Pokročilá algoritmizace

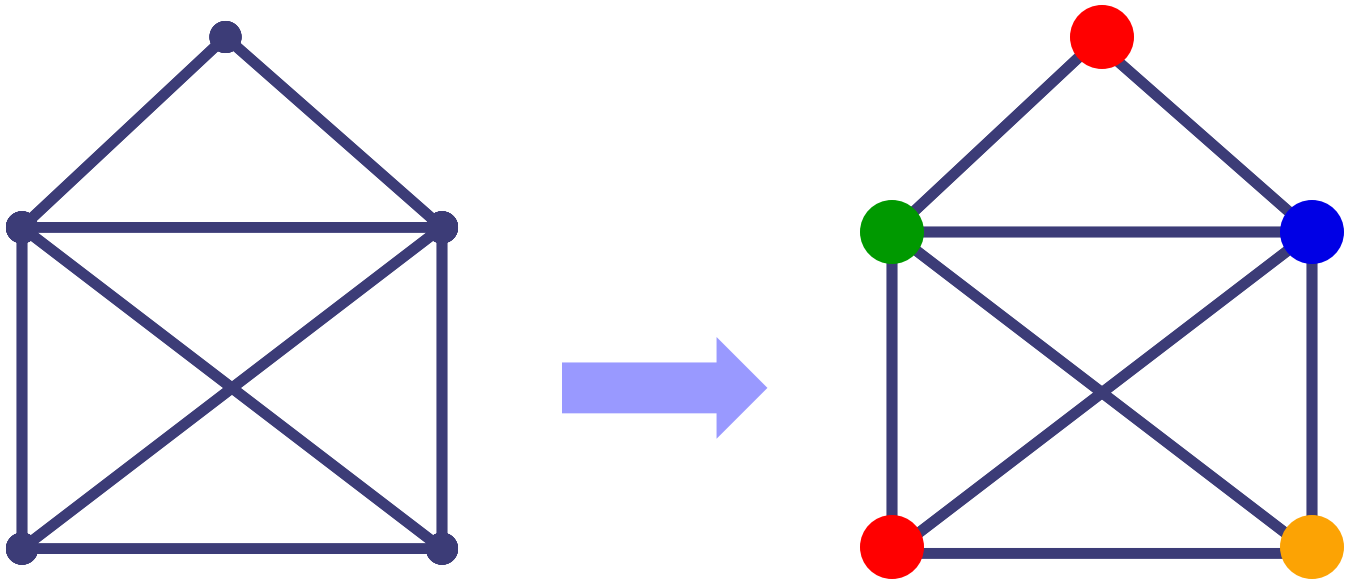
barevnost grafu, rovinné grafy, SAT, CNF, DPLL, souvislost barevnosti se SATem

Jiří Vyskočil, Marko Genyg-Berezovskyj

2010

Barevnost grafu

- Cíl: Nechť G je libovolný neorientovaný graf. Chceme obarvit jeho vrcholy co nejmenším počtem barev tak, aby platilo, že každé dva vrcholy spojené hranou mají různou barvu.



Barevnost grafu

- Nechť $G = (V, E)$ je neorientovaný graf a k přirozené číslo. Zobrazení

$$b: V \to \{1,2, \ldots, k\}$$

nazveme obarvením grafu G pomocí k barev, pokud platí:

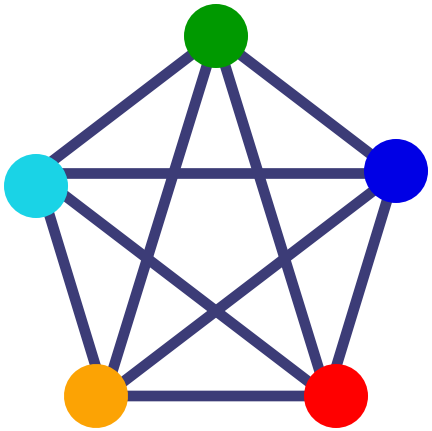
$$\forall \{x, y\} \in E : b(x) \neq b(y)$$

- **Barevnost grafu** (také **chromatické číslo**) G je minimální počet barev potřebný pro obarvení G.

