FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

# Semestrální práce z KIV/UZI

## OBECNÉ HLEDÁNÍ OPTIMÁLNÍ (HERNÍ) STRATEGIE BARVENÍM GRAFU

Lukáš Runt (A20B0226P)

*lrunt@students.zcu.cz*

15. prosince 2023

# Obsah

# 1 Zadání

Je dán graf, ve kterém vrcholy odpovídají proměnným s doménami (obory hodnot) a hrany jsou označeny jménem podmínky mezi proměnnými. Obarvěte graf, tj. přiřaďte každé proměnné hodnotu z její domény tak, aby byly splněny všechny podmínky. Navrhněte vlastní reprezentaci grafu a vytvořte generátor, který připraví graf např. pro řešení problému N-dam. Realizujte lokálním prohledávání s Tabu seznamem: Nejprve se "náhodně"obarví všechny vrcholy. Pokud existuje nějaký vrchol, který je v konfliktu se svým okolím (není splněna podmínka na hraně), program najde pro tento vrchol jinou barvu tak, aby se konflikt zmenšil. Pro odstranění lokálního minima použijte techniku Tabu seznamu. [1]

V rámci semestrální práce jsem zvolil problém řešení sudoku pomocí strategie barvení grafu. Vstupem se očekává sudoku zadané uživatelem. Výstupem bude vyřešené sudoku.

# 2 Analýza úlohy

## 2.1 Sudoku

Sudoku je logická hra, ve které hráč doplňuje čísla 1 až 9 do mřížky 9x9 tak, aby se v řádku, sloupci a vyznačeném čtverci 3×3 neobjevila stejná čísla. Cílem hry je doplnit všechna čísla.

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

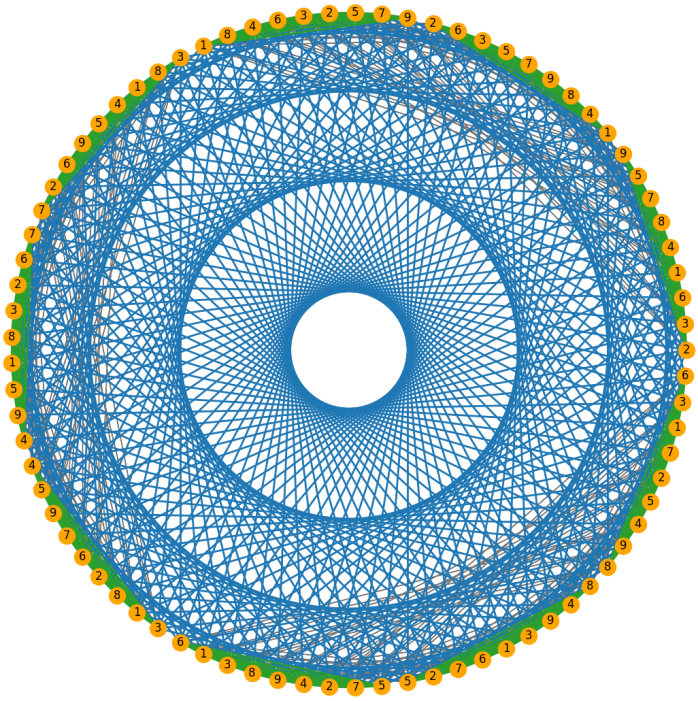
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *4* | 2 | *6* | 5 | *7* | 1 | *3* | 9 | *8* |
| 8 | *5* | *7* | 2 | *9* | 3 | *1* | *4* | 6 |
| *1* | 3 | *9* | *4* | 6 | *8* | *2* | 7 | *5* |
| *9* | *7* | 1 | *3* | *8* | *5* | 6 | *2* | *4* |
| 5 | 4 | *3* | *7* | *2* | *6* | *8* | 1 | 9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

Obrázek 1: Příklad zadání (vlevo) a řešení (vpravo)

## 2.2 Graf

Graf je struktura, která se skládá z množin vrcholů a hran, přičemž hrany znázorňují relace mezi objekty. Grafy se používají k modelování a řešení růz-

ných situací a problémů. Sudoku si můžeme představit jako neorientovaný graf, ve kterém jsou hrací pole vrcholy a hrany jsou relace mezi těmito poli, ve kterých nesmí být stejné číslo. Výsledný graf tedy vytvoříme tak, že vytvoříme hranu mezi všemi poli v řadě, sloupci a čtverci 3×3.



