Kantiansk anskuelse og høyere aritmetikk

Herman Ruge Jervell

Vi gir en vei fra Kants syn på aritmetikk til ufullstendighetsteoremet av Gerhard Gentzen.

Utgangspunkt

I festskriftet til Johan Arnt Myrstad i anledning 50-års dagen har Frode Kjosavik en interessant artikkel "Kants syn på aritmetikk". Der avslutter han med

> Det store spørsmålet er imidlertid hvilken rolle kantiansk anskuelse kan spille ved dannelsen av nye matematiske begreper, m.a.o. om denne typen anskuelse strekker til når vi skal redegjøre for utviklingen av nye formale systemer, eller om vi trenger en rikere form for anskuelse, som den vi finner hos Husserl og Gödel.

Vi tar opp dette spørsmålet og bruker da kantianske begreper fra Kjosaviks artikkel

- Anskuelse
- Ostensive og symbolske konstruksjoner
- Tidsanskuelse
- Romanskuelse

Vi har valgt å se på ufullstendighetsteoremet til Gerhard Gentzen (1936 og 1938) og ikke det av Kurt Gödel (1931). Vi vil begrunne dette valget nedenfor – det er Gentzens argument som knytter an til anskuelse og ikke Gödels argument.

Gödel versus Gentzen

La meg kort skissere Gödels ufullstendighetsteorem. Det er gitt et formelt system for aritmetikk **S** som tilfredsstiller to krav

1. Språket til **S** er rikt nok til å representere syntaksen til **S** som aritmetiske uttrykk (ved hjelp av såkalt Gödel-nummerering)
2. Kalkylen til **S** er kraftig nok til at vi kan vise alle de syntaktiske uttrykkene som vi oppfatter som vanlige regnestykker (kvantorfrie utsagn)

Dette er to ganske svake krav – alle rimelige formelle system for aritmetikk tilfredsstiller dem. Hvis det formelle systemet S er konsistent, så følger:

3. Det fins utsagn F slik at vi verken kan vise F eller negasjonen ¬F i systemet **S** (Gödels første ufullstendighetsteorem)
4. Utsagnet Con(**S**) som uttrykker at S er konsistent er et slikt utsagn (Gödels andre ufullstendighetsteorem)

Det er flere detaljer vi her har hoppet over – men for vår diskusjon skulle dette rekke. Det vesentlige punktet nå er at det i Gödels argument ligger bare en svak forståelse av systemet vi skal vise ufullstendighet for. Det er ingen underliggende analyse av anskuelsen knyttet til system **S**.

Gerhard Gentzens argument involverer systemet på en helt annen måte. Han tar for seg spesielle formelle systemer og bruker ufullstendighetsargumenter for å karakterisere hvilken type anskuelse de formelle systemene klarer å beskrive. Mer spesielt tar han for seg

- Første ordens aritmetikk. Dette formelle systemet innholder elementære aritmetiske aksiomer og i tillegg aksiomskjemaet for induksjon for et hvilket som helst første ordens utsagn. Dette systemet blir ofte kalt Peano-aritmetikk og det fins også en intuisjonistisk variant, Heyting-aritmetikk, med tilsvarende analyse.

Han viser så at det formelle systemet er konsistent ved å bruke en prosess som er styrt av et bestemt ordinaltall, ε_0 – uttalt epsilon-null. Ordinaltallet gir også et ufullstendighetsteorem

- I første ordens aritmetikk kan vi vise at ethvert ordinaltall mindre enn ε_0 er velordnet, mens vi ikke kan vise at ε_0 er velordnet. Velordningen av ε_0 er et utsagn der verken det eller negasjonen kan vises innenfor første ordens aritmetikk.

I resten av artikkelen skal vi vise – i fire trinn – hvordan vi kan få en kantiansk anskuelse av ε_0 . Vi skal ikke gå gjennom Gentzens argument. Det krever for mye forberedelse og plass til å gjennomføres i en artikkel som denne. Men med en anskuelse av ε_0 vil dette sammen med Gentzens argument gi en anskuelse av første ordens aritmetikk.

Det fins analyser av andre formelle systemer à la Gentzen og med andre ordinaltall. Disse kan involvere helt andre prosesser enn Gentzens mest berømte analyse – den for første ordens aritmetikk med bruk av ordinaltallet ε_0 .

Trinn 1 – unære tall

Vi kjenner de unære tall som merker

| || ||| |||| ||||| |||||| ||||||| …

Kant knytter vår forståelse av tall til slike merker og til bruk av dem i telleprosessen. Fra et logisk synspunkt er det mest interessante forståelsen av de tre prikkene til slutt, …, som markerer ”og så videre”. Tidsanskuelsen ligger til grunn for hele telleprosessen, selv om vi avslutter den uten prikker. Men tidsanskuelsens ubegrensethet kommer eksplisitt til uttrykk i de tre prikkene.

Menneskene har brukt tilsvarende merker ved sin telling. Ordene ”komputasjon” og ”kalkyle” henger sammen med henholdsvis ”skjære” og ”småstein”. I moderne versjon snakker vi om datastrukturen unære tall bygd opp fra et startsymbol – vanligvis tatt som 0 – og en konstruktor s som fra et tall gir tallet etter. Vi formulerer det som systemet

$$0 : \mathcal{N}$$
$$\forall x : \mathcal{N}.\ sx : \mathcal{N}$$

Og leser setningene som 0 er med i $\mathcal{N}$, og for alle x om x er med i $\mathcal{N}$, så er også sx med i $\mathcal{N}$.

Her kunne en gå videre – slik det er vanlig i informatikk – å innføre datastrukturer med flere startobjekter og flere konstruktorer. Her er f eks datastrukturen par (med ε som det tomme paret):

$$\varepsilon : \mathcal{P}$$
$$\forall x,y : \mathcal{P}.\ (x.y) : \mathcal{P}$$

og datastrukturen binære tall (med dobbelt bruk av symbolet 1)

$$1 : \mathcal{B}$$
$$\forall x : \mathcal{B}.\ x0 : \mathcal{B}$$
$$\forall x : \mathcal{B}.\ x1 : \mathcal{B}$$

Jeg har moret meg litt her med å skrive konstruktorene på litt forskjellig måte – først som prefiks i de unære tallene, som infiks i par og som postfiks i de binære tallene. Dette er uvesentlig. Det vesentlige er at med prosessen å bygge opp datastrukturen følger det konstruksjonsprinsippet som kalles primitiv rekursjon. Kjosavik tar dette som en naturlig fortsettelse av Kants forståelse av de naturlige tall. Her er en primitiv rekursiv definisjon av addisjon:

$$x + 0 = x$$
$$x + sy = s(x + y)$$

Slike definisjoner ble for første gang brukt på en systematisk måte i en av de store artiklene om aritmetikkens grunnlag – Thoralf Skolems 1923-artikkel "Begründung der elementärer Arithmetik durch die rekurