Úvod do umělé inteligence (NAIL120)

3. cvičení

Jirka Fink

https://ktiml.mff.cuni.cz/~fink/

Katedra teoretické informatiky a matematické logiky
Matematicko-fyzikální fakulta
Univerzita Karlova v Praze

Letni semestr 2021/22

Poslední změna 3. března 2022

1. domácí úkol: A* heuristiky

Zadání (zkráceno)

Implementujte **monotónní** heuristiky pro **A*** algoritmus spuštěný na **podgrafy** následujících nekonečných mřížek.

- Klasická dvourozměrná mřížka
- Klasická třírozměrná mřížka
- Dvourozměrná mřížka obsahující i úhlopříčky
- Třírozměrná mřížka obsahující stěnové i prostorové úhlopříčky
- Třírozměrná mřížka obsahující stěnové úhlopříčky ale nikoliv prostorové
- Hrany odpovídají právě pohybům koně po šachovnici

Hledání specializovaných algoritmů na konkrétní úlohy

Hledání specializovaných algoritmů na konkrétní úlohy

- Dijkstrův algoritmus na hledání nejkratší cesty
- Borůvkův algoritmus na hledání minimální kostry
- Aho-Corasic algoritmus na vyhledávání v textu
- Dinicův algoritmus na hledání maximálního toku
- Strassenův algoritmus na násobení matic

Přístupy k řešení algoritmických úloh

Hledání specializovaných algoritmů na konkrétní úlohy

- Dijkstrův algoritmus na hledání nejkratší cesty
- Borůvkův algoritmus na hledání minimální kostry
- Aho-Corasic algoritmus na vyhledávání v textu
- Dinicův algoritmus na hledání maximálního toku
- Strassenův algoritmus na násobení matic

Nebylo by lepší mít obecný algoritmus na řešení podobných úloh?

Přístupy k řešení algoritmických úloh

Hledání specializovaných algoritmů na konkrétní úlohy

- Dijkstrův algoritmus na hledání nejkratší cesty
- Borůvkův algoritmus na hledání minimální kostry
- Aho-Corasic algoritmus na vyhledávání v textu
- Dinicův algoritmus na hledání maximálního toku
- Strassenův algoritmus na násobení matic

Nebylo by lepší mít obecný algoritmus na řešení podobných úloh?

Je možné na každou úlohu najít algoritmus?

Přístupy k řešení algoritmických úloh

Hledání specializovaných algoritmů na konkrétní úlohy

- Dijkstrův algoritmus na hledání nejkratší cesty
- Borůvkův algoritmus na hledání minimální kostry
- Aho-Corasic algoritmus na vyhledávání v textu
- Dinicův algoritmus na hledání maximálního toku
- Strassenův algoritmus na násobení matic

Nebylo by lepší mít obecný algoritmus na řešení podobných úloh?

Je možné na každou úlohu najít algoritmus?

- Algoritmů je spočetně mnoho

Přístupy k řešení algoritmických úloh

Hledání specializovaných algoritmů na konkrétní úlohy

- Dijkstrův algoritmus na hledání nejkratší cesty
- Borůvkův algoritmus na hledání minimální kostry
- Aho-Corasic algoritmus na vyhledávání v textu
- Dinicův algoritmus na hledání maximálního toku
- Strassenův algoritmus na násobení matic

Nebylo by lepší mít obecný algoritmus na řešení podobných úloh?

Je možné na každou úlohu najít algoritmus?

- Algoritmů je spočetně mnoho
- Rozhodovacích úloh je nespočetně mnoho

Přístupy k řešení algoritmických úloh

Hledání specializovaných algoritmů na konkrétní úlohy

- Dijkstrův algoritmus na hledání nejkratší cesty
- Borůvkův algoritmus na hledání minimální kostry
- Aho-Corasic algoritmus na vyhledávání v textu
- Dinicův algoritmus na hledání maximálního toku
- Strassenův algoritmus na násobení matic

Nebylo by lepší mít obecný algoritmus na řešení podobných úloh?

Je možné na každou úlohu najít algoritmus?

- Algoritmů je spočetně mnoho
- Rozhodovacích úloh je nespočetně mnoho
- Pro většinu úloh neexistuje algoritmus

Přístupy k řešení algoritmických úloh

Hledání specializovaných algoritmů na konkrétní úlohy

- Dijkstrův algoritmus na hledání nejkratší cesty
- Borůvkův algoritmus na hledání minimální kostry
- Aho-Corasic algoritmus na vyhledávání v textu
- Dinicův algoritmus na hledání maximálního toku
- Strassenův algoritmus na násobení matic

Nebylo by lepší mít obecný algoritmus na řešení podobných úloh?

Je možné na každou úlohu najít algoritmus?

- Algoritmů je spočetně mnoho
- Rozhodovacích úloh je nespočetně mnoho
- Pro většinu úloh neexistuje algoritmus
- Příklad: Neexistuje algoritmus, který rozhodne, zda se daný program zastaví

Obecnější postupy řešení algoritmických úloh

Jak zkonstruovat postup na řešení obecnější třídy úloh?

- Zvolit rozumnou třídu úloh
- Vymyslet zápis úloh této třídy
- Najít obecný algoritmus na řešení těchto úloh

Obecnější postupy řešení algoritmických úloh

Jak zkonstruovat postup na řešení obecnější třídy úloh?

- Zvolit rozumnou třídu úloh
- Vymyslet zápis úloh této třídy
- Najít obecný algoritmus na řešení těchto úloh

Příklady tříd úloh

- Lineární programování

Obecnější postupy řešení algoritmických úloh

Jak zkonstruovat postup na řešení obecnější třídy úloh?

- Zvolit rozumnou třídu úloh
- Vymyslet zápis úloh této třídy
- Najít obecný algoritmus na řešení těchto úloh

Příklady tříd úloh

- Lineární programování
- Konvexní optimalizace

Obecnější postupy řešení algoritmických úloh

Jak zkonstruovat postup na řešení obecnější třídy úloh?

- Zvolit rozumnou třídu úloh
- Vymyslet zápis úloh této třídy
- Najít obecný algoritmus na řešení těchto úloh

Příklady tříd úloh

- Lineární programování
- Konvexní optimalizace
- Constraint satisfaction programming (splňování podmínek)

Obecnější postupy řešení algoritmických úloh

Jak zkonstruovat postup na řešení obecnější třídy úloh?

- Zvolit rozumnou třídu úloh
- Vymyslet zápis úloh této třídy
- Najít obecný algoritmus na řešení těchto úloh

Příklady tříd úloh

- Lineární programování
- Konvexní optimalizace
- Constraint satisfaction programming (splňování podmínek)
- SAT (splnitelnost logických formulí)

Obecnější postupy řešení algoritmických úloh

Jak zkonstruovat postup na řešení obecnější třídy úloh?

- Zvolit rozumnou třídu úloh
- Vymyslet zápis úloh této třídy
- Najít obecný algoritmus na řešení těchto úloh

Příklady tříd úloh

- Lineární programování
- Konvexní optimalizace
- Constraint satisfaction programming (splňování podmínek)
- SAT (splnitelnost logických formulí)
- Automatické plánování

Popis úlohy pomocí CSP

- Konečná množina proměnných

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných
  - Podmínkou je, aby k-tice hodnot proměnných ležela v této podmnožině

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných
  - Podmínkou je, aby k-tice hodnot proměnných ležela v této podmnožině
- Řešiče CSP hledají ohodnocení proměnných splňující všechny podmínky

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných
  - Podmínkou je, aby k-tice hodnot proměnných ležela v této podmnožině
- Řešiče CSP hledají ohodnocení proměnných splňující všechny podmínky
- Některé řešiče povolují jen některé typy podmínek
  Python-constraint, IBM CPLEX CP, Google OR-tools

Constraint satisfaction programming (CSP)

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných
  - Podmínkou je, aby k-tice hodnot proměnných ležela v této podmnožině
- Řešiče CSP hledají ohodnocení proměnných splňující všechny podmínky
- Některé řešiče povolují jen některé typy podmínek
  Python-constraint, IBM CPLEX CP, Google OR-tools

Cvičení: Popište sudoku jako úlohu CSP

Constraint satisfaction programming (CSP)

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných
  - Podmínkou je, aby k-tice hodnot proměnných ležela v této podmnožině
- Řešiče CSP hledají ohodnocení proměnných splňující všechny podmínky
- Některé řešiče povolují jen některé typy podmínek
  Python-constraint, IBM CPLEX CP, Google OR-tools

Cvičení: Popište sudoku jako úlohu CSP

- Proměnné $x_{ij} \in \{1, \ldots, 9\}$ pro $i, j \in \{1, \ldots, 9\}$

Constraint satisfaction programming (CSP)

