A számítástudomány alapjai 2017. I. félév

11. gyakorlat. Összeállította: Fleiner Tamás (fleiner@cs.bme.hu)

Tudnivalók

Megfigyelés: Ha $a \equiv b(m)$ akkor $(a, m) = (b, m)$.

Köv.: Az m-hez relatív prím számok halmaza néhány mod m maradékosztály uniója.

Def: Az m-hez relatív prím maradékosztályok számát $\varphi(m)$ jelöli, ez az *Euler-féle φ függvény.*

Def: Az $\{a_1, a_2, \ldots, a_n\} \subset \mathbb{Z}$ halmaz *redukált maradékrendszer modulo m* (röviden *RMR mod m*), ha minden m-hez relatív prím mod m maradékosztályból pontosan egy elemet tartalmaz.

Megfigyelés: Tetszőleges mod m TMR-ből mod m RMR-t kapunk, ha elhagyjuk belőle az m-hez nem relatív prím számokat. Tetszőleges mod m RMR mérete $\varphi(m)$, így azé is, amely az m-nél kisebb, m-hez relatív prím természetes számokból áll.

Tétel: Ha p prím, akkor (1) $\varphi(p) = p - 1$, (2) $\varphi(p^\alpha) = (p-1)p^{\alpha-1}$.
(3) $(a, b) = 1 \Rightarrow \varphi(ab) = \varphi(a)\varphi(b)$. (4) Ha $n = \prod_{i=1}^{k} p_i^{\alpha_i}$, akkor $\varphi(n) = n \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right)$.

Tétel: Ha $\{a_1, a_2, \ldots, a_m\}$ RMR mod m, és $(b, m) = 1$, akkor $\{ba_1, \ldots, ba_m\}$ RMR mod m.

Euler-Fermat tétel: Ha $(a, m) = 1$, akkor $a^{\varphi(m)} \equiv 1(m)$.

Kis Fermat tétel: Ha p prím és $a \in \mathbb{Z}$, akkor $a^p \equiv a(p)$.

Def: Az algoritmus fogalmát nem definiáljuk, de algoritmusra úgy gondolunk, mint valamiféle elméleti számítógépen futtatott programra, mely egy, az input által megadott konkrét feladatot old meg, azaz véges sok lépés megtétele után a helyes megoldás lesz az adott inputhoz tartozó output. Az A algoritmus ahhoz a Π problémához tartozik, amit az A algoritmus megold. Természetesen ugyanahhoz a Π problémához több különböző algoritmus is tartozhat. Az A algoritmus *inputja* (bemenete) jelöli ki, hogy a szóban forgó Π problémacsoportnak pontosan melyik konkrét feladatáról is van szó. Az input *mérete* az inputot leíró bitek száma. Tetszőleges A algoritmushoz tartozik egy f_A függvény: $f_A(n)$ azt adja meg, hogy legfeljebb hány lépést tesz meg az A algoritmus a legfeljebb n hosszúságú inputokon. A lépésszámfüggvény szempontjából tehát lényegtelen, ha az A algoritmus a legfeljebb n hosszúságú inputok $99,999\%$-án néhány lépésben végez, $f_A(n)$ a legrosszabbul viselkedő, legfeljebb n hosszúságú inputhoz tartozó lépésszám. A lépésszámra vonatkozó felső becslés garanciát ad a lépésszámra, és olyan algoritmust próbálunk keresni a vizsgált problémákra, amelyek minél jobb lépésszámgaranciával rendelkeznek.

Példa: Ha az algoritmus inputja egy pozitív egész n szám, akkor az input mérete az n bináris alakjában található jegyek száma, azaz $1 + \lfloor \log_2(n) \rfloor$. Ha az input egy n csúcsú egyszerű $G = (V, E)$ gráf, akkor G-t a szomszédossági mátrixával megadva az input mérete n^2. (Vannak persze értelmesebb megadások is, sőt, szinte csak azok vannak. Éllistával pl. az input mérete $konst \cdot (n+m)$, ahol m a G éleinek száma.)