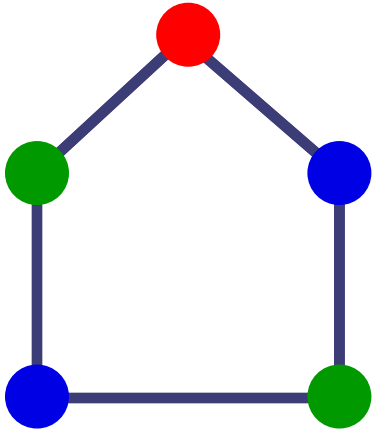
Značí se $\chi(G)$.

Barevnost grafu

- $\chi(G) = 1$ právě tehdy, skládá-li se G z izolovaných vrcholů.
- $\chi(G) = |V|$ pro libovolný úplný graf.



- $\chi(G) \geq 3$ právě tehdy, obsahuje-li G cyklus liché délky.

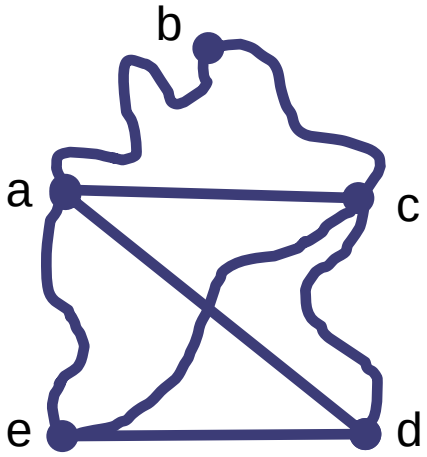


Rovinný graf, rovinné nakreslení

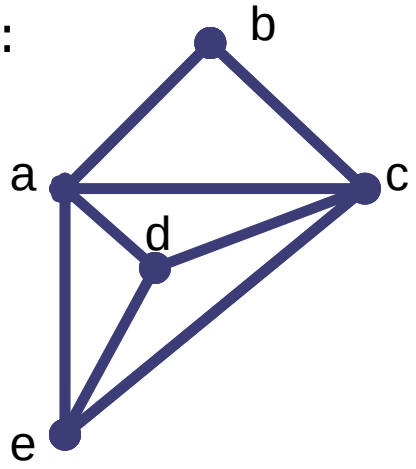
- rovinný graf (planární graf)
  - **Rovinný graf** je graf, pro který existuje takové **rovinné nakreslení**, že se žádné dvě hrany nekříží.
- rovinné nakreslení
  - **Rovinné nakreslení** je zobrazení b, které každému vrcholu v přiřazuje bod roviny $b(v)$ a hraně $\{i, j\}$ přiřadí oblouk s koncovými body $\sigma(i)$ a $\sigma(j)$. Zobrazení je prosté (různým vrcholům odpovídají různé body roviny) a žádný bod $b(v)$ není nekoncovým bodem žádného oblouku. Graf spolu s takovýmto zobrazením nazveme *topologický graf*.
- oblouk
  - **Oblouk** je podmnožina roviny tvaru σ(<0,1>), kde σ : <0,1> $\rightarrow \mathbb{R}^2$ je nějaké spojité a prosté (až na koncové body) zobrazení intervalu <0,1> do roviny. Body σ(0) a σ(1) se nazývají *koncové body* oblouku.
- stěna
  - Nechť A$\subseteq\mathbb{R}^2$ je nějaká podmnožina roviny. Nazveme ji *souvislou*, pokud pro $\forall x,y \in$A platí, že existuje oblouk *o* s koncovými body *x* a *y* takový, že $o \subseteq$ A. Oblouky příslušné hranám nějakého topologického grafu pak podle této relace souvislosti rozdělují rovinu na třídy ekvivalence, které se nazývají **stěny grafu**.

Rovinný graf, rovinné nakreslení

- příklad rovinného nakreslení grafu s křížením hran:



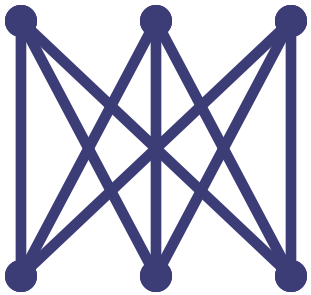
- příklad rovinného nakreslení téhož grafu, ale bez křížení hran (graf je tedy rovinný):



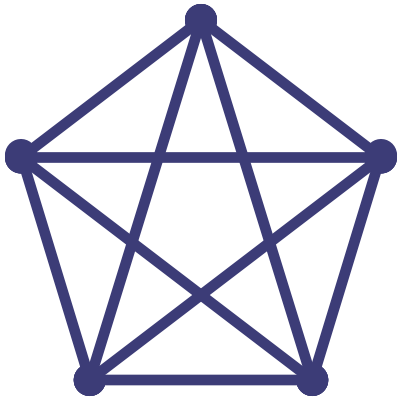
Kuratowského věta

- Graf G je rovinný, právě když žádný jeho podgraf není isomorfní dělení grafu $K_{3,3}$ ani dělení grafu K_5.