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných
  - Podmínkou je, aby k-tice hodnot proměnných ležela v této podmnožině
- Řešiče CSP hledají ohodnocení proměnných splňující všechny podmínky
- Některé řešiče povolují jen některé typy podmínek
  Python-constraint, IBM CPLEX CP, Google OR-tools

Cvičení: Popište sudoku jako úlohu CSP

- Proměnné $x_{ij} \in \{1, \ldots, 9\}$ pro $i, j \in \{1, \ldots, 9\}$
- Podmínky $(x_{ij}, x_{i'j'}) \in \{(a, b);\ a, b \in \{1, \ldots, 9\},\ a \neq b\}$ kdykoliv pozice ij a $i'j'$ mají mít různé hodnoty

Constraint satisfaction programming (CSP)

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných
  - Podmínkou je, aby k-tice hodnot proměnných ležela v této podmnožině
- Řešiče CSP hledají ohodnocení proměnných splňující všechny podmínky
- Některé řešiče povolují jen některé typy podmínek
  Python-constraint, IBM CPLEX CP, Google OR-tools

Cvičení: Popište sudoku jako úlohu CSP

- Proměnné $x_{ij} \in \{1, \ldots, 9\}$ pro $i, j \in \{1, \ldots, 9\}$
- Podmínky $(x_{ij}, x_{i'j'}) \in \{(a, b);\ a, b \in \{1, \ldots, 9\},\ a \neq b\}$ kdykoliv pozice ij a $i'j'$ mají mít různé hodnoty
- Je-li na pozici ij předepsaná hodnota a, tak přidáme podmínku $x_{ij} \in \{a\}$

Constraint satisfaction programming (CSP)

Popis úlohy pomocí CSP

- Konečná množina proměnných
- Každá proměnná má danou konečnou množinu hodnot, kterých může nabývat
- Množina podmínek popsané pomocí
  - k-tice proměnných
  - Podmnožina kartézského součinu množin hodnot těchto proměnných
  - Podmínkou je, aby k-tice hodnot proměnných ležela v této podmnožině
- Řešiče CSP hledají ohodnocení proměnných splňující všechny podmínky
- Některé řešiče povolují jen některé typy podmínek
  Python-constraint, IBM CPLEX CP, Google OR-tools

Cvičení: Popište sudoku jako úlohu CSP

- Proměnné $x_{ij} \in \{1, \ldots, 9\}$ pro $i, j \in \{1, \ldots, 9\}$
- Podmínky $(x_{ij}, x_{i'j'}) \in \{(a, b);\ a, b \in \{1, \ldots, 9\},\ a \neq b\}$ kdykoliv pozice ij a $i'j'$ mají mít různé hodnoty
- Je-li na pozici ij předepsaná hodnota a, tak přidáme podmínku $x_{ij} \in \{a\}$
- Ohodnocení proměnných splňující všechny podmínky odpovídá řešení sudoku a opačně

Perfektní párování

Příklad

- Perfektní párování grafu je podmnožina hran P taková, že každý vrchol má právě jednu incidentní hranu v P

Perfektní párování

Příklad

- Perfektní párování grafu je podmnožina hran P taková, že každý vrchol má právě jednu incidentní hranu v P
- Popište hledání perfektního párování pomocí CSP

Perfektní párování

Příklad

- Perfektní párování grafu je podmnožina hran P taková, že každý vrchol má právě jednu incidentní hranu v P
- Popište hledání perfektního párování pomocí CSP

Řešení pro obecné grafy

- Proměnné $x_e \in \{0, 1\}$ pro všechny hrany e

Perfektní párování

Příklad

- Perfektní párování grafu je podmnožina hran P taková, že každý vrchol má právě jednu incidentní hranu v P
- Popište hledání perfektního párování pomocí CSP

Řešení pro obecné grafy

- Proměnné $x_e \in \{0, 1\}$ pro všechny hrany e
- Podmínka pro každý vrchol u stupně k:
  Hodnota k-tice proměnných hran incidentních s u musí být rovna jednomu z k vektorů $(0, \ldots, 0, 1, 0, \ldots, 0)$

Perfektní párování

Příklad

- Perfektní párování grafu je podmnožina hran P taková, že každý vrchol má právě jednu incidentní hranu v P
- Popište hledání perfektního párování pomocí CSP