Obrázek 2: Vizualizace grafu sudoku [2]

## 2.3 Obarvování grafu

Obarvování grafu je úloha ve které je cílem přiřadit všem vrcholům v grafu hodnotu (barvu) tak, aby žádné dva sousední vrcholy neměly stejnou barvu. Problém obarvování grafu patří do skupiny NP-úplných problémů, to znamená, že neexistuje žádný známý algoritmus, který by našel optimální řešení v polynomiálním čase vzhledem k velikosti grafu. Obarvování grafu má široké využití v mnoha úlohách jako je vytváření rozvrhu, obarvování map, optimalizace využití frekvence v telekomunikacích, řešení sudoku a ve spoustě dalších úlohách. [3] [4]

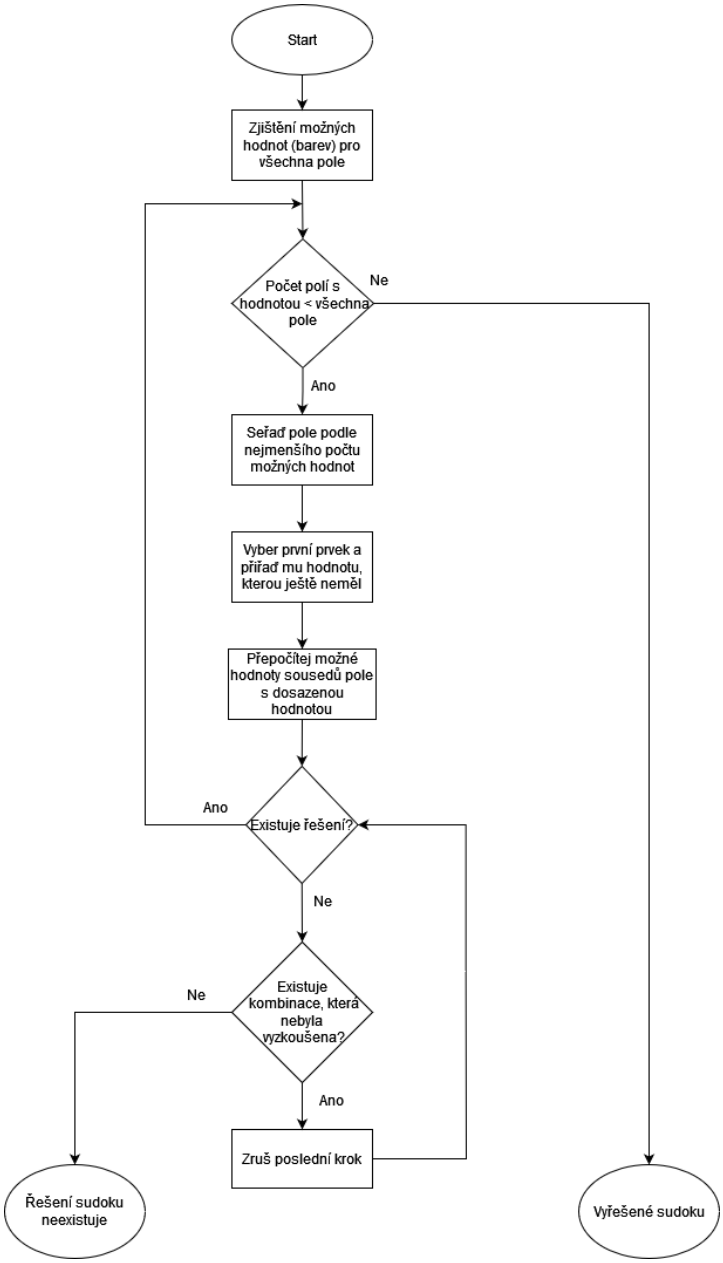
Sudoku se řadí mezi rozhodovací problémy m-barev. Tedy máme dáno m barev (čísla 1 až 9), které přiřazujeme jednotlivým vrcholům tak, aby žádný vrchol nesousedil s vrcholem stejné barvy. Budeme tedy postupně obarvovat námi vytvořený graf s počátečními hodnotami. Při každém obarvení zkontrolujeme sousední vrcholy a přiřadíme hodnotu, kterou žádný ze sousedů nemá. Při barvení se může stát, že jsme vrcholy obarvili špatně a nelze přiřadit zbývajícím vrcholům žádnou hodnotu bez porušení pravidel hry. Pro tento případ je vhodné použít bactracking, tedy vrátit se libovolný počet kroků a zkusit jiné řešení. [5]

