογισμού των κυττάρων. Ωστόσο, οι πόρτες των κυττάρων–widgets μπορεί να δέχονται νέες τιμές χωρίς να επιλεγτούν, όπως την περίπτωση του ScrollBar widget όπου η τιμή της πόρτας sliderValue μεταβάλλεται καθώς μετακινείται η λαβή. Γι’ αυτό ο σχεδιαστής πρέπει να ενεργοποιεί τον μηχανισμό επανυπολογισμού μέσα από τις διαδικασίες–callbacks των κυττάρων–widgets. Αυτό δεν επηρεάζει τους υπόλοιπους μηχανισμούς του συστήματος, εκτός – ίσως – από εκείνον της επιλογής, για τον οποίο θα μιλήσουμε παρακάτω με λεπτομέρεια.

3.4.4 Οι Διαδικασίες του Κυττάρου–widget

Ο σχεδιαστής των πρωτογενών κυττάρων, αφού τελειώσει με την μελέτη της λειτουργικότητας του κυττάρου–widget θα πρέπει να ορίσει όλες τις επιμέρους διαδικασίες που είναι απαραίτητες σε ένα πρωτογενές κύτταρο:

- Ο **"κατασκευαστής" στιγμιοτύπων** (constructor) είναι μία τέτοια διαδικασία, με την οποία ο σχεδιαστής ορίζει τις πόρτες του κυττάρου, το όνομα της κάθε πόρτας, τις τιμές που αυτές επιτρέπεται να δέχονται, τι ιδιότητες θα έχουν, καθώς και τις αρχικές τιμές τους.

  Εδώ ο σχεδιαστής θα πρέπει να δημιουργεί το widget και να δίνει τιμές σε χαρακτηριστικά του που δεν είναι προσπελάσιμα από τις πόρτες του. Ακόμα θα πρέπει να φυλάσσει σε μία "ιδιωτική μεταβλητή" (private data του κυττάρου) την ταυτότητα του widget (widget–ID). Μέσω αυτής το σύστημα θα μπορεί να χειρίζεται το widget και να επικοινωνεί με αυτό.

  Τέλος όλες οι άλλες εσωτερικές δομές του νέου πρωτογενούς κυττάρου θα μείνουν ως έχουν για τα υπόλοιπα πρωτογενή κύτταρα.

- Μια άλλη **διαδικασία** είναι αυτή της **εύρεσης του ορθογωνίου περιγράμματος** του κυττάρου–widget. Τα widgets είναι τις περισσότερες φορές ορθογώνια παράθυρα και έτσι αρκεί ο σχεδιαστής να ορίσει την διαδικασία αυτή ώστε να υπολογίζει τις διαστάσεις του παραθύρου του widget, να τις μετατρέπει σε συντεταγμένες κόσμου και να ενημερώνει το ορθογώνιο περίγραμμά του.

- Στην **διαδικασία υπολογισμού** του κυττάρου ο σχεδιαστής θα πρέπει να υπολογίζει τις τιμές των πορτών εξόδου του κυττάρου με βάση τις τιμές των πορτών εισόδου που έχουν δεχτεί γεγονότα. Η διαδικασία αυτή μπορεί να συνεργάζεται με την διαδικασία σχεδίασης του κυττάρου, όπως περιγράφεται αμέσως παρακάτω.

- Στην **διαδικασία σχεδίασης**, ο σχεδιαστής πρέπει να διαβάζει τις τιμές των πορτών του κυττάρου και να αλλάζει τα αντίστοιχα χαρακτηριστικά του κυττάρου. Η αλλαγή αυτή, συνήθως, προκαλεί την αυτόματη επανασχεδίαση του widget. Ισως η αλλαγή των χαρακτηριστικών του widget θα χρειαστεί να γίνεται στη διαδικασία υπολογισμού του, γιατί κάποιες από τις πόρτες εξόδου μπορεί να εξαρτώνται από τα χαρακτηριστικά αυτά.

  Για να αποφεύγει ο σχεδιαστής την αυτόματη επανασχεδίαση του widget θα πρέπει, μέσα στην διαδικασία υπολογισμού, να το σβήνει από την οθόνη (δίνοντας την εντολή XtUnmapWidget στον X server) και να το επανεμφανίζει – με τα νέα χαρακτηριστικά του – στην διαδικασία επανασχεδίασης (με την εντολή XtMapWidget).

- Τέλος, η **διαδικασία δημιουργίας κώδικα PostScript** είναι η δυσκολότερη να υλοποιηθεί από τον σχεδιαστή των πρωτογενών κυττάρων. Η γραφική αναπαράσταστη ενός κυττάρου–widget είναι σύνθετη και πραγματοποιείται με περισσότερες από μία εντολές σχεδίασης προς τον X server. Ο μόνος τρόπος για να περιγραφεί το κύτταρο–widget με εντολές PostScript είναι ως εικόνα. Για τον λόγο αυτό, θα πρέπει να αναπτύξουμε ένα μηχανισμό, ο οποίος θα ενεργοποιείται από κάθε τέτοια διαδικασία, και θα "αποτυπώνει" την εικόνα της γραφικής απαναπαράστασης του κυττάρου–widget στο αρχείο. Ο μηχανισμός αυτός θα κάνει κάτι παρόμοιο με τις εφαρμογές X xwd και xpr [18], οι οποίες δημιουργούν ένα "αποτύπωμα" ενός παραθύρου (xwd) και μεταπρέπουν το "αποτύπωμα" αυτό σε κώδικα PostScript (xpr).

Ωστόσο είναι πολύ πιθανό η διαδικασία αυτή να δημιουργεί ένα "αποτύπωμα" με διαστάσεις διαφορετικές από εκείνες τις οθόνης. Στην περίπτωση αυτή, μπορούμε να τις μεταβάλουμε με την εντολή scale της PostScript, ώστε να ταιριάζουν.

Προφανώς, όλες οι παραπάνω διαδικασίες θα κάνουν χρήση της "προσωπικής μεταβλητής" widget–ID για να μπορούν να εντοπίζουν το κύτταρο–widget και να μεταβάλλουν τα χαρακτηριστικά του.

Οπως φάνηκε, ο σχεδιαστής των πρωτογενών κυττάρων–widgets κινδυνεύει να κάνει αρκετά όμοια πράγματα κατά τον ορισμό τους. Για να αποφευχθεί αυτό, είναι καλό να ορίσουμε μία κλάση πρωτογενών κυττάρων, της οποίας όλα τα κύτταρα–widgets να είναι "απόγονοί", και στην οποία θα είναι οριμένες οι κοινές διαδικασίες για όλα αυτά τα κύτταρα. Ετσι, ο σχεδιαστής θα ορίζει μόνο τα χαρακτηριστικά εκείνα των κυττάρων–widgets που τα διαφοροποιούν από τα υπόλοιπα.

3.4.5 Αλλαγές στους Μηχανισμούς του Λαβύρινθου

Ο μηχανισμός επιλογής του συστήματος θα χρειαστεί αλλαγές εξ’ αιτίας της χρήσης των κυττάρων–widgets. Το πρόβλημα οφείλεται στο ότι τα κύτταρα–widgets "κλέβουν" τα γεγονότα X που συμβαίνουν όταν το ποντίκι βρίσκεται πάνω τους. Ετσι το σύστημα δεν μπορεί να γνωρίζει ότι το ποντίκι υπέδειξε ένα σημείο του σχεδιαστικού παραθύρου, οπότε ο μηχανισμός επιλογής δεν ενεργοποιείται. Για το λόγο αυτό, η πόρτα p του παραδείγματος δεν μπορεί να επιλεγτεί.

Στο πρόβλημα αυτό μπορούμε να δώσουμε δύο λύσεις. Η πρώτη είναι να απαγορεύσουμε στα κύτταρα–widgets να έχουν αυτόνομη λειτουργικότητα. Ετσι, οποιαδήποτε μεταβολή τους θα γίνεται μόνο μέσω εντολών του συστήματος. Για τα κύτταρα του παραδείγματός μας, αυτό σημαίνει ότι η λαβή δεν θα κινείται καθώς ο χρήστης μετακινεί το ποντίκι πάνω στο widget αυτό. Το σύστημα θα δέχεται τα γεγονότα ButtonPress, και με διαδικασίες ορισμένες από τον σχεδιαστή των πρωτογενών κυττάρων, θα δίνει στην πόρτα sliderValue την κατάλληλη τιμή, ενώ θα επανασχεδιάζει την λαβή στη νέα θέση μέσω εντολών της βιβλιοθήκης Xm [11] προς το widget ScrollBar. Ανάλογες μεταβολές θα έχουμε και για το Text κύτταρο–widget. Η διαδικασία αυτή θα στηρίζεται στην ύπαρξη ενός αόρατου παραθύρου πάνω από το σχεδιαστικό παράθυρο του Λαβύρινθου, το οποίο θα "κλέβει" τα γεγονότα X που προορίζονται για τα κύτταρα–widgets. Ετσι, ο Λαβύρινθος θα τα "επεξεργάζεται" όπως αυτός θέλει. Η λύση αυτή δεν συνεπάγεται αλλαγές στον μηχανισμό επιλογής, ή σε άλλους μηχανισμούς, αλλά δημιουργεί ένα νέο μηχανισμό που χειρίζεται κατάλληλα τα γεγονότα αυτά. Ωστόσο, θεωρούμε ότι η λύση αυτή δεν παρουσιάζει κανένα ενδιαφέρον και μάλλον συνηγορεί στη υπολειτουργία των widgets, αφού αντιμετωπίζει τα widgets μόνο ως οντότητες με πολύπλοκες σχεδιαστικές διαδικασίες. Ο σχεδιαστής θα χρειάζονταν να υλοποιήσει εξ’ αρχής μέρος της λειτουργικότητάς τους, γεγονός που θα έφερνε πρόσθετες δυσκολίες στην διαδικασία ορισμού τους.

Η δεύτερη λύση προσπαθεί να εκμεταλευτεί την λειτουργικότητα των widgets κατά το μέγιστο. Προτείνουμε ο σχεδιαστής να ορίζει τις διαδικασίες–callbacks και τους χειριστές γεγονότων για το κύτταρο–widget, ώστε να ενεργοποιούν τον μηχανισμό επιλογής. Ετσι, θα επιλέγεται το κύτταρο–widget που δέχτηκε ένα γεγονός, καθώς και όλα τα κύτταρα που τέμνονται με αυτό (κάνοντας χρήση των ορθογωνίων περιγραμμάτων τους). Αυτό συνεπάγεται ότι θα πρέπει να τροποποιηθεί ελαφρώς ο μηχανισμός επιλογής. Ετσι όμως, αποφεύγεται η επιβάρυνση του σχεδιαστή με περιττό προγραμματισμό για την υλοποιήση εξ' αρχής μέρους της λειτουργικότητάς των κυττάρων–widgets.

4. Χρώματα, Είδη Γραμμών, Είδη Γραφής

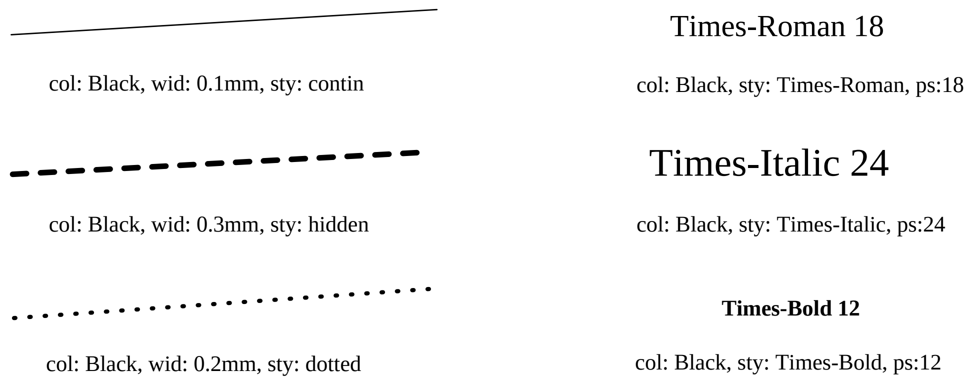
4.1 Περιγραφή Σχεδιαστικών Χαρακτηριστικών

Ο Λαβύρινθος παρέχει κύτταρα με πόρτες που καθορίζουν το χρώμα, το είδος και το πάχος των γραμμών τους, το είδος γραφής (οικογένεια γραφής), και το μέγεθός του κειμένου τους. Ο χρήστης μπορεί να δίνει τιμές στις πόρτες αυτές, καθορίζοντας έτσι τα συγκεκριμένα χαρακτηριστικά των κυττάρων.

Παραδείγματα τέτοιων κυττάρων είναι το "ευθύγραμμο τμήμα", το "τόξο", ο "κύκλος", το "κείμενο", το "μέγεθος κειμένου", κ.α. (βλέπε παράρτημα Β). Συγκεκριμένα, το πρωτογενές κύτταρο "ευθύγραμμο τμήμα" έχει τις πόρτες *col, sty, wid* για να μπορεί ο χρήστης να καθορίζει το χρώμα, το είδος και το πάχος της γραμμής του αντίστοιχα, ενώ το "κείμενο" έχει τις πόρτες *col, sty, ps* για το καθορισμό του χρώματος, της οικογένειας γραφής και του μεγέθους γραμμάτων του αντίστοιχα. Στο σχήμα 4.1 βλέπουμε πώς η γραφική αναπαράσταση των δύο αυτών κυττάρων επηρεάζεται από τις διαφορετικές τιμές των σχεδιαστικών χαρακτηριστικών τους. Ο χρήστης μπορεί να ορίσει σε ένα σύνθετο κύτταρο πόρτες για χρώμα, είδη γραμμών και γραφής συνδέοντας τις πόρτες αυτές με πόρτες αντίστοιχων χαρακτηριστικών των συνιστώντων κυττάρων του.

Οι τιμές που μπορούν να πάρουν οι παραπάνω πόρτες είναι μικροί θετικοί αριθμοί, οι οποίοι στην πραγματικότητα είναι δείκτες (indices) σε πίνακες, στατικά ορισμένους σε πηγαίο κώδικα, στους οποίους το σύστημα φυλάσσει την πραγματική πληροφορία. Οι τιμές που έχουν για κάθε ένα από τα χαρακτηριστικά φαίνονται στους πίνακες που ακολουθούν.

Για το παράδειγμα του σχήματος, οι τιμές των πορτών για το μεσαίο "ευθύγραμμο τμήμα" είναι *col: 0, wid: 3, sty: 3,* ενώ για το αντίστοιχο "κείμενο" είναι *col: 0, sty: 2, ps: 6.* Επιλέξαμε να δίνονται αριθμητικές τιμές στις παραπάνω πόρτες για να αποφύγουμε την σπατάλη χώρου εξ' αιτίας των πολλαπλών τρόπων περιγραφής ιδιοτήτων (όπως πολλαπλές αλφαριθμητικές σταθερές, π.χ. "κόκκινο", "Times Roman", για τα χρώματα και την οικογένειες γραφής, αριθμοί για τα πάχη γραμμών και τα μεγέθη των γραμμάτων, καθώς και λίστες αριθμών για τα είδη γραμμών που περιγράφουμε παρακάτω), καθώς και για να μπορούμε, αλλάζοντας τα περιεχόμενα ενός πίνακα, να προκαλούμε "ομαδικές" αλλαγές στα χαρακτηριστικά των



Σχήμα 4.1: Κύτταρα με διαφορετικά είδη γραμμών και γραφής

Προκαθορισμένα Χρώματα							
0	black	4	blue	8	cyan	12	coral
1	white	5	orange	9	grey	13	navy
2	red	6	pink	10	yellow	14	violet
3	green	7	magenta	11	brown	15	gold

Προκαθορισμένες Οικογένειες Γραφής	
0	system default
1	Times Roman
2	Times Italic
3	Times Bold
4	Times Bold-Italic

Προκαθορισμένα Μεγέθη Κειμένου	
0	‘‘normal’’
1	8 points
2	10 points
3	12 points
4	14 points
5	18 points
6	24 points

κυττάρων.