Řešení pro obecné grafy

- Proměnné $x_e \in \{0, 1\}$ pro všechny hrany e
- Podmínka pro každý vrchol u stupně k:
  Hodnota k-tice proměnných hran incidentních s u musí být rovna jednomu z k vektorů $(0, \ldots, 0, 1, 0, \ldots, 0)$

Alternativní řešení pro bipartitní grafy

- Máme bipartitní graf mezi vrcholy A a B, kde $|A| = |B|$

Perfektní párování

Příklad

- Perfektní párování grafu je podmnožina hran P taková, že každý vrchol má právě jednu incidentní hranu v P
- Popište hledání perfektního párování pomocí CSP

Řešení pro obecné grafy

- Proměnné $x_e \in \{0, 1\}$ pro všechny hrany e
- Podmínka pro každý vrchol u stupně k:
  Hodnota k-tice proměnných hran incidentních s u musí být rovna jednomu z k vektorů $(0, \ldots, 0, 1, 0, \ldots, 0)$

Alternativní řešení pro bipartitní grafy

- Máme bipartitní graf mezi vrcholy A a B, kde $|A| = |B|$
- Proměnná $x_a \in N(a)$ udává vrchol z B, se kterým je $a \in A$ spárovaný, kde $N(a) \subseteq B$ jsou sousední vrcholy a

Perfektní párování

Příklad

- Perfektní párování grafu je podmnožina hran P taková, že každý vrchol má právě jednu incidentní hranu v P
- Popište hledání perfektního párování pomocí CSP

Řešení pro obecné grafy

- Proměnné $x_e \in \{0, 1\}$ pro všechny hrany e
- Podmínka pro každý vrchol u stupně k:
  Hodnota k-tice proměnných hran incidentních s u musí být rovna jednomu z k vektorů $(0, \ldots, 0, 1, 0, \ldots, 0)$

Alternativní řešení pro bipartitní grafy

- Máme bipartitní graf mezi vrcholy A a B, kde $|A| = |B|$
- Proměnná $x_a \in N(a)$ udává vrchol z B, se kterým je $a \in A$ spárovaný, kde $N(a) \subseteq B$ jsou sousední vrcholy a
- Pro každou dvojici různých vrcholů $a, a' \in A$ máme podmínku $x_a \neq x_{a'}$

Další příklady

Testování lidí na covid

- Každé testovací místo má danou kapacitu

Další příklady

Testování lidí na covid

- Každé testovací místo má danou kapacitu
- Každý člověk smí jet jen do některých testovacích míst, aniž by porušil nařízení premiéra

Další příklady

Testování lidí na covid

- Každé testovací místo má danou kapacitu
- Každý člověk smí jet jen do některých testovacích míst, aniž by porušil nařízení premiéra
- Popište hledání přiřazení lidí testovacím místům pomocí CSP

Další příklady

Testování lidí na covid

- Každé testovací místo má danou kapacitu
- Každý člověk smí jet jen do některých testovacích míst, aniž by porušil nařízení premiéra
- Popište hledání přiřazení lidí testovacím místům pomocí CSP

Souvislost

Rozhodnout, zda dané dva vrcholy leží ve stejné komponentě grafu

Logické puzzle

V pěti domech pěti barev žijí lidé pěti národností pěti chutí s pěti mazlíčky . . . .
Pomocí CSP zjistěte kdo kde s kým . . . žije, víte-li například

- Angličan žije v červeném domě.
- Španěl chová psa.
- Nor žije v nejlevějším domě.
- Zelený dům je vedle domu s liškou.
- Kit Kats se jí v žlutém domě.
- Nor žije v pravo od modrého domu.
- Chovatel hadů má rád Snickers.
- Čech pije pivo.
- . . .

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$
- Ale CSP umí řešit jen rozhodovací problémy

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$
- Ale CSP umí řešit jen rozhodovací problémy
- Předpokládejme, že umíme pomocí CSP popsat problém nalezení hledání $x \in M$ splňující $f(x) \geq A$ pro libovolné A

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$
- Ale CSP umí řešit jen rozhodovací problémy
- Předpokládejme, že umíme pomocí CSP popsat problém nalezení hledání $x \in M$ splňující $f(x) \geq A$ pro libovolné A
- Maximální $x \in M$ najdeme opakováným řešením rozhodnovací úlohy pomocí
  - postupného zmenšování/zvětšování A

