BI-ZUM.21
Tomáš Novotný, 14.5. 2023

SUDOKU NA 100%

POPIS DOPLŇOVAČKY SUDOKU S ROZMĚREM N, KDE $\sqrt{N}$ JE PŘIROZENÉ ČÍSLO:

Čtvercová tabulka N x N s N vnitřními bloky. Buňky mohou obsahovat čísla 1 až N. Na začátku jsou některé buňky vyplněné. Úkolem je nevyplněné buňky vyplnit tak, aby bylo splněno, že jsou různé číslice v řádcích, sloupcích a v blocích.

PROBLÉM SPLNITELNOSTI (CONSTRAINT SATISFACTION PROBLEM)

Je speciální případ prohledávání stavového prostoru. Stav reprezentují proměnné s určitými doménami. Všechny cílové stavy musí splňovat podmínky, které reprezentují daný problém. Při přiřazení hodnoty všem proměnným tak, aby byly splněny podmínky, je dosaženo cíle. Cesta není důležitá a umožňuje efektivnější řešení.

SUDOKU JAKO CSP

- Konečná množina proměnných **X** – buňky v tabulce $x_{1,1}$ až $x_{N,N}$
- Doména proměnných **D** – číslice 1 až N
- Množina omezení **C** – různé číslice v řádcích, různé číslice ve sloupcích, různé číslice v blocích

KOMENTÁŘ K IMPLEMENTACI

K řešení úlohy jsem použil knihovnu **python-constraint**.

K inicializaci problém se použije třída **Problem**.

Následně lze proměnné a jejich doménu definovat metodou **addVariable**(„název proměnné“, „doména“).

Omezení lze definovat metodou **addConstraint**(„omezení“, „seznam proměnných“).

Jako omezení různosti čísel v řádcích, sloupcích a blocích jsem použil implementaci **AllDifferentConstraint**(), která využívá heuristiku pro prohledávání forward check.

Řešení se nalezne metodou **getSolution**(). Metoda prohledává stavový prostor backtrackingem.

POZNÁMKY K OPTIMALIZACÍM PROHLEDÁVÁNÍ STAVOVÉHO PROSTORU

Forward check minimalizuje počet proměnných, které se musí přiřadit tak, že dopředu prozkoumá neplatné kombinace.

Backtracking je systematické rekurzivní prohledávání stavového prostoru. Hledá řešení tak, že přiřadí hodnotu proměnné a postupně doplňuje proměnným hodnoty a pokud zjistí, že neexistuje ani jedna proměnná, které lze přiřadit hodnotu, vrací se zpět. Řešení je navíc vždy zaručeno.

POZNÁMKY K TESTOVÁNÍ ŘEŠIČE

Úlohu jsem pojal obecněji tak, aby byla schopná řešit čtvercové Sudoku s dynamickým rozměrem. Moje implementace však dokáže problém řešit efektivně pouze při rozměrech 4x4 až 5x5.

Složitější problémy na poli 5x5 už moje implementace v rozumném čase nedokáže. Přesto Úlohy o rozměru 3x3 a 4x4 dokáže řešit v řádu nižších jednotek vteřin (viz prezentační notebook).

V rychlosti nalezení řešení hraje jistou roli náhoda. Může se stát, že optimalizačním technikám nahrává zadání a výpočet sudoku 25x25 může trvat 3 vteřiny, ale i přes 20 minut (viz prezentační notebook).

Detaily implementace jsou v komentářích skriptu.