Korespondenční Seminář z Programování

31. ročník KSP **Září 2019**

Milí řešitelé, řešitelky a řešitelčata!

Právě si prohlížíte komentáře k úlohám čtvrté série KSP-H (přesněji k těm, ke kterým jsme uznali, že se komentář hodí). Připomínáme, že od letoška jsou totiž řešení každé série rozdělena na dvě části: na samotná autorská řešení, která vydáváme brzy po termínu série, a komentáře k došlým řešením, která vydáváme až po opravení vašich řešení.

Pokud se vám cokoliv nezdá nebo máte nějaký dotaz, neváhejte se ozvat na našem fóru nebo emailem na známou adresu.

Komentáře k čtvrté sérii třicátého prvního ročníku KSP

31-4-1 Kouzelné zrcadlo

Na začátku bych rád zmínil, že v podobných úlohách již počítáme s tím, že zadané posloupnosti jsou již načtené v paměti (třeba v podobě polí) – část z vás si tím nebyla úplně jistá. Jde nám tedy jenom o čas vyhledání bez času nutného k načtení. V reálném světě se s tímto konceptem často potkáme, když je náš program součástí řešení nějaké větší úlohy a data pro nás již připravil v paměti někdo jiný.

Většina vašich řešení šla na věc správně, ale dvěma rozdílnými způsoby založenými na binárním vyhledávání. Ten první odpovídá našemu vzorovému řešení, kdy postupně odebíráme části z prohledávaných polí a zmenšujeme k, a je asi jednodušší na dokázání – alespoň v jeho dokazování nikdo chybu neudělal.

Druhým postupem bylo použití binárního vyhledávání napřímo – podívat se na prvek v první posloupnosti, dopočítat si index prvku ve druhé a podle výsledku jejich porovnání se v první posloupnosti posunout doleva nebo doprava. To je správná idea, ale je potřeba nezapomenout na kontrolu, jestli již číslo z právě zpracovávané dvojice není správným řešením. Vezměme si číslo $A[i]$ z první posloupnosti a $B[j]$ (pro $j = k - i$) ze druhé a nechť $A[i] < B[j]$. Pokud $A[i+1] > B[j]$, tak jsme již našli k-tý nejmenší prvek, protože $A[i+1]$ je už určitě větší, než k prvních prvků. Na tuto koncovou podmínku část z vás zapomněla, a přišla tak o velkou část bodů.

Pokud vaše řešení procházelo posloupnosti prvek za prvkem, tak jste dosáhli časové složitosti $\mathcal{O}(k)$ a v moc bodů jste doufat nemohli. Co mě však překvapilo, bylo několik řešení, které vůbec nevyužívaly toho, že posloupnosti jsou již setříděné a pokoušely se je třídit znovu. Na to mohu doporučit asi jen pořádně číst zadání – většinou, když vám slíbíme něco setříděné, tak je klíčové této vlastnosti využít.

Jirka Setnička

31-4-2 Stromy na mýtince

Nápady, které vedou k efektivnímu řešení úlohy, měla tentokrát většina z vás. Komplikovanější část řešení spočívala v jejich správné aplikaci a následných „implementačních detailech“.

Několik řešení velmi připomínalo vzorové řešení, jen jste neznačili pohyb nahoru a dolů, ale zapisovali jste si počty synů jednotlivých vrcholů. Představte si, že na vstupu dostanete stromy, které mají pouze kořen a k listů. Každý strom potom lze reprezentovat posloupností začínající právě číslem k, následovaným k nulami. Takové posloupnosti se nám budou těžko ukládat do trie. Po kořenu trie bychom totiž chtěli, aby mohl mít libovolně mnoho potomků – pro každé k potřebujeme jednoho. V klasické implementaci trie máme však počet potomků jednoho vrcholu omezený konstantou – velikostí abecedy. Díky tomu pak zvládneme jeden vrchol projít opět v konstantním čase. Bez tohoto omezení neplatí rozbor časové složitosti a dostaneme nakonec pomalé řešení.

Jedno z vašich řešení se tento problém snažilo opravit tím, že čísla zapsalo do textového řetězce v desítkové soustavě. Tím se omezila velikost abecedy na počet možných cifer (`0-9`), ale délka řetězce se až logaritmicky prodloužila. Přesněji: k prodloužení by došlo, pokud by všechna čísla v kódu stromu byla velká. V našem případě čísla odpovídají stupňům vrcholů stromu a pro stromy lze dokázat, že celková délka takového řetězce je stále lineární s počtem vrcholů.

Toto řešení však narazilo na jiný problém: Když čísla zapíšeme do řetězce, musíme je v řetězci také oddělit od sebe. Jinak vygenerujeme stejný řetězec pro dva neizomorfní stromy: `1, 11, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0` a `11, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0`.

Občas jste se snažili problémům s trií vyhnout tím, že jste místo ní použili hešování. Při reálném programování úlohy to samozřejmě může být výhodné. V mnoha jazycích již někdo hešování naprogramoval a můžeme takovou implementaci použít. Nicméně řešení používající hešování mají lineární časovou složitost pouze v průměrném případě. Pokud neřekneme jinak, myslíme zpravidla časovou složitostí rozbor nejhoršího případu.

Když už mluvíme o průměrném případě, pojďme si připomenout, co to vlastně znamená. Hešování je pravděpodobnostní technika – program si náhodně zvolí funkci, kterou bude řetězec hešovat na jedno číslo. Pokud budeme program spouštět opakovaně na stejném vstupu, pokaždé si vybere hešovací funkci znovu nějak náhodně. Průměrným případem pak myslíme průměr počítaný přes opakované běhy programu na stejném vstupu.

Občas se ve vašich řešeních objevují tvrzení typu „V nejhorším případě je program kvadratický, ale takové vstupy jsou vzácné. V průměrném případě je moje řešení lineární.“ S tím však často nelze souhlasit. Pokud to není v zadání explicitně uvedeno, nemůžete nic předpokládat o tom, jak vypadá průměrný vstup. Problém je, že pokud pro jeden konkrétní vstup váš program poběží vždy kvadraticky, může vám jej protivník stále předhazovat dokola. A pak bude i váš průměr kvadratický.

Jenda Hadrava

31-4-4 Otrávené ovoce

Tato úloha se ukázala jako oříšek – ze sedmi došlých řešení jen jedno dokázalo, že je úloha otráveného ovoce $\mathcal{NP}$-úplná. Připomeňme si, co to vlastně znamená. Abychom ukázali, že problém ovoce je $\mathcal{NP}$-úplný, musíme ukázat ukázat, že o něm platí dvě věci:

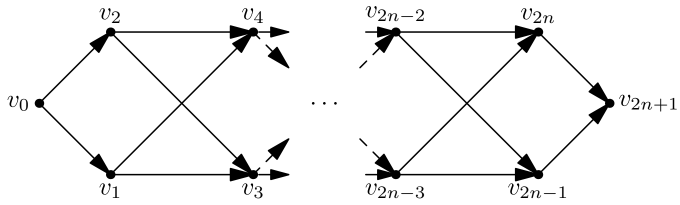
- *(je těžký)* Jde na něj převést nějaký jiný $\mathcal{NP}$-úplný problém, tedy kdybychom měli magickou krabičku řešící problém ovoce „efektivně" (lépe řečeno polynomiálně), uměli bychom „efektivně" vyřešit i onen $\mathcal{NP}$-úplný problém, a tím pádem i všechny ostatní problémy v $\mathcal{NP}$.
- *(s nápovědou už je lehký)* Problém leží v $\mathcal{NP}$, tj. existuje nějaký polynomiálně velký kus informace, se kterým už úlohu umíme vyřešit v polynomiálním čase.

Častou chybou bylo, že jste se pokoušeli převod provést obráceně, tedy vyřešit problém ovoce pomocí nějakého $\mathcal{NP}$-úplného problému. To nám ale o obtížnosti problému ovoce nic neříká! (Tedy kromě toho, že také leží v $\mathcal{NP}$.) Stejně tak bychom pak mohli tvrdit, že problém „Najdi v posloupnosti čísel maximum" je těžký, protože ho umíme převést na problém batohu.

