Cvičenie 6

Príklad 6.1.

Na vstupe máte konvexný n-uholník, zadaný pomocou súradníc jeho vrcholov. Vstup ďalej obsahuje k bodov v rovine, zadaných pomocou súradníc. Pre každý bod zistite, či leží vo vnútri daného n-uholníka.

Náčrt riešenia 6.1.

Túto úlohu vieme riešiť triviálne v čase $O(nk)$ tak, že pre každý z k bodov zistíme, či leží "naľavo" od každej strany n-uholníka (pomocou vektorového súčinu). Pokúsime sa to riešiť rýchlejšie.

Vyberme si nejaký z vrcholov n-uholníka, a vytvorme[1] si lúče z neho do zvyšných vrcholov (viď obrázok 1). Tieto lúče rozdelia rovinu na $n-1$ sektorov. Všimnite si, že ak vieme, v ktorom sektore sa nachádza bod, tak na to, aby sme zistili, či leží vo vnútri n-uholníka, stačí skontrolovať už len to, či leží "naľavo" od príslušnej strany n-uholníka (prípadne netreba kontrolovať ani to, ak leží "napravo" od najpravejšieho lúča alebo "naľavo" od najľavejšieho lúča).

Čiže, ak poznáme sektor, do ktorého patrí daný bod, tak vieme zodpovedať, či leží vo vnútri, v čase $O(1)$. Vyvstáva otázka, ako rýchlo vieme zistiť, v ktorom sektore leží?

Vieme to urobiť triviálne v čase $O(n)$ tak, že si vyskúšame všetky sektory. To by nám ale nedalo žiadne zrýchlenie oproti pôvodnému riešeniu. Pre zrýchlenie použijeme myšlienku *binárneho vyhľadávania odpovede* (viď obrázok 2). To nám umožní určovať sektor pre daný bod v čase $O(\log n)$. A teda, výsledná časová zložitosť pre k bodov je $O(n + k \log n)$.

(Opakovanie) Polynomiálna redukcia úlohy A na úlohu B ($A \leq_p B$)

```
def solve_A(instance):
    instance_B = transform_instance_A_to_B(instance)
    result = solve_B(instance_B)
    return result

def transform_instance_A_to_B(instance):
    ... # polynomial time transformation

def solve_B(instance):
    ...
```

(Opakovanie) Polynomiálna redukcia a trieda $\mathcal{P}$

Ak vieme polynomiálne redukovať úlohu A na úlohu B, tak platí:

1. Ak úloha B má (deterministicky) polynomiálne riešenie, tak aj úloha A má polynomiálne riešenie (z podstaty redukcie):

$$A \leq_p B \vDash B \in \mathcal{P} \Rightarrow A \in \mathcal{P},$$

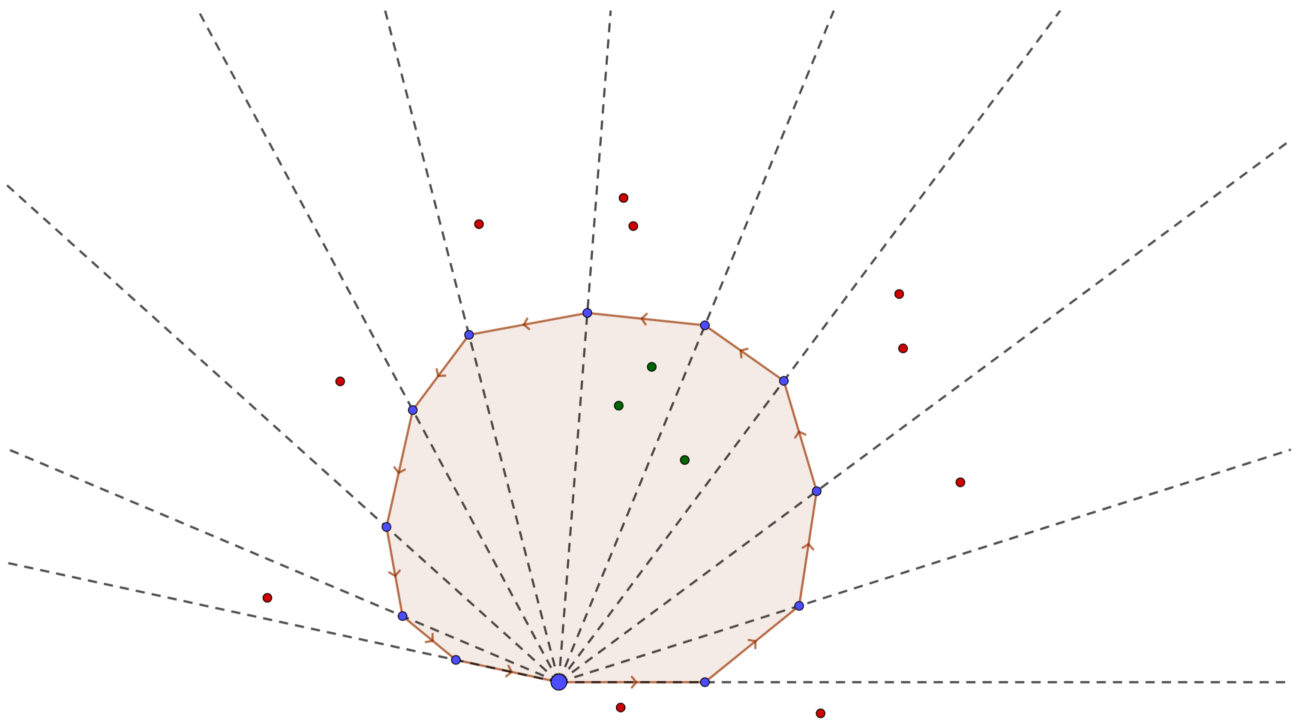
2. Ak úloha A nemá (deterministicky) polynomiálne riešenie, tak ani úloha B nemá polynomiálne riešenie (obratená implikácia):

$$A \leq_p B \vDash A \notin \mathcal{P} \Rightarrow B \notin \mathcal{P}$$

Príklad 6.2.

Dokážte, že platí CLIQUE $\leq_p$ SAT!

[1] skor "predstavme"



Obr. 1: Rozdelenie roviny na sektory podľa daného konvexného n-uholníka

CLIQUE: obsahuje daný graf úplný podgraf (klika) veľkosti aspoň k?

Náčrt riešenia 6.2.

Majme n premenných x_i, pričom x_i je 1 práve vtedy, keď i-tý prvok je v klike. Ďalej nech konštanta $e_{i,j}$ je rovná 1 práve vtedy, keď hrana (i, j) je v grafe. Musíme zaistiť tieto podmienky:

1. Ak dva vrcholy sú v klike, tak medzi nimi musí byť hrana:

$$\forall i, j, i \neq j : (x_i \wedge x_j) \Rightarrow e_{i,j}.$$

   Ekvivalentne (obrátená implikácia), ak hrana (i, j) neexistuje, tak v klike sa nemôžu zároveň nachádzať obidva vrcholy i a j:

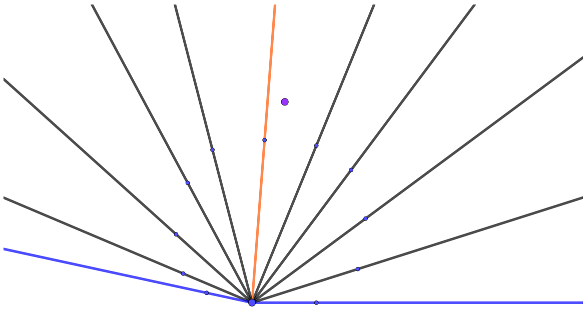
$$\forall (i, j) \notin E : \neg x_i \vee \neg x_j$$

2. Musíme zaistiť, že v klike je aspoň k vrcholov. Naivný spôsob je vymenovať všetkých $\binom{n}{k}$ k-tíc vrcholov, ale výsledný počet klauzúl je exponenciálny.