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$
- Ale CSP umí řešit jen rozhodovací problémy
- Předpokládejme, že umíme pomocí CSP popsat problém nalezení hledání $x \in M$ splňující $f(x) \geq A$ pro libovolné A
- Maximální $x \in M$ najdeme opakováným řešením rozhodnovací úlohy pomocí
  - postupného zmenšování/zvětšování A
  - binárního půlení

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$
- Ale CSP umí řešit jen rozhodovací problémy
- Předpokládejme, že umíme pomocí CSP popsat problém nalezení hledání $x \in M$ splňující $f(x) \geq A$ pro libovolné A
- Maximální $x \in M$ najdeme opakováným řešením rozhodnovací úlohy pomocí
  - postupného zmenšování/zvětšování A
  - binárního půlení

Plánování on-line schůzek

- Každé schůzky se musí účastnit daná množina lidí

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$
- Ale CSP umí řešit jen rozhodovací problémy
- Předpokládejme, že umíme pomocí CSP popsat problém nalezení hledání $x \in M$ splňující $f(x) \geq A$ pro libovolné A
- Maximální $x \in M$ najdeme opakováným řešením rozhodnovací úlohy pomocí
  - postupného zmenšování/zvětšování A
  - binárního půlení

Plánování on-line schůzek

- Každé schůzky se musí účastnit daná množina lidí
- Schůzky jsou stejně dlouhé a trvají jeden blok

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$
- Ale CSP umí řešit jen rozhodovací problémy
- Předpokládejme, že umíme pomocí CSP popsat problém nalezení hledání $x \in M$ splňující $f(x) \geq A$ pro libovolné A
- Maximální $x \in M$ najdeme opakováným řešením rozhodnovací úlohy pomocí
  - postupného zmenšování/zvětšování A
  - binárního půlení

Plánování on-line schůzek

- Každé schůzky se musí účastnit daná množina lidí
- Schůzky jsou stejně dlouhé a trvají jeden blok
- Všechny schůzky jednoho člověka musí být v různých blocích

Převod na rozhodovací problém

- Chtěli bychom najít řešení $x \in M$ maximalizující funkci $f(x)$
- Ale CSP umí řešit jen rozhodovací problémy
- Předpokládejme, že umíme pomocí CSP popsat problém nalezení hledání $x \in M$ splňující $f(x) \geq A$ pro libovolné A
- Maximální $x \in M$ najdeme opakováným řešením rozhodnovací úlohy pomocí
  - postupného zmenšování/zvětšování A
  - binárního půlení

Plánování on-line schůzek

- Každé schůzky se musí účastnit daná množina lidí
- Schůzky jsou stejně dlouhé a trvají jeden blok
- Všechny schůzky jednoho člověka musí být v různých blocích
- Pomocí CSP najděte plán s minimálním počtem bloků

2. domácí úkol: CSP

Zadání (zkráceno)

Pomocí CSP najděte úplné obarvení grafu (total chromatic index)

- Máme obarvit vrcholy i hrany grafu pomocí minimálního počtu barev
- Každé dva sousední vrcholy musí mít různou barvu
- Každé dvě hrany sdílející společný vrchol musí mít různou barvu
- Hrana a její koncové vrcholy musí mít různou barvu

2. domácí úkol: CSP

Zadání (zkráceno)

Pomocí CSP najděte úplné obarvení grafu (total chromatic index)

- Máme obarvit vrcholy i hrany grafu pomocí minimálního počtu barev
- Každé dva sousední vrcholy musí mít různou barvu
- Každé dvě hrany sdílející společný vrchol musí mít různou barvu
- Hrana a její koncové vrcholy musí mít různou barvu

Knihovny pro Python

- networkx: Knihovna pro práci s grafy
  `https://pypi.org/project/networkx/`
- python-constraint: Triviální řešič CSP
  `https://pypi.org/project/python-constraint/`

2. domácí úkol: CSP

Rady

- Přečtěte si pořádně definici úplného barvení a základní pozorování `https://en.wikipedia.org/wiki/Total_coloring`
- Přečtěte si dokumentaci a příklady na CSP `https://github.com/python-constraint/python-constraint/tree/master/examples`
- Ověřte si, že instalujete knihovnu python-constraint a nikoliv jinou
- Přečtěte si zdrojové kódy k domácí úloze
- Buďte trpělivý, v umělé inteligenci nemusí všechny teoreticky správné postupy fungovat v reálu ideálně
- Cílem úkolu je vyzkoušet si jednoduchý CSP řešič na jednoduché úloze