Προβλήματα Ικανοποίησης Περιορισμών

(Επιπλέον Διαφάνειες)

Μανόλης Κουμπαράκης

1

Περιεχόμενα

- Παραδείγματα CSP
- Παράδειγμα εκτέλεσης του αλγόριθμου BT
- Sudoku
- k-consistency
- Η έννοια της αποσύνθεσης δένδρου

2

Το Παζλ της Ζέβρας

1	2	3	4	5

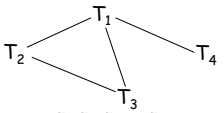
N_i = {English, Spaniard, Japanese, Italian, Norwegian}
C_i = {Red, Green, White, Yellow, Blue}
D_i = {Tea, Coffee, Milk, Fruit-juice, Water}
J_i = {Painter, Sculptor, Diplomat, Violinist, Doctor}
A_i = {Dog, Snails, Fox, Horse, Zebra}

The Englishman lives in the Red house
The Spaniard has a Dog
The Japanese is a Painter
The Italian drinks Tea
The Norwegian lives in the first house on the left
The owner of the Green house drinks Coffee
The Green house is on the right of the White house
The Sculptor breeds Snails
The Diplomat lives in the Yellow house
The owner of the middle house drinks Milk
The Norwegian lives next door to the Blue house
The Violinist drinks Fruit juice
The Fox is in the house next to the Doctor's
The Horse is next to the Diplomat's

Who owns the Zebra?
Who drinks Water?

3

Χρονοπρογραμματισμός Εργασιών



Πρόβλημα: Δίνονται οι εργασίες T_1, T_2, T_3 και T_4 που σχετίζονται με τους εξής χρονικούς περιορισμούς:

- Η T_1 πρέπει να γίνει κατά τη διάρκεια της T_3
- Η T_2 πρέπει να έχει τελειώσει όταν αρχίζει η T_1
- Η T_2 δεν πρέπει να γίνεται παράλληλα με την T_3
- Η T_4 πρέπει να αρχίσει αφού τελειώσει η T_1

Ερωτήσεις:

- Πως θα ορίσουμε το παραπάνω πρόβλημα σαν CSP?
- Είναι οι παραπάνω περιορισμοί συνεπείς?
- Ποιες είναι οι δυνατές χρονικές σχέσεις ανάμεσα στις παραπάνω ενέργειες?
- Τι γίνεται αν οι παραπάνω ενέργειες χρησιμοποιούν κάποιους κοινούς πόρους?

4

Ο Αλγόριθμος της Χρονολογικής Υπαναχώρησης (BT)

Ο αλγόριθμος αυτός είναι ουσιαστικά μια έκδοση του αλγόριθμου αναζήτησης πρώτα κατά βάθος που έχει επεκταθεί με έλεγχο των περιορισμών σε κάθε βήμα.

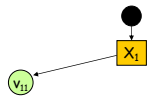
5

Χρονολογική Υπαναχώρηση (3 μεταβλητές)

Ανάθεση = {}

6

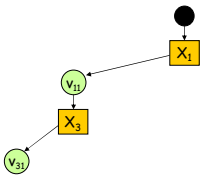
Χρονολογική Υπαναχώρηση
(3 μεταβλητές)



Ανάθεση = {(X_1,v_{11})}

7

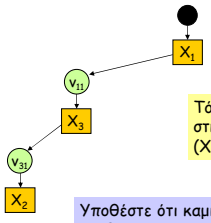
Χρονολογική Υπαναχώρηση
(3 μεταβλητές)



Ανάθεση = {(X_1,v_{11}), (X_3,v_{31})}

8

Χρονολογική Υπαναχώρηση
(3 μεταβλητές)



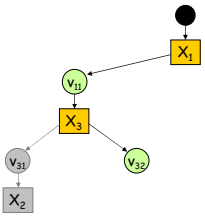
Τότε ο αλγόριθμος υπαναχωρεί στην προηγούμενη μεταβλητή (X_3) και δοκιμάζει μια άλλη τιμή.