Τα χρώματα στο περιβάλλον X (τα οποία κληρονομούμε στον Λαβύρινθο) είναι ορισμένα με ονόματα (red, black, κλπ). Μια άλλη εκδοχή για τον ορισμό τους είναι οι τιμές RGB (Red, Green, Blue); το κόκκινο (red) αντιστοιχεί στις τιμές [255, 0, 0], ενώ το πορτοκαλί (orange) στις [255, 165, 0]. Στο μέλλον ο χρήστης θα μπορεί να ορίζει τα χρώματα και με τους δύο τρόπους, μέσω εντολών που θα ορίζουν τα περιεχόμενα και το μέγεθος του παραπάνω πίνακα. Ανάλογα, θα μπορεί να ορίζει δυναμικά τους πίνακες και για τα άλλα χαρακτηριστικά των κυττάρων, ενώ θα υπάρχουν εντολές που θα δείχνουν στον χρήστη τις τρέχουσες τιμές των πινάκων αυτών. Στο παρόν σύστημα τα παραπάνω προκαθορισμένα χαρακτηριστικά μπορούν να αλλάξουν μόνο με re-compilation.

Οι οικογένειες γραφής που παρέχει το σύστημα[9], αυτή την στιγμή, είναι πέντε. Ο πίνακας

[9] Για την υποστήριξη, από τον Λαβύρινθο, των πολλαπλών ειδών γραφής και μεγεθών κειμένου, εργάστηκε ο Frank Jansen, ο οποίος συμμετείχε προσωρινά στην ερευνητική ομάδα του Λαβύρινθου.

των οικογενειών γραφής περιέχει τα ονόματα των οικογενειών αυτών. Ο Λαβύρινθος υποστηρίζει μόνο την οικογένεια γραφής Times στους τύπους απλή, πλάγια, έντονη και πλάγια–έντονη σε οκτώ μεγέθη[10]. Ωστόσο, στο μέλλον, το σύστημα θα υποστηρίζει περισσότερες οικογένειες γραφής και μεγέθη.

Προκαθορισμένα είδη γραμμών

0: contin		4: dashDot	2, 1, 0, 1
1: dashed	2, 1	5: dashDotDot	2, 1, 0, 1, 0, 1
2: hidden	1, 1	6: dashDashDot	2, 1, 2, 1, 0, 1
3: dotted	0, 1	7: longDashDot	4, 1, 1, 1

Προκαθορισμένα Πάχη Γραμμών					
0	‘‘usual’’	3	0.3 mm	6	0.7 mm
1	0.1 mm	4	0.4 mm	7	1.0 mm
2	0.2 mm	5	0.5 mm		

Παραπάνω φαίνονται τα διαφορετικά είδη γραμμών του συστήματος[11]. Οι αριθμοί κάτω από καθε είδος γραμμής δείχνουν τις αναλογίες μεταξύ των μηκών των γραμμών που σχεδιάζονται και εκείνων που δεν σχεδιάζονται. Με τους αριθμούς αυτούς περιγράφονται οι γραμμές αυτές τόσο στο περιβάλλον X, όσο και σε κώδικα PostScript. Οι γραμμές αυτές μπορούν να έχουν οκτώ διαφορετικά πάχη (μετρημένα σε χιλιοστά, mm). Ο τρόπος με τον οποίο το σύστημα μετατρέπει τα πάχη αυτά σε *pixels* στην οθονη είναι προσεγγιστικός, αλλά αξιόπιστος. Στο μέλλον η μετατροπή αυτή θα πρέπει να βελτιωθεί ακόμη περισσότερο.

Ο Λαβύρινθος παρέχει διαδικασίες με τις οποίες ο σχεδιαστής πρωτογενών κυττάρων μπορεί να έχει πρόσβαση στους παραπάνω πίνακες, και οι οποίες δέχονται, ως παράμετρο, τον δείκτη σε ένα πίνακα και επιστρέφουν κατάλληλες τιμές που ο σχεδιαστής μπορεί να χρησιμοποιήσει για σχεδιάσει στη οθόνη ή να καταχωρίσει στο αρχείο PostScript. Για παράδειγμα, η διαδικασία *getLabyDashes()* δεχόμενη ως παράμετρο την τιμή 3 επιστρέφει, από τον αντίστοιχο πίνακα με τα προκαθορισμένα είδη γραμμών, την λίστα των αριθμών: {0, 1} που περιγράφουν το είδος της γραμμής.

[10] Ο λόγος για τον οποίο ο Λαβύρινθος δεν προσφέρει περισσότερες οικογένειες γραφής είναι, γιατί δεσμεύεται από τον περιορισμένο αριθμό αρχείων με πληροφορίες για τις διαστάσεις των οικογενειών αυτών σε κώδικα PostScript, τα οποία διαθέτουμε στο υπολογιστικό σύστημα που αναπτύσσεται σήμερα ο Λαβύρινθος. Ακόμα, η αδυναμία του συστήματος να υποστηρίξει περισσότερα μεγέθη για την συγκεκριμένη, ή και για περισσότερες, οικογένειες γραφής οφείλεται στις λίγες επιλογές σε μεγέθη που παρέχει το περιβάλλον X για τις οικογένειες γραφής της οθόνης (screen fonts).

[11] Στα αρχικά στάδια της ανάπτυξης για την υποστήριξη, από τον Λαβύρινθο, των πολλαπλών ειδών γραμμής και χρωμάτων, βοήθησε η Μαρία Καραβασίλη, μέλος της ερευνητικής ομάδας του Λαβύρινθου.

Οπως αναφέραμε παραπάνω, ο Λαβύρινθος θα έχει την δυνατότητα, στο μέλλον, δυναμικού ορισμού των πινάκων των σχεδιαστικών χαρακτηριστικών. Η διαδικασία μεταβολής των περιεχομένων τους θα είναι ιδιαίτερα απλή. Μόνο στην περίπτωση ορισμού των σχεδιαστικών χαρακτηριστικών του κειμένου ο χρήστης θα πρέπει να είναι περισσότερο προσεκτικός, διότι το σύστημα, παράλληλα με τους πίνακες των οικογενειών γραφής και μεγεθών κειμένου, κρατάει και άλλους δύο πίνακες: έναν ίδιου μεγέθους με τον πίνακα των ονομάτων των οικογενειών γραφής, που περιέχει τα ονόματα των αρχείων με τις διαστάσεις της κάθε οικογένειας για την εκτύπωση (διαστάσεις του κειμένου σε PostScript) και έναν που έχει τόσα στοιχεία όσα και ο αριθμός των οικογενειών γραφής επί τον αριθμό των μεγεθών που υπάρχουν, και περιέχει τα ονόματα των αρχείων με τις διαστάσεις για την οθόνη κάθε οικογένειας για κάθε ένα συγκεκριμένο μέγεθος. Ετσι ο χρήστης στις εντολές του, εκτός από τα ονόματα των οικογενειών γραφης και τα μεγέθη του κειμένου, θα πρέπει να δηλώνει τις παραπάνω πληροφορίες που συμπληρώνουν τους άλλους δύο παράλληλους πίνακες του συστήματος. Ωστόσο, το σύστημα θα πρέπει να προσφέρει ένα μηχανισμό που να αυτοματοποιεί – κατά το δυνατό – αυτή την διαδικασία.

4.2 Υλοποίηση – Αποδοτική χρήση της Xlib κατά τη σχεδίαση

Στο περιβάλλον X, όλες οι γραφικές εντολές προς τον X server χρησιμοποιούν μία ειδική δομή, που ονομάζεται "Γραφικά Συμφραζόμενα" (Graphic Context – GC) και περιλαμβάνει σχεδιαστικά χαρακτηριστικά όπως το πάχος γραμμής, το είδος γραμμής, το χρώμα, το είδος γραφής, η "περιοχή ψαλιδίσματος", κ.α [14]. Οι γραφικές εντολές, όπως οι εντολές σχεδίασης γραμμών, χρησιμοποιούν τις τιμές των πεδίων των GCs για να καθορίζουν την λειτουργία τους, δηλαδή τί γραμμές θα σχεδιάσουν και πώς. Τα πεδία της δομής αυτής είναι "ιδιωτικά" (private) στην βιβλιοθήκη Xlib, και ο προγραμματιστής για να δώσει τιμές σ' αυτά, πρέπει να καλέσει τις διαδικασίες διαχείρισής τους, που παρέχει η Xlib.

4.2.1 Περιγραφή του προβλήματος

Στον Λαβύρινθο, οι διαδικασίες σχεδίασης των πρωτογενών κυττάρων χρησιμοποιούν GCs για να σχεδιάζουν στην οθόνη. Συγκεκριμένα, το σύστημα κάνει χρήση των πεδίων: *foreground* (για το χρώμα), *line_style*, *line_width*, *dash_offset*, και *dashes* (για το είδος και το πάχος των γραμμών) και *font* (για τη οικογένεια γραφής και το μέγεθος του κειμένου).

Ετσι, μία τυπική διαδικασία σχεδίασης, π.χ. η διαδικασία σχεδίασης του "ευθύγραμμου τμήματος", χρειάζεται πρώτα να ορίσει ένα νέο GC, μετά να δώσει με κατάλληλες εντολές τις κατάλληλες τιμές στα πεδία του, και μετά να δώσει την εντολή σχεδίασης της γραμμής XDraw-Line() με όρισμα το συγκεριμένο GC. Ανάλογα ορίζουμε τις διαδικασίες σχεδίασης για όλα τα πρωτογενή κύτταρα που έχουν σχεδιαστκές ιδιότητες, όπως χρώμα, είδος και πάχος γραμμής κλπ. Εάν ακολουθούσαμε αυτόν τον τρόπο, η διαδικασία σχεδίασης του "ευθύγραμμου τμήματος" θα ήταν η εξής:

α. Διάβασε τις τιμές των πορτών "χρώμα", "είδος γραμμής", και "πάχος γραμμής"

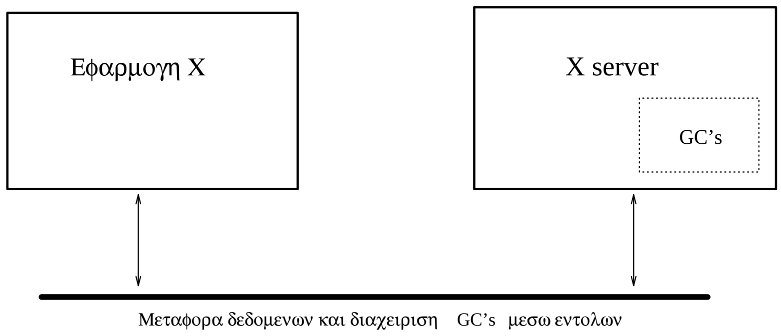
β. Βρες τις πραγματικές τιμές των παραπάνω χαρακτηριστικών από τους αντίστοιχους πίνακες του συστήματος και δημιούργησε ένα GC με τις τιμές αυτές στα αντίστοιχα πεδία

γ. Διάβασε τις τιμές των πορτών "αρχικό σημείο" και "τελικό σημείο"

δ. Σχεδίασε μια γραμμή μεταξύ των δύο σημείων χρησιμοποιώντας το παραπάνω GC

ε. Ελευθέρωσε τον χώρο που κρατήθηκε για το GC.

Ο μηχανισμός επανασχεδίασης του Λαβύρινθου καλεί αναδρομικά τις διαδικασίες σχεδίασης των κυττάρων του σχεδίου μέχρι να φτάσει σε πρωτογενή κύτταρα. Αρα με τον παραπάνω τρόπο, το σύστημα θα δημιουργούσε, θα χρησιμοποιούσε και μετά θα κατέστρεφε ένα GC σε κάθε κλήση της διαδικασίας σχεδίασης κάθε πρωτογενούς κυττάρου του σχεδίου. Τα GCs είναι δομές που δημιουργούνται, φυλάσσονται, και ενημερώνονται από τον X server, και οι εφαρμογές έχουν πρόσβαση σε αυτά μόνο μέσω των ταυτοτήτων τους (GC–IDs).



Σχήμα 4.2: Η μεταφορά και η διαχείριση των GCs από τον X server

Κάθε εφαρμογή X που δημιουργεί και χρησιμοποιεί ένα ή περισσότερα GCs πρέπει να στέλνει εντολές διαχείρισης τους στον X server (σχήμα 4.2). Ειδικά οι εντολές δημιουργίας, ή μεταβολής, ενός GC συνεπάγονται και μετακίνηση δεδομένων (τις τιμές των πεδίων του GC) μέσω του X πρωτοκόλου. Δηλαδή, το πρόγραμμα θα έστελνε συνεχώς προς τον X server εντολές δημιουργίας και καταστροφής των GCs, εντολές σχεδίασης, καθώς και τις τιμές των πεδίων των GCs, ενώ ο X server θα κρατούσε και ελευθέρωνε συνεχώς χώρο στον υπολογιστή όπου έτρεχε. Το γεγονός αυτό, σε συνδυασμό με το ότι ο Λαβύρινθος και ο X server μπορεί να βρίσκονται σε διαφορετικούς υπολογιστές και η επικοινωνία μεταξύ τους να γίνεται μέσω δικτύου, θα δημιουργούσε καθυστερήσεις στο σχεδιαστικό μηχανισμό, ιδιαίτερα ενοχλητικές για τον χρήστη του συστήματος.

4.2.2 Χρήση ενός ‘‘γενικού’’ GC

Μια λύση στο παραπάνω πρόβλημα θα ήταν να μεταβάλλουμε τις διαδικασίες σχεδίασης των πρωτογενών κυττάρων, ώστε να μην δημιουργούν και καταστρέφουν το GC σε κάθε κλήση τους, αλλά να ορίζαμε ένα "γενικό" GC (global variable), στο οποίο να δίναμε κάθε φορά νέες τιμές στα πεδία που ενδιαφέρουν την σχεδιαστική εντολή. Ετσι, στην διαδικασία της προηγούμενης παραγράφου το βήμα (ε) θα ήταν περιττό, ενώ το βήμα (β) θα μεταβάλλονταν ως εξής:

β. Βρες τις πραγματικές τιμές των παραπάνω χαρακτηριστικών από τους αντίστοιχους πίνακες του συστήματος και καταχώρησε τις στα αντίστοιχα πεδία του "γενικού" GC του συστήματος

Για κάθε κλήση της διαδικασίας σχεδίασης κάθε κυττάρου του σχεδίου, το σύστημα θα μετέβαλε τα πεδία του "γενικού" GC και θα χρησιμοποιούσε αυτό στην εντολή σχεδίασης προς τον X server. Ετσι ο X server δεν θα δέσμευε και θα ελευθέρωνε χώρο κάθε φορά που το σύστημα επρόκειτο να σχεδιάσει κάτι.

Ωστόσο, κατά την διαδικασία της σχεδίασης ο μηχανισμός επανασχεδίασης ενεγοποιείται με μεγάλη συχνότητα: ένα μέρος το σχεδίου μπορεί να επανασχεδιάζεται σχεδόν κάθε δευτερόλεπτο, όταν ο χρήστης κάνει συνδέσεις, αποσυνδέσεις, μετακινήσεις κυττάρων στο σχέδιο. Ακόμα, οι διαδικασίες σχεδίασης των πρωτογενών κυττάρων καλούνται συχνά για να επανασχεδιάσουν το κύτταρο σε μία νέα θέση, επειδή ο χρήστης έδωσε – έμμεσα, ή άμεσα – νέα τιμή σε κάποια γεωμετρική πόρτα του, χωρίς ωστόσο να έχουν αλλάξει τα σχεδιαστικά χαρακτηριστικά του, π.χ. το χρώμα του. Ανάλογα συχνή είναι η επανασχεδίαση κυττάρων, τα οποία δεν έχουν υποστεί καμία μεταβολή, αλλά "έτυχε" να βρίσκονται ολόκληρα ή μέρος τους στην "περιοχή σβησίματος".

4.2.3 Χρήση ενός Hash Table

Σε κάθε τέτοια περίπτωση, θα μπορούσαμε να φυλάσσουμε το GC, για κάθε ένα κύτταρο, και να το χρησιμοποιούμε στην διαδικασία σχεδίασης του κυττάρου, μέχρι να αλλάξει κάποιο από τα σχεδιαστικά χαρακτηριστικά του. Ετσι θα αποφεύγαμε τις περιττές εντολές προς τον X server. Ωστόσο, θα πρέπει να προσέξουμε ότι περισσότερα από ένα κύτταρα του σχεδίου μπορεί να έχουν ίδια σχεδιαστικά χαρακτηριστικά. Ενα σχέδιο με εκατοντάδες κύτταρα μπορεί να περιέχει περισσότερα από ένα διακεκκομένα κόκκινα "ευθύγραμμα τμήματα" πάχους 0.3 mm. Ανάλογα, μπορεί να υπάρχουν περισσότερα από ένα Times-Italic "κείμενα" μεγέθους 14 points και χρώματος μαύρου. Θα μπορούσαμε λοιπόν να χρησιμοποιούμε κοινά GCs για να σχεδιάζουμε περισσότερα κύτταρα με κοινές σχεδιαστικές ιδιότητες.

