Οικονομικό Πανεπιστήμιο Αθηνών, Τμήμα Πληροφορικής
Μάθημα: Τεχνητή Νοημοσύνη
Ακαδημαϊκό έτος: 2024–25
Διδάσκων: Ι. Ανδρουτσόπουλος

1η Εργασία

Μέρος Α' (Επίλυση προβλημάτων μέσω αναζήτησης): Αναπτύξτε λογισμικό σε Java, C++, Python (ή άλλη γλώσσα που θα σας επιτρέψουν οι βοηθοί των φροντιστηρίων) που να λύνει **ένα από τα ακόλουθα τρία προβλήματα** (πρέπει να επιλέξετε ακριβώς ένα πρόβλημα).

α) Το πρόβλημα των κανιβάλων και ιεραποστόλων[1] χρησιμοποιώντας τον αλγόριθμο A* με κλειστό σύνολο. Μπορείτε να χρησιμοποιήσετε ευρετικές συναρτήσεις παρόμοιες εκείνων της λύσης της άσκησης μελέτης 4.3 ή άλλες δικές σας (εξηγώντας γιατί τις επιλέξατε στο έγγραφο που θα υποβάλετε). Θεωρήστε ότι στην αρχική κατάσταση έχουμε N ιεραποστόλους στη μία όχθη και τον ίδιο αριθμό (N) κανιβάλων στην ίδια όχθη αλλά το N είναι παράμετρος που ορίζεται κατά την κλήση του προγράμματος. Η μέγιστη χωρητικότητα (σε άτομα) της βάρκας είναι M, όπου το M επίσης ορίζεται κατά την κλήση του προγράμματος. Κατά την έναρξη εκτέλεσης του λογισμικού ορίζεται, επίσης, ο μέγιστος επιτρεπόμενος αριθμός διασχίσεων του ποταμού, έστω K. Το λογισμικό σας θα πρέπει να βρίσκει τη βέλτιστη λύση που δεν υπερβαίνει τις K διασχίσεις, αν υπάρχει τέτοια λύση. Μπορείτε να κάνετε δοκιμές με σχετικά μικρές τιμές των N, M (π.χ. 3, 4) και K (π.χ. ≤ 100). Θα πρέπει να αναφέρετε στο έγγραφο που θα υποβάλετε πόσο περίπου χρόνο (ανάλογα και με τον υπολογιστή που χρησιμοποιείτε) χρειάζεται το λογισμικό σας για να βρει λύση για τις ενδεικτικές τιμές των N, M, K που δοκιμάσατε, καθώς και τις αντίστοιχες λύσεις που βρίσκει. Όταν M > 2 και υπάρχει τουλάχιστον ένας ιεραπόστολος στη βάρκα, θεωρήστε ότι πρέπει και στο εσωτερικό της βάρκας ο αριθμός των κανιβάλων να μην υπερβαίνει τον αριθμό των ιεραποστόλων.

β) Πρόγραμμα μαθημάτων. Ζητείται να αναπτύξετε λογισμικό που να κατασκευάζει το ωρολόγιο πρόγραμμα μαθημάτων ενός γυμνασίου χρησιμοποιώντας μεθόδους αναζήτησης σε χώρους καταστάσεων που διδάσκονται στο μάθημα (π.χ. γενετικούς αλγορίθμους, simulated annealing). Θεωρήστε ότι κάθε μία από τις τρεις τάξεις έχει τρία τμήματα (Α1, Α2, Α3, Β1, Β2, Β3, Γ1, Γ2, Γ3). Το λογισμικό σας θα πρέπει να δέχεται ως είσοδο τα ακόλουθα αρχεία. Μπορείτε να επιλέξετε ελεύθερα τη μορφή των αρχείων (π.χ. αρχεία απλού κειμένου ή XML), αρκεί να είναι εύκολη η κατανόηση των περιεχομένων τους.

- Ένα αρχείο με όνομα lessons που να παραθέτει τους κωδικούς και τα ονόματα των μαθημάτων, την τάξη (Α, Β, Γ) στην οποία διδάσκεται το κάθε μάθημα και πόσες ώρες ανά εβδομάδα πρέπει να διδάσκεται (σε κάθε τμήμα της τάξης).

[1] Βλ. και https://en.wikipedia.org/wiki/Missionaries_and_cannibals_problem. Ο διδάσκων θα εκτιμούσε προτάσεις σας για καλύτερο όνομα και αφήγημα του παιχνιδιού, που θα διατηρούν τα μαθηματικά του χαρακτηριστικά, χωρίς να υπονοούν βία ή ρατσιστικές αντιλήψεις.

- Ένα αρχείο με όνομα teachers που να παραθέτει τους κωδικούς και τα ονόματα των καθηγητών, τους κωδικούς των μαθημάτων που μπορεί να διδάξει ο καθένας και το μέγιστο αριθμό ωρών ανά ημέρα και εβδομάδα που μπορεί να διδάξει (π.χ. λόγω διοικητικών καθηκόντων).

