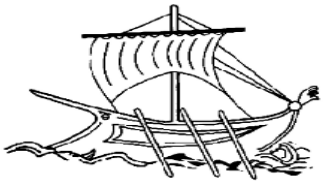


ΑΝΩΤΑΤΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΠΕΙΡΑΙΑ
ΤΕΧΝΟΛΟΓΙΚΟΥ ΤΟΜΕΑ

ΟΙ ΕΥΡΕΤΙΚΟΙ ΚΑΙ ΜΕΤΑΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ ΓΙΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΧΡΟΝΟΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΤΩΝ ΦΩΤΕΙΝΩΝ ΣΗΜΑΤΟΔΟΤΩΝ

ΠΑΣΧΟΣ ΑΛΕΞΑΝΔΡΟΣ
ΑΜ:44697
ΣΚΟΥΤΕΛΙΑΣ ΠΑΡΙΣ
ΑΜ:44698

Επιβλέπων καθηγητής: ΔΡΟΣΟΣ ΧΡΗΣΤΟΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΑΥΤΟΜΑΤΙΣΜΟΥ

ΜΑΡΤΙΟΣ 2017

ΤΜΗΜΑ ΑΥΤΟΜΑΤΙΣΜΟΥ
Π.Ράλλη & Θηβών 250,12244,Αιγάλεω,Αθήνα-Ελλάδα
Τηλ. 210-5381488

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΕΦΑΛΑΙΟ 1

ΕΥΡΕΤΙΚΕΣ ΚΑΙ ΜΕΤΑΕΥΡΕΤΙΚΕΣ ΜΕΘΟΔΟΙ

1.1. Ευρετικοί αλγόριθμοι

Η εύρεση της καλύτερης λύσης σε ένα δεδομένο πρόβλημα είναι ένα δυσεπίλυτο πρόβλημα. Δεδομένου ότι τα προβλήματα που παρουσιάζονται στις διάφορες σύγχρονες επιστημονικές κοινότητες είναι προβλήματα συνδυαστικά και πολύπλοκα, τότε καταλαβαίνει κανείς ότι η μελέτη όλων των πιθανών λύσεων με σκοπό την εύρεση της καλύτερης, είναι τις περισσότερες φορές απαγορευτική αν όχι αδύνατη. Αυτή η αδυναμία ώθησε στην ανάπτυξη των λεγόμενων ευρετικών αλγορίθμων. Τα δυο κυριότερα στοιχεία ενός αλγορίθμου είναι η δυνατότητα εύρεσης της βέλτιστης λύσης και ο υπολογιστικός χρόνος που χρειάζεται για την εύρεση αυτή. Οι ευρετικοί αλγόριθμοι προσανατολίζονται στην εύρεση μιας λύσης σε κάποιο υπολογιστικό χρόνο θυσιάζοντας πολλές φορές τον έναν από τους δυο παράγοντες. Βρίσκουν έτσι, αρκετά καλές λύσεις στα προβλήματα στα οποία εφαρμόζονται αλλά χωρίς απόδειξη ότι η λύσεις αυτές είναι πραγματικά καλές αλλά και δεν υπάρχει εγγύηση ότι θα δοθεί λύση σε σύντομο υπολογιστικό χρόνο. Η έρευνα έχει δείξει ότι ο υπολογιστικός χώρος αναζήτησης λύσεων για ρεαλιστικά προβλήματα είναι πολύ μεγάλος, ώστε να παρθούν λύσεις σε μικρούς σχετικά υπολογιστικούς χρόνους.

Οι ευρετικοί αλγόριθμοι εκμεταλλεύονται μια αρχική προσέγγιση ενός τυχαίου αριθμού του πληθυσμού και τυχαίας επιλογής του αρχικού χώρου λύσεων ώστε να φτάσουν με ευρετικούς μηχανισμούς σε μια καλύτερη περιοχή λύσεων μέσω του ανασυνδυασμού και της μετάλλαξης. Αυτό επιτυγχάνεται με την αξιολόγηση των ενδιάμεσων καταστάσεων ως προς την εκτιμώμενη απόστασή τους από μια τελικά κατάσταση και επεκτείνουν τις λύσεις που βρίσκονται κοντά σε αυτή, ενώ ταυτόχρονα μειώνουν την ισχύ αυτών που απέχουν πολύ ή και τις διαγράφουν

εντελώς από το δέντρο των λύσεων. Στην ουσία καταγωγή των αλγορίθμων είναι η θεωρία της μαθηματικής βελτιστοποίησης η οποία εντοπίζει το ελάχιστο ή και το μέγιστο μιας συνάρτησης διακριτής μεταβλητής. Το ρόλο της πραγματικής συνάρτησης στους αλγορίθμους αυτούς παίζει η (μετα) ευρετική συνάρτηση και ο χώρος των λύσεων παρουσιάζεται σαν ένας χώρος στον οποίο οι λύσεις μεταξύ τους διαφέρουν με βάση τη μεταξύ τους απόσταση, ενώ όσο καλύτερη ευρετική τιμή έχει μια λύση, τόσο υψηλότερα σε σχέση με τις άλλες τοποθετείται σε αυτό το χώρο του συνόλου των λύσεων. Μπορούμε να παρουσιάσουμε αυτό το χώρο των λύσεων σαν ένα γράφο με κορυφές τις καταστάσεις και ακμές τους τελεστές μετάβασης.

Σε μεγάλους χώρους αναζήτησης μπορούμε να πετύχουμε πολύ καλύτερη απόδοση αν εκμεταλλευτούμε τη γνώση που έχουμε σχετικά με το συγκεκριμένο πρόβλημα που προσπαθούμε να επιλύσουμε (domain specific knowledge). Οι ευρετικοί μηχανισμοί (heuristics) είναι κανόνες βασισμένοι σε τέτοιου είδους γνώση σύμφωνα με τους οποίους επιλέγουμε τον επόμενο κόμβο για επέκταση κατά τη διάρκεια της αναζήτησης. Ένα βασικό συστατικό αυτών των αλγορίθμων είναι μια ευρετική συνάρτηση (heuristic function) h, τέτοια ώστε:

- h(n) είναι το εκτιμώμενο κόστος του συντομότερου μονοπατιού από την κατάσταση στον κόμβο n ως μια κατάσταση στόχου
- Η h μπορεί να είναι οποιαδήποτε συνάρτηση τέτοια ώστε h(n) = 0 αν το n αντιστοιχεί στην κατάσταση-στόχο

Για να βρούμε όμως μια καλή ευρετική συνάρτηση χρειαζόμαστε πληροφορία σχετικά με το συγκεκριμένο πρόβλημα. Ένα παράδειγμα στο οποίο θα μπορούσε να εφαρμοστεί ένας τέτοιος αλγόριθμος είναι η συσκευασία στοιχείων που δεν έχουν κανονισμένο και το ίδιο σχήμα σε έναν περιορισμένο χώρο. Η εύρεση της καλύτερης λύσης είναι ένα πολύ δύσκολο πρόβλημα με αποτέλεσμα, αν κάποιος θέλει να βρει την τέλεια λύση να πρέπει να δοκιμάσει όλους τους πιθανούς συνδυασμούς. Αυτό για μικρό αριθμό αντικειμένων ίσως να φαντάζει απλό αλλά αν ο αριθμός αυτός αυξηθεί έστω και λίγο τότε το πρόβλημα που προκύπτει είναι δυσεπίλυτο. Η λογική που μπορεί να χρησιμοποιηθεί σε αυτήν την περίπτωση είναι να τοποθετηθούν πρώτα τα μεγαλύτερα αντικείμενα και προχωρώντας προς τα μικρότερα να μπορέσουμε να τα τοποθετήσουμε ανάλογα με το χώρο που διαθέτουμε. Αυτός είναι και ο τρόπος που

θα λειτουργούσε μια ευρετική συνάρτηση σε αυτήν την περίπτωση με τη διαφορά ότι η λύση που θα παίρναμε θα ήταν πιο άμεση.

Οι ευρετικές μέθοδοι είναι βασισμένες σε ευφυείς στρατηγικές αναζήτησης που χρησιμοποιούνται για την επίλυση υπολογιστικών προβλημάτων χρησιμοποιώντας διάφορες εναλλακτικές προσεγγίσεις. Υπάρχουν στην πραγματικότητα και περιπτώσεις στις οποίες ο ευρετικός αλγόριθμος τελικά θα επιστρέψει χαμηλής ποιότητας αποτελέσματα ή θα τα επιστρέψει με μεγάλη καθυστέρηση, παρόλα αυτά γίνεται αυτές οι δυσλειτουργίες να διορθωθούν για κάθε περίπτωση προβλήματος και ο αλγόριθμος τελικά να γίνει αποδοτικός. Η χρήση των ευρετικών αλγορίθμων είναι πλέον ευρέως διαδεδομένη και στην πραγματικότητα σε πολλές από τις περιπτώσεις προβλημάτων είναι και ο μόνος τρόπος προσέγγισης λύσεων σε ένα λογικό χρονικό διάστημα.

1.2. Μεταευρετικοί αλγόριθμοι

Οι μεταευρετικοί αλγόριθμοι είναι μια υποκατηγορία και παράλληλα μια εξέλιξη των ευρετικών αλγορίθμων και οι οποίοι δε βελτιώνουν μια λύση απευθείας, αλλά διαχειρίζονται την επιλογή ευρετικών υπο-μεθόδων για τη βελτίωση της τρέχουσας λύσης −«μια ευρετική που επιλέγει ευρετικές». Αυτές οι λεγόμενες «χαμηλού επιπέδου» ευρετικές αναλαμβάνουν την αναζήτηση στον χώρο των λύσεων.

Ο όρος μεταευρετικός αποτελεί σύνθεση των λέξεων «μετά» και «ευρίσκω» και χρησιμοποιείται έχοντας την έννοια της αναζήτησης σε υψηλότερο επίπεδο. Οι μεταευρετικές τεχνικές στηρίζονται στην παραδοχή ότι ενδεχομένως δεν θα εντοπίσουν την βέλτιστη λύση αλλά στην πράξη έχει αποδειχθεί ότι συχνά αποτελούν τον πλέον ενδεδειγμένο τρόπο για την αντιμετώπιση δύσκολων προβλημάτων συνδυαστικής βελτιστοποίησης. Οι Osman και Laporte περιγράφουν τις μεταευρετικές τεχνικές ως εξής: “Μια μεταευρετική τεχνική ορίζεται ως μια επαναληπτική διαδικασία δημιουργίας λύσεων, η οποία καθοδηγεί μια δευτερεύουσα ευρετική τεχνική συνδυάζοντας με ευφυή τρόπο διαφορετικούς τρόπους εξερεύνησης και ανάλυσης του χώρου αναζήτησης, ενώ στρατηγικές μάθησης χρησιμοποιούνται

προκειμένου να εντοπίσουν με αποδοτικό τρόπο λύσεις οι οποίες βρίσκονται κοντά στις βέλτιστες.”

Οι μεταευρετικές τεχνικές φαίνεται να έχουν ισχυρή δυναμική για την περαιτέρω εξάπλωση και χρήση τους. Σε αυτό το συμπέρασμα συνηγορεί το γεγονός ότι εκτός από τις παραδοσιακές μεταευρετικές τεχνικές (Simulated Annealing, Tabu Search, κ.α.) τα τελευταία χρόνια έχουν αναπτυχθεί νέες τεχνικές (Variable Neighborhood Search, Guided Local Search, κ.α.). Οι μεταευρετικές τεχνικές μπορούν να λειτουργήσουν είτε ανεξάρτητα είτε σε υβριδικούς συνδυασμούς είτε οργανωμένες σε διάφορα επίπεδα αφαίρεσης (Hyperheuristics). Θα πρέπει να τονιστεί ότι για ορισμένες τεχνικές τα όρια ανάμεσα στην τεχνητή νοημοσύνη, την υπολογιστική νοημοσύνη και τις μεταευρετικές τεχνικές δεν είναι σαφή. Για παράδειγμα οι γενετικοί αλγόριθμοι αναφέρονται στην βιβλιογραφία ως παράδειγμα τεχνικής και στις τρεις προαναφερθείσες κατηγορίες.

1.3. Παραδείγματα για τους ευρετικούς αλγορίθμους

1.3.1. Αλγόριθμοι κατασκευής

Οι αλγόριθμοι αυτοί επιχειρούν να δώσουν λύσεις στο υπό εξέταση πρόβλημα ξεκινώντας από μια κενή αρχική λύση και λειτουργώντας επαυξητικά και επαναληπτικά προσθέτει καίρια καθορισμένα τμήματα της λύσης χωρίς να επιστρέφει σε προηγούμενα βήματα της διαδικασίας μέχρι να δώσει μια ολοκληρωμένη λύση. Στην πιο απλή μορφή του αλγόριθμου οι λύσεις προστίθενται με τυχαία σειρά. Οι «άπληστοι» (greedy) ευρετικοί αλγόριθμοι κατασκευής προσθέτουν σε κάθε βήμα ένα κομμάτι λύσης, και έτσι επιτυγχάνεται το μέγιστο «μυωπικό» όφελος όπως αυτό υπολογίζεται από κάποια συγκεκριμένη ευρετική πληροφορία.

1.3.2. Lagrangian Relaxation

Η μέθοδος προσεγγίζει ένα δύσκολο πρόβλημα της βελτιστοποίησης υπό περιορισμούς μέσω ενός απλούστερου προβλήματος. Η μέθοδος Lagrangian relaxation (LR) είναι μια μαθηματική προγραμματιστική τεχνική για βελτιστοποίηση προβλημάτων που περιλαμβάνουν περιορισμούς (constrained optimization). Όμοια με τον μηχανισμό των τιμών σε μια αγορά, η μέθοδος αντικαθιστά τους "σκληρούς" περιορισμούς, όπως περιορισμούς στην διαθεσιμότητα των μηχανών, με την πληρωμή ορισμένων τιμών, δηλαδή των πολλαπλασιαστών Lagrange, βασιζόμενη στην "ζήτηση" για τη χρήση μιας μηχανής σε κάθε μονάδα χρόνου. Το αρχικό πρόβλημα μπορεί με αυτό τον τρόπο να αναλυθεί σε πολλά μικρότερα και ευκολότερα στην επίλυση υποπροβλήματα.

Πιο συγκεκριμένα, η μέθοδος αυτή είναι καλή για προβλήματα στα οποία μπορούμε να διαχωρίσουμε τους περιορισμούς σε δυο κατηγορίες: στους "καλούς" και "κακούς". Με την πρώτη κατηγορία το πρόβλημα λύνεται εύκολα ενώ με την προσθήκη των δεύτερων το πρόβλημα γίνεται πολύ περίπλοκο στη λύση του. Αυτό που γίνεται στη συνέχεια είναι ότι απομακρύνονται οι «κακοί» περιορισμοί και τοποθετούνται στην αντικειμενική συνάρτηση με αντίστοιχα βάρη τα οποία αντιπροσωπεύουν μια ποινή που λέγεται πολλαπλασιαστής Lagrange και που προστίθεται σε μια λύση που δεν ικανοποιεί το συγκεκριμένο περιορισμό.

1.3.3. Γενετικοί Αλγόριθμοι

Οι γενετικοί αλγόριθμοι αναφέρονται σε όλες τις διαδικασίες αναζήτησης λύσεων οι οποίες προσομοιώνουν τη φυσική εξελικτική διαδικασία επιλογής. Αναφέρονται λοιπόν σε συγκεκριμένα χαρακτηριστικά και μεθόδους εξέλιξης πληθυσμών ανάλογα με αυτούς που συναντάμε σε πληθυσμούς (συνήθως εντόμων) στη φύση. Οι γενετικοί αλγόριθμοι εφαρμόζονται σε πεπερασμένους πληθυσμούς υποψήφιων λύσεων και ιδιαίτερα, όταν εφαρμόζονται στον χρονοπρογραμματισμό, θεωρούν τις ακολουθίες ή τους διάφορους χρονοπρογραμματισμούς ως άτομα (individuals) ή μέλη ενός πληθυσμού (population). Κάθε άτομο του πληθυσμού έχει κάποια χαρακτηριστικά όπως είναι η υγεία του (fitness). Η υγεία ενός ατόμου του πληθυσμού μετράται από

την αντικειμενική συνάρτηση. Η διαδικασία δουλεύει επαναληπτικά και κάθε επανάληψη είναι μια γενιά (generation).

Όσων αφορά στις λύσεις που αντιμετωπίζονται από τον αλγόριθμο, σε κάθε επανάληψη του αλγορίθμου οι χειρότερες λύσεις μετακινούνται από τον πληθυσμό και αντικαθίστανται από νέες υποψήφιες λύσεις που παράγονται με: (1) εφαρμογή μεταλλάξεων (mutations) σε ορισμένες λύσεις του πληθυσμού, (2) εφαρμογή διασταυρώσεων (crossover) σε ζευγάρια λύσεων του πληθυσμού. Η ίδια διαδικασία συνεχίζεται για έναν προκαθορισμένο αριθμό επαναλήψεων και ο πιο επιτυχημένος υποψήφιος επιλέγεται ως η καλύτερη λύση. Όπως η εξελικτική διαδικασία στη φύση, ο πληθυσμός συνήθως παραμένει σταθερός, χάρη στον αποκλεισμό των πιο αδύναμων λύσεων που παράγονται (weaker child solutions), στην αναπαραγωγή ή τη μετάλλαξη των μελών του πληθυσμού της προηγούμενης γενιάς και στο γενικό κανόνα της επιβίωσης του υγιέστερου (survival of the fittest).

1.3.4. Αλγόριθμος Tabu Search

Στην αναζήτηση tabu ο αλγόριθμος αναζήτησης μπορεί να μετακινηθεί από μία τρέχουσα υποψήφια λύση σε οποιαδήποτε «γειτονική»-καλύτερη ή χειρότερη-στηριζόμενος μόνο στο ότι η μετάβαση στην επόμενη λύση δεν περιέχεται στην λίστα-tabu (tabu list). Σε οποιοδήποτε στάδιο της διαδικασίας κρατείται μια tabu list των μεταβολών (mutations) που η διεργασία δεν επιτρέπεται να επανεκτελέσει. Οι μεταβολές αυτές μπορεί να είναι για παράδειγμα ζεύγη διεργασιών που δεν μπορούν να υποστούν αμοιβαία ανταλλαγή. Οι πληροφορίες που μεταφέρονται κατά τη διάρκεια της εκτέλεσης της μεθόδου tabu search, συνίστανται από την tabu list καθώς και τη βέλτιστη λύση που λαμβάνεται μέσω της διαδικασίας αναζήτησης.

Όταν μία μετατροπή εφαρμόζεται σε μία τρέχουσα υποψήφια λύση αυτή η μετατροπή προστίθεται στην κεφαλή της λίστας και στη συνέχεια η ουρά της λίστας απομακρύνεται. Με αυτό τον τρόπο ο αλγόριθμος οδηγεί τη διαδικασία αναζήτησης μακριά από λύσεις που (με βάση τις διαθέσιμες πληροφορίες) εμφανίζονται να επαναλαμβάνονται ή να μοιάζουν σε λύσεις που έχουν βρεθεί σε προηγούμενα στάδια.

1.3.5. Αλγόριθμος Simulated Annealing

Η μέθοδος simulated annealing είναι μια μέθοδος αναζήτησης που έχει την καταγωγή της στα πεδία της φυσικής και της επιστήμης των υλικών. Αναπτύχθηκε πρώτα σαν μοντέλο προσομοίωσης για την περιγραφή της σκλήρυνσης με πυράκτωσης συμπυκνωμένου μετάλλου. Η τεχνική αυτή είναι μία παραλλαγή της τοπικής αναζήτησης. Στη μέθοδο simulated annealing επιτρέπονται μετακινήσεις προς χειρότερες λύσεις (ανηφορικές–uphill μετακινήσεις). Ο λόγος που επιτρέπονται τέτοιες μετακινήσεις είναι ότι δίνεται στη διαδικασία η δυνατότητα να μετακινηθεί μακριά από ένα τοπικό ελάχιστο και να βρει μια καλύτερη λύση αργότερα. Αν η τιμή της αντικειμενικής συνάρτησης της γειτονικής λύσης είναι καλύτερη από αυτή της τρέχουσας υποψήφιας λύσης τότε επιλέγεται η γειτονική. Αν δεν είναι, τότε επιλέγεται και πάλι με τη διαφορά ότι στην επόμενη επανάληψη μειώνεται η πιθανότητα να επιλεγεί η επόμενη χειρότερη γειτονική λύση. Με αυτό τον τρόπο, αν κατά τη διάρκεια της αναζήτησης βρίσκουμε συνεχώς χειρότερες γειτονικές λύσεις τότε κάποια στιγμή η πιθανότητα να επιλεχθεί μία επόμενη χειρότερη θα είναι μηδέν και ο αλγόριθμος θα σταματήσει εκεί και μάλιστα σε σύντομο χρονικό διάστημα έχοντας βρει ένα τοπικό βέλτιστο. Διαφορετικά βρίσκοντας καλύτερες λύσεις συνεχώς, φθάνει στο ολικό μέγιστο.

1.3.6. Νοημοσύνη των σμηνών

Η Νοημοσύνη των σμηνών (Swarm Intelligence-SI) είναι μια τεχνική τεχνητής νοημοσύνης που βασίζεται γύρω από τη μελέτη της συμπεριφοράς συστημάτων συνόλων πληθυσμών. Η έκφραση "νοημοσύνη σμηνών" εισήχθη από τους Beni & WANG το 1989, για τη μελέτη των κυψελοειδών ρομποτικών συστημάτων. Ο όρος νοημοσύνη των σμηνών αναφέρεται σε ένα κλάδο της τεχνητής ζωής, ο οποίος στηρίζεται στη μελέτη της συλλογικής συμπεριφοράς σε μη κατανεμημένα, αυτό-οργανωμένα σύνολα ατόμων όπως μυρμήγκια, μέλισσες κ.α. που εκτελούν διάφορες

εργασίες ζωτικής σημασίας για το σύνολο του πληθυσμού που ταυτόχρονα σαν ξεχωριστά άτομα δε θα μπορούσαν να πετύχουν. Τα συστήματα αυτά αποτελούνται από έναν πληθυσμό απλών πρακτόρων που αλληλεπιδρούν ατομικά και συλλογικά μεταξύ τους και με το περιβάλλον τους. Γενικά, δεν υπάρχει κάποια συγκεντρωτική δομή ελέγχου του τρόπου της αλληλεπίδρασης, ενώ αυτό γίνεται αντιληπτό από τη συλλογική συμπεριφορά. Παραδείγματα τέτοιων συστημάτων μπορούν να βρεθούν στη φύση είναι οι αποικίες των μυρμηγκιών, τα σμήνη των πουλιών, οι αποικίες βακτηριδίων και τα κοπάδια των ψαριών.

Από τις πιο επιτυχημένες τέτοιες τεχνικές που έχουν αναπτυχθεί μέχρι τώρα είναι η Βελτιστοποίηση με τη μέθοδο των αποικιών μυρμηγκιών (Ant Colony Optimization-ACO) και η Βελτιστοποίηση με τη μέθοδο σμηνών μορίων (Particle swarm optimization-PSO). Οι ACO αλγόριθμοι είναι μεταευρετικοί αλγόριθμοι που δίνουν κατά προσέγγιση λύσεις σε δύσκολα συνδυαστικά προβλήματα βελτιστοποίησης. Οι αλγόριθμοι PSO αποτελούν τεχνικές ελαχιστοποίησης για τα προβλήματα στα οποία μια καλύτερη λύση μπορεί να αντιπροσωπευθεί ως σημείο ή επιφάνεια σε ένα n-διάστατο σύστημα. Μια ακόμα τεχνική που αξίζει να δούμε και ανήκει στις τεχνικές των σμηνών είναι και τεχνική αποικίας τεχνητών μελισσών (Artificial Bee Colony - ABC). Η πρώτη φορά που μελετήθηκε μια τέτοια τεχνική ήταν από τον Karaboga το 2005. Σε αυτήν περίπτωση γίνεται προσομοίωση της διαδικασίας εύρεσης τροφής των μελισσών. Οι αλγόριθμοι που παρουσιάζουν αυτήν την τεχνική έχουν τρεις φάσεις: της μέλισσας εργάτριας, της μέλισσας ανιχνευτή και της μέλισσας επιστάτη.

