Lineární programování a kombinatorická optimalizace – praktický domácí úkol*

18. dubna 2023

1 Pokyny k vypracování a odevzdávání

Řešení praktického úkolu odevzdávejte e-mailem. Úkolem je si na několika příkladech zkusit napsat program, který ze vstupu vypíše úlohu lineárního programu zapsanou ve formátu, na kterém jde pustit řešič glpsol, který k dané úloze najde optimální řešení. Vypracované řešení každé úlohy se tedy skládá z **generátoru** úlohy lineárního programování a **dokumentace**.

Svá řešení můžete odevzdávat až do **16. května**, nejpozději ale v 17:20. Jsou povolena i opakovaná odevzdání a neváhejte mi napsat, je-li cokoli nejasné. Podrobnější informace k vypracování a odevzdání jsou uvedené níže.

1.1 Generátor

Generátorem je program, který transformuje vstup daného příkladu na úlohu lineárního programování zapsanou v jazyce GNU MathProg pro řešič glpsol. Jednoduchý manuál ke GNU MathProg naleznete na `https://iuuk.mff.cuni.cz/~bohm/texts/mathprog_intro_cz.html`, oficiální dokumentace je součástí distribuce glpk, případně rozumně aktuální verze je online na `https://ktiml.mff.cuni.cz/~micka/let1718/optimalizace/gmpl.pdf`. Několik příkladů jde najít i tady: `https://www.cs.unb.ca/~bremner/teaching/cs6375/examples/`

Vstup načítejte ze standardního vstupu či z textových souborů (viz archiv testovacích vstupů popsaný níže) a výstup vypisujte na standardní výstup. Pro potřeby debugovacích hlášek používejte standardní chybový výstup. Formát výstupu vygenerovaného lineárního programu, neboli to, co vypíše příkaz `glpsol -m vygenerovane_lp.mod`, je popsán u jednotlivých příkladů. Můžete předpokládat, že vstup je vždy ve validním formátu.

Na adrese `https://kam.mff.cuni.cz/~balko/lpko2223/ukolPrakticky.zip` naleznete archiv s připravenými vstupy pro jednotlivé příklady, na nichž můžete své generátory testovat. Váš generátor samozřejmě musí fungovat i pro jiné podobně velké vstupy. Archiv obsahuje pro každý příklad základní vstupy označené `vstup-`x`.txt`, kde x je číslo vstupu, malé vstupy označené `vstup-s`x`.txt`, a soubor `reseni.txt`, který obsahuje informace o výsledku každého vstupu (optimální hodnotu účelové funkce a časem běhu v sekundách vzorového řešení na testovacím stroji). Testovací stroj má procesor Intel Celeron 3865U (1.80GHz, Kaby Lake) a není tedy příliš rychlý.

Odevzdávejte zdrojový kód, ne přeložený program. Také si dejte pozor, aby k vašemu řešení nebyly přibaleny nadbytečné soubory.

Programovací jazyky, ve kterých můžete napsat kód generátoru, jsou C, C++, Java, C#, a Python. Pokud byste chtěli použít něco jiného, tak mi nejdřív dejte vědět a domluvíme se. Zdrojový kód musí být možné spustit či zkompilovat v počítačové laboratoři Rotunda na libovolném počítači s Linuxem či Windows. Uživatel by neměl být nijak nucen zasahovat do zdrojového kódu. Můžete používat standardní knihovny příslušného jazyka, jsou-li k dispozici v labu. Pokud si nejste jistí, jestli se daná knihovna dá považovat za standardní, tak mi napište, ale žádné speciální knihovny by neměly být potřeba. Odevzdaný zdrojový kód generátoru by měl být čitelný, formátovaný a v rozumné míře okomentovaný.

1.2 Dokumentace

Nedílnou součástí řešení je dokumentace, která musí obsahovat následující čtyři části:

(a) popis, jak sestavit program generátoru,

(b) informace, jak generátor ovládat, a

*Informace o cvičení naleznete na http://kam.mff.cuni.cz/~balko/

(c) stručný popis, jak bude vypadat vygenerovaný lineární program.

Dokumentaci odevzdejte ve formátu pdf nebo plain text. Dokumentace nemusí být dlouhá; měla by se vejít na 1 nebo 2 stránky.

Upozorňuji, že řešení, která kombinatoricky vyřeší zadanou úlohu a pak na ní pustí triviální lineární program, nebudou uznávána. Cílem tohoto domácího úkolu je seznámit se s automatickou tvorbou lineárních programů pro řešiče. Na plný počet bodů je také potřeba využít automatizaci.

2 Jednotlivé příklady

Vyřešené příklady můžete odevdávat i zvlášť a v rozumné míře i opakovaně. Celkem je možné získat až **30 bodů**.