### 2.4 Backtracking

Backtracking neboli zpětné hledání je algoritmická technika, využívající rekurzivní řešení problémů. Tato technika se snaží systematicky procházet možná řešení (obvykle prohledáváním stromu do hloubky), přičemž se odstraňují ta řešení, která nesplňují pravidla problému. Typickými úlohami bactrackingu jsou permutační, herní nebo grafové úlohy. [6]

## 3 Popis implementace

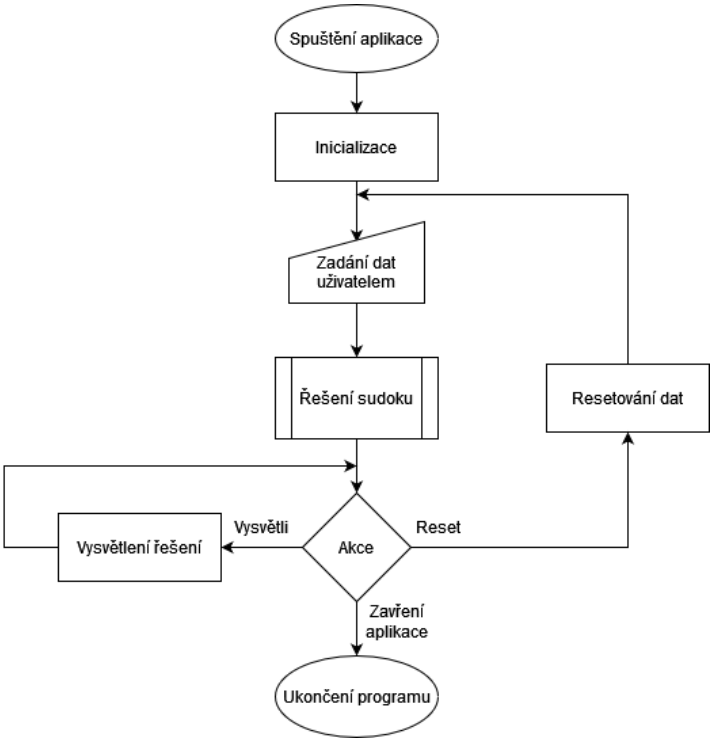
Implementace využívá programátorských strategií, algoritmů a struktur uvedených v analýze úlohy 2. Řešení sudoku využívá algoritmu pro obarvení grafu a bactraking. Algoritmus je navržen tak, že se vždy obarvuje vrchol grafu s nejmenším počtem možných hodnot, neboť je zde největší pravděpodobnost, že se trefíme do správné hodnoty a naše rozhodnutí nebudeme muset později měnit. Na začátku algoritmu tedy vypočteme všechny možné hodnoty (barvy) všem vrcholům, kterým není přiřazena žádná hodnota. Následně všechny vrcholy seřadíme podle počtu možných hodnot a vybereme první v seřazeném pořadí. V dalším kroku upravíme možné hodnoty sousedů vrcholu, kterému byla přiřazena hodnota. V této části kontrolujeme, za má soused v možných hodnotách k přiřazení hodnotu, která byla právě přiřazena. Po této aktualizaci grafu zkontrolujeme, zda lze obarvit ostatní vrcholy. Zde se kontroluje, zda mají všechny vrcholy bez hodnoty alespoň jednu možnou barvu k přiřazení, pokud je zde nějaký vrchol, kterému nelze přiřadit žádná hodnota, musíme se vrátit o jeden nebo více kroků zpět a zkusit jiné řešení. Tímto způsobem jsou vyzkoušeny všechny možné kombinace. Algoritmus opakujeme do té doby, dokud nemají všechny vrcholy přiřazenou hodnotu nebo byly vyzkoušeny všechny kombinace a sudoku nemá řešení.




