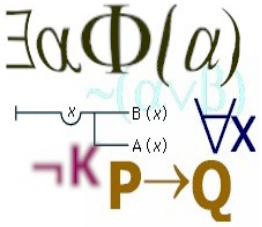
Πανεπιστήμιο Μακεδονίας
Τμ. Εφαρμοσμένης Πληροφορικής

Ικανοποίηση Περιορισμών (Constraint Satisfaction)

Ηλίας Σακελλαρίου



Δομή

- Διαχείριση Αριθμητικών Εκφράσεων σε Prolog
- Προβλήματα ικανοποίησης περιορισμών
- Μέθοδοι Επίλυσης
  - Αλγόριθμοι Ελέγχου Συνέπειας
- Προγραμματισμός με περιορισμούς στον ΛΠ

Πρόβλημα: Διαχείριση Αριθμητικών Εκφράσεων

- Η διαχείριση των αριθμητικών εκφράσεων στην Prolog **δεν ακολουθεί τη συνηθισμένη δηλωτική ερμηνεία.**

  ?- X > 3, X < 6.

  … instantiation fault in X > 3

Μια Λύση

- Για να μπορέσει η Prolog να απαντήσει, αν γνώριζα ότι το X είναι στο διάστημα [1..7]:

  ?- member(X,[1,2,3,4,5,6,7]), X > 3, X < 6.

  X = 4

  Yes

  X = 5

  Yes

Σχόλια

- Προηγούμενο είναι παραγωγή και δοκιμή.
- Σύνθετο πρόβλημα.
- Χαρακτηριστικά:
  - Μεταβλητές & Πεδία τιμών.
  - Λογικές Σχέσεις.

```
solve(X,Y,Z):-
  member(X,[1,2,3,4,5,6,7,8,9,10]),
  member(Y,[1,2,3,4,5,6,7,8,9,10]),
  member(Z,[4,5,6,7,8,9,10,11]),
  X > 3, X < 9,
  Y > 2, Y < 9,
  X =:= Y + 1,
  Z > 5,
  Z < X.
```

Ικανοποίηση Περιορισμών (1/2)

- Ένα πρόβλημα ικανοποίησης περιορισμών (constraint satisfaction problem) αποτελείται από:
  - Ένα σύνολο **n μεταβλητών** $V_1, V_2,...,V_n$,
  - Ένα σύνολο **n πεδίων τιμών** $D_1, ...D_n$, που αντιστοιχούν σε κάθε μεταβλητή έτσι ώστε $V_i \in D_i$
  - Ένα σύνολο **σχέσεων (περιορισμών)** $C_1, C_2,...C_m$ όπου $C_i(V_k,...,V_m)$ μια σχέση μεταξύ των μεταβλητών του προβλήματος.

Ικανοποίηση Περιορισμών (2/2)

- Ανάλογα με το πόσες μεταβλητές περιλαμβάνει ένας περιορισμός χαρακτηρίζεται ως:
  - **μοναδιαίος (unary)** όταν αφορά μια μεταβλητή,
  - **δυαδικός (binary)** όταν αφορά δύο μεταβλητές ή
  - **ανώτερης τάξης (higher order)** όταν αφορά περισσότερες των δύο μεταβλητές.
- **Λύση** αποτελεί μια ανάθεση τιμών στις μεταβλητές του προβλήματος, τέτοια ώστε να ικανοποιούνται οι περιορισμοί, δηλαδή:

$$V_1 = d_1 \wedge V_2 = d_2 \wedge \ldots \wedge V_n = d_n$$

$$\wedge \quad d_1 \in D_1 \wedge d_2 \in D_2 \wedge \ldots \wedge d_i \in D_n \quad \wedge C_1 \wedge C_2 \ldots \wedge C_m$$

Περιορισμοί

- Είναι **λογικές σχέσεις** μεταξύ **μεταβλητών**, όπου κάθε μεταβλητή μπορεί να πάρει τιμές από ένα **συγκεκριμένο πεδίο**.
  - Περιορίζει τις πιθανές τιμές που μπορούν να πάρουν οι μεταβλητές, δηλ. εκφράζει **μερική πληροφορία** για το πρόβλημα.
  - Για παράδειγμα σε μια εφαρμογή χρονοπρογραμ-ματισμού, αν S_A και S_B είναι οι χρόνοι έναρξης των εργασιών A και B, και D_A η διάρκεια της A, τότε ο περιορισμός

    $$\mathbf{S_A + D_A < S_B}$$

    δηλώνει ότι η εργασία B πρέπει να γίνει μετά την A.