Def: Az A algoritmus *polinomidejű* (néha *polinomiális* vagy *hatékony*), ha létezik olyan $p(n)$ polinom, amelyre $f_A(n) \leq p(n)$ teljesül minden $n \geq 1$-re. Az A algoritmus *exponenciális idejű*, ha létezik olyan pozitív K és $c > 1$ konstansok, melyekre $f_A(n) \leq K \cdot c^n$ teljesül minden $n \geq 1$-re.

Megjegyzés: Ha a számítógép műveleti sebessége (mondjuk) kétszeresére gyorsul, akkor egy polinomidejű algortimussal egységnyi idő alatt egy konstansszor nagyobb inputméretű problémát oldhatunk meg, mint a régi számítógéppel, míg exponenciális futásídeű algoritmus esetén csak konstanssal növekszik az egységnyi idő alatt megoldható input mérete.
Itt és most a polinomidejű algoritmust hatékonynak tekintjük, az olyat pedig, nem szeretjük, amire az exponenciális becslésnél nem tudunk jobbat mondani. Minden polinomidejű algoritmus egyúttal exponenciális idejű is.

Példa: A BFS algoritmus hatékony, hiszen egy n élű m csúcsú gráfot (amelyet meg lehet adni n^2 méretű vagy $konst \cdot (n+m)$ méretű inputtal) a lépésszámra $f_{BFS}(n) \leq c \cdot (n+m)$ teljesül alkalmas c konstansra, tehát $p(n) = c' \cdot n$ megfelelő polinom, ahol c' alkalmas konstans.

Def: *Döntési probléma* az olyan probléma, amelyek outputja egyetlen bit, azaz minden értelmes inputhoz az IGEN vagy a NEM outputok valamelyike tartozik. A P problémaosztályt mindazon döntési problémák alkotják, amelyekre van polinomidejű algoritmus.

Def: NP-beli probléma alatt olyan Π döntési problémát értünk, amelyhez létezik egy A algoritmus az alábbi tulajdonsággal. (1) Az A algoritmus inputja tetszőleges I, T pár, és az A algoritmus lépésszáma felülről becsülhető I méretének polinomjával. (Tehát T-től nem függ.) (2) A Π minden

olyan I inputjához, amelyhez IGEN a helyes output, létezik olyan T tanú, hogy az A algoritmust az (I,T) inputtal elindítva IGEN választ ad, valamint (3) minden olyan I inputhoz, amelyhez NEM a helyes output, nem létezik olyan T tanú, hogy A az (I,T) inputra elindítva IGEN választ adjon. (Röviden: egy döntési probléma akkor NP-beli, ha az IGEN válasz polinomidőben bizonyítható.) A $co-NP$-beli problémaosztályt azon Π döntési problémák alkotják, amelyekhez a NEM válaszra van a fentiekhez hasonlóan definiálható tanú.

Állítás: $P \subseteq NP \cap co-NP$.

Def: A Π probléma *polinomidőben visszavezethető* a Π' problémára (jelölése $\Pi \prec \Pi'$), ha a Π tetszőleges I inputjához polinomidejű algoritmussal konstruálható a Π' problémának olyan I' inputja, melyre (Π'-ben) ugyanaz a válasz, mint I-re Π-ben.

Megfigyelés: (1) $\Pi \prec \Pi' \prec \Pi'' \rightarrow \Pi \prec \Pi''$. (2) $\Pi \prec \Pi' \in P \Rightarrow \Pi \in P$.