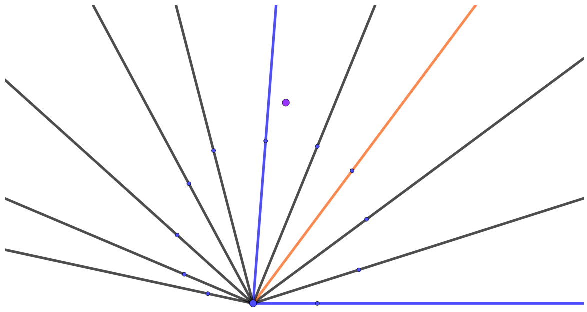
Efektívny spôsob zaistenia veľkosti kliky Nech premenná $s_{i,j}$ ($0 \leq i \leq n$, $0 \leq j \leq k$) je rovná 1 práve vtedy, keď spomedzi prvých i vrcholov je aspoň j v klike. Musíme zaistiť nasledovné podmienky:

- Každý prefix má aspoň 0 vrcholov v klike:

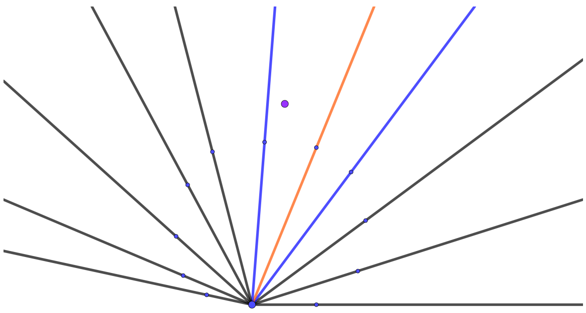
$$\forall i : s_{i,0}$$



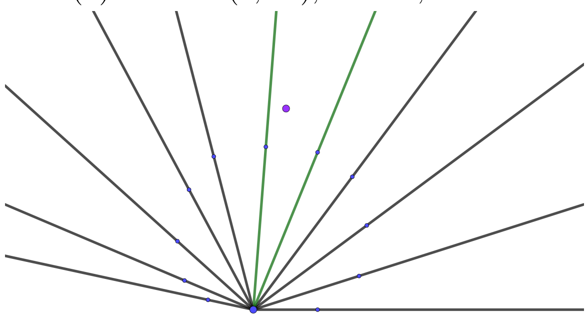
(a) Interval (0, 10), stred 5, napravo



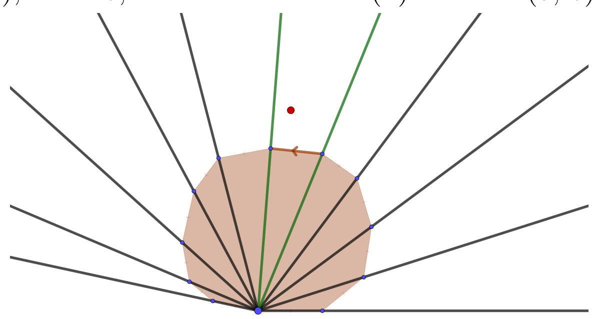
(b) Interval (5, 10), stred 7, naľavo



(c) Interval (5, 7), stred 6, naľavo



(d) Interval (5, 6), sektor je určený



(e) Napravo od hrany, bod je mimo

Obr. 2: Binárne vyhľadávanie sektora

- Nulový prefix nemôže mať kladný počet vrcholov v klike:

$$\forall j \geq 1 : \neg s_{0,j}$$

- Prefix dĺžky i môže mať aspoň j vrcholov v klike práve vtedy, keď o jedna kratší prefix má aspoň j vrcholov, alebo keď o jedna kratší prefix má aspoň $j-1$ vrcholov v klike a i-tý vrchol je v klike:

$$\forall i, j \geq 1 : s_{i,j} \Leftrightarrow s_{i-1,j} \vee (s_{i-1,j-1} \wedge x_i)$$

