

Föreläsning 4: Sammanfattning av alla räkneproblem, samt cykler · 1MA020

Vilhelm Agdur¹

¹ vilhelm.agdur@math.uu.se

29 januari 2024

Vi blickar tillbaka på de olika räkneproblem vi har studerat i den första delen av kursen, och börjar skönja ett mönster. Vi skriver upp tabellen av problem vi studerat hittills, och fyller sedan i de återstående cellerna.

Till slut diskuterar vi en jobbigare variant av problemet med människor som sitter runt ett runt bord, och använder denna för att introducera konceptet av cykler i en permutation.

Den tolvfaldiga vägen

Vi har studerat ett antal olika räkneproblem hittills i denna kurs, från de relativt enkla – antalet permutationer – till de mer invecklade, som antalet surjektioner. Vi har ett antal gånger talat om ifall objekt skall vara särskiljbara eller osärskiljbara, och också ifall lådor får lov att vara tomma eller måste innehålla minst ett objekt.

När vi talade om kompositioner fördelade vi osärskiljbara objekt i särskiljbara lådor – och studerade både fallet där lådor fick vara tomma, och när de inte fick det. För surjektioner fördelade vi särskiljbara objekt i särskiljbara lådor, och kravet var att varje låda får ett objekt.

Vi börjar ana ett mönster här i hur våra problem kan se ut. Vi kan ha

1. Särskiljbara objekt och särskiljbara lådor
2. Osärskiljbara objekt och särskiljbara lådor
3. Särskiljbara objekt och osärskiljbara lådor
4. Osärskiljbara objekt och osärskiljbara lådor

och vi kan ha olika krav på hur objekten fördelas i lådorna

1. Generell – lådor får vara tomma och får innehålla hur många objekt som helst
2. Surjektiv – varje låda måste innehålla något objekt
3. Injektiv – ingen låda får innehålla mer än ett objekt

så sammanfattningsvis har vi en tabell med tolv stycken tänkbara kombinatorikproblem. Det kommer visa sig att vi i själva verket redan studerat sju av dem.

Så låt oss rita denna tabell – problem vi redan studerat är i svart text, problem vi inte sett innan är i **grön text**. Låt N vara en mängd

av n objekt, och X vara en mängd av x lådor. Vi ser en fördelning av objekt i lådor som en funktion $f : N \rightarrow X$. Varje cell i tabellen beskriver vad en sådan funktion representerar samt hur många sådana funktioner som finns.

	Generellt f	Injektivt f	Surjektivt f
Bägge särskiljbara	Ord ur X av längd n x^n	Permutation ur X av längd n $\frac{x!}{(x-n)!}$	Surjektion från N till X $x! \{x\}^n$
Osärskiljbara objekt	Multi-delmängd av X av storlek n $\binom{n+x-1}{n}$	Kombinationer: delmängd av X av storlek n $\binom{x}{n}$	Kompositioner av n av längd x $\binom{n-1}{n-x}$
Osärskiljbara lådor	Mängdpartition av N $i \leq x$ delar $\sum_{k=1}^x \{n\}_k$	Mängdpartition av N $i \leq x$ delar av storlek 1 1 om $n \leq x$, 0 annars	Mängdpartition av N i x delar $\{x\}^n$
Bägge osärskiljbara	Heltalspartition av n i $\leq x$ delar $p_x(n+x)$	Sätt att skriva n som summan av $\leq x$ ettor 1 om $n \leq x$, 0 annars	Heltalspartitioner av n i x delar $p_x(n)$

Låt oss ge argument för varför varje av cellerna i denna tabell faktiskt är vad som påstås.

Raden med bägge särskiljbara är den enklaste att fundera på. Om vi tar ett generellt f räknar vi *alla* funktioner från N till X , utan begränsningar och utan att bekymra oss om särskiljbarhet. Ett annat ord för "funktion från N till X " kan vara "ord ur X av längd n " ifall vi tänker oss $N = [n]$. Att kräva att funktionen är injektiv är samma sak som att kräva att ingen bokstav dyker upp två gånger i ordet, vilket ju var vår definition av en permutation. Om det enda vi kräver är att funktionen är surjektiv är så klart vad vi räknar just surjektioner.

