ΞKSI

Korespondenční Seminář z Informatiky

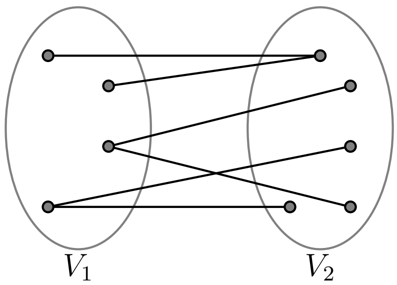
Milé řešitelky, milí řešitelé,

je tu závěrečná sada letošního ročníku. Příklady v této sadě jsou všechny založeny na jednoduchém principu, který se v informatice hojně využívá. Máme-li jeden problém efektivně (nebo třeba zázračně) vyřešený, můžeme na něj převést nějaký jiný problém, jehož podstata dá vyjádřit pomocí už vyřešeného problému. Vyřešený problém vystupuje v našem algoritmu jako *blackbox.*

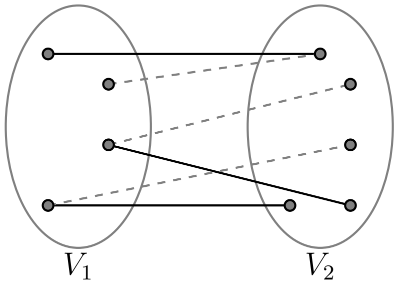
Důkaz korektnosti takového řešení zahrnuje zejména *důkaz, že řešení nalezené pomocí blackboxu skutečně je správným řešením našeho problému* a také naopak *důkaz, že pokud existuje řešení našeho problému, algoritmus používající blackbox ho nepřehlédne.* Hned po představení několika pojmů z teorie grafů, které se Vám budou hodit v této sadě, se dostaneme ke dvěma ukázkám takových převodů.[1]

Pojmy

- *Bipartitní graf* je takový graf, jehož vrcholy se dají rozdělit do dvou množin $V = V_1 \cup V_2$, jejichž průnik $V_1 \cap V_2$ je prázdný, a pro každou hranu platí, že jeden z jejích koncových vrcholů je ve V_1 a druhý ve V_2.



- *Párování velikosti k v bipartitním grafu* je množina obsahující k různých hran, z nichž žádné dvě nemají společný koncový vrchol.

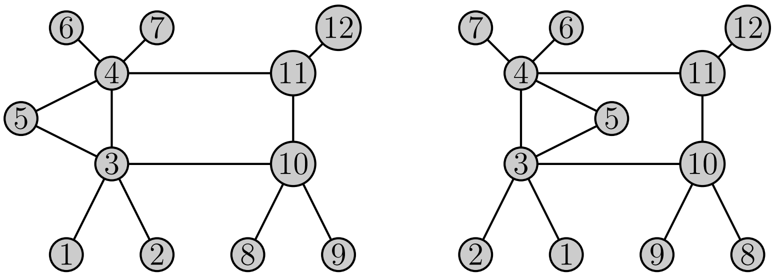


- Řekneme, že určitých k cest v grafu je *disjunktních,* jestliže žádné dvě z těchto cest nemají společnou hranu.

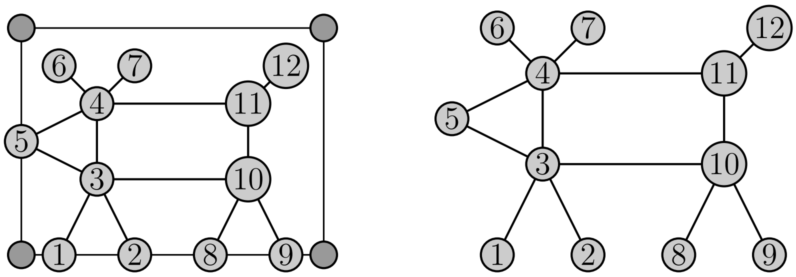
- *Hranové pokrytí* grafu $G = (E, V)$ je podmnožina $E' \subseteq E$ taková, že každý vrchol $v \in V$ je jedním z koncových vrcholů některé hrany z E'.

[1]Zatímco u příkladů v úvodníku jde spíše o vysvětlení principu a může existovat i rychlý algoritmus bez použití blackboxu, v ostatních příkladech blackboxy významně pomůžou s efektivitou:-)

- Hranové pokrytí E' grafu $G = (E, V)$ se nazývá *minimální*, pokud je to hranové pokrytí s minimálním možným počtem prvků (hran).
- Grafy $G_1 = (V_1, E_1)$ a $G_2 = (V_2, E_2)$ jsou *izomorfní*, pokud mají oba grafy stejný počet vrcholů n a existuje očíslování vrcholů prvního grafu čísly $1, \ldots, n$ a očíslování vrcholů druhého grafu těmi stejnými čísly takové, že i-tý a j-tý vrchol v prvním grafu jsou spojené hranou právě tehdy, když i-tý a $j-$tý vrchol ve druhém grafu jsou spojeny hranou.