Obrázek 3: Vývojový diagram řešení sudoku

Kromě řešení sudoku se v programu řeší také samotný běh aplikace, který zahrnuje inicializaci, vysvětlení řešení a připravení grafu pro řešení jiného sudoku. Při spuštění aplikace je provedena inicializace sudoku. V této části je vytvořen graf, ve kterém je každý vrchol spojen s vrcholy reprezentující hrací pole v řadě, sloupci a čtverci 3×3. Po inicializaci je očekáván vstup

dat od uživatele, které jsou následně uloženy do námi vytvořeného grafu. Následně se spustí výše popsaný algoritmus a sudoku je vyřešeno, pokud má vstup od uživatele řešení. Vyřešené sudoku je následně zobrazeno uživateli. V této chvíli má uživatel možnost vysvětlení, jak bylo řešeno některé z polí nebo samotný postup řešení celého sudoku. Toto je implementováno voláním příslušných metod ve vysvětlovacím modulu za použití seznamu s uloženým postupem. Další možností je vyčištění hracího pole pro účel vyřešení dalšího zadání sudoku, které je implementováno vynulováním hodnot všech vrcholů v grafu.




Obrázek 4: Vyvojový diagram běhu aplikace

# 4 Popis programového řešení

Program je navržen jako znalostní systém, je tedy rozdělen do pěti modulů a modulu `main.py`, který slouží ke spuštění programu. Aplikace je napsána v programovacím jazyce `Python`. V rámci implementace bylo vytvořeno grafické uživatelské rozhraní (dále GUI) pro větší komfort uživatele. K vytvoření GUI byla využita knihovna `PyQt5`. [7]

## 4.1 Inferenční modul

Modul `InferencniModul.py` slouží k zobrazení dat z komunikačního modulu, obsluze požadavků uživatele a jeho následnému zpracování.

Obsluhou požadavků uživatele se rozumí jakákoliv interakce uživatele s programem, to jest obsluha tlačítek, textových polí a rozbalovacích listů. Při zaznamenání požadavku uživatele inferenční modul identifikuje o jaký požadavek se jedná a dle toho se rozhodne jaké metody spustit, aby se uživateli dostalo kýženého výsledku. Modul se kromě obsluhy stará také o stabilitu aplikace, když se snaží zabránit neočekávanému chování uživatele, a to uzamčením textových polí, listů nebo tlačítek.

## 4.2 Báze znalostí

Modul `BazeZnalosti.py` obsahuje znalosti a pravidla jak řešit sudoku. Znalosti jsou definovány pomocí tříd `Node`, `Graph` a `Sudoku`.

Třída `Node` reprezentuje vrchol grafu. Jejími atributy jsou `neighbours`, `value`, `defined`, `grid`.

- `neighbours` je pole čísel, která reprezentují indexy sousedících vrcholů.
- `value` je hodnota, kterou vrchol obsahuje. Pokud vrchol obsahuje hodnotu 0, znamená to, že jeho hodnota zatím nebyla určena.
- `defined` značí, zda byla hodnota vrcholu definována uživatelem.
- `grid` obsahuje informaci o indexech vrcholů, které jsou ve stejném čtverci $3\times3$.

Třída `Graph` reprezentuje graf. Jejím atributem je atribut `nodes`, který pole vrcholů grafu. Dále třída `graph` obsahuje metodu pro přidání hrany a kontrolu, zda mezi dvěma vrcholy existuje hrana.

Třída `Sudoku` reprezentuje sudoku. Tato třída obsahuje metody pro inicializaci grafu, kontrolu a vyčištění dat v grafu a metody pro řešení sudoku. Jejími atributy jsou `stack`, `possible_colors`, `number_of_filled_fields` a `graph`.

- `stack` je zásobník, do kterého se ukládá postup řešení sudoku s informacemi jaký vrchol byl řešen, jaká hodnota byla přiřazena a jaké hodnoty je možné podle pravidel dosadit.
- `possible_colors` Je seznam zakázaných a povolených hodnot, které mohou být vrcholu přiřazeny.