I raden med osärskiljbara objekt så kan vi tänka oss objekten som bollar vi lägger i lådorna. Om vi kan acceptera varje fördelning av bollar – lådor får vara tomma eller innehålla mer än en boll – är ju detta precis vårt scenario med att fördela mynt bland personer, vilket vi just gett namnet multi-delmängder.

Om varje låda bara får innehålla noll eller en boll, och vi inte kan se skillnad på bollarna, bestäms varje fördelning av bollar bara av vilka lådor som har en boll och vilka som inte har en – alltså av en delmängd av lådorna, och delmängden måste ha samma storlek som antalet bollar.

Kräver vi att varje låda måste få en boll är vi i fallet med kompositioner, som vi definierade kort i slutet av biten om multi-delmängder – vi kan få en sådan genom att först ge varje person ett mynt och sedan fördela ut resten av mynten fritt, så vi härleder lätt den formeln ur formeln för multi-delmängder.

I raden med osärskiljbara lådor får vi i stället tänka oss att objekten är distinkta, men vi lägger dem i högar (och tillåter oss högst x olika

högar) istället för i lådor – så vi kan inte se skillnad på olika högar annat än på deras innehåll.

Om vi kräver att vår funktion är surjektiv, alltså att varje av våra x högar måste innehålla ett objekt, så begränsar vi oss till att dela upp våra objekt i exakt x högar, och det problemet har vi ju redan studerat: Det är alltså att vi gör en mängdpartition av objekten i x delar.

Om vi inte kräver att funktionen är surjektiv tillåter vi ”tomma högar”, vilka ju resulterar i att vi har en uppdelning i ett mindre antal faktiska högar. Så formeln är bara att vi summerar den tidigare formeln över varje möjligt antal högar.

Fallet med injektiva f och osärskiljbara lådor är korkat. Vi vill alltså dela upp våra objekt i högst x stycken högar, men kräver att varje hög har som mest ett objekt. Så om $n \leq x$ är detta möjligt – varje objekt får sin egen hög – annars är det omöjligt. Så det finns alltid noll eller ett sätt att göra detta på.

I sista raden, när både objekten och lådorna är osärskiljbara, får vi tänka oss att vi har identiska bollar som vi lägger i högar. Vi kan varken fråga ”vilka bollar ligger i den högen” eller ”vilken av högarna är det där”, vi kan bara se hur många högar av varje storlek det finns. Så låt oss definiera det räkningsproblem detta motsvarar:

Definition 1. En *heltalspartition* av ett heltal n i k delar är ett sätt att skriva n som summan av k stycken heltal större än noll. Ordningen vi skriver heltalen i i summan spelar ingen roll.

Vi betecknar antalet heltalspartitioner av n i k delar med $p_k(n)$.

Till skillnad från varje annat räkneproblem vi studerat hittills finns det ingen enkel formel för $p_k(n)$ i termer av k och n . Men det betyder inte att vi inte kommer kunna säga intressanta kombinatoriska saker om heltalspartitioner.

Exempel 2. Det finns fem heltalspartitioner av 8 i 4 delar. Dessa är:

$$5 + 1 + 1 + 1 = 8$$

$$4 + 2 + 1 + 1 = 8$$

$$3 + 3 + 1 + 1 = 8$$

$$3 + 2 + 2 + 1 = 8$$

$$2 + 2 + 2 + 2 = 8$$

Så i sista raden, när vi kräver att funktionen är surjektiv, alltså att vi har exakt x högar, ges antalet sådana funktioner av $p_x(n)$. Om vi inte kräver att funktionen är surjektiv så räknar vi istället alla heltalspartitioner av n i högst x delar. Antalet sådana ges av $p_x(n+x)$ – tänk att vi ger en etta till varje av de x lådorna, och sedan fördelar resten fritt, för att motivera formeln.