- Řekneme, že $G_1 = (V_1, E_1)$ *obsahuje podgraf izomorfní grafu* $G_2 = (V_2, E_2)$, pokud existuje $V \subseteq V_1$ a $E \subseteq E_1$ tak, že graf $G = (V, E)$ je izomorfní grafu $G_2 = (V_2, E_2)$.



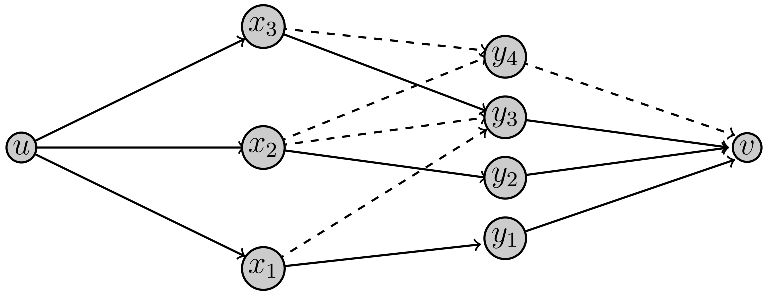
Ukázka s párováním

Představme si, že máme blackbox, který dokáže pro libovolný *orientovaný* graf $G = (V, E)$ s n vrcholy a m hranami, a dva jeho zadané vrcholy $v_1, v_2 \in V$ najít v čase $O(n + m)$ odpověď na otázku Ëxistuje z v_1 do v_2 alespoň k disjunktních cest (tj. posloupností různých orientovaných na sebe navazujících hran) takových, že žádné dvě cesty nemají společnou hranu?"

Ukázkový problém párování. Problém, který chceme vyřešit (a můžeme si k řešení pomoci použitím blackboxu) je Ëxistuje v zadaném bipartitním grafu párování velikosti alespoň k ?"

Řešení problému párování. Problém vyřešíme s použitím blackboxu v čase $O(n+m)$. Stačí ze zadaného bipartitního grafu $G = (V, E)$ s $V = V_1 \cup V_2$ (kde V_1 a V_2 jsou množiny z definice bipartitního grafu) vytvořit nový graf přidáním vrcholů u, v a orientovaných hran (u, x) pro všechna $x \in V_1$, a orientovaných hran (v, y) pro všechna $y \in V_2$. Všechny původní hrany (x, y) zorientujeme směrem z vrcholu z množiny V_1 do vrcholu z množiny V_2.

Na takto vytvořený graf G' (ověřte si, že k jeho vytvoření nám stačí $O(n + m)$ kroků) pak pustíme blackbox a dostaneme odpověď v čase $O(n + 2 + m + n)$. Celkově tedy časová složitost řešení je v $O(3n + 2m) = O(n + m)$.



Důkaz, že ve vytvořeném grafu G' existuje k disjunktních cest z u do v právě tehdy, když v původním grafu existovalo párování velikosti k :

$\Rightarrow$ Pokud v nově vytvořeném grafu existuje k disjunktních cest z u do v, pak každá z těchto cest začíná hranou (u, x_i) z u do k různých vrcholů z V_1. Druhá hrana na této cestě vede z x_i do některého vrcholu y_i z V_2. Z y_i vede jen jediná hrana ven, a to do vrcholu v. Protože cesty jsou disjunktní, máme pro každou z k cest různé první hrany (u, x_i) a různé druhé hrany (x_i, y_i). Také máme různé poslední hrany y_i, v. Když uvážíme množinu k druhých hran těchto cest, víme, že mají různé první vrcholy (protože cesty mají různé první hrany) a mají různé druhé vrcholy (protože cesty mají různé třetí hrany). Máme tedy párování v původním grafu velikosti k.

$\Leftarrow$ Pokud v původním grafu existovalo párování velikosti k, uvažme hrany $e_1 = (x_1, y_1), e_2 = (x_2, y_2), \ldots, e_k = (x_k, y_k)$, které tvoří toto párování (zapsané s prvním vrcholem z V_1 a druhým z V_2). Protože mají navzájem různé koncové vrcholy z V_1 i koncové vrcholy z V_2, můžeme vytvořit k navzájem disjunktních cest z u do v tím, že v i. cestě jsou hrany $(u, x_i), (x_i, y_i)$ a (y_i, v).

