Grunder i matematik och logik (2021)

Talteori

Marco Kuhlmann och Victor Lagerkvist

3.01 RSA (uppkallad efter matematikerna Ron Rivest, Adi Shamir och Leonard Adleman) är en välkänd krypteringsalgoritm. Den använder två olika nycklar för kryptering och dekryptering: en *offentlig nyckel*, som består av två heltal n och e, och en *hemlig nyckel*, som består av n och ett tredje heltal d. När man krypterar ett meddelande m så beräknar man ett tal c enligt formeln $c = m^e \bmod n$. Med detta menas att c är den rest som man får när man dividerar m^e med n. Klartexten kan fås tillbaka från det krypterade meddelandet enligt formeln $m = c^d \bmod n$. Finessen med RSA är att man inte kan beräkna d, och därmed inte heller m, utifrån n och e – åtminstone inte inom rimlig tid.

3.02 För att beskriva och beräkna RSA behöver man känna till ett antal grundläggande begrepp från den gren av matematiken som kallas *talteori*, bland annat primtal, största gemensamma delare och heltalsdivision. Dessa begrepp är t.ex. relevanta för att förstå hur talen n, e och d genereras:[1]

1. Välj två olika, stora primtal p och q.
2. Låt $n = pq$ och låt $\phi = (p-1)(q-1)$.
3. Välj ett tal $1 < e < \phi$ så att den största gemensamma delaren av e och ϕ är 1.
4. Beräkna sedan ett heltal d sådant att $ed \bmod \phi = 1$.

Målet med denna föreläsning är att introducera dessa begrepp.

3.03 Denna föreläsning har även som ambition att ge exempel på hur större matematiska argumentkedjor kan se ut. Sådana kedjor följer ofta mönstret definition–sats–bevis, där man först definierar matematiska koncept och sedan bevisar relevanta egenskaper hos dessa koncept. När en sats (eller en *lemma*, en hjälpsats) väl har bevisats kan den användas som ”hjälpfunktion” i andra bevis. Föreläsningen exemplifierar även ett antal bevistekniker, som t.ex. direkt bevis, indirekt bevis och induktionsbevis.

3.04 Symbolen $\mathbb{Z}$ betecknar mängden av alla heltal; symbolen $\mathbb{N}$ betecknar mängden av alla icke-negativa heltal (naturliga tal) $n \geq 0$.

[1] baserad på `http://sv.wikipedia.org/wiki/RSA`

Heltalsdivision

3.05 **Heltalsdivision.** När vi dividerar 14 med 4, så blir resultatet inte ett heltal. Ett sätt att uttrycka resultatet som heltal är med hjälp av **kvot** och **rest**. Med *kvot* betecknar vi det hela antal gånger som 4 går i 14, i det här fallet alltså 3; med *rest* betecknar vi det som blir kvar: $14 - 3 \cdot 4 = 2$.

$$\frac{14}{4} = \text{kvot } 3 \text{ rest } 2$$

I fortsättningen kommer vi alltid mena den *principala resten* när vi skriver *rest*. Detta är resten som vi får när vi låter kvoten vara det maximala antalet gånger som det ena talet går i det andra talet.

3.06 Om två heltal a och b har samma rest vid division med ett heltal n, så säger vi att a och b är **kongruenta** modulo n och skriver

$$a \equiv b \pmod{n} \qquad \text{eller} \qquad a \bmod n = b \bmod n$$

Exempelvis gäller att $2 \equiv 26 \bmod 24$.

3.07 I Python kan vi använda dessa operatorer för heltalsdivision:

```
14 // 4           # returnerar kvoten (3)
14 % 4            # returnerar den principala resten (2)
divmod(14, 4)     # returnerar paret (14 // 4, 14 % 4)
```

För den matematiska operation som används vid kryptering och dekryptering i RSA finns en speciell funktion i Python:

```
pow(a, b, n)      # ger samma resultat som pow(a, b) % n
```

Delbarhet

3.08 Vi börjar med en fundamental definition.

Definition 3.1 Vi säger att ett heltal $a \neq 0$ **delar** ett heltal b om det finns ett heltal m sådant att $b = ma$. Detta skriver vi som $a \mid b$. Vi säger att a är en **delare** till b.