Navíc to, že problém ovoce jde na daný $\mathcal{NP}$-úplný problém převést, už víme: z definice jdou na $\mathcal{NP}$-úplné problémy převést všechny problémy v $\mathcal{NP}$, takže stačí vědět, že problém ovoce je v $\mathcal{NP}$, a převod tímto směrem máme zadarmo. To ale neznamená, že vymýšlet převody na $\mathcal{NP}$-úplné problémy je k ničemu – v praxi se často vyplatí převést $\mathcal{NP}$-úplný na nějaký jiný, který je sice také $\mathcal{NP}$-úplný, ale pro rozumně velké vstupy ho umíme řešit efektivně.

Dalším problémem bylo dokazování, že problém leží v $\mathcal{NP}$. Všechny správné důkazy volily jako certifikát popis rozdělení ovoce mezi služebníky, případně i s popisem toho, jakou cestou se který služebník vydá. Co ale nefungovalo, bylo vzít za certifikát jediné číslo označující nejmenší nutný počet služebníků. Program, který certifikát ověřuje, se totiž nesmí nechat zmást chybnými nebo zákeřnými certifikáty, které tvrdí, že úloha má řešení, i když nemá (nebo v našem případě, že úloha má lepší řešení, než ve skutečnosti má). V případě, kdy za certifikát zvolíme číslo označující nejmenší nutný počet služebníků, musí ověřovací program stejně spočítat, zda tento počet služebníků stačí, aby mohl certifikát buď přijmout, nebo zamítnout. To je ale stejně těžké jako vyřešit původní úlohu.

Někteří řešitelé také ve svém řešení hledali všechny cesty v grafu mezi dvěma vrcholy. To je však těžká úloha – ne ani tak proto, že by byla $\mathcal{NP}$-úplná (což ani není, protože dokonce neleží v $\mathcal{NP}$), ale prostě proto, že cest může být exponenciálně, takže jejich vypsání nestihneme v polynomiálním čase. Příkladem budiž následující graf na $2n+2$ vrcholech s 2^n cestami z v_0 do v_{2n+1}.



Na závěr si předvedeme pěkný převod, na který přišel Jiří Kalvoda, a který je řádově jednodušší než vzorové řešení. Převádět budeme problém 3D párování v podobě, v jaké je popsán v kuchařce:[1] Na vstupu máme seznam k mužů, k žen a k zvířátek. Dále máme zadaný seznam uspořádaných trojic (*muž, žena, zvířátko*) určující, které trojice muž, žena, zvířátko se spolu snesou. Úkolem je rozdělit bytosti do k trojic tak, aby se spolu snesli a každá bytost byla právě v jedné trojici.

Vyrobíme si graf sestávající pouze z počátečního a cílového vrcholu a za každou trojici do něj přidáme hranu ze startu do cíle ohodnocenou (tříprvkovou) množinou bytostí z této trojice. Za ovoce prohlásíme bytosti. Předhodíme takový vstup naší kouzelné krabičce a rozmyslíme si, že původní problém měl řešení právě tehdy, když nám bude stačit k služebníků, neboť v takovém případě musí každý služebník nést právě jednu trojici. Tím je převod hotov.

Ríša Hladík

31-4-5 Dělení království

Menší část z vás šla na úlohu podobnou grafovou představou, jakou máme v našem vzorovém řešení, a je pravda, že tam jste většinou žádné dokazovací chyby nedělali. Přecejenom představa nějakých puntíků putujících po cestičce a kolečku je docela intuitivní :)

Větší část z vás ale zkoušela jiný postup, a to spočítat si, jak je dlouhá část před periodou. Pokud se vám toto povedlo, tak již řešení bylo jednoduché – zapamatovali jste si první číslo periody a dělili tak dlouho, dokud jste ho opět nepotkali. Větší problém byl ale v dokazování toho, že vaše spočítání spočítá délku před periodou správně.