Χαρακτηριστικά Περιορισμών

- Οι περιορισμοί είναι:
  - **δηλωτικοί**: ορίζουν μια σχέση μεταξύ των οντοτήτων του προβλήματος χωρίς να ορίζουν μια συγκεκριμένη υπολογιστική διαδικασία.
  - **προσθετικοί**: ενδιαφέρει συνήθως η σύζευξη των περιορισμών και όχι η σειρά με την οποία τέθηκαν.
  - **σπανίως ανεξάρτητοι:** στη συνηθέστερη περίπτωση οι περιορισμοί έχουν κοινές μεταβλητές.
- Είναι ένας φυσικός τρόπος έκφρασης προβλημάτων σε ένα εξαιρετικό φάσμα πεδίων.

Προγραμματισμός με Υποστήριξη Περιορισμών

- CP (constraint programming): Μελέτη συστημάτων βασισμένων στους περιορισμούς.
- Δηλωτικό Παράδειγμα Προγραμματισμού:

"Ο προγραμματιστής δηλώνει ποιοι είναι οι περιορισμοί του προβλήματος και η πλατφόρμα προσφέρει την υποδομή για την επίλυση τους".

- Συνδυάζει αποτελέσματα από διάφορα πεδία: *τεχνητή νοημοσύνη, επιχειρησιακή έρευνα, λογική, νευρωνικά δίκτυα*, κλπ.
- Έχει αναγνωριστεί από την ACM σαν μια από τις **στρατηγικές κατευθύνσεις** στην έρευνα στο πεδίο των υπολογιστών.

Επίλυση

- Αναζήτηση
  - αλλά σε κάθε βήμα ανάθεσης τιμής ελέγχω ότι μπορώ.

```
solve(X,Y,Z):-
  member(X,[1,2,3,4,5,6,7,8,9,10]),
  member(Y,[1,2,3,4,5,6,7,8,9,10]),
  member(Z,[4,5,6,7,8,9,10,11]),
  X > 3, X < 9,
  Y > 2, Y < 9,
  X =:= Y + 1,
  Z > 5,
  Z < X.
```

```
solve_dfs(X,Y,Z):-
  member(X, [1,2,3,4,5,6,7,8,9,10]),
  X > 3, X < 9,
  member(Y, [1,2,3,4,5,6,7,8,9,10]),
  Y > 2, Y < 9,
  X =:= Y + 1,
  member(Z, [4,5,6,7,8,9,10,11]),
  Z > 5,
  Z < X.
```

Επίλυση

- Αναδιάταξη Μεταβλητών.

```
solve_dfs(X,Y,Z):-
  member(X, [1,2,3,4,5,6,7,8,9,10]),
  X > 3, X < 9,
  member(Y, [1,2,3,4,5,6,7,8,9,10]),
  Y > 2, Y < 9,
  X =:= Y + 1,
  member(Z, [4,5,6,7,8,9,10,11]),
  Z > 5,
  Z < X.
```

```
solve_dfs_h(X,Y,Z):-
  member(Z, [4,5,6,7,8,9,10,11]),
  Z > 5,
  member(X, [1,2,3,4,5,6,7,8,9,10]),
  X > 3, X < 9,
  Z < X,
  member(Y, [1,2,3,4,5,6,7,8,9,10]),
  Y > 2, Y < 9,
  X =:= Y + 1.
```

Μείωση Αριθμού Τιμών

- Μερική πληροφορία περιορισμών.
  - πχ. για το Z λόγω των περιορισμού Z > 5.
- Έτσι οι ακόλουθοι υποστόχοι:

  member(Z, [4, 5, 6, 7, 8, 9, 10, 11]),

  Z > 5,
- μετατρέπονται στον:

  member(Z, [6, 7, 8, 9, 10, 11]),

Εφαρμογή Μοναδιαίων Περιορισμών

- Μοναδιαία Συνέπεια Κόμβου

```
solve_dfs_h(X,Y,Z):-
  member(Z, [4, 5, 6, 7, 8, 9, 10, 11]),
  Z > 5,
  member(X, [1,2,3,4,5,6,7,8,9,10]),
  X > 3, X < 9,
  Z < X,
  member(Y, [1,2,3,4,5,6,7,8,9,10]),
  Y > 2, Y < 9,
  X =:= Y + 1.
```

```
solve_filter(X,Y,Z):-
  member(Z, [6,7,8,9,10,11]),
  member(X, [4,5,6,7,8]),
  Z < X,
  member(Y, [3,4,5,6,7,8]),
  X =:= Y + 1.
```

