Εργασία hw3, Τεχνητή Νοημοσύνη Ι, 5ο εξάμηνο
Ονοματεπώνυμο: TSVETOMIR IVANOV
AM: sdi1115201900066

Άσκηση 1.

README:

1. Μοντελοποίηση:

Για το CSP που μας δίνεται ορίζουμε τα εξής:
- Variables: Μαθήματα : Ενα variable ειναι ενα tuple με τα εξής πεδία:
(Semester, Course, Teacher, Difficulty, With_Lab), που αντιστοιχούν στο εξάμηνο, το ονομα του μαθήματος, ο καθηγητης, η δυσκολία και αν εχει εργαστήριο

- Domains: Οι τιμές για όλα τα μαθήματα ειναι ενα tuple (Day, Slot). Οι δυνατοί συνδυασμοί δηλαδή ειναι: (1, 1), (1,2), (1,3) … (21, 1), (21, 2), (21, 3) , σύμφωνα με την εκφώνηση αφού οι εξετάσεις διαρκούν 21 μέρες και υπάρχουν 3 διαθέσιμα slot (ωρες) όπου εξετάζονται τα μαθήματα

- Constraints: Οι περιορισμοί δίνονται από την var_constraints συνάρτηση, η οποία για δεδομένα δυο μαθήματα A, B στα οποια εχουν ανατεθεί δυο τιμές a, b ελέγχει αν ικανοποιείται ή όχι ο κάθε περιορισμός επιστρέφοντας True ή False

- Neighbours: Για κάθε μάθημα ορίζονται ως γείτονες τού, ολα τα αλλα μαθήματα αφού ολα συμμετέχουν στον γενικό περιορισμό, ο οποίος είναι να μην υπάρχουν δυο μαθήματα που εξετάζονται την ίδια μέρα και ίδια ώρα.

2. Τεχνικές Λεπτομέρειες:

Εντολή εκτέλεσης προγράμματος: python main.py
Η εργασία έγινε σε περιβάλλον Conda Enviroment με Python Version 3.8.12

Το αρχείο table.csv είναι τα δεδομένα που δίνονται από την εκφώνηση. Στο αρχείο main.py γίνεται η κύρια υλοποίηση του προβλήματος, και συμπεριλαμβάνεται μια main η οποία εκτελεί τα ζητούμενα.
ΔΕΝ δίνονται ορίσματα γραμμής εντολών, η main εκτελεί όλους τους δυνατούς συνδυασμούς των αλγορίθμων και εμφανίζει για κάθε έναν τα αποτελέσματα σε μορφή
Αλγόριθμος, Χρόνος, Αριθμός Αναθέσεων, Αριθμός ελέγχων

3. Υλοποίηση dom/wdeg:
Για την υλοποίηση του ευρετικού κανόνα δυναμικής διάταξης μεταβλητών dom/wdeg που περιγράφεται απο το paper http://www.frontiersinai.com/ecai/ecai2004/ecai04/pdf/p0146.pdf στην παράγραφο 3.3 :

- προσθέτουμε στην parent class CSP το field, self.weights.
Ενα λεξικό(dictionary): (key = tuple(A,B), value = weight) στο οποίο για συνδυασμό μεταβλητών (ζευγάρι A,B) αποθηκεύεται το βάρος του περιορισμού, το οποιο αρχικοποείται στο 1 (counter weight).

- ***Τροποποίηση/Προσθήκη συναρτήσεων στο αρχείο csp.py:***

1) revise(): Στην revise() προστίθεται ο έλεγχος για το domain wipe out για την τρέχουσα μεταβλητή , οπότε αυξάνουμε το βάρος για τον περιορισμο στο οποιο συμμετέχει η μεταβλητή.
2) forward_checking(): Επειδή η revise() χρησιμοποιείται μονο απο τον MAC αλγόριθμο, προσθέτουμε τον ίδιο έλεγχο στην forward_checking.
3) dom_wdeg(): Η συνάρτηση η οποία υλοποιεί τον κάνονα dom/wdeg. Για κάθε Variable η οποια συμμετεχει σε περιορισμο με μια μεταβλητή στην οποια δεν εχει ανατεθεί τιμή, υπολογίζεται το άθροισμα των weigths και τέλος επιλέγεται η μεταβλητή με το μικρότερο λόγο αριθμός_διαθέσιμων_τιμων/άθροισμα.
4) Για την επιστροφή του αριθμού ελέγχων για κάθε αλγόριθμο προστίθενται κάποιες εντολές στις συναρτήσεις backtrackig(), revise(), forward_checking().

Σε περίπτωση που δεν υπάρχει πλήθος διαθέσιμων τιμών επιστρέφεται η πρώτη μεταβλητή που στην οποία δεν έχει ανατεθεί κάποια τιμή.