Ετσι, καταλήξαμε στο ότι το σύστημα πρέπει να φυλάσσει τα GCs που δημιουργεί, και να τα χρησιμοποιεί και για την σχεδίαση όσων άλλων κυττάρων τυχαίνει να έχουν τον ίδιο συνδυασμό χαρακτηριστικών. Κάθε φορά, που μια διαδικασία σχεδίασης ενός πρωτογενούς κυττάρου χρειάζεται να σχεδιάσει, θα πρέπει να ψάχνει αν υπάρχει φυλαγμένο ένα GC με σχεδιαστικά χαρακτηριστικά όμοια με αυτά του εν λόγω κυττάρου. Αν υπάρχει, αρκεί να περνάει αυτό το GC, ως όρισμα, στην εντολή σχεδίασης που στέλνει στον X server. Με τον τρόπο αυτό, αποφεύγεται η άσκοπη επικοινωνία με τον X server για πληροφορίες (GCs) που είναι ήδη γνωστές, καθώς και η συνεχής δέσμευση και άμεση απελευθέρωση χώρου μαζί με κάθε σχεδιαστική εντολή προς αυτόν. Προφανώς, αν δεν υπάρχει τέτοιο GC, θα πρέπει να το δημιουργεί και να το φυλάσσει για μελλοντική χρήση.

Αυτό σημαίνει ότι χρειαζόμαστε μία δυναμική δομή στην οποία το σύστημα θα μπορεί να κρατάει τις ταυτότητες των GCs, που έχουν ήδη κατασκευαστεί. Η πιό κατάλληλη δομή είναι ένας Hash Table, ο οποίος παρέχει γρήγορη αναζήτηση. Ετσι, η διαδικασία σχεδίασης του πρωτογενούς κυττάρου θα πρέπει να αναζητά στον πίνακα αυτό ένα GC όμοιο με αυτό που χρειάζεται για να σχεδιάσει, και μόνο αν δεν το βρεί να προβαίνει στην δημιουργία του. Ετσι, το βήμα (β) της διαδικασίας σχεδίασης ενός πρωτογενούς κυττάρου της παραγράφου 4.2.1 μεταβάλλεται ως εξής:

β. Βρές τις πραγματικές τιμές των παραπάνω χαρακτηριστικών από τους αντίστοιχους πίνακες του συστήματος και ψάξε, μέσα στον Hash Table των GCs του συστήματος, για ένα GC με τις τιμές αυτές στα αντίστοιχα πεδία. **Εάν** δεν υπάρχει ένα τέτοιο GC, **τότε** δημιούργησε ένα νέο, δίνοντας τις παραπάνω τιμές στα αντίστοιχα πεδία αυτού, και ενημέρωσε κατάλληλα τον Hash Table

---

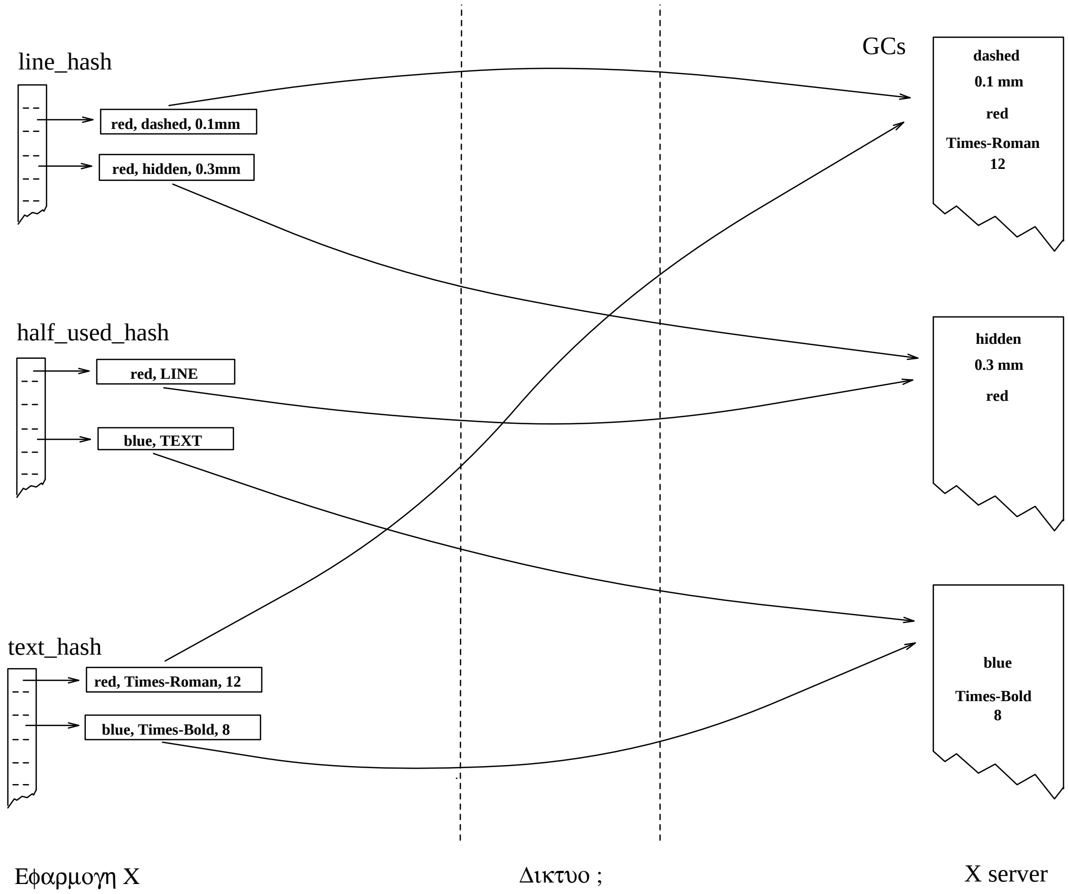
Προφανώς χρειαζόμαστε μιά διαδικασία για αναζήτηση και εισαγωγή στον παραπάνω πίνακα. Η διαδικασία θα δέχεται, ως ορίσματα, τα γραφικά χαρακτηριστικά ενός κυττάρου, θα αναζητά το κατάλληλο GC, και θα επιστρέφει την ταυτότητα του, αν το βρεί. Στην αντίθετη περίπτωση, θα ζητάει από τον X server να δημιουργήσει ένα GC με τα χαρακτηριστικά αυτά, την ταυτότητα του οποίου θα καταχωρεί στον πίνακα και θα επιστρέφει στη διαδικασία από την οποία εκλήθει.

4.2.4 Χρήση τριών Hash Tables

Ωστόσο, παρατηρούμε ότι ορισμένα από τα σχεδιαστικά χαρακτηριστικά των πρωτογενών κυττάρων είναι αμοιβαία αποκλειόμενα: το "κείμενο" χρησιμοποιεί μόνο τα πεδία του GC που αντιστοιχούν στο χρώμα, την οικογένεια γραφής, και το μέγεθος του κειμένου, ενώ το "ευθύγραμμο τμήμα", ο "κύκλος", το "τόξο" κ.α. χρησιμοποιούν τα πεδία για το χρώμα, το είδος και το πάχος της γραμμής. Αρα, κάθε GC για τα παραπάνω κύτταρα έχει κάποια από τα πεδία του αχρησιμοποίητα. Ετσι, έχουμε δύο "ομάδες" από GCs: τα GCs–γραμμών και τα GCs–κειμένου, όπου τα GCs της μίας δεν χρησιμοποιούν τα πεδία που χρησιμοποιούν τα GCs της άλλης, με εξαίρεση το πεδίο για το χρώμα.

Μπορούμε να εκμεταλευτούμε το γεγονός αυτό, ώστε το σύστημα να δημιουργεί όσο το δυνατό λιγότερα GCs για όσο το δυνατό περισσότερα κύτταρα. Με εξαίρεση το χρώμα, τα πεδία για υπόλοιπα χαρακτηριστικά μπορούν να φυλάσσονται στο ίδιο GC χωρίς να επικαλύπτονται, π.χ. διακεκκομένα – hidden – "ευθύγραμμα τμήματα" πάχους 0.3 mm και Times-Roman "κείμενα" μεγέθους 12 points μπορούν να μοιράζονται το ίδιο GC αρκεί να έχουν το ίδιο χρώμα (π.χ. να είναι κόκκινα). Με τον τρόπο αυτό, το σύστημα μπορεί να δημιουργεί μόνο τα GCs εκείνα που είναι απαραίτητα, δεσμεύοντας έτσι τον λιγότερο δυνατό χώρο στον X server.

Με την παραπάνω τακτική, οι "ομάδες" των GCs αυξάνονται σε τρείς: τα GCs–γραμμής, τα GCs–κειμένου και τα "κοινά"–GCs. Τα "κοινά"–GCs προκύπτουν από την "ένωση" δύο GCs, ενός GC–γραμμής και ενός GC–κειμένου που έχουν κοινό χρώμα. Οι δύο άλλες "ομάδες" έχουν GCs που δεν είναι συμβατά μεταξύ τους, δηλαδή δεν υπάρχουν δύο GCs – ένα από καθε ομάδα – που έχουν την ίδια τιμή για το πεδίο του χρώματος.



Σχήμα 4.3: Οι τρείς Hash Tables και τα τρία είδη GCs

Για να υλοποιήσουμε ένα μηχανισμό, ο οποίος να εντοπίζει και να φυλάσσει τα "κοινα"–GCs του συστήματος, χρησιμοποιήσαμε τρείς Hash Tables. Στον πρώτο πίνακα, που ονομάζουμε πίνακα γραμμών (*line_hash*), φυλάσσουμε τα GCs που είναι χρήσιμα για την σχεδίαση γραμμών: τα GCs–γραμμής και τα "κοινά"–GCs (τα οποία έχουν επιπλέον πληροφορία για σχεδίαση κειμένου). Ανάλογος είναι ο πίνακας κειμένων (*text_hash*), ο οποίος περιέχει GCs που είναι χρήσιμα για την σχεδίαση κειμένου: τα GCs–κειμένου και τα "κοινά"–GCs. Ο τρίτος πίνακας περιέχει τα GCs που δεν είναι "πλήρη" (*half_used_hash*), δηλαδή τα GCs των οποίων ορισμένα "χρήσιμα" πεδία δεν έχουν ακόμα οριστεί και περιέχει τόσο τα GCs–γραμμής, όσο και τα GCs–κειμένου (όπως φαίνεται στο σχήμα 4.3). Ετσι κάθε GC που ορίζεται από το σύστημα θα πρέπει να βρίσκεται ταυτόχρονα σε δύο από τους τρείς Hash Tables: *line_hash* και *half_used_hash* ή *text_hash* και *half_used_hash* ή *line_hash* και *text_hash*. Τα κλειδιά αναζήτησης για τους πίνακες αυτούς είναι για τους *line_hash* και *text_hash* οι πραγματικές τιμές των τριάδων ["χρώμα", "είδος γραμμής", "πάχος γραμμής"] και ["χρώμα", "οικογένεια γραφής", "μέγεθος γραμμάτων"] αντίστοιχα, ενώ για τον *half_used_hash* οι πραγματικές τιμές της δυάδας ["χρώμα", "είδος GC"].

Κάθε διαδικασία σχεδίασης ενός πρωτογενούς κυττάρου, θα πρέπει να αναζητήσει ένα GC με συγκεκριμένα σχεδιαστικά χαρακτηριστικά. Το GC αυτό, θα πρέπει να αναζητηθεί στον πίνακα γραμμών, ή τον πίνακα κειμένων, ανάλογα με το τί κύτταρο είναι αυτό ("κείμενο", "ευθύγραμμο τμήμα", "κύκλος", κτλ). Αν το ζητούμενο GC δεν εντοπιστεί θα πρέπει να δημιουργηθεί ένα νέο GC. Ωστόσο, πριν από αυτό, η διαδικασία θα πρέπει να εξετάσει αν έχει ήδη δημιουργηθεί κάποιο άλλο GC με το ίδιο χρώμα, αλλά με αχρησιμοποίητα τα πεδία για τα σχεδιαστικά χαρακτηριστικά που την ενδιαφέρουν. Το GC αυτό θα πρέπει να αναζητηθεί στον πίνακα με τα "μη–πλήρη" GCs. Αν υπάρχει, η διαδικασία θα πρέπει πρώτα να δώσει τις τιμές των παραπάνω χαρακτηριστικών στα αντίστοιχα πεδία αυτού του GC και μετά να το χρησιμοποιήσει για να σχεδιάσει το κύτταρο. Μόνο αν δεν υπάρχει ένα τέτοιο "μη–πλήρες" GC, θα πρέπει να δημιουργήσει ένα νέο και να ενημερώσει κατάλληλα τους παραπάνω πίνακες.

Στον Λαβύρινθο έχουμε ορίσει δύο διαδικασίες, τις *findLineGC()* και *findTextGC()*, για την αναζήτηση των GCs και την ενημέρωση των πινάκων (Hash Tables), στους οποίους αυτά φυλάσσονται. Η *findLineGC()* δέχεται, ως ορίσματα, το χρώμα, το είδος, και το πάχος γραμμής ενός πρωτογενούς κυττάρου, και επιστρέφει ένα GC κατάλληλο για την σχεδίαση των γραμμών αυτού. Η *findTextGC()*, αντίστοιχα, δέχεται, ως ορίσματα, το χρώμα, την οικογένεια γραφής, και το μέγεθος του κειμένου του κυττάρου, και επιστρέφει ένα GC κατάλληλο για την σχεδίαση του κειμένου αυτού. Τα παραπάνω ορίσματα είναι οι δείκτες στους πίνακες των προκαθορισμένων χαρακτηριστικών των κυττάρων. Ανάλογα λοιπόν με το κύτταρο ("κείμενο", "ευθύγραμμο τμήμα", "κύκλος", κτλ), η μία από τις δύο αυτές διαδικασίες καλείται στο βήμα (δ) της διαδικασίας σχεδίασής του. Παρακάτω, παρουσιάζουμε την διαδικασία αναζήτησης ενός GC για σχεδίαση γραμμών (*findLineGC()*), που χρησιμοποιείται στις διαδικασίες σχεδίασης των πρωτογενών κυττάρων: "ευθύγραμμο τμήμα", "κύκλος", "τόξο" κ.α.