Det gäller att $8 \mid 32$ (ty $32 = 4 \cdot 8$) och att $16 \mid 256$ (ty $256 = 16 \cdot 16$), men däremot inte att 8 är en delare till 30 eller att 16 är en delare till 100.

3.09 Det finns flera välkända **delbarhetsregler**. Ett tal är …

delbart med	om
2	talets sista siffra är ett 0, 2, 4, 6 eller 8
3	talets siffersumma är delbar med 3
4	talet som bildas av de två sista siffrorna är delbart med 4
5	talets sista siffra är 0 eller 5
6	talet är delbart med 2 och 3
8	talet som bildas av de tre sista siffrorna är delbart med 8
9	talets siffersumma är delbar med 9
10	talet sista siffra är 0

3.10 En mer formell delbarhetsregel ges av följande:

Lemma 3.1 För alla $a, b_1, b_2 \in \mathbb{Z}$ gäller:

1. Om $a \mid b_1$ och $a \mid b_2$ så även $a \mid b_1 + b_2$.

2. Om $a \mid b_1$ och $a \mid b_2$ så även $a \mid b_1 - b_2$.

Bevis Vi bevisar endast påstående 1; beviset av påståendet 2 är nästan identiskt. Om $a \mid b_1$ och $a \mid b_2$ så finns $m_1, m_2 \in \mathbb{Z}$ sådana att $b_1 = m_1 a$ och $b_2 = m_2 a$. Vi kan skriva

$$b_1 + b_2 = m_1 a + m_2 a = (m_1 + m_2) \cdot a \,.$$

Alltså finns det ett $m \in \mathbb{Z}$ sådant att $b_1 + b_2 = ma$, nämligen $m = m_1 + m_2$. Detta visar att $a \mid b_1 + b_2$.

Lemma 3.2 För alla $a, b_1, b_2 \in \mathbb{Z}$ gäller: Om $a \mid b_1$ och $a \mid b_2$ så gäller även att $a \mid n_1 b_1 + n_2 b_2$, för godtyckliga $n_1, n_2 \in \mathbb{Z}$.

Bevis Generalisering av beviset till Lemma 3.1.

Bevis Om $a \mid b_1$ och $a \mid b_2$ så finns $m_1, m_2 \in \mathbb{Z}$ sådana att $b_1 = m_1 a$ och $b_2 = m_2 a$. Låt $n_1, n_2 \in \mathbb{Z}$. Vi kan skriva

$$n_1 b_1 + n_2 b_2 = n_1 m_1 a + n_2 m_2 a = (n_1 m_1 + n_2 m_2) \cdot a \,.$$

Alltså finns det ett $m \in \mathbb{Z}$ sådant att $n_1 b_1 + n_2 b_2 = ma$, nämligen $m = n_1 m_1 + n_2 m_2$. Detta visar att $a \mid n_1 b_1 + n_2 b_2$.

Primtal

3.11 Vi börjar med att definiera primtal.

Definition 3.2 Ett heltal $p > 1$ är ett **primtal** om dess enda positiva delare är 1 och p.

De tio första primtalen är 2, 3, 5, 7, 11, 13, 17, 19, 23, 29. Det största idag kända primtalet har 24 862 048 siffror. Talet upptäcktes 2018-12-07.[2]

[2] `http://en.wikipedia.org/wiki/Largest_known_prime_number`

3.12 Den klassiska metoden för att hitta primtal är en algoritm som heter **Eratosthenes' såll**: Man börjar med att lista alla heltal från och med 2 till och med ett största tal n. Sedan fortsätter man i ett antal omgångar där man i varje omgång markerar det minsta talet som finns kvar i listan som primtal och stryker alla multipler av det just markerade talet, förutom talet själv. (En del av dessa må redan vara strukna.) På det viset sållar man så småningom bort alla tal som *inte* är primtal.

3.13 Primtal är intressanta eftersom varje tal kan skrivas som en produkt av primtal:

Lemma 3.3 Varje heltal $a > 1$ kan skrivas som en produkt av primtal.

Exempelvis är $693 = 3^2 \cdot 7 \cdot 11$ och $286 = 2 \cdot 11 \cdot 13$.

Bevis Induktion över a.

- För $a = 2$ är saken klar eftersom talet 2 i sig självt är ett primtal och därmed sin egen primtalsfaktorisering (en produkt med en enda faktor).
- För att bevisa lemmat för $a > 2$ antar vi att varje naturligt tal n sådant att $1 < n < a$ har en primtalsfaktorisering. Nu finns det två möjligheter: Antingen är a ett primtal och därmed sin egen primtalsfaktorisering (en produkt med en enda faktor), eller så har a en delare $d \in \mathbb{Z}$ med $1 < d < a$. Då kan a skrivas som $a = dn$ med $1 < n < a$. Nu kan vi utnyttja induktionsantagandet och skriva både d och n som produkter av primtal. Då är även produkten av dessa primtalsfaktoriseringar en primtalsfaktorisering, och närmare bestämd en primtalsfaktorisering av a.

3.14 Lemma 3.3 kan förstärkas till följande sats:

Sats 3.1 (Aritmetikens fundamentalsats) Varje heltal $a > 1$ kan skrivas som en produkt av primtal. Denna faktorisering är unik sånär som på ordningen av faktorerna.

Denna sats kommer vi tyvärr inte kunna bevisa i ramen av denna kurs.

3.15 Man känner inte till någon effektiv metod för att beräkna primtalsfaktoriseringen för ett givet tal. Frågan om det finns en algoritm som beräknar svaret i polynomisk tid är ett av de viktigaste olösta problem inom datalogin. Om det skulle finnas en sådan metod skulle metoden RSA vara värdelös, eftersom man då skulle kunna räkna ut primtalen p och q och därmed den hemliga nyckeln utifrån talet n i den publika nyckeln.

3.16 En annan viktig sats inom talteori är denna:[3]

Sats 3.2 (Euklides' sats) Det finns oändligt många primtal.

[3]Euklides (ca. 325–265 f.Kr.), grekisk matematiker

Denna sats innebär goda nyheter för RSA eftersom den säger att det är omöjligt att knäcka algoritmen genom att skapa en databas över alla primtal: En sådan databas kommer nödvändigtvis vara ofullständig.

Bevis Låt $p_1, \ldots, p_n$ vara en godtycklig men ändligt lång lista av primtal, $n \geq 1$. Vi kommer att visa att det finns minst ett primtal som inte finns med i denna lista; det måste därför finnas oändligt många primtal. Låt $a = p_1 \cdot \cdots \cdot p_n$ och låt $b = a + 1$. Om b är ett primtal är beviset klart, för då kan vi konstatera att det finns det åtminstone ett primtal (nämligen b) som inte finns med i listan. Om b inte är ett primtal kan vi skriva det som en produkt av primtal (Lemma 3.3). Låt oss välja något primtal p ur denna primtalsfaktorisering. Vi kommer nu att göra ett indirekt bevis: Vi kommer att anta att p finns med på vår lista och visa att detta antagande leder till någonting som är omöjligt. Detta innebär att antagandet måste vara falskt, dvs. att p inte kan finnas med på vår lista. Därmed kommer beviset vara klart. Antag alltså att p finns med på vår lista. Då skulle vi ha $p \mid a$ och $p \mid b$ och därmed även $p \mid b - a$, dvs. $p \mid 1$ (Lemma 3.1). Detta är omöjligt eftersom den enda delaren som 1 har är 1 och $p > 1$.

Beviset är ett klassiskt exempel på ett **indirekt bevis** där man bevisar motsatsen till det påstående som man egentligen vill bevisa och sedan härleder en motsägelse.

Största gemensamma delare

3.17 Två tal kan ha många gemensamma delare. I detta avsnitt ska vi titta på den största.

Definition 3.3 Låt $a, b \in \mathbb{N}$ sådana att inte båda är lika med 0. Den **största gemensamma delaren** till a och b är det största positiva talet bland deras gemensamma delare. Detta tal betecknar vi med $\text{sgd}(a, b)$.

Exempelvis är $\text{sgd}(693, 286) = 11$.

3.18 Ett sätt att beräkna den största gemensamma delaren till två tal är att bestämma talens primtalsfaktoriseringar (Lemma 3.3) och multiplicerar de faktorer som förekommer i båda. Till exempel är $693 = 3^2 \cdot 7 \cdot 11$ och $286 = 2 \cdot 11 \cdot 13$. Dessa gemensamt är faktorn 11. Detta tal är den största gemensamma delaren till 693 och 286. En mycket mer effektiv metod att beräkna den största gemensamma delaren är Euklides' algoritm, som vi kommer att behandla senare i detta avsnitt.

3.19 För att motivera Euklides' algoritm bevisar vi först ett antal räkneregler för den största gemensamma delaren:

Lemma 3.4 För alla $a, b \in \mathbb{N}$ gäller:

$$
\begin{aligned}
\operatorname{sgd}(693, 286) &= \operatorname{sgd}(693 - 286, 286) = \operatorname{sgd}(407, 286) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(407 - 286, 286) = \operatorname{sgd}(121, 286) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(286, 121) && \text{(med identitet 2)} \\
&= \operatorname{sgd}(286 - 121, 121) = \operatorname{sgd}(165, 121) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(165 - 121, 121) = \operatorname{sgd}(44, 121) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(121, 44) && \text{(med identitet 2)} \\
&= \operatorname{sgd}(121 - 44, 44) = \operatorname{sgd}(77, 44) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(77 - 44, 44) = \operatorname{sgd}(33, 44) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(44, 33) && \text{(med identitet 2)} \\
&= \operatorname{sgd}(44 - 33, 33) = \operatorname{sgd}(11, 33) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(33, 11) && \text{(med identitet 2)} \\
&= \operatorname{sgd}(33 - 11, 11) = \operatorname{sgd}(22, 11) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(22 - 11, 11) = \operatorname{sgd}(11, 11) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(11 - 11, 11) = \operatorname{sgd}(0, 11) && \text{(med identitet 3)} \\
&= \operatorname{sgd}(11, 0) && \text{(med identitet 2)} \\
&= 11 && \text{(med identitet 1)}
\end{aligned}
$$

Figure 1: Beräkning av $\operatorname{sgd}(693, 286) = 11$ med hjälp av reglerna i Lemma 3.4.

1. $\operatorname{sgd}(a, 0) = a$
2. $\operatorname{sgd}(a, b) = \operatorname{sgd}(b, a)$
3. $\operatorname{sgd}(a, b) = \operatorname{sgd}(a - b, b)$

Med dessa identiteter kan vi beräkna $\operatorname{sgd}(693, 286)$ som i Figur 1.

Bevis Vi bevisar de tre utsagorna separat:

1. Vi antar att $a \neq 0$. Den största delaren till a är a. Mängden av alla delare till 0 är mängden av alla heltal, förutom 0. Den största *gemensamma* delaren är alltså a.
2. Kan bevisas med ett venndiagram över de gemensamma delarna.
3. Vi vet att $\operatorname{sgd}(a, b) \mid a$ och att $\operatorname{sgd}(a, b) \mid b$, och från detta får vi $\operatorname{sgd}(a, b) \mid a - b$ med hjälp av Lemma 3.1. Nu vill vi visa att $\operatorname{sgd}(a, b)$ är den *största* gemensamma delaren till $a - b$ och b. Indirekt bevis. Antag att det finns ett $g > \operatorname{sgd}(a, b)$ sådant att $g \mid a - b$ och $g \mid b$. Återigen med Lemma 3.1 får vi $g \mid a$, så g är en gemensam delare till både a och b. Men detta är omöjligt, eftersom $\operatorname{sgd}(a, b)$ per definition är den *största* gemensamma delaren till a och b; det kan inte finnas en gemensam delare som är större än denna. Detta betyder att $\operatorname{sgd}(a, b)$ även måste vara den största gemensamma delaren till $a - b$ och b.

a	b	a % b
693	286	121
286	121	44
121	44	33
44	33	11
33	11	0
11	0	–

Figure 2: Euklides' algoritm, exempel.