Υποθέστε ότι καμιά τιμή της X_2 δεν οδηγεί σε ανάθεση μεταβλητών που ικανοποιεί τους περιορισμούς.

Ανάθεση = {(X_1,v_{11}), (X_3,v_{31})}

9

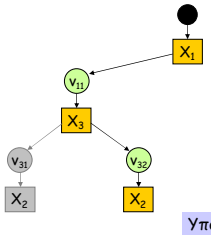
Χρονολογική Υπαναχώρηση
(3 μεταβλητές)



Ανάθεση = {(X_1,v_{11}), (X_3,v_{32})}

10

Χρονολογική Υπαναχώρηση
(3 μεταβλητές)



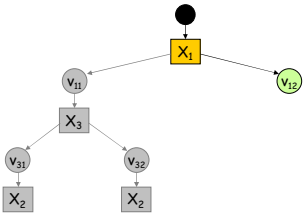
Ο αλγόριθμος υπαναχωρεί στην προηγούμενη μεταβλητή (X_3) και δοκιμάζει την επόμενη τιμή. Αλλά ας υποθέσουμε ότι η X_3 έχει μόνο δύο δυνατές τιμές. Τότε, ο αλγόριθμος υπαναχωρεί στη X_1.

Υποθέστε ξανά ότι καμιά τιμή της X_2 δεν οδηγεί σε ανάθεση που ικανοποιεί τους περιορισμούς.

Ανάθεση = {(X_1,v_{11}), (X_3,v_{32})}

11

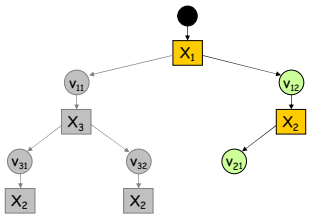
Χρονολογική Υπαναχώρηση
(3 μεταβλητές)



Ανάθεση = {(X_1,v_{12})}

12

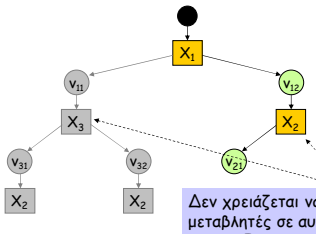
Χρονολογική Υπαναχώρηση
(3 μεταβλητές)



Ανάθεση = {(X_1,v_{12}), (X_2,v_{21})}

13

Χρονολογική Υπαναχώρηση
(3 μεταβλητές)

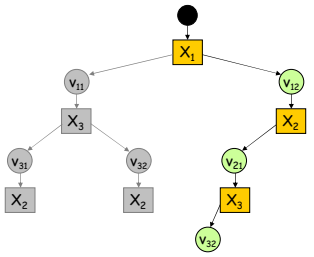


Δεν χρειάζεται να εξετάσουμε τις μεταβλητές σε αυτό το υποδένδρο με την ίδια σειρά που τις εξετάσαμε στο διπλανό.

Ανάθεση = {(X_1,v_{12}), (X_2,v_{21})}

14

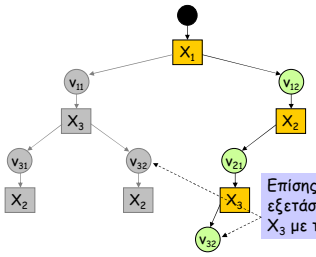
Backtracking Search
(3 variables)



Ανάθεση = {(X_1,v_{12}), (X_2,v_{21}), (X_3,v_{32})}

15

Χρονολογική Υπαναχώρηση
(3 μεταβλητές)

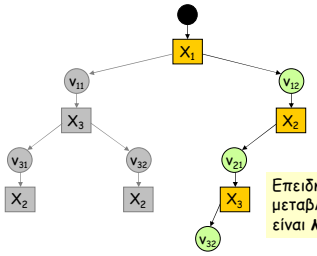


Επίσης, δεν χρειάζεται να εξετάσουμε τις τιμές της X_3 με την ίδια σειρά.

