

SVEUČILIŠTE U ZAGREBU  
PRIRODOSLOVNO-MATEMATIČKI FAKULTET  
MATEMATIČKI ODSJEK

Lovro Gorup

RJEŠAVANJE IGRE NONOMINO  
SUDOKU

Zagreb, 2022./2023.

## SADRŽAJ

|   |   |
|---|---|
| 1. UVOD .....   | 2 |
| 2. ANALIZA PROBLEMA .....                                       | 3 |
| 3. PRISTUPI RJEŠAVANJA.....                                     | 3 |
| 3.1. NAIVNI PRISTUP .....                                       | 3 |
| 3.1. BACKTRACKING ALGORITAM .....                               | 3 |
| 3.1.1. OSNOVNI BACKTRACKING ALGORITAM .....                     | 4 |
| 3.1.2. PRVO POBOLJŠANJE BACKTRACKING ALGORITMA.....             | 5 |
| 3.1.3. DRUGO POBOLJŠANJE BACKTRACKING ALGORITMA .....           | 5 |
| 4. USPOREDBA VREMENSKIH SLOŽENOSTI BACKTRACKING ALGORITAMA..... | 5 |
| 5. ZAKLJUČAK .....  | 6 |
| 6. LITERATURA .....   | 6 |

## 1. UVOD

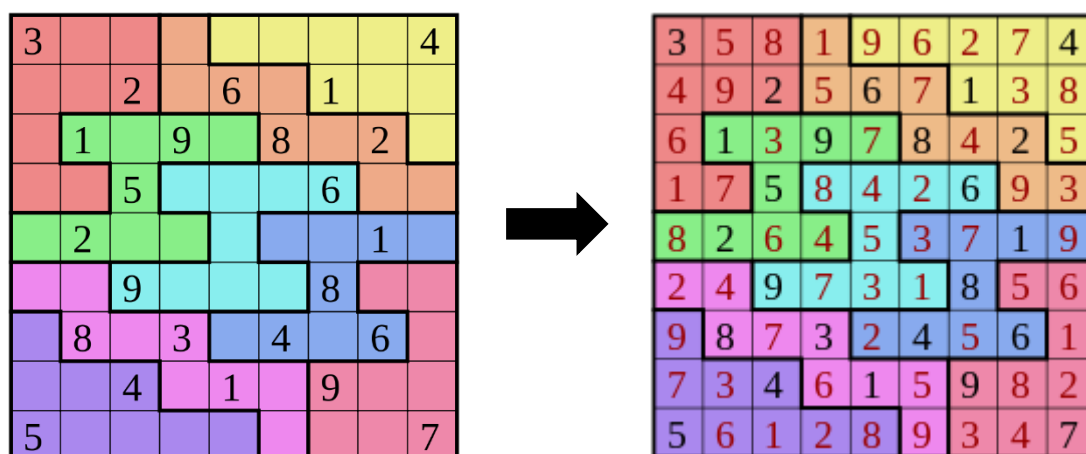
Igra sudoku japanska je igra čije se rješavanje temelji na logici i kombinatorici. Iako su postojale neke varijante i ranije, smatra se da je današnji, moderni sudoku, prvi puta objavljen 1979. godine.

Igra sudoku stekla je veliku popularnost i postoje mnoge varijante. U ovom radu bit će predstavljena varijanta zvana Nonomino sudoku i implementirat će se efikasan algoritam rješavanja.

Nonomino (Jigsaw) sudoku započinje 9x9 tablicom koja sadrži polja koja su prazna ili se u njima nalazi jedan od brojeva 1, 2, 3, 4, 5, 6, 7, 8, 9. Dodatno, tablica je podijeljena u 9 područja koji se sastoje od 9 povezanih polja i svako područje je obojan različitom bojom.