Slutligen är sista radens mellersta cell, när vi kräver att funktionen är injektiv, korkad av ungefär samma skäl som varför cellen ovanför var det. Vad vi kräver är alltså att ingen hög får ha mer än ett objekt – så vi skall skriva n som summan av högst x stycken ettor! Om $n \leq x$ kan vi göra det på ett enda sätt – vi skriver n som summan av n stycken ettor – annars går det inte.

Udda och distinkta heltalspartitioner

Låt oss studera heltalspartitioner lite mer, genom att introducera ett problem som vi i denna föreläsning löser med ett elementärt bevis,² men också kommer att återkomma till senare i kursen och lösa med hjälp av genererande funktioner.

Definition 3. Ett sätt att skriva heltalet n som en summa av ett godtyckligt antal positiva udda tal kallar vi för en *udda* partition av n , och antalet sådana kallar vi för $p_o(n)$.³

Ett sätt att skriva heltalet n som en summa av ett godtyckligt antal positiva tal som alla är olika kallar vi för en *distinkt* partition av n , och antalet sådana kallar vi för $p_d(n)$.⁴

Teorem 4. Det gäller för varje $n \geq 1$ att $p_o(n) = p_d(n)$.

Bevis. Vi bevisar detta genom att hitta en bijektion mellan udda partitioner och distinkta partitioner. Så antag att vi börjar med en udda partition

$$n = \underbrace{\lambda_1 + \dots + \lambda_1}_{n_1} + \underbrace{\lambda_2 + \dots + \lambda_2}_{n_2} + \dots + \underbrace{\lambda_k + \dots + \lambda_k}_{n_k}, \quad (1)$$

där vi alltså skrivit n som summan av n_1 stycken λ_1 , n_2 stycken λ_2 , och så vidare, där varje λ_i är ett udda tal.⁵ Vi kan alltså lika väl skriva att

$$n = n_1\lambda_1 + n_2\lambda_2 + \dots + n_k\lambda_k. \quad (2)$$

Vad vi gör nu är att vi skriver varje n_i i dess binära representation, så att vi har^{6,7}

$$n_1 = 2^{m_1^1} + 2^{m_2^1} + \dots + 2^{m_{j_1}^1},$$

$$n_2 = 2^{m_1^2} + 2^{m_2^2} + \dots + 2^{m_{j_2}^2},$$

...

$$n_k = 2^{m_1^k} + 2^{m_2^k} + \dots + 2^{m_{j_k}^k}.$$

Så om vi stoppar in dessa binära expansioner av talen n_i i (2) och multiplicerar ut uttrycket får vi att

$$n = 2^{m_1^1}\lambda_1 + 2^{m_2^1}\lambda_1 + \dots + 2^{m_{j_1}^1}\lambda_1 + \dots + 2^{m_1^k}\lambda_k + \dots + 2^{m_{j_k}^k}\lambda_k. \quad (3)$$

² Det vill säga ett som inte kräver några av de verktyg vi kommer se senare i kursen – det är inte alls elementärt i betydelsen “enkelt att finna”.

³ Så till exempel kan vi skriva

$$25 = 5 + 5 + 5 + 3 + 3 + 1 + 1 + 1 + 1$$

för en udda partition av 25.

⁴ Så till exempel kan vi skriva

$$25 = 10 + 6 + 5 + 4$$

för en distinkt partition av 25.

⁵ Och eftersom vi grupperat termerna i summan så så har vi också att $\lambda_i \neq \lambda_j$ för $i \neq j$.

⁶ Här får man ha viss tolerans för många index, för att kunna uttrycka det hela exakt.

⁷ Och här har vi alltså att $m_i^a \neq m_j^a$ för $i \neq j$ för något a , eftersom detta är en binär representation av talen.

Vi hävdar att (3) är en *distinkt* partition av n , och att funktionen som skickar partitionen (1) på partitionen (3) är en bijektion. Om vi kan visa detta kommer det alltså att följa att antalet udda och distinkta partitioner är lika.

Låt oss börja med att bevisa att (3) faktiskt är en distinkt partition. Om två av talen i den summan var lika hade det betytt att vi hade $2^{m_a^i} \lambda_i = 2^{m_b^j} \lambda_j$ för $a \neq b$ och/eller $i \neq j$.