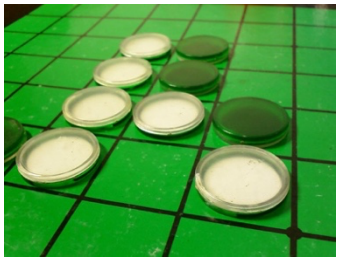
Θεωρήστε ότι κάθε εβδομάδα έχει 5 διδακτικές ημέρες, κάθε μία με το πολύ 7 διδακτικές ώρες. Το λογισμικό θα πρέπει να αναζητεί ωρολόγιο πρόγραμμα που να ικανοποιεί τους περιορισμούς των αρχείων lessons και teachers, καθώς και άλλους προφανείς περιορισμούς, όπως π.χ. ότι δεν επιτρέπεται ένας καθηγητής να διδάσκει την ίδια ώρα σε δύο διαφορετικές τάξεις. Επιπλέον να ικανοποιούνται κατά το δυνατόν οι παρακάτω περιορισμοί:

1. Να μην υπάρχουν κενά (π.χ. τη Δευτέρα μάθημα 8:00-11:00, κενό 11:00-13:00, μάθημα 12:00-15:00) στο πρόγραμμα κανενός τμήματος.
2. Να μη διδάσκει κανένας καθηγητής περισσότερες από δύο συνεχόμενες ώρες (π.χ. αν κάποιος καθηγητής έχει μάθημα τρεις ώρες κάποια μέρα, να έχει τουλάχιστον μία κενή ώρα μεταξύ των πρώτων δύο ή μεταξύ των δύο τελευταίων ωρών).
3. Ο ημερήσιος αριθμός ωρών διδασκαλίας κάθε τμήματος να είναι κατά το δυνατόν ομοιόμορφος όλες τις ημέρες (π.χ. όχι 4ωρο τη Δευτέρα, 7ωρο την Τρίτη, 3ωρο την Τετάρτη, 7ωρο Πέμπτη, 4ωρο Παρασκευή).
4. Οι ώρες διδασκαλίας κάθε μαθήματος σε ένα τμήμα να είναι κατά το δυνατόν ομοιόμορφα κατανεμημένες σε όλες τις ημέρες της εβδομάδας (π.χ. να μη διδάσκονται όλες οι ώρες του μαθήματος την ίδια ημέρα).
5. Ο αριθμός ωρών διδασκαλίας ανά εβδομάδα να είναι κατά το δυνατόν ομοιόμορφος για όλους τους καθηγητές (π.χ. να μη διδάσκει ένας 25 ώρες την εβδομάδα και άλλος μόνο 5).

Μπορείτε να προσθέσετε δικούς σας περιορισμούς ή/και να τους δώσετε βάρη (προτεραιότητες). Μπορείτε να προσθέσετε επιπλέον δυνατότητες στο λογισμικό σας (π.χ. τη δυνατότητα να χειρίζεται διαφορετικό αριθμό τμημάτων ανά τάξη). Θα προσμετρηθεί θετικά η δοκιμή του λογισμικού σας με κατά το δυνατόν ρεαλιστικά δεδομένα γυμνασίων.

Το λογισμικό πρέπει να παράγει ένα αρχείο schedule που να παριστάνει το εβδομαδιαίο ωρολόγιο πρόγραμμα όλου του γυμνασίου. Μπορείτε να διαλέξετε ελεύθερα τη μορφή αυτού του αρχείου, αρκεί να είναι εύκολα κατανοητά τα περιεχόμενά του.

γ) Reversi (ή την παραλλαγή του Othello).
Μπορείτε να βρείτε τους κανόνες του παιχνιδιού στην παρακάτω διεύθυνση:[2]



http://en.wikipedia.org/wiki/Reversi

Αναπτύξτε λογισμικό που θα επιτρέπει στον χρήστη να παίξει Reversi (ή Othello) με αντίπαλο τον υπολογιστή. Για την επιλογή των κινήσεων του υπολογιστή, το λογισμικό σας πρέπει να

[2] Η εικόνα της εκφώνησης προέρχεται επίσης από τη σελίδα https://en.wikipedia.org/wiki/Reversi.

χρησιμοποιεί τον αλγόριθμο MiniMax (6η διάλεξη), κατά προτίμηση με πριόνισμα α-β. Κατά την έναρξη του παιχνιδιού, ο χρήστης θα πρέπει να μπορεί να επιλέξει το μέγιστο βάθος αναζήτησης του αλγορίθμου MiniMax. Ο χρήστης θα πρέπει να μπορεί να επιλέξει, επίσης, αν θα παίξει πρώτος ή όχι. Το λογισμικό πρέπει να απορρίπτει κινήσεις που παραβιάζουν τους κανόνες του παιχνιδιού. Αν ο παίκτης του οποίου είναι η σειρά να παίξει δεν μπορεί να τοποθετήσει πουθενά νέο πούλι χωρίς να παραβιάσει τους κανόνες, το λογισμικό πρέπει να εμφανίζει αυτόματα σχετικό μήνυμα και να ζητά να παίξει ο άλλος παίκτης. Μετά από κάθε κίνηση, το λογισμικό θα πρέπει να δείχνει την κατάσταση του παιχνιδιού (π.χ. τυπώνοντας κενά, Χ και Ο). Δεν μας ενδιαφέρει σε αυτό το μάθημα η διεπαφή χρήστη (π.χ. μην αφιερώσετε χρόνο στην ανάπτυξη γραφικής διεπαφής). Θα πρέπει να περιλάβετε στο έγγραφο που θα παραδώσετε και ενδεικτικά παραδείγματα παρτίδων που έπαιξε το λογισμικό σας με αντίπαλο άνθρωπο ή προαιρετικά και με λογισμικό άλλων ομάδων. Εναλλακτικά μπορείτε να ασχοληθείτε με κάποιο άλλο παιχνίδι δύο αντιπάλων με ή χωρίς ζάρια (π.χ. τάβλι, σκάκι, όχι όμως π.χ. απλή τρίλιζα) που θα σας επιτρέψει ο βοηθός του φροντιστηρίου σας (και το οποίο θα περιγράφετε στο έγγραφο που θα παραδώσετε).