---

findLineGC("χρώμα", "είδος γραμμής", "πάχος γραμμής")

α. Βρες τις πραγματικές τιμές για τα ορίσματα "χρώμα", "είδος γραμμής" και "πάχος γραμμής" από τους αντίστοιχους πίνακες του συστήματος

β. Χρησιμοποίησε τις τιμές αυτές, ως κλειδί αναζήτησης, και ψάξε στον πίνακα *line_hash* ένα GC με ίδιο κλειδί

  Εάν υπάρχει τέτοιο GC **τότε**

    Επέστρεψε την τιμή της ταυτότητάς του (GC–ID) στην διαδικασία που σε κάλεσε

γ. **Αλλιώς**

  Χρησιμοποίησε την τιμή του "χρώματος" και του είδος του GC (GC–κειμένου), ως κλειδί αναζήτησης, και αναζήτησε στον πίνακα *half_used_hash* ένα GC με ίδιο κλειδί

    Εάν υπάρχει τέτοιο GC **τότε**

    i. Ζήτησε από τον X server να εκχωρίσει τις πραγματικές τιμές για τα ορίσματα: "είδος γραμμής" και "πάχος γραμμής", στα αντίστοιχα (αχρησιμοποίητα) πεδία του

    ii. Πρόσθεσε το στον πίνακα *line_hash*, με κλειδιά το "χρώμα", "είδος γραμμής", και "πάχος γραμμής"

    iii. Αφαίρεσέ το από τον πίνακα *half_used_hash* με κλειδιά το "χρώμα" και το είδος του GC: GC–κειμένου

iv. **Επέστρεψε** την τιμή της ταυτότητάς του στην διαδικασία που σε κάλεσε

Αλλιώς

i. Ζήτησε από τον X server να δημιουργήσει ένα νέο GC δίνοντας τις πραγματικές τιμές για τα ορίσματα: "χρώμα", "είδος γραμμής", και "πάχος γραμμής", στα αντίστοιχα πεδία του

ii. Πρόσθεσε το στον πίνακα *line_hash* με κλειδιά το "χρώμα", "είδος γραμμής", και "πάχος γραμμής"

iii. Πρόσθεσε το στον πίνακα *half_used_hash* με κλειδιά το "χρώμα" και το είδος του GC: GC–γραμμής

iv. **Επέστρεψε** την τιμή της ταυτότητάς του στην διαδικασία που σε κάλεσε

Ανάλογη διαδικασία χρησιμοποιείται στην διαδικασία σχεδίασης του "κειμένου" με ορισμένες προφανείς αλλαγές. Εστω ότι ο μηχανισμός επανασχεδίασης καλεί τη διαδικασία σχεδίασης ενός "κειμένου". Αυτή θα αναζητήσει ένα GC που να ταιρίαζει με τα σχεδιαστικά χαρακτηριστικά του "κειμένου". Η αναζήτηση ξεκινάει από τον πίνακα *text_hash*. Αν το ζητούμενο GC δεν εντοπιστεί εκεί, η αναζήτηση συνεχίζεται στον πίνακα *half_used_hash*, με την ένδειξη ότι αναζητείται ένα "μη-πλήρες" GC–γραμμής με μόνη προϋπόθεση να έχει το χρώμα του "κειμένου". Αν και πάλι δεν εντοπιστεί τέτοιο GC, ο μηχανισμός αυτός ζητά από τον X server να δημιουργήσει ένα GC με τα κατάλληλα πεδία και το φυλάσσει στους πίνακες *text_hash* και *half_used_hash*.

4.2.5 Αξιολόγηση

Η χρήση των πινάκων *line_hash, text_hash* και *half_used_hash* προφανώς επιβαρύνει με πρόσθετο χώρο τον Λαβύρινθο. Ωστόσο, πιστεύουμε ότι πετυχαίνουμε σημαντική βελτίωση στην ταχύτητα επανασχεδίασης, και έχουμε εξοικονόμιση χώρου στον X server, χάρη στην κατά το δυνατό βέλτιστη χρήση των GCs.

Την άποψή μας δικαίωσαν και ορισμένες μετρήσεις που κάναμε σχετικά με την απόδοση των τριών τρόπων που προτάθηκαν στις παραγράφους 4.2.1, 4.2.2, και 4.2.4 αντίστοιχα, και οι οποίες φαίνονται στον παρακάτω πινακα. Ο πίνακας περιέχει τα αποτελέσματα ενός δοκιμαστικού προγράμματος (benchmark) που εκτελεί 432 διαφορετικές εντολές σχεδίασης κειμένου και γραμμών οι οποίες κάνουν χρήση 216 διαφορετικών GCs. Συγκεριμένα, περιέχει το χρόνο σε secs (elapsed time) που χρειάζεται το πρόγραμμά μας για να σχεδιάσει 432 γραμμές και κείμενα με κάθε ένα από από τους τρείς διαφορετικούς τρόπους που περιγράψαμε στις παραγράφους 4.2.1, 4.2.2, και 4.2.4 αντίστοιχα, και σε τρία διαφορετικά περιβάλλοντα λειτουργίας της συγκεριμένης εφαρμογής X. Μέσα σε παρενθέσεις δίνονται οι χρόνοι που χρειάζονται για την σχεδίαση μιάς απλής και μίας χρονοβόρας σχεδιαστικής εντολής στο συγκεκριμένο περιβάλλον.

Τα αποτελέσματα που δίνονται στον πίνακα αφορούν σε τρείς δοκιμές (tests). Η πρώτη για κάθε σχεδιαστική εντολή δημιουργεί, χρησιμοποιεί και μετά καταστρέφει ένα GC (βλέπε παράγραφο 4.2.1). Η δεύτερη για κάθε σχεδιαστική εντολή πρώτα ενημερώνει και μετά

Περιβάλλοντα Επικοινωνίας	Δοκιμή 1η	Δοκιμή 2η	Δοκιμή 3η	
			Δημιουργία	Προσπέλαση
Unix domain sockets (0.0016 - 0.116)	30.2	29.5	29.3	26.7
Μικρό Τοπικό Δίκτυο (0.0016 - 0.119)	36.3	35.5	34.9	29.6
Δίκτυο Ευρύτερης Περιοχής (0.0023 - 0.119)	108.5	106.5	103.8	47.2

χρησιμοποιεί ένα "γενικό" GC (βλέπε παράγραφο 4.2.2). Στην τρίτη οι σχεδιαστικές εντολές κάνουν χρήση των τριών Hash Tables που χρησιμοποιεί ο Λαβύρινθος (βλέπε παράγραφο 4.2.4) Στην δοκιμή αυτή μετρήθηκαν ξεχωριστά ο χρόνος σχεδίασης που περιλαμβάνει και την δημιουργία των GCs, και εκείνος όπου η σχεδίαση γίνεται με απλή προσπέλαση στους Hash Tables, εφόσον τα GCs έχουν δημιουργηθεί και καταχωρηθεί σ’ αυτούς.

Τρέξαμε το πρόγραμμά μας σε τρία διαφορετικά περιβάλλοντα επικοινωνίας του X server και της εφαρμογής X μας. Και στις τρείς περιπτώσεις ο X server, όπως και η εφαρμογή μας έτρεξαν σε υπολογιστές Sun 3/60 με λειτουργικό SunOS 4.1.1 και έγχρωμη οθόνη διακριτικότητας 1152x900 pixels. Συγκεριμένα μετρήσαμε την περίπτωση που η εφαρμογή X τρέχει στον ίδιο υπολογιστή με τον X server και επικοινωνούν μέσω Unix domain sockets, καθώς και τις περιπτώσεις που η εφαρμογή X και ο X server τρέχουν σε διαφορετικούς υπολογιστές και επικοινωνούν είτε μέσω ενός μικρού τοπικού δικτύου (Ethernet), είτε μέσω ενός δικτύου ευρύτερης περιοχής που συνδέει δύο τοπικά δίκτυα (Ethernet) μέσω modem ταχύτητας 72 Kbits/sec.

Οι χρόνοι δείχνουν ότι η χρήση των Hash Tables είναι επικερδής, ιδιαίτερα στην περίπτωση επικοινωνίας μέσω δικτύου ευρύτερης περιοχής. Βλέπουμε ότι οι χρόνοι στους οποίους δημιουργούνται τα GCs και ενημερώνονται οι Hash Tables είναι παρόμοιοι με αυτούς των δοκιμών 1 και 2. Ομως, οι χρόνοι προσπέλασης των Hash Tables και χρήσης των GCs, που είναι καταχωρημένα εκεί, είναι αισθητά καλύτεροι από τους αντίστοιχους των δοκιμών 1 και 2. Η διαπίστωση αυτή είναι σημαντική, εφόσον περιμένουμε το σχέδιο να σχεδιάζεται πολύ συχνά χωρίς (μαζικές) μεταβολές στα σχεδιαστικά χαρακτηριστικά των κυττάρων.

Εντούτοις, πραγματική εκτίμιση της κατάστασης θα μπορέσουμε να έχουμε μόνο κρατώντας στατιστικά στοιχεία από την χρήση των παραπάνω πινάκων σε σχέση με τον αριθμό των εντολών που στέλνονται προς τον X server. Αυτό δεν μπορεί ακόμα να γίνει, γιατί ο Λαβύρινθος βρίσκεται υπό ανάπτυξη και δεν υπάρχουν χρήστες που να κάνουν πραγματική χρήση του συστήματος, οπότε τα αποτελέσματα των στατιστικών μετρήσεων θα είναι πλασματικά, και κατά συνέπεια μη αξιόπιστα.

Ωστόσο, ο παραπάνω μηχανισμός φύλλαξης των GCs έχει ένα μειονέκτημα: όταν, κατά την διάρκεια της σχεδίασης, τα σχεδιαστικά χαρακτηριστικά των κυττάρων αλλάξουν, η διαδικασία που μόλις περιγράψαμε δημιουργεί νέα GCs. Ετσι είναι δυνατό να υπάρχουν στους πίνακες *line_hash*, *text_hash* και *half_used_hash*, άχρηστα GCs που δεν αντιπροσωπεύουν την τρέχουσα κατάσταση κανενός από τα κύτταρα του σχεδίου, όσον αφορά τα σχεδιαστικά χαρακτηριστικά τους, αλλά κάποια παλαιότερη. Ακόμα, καθώς ο χρήστης σχεδιάζει μετακινείται από μία μόνο περιοχή του σχεδίου σε άλλη. Αν παραμείνει για πολύ ώρα σε μία περιοχή, τότε τα GCs που έχουν οριστεί για τα κύτταρα μιας άλλης περιοχής δεν χρησιμοποιούνται για πολύ ώρα και κατά συνέπεια είναι "άχρηστα".

Είναι προφανές, ότι το σύστημα θα πρέπει να έχει ένα τρόπο να εντοπίζει τα GCs αυτά, και να ελευθερώνει το χώρο πού αυτά έχουν δεσμέσει. Για τον λόγο αυτό πρέπει να κρατάει και κάποια πρόσθετη πληροφορία μαζί με τα GCs στους παραπάνω πίνακες. Η πληροφορία αυτή θα μπορούσε να είναι κάτι ανάλογο του *reference bit* στους αλγόριθμους για εύρεση αντικαταστατάσιμης σελίδας κατά το *paging*. Κάθε φορά που μια διαδικασία σχεδίασης καλεί την διαδικασία εύρεσης ένος GC, πρέπει να θέτει το *reference bit* του. Αντίστοιχα, ο Λαβύρινθος, ανά καθορισμένο χρονικό διάστημα, πρέπει να καλεί μία διαδικασία, η οποία να ελέγχει τα *reference bits* σε όλα τα GCs στούς πίνακες *line_hash* και *text_hash*[12] και μετά να τα μηδενίζει. Αν σε κάποιο GC, το *reference bit* είναι μηδέν, σημαίνει ότι αυτό δεν χρησιμοποιήθηκε – τουλάχιστον για το προηγούμενο χρονικό διάστημα – και συνεπώς είναι άχρηστο. Ετσι ολα αυτά τα GCs, με *reference bit* μηδέν, πρέπει να σβηστούν από τους τρεις πίνακες, καθώς και να σταλούν στον X server εντολές αποδέσμευσης του χώρου που αυτά δέσμευαν.

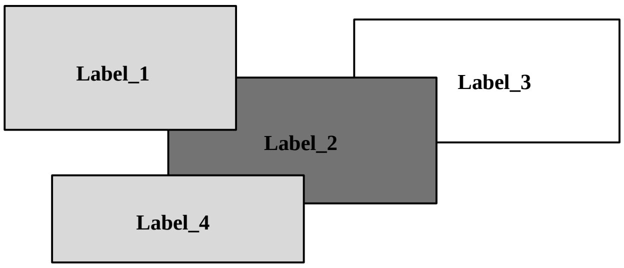
Η διάσχιση των πινάκων *line_hash, text_hash* και *half_used_hash* για την εύρεση και την διαγραφή των άχρηστων GCs, ανά ορισμένο χρονικό διάστημα, είναι χρονοβόρα και για τον λόγο αυτό ανεπιθύμιτη. Ωστόσο, με τον τρόπο αυτό, το σύστημα δεν δεσμεύει απεριόριστα πολύ χώρο στον X server, καθώς και οι παραπάνω πίνακές δεν μεγαλώνουν συνεχώς. Ο Λαβύρινθος δεν έχει, ακόμα, ένα τέτοιο μηχανισμό για την ανίχνευση των άχρηστων GCs. Κρίνουμε όμως πως είναι απαραίτητος για την καλή λειτουργία του συστήματος και στο μέλλον πρέπει να υλοποιηθεί.

[12] Οι δύο πίνακες περίεχουν όλα τα GCs που χρησιμοποιεί το σύστημα.

5. Μηχανισμοί για Επικαλυπτόμενα κύτταρα

5.1 Περιγραφή του προβλήματος

Ενα από τα σημαντικά μειονεκτήματα που ο χρήστης διαπιστώνει στο σχεδιαστικό μέρος του Λαβύρινθου είναι η απουσία πολιτικής για τον χειρισμό των επικαλυπτομένων κυττάρων του σχεδίου. Το σχήμα 5.1 είναι ένα παράδειγμα του τί είναι τα επικαλυπτόμενα κύτταρα.

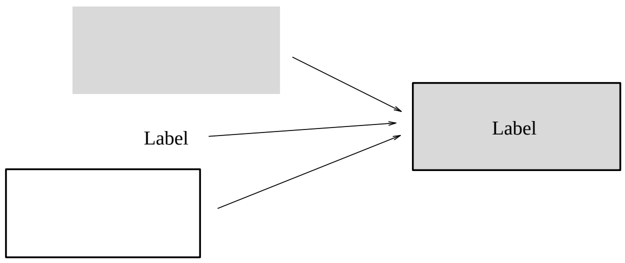


Σχήμα 5.1: Παράδειγμα επικαλυπτόμενων κυττάρων

Στο σχέδιο που δημιουργεί ένας χρήστης μπορεί να υπάρχουν κύτταρα των οποίων μέρη ή ολόκληρα τα κύτταρα βρίσκονται "σκεπασμένα" από άλλα. Η σειρά επικάλυψης μπορεί να είναι τυχαία ή σκόπιμη. Στην δεύτερη περίπτωση, το σύστημα πρέπει να την ξέρει και να την διατηρεί.

Στην παρούσα μορφή ο Λαβύρινθος συμπεριφέρεται με "σχετικά τυχαίο" τρόπο κατά τις περιπτώσεις επικάλυψης κυττάρων σε ένα σχέδιο: το κύτταρο, που επανυπολογίστηκε πιο πρόσφατα βρίσκεται πάντα πάνω από τα υπόλοιπα κύτταρα του σχεδίου. Ωστόσο, με την εντολή επανασχεδίασης ολόκληρου του σχεδίου (redraw), η σειρά επικάλυψης είναι εκείνη με την οποία τα κύτταρα ορίστηκαν κατά την σύνθεση του σχεδίου. Υπάρχουν σχεδιαστικά προγράμματα πού έχουν ανάλογη συμπεριφορά, όπως το xfig [17], και άλλα όπως το MacPaint, το MacDraw, το ReadySetGo!!!, στα οποία ο χρήστης μπορεί να επιλέγει την σειρά με την οποία τα κύτταρα επικαλύπτονται, με δύο εντολές: "τοποθέτησε στην κορυφή" και "τοποθέτησε στο βάθος".

Το σχήμα 5.2 δείχνει μια από τις περιπτώσεις όπου η σειρά με την οποία σχεδιάζονται τα κύτταρα του σχεδίου είναι σημαντική: είναι ο τρόπος με τον οποίο συνθέτουμε ένα από τα κύτταρα του σχήματος 5.1. Κάθε ένα από τα κύτταρα αυτά αποτελείται από τρία άλλα πρωτογενή κύτταρα: ένα "ορθογώνιο παραλληλόγραμμο περίγραμμα", ένα "κείμενο" και μια "ορθογώνια παραλληλόγραμμη επιφάνεια". Πρώτα πρέπει να σχεδιαστεί η "ορθογώνια παραλληλόγραμμη επιφάνεια", μετά το "κείμενο" και τέλος το "ορθογώνιο παραλληλόγραμμο περίγραμμα".



Σχήμα 5.2: Ορισμός σύνθετου κυττάρου

Αν η σειρά αυτή ήταν διαφορετική, π.χ. αν σχεδιάζονταν πρώτα το "κείμενο", και μετά η "ορθογώνια παραλληλόγραμμη επιφάνεια", θα προέκυπτε ένα κύτταρο στο οποίο το "κείμενο" δεν θα φαινόταν, γιατί θα καλύπτονταν από την δεύτερη.

Βλέπουμε λοιπόν ότι, κάθε σύνθετο κύτταρο θα πρέπει να φυλάσσει τη σειρά επανασχεδίασης των άμεσων παιδιών του στις εσωτερικές του δομές. Παρακάτω, παρουσιάζουμε τις αλλαγές που θα πρέπει να υποστεί ο Λαβύρινθος για να μπορεί να υποστηρίζει τον δυναμικό ορισμό της σειράς επανασχεδίασης των κυττάρων.

