

## **Probleme de satisfacere a constrângerilor**

O problemă de satisfacere a constrângerilor (*Constraint Satisfaction Problem*):

- o mulțime de variabile  $X = \{X_1, \dots, X_n\}$
- fiecare variabilă  $X_i$  poate lua valori dintr-un domeniu  $D_i$
- o mulțime de constrângeri  $C = \{C_1, \dots, C_j\}$  care specifică combinațiile permise de valori.

O soluție pentru o astfel de problemă este o asignare de valori variabilelor a.î. toate constrângerile să fie satisfăcute.

În definiție nu se impune nici o condiție asupra tipului variabilelor. Acestea pot fi întregi, logice, mulțimi, sau de orice alt tip. Nici modul de definire a constrângerilor nu este limitat. Constrângerile pot fi date atât explicit, prin specificarea tupelurilor de valori permise, cât și implicit, prin relații (ex:  $X_i > 2$ ).

Graful restricțiilor: nodurile reprezintă variabilele, muchiile reprezintă restricțiile între variabile.

### **Exemplul 1: Problema colorării unei hărți**

Considerăm o hartă cu  $n$  țări. Fiecare regiune/țară poate fi colorată cu o culoare dintr-o mulțime de culori asociate. Să se coloreze harta a.î. regiunile/țările vecine să fie colorate diferit.

Modelare: asignăm fiecărei regiuni/țări de pe hartă o variabilă; domeniul fiecărei variabile este o mulțime de culori specificată. Restricțiile sunt de forma:  $X_i \neq X_j$  (două țări vecine au culori diferite). Obs: între două țări vecine în graful constrâns vom adăuga o muchie.

### **Exemplul 2: Sudoku**

Un puzzle Sudoku de ordin 3 constă dintr-o tablă de 9x9 în care fiecare pătrat poate avea un număr de la 1 la 9. Dată o asignare parțială a tablei, să se completeze pozițiile rămase libere a.î. fiecare număr să apară o singură dată pe o linie, pe o coloană și într-o regiune 3x3.

O posibilă modelare ca o problemă CSP: fiecare poziție a tablei este reprezentată de o variabilă  $X_{ij}$ , iar domeniul unei variabile este un număr de la 1 la 9.

Pentru fiecare linie, coloană și regiune 3x3 vom asocia o constrângere de tip *alldifferent* (toate variabilele din constrângere trebuie să aibă valori diferite de restul variabilelor).

### **Exemplul 3: problema reginelor**

Dată fiind o tablă de șah  $n \times n$  dorim să plasăm  $n$  regine pe tablă a.î. nici o regină să nu fie atacată de nici o altă regină.

O posibilă formulare a problemei reginelor ca o problemă CSP: pentru fiecare coloană a tablei de șah asociem o variabilă  $X_i$ , iar domeniul variabilei sunt liniile, adică  $D_i = \{1, \dots, n\}$ .

Constrângerile sunt asociate fiecărei perechi de coloane și precizează că două regine nu se pot afla pe aceeași linie sau pe aceeași diagonală; se exprimă prin relațiile:

$$X_i \neq X_j$$

$$|X_i - X_j| \neq |i - j|, \text{ pentru orice } i, j \text{ din intervalul } 1, \dots, n.$$

## Abordări

- Algoritmul **Backtracking**: menține o soluție parțială (o mulțime de variabile instanțiate corect) pe care o extinde pas cu pas. Inițial, mulțimea este vidă. La fiecare pas se selectează următoarea variabilă din ordonare și se încearcă asignarea variabilei cu o valoare consistentă cu instanțierea parțială. Dacă este găsită o astfel de valoare, algoritmul continuă procedeul cu următoarea variabilă din ordonare. În caz contrar, ne întoarcem la variabila anterioară și îi asignăm o altă valoare consistentă.

## - Propagarea constrângerilor

*Forward-checking*: verifică ca fiecare valoare să fie compatibilă cu cel puțin o valoare din domeniul fiecărei variabile viitoare. Se instanțiază variabila cu o valoare și apoi se elimină valori din domeniul variabilelor viitoare care sunt în conflict cu instanțierea curentă. Dacă domeniul unei variabile viitoare devine vid, algoritmul consideră următoarea valoare posibilă pentru variabila curentă.

Ordonarea variabilelor

**MRV** (*Minimum remaining values*): alege variabila cu cele mai puține valori ramase în domeniu

## Temă

Implementați algoritmul *Forward-checking* + *MRV* pentru problema de colorare a unei hărți, modelată ca problemă de satisfacere a constrângerilor.

**Exemplul 1**: considerăm o hartă cu regiunile WA, SA, NT; regiunile adiacente sunt precizate mai jos:

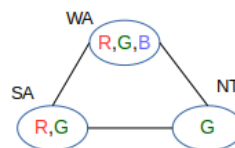
WA: {SA, NT}

SA: {WA, NT}

NT: {WA, SA}

Culorile disponibile pentru fiecare regiune sunt:

WA: {red, green, blue}, SA: {red, green}, NT: {green}.



Aplicarea algoritmului *Forward-checking* + *MRV*:

	WA	SA	NT
Initial	RGB	RG	<b>G</b>
Dupa NT=G	RB	<b>R</b>	<b>G</b>
Dupa SA=R	<b>B</b>	<b>R</b>	<b>G</b>

Exemplul 2:

T: {V}

WA: {NT, SA}

NT: {WA, Q, SA}

SA: {WA, NT, Q, NSW, V}

Q: {NT, SA, NSW}

NSW: {Q, SA, V}

V: {SA, NSW, T}

Mulțimea de culori asociată fiecărei regiuni:

WA: {red}, NT: {red, blue, green}, SA: {red, blue, green}, Q: {green}, NSW: {red, blue, green}, V: {red, blue, green}, T: {red, blue, green}

Vezi secțiunea 5.2 *Backtracking Search for CSPs* din [1].

Etape

(0.3) 1. Modelarea problemei ca o problemă de satisfacere a constrângerilor

(0.5) 2. Implementarea metodei FC

(0.2) 3. Implementarea MRV

În cadrul laboratorului trebuie rezolvat punctul 1.

### **Bibliografie**

[1] *Artificial Intelligence: A Modern Approach*, capitolul 5

<http://aima.cs.berkeley.edu/newchap05.pdf>