Většinou jste si někde našli tvrzení o rozkladu na mocniny 2 a 5 a aplikovali jste ho – v lepším případě s odkazem, v horším jste ani odkaz na žádný zdroj neuvedli. Ale u takovéhoto ne úplně zjevného tvrzení od vás očekáváme alespoň nějaký nástin důkazu, proč platí. Obecně když vám uvěříme, že jste tvrzení sami pochopili, tak ho pak můžete v řešení použít.

Pouze jedno řešení se pokoušelo důkaz popsat (a docela dobře), ostatní řešení, která se pokusila použít tvrzení bez důkazu, bohužel několik bodů ztratila.

Jirka Setnička

[1] `http://ksp.mff.cuni.cz/viz/kucharky/tezke-problemy`

Výsledková listina čtvrté série třicátého prvního ročníku KSP

	řešitel	*škola*	*ročník*	*sérií*	*4-1*	*4-2*	*4-3*	*4-4*	*4-5*	*4-6*	*série*	*celkem*
0.					10	11	8	12	12	15	60,0	240,0
1.	Jiří Kalvoda	GJarošeBO	2	4	10	11	8	12	12	15	60,0	236,2
2.	Jiří Kvapil	GTomkovaOL	1	9	0	10	8	7	12	15	53,0	193,3
3.	Petr Budai	G JGJ PH	2	4	2,5		8		9	3	28,8	181,3
4.	Jan Provazník	GVoděraPH	3	4	10	10	8		4	12	49,0	181,1
5.	Daniel Skýpala	GTomkovaOL	1	12	10	10	8		10	9	46,3	172,7
6.	Lucie Vomelová	GŠpitálsPH	3	5	4		8	2	4	11	36,9	162,4
7.	Daniel Kurek	GTomkovaOL	3	4	10	6	5	2		15	43,7	160,5
8.	Ondřej Jamelský	G Cheb	1	3							0,0	155,4
9.	Dalibor Kramář	G BO-Řeč	4	4							0,0	152,1
10.	Jan Piroutek	GŠpitálsPH	3	5	2,5	4	8	6	0	10	39,0	150,2
11.	Vladimír Chudý	G Chrudim	2	9	8	6	8		4	5	34,0	149,6
12.	Vojtěch Žák	GŠpitálsPH	3	5	10		8			11	31,0	148,8
13.	Jiří Šáda	GVoděraPH	3	4			8			15	23,0	146,3
14.	Petr Kolář	GMilevsko	3	4	10	3	8	2	5		34,7	143,0
15.	Jakub Komárek	GUHradiště	4	9			8			15	23,0	141,2
16.	David Klement	GNAlejíPH	3	6							0,0	132,2
17.	Petr Zahradník	GaSOŠ ÚL	4	6							0,0	128,6
18.	Kristýna Petrlíková	VOŠJičín	1	4	4	10	8			9	36,9	126,2
19.	Tomáš Černý	GArabskáPH	3	6	1		3			9	17,3	125,1
20.	Martin Zimen	GJMasarJI	4	4							0,0	90,3
21.	Václav Pavlíček	SPSEPard	3	16	10					9	16,6	90,1
22.	Lucia Krajčoviechová	GJHroncaBA	3	5			8				8,0	77,1
23.	Daniel Oravec	GVaršŽilina	4	2							0,0	65,7
24.	Ondřej Gonzor	G Brandýs	2	12							0,0	64,6
25.	Michal Kodad	SPŠSmíchov	3	15							0,0	62,1
26.	Matěj Kripner	GEBenešeKL	4	8							0,0	56,2
27.	Josef Minařík	GJarošeBO	4	4							0,0	54,3
28.	Janek Hlavatý	GJirsíkaČB	0	2							0,0	53,4
29.	Jakub Pánek	SPŠERožnov	4	2							0,0	43,9
30.	Daniil Barabashev	GNadKavaPH	3	2							0,0	42,7
31.	Tomáš Sláma	GTurnov	4	1							0,0	40,6
32.	Jan Kaifer	GKepleraPH	3	12	9,5	11	8				28,4	40,4
33.	František Kmječ	StOlavVGS	3	10							0,0	39,8
34.	Jindřich Dítě	VOSPŠŽďár	3	4							0,0	37,8
35.	Marek Černoch	GFPValMez	3	1							0,0	31,5
36.	Ondřej Sladký	GMikulášPL	2	1	10	4	3	4			30,5	30,5
37.	Jakub Profota	GŘíč	4	1							0,0	30,3
38.	Vít Skalický	GPísnickáPH	1	8			5				5,7	29,0
39.	Vojtěch Březina	GCoubTábor	2	3							0,0	25,6
40.	Jáchym Mierva	BiGy Žďár	2	4							0,0	23,7
41.	Martin Miller	GVoděraPH	4	3							0,0	23,0
42.	Jakub Šťastný	G BO-Řeč	4	1							0,0	22,8
43.	Martin Hubata	GMikulášPL	3	1							0,0	22,2
44.	Ondra Müller	GTurnov	2	2							0,0	22,0
45.	Robert Gemrot	GKomHavíř	2	1	10				11		21,8	21,8
46.	Linda Kimrová	GEvolutionJM	3	1							0,0	21,2
47.	Matěj Volf	GCoubTábor	1	1							0,0	19,7
48.	Filip Hejsek	GPísnickáPH	2	2							0,0	12,0
49.	Patrik Vácal	SPŠEPlzeň	2	1							0,0	9,5
50.	Ondřej Bleha	GBNěmcovHK	4	3							0,0	9,0
51.–54.	Ondřej Daniš	GFPValMez	4	1							0,0	8,0
	Kristýna Prokopová	GJosBožČT	3	1							0,0	8,0
	Petr Šejvl	SPŠPísek	4	1							0,0	8,0
	Roman Šíp	SPŠPísek	4	1							0,0	8,0
55.	Anna Hollmannová	GSRandyJN	2	5							0,0	7,8