- $K_{3,3}$



- K_5



- **Dělení grafu** je graf, který vznikne z původního grafu opakovanou aplikací dělení hrany. **Dělení hrany** lze popsat operací % následovně:

$$G\%e = (V \cup \{z\}, (E \backslash \{\{x, y\}\}) \cup \{\{x, z\}, \{z, y\}\})$$

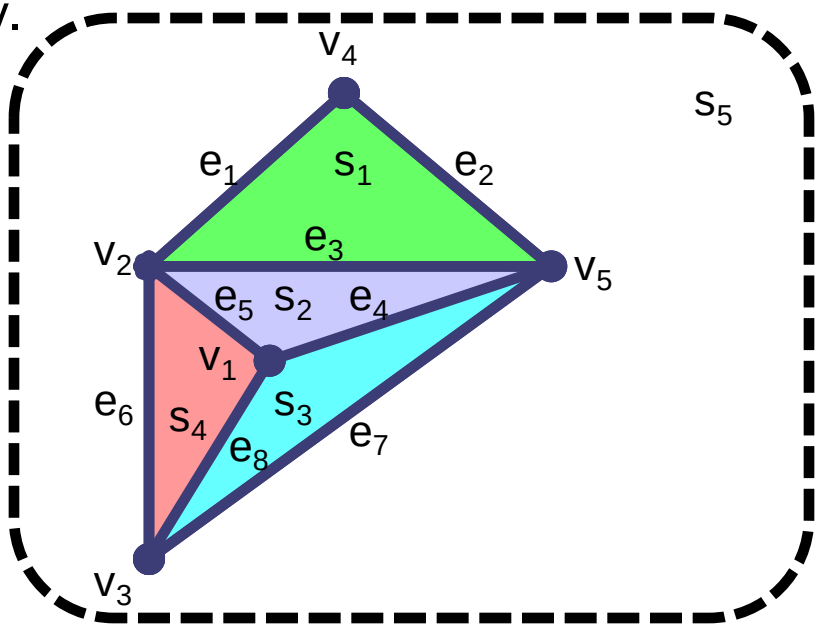
kde $e=\{x,y\} \in E$ je hrana, a $z \notin V$ je nový vrchol (na hranu $\{x,y\}$ tedy „přikreslíme“ nový vrchol z).

Eulerův vzorec

- Eulerův vzorec
  - Nechť $G=(V,E)$ je souvislý rovinný graf, a nechť s je počet stěn nějakého rovinného nakreslení G, kde se nekříží hrany. Potom platí

$$|V| - |E| + s = 2$$

- Důsledek
  - Počet stěn nezávisí na způsobu rovinného nakreslení, kde se nekříží hrany.



Eulerův vzorec

- důkaz indukcí:
  - Pro $|E| = \emptyset$ je potom $|V| = 1$ a $s = 1$. Vzorec tedy platí.
  - Dále rozlišíme dva případy:
    - Graf G neobsahuje cyklus. Potom G je strom a tedy $|V| = |E| + 1$. Každý strom má pouze jedinou stěnu.
    - Graf G obsahuje cyklus. Existuje tedy nějaká hrana $e \in E$, která je obsažena v cyklu. $G - e$ je tedy souvislý. Pro tento graf platí podle indukčního předpokladu Eulerův vzorec:

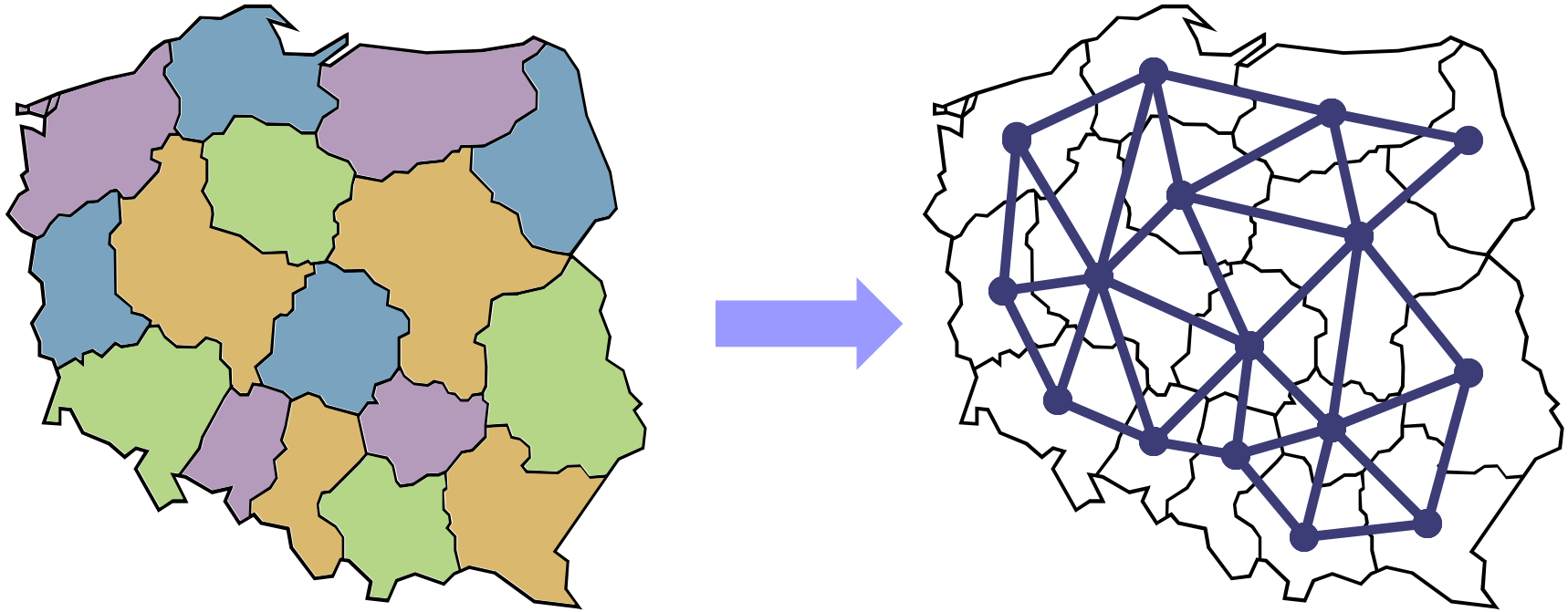
      $|V(G - e)| - |E(G - e)| + s(G - e) = 2.$

      Přidáním hrany e vznikne cyklus který odštěpí novou stěnu. Přidáním 1 hrany tedy vzroste počet stěn o 1. Vzorec tedy platí:

      $|V(G - e)| - (|E(G - e)| + 1) + (s(G - e) + 1) = 2.$

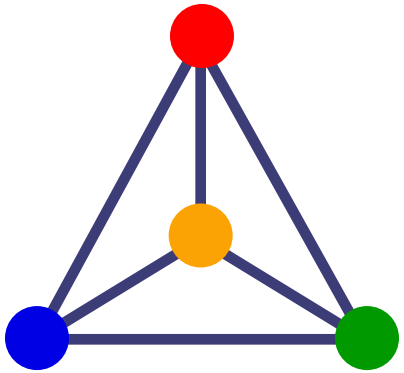
Barevnost rovinných grafů

- problém: Jaké minimální množství barev je potřeba na obarveni libovolné mapy států tak, aby státy se společnou hranicí neměly stejnou barvu?
- převedeme tento problém na barvení rovinného grafu



Barevnost rovinných grafů

- Je zřejmé, že na vyřešení problému potřebujeme alespoň 4 barvy.



- Tento problém byl v roce 1976 s pomocí počítače vyřešen.
- Každý rovinný graf lze obarvit 4 barvami.
  - Je to jeden z prvních výsledků, který byl dokázán s pomocí počítače.
  - Důkaz obsahuje 1936 speciálních případů, které pokrývají všechny možnosti.
- Existuje lineární algoritmus pro obarvení rovinného grafu 5 barvami.

Co je SAT a co nabízí?