1.4. Παραδείγματα μεταυερετικών αλγορίθμων

1.4.1. Βελτιστοποίηση με Σμήνος Σωματιδίων

Ο αλγόριθμος Βελτιστοποίηση με Σμήνος Σωματιδίων (Particle Swarm Optimization PSO) είναι ένας στοχαστικός μεταευρετικός αλγόριθμος βασισμένος στην έννοια του πληθυσμού. Προτάθηκε από τους Eberhart και Kennedy (1995) ως μία εναλλακτική λύση στους κυρίαρχους μέχρι τότε Εξελικτικούς Αλγορίθμους (Evolutionary

Algorithms - EA) για την επίλυση προβλημάτων αριθμητικής βελτιστοποίησης. Από την δημιουργία του, ο αλγόριθμος PSO έχει αποκτήσει αυξανόμενο ερευνητικό ενδιαφέρον. Αυτό οφείλεται στην επιβεβαιωμένη αποδοτικότητά του σε πληθώρα δύσκολων προβλημάτων βελτιστοποίησης, καθώς και στην εύκολη υλοποίηση του. Σήμερα, ο αλγόριθμος PSO κατατάσσεται ανάμεσα στους κορυφαίους μεταευρετικούς αλγορίθμους βελτιστοποίησης, μετρώντας ένα μεγάλο πλήθος εφαρμογών σε διάφορα επιστημονικά πεδία, εκτενή βιβλιογραφία, καθώς και δημοσιευμένο ερευνητικό λογισμικό. Ο κύριος μηχανισμός του αλγορίθμου PSO αποτελείται από μία ομάδα συνεργαζόμενων πρακτόρων αναζήτησης, οι οποίοι επαναληπτικά εξετάζουν τον χώρο αναζήτησης. Πιο συγκεκριμένα, θεωρούμε το n-διάστατο πρόβλημα ολικής ελαχιστοποίησης:

$$\min_{x \in X \subset \mathbb{R}^n} C(x).$$

Ένα σμήνος (swarm) είναι ένα σύνολο από σημεία αναζήτησης και ορίζεται ως:

$$S = \{x_1, x_2, \dots, x_N\}, \qquad x_i \in X, \qquad i \in I = \{1, 2, \dots, N\}.$$

Κάθε σημείο αναζήτησης καλείται σωματίδιο (particle) και είναι ένα -διάστατο διάνυσμα που ορίζεται ως:

$$x_i = (x_{i1}, x_{i2}, \dots, x_{in})^T \in X, \qquad i \in I.$$

Κάθε σημείο αναζήτησης μπορεί να κινηθεί μέσα στον χώρο αναζήτησης. Για τον σκοπό αυτό χρησιμοποιείται μια προσαρμοζόμενη μετατόπιση θέσης, η οποία καλείται ταχύτητα (velocity) του σωματιδίου και ορίζεται ως ακολούθως:

$$v_i = (v_{i1}, v_{i2}, \dots, v_{in})^T, \qquad i \in I.$$

Επιπλέον, κάθε σωματίδιο διατηρεί στην μνήμη την καλύτερη θέση (best position) που βρήκε στον χώρο αναζήτησης, δηλαδή αυτή με την μικρότερη συναρτησιακή τιμή:

$$p_i = (p_{i1}, p_{i2}, \dots, p_{in})^T, \qquad i \in I.$$

Αν συμβολίσουμε τον μετρητή επαναλήψεων του αλγορίθμου, τότε ισχύει ότι:

$$p_i(t) = x_i(\tau), \qquad \tau = arg\min_{\kappa \in \{0,1,\dots,t\}} \{C(x_i(\kappa))\}.$$

Οι καλύτερες θέσεις αποτελούν ένα είδος εμπειρίας (experience) για τα σωματίδια. Διαμοιράζοντας την εμπειρία αυτή, επιτυγχάνεται συνεργασία ανάμεσα στα σωματίδια κατευθύνοντας την αναζήτησή τους προς τις πιο ελπιδοφόρες περιοχές του χώρου αναζήτησης. Ο διαμοιρασμός της πληροφορίας βασίζεται στην έννοια της γειτονιάς (neighborhood). Η γειτονιά του -οστού σωματιδίου ορίζεται από το σύνολο δεικτών των σωματιδίων με τα οποία ανταλλάσσεις πληροφορίες:

$$NB_{i,s} = \{j_1, j_2, \dots j_3\} \subseteq I, \qquad i \in NB_{i,s}.$$

Η καλύτερη θέση της γειτονιάς, όπου:

$$g_i = arg\min_{j \in NB_{i,s}} \{C(\boldsymbol{p}_j)\},$$

χρησιμοποιείται μαζί με το p_i για να ανανεωθεί το i-οστό σωματίδιο σε κάθε επανάληψη. Η παράμετρος s ορίζεται από τον χρήστη και καλείται μέγεθος γειτονιάς (neighborhood size) ενώ έχει άμεσο αντίκτυπο στο εύρος ανταλλαγής πληροφορίας ανάμεσα στα σωματίδια. Ως εκ τούτου, μπορεί να επηρεάσει τις ιδιότητες εξερεύνησης (exploration)/εκμετάλλευσης (exploitation) του αλγορίθμου. Στην ειδική περίπτωση όπου s = N, ολόκληρο το σμήνος αποτελεί την γειτονιά κάθε σωματιδίου. Αυτή η περίπτωση ορίζει το λεγόμενο ολικό μοντέλο (global model) της PSO, το οποίο συμβολίζεται και ως gbest. Αντίθετα, αυστηρά μικρότερες γειτονιές ορίζουν το τοπικό μοντέλο (local model), το οποίο συμβολίζεται και ως lbest.

Τα σωματίδια που θα αποτελέσουν κάθε γειτονιά συνήθως καθορίζονται διαμέσου γενικών σχημάτων, στα οποία θεωρείται μια χωρική οργάνωση των δεικτών των σωματιδίων. Αυτά τα σχήματα ονομάζονται τοπολογίες γειτονιών (neighborhood topologies) και μπορεί να έχουν σημαντικό αντίκτυπο στην απόδοση της PSO επειδή

καθορίζουν την ροή της πληροφορίας ανάμεσα στα σωματίδια. Μια ευρέως χρησιμοποιούμενη τοπολογία γειτονιών είναι ο δακτύλιος (ring), στην οποία όλοι οι δείκτες σωματιδίων θεωρούνται διατεταγμένοι σε αύξουσα σειρά, με τους δείκτες να ανακυκλώνονται. Σύμφωνα με αυτή την τοπολογία, κάθε σωματίδιο θεωρεί ως γειτονιά τα σωματίδια με γειτονικούς σε αυτό δείκτες.

Μέχρι σήμερα, έχουν προταθεί στην βιβλιογραφία διάφορες παραλλαγές του αλγορίθμου PSO. Μία από τις πιο δημοφιλής είναι η παραλλαγή με χρήση συντελεστή περιορισμού (constriction coefficient) που προτάθηκε από τους Clerc και Kennedy (2002). Σύμφωνα με αυτή, η κίνηση του σωματιδίου περιγράφεται από τις ακόλουθες σχέσεις:

$$v_{ij}(t+1) = \chi\left[v_{ij}(t) + c_1 R_1\left(p_{ij}(t) - x_{ij}(t)\right) + c_2 R_2\left(p_{g_i j}(t) - x_{ij}(t)\right)\right],$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1),$$

Όπου i=1,2,…,n και j=1,2,…,n, η παράμετρος χ είναι ο συντελεστής περιορισμού, τα c_1 και c_2 και είναι σταθερές γνωστές και ως γνωσιακή και κοινωνική σταθερά αντίστοιχα, και τα R_1 και R_2 είναι τυχαίες μεταβλητές ομοιόμορφα κατανεμημένες στο διάστημα [0,1]. Θα πρέπει να σημειωθεί πως χρησιμοποιείται διαφορετική τιμή των R_1 και R_2 για κάθε i και j της παραπάνω σχέσης σε κάθε επανάληψη. Οι καλύτερες θέσεις των σωματιδίων ανανεώνονται σε κάθε επανάληψη με βάση την παρακάτω σχέση:

$$\boldsymbol{p}_i(t+1) = \begin{cases} \boldsymbol{x}_i(t+1), & \text{αν } C\left(\boldsymbol{x}_i(t+1)\right) < C\left(\boldsymbol{p}_i(t)\right), \\ \boldsymbol{p}_i(t), & \text{διαφορετικά,} \end{cases} \qquad i \in I.$$

Η παραπάνω μορφή της PSO προέκυψε από την ανάλυση ευστάθειας του αλγορίθμου. Από την ίδια ανάλυση προέκυψε και το σύνολο τιμών των παραμέτρων, χ=0.729, c_1=c_2=2.05, το οποίο θεωρείται εν γένει ικανοποιητικό, παρέχοντας στον αλγόριθμο σταθερότητα και ταχύτητα. Εναλλακτικά σύνολα παραμέτρων έχουν δοθεί στην βιβλιογραφία.

1.4.2. Διαφοροεξελικτικός Αλγόριθμος

Ο Διαφοροεξελικτικός Αλγόριθμος (Differential Evolution - DE) είναι ένας μεταευρετικός αλγόριθμος βασισμένος στην έννοια του πληθυσμού και έχει πολλές ομοιότητες με την PSO. Προτάθηκε το 1997 από τους Storn και Price και, ομοίως με την PSO, χρησιμοποιεί μία ομάδα από σημεία αναζήτησης που καλείται πληθυσμός (population) και ορίζεται ως:

$$S = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\},$$

για να εξερευνήσει τον χώρο αναζήτησης. Κάθε σημείο αναζήτησης καλείται άτομο (individual) και ορίζεται ως ένα n-διάστατο διάνυσμα:

$$\boldsymbol{x}_i = (x_{i1}, x_{i2}, \dots, x_{in})^T \in X, \qquad i \in I$$

το οποίο ανήκει στο χώρο αναζήτησης , σε άμεση συμφωνία με τα σωματίδια της PSO. Επίσης, ο πληθυσμός αρχικοποιείται τυχαία, συνήθως με βάση την ομοιόμορφη κατανομή στο χώρο αναζήτησης. Ωστόσο, η κύρια διαδικασία ανανέωσης του DE διαφέρει από της PSO. Συγκεκριμένα, αντί της υπόθεσης πως τα άτομα κινούνται μέσα στον χώρο αναζήτησης, η ανανέωση γίνεται με χρήση τριών τελεστών που δειγματοληπτούν άτομα από τον πληθυσμό, τα συνδυάζουν παράγοντας νέες πιθανές λύσεις και επιλέγοντας τα καλύτερα ανάμεσα στα νέα και τα ήδη υπάρχοντα άτομα. Οι τελεστές αυτοί καλούνται μετάλλαξη (mutation), διασταύρωση (crossover) και επιλογή (selection), και αναλύονται στην συνέχεια.

Ο τελεστής της μετάλλαξης παράγει ένα νέο διάνυσμα, v_i, για κάθε άτομο x_i, i=1,2,…,N, συνδυάζοντας κάποια από τα άτομα του πληθυσμού. Υπάρχει πλήθος τελεστών μετάλλαξης για τον DE που αναφέρονται στη βιβλιογραφία. Οι παρακάτω πέντε τελεστές είναι οι επικρατέστεροι:

$$(DE1) \quad \boldsymbol{v}_i\,(t+1) = \boldsymbol{x}_g\,(t) + F\left(\boldsymbol{x}_{r_1}\,(t) - \boldsymbol{x}_{r_2}\,(t)\right),$$

$$(DE2) \quad \boldsymbol{v}_i\,(t+1) = \boldsymbol{x}_{r_1}\,(t) + F\left(\boldsymbol{x}_{r_2}\,(t) - \boldsymbol{x}_{r_3}\,(t)\right),$$

$$(DE3) \quad \boldsymbol{v}_i\,(t+1) = \boldsymbol{x}_i\,(t) + F\left(\boldsymbol{x}_g\,(t) - \boldsymbol{x}_i\,(t) + \boldsymbol{x}_{r_1}(t) - \boldsymbol{x}_{r_2}(t)\right),$$

$$(DE4) \quad \boldsymbol{v}_i\,(t+1) = \boldsymbol{x}_g(t) + F\left(\boldsymbol{x}_{r_1}(t) - \boldsymbol{x}_{r_2}(t) + \boldsymbol{x}_{r_3}(t) - \boldsymbol{x}_{r_4}(t)\right),$$

$$(DE5) \quad \boldsymbol{v}_i\,(t+1) = \boldsymbol{x}_{r_1}(t) + F\left(\boldsymbol{x}_{r_2}(t) - \boldsymbol{x}_{r_3}(t) + \boldsymbol{x}_{r_4}(t) - \boldsymbol{x}_{r_5}(t)\right),$$

ΚΕΦΑΛΑΙΟ 2

ΕΙΣΑΓΩΓΗ ΠΡΟΒΛΗΜΑΤΩΝ ΣΤΟ MATLAB

Το MATLAB, έχει ονομαστεί από τις λέξεις MATrix LABoratory, και αποτελεί ένα λογισμικό περιβάλλον προγραμματισμού αριθμητικών μεθόδων, προσομοίωσης και γραφικής οπτικοποίησης. Το βασικό στοιχείο/ μεταβλητή του MATLAB είναι ο πίνακας και έτσι όλοι οι φιλόδοξοι μελλοντικοί προγραμματιστές πρέπει να γνωρίζουν τις βασικές μαθηματικές αρχές πινάκων και γραμμικής άλγεβρας. Το εγχειρίδιο αποσκοπεί να αποτελέσει ένα σύντομο βοήθημα αναφορικά με τις κυριότερες συναρτήσεις και δυνατότητες του MATLAB και του SIMULINK. Αν κάποιος θέλει να εμβαθύνει και να εκμεταλλευτεί όλες τις δυνατότητες που προσφέρει το λογισμικό μπορεί να χρησιμοποιήσει το User's Manual ή την εντολή HELP. Επίσης κάποιος μπορεί να χρησιμοποιήσει ένα από τα πολλά FORUMS του διαδικτύου στα οποία μπορεί να βρει ειδικευμένες συναρτήσεις γραμμένες από άλλους χρήστες. Τέτοια FORUMS μπορούν να βρεθούν με ένα απλό search στο Google. Σημαντικό να αναφερθεί ότι το MATLAB έχει αναπτυχθεί για την πλατφόρμα Windows στην οποία το mouse είναι το κύριο μέσο επικοινωνίας. Πολλές από τις εντολές του MATLAB, μπορούν να γίνουν είτε μέσω του menu ή μέσω εντολών στο Command Prompt. Οι μέθοδοι αυτοί θα αναλυθούν και αφήνεται στις προτιμήσεις του χρήστη για το ποια μέθοδο θα χρησιμοποιεί.

2.1. Δηλώσεις και Μεταβλητές

Η συλλογή προγραμμάτων του MATLAB περιέχει το βασικό πρόγραμμα συν μια ποικιλία από βοηθήματα (toolboxes). Τα toolboxes είναι μια συλλογή ειδικών αρχείων, τα Μ-αρχεία, τα οποία επεκτείνουν τις δυνατότητες του βασικού προγράμματος. Το βασικό πρόγραμμα μαζί με το Control System Toolbox μας δίνουν τη δυνατότητα να χρησιμοποιούμε το MATLAB για σχεδιασμό και ανάλυση συστημάτων ελέγχου.

Οι δηλώσεις στο MATLAB γίνονται ως εξής :

>> μεταβλητή =έκφραση

Το (‘’=’’)αναθέτει την έκφραση στη μεταβλητή. Η προτροπή εντολής είναι δύο δεξιά βέλη ‘’>>’’.

Υποθετικά, εισάγουμε ένα πίνακα 2x2 με όνομα μεταβλητής A. Ο πίνακας A εμφανίζεται αυτόματα μετά την εκτέλεση της δήλωσης (μετά το carriage return).

Το MATLAB μπορεί να χρησιμοποιηθεί επίσης σε ‘’ calculator ‘’ mode.

Οι πιο συνηθισμένοι μαθηματικοί τελεστές (+.-, *, /, ^)

Στοιχειώδης μαθηματικές και τριγωνομετρικές συναρτήσεις :(π.χ. sin (x) ημίτονο του X)

Υπάρχουν επίσης ορισμένες προκαθορισμένες μεταβλητές , όπως pi, Inf, Nan, I και j.

2.2. Πίνακες και Διανύσματα

Όλες οι μεταβλητές στο MATLAB είναι arrays. Ένα array είναι μια συλλογή τιμών στην οποία η αναφορά γίνεται με ένα μόνο όνομα μεταβλητής. Κάθε ένα στοιχείο του πίνακα αποθηκεύεται και εξάγεται (υπολογίζεται) καθορίζοντας το δείκτη του στοιχείου με βάση την αρχή του array. Από την έκδοση 5 του MATLAB και μετά μπορούν να δημιουργηθούν arrays με οποιονδήποτε αριθμό δεικτών. Στις εφαρμογές αυτής της εισαγωγής μας στο MATLAB τα arrays θα έχουν έναν ή δύο δείκτες που περιέχουν είτε αριθμητικές, ή χαρακτήρες. Ένας πίνακας (matrix) είναι δισδιάστατο array με αριθμητικές τιμές που υπακούει στους κανόνες της γραμμικής άλγεβρας, όπως αυτοί περιγράφονται σε ένα βιβλίο Γραμμικής Άλγεβρας. Εδώ θα αναφερθούμε κυρίως σε ότι αφορά τον ορισμό και τη χρήση μεταβλητών για πίνακες του MATLAB σε απλές υπολογιστικές πράξεις. Ένα βαθμωτό μέγεθος θεωρείται ως ένας πίνακας με μια γραμμή και μια στήλη. Ένα διάνυσμα (vector) είναι ένας πίνακας είτε με μία

γραμμή ή με μία στήλη. Όταν εισάγεται μια μαθηματική έκφραση ή παράσταση (expression) ο διερμηνέας του MATLAB την διαπερνάει (parse) και την υπολογίζει κάνοντας χρήση των κανόνων της γραμμικής άλγεβρας.

Οι αγκύλες, [ ], χρησιμοποιούνται για να οροθετήσουν διανύσματα και πίνακες κατά την εισαγωγή τους στη γραμμή εντολών.

Τα στοιχεία σε μια γραμμή διαχωρίζονται με κενά ή κόμματα. Τα ερωτηματικά χρησιμοποιούνται για το διαχωρισμό των γραμμών.

Όταν ένας πίνακας εισάγεται απευθείας στη γραμμή εντολών, για το διαχωρισμό των γραμμών μπορεί να χρησιμοποιηθεί το πλήκτρο enter.

Οι διαστάσεις του πίνακα (ή διανύσματος) δεν καθορίζονται ρητά. Το MATLAB κρατάει πληροφορία για τον αριθμό των γραμμών και των στηλών, αυτόματα.

Δημιουργία Πινάκων

Στο MATLAB, μεταβλητές πινάκων μπορούν να δημιουργηθούν με οποινδήποτε από τους παρακάτω τρόπους:

Απευθείας εισαγωγή (με πληκτρολόγηση)

Υπολογισμός παραστάσεων πινάκων

Ενσωματωμένες συναρτήσεις που επιστρέφουν πίνακες

Συναρτήσεις που δημιουργούνται από το χρήστη και που επιστρέφουν πίνακες

Εισάγοντας πίνακες δεδομένα από αρχεία σε δίσκο

Για την εισαγωγή διανυσμάτων είναι σημαντικό να διακρίνουμε τα διανύσματα γραμμές από τα διανύσματα στήλες. Σύμφωνα με τους κανόνες της γραμμικής άλγεβρας, δεν είναι εναλλάξιμα τα διανύσματα στήλες με τα διανύσματα γραμμές και ο διερμηνευτής (interpreter) του MATLAB δεν επιτρέπει μη αποδεκτές παραστάσεις.

Ένα διάνυσμα γραμμή μπορεί να μετατραπεί σε ένα διάνυσμα στήλη, και το αντίθετο, με τον τελεστή αναστροφής (transpose operator). Στον γνωστό από την γραμμική άλγεβρα συμβολισμό, αν v είναι ένα διάνυσμα, τότε το v T είναι το ανάστροφό του. Στο MATLAB ο τελεστής αναστροφής συμβολίζεται με τη μονή απόστροφο ('). Δηλ. ο ανάστροφος του πίνακα A συμβολίζεται με A'. Υπάρχουν πολλές ενσωματωμένες συναρτήσεις για τη δημιουργία διανυσμάτων και πινάκων. Η eye συνάρτηση δέχεται ένα ή δύο ορίσματα και δημιουργεί έναν πίνακα με 1 στην κύρια διαγώνιο. Η μορφή της eye συνάρτησης με ένα όρισμα δημιουργεί έναν (τετραγωνικό) μοναδιαίο πίνακα με διαστάσεις που ορίζονται από την τιμή της εισαχθείσας παραμέτρου. Αν στην συνάρτηση eye εισαχθούν δύο παράμετροι, η πρώτη ορίζει το πλήθος των γραμμών και η δεύτερη το πλήθος των στηλών του πίνακα που θα δημιουργηθεί. Η διαγώνιος με i = j, όπου i είναι ο δείκτης της γραμμής ενώ j είναι ο δείκτης της στήλης, έχει στοιχεία μονάδες. Η συνάρτηση diag είτε δημιουργεί έναν πίνακα με καθορισμένες τις τιμές της διαγωνίου ή εξάγει τα στοιχεία της διαγωνίου. Για τη δημιουργία ενός διαγώνιου πίνακα με την diag συνάρτηση, το εισαγόμενο όρισμα πρέπει να είναι διάνυσμα. Το διάνυσμα μπορεί να δημιουργηθεί ρητά ή να εισαχθεί κατευθείαν ως όρισμα. Για την εξαγωγή των στοιχείων της διαγωνίου ενός πίνακα χρησιμοποιείται η ίδια συνάρτηση με τη διαφορά ότι τώρα το εισαγόμενο όρισμα είναι πίνακας και όχι διάνυσμα. Η εξάρτηση της λειτουργίας της συνάρτησης diag από το εισαγόμενο όρισμα είναι ένα σημαντικό χαρακτηριστικό το οποίο συναντάται σε πολλές συναρτήσεις του MATLAB. Για κάποιον που μαθαίνει τώρα το MATLAB, η δυνατότητα μιας συνάρτησης να εκτελεί διαφορετικές – αν και σχετιζόμενες – εργασίες, μπορεί να προκαλεί σύγχυση. Όταν όμως εξοικειωθεί με την δυνατότητα αυτή θα είναι ένα σημαντικό πλεονέκτημα για τον ίδιο καθώς, θα πρέπει να θυμάται λιγότερες συναρτήσεις. Οι συναρτήσεις ones και zeros είναι παρόμοιες: η ones δημιουργεί έναν πίνακα με μονάδες και η zeros έναν πίνακα με μηδενικά. Και οι δυο αυτές συναρτήσεις δαίχονται δύο ορίσματα. Το πρώτο όρισμα είναι το πλήθος των γραμμών του πίνακα και το δεύτερο είναι το πλήθος των στηλών του. Φυσικά, ο πίνακας δεν είναι απαραίτητο να είναι τετραγωνικός. Βέβαια, οι ones και zeros μπορούν να χρησιμοποιηθούν για τη δημιουργία διανυσμάτων γραμμών ή στηλών. Οι εντολές του MATLAB που ακολουθούν, δημιουργούν διανύσματα που περιέχουν τιμές τριγωνομετρικών συναρτήσεων. Κατόπιν, τα διανύσματα αυτά συνδυάζονται για να σχηματίσουν τις στήλες ενός πίνακα έτσι ώστε οι τιμές να εμφανιστούν σε πινακοειδή μορφή. Η έκφραση [x' s' c' t'] δημιουργεί έναν πίνακα του οποίου οι

στήλες είναι τα ανάστροφα των διανυσμάτων x, s, c και t. Τα στοιχεία των πινάκων αναφέρονται χρησιμοποιώντας συμβολισμό δεικτών με τρόπο παρόμοιο με αυτόν της Fortran. Αν v είναι ένα διάνυσμα γραμμή ή στήλη, το v(k) αναφέρεται στο k-οστό στοιχείο του διανύσματος. Αν A είναι ένας πίνακας, το A(m,n) αναφέρεται στο στοιχείο της m-οστής γραμμής και n-οστής στήλης. Κατά την εξαγωγή τιμών από πίνακα, η χρήση δεικτών πέραν των διαστάσεων του πίνακα έχει σαν αποτέλεσμα σφάλμα. Ο συμβολισμός δεικτών μπορεί να χρησιμοποιηθεί για την ανάθεση τιμών σε στοιχεία πίνακα. Αντίθετα από την προσπάθεια εξαγωγής στοιχείου εκτός του διαστήματος δεικτών ενός πίνακα, μια πράξη ανάθεσης μπορεί, νόμιμα, να αναφέρεται σε δείκτες γραμμών και στηλών που δεν αποτελούν μέρος του πίνακα. Αν μια εντολή ανάθεσης χρησιμοποιεί έναν δείκτη μεγαλύτερο από τις τρέχουσες διαστάσεις του πίνακα το MATLAB απλώς αυξάνει το μέγεθος του πίνακα, προκειμένου να συμπεριλάβει το νέο στοιχείο(α). Οι πίνακες που μεγαλώνουν αυτόματα για να ανταποκριθούν σε ανάγκες είναι ένα ισχυρό χαρακτηριστικό που ελαττώνει τη μονοτονία και την πολυπλοκότητα κάποιων διεργασιών της γραμμικής άλγεβρας. Ωστόσο, όπως πολλά από τα εξαίρετα χαρακτηριστικά στοιχεία του MATLAB, αυτό μπορεί να οδηγεί σε ολέθρια σφάλματα. Οι συναρτήσεις length και size χρησιμοποιούνται για να προσδιορίσουν το πλήθος των στοιχείων σε διανύσματα και πίνακες. Οι συναρτήσεις αυτές είναι ιδιαίτερα χρήσιμες όταν κάποιος δουλεύει με πίνακες μεταβλητού ή αγνώστου μεγέθους, ειδικότερα δε, όταν γράφει βρόχους.