	řešitel	*škola*	*ročník*	*sérií*	*4-1*	*4-2*	*4-3*	*4-4*	*4-5*	*4-6*	*série*	*celkem*
56.–62.	Robert Jaworski	GÚstavníPH	1	1							0,0	7,6
	Vojtěch Jedlička	GCoubTábor	2	1							0,0	7,6
	Petr Khartskhaev	PORGPha	2	1							0,0	7,6
	David Krásný	SPŠEPlzeň	2	1							0,0	7,6
	Petr Macháček	GTýnNVlt	3	1							0,0	7,6
	Jan Najman	SPSEPard	2	1							0,0	7,6
	Jakub Vybiral	GLovosice	2	1							0,0	7,6
63.	Tomáš Vesecký	SSŠVTPraha	2	1	4						7,1	7,1
64.	Marie Kalousková	GNAlejíPH	3	2							0,0	6,8
65.–66.	Vít Gardoň	GPří	3	1							0,0	5,5
	Ondřej Chlubna	GOrlová	2	1							0,0	5,5
67.–69.	Matyáš Boháček	ZŠKladskáPH	1	1							0,0	4,7
	Tomáš Pelák	SŠkybernHK	3	1							0,0	4,7
	Matej Straka	SPŠEPrešov	4	1							0,0	4,7
70.	Ondřej Cach	SPSEPard	3	2							0,0	4,4
71.	Vojtěch Crha	GČeskoliPH	4	1							0,0	4,1
72.	Martin Havelka	Gym Třeboň	1	1	1						2,5	2,5

KSP pro vás připravují studenti Matematicko-fyzikální fakulty Univerzity Karlovy.

Webové stránky: *https://ksp.mff.cuni.cz/*
E-mail: *ksp@mff.cuni.cz*
Diskusní fórum: *https://ksp.mff.cuni.cz/forum/*

Chcete-li s námi komunikovat bezpečně, můžete si ověřit náš HTTPS certifikát – jeho SHA1 fingerprint je: `E9:DB:EE:C6:62:BC:14:DE:09:E4:E8:97:DC:36:0E:87:B3:50:B0:01`.