5.2 Επανασχεδίαση επικαλυπτόμενων κυττάρων

5.2.1 ‘‘Επίπεδο Σχεδίασης’’

Η σειρά σχεδίασης μπορεί να περιγραφεί με έναν ακέραιο για κάθε κύτταρο, που να δηλώνει την σειρά επανασχεδίασής του σχετικά με τα άλλα κύτταρα του σχεδίου που έχουν κοινό πατέρα. Την πληροφορία αυτή θα την ονομάζουμε "επίπεδο σχεδίασης". Στο παράδειγμα του σχήματος 5.2, το "επίπεδο σχεδίασης" για κάθε ένα από τα συνιστώντα κύτταρα του σύνθετου κυττάρου είναι: "ορθογώνια παραλληλόγραμμη επιφάνεια": 0, "κείμενο": 1, και "ορθογώνιο παραλληλόγραμμο περίγραμμα": 1. Το μικρότερο "επίπεδο σχεδίασης" δηλώνει ότι το κύτταρο πρέπει να σχεδιαστεί πρώτο. Κύτταρα, που δεν αλληλοεπηρεάζονται, μπορούν να έχουν (βρίσκονται στο) ίδιο "επίπεδο σχεδίασης" ("κείμενο" και "ορθογώνιο παραλληλόγραμμο περίγραμμα"). Για τα κύτταρα στο παράδειγμα του σχήματος 5.1 τα "επίπεδο σχεδίασης" είναι: Label_4: 2, Label_1: 2, Label_2: 1, Label_3: 0.

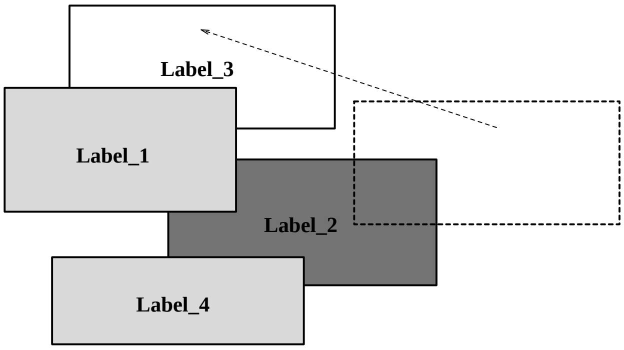
Αν χρήστης θέλει να μεταβάλλει την σειρά που σχεδιάζονται τα συνιστώντα κύτταρα του σχεδίου, αρκεί να αλλάξει το "επίπεδο σχεδίασής" τους. Η αλλαγή αυτή θα είναι δυνατή μέσω μιάς εντολής. Ο χρήστης θα επιλέγει ένα κύτταρο και δίνοντας την εντολή: "επίπεδο σχεδίασης" = < ακέραιος >, θα μπορεί να μεταβάλλει το "επίπεδο σχεδίασης" του επιλεγμένου κυττάρου. Η αρχική τιμή για το "επίπεδο σχεδίασης" κάθε κυττάρου θα είναι 0.

Ετσι ο μηχανισμός επανασχεδίασης του συστήματος θα πρέπει να μεταβληθεί, ώστε να λαμβάνει υπ' όψη του το "επίδεδο σχεδίασης" κάθε κυττάρου στο σχέδιο. Συγκεκριμένα, για την επανασχεδίαση ενός σύνθετου κυττάρου θα πρέπει να διασχίζονται όλα τα συνιστώντα κύτταρά του, να ταξινομούνται και να καλούνται οι διαδικασίες σχεδίασής τους. Για λόγους ταχύτητας, θα πρέπει τα παιδιά του να κρατιώνται μόνιμα ταξινομιμένα σε μία "λίστα σειράς σχεδίασης".

Ο Λαβύρινθος δεχόμενος, μέσω μίας εντολής, το νέο "επίδεδο σχεδίασης" για ένα κύτταρο, θα καταχωρεί τον αριθμό αυτό, ως "επίδεδο σχεδίασης", στην εσωτερική αναπαράσταση του κυττάρου και θα αναδιοργανώνει κατάλληλα την "λίστα της σειράς σχεδίασης". Ακόμα, θα υπάρχουν οι εντολές: "τοποθέτησε στην κορυφή" και "τοποθέτησε στο βάθος", με τις οποίες το επιλεγμένο κύτταρο θα μπαίνει στο τέλος, ή στη αρχή, της λίστας αντίστοιχα. Ωστόσο, στην περίπτωση που δύο κύτταρα με ίδιο "επίπεδο σχεδίασης" επικαλύπτονται, το σύστημα αρχικά θα τα σχεδιάζει με την σειρά με την οποία τα κύτταρα ορίστηκαν κατά την δημιουργία του σύνθετου κυττάρου, ενώ θα υπάρχουν και οι εντολές: "τοποθέτησε στην κορυφή του επιπέδου" και "τοποθέτησε στο βάθος του επιπέδου", οι οποίες θα διατάσουν "σχετικά" κύτταρα με ίδιο "επίδεδο σχεδίασης". Η διαδικασία σχεδίασης για τα σύνθετα κύτταρα πρέπει να μεταβληθεί, ώστε να ακολουθεί την "λίστα της σειράς σχεδίασης".

5.2.2 Μηχανισμός Επανασχεδίασης

Κατά την επανασχεδίαση λόγω επανυπολογισμού παρουσιάζονται ορισμένα προβλήματα, εξ' αιτίας της επικάλυψης κυττάρων. Το σχήμα 5.3 δείχνει τα δύο κυριότερα από αυτά. Εδώ, ο χρήστης μετακίνησε το κύτταρο Label_3. Το κύτταρο αυτό πρέπει να σβηστεί από την παλαιά του θέση και να επανασχεδιαστεί στη νέα. Η "περιοχή σβησίματος" φαίνεται περιγεγραμμένη με διακεκκομένη γραμμή.



Σχήμα 5.3: Προβλήματα της υποστήριξης των επικαλυπτόμενων κυττάρων

Το πρώτο πρόβλημα αφορά τα κύτταρα που τέμνονται με την "περιοχή σβησίματος", και κατά συνέπεια πρέπει να επανασχεδιαστούν. Το βήμα (ε) του μηχανισμού επανυπολογισμού και επανασχεδίασης που παρουσιάσαμε στην παραγράφο 3.2.2, δεν μπορεί να εφαρμοστεί όπως είναι στην περίπτωση που υπάρχουν επικαλυπτόμενα κύτταρα μέσα στην "περιοχή σβησίματος", γιατί στο βήμα αυτό επανασχεδιάζαμε τα κύτταρα που "επηρεάζονται" από το σβήσιμο της περιοχης, χωρίς να λαμβάνουμε υπ’ όψη μας το "επίπεδο σχεδίασής" τους. Ετσι, τα παραπάνω κύτταρα μπορεί να εμφανιστούν σε "τυχαία" διάταξη. Για την αντιμετώπιση του προβλήματος αυτού πρέπει όταν ψάχνουμε για όλα τα παιδιά που τέμνουν την περιοχή σβησίματος, να ψάχνουμε ακολουθώντας την "λίστα σειράς σχεδίασης", και να εξασφαλίζουμε ότι θα σχεδιαστούν με την σειρά που τα συναντάμε.

Ο Pike στο [13] υποστηρίζει την διατήρηση των περιοχών που καλύπτονται, σε δομές του συστήματος, υπό την μορφή διδιάστατων πινάκων σημείων (bitmaps). Η λύση αυτή δεν μπορεί να εφαρμοστεί στην περίπτωσή μας, γιατί πρώτον, το περιβάλον X δεν υποστηρίζει γρήγορες λειτουργίες πάνω σε διδιάστατους πινάκες σημείων, και δεύτερον διότι οι περιοχές που καλύπτονται μπορούν με εύκολο τρόπο να επανέλθουν στην αρχική τους κατάσταση (με επανασχεδίαση των περιεχομένων της "περιοχής σβησίματος").

Το δεύτερο πρόβλημα έχει να κάνει με τα κύτταρα που επανυπολογίστηκαν και πρέπει να επανασχεδιαστούν στις νέες τους θέσεις (κύτταρα της λίστας επανασχεδίασης). Ο μηχανισμός που περιγράψαμε στην παράγραφο 3.2.2 δεν μπορεί, και πάλι, να εφαρμοστεί εδώ, διότι τα κύτταρα της λίστας επανασχεδίασης έχουν "τυχαία" διάταξη, και έτσι κατά την επανασχεδίασή τους προκύπτουν ανάλογα προβλήματα με αυτά που περιγράψαμε προηγουμένως. Εδώ έχουμε μια επιπλέον δυσκολία: οι θέσεις των κυττάρων που επανυπολογίστηκαν μπορεί να συμπίπτουν με τις θέσεις άλλων κυττάρων που παρέμειναν αμετάβλητα κατά την διαδικασία επανυπολογισμού και δεν ανήκουν στην "περιοχή σβησίματος". Κάποια από τα κύτταρα αυτά μπορεί να έχουν "επίπεδο σχεδίασης" μεγαλύτερο από εκείνο των κυττάρων που επανυπολογίστηκαν, και να πρέπει επομένως να είναι "από πάνω", ενώ αν απλά σχεδιάσουμε τα κύτταρα που επανυπολογίστηκαν τότε τα αμετάβλητα κύτταρα θα βρεθούν από "κάτω". Στο παράδειγμα του σχήματος 5.3, το κύτταρο Label_1 μπορεί να βρεθεί "σκεπασμένο" από το Label_3, μετά την μετακίνηση του τελευταίου.

Για να λυθεί το πρόβλημα αυτό, το σύστημα θα πρέπει να υπολογίζει το ελάχιστο ορθογώνιο περίγραμμα που περιβάλλει όλα τα κύτταρα της λίστας επανασχεδίασης ("περιοχή επανασχεδίασης"). Μέσω αυτού, μπορεί να εντοπίσει τα κύτταρα που "επηρεάζονται" από την επανασχεδίαση των κυττάρων της λίστας αυτής και να τα επανασχεδιάσει. Για να σχεδιαστούν με την σωστή σειρά τα κύτταρα αυτά, πρέπει να "προστεθούν" στην λίστα επανασχεδίασης μαζί με τα υπόλοιπα, προς επανασχεδίαση, κύτταρα. Το σύστημα πρέπει να κρατάει τα στοιχεία της λίστας αυτής διατεταγμένα, ανάλογα με την "επίπεδο σχεδίασης" τους. Αυτό μπορεί να γίνει εύκολα κατά το δημιουργία της λίστας: τα κύτταρα με μικρότερο "επίπεδο σχεδίασης" πρέπει να μπαίνουν στην αρχή της λίστας, ενώ εκείνα με μεγαλύτερο "επίπεδο σχεδίασης" προς το τέλος. Το σύστημα χρειάζεται, απλά, να παρέχει ένα μηχανισμό που θα διατάσσει τα στοιχεία της λίστας επανασχεδίασης, κατά την διάρκεια της κατασκευής της.

Ωστόσο, οι λύσεις που μόλις προτείναμε δεν λαμβάνουν υπ’ όψη τους την περίπτωση στην οποία το ορθογώνιο περίγραμμα της "περιοχής σβησίματος" τέμνει εκείνο της λίστας επανασχεδίασης. Στην περίπτωση αυτή, κύτταρα με μεγάλο "επίπεδο σχεδίασης", που επανασχεδιάστηκαν λόγω της "περιοχής σβησίματος", μπορεί να τύχει να καλύπτονται από κύτταρα της λίστας επανασχεδίασης με μικρότερο "επίπεδο σχεδίασης". Για να αποφύγουμε μια τέτοια περίπτωση προτείνουμε, η λίστα επανασχεδίασης να περιλαμβάνει και τα κύτταρα που τέμνουν την "περιοχή σβησίματος". Ετσι, όλα τα κύτταρα θα σχεδιαστούν με ένα πέρασμα της λίστας επανασχεδίασης και με την κατάλληλη σειρά, εφόσον η λίστα αυτή δημιουργείται με βάση το "επίπεδο σχεδίασής" τους.

Για να λειτουργήσει σωστά ο μηχανισμός που παρουσιάσαμε παραπάνω, πρέπει να λύσουμε ακόμη ένα πρόβλημα: να αποκλείσουμε την περίπτωση "ανεπιθύμιτων παρενεργειών" (side effects) κατά την σχεδίαση της νέας λίστας επανασχεδίασης. Οι "παρενέργειες" αυτές οφείλονται στο ότι, καθώς ένα κύτταρο από την λίστα επανασχεδίασης σχεδιάζεται ολόκληρο στην οθόνη, μπορεί να καλύπτει ένα μέρος ενός άλλου κυττάρου με μεγαλύτερο "επίπεδο σχεδίασης" από το δικό του, και το οποίο δεν τέμνεται με καμία από τις δύο παραπάνω "περιοχές" (σβησίματος και επανασχεδίασης). Ορίζοντας ως "περίοχή ψαλιδίσματος" τα ορθογώνια περιγράμματα των δύο αυτών περιοχών, κανένα κύτταρο έξω από την περιοχή αυτή δεν θα υποστεί αλλαγές εξαιτίας της επανασχεδίασης ορισμένων κυττάρων του σχεδίου.

Ετσι, ο αλγόριθμος της παραγράφου 2.1.3 πρέπει να συμπληρωθεί, ώστε παράλληλα με την "περιοχή σβησίματος", να δημιουργεί και την "περιοχή επανασχεδίασης" ως εξής:

Για κάθε κύτταρο στην λίστα επανυπολογισμού:

Εάν το κύτταρο είναι πρωτογενές **τότε**

α. Κάλεσε την διαδικασία υπολογισμού του (επανυπολόγισέ το) και πρόσθεσέ το στην λίστα επανασχεδίασης, με βάση το "επίπεδο σχεδίασής" του

β. "Πρόσθεσε" το παλαιό ορθογώνιο περίγραμμά του στην "περιοχή σβησίματος"

γ. **Εάν** το κύτταρο είναι ορατό **τότε** κάλεσε την διαδικασία υπολογισμού του νέου ορθογωνίου περιγράμματός του (υπολόγισε το περίγραμμά του) και "πρόσθεσε" το νέο περίγραμμα στη "περιοχή επανασχεδίασης"

Εάν το κύτταρο είναι σύνθετο **τότε**

(όμοια με την παράγραφο 2.1.3) ...

Αντίστοιχα ο μηχανισμός επανυπολογισμού και επανασχεδίασης, που περιγράφουμε στην παράγραφο 3.2, μεταβάλλεται με βάση τα παραπάνω ως εξής:

α. Δώσε αρχικές τιμές στην "περιοχή σβησίματος" και στην "περιοχή επανασχεδίασης"

β. Επανυπολόγισε τα κύτταρα στην λίστα επανυπολογισμού (αλγόριθμος της παραγράφου 2.1.3)

γ. Σβήσε τα περιεχόμενα της "περιοχής σβησίματος" από την οθόνη

δ. **Για κάθε** κύτταρο στην "λιστα της σειράς σχεδίασης" των συνιστώντων κυττάρων που συνθέτουν το "κύτταρο εργασίας"

Εάν το ορθογώνιο περίγραμμά του τέμνει, είτε την "περιοχή σβησίματος", είτε την "περιοχή επανασχεδίασης", **και** δεν ανήκει στην λίστα επανασχεδίασης **τότε**

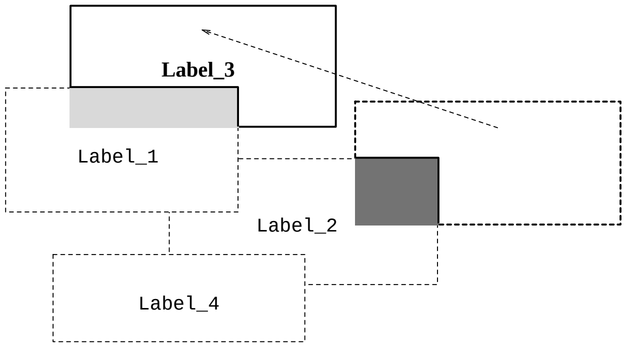
Πρόσθεσέ το στην λίστα επανασχεδίασης, με βάση το "επίπεδο σχεδίασής" του, χωρίς να ενημερώσεις την "περιοχή επανασχεδίασης" για την αλλαγή αυτή ε. Ορισε την "περιοχή σβησίματος" καθώς και την "περιοχή επανασχεδίασης" ως "περιοχή ψαλιδίσματος"

στ. **Για κάθε** κύτταρο την λίστα επανασχεδίασης

Κάλεσε την διαδικασία επανασχεδίασής του

ζ. Ακύρωσε την "περιοχή ψαλιδίσματος" (ορισέ να είναι ολόκληρο το σχεδιαστικό παράθυρο του συστήματος).

