Att dela en hemlighet

Johan Håstad

KTH

1. Inledning. Anta att du har en hemlighet som du inte vill att någon annan skall veta. För att vara konkret, låt denna hemlighet vara numret på ditt hemliga bankkonto i Schweiz. Naturligtvis vill du inte att någon känner till detta kontonummer, men ändå vill du att dina tre barn skall kunna komma åt kontonumret när du är död. Du litar inte på dina barn och de litar inte på varandra. (Ett olagligt bankkonto i Schweiz medför att man blir litet försiktig och inte litar på folk.) Du kommer på följande idé. Anta att kontonumret är 31784 (i själva verket är det längre, men vi sparar skrivarbete). Du väljer två slumpartade 5–siffriga tal, låt oss säga 69241 och 77431. Du ger dina tre barn talen 69241, 77431 och 05112, ett till dem var. Dessutom ger du instruktionerna att om de adderar förstasiffrorna i deras tre tal och stryker tiotals–siffran så får de den första siffran i kontonumret, likadant för den andra siffran o.s.v. Du är mycket nöjd med dig själv och har åstadkommit följande:

1) Dina tre barn kan tillsammans få fram kontonumret.
2) Två av dem kan inte lista ut något om kontonumret förutom att det har fem siffror.

Visa detta påstående.

Med andra ord har du givit bort din hemlighet utan att någon enstaka person vet den. Den här uppgiften går ut på att studera algoritmer för att fördela hemligheter och deras egenskaper och svagheter.

Låt oss börja med att påpeka några problem med ovanstående algoritm.

1) Om ett av barnen dör eller glömmer sitt tal är kontonumret försvunnet för alltid.
2) Anta att den som fått 69241 istället påstår att han fått 23716. När de andra avslöjar sina tal tror de att kontonumret är 95259, medan den som ljugit vet det rätta numret (förutsatt att ingen annan varit lika listig).

2. Bakgrundsfakta. För att beskriva en algoritm som ger ett annat resultat behöver vi litet bakgrundsinformation.

Låt p vara ett primtal. Vi kommer att räkna med talen $0, 1, 2, \ldots, p-1$ modulo p. Detta innebär att vi har de vanliga räknesätten $+$, - och $\cdot$ men att vi bara är intresserade av vilken rest svaret ger vid division med p. För att det ska fungera bra att fortsätta att räkna är det viktigt att notera att resultatets rest vid division med p endast beror på operandernas rest vid division med p. (Visa detta.) Vi kommer att skriva $+$, - och $\cdot$ som vanligt medan vi använder likhetstecken med 3 streck och lägger till $\pmod p$ efteråt för att vi bara är intresserade av vilken rest talen ger vid division med p. Till exempel har vi

$$4 \cdot 6 \equiv 3 \pmod 7$$
$$16 + 4 \equiv 1 \pmod{19}$$
$$3 - 7 \equiv 7 \pmod{11}$$

Du kan läsa mer om detta sätt att räkna i boken av Hardy och Wright som nämns i litteraturlistan. För att den här definitionen skall fungera behöver p inte vara primtal men när vi nu skall definiera division underlättar det.

Om $b \not\equiv 0 \pmod p$ definieras a/b som det tal $c \pmod p$ som uppfyller $c \cdot b \equiv a \pmod p$. T.ex. har vi $6/11 \equiv 16 \pmod{17}$.

Visa att ett sådant tal finns och är unikt.

(Studera talen b, $2\cdot b \ldots p\cdot b \pmod p$. Visa att de olika $\pmod p$. Ett måste vara a.)

Det finns effektivare sätt att göra division $\pmod p$ om p är stort. *Försök finna något.* Ett sätt presenteras under *Bra att veta* i slutet av uppgiften.

3. En ny delningsalgoritm. Med vår nya notation kan vi säga att om den första siffran i kontonumret är s_1 och första siffran i de tre barnens tal är $s_1^{(1)}, s_1^{(2)}$ och $s_1^{(3)}$, * så gäller

$$s_1 \equiv s_1^{(1)} + s_1^{(2)} + s_1^{(3)} \pmod{10}$$

och att $s_1^{(1)}$ och $s_1^{(2)}$ valdes slumpmässigt. (Vi kommer i fortsättningen bara att beskriva vad som händer med första siffran i kontonumret. Vi förutsätter att de andra siffrorna behandlas på samma sätt och eventuella slumptal väljs oberoende.)

