Universiteti i Prishtinës
Fakulteti i Inxhinierisë Elektrike dhe Kompjuterike
Drejtimi: Inxhinieri Kompjuterike

Dokumentimi i detyrës

Lënda: Inteligjenca Artificiale

Zgjidhja e problemit N-Queens me Constraint Programming

Emri profesorit/Asistentit

Prof. Dr. Nysret Musliu

Msc. Adrian Ymeri

Emri & mbiemri i studentëve

1. Altin Shabani

2. Erza Gashi

Prishtinë, 2023

1. Përshkrimi i problemit

Në problemin Blocked N-Queens, kemi gjithashtu tabelë shahu NxN dhe N mbretëresha. Çdo katror mund të mbajë më së shumti një mbretëreshë. Disa katrorë në tryezë janë të bllokuara dhe nuk mund të mbajnë asnjë mbretëreshë. Kushti është që mbretëreshat të mos sulmojnë njëra-tjetrën. Hyrja e këtij problemi është numri i mbretëreshave dhe fushave të bllokuara.
Problemi Queens është një variant ku, si dhe n, hyrja përmban një listë katrorësh të cilët janë të bllokuar. Një zgjidhje për problemin është një zgjidhje për n
Problemi Queens është i lidhur ngushtë me problemin e plotësimit n-Queens dhe problemin e Diagonaleve të përjashtuara n-Queens.

1.1 Si i zgjidhim N-Queens

Njerëzit e zgjidhin këtë problem duke eksperimentuar me konfigurime të ndryshme.
Ata përdorin njohuri të ndryshme rreth problemit për të eksploruar vetëm një numër të vogël konfigurimesh përpara se të gjejnë një përgjigje.
Problemi është se është e paqartë saktësisht se cilat janë këto njohuri. Për më tepër, njerëzit do ta kishin të vështirë të zgjidhnin një problem 1000 Mbretëreshës!
Kompjuterët janë të mirë për të bërë shpejt një numër të madh gjërash të thjeshta. Pra, një zgjidhje e mundshme është të provojmë sistematikisht çdo vendosje të mbretëreshave derisa të gjejmë një zgjidhje.

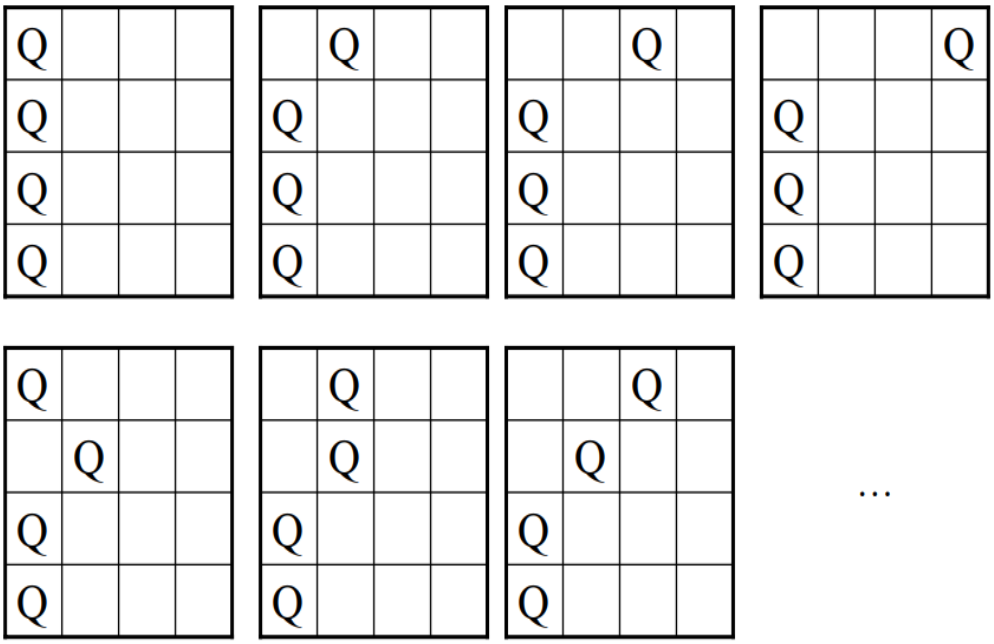

Figure 1. Nje zgjidhje sistematike

1. Një grup variablash.
2. Një grup vlerash për secilën prej variablave.
3. Një grup kufizimesh midis koleksioneve të ndryshme të variablave.

