### **Rezolvarea jocului Sudoku folosind algoritmul de verificare înainte (Forward Checking).**

### **1. Introducere**

Această documentație prezintă implementarea unui program de rezolvare a Sudoku utilizând algoritmul Forward Checking. Programul este dezvoltat în limbajul C# și are ca scop demonstrabilitatea aplicării algoritmilor de inteligență artificială în probleme de căutare cu spațiu mare de soluții.

### **2. Descrierea problemei**

Sudoku este un joc logic popular, în care jucătorul trebuie să completeze un grid 9x9 astfel încât:

* Fiecare rând să conțină cifrele de la 1 la 9 fără repetări.
* Fiecare coloană să conțină cifrele de la 1 la 9 fără repetări.
* Fiecare dintre cele 9 sub-grile 3x3 să conțină cifrele de la 1 la 9 fără repetări.

Problema poate fi considerată o problemă de satisfacere a constrângerilor (CSP - *Constraint Satisfaction Problem*), ceea ce o face potrivită pentru aplicarea algoritmilor precum Backtracking sau Forward Checking.

### **3. Aspecte teoretice privind algoritmul Forward Checking**

Algoritmul Forward Checking este o tehnică de optimizare aplicată în problemele de CSP pentru reducerea spațiului de căutare.  
**Principiul de bază**: După atribuirea unei valori unei variabile, se verifică imediat efectele acestei atribuiri asupra variabilelor neasignate. Dacă cel puțin o variabilă rămâne fără opțiuni valide, se efectuează backtracking.

**Pași principali**:

1. Se selectează o variabilă de asignat (euristică MRV - *Minimum Remaining Values* este des utilizată).
2. Se atribuie o valoare variabilei din domeniul său curent.
3. Se verifică impactul asupra variabilelor neasignate.
4. Dacă domeniul unei variabile devine gol, se revine la pasul anterior (backtracking).
5. Se continuă până când gridul este complet sau s-a determinat că soluția nu există.

Avantaje:

* Reducerea semnificativă a numărului de verificări inutile.
* Îmbunătățirea eficienței față de backtracking-ul clasic.

### **4. Modalitatea de rezolvare**

#### **a. Modelarea problemei**

* Gridul Sudoku este reprezentat ca o matrice 9x9.
* Fiecare poziție conține o valoare între 1 și 9 sau este goală (0).
* Constrângerile Sudoku sunt implementate sub formă de reguli.

#### **b. Implementarea algoritmului**

1. **Inițializare:** Se citește gridul și se inițializează domeniile posibile pentru fiecare poziție necompletată.
2. **Selectarea variabilei:** Se aplică o euristică (ex. MRV).
3. **Asocierea valorilor:** Se verifică dacă o valoare respectă constrângerile curente.
4. **Forward Checking:** Se actualizează domeniile posibile pentru variabilele neasignate.
5. **Backtracking:** În cazul unui blocaj, se revine la pasul anterior.

#### **c. Detalii de implementare în C#**

Programul este scris în C# utilizând paradigma orientată pe obiecte pentru o mai bună structurare a codului. Clasele principale includ:

* SudokuGrid pentru reprezentarea tablei de Sudoku.
* ForwardCheckingSolver pentru implementarea algoritmului.
* Utilities pentru funcționalități auxiliare (citire, afișare rezultate).

### **5. Cod sursă și explicații**

#### **Exemplu de cod: Inițializarea domeniilor**

**Explicație:**

* Pentru fiecare poziție goală din grid, se determină valorile posibile pe baza regulilor Sudoku.
* Rezultatele sunt stocate într-un dicționar având coordonatele drept cheie și lista de valori posibile drept valoare.

### **6. Rezultate obținute**

Se vor include capturi de ecran din rularea aplicației în diferite scenarii, precum:

* Grid-uri complet rezolvate.
* Situații în care soluția nu există.
* Timpul de execuție pentru diverse dificultăți ale puzzle-ului.

### **7. Concluzii**

Implementarea algoritmului Forward Checking în rezolvarea Sudoku a demonstrat o performanță superioară metodei de backtracking simplu, reducând semnificativ timpul de căutare. Cu toate acestea, complexitatea problemei crește exponențial pentru grid-uri de dimensiuni mai mari.

### **8. Bibliografie**

..

### **9. Contribuții individuale ale membrilor echipei**

* **Pintilie Justinian:** Designul clasei SudokuGrid și logica regulilor Sudoku.
* **Florea Alexandra:** Implementarea algoritmului Forward Checking și optimizări.
* **:** Testarea programului și realizarea capturilor de ecran.