Ukázka s hranovým pokrytím

Budeme mít blackbox, který na vstupu bere

- Seznam n proměnných, které můžou nabývat hodnoty 0 a 1. Zatím nevíme, která proměnná bude 0 a která 1.
- Seznam m nerovností, ve kterých vystupuje těchto n proměnných a kde $a_{i,j}$ a b_i jsou celá čísla pro všechna $i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}$. Nerovnosti jsou následujícího tvaru:

$$\begin{aligned} a_{1,1} \cdot x_1 \ldots a_{1,n} \cdot x_n &\geq b_1, \\ a_{2,1} \cdot x_1 \ldots a_{2,n} \cdot x_n &\geq b_2, \\ &\vdots \\ a_{m,1} \cdot x_1 \ldots a_{m,n} \cdot x_n &\geq b_m. \end{aligned}$$

- Seznam n celých čísel $c_1, c_2, \ldots, c_n$.

Blackbox v čase $O(n + m)$ najde ohodnocení $x_1, \ldots, x_n$ (přiřazení nul a jedniček), které splňuje všech m nerovností a pro které je součet $c_1 \cdot x_1 + c_2 \cdot x_2 + \ldots + c_n \cdot x_n$ minimální. Takových ohodnocení může být i víc, blackbox vrátí jedno z nich. Může se také stát, že uvedených m nerovností nelze splnit. Pak blackbox řekne, že řešení neexistuje.

Příklad použití blackboxu: Mějme proměnné x_1, x_2, nerovnosti

$$\begin{aligned} 1 \cdot x_1 + 3 \cdot x_2 &\geq 0 \\ (-3) \cdot x_1 + 0 \cdot x_2 &\geq -2 \end{aligned}$$

a $c_1 = 2, c_2 = -5$. Blackbox by řekl, že x_1 musí být 0 (kvůli druhé nerovnosti) a x_2 může být 0 i 1 (první nerovnost platí vždy). Hledáme ohodnocení, při kterém je $c_1 \cdot x_1 + c_2 \cdot x_2$ minimální. Možné hodnoty jsou $2 \cdot 0 + (-5) \cdot 0 = 0$ a $2 \cdot 0 + (-5) \cdot 1 = -5$. Minimální hodnoty tedy dosáhneme s $x_1 = 0$ a $x_2 = 1$.

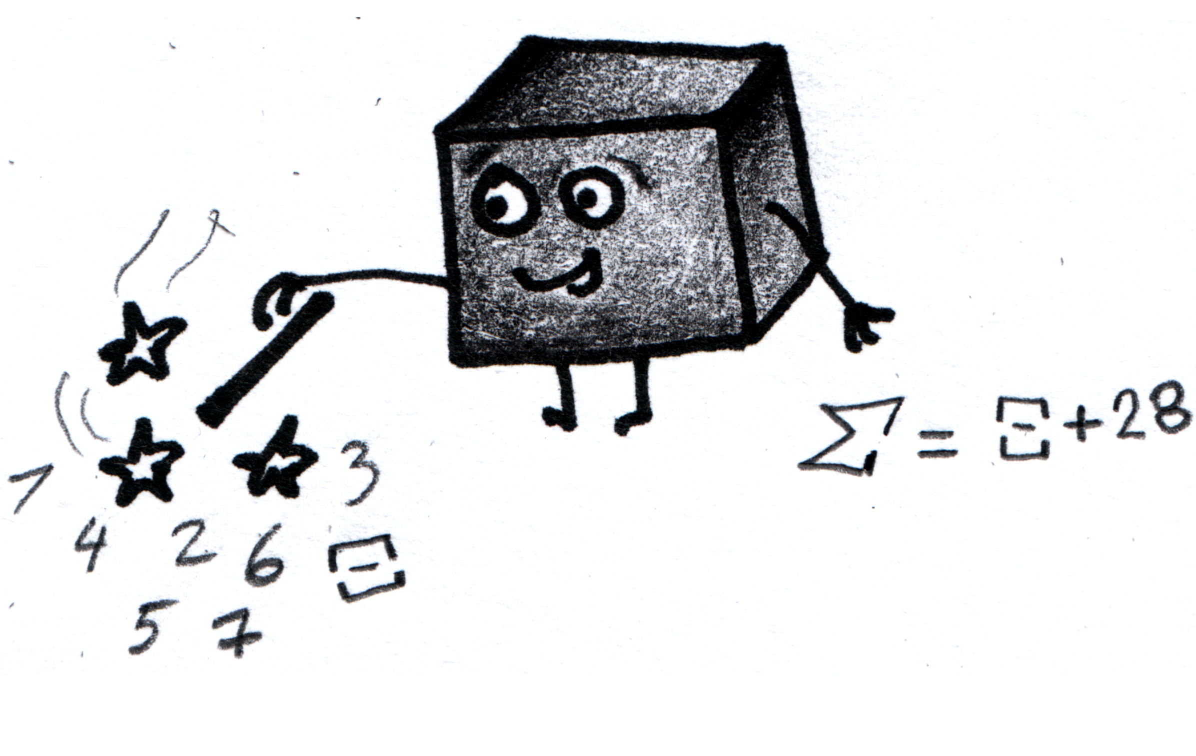
Ukázkový problém hranového pokrytí. Problém, který chceme vyřešit (pomocí blackboxu) je "Která množina hran zadaného grafu $G = (V, E)$ (s k vrcholy a p hranami) tvoří jeho minimální hranové pokrytí?"