Problemi n-mbretëresha pyet se si të vendosen n mbretëresha në një tabelë shahu $n \times n$ në një mënyrë që të mos ndërhyjnë dy mbretëresha. Në mënyrë të detajuar kjo do të thotë:

Në çdo vijë vertikale të tabelës lejohet vetëm një mbretëreshë, ne do t'i referohemi këtyre rreshtave si kolona.
Në çdo vijë horizontale të tabelës lejohet vetëm një mbretëreshë, këto rreshta do të quhen rreshta më vonë.
Në çdo vijë diagonale lejohet vetëm një mbretëreshë.
Ky mund të modelohet si një program binar në mënyrën e mëposhtme:
Le të tregojmë $x_{i,j} \in \{0,1\}$ nëse një mbretëreshë vendoset në rreshtin e i dhe në kolonën j të tabelës së shahut. Meqenëse problemi është gjetja e një vendosjeje, funksioni objektiv është i parëndësishëm. Ne shtojmë, megjithatë, objektivin e tepërt për të maksimizuar numrin e mbretëreshave të vendosura:

$$\max \sum_{i=1}^{n} \sum_{j=1}^{n} x_{i,j}$$

Tani ne detyrojmë saktësisht një mbretëreshë të vendoset në çdo kolonë dhe çdo rresht:

$$\sum_{i=1}^{n} x_{i,j} = 1, \quad j = 1,\ldots,n$$
$$\sum_{j=1}^{n} x_{i,j} = 1, \quad i = 1,\ldots,n$$

Rreshtat diagonale janë pak më të komplikuara për t'u shkruar:

$$\sum_{i=1}^{n-j+1} X_{i,j+i-1} <= 1, \quad j = 1,\ldots,n$$
$$\sum_{j=1}^{n-i+1} X_{i+j-1,j} <= 1, \quad i = 1,\ldots,n$$
$$\sum_{i=1}^{n-j+1} X_{i,n-j-i+2} <= 1, \quad j = 1,\ldots,n$$
$$\sum_{j=1}^{n-i+1} X_{i+j-1,n-j+1} <= 1, \quad i = 1,\ldots,n$$

- **Qasja CP ndaj problemit N-queens**

Një zgjidhës CP punon duke provuar sistematikisht të gjitha caktimet e mundshme të vlerave për variablat në një problem, për të gjetur zgjidhjet e realizueshme. Në problemin me 4 mbretëresha, zgjidhësi fillon në kolonën më të majtë dhe vendos me radhë një mbretëreshë në secilën kolonë, në një vend që nuk sulmohet nga asnjë mbretëreshë e vendosur më parë.
Supozojmë se fillojm duke vendosur në mënyrë arbitrare një mbretëreshë në këndin e sipërm të majtë. Kjo është një lloj hipoteze; që do të rezultojë se nuk ekziston asnjë zgjidhje me një mbretëreshë në këndin e sipërm të majtë.
Një kufizim është se mund të ketë vetëm një mbretëreshë në një kolonë (X-të gri poshtë), dhe një kufizim tjetër ndalon dy mbretëresha në të njëjtën diagonale (X-të e kuqe më poshtë).