Το MATLAB έχει μια ισχυρή και πολύ συμπαγή σύνταξη, η οποία αναφέρεται ως συμβολισμός άνω - κάτω τελεία. Ο συμβολισμός αυτός μπορεί να χρησιμοποιηθεί είτε στη δημιουργία διανυσμάτων ή, σε συνδυασμό με τον συμβολισμό δεικτών, στην εξαγωγή πεδίων τιμών των στοιχείων πινάκων. Αν και ο συμβολισμός άνω - κάτω τελεία είναι πολύ λακωνικός και κατά κάποιο τρόπο «ξυπναδιάρικος» για να τον κατέχει κάποιος, μπορεί να χρησιμοποιηθεί παρέχοντας πολλά πλεονεκτήματα. Δύο μορφές συμβολισμού άνω - κάτω τελείας χρησιμοποιούνται για τη δημιουργία διανυσμάτων. Εδώ, οι τιμές αρχικήΤιμή και τελικήΤιμή είναι τα άκρα του διαστήματος, η τιμή αύξηση είναι η διαφορά ανάμεσα στα σημεία του διαστήματος και διάνυσμα είναι το αποτέλεσμα. Οι παρενθέσεις είναι απαραίτητες γιατί ο τελεστής αναστροφής έχει υψηλότερη προτεραιότητα από το σύμβολο (:)2 . Ένα σύμβολο (:) μπορεί να χρησιμοποιηθεί σαν μπαλαντέρ για να αναφερθούμε σε μια ολόκληρη γραμμή ή στήλη. Εναλλακτικά, ένα πεδίο τιμών μπορεί να επιλεγεί

χρησιμοποιώντας μια παράσταση με (:) στη θέση ενός απλού δείκτη. Ο συμβολισμός άνω - κάτω τελεία μπορεί επίσης να χρησιμοποιηθεί σε πράξεις ανάθεσης τιμών. Σε ορισμένες περιπτώσεις μπορεί κανείς να χρειαστεί να ανακτήσει την τελευταία τιμή, την προτελευταία κ.λπ. τιμή από έναν πίνακα ή διάνυσμα. Σε αυτό το σημείο, μπορεί να χρησιμοποιηθεί η λέξη-κλειδί end αντί να ορίσουμε μια μεταβλητή να είναι ίση με το μήκος του διανύσματος.

Διαγραφή Στοιχείων από Διανύσματα και Πίνακες:

Το MATLAB δίνει την δυνατότητα διαγραφής μεμονωμένων ή ομαδοποιημένων στοιχείων από διανύσματα και πίνακες αναθέτοντας στα στοιχεία αυτά τον κενό πίνακα,[ ]. Αν x είναι ένα ήδη ορισμένο διάνυσμα, η ανάθεση x = [ ] απαλείφει όλα τα στοιχεία του διανύσματος. Συγκεκριμένα στοιχεία ενός διανύσματος απαλείφονται χρησιμοποιώντας τις κατάλληλες εντολές. Χρησιμοποιώντας τον κατάλληλο συμβολισμό μπορούμε να διαγράψουμε τμήματα στοιχείων ενός διανύσματος. Οι προηγούμενες εντολές μπορούν να περιοριστούν χρησιμοποιώντας τη λέξη-κλειδί end. Οι πράξεις απαλοιφής σε πίνακες πρέπει να αφορούν ολόκληρες γραμμές ή στήλες.

Πράξεις με Πίνακες:

Οι βασικές αριθμητικές πράξεις της πρόσθεσης, της αφαίρεσης και του πολλαπλασιασμού μπορούν να εφαρμοστούν άμεσα σε πίνακες μεταβλητές, υπό τον όρο ότι η συγκεκριμένη πράξη διέπεται από τους κανόνες της Γραμμικής Άλγεβρας. Για παράδειγμα, η πρόσθεση και η αφαίρεση δύο διανυσμάτων γραμμής ίσου μεγέθους είναι επιτρεπτή. Ο τελεστής * εκτελεί τον κατάλληλο πολλαπλασιασμό πίνακα με πίνακα, πίνακα με διάνυσμα, εσωτερικό και εξωτερικό γινόμενο, ανάλογα με το είδος των τελεστέων. Δοθέντων δύο συμβατών πινάκων A και B, η παράσταση A*B του MATLAB υπολογίζει το γινόμενο των A και B όπως ορίζεται από τους κανόνες της Γραμμικής Άλγεβρας. Το εσωτερικό γινόμενο δύο διανυσμάτων είναι αριθμός. Στην Γραμμική Άλγεβρα το εσωτερικό γινόμενο είναι αποτέλεσμα του πολλαπλασιασμού ενός διανύσματος γραμμή και ενός διανύσματος στήλης. Το εσωτερικό γινόμενο μπορεί, επίσης, να υπολογιστεί με την ενσωματωμένη συνάρτηση dot, η οποία δέχεται σαν ορίσματα δύο διανύσματα με τον ίδιο αριθμό

στοιχείων. Σε αντίθεση με την χρήση του τελεστή *, η χρήση της συνάρτησης dot γίνεται ανεξάρτητα από το είδος των διανυσμάτων, αν είναι δηλαδή και τα δύο διανύσματα, διανύσματα γραμμή ή διανύσματα στήλη. Η συνάρτηση dot είναι κατά κάποιο τρόπο λιγότερο αποτελεσματική από τον τελεστή (*) γιατί εκτελεί κάποιους επιπλέον ελέγχους συμβατότητας και επανασχηματισμού. Διανυσματικοποίηση και Τελεστές Πινάκων Όλες οι ενσωματωμένες συναρτήσεις στο MATLAB είναι «διανυσματικοποιημένες», πράγμα που σημαίνει ότι, δοθέντος ενός διανύσματος ως εσωτερικού ορίσματος, η πράξη που σημειώνεται με το όνομα της συγκεκριμένης συνάρτησης εφαρμόζεται σε όλα τα στοιχεία του διανύσματος. Η διανυσματικοποίηση έχει ως αποτέλεσμα οι πράξεις να εκτελούνται με μια συνοπτική σύνταξη. Ωστόσο, περισσότερο σημαντικό είναι το κέρδος από την απόδοση των υπολογισμών, όποτε χρησιμοποιούνται διανυσματικοποιημένες πράξεις αντί για βρόχους (αλληλουχία επαναλαμβανόμενων εντολών) που διαπερνούν όλα τα στοιχεία ενός πίνακα ή διανύσματος. Προς υποστήριξη της διανυσματικοποίησης, το MATLAB ορίζει νέα αριθμητικά σύμβολα που ονομάζονται array operators, που εκτελούν πράξεις στοιχείο με στοιχείο σε ένα ζεύγος πινάκων (ή διανυσμάτων) με ίσο αριθμό γραμμών και στηλών. Το αποτέλεσμα μιας πράξης στοιχείο με στοιχείο ανάμεσα σε δυο πίνακες, είναι ένας άλλος πίνακας της αυτής μορφής. Στη Γραμμική Άλγεβρα δεν υπάρχει άμεσα ισοδύναμο σε κάποιες από τις πράξεις αυτές. Οι πράξεις αυτές (στοιχείο με στοιχείο) θεωρούν τους πίνακες σαν δομές δεδομένων, όχι σαν μαθηματικές οντότητες. Τα σύμβολα των array operators είναι ο συνδυασμός μιας τελείας (.) και ενός συνήθους τελεστή από τους *, / και ^. Ο στοιχείο με στοιχείο πολλαπλασιασμός εκτελείται με τον τελεστή .* και η στοιχείο με στοιχείο διαίρεση εκτελείται με τον τελεστή.

Οι στοιχείο με στοιχείο πράξεις εφαρμόζονται σε πίνακες αλλά και σε διανύσματα. Ο τελεστής στοιχείο με στοιχείο εκθετοποίησης, .^, υψώνει όλα τα στοιχεία ενός πίνακα σε μια δύναμη. Η σύνταξη των array operators απαιτεί την με προσοχή σωστή τοποθέτηση ενός μικρού τυπογραφικού συμβόλου, μιας τελείας (.), σε ένα, μάλλον, σύνθετο τύπο. Αν και το MATLAB θα εντοπίσει τα συντακτικά λάθη, είναι πιθανό, να γίνουν υπολογιστικά σφάλματα με νόμιμες πράξεις. Για παράδειγμα, A.^2 και A^2 είναι και τα δυο νόμιμα, αλλά όχι ισοδύναμα. Στην Γραμμική Άλγεβρα, η πρόσθεση και η αφαίρεση πινάκων ή διανυσμάτων είναι στοιχείο με στοιχείο πράξεις. Για τον λόγο αυτό, δεν υπάρχουν ειδικοί τελεστές για την πρόσθεση και την αφαίρεση.

Μετασχηματισμός της Μορφής Πινάκων:

Μερικές φορές χρειάζεται αναδιοργάνωση των δεδομένων που είναι αποθηκευμένα σε έναν πίνακα χωρίς να μετασχηματισθούν μαθηματικά. Για παράδειγμα, για να κάνουμε έναν υπολογισμό πιο αποτελεσματικά, μπορεί να πρέπει να μετατρέψουμε έναν πίνακα σε διάνυσμα. Αν η οπτική εικόνα του πίνακα είναι ένα ορθογώνιο, τότε η μετατροπή πίνακα σε διάνυσμα περιλαμβάνει αλλαγή του σχήματος του πίνακα από ένα κανονικό ορθογώνιο σχήμα σε ένα στενόμακρο ορθογώνιο σχήμα. Για την εκτέλεση της πράξης αυτής χρησιμοποιείται η συνάρτηση reshape. Σε άλλες περιπτώσεις, είναι πιο κατάλληλο να χρησιμοποιείται ένα τέχνασμα με το συμβολισμό της ανω – κάτω τελείας (:).

Η συνάρτηση reshape έχει τρία εισερχόμενα ορίσματα:

ΕξόδουΠίνακας= reshape(εισόδουΠίνακας,νέεςΓραμμές,νέεςΣτήλες)

όπου εισόδουΠίνακας είναι ο πίνακας που θα επανασχηματιστεί και νέεςΓραμμές, νέεςΣτήλες είναι ο αριθμός γραμμών και στηλών, αντιστοίχως, του εξόδουΠίνακας που δημιουργούνται από τα στοιχεία του εισόδου Πίνακας. Δεν επιτρέπεται μια μερικώς συμπληρωμένη τελευταία στήλη. Οι επόμενες εντολές παρουσιάζουν την reshape συνάρτηση. Τα στοιχεία του πίνακα A είναι επιλεγμένα για να επισημάνουν πώς η reshape συνάρτηση αναδιοργανώνει τον εισόδου Πίνακας κατά στήλες. Επειδή ένα διάνυσμα γραμμή ή στήλη είναι πίνακας, η reshape συνάρτηση μπορεί να χρησιμοποιηθεί για τη δημιουργία πινάκων ορθογώνιου σχήματος, από διανύσματα. Είναι επίσης πιθανό να αλλάξει το σχήμα ενός πίνακα επί τόπου (δηλ., χωρίς την αντιγραφή του σε άλλο πίνακα). Ο τελεστής (:) μπορεί να χρησιμοποιηθεί στον επανασχηματισμό ενός πίνακα σεδιάνυσμα στήλη. Δεν είναι και περιοριστικό το ότι η χρήση του συμβόλου (:) έχει σαν αποτέλεσμα διάνυσμα στήλη. Αν απαιτείται ένα διάνυσμα γραμμή, απλώς, προσθέστε τον τελεστή αναστροφής. Όπως με την reshape, η μέθοδος με την χρήση του συμβόλου (:) μπορεί να εφαρμοστεί άμεσα σε ένα διάνυσμα. Ο επανασχηματισμός με τη χρήση του συμβόλου (:) είναι ιδιαίτερα χρήσιμος όταν το εισερχόμενο όρισμα μιας διαδικασίας πρέπει να είναι ένα διάνυσμα στήλη ή γραμμή. Ο τελεστής αναστροφής απλώς μετατρέπει ένα διάνυσμα γραμμή σε ένα διάνυσμα στήλη και αντιστρόφως. Σε αντίθεση, η πράξη με την (:), x(:), παράγει

πάντα ένα διάνυσμα στήλη. Οι πράξεις επανασχηματισμού επωφελούνται των ενσωματωμένων δυνατοτήτων διανυσματικοποίησης του MATLAB. Ως αποτέλεσμα, οι πράξεις επανασχηματισμού είναι πολύ πιο αποδοτικές από την αλγοριθμικά ισοδύναμη διαδικασία αντιγραφής στοιχείων, ένα κάθε φορά, σε έναν πίνακα του επιθυμητού σχήματος.

2.3. Γραφικές Παραστάσεις και Διαγράμματα

Το MATLAB μπορεί να σχεδιάσει διαγράμματα με αριθμητικά δεδομένα που είναι αποθηκευμένα σε διανύσματα ή πίνακες. Τα δεδομένα μπορούν να προκύψουν από τον υπολογισμό μιας αναλυτικής συνάρτησης ή να διαβαστούν από ένα αρχείο. Βασικές συναρτήσεις σχεδίασης γραμμών χρησιμοποιούνται για τη δημιουργία μιας καμπύλης με εξίσωση y = f(x) ή περισσοτέρων, με εξισώσεις y1 = f(x1), y2 = f(x2) κ.λ.π. σε ένα μόνο διάγραμμα. Διδιάστατα δεδομένα της μορφής z = f(x,y) αναπαριστώνται σε μια γραφική παράσταση επιφάνειας. Τρισδιάστατα αντικείμενα (δηλ. φυσικά αντικείμενα, όπως καρέκλες, τραπέζια, εξαρτήματα και άνθρωποι) μπορεί να παρουσιάζονται σαν φωτοσκιασμένες γραφικές παραστάσεις επιφανειών. Κάθε ένα από τα διαγράμματα αυτά, μπορούν να αποκτήσουν κίνηση (animation) με την εμφάνιση μιας ακολουθίας εικόνων σε ταχύτατη αλληλουχία.

Γραφικές Παραστάσεις Γραμμών:

Οι γραφικές παραστάσεις χρησιμοποιούνται για την παρουσίαση της μεταβολής μιας μεταβλητής – της εξαρτημένης μεταβλητής – σαν συνάρτηση μιας άλλης μεταβλητής – της ανεξάρτητης μεταβλητής. Υποθέτουμε ότι τα διανύσματα x και y περιέχουν δεδομένα, αποτελέσματα υπολογισμών μιας συνάρτησης με τύπο, y = f(x).

Εμφάνιση Σχολίων σε Διαγράμματα:

Οι ετικέτες των αξόνων, οι λεζάντες και άλλα σχόλια εμπλουτίζουν την πληροφορία που δίνεται από ένα διάγραμμα. Τα στοιχεία εμφανίζονται στα διαγράμματα με τη χρήση των συναρτήσεων. Οι συναρτήσεις xlabel, ylabel και title προσθέτουν κείμενο στους άξονες x και y και στο πάνω μέρος του διαγράμματος, αντίστοιχα. Οι συναρτήσεις αυτές παίρνουν σαν όρισμα ένα απλό αλφαριθμητικό κειμένου. Η

τοποθέτηση την ετικετών γίνεται αυτόματα. Η συνάρτηση legend τοποθετεί μια λεζάντα στο διάγραμμα για να δείχνει τη χρήση διαφορετικών συμβόλων για τη σχεδίαση κάθε καμπύλης. Η σύνταξη της legend είναι

legend(label1, label2, label3, …)

legend(label1, label2, label3, …, position)

όπου label1, label2, label3, …, είναι αλφαριθμητικά που συσχετίζονται με τα σύμβολα και τους τύπους γραμμών που χρησιμοποιούνται σε προηγούμενες κλήσεις της συνάρτησης plot (ή κάποιας ισοδύναμης). Η προαιρετική παράμετρος position καθορίζει το σημείο στο διάγραμμα όπου εμφανίζεται η λεζάντα. Εξ ορισμού, το MATLAB προσπαθεί να τοποθετήσει την λεζάντα έτσι ώστε να μην εμπλέκεται στις γραμμές και τα σύμβολα των δεδομένων που αντιπροσωπεύουν. Για παράδειγμα, ας υποθέσουμε ότι έχουμε να σχεδιάσουμε ένα διάγραμμα από μερικά αναλυτικά ή πειραματικά δεδομένα, περιγράφοντας τη θέση x, την ταχύτητα υ και την επιτάχυνση α ενός αντικειμένου, τιμές οι οποίες μεταβάλλονται χρονικά. Οι επόμενες εντολές δημιουργούν το διάγραμμα δεδομένων και τη σχετική λεζάντα.

Εμφάνιση Πολλών Διαγραμμάτων σε ένα Παράθυρο (Υποδιαγράμματα):

Μερικές φορές είναι επιθυμητό να εμφανίσουμε περισσότερα του ενός διαγράμματα σε μία εικόνα παράθυρο. Αυτό γίνεται με τη χρήση της συνάρτησης subplot, η οποία καλείται πάντα με τρία ορίσματα, όπως

subplot(nrows, ncols, thisPlot)

όπου nrows και ncols ορίζουν έναν οπτικό πίνακα διαγραμμάτων που είναι διατεταγμένα στο παράθυρο και η thisPlot καθορίζει τον αριθμό των διαγραμμάτων που επί του παρόντος είναι σχεδιασμένα. Η thisPlot είναι ένας ακέραιος που μετράει τις γραμμές και μετά τις στήλες. Για ένα παράθυρο στο οποίο έχουν ήδη τοποθετηθεί τα διαγράμματα, τα ορίσματα nrows και ncols δεν αλλάζουν. Πριν τη σχεδίαση κάθε διαγράμματος στον πίνακα χρησιμοποιείται η subplot με την κατάλληλη τιμή της thisPlot.

Γραφικές Παραστάσεις Επιφανειών:

Μια αριθμητική συνάρτηση με δύο ανεξάρτητες μεταβλητές, z = f(x,y), ορίζει μια επιφάνεια στον τρισδιάστατο χώρο. Γραφικά, η επιφάνεια αυτή μπορεί να αναπαρασταθεί με πολλούς τρόπους. Οι συναρτήσεις mesh, meshc, surf, surfc και surfl δημιουργούν γραφικές παραστάσεις μιας επιφάνειας με διαφορετικούς τρόπους εμφάνισης. Οι συναρτήσεις mesh και meshc σκιαγραφούν την επιφάνεια σαν πλέγμα καλωδίων. Οι συναρτήσεις surf, surfc και surfl σκιαγραφούν την επιφάνεια σαν μια συλλογή μικρών επιπέδων, χρωματισμέναων με βάση διαφορετικό φωτισμό και σκιές. Οι συναρτήσεις meshc και surfc συνδυάζουν τη φωτοσκίαση της αντίστοιχης επιφάνειας με το περίγραμμα ενός σταθερού επιπέδου z κάτω από την επιφάνεια. Στις πιο συνηθισμένες μορφές τους, οι συναρτήσεις mesh και surf παίρνουν τις ίδιες παραμέτρους. Οι συναρτήσεις surf και mesh χρησιμοποιούνται για τη σχεδίαση διαγραμμάτων δεδομένων από εξωτερικά αρχεία ή συναρτήσεις που υπολογίζονται στο MATLAB. Για τον υπολογισμό αριθμητικών συναρτήσεων, η συνάρτηση meshgrid είναι ο εύκολος τρόπος δημιουργίας δεδομένων (x, y) που χρησιμοποιούνται για τον υπολογισμό της z = f(x,y).

Περιγράμματα:

Τα διαγράμματα αυτά δείχνουν τα «επίπεδα γραμμών» της συνάρτησης z = f(x,y) (π.χ., οι καμπύλες για τις οποίες z = σταθερά). Ένα συνηθισμένο περίγραμμα είναι μια δισδιάστατη προβολή του επιπέδου γραμμών στο επίπεδο (x,y). Ενώ τα διαγράμματα επιφάνειας δίνουν μια δημιουργική και παραστατική εμφάνιση των δεδομένων της συνάρτησης z = f(x,y), τα περιγράμματα περισσότερη πληροφορία. Οι τοπογραφικοί χάρτες είναι ένας συνηθισμένος τύπος περιγράμματος, που περιέχει πολλά δεδομένα, ακριβής πληροφορίας, που χρησιμοποιείται στην πλοήγηση στην ξηρά. Οι γραμμές του περιγράμματος σε έναν τοπογραφικό χάρτη είναι γραμμές με σταθερή ανύψωση πάνω από το επίπεδο της θάλασσας. Ένας έμπειρος στο διάβασμα χαρτών μπορεί να κοιτάξει έναν τοπογραφικό χάρτη και να δει τις ανυψώσεις επιφάνειας που δείχνουν οι γραμμές του περιγράμματος.

Αυτό το κεφάλαιο παρουσίασε τις βασικές λειτουργίες των υπολογισμών με αλληλεπίδραση του MATLAB. Οι περισσότερες συναρτήσεις του MATLAB μαθαίνονται καλύτερα με τον πειραματισμό.