Gyakorlatok

1. Oldogassunk lineáris kongruenciákat. (a) $49^{49}x \equiv 3(15)$ (b) $3^{80}x \equiv 23(100)$.
2. Hogyan számíthatjuk ki gyorsan a 7^{73} hatvány 19-es maradékát? Ugyanez a kérdés, de a φ függvény értékét nem használhatjuk. Mi a helyzet az n^k kiszámításával modulo m?
3. Számítsuk ki a $\varphi(533)$, $\varphi(2007)$ és $\varphi(540)$ értékeket.
4. Bizonyítsuk be, hogy $11 \mid n^{11}+10n$ és $42 \mid n^7-n$ teljesül tetszőleges $n \in N$ esetén.
5. Bizonyítsuk be, hogy tetszőleges $h_1, h_2, \ldots, h_k$ pozitív egészekre és p prímszámra fennáll, hogy $(h_1+h_2+\ldots+h_k)^p \equiv h_1^p+h_2^p+\ldots+h_k^p(p)$. (ZH '02)
6. Milyen maradékot ad a 31-gyel osztva, ha $a^{100} \equiv 5(31)$ és $a^{101} \equiv 19(31)$? (V '00)
7. Mi a 403^{402} utolsó három, a $29^{39^{49}}$ utolsó két és a $7^{6^{5^{4^{3^2}}}}$ szám utolsó jegye tízes számrendszerben?
8. Milyen maradékot ad 59^{99} 101-gyel osztva? (ZH '03)
9. Mi az utolsó három jegye a $999^{777^{8888}}$ számnak? Mi az utolsó két jegye az $1997^{2001^{2005}}$ számnak?
10. Bb: ha $p>5$ prím, akkor az $1, 11, 111, \ldots$ számok között végtelen sok többszöröse van! (ZH '01)
11. Bb: $17 \mid 2002^{2002}+1$ (ZH '02)
12. Legyenek m és n pozitív egészek, továbbá $m \mid n$. Bizonyítsuk be, hogy $\varphi(m) \mid \varphi(n)$. (ZH '00)
13. Mely n számokra lesz $\varphi(n)$ prímszám? Hát aztán mikor lesz $\varphi(n)$ páratlan? (ZH '99)
14. Mely n természetes számokra igaz, hogy $\varphi(5n)+\varphi(3n)=7\varphi(n)$? (ZH '03)
15. Bb: ha $d \mid n$, akkor $d-\varphi(d) \leq n-\varphi(n)$. (V '00)
16. Bb: $\sum_{0<i<n,\ (i,n)=1} i = \frac{n \cdot \varphi(n)}{2}$, ha $n>1$, egész. (V '99)
17. Ha $r_1, r_2, \ldots, r_{\varphi(n)}$ redukált maradékrendszer modulo n, akkor $\sum_{i=1}^{\varphi(n)} r_i \equiv 0(\text{mod } n)$. (V '00)
18. Mutassuk meg, hogy tetszőleges $n>1$ egész számra $\varphi(\varphi(n)) \leq \frac{n}{2}$ teljesül. Mutassunk olyan n-et, amire $\varphi(\varphi(n)) \geq 0,47n$.
19. Ismételjük át, hogy az egész számok összeadására és szorzására van polinomidejű algoritmus.
20. Négyzetreemelések segítségével számítsuk ki $10^{133}(mod\ 13)$ értékét, azaz határozzuk meg, milyen maradékot ad 13-mal osztva a 10^{133}. Határozzuk meg ugyanezt most az Euler-Fermat tételre támaszkodva.
21. Tegyük fel, hogy A egy polinomidejű algoritmus a Π problémára. Legyen Π' egy másik probléma, és legyen A' olyan polinomidejű algoritmus, amely Π' tetszőleges I' inputjához a Π olyan I inputját készíti el, amelyhez a Π problémában ugyanaz a válasz tartozik, mint I'-höz a Π' problémában. Helyes és polinomidejű-e az az A^* algoritmus a Π' problémára, amelyet úgy kapunk, hogy Π' tetszőleges I' inputján lefuttatjuk az A' algotitmust, majd a kapott I outputot inputként felhasználva lefuttatjuk az A algoritmust?
22. Tegyük fel, hogy az A algoritmus a Π problémát oldja meg olyan módon, hogy Π tetszőleges n méretű inputjához A n lépésben elkészíti a Π probléma egy $\lceil n/2 \rceil$ és egy $\lfloor n/2 \rfloor$ méretű inputját, és azokat megoldja saját maga meghívásával. Polinomidejű-e az A algoritmus? Mi a helyzet akkor, ha tetszőleges n méretű inputból a két elkészített input mérete $n-10$?