Cilj je popuniti sva prazna polja brojevima 1, 2, 3, 4, 5, 6, 7, 8, 9, tako da vrijede sljedeća tri uvjeta:

- U svakom stupcu pojavljuju se svi mogući brojevi i to točno jednom
- U svakom retku pojavljuju se svi mogući brojevi i to točno jednom
- U svakom području iste boje pojavljuju se svi mogući brojevi i to točno jednom



|   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|
| 3 |   |   |   |   |   |   |   | 4 |
|   |   | 2 |   | 6 |   | 1 |   |   |
|   | 1 |   | 9 |   | 8 |   | 2 |   |
|   |   | 5 |   |   |   | 6 |   |   |
|   | 2 |   |   |   |   |   | 1 |   |
|   |   | 9 |   |   |   | 8 |   |   |
|   | 8 |   | 3 |   | 4 |   | 6 |   |
|   |   | 4 |   | 1 |   | 9 |   |   |
| 5 |   |   |   |   |   |   |   | 7 |

|   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|
| 3 | 5 | 8 | 1 | 9 | 6 | 2 | 7 | 4 |
| 4 | 9 | 2 | 5 | 6 | 7 | 1 | 3 | 8 |
| 6 | 1 | 3 | 9 | 7 | 8 | 4 | 2 | 5 |
| 1 | 7 | 5 | 8 | 4 | 2 | 6 | 9 | 3 |
| 8 | 2 | 6 | 4 | 5 | 3 | 7 | 1 | 9 |
| 2 | 4 | 9 | 7 | 3 | 1 | 8 | 5 | 6 |
| 9 | 8 | 7 | 3 | 2 | 4 | 5 | 6 | 1 |
| 7 | 3 | 4 | 6 | 1 | 5 | 9 | 8 | 2 |
| 5 | 6 | 1 | 2 | 8 | 9 | 3 | 4 | 7 |

Slika 1 Nonomino tablica i rješenje

## 2. ANALIZA PROBLEMA

Problem traženja rješenja igre sudoku, pa tako i varijante nonomino, je tzv. *constraint satisfaction problem (CSP)*, čije rješenje zahtijeva skup objekata koji zadovoljavaju određena ograničenja. U ovom primjeru su ta ograničenja upravo tri uvjeta iz prethodnog poglavlja. Kažemo da je tablica konzistentna ako su zadovoljena sva tri uvjeta. Cilj je pronaći popunjenu tablicu koja je konzistentna.

Uočava se da postojanje i jedinstvenost rješenja ovise o zadanoj početnoj tablici. Ako početna tablica ne zadovoljava zadana pravila, onda očito nema rješenje. Inače, u ovisnosti o broju zadanih elemenata i njihovom položaju zadana tablica ili ima jedinstveno rješenje ili ih ima više (tada je cilj pronaći barem jedno)

## 3. PRISTUPI RJEŠAVANJA

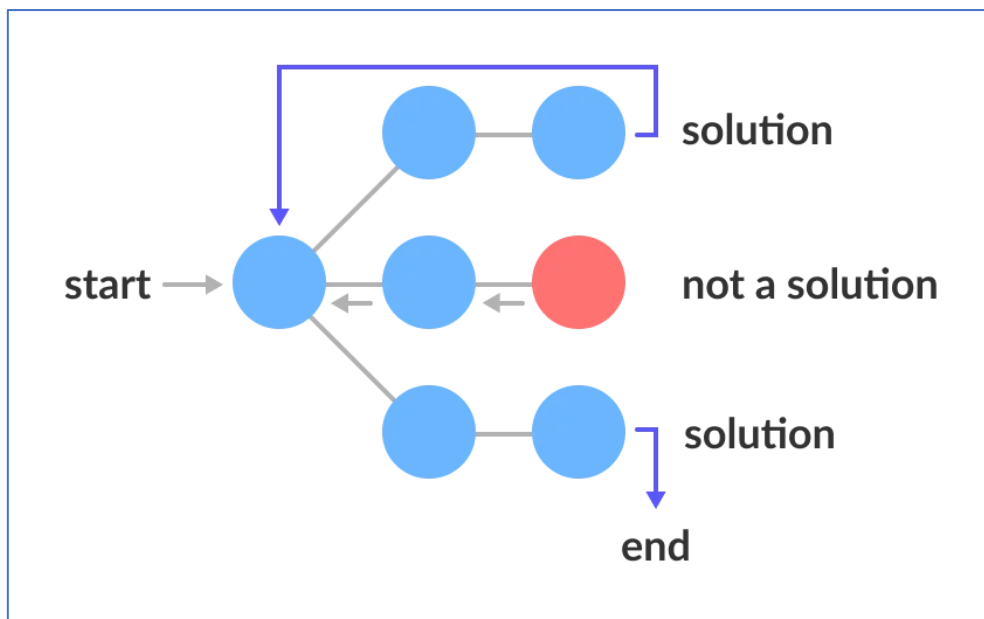
U ovom poglavlju razmatraju se dva različita pristupa problemu. To su naivni pristup i backtracking algoritam.