2.4. Πολυώνυμα

Το MATLAB παρέχει μερικές ενσωματωμένες συναρτήσεις που εφαρμόζονται στα πολυώνυμα. Ένα πολυώνυμο στο MATLAB είναι απλά ένα διάνυσμα πραγματικών ή μιγαδικών συντελεστών, και σαν τέτοιο, δεν αποτελεί ξεχωριστό τύπο δεδομένων. Συναρτήσεις για υπολογισμούς πολυωνύμων χειρίζονται αυτά τα διανύσματα των συντελεστών. Οι συνηθισμένοι αριθμητικοί τελεστές (+, -, *) και οι στοιχείο με στοιχείο τελεστές (.*, ./, .^) μπορούν να εφαρμοστούν σε διανύσματα πολυωνυμικών συντελεστών, αλλά τα αποτελέσματα προκύπτουν μέσω των κανόνων της γραμμικής άλγεβρας και των κανόνων προτεραιότητας των τελεστών του MATLAB. Ένα πολυώνυμο βαθμού n αναπαριστάται με το διάνυσμα των συντελεστών όταν το πολυώνυμο είναι γραμμένο με φθίνουσες δυνάμεις του x. Για παράδειγμα το πολυώνυμο x3 -2x+12, καθορίζεται από το διάνυσμα [1 0 -212]. Δοθέντος ενός πολυωνύμου καθορισμένου από τους συντελεστές του, η συνάρτηση polyval υπολογίζει το πολυώνυμο για μια τιμή του x.

2.5. Αρχεία εντολών και Αρχεία συναρτήσεων

Είναι απαραίτητο να γίνει ένας σημασιολογικός διαχωρισμός ανάμεσα στις συναρτήσεις και τις εντολές που εισάγονται στο παράθυρο εντολών. Μια συνάρτηση έχει εισαγόμενα ορίσματα και, συνήθως, έχει και εξαγόμενα ορίσματα. Για παράδειγμα, η παράσταση

```
>> y = sin(pi/6)
```

χρησιμοποιεί τη συνάρτηση sin με εισαγόμενο όρισμα pi/6 και αναθέτει το αποτέλεσμα στη μεταβλητή y. Σε αντίθεση, η εντολή help χρησιμοποιείται με ένα κενό ανάμεσα στο όνομα της εντολής και το όρισμά της, όπως για παράδειγμα

```
>> help sin
```

Γενικά, οι εντολές χρησιμοποιούνται για τη διαχείριση της κατάστασης της τρέχουσας συνεδρίας του MATLAB, ενώ οι συναρτήσεις χρησιμοποιούνται για τη διαχείριση των μεταβλητών του MATLAB. Ο διαχωρισμός ανάμεσα στις εντολές και τις συναρτήσεις είναι κατά κάποιο τρόπο τεχνητός, γιατί όλες οι εντολές μπορούν να χρησιμοποιηθούν σαν συναρτήσεις αν το όρισμα της εντολής αποδοθεί ως εισαγόμενη συμβολοσειρά (string), όπως

```
>> help('sin')
```

Αν και οι εντολές μπορούν να χειριστούν ως συναρτήσεις, το αντίθετο δεν συμβαίνει. Η παράσταση

```
>> sin pi

ans =

 -0.8900 –0.9705
```

έχει ένα μη αναμενόμενο αποτέλεσμα. Βασικά, η χρήση του MATLAB έχει να κάνει με τη χρήση συναρτήσεων. Η διαφορά στη σύνταξη μεταξύ των συναρτήσεων και των εντολών είναι ένα έλασσον μπέρδεμα, γιατί υπάρχει μόνον ένας μικρός αριθμός εντολών και όλες αυτές έχουν να κάνουν με μη αριθμητικές δουλειές.

2.6. Δομές Ελέγχου ροής

Το MatLab περιλαμβάνει κάποιες εντολές οι οποίες μας βοηθάνε να δημιουργήσουμε δικές μας ακολουθίες συναρτήσεων με τις οποίες να επιτύχουμε την επίλυση

προβλημάτων υψηλού δείκτη δυσκολίας. Από αυτές τις εντολές εμείς θα επεκταθούμε στις εξής τρεις γιατί αυτές είναι που θα χρειαστούμε στα συστήματα αυτόματου ελέγχου:

Η εντολή if

Η εντολή for και

Η εντολή while.

Σις εντολές αυτές συνηθίζεται να τις συντάσσουμε μέσα σε αρχεία συναρτήσεων, αυτό όμως δεν σημαίνει ότι δεν μπορούν να χρησιμοποιηθούν και απευθείας στο παράθυρο εντολών. Εμείς θα τις χρησιμοποιούμε μέσα σε αρχεία εντολών ή συναρτήσεων (M-files).

Εντολή if

Η εντολή if κάνει ακριβώς αυτό που λέει η ίδια η ονομασία της «εάν» ισχύει μια συνθήκη τότε εκτέλεσε την εκάστοτε εντολή, η μορφή της εντολής if είναι η εξής:

```
if συνθήκη

 εντολή;

end
```

Η εντολή ελέγχει τη συνθήκη και αν ικανοποιείται τότε εκτελείται η εντολή ενώ αν δεν ικανοποιείται η συνθήκη δεν εκτελείται η εντολή και η ροή του προγράμματος συνεχίζεται με τον τερματισμό της εντολής if. Η εντολή if μπορεί να περιέχει περισσότερες από μια εντολές τις οποίες τις γράφουμε τη μια κάτω από την άλλη.

```
if συνθήκη

 εντολή1;
```

….

εντολή N;

end

Εκτός από την παραπάνω κλασική μορφή, η εντολή if μπορεί να έχει και την παρακάτω μορφή.

if συνθήκη

εντολή 1;

else

εντολή 2;

end

Σε αυτή τη περίπτωση ελέγχουμε τη συνθήκη και αν ικανοποιείται εκτελείται η εντολή 1 αλλιώς αν δεν ικανοποιείται εκτελείται η εντολή που ακολουθεί την else δηλαδή η εντολή 2 και στη συνέχεια τερματίζεται η εντολή if. Επίσης υπάρχει και ο συνδυασμός των δύο παραπάνω μορφών που έχουμε δει μέχρι στιγμής.

if συνθήκη

εντολή 1;

elseif

εντολή 2;

else

εντολή 3;

end

Ακολουθούμε την ίδια λογική που έχουμε αναφέρει στην εντολή if, αυτό που πρέπει να προσέξουμε είναι ότι το elseif γράφεται χωρίς κενό σαν να είναι μια λέξη.

Εντολή for

Η εντολή for επαναλαμβάνει την εκτέλεση μιας εντολής ή μιας σειράς εντολών για ένα αριθμό επαναλήψεων που έχουμε ορίσει με τη χρήση ενός μετρητή. Παρατηρώντας την μορφή της εντολής for θα γίνει πιο αντιληπτή η λειτουργία της.

for μετρητής = αρχική τιμή: βήμα μέτρησης: τελική τιμή

εντολή1; …

εντολήN;

end

Κατανοούμε λοιπόν ότι η εντολή επαναλαμβάνεται για όλες τις τιμές του μετρητή και κάθε φορά ο μετρητής αυξάνεται κατά το βήμα μέτρησης που έχουμε ορίσει μέχρι και την τελική τιμή. Μπορούμε στην εντολή for ναμ παραλείψουμε το βήμα μέτρησης, τότε σε αυτή τη περίπτωση το MatLab αντιλαμβάνεται ως βήμα μέτρησης την μονάδα.

Εντολή while

Η εντολή while θα μπορούσαμε να πούμε ότι αποτελεί ένα συνδυασμό των δύο παραπάνω εντολών. Η εντολή while επαναλαμβάνει μια εντολή ή μια σειρά εντολών για όσο ισχύει κάποια συνθήκη. Η μορφή της εντολή είναι η εξής :

while συνθήκη

εντολή 1;

…

εντολή N;

end

Στις παραπάνω εντολές αναφερθήκαμε σε συνθήκες οι οποίες έχουν άμεση σχέση με την εκτέλεση ή μη κάποιων εντολών. Ένας από τους τρόπους για να εκφράσουμε μια συνθήκη αποτελεί η χρήση σχεσιακών τελεστών, βέβαια δεν αποτελούν το μόνο τρόπο

Στον παρακάτω πίνακα παρουσιάζονται σχεσιακοί τελεστές.

Σχεσιακοί τελεστές

== Έλεγχος ισότητας

~= Έλεγχος ανισότητας

< Μικρότερος

<= Μικρότερος ίσος

> Μεγαλύτερος

>= Μεγαλύτερος ίσος

2.7. Βασικές συναρτήσεις MatLab

Στον παρακάτω πίνακα παρουσιάζονται οι σημαντικότερες συναρτήσεις και εντολές του MatLab τις οποίες πρόκειται να χρησιμοποιήσουμε καθώς και η περιγραφή λειτουργία της καθεμιάς.:

Βασικές συναρτήσεις του MatLab

sin(x) Ημίτονο του x

cos(x) Συνημίτονο του x

tan(x) Εφαπτομένη του x

asin(x) Τόξο ημιτόνου του x

acos(x) Τόξο συνημίτονου του x

atan(x) Τόξο εφαπτομένης του x

exp(x) Εκθετική συνάρτηση ( )

log(x) Συνάρτηση φυσικού λογαρίθμου ( )

log10(x) Συνάρτηση δεκαδικού λογαρίθμου του x

log2(x) Συνάρτηση λογαρίθμου του x με βάση το 2

sqrt(x) Σετραγωνική ρίζα του x

abs(x) Απόλυτη τιμή του x

complex(x,y)

Σύνθεση μιγαδικών δεδομένων από πραγματικό (x) και φαν-

Σταστικό μέρος (y)

conj(x) Συζυγής μιγαδικός

imag(x) Φανταστικό μέρος μιγαδικής ποσότητας

real(x) Πραγματικό μέρος μιγαδικής ποσότητας

ans Μεταβλητή συστήματος γενικής χρήσης

clear

Καθολικό σβήσιμο δεδομένων της οθόνης του τρέχοντος περιβάλλοντος εργασίας

help Εμφάνιση θεμάτων βοήθειας

i Φανταστική μονάδα ( )

j impulse(F) Υπολογισμός της κρουστικής της συνάρτησης F

lsim(f,u,t)

Υπολογισμός της χρονικής απόκρισης ενός συστήματος f με

είσοδο u και χρονικό διάστημα t

max(x) Προσδιορίζει το μεγαλύτερο στοιχείο ενός πίνακα x

min(x) Προσδιορίζει το μικρότερο στοιχείο ενός πίνακα x

pole(x) Υπολογισμός τον πόλων του συστήματος x

residue Υπολογισμός ανάλυσης σε μερικά κλάσματα

rlocus Υπολογισμός του γεωμετρικού τόπου ριζών

step(F) Υπολογισμός της βηματικής απόκρισης ενός συστήματος

va = abs(v) Δημιουργεί το διάνυσμα va του οποίου κάθε στοιχείο ισούται με την απόλυτη

τιμή του αντίστοιχου στοιχείου του ορίσματος v.

sq = sqrt(v) Δημιουργεί το διάνυσμα sq του οποίου κάθε στοιχείο ισούται με την

τετραγωνική ρίζα του αντίστοιχου στοιχείου του ορίσματος v. Αν κάποιο

στοιχείο είναι αρνητικό, το αντίστοιχο στοιχείου του sq θα είναι μιγαδικός.

y = sin(pi/6) Επιστρέφει το ημίτονο του π/6. Όμοια για cos (συνημίτονο), tan (εφαπτομένη),

asin (τόξο ημ.), acos (τόξο συν.), atan (τόξο εφ.).

y = exp(4) Επιστρέφει το e4, όπου e η βάση των νεπέριων λογαρίθμων.

v' Επιστρέφει τον ανάστροφο του πίνακα v.

C1 = eye(4) Δίνει το μοναδιαίο τετραγωνικό πίνακα με διάσταση 4.

C2 = eye(5,3) Δίνει το μοναδιαίο πίνακα με 5 γραμμές και 3 στήλες.

C3 = ones(3) Δίνει τον τετραγωνικό πίνακα με διάσταση 3 και όλα τα στοιχεία μονάδες.

C4 = eye(2,3) Δίνει τον πίνακα με 5 γραμμές και 3 στήλες και όλα τα στοιχεία μονάδες.

C5 = zeros(2) Δίνει τον τετραγωνικό πίνακα με διάσταση 2 και όλα τα στοιχεία μηδέν.

A1 = diag(v) Επιστρέφει το διαγώνιο (τετραγωνικό) πίνακα με στοιχεία διαγωνίου τα

στοιχεία του v, αφού το όρισμα v είναι διάνυσμα.

A2 = diag(v,k) Επιστρέφει τον πίνακα με όλα τα στοιχεία 0, εκτός αυτών που ανήκουν στην k-διαγώνιο και που θα παίρνουν τις τιμές από τα στοιχεία του διανύσματος v.

Για k = 0 τα στοιχεία του v μπαίνουν στην κύρια διαγώνιο, για k = 1 μπαίνουν στη διαγώνιο που βρίσκεται ακριβώς πάνω από την κύρια διαγώνιο, για k = -1 μπαίνουν στη διαγώνιο που βρίσκεται ακριβώς κάτω από την κύρια διαγώνιο κ.ο.κ. Οι διαστάσεις του πίνακα θα είναι οι ελάχιστες δυνατές ώστε να χωρούν το διάνυσμα v.

w = diag(A,k) Παίρνει τα στοιχεία της k-διαγωνίου του τετραγωνικού πίνακα A και σχηματίζει με αυτά ένα διάνυσμα-στήλη. Το k ορίζεται όπως πριν. Η συνάρτηση diag είναι η ίδια με τις δύο προηγούμενες περιπτώσεις, απλώς συμπεριφέρεται διαφορετικά ανάλογα με τη μορφή του πρώτου ορίσματος.

Όλες οι συναρτήσεις (εκτός αν αναφέρεται διαφορετικά), όταν παίρνουν σαν όρισμα πίνακα αντί για αριθμό, επιστρέφουν ένα πίνακα με ίδιες διαστάσεις, του οποίου το κάθε στοιχείο ισούται με την τιμή της συνάρτησης του αντίστοιχου στοιχείου του αρχικού πίνακα. Ισχύει π.χ. για sqrt, abs, sin κτλ.

ΚΕΦΑΛΑΙΟ 3

ΠΩΣ ΛΕΙΤΟΥΡΓΟΥΝ ΟΙ ΦΩΤΕΙΝΟΙ ΣΗΜΑΤΟΔΟΤΕΣ

3.1. Γενικά για τους φωτεινούς σηματοδότες και πως λειτουργούν

Το πρώτο φανάρι τοποθετήθηκε έξω από τη Βρετανική Βουλή στο Λονδίνο τον Δεκέμβρη του 1868. Το τοποθέτησε εκεί ο μηχανικός τρένων J. P. Knight. Ήταν ένα ανακατασκευασμένο σιδηροδρομικό φανάρι, με σηματοφόρους βραχίονες και με κόκκινες και πράσινες λάμπες. Οι λυχνίες αερίου ελέγχονταν από έναν μοχλό στη βάση τους, ώστε να φαίνεται προς την κυκλοφορία το κατάλληλο φως. Το συγκεκριμένο σύστημα καταστράφηκε από έκρηξη το 1869 τραυματίζοντας τον αστυνομικό που το χειριζόταν εκείνη τη στιγμή.

Συνεχίζοντας με την ιστορία των φαναριών, το ηλεκτρικό αυτόματο - θα λέγαμε - φανάρι κατασκευάστηκε στην Αμερική το 1912. Εφευρέτης του θεωρείται ο αστυνομικός Lester Wire από το Οχάιο. Το 1914 η Αμερικάνικη Εταιρεία Φωτεινών Σηματοδοτών τοποθέτησε ένα σύστημα με δύο χρώματα, κόκκινο και πράσινο, και έναν βομβητή (buzzer) για να προειδοποιεί για τις αλλαγές. Εμπνευστής του ήταν ο James Hoge.

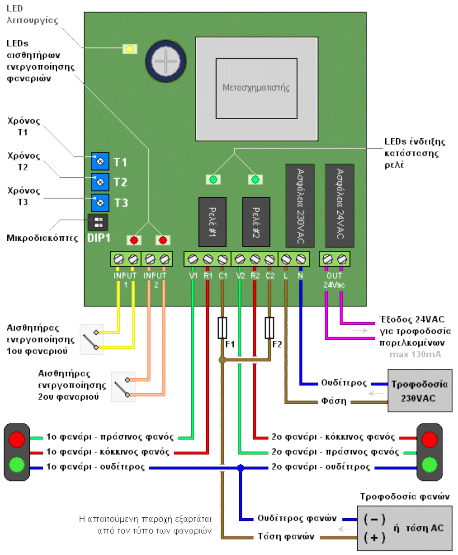
Το συγκεκριμένο σύστημα επέτρεπε στην Αστυνομία και την Πυροσβεστική να ελέγχουν τα φανάρια σε περίπτωση ανάγκης. Το πρώτο τρίχρωμο φανάρι τεσσάρων κατευθύνσεων κατασκευάστηκε από τον αστυνομικό William Potts στο Ντιτρόιτ το 1920. Επειδή ήταν υπάλληλος της κυβέρνησης (αστυνομικός) δεν μπορούσε να πάρει άδεια της πατέντας για την εφεύρεση του. Έτσι, το 1922 ο T.E. Hayes κατοχύρωσε το "Συνδυασμένο οδηγό κυκλοφορίας και ρυθμιστικού σήματος" ("Combination traffic guide and traffic regulating signal").

Το πρώτο διασυνδεδεμένο σύστημα κυκλοφορίας εγκαταστάθηκε στο Σολτ Λέικ Σίτυ το 1917 σε δρόμο με έξι διασταυρώσεις, και ελεγχόταν από χειροκίνητους διακόπτες. Ο αυτόματος χειρισμός του συστήματος μπήκε τον Μάρτιο του 1922 στο Χιούστον του Τέξας. Το 1923 ο Garrett Morgan πατεντάρισε τη δικιά του έκδοση. Ήταν ένας στύλος σε σχήμα Τ με τρεις θέσεις: σταμάτημα, ξεκίνημα και σταμάτημα προς όλες τις κατευθύνσεις. Η τρίτη κατάσταση έδινε στους οδηγούς τη δυνατότητα να σταματήσουν μέχρι να ξεκινήσει η κυκλοφορία του αντίθετου ρεύματος, και επίσης, για την ασφάλεια των πεζών. Το μεγάλο του πλεονέκτημα ήταν η δυνατότητα χειρισμού του από απόσταση μέσω μηχανικής σύζευξης.

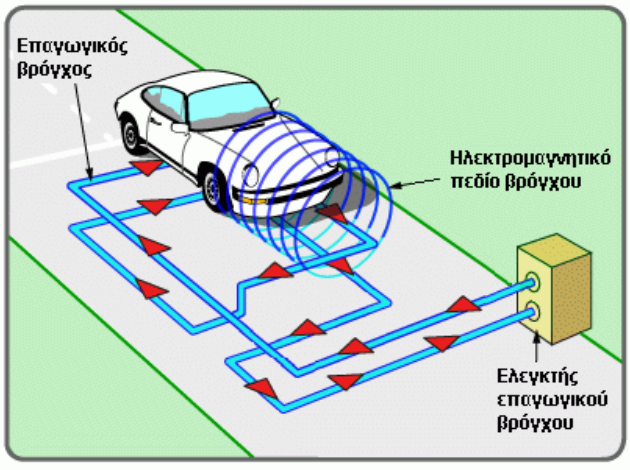
Η πρώτη πόλη που συνέδεσε με υπολογιστές το σύστημα φαναριών των δρόμων της ήταν το Τορόντο το 1963. Τα χρώματα των φαναριών που αναπαριστούν σταμάτημα και ξεκίνημα ενδέχεται να προήλθαν από αυτά που ταυτοποιούσαν το λιμάνι (κόκκινο) και το στρίψιμο προς τα δεξιά (πράσινο) στη ναυσιπλοΐα.

Σύμφωνα με τους κανόνες ναυτικής κυκλοφορίας, το πλοίο από τα αριστερά έπρεπε να σταματήσει για αυτό που έρχεται από τα δεξιά - πράγμα που ισχύει πλέον και στην χερσαία κυκλοφορία. Σε περίπτωση μη ύπαρξης σήμανσης, προτεραιότητα έχει ο κινούμενος από τα δεξιά πάντα, όπως μαθαίνουν οι δάσκαλοι οδήγησης. Τέλος, αξίζει

να αναφερθούμε και στους φωτεινούς σηματοδότες με χρονόμετρο. Το σύστημα εισάχθηκε τη δεκαετία του ’90. Η αντίστροφη μέτρηση βοηθάει τους οδηγούς και τους πεζούς να ξέρουν πόσο χρόνο έχουν μέχρι να αλλάξει το φανάρι, ώστε να αποφασίσουν αν θα περάσουν τη διασταύρωση με ασφάλεια ή όχι.



Τα φανάρια στην Ελλάδα λειτουργούν με προρυθμισμένο χρονοδιακόπτη. Αυτο έχει το μειονέκτημα ότι το φανάρι δεν "αισθάνεται" την κίνηση του δρόμου αλλά λειτουργεί προκαθορισμένα. Σαφώς και υπάρχουν ωράρια (πρωινά, απογευματινά κλπ.) με διαφορετικούς χρόνους διάρκειας του κάθε φαναριού που έχουν προκύψει από στατιστική μελέτη της κάθε διασταύρωσης, αλλά αυτο δεν είναι αρκετό.



Το καλύτερο είναι τα φανάρια να λειτουργούν με αισθητήρα. Ο αισθητήρας βέβαια δεν είναι αισθητήρας βάρους, αλλά αισθητήρας επαγωγής ρεύματος.

Η άσφαλτος έχει σκαφτεί και έχουν βάλει ένα σύρμα σε σχήμα ορθογώνιο/ τετράγωνο που λειτουργεί ως ένα μεγάλο πηνίο. Όταν περνά από πάνω η μπαταρία του οχήματος, το ηλεκτρικό πεδίο της προκαλεί μεταβαλλόμενο μαγνητικό πεδίο που επάγει ρεύμα στο πηνίο και δίνει σήμα στο φανάρι.

Το φανάρι με αισθητήρα έχει το προφανές πλεονέκτημα ότι "αισθάνεται" την κίνηση και προσαρμόζεται ανάλογα, σε αντίθεση με το φανάρι προκαθορισμένου χρόνου. Έτσι, πχ, σε μια άδεια διασταύρωση, αμέσως μόλις φτάνεις, ανάβει το πράσινο και περνάς. Αντίθετα, στο φανάρι με τον χρονοδιακόπτη, κάθεσαι και περιμείνεις άσκοπα να περάσει ο προκαθορισμένος χρόνος, ακόμα κι αν δεν υπάρχει κανένα άλλο όχημα στην διασταύρωση.

Στην Αθήνα δεν υπάρχει φανάρι με αισθητήρα. Οι αισθητήρες έχουν τοποθετηθεί αλλά όχι για να ελέγχουν τη λειτουργία του φαναριού, αντίθετα ελέγχουν την πυκνότητα της ροής και προβάλλουν του αποτέλεσμα στις ηλεκτρονικές πινακίδες ενημέρωσης.

Στην Αμερική όλα τα φανάρια έχουν αισθητήρα, ακόμα και στην τελευταία γειτονία της τελευταίας κωμόπολης. Μόλις φθάνει το αυτοκίνητο στην διασταύρωση, ανάβει το πράσινο και φεύγει.