Řešení problému hranového pokrytí. Vyřešíme tento problém pomocí našeho blackboxu v čase $O(p + k)$. Pro zadaný graf G uvážíme proměnné $x_1, \ldots, x_p$ (pro každou hranu jednu proměnnou nabývající hodnoty 0 nebo 1 podle toho, jestli byla hrana vybraná do hranového pokrytí nebo ne). Dále uvažme k nerovností tvaru

$$a_{i,1} \cdot x_1 + a_{i,2} \cdot x_2 + \ldots + a_{i,p} \cdot x_p \geq 1,$$

pro každý vrchol v_i. Nastavme celočíselnou konstantu $a_{i,j}$ na 0 pokud i-tý vrchol není koncovým vrcholem j-té hrany a na 1 pokud i-tý vrchol je koncovým vrcholem j-té hrany. Zvolme $c_1 = c_2 = \ldots = c_p = 1$. Následně necháme blackbox najít takové ohodnocení $x_1, \ldots, x_p$, pro které je $c_1 \cdot x_1 + \ldots + c_p \cdot x_p$ minimální. Minimálním hranovým pokrytím je množina hran e_i, pro které je při nalezeném minimálním ohodnocení $x_i = 1$.

Důkaz, že řešení minimalizačního problému je právě minimální hranové pokrytí G: Pokud máme řešení minimalizačního problému - ohodnocení proměnných $x_1, \ldots, x_p$ nulami a jedničkami. Toto přiřazení jedniček a nul splňuje všech k nerovností, takže pro každý vrchol v_i je vybraná alespoň jedna hrana s koncovým vrcholem v_i. Jde tedy skutečně o hranové pokrytí. Zároveň každé hranové pokrytí grafu G zadává ohodnocení proměnných nulami a jedničkami splňující našich k nerovností (neboť je vždy vybraná alespoň jedna z hran sousedících s v_i pro všechna i). To znamená, že blackbox nemůže žádné hranové pokrytí přehlédnout, existuje-li. Součet $x_1 + \ldots + x_p$ je právě počet vybraných hran - za každou hranu započítáme 1, je-li vybraná a 0 jinak. Tento součet je tedy minimální možný, právě tehdy když množina vybraných hran má minimální možný počet prvků. Takže jsme pomocí minimalizačního problému skutečně našli minimální hranové pokrytí.



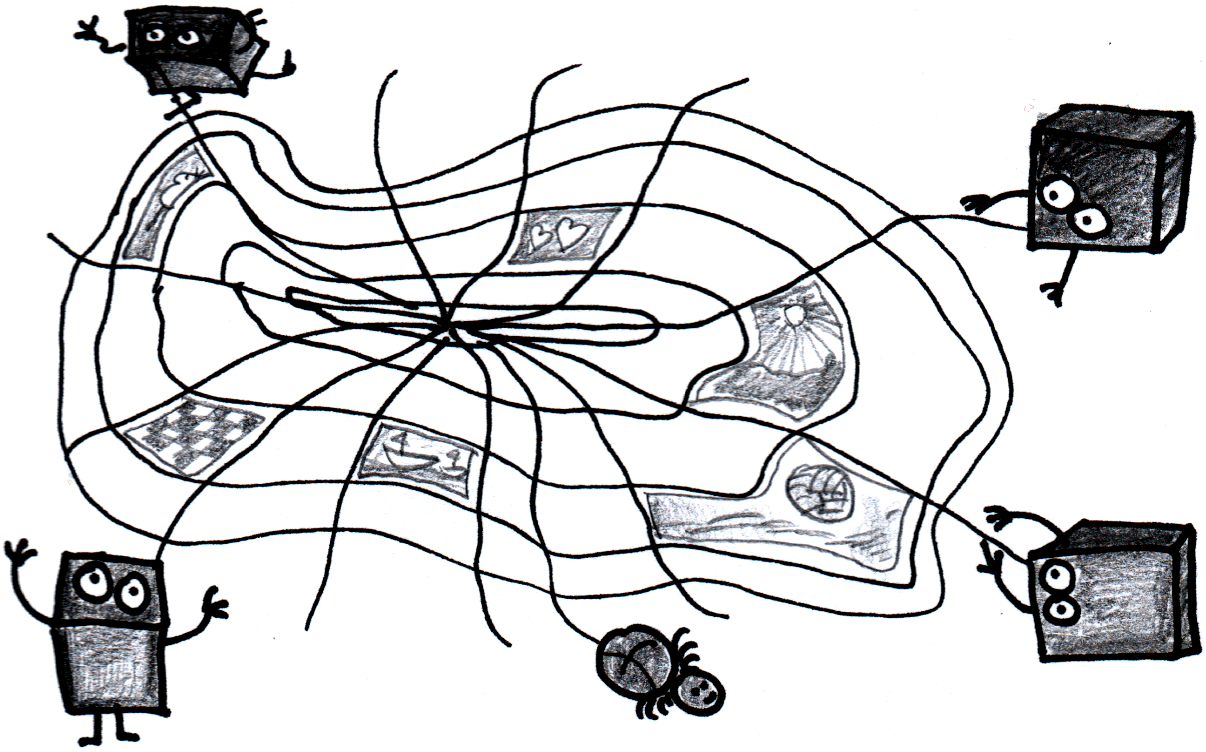
Zadání 5. sady úloh KSI (termín odevzdání: 14. 4. 2013)