- SAT = Boolean SATisfiability problem
- Řeší problém nalezení ohodnocení proměnných v booleovské formuli tak, že je formule splněna.
- Příklad:

  $$\exists x_1 x_2 x_3 x_4 \;\; (x_1 \vee \neg x_2 \vee \neg x_3) \,\&\, (x_1 \vee x_2 \vee x_4)$$

  Řešení:

  Formule je splněna když: (x_1, x_2, x_3, x_4=TRUE)
- Vstupem většiny SAT solverů je CNF (Conjunctive Normal Form).
- Nalezení řešení SAT je NP-úplný problém. To znamená, že zatím nejlepší známý algoritmus řeší úlohu v čase O(exp(n)) (tj. není znám polynomiální algoritmus).

CNF

- CNF (Conjunctive Normal Form)
  - CNF (česky: konjunktivní normální tvar formule) je konjunkce klauzulí, kde klauzule je disjunkce literálů, a kde literál je buď výroková proměnná nebo znegovaná výroková proměnná.

$$\bigwedge_{i \in \text{indexy klauzulí}} \bigvee_{j \in \text{indexy literálů v klauzuli } i} \text{literál}_{i,j}$$

- Příklad CNF: $(x_1 \vee x_3 \vee \neg x_2 \vee x_1) \wedge (\neg x_1 \vee \neg x_4)$
- Každá výroková formule lze převést na CNF.
  - Nejprve převedeme všechny logické spojky na $\wedge$, $\vee$ a $\neg$.
  - Potom pomocí De Morganových pravidel a roznásobení převedeme na CNF.
  - Existuje algoritmus, který umí převést libovolnou výrokovou formuli na jí ekvivalentní v lineárním čase a navíc výsledná CNF formule zabere lineární prostor vůči vstupní formuli.

Jak fungují SAT solvery - DPLL

- Většina nejrychlejších současných SAT solverů používá algoritmus založený na DPLL (Davis-Putnam-Logemann-Loveland) algoritmu:

function DPLL(Φ : CNF) : boolean

    if Φ neobsahuje žádnou klauzuli **then return** true;

    if Φ obsahuje prázdnou klauzuli **then return** false;

    for every *unit clause* L **in** Φ

        Φ := unit-propagate(L, Φ);

    for every *pure literál* L **in** Φ

        Φ := pure-literal-assign(L, Φ);

    L := vyber-literál(Φ);

return DPLL(Φ & L) **or** DPLL(Φ & not(L));

Jak fungují SAT solvery - DPLL

- **unit clause** je klauzule, která obsahuje pouze jeden nepřiřazený literál (literál, který ještě nemá hodnotu).
- **pure literál** je literál, který se v CNF formuli vyskytuje pouze v jedné polaritě (buď jako $\neg x$ nebo jako x).
- **unit-propagate**(L, Φ) a **pure-literal-assign**(L, Φ) nastaví všem literálům L z Φ stejnou hodnotu podle jejich tvaru a zjednoduší výslednou formuli.

  Jinými slovy: Funkce nahradí všechny výskyty L za *true* a opačně polarizované L za *false* a následně zjednoduší výslednou formuli Φ tak, že smaže všechny klauzule obsahující *true* a smaže všechny *false* ze všech klauzulí (Pozor! Prázdná klauzule je stále klauzule)

Jak fungují SAT solvery - DPLL

- Příklad: $(x_1) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_4 \vee \neg x_5) \wedge (x_5 \vee x_4 \vee \neg x_1)$

$(\mathbf{x_1}) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_4 \vee \neg x_5) \wedge (x_5 \vee x_4 \vee \neg x_1)$ - unit-propagate (x_1 , …

$(\mathbf{TRUE}) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_4 \vee \neg x_5) \wedge (x_5 \vee x_4 \vee \mathbf{FALSE})$

$(\neg \mathbf{x_2} \vee x_3) \wedge (\neg x_4 \vee \neg x_5) \wedge (x_5 \vee x_4)$ - pure-literal-assign($\neg x_2$, …

$(\mathbf{TRUE} \vee x_3) \wedge (\neg x_4 \vee \neg x_5) \wedge (x_5 \vee x_4)$

$(\neg \mathbf{x_4} \vee \neg x_5) \wedge (x_5 \vee \mathbf{x_4})$ - vyber-literál $\neg x_4$

$(\neg x_4 \vee \neg x_5) \wedge (x_5 \vee x_4) \wedge (\neg \mathbf{x_4})$ - unit-propagate ($\neg x_4$, …

$(\mathbf{TRUE} \vee \neg x_5) \wedge (x_5 \vee \mathbf{FALSE}) \wedge (\mathbf{TRUE})$