- `number_of_filled_fields` je číslo, kolik vrcholů je aktuálně vyplněno.
- `graph` je reprezentace sudoku pomocí grafu.

## 4.3 Báze dat

Modul `BazeDat.py` představuje bázi dat znalostního systému. Tato báze obsahuje informace o počtu políček v hracím poli, velikosti hracího pole, definici čtverců 3×3, výčet možných hodnot, které lze doplnit do hracího pole, informace o souřadnicích a uložení dat sudoku.

## 4.4 Komunikační modul

Modul `KomunikacniModul.py` obsahuje definici GUI a textů, které jsou uživateli zobrazovány.

Rozvržení GUI je definováno třídou `SudokuGUI`, která je potomkem třídy `QWidget` knihovny `PyQt5`. Tato třída se stará o uspořádání, vzhled a obsah všech položek v okně. Další pomocnou třídou, která se stará o správnou funkcionalitu textových polí, je třída `NoLeadingZeroValidator`, která je potomkem `QIntValidator`, jejíž účelem je zakázat jiné znaky v textových polích, než jsou čísla 1 až 9.

Dále jsou v modulu definovány české texty, které jsou navrženy tak, aby uživatel dostával odpovědi v přirozeném jazyce. Dále je implementována metoda pro zobrazení chyby vyskakovacím oknem.

## 4.5 Vysvětlovací modul

Modul `VystvetlovaciModul.py` slouží k vysvětlení řešení sudoku. Jsou implementovány dva druhy vysvětlování, a to vysvětlení postupu a vysvětlení dosazení hodnoty na určité pole.

Postup je vysvětlen funkcí `explain_solution_procedure`. Parametrem je seznam, který obsahuje potřebné informace o postupu řešení sudoku. Metoda data upraví a pomocí komunikačního modulu zobrazí postup převedený do přirozeného jazyka.

Vysvětlení, proč je hodnota přiřazena do určitého pole, je vysvětleno funkcí `explain_tile_value_choice`, která využívá parametry souřadnice a sudoku. Při vysvětlování se simuluje řešení sudoku, přičemž se dostaneme do kroku, kdy byla dosazena hodnota pole, které uživatel zadal. V rámci vysvětlení se vypíše do pole, určeného k vysvětlení, stav sudoku před krokem než se hodnota přiřadila, dále jaké hodnoty jsou sousedy našeho pole,

které hodnoty mohou být doplněna vzhledem k pravidlům hry a následné rozhodnutí algoritmu s vysvětlením rozhodnutí.

# 5 Uživatelská příručka

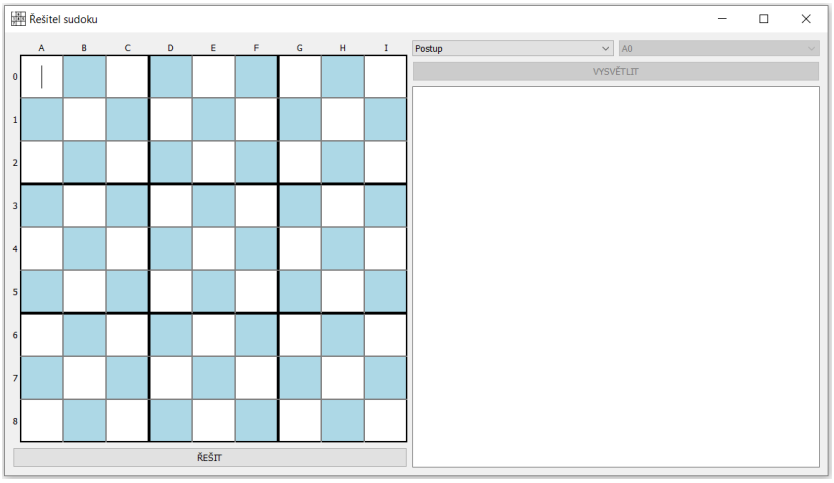
## 5.1 Spuštění aplikace

Aplikaci lze spustit dvěma způsoby. První způsob je přes spustitelný soubor `SudokuSolver.exe`. Tento soubor s příponou `.exe` byl vytvořen pomocí knihovny `PyInstaller` [8] a obsahuje všechny potřebné knihovny ke spuštění, takže uživatel nemusí mít `Python` a jeho knihovny ve svém počítači.