Ανάθεση = {(X_1,v_{12}), (X_2,v_{21}), (X_3,v_{32})}

16

Χρονολογική Υπαναχώρηση
(3 μεταβλητές)



Επειδή υπάρχουν μόνο 3 μεταβλητές, η ανάθεση είναι **λύση**.

Ανάθεση = {(X_1,v_{12}), (X_2,v_{21}), (X_3,v_{32})}

17

Παράδειγμα: Sudoku

	1	2	3	4	5	6	7	8	9
A			3		2		6		
B	9			3		5			1
C			1	8		6	4		
D			8	1		2	9		
E	7								8
F			6	7		8	2		
G			2	6		9	5		
H	8			2		3			9
I			5		1		3		

(a)

	1	2	3	4	5	6	7	8	9
A	4	8	3	9	2	1	6	5	7
B	9	6	7	3	4	5	8	2	1
C	2	5	1	8	7	6	4	9	3
D	5	4	8	1	3	2	9	7	6
E	7	2	9	5	6	4	1	3	8
F	1	3	6	7	9	8	2	4	5
G	3	7	2	6	8	9	5	1	4
H	8	1	4	2	5	3	7	6	9
I	6	9	5	4	1	7	3	8	2

(b)

18

Το Sudoku σαν CSP

- **Μεταβλητές**: 81, μία για κάθε τετράγωνο. Θα τις ονομάζουμε A1,…,A9, …, I1,…,I9.
- **Πεδία**: {1, …, 9} για τις μεταβλητές που αντιστοιχούν σε κενά τετράγωνα. Για τις υπόλοιπες μεταβλητές, ένα μονοσύνολο που περιέχει τον αριθμό στο αντίστοιχο τετράγωνο.
- **Περιορισμοί**: 27 περιορισμοί *Alldiff* (ένας για κάθε γραμμή, ένας για κάθε στήλη και ένας για κάθε περιοχή που αποτελείται από εννέα τετράγωνα). Υποθέτουμε ότι κάθε περιορισμός *Alldiff* δίνεται σαν ένα σύνολο δυαδικών περιορισμών ≠.

19

Εφαρμόζουμε Συνέπεια Ακμής

- Θεωρήστε την μεταβλητή E6.
- Από τους περιορισμούς για την περιοχή της, μπορούμε να απαλείψουμε τις τιμές 1, 2, 7 και 8 από το πεδίο της.
- Από τους περιορισμούς για την στήλη της, μπορούμε να απαλείψουμε τις τιμές 5, 6, 2, 8, 9 και 3.
- Άρα το πεδίο της E6 γίνεται { 4 }.

20

Εφαρμόζουμε Συνέπεια Ακμής

- Θεωρήστε την μεταβλητή I6.
- Εφαρμόζοντας συνέπεια ακμής στη στήλη του, μπορούμε να απαλείψουμε τις τιμές 5, 6, 2, 4 (λόγω της E6), 8, 9 και 3 από το πεδίο της.
- Εφαρμόζοντας συνέπεια ακμής με την I5, απαλείφουμε την τιμή 1 από το πεδίο της.
- Άρα το πεδίο της I6 γίνεται { 7 }.
- Βλέπουμε επίσης τώρα ότι το πεδίο της A6 γίνεται { 1 }.

21

Εφαρμόζουμε Συνέπεια Ακμής

- Συνεχίζοντας με αυτό τον τρόπο μπορούμε να δούμε ότι εφαρμόζοντας συνέπεια ακμής, **επιλύουμε πλήρως το παζλ**.
- Εύκολα Sudoku μπορούν να επιλυθούν με συνέπεια ακμής.
- Πιο δύσκολα Sudoku μπορούν να επιλυθούν με **συνέπεια μονοπατιού** (path-consistency ή 3-consistency) με πολύ μεγαλύτερο υπολογιστικό κόστος. Υπάρχουν 255,960 διαφορετικοί περιορισμοί μονοπατιού που πρέπει να θεωρήσουμε σε ένα παζλ Sudoku.

22

k-consistency

- Αν εφαρμόσουμε k-consistency σε ένα δοσμένο CSP για k=1,…,n (n ο αριθμός των μεταβλητών) τότε το CSP είναι άμεσα επιλύσιμο με τον παρακάτω αλγόριθμο που δεν χρειάζεται να υπαναχωρήσει.