Om dessa två tal är lika hade de också behövt ha precis samma primtalsfaktorer. Eftersom vi började med en udda partition är både λ_i och λ_j udda tal, och har inga tvåor i sin primtalsfaktorisering – så vi måste ha $m_a^i = m_b^j$, och alltså också $\lambda_i = \lambda_j$. Men vi vet att $\lambda_i = \lambda_j$ enbart kan inträffa om $i = j$, och från att $i = j$ ser vi att vi har $m_a^i = m_b^i$, vilket vi ju sett bara inträffar om $a = b$. Så i själva verket har vi också $a = b$, och vi har en motsägelse. Alltså är alla termerna distinkta.

För att se att detta är en bijektion räcker det att se att denna funktion har en väldefinierad invers – alltså att vi givet en partition i distinkta heltal alltid kan återskapa en partition i udda heltal, och att vi, om vi går först i ena riktningen och sedan andra, alltid kommer tillbaka dit vi startade.

Så antag att vi har en partition i distinkta heltal, säg

$$n = \mu_1 + \mu_2 + \dots + \mu_k.$$

Vad vi gör är att vi bryter ut alla tvåor ur varje tal, så att vi skriver dem som en produkt av en potens av två och ett udda tal, alltså

$$n = 2^{\ell_1} \mu'_1 + 2^{\ell_2} \mu'_2 + \dots + 2^{\ell_k} \mu'_k,$$

och sedan samlar ihop termer med samma värde på μ'_i , så att vi får ett uttryck på formen⁸

$$n = \mu'_1 (2^{\ell_1^1} + 2^{\ell_2^1} + \dots + 2^{\ell_{j_1}^1}) + \dots + \mu'_k (2^{\ell_1^k} + 2^{\ell_2^k} + \dots + 2^{\ell_{j_k}^k}).$$

Om vi nu helt enkelt skriver ner partitionen av n i udda heltal μ'_i , där varje uppträder det antal gånger som ges av summan av tvåpotenser, så får vi alltså

$$n = \underbrace{\mu'_1 + \dots + \mu'_1}_{2^{\ell_1^1} + 2^{\ell_2^1} + \dots + 2^{\ell_{j_1}^1}} + \underbrace{\mu'_2 + \dots + \mu'_2}_{2^{\ell_1^2} + 2^{\ell_2^2} + \dots + 2^{\ell_{j_2}^2}} + \dots + \underbrace{\mu'_k + \dots + \mu'_k}_{2^{\ell_1^k} + 2^{\ell_2^k} + \dots + 2^{\ell_{j_k}^k}},$$

vilket vi ser någorlunda enkelt kommer att skickas tillbaka på den distinkta partition vi började med av vår bijektion. Alltså har vi verkligen hittat en invers, och bijektionen är verkligen en bijektion. \square

⁸ Också detta blir ett uttryck som är lite krångligt att skriva ner i full generalitet – notera att vi implicit antagit att $\mu'_1 \neq \mu'_k$ i hur vi skrev detta, eftersom de inte är grupperade tillsammans, men det behöver så klart inte vara sant. Men för att undvika att skriva så hade vi behövt ännu ett lager av index...

Stirlingtalen av första sorten

I förra föreläsningen introducerade vi Stirlingtalen av *andra* sorten, vilket ju leder en till att fråga vad den första sorten är. Så låt oss introducera ett problem till vilket den är lösningen:

Definition 5. Antag att n personer skall sitta runt k stycken osärskiljbara runda bord, och varje bord måste ha minst en person som sitter vid det. Då ges antalet sätt att placera personerna runt borden av $[n]_k$, *Stirlings cykeltal*. Stirlings cykeltal kallas också för *Stirlingtalen av första sorten*.

