Τεχνητή Νοημοσύνη 1 – Χειμερινό 2021-2022

Εργασία τρίτη

Κατσαούνη Σοφία Μερόπη , sdi1800070

Πρόβλημα 2:

α) Ορίζω το πρόβλημα ικανοποίησης περιορισμών ως εξής

Variables:

	Χρόνος μετάβασης από χρηματοκιβώτιο σε αίθουσα	**Χρόνος μετάβασης από αίθουσα σε χρηματοκιβώτιο**	**Χρόνος για παραβίαση χρηματοκιβώτιου**
Γιάννης	X1	X4	X7
Μαρία	X2	X5	X8
Όλγα	X3	X6	X9

Domains: σύνολο τιμών ώρας για να πάει από την αίθουσα στο χρηματοκιβώτιο [20-30] λεπτά, σύνολο τιμών ώρας για να γυρίσει από το χρηματοκιβώτιο στην αίθουσα [20-30] λεπτά, σύνολο τιμών ώρας για να κλέψει από το χρηματοκιβώτιο [45-90] λεπτά

Στην ουσία από αυτά τα διαστήματα θα γίνει ανάθεση μίας τιμής για κάθε μια από τις μεταβλητές.

Constraints:

1.χρόνος ολοκλήρωσης ομιλίας Γιάννη στις 9.30

2.χρόνος ολοκλήρωσης ομιλίας Μαρίας στις 10,

3. χρόνος ολοκλήρωσης ομιλίας Όλγας στις 10.30,

4.ληξη ομιλιών στις 11,

5. [20-30] λεπτά από χρηματοκιβώτιο σε αίθουσα,

6. [20-30] λεπτά από αίθουσα σε χρηματοκιβώτιο,

7. [45-90] λεπτά για παραβίαση χρηματοκιβώτιου

8. έναρξη συνομιλιών στις 9

β)

Ο Σιεσπής θα συλλάβει τον Γιάννη. Δεν υπάρχει ενδεχόμενο κάποιος να πήγε στο δωμάτιο του και να έκλεψε το έπαθλο επομένως κάποιος είπε ψέματα. Από το ερώτημα 1 βλέπουμε ότι εάν πάρουμε τις ελάχιστες τιμές από τα domain 20+20+45 = 85 λεπτά έτσι ώστε κάποιος να κλέψει το μήλο. Εάν ορίσουμε μια μεταβλητή Y που αναπαριστά τον χρόνο που απομένει στον κάθε ύποπτο ισχύει Y1=90 λεπτά εφόσον ο Γιάννης τελειώνει την ομιλία του στις 9:30, αντίστοιχα Y2=60 λεπτά για την Μαρία εφόσον τελειώνει στις 10:00, και Y3=30 λεπτά εφόσον η Όλγα τελειώνει στις 10:30 και εφόσον όλοι οι ύποπτοι παρακολούθησαν την απονομή βραβείου που γίνεται στις 11:00. Βλέπουμε ότι τα 85 λεπτά χωράνε μόνο στον ελεύθερο χρόνο του Γιάννη και άρα αυτός έκλεψε.

γ) Μια μέθοδος διάδοσης περιορισμών είναι η χρήση του αλγορίθμου Forward checking διότι στον συγκεκριμένο αλγόριθμο όταν μια τιμή εκχωρείται στην τρέχουσα μεταβλητή, οποιαδήποτε τιμή στο domain μιας "μελλοντικής" μεταβλητής που έρχεται σε διένεξη με αυτήν την εκχώρηση αφαιρείται (προσωρινά) από το domain. π.χ. εάν επιλέξει μια τιμή ανάμεσα στο διάστημα [20-30] για την μεταβλητή **Χρόνος μετάβασης από χρηματοκιβώτιο σε αίθουσα** τότε οι τιμές των μεταβλητών d1:**Χρόνος μετάβασης από αίθουσα σε χρηματοκιβώτιο** και d2:**Χρόνος για παραβίαση χρηματοκιβώτιου** θα έχουν την ιδιότητα d1+d2 < συνολικός ελεύθερος χρόνος υπόποτου - **Χρόνος μετάβασης από χρηματοκιβώτιο σε αίθουσα** και άρα οι πιθανές τιμές που λαμβάνουν οι d1 και d2 βρίσκονται αναγκαστικά σε ένα υποσύνολο των συνόλων [20-30] και [45-90].

Πρόβλημα 3:

1) Ορίζουμε το πρόβλημα ικανοποίησης περιορισμών ως εξής:

Variables: X1 = (εργασία 1, ενέργειες m), X2 = (εργασία 2, ενέργειες m), X… = (εργασία…, ενέργειες m), Xn = (εργασία n, ενέργειες m)

Domains: (αριθμός μηχανής που απασχολεί μια εργασία, χρόνος di για τον οποίο απασχολείται αυτήν η μηχανή)

Constraints:

1. Κάθε εργασία i πρέπει να τερματίζει πριν από την δοσμένη προθεσμία D>0, επομένως d1 + d2 + … + dm <= D
2. Κάθε μηχανή εκτελεί αποκλειστικά μία μόνο ενέργεια την φορά

3. Μία ενέργεια εκτελείται μόνο όταν εκτελεστούν όλες οι προηγούμενες από αυτήν την ενέργεια ενέργειες
4. Εφόσον μια ενέργεια ξεκινήσει δεν μπορεί να διακοπεί η λειτουργία της

2)

Εάν το συνολικό άθροισμα των χρόνων d για την εκτέλεση μίας ενέργειας είναι μικρότερο από την δοσμένη προθεσμία D δηλαδή αν $d1 + d2 + \ldots + dm \le D$, τότε μία λυση για n=3 και m=4 θα ήταν κάθε μία από τις 4 μηχανές να αναλάβει 1 εργασία.

Ένα μη συνεπές παράδειγμα προκύπτει εάν συμβαίνει το αντίθετο δηλαδή εάν $d1 + d2 + \ldots + dm > D$. Σε αυτήν την περίπτωση ο συνολικός χρόνος δεν θα μας φτάσει για να εκτελεστούν όλες οι ενέργειες όλων των εργασιών και άρα δεν θα υπάρχει λύση στο πρόβλημα.

3)

Ένας αλγόριθμος οπισθοδρόμησης που μπορεί να χρησιμοποιηθεί γι' αυτό το πρόβλημα είναι ο MAC (maintaining arc consistency). Η μεταβλητή Xi είναι arc-consistent σε σχέση με μια άλλη μεταβλητή Xj εάν και μόνο εάν για κάθε τιμή vi στο Di, υπάρχει μια τιμή vj σε Dj έτσι ώστε (vi, vj) ικανοποιεί τον περιορισμό (Xi, Xj).Αν η Xi δεν είναι συνεπής με τη μεταβλητή Xj, μπορούμε να τη κάνουμε συνεπής αφαιρώντας τιμές στο Di που δεν είναι συνεπείς με καμία τιμή στον Dj. Αυτή η αφαίρεση δεν μπορεί ποτέ να αποκλείσει οποιαδήποτε λύση.

Πρόβλημα 6:

Από τις διαφάνειες του μαθήματος http://cgi.di.uoa.gr/~ys02/dialekseis2020/propositional.pdf και συγκεκριμένα από το slide 25 προκύπτει ότι:

- (A) , η έκφραση είναι καλά ορισμένη πρόταση της προτασιακής λογικής εφόσον ανάγεται στην κατηγορία των ComplexSentence του slide.

- $(A \rightarrow B)$, η έκφραση **δεν** είναι καλά ορισμένη πρόταση της προτασιακής λογικής εφόσον το σύμβολο -> δεν υπάρχει στις καλά ορισμένες προτάσεις.(εκτός εάν εννοήθηκε αυτό το σύμβολο $\Rightarrow$ τότε σε αυτήν την περίπτωση ανήκει στις καλά ορισμένες προτάσεις)

- $A \equiv B$, η έκφραση **δεν** είναι καλά ορισμένη πρόταση της προτασιακής λογικής εφόσον το σύμβολο ανάμεσα στα προτασιακά σύμβολα δεν υπάρχει στις καλά ορισμένες προτάσεις.
- $A \models B$, η έκφραση **δεν** είναι καλά ορισμένη πρόταση της προτασιακής λογικής εφόσον το σύμβολο ανάμεσα στα προτασιακά σύμβολα δεν υπάρχει στις καλά ορισμένες προτάσεις.
- $(A \wedge 1)$,η έκφραση **δεν** είναι καλά ορισμένη πρόταση της προτασιακής λογικής εφόσον το σύμβολο 1 δεν υπάρχει στις καλά ορισμένες προτάσεις.

Πρόβλημα 4:

- $(A \wedge B \wedge C \Rightarrow D) \Leftrightarrow (A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$

A B C D	(A ∧ B)	(A ∧ B) ∧ C	$(A \wedge B \wedge C \Rightarrow D)$
1 1 1 1	1	1	1
1 1 1 0	1	1	0
1 1 0 1	1	0	1
1 1 0 0	1	0	1
1 0 1 1	0	0	1
1 0 1 0	0	0	1
1 0 0 1	0	0	1
1 0 0 0	0	0	1
0 1 1 1	0	0	1
0 1 1 0	0	0	1
0 1 0 1	0	0	1
0 1 0 0	0	0	1
0 0 1 1	0	0	1
0 0 1 0	0	0	1
0 0 0 1	0	0	1
0 0 0 0	0	0	1

A	B	C	D	$(C \Rightarrow D)$	$(B \Rightarrow (C \Rightarrow D))$	$(A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$
1	1	1	1	1	1	1
1	1	1	0	0	0	0
1	1	0	1	1	1	1
1	1	0	0	1	1	1
1	0	1	1	1	1	1
1	0	1	0	0	1	1
1	0	0	1	1	1	1
1	0	0	0	1	1	1
0	1	1	1	1	1	1
0	1	1	0	0	0	1
0	1	0	1	1	1	1
0	1	0	0	1	1	1
0	0	1	1	1	1	1
0	0	1	0	0	1	1
0	0	0	1	1	1	1
0	0	0	0	1	1	1

$(A \wedge B \wedge C \Rightarrow D)$	$(A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$	$(A \wedge B \wedge C \Rightarrow D) \Leftrightarrow (A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$
1	1	1
0	0	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	1	1

- $A \wedge (A \Rightarrow B) \wedge (A \Rightarrow \neg B)$

A	B	$(A \Rightarrow B)$	$A \wedge (A \Rightarrow B)$	$(A \Rightarrow \neg B)$	$A \wedge (A \Rightarrow B) \wedge (A \Rightarrow \neg B)$
1	1	1	1	0	0
1	0	0	0	1	0
0	1	1	0	1	0
0	0	1	0	1	0

- $(A \vee B) \wedge (\neg A \vee C) \wedge \neg B \wedge \neg C$

A B C	(A ∨ B)	(¬A ∨ C)	$(A \vee B) \wedge (\neg A \vee C)$	$(A \vee B) \wedge (\neg A \vee C) \wedge \neg B$	$(A \vee B) \wedge (\neg A \vee C) \wedge \neg B \wedge \neg C$
1 1 1	1	1	1	0	0
1 1 0	1	0	0	0	0
1 0 1	1	1	1	1	0
1 0 0	1	0	0	0	0
0 1 1	1	1	1	0	0
0 1 0	1	1	1	0	0
0 0 1	0	1	0	0	0
0 0 0	0	1	0	0	0
					0

- $(A \vee B) \wedge (\neg A \vee C) \wedge (B \vee C)$

A B C	(A ∨ B)	(¬A ∨ C)	$(A \vee B) \wedge (\neg A \vee C)$	(B ∨ C)	$(A \vee B) \wedge (\neg A \vee C) \wedge (B \vee C)$
1 1 1	1	1	1	1	1
1 1 0	1	0	0	1	0
1 0 1	1	1	1	1	1
1 0 0	1	0	0	0	0
0 1 1	1	1	1	1	1
0 1 0	1	1	1	1	1
0 0 1	0	1	0	1	0
0 0 0	0	1	0	0	0

Σύμφωνα με τους παραπάνω πίνακες τα ερωτήματα απαντώνται ως εξής:

1. Ποιες από τις προτάσεις είναι έγκυρες;

$$(A \wedge B \wedge C \Rightarrow D) \Leftrightarrow (A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$$

2. Ποιες από τις προτάσεις είναι ικανοποιήσιμες;

$(A \wedge B \wedge C \Rightarrow D) \Leftrightarrow (A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$ και

$(A \vee B) \wedge (\neg A \vee C) \wedge (B \vee C)$

3. Ποιες από τις προτάσεις είναι μη ικανοποιήσιμες;

$A \wedge (A \Rightarrow B) \wedge (A \Rightarrow \neg B)$ και $(A \vee B) \wedge (\neg A \vee C) \wedge \neg B \wedge \neg C$

4. Ποιες από τις προτάσεις έχουν τουλάχιστον ένα μοντέλο;

$(A \wedge B \wedge C \Rightarrow D) \Leftrightarrow (A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$ και

$(A \vee B) \wedge (\neg A \vee C) \wedge (B \vee C)$

5. Ποιες από τις προτάσεις είναι ταυτολογίες;

$$(A \wedge B \wedge C \Rightarrow D) \Leftrightarrow (A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$$

6. Ποιες από τις προτάσεις είναι σε μορφή Horn ή μπορούν να μετατραπούν σε ένα σύνολο φράσεων που είναι Horn;

Για τις προτάσεις $(A \vee B) \wedge (\neg A \vee C) \wedge \neg B \wedge \neg C$ και $(A \vee B) \wedge (\neg A \vee C) \wedge (B \vee C)$ εύκολα προκύπτει ότι δεν βρίσκονται σε μορφή Horn, οι άλλες 2 προτάσεις όμως χρειάζονται μετατροπή:

• $A \wedge (A \Rightarrow B) \wedge (A \Rightarrow \neg B) \equiv A \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B)$

και

• $(A \wedge B \wedge C \Rightarrow C) \Leftrightarrow (A \Rightarrow (B \Rightarrow (C \Rightarrow D)))$

Ονομάζοντας $(A \wedge B \wedge C \Rightarrow C) = \alpha$ και $(A \Rightarrow (B \Rightarrow (C \Rightarrow D))) = \beta$ έχουμε

$\alpha \Leftrightarrow \beta$ δηλαδή $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha) \equiv (\neg \alpha \vee \beta) \wedge (\neg \beta \vee \alpha)$

α: $\neg(A \wedge B \wedge C) \vee D \equiv \neg A \vee \neg B \vee \neg C \vee D$

$\neg\alpha$: $A \wedge B \wedge C \wedge \neg D$

β: $(A \Rightarrow (B \Rightarrow (C \Rightarrow D))) \equiv (A \Rightarrow (B \Rightarrow (\neg C \vee D))) \equiv$

$(A \Rightarrow (\neg B \vee (\neg C \vee D))) \equiv (A \Rightarrow (\neg B \vee \neg C \vee D)) \equiv$

$\neg A \vee (\neg B \vee \neg C \vee D) \equiv \neg A \vee \neg B \vee \neg C \vee D$

Παρατηρούμε ότι $\alpha \equiv \beta$

Επομένως $(\neg \alpha \vee \beta) \wedge (\neg \beta \vee \alpha) \equiv$

$\equiv (\neg \alpha \vee \alpha) \wedge (\neg \alpha \vee \alpha) \equiv \alpha \wedge \alpha \equiv \alpha \equiv \neg A \vee \neg B \vee \neg C \vee D$

Και άρα από τις τελικές μορφές παρατηρούμε ότι βρίσκονται σε μορφή Horn εφόσον υπάρχει το πολύ 1 θετικό λεκτικό σε κάθε φράση.

Πρόβλημα 5:

Μετατρέποντας την πρόταση $A \wedge (B \Leftrightarrow C)$ σε CNF προκύπτει:

$$A \wedge (B \leftrightarrow C) = A \wedge (B \rightarrow C) \wedge (C \rightarrow B) = A \wedge (\neg B \vee C) \wedge (\neg C \vee B)$$

Αντίστοιχα εφαρμόζοντας άρνηση πάνω στην πρόταση
$(A \wedge B) \Leftrightarrow (A \wedge C)$ προκύπτει:

$$\neg((A \wedge B) \leftrightarrow (A \wedge C)) = (A \wedge B) \leftrightarrow (\neg(A \wedge C)) = (A \wedge B) \leftrightarrow (\neg A \vee \neg C)) =$$

και στην συνέχεια μετατρέπουμε την πρόταση σε CNF:

$$((A \wedge B) \rightarrow (\neg A \vee \neg C)) \wedge ((\neg A \vee \neg C) \rightarrow (A \wedge B)) =$$

$$((\neg A \vee \neg B) \vee (\neg A \vee \neg C)) \wedge ((A \wedge C) \vee (A \wedge B)) =$$

$$(\neg A \vee \neg B \vee \neg C) \wedge (A \wedge (B \vee C))$$

Άρα:

A , $(\neg B \vee C)$, $(\neg C \vee B)$, $(\neg A \vee \neg B \vee \neg C)$, $(B \vee C)$

Από τον κανόνα της μοναδιαίας ανάλυσης μπορούμε να συμπεράνουμε την $\neg A$ από τα ,$(\neg A \vee \neg B \vee \neg C)$, $(B \vee C)$. Έτσι προκύπτει κενή φράση.

Επομένως η μια πρόταση καλύπτει λογικά την άλλη.

Πρόβλημα 1:

τρέχετε το πρόγραμμα με την εντολή:

python main.py <algorithm> <display>

Για το πεδίο ***<display>*** υπάρχουν 2 επιλογες, display ή mydisplay

το όρισμα display: καλεί την έτοιμη display του csp

το όρισμα mydisplay: καλεί μια συνάρτηση display που έφτιαξα εγώ για να φαίνεται το πρόγραμμα πιο "ανθρώπινο" και να ειναι γίνεται πιο εύκολα το error checking. Αν επιλέξετε αυτό το όρισμα τότε θα εμφανιστεί κάτι αυτού του στύλ στην οθόνη

```
(new_env) sofia@sofia-HP-Laptop-15-da0xxx:~/Desktop/h3$ python main.py bt mydisplay
                         9:00-12:00              12:00-3:00              3:00-6:00

___________________
DAY  1
                         Γραμμική Άλγεβρα   | Πιθανότητες και στοιχεί |  Αρχές Γλωσσών Προγραμμα

___________________
DAY  2
                         Διακριτά Μαθηματικά | Εργαστήριο Κυκλωμάτων κ |         Γραφικά Ι

___________________
DAY  3
                         Εισαγωγή στην Πληροφορι | Εφαρμοσμένα Μαθηματικά |  Αλγοριθμική Επιχειρησια

___________________
DAY  4
                         Λογική Σχεδίαση    |      Ανάλυση ΙΙ      |   Λειτουργικά Συστήματα

___________________
DAY  5
                         Εισαγωγή στον Προγραμμα |                      |  Σχεδίαση Ψηφιακών Συστη

___________________
DAY  6
                         Αντικειμενοστραφής Προγ |                      |  Τηλεπικοινωνιακά Δίκτυα

___________________
DAY  7
                         Σήματα και Συστήματα | Ψηφιακή Επεξεργασία Σήμ |  Προηγμένα Θέματα Αλγορί

___________________
DAY  8
                         Τεχνητή Νοημοσύνη (ΤΟ Κ |                      |  Ειδικά Θέματα Θεωρητική

___________________
DAY  9
                         Δίκτυα Επικοινωνιών ΙΙ | Αλγόριθμοι Βιοπληροφορι |
```

κλπ

Για το πεδίο ***<algorithm>*** υπάρχουν οι επιλογές:

mac+mrv --> *algorithm*: mac *heuristic*: mrv

fc+mrv --> *algorithm*: fc *heuristic*: mrv

mincon --> *algorithm*: min conflicts

bt --> *algorithm*: backtracking

bt+mrv --> *algorithm*: backtracking *heuristic*: mrv

fc+dom --> *algorithm*: fc *heuristic*: dom wdeg

mac+dom --> *algorithm*: mac *heuristic*: dom wdeg

bt+dom --> *algorithm*: backtracking *heuristic*: dom wdeg

Στον αλγόριθμο **mincon** δεν γίνεται επιλογή display ή mydisplay. Εμφανίζεται αποτέλεσμα εαν τρέξετε απλά **python main.py mincon**

Το πρόγραμμα δοκιμάστηκε σε *anaconda enviroment και Python 3.6.13*

Εξήγηση σχεδιαστικών επιλογών:

ακολούθησα πιστά τις προτάσεις των φροντιστηρίων σχετικά με την μορφή των συναρτησεων την δημιουργία κλάσης κλπ, για variables έχω μια λίστα με όλα τα μαθήματα, domain είναι ένα dictionary που για κάθε μάθημα αντιστοιχεί μια λίστα απο όλες τις πιθανές ώρες και μέρες που μπορεί να εξεταστεί το μάθημα. Για neighbors έχω ένα dictionary που για κάθε μάθημα αντιστοιχεί μια λίστα με όλα τα υπόλοιπα μαθήματα εκτός αυτού, επιλέγω αυτούς τούς γείτονες διότι υπάρχει μόνο μία αίθουσα και άρα αναγκαστικά όλα τα μαθήματα συμμετέχουν στα constraints με όλα τα υπόλοιπα. Ο κώδικα έχει και σχόλια για καλύτερη κατανόηση.

Ευρετική συνάρτηση dom_wdeg:

Ακολουθώντας τις οδηγίες του δεδομένου paper, φτιάχνω στην κλάση στο main.py μια λίστα απο όλα τα ζεύγη μαθήματος-γείτονα, τα οποία περνάω στην init του csp. Εκεί για κάθε ένα απο αυτά τα ζεύγη, δημιουργώ έναν weight counter τον οποιο θέτω σε 1.

One limitation of this approach is that no information about previous states of the search is exploited. We propose to capture such information by associating a counter, called *weight*, with any constraint of the problem. These counters will be updated during search whenever a dead-end (domain wipe-out) occurs. As systematic solvers such as FC or MAC involve successive revisions in order to remove values

It is important to note that this new heuristic is related to *ddeg* as only constraints involving a variable and at least another uninstantiated one are considered. In fact, setting all *weight* counters to 1 is equivalent to define *ddeg*. Then, in order to benefit, at the beginning of the search, from relevant information about current variable degrees, we propose to initialize all *weight* counters to 1. Finally,

Στην συνέχεια πάω στις συναρτήσεις revise του αλγορίθμου mac και forward_checking και κάθε φορά που συμβαίνει domain wipe-out αυξάνω τα weight +1.

filtering algorithms (see, e.g., [19]). At the end of the algorithm (lines 4 and 5), when a domain wipe-out occurs, the weight of the "revising" constraint is incremented.

Using these counters, it is possible to define a new variable or-

Και στην συνέχεια

$$\alpha_{wdeg}(X_i) = \sum_{C \in \mathscr{C}} weight[C] \mid vars(C) \ni X_i \wedge \mid FutVars(C) \mid > 1$$

where $FutVars(C)$ denotes the uninstantiated variables in $vars(C)$. Hence, the weighted degree of a variable X_i corresponds

Algorithm 1 revise(C : Constraint, X : Variable) : boolean

```
for each a ∈ dom(X) do
    if seekSupport(C, X, a) = false then
        remove a from dom(X)
if dom(X) = ∅ then
    weight[C]++
return Dom(X) ≠ ∅
```

to the sum of the weights of the constraints involving X_i and at least another uninstantiated variable. Intuitively, locally inconsistent or hard parts of CSPs should be first examined by the search algo-

Μετά φτίαχνω 2 συναρτήσεις τις dom_wdeg και domain_weight_degree_ratio.

dom_wdeg: Ακολουθεί πιστά την μορφή της ευρετικής mrv και για κάθε μια απο τις μεταβλητές που δεν υπάρχουν ήδη στο assignement καλεί την domain_weight_degree_ratio. Επιστρέφει μια μεταβλητή.

domain_weight_degree_ratio: Κρατάει το πλήθος των domain για την συγκεκριμένη μεταβλητή και για κάθε γείτονα της μεταβλητής που δεν είναι ήδη στο assignement αθροίζει συνολικά όλα τα weights των ζευγών μεταβλητη-γείτονα. Τελικά εφόσον μιλάμε για ratio επιστρέφεται η διαίρεση του πλήθους domain προς το συνολικό βάρος.

ning of the search, from relevant information about current variable degrees, we propose to initialize all $weight$ counters to 1. Finally, combining weighted degrees and domain sizes yields $dom/wdeg$, an heuristic that selects first the variable with the smallest ratio current domain size to current weighted degree. In the rest of the paper, $wdeg$ and $dom/wdeg$ will be called conflict-directed (variable or-

Είναι αξιόλογο οτι παρατηρήσα οτι όσες φορές και να τρέξεις την συγκεκριμένη ευρετική, τα weights δεν γίνονται update δηλαδή παραμένουν 1 από την αρχικοποίηση, αυτό σημαίνει οτι δεν συμβαίνει ποτέ domain wipe-out.

2)

Τρέχω κάθε αλγόριθμο 3 φορές για μια καλύτερη εκτίμηση των ικανοτήτων του καθενός.

Τα κριτήρια αξιολόγησης μου στο πινακακι είναι:

1) Χρόνος εκτέλεσης

2) Πλήθος κόμβων στο search tree

3) Πλήθος ελεγχών συνέπειας (περιορισμών)

Ο χρόνος είναι περισσότερο διακοσμητικός καθώς παρατήρησα οτι οι δοκιμές μου σε διαφορετικούς υπολογιστές καθώς και με προσθαφαίρεση άλλων εντολών υπάρχουν μικροδιαφορές. Το πλήθος κόμβων που εξετάζονται σε όλους τους αλγορίθμους με οποιοδήποτε συμδυασμό ευρετικής είναι σταθερά 38. Άρα τελικά το κριτήριο σύγκρισης που χρησιμοποίησα είναι το πλήθος ελέγχων συνέπειας.

Παραθέτω 3 δοκιμές απόδοσης για τον κάθε αλγόριθμο:

```
Running  mac+mrv  ...
                      time elapsed nodes expanded   constraint checks
                      2.15         38.00            898653
Running  mac+mrv  ...
                      time elapsed nodes expanded   constraint checks
                      2.15         38.00            893667
Running  mac+mrv  ...
                      time elapsed nodes expanded   constraint checks
                      2.19         38.00            892316
```

Όπως παρατηρούμε υπάρχει μια πολύ μικρή διαφοροποίηση στο πλήθος constraint checks. Κατά προσέγγιση λοιπόν constraint checks = 890000

```
Running  fc+mrv  ...
                      time elapsed nodes expanded   constraint checks
                      0.11         38.00            51242
Running  fc+mrv  ...
                      time elapsed nodes expanded   constraint checks
                      0.08         38.00            50406
Running  fc+mrv  ...
                      time elapsed nodes expanded   constraint checks
                      0.09         38.00            49935
```

Εδώ οι διαφορές ανάμεσα στα constraint checks είναι πάλι μικρές αλλα υπάρχει περισσότερη διαφορά σε σχέση με τον mac+mrv

```
Running  mincon  ...
                            time elapsed nodes expanded   constraint checks
                            2.16            399.00           1394752
```

Η ευρετική min conflicts όπως παρατηρούμε στο συγκεκριμένο πρόβλημα δεν προσφέρει τα πίο καλές επιδόσεις.

```
Running  bt  ...
                          time elapsed nodes expanded   constraint checks
                          0.08           38.00            25016
Running  bt  ...
                          time elapsed nodes expanded   constraint checks
                          0.04           38.00            25016
Running  bt  ...
                          time elapsed nodes expanded   constraint checks
                          0.04           38.00            25016
```

O backracking σκέτος ήταν ο μόνος αλγόριθμος που όσες δοκιμές και να έκανα τα αποτελέσματα ήταν σταθερά, αυτο οφείλεται στο γεγονός ότι δεν υπήρχε διαφορετική επιλογή μεταβλητών απο ευρετική καθώς δεν χρησιμοποιήθηκε καμία.

```
Running  bt+mrv  ...
                          time elapsed nodes expanded   constraint checks
                          0.08           38.00            22601
Running  bt+mrv  ...
                          time elapsed nodes expanded   constraint checks
                          0.05           38.00            22948
Running  bt+mrv  ...
                          time elapsed nodes expanded   constraint checks
                          0.05           38.00            22583
```

Η χρήση ευρετικής mrv όπως παρατηρούμε δίνει πλέον διαφορετικά αποτελέσματα στα contraint checks.

```
Running  fc+dom  ...
                          time elapsed nodes expanded   constraint checks
                          0.07           38.00            51249
Running  fc+dom  ...
                          time elapsed nodes expanded   constraint checks
                          0.10           38.00            50757
Running  fc+dom  ...
                          time elapsed nodes expanded   constraint checks
                          0.07           38.00            51882
```

Παρατηρούμε οτι για τον αλγόριθμο fc η χρήση dom wdeg ευρετικής αυξάνει ελάχιστα τα constraint checks.

```
Running  mac+dom  ...
                        time elapsed nodes expanded   constraint checks
                        2.17           38.00              846873
Running  mac+dom  ...
                        time elapsed nodes expanded   constraint checks
                        2.40           38.00              917754
Running  mac+dom  ...
                        time elapsed nodes expanded   constraint checks
                        2.39           38.00              924055
```

Εδώ παρατηρούμε οτι για τον αλγόριθμο mac η χρήση dom wdeg ευρετικής αυξάνει ακόμα πιο πολύ τα constraint checks.

```
Running  bt+dom  ...
                        time elapsed nodes expanded   constraint checks
                        0.08           38.00              46398
Running  bt+dom  ...
                        time elapsed nodes expanded   constraint checks
                        0.08           38.00              46398
Running  bt+dom  ...
                        time elapsed nodes expanded   constraint checks
                        0.09           38.00              46398
```

Εδώ παρατηρούμε οτι για τον αλγόριθμο backtracking η χρήση dom wdeg ευρετικής σχεδόν διπλασιάζει τα constraint checks σε σχέση με την ευρετική mrv.

Απο τις διαφάνειες του μαθήματος για τις 2 ευρετικές dom και mrv έχουμε:

Η συνάρτηση SELECT-UNASSIGNED-VARIABLE επιλέγει την επόμενη μεταβλητή που δεν έχει τιμή με τη σειρά που προκύπτει από τη λίστα VARIABLES[CSP]. Αυτή η **στατική** επιλογή μεταβλητών σπάνια καταλήγει σε αποδοτική αναζήτηση.

Ο ευρετικός μηχανισμός των **ελάχιστων απομενουσών τιμών (minimum remaining values - MRV)** επιλέγει τη μεταβλητή με τις λιγότερες νόμιμες τιμές που απομένουν, δηλαδή ελαχιστοποιεί τον παράγοντα διακλάδωσης.

Με τον MRV έχουμε **δυναμική** επιλογή μεταβλητών.

Ο ευρετικός μηχανισμός **βαθμού (degree heuristic)** είναι ο εξής: επιλέγουμε τη μεταβλητή που εμπλέκεται στο μεγαλύτερο πλήθος περιορισμών με άλλες μεταβλητές που δεν τους έχει ανατεθεί τιμή (δηλαδή τη μεταβλητή-κόμβο με το **μεγαλύτερο βαθμό** στον γράφο των περιορισμών). Αυτή η επιλογή είναι στατική.

Βαθμός κόμβου σ' ένα γράφο είναι το πλήθος των ακμών που πρόσκεινται στον κόμβο.

Διαίσθηση: Αυξάνουμε τη μελλοντική απαλειφή τιμών, προκειμένου να μειώσουμε τους μελλοντικούς παράγοντες διακλάδωσης.

Όπως λέει και στις διαφάνειες η στατική επιλογή μεταβλητών σπάνια καταλήγει σε αποδοτική αναζήτηση, ο mrv κάνει δυναμική επιλογή μεταβλητών και ο dom wdeg στατική, επομένως καταλήγουμε οτι για το συγκεκριμένο πρόβλημα ίσως ταιρίαζει πιο πολύ η ευρετική mrv για περισσότερη αποδοτικότητα.

3)

Από τον περιορισμό που απογορεύει ένα μάθημα κοινού εξαμήνου να εξετάζεται την ίδια μέρα με άλλο προκύπτει ότι οι ελάχιστες μέρες που θα έχει το πρόγραμμα είναι 16. Αυτό γιατί τα μαθήματα που 7ου εξαμήνου είναι 16.