Řešení zasílejte pomocí internetového systému na adrese `http://ksi.fi.muni.cz`.

Příklad 1: Sociálni sieť (10 bodů)

Martin je nadšeným používateľom nemenovanej sociálnej siete. Má na nej mnoho priateľov, no ako nedávno zistil, mnohí z nich sa medzi sebou vôbec nepoznajú. Martin by preto chcel usporiadať veľkú zoznamovaciu párty, na ktorej by sa mohli všetci spoznať. Potrebuje preto zistiť, aká je najväčšia skupina medzi jeho priateľmi, v ktorej sa žiadni dvaja ľudia nepoznajú. On sám nie je žiaden programátor, no podarilo sa mu nájsť na internete program, ktorý funguje nasledovne: program dostane na vstup mená ľudí a zoznam dvojíc priateľov. Vzťah priateľstva je pritom obojstranný. Programu sa potom môžeme spýtať, či medzi týmito ľuďmi existuje skupina n ľudí, v ktorej sú všetci medzi sebou priatelia, t.j. každý pozná každého. Program odpovie iba **ÁNO** alebo **NIE**.

Vašou úlohou je navrhnúť čo najefektívnejší postup, ako by mohol Martin použiť tento program na vyriešenie svojho problému.



Příklad 2: Prales (10 bodů)

Firma, které jste již dříve pomohli s rozmístěním hotelů v nádherných Velehorách, nyní díky vám velmi prosperuje, a rozhodla se rozšířit své služby také na sousední oblast pralesa. K tomu bude opět potřebovat vaši pomoc při plánování rozmístění nových hotelů v této lokalitě. V této oblasti jsou totiž podmínky pro rozmisťování hotelů úplně jiné, takže nelze použít algoritmus, který jste navrhli pro Velehory. Musíte proto vymyslet algoritmus nový, který dokáže naleznout co nejlepší rozmístění hotelů okolo pralesa.

Hotely mohou stát pouze ve městech, která leží v blízkosti pralesa (města označíme třeba $m_1, m_2, ..., m_n$). Vstup do pralesa je nebezpečný, a proto je povolen pouze v doprovodu průvodce. Každý průvodce ale má svou vlastní cestičku, po které bude turisty provázet (cestičky označme $e_1, ..., e_k$). O každé cestičce navíc víme, že začíná v nějakém městě (m_i) a končí v jiném (m_j).

Firma chce, aby každý návštěvník pralesa musel projít kolem některého z jejích hotelů. Každá cestička tedy musí mít hotel buď ve městě ve kterém začíná, nebo v tom kde končí.

Firma zároveň nechce utrácet peníze na stavbu přebytečných hotelů, proto musí váš algoritmus najít takové rozmístění, kdy jich bude potřeba postavit nejméně. Protože je ale prales opravdu veliký a měst i průvodců je hodně, zapůjčí vám firma pro tento úkol svůj nejnovější vynález: zařízení pro řešení problémů 0/1 programování v čase $O(n)$. Vaším úkolem je tedy naleznout algoritmus, který dokáže co nejefektivněji řešit problém rozmístění hotelů okolo pralesa za pomoci tohoto zařízení (v pseudokódu tedy můžete použít funkci `TSP2()`, která má složitost $\mathcal{O}(n)$).

P.S.: Pro vaši lepší představu přiložila firma i text, který bude používat jako reklamu pro svoje nové zařízení na řešení 0/1 programování:

Jak jistě víte, doba pokračuje mílovými kroky a s ní také dovednosti počítačů. Řešení TSP — problému obchodního cestujícího — v čase $\mathcal{O}(n)$ je už dlouhou dobu samozřejmostí. Tento pokrok sice umožnil obchodním cestujícím jednodušší plánování svých cest, ovšem přinesl i zcela nový problém: TSP Vol. 2 (dříve známý spíše jako problém 0/1 programování).