$(\mathbf{x_5})$ - unit-propagate (x_5 , …

$(\mathbf{TRUE})$

$\emptyset$

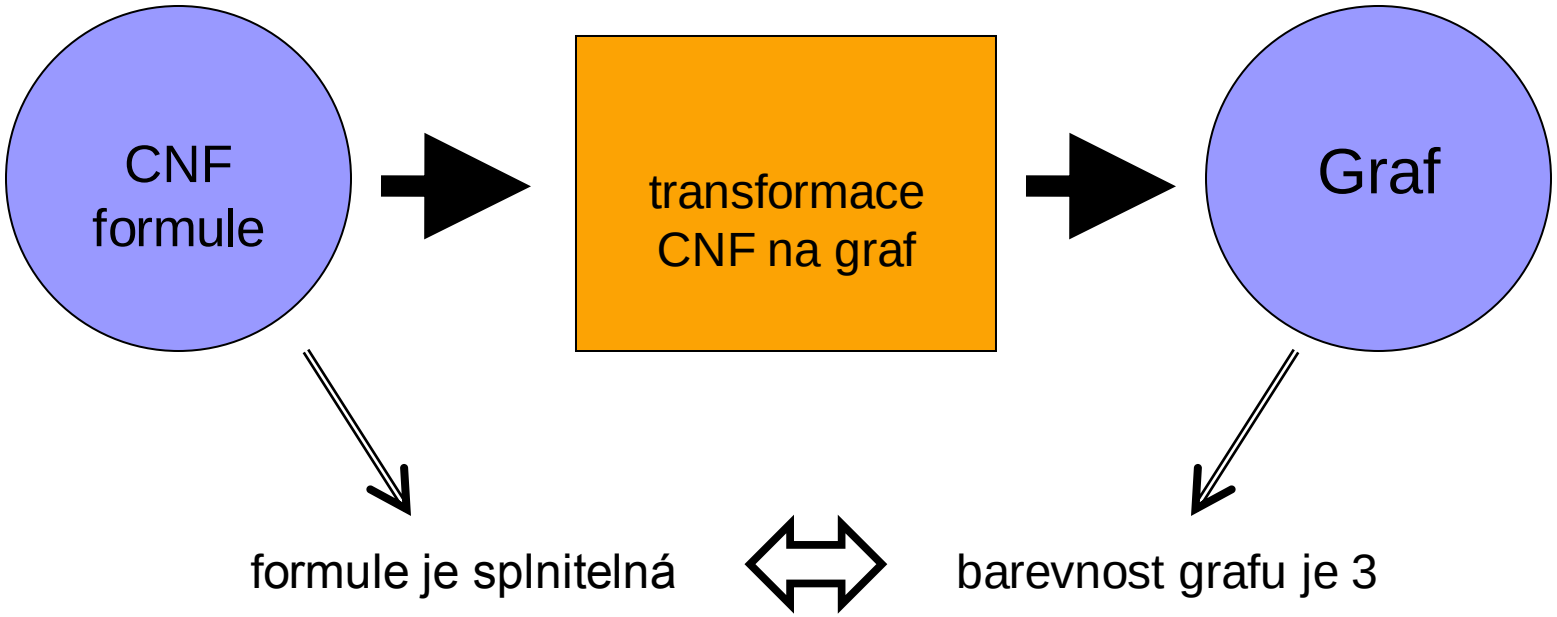
CNF neobsahuje žádnou klauzuli => existuje ohodnocení výrokových proměnných takové, že je formule splněná (x_1=TRUE, x_2=FALSE, x_3=libovolné, x_4=FALSE, x_5=TRUE).

Proč převádět na SAT?

- Dnes existují desítky výkonných SAT solverů (např. MiniSAT, zChaff, …), které se každoročně testují na stovkách testovacích příkladů v soutěžích.
- SAT má dlouhodobě standardizovaný a jednoduchý formát, lze tedy snadno používat více různých solverů.
- Existují preprocesory pro SAT, které na některých typech úloh výrazně zvyšují výkonnost (např. SatELite)
- Existují rozšíření, které řeší i složitější úlohy než SAT (např. Paradox, Equinox, některé QBF solvery, …)