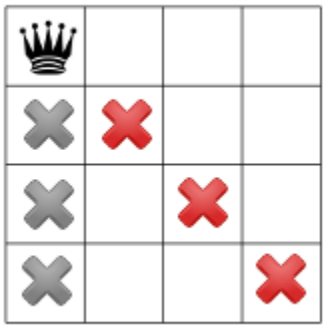

Figure 2. Queen 1

Kufizimi tjetër i ndalon mbretëreshat në të njëjtin rresht:

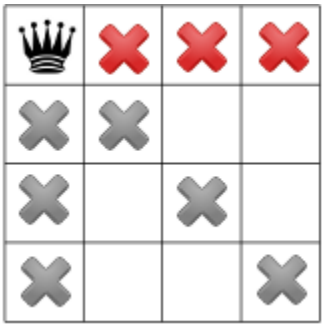

Figure 3. Ndalimi i Queens ne te njejtin rresht

Kufizimet tona u përhapën, ne mund të testojmë një hipotezë tjetër dhe të vendosim një mbretëreshë të dytë në një nga katrorët e mbetur në dispozicion. Zgjidhësi ynë mund të vendosë të vendosë në të katrorin e parë të disponueshëm në kolonën e dytë:

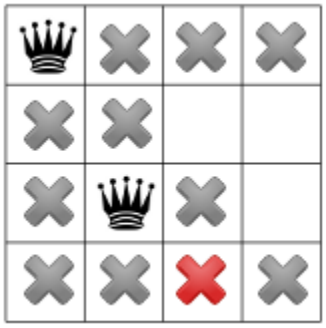

Figure 4. Queen 2

Pas përhapjes së kufizimit diagonal, mund të shohim se ai nuk lë katrorë të disponueshëm as në kolonën e tretë, as në rreshtin e fundit:

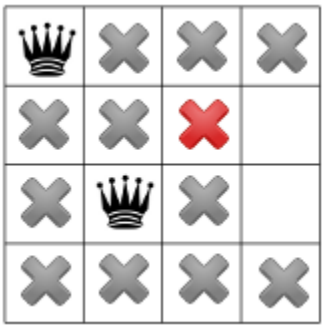

Figure 5. Pamja e Queen 2

Pa zgjidhje të mundshme në këtë fazë, ne duhet të tërhiqemi. Një opsion është që zgjidhësi të zgjedhë katrorin tjetër të disponueshëm në kolonën e dytë. Sidoqoftë, përhapja e kufizimeve më pas detyron një mbretëreshë në rreshtin e dytë të kolonës së tretë, duke mos lënë asnjë pikë të vlefshme për mbretëreshën e katërt:

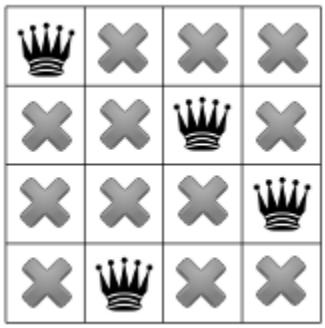

Figure 6. Pamja Finale

Në vazhdim gjendet përshkrimi i kodit:

Importimi i librarive . Kodi i mëposhtëm importon librarit kryesore.

```
import com.google.ortools.Loader;
import com.google.ortools.sat.CpModel;
import com.google.ortools.sat.CpSolver;
import com.google.ortools.sat.CpSolverSolutionCallback;
import com.google.ortools.sat.IntVar;
import com.google.ortools.sat.LinearExpr;
```

Deklarimi i modelit: Kodi i mëposhtëm deklaron modelin CP-SAT.

```
CpModel model = new CpModel();
```

Krijimi i variablat: Zgjidhësi krijon variablat për problemin si një grup me emrin queens.

```
int boardSize = 8;
IntVar[] queens = new IntVar[boardSize];
for (int i = 0; i < boardSize; ++i) {
  queens[i] = model.newIntVar(0, boardSize - 1, "x" + i);
}
```

Supozojmë se queens[j] është numri i rreshtit për mbretëreshën në kolonën j. Ku , queens[j] = i do të thotë se ka një mbretëreshë në rreshtin i dhe kolonën j.