Οι αισθητήρες χρησιμοποιούνται και για να ελέγχουν τη ροή αυτοκίνητων που μπαίνουν στους αυτοκινητοδρόμους. Σε κάθε είσοδο αυτοκινητοδρόμου υπάρχει φανάρι. Γενικά είναι πάντα πράσινο και μπαίνεις άνετα. Όταν όμως έχει πυκνή ροή ο αυτοκινητόδρομος ανάβει πράσινο/κόκκινο εκ περιτροπής και έτσι μειώνει τον ρυθμό εισόδου νέων αυτοκινήτων στον ήδη κορεσμένο αυτοκινητόδρομο. Επίσης χρησιμοποιούνται και για την εξαγωγή στατιστικών/ενημερώσεων σε ηλεκτρονικές πινακίδες, όπως και στην Αθήνα.

3.2. Ηλεκτρονικός πίνακας ελέγχου για φωτεινούς σηματοδότες

Ο συγκεκριμένος πίνακας συνδυάζεται συνήθως με φωτεινούς σηματοδότες 2 πεδίων όπως οι Stagnoli ASF2RV ή 3 πεδίων όπως οι Stagnoli SF3GRV.

Χρησιμοποιείται για έλεγχο 2 φωτεινών σηματοδοτών (φαναριών ρύθμισης κυκλοφορίας) με 2 φωτεινά πεδία (κόκκινο & πράσινο) ή με 3 φωτεινά πεδία (κόκκινο, πορτοκαλί & πράσινο) ο καθένας.

MADE IN ITALY

Τα τεχνικά χαρακτηριστικά παρουσιάζονται στον παρακάτω πίνακα:

Τροφοδοσία από δίκτυο	**230VAC, 50 Hz**
Τροφοδοσία προς παρελκόμενα	24VAC (0.4 A max)
Πρόκειται για έξοδο τροφοδοσίας από την πλακέτα για ραντάρ, ελεγκτές επαγωγικών βρόχων και λοιπές συσκευές (αισθητήρες) ενεργοποίησης των φωτεινών σηματοδοτών.	
Ασφάλειες	2

Ασφάλεια τροφοδοσίας από δίκτυο	500mA, ταχείας τήξεως
Ασφάλεια τροφοδοσίας προς σηματοδότες	2A, ταχείας τήξεως
Έξοδοι	Μέσω ρελέ κατάλληλων για μέγιστο ρεύμα 5A στα 230VAC
Τα ρελέ τροφοδοτούν τα φανάρια με τέτοιο τρόπο ώστε να ανάβουν το κόκκινο, το πορτοκαλί ή το πράσινο φωτεινό πεδίο.	
Είσοδοι αισθητήρων	3 είσοδοι είτε τύπου Κανονικά Ανοιχτού (NO) είτε τύπου Κανονικά Κλειστού (NC) διακόπτη (ο τύπος κάθε επαφής είναι προγραμματιζόμενος).
Οι 2 από τις επαφές χρησιμεύουν για να δέχεται ο πίνακας σήματα από κατάλληλους αισθητήρες (πχ ραντάρ ή επαγωγικούς βρόχους). Η ενεργοποίηση ενός αισθητήρα ενημερώνει τον πίνακα για την παρουσία οχήματος η οποία πρέπει να ενεργοποιήσει (ή να διατηρήσει ενεργοποιημένο) το πράσινο φωτεινό πεδίο στο αντίστοιχο φανάρι.	
Η 3η επαφή χρησιμεύει για να δέχεται ο πίνακας εντολή έκτακτης ανάγκης.	
Μέθοδος προγραμματισμού	Με 3 πλήκτρα (+, - και E) και 7-segment οθόνη 4 χαρακτήρων
Ενδεικτικά LEDs	2 συνολικά
Κάθε LED δείχνει σε πραγματικό χρόνο το χρώμα του αντίστοιχου σηματοδότη (κόκκινο, πορτοκαλί ή πράσινο).	

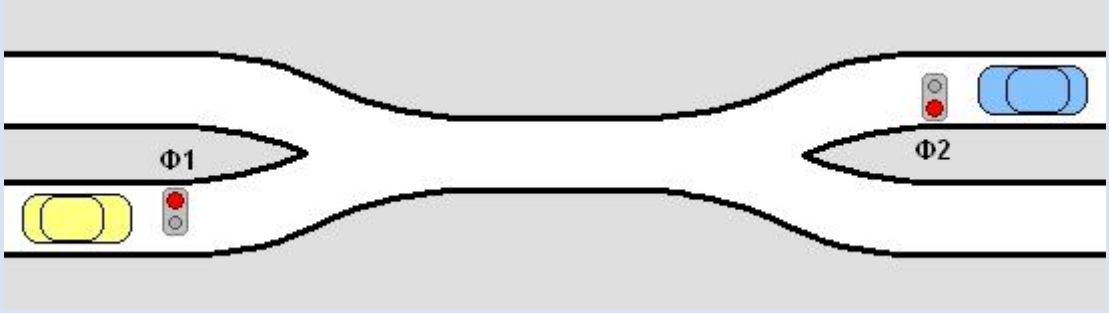
3.2.1. Ενδεικτικές περιπτώσεις χρήσης

Ο γενικός τρόπος λειτουργίας των φωτεινών σηματοδοτών που ελέγχει ο πίνακας είναι όπως φαίνεται στην παρακάτω τυπική περίπτωση χρήσης τους. Πρόκειται για το πρόβλημα κυκλοφορίας που εμφανίζεται όταν έχουμε 2 αντίθετες κατευθύνσεις κυκλοφορίες οι οποίες αναγκάζονται σε κάποιο σημείο να χρησιμοποιήσουν, εναλλάξ, μία κοινή λωρίδα.

Για απλοποίηση, στα παρακάτω σχήματα χρησιμοποιούμε σηματοδότες χωρίς πορτοκαλί πεδίο (ο πίνακας πάντως υποστηρίζει σηματοδότες με ή χωρίς πορτοκαλί πεδίο).

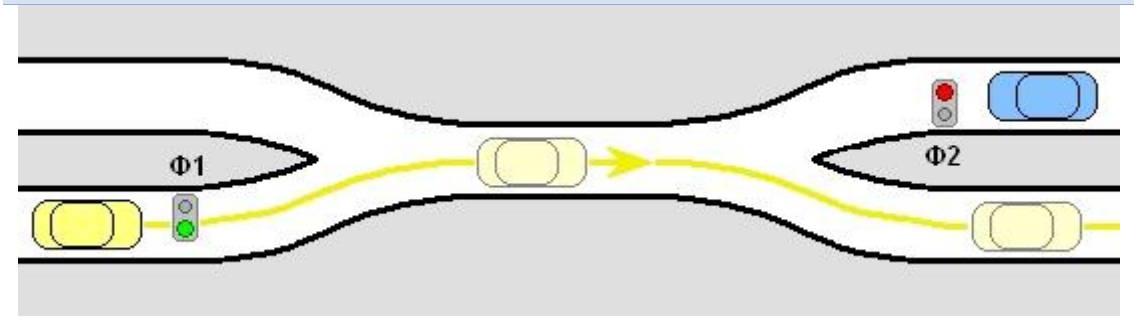
Το ζητούμενο είναι να διέρχονται, εκ' περιτροπής, για ένα διάστημα τα οχήματα της μίας κατεύθυνσης και για ένα διάστημα τα οχήματα της άλλης. Προφανώς, ποτέ δεν πρέπει να επιτρέπεται ταυτόχρονα και στις δύο κατευθύνσεις η διέλευση, ενώ, για να αποφύγουμε συγκρούσεις, πρέπει ανάμεσα στις αλλαγές κατεύθυνσης να μεσολαβεί αρκετός χρόνος όπου και οι δύο φωτεινοί σηματοδότες θα είναι κόκκινοι, ώστε να προλάβει να ελευθερωθεί η κοινή λωρίδα από τυχόν διερχόμενα οχήματα.

Φανάρι 1 & Φανάρι 2: κόκκινο



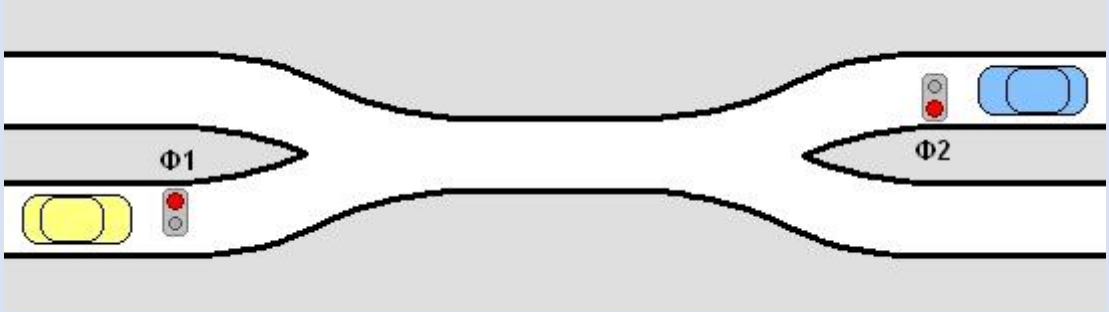
Αυτή είναι η ουδέτερη κατάσταση λειτουργίας (εφόσον ισχύει η εργοστασιακή ρύθμιση d6=1). Και οι δύο φωτεινοί σηματοδότες είναι κόκκινοι.

Φανάρι 1: πράσινο - Φανάρι 2: κόκκινο



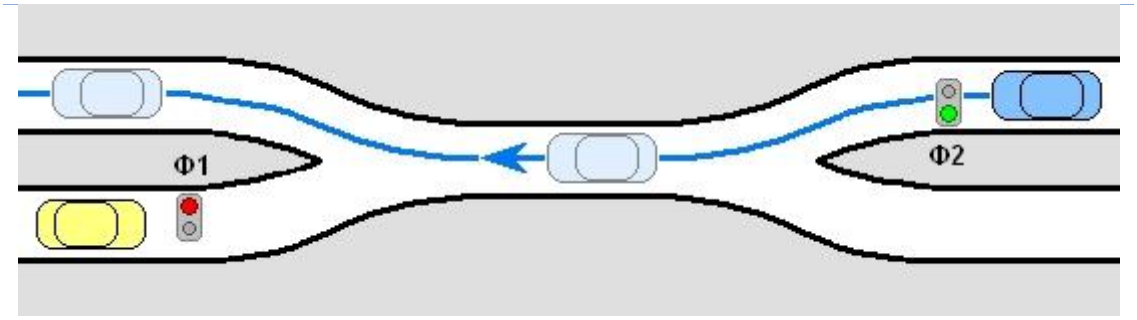
Το Φανάρι 1 γίνεται πράσινο ενώ ταυτόχρονα το Φανάρι 2 παραμένει κόκκινο.

Φανάρι 1 & Φανάρι 2: κόκκινο



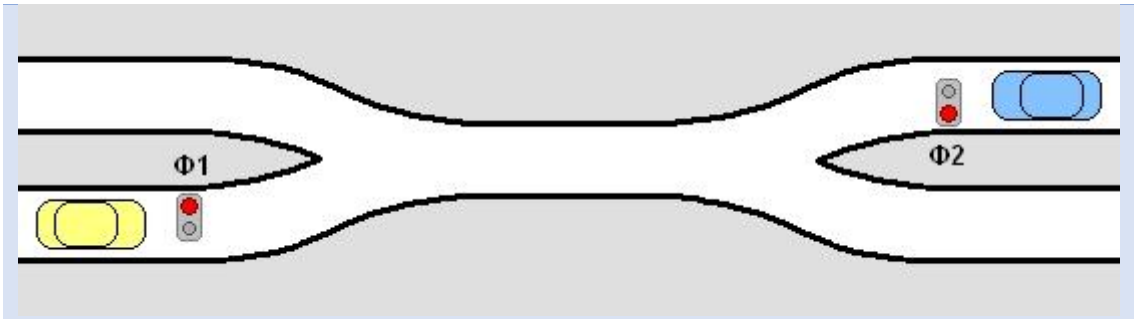
Πριν αλλάξουν χρώμα οι δύο φωτεινοί σηματοδότες, παραμένουν για ένα διάστημα, ταυτόχρονα, κόκκινοι. Με τον τρόπο αυτό, δίνεται η δυνατότητα να ελευθερωθεί ο ενδιάμεσος χώρος από οχήματα ώστε, όταν το Φανάρι 2 γίνει πράσινο, να μην έχουμε συγκρούσεις.

Φανάρι 1: κόκκινο - Φανάρι 2: πράσινο



Το Φανάρι 2 γίνεται πράσινο ενώ ταυτόχρονα το Φανάρι 1 παραμένει κόκκινο.

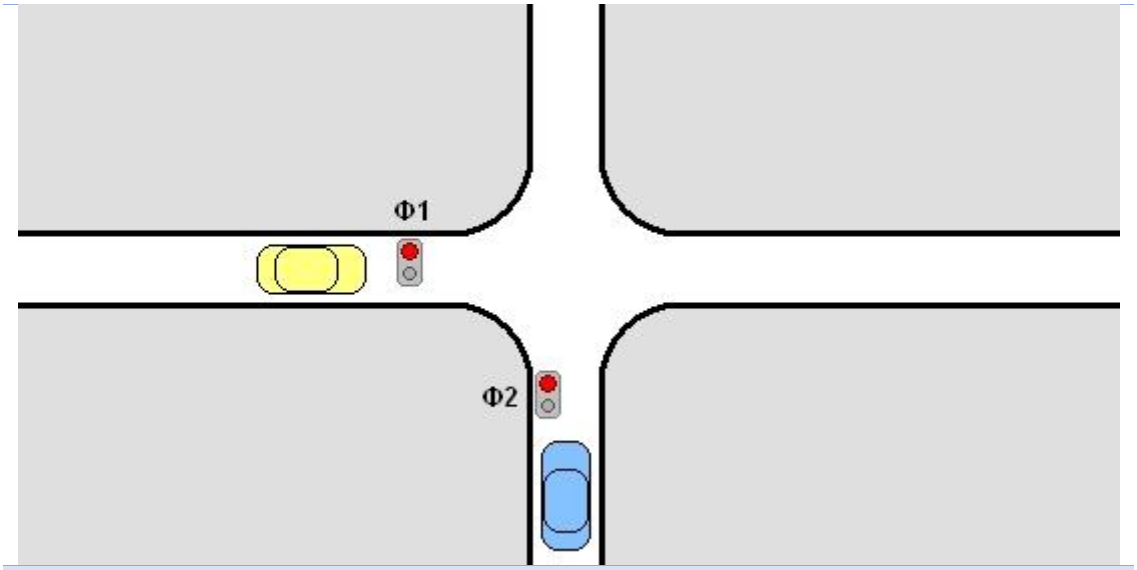
Φανάρι 1 & Φανάρι 2: κόκκινο



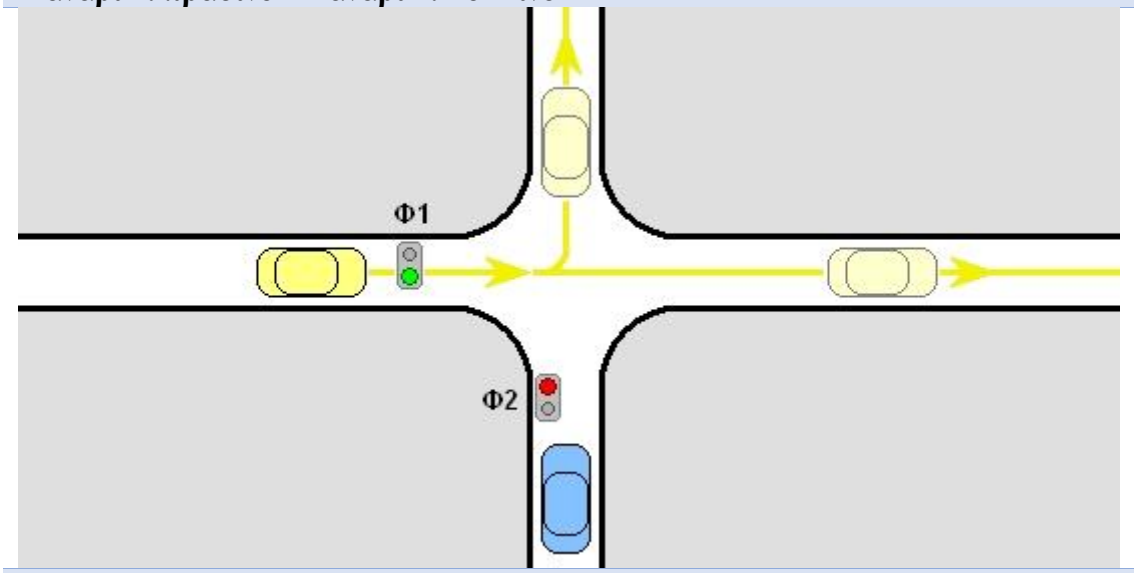
Πριν αλλάξουν χρώμα οι δύο φωτεινοί σηματοδότες, παραμένουν για ένα διάστημα, ταυτόχρονα, κόκκινοι. Με τον τρόπο αυτό, δίνεται η δυνατότητα να ελευθερωθεί ο ενδιάμεσος χώρος από οχήματα ώστε, όταν το Φανάρι 1 γίνει πράσινο, να μην έχουμε συγκρούσεις. Από την κατάσταση αυτή, ο κύκλος λειτουργίας επαναλαμβάνεται με τον παραπάνω τρόπο.

Η εναλλάξ χρήση μίας μόνο λωρίδας, όπως στα παραπάνω σχήματα, δεν είναι η μόνη περίπτωση χρήσης. Παρόμοια εφαρμογή είναι και οι απλές διασταυρώσεις (δύο κατευθύνσεις που διασταυρώνονται), όπως στο επόμενο σχήμα:

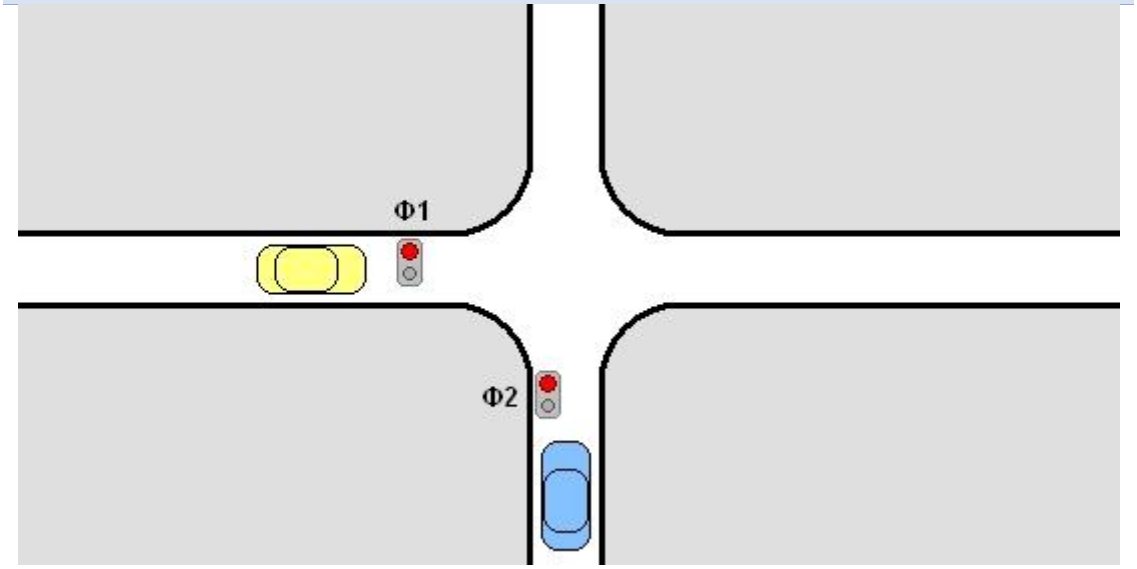
Φανάρι 1 & Φανάρι 2: κόκκινο



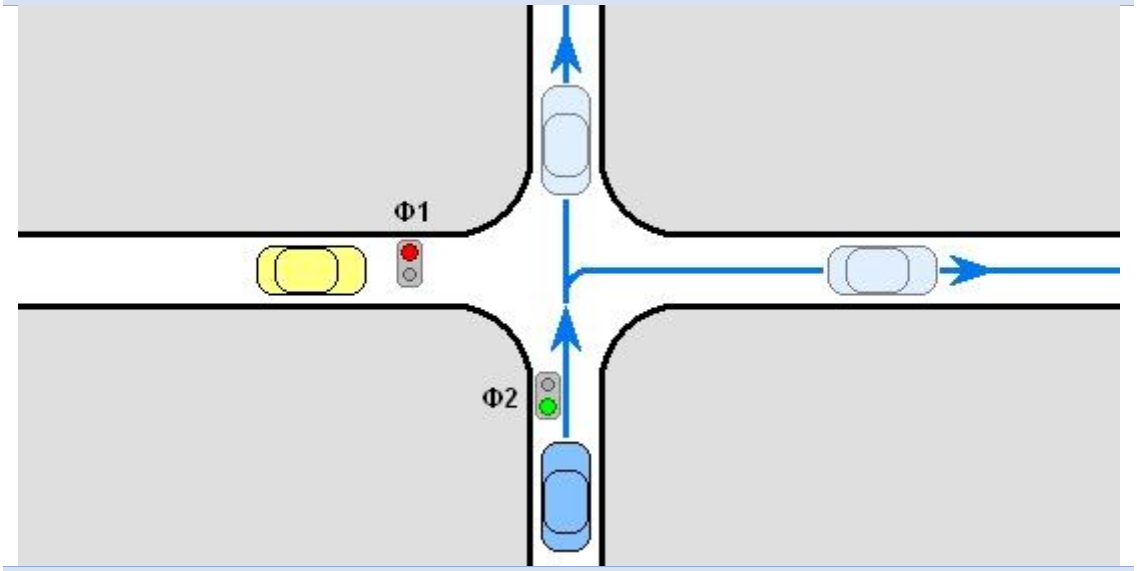
Φανάρι 1: πράσινο - Φανάρι 2: κόκκινο



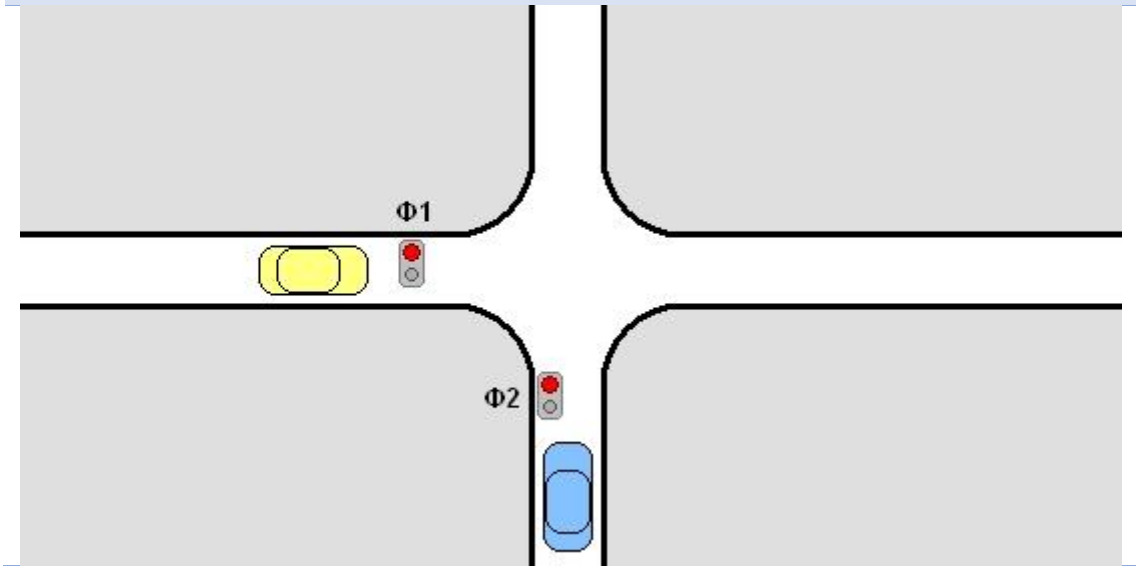
Φανάρι 1 & Φανάρι 2: κόκκινο



Φανάρι 1: κόκκινο - Φανάρι 2: πράσινο



Φανάρι 1 & Φανάρι 2: κόκκινο



3.2.2. Συνδεσμολογία

Ορισμένες βασικές παρατηρήσεις για την συνδεσμολογία:

