~~Σύνοψη~~ πρόβλημα 9

Ο χώρος μας είναι ένας πίνακας $A^{400\times 300}$

Οι μεταβλητές μας είναι το γραφείο, το κρεβάτι, η καρέκλα γραφείου και ο καναπές

Στην περίπτωση μας κάθε μεταβλητή παίρνει ως τιμή ένα καρτεσιανό γινόμενο $[x,y]\times[z,w]$ όπου $x,y\in[0,300]$ και $[z,w]\in[0,400)$ και επίσης ισχύει για τις τιμές αυτές πως:

$(|x-y|=$ πλάτος και $|z-w|=$ μήκος$)$ ή $(|x-y|=$ μήκος και $|z-w|=$ πλάτος$)$

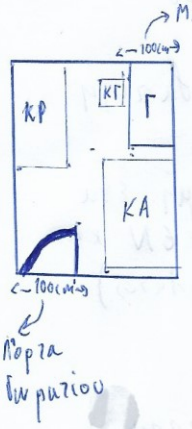
Το πεδίο τιμών είναι ίδιο για όλες τις μεταβλητές. Είναι όλα τα καρτεσιανά γινόμενα όπως τα αναφέραμε πριν όμως ισχύει επιπλέον πως λόγω της πόρτας $[x,y]\subseteq[101,300]$ και $[z,w]\subseteq[0,399]$

Στην συνέχεια βλέπουμε τα επίπεδα σαν μικρούς πίνακες οι οποίοι περιέχουν ως στοιχεία όλα τα στοιχεία του μεγάλου πίνακα που ο αριθμός της γραμμής τους ανήκει στο $[x,y]$ και ο αριθμός της στήλης τους στο $[z,w]$.

Οι περιορισμοί μας είναι πως γραφείο ≠ κρεβάτι ≠ καρέκλα γραφείου ≠ καναπές και έστω πως επιλέγω τυχαία δυο από τις παραπάνω μεταβλητές και τις ονομάζω Var1, Var2. Έστω επίσης πως η μεταβλητή Var1 παίρνει τιμή $[x,y]\times[z,w]$ και η Var2 παίρνει τιμή $[a,\beta]\times[\gamma,\delta]$ από το πεδίο τιμών τους. Τότε για να ισχύει πως Var1 ≠ Var2 πρέπει να ισχύει υποχρεωτικά πως

$\Big(([\alpha-1,\beta+1]\cap[x,y]=\emptyset$ και $[\gamma-1,\delta+1]\cap[z,w]=\emptyset)$ ή

$([\alpha,\beta]\cap[x-1,y-1]=\emptyset$ και $[\gamma,\delta]\cap[z-1,w+1]=\emptyset)\Big)$



Γ: γραφείο
ΚΑ: καναπές
ΚΡ: κρεβάτι
ΚΓ: καρέκλα γραφείου

Λύση του προβλήματος 9

Πρόβλημα 3

Οι μεταβλητές μας είναι οι $A1, A2, A3, A4, A5$

κάθε μεταβλητή έχει πεδίο τιμών το $\{9, 10, 11\}$

οι περιορισμοί είναι:

(1) $A_1 > A_3$

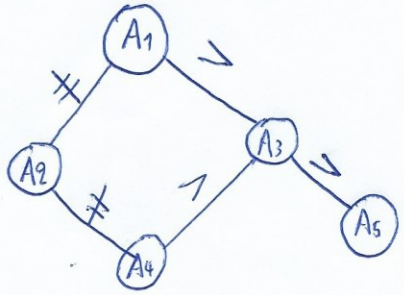
(2) $A_3 > A_5$

(3) $A_3 < A_4$

(4) $A_2 \neq A_1$

(5) $A_2 \neq A_4$

(6) $A_4 \neq 10:00$



Ξεκινάω με $A_1 = 9:00$ τοτε $D_2 = \{10, 11\}$ και $D_3 = \{\}$ αρα το A_1 θα αλλάξει τιμή