2.1 Orientované grafy bez krátkých orientovaných cyklů [10 bodů]

Na vstupu máte orientovaný graf s hranami ohodnocenými nezápornými celými čísly. Vaším úkolem je napsat celočíslenou úlohu lineárního programování, která najde podmnožinu hran takovou, že po jejím odebrání graf nebude obsahovat žádnou **orientovanou** kružnici délky 4 či kratší, a mezi takovými množinami hran má tato minimální váhu. Graf na vstupu neobsahuje smyčky ani cykly délky 2. Očekávaná doba běhu na připravených vstupech je zhruba do 120 sekund.

Formát vstupu: Soubor s orientovaným grafem má následující formát: první řádek začíná slovem `WEIGHTED DIGRAPH` a za ním následuje počet vrcholů a počet hran, obojí je odděleno mezerami, a na konci prvního řádku je dvojtečka. Vrcholy jsou číslovány od nuly. Další řádky mají tvar i `-->` j `(` w`)` a určují jednotlivé hrany, w je nezáporná celočíselná váha hrany. Příklad vstupu pro graf K_4:

```
WEIGHTED DIGRAPH 4 6:
 0 --> 1 ( 4)
 0 --> 2 ( 3)
 0 --> 3 ( 1)
 1 --> 2 ( 4)
 2 --> 3 ( 2)
 3 --> 1 ( 5)
```

Formát výstupu: Program může vypisovat jakékoli informace uznáte za vhodné, ale výstup vždy musí obsahovat následující povinnou část: povinná část je ohraničena řádky `#OUTPUT:` W a `#OUTPUT END`, W je celková váha odebraných hran. Mezi nimi je výpis odebraných hran ve tvaru i `-->` j. Příklad pro vstup výše:

```
#OUTPUT: 2
2 --> 3
#OUTPUT END
```

2.2 Kocourkovské volby [20 bodů]

V Kocourkově se schyluje k volbám a místní radní by zajímalo, jaký minimální počet politických stran se může ve volbách objevit. Místní obyvatelé jsou velmi angažovaní a každý z nich chce být členem nějaké strany, ale všichni členové každé politické strany se musí navzájem snést. Vaším úkolem je pomocí dat získaných ze sociálních sítí určit optimální rozdělení obyvatel do politických stran.

Na vstupu máte neorientovaný graf, kde vrcholy odpovídají obyvatelům a hrany představují relaci „snesou se“. Všichni členové každé politické strany se musí navzájem snést. Výstupem je rozdělení obyvatel do stran, které minimalizuje počet stran. Dále analýza ukázala, že tento graf má velmi podobnou strukturu, jako graf známostí, tedy je tvořen převážně velkými klikami, které se nepřekrývají, a relativně málo dalšími hranami.

Poznámky k řešení: Za správně formulované přímočaré řešení dostanete 10 bodů. K zisku více bodů je nutné mít řešení výrazně rychlejší než naivní. Naivní řešení pravděpodobně doběhne v rozumném čase pouze na malých vstupech, zatímco úplně správné řešení by mělo na připravených vstupech doběhnout prakticky vždy v řádu desítek sekund (počítá se čas řešení i generování úlohy). Řešiči výrazně pomůže, pokud je řešení úlohy co nejvíce jednoznačné, a také mu může pomoci přidání podmínek, které jsou sice implikovány již uvedenými podmínkami, ale jejich vyvození vyžaduje mnoho kroků.

Formát vstupu: Soubor s neorientovaným grafem má následující formát: první řádek začíná slovem `GRAPH` a za ním následuje počet vrcholů a počet hran, obojí je odděleno mezerami, a na konci prvního řádku je dvojtečka. Vrcholy jsou číslovány od nuly. Další řádky mají tvar i `--` j a určují jednotlivé hrany $\{i, j\}$ s $i < j$. Příklad zápisu grafu $K_{1,2}$:

```
GRAPH 3 2:
 0 -- 1
 0 -- 2
```

Formát výstupu: Program může vypisovat jakékoli informace uznáte za vhodné, ale výstup vždy musí obsahovat následující povinnou část: povinná část je ohraničena řádky `#OUTPUT:` M a `#OUTPUT END`, kde M je počet politických stran. Mezi nimi je výpis stran přiřazených jednotlivým vrcholům ve tvaru `v_`i`:` x, kde i je číslo vrcholu a x číslo strany (strany jsou číslovány celými čísly od 0 do $M - 1$). Příklad povinné části výstupu pro graf $K_{1,2}$ uvedený výše:

```
#OUTPUT: 2
v_0: 0
v_1: 0
v_2: 1
#OUTPUT END
```