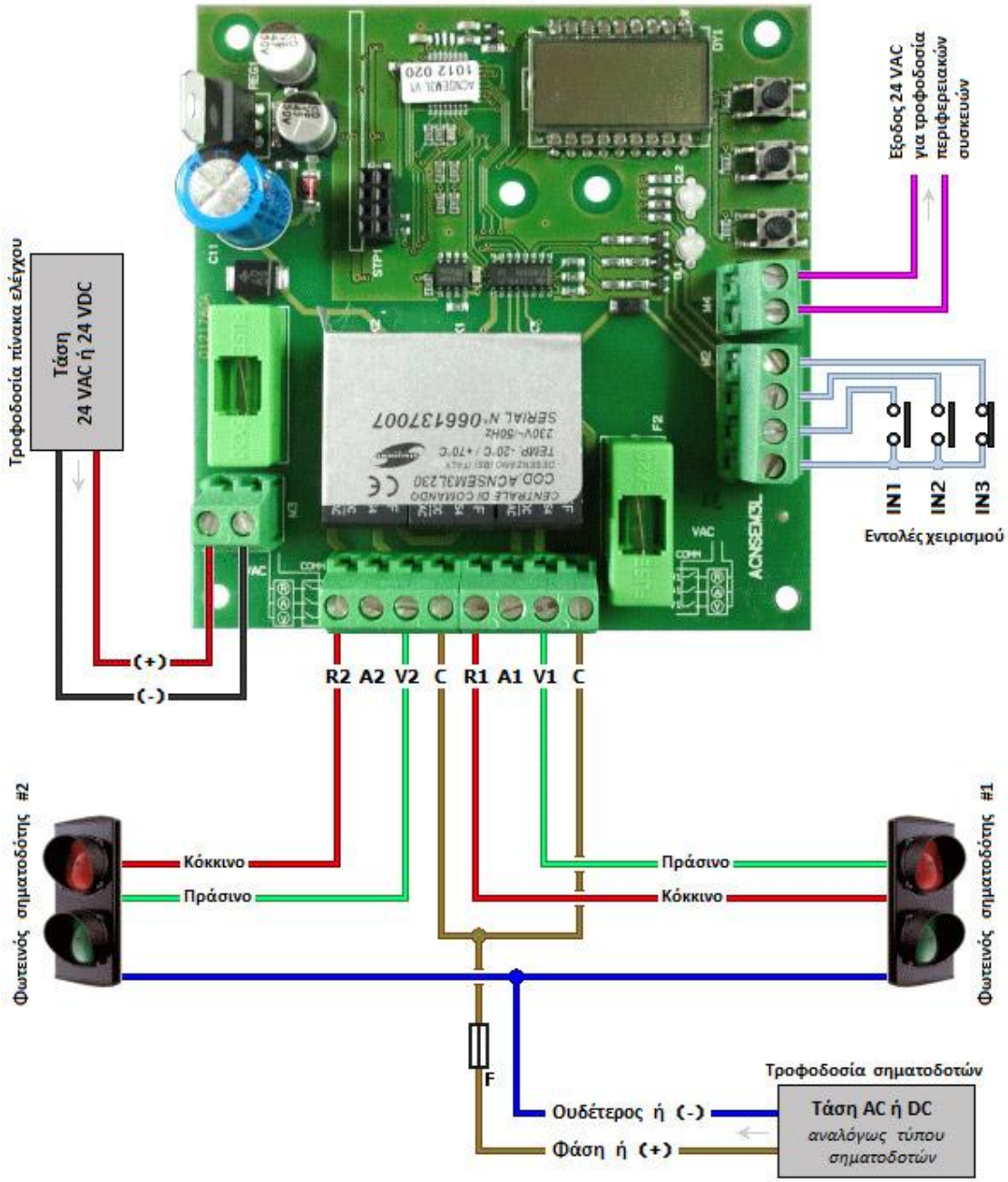
- Η σύνδεση αισθητήρων πρέπει να γίνεται μόνο εάν επιλεγεί η λογική λειτουργίας με σήματα από τους αισθητήρες.
- Εάν επιλεγεί η λογική λειτουργίας με σήματα από τους αισθητήρες πρέπει οπωσδήποτε να εγκαταστήσουμε 2 αισθητήρες (από έναν σε κάθε λωρίδα κυκλοφορίας, εγκατεστημένο πριν από το αντίστοιχο φανάρι). Τέτοιοι αισθητήρες περιλαμβάνουν τα φωτοκύτταρα, τους αισθητήρες ραντάρ και τους ελεγκτές επαγωγικών βρόχων.
- Υπάρχει ξεκάθαρη προτίμηση στους ελεγκτές επαγωγικών βρόχων και μάλιστα στους ελεγκτές διπλού βρόχου με ικανότητα ανίχνευσης κατεύθυνσης οχήματος ή στους αισθητήρες radar doppler διότι προσφέρουν την μέγιστη λειτουργικότητα και αξιοπιστία.
- Ο πίνακας ανταποκρίνεται σε παλμούς από τους αισθητήρες και όχι σε διατηρούμενα σήματα από αυτούς. Εάν για οποιοδήποτε λόγο (πχ βλάβη, παρεμβολή ή ακινητοποιημένο όχημα) κάποιος αισθητήρας δίνει διατηρούμενο σήμα, αυτό θα αγνοηθεί.
- Εάν επιλεγεί η λογική λειτουργίας με κατανομή χρόνου, δεν κάνουμε καμία σύνδεση στις αντίστοιχες επαφές.
- Ο πίνακας δεν τροφοδοτεί άμεσα τους φωτεινούς σηματοδότες. Ο πίνακας ελέγχει ρελέ, στα οποία συνδέεται σαν είσοδος η τάση που τροφοδοτεί τους φωτεινούς σηματοδότες, και δίνουν σαν έξοδο την ίδια τάση προς το κόκκινο, το πορτοκαλί ή το πράσινο φωτεινό πεδίο του αντίστοιχου φωτεινού σηματοδότη. Επίσης, η τάση εισόδου πρέπει να συμβαδίζει με τον τύπο των φωτεινών σηματοδοτών (δηλαδή το είδος του λαμπτήρα κάθε φανού). Υπάρχουν φωτεινοί σηματοδότες 230VAC, 24VAC ή 24VDC κλπ. Όσον αφορά τον πίνακα, οποιαδήποτε τάση τροφοδοσίας των φωτεινών σηματοδοτών είναι αποδεκτή, αρκεί να είναι μέσα στα επιτρεπτά για τα ρελέ όρια. Επειδή ακριβώς ο πίνακας δεν παρέχει άμεσα την τάση προς τους φωτεινούς σηματοδότες, δεν διαθέτει και σχετικές ενσωματωμένες ασφάλειες: η ασφάλιση της τάσης τροφοδοσίας των φωτεινών σηματοδοτών πρέπει να

γίνεται πριν από την σύνδεσή της στον πίνακα. Η ασφάλεια αυτή συμβολίζεται στο σχήμα ως F. Σημειώνουμε ότι επειδή οι 2 επαφές του πίνακα που είναι σημειωμένες ως C είναι μεταξύ τους γεφυρωμένες (εσωτερικά στο κύκλωμα του πίνακα), δεν έχει νόημα το να ασφαλίσουμε ξεχωριστά την τροφοδοσία κάθε σηματοδότη.

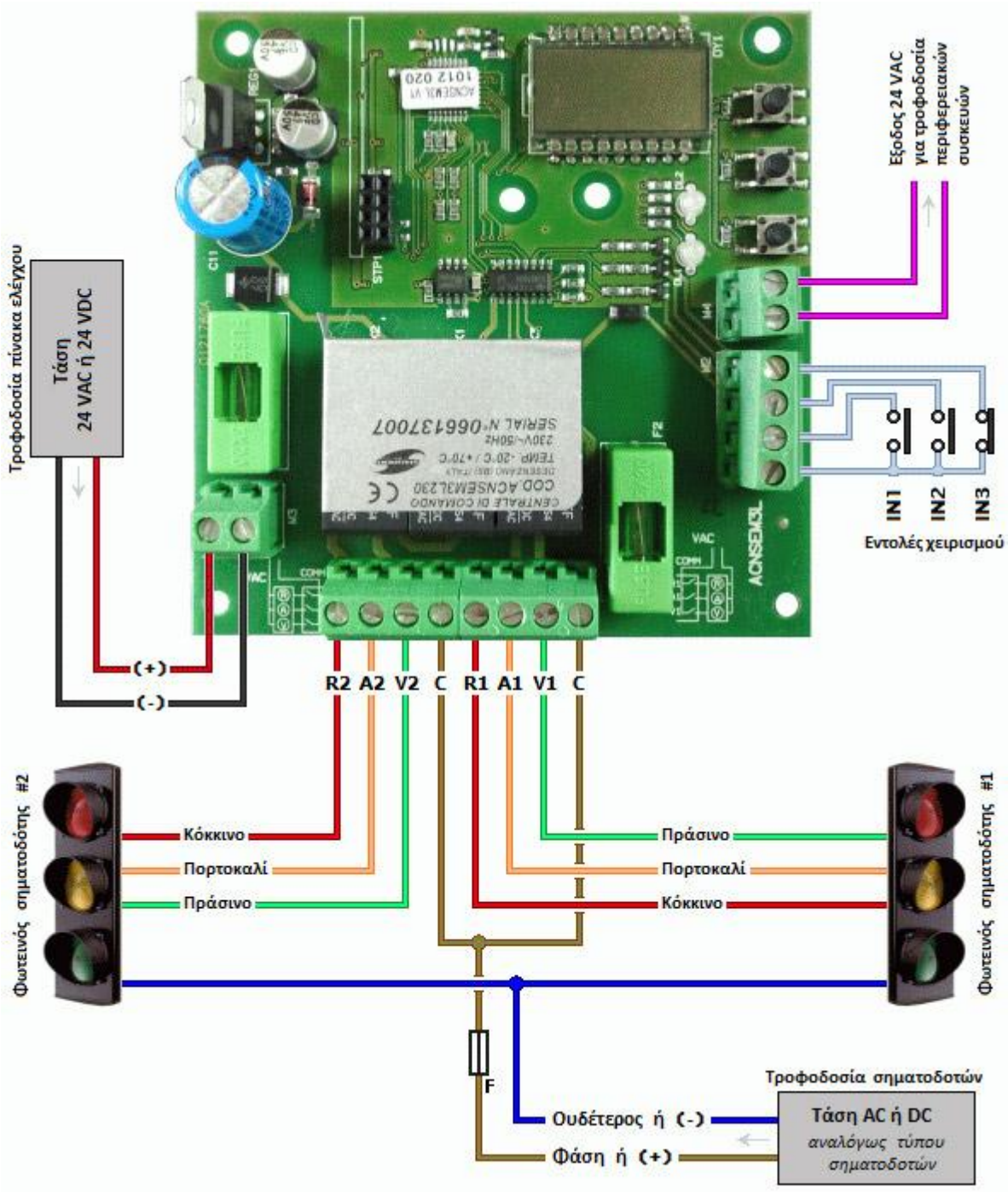
Η τάση 24VAC για τροφοδοσία παρελκομένων αποτελεί διευκόλυνση για την σύνδεση των αισθητήρων. Πρέπει όμως να γίνεται έλεγχος εάν όντως οι αισθητήρες λειτουργούν με τέτοια τάση και έχουν συνολική κατανάλωση την οποία μπορεί να καλύψει ο πίνακας. Εάν κάτι από αυτά δεν ισχύει, απαιτείται εξωτερική πηγή τροφοδοσίας των αισθητήρων με ρεύμα (είτε άμεσα από το δίκτυο εάν λειτουργούν με 230VAC, είτε μέσω ξεχωριστού τροφοδοτικού).

Η συνδεσμολογία του πίνακα περιγράφεται αναλυτικά στα παρακάτω σχήματα:

Συνδεσμολογία με σηματοδότες 2 πεδίων (κόκκινο & πράσινο)



Συνδεσμολογία με σηματοδότες 3 πεδίων (κόκκινο, πορτοκαλί & πράσινο)


Τροφοδοσία πίνακα ελέγχου
Τάση
24 VAC ή 24 VDC
(+)
(-)
Έξοδος 24 VAC
για τροφοδοσία
περιφερειακών
συσκευών
IN1
IN2
IN3
Εντολές χειρισμού
R2 A2 V2 C R1 A1 V1 C
Φωτεινός σηματοδότης #2
Κόκκινο
Πορτοκαλί
Πράσινο
Πράσινο
Πορτοκαλί
Κόκκινο
Φωτεινός σηματοδότης #1
F
Τροφοδοσία σηματοδοτών
Ουδέτερος ή (-)
Φάση ή (+)
Τάση AC ή DC
αναλόγως τύπου
σηματοδοτών

ΚΕΦΑΛΑΙΟ 4

ΟΙ ΕΥΡΕΤΙΚΟΙ ΚΑΙ ΜΕΤΑΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ ΓΙΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΧΡΟΝΟΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΤΩΝ ΦΩΤΕΙΝΩΝ ΣΗΜΑΤΟΔΟΤΩΝ

Η αυξημένη χρήση των οχημάτων στους δρόμους και κυρίως των πόλεων έχει σαν αποτέλεσμα την δημιουργία κυκλοφοριακών 'κουμφούζιων' δημιουργώντας έτσι προβλήματα στην μετακίνηση των οχημάτων έκτακτης ανάγκης. Γι' αυτό το λόγο η παρούσα πτυχιακή έχει ως σκοπό την διευκόλυνση των οχημάτων αυτών ώστε να μπορούν να κινηθούν άμεσα στο σημείο όπου υπάρχει ανάγκη.

Η σημασία του έργου είναι πολύ μεγάλη για τα οχήματα εκτάκτως ανάγκης, καθώς επιτυγχάνεται βελτιστοποίηση της κυκλοφορίας με την άμεση προσαρμογή της λειτουργίας των φωτεινών σηματοδοτών στις κυκλοφοριακές συνθήκες, συμπεριλαμβανομένων επεμβάσεων για την προτεραιότητα των οχημάτων αυτών. Ταυτόχρονα, το διαρκώς αυξανόμενο κυκλοφοριακό πρόβλημα όλων των μεγάλων πόλεων του κόσμου επιβάλλει μία νέα προσέγγιση στη διαχείριση της σηματοδότησης, που ξεφεύγει από την κλασική αντιμετώπιση με τους στατικούς σηματοδότες, και που οδηγεί στην εφαρμογή ολοκληρωμένων «έξυπνων» συστημάτων διαχείρισής της, στα πλαίσια ευρύτερων περιοχών μέσα σε ένα αστικό δίκτυο. Προς το σκοπό αυτό, την καθοριστική ώθηση δίνουν η ανάπτυξη των ηλεκτρονικών υπολογιστών, της τηλεματικής των τηλεπικοινωνιών και των αυτοματισμών. Τέλος, η ανάπτυξη ενός συστήματος διαχείρισης και παρακολούθησης της κυκλοφορίας δίνει τη δυνατότητα υποστήριξης και συντονισμού των ενεργειών των αρμοδίων υπηρεσιών στην αντιμετώπιση των συμβάντων με αποτέλεσμα τη βελτίωση εξυπηρέτησης των πολιτών.

Στήνοντας τώρα έναν αλγόριθμό για την βελτιστοποίηση του χρονοπρογραμματισμού των φωτεινών σηματοδοτών θα πρέπει να υποθέσουμε ότι οι διεργασίες που θα πρέπει να συνυπολογιστούν είναι απόλυτα εξαρτημένες μεταξύ τους.

Στη πραγματικότητα αυτή η συνεργασία μεταξύ των διεργασιών μπορεί να εκφραστεί με τους παρακάτω τρόπους: μερικές διεργασίες πρέπει να σεβαστούν μία καθορισμένη διάταξη, μία ανταλλαγή δεδομένων μεταξύ τους ή ακόμα και τη χρήση

πόρων, συνήθως με αποκλειστικό τρόπο. Απ' τη σκοπιά του προσαρμοστικού μοντέλου, υπάρχουν δύο είδη εξαρτήσεων στα συστήματα πραγματικού χρόνου:

- Περιορισμοί προτεραιότητας που αντιστοιχούν στο συγχρονισμό ή στην επικοινωνία μεταξύ των διεργασιών

- Περιορισμοί αμοιβαίου αποκλεισμού με σκοπό την προστασία κοινόχρηστων πόρων. Αυτοί οι κρίσιμοι πόροι μπορεί να είναι δομές δεδομένων, περιοχές μνήμης, εξωτερικές συσκευές, καταχωρητές, κ.τ.λ.

Έστω ότι έχουμε ένα αλγόριθμο πραγματικού χρόνου με δικαίωμα διακοπής, όπου οι προτεραιότητες των διεργασιών είναι σταθερές. Ερευνούμε το άνω όριο του χρόνου απόκρισης μίας διεργασίας τ_0, η οποία έχει χειρότερο χρόνο εκτέλεσης C_0. Να θεωρηθεί ότι ο παράγοντας του φόρτου του επεξεργαστή είναι όσο χαμηλός απαιτείται έτσι ώστε να επιτρέψει στο σύνολο των διεργασιών, συμπεριλαμβανομένου της διεργασίας τ_0, να χρονοδρομολογηθεί ανεξάρτητα από το χρόνο διακοπής εξαιτίας των κοινόχρηστων πόρων.

Αρχικά υποθέτουμε ότι οι διεργασίες είναι ανεξάρτητες, π.χ. χωρίς καθόλου κοινούς πόρους. Εάν η διεργασία τ_0 έχει την υψηλότερη προτεραιότητα, είναι προφανές ότι ο χρόνος απόκρισης TR_0 της διεργασίας τ_0 ισούται με το χρόνο εκτέλεσης της C_0. Ενώ εάν μία διεργασία έχει μία ενδιάμεση προτεραιότητα, το άνω όριο της απόκρισης της μπορεί να υπολογιστεί εύκολα σε συνάρτηση με τις διεργασίες με υψηλότερη προτεραιότητα από την $\tau_0(\tau_{hpt})$:

- Όταν όλες οι διεργασίες είναι περιοδικές με την ίδια περίοδο ή απεριοδικές, τότε:

$$TR_0 \leq C_0 + \sum_{i \in HPT} C_i$$

- Όταν όλες οι διεργασίες είναι περιοδικές, αλλά με διαφορετικές περιόδους τότε:

$$TR_0 \leq C_0 + \sum_{i \in HPT} \left\lceil \frac{T_0}{T_i} \right\rceil C_i$$

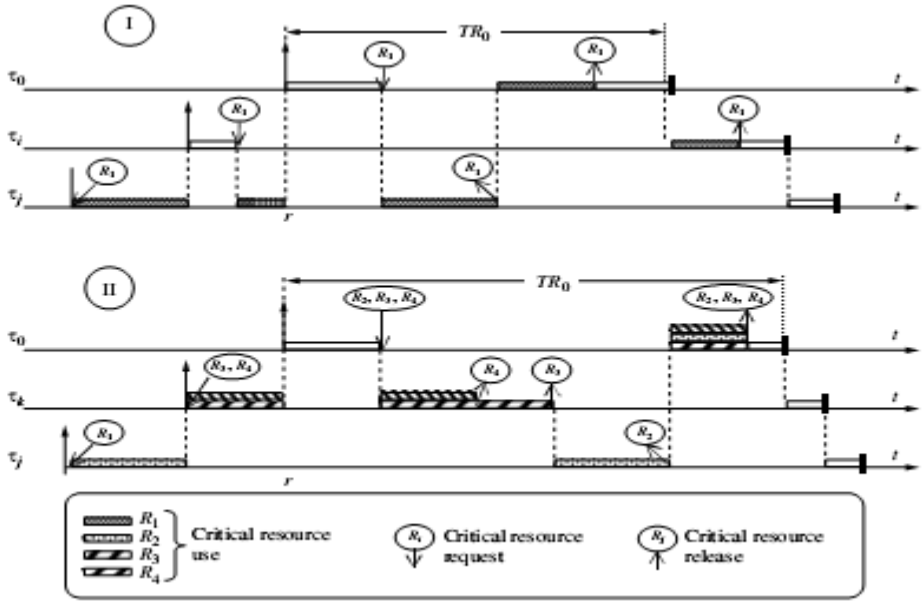
Στο δεύτερο βήμα, θεωρούμε ότι οι διεργασίες μοιράζονται πόρους. Λαμβάνοντας υπόψη τη διεκπεραίωση των διεργασιών ή την πρόσβαση στους

πόρους, ο χειρισμός όλων των ουρών γίνεται σύμφωνα με τις προτεραιότητες των διεργασιών. Εξ' άλλου, θεωρούμε ότι η υπερφόρτωση εξαιτίας των μηχανισμών πυρήνα (πρόσβαση στους πόρους, καταμερισμός των διεργασιών σε ουρές, διακοπές) είναι αμελητέα. Φυσικά, αυτές οι υπερφορτώσεις μπορούν να ληφθούν υπόψη ως ένας ακόμα όρος των χρόνων εκτέλεσης των διεργασιών.

Σε ένα σύνολο από *n+1* διεργασίες και *m* πόρους, υπολογίζουμε το άνω όριο του χρόνου απόκρισης της διεργασίας τ_0 (i) όταν έχει και (ii) όταν δεν έχει τη μέγιστη προτεραιότητα. Όταν η τ_0 έχει τη μέγιστη προτεραιότητα , η εκτέλεση της μπορεί να καθυστερήσει μόνο από τις διεργασίες που έχουν χαμηλότερες προτεραιότητες και χειρίζονται κοινούς πόρους m_0. Αυτή η περίπτωση πρέπει να ληφθεί υπόψη για δύο περιπτώσεις:

- Περίπτωση Ι: Οι κοινοί πόροι m_0 ελέγχονται από τουλάχιστον m_0 διεργασίες όπως φαίνεται στο παρακάτω σχήμα, όπου κάθε διεργασία τ_j ελέγχει τον πόρο R_1 που ζητείται να προσπελασθεί και από τη διεργασία τ_0. Είναι σημαντικό να κατανοηθεί ότι η διεργασία τ_i διακόπτεται από τη διεργασία τ_0 εξαιτίας της διαχείρισης των προτεραιοτήτων των ουρών. Έστω $CR_{i,q}$ ο μέγιστος χρόνος όπου η διεργασία τ_i χρησιμοποιεί τον πόρο R_q , $CR_{max,q}$ είναι το μέγιστο $CR_{i,q}$ από όλες τις διεργασίες τ_i , $CR_{i,max}$ το μέγιστο $CR_{i,q}$ από όλους τους πόρους R_q και CR_{max} το μέγιστο $CR_{i,q}$ όλων των διεργασιών και πόρων. Συνεπώς, το άνω όριο του χρόνου απόκρισης της διεργασίας τ_0 δίνεται από τη σχέση:

$$TR_0 \leq C_0 + \sum_{i=1}^{m_0} \mathrm{CR}_{\mathrm{i,max}}$$



Χρόνος απόκρισης υψηλής προτεραιότητας με κοινούς πόρους: Περίπτωση I: δύο χαμηλότερης προτεραιότητας διεργασίες μοιράζονται ένα κρίσιμο πόρο με τη διεργασία τ_0. Περίπτωση II: δύο χαμηλότερης προτεραιότητας διεργασίες μοιράζονται τρεις κρίσιμους πόρους με τη διεργασία τ_0.