Δυαδικοί Περιορισμοί

- ***Z < X:***
  - Z **δεν** μπορεί να είναι >= 8.
  - X **δεν** μπορεί να είναι < 6

```
solve_filter(X,Y,Z):-
  member(Z, [6,7,8,9,10,11]),
  member(X, [4,5,6,7,8]),
  Z < X,
  member(Y, [3,4,5,6,7,8]),
  X =:= Y + 1.
```

Δυαδικοί Περιορισμοί (cont)

- ***X =:= Y + 1:***
  - Y **δεν** μπορεί να είναι 3,4,5 και 8.

```
solve_filter(X,Y,Z):-
  member(Z, [6,7]),
  member(X, [7,8]),
  Z < X,
  member(Y, [6,7]),
  X =:= Y + 1.
```

Παρατηρήσεις

- Δραματική μείωση τιμών.
  - Από τους αρχικούς 800 συνδυασμούς τιμών (10 * 10 * 8), απέμειναν 8 συνδυασμοί.
- Ενδιαφέρουσες παρατηρήσεις:
  - μοναδιαίοι περιορισμοί εξετάστηκαν μια φορά,
  - οι δυαδικοί περιορισμοί διατηρήθηκαν ακόμη και μετά την μείωση των αντίστοιχων πεδίων τιμών,
  - οι αλλαγές στο πεδίο τιμών της μεταβλητής X λόγω του περιορισμού Z < X, οδήγησε σε μείωση τους πεδίου της Y λόγω του περιορισμού X =:= Y + 1.
    - Επαναληπτική εξέταση περιορισμών.

Αλγόριθμοι Συνέπειας

- Προηγούμενα ήταν ο βασικός κορμός των αλγορίθμων συνέπειας.
  - AC-3, AC-2000, κλπ.
- Συνέπεια δεν είναι αρκετή.
  - Απαιτείται και αναζήτηση.

Συνδυασμός αναζήτησης και αλγορίθμων συνέπειας/διήθησης τιμών.

Prolog + Constraints = CLP

- Πως φιλοξενεί ο LP τους περιορισμούς;
  - Νέος τύπος μεταβλητών, των μεταβλητών περιορισμών,
    - συνδέονται με ένα πεδίο τιμών.
  - Αντικατάσταση της κλασικής ενοποίησης για αυτές τις μεταβλητές.
    - Επίλυση περιορισμών (αλγόριθμοι συνέπειας).
  - “Αποθήκη περιορισμών” (constraint store),
    - διατήρησης των περιορισμών πάνω στις μεταβλητές
    - χρήση από αλγόριθμους συνέπειας

Παράδειγμα Μεταβλητών και Πεδίων.

?- X #:: [1..10].

X = X{1..10}

yes

?- X #:: [1..10], X = 3.

X = 3

Yes

?- X #:: [1..10], X = 12.

No

Εφαρμογή Μοναδιαίων Περιορισμών

?- X #:: [1..10], X #> 5.

X = X{6..10}

Yes

Δυαδικοί Περιορισμοί

- Delayed Goals
  - Περιορισμοί που έχουν διατηρηθεί όσο χρειάζεται.
- *?- X #:: [1..10], Y #:: [1..10], X #< Y.*

X = X{1..9}

Y = Y{2..10}

There is 1 delayed goal.

Yes

- *?- X #:: [1..10], Y #:: [1..10], X #< Y, X = 3.*

X = 3

Y = Y{4..10}

Yes

Αποτυχία λόγω διαγραφής τιμών.

- *?- X #:: [1..10], X #> 8.*

  X = X{[9, 10]}

  Yes

- *?- X #:: [1..10], X #> 8, X #< 9.*

  No

Ανάθεση τιμών

- Τιμές από πεδία τιμών.
- Εναλλακτικές κατά την οπισθοδρόμηση.

?- X #:: [1..10], X #> 8, indomain(X).

X = 9

Yes ;

X = 10

Yes

Ανάθεση τιμών σε πολλές μεταβλητές.

- Κατηγόρημα labeling/1

?- X #:: [1..10], Y #:: [1..10], X #< Y, X #> 7, labeling([X,Y]).