23

k-consistency

- Βρίσκουμε μια τιμή από το πεδίων τιμών της X1. Αυτό είναι δυνατό αφού το CSP είναι 1-consistent.
- Επεκτείνουμε αυτή τη μερική ανάθεση βρίσκοντας μια τιμή στο πεδίο της μεταβλητής X2 η οποία, μαζί με την τιμή της X1, ικανοποιεί όλους τους περιορισμούς για τις μεταβλητές X1 και X2. Αυτό είναι δυνατό γιατί το CSP είναι 2-consistent.
- Συνεχίζουμε με τον ίδιο τρόπο μέχρι να φτάσουμε στη μεταβλητή Xn.
- Ο αλγόριθμος αυτός έχει πολυπλοκότητα $O(nd)$.

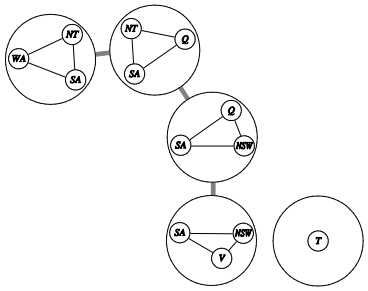
24

k-consistency

- Όμως αν εφαρμόσουμε k-consistency σε ένα δοσμένο CSP για k=1,…,n, τότε θα χρειαστούμε χρόνο και χώρο εκθετικό στο n στην χειρότερη περίπτωση.
- Οπότε ο προηγούμενος αλγόριθμος δεν είναι χρήσιμος στην πράξη.
- Στην πράξη συνήθως εφαρμόζουμε 2-consistency ή σπανιότερα 3-consistency.

25

Αποσύνθεση Δένδρου (Tree Decomposition)



26

Αποσύνθεση Δένδρου

- Μια **αποσύνθεση δένδρου** μετατρέπει το γράφο των περιορισμών (δηλ. το δοσμένο πρόβλημα) σε ένα δένδρο που αποτελείται από μικρότερους γράφους (δηλ. μικρότερα υποπροβλήματα).
- Κάθε υποπρόβλημα λύνεται ξεχωριστά και μετά οι λύσεις που βρέθηκαν συνδυάζονται για να προκύψει μια λύση του αρχικού προβλήματος.

27

Αποσύνθεση Δένδρου

- Μια **αποσύνθεση δένδρου** θα πρέπει να ικανοποιεί τους παρακάτω περιορισμούς:
  - Κάθε μεταβλητή του αρχικού προβλήματος εμφανίζεται σε τουλάχιστον ένα από τα υποπροβλήματα.
  - Αν δυο μεταβλητές συνδέονται με ένα περιορισμό στο αρχικό πρόβλημα, πρέπει να βρίσκονται μαζί (και μαζί με τον περιορισμό) σε τουλάχιστον ένα από τα υποπροβλήματα.
  - Αν μια μεταβλητή εμφανίζεται σε δύο υποπροβλήματα στο δένδρο, πρέπει να εμφανίζεται και σε κάθε υποπρόβλημα κατά μήκος της διαδρομής που συνδέει αυτά τα υποπροβλήματα.

28

Αποσύνθεση Δένδρου

- Οι δύο πρώτες συνθήκες εξασφαλίζουν ότι όλες οι μεταβλητές και όλοι οι περιορισμοί αναπαρίστανται στην αποσύνθεση.
- Η τρίτη συνθήκη φροντίζει ώστε οποιαδήποτε μεταβλητή να έχει την ίδια τιμή σε κάθε υποπρόβλημα το οποίο εμφανίζεται.

29

Αλγόριθμος Επίλυσης

- Επιλύουμε το κάθε υποπρόβλημα ανεξάρτητα.
- Αν οποιοδήποτε υποπρόβλημα δεν έχει λύση, τότε γνωρίζουμε ότι ολόκληρο το πρόβλημα δεν έχει λύση.
- Αν μπορούμε να επιλύσουμε όλα τα υποπροβλήματα, τότε επιχειρούμε να κατασκευάσουμε μια καθολική λύση ως εξής.