Když ráno vstane, musí si obchodní cestující kromě svačiny a dostatku propisek nabalit také spoustu dalších materiálů, kterými pak přesvědčuje své zákazníky o kvalitě, výhodnosti a dalších důležitých vlastnostech nabízených produktů. Tyto předměty jsou ale těžké a obchodníci se s nimi musejí tahat celý den. Díky řešitelnosti problému TSP ale obchodní cestující stihnou obejít za jeden den mnohem více klientů, takže potřebují i mnohem více různých přesvědčovacích nástrojů. Je proto nutné lépe rozmyslet, jak udělat svůj cestovní kufr co nejlehčí. Zároveň je ale samozřejmě nutné, aby pomocí předmětů, které do kufru přidá, dokázal přesvědčit všechny klienty, které v daný den navštíví.

O každém klientovi (nazvěme jej třeba k) vědí, jakou přesvědčovací sílu na něj má předmět (i). Tuto hodnotu lze vyjádřit jako celé číslo $a_{k,i}$. Dále ke k známe jeho nedůvěřivost b_k a víme, že si nabízený produkt koupí pouze tehdy, pokud přesvědčovací síla všech předmětů, které má obchodník u sebe, bude větší než klientova nedůvěřivost.

Trochu více matematicky: pro každý předmět i zavedeme proměnnou x_i, která nabývá pouze hodnoty 0 (nepřibalit do kufru) nebo 1 (přibalit). Součin $a_{k,i} \cdot x_i$ potom udává, jakou skutečnou přesvědčovací sílu má daný předmět na klienta k (Pokud ho obchodník nechal doma ve skříni, pak se tento součin rovná 0, takže klienta tento předmět nijak neovlivní). Nyní už můžeme sestavit nerovnici

$$a_{k,1} \cdot x_1 \ + \ a_{k,2} \cdot x_2 \ + \ \dots \ + \ a_{k,n} \cdot x_n \geq b_k$$

Na levé straně je přesvědčovací síla všech předmětů, které si obchodník přinesl sebou. Vpravo je nedůvěřivost klienta. Znaménko $\geq$ říká, že síla předmětů, které si přinesl, musí být větší než odolnost klienta. Takovouto nerovnici sestavíme pro všechny klienty, které v daný den obchodní cestující navštíví.

Dále je zcela zbytečné, aby cestoval s těžším kufrem, než je nezbytně nutné. Ke každému předmětu i určíme jeho hmotnost c_i, a podobným způsobem jako dříve vyjádříme hmotnost kufru

$$C = c_1 \cdot x_1 \ + \ \dots \ + \ c_n \cdot x_n$$

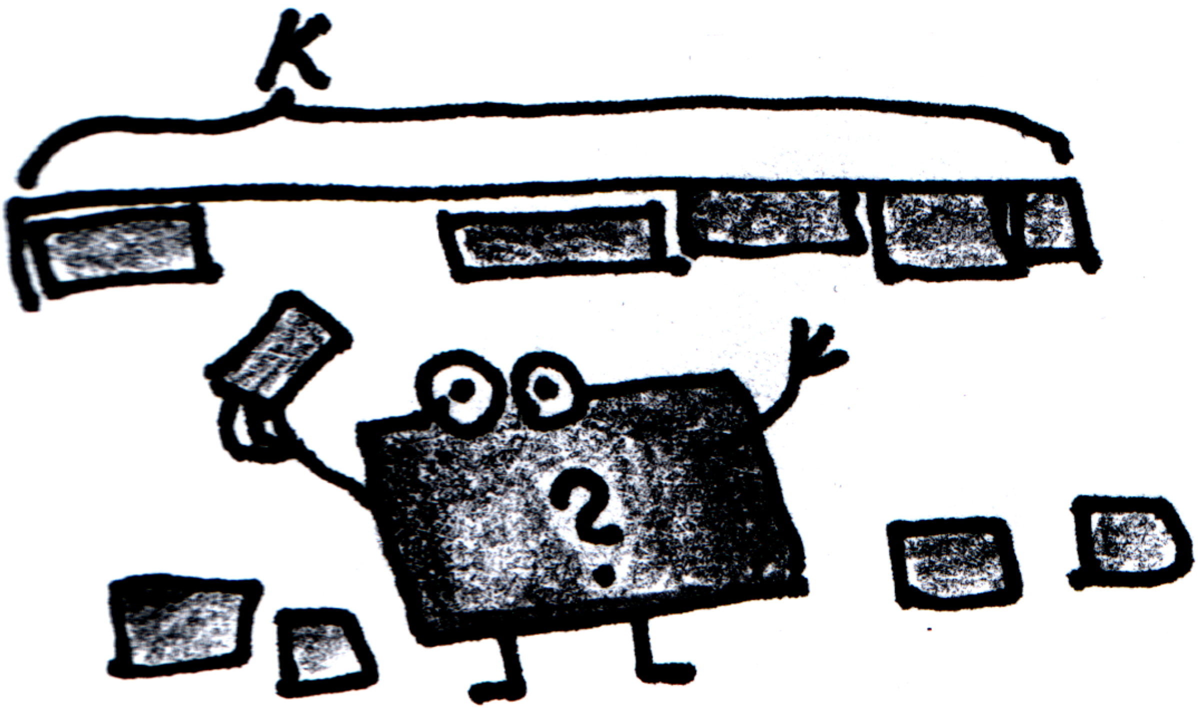
Po ní budeme požadovat, aby byla co možná nejmenší.