** Επίσης στην κλήση του Mac αλγορίθμου αντί για την constraitt propagation AC3b, δίνεται ως όρισμα η AC3 η οποία καλεί την revise. Η AC3b δεν εχει τροποποιηθεί κατάλληλα για την dom/wdeg(δεν μπορούσα να καταλάβω ποτε πρεπει να γίνεται ο ελεγχος για το Domain wipe out) παρόλο που εβαλα εκτυπωση δεν φαινόταν να ικανοποιειται πουθενα η συνθηκη ωστε να αυξήσω τον counter weight.

Σύγκριση αλγορίθμων, Αποτελέσματα:

	MAC + MRV	MAC + dom/wdeg	FC + MRV	FC + dom/wdeg	MIN-CONFLICTS	SIMPLE-BACKTRACKING
TIME	0.55675	0.54792	0.04626	0.03606	0.05190	0.01765
ASSIGNMENTS	38	38	38	38	45	38
CHECKS	805781	800181	31160	31164	-	-

Μετρικές: Χρόνος, αριθμός αναθέσεων, αριθμός ελέγχων
Συμπεράσματα:

Με αυτές τις μετρικές δεν είναι ξεκάθαρο ποιος αλγόριθμος είναι αποδοτικότερος. Σε κάποιες περιπτώσεις φαίνεται ο MAC και FC να υστερούν στον χρόνο εκτέλεσης έναντι του MIN-CONFLICTS ακόμα και του SIMPLE BT.

Οσον αφορά την σύγκριση mrv, dom/wdeg στο συγκεκριμένο στυγμιότυπο πίνακα φαίνεται μικρή η διαφορά του χρόνου εκτέλεσης τους. Ο dom/wdeg εκτελείται πιο γρήγορα.

Ο αριθμός των τελικών αναθέσεων είναι ίδιος για όλους τους αλγορίθμους εκτός από τον MIN-CONFLICTS ο οποιες σε καποιες περιπτώσεις μπορεί να αυξηθεί σημαντικά και να καταστήσει τον αλγόριθμο πιο αργό. Αυτο συμβαίνει επειδή έχουμε ως περιορισμό για τα μαθήματα με εργαστήριο, να επιστρέφουμε False αν δει ότι πάει να δοκιμάσει να βάλει άλλο μάθημα αμέσως μετά από αυτά. Αυτός ο

περιορισμός είναι γρήγορος και καλύπτει επαρκώς το πρόβλημα για αυτό βγάζει πάντα ίδιο αριθμό assignments.

Τέλος αν πάρουμε κριτήριο τον αριθμό των ελέγχων φαίνεται ότι ο FC είναι αποδοτικότερος από τον MAC γιατί κάνει πολύ λιγότερους αριθμούς ελέγχων 31000 έναντι 700000+.

Πιθανόν, για το συγκεκριμένο στιγμιότυπο δεδομένων (το οποίο δεν έχει μεγάλη πολυπλοκότητα), να αρκεί ο απλός αλγόριθμος backtracking ο οποίος δεν χρησιμοποιεί άλλους πόρους (για την διάταξη μεταβλητών και τιμών) και λύνει “εύκολα και γρήγορα” το πρόβλημα, γι’αυτο και φαίνεται να έχει τον μικρότερο χρόνο εκτέλεσης.

Ελάχιστος Χρόνος Διαρκείας Εξετάσεων:
Παρατηρούμε ότι έχουμε περιορισμό να μην εξετάζονται δυο μαθήματα του ίδιου εξαμήνου την ίδια μέρα, και εφόσον το 7ο εξάμηνο έχει 16 μαθήματα οπότε χρειαζόμαστε τουλάχιστον 16 μέρες.

Σχόλια για το πρόγραμμα:
-Παρατηρούμε οτι αν τρέξουμε το πρόγραμμα με μικρότερο πεδίο τιμών, δηλαδή λιγότερες μέρες (π.χ 20 ή 18), το πρόγραμμα αργεί πολύ να βγάλει αποτέλεσμα στους απλούς αλγορίθμους όπως simple BT, fc, Mac με mrv.
-Αυτο συμβαίνει επειδή γίνεται πολύ γρήγορα domain wipe out λόγω των δύσκολων περιορισμών (ειδικότερα για τα μαθήματα που ειναι και δύσκολα και έχουν εργαστήριο).
-Αν τρέξουμε το πρόγραμμα μόνο για Mac+dom/wdeg ή και FC+dom/wdeg παρατηρούμε οτι (για 20 μέρες) βγάζει άμεσα λύση. Για κάτω από 20 μέρες πάλι αργεί.