Στη χειρότερη περίπτωση, για το σύνολο των διεργασιών ο χρόνος απόκρισης θα είναι το πολύ:

$$TR_0 \leq C_0 + m * CR_{max}$$

Ή με περισσότερη ακρίβεια:

$$TR_0 \leq C_0 + \sum_{i=1}^{m} \mathrm{CR}_{\mathrm{i,max}}$$

- Περίπτωση II: Οι κοινοί πόροι της m_0 ελέγχονται από n_1 διεργασίες όπου $n_1 < m_0$, όπως φαίνεται στο παρακάτω σχήμα, όπου οι διεργασίες τ_k και τ_j προσπελαύνουν τους πόρους R_2, R_3 και R_4 οι οποίοι απαιτείται να προσπελασθούν και από την τ_0. Μπορούμε να διαπιστώσουμε, ότι τουλάχιστον μία διεργασία προσπελαύνει δύο πόρους. Εάν υποθέσουμε, ότι τα κρίσιμα τμήματα μίας διεργασίας είναι κατάλληλα εμφωλευμένα, η μέγιστη διάρκεια κρίσιμου τμήματος μίας η οποία χειρίζεται πάνω από ένα πόρο καθορίζεται από το μέγιστο κρίσιμο τμήμα. Έτσι ο χρόνος απόκρισης της διεργασίας τ_0 έχει πάνω όριο:

$$TR_0 \leq C_0 + n_1 * CR_{max}$$

Ή με περισσότερη ακρίβεια:

$$TR_0 \leq C_0 + \sum_{q=1}^{n_1} \mathrm{CR}_{\max,q}$$

Στη χειρότερη περίπτωση, για αυτό το σύνολο από τις διεργασίες (*n* διεργασίες και *m πόροι*, με *n*<*m*), ο χρόνος απόκρισης της διεργασίας τ_0 είναι το πολύ:

$$TR_0 \leq C_0 + n * CR_{max}$$

Ή με περισσότερη ακρίβεια:

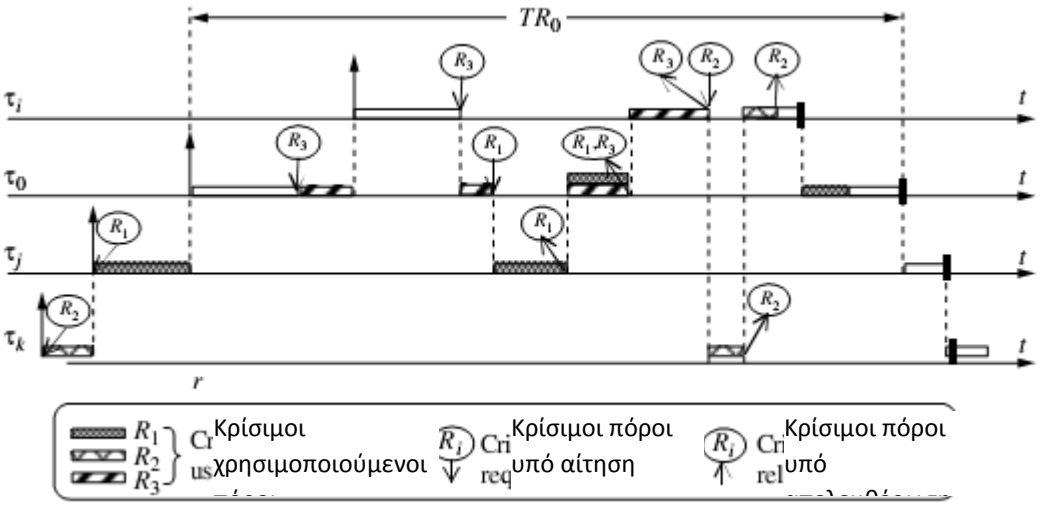
$$TR_0 \leq C_0 + \sum_{q=1}^{n} \mathrm{CR}_{\max,q}$$

Συνοψίζοντας, μία γενική έκφραση του χρόνου απόκρισης τις διεργασίες υψηλής προτεραιότητας σε εφαρμογές πραγματικού χρόνου, που απαρτίζεται από *n+1* διεργασίες και *m* πόρους είναι η παρακάτω:

$$TR_0 \leq C_0 + \inf(n,m) * \mathrm{CR}_{\max}$$

Έστω η διεργασία τ_0 έχει κάποια ενδιάμεση προτεραιότητα. Το σύνολο των διεργασιών περιλαμβάνει n_1 διεργασίες με προτεραιότητες υψηλού επιπέδου (High Priority Level Task– HPT σύνολο) και n_2 διεργασίες με προτεραιότητες χαμηλότερου επιπέδου και οι οποίες μοιράζονται *m* κρίσιμους πόρους με τη διεργασία τ_0. Η περίπτωση αυτή απεικονίζεται στο παρακάτω σχήμα με τις παρακάτω τιμές: $n_1 = 1$, $n_2 = 2$, $m = 3$. Με την υπόθεση ότι οι n_2 διεργασίες χαμηλής προτεραιότητας έχουν εξαρτήσεις μόνο με την τ_0 και όχι με τις n_1 διεργασίες υψηλότερης προτεραιότητας, είναι δυνατών να υπολογίσουμε το ανώτατο όριο του χρόνου απόκρισης της διεργασίας τ_0 συνδυάζοντας τις ανισότητες. Ο χρόνος απόκρισης είναι:

$$TR_0 \leq C_0 + \inf(n_1,m) * \mathrm{CR}_{\max} + \sum_{i \in HPT} \left\lceil \frac{T_0}{T_i} \right\rceil C_i$$



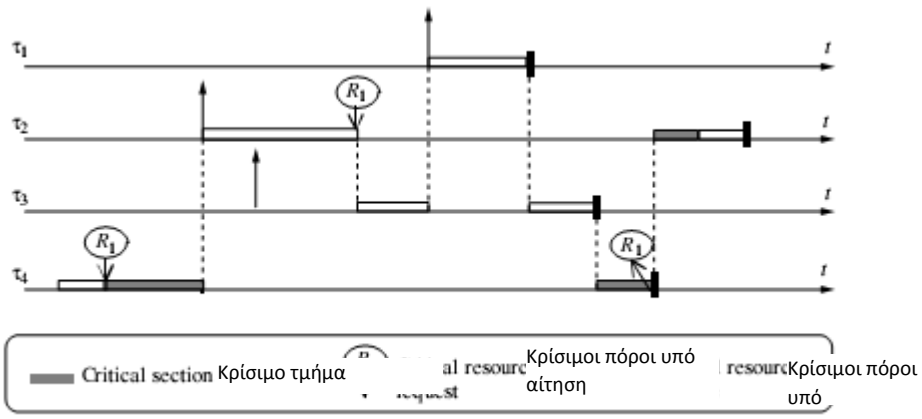
Χρόνος απόκρισης διεργασιών που μοιράζονται κρίσιμους πόρους: $Prio_i > Prio_0 > Prio_j > Prio_k$.

Ωστόσο, ο υπολογισμός του ανώτατου ορίου της κάθε διεργασίας βασίζεται σε κάποιες προϋποθέσεις σχετικά με τους κανόνες χρονοδρομολόγησης. Συγκεκριμένα, για ένα αλγόριθμο που επιτρέπονται οι διακοπές με καθορισμένες προτεραιότητες εξ αρχής, υπάρχει μία απόλυτη συνθήκη που δεν πρέπει να παραβούμε: στο χρόνο ενεργοποίησης της, μία διεργασία τ_0 πρέπει να εκτελεστεί μόλις όλες οι διεργασίες υψηλής προτεραιότητας εκτελεστούν και όλες με χαμηλή προτεραιότητα οι οποίες χειρίζονται κρίσιμους πόρους, τους οποίους ζητάει να προσπελάσει και η τ_0, απελευθερώσουν τα κρίσιμα σημεία που εμπλέκονται. Στην πραγματικότητα δύο προβλήματα χρονοδρομολόγησης μπορούν να ακυρώσουν αυτή την προϋπόθεση: Το φαινόμενο προτεραιότητας αντιστροφής (priority inversion phenomenon) και το αδιέξοδο (deadlock).

Στην χρονοδρομολόγηση με δικαίωμα διακοπής η οποία καθοδηγείται από σταθερές προτεραιότητες και κρίσιμοι πόροι προστατεύονται από ένα μηχανισμό αμοιβαίου αποκλεισμού, εμφανίζονται φαινόμενα προτεραιότητας αντιστροφής. Παρακάτω έχουμε ένα παράδειγμα όπου φαίνεται αυτό το πρόβλημα. Έστω τέσσερις διεργασίες {τ1, τ2, τ3, τ4 } οι οποίες έχουν φθίνουσες προτεραιότητες. Οι διεργασίες τ2 και τ4 μοιράζονται κρίσιμους πόρους R1 και η πρόσβαση σε αυτούς γίνεται με αμοιβαίο αποκλεισμό. Η μελέτη αυτή εστιάζεται στο χρόνο απόκρισης της διεργασίας τ2. Η διεργασία με τη χαμηλότερη προτεραιότητα τ4 ξεκινάει την εκτέλεση της πρώτη και μετά από κάποιο διάστημα μπαίνει σε κρίσιμο τμήμα με τη χρήση των

πόρων R1. Όταν η διεργασία τ4 βρίσκεται στο κρίσιμο τμήμα της, η διεργασία τ2 με υψηλότερη προτεραιότητα, απελευθερώνεται και διακόπτει τη διεργασία τ4 . Κατά τη διάρκεια της εκτέλεσης της τ2 , απελευθερώνεται και η τ3 . Εν τούτοις, η διεργασία τ3 , αφού έχει μικρότερη προτεραιότητα από την τ2 πρέπει να περιμένει. Όταν η τ2 χρειάζεται να μπει στο κρίσιμο της τμήμα, δηλαδή τους πόρους R1 (οι οποίοι είναι κοινοί με το τ4), βλέπει ότι οι πόροι αυτοί είναι δεσμευμένοι από τη διεργασία τ4 . Έτσι η εκτέλεση της εμποδίζεται. Η διεργασία με την υψηλότερη προτεραιότητα που επιτρέπεται να εκτελεστεί είναι η διεργασία τ3. Έτσι η διεργασία τ3 είναι αυτή που απασχολεί τον επεξεργαστεί και εκτελείται.

Κατά τη διάρκεια της εκτέλεσης, η διεργασία με τη μέγιστη προτεραιότητα αφυπνίζεται. Αυτό έχει ως συνέπεια η τ3 να ανασταλεί και η υπολογιστική ισχύ να εκχωρηθεί στη διεργασία τ1 . Μετά την ολοκλήρωση της εκτέλεσης της τ1, η τ3 μπορεί να συνεχίσει την εκτέλεσή της μέχρι να ολοκληρωθεί και αυτή. Τώρα μόνο η διεργασία με τη χαμηλότερη προτεραιότητα, η οποία είχε διακοπεί πριν, μπορεί να εκτελεστεί ξανά. Συνεχίζει την εκτέλεση της μέχρι να απελευθερώσει τους κρίσιμους πόρους R1, οι οποίοι χρειάζονται από τη διεργασία υψηλότερης προτεραιότητας τ2. Μετά μπορεί να συνεχίσει την εκτέλεσή της και αυτή η διεργασία χρησιμοποιώντας τους κρίσιμους πόρους R1, οι οποίοι και είναι απαραίτητοι για τις δραστηριότητές της.



Παράδειγμα φαινόμενου προτεραιότητας αντιστροφής (Priority inversion phenomenon).

Είναι μεγάλης σημασίας να αναλυθεί αυτό το απλό παράδειγμα επακριβώς. Ο μέγιστος χρόνος διακοπής της διεργασία τ2 μπορεί να εξαρτηθεί από τη διάρκεια των κρίσιμων τμημάτων των διεργασιών χαμηλότερης προτεραιότητας, που μοιράζονται

πόρους με αυτήν, όπως η διεργασία τ4, αλλά και από τους χρόνους εκτέλεσης των διεργασιών υψηλότερης προτεραιότητας, όπως της τ1. Αυτοί οι δύο λόγοι αύξησης του χρόνου απόκρισης της τ2 είναι απολύτως συνεπείς με τους κανόνες χρονοδρομολόγησης. Όμως, μια άλλη διεργασία, η τ3, η οποία έχει χαμηλότερη προτεραιότητα και δε μοιράζεται κρίσιμους πόρους με την τ2, συμμετέχει στην αύξηση του χρόνου διακοπής της. Αυτή η κατάσταση ονομάζεται αναστροφή προτεραιότητας (priority inversion), και η οποία καταστρατηγεί τις προδιαγραφές της χρονοδρομολόγησης και μπορεί να προκαλέσει το ξεπέρασμα των προθεσμιών (deadlines).

Η χρονοδρομολόγηση διεργασιών που μοιράζονται κρίσιμους πόρους οδηγεί σε ορισμένα προβλήματα σε όλες τις εφαρμογές της επιστήμης των υπολογιστών:

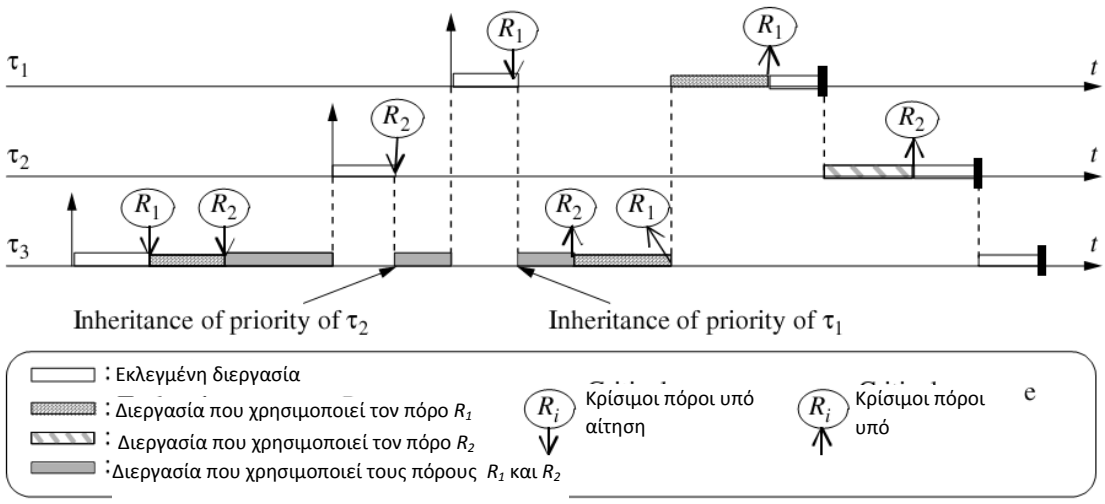
- συγχρονισμός των προβλημάτων μεταξύ των διεργασιών και ιδιαιτέρως η κατάσταση αντιστροφής προτεραιοτήτων όταν μοιράζονται πόρους αμοιβαίου αποκλεισμού
- προβλήματα αδιεξόδου και αλυσιδωτών δεσμευμένων διεργασιών

Στα συστήματα πραγματικού χρόνου, μία απλή μέθοδος που αντιμετωπίζει αυτά τα προβλήματα είναι η προκράτηση και προδέσμευση των πόρων στην αρχή της εκτέλεσης της διεργασίας. Όμως, αυτή η τεχνική οδηγεί σε χαμηλού επιπέδου χρησιμοποίηση των πόρων. Γι’ αυτό και έχουν σχεδιαστεί ορισμένα πρωτόκολλα πρόσβασης πόρων έτσι ώστε να αποφευχθούν αυτά τα ελαττώματα αλλά και να οριοθετηθεί ο μέγιστος χρόνος απόκρισης των διεργασιών.

Διάφορα πρωτόκολλα έχουν αναπτυχθεί για την αποτροπή της αντιστροφής της προτεραιότητας σε RM και EDF χρονοδρομολογητές. Αυτά τα πρωτόκολλα επιτρέπουν να καθοριστεί το ανώτατο όριο του χρόνου διακοπής, ο οποίος οφείλεται σε πρόσβαση σε κρίσιμους πόρους για κάθε διεργασία ti. Το όριο αυτό ορίζεται ως Bi. Αυτός ο μέγιστος χρόνος διάρκειας κατά την οποία μένει σταματημένη μία διεργασία κατόπιν ενσωματώνεται στο χρονοπρογραμματισμό κλασικών αλγορίθμων χρονοπρογραμματισμού όπως του RM και του EDF(βλέπε κεφ. 2). Αυτή η ενσωμάτωση υλοποιείται εύκολα λαμβάνοντας υπόψη ότι μία διεργασία ti έχει χρόνο εκτέλεσης ίσο με Ci + Bi. Ορισμένοι από αυτούς τους αλγόριθμους πρόσβασης πόρων αποτρέπουν επίσης φαινόμενα αδιεξόδου.

Η βασική ιδέα του πρωτοκόλλου κληρονομικής προτεραιότητας είναι να αλλάζει δυναμικά τις προτεραιότητες ορισμένων διεργασιών. Έτσι μία διεργασία τi., η οποία χρησιμοποιεί ένα κρίσιμο πόρο μέσα σε ένα κρίσιμο τμήμα παίρνει την τιμή της προτεραιότητας κάθε διεργασίας τj που περιμένει για αυτόν τον πόρο, εάν η προτεραιότητα της διεργασίας τj είναι μεγαλύτερη από αυτήν της τi. Συνεπώς, η διεργασία τi χρονοδρομολογείται σε ένα υψηλότερο επίπεδο από το επίπεδο της αρχικής της προτεραιότητας. Αυτό το νέο στοιχείο απελευθερώνει τους κρίσιμους πόρους νωρίτερα και ελαχιστοποιεί το χρόνο αναμονής της διεργασίας υψηλής προτεραιότητας τj. Το πρωτόκολλο κληρονομικής προτεραιότητας δεν αποτρέπει τα αδιέξοδα, τα οποία πρέπει να αποφευχθούν με τη χρήση των τεχνικών που συζητήθηκαν παραπάνω. Έτσι, το πρωτόκολλο κληρονομικής προτεραιότητας πρέπει να χρησιμοποιείται σε κώδικα διεργασιών με σωστά εμφωλευμένα τμήματα. Σε αυτήν την περίπτωση το πρωτόκολλο εφαρμόζεται με επαναλαμβανόμενο τρόπο. Το πρωτόκολλο αυτό έχει υλοποιηθεί στο λειτουργικό σύστημα πραγματικού χρόνου DUNE-IX.

Στο παρακάτω σχήμα δίνεται ένα παράδειγμα αυτού του πρωτοκόλλου για ένα σύνολο τριών διεργασιών {τ1, τ2, τ3}, οι οποίες έχουν φθίνουσες προτεραιότητες και δύο κρίσιμους πόρους { R1, R2}. Η διεργασία τ1 χρησιμοποιεί τον πόρο R1, η τ2 τον πόρο R2 και η τ3 και τους δύο πόρους R1 και R2. Η διεργασία τ3 ξεκινάει πρώτη να εκτελείται και χρησιμοποιεί τους πόρους R1 και R2. Αργότερα η διεργασία τ2 αφυπνίζεται και διακόπτει την τ3 στο εμφωλευμένο κρίσιμο τμήμα της. Όταν η διεργασία τ2 χρειαστεί τον πόρο R2, επειδή έχει δεσμευθεί από τη διεργασία τ3, η τ3 παίρνει την προτεραιότητας από την τ2. Λέμε ότι η τ3 κληρονομεί την προτεραιότητα από την τ2. Μετά, με τον ίδιο τρόπο, η τ1 αφυπνίζεται και διακόπτει την τ3 στο κρίσιμό της σημείο. Όταν η διεργασία τ1 ζητήσει την πρόσβαση στον πόρο R1, αυτός είναι δεσμευμένος από την τ3 και κατά συνέπεια η διεργασία τ3 κληρονομεί την προτεραιότητα από την τ1. Έτσι η διεργασία τ3 συνεχίζει την εκτέλεσή της, αφού πλέον έχει τη μέγιστη προτεραιότητα του συνόλου των διεργασιών. Όταν η διεργασία τ3 απελευθερώσει τους πόρους R2 και R1, συνεχίζει με την αρχική της προτεραιότητα. Έτσι άμεσα, η υψηλής προτεραιότητας διεργασία τ1 και η οποία περίμενε για ένα πόρο να απελευθερωθεί, διακόπτει τη διεργασία τ3 και καταλαμβάνει τον επεξεργαστή. Το τέλος της ακολουθίας εκτέλεσης τηρεί τους κλασικούς κανόνες της χρονοδρομολόγησης.



Παράδειγμα εφαρμογής του πρωτοκόλλου κληρονομικής προτεραιότητας.

Όταν χρησιμοποιείται το πρωτόκολλο κληρονομικής προτεραιότητας, είναι δυνατών να εκτιμήσουμε το ανώτατο όριο του χρόνου διακοπής της κάθε διεργασίας. Με την εφαρμογή αυτού του πρωτοκόλλου, μία διεργασία τ1 μπορεί να διακοπεί το πολύ σε n κρίσιμα τμήματα από χαμηλότερης προτεραιότητας διεργασίες ή από m κρίσιμα τμήματα που συμπίπτουν με τους κοινόχρηστους πόρους με διεργασίες χαμηλότερης προτεραιότητας. Ο χρόνος αυτό ισούται:

$$Bi \leq \inf(n,m) * CRmax$$

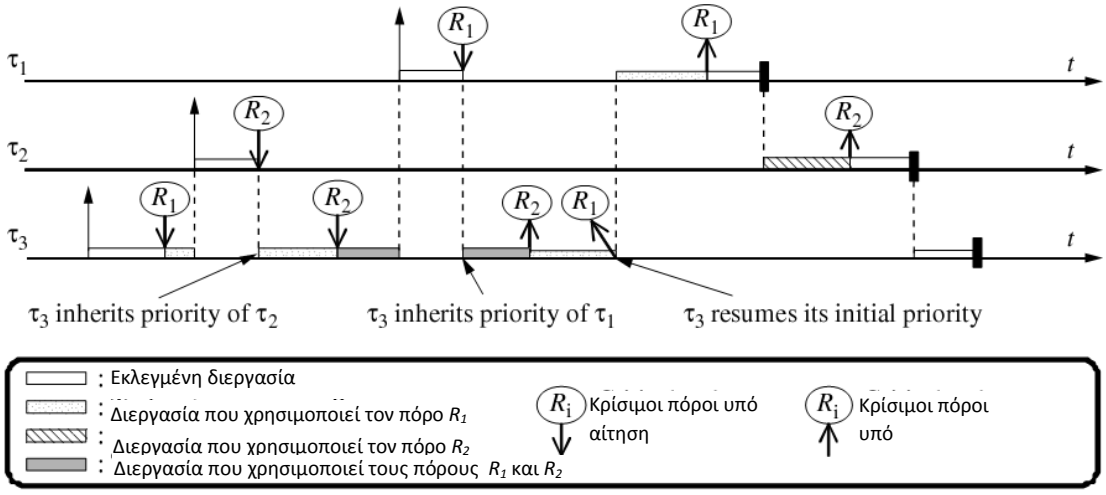
Όπως μπορούμε να δούμε στο παραπάνω σχήμα, η διεργασία τ2 έχει καθυστερήσει πιο πολύ από το κρίσιμο τμήμα της διεργασίας τ3 (επισημαίνεται ότι τα κρίσιμα τμήματα που χρησιμοποιούνται σε μία διεργασία πρέπει να είναι σωστά εμφωλευμένα. Στο παράδειγμά μας, η R1 απελευθερώνεται μετά την R2).