### 3.1. NAIVNI PRISTUP

Naivni pristup problemu bio bi generiranje svih mogućnosti i provjeravanje svake zadovoljava li uvjet konzistentnosti. Ako označimo s  $n$  broj praznih polja u početnoj tablici, onda je broj različitih mogućnosti jednak  $9^n$ , što je izuzetno neefikasno, pa takav algoritam nije uporabljiv u praksi.

### 3.1. BACKTRACKING ALGORITAM

Backtracking je standardna klasa algoritama u kojima se iterativno konstruiraju kandidati za rješenje određenog problema, a kad je moguće zaključiti da konstruirani kandidat nije rješenje, onda se kandidat eliminira.



*Slika 2 Shematski prikaz backtracking algoritma*

Ideja je sljedeća:

1. Proći po svim praznim poljima u tablici i umetnuti najmanji broj iz skupa  $\{1, \dots, 9\}$  koji čuva konzistentnost
2. Ako takav broj ne postoji, vratiti se na prethodno polje i umetnuti najmanji broj veći od trenutnog koji čuva konzistentnost

Koraci 1 i 2 provode se iterativno sve dok se ne dođe kraja tablice (rješenje postoji) ili dok se ne ustanovi da ne postoji broj koji zadovoljava konzistentnost u prvom slobodnom polju (rješenje ne postoji).

### 3.1.1. OSNOVNI BACKTRACKING ALGORITAM

Uočava se da prolaz po praznim poljima tablice može biti u mnogo redosljeda. U osnovnom tipu prolazimo po poljima odozgo prema dolje i slijeva na desno. Tada za svaki broj iz  $\{1, \dots, 9\}$  provjeravamo je li tablica (nakon umetanja tog elementa) konzistentna.

Problem ovog tipa algoritma je u puno provjera konzistentnosti cijele tablice koja onda utječe na veliku složenost algoritma. Tako implementirani algoritam je potpun (pronalaži rješenje kada ono postoji), ali ima veliku vremensku složenost pa je sada cilj smanjiti ju. Navedeno pretraživanje je slijepo, ali se može dopuniti do usmjerenog korištenjem heuristika.

### 3.1.2. PRVO POBOLJŠANJE BACKTRACKING ALGORITMA

Prvo poboljšanje algoritma je u primjedbi da ne treba za svaki element tablice provjeravati sve moguće kandidate (1, ... , 9), već je dovoljno samo neke. Za to je dovoljno kreirati strukturu podataka koja pamti kandidate svakog polja u tablici. Ta struktura kreira se na početku, prije samog iteriranja algoritma.

Time se postiže to da nije potrebno uopće provjeravati konzistentnost za elemente koji ionako ne mogu biti povoljni za rješavanje s obzirom na danu početnu tablicu, što uvelike smanjuje složenost algoritma.

### 3.1.3. DRUGO POBOLJŠANJE BACKTRACKING ALGORITMA

Drugo poboljšanje je u odabiru poretka kojim se popunjava zadana tablica. U prvim dvama pristupima je poredak s lijeva na desno i odozgo prema dolje.

Algoritam se ubrzava ako se poredak promijeni tako da se prvo krene s retkom čija je ukupna suma kandidata po svim poljima u retku najmanja. Dodatno, svaki redak se sortira po broju kandidata i onda se najprije pristupa poljima s najmanjem brojem kandidata.