Druhý způsob je spuštění přes příkazovou řádku. Pro úspěšné spuštění aplikace je potřeba mít stažený `Python` a knihovnu `PyQt5`, bez těchto předpokladů bude pokus o spuštění programu neúspěšný. Při splnění předpokladů se aplikace spustí pomocí příkazu:

```
python main.py
```

Při úspěšném spuštění se nám otevře okno s aplikací.

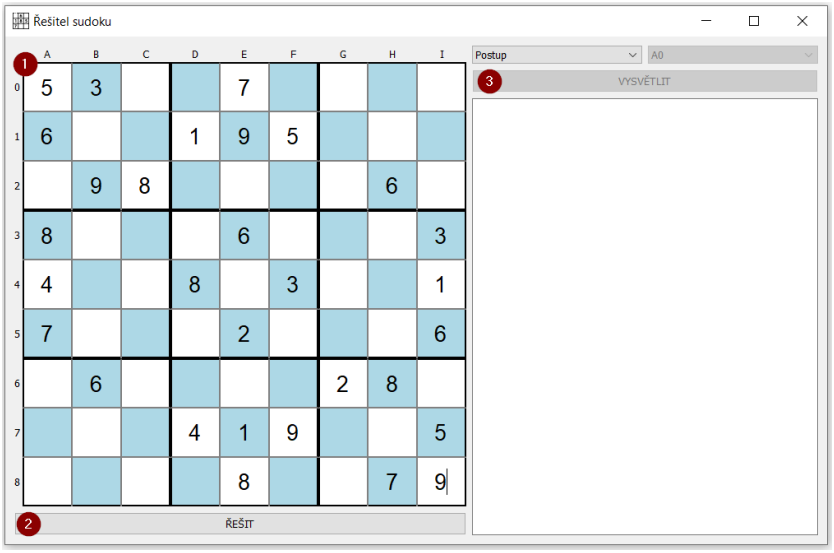



Obrázek 5: Vzhled aplikace při úspěšném spuštění

## 5.2 Ovládání aplikace

Aplikace se může vyskytovat ve dvou režimech, a to režimu řešícím a vysvětlovacím.

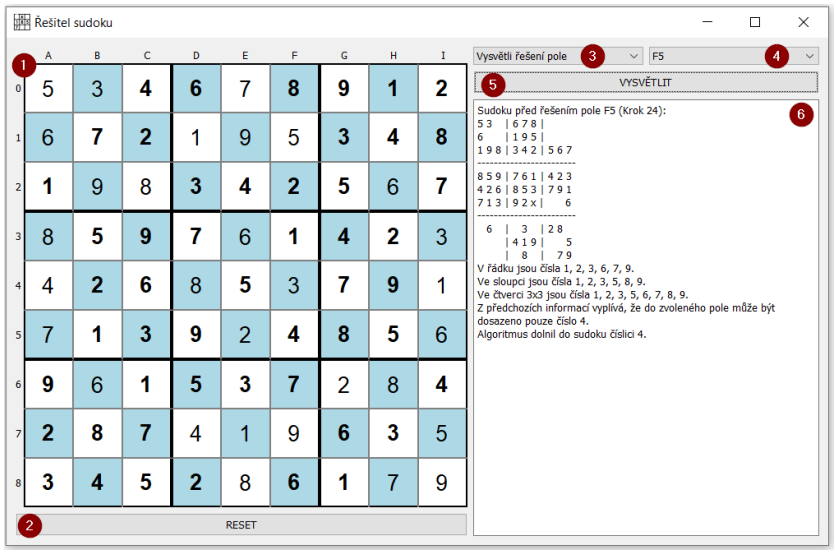
Režim řešení je stav aplikace, kdy ještě sudoku není vyřešeno. V tomto režimu je zablokováno vysvětlování, protože není co vysvětlovat. Uživatel v tomto režimu aplikace může zadávat čísla do mřížky a následně stisknout tlačítko `ŘEŠIT`, které vyřeší sudoku a doplní zbývající pole v mřížce. Vyřešením sudoku se aplikace dostává do režimu vysvětlování.