Den nya algoritmen kommer att vara liknande och fungerar på följande sätt. Du väljer ett slumpmässigt tal b_1, $0 \le b_1 \le 10$ och nu blir de tre bitarna

$$\begin{aligned} s_1^{(1)} &\equiv s_1 + b_1 \pmod{11} \\ s_1^{(2)} &\equiv s_1 + 2\cdot b_1 \pmod{11} \\ s_1^{(3)} &\equiv s_1 + 3\cdot b_1 \pmod{11} \end{aligned}$$

Dessa bitar är nu inte siffror i ett tal utan tal i intervallet $[0, 10]$ men det spelar inte så stor roll för problemet. Om vi således delar

*Här använder vi litet matematisk notation. Dessa tal är inte potenser, vi använder övre och undre index för att hålla ordning på informationen. Vi låter $s_2^{(3)}$ t.ex. betyda det tredje barnets andra siffra och $s_1^{(2)}$ betyder det andra barnets första siffra o.s.v.

numret 31784 på detta sätt med $b_1 = 2 \quad b_2 = 5 \quad b_3 = 0 \quad b_4 = 7 \quad b_5 = 5$ får de tre syskonen:

$(5, 6, 7, 4, 9)$	Syskon 1
$(7, 0, 7, 0, 3)$	Syskon 2
$(9, 5, 7, 7, 8)$	Syskon 3.

Detta sätt att dela hemligheten har följande egenskaper:

1) Två syskon tillsammans kan ta reda på kontonumret.
2) Inget syskon har någon aning om numret, förutom antalet siffror.

Visa dessa egenskaper.

Diskutera eventuella problem med denna fördelningsalgoritm.

4. Generalisering. En naturlig fråga är nu i vilken grad de givna algoritmerna går att generalisera. Anta att vi vill fördela en hemlighet bland n personer på ett sådant sätt att t personer tillsammans kan ta reda på hemligheten medan $t-1$ personer inte får någon information även om de samarbetar. Vi har följande grundidé. Låt s vara en siffra i hemligheten. Ta ett polynom av gradtal $t-1$

$$Q(x) = s + b_1 \cdot x + b_2 \cdot x^2 \ldots b_{t-1} x^{t-1}$$

där $b_1, b_2 \ldots b_{t-1}$ är slumpmässigt valda heltal under villkoret att $0 \leq b_i \leq 10$. Sedan får den i'te personen $Q(i) \pmod{11}$.

Visa att denna delningsalgoritm har de önskade egenskaperna om $n < 11$. D.v.s. om t personer tillsammans kan rekonstruera s medan $t-1$ personer inte får någon information om s. Det finns information som kan vara till hjälp under rubriken *Bra att veta* i slutet av denna beskrivning.

Hur klarar man större n?

Om exakt t personer försöker rekonstruera hemligheten har vi ett liknande problem som tidigare, nämligen att en oärlig person kan ge en felaktig bit och förstöra resultatet. För att motverka det kan man försöka följande. Vi byter ut 11 mot ett stort primtal p och sätter

$$Q(x) = s + b_1 \cdot x + b_2 \cdot x^2 \ldots b_{t-1}x^{t-1}$$

med $0 \leq b_i \leq p-1$ slumpmässigt valda. Som förut är den i'te biten $Q(i) \pmod p$.

Om man nu fuskar genom att ändra sin bit slumpmässigt blir s antagligen något slumpmässigt tal mellan 0 och $p-1$. Eftersom vi vet att $0 \leq s \leq 9$ upptäcker man troligtvis att någon fuskar. (Dock lyckas ju fusket i den mån att den som fuskar kan räkna ut s.)

Visa att denna metod inte är särskilt bra. I själva verket kan en person som vet de andra personernas i byta ut s mot $s+1$ eller $s+d$ för något d personen väljer själv. Det kan vara bättre att som i'te bit ge ut två tal. Dels ett slumpvis valt tal a_i och dels värdet av polynomet i denna punkt, $Q(a_i) \pmod p$.

5. Förslag till uppgifter. Efter denna inledning kan din egen kreativitet ta vid.

FÖRSLAG.

1. Försök att hitta andra problem med de presenterade algoritmerna och försök hitta motåtgärder.

2. Försök att hitta praktiska användningar av de givna idéerna. I dagens digitaliserade samhälle finns mycket information som skall hållas hemlig men som en grupp personer bör veta. Utveckla gärna program och sälj till företag som saknar matematisk expertis.