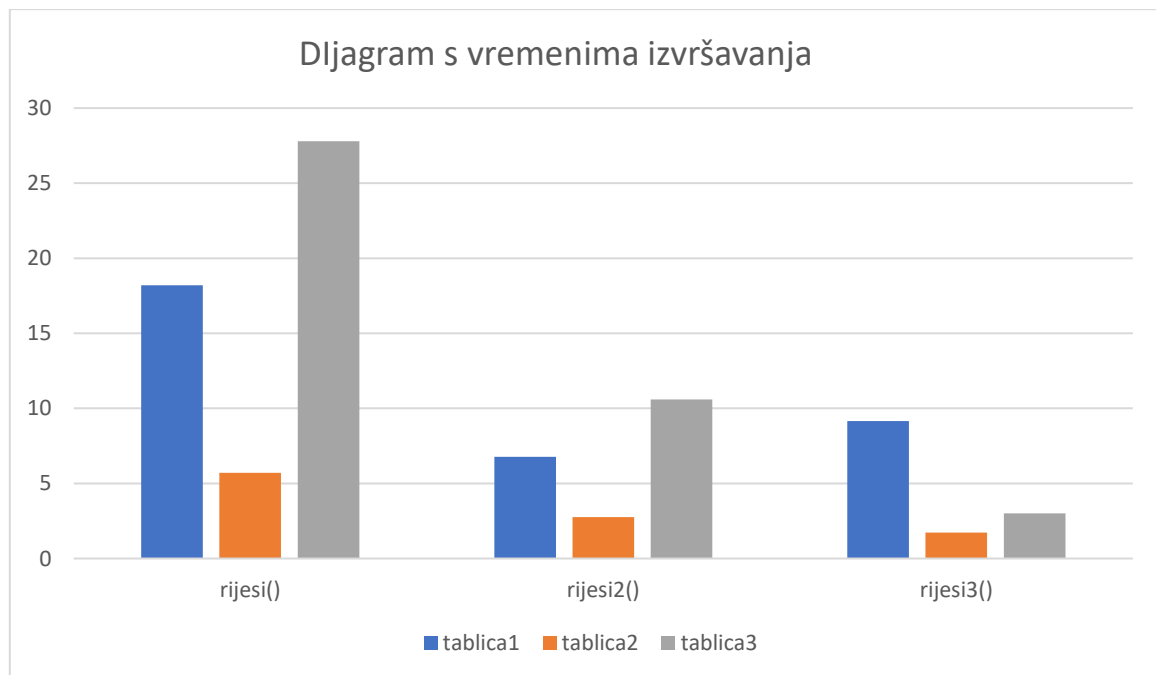
## 4. USPOREDBA VREMENSKIH SLOŽENOSTI BACKTRACKING ALGORITAMA

Neka su *tablica1* i *tablica2* dvije konkretne tablice za rješavanje. Označimo s *rijesi()*, *rijesi2()*, *rijesi3()* funkcije koje implementiraju tri opisana algoritma, respektivno. Tada se za jedno konkretno izvršavanje implementiranog algoritma u programskom jeziku *Python* dobivaju sljedeća vremena izvršavanja za zadane tablice:

|                  | <b>tablica1</b> | <b>tablica2</b> | <b>tablica3</b> |
|------------------|-----------------|-----------------|-----------------|
| <b>rijesi()</b>  | 18.2 s          | 5.7 s           | 27.8 s          |
| <b>rijesi2()</b> | 6.77 s          | 2.75 s          | 10.6 s          |
| <b>rijesi3()</b> | 9.15 s          | 1.72 s          | 1.08 s          |

*Tablica 1 Vremena izvršavanja algoritama na primjerima*

Dobiveni podaci mogu se prikazati i pomoću dijagrama:



U svim slučajevima osim za primjenu `rijesi3()` za tablicu `tablica1` vide se značajna ubrzanja.

## 5. ZAKLJUČAK

Problem nalaženja rješenja igre nonomino sudoku uspješno se implementira backtracking metodama koje se mogu svesti na vrlo malu vremensku složenost postepenim poboljšavanjem. Dobivena vremena izvršavanja pomoću računala (manje od deset sekundi), za razliku od standardnog rješavanja “na papiru”, pokazuje nevjerovatnu snagu i važnost tehnologije za čovjeka.

## 6. LITERATURA

- <https://www.activestate.com/blog/solving-sudoku-with-python-and-artificial-intelligence/>
- <https://medium.com/mlearning-ai/ai-sudoku-solver-cb41b36e446b>
- <http://praneethas.github.io/projects/AI/AIProject-UG201110023.pdf>