Řešením problému TSP Vol. 2 je pak takové přiřazení 0 a 1 všem proměnným x, které umožní přesvědčit všechny klienty a zároveň má ze všech takových přiřazení nejmenší hodnotu C.

Příklad 3: Ládíkův problém (10 bodů)

Ládík se ve svém volném čase velmi nudí, a proto se rozhodl vyřešit velice zajímavou úlohu, kterou viděl na informatické olympiádě starších. Ládík má vymyslet postup, který ze sady celých čísel vybere alespoň jedno číslo tak, aby součet těchto čísel byl přesně předem zvolené celé číslo C. Aby to měl jednodušší, tak mu jeho profesor z informatiky dal program, který mu ze seznamu celých čísel vybere taková čísla, která v součtu dávají celé kladné číslo K. Ládík si s úlohou neví rady a proto vás požádal o pomoc.

Vaší úlohou je vymyslet algoritmus, který z množiny celých čísel vybere neprázdnou podmnožinu čísel takových, že jejich součet je $C \in \mathbb{Z}$. K tomuto úkolu máte k dispozici blackbox, který vám v lineárním čase z množiny celých čísel vybere taková, jejichž součet je $K \in \mathbb{N} \setminus \{0\}$.



Příklad 4: Obchodníci (10 bodů)

Bolo raz jedno kráľovstvo. Bývali v ňom zruční remeselníci a obchodníci. Do ich ponuky patria dobroty ako žemličky s cestárskou soľou, zmrzliny s prípravkom na vši, vodka z fridexu a krowky. Majú len jediný problém. Trápi ich otázka, ako tieto mňamky rozviesť po celej Európe.

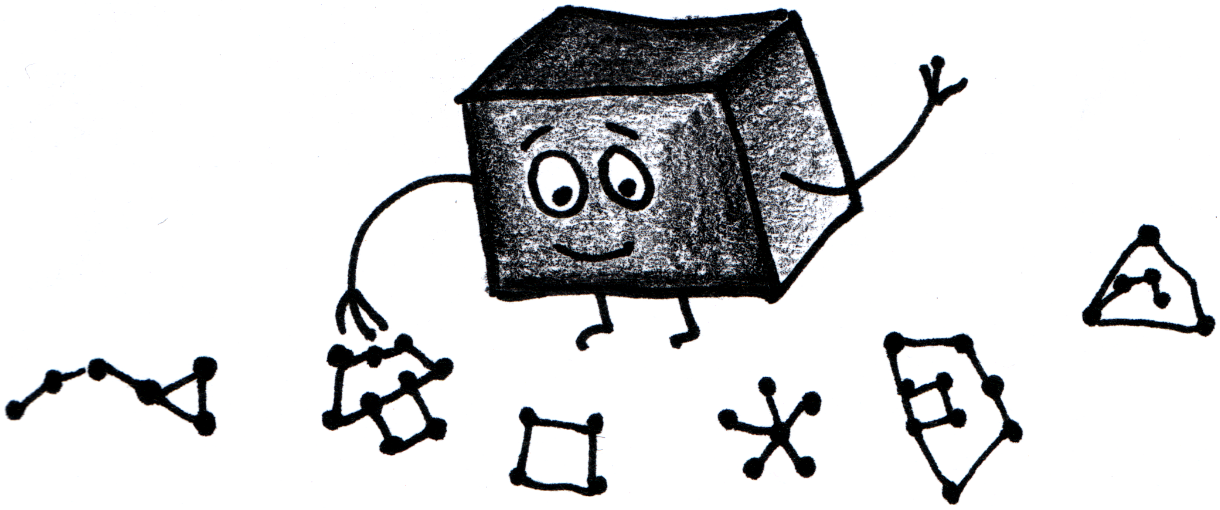
Obchodníci majú dokonalé mapy celej cestnej siete, na ktorej sú vyznačené mestá, do ktorých potrebujú dobroty rozviesť. Mestá sú spojené cestami známej dĺžky. Hrany sa pretínajú iba v mestách, všetky križovatky mimo miest sú mimoúrovňové. Zámer obchodníkov je urobiť okruh tak, aby prešli všetky mestá, ktoré si určili ako cieľové a vrátili sa naspäť do mesta, z ktorého vyštartovali. Samozrejme im ide o to, aby prešli čo najmenšiu vzdialenosť. Cesta totiž stojí peniaze, navyše môžu nastať problémy. Napríklad… hygienické kontroly…