Souvislost SATu a barevnosti

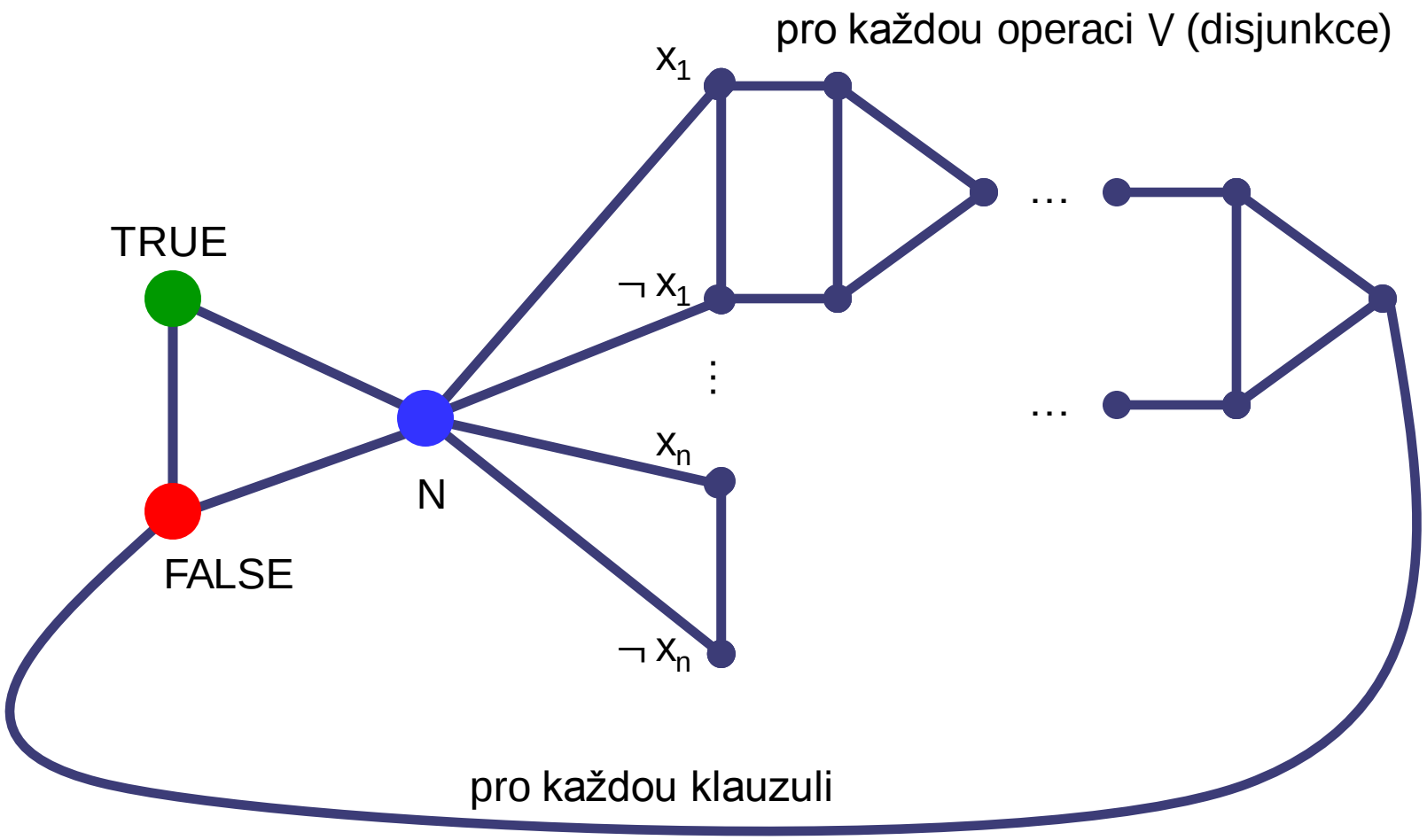
- Souvislost SATu a barevnosti
  - Ke každé boolovské formuli F existuje graf G takový, že pokud je F splnitelná, je barevnost grafu G rovna 3. Navíc jsme tento graf G schopni vygenerovat vůči CNF tvaru formule F v lineárním čase.



  - Důsledek: Kdybychom uměli „rychle“ (např. polynomiálně) řešit *k*-barevnost grafu, potom bychom uměli i „rychle“ řešit SAT (to ale zatím neumíme) výše uvedenou transformací.

Souvislost SATu a barevnosti

- Převod SATu na barvení grafu třemi barvami



Souvislost SATu a barevnosti

- Příklad: $(x_1 \vee x_3 \vee \neg x_2 \vee x_1) \wedge (\neg x_1 \vee \neg x_4)$