Ο μηχανισμός αυτός στό βήμα (β) υπολογίζει σταδιακά τις περιοχές "σβησίματος" και "επανασχεδίασης" και αφού σβήσει τα κύτταρα που επανυπολογίστηκαν από τις παλαιές τους θέσεις, επανασχεδιάζει εκείνα μόνο τα κύτταρα που βρίσκονται στην λίστα επανασχεδίασης (βήμα (στ)). Ωστόσο, στο βήμα (δ) "αυξάνει" την λίστα αυτή με τα κύτταρα που μπορεί να επηρεάζονται από την επικείμενη επανασχεδίαση.



Σχήμα 5.4: Επανασχεδίαση επικαλυπτόμενων κυττάρων

Ας δούμε πώς ο μηχανισμός αυτός εφαρμόζεται στο παράδειγμά μας. Στο σχήμα 5.4 φαίνονται τα κύτταρα που πρέπει να επανασχεδιαστούν μετά τον επανυπολογισμό του κυττάρου Label_3. Η "περιοχή σβησίματος" φαίνεται με διακεκομμένο περίγραμμα, ενώ η "περιοχή επανασχεδίασης" είναι το νέο ορθογώνιο περίγραμμα του Label_3. Στο σχήμα φαίνεται πώς λειτουργεί η "περιοχή ψαλιδίσματος" (clipping area) για το παράδειγμά μας. Εδώ, η λίστα επανασχεδίασης περιλαμβάνει αρχικά το κύτταρο Label_3 που επανυπολογίστηκε (βήμα (β)). Στο βήμα (δ), ο μηχανισμός προσθέτει στην λίστα το Label_2 και το Label_1. Αρα στο βήμα (ε), η λίστα περιέχει τα: Label_3, Label_2, και Label_1, με αυτή τη σειρά. Στο βήμα αυτό, ορίζεται η "περιοχή ψαλιδίσματος" να αποτελείται από τα ορθογώνια περιγράμματα των περιοχών "σβησίματος" και "επανασχεδίασης". Στο βήμα (στ) σχεδιάζονται τα στοιχεία της λίστας επανασχεδίασης. Οπως βλέπουμε, πρώτα σχεδιάζεται ολόκληρο το κύτταρο Label_3, και μετά μέρη μόνο των κυττάρων Label_2 και Label_1, λόγω του ψαλιδίσματος.

Ο νέος μηχανισμός επανυπολογισμού και επανασχεδίασης κυττάρων δεδομένου του ότι τα ορθογώνια περιγράμματα των κυττάρων παρέχουν προσεγγιστικά και μόνο την επιφάνεια που αυτά καταλαμβάνουν λειτουργεί με αποδοτικό τρόπο, διότι σχεδιάζει μόνο όσα κύτταρα – ολόκληρα ή μερη αυτών – είναι απαραίτητο να επανασχεδιαστούν, αποφεύγοντας έτσι πιθανές "ανεπιθύμιτες παρενέργειες". Για το παράδειγμα αυτό, μία τέτοια "παρενέργεια" θα ήταν η επανασχεδίαση ολόκληρου του κυττάρου Label_2, του οποίου ένα μέρος σβήστηκε από την "περιοχή σβησίματος". Η επανασχεδίαση ολόκληρου του Label_2 θα "χάλαγε" την διάταξη των κυττάρων του σχεδίου, γιατί το κύτταρο Label_4 θα βρισκόταν κάτω από το το Label_2 ενώ έχει μεγαλύτερο "επίπεδο σχεδίασης".

5.3 Αλλοι μηχανισμοί για την υποστήριξη επικαλυπτόμενων κυττάρων

Ανάλογα με τον μηχανισμό επανυπολογισμού και επανασχεδίασης χρειάζεται να μεταβληθούν , οι μηχανισμοί επιλογής και δημιουργίας κώδικα PostScript ώστε να λαμβάνουν υπ’ όψη τους την σχετική διάταξη των κυττάρων στην οθόνη.

5.3.1 Επιλογή, Δημουργία κώδικα PostScript

Τα επικαλυπτόμενα κύτταρα δίνουν στο χρήστη την εντύπωση ότι ορισμένα κύτταρα του σχεδίου είναι κοντά του, ενώ άλλα βρίσκονται "προς τα πίσω". Ο μηχανισμός επιλογής του Λαβύρινθου πρέπει να ενισχύει την εντύπωση αυτή, επιλέγοντας το "πάνω" κύτταρο όταν υπάρχουν περισσότερα από ένα κύτταρα στο σημείο επιλογής.

Η μετατροπή που απαιτείται στη διαδικασία επιλογής της παραγράφου 2.2.2 είναι σχετικά μικρή: χρειάζεται αλλαγή της σειράς διάσχισης της ιεραρχίας των κυττάρων κατά την αναζήτηση πιθανών κυττάρων προς επιλογή. Η διάσχιση αυτή, η οποία μέχρι τώρα γινόταν με "τυχαίο" τρόπο, θα πρέπει να γίνεται ακολουθώντας τα "επίπεδα σχεδίασης" των κυττάρων. Η διαδικασία επιλογής θα πρέπει να εντοπίζει τα επιλεγμένα κύτταρα ακολουθώντας αναδρομικά την "λίστα της σειράς σχεδίασης", για κάθε κύτταρο (παιδί) του κυττάρου που επεξεργάζεται ο χρήστης. Προφανώς, με τον τρόπο αυτό, καταχωρούνται στον πίνακα επιλεγμένων, πρώτα τα κύτταρα που είναι "στο βάθος" του σχεδίου και τελευταία τα κύτταρα "στην κορυφή". Σε περίπτωση που τα επιλεγμένα κύτταρα είναι περισσότερα από 64, ο πίνακας επιλεγμένων συνεχίζει να συμπληρώνεται κυκλικά, αγνοώντας έτσι κύτταρα με μικρά "επίπεδα σχεδίασης" Ο μηχανισμός επιλογής δεν έχει παρά να επιλέγει το κύτταρο που βρίσκεται στη τελευταία τρέχουσα έγκυρη θέση του πίνακα αυτού.

Ετσι η διαδικασία επιλογής πρέπει να μετατραπεί ως εξής:

Δοθέντος του σημείου επιλογής

α. Δημιούργησε ένα BBOX γύρω από το σημείο αυτό (με διαστάσεις 2*pointingTolerance* $\times$ 2*pointingTolerance*)

β. Ακύρωσε τα προηγούμενα περιεχόμενα του πίνακα

γ. **Για κάθε** κύτταρο (παιδί) στην "λίστα της σειράς σχεδίασης" του "κυττάρου εργασίας"

Εάν το BBOX του κυττάρου τέμνει το BBOX επιλογής **τότε**

(όμοια με την παράγραφο 2.2.2) ...

Ανάλογη μετατροπή πρέπει να γίνει και στην διαδικασία δημιουργίας κώδικα PostScript. Η σειρά με την οποία καταχωρούνται στο αρχείο οι εντολές PostScript που περιγράφουν τα κύτταρα του σχεδίου καθορίζει τη διάταξή τους στη εκτύπωση. Το αρχείο με τις εντολές PostScript που δημιουργεί ο Λαβύρινθος διαβάζεται από τον εκτελεστή εντολών του εκτυπωτή και εκτελείται σειριακά. Για τον λόγο αυτό, ο μηχανισμός δημιουργίας κώδικα PostScript του συστήματος πρέπει να διασχίζει την ιεραρχία των κυττάρων ακολουθώντας (αναδρομικά) την "λίστα της σειράς σχεδίασης", και για κάθε παιδί του "κυττάρου εργασίας" να καλεί την διαδικασία παραγωγής κώδικα PostScript του.

5.3.2 Δυναμικός Ορισμός του ‘‘Επιπέδου Σχεδίασης’’

Ο χρήστης θα πρέπει να έχει την δυνατότητα να αλλάζει δυναμικά την διάταξη των επικαλυπτόμενων κυττάρων του σχεδίου εφαρμόζοντας την εντολή: "επίπεδο σχεδίασης" = < ακέραιος > στο επιλεγέν κύτταρο. Ακόμα, με την εντολή "επίπεδο σχεδίασης" = ?, ο χρήστης θα μπορεί να βλέπει το "επίπεδο σχεδίασης" του επιλεγμένου κυττάρου.

Το "επίπεδο σχεδίασης" είναι ένα χαρακτηριστικό που έχουν όλα τα ορατά κύτταρα στο σύστημα. Ωστόσο, ο μοναδικός τρόπος που ο Λαβύρινθος παρέχει στον χρήστη για να μεταβάλλει τα χαρακτηριστικά των κυττάρων του σχεδίου είναι δίνοντας τιμές στις πόρτες τους. Αυτό θα μας οδηγούσε στο να υπάρχει μία πόρτα σε κάθε κύτταρο της οποίας η τιμή να δίνει το "επίπεδο σχεδίασης" του κυττάρου. Κάτι τέτοιο θα ήταν δύσχρηστο, εφόσον η αναγκαιότητα του "επιπέδου σχεδίασης" για κάθε κύτταρο του συστήματος θα ανάγκαζε τον χρήστη να ορίζει μια τέτοια πόρτα σε κάθε σύνθετο κύτταρο που δημιουργεί. Ο χρήστης, ωστόσο, είναι επηρεπής σε παραλλείψεις κατά την διαδικασία σχεδίασης, και γι’ αυτό δεν μπορούμε να τον εμπιστευόμαστε, δεδομένου του ότι οι περισσότεροι μηχανισμοί, όπως παρουσιάστηκαν νωρίτερα, βασίζονται στην ύπαρξη αυτού του "επιπέδου σχεδίασης". Το "επίπεδο σχεδίασης" πρέπει να είναι ένα εγγενές χαρακτηριστικό του κυττάρου, που ο χρήστης θα μπορεί να αλλάζει χωρίς να επιλέγει μια συγκεριμένη πόρτα για τον σκοπό αυτό.

Ακόμα, η αλλαγή στο "επίπεδο σχεδίασης" ενός κυττάρου θα συνεπάγεται και αναδιοργάνωση που σχεδίου στην οθόνη. Ενας απλός τρόπος για να γίνει αυτό θα ήταν, κάθε φορά που δίνεται μια εντολή αλλαγής "επιπέδου σχεδίασής", να εκαλείτο ο μηχανισμός επανασχεδίασης όλου του σχεδίου. Αυτό όμως θα συνεπάγετο σοβαρή καθυστέρηση για το σύστημα. Ο καλύτερος τρόπος είναι να κληθεί η διαδικασία επανασχεδίασης του συγκεκριμένου κυττάρου και οι αντίστοιχες διαδικασίες εκείνων που τέμνονται με αυτό. Ετσι, κάθε φορά που ο χρήστης δίνει μια εντολή αλλαγής του "επιπέδου σχεδίασης" για ένα συγκεκριμένο κύτταρο, ο Λαβύρινθος θα καλεί την παρακάτω διαδικασία:

Για το συγκεκριμένο κύτταρο

α. Ορισε το ορθογώνιο περίγραμμά του ως την "περιοχή σβησίματος"

β. Σβήσε τα περιεχόμενα της "περιοχής σβησίματος" από την οθόνη

γ. **Για κάθε** κύτταρο στην "λιστα της σειράς σχεδίασης" των συνιστώντων κυττάρων που συνθέτουν το "κύτταρο εργασίας"

> **Εάν** το ορθογώνιο περίγραμμά του τέμνει την "περιοχή σβησίματος" **και** δεν ανήκει στην λίστα επανασχεδίασης **τότε**
>
> > Πρόσθεσέ το στην λίστα επανασχεδίασης, με βάση το "επίπεδο σχεδίασής" του

δ. Ορισε την "περιοχή σβησίματος" ως "περιοχή ψαλιδίσματος"

στ. **Για κάθε** κύτταρο την λίστα επανασχεδίασης

Κάλεσε την διαδικασία επανασχεδίασής του

στ. Ακύρωσε την "περιοχή ψαλιδίσματος" (ορισέ να είναι ολόκληρο το σχεδιαστικό παράθυρο του συστήματος).

Με τον μηχανισμό αυτό, δεδομένου του ότι τα ορθογώνια περιγράμματα των κυττάρων παρέχουν προσεγγιστικά και μόνο την επιφάνεια που αυτά καταλαμβάνουν, το σχέδιο θα υφίσταται τις λιγότερες δυνατές μεταβολές καθώς έχουμε τις λιγότερες δυνατές επανασχεδιάσεις κυττάρων.

5.3.3 Προβλήματα που παραμένουν

Ο Λαβύρινθος, όπως αναφέραμε σε προηγούμενη παράγραφο, θα μπορεί να χρησιμοποιεί widgets ως κύτταρα. Η χρήση τους δημιουργεί μία σειρά προβλημάτων που αφορούν στο χειρισμό των επικαλυπτόμενων κυττάρων: οι οντότητες–widgets έχουν για γραφική αναπαράσταση μία ιεραρχία από αδιαφανή παράθυρα που καλύπτουν οτιδήποτε είναι σχεδιασμένο πάνω στο παράθυρο από το οποίο δημιουργήθηκαν. Πιο συγκεκριμένα, τα κύτταρα–widgets του Λαβύρινθου θα καλύπτουν τα κύτταρα που είναι σχεδιασμένα πάνω στο σχεδιαστικό παράθυρο του συστήματος, ανεξάρτητα από το "επίπεδο σχεδίασης" τους, γεγονός που δημιουργεί ένα σοβαρό πρόβλημα στον μηχανισμό υποστήριξης των επικαλυπτομένων κυττάρων.

Το πρόβλημα συνίσταται στην ύπαρξη δύο τύπων κυττάρων: αυτά που αναπαρίστανται με παράθυρα και αυτά που αναπαρίστανται με σχεδιαστικές εντολές. Το "επίπεδο σχεδίασης" μπορεί να εφαρμοστεί με επιτυχία για το κάθε ένα από τα δύο είδη ξεχωριστά αλλά όχι μεταξύ τους. Ο μηχανισμός επανασχεδίασης θα λαμβάνει υπ’ όψη του το "επίπεδο σχεδίασης" των κυττάρων–widgets και θα τα αναδιατάσσει, κάνοντας χρήση των εντολών XtRaizeWidget και XtLowerWidget [8]. Αντίστοιχα, τα υπόλοιπα κύτταρα του συστήματος θα διατάσσονται με βάση τον μηχανισμό που περιγράψαμε στην προηγούμενη παράγραφο. Ωστόσο, τα κύτταρα–widgets θα καλύπτουν πάντοτε τα υπόλοιπα κύτταρα του συστήματος, δημιουργώντας μια ασυνέπεια για τον μηχανισμό σχεδίασης για τα επικαλυπτόμενα κύτταρα. Δυστυχώς, το πρόβλημα αυτό δεν είναι δυνατόν να λυθεί με τις υπάρχουσες επιλογές στον Λαβύρινθο.

6. Μελλοντικές Δυνατότητες Εξέλιξης

Στην εργασία αυτή παρουσιάστηκαν: το τμήμα της επικοινωνίας του χρήστη και του γραφικού editor του Λαβύρινθου, οι μηχανισμοί επιλογής και επανασχεδίασης, οι μηχανισμοί υποστήριξης πολλαπλών σχεδιαστικών χαρακτηριστικών και πολύπλοκων σχεδιαστικών αντικειμένων, ο μηχανισμός για την δημιουργία κώδικα PostScript, καθώς και οι αλλαγές που χρειάζεται να γίνουν στους παραπάνω μηχανισμούς για να μπορεί ο Λαβύρινθος να χειρίζεται επικαλυπτόμενα κύτταρα.