Η βασική ιδέα αυτού του πρωτοκόλλου είναι η επέκταση του προηγούμενου πρωτοκόλλου με σκοπό να αποφευχθούν τα αδιέξοδα και οι αλυσιδωτές διακοπές, τα οποία εμποδίζουν μία διεργασία να μπει σε ένα κρίσιμο της τμήμα. Έτσι σε κάθε πόρο ανατίθεται μία προτεραιότητα, η οποία ονομάζεται προτεραιότητα οροφής (ceiling priority) και η οποία ισούται με την προτεραιότητα της διεργασίας με την υψηλότερη προτεραιότητα που μπορεί να τον χρησιμοποιήσει. Η προτεραιότητα οροφής είναι παρόμοια με ένα κατώφλι (threshold). Με τον ίδιο τρόπο με το πρωτόκολλο κληρονομικής προτεραιότητας, μία διεργασία τi η οποία χρησιμοποιεί ένα κρίσιμο πόρο σε ένα κρίσιμο τμήμα, παίρνει την τιμή της προτεραιότητας τj η

οποία αναμένει αυτόν τον πόρο να απελευθερωθεί, εάν η προτεραιότητα της tj είναι υψηλότερη από αυτή της ti. Συνεπώς, η διεργασία τι χρονοδρομολογείται σε ένα υψηλότερο επίπεδο προτεραιότητας από το αρχικό της και ο χρόνος αναμονής της υψηλότερης προτεραιότητας διεργασίας tj ελαχιστοποιείται. Επιπλέον, με σκοπό να απαλείψουμε τα αδιέξοδα, όταν μία διεργασία ζητάει έναν πόρο, ο πόρος αυτό ανατίθεται μόνο εάν αυτός είναι ελεύθερος και η προτεραιότητα αυτής της διεργασίας είναι υψηλότερη από την προτεραιότητα οροφής των πόρων που χρησιμοποιούνται από τις άλλες διεργασίες. Αυτός ο κανόνας παρέχει έγκαιρο σταμάτημα των διεργασιών μπορούν να προκαλέσουν αδιέξοδο και εγγυάται ότι οι μελλοντικές διεργασίες υψηλής προτεραιότητας θα αποκτήσουν πρόσβαση στους πόρους τους.

Στο παρακάτω σχήμα φαίνεται ένα παράδειγμα πρωτοκόλλου για ένα σύνολο διεργασιών {τ1,τ2,τ3} με φθίνουσες προτεραιότητες των δύο κρίσιμων πόρων {R1, R2}. Η διεργασία τ1 χρησιμοποιεί τον πόρο R1, η διεργασία τ2 τον πόρο R2 και η διεργασία τ3 και τους δύο πόρους R1 και R2. Η διεργασία τ3 ξεκινά πρώτη να τρέχει και δεσμεύει τον πόρο R1, ο οποίος είναι ελεύθερος.

Το ανώτατο όριο προτεραιότητας της του πόρου R1 (αντίστοιχα του R2) ισούται με την προτεραιότητα της διεργασίας τ1 (αντίστοιχα του τ2). Αργότερα, αφυπνίζεται η διεργασία τ2 και διακόπτει την τ3 , αφού έχει μεγαλύτερη προτεραιότητα από αυτήν. Όταν η τ2 ζητήσει πρόσβαση στον πόρο R2 , εμποδίζεται η ενέργειά της από το πρωτόκολλο γιατί η προτεραιότητα της δεν είναι απόλυτα μεγαλύτερη από την ανώτατη προτεραιότητα του πόρου R1. Απ' την στιγμή που η τ2 αναμένει, η διεργασία τ3 κληρονομεί την προτεραιότητα της τ2 και συνεχίζει την εκτέλεση της. Με τον ίδιο τρόπο, η διεργασία τ1 αφυπνίζεται και διακόπτει την τ3, από τη στιγμή που έχει μεγαλύτερη προτεραιότητα από αυτήν. Όταν η τ1 ζητήσει να έχει πρόσβαση στον πόρο R1, εμποδίζεται από το πρωτόκολλο, επειδή η προτεραιότητα δεν είναι αυστηρά μεγαλύτερη από την ανώτατη προτεραιότητα του πόρου R1 που χρησιμοποιείται. Από την στιγμή που η τ1 αναμένει, η διεργασία τ3 κληρονομεί την προτεραιότητα της τ1 και συνεχίζει την εκτέλεσή της. Όταν η τ3 φύγει από τα κρίσιμα τμήματα και των δύο πόρων R2 και R1, συνεχίζει με την αρχική της προτεραιότητα και διακόπτεται άμεσα από τη διεργασία τ1, η οποία και έχει υψηλότερη προτεραιότητα. Το τέλος της ακολουθίας εκτέλεσης ακολουθεί τους κλασσικούς κανόνες χρονοδρομολόγησης.



Παράδειγμα εφαρμογής του πρωτοκόλλου προτεραιότητας οροφής.

Είναι σημαντικό να επισημανθεί ότι αυτό το πρωτόκολλο απαιτεί να γνωρίζουμε εκ των προτέρων όλες τις προτεραιότητες των δραστηριοτήτων και όλους τους πόρους που χρησιμοποιούνται από κάθε διεργασία με σκοπό να οριστούν οι ανώτατες τιμές των προτεραιοτήτων.

Τα τελευταία χρόνια έχουν εμφανιστεί κάποιες προτάσεις επέκτασης του πρωτοκόλλου αυτού, π.χ. (immediate priority ceiling protocol), η οποία είναι μία πιο άμεση προσέγγιση και αυξάνει τις προτεραιότητες των διεργασιών όταν αυτή κλειδώνει ένα πόρο, παρά όταν αυτή σταματάει μία υψηλότερης προτεραιότητας διεργασία. Η συμπεριφορά και των δύο αλγορίθμων στην χειρότερη περίπτωση (worst-case) είναι ακριβώς ίδια.

Το πρωτόκολλο συσσώρευσης των πόρων επεκτείνει το προηγούμενο πρωτόκολλο με δύο τρόπους: επιτρέπει τη χρήση πολυδύναμων πόρων και μπορεί να εφαρμοστεί σε ένα αλγόριθμο χρονοδρομολόγησης μεταβλητών προτεραιοτήτων όπως τον αλγόριθμο προτεραιότητας μικρότερης προθεσμίας (earliest deadline first) [24]. Σε συμπλήρωμα προς την κλασσική προτεραιότητα, σε κάθε διεργασία ανατίθεται μία νέα παράμετρος π, η οποία ονομάζεται επίπεδο της διακοπής και σχετίζεται με το χρόνο που αφιέρωσε για να εκτελεστεί (π.χ. το π είναι αντιστρόφως ανάλογο στην προθεσμία ολοκλήρωσής της – deadline D). Το επίπεδο της διακοπής είναι τέτοιο, ώστε μία διεργασία τi να μη μπορεί να διακόψει μία διεργασία τj παρά μόνο εάν π(τi)> π(τj). Το επίπεδο της τρέχουσας διακοπής του συστήματος ορίζεται

σε συνάρτηση της πρόσβασης στους πόρους. Μία διεργασία δεν μπορεί να εκτελεστεί εάν το επίπεδο της διακοπής είναι χαμηλότερο από το συνολικό επίπεδο διακοπής. Η βασικότερη διαφορά αυτού του αλγορίθμου από τον προηγούμενο έγκειται στο χρόνο που εμποδίζεται η εκτέλεση μίας διεργασίας. Με το πρωτόκολλο προτεραιότητας οροφής, για μία διεργασία εμποδίζεται η εκτέλεση της όταν επιθυμεί να χρησιμοποιήσει ένα πόρο, ενώ με τη συσωρευτική πολιτική για τους πόρους εμποδίζεται η εκτέλεση μίας διεργασίας όταν επιθυμεί να έχει πρόσβαση στον επεξεργαστή.

Σε ένα σύστημα πραγματικού χρόνου, οι διεργασίες έχουν χρονικούς περιορισμούς και η εκτέλεσή τους πρέπει να πραγματοποιηθεί μέσα σε σαφή χρονικά όρια. Ο αντικειμενικός σκοπός της χρονοδρομολόγησης, είναι να επιτρέψει στις διεργασίες να εκτελεστούν ικανοποιώντας αυτούς τους χρονικούς περιορισμούς. Ο τρόπος χρονοδρομολόγησης πρέπει να προβλέπει ότι όλοι οι χρονικοί περιορισμοί θα εκπληρωθούν, όταν η εφαρμογή διεκπεραιώνεται με κανονικό τρόπο. Όταν συμβεί κάτι ασυνήθιστο ή μια βλάβη στην υπό έλεγχο διαδικασία, πρέπει να είναι σε θέση να ενεργοποιηθούν διεργασίες που θα σημάνουν συναγερμό (alarm tasks) και τότε ο σκοπός της χρονοδρομολόγησης είναι να κρατήσει την όλη διαδικασία ασφαλή έχοντας ανοχή στο επίπεδο της ποιότητας λειτουργίας της εφαρμογής.

Οι διεργασίες μιας εφαρμογής πραγματικού χρόνου μπορεί να προκαλούνται ταυτόχρονα και έχουν τον ίδιο και ταυτόχρονο χρόνο άφιξης ή να προκαλούνται προοδευτικά.

Μερικοί ακόμα χρήσιμοι ορισμοί για τη λειτουργία των διεργασιών, είναι οι παρακάτω:

- Ορίζεται ως *συντελεστής χρήσης* του επεξεργαστή του υπολογιστικού συστήματος *(processor utilization factor)* για ένα πλήθος από n περιοδικές διεργασίες, η έκφραση:

$$U = \sum_{i=1}^{n} \frac{Ci}{Ti}$$

- Κατά παρόμοιο τρόπο ορίζεται και ο συντελεστή φόρτου του επεξεργαστή (processor load factor) για ένα πλήθος από n περιοδικές διεργασίες, ως η έκφραση:

$$CH = \sum_{i=1}^{n} \frac{Ci}{Di}$$

- Ανοχή του επεξεργαστή (processor laxity):Επειδή οι παραπάνω εκφράσεις του processor utilization factor και του processor load factor δεν επαρκούν για να εκτιμήσουν εάν μια διεργασία κατά την εκτέλεσή της θα υπερβεί τις χρονικές προθεσμίες, εισάγεται η έννοια της ανοχή του επεξεργαστή (processor laxity) LP(t), που προσδιορίζει το μέγιστο χρόνο που ο επεξεργαστής μπορεί να παραμείνει ανενεργός χωρίς αυτό το γεγονός να κοστίσει στη διεργασία να χάσει κάποια χρονική προθεσμία. Ο χρόνος αυτός ποικίλλει συνεχώς ανάλογα σε πιο χρονικό σημείο εξέλιξης βρίσκεται η διεργασία και σε κάθε περίπτωση είναι LP(t) > 0. Υπολογίζεται και η εξαρτώμενη (conditional) laxity LCi(t) μιας διεργασίας i, όταν έχουν προηγηθεί j διεργασίες ως:

$$LCi(t) = Di - \sum_{i=1}^{n} Cj(t)$$

- *Ανενεργός χρόνος του επεξεργαστή (processor idle time): το χρονικό διάστημα στο οποίο δεν έχει ανατεθεί κάποια διεργασία στον επεξεργαστή.*

Η χρονοδρομολόγηση ενός πλήθους *(set)* από διεργασίες απαιτεί την κατάρτιση ενός αυστηρού πλαισίου με σκοπό την τήρηση των χρονικών περιορισμών, που να λαμβάνει υπόψη:

- την εκτέλεση όλων των διεργασιών όταν το σύστημα λειτουργεί με κανονικό τρόπο.

- Την εκτέλεση τουλάχιστον των διεργασιών που θεωρούνται ότι έχουν τον υψηλότερο βαθμό από άποψης σπουδαιότητας (για παράδειγμα αυτές που θα κρατήσουν το υπολογιστικό σύστημα και την υπό έλεγχο διαδικασία ασφαλή), όταν το σύστημα λειτουργεί με ανώμαλο τρόπο.

Ένας ανώμαλος τρόπος λειτουργίας μπορεί να προέλθει είτε από αστοχία του υλικού *(hardware faults)* είτε από άλλα μη αναμενόμενα συμβάντα.

Οι αλγόριθμοι χρονοδρομολόγησης αναθέτουν την εκτέλεση των διεργασιών στους επεξεργαστές και παρέχουν μια αυστηρά διατεταγμένη λίστα διεργασιών στην οποία

καθορίζεται η αλληλουχία και η προτεραιότητα εκτέλεσής τους. Οι κατηγορίες αλγορίθμων που χρησιμοποιούνται είναι οι παρακάτω:

- *Off-line*: ένας αλγόριθμος off-line καταρτίζει μια ολοκληρωμένη λίστα με την αλληλουχία των διεργασιών και τις παραμέτρους που καθορίζουν την κάθε μία. Η σειρά εκτέλεσης των διεργασιών είναι εκ των προτέρων γνωστή και δεν αλλάζει σε καμία περίπτωση. Προφανώς αυτή η προσέγγιση είναι στατική και άκαμπτη. Θεωρεί ότι όλες οι παράμετροι, όπως οι χρόνοι άφιξης των διεργασιών, οι χρόνοι εκτέλεσης, κλπ., είναι σταθεροί και έτσι δεν υπάρχει δυνατότητα προσαρμογής σε μεταβολές του εξωτερικού περιβάλλοντος.

- *On-line*: η κατηγορία αυτή έχει γνώση των παραμέτρων όλων των διεργασιών που έχουν προκαλέσει αίτηση για εκτέλεση και επιτρέπει κάθε χρονική στιγμή την επιλογή της επόμενης διεργασίας. Έτσι ανάλογα με τα εξωτερικά συμβάντα, ο αλγόριθμός αποφασίζει για το ποιά διεργασία θα ακολουθήσει αυτήν που εκτελείται. Αυτή η κατηγορία αλγορίθμων επιλέγεται όταν υπάρχει η ανάγκη για δυναμικό χειρισμό των διεργασιών και όταν υπάρχουν λίγες πληροφορίες για τη συμπεριφορά της υπό έλεγχο διαδικασίας. Χρησιμοποιείται κυρίως για χειρισμό απεριοδικών – σποραδικών διεργασιών και για συστήματα που θεωρείται σίγουρο ότι θα λειτουργήσουν με ανώμαλο τρόπο.

- *Preemptive*: Σε μια τέτοια κατηγορία αλγορίθμων χρονοδρομολόγησης, η εκτέλεση μιας διεργασίας μπορεί να διακοπεί βίαια πριν τον προβλεπόμενο χρόνο ολοκλήρωσης της όλης ενέργειάς της και να ανατεθεί στον επεξεργαστή μια άλλη διεργασία με μεγαλύτερη προτεραιότητα. Η διεργασία που διακόπτεται μεταπίπτει στην κατάσταση ready και αναμένει για το πότε θα μεταβεί στην κατάσταση elected. Πλεονέκτημα αυτού του τύπου χρονοδρομολόγησης είναι ότι με κατάλληλο χειρισμό των διεργασιών αποφεύγονται οι παραβιάσεις των χρονικών περιορισμών.

- *Non – preemptive*: Σε μια non-preemptive χρονοδρομολόγηση, οι διεργασίες από τη στιγμή που θα αρχίσουν την εκτέλεσή τους θα την ολοκληρώσουν οπωσδήποτε. Με αυτόν τον τρόπο όμως είναι πολύ πιθανό να το σύστημα να μην είναι σε θέση να τηρήσει τις χρονικές προθεσμίες.

- *Best effort – timing fault intolerance*: Η τεχνική αυτή χρησιμοποιείται στα soft συστήματα όταν η εφαρμογή επιτρέπει την ανοχή στην τήρηση των χρονικών περιορισμών.

- *Centralized – distributed* : η χρονοδρομολόγηση τέτοιου είδους χρησιμοποιείται όταν οι παράμετροι όλων των διεργασιών είναι εκ των προτέρων γνωστοί και είναι δυνατό να δομηθούν σε μια κατανεμημένη αρχιτεκτονική. Η τεχνική αυτή χρησιμοποιείται όταν η εφαρμογή αποτελείται από διάφορα επίπεδα και σε κάθε επίπεδο τηρείται μια «τοπική» λογική χρονοδρομολόγησης.

ΣΥΜΠΕΡΑΣΜΑΤΑ

Συνήθως οι περιοδικές διεργασίες έχουν αυστηρούς (hard) χρονικούς περιορισμούς και χρονοδρομολογούνται με έναν από τους τέσσερις αλγορίθμους που παρουσιάσθηκαν στην προηγούμενη ενότητα. Οι σποραδικές διεργασίες παρουσιάζουν και αυστηρούς και χαλαρούς (soft) χρονικούς περιορισμούς.

Ο βασικός αντικειμενικός σκοπός της χρονοδρομολόγησης σε αυτήν την περίπτωση είναι να εγγυηθεί την ορθή εκτέλεση όλων των περιοδικών διεργασιών. Όσο αφορά στις σποραδικές διεργασίες, η χρονοδρομολόγηση παρέχει ένα σχετικά καλό μέσο όρο απόκρισης για τις διεργασίες που έχουν soft περιορισμούς (best effort αλγόριθμοι), ενώ για τις διεργασίες που έχουν hard περιορισμούς, προσπαθεί μα μεγιστοποιήσει το βαθμό απόκρισής τους.

Η απλούστερη μέθοδος που χρησιμοποιείται για τη χρονοδρομολόγηση σποραδικών διεργασιών είναι αυτή της background χρονοδρομολόγησης, με μικρή όμως απόδοση. Ο μέσος χρόνος απόκρισης των σποραδικών διεργασιών μπορεί να βελτιωθεί με τη χρήση μιας διεργασίας που ονομάζεται εξυπηρετητής (server) διεργασιών. Τέλος, ο slack stealing αλγόριθμος παρέχει ουσιώδεις βελτιώσεις στο χρόνο απόκρισης των σποραδικών διεργασιών «κλέβοντας» χρόνο από τις περιοδικές διεργασίες

Κυρίαρχος παράγοντας με τον οποίο αξιολογείται μια εφαρμογή πραγματικού χρόνου είναι ο χρόνος. Οι απαιτήσεις της εκάστοτε εφαρμογής σε αυτόν το βασικό παράγοντα, είναι αυτές που θα οδηγήσουν στο διαχωρισμό σε συστήματα αυστηρά πραγματικού χρόνου (hard real-time systems) ή σε ήπια συστήματα πραγματικού χρόνου (soft real-time systems).

Μια απλή αποτυχία στους χρονικούς περιορισμούς που υποδεικνύει μια hard real – time εφαρμογή, μπορεί να οδηγήσει σε μια οικονομική ή οικολογική ή ανθρώπινη τραγωδία. Αυτή η αποτυχία στους χρονικούς περιορισμούς μπορεί να συνίσταται σε ένα καθυστερημένο μήνυμα ή σε μια λάθος μέτρηση ή σε μια χρονική προθεσμία που για κάποιο λόγο δεν τηρήθηκε.

Οι διεργασίες πραγματικού χρόνου είναι οι βασικές οντότητες οι οποίες χρονοδρομολογούνται. Οι διεργασίες είναι είτε περιοδικές είτε σποραδικές. Μια εφαρμογή πραγματικού χρόνου απαιτεί και τους δυο τύπους διεργασιών. Οι περιοδικές διεργασίες έχουν συνήθως hard περιορισμούς, ενώ οι σποραδικές έχουν και soft και hard περιορισμούς.

Κύριο μέλημα του σχεδιαστή ενός συστήματος πραγματικού χρόνου είναι η προσεκτική επιλογή των αλγορίθμων χρονοδρομολόγησης, έτσι ώστε να ικανοποιούνται όλες οι χρονικές προθεσμίες. Πολύ σημαντικό στοιχείο για ένα σύστημα χρονοδρομολόγησης να έχει τη δυνατότητα να επιλύει άμεσα προβλήματα αδιεξόδων και φαινόμενα αντιστροφής προτεραιότητας ειδικά σε hard RTS.

Εάν το σύστημα δεν έχει σχεδιαστεί να ανταπεξέρχεται σε υπερφορτώσεις, το αποτέλεσμα μπορεί να είναι καταστροφικό και ορισμένες υψίστης σημασίας διεργασίες της εφαρμογής μπορεί να χάσουν τις προθεσμίες τους.

Η χρονοδρομολόγηση σε περιβάλλον πολυεπεξεργαστών, παρουσιάζει δυσχέρειες και ανωμαλίες. Πολλά προβλήματα παραμένουν και πρέπει να βρεθεί λύση για αυτά. Καινούριες έρευνες σε άλλα πεδία των επιστημών, όπως η ασαφής λογική (fuzzy logic) ή τα νευρωνικά δίκτυα, ίσως είναι σε θέση να δώσουν λύση σε πολλά από τα υπόψη προβλήματα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Yang, X. S. (2010). Nature-inspired metaheuristic algorithms. Luniver press.

Dorigo, M., & Stützle, T. (2003). The ant colony optimization metaheuristic: Algorithms, applications, and advances. In Handbook of metaheuristics (pp. 250-285). Springer US.

Hosny, M. I., & Mumford, C. L. (2010). The single vehicle pickup and delivery problem with time windows: intelligent operators for heuristic and metaheuristic algorithms. Journal of Heuristics, 16(3), 417-439.

Lodi, A., Martello, S., & Vigo, D. (1999). Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. INFORMS Journal on Computing, 11(4), 345-357.

Jin, Z., Yang, Z., & Ito, T. (2006). Metaheuristic algorithms for the multistage hybrid flowshop scheduling problem. International Journal of Production Economics, 100(2), 322-334.

Kolisch, R., & Hartmann, S. (1999). Heuristic algorithms for the resource-constrained project scheduling problem: Classification and computational analysis. In Project scheduling (pp. 147-178). Springer US.

Puchinger, J., & Raidl, G. R. (2005, June). Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. In International Work-Conference on the Interplay Between Natural and Artificial Computation (pp. 41-53). Springer Berlin Heidelberg.

Yagiura, M., & Ibaraki, T. (2001). On metaheuristic algorithms for combinatorial optimization problems. Systems and Computers in Japan, 32(3), 33-55.

Glover, F. W., & Kochenberger, G. A. (Eds.). (2006). Handbook of metaheuristics (Vol. 57). Springer Science & Business Media.

Sayadi, M., Ramezanian, R., & Ghaffari-Nasab, N. (2010). A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. International Journal of Industrial Engineering Computations, 1(1), 1-10.

Haddad, O. B., Afshar, A., & Mariño, M. A. (2006). Honey-bees mating optimization (HBMO) algorithm: a new heuristic approach for water resources optimization. water resources management, 20(5), 661-680.

Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys (CSUR), 35(3), 268-308.

Yang, X. S. (2011). Review of meta-heuristics and generalised evolutionary walk algorithm. International Journal of Bio-Inspired Computation, 3(2), 77-84.

Tarantilis, C. D., Ioannou, G., Kiranoudis, C. T., & Prastacos, G. P. (2005). Solving the open vehicle routeing problem via a single parameter metaheuristic algorithm. Journal of the Operational Research Society, 56(5), 588-596.

Wiering, M. (2000, June). Multi-agent reinforcement learning for traffic light control. In ICML (pp. 1151-1158).

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., ... & Sokolsky, M. (2011, June). Towards fully autonomous driving: Systems and algorithms. In Intelligent Vehicles Symposium (IV), 2011 IEEE (pp. 163-168). IEEE.

Prashanth, L. A., & Bhatnagar, S. (2011). Reinforcement learning with function approximation for traffic signal control. IEEE Transactions on Intelligent Transportation Systems, 12(2), 412-421.

Wiering, M., Van Veenen, J., Vreeken, J., & Koopman, A. (2004). Intelligent traffic light control. Institute of Information and Computing Sciences. Utrecht University.

Dunne, M. C., & Potts, R. B. (1964). Algorithm for traffic control. Operations Research, 12(6), 870-881.

Sánchez-Medina, J. J., Galán-Moreno, M. J., & Rubio-Royo, E. (2010). Traffic signal optimization in “La Almozara” district in Saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing. IEEE Transactions on Intelligent Transportation Systems, 11(1), 132-141.

Singh, L., Tripathi, S., & Arora, H. (2009). Time optimization for traffic signal control using genetic algorithm. International Journal of Recent Trends in Engineering, 2(2), 4-6.