Έστω $A_1 = 10:00$ τοτε $D_2 = \{9, 11\}$, $D_3 = \{9\}$

(A_4, A_3) → μη συνεπής → $D_4 = \{\}$

Αρα το A_1 θα αλλάξει τιμή

Έστω $A_1 = 11:00$

Ξεκινάω με $A_1 = 9:00$ τοτε $D_2 = \{10:00, 11:00\}$ και $D_3 = \{\}$

Έστω $A_1 = 10:00$ τοτε $D_2 = \{9:00, 11:00\}$, $D_3 = \{9:00\}$

(A_4, A_3) → μη συνεπής → $D_4 = \{\}$

Έστω $A_1 = 11:00$ τοτε $D_2 = \{9:00, 10:00\}$, $D_3 = \{9:00, 10:00\}$

(A_4, A_3) → ασυνεπής → $D_4 = \{10:00, 11:00\}$ → πρέπει να εξετάσουμε και (A_2, A_4) και (A_3, A_4)

(A_5, A_3) → ασυνεπής → $D_5 = \{9:00\}$ → πρέπει να εξετάσουμε (A_5, A_3)

(A_4, A_2) → συνεπής

(A_2, A_4) → συνεπής

(A_3, A_4) → συνεπής

(A_5, A_3) → συνεπής

Έστω $A_2 = 9:00$ όλοι οι περιορισμοί ικανοποιούνται οπότε το κρατάμε

Έστω $A_3 = 9:00$ τότε $D_5 = \{\}$

Έστω $A_3 = 10:00$ τότε $D_4 = \{11:00\} \rightarrow$ ελέγχω $(A_3, A_4), (A_2, A_4)$

$(A_3, A_4) \rightarrow$ συνεπής

$(A_2, A_4) \rightarrow$ συνεπής

Τέλος $A_4 = 11:00$ ικανοποιεί όλους τους περιορισμούς και το κρατάμε και όμοια $A_5 = 9:00$

Όνομα: Δημήτριος
Επώνυμο:Φούντας
ΑΜ:1112201600236
ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ
PROJECT 3

Με σκοπό την πραγματοποίηση της εργασίας αυτής δημιούργησα δύο διαφορετικά αρχεία. Το problem1.py και το csp.py. Αρχικά θα σας αναλύσω την εργασία μου στο πρώτο αρχείο. Αποφάσισα πως η σωστότερη προσέγγιση για την επίλυση του προβλήματος ήταν να δημιουργήσω μια κλάση PROBLEM , η οποία είναι υποκλάση της CSP. Σε αυτήν αρχικοποίησα μια λίστα για τις μεταβλητές , και δύο άδεια λεξικά για να αποθηκεύσω το πεδίο τιμών της κάθε μετάβλητής και τους γείτωνες της κάθε μεταβλητής. Επίσης χρειάστηκε να αρχικοποιήσω δύο βοηθητικές μεταβλητές και δύο βοηθητικά λεξικά της κλάσης PROBLEM. Η μεταβλητή self.size μας βοηθάει λίγες γραμμές κώδικα πιο κάτω να εισάγουμε όλες τις μεταβλητές στην λίστα που αρχικοποιήσαμε για αυτές το λεξικο self.isindomain είναι δημιουργημένο ώστε να του δίνεις ως κλειδί μια μεταβλητή και να σου γυρνάει τον αριθμό του πεδίου τιμών της. Η μεταβλητή self.num_domains όπως λέει και το όνομα της αρχικοποιεί το πόσα διαφορετικά πεδία τιμών έχουμε για τις μεταβλητές μας.