Επίσης μελετήσαμε την απόδοση των αλγόριθμων που επιλέκτηκαν. Ωστόσο, οι λύσεις που παρουσιάστηκαν αποτελούν μία βάση για παραπέρα ανάλυση και σχεδιασμό του γραφικού τμήματος του Λαβύρινθου. Οπως αναφέραμε, το σύστημα αναπτύσσεται και επεκτείνεται συνεχώς, και καθώς οι απαιτήσεις του μεταβάλλονται, ορισμένες από τις ιδέες και τους αλγόριθμους που παρουσιάστηκαν ενδέχεται να αναπροσαρμοστούν. Ηδη αναφέρθηκε ο τρόπος με τον οποίο ο Λαβύρινθος θα μπορεί να υποστηρίζει σχεδιαστικά αντικείμενα, όπως τα widgets. Ακόμα αναφερθήκαμε στις αλλαγές που πρέπει να υποστούν οι μηχανισμοί που περιγράφονται στην εργασία αυτή, ώστε ο Λαβύρινθος να χειρίζεται επικαλυπτόμενα κύτταρα.

Ενα από τα κύρια μειονεκτήματα του γραφικού editor του Λαβύρινθου είναι ότι δεν έχει δωθεί η πρέπουσα σημασία στην φιλικότητα της διαλογικής επικοινωνίας του συστήματος. Ο γραφικός editor είναι μάλλον δύσχρηστος. Η χρήση παλεττών με γραφικά αντικείμενα, από τις οποίες ο χρήστης θα μπορεί εύκολα να διαλέγει αυτά που θα συνθέσουν το σχέδιό του, θα βελτιώσουν πολύ την φιλικότητα του Λαβύρινθου. Σε αυτό θα συμβάλει και η ύπαρξη μενού εντολών για τον χειρισμό των γραφικών αντικείμενων, για την φύλαξη (και φόρτωση) του σχεδίου σε (και από) αρχείο, καθώς και για άλλες χρήσιμες λειτουργίες.

Ακόμα χρειάζεται να γίνουν αλλαγές στις εσωτερικές δομές του συστήματος και τους μηχανισμούς διαχείρισής τους, ώστε να μπορούν να υποστηρίζουν πολλαπλά παράθυρα για την εμφάνιση των γραφικών αντικειμένων. Ετσι ο μελλοντικός χρήστης – σχεδιαστής θα μπορεί να βλέπει πολλαπλές όψεις ενός σχεδίου.

Θα χρειαστεί να γίνει μελέτη τόσο της χρήσης όσο και της επέκτασης των τριών Hash Tables που ο Λαβύρινθος χρησιμοποιεί για να έχει γρήγορη πρόσβαση στα γραφικά χαρακτηριστικά των σχεδίων. Εκτός από τα χαρακτηριστικά που παρουσιάσαμε στην εργασία αυτή (χρώμα, πολλαπλά είδη γραμμών και γραφής), ο Λαβύρινθος πρέπει να υποστηρίζει και άλλα ανάλογα χαρακτηριστικά, όπως η δυνατότητα χρωματισμού της επιφάνειας που καλύπτει ένα τόξο, τα οποία υποστηρίζονται από το περιβάλλον X, μέσω της δομής Graphic Contexts (GC).

Ακόμα χρειάζεται βελτίωση ο μηχανισμός μεγένθυσης – σμίκρυνσης: κατά την μεγένθυση ενός σχεδίου τόσο τα πάχη των γραμμών, όσο και τα μεγέθη των κειμένων πρέπει να μεγαλώνουν κατά τρόπο ανάλογο, ακριβώς όπως κοιτάμε την λεπτομέρεια ενός σχεδίου με ένα μεγενθυτικό φακό.

Τέλος, το σύστημα θα πρέπει να μπορεί να υποστηρίζει την ομαλή μετακίνηση γραφικών αντικειμένων (smooth dragging - rubber banding). Αυτό συνεπάγεται βέβαια την δημιουργία μιας νέας διαδικασίας για τα πρωτογενή κύτταρα, η οποία θα σκιαγραφεί με γρήγορο τρόπο την γραφική αναπαράσταση του κυττάρου χρισημοποιώντας XOR για την σχεδίαση και μετέπειτα το σβήσιμό του, καθώς και αλλαγές στους μηχανισμούς επανυπολογισμού και επανασχεδίασης.

Γλωσσάρι

Δημιουργία κώδικα PostScript Ο Λαβύρινθος μπορεί να παράγει ένα αρχείο που περιέχει το σχέδιο του χρήστη σε κώδικα PostScript. Ετσι ο χρήστης μπορεί να εκτυπώσει το σχέδιό του σε ένα εκτυπώτη ή να το συμπεριλάβει ως εικόνα σε ένα κείμενό του.

Επανασχεδίαση Ο Λαβύρινθος έχει ένα μηχανισμό επανασχεδίασης των κυττάρων που μεταβλήθηκαν από την διαδικασία επανυπολογισμού του συστήματος. Σκοπός του είναι να ενημερώσει το παράθυρο σχεδίασης για πιθανές αλλαγές στην γραφική αναπαράσταση του κυττάρου εργασίας.

Επανυπολογισμός Ο Λαβύρινθος έχει ένα μηχανισμό επανυπολογισμού που ενεργοποιείται κάθε φορά που ο χρήστης μεταβάλλει τις τιμές των πορτών εισόδου ενός κυττάρου. Ο μηχανισμός αυτός υπολογίζει το κύτταρο αυτό καθώς και όσα κύτταρα συνδέονται με αυτό και επηρεάζονται από τις τιμές των πορτών εξόδου του πρώτου.

Επιλογή Ο Λαβύρινθος έχει ένα μηχανισμό επιλογής με τον οποίο ο χρήστης μπορεί να επιλέγει ένα από τα συνιστώντα κύτταρα του "κυττάρου εργασίας" και να μεταβάλλει τόσο τα χαρακτηριστικά του όσο και τον τρόπο με τον οποίο αυτό συνδέεται με άλλα κύτταρα. Η επιλογή γίνεται μέσω του ποντικιού.

Εφαρμογή X Είναι μια γραφική εφαρμογή που συνεργάζεται με το γραφικό περιβάλλον X. Οι εφαρμογές X στέλνουν αιτήσεις για γραφικά στον X server (π.χ. αιτήσεις εμφάνισης γραμμών, τόξων, κειμένου κ.α.), και τον πληροφορούν ότι ενδιαφέρονται για ορισμένα γεγονότα, όπως π.χ. η αλλαγή των διαστάσεων ενός παραθύρου. Οι εφαρμογές X μπορούν να τρέχουν σε ένα υπολογιστή και να εμφανίζουν τα γραφικά τους στην οθόνη ενος άλλου, ενώ δέχονται την είσοδό τους από το αντίστοιχο πληκτρολόγιο ή ποντίκι.

Κύτταρο Κύτταρα (cells) ονομάζονται οι οντότητες στον Λαβύρινθο. Αυτές μπορεί να έχουν γραφική αναπαράσταση ή όχι (ορατά, αόρατα). Εχουν παραμέτρους που ονομάζουμε πόρτες – επικοινωνίας – μέσω των οποίων μπορούν να καθορίζονται τα χαρακτηριστικά τους. Τα κύτταρα στον Λαβύρινθο είναι ιεραρχικά οργανωμένα σε πρωτογενή και σύνθετα.

Κύτταρο Εργασίας Είναι το σύνθετο εκείνο κύτταρο (σχέδιο) το οποίο ο χρήστης επεξεργάζεται (μεταβάλλει) σε μία δεδομένη στιγμή μέσω του γραφικού editor του Λαβύρινθου. Η γραφική αναπαράστασή του εμφανίζεται στο παράθυρο σχεδίασης του γραφικού editor. Ο χρήστης έχει την δυνατότητα να επιλέγει τα συνιστώντα κύτταρά του, να αλλάζει τις τιμές που έχουν οι πόρτες τους, να τα συνδέει μεταξύ τους ή να τα αποσυνδέει κλπ.

Λίστα Επανασχεδίασης Είναι μία λίστα που δημιουργείται κατά την διαδικασία επανυπολογισμού των κυττάρων του σχεδίου, και περιέχει τα κύτταρα που επανυπολογίστηκαν και χρειάζονται επανασχεδίαση.

Οντότητα–widget (ή widget) Τα widgets είναι προγράμματα που περιέχονται σε διάφορες βιβλιοθήκες που διανέμονται μαζί με το σύστημα παραθύρων X. Αυτά τα προγράμματα ξέρουν να διαχειρίζονται αυτόνομες οντότητες με συγκεκριμένη λειτουργικότητα, τις οποίες ο προγραμματιστής μπορεί να χρησιμοποιεί, διαμορφώνοντάς τις κατάλληλα για να ταιρίαζουν στις ανάγκες ενός προγράμματος. Παραδείγματα τέτοιων widgets είναι οι κυλιόμενοι ενδεικτές (ScrollBars) και τα μενού (Menu's). Ο προγραμματιστής επωφελείται της λειτουργικότητας αυτής κάνοντας χρήση του ειδικού τρόπου επικοινωνίας που παρέχει το κάθε widget.

Ορθογώνιο Περίγραμμα Κυττάρου (ή BBOX) Ως ορθογώνιο περίγραμμα ενός κυττάρου ορίζουμε το ελάχιστο ορθογώνιο παραλληλόγραμμο με πλευρές παράλληλες στους άξονες των X και Y που το περιβάλλει πλήρως. Ο ορισμός αυτός ισχύει για κύτταρα που έχουν ένα σχήμα ορατό στην οθόνη. Το σύστημα κρατάει το ορθογώνιο περίγραμμα κάθε κυττάρου σε μία δομή (structure BBOX), η οποία περιέχει τις ελάχιστες και μέγιστες τιμές του ορθογώνιου περιγράμματος του κυττάρου στους άξονες X και Y. Οι τιμές αυτές, που χαρακτηρίζουν τις πλευρές του, εκφράζονται σε συντεταγμένες κόσμου.

Παράθυρα, Περιβάλλον Παραθύρων Με τον όρο παράθυρα εννοούμε ορθογώνιες περιοχές στην οθόνη. Σε ένα περιβάλλον παραθύρων, ο χρήστης μπορεί να δημιουργήσει περιοχές στην οθόνη (παράθυρα) και να τις χρησιμοποιεί ως ανεξάρτητες οθόνες των οποίων τα χαρακτηριστικά μπορεί να μεταβάλλει κατά προτίμιση. Το περιβάλλον αυτό είναι συνήθως πολύ φιλικό για τους χρήστες, οι οποίοι επικοινωνούν με αυτό σχεδον αποκλειστικά, μέσω του ποντικιού και των μενού.

Παράθυρο Εντολών Είναι η περιοχή του γραφικού editor του συστήματος, στην οποία εμφανίζονται οι εντολές που έδωσε ο χρήστης, καθώς και μηνύματα λάθους. Οι εντολές δίνονται μέσω του πληκτρολογίου.

Παράθυρο Σχεδίασης Είναι η περιοχή σχεδίασης του γραφικού editor του συστήματος. Στο παράθυρο αυτό στέλνονται οι εντολές δημιουργίας γραφικών από τις διαδικασίες σχεδίασης των πρωτογενών κυττάρων. Εδώ ο χρήστης μπορεί, με την βοήθεια του ποντικιού, να υποδείξει ένα σημείο της οθόνης, και το σύστημα αυτόματα επιλέγει το κύτταρο που βρίσκεται πιο κοντά στο σημείο αυτό. Στο κάτω μέρος της περιοχής σχεδίασης υπάρχει μια άλλη περιοχή, όπου εμφανίζονται πληροφορίες σχετικές με τα επιλεγμένα κύτταρα.

Περιοχή Σβησίματος (erase area) Είναι η περιοχή του σχεδίου που χρειάζεται να "σβηστεί". Σε αυτή περιλαμβάνονται τα κύτταρα που επανυπολογίστηκαν και θα επανασχεδιαστούν σε νέες θέσεις ή με άλλα γραφικά χαρακτηριστικά. Η περιοχή σβήσίματος είναι το μικρότερο ορθογώνιο περίγραμμα που περιλαμβάνει τα ορθογώνια περιγράμματα των κυττάρων αυτών.

Περιοχή Ψαλιδίσματος (clipping area) Είναι η περιοχή που ορίζουμε να εφαρμόζονται οι σχεδιαστικές λειτουργίες. Οποιαδήποτε σχεδιαστική λειτουργία έξω από την περιοχή αυτή "ψαλιδίζεται" (δεν πραγματοποιείται).

Συντεταγμένες Κόσμου και Οθόνης Οι πόρτες τύπου σημείου των κυττάρων περιγράφονται σε συντεταγμένες κόσμου (world coordinates). Το σύστημα τις μετατρέπει σε συντεταγμένες οθόνης (screen coordinates – pixels) όταν πρόκειται να σχεδιάσει στην οθόνη.

Σχεδιαστικά Συμφραζόμενα (ή GCs) Είναι μία δομή που περιλαμβάνει χαρακτηριστικά σχεδίασης, όπως το πάχος γραμμής, το είδος γραμμής, το χρώμα, το είδος γραφής, η "περιοχή ψαλιδίσματος", κ.α. Οι γραφικές εντολές προς τον X server, παραδείγματος χάριν οι εντολές σχεδίασης γραμμών, χρησιμοποιούν τις τιμές των πεδίων των GCs για να καθορίζουν την λειτουργία τους (δηλαδή τί γραμμές θα σχεδιάσουν και πώς). Τα πεδία της δομής αυτής είναι "ιδιωτικές" (private) στην βιβλιοθήκη Xlib, και ο προγραμματιστής, που θέλει να δώσει τιμές σ' αυτά, πρέπει να καλεί τις διαδικασίες διαχείρισής τους, που παρέχει η Xlib.

X server Είναι πρόγραμμα "υπηρέτης" που είναι υπεύθυνο να σχεδιάζει γραφικά στη οθόνη (ή οθόνες) ενός υπολογιστή. Το πρόγραμμα αυτό γνωρίζει τα χαρακτηριστικά του υλικού (hardware) του υπολογιστή στον οποίο τρέχει και αναλαμβάνει να ικανοποιεί τις αιτήσεις για γραφικά που έρχονται από διάφορες εφαρμογές X. Οι συσκευές για τις οποίες είναι υπεύθυνο είναι το ποντίκι, το πληκτρολόγιο και η μία ή οι περισσότερες οθόνες πού είναι συνδεδεμένες με τον υπολογιστή στον οποίο τρέχει. Επικοινωνεί με τις εφαρμογές X μέσω του X πρωτοκόλου και τις ενημερώνει για γεγονότα που συμβαίνουν στις παραπάνω συσκευές.