3. Hitta på andra sätt att dela hemligheter.

4. Implementera någon hemlighets–delningsalgoritm. Antingen den allmänna givna eller din egen.

Om du vill göra något mer avancerat, läs också *Offentlig kryptering* i denna volym och kombinera metoderna.

Bra att veta. Faktorsatsen är sann om vi räknar med polynom modulo p för ett primtal p. Det vill säga om $Q(a) \equiv 0 \pmod{p}$ kan vi skriva

$$Q(x) \equiv (x-a)(P(x) \pmod{p} \quad \text{för något polynom } P(x).$$

Använd detta för att visa att ett polynom av gradtal t har högst t nollställen.

Med ett givet talpar $(a_i, b_i) \quad i = 1, 2 \ldots t$ kan vi hitta ett polynom av gradtal $t-1$ som uppfyller $Q(a_i) \equiv b_i \pmod{p}$ förutsatt att $a_i \not\equiv a_j \mod p$ för $i \not\equiv j$. Vi kan nämligen ta

$$Q(x) = \sum_{i=1}^{t} b_i \prod_{j \neq i} \frac{(x - a_j)}{(a_i - a_j)}.$$

Visa att Q är unikt. (Om Q_1 och Q_2 är två polynom som interpolerar (a_i, b_i) så har $Q_1 - Q_2$ t nollställen.)

Slutligen är Euklides' algoritm användbar.

Euklides används till att beräkna största gemensamma delare av två tal och att med givna y_1, y_2 och m beräkna $y_1/y_2 \pmod{p}$.

Låt oss börja med största gemensamma delare. Största gemensamma delare av två tal a och b betecknas med (a, b) och är det största tal som delar både a och b. Idén bakom algoritmen är att om ett tal delar a och b så delar det också $a - k \cdot b$ för alla heltal k. Algoritmer beskrivs nu kanske enklast med ett exempel. Låt oss ta talen 534 0ch 114. Anta att d delar dessa två tal, då delar det också

$$534 - 4 \cdot 114 = 78$$

och också

$$114 - 78 = 36$$

och också

$$78 - 2 \cdot 36 = 6$$

och också

$$36 - 6 \cdot 6 = 0.$$

Nu slutar algoritmen eftersom vi fick talet 0. Det sista talet 6 kontrolleras lätt vara svaret.

Låt oss beskriva hur man beräknar $y_1/y_2 \pmod{p}$.

Först beräknar man $e \equiv 1/y_2 \pmod{p}$. Sedan blir svaret $y_1 \cdot e \pmod{p}$. Eftersom p är primtal är $(p_1, y_2) = 1$ då $1 \leq y_2 \leq p-1$. Vi beräknar ändå (p_1, y_2) med Euklides' algoritm. Låt $p = 91$ och $y_2 = 53$, vilket ger

$$\begin{aligned} 91 - 53 &= 38 \\ 53 - 38 &= 15 \\ 38 - 2 \cdot 15 &= 8 \\ 15 - 8 &= 7 \\ 8 - 7 &= 1. \end{aligned}$$

Låt oss använda ekvationerna baklänges

$$\begin{aligned} 1 &= 8 - 7 = 8 - (15 - 8) = 2 \cdot 8 - 15 = 2 \cdot (38 - 2 \cdot 15) - 15 \\ &= 2 \cdot 38 - 5 \cdot 15 = 2 \cdot 38 - 5 \cdot (53 - 38) \\ &= 7 \cdot 38 - 5 \cdot 53 = 7 \cdot 91 - 12 \cdot 53. \end{aligned}$$

Ur detta följer att

$$12 \cdot 53 \equiv -1 \pmod{91}$$

och

$$(-12) \cdot 53 \equiv 1 \pmod{91}$$

och således

$$79 \cdot 53 \equiv 1 \pmod{91}$$

d.v.s.

$$\frac{1}{53} \equiv 79 \pmod{91}.$$

Litteratur

Att dela en hemlighet på det angivna sättet föreslogs först i

Shamir, A., How to share a secret. *Communications of ACM* 22 (11) (1979).

Ett ställe där du kan läsa mer om modulo räkning och elementär talteori är

Hardy, G.H., & Wright, E.M., *An introduction to the theory of numbers.* Fifth edition, Oxford Univ. Press, Oxford 1979.