Μέρος Β' (Παράσταση γνώσεων και συλλογιστική): Ζητείται να υλοποιήσετε σε Java, C++, Python (ή άλλη γλώσσα που θα σας επιτρέψουν να χρησιμοποιήσετε οι βοηθοί των φροντιστηρίων) **δύο από τις ακόλουθες μεθόδους εξαγωγής συμπερασμάτων** (πρέπει να επιλέξετε ακριβώς δύο). Ειδικά **όσοι επιτρέπεται να παραδώσετε την εργασία ατομικά** (δείτε το έγγραφο με τις γενικές οδηγίες των εργασιών), μπορείτε να υλοποιήσετε μόνο **μία μέθοδο**.

- **Εξαγωγή συμπερασμάτων προς τα εμπρός** (forward chaining) **για προτάσεις Horn** (ακριβέστερα, οριστικές προτάσεις) **προτασιακής λογικής** (9η διάλεξη). Θεωρήστε ότι ο προς απόδειξη τύπος είναι ένα μόνο σύμβολο χωρίς άρνηση (π.χ. P).
- **Εξαγωγή συμπερασμάτων προς τα εμπρός για προτάσεις Horn** (ακριβέστερα, οριστικές προτάσεις) **πρωτοβάθμιας κατηγορηματικής λογικής** (12η διάλεξη). Θεωρήστε ότι ο προς απόδειξη τύπος είναι ένας ατομικός τύπος χωρίς άρνηση (π.χ. Loves(John, x)).
- **Εξαγωγή συμπερασμάτων προς τα πίσω** (backward chaining) **για προτάσεις Horn** (ακριβέστερα, οριστικές προτάσεις) **πρωτοβάθμιας κατηγορηματικής λογικής** (12η διάλεξη). Θεωρήστε ότι ο προς απόδειξη τύπος είναι ένας ατομικός τύπος χωρίς άρνηση.

Το λογισμικό που θα υλοποιήσετε σας θα πρέπει να διαβάζει ένα αρχείο το οποίο θα περιέχει τους τύπους (της αντίστοιχης μορφής λογικής) που θα αποτελούν τη βάση γνώσης. Το αρχείο αυτό θα πρέπει να είναι εύκολο να τροποποιηθεί με έναν επεξεργαστή κειμένου. Το λογισμικό σας θα πρέπει να δέχεται επίσης ως είσοδο (π.χ. από το πληκτρολόγιο ή από άλλο αρχείο) τον προς απόδειξη τύπο και θα πρέπει να αποκρίνεται *αληθές* ή *ψευδές*, ανάλογα με το αν κατάφερε ή όχι να τον αποδείξει. Προαιρετικά, μπορείτε να φροντίσετε το λογισμικό σας να επιστρέφει και άλλες πληροφορίες (π.χ. δέντρο απόδειξης, τιμές των μεταβλητών του προς απόδειξη τύπου κ.λπ.).

Ζητείται επίσης να επιδείξετε (στο έγγραφο που θα υποβάλετε και στην προφορική εξέταση) τις δυνατότητες του λογισμικού σας με παραδείγματα βάσεων γνώσεων και συμπερασμάτων (π.χ. παραδείγματα των διαφανειών).

Η **προθεσμία παράδοσης** της εργασίας θα ανακοινωθεί στο e-class.

Περαιτέρω διευκρινίσεις θα δοθούν από τους υπευθύνους των φροντιστηρίων ή/και μέσω e-class.

Διαβάστε προσεκτικά και το έγγραφο με τις γενικές οδηγίες των εργασιών του μαθήματος (βλ. έγγραφα μαθήματος στο e-class). Θα πρέπει να υποβάλετε (μέσω e-class) **ένα μόνο συμπιεσμένο αρχείο και για τα δύο μέρη της εργασίας μαζί**. Το συμπιεσμένο αρχείο θα πρέπει να περιλαμβάνει τον **πηγαίο κώδικα και των δύο μερών** της εργασίας, ένα **ξεχωριστό README.txt για κάθε μέρος** και **ξεχωριστό έγγραφο PDF** (το πολύ 10 σελίδων το κάθε έγγραφο) **για κάθε μέρος της εργασίας**.