Παράρτημα A

Υλοποίηση της Κλάσσης BBOX

```
/* class Implementation of BBOX */
/* -------------------------------------- */

class BBOX {
	private:
		Wcoor xmin;
		Wcoor ymin;
		Wcoor xmax;
		Wcoor ymax;
	public:
		inline BBOX();
		inline void set();
		inline void set(Wcoor xMin, Wcoor yMin, Wcoor xMax, Wcoor yMax);
		inline void get(Wcoor *xMin, Wcoor *yMin, Wcoor *xMax, Wcoor *yMax);
		void expand_by(BBOX* bbox);
		BOOLEAN intersects(BBOX* bbox);
		BOOLEAN update_parent_bbox(BBOX* child_old_bbox, BBOX* child_new_bbox);
};

/*
 * Constructor for the BBOX class. Gives to a newly constructed BBOX its initial values.
 */
inline BBOX::BBOX()
{
	xmin = ymin = MAX_WCOOR;
	xmax = ymax = MIN_WCOOR;
}

/*
 * Initialization function : initializes an "illegal" BBOX.
 * The initial BBOX gets its correct value by the first expand_by(), otherwise it remains illegal.
 */
inline void BBOX::set()
{
	set(MAX_WCOOR, MAX_WCOOR, MIN_WCOOR, MIN_WCOOR);
}
```

```
/*
 * Function that passes the parameter values in the appropriate fields of 'this'.
 */
inline void BBOX::set(Wcoor xMin, Wcoor yMin, Wcoor xMax, Wcoor yMax)
{
	xmin = xMin;
	xmax = xMax;
	ymin = yMin;
	ymax = yMax;
}

/*
 * Function that returns into the parameter the values of the appropriate fields of 'this'.
 */
inline void BBOX::get(Wcoor *xMin, Wcoor *yMin, Wcoor *xMax, Wcoor *yMax)
{
	*xMin = xmin;
	*xMax = xmax;
	*yMin = ymin;
	*yMax = ymax;
}

/*
 * Gets the maximum of the two BBOXes (this and other) and assigns it to 'this'.
 */
void BBOX::expand_by(BBOX* other)
{
	if (other->xmin < xmin)
		xmin = other->xmin;
	if (other->ymin < ymin)
		ymin = other->ymin;
	if (other->xmax > xmax)
		xmax = other->xmax;
	if (other->ymax > ymax)
		ymax = other->ymax;
}

/*
 * Checks if 'this' BBOX intersects with the 'other' BBOX.
 */
BOOLEAN BBOX::intersects(BBOX* other)
{
return( (xmin <= other->xmax) && (other->xmin <= xmax) &&
       (ymin <= other->ymax) && (other->ymin <= ymax) );
}
```

```
/*
 * This function, given the old and the new BBOXes of a child, checks if the parent BBOX is to
 * be changed. It updates the parent BBOX ('this') and returns a boolean value indicating if
 * the updated BBOX is correct (TRUE), or not (this means that it can't be determined by the
 * given data) and therefore further action must be taken (FALSE).
 */
BOOLEAN BBOX::update_parent_bbox(BBOX* child_old_bbox, BBOX* child_new_bbox)
{
	if (child_new_bbox->xmin <= child_old_bbox->xmin)
		xmin = min(xmin, child_new_bbox->xmin);
	else if (xmin == child_old_bbox->xmin)
		return(FALSE);
	if (child_new_bbox->ymin <= child_old_bbox->ymin)
		ymin = min(ymin, child_new_bbox->ymin);
	else if (ymin == child_old_bbox->ymin)
		return(FALSE);
	if (child_new_bbox->xmax >= child_old_bbox->xmax)
		xmax = max(xmax, child_new_bbox->xmax);
	else if (xmax == child_old_bbox->xmax)
		return(FALSE);
	if (child_new_bbox->ymax >= child_old_bbox->ymax)
		ymax = max(ymax, child_new_bbox->ymax);
	else if (ymax == child_old_bbox->ymax)
		return(FALSE);

	return(TRUE);
}
```

Παράρτημα Β

Περίληψη Πρωτογενών Κυττάρων

Ορατά Πρωτογενή Κύτταρα

‘‘ευθύγραμμο τμήμα’’ (line) : Σχεδιάζει το ευθύγραμμο τμήμα ανάμεσα στα δύο σημεία εισόδου *p* και *q*, με χρώμα *col*, πάχος *wid*, και είδος γραμμής *sty*.

‘‘βέλος’’ (arrow) : Σχεδιάζει ένα βέλος ανάμεσα από το σημείο εισόδου *p* προς το σημείο εισόδου *q*, με χρώμα *col*, πάχος *wid*, και είδος γραμμής *sty*. Η "κορυφή" του (tip) έχει μήκος *l* και κλίση *r*.

‘‘κύκλος’’ (circle) : Σχεδιάζει τον κύκλο με κέντρο *c*, που περνάει από το σημείο *p*, με δοθέντα χρώμα, πάχος και είδος γραμμής.

‘‘τόξο τριών σημείων’’ (arc) : Σχεδίαζει ένα τόξο δοθέντων τριών σημείων *p1*, *p2*, *p3* της περιφέριάς του, με δοθέντα χρώμα, πάχος, και είδος γραμμής. Τα σημεία *p1* και *p3* είναι τα άκρα του τόξου, ενώ το σημείο *p2* είναι εσωτερικό σημείο του τόξου.

‘‘τόξο από κέντρο και περιεχόμενη γωνία’’ (arcc) : Σχεδίαζει ένα τόξο δοθέντος του κέντρου *c* του κύκλου, ενός άκρου *p1* του τόξου ,και της περιεχόμενης γωνίας *angl*, με δοθέντα χρώμα, πάχος και είδος γραμμής. Το πρόσημο της γωνίας καθορίζει τον προσανατολισμό του τόξου.

‘‘ορθογώνιο παραλληλόγραμμο περίγραμμα’’ (rect) : Σχεδιάζει ένα ορθογώνιο παραλληλόγραμμο περίγραμμα με πλευρές παράλληλες στους αξονες X και Y, χρώματος *col*, δεδομένων δύο αντιδιαμετρικών κορυφών του *p* και *q*. Παράγει ως εξόδους τις άλλες δύο κορυφές, τα πέντε μεσαία σημεία, καθώς και το ύψος και το πλάτος του παραλληλογράμμου περιγράμματος.

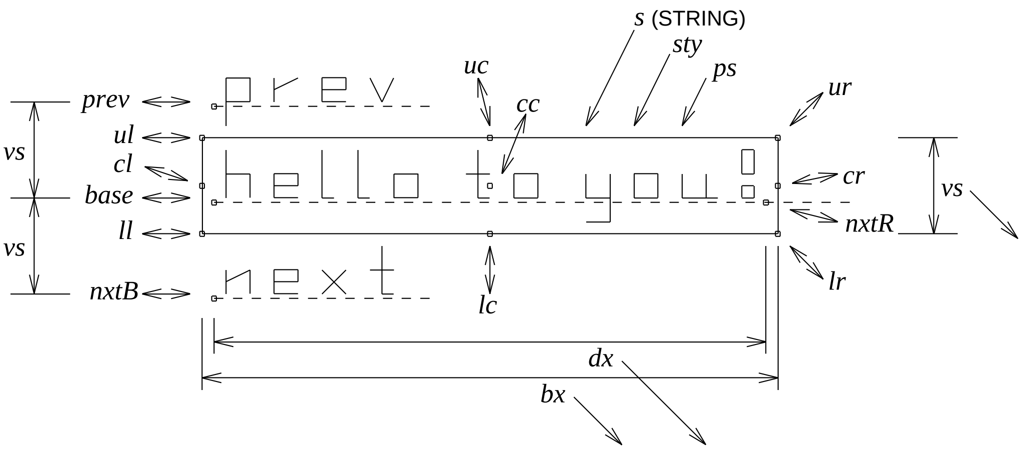
‘‘ορθογώνια παραλληλόγραμμη επιφάνεια 1’’ (rectmf – Manhattan filled rectangle) : Σχεδιάζει ένα αδιαφανές ορθογώνιο παραλληλόγραμμο με πλευρές παράλληλες στους αξονες X και Y, "γεμίζοντάς" το με χρώμα *col*, δεδομένων δύο αντιδιαμετρικών κορυφών του *p* και *q*. Παράγει ως εξόδους τις τιμές των σημείων στις δύο άλλες κορυφές, τα πέντε μεσαία σημεία, καθώς και το ύψος και το πλάτος του παραλληλογράμμου.

‘‘ορθογώνια παραλληλόγραμμη επιφάνεια 2’’ (rectmfc) : Σχεδιάζει ένα αδιαφανές ορθογώνιο παραλληλόγραμμο με πλευρές παράλληλες στους αξονες X και Y, "γεμίζοντάς" το με χρώμα *col*, δεδομένου του κέντρου του *c*, του ύψους *dy* και του πλάτους του *dx*. Παράγει ως εξόδους τις τέσσερεις κορυφές, καθώς και τα τέσσερα μεσαία σημεία.

‘‘κείμενο’’ (text) : Τυπώνει δοθέν κείμενο *s* στο σημείο *p*, με χρώμα *col*, οικογένεια γραφής *sty* (style – font), και μέγεθος γραφής *ps* (point-size).

‘‘δείκτης’’ (marker) : Σχεδιάζει ένα μικρό σημάδι στο σημείο *p*.

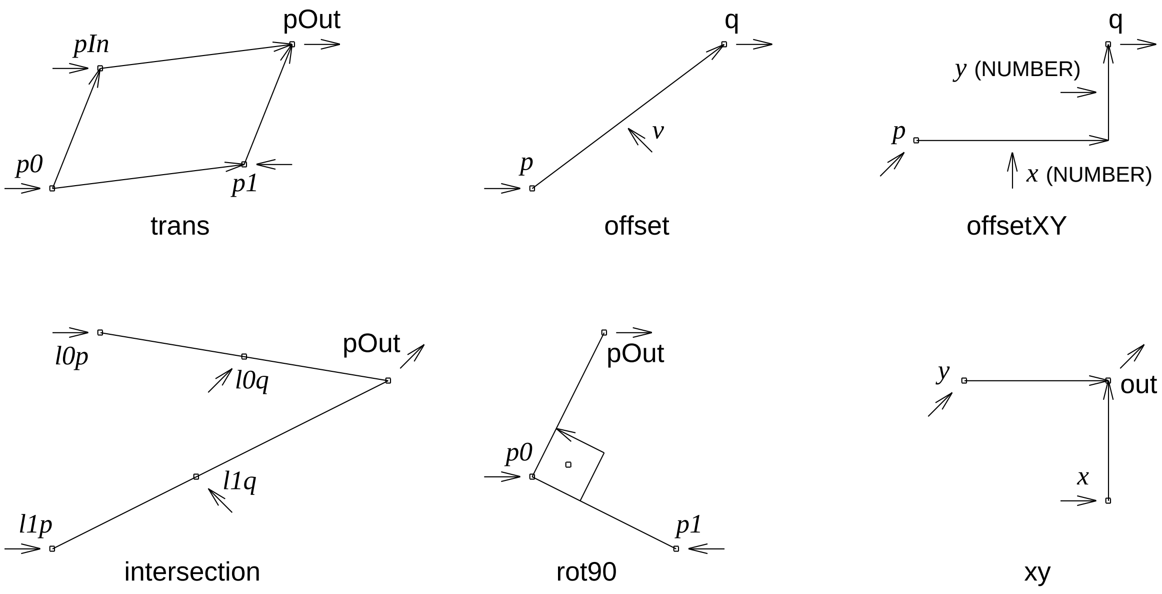
‘‘μεγεθος κειμένου’’ (textSize) : Δοθέντος ενός κειμένου, της οικογένειας γραφής και του μεγέθους γραφής, και ενός από τα δεκατρία "χαρακτηριστικά" σημεία του κυττάρου αυτού, παράγει τα υπόλοιπα σημεία του (σχήμα B.1).



Σχήμα B.1: Κύτταρο ‘‘μεγεθος κειμένου’’ (textSize)

trans : Κάνει παράλληλη μετάθεση ενός διανύσματος *p0*→*p1* σε *pIn*→*pOut* (σχήμα B.2).

offset : Παράγει το σημείο *q* προσθέτοντας το διάνυσμα *v* στο σημείο *p* (σχήμα B.2).



Σχήμα B.2: Κύτταρα trans, offset, offsetXY, ‘‘τομή’’, ‘‘ορθή γωνία’’, xy

offsetXY : Παράγει το σημείο *q* προσθέτοντας στο σημείο *p*, αντί ενός διανύσματος, δύο αριθμούς *x*,*y* που περιγράφουν το διάνυσμα (τα μεγέθη εκφράζονται σε χιλιοστά) (σχήμα

B.2).

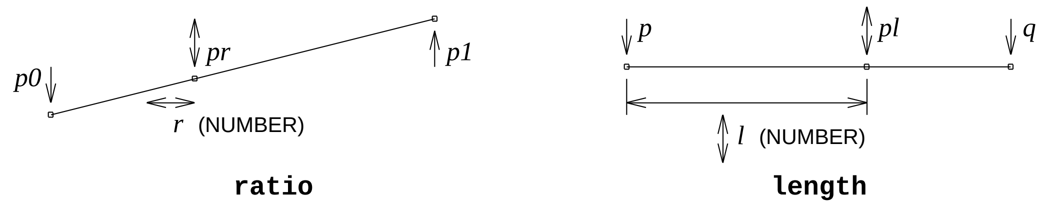
‘‘τομή’’ (intersection) : Παράγει το σημείο *pOut*, ως σημείο τομής της ευθείας *line-0* που ορίζεται από τα σημεία *l0p* και *l0q*, και της ευθείας *line-1* που ορίζεται από τα σημεία *l1p* και *l1q* (σχήμα B.2).

‘‘ορθή γωνία’’ (rot90) : Περιστρέφει αντίθετα με την φορά του ρολογιού κατά 90° το σημείο *p1* γύρω από το *p0* και παράγει το *pOut* (σχήμα B.2).

xy : Το σημείο *out* παράγεται από την τομή μιας οριζόντιας γραμμής που περνάει από το σημείο *y* και μιας κάθετης που περνάει από το *x* (σχήμα B.2).

‘‘λόγος’’ (ratio) : Δεδομένων δύο σημείων *p0* και *p1* παράγει ένα συνευθειακό σημείο *pr* τέτοιο ώστε το διάνυσμα *p0*→*pr* να είναι *r* φορές το διάνυσμα *p0*→*p1* (σχήμα B.3).

‘‘μήκος’’ (length) : Παράγει το μήκος *l* σε χιλιοστά του διανύσματος *p*→*pl*, με δεδομένα τα σημεία *p* και *q*, και το περιεχόμενο σημείο *pl* (σχήμα B.3).



Σχήμα B.3: Κύτταρα ‘‘λόγος’’ και ‘‘μήκος’’

extremum : Παράγει τις μέγιστες και ελάχιστες τιμές των συντεταγμένων (*x*,*y*) για τα σημεία εισόδου *in0*, *in1*, *in2* και *in3*.

floatAscii : Μετατρέπει ένα αριθμό *n* στην ASCII αναπαράστασή του *s* και αντίστροφα.

‘‘πολλαπλασιασμός’’ (mul) : Παράγει ένα αριθμό *a*b* πολλαπλασιάζοντας τους αριθμούς *a* και *b*.

Αναφορές

[1] Autodesk, Inc., *AutoCAD, Reference Manual*, 1988.

[2] A. Borning, ‘‘The Pragramming Language Aspects of ThingLab, A Constraint Oriented Simulation Laboratory,’’ *ACM Transactions on Programming Language and System*, vol. 3, no. 4, pp. 353-387, October 1981.

[3] P. Constantopoulos, Martin Doerr, Elena Pataki, Eleni Petra, George Spanoudakis, and Y. Vassiliou, ‘‘The Software Information Base – Selection Tool integrated prototype,’’ ITHACA.FORTH.91.E2.#3, Computer Science Institute, Foundation of Research and Technology, Hellas, January 1991.

[4] Frame Technology Corp., *FrameMaker*, 1989.

[5] J. D. Foley and A. Van Dam, *Fundamentals of Interactive Computer Graphics*, Addison-Wesley Publ. Co., Inc, 1984.

[6] Adobe Systems Inc., *PostScript Language, Reference Manual*, Addison-Wesley Publ. Co., Inc, 1985.

[7] M. Katevenis, T. Sorilos, C. Georgis, and P. Kalogerakis, ‘‘LABY User’s Manual,’’ ITHACA.FORTH.90.E.3.3.#7, Computer Science Institute, Foundation of Research and Technology, Hellas, December 1990.

[8] J. McCormack, P. Asente, and R. R. Swick, *X Toolkit Intrinsics - C Language Interface*, MIT X Consortium, 1989.

[9] G. Nelson, ‘‘Juno, a constraint-based graphics system,’’ *Computer Graphics: SIGGRAPH’85 Conference Proceedings*, vol. 19, no. 3, San Francisco, CA, July 1985.

[10] OSF, *Motif Programmer’s Guide*, Prentice Hall, 1990.

[11] OSF, *Motif Programmer’s Reference*, Prentice Hall, 1990.

[12] Ch. D. Peterson, *Athena Widget Set - C Language Interface*, MIT X Consortium, 1989.

[13] Rob Pike, ‘‘Graphics in overlapping bitmap layers,’’ *ACM Transactions on Graphics*, vol. 2, no. 2, pp. 135-160, April 1983.

[14] R. Scheifler, J. Gettys, and R. Newman, *X Window System - C Library and Protocol Reference*, Digital Press, 1989.

[15] T. Sorilos, ‘‘Object Oriented Design of Labyrinth’s Graphical Entities,’’ M. Sc. Thesis, Univercity of Crete, Computer Science Dept., Hellas, July 1991.

[16] B. Stroustrup, *The C++ Programming Language*, Addison-Wesley Publ. Co., Inc, 1986.

[17] ‘‘xfig - Facility for Interactive Generation of figures under X11,’’ manual pages.

[18] ‘‘xwd - dump an image of an X window, xpr - print a window dump,’’ User Commands, manual pages.