Επόμενο στάδιο είναι να διαβάσουμε ένα ένα τα αρχεία για να πάρουμε τις πληροφορίες συνάρτηση dread διαβάζει dom αρχεία η pread διαβάζει var αρχεία και η pctr διαβάζει ctr αρχεία. Ας ξεκινήσουμε να εξηγούμε την πρώτη, αρχικά διαβάζουμε το αρχείο (readlines) , μία μία τις γραμμές η οποίες μπαίνουν σε λίστα. Η πρώτη μας μέριμνα είναι να διώξουμε τον ενοχλητικό χαρακτήρα αλλαγής γραμμής με χρήση της εντολής strip. Στην συνέχεια χρησιμοποιούμε την εντολή split για να ξεφορτωθούμε τα κενά μεταξύ τον λέξεων και να δημιουργήσουμε μια λίστα που να περιέχει καθαρά όλη την πληροφορία του text. Τέλος τρέχω κάθε στοιχείο της λίστας με σκοπό να δημιουργήσω το λεξικό self.isindomain που εξήγησα παραπάνω. Η αμέσως επόμενη συνάρτηση ακολουθεί την ίδια λογικη με την διαφορά ότι χρησιμοποιούμε την λογική μεταβλητή logic για να αποφύγω το πρώτο στοιχείο της λίστας καθώς δεν περιέχει χρήσιμη πληροφορία και τέλος αρχικοποιώ και τα domains χρησιμοποιώντας το λεξικό isindomain. Τέλος πάλι όμοια χρησιμοποιώ την ίδια λογική για να δημιουργήσω το λεξικό self.info το οποίο έχει ως μονάδικό σκοπό να αποθηκεύσει την πληροφορία που μας δίνει το ctr αρχείο και να την χρησιμοποιήσω στην συνάρτηση constraints η οποία παίρνει ως είσοδο δύο μεταβλητές και δύο τιμές αυτών , ελέγχει αν είναι γείτωνες και αν είναι ελέγχει αν οι τιμές τους ικανοποιούν τον ανάλογο περιορισμό. Τέλος οι συναρτήσεις getvar , getneig , getdomains έχουν ως σκοπό να βοηθήσουν τους διορθωτές να ελέγξουν αν τα στοιχεία εκχωρήθηκαν σωστά και αν η κλάση PROBLEM είναι καλά ορισμένη για τις ανάγκες του προβλήματος.

Προκειμένου να δουλέψουν καλά οι αλγόριθμοι που καλούμαστε να υλοποιήσουμε πρέπει να χρησιμοποιήσουμε ευρετική συνάρτηση. Για αυτό τον σκοπό αρχικοποιούμε ένα λεξικό στην κλάση PROBLEM που το ονομάζουμε self.weight, το λεξικό αυτό θα έχει κλειδια πλειάδες δύο στοιχείων που θα περιέχουν στην μία θέση μια μεταβλητή και στην άλλη έναν γείτωνα της και αντίστροφα για λόγους συμμετριας. Αρχικοποιούμε τέτοιου είδους πλειάδες μέσα σε αυτό το λεξικό , για κάθε μεταβλητή και για κάθε γείτωνα της και οι τιμές του λεξικού είναι όλες 1. Προκειμένου η ευρετική μας να δουλέψει για τον αλγόριθμο mac πρέπει να αλλάξουμε στο csp.py αρχείο το inference της συνάρτησης mac σε AC3 και την συνάρτηση revise έτσι ώστε κάθε φορά που μηδενίζεται το domain της μεταβλητής Xi να προστίθετε μια μονάδα στο βάρος του περιορισμού που προκάλεσε τον περιορισμό. Όμοια στην συνάρτηση forward_checking κάνω το ίδιο όταν μηδενίζεται το μήκος του πεδίου τιμών της var. Αφού καταφέρουμε να κάνουμε το βάρος τον περιορισμών να αυξάνεται κάθε φορα που ο περιορισμος αποτυγχάνει με τον τρ΄πο που είπαμε παραπάνω, είμαστε έτοιμοι να δημιουργήσουμε την ευρετική.

Στο αρχείο csp.py δημιούργησα την συνάρτηση dweg η οποία παίρνει μια μεταβλητή και αφου αρχικοποιήσει μια τιμή αθροίσματος .Για κάθε γείτωνα της ο οποίος δεν έχει πάρει ακόμα τιμή προσθέτει στην μεταβλητή του αθροίσματος το βάρος που έχουν οι περιορισμοί που τις περιλαμβάνουν. Τέλος εφαρμόζω την συνάρτηση αυτή μέσα στην συνάρτηση mrv έτσι ώστε να μήν επιλέγει πια την μεταβλητή με το μικρότερο πεδίο τιμών ,αλλά αυτή με τον μικρότερο λόγο του πλήθους στοιχείων πεδίου τιμών/ βάρος της μεταβλητής)

ΥΓ: Δυστυχώς ο αλγόριθμος μου δεν έδωσε αποτελέσματα τα οποία να μπορέσω να μετρήσω , έχει ακόμα κάποια bugs που δεν μου επέτρεψαν να συνεχίσω την εργασία. Αλλά θα σας παρακαλούσα πολύ να την διορθώσετε παρ αυτα καθώς η λογική της είναι σωστά και έχει γίνει αρκετή δουλειά που είναι κρίμα να μείνει απαρατήρητη . Ευχαριστώ πολύ.