Obrázek 6: Aplikace v režimu řešení

1. Mřížka sudoku, zde uživatel zadává zadání sudoku, které chce vyřešit.
2. Tlačítko `ŘEŠIT` vyřeší sudoku, pokud má sudoku řešení.
3. Vysvětlovací část je zakázána, neboť zatím není co vysvětlovat.

Režim vysvětlování je stav aplikace, kdy je sudoku vyřešené. V tomto je zablokováno psaní do mřížky. Pokud chce uživatel vyřešit jiné sudoku, musí zmáčknout tlačítko `RESET`, které vymaže dosavadní řešení a vyčistí mřížku. Tento režim je zaměřen primárně na vysvětlování v pravé části okna. Zde se nachází dva seznamy. První seznam obsahuje popis toto, co se bude vysvětlovat. V aktuální verzi lze vysvětlit postup řešení (Postup) nebo vysvětlení proč do daného pole přiřadila určitá hodnota (Vysvětli řešení pole). Ve druhé zmíněné metodě vysvětlování se též zpřístupní druhý seznam se souřadnicemi. V tomto seznamu si uživatel vybere, které řešení pole chce vysvětlit. Pro zobrazení vysvětlení musí uživatel zmáčknout tlačítko `VYSVĚTLIT`, po zmáčknutí tohoto tlačítka se vygeneruje příslušný text, který se poté zobrazí ve velkém textovém poli pod tlačítkem.




Obrázek 7: Aplikace v režimu vysvětlování

1. Mřížka s řešením sudoku. Číslice vyřešené znalostním systémem jsou zvýrazněny tučně. Je zakázáno upravovat obsah mřížky.
2. Tlačítko `RESET` vymaže aktuální řešení sudoku a vyčistí mřížku.
3. List s požadavky, co chceme vysvětlit. V aktuální verzi lze vysvětlit postup, nebo řešení konkrétního pole.
4. List se souřadnicemi. Tento list je zpřístupněn pouze pokud chceme vysvětlit konkrétní pole.
5. Tlačítko `VYSVĚTLIT` vygeneruje text s vysvětlením a zobrazí jej uživateli.
6. Textové pole s vysvětlením.

# 6 Závěr

V rámci semestrální práce byla vytvořena aplikace znalostního systému pro řešení sudoku. Výsledné řešení bylo vytvořeno se snahou udělat algoritmus obarvení grafu co nejvíce efektivní, čímž jsem se trošku odchýlil od zadání, neboť nedochází k náhodnému obarvení, ale k systematickému obarvování všech vrcholů. Aplikace je dekomponována do několika modulů, díky kterým je aplikace dobře modifikovatelná pro další úpravy. Další modifikací programu by mohlo být přidání lokalizace (jiných jazyků), krokování sudoku

při řešení, přidání herního módu a další. Aplikace byla testována testovými scénáři a ošetřena na nečekané vstupy od uživatele.

# Reference


[1] Vaclav Matousek. *Zadání semestrální práce*. Zář. 2023. URL: https://www.kiv.zcu.cz/studies/predmety/uzi/Zadani7/Zadani7_2023.pdf.

[2] *Sudoku and Graph Coloring — NetworkX Notebooks*. URL: https://networkx.org/nx-guides/content/generators/sudoku.html.

[3] Henri Maltby, Khang Nguyen Thanh a Christopher Williams. *Graph Coloring and Chromatic Numbers*. URL: https://brilliant.org/wiki/graph-coloring-and-chromatic-numbers/.

[4] Zdeněk Ryjáček. *Teorie grafů a diskrétní optimalizace 1*. Led. 2007. URL: http://www.kma.zcu.cz/TGD1.

[5] Ishaan Gupta. *Sudoku solver - graph coloring*. Srp. 2020. URL: https://medium.com/code-science/sudoku-solver-graph-coloring-8f1b4df47072.

[6] GeeksforGeeks. *Backtracking Algorithms*. Zář. 2023. URL: https://www.geeksforgeeks.org/backtracking-algorithms/.

[7] *PyQt5 5.15.10*. Říj. 2023. URL: https://pypi.org/project/PyQt5/.

[8] *PyInstaller 6.2.0 documentation*. URL: https://pyinstaller.org/en/stable/.


# Seznam obrázků