X = 8

Y = 9 ;

Yes

X = 8

Y = 10 ;

Yes

…

Μεταβλητές και Πεδία

Περιορισμοί

Αναζήτηση και ανάθεση τιμών

Παράδειγμα

Prolog

```
solve(X,Y,Z):-
  member(X,[1,2,3,4,5,6,7,8,9,10]),
  member(Y,[1,2,3,4,5,6,7,8,9,10]),
  member(Z,[4,5,6,7,8,9,10,11]),
  X > 3, X < 9,
  Y > 2, Y < 9,
  X =:= Y + 1,
  Z > 5,
  Z < X.
```

CLP

```
solve_clp(X,Y,Z):-
  [X,Y] #:: [1..10],
  Z #:: [4..11],
  X #> 3, X #< 9,
  Y #> 2, Y #< 9,
  X #= Y + 1,
  Z #> 5, Z #< X,
  labeling([X,Y,Z]).
```

Προγραμματισμός με περιορισμούς

- Δημιουργία μιας νέας "σχολής" προγραμματισμού, του *προγραμματισμού με περιορισμούς* (***constraint programming-CP***), στο πλαίσιο της οποίας αναπτύσσονται εργαλεία και νέες γλώσσες προγραμματισμού για την επίλυση τέτοιων προβλημάτων.
- Ένα **CP package** μπορεί να είναι
  - μια βιβλιοθήκη που χρησιμοποιείται με μια “κλασική” γλώσσα προγραμματισμού (C, C++, Java), όπως για παράδειγμα ο ILOG Solver, JCL, Choco, κλπ.
  - επέκταση μιας κατάλληλης γλώσσας προγραμματισμού, πχ. του λογικού προγραμματισμού (Prolog), που οδηγεί στον λογικό προγραμματισμό με υποστήριξη περιορισμών (CLP).

Δηλωτικότητα (Declarativeness)

- **Δηλωτικός Προγραμματισμός:** Ο χρήστης δηλώνει ποιο είναι το πρόβλημα και ο υπολογιστής το λύνει.
  - Ο προγραμματισμός με περιορισμούς είναι δηλωτικός.
  - Κατάλληλη γλώσσα για επέκταση πρέπει να εμφανίζει το παραπάνω χαρακτηριστικό.

"Constraint Programming represents one of the closest approaches computer science has yet made to the Holy Grail of programming: the user states the problem and the computer solves it"

E. Freuder

- Επέκταση των γλωσσών λογικού προγραμματισμού
  - *Λογικός Προγραμματισμός με Περιορισμούς* (*Constraint Logic Programming - CLP*).

Λογικός Προγραμματισμός με Περιορισμούς

- Γιατί ο Λογικός Προγραμματισμός είναι κατάλληλό όχημα για CP?
  - Ύπαρξη Λογικών σχέσεων και στα δύο παραδείγματα.
  - Εγγενής υποστήριξη μεθόδων αναζήτησης (backtracking, depth first search)
  - Μεταβλητές ίδιας φύσης – μαθηματικές (και στις δύο περιπτώσεις μπορεί να είναι ελεύθερες μεταβλητές)

Λογικός Προγραμματισμός με Περιορισμούς

- ... για την ακρίβεια το CP προήλθε σε μεγάλο βαθμό από το λογικό προγραμματισμό.
  - Ανάγκη για διαχείριση μαθηματικών σχέσεων.
  - πχ.
    - X is Y + 3, Y = 10
  - θα δώσει σφάλμα στον κλασσικό Λογικό προγραμματισμό, αλλά μπορεί να υπολογιστεί θαυμάσια στο CLP!

Λογικός Προγραμματισμός με Περιορισμούς

- Παράδειγμα τέτοιου συστήματος είναι το Chip με πλήθος βιομηχανικών εφαρμογών:
  - σύνταξη ωρολογίου προγράμματος για την κατανομή ωρών εργασίας σε νοσοκομείο (Gymnaste, στο νοσοκομείο Blingy),
  - σχεδιασμό ενεργειών (planning) για την οργάνωση γραμμών παραγωγής στην αεροπορική βιομηχανία (PLANE στη Dassault), κτλ.
- Άλλες ιδιαίτερα διαδεδομένες CLP γλώσσες είναι η SICStus, **ECLiPSe Prolog**, η Oz και η gnu-prolog, κλπ
- Πλέον οι περισσότερες εκδόσεις της γλώσσας Prolog υποστηρίζουν σε μεγαλύτερο ή μικρότερο βαθμό την νέα αυτή σχολή προγραμματισμού.