Krijoni i Constrain: Këtu është kodi që krijon kufizimet për problemin.

```
// All rows must be different.
model.addAllDifferent(queens);

// All columns must be different because the indices of queens are all different.
// No two queens can be on the same diagonal.
LinearExpr[] diag1 = new LinearExpr[boardSize];
LinearExpr[] diag2 = new LinearExpr[boardSize];
for (int i = 0; i < boardSize; ++i) {
  diag1[i] = LinearExpr.newBuilder().add(queens[i]).add(i).build();
  diag2[i] = LinearExpr.newBuilder().add(queens[i]).add(-i).build();
}
model.addAllDifferent(diag1);
model.addAllDifferent(diag2);
```

Kodi përdor metodën AddAllDifferent, e cila kërkon që të gjithë elementët e një grupi të ndryshueshëm të jenë të ndryshëm.

Kufizimet garantojnë tre kushtet për problemin N-queens (mbretëresha në rreshta, kolona dhe diagonale të ndryshme).

- **Nuk ka dy mbretëresha në të njëjtin rresht**

Zbatimi i metodës AllDifferent të zgjidhësit për mbretëreshat detyron vlerat e mbretëreshave[j] të jenë të ndryshme për secilën j, që do të thotë se të gjitha mbretëreshat duhet të jenë në rreshta të ndryshëm.

- **Nuk ka dy mbretëresha në të njëjtën kolonë**

Meqenëse asnjë element i mbretëreshave nuk mund të ketë të njëjtin indeks, asnjë dy mbretëreshë nuk mund të jetë në të njëjtën kolonë.

- Nuk ka dy mbretëresha në të njëjtën diagonale

Kufizimi diagonal është pak më i ndërlikuar se kufizimet e rreshtave dhe kolonave. Së pari, nëse dy mbretëresha shtrihen në të njëjtën diagonale, një nga kushtet e mëposhtme duhet të jetë i vërtetë:

Numri i rreshtit plus numri i kolonës për secilën nga dy mbretëreshat janë të barabartë. Me fjalë të tjera, mbretëreshat(j) + j kanë të njëjtën vlerë për dy indekse të ndryshëm j.

Numri i rreshtit minus numrin e kolonës për secilën nga dy mbretëreshat janë të barabartë. Në këtë rast, queens(j) - j ka të njëjtën vlerë për dy indekse të ndryshëm j.

Renditja nuk ka asnjë efekt në grupin e zgjidhjeve, vetëm në mënyrën se si i përfytyroni ato.

Pra, kufizimi diagonal është që vlerat e mbretëreshave (j) + j duhet të jenë të gjitha të ndryshme, dhe vlerat e mbretëreshave (j) - j duhet të jenë të gjitha të ndryshme, për j të ndryshme.

Aplikimi i metodës AddAllDifferent te queens(j) + j, ne vendosim shembujt N të ndryshores, për j nga 0 në N-1, në një grup, diag1, si më poshtë:

```
q1 = model.NewIntVar(0, 2 * board_size, 'diag1_%i' % i)
diag1.append(q1)
model.Add(q1 == queens[j] + j)
```

Pastaj aplikojmë AddAllDifferent në diag1.

```
model.AddAllDifferent(diag1)
```

Programi gjen 92 zgjidhje të ndryshme për një tabelë 8x8. Këtu është i pari.

```
Q _ _ _ _ _ _ _
_ _ _ _ _ _ Q _
_ _ _ _ Q _ _ _
_ _ _ _ _ _ _ Q
_ Q _ _ _ _ _ _
_ _ _ Q _ _ _ _
_ _ _ _ _ Q _ _
_ _ Q _ _ _ _ _
...91 other solutions displayed...
Solutions found: 92
```

Figure 7. Zgjidhje per 8 Queens

Numri i zgjidhjeve rritet afërsisht në mënyrë eksponenciale me madhësinë e tabelës:

Board size	Solutions
1	1
2	0
3	0
4	2
5	10
6	4
7	40
8	92
9	352
10	724
11	2680
12	14200
13	73712
14	365596
15	2279184

Figure 8. Numri i zgjidhjeve per N-parametra