Preto kráľovskí vedci vyvinuli úžasnú čiernu krabičku, ktorá na 100% funguje (NO FAKE) a má pomôcť riešiť všetky ich problémy. Táto krabička pracuje s grafmi a má naozaj zázračné schopnosti. Do krabičky hodíme dva grafy. Krabička v konštantnom čase nájde podgraf prvého zadaného grafu, ktorý je izomorfný s druhým zadaným grafom a vráti ho, ak taký graf existuje. Navyše ak takých grafov existuje viac, vráti taký, ktorý má najmenší súčet hodnôt na vrcholoch. Ak aj takých existuje viac, vráti taký, ktorý má najmenší súčet dĺžok hran.

Je na vás, aby ste našli algoritmus, ktorý nájde najkratší okruh prechádzajúci zadanými mestami v cestnej sieti tak, že využijete vlastnosti čiernej krabičky. Pri riešení môžete použiť funkciu $subGraph(grafG_1, grafG_2)$. Funkcia vracia graf G izomorfný s G_2, ktorý je podgrafom G_1. Vrcholy a hrany grafu G_1 si môžete ľubovoľne (vhodne) ohodnotiť. Vrátený graf G potom tieto hodnoty zachováva.

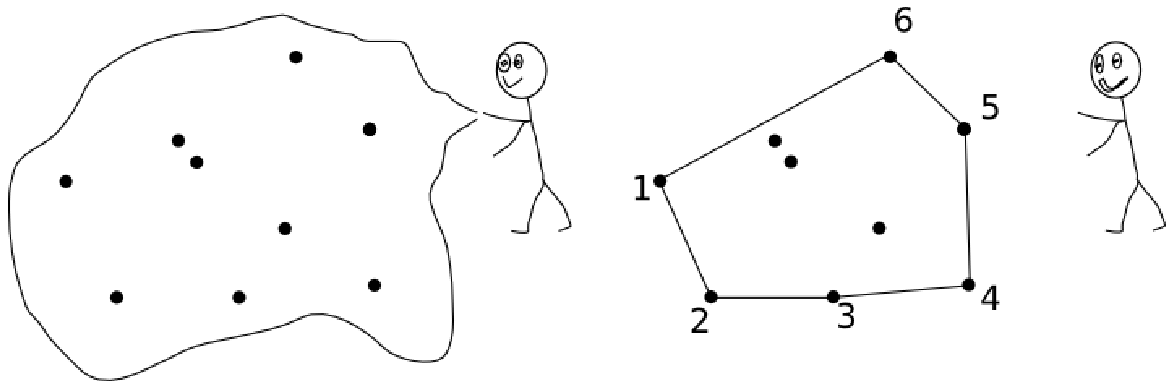
Příklad 5: Ovčák Jura (10 bodů)

V horách Beraní hlavy žije geniální ovčák Jura. Podívá-li se na libovolnou plochou louku (kterých je v horách Beraní hlavy docela málo), na které je zapíchnutých n kůlů, dokáže v čase



$\mathcal{O}(n)$ přijít na způsob, jak natáhnout kolem kůlů dokola provaz, aby byla na louce pěkná ohrada ohraničující všechny kůly (ovce pak pustí do doplňku této ohrady, aby si volně pobíhaly po horách, důležité je jen aby se nenapíchly na kůly, které musí zůstat v ohradě). Ohrada, kterou Jura najde, má tyto vlastnosti:

- všechny kůly jsou uvnitř,
- délka použitého provazu je nejmenší možná, (tj. Jura nebude okolo jednotlivých kůlů kreslit žádné složité ornamenty, prostě obejde celou oblast s kůly dokola s provazem a pak ho stáhne, aby nekreslil kružnici, ani bramboru, ale mnohoúhelník)
- hranici tvoří mnohoúhelník, jehož stěny jsou provazy natažené mezi krajními kůly,
- Jura vypíše na výstup seznam krajních kůlů, jak jdou po sobě pokud je vyjmenovává proti směru hodinových ručiček a začne jedním z kůlů, které mají minimální x-ovou souřadnici.



Vaším úkolem je setřídit pole k kladných reálných čísel, můžete se ve svém algoritmu ptát Jury, jak by ohraničil nějakou skupinu kůlů na louce danou souřadnicemi $[x, y]$, a počítat s jeho časovou složitostí.

A to je z páté, tedy pro letošek poslední, sady KSI vše. Přejeme ti hodně úspěchů při řešení, a když budeš mít jakékoliv otázky, neváhej se na nás obrátit e-mailem na adresu `ksi@fi.muni.cz` nebo v diskuzním fóru na webových stránkách.

Termín odevzdání 5. sady úloh KSI: 14. 4. 2013

`http://ksi.fi.muni.cz`