Till skillnad från Stirlings partitionstal finns det inte någon enkel formel för cykeltalen. Vi kan däremot bevisa följande resultat:

Teorem 6. Det gäller, för alla $n \geq 1$, att

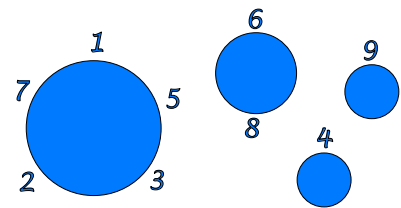
$$\sum_{k=1}^n [n]_k = n!.$$

Bevis. Vi bevisar detta genom att uppvisa en bijektion mellan mängden av permutationer av längd n ur $[n]$, vilka vi vet att det finns $n!$ av, och samlingen av sätt att placera personer runt ett godtyckligt antal runda bord, vilka vi vet per definition räknas av $\sum_{k=1}^n [n]_k$.

Givet ett sätt att placera n personer runt något antal runda bord kan vi definiera en permutation σ genom att, på plats i i permutationen, skriva den person som sitter till vänster om person i runt deras bord. (En person som sitter ensam anser vi sitta till vänster om sig själv.) Detta kommer ge oss en permutation eftersom varje person bara har en person till höger om sig, så ingen person kommer dyka upp två gånger, och varje person har bara en person till vänster om sig, så $\sigma(i)$ är väldefinierat för varje i .

Givet en permutation σ kan vi placera ut personer runt runda bord som följer: Vi börjar med att ställa fram ett bord, och sätta person ett vid det bordet. Sedan sätter vi person $\sigma(1)$ till vänster om person ett, och person $\sigma(\sigma(1))$ till vänster om person $\sigma(1)$, och så vidare. Förr eller senare måste vi komma tillbaka till person 1, eftersom det bara finns ändligt många personer. Om vi placerat alla personerna runt vårt första bord är vi klara.

Om vi har några personer kvar att placera plockar vi fram ett till bord, och sätter den person med lägst nummer som inte redan har en sittplats vid det bordet – säg att det är person j . Sedan upprepar vi processen från innan, och placerar person $\sigma(j)$ till vänster om henne, person $\sigma(\sigma(j))$ till vänster om person $\sigma(j)$, och så vidare. Återigen kommer vi förr eller senare komma tillbaka till person k , och ha gått full cirkel.



Figur 1: Ett exempel på ett sätt placera nio personer vid runda bord. Den motsvarande permutationen till detta sätt att placera personer är 572438169. Det vanliga sättet att skriva detta sätt att placera personer vid bord i text är $(15327)(4)(68)(9)$.

Vi upprepar denna process med fler bord ända tills varje person har fått ett bord, och vi har fått oss ett sätt att placera n personer runt något antal bord mellan 1 och n .

Det är någorlunda enkelt att se, efter att man funderat en stund, att vi alltid kommer komma tillbaka till samma permutation vi började med om vi först skapar en bordsplacering av permutationen, och sedan skapar en permutation av den bordsplaceringen. Alltså är detta en bijektion, och vi har bevisat vår formel. \square

Kommentar 7. Vi har introducerat detta som "personer runt runda bord", men den vanliga matematiska terminologin runt detta är "cykler i en permutation". Hittills har vi bara sett permutationer som en lista av tal i någon ordning, där varje tal mellan 1 och n dyker upp exakt en gång, men detta är bara ett perspektiv på vad en permutation är.

Perspektivet med cykler är ett minst lika vanligt perspektiv på permutationer, och framhäver andra saker man kan använda dem för. För att representera dessa i text skriver vi vanligen i formatet $(15327)(4)(68)(9)$, såsom i Figur 6, i stället för att rita cirklar med tal runt dem.

Låt oss avsluta denna föreläsning med ett enkelt resultat om dessa cykeltal.

Proposition 8. *Det gäller för alla $n \geq 3$ att*

$$\left[\begin{matrix} n \\ n-3 \end{matrix} \right] = \frac{1}{6}(n-2)(n-1)n(n^3 - 6n^2 + 11n - 6).$$

Bevis. Vi har alltså n personer som skall placeras runt exakt $n-3$ stycken runda bord, och inget av borden får bli tomt. Alltså kan vi börja med att välja $n-3$ personer att placera ut runt borden, vilket kan göras på $\binom{n}{n-3}$ sätt.

När det är gjort har vi tre personer kvar att placera vid ett bord – och nu är borden särskiljbara, eftersom vi kan betrakta personen som redan sitter där som en etikett.