3.20 **Euklides' algoritm** är en effektiv algoritm för att beräkna den största gemensamma delaren till två tal $a, b \in \mathbb{N}$. Dess effektivitet bygger på observationen att beräkningen i Figur 1 har en speciell struktur: Om vi bortser från den sista raden så kan den delas in i *faser* där varje fas ser ut så här:

- Tillämpa identitet 3 så länge som $a - b \geq 0$.
- Tillämpa identitet 2.

I slutet hamnar vi ett läge där vi kan tillämpa identitet 1. Den upprepade tillämpningen av identitet 3 kan undvikas genom att inse att det vi gör med dessa är att beräkna resten av en heltalsdivision (se 3.05). Om vi som exempel tar kedjan från $\text{sgd}(693, 286)$ till $\text{sgd}(121, 286)$, som består av 2 stycken tillämpningar av identitet 3, så inser vi att 121 är resten av heltalsdivisionen 693/286.

3.21 Med hjälp av heltalsdivision kan vi förkorta varje fas i vår beräkning genom följande identitet:

Lemma 3.5 För alla $a, b \in \mathbb{N}$ gäller $\text{sgd}(a, b) = \text{sgd}(b, a \bmod b)$.

Bevis Vi kombinerar identiteterna 2 och 3 i Lemma 3.4:

$$\begin{aligned} \text{sgd}(a, b) &= \text{sgd}(a \bmod b, b) \quad (a \text{ div } b \text{ stycken tillämpningar av identitet 3}) \\ &= \text{sgd}(b, a \bmod b) \quad (\text{identitet 2}) \end{aligned}$$

3.22 Nu kan vi ange Euklides' algoritm som en enradare i Python:

```
def gcd(a, b):    # greatest common divisor
    return a if b == 0 else gcd(b, a % b)
```

Ett konkret anrop illustreras i Figur 2.

3.23 För att se att algoritmen terminerar, notera att $0 \leq a \bmod b < b$. Funktionens andra argument blir alltså mindre och mindre med varje steg, utan att någon gång bli mindre än 0. Detta innebär att funktionen så småningom anropas med ett andra argument som är 0, och då terminerar den.

3.24 I samband med RSA-algoritmen är det intressant att förutom den största gemensamma delaren av två tal a och b också beräkna två koefficienter x och y sådana att

$$ax + by = \text{sgd}(a, b)$$

Denna ekvation kallas **Bézouts identitet**[4] Det är inte uppenbart att två tal x och y som uppfyller den önskade egenskapen existerar, och vi kommer inte heller bevisa det i denna kurs; men det visar sig att Bézout-koefficienterna inte bara finns utan även kan beräknas, på ett effektivt sätt, med en utvidgad version av Euklides' algoritm (eng. *extended Euclidean algorithm*).

3.25 Två tal a och b kallas **relativt prima** om $\text{sgd}(a, b) = 1$. Detta innebär alltså att a och b inte har några gemensamma delare, förutom den triviala delaren 1. Exempelvis är 12 och 25 relativt prima (men inga primtal).

3.26 Koefficienterna i Bézouts identiet är intressanta eftersom man med hjälp av dessa kan beräkna ett tals **invers** (eng. *modular multiplicative inverse*), vilket man behöver göra i sista steget av nyckelgenereringsprocessen i RSA. Där har man två tal e och ϕ som är relativt prima, och uppgiften är att beräkna ett tal d sådant att $ed \bmod \phi = 1$, dvs. den multiplikativa inversen till e modulo ϕ. Med hjälp av den utvidgade versionen av Euklides' algoritm får vi två Bezóut-koefficienter x och y sådana att

$$ex + \phi y = \text{sgd}(e, \phi) \quad \iff \quad ex + \phi y = 1 \quad \iff \quad ex - 1 = (-y)\phi$$

Detta innebär alltså att $\phi \mid ex-1$ (se definition 3.1), och därmed att $ex \bmod \phi = 1$. Bézout-koefficienten x är alltså exakt det tal d som vi söker.

[4] Étienne Bézout (1730–1783), fransk matematiker