- Aby sme dosiahli celkový počet vrcholov v klike aspoň k, musíme zaistiť, že premenná $s_{n,k}$ má hodnotu 1:

$$s_{n,k}.$$

Ako previesť $a \Leftrightarrow (b \vee (c \wedge d))$ do CNF?

$$\begin{aligned}
a &\Rightarrow (b \vee (c \wedge d)) \\
\neg a &\vee (b \vee (c \wedge d)) \\
\neg a &\vee ((b \vee c) \wedge (b \vee d)) \\
(\neg a \vee b \vee c) &\wedge (\neg a \vee b \vee d)
\end{aligned}$$

$$\begin{aligned}
a &\Leftarrow (b \vee (c \wedge d)) \\
a &\vee \neg(b \vee (c \wedge d)) \\
a &\vee (\neg b \wedge \neg(c \wedge d)) \\
(a \vee \neg b) &\wedge (a \vee \neg(c \wedge d)) \\
(a \vee \neg b) &\wedge (a \vee \neg c \vee \neg d)
\end{aligned}$$

A teda, CNF pôvodnej formuly je konjunkcia týchto dvoch implikácií:

$$(\neg a \vee b \vee c) \wedge (\neg a \vee b \vee d) \wedge (a \vee \neg b) \wedge (a \vee \neg c \vee \neg d)$$

Príklad 6.3.

Dokážte, že platí 3-SAT $\leq_p$ SUBSET-SUM!
SUBSET-SUM: Existuje podmnožina čísel $s_1, \ldots, s_n$ so súčtom presne t?

Náčrt riešenia 6.3.

Pre každú premennú vytvoríme dve čísla (pre kladnú hodnotu a pre zápornú). Donútime vybrať vždy práve jedno z týchto čísel.

Nech je daná CNF $(a_{1,1} \vee a_{1,2} \vee a_{1,3}) \wedge \ldots \wedge (a_{n,1} \vee a_{n,2} \vee a_{n,3})$, ktorá obsahuje m premenných.

Pre každú premennú u_i vytvoríme čísla v_i a v_i' s $m+n$ číslicami v desiatkovej sústave:

	u_1	u_2	$\ldots$	u_i	$\ldots$	u_m	$C_1 \ldots C_n$
v_i (zodpovedá u_i)	0	0	$\ldots$	1	$\ldots$	0	$C_k = 1$ ak u_i je v k-tej klauzule
v_i' (zodpovedá $\neg u_i$)	0	0	$\ldots$	1	$\ldots$	0	$C_k = 1$ ak $\neg u_i$ je v k-tej klauzule

Platí, že ak vyberieme nejakú podmnožinu, ktorá zodpovedá splňujúcemu priradeniu, tak ich súčet bude mať podobu

u_1	u_2	$\dots$	u_m	C_1	C_2	$\dots$	C_n
1	1	$\dots$	1	≥ 1	≥ 1	$\dots$	≥ 1
				≤ 3	≤ 3	$\dots$	≤ 3

Ako cieľ si zvolíme číslo t:

u_1	u_2	$\dots$	u_m	C_1	C_2	$\dots$	C_n
1	1	$\dots$	1	4	4	$\dots$	4

Pre každú klauzulu pridáme doplnkové čísla pre prípad, že stĺpec danej klauzuly bude mať menej než 4 v súčte:

	u_1	u_2	$\dots$	u_m	C_1	C_2	$\dots$	C_i	$\dots$	C_n
s_i	0	0	$\dots$	0	0	0	$\dots$	1	$\dots$	0
s_i'	0	0	$\dots$	0	0	0	$\dots$	2	$\dots$	0

(Opakovanie) Optimalizačné úlohy a vzťah s triedou $\mathcal{NP}$

Pre minimalizačnú úlohu U vieme definovať *hraničnú* rozhodovaciu úlohu A "Má daný vstup riešenie s cenou nanajvýš k?" (obdobne pre maximalizačné úlohy). Hovoríme, že optimalizačná úloha U je $\mathcal{NP}$-ťažká, ak jej hraničná rozhodovacia úloha A je $\mathcal{NP}$-ťažká.