Det finns tre olika sätt vi kan välja att placera dessa tre personer vid bord:

1. Vi kan låta varje person få sitt eget bord. Detta kan vi göra på $(n-3)(n-4)(n-5)$ sätt. Eftersom varje bord då bara har en eller två personer vid sig har vi ingen ordning av personerna runt borden att ta hänsyn till.
2. Vi kan placera en person vid ett bord och två personer vid ett annat. För att göra detta behöver vi
 - (a) välja vilka två personer som får sitta vid bordet med tre, vilket kan göras på $\binom{3}{2} = 3$ sätt,

- (b) välja ett bord för den som skall sitta vid ett bord med två, vilket vi kan göra på $n - 3$ sätt,
 - (c) välja ett bord för de som skall sitta vid bordet med tre, vilket vi kan göra på $(n - 4)$ sätt (eftersom ett bord redan är taget av den andre),
 - (d) och till sist välja en ordning på de två personerna som skall hamna på ett bord med tre, vilket vi kan göra på 2 sätt.
3. Vi kan placera alla tre vid samma bord. Vi kan välja vilket bord detta skall bli på $(n - 3)$ sätt, och sedan kan vi välja en ordning de skall sitta i på $3! = 6$ sätt.

Sammanställer vi allt detta, med additions- och multiplikationsprinciperna, så ser vi att vi fått att

$$\begin{aligned} \begin{bmatrix} n \\ n-3 \end{bmatrix} &= \begin{pmatrix} n \\ n-3 \end{pmatrix} ((n-3)(n-4)(n-5) + 6(n-3)(n-4) + 6(n-3)) \\ &= \frac{1}{6}(n-2)(n-1)n(n^3 - 6n^2 + 11n - 6) \end{aligned}$$

såsom önskat. □

Övningar

Övning 1. Antag att vi börjar med den udda partitionen

$$25 = 5 + 5 + 5 + 3 + 3 + 1 + 1 + 1 + 1.$$

Vilken distinkt partition skickas denna på av vår bijektion i Teorem 4?

Antag att vi börjar med den distinkta partitionen

$$30 = 12 + 6 + 5 + 4 + 3.$$

Vilken udda partition skickas på denna av vår bijektion? Räkna också, när du funnit den udda partitionen, ut bijektionen i motsatt riktning för att bekräfta att du får tillbaka vad vi startade med, så att vår invers verkligen är en invers.

Övning 2. Ge ett kombinatoriskt bevis för följande rekursion för Stirlings cykeltal

$$\begin{bmatrix} n+1 \\ k \end{bmatrix} = \begin{bmatrix} n \\ k-1 \end{bmatrix} + n \begin{bmatrix} n \\ k \end{bmatrix}.$$

Övning 3. I slutet av förra föreläsningen talade vi om *derangemang* – alltså permutationer σ sådana att $\sigma(i) \neq i$ för alla i . Om vi i stället tänker på permutationer som sätt att placera personer runt runda bord, hur kan vi se på vår bordsplacering om vår permutation är ett derangemang?

Övning 4. Bevisa att⁹

$$\left\{ \begin{matrix} n+1 \\ k+1 \end{matrix} \right\} = \sum_{j=k}^n \binom{n}{j} \left\{ \begin{matrix} j \\ k \end{matrix} \right\}.$$

Övning 5. Skriv följande permutation av $[10]$ i cykelform

$$8, 9, 4, 10, 5, 7, 3, 2, 6, 1.$$

Övning 6. Härled ett uttryck för

$$\left[\begin{matrix} n \\ n-4 \end{matrix} \right]$$

på liknande sätt som vi gjorde för $\left[\begin{matrix} n \\ n-3 \end{matrix} \right]$.

⁹ Här ber vi alltså inte specifikt om ett kombinatoriskt bevis, även om beviset jag spontant kommer på är sådant. Om ni hittar ett induktionsbevis vore det också intressant.

Ledtråd för det kombinatoriska beviset: Tänk att vi har ett speciellt objekt som är det $n+1$ te, och det får hamna i sin speciella del. Så vi väljer hur stor den delen är och sedan fördelar vi ut resten av objekten.