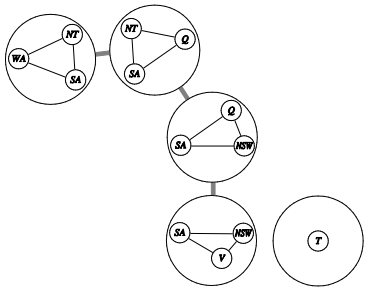
30

Αλγόριθμος Επίλυσης

- Πρώτα βλέπουμε το κάθε υποπρόβλημα ως μια «υπερμεταβλητή» που το πεδίο της είναι το σύνολο όλων των λύσεων του υποπροβλήματος.
- **Παράδειγμα**: Το αριστερό υποπρόβλημα του προηγούμενου παραδείγματος είναι πρόβλημα χρωματισμού χάρτη με τρείς μεταβλητές και επομένως έχει 3!=6 λύσεις. Μια λύση είναι η εξής: { WA=red, SA=blue, NT=green }

31

Παράδειγμα



32

Αλγόριθμος Επίλυσης

- Μετά επιλύουμε το πρόβλημα ικανοποίησης περιορισμών που σχηματίζουν οι «υπερμεταβλητές» χρησιμοποιώντας την αποδοτική μέθοδο για δένδρα που παρουσιάσαμε.
- Οι περιορισμοί μεταξύ των υποπροβλημάτων απλώς απαιτούν οι λύσεις των υποπροβλημάτων να συμφωνούν ως προς τις κοινές μεταβλητές.
- **Παράδειγμα**: Με δεδομένη τη λύση { WA=red, SA=blue, NT=green } για το αριστερό υποπρόβλημα του προηγούμενου παραδείγματος, η μόνη συνεπής λύση για το επόμενο υποπρόβλημα είναι η { SA=blue, NT=green, Q=red }.

33

Πλάτος Δένδρου

- Ένα δοσμένος γράφος περιορισμών επιδέχεται πολλές αποσυνθέσεις δένδρου.
- Κατά την επιλογή μιας αποσύνθεσης, θέλουμε να κάνουμε τα υποπροβλήματα όσο το δυνατόν μικρότερα.
- **Ορισμός**. Το **πλάτος δένδρου (tree-width)** μιας αποσύνθεσης δένδρου ενός γράφου περιορισμών είναι ένας ακέραιος κατά ένα μικρότερος από το μέγεθος του μεγαλύτερου υποπροβλήματος του γράφου.
- **Ορισμός**. Το **πλάτος δένδρου** ενός γράφου περιορισμών είναι το ελάχιστο πλάτος δένδρου μεταξύ όλων των δυνατών αποσυνθέσεων δένδρου αυτού του γράφου.

34

Πρόταση

- Αν ένας γράφος περιορισμών έχει πλάτος w και μας δοθεί η αντίστοιχη αποσύνθεση δένδρου, τότε το πρόβλημα μπορεί να επιλυθεί σε χρόνο $O(nd^{w+1})$ όπου n είναι ο αριθμός των μεταβλητών και d το μέγιστο μέγεθος πεδίου.
- Επομένως, τα προβλήματα ικανοποίησης περιορισμών που έχουν γράφους περιορισμών με **σταθερό πλάτος δένδρου** είναι **επιλύσιμα σε πολυωνυμικό χρόνο.**
- Η εύρεση της αποσύνθεσης με το μικρότερο πλάτος είναι **NP-hard πρόβλημα**, υπάρχουν όμως ευρετικές μέθοδοι που λειτουργούν καλά στην πράξη.

35

Ευχαριστίες

- Οι διαφάνειες 1-17 είναι ακριβής μετάφραση διαφανειών του Jean-Claude Latombe από το Πανεπιστήμιο Stanford που είναι διαθέσιμες στην ιστοσελίδα http://ai.stanford.edu/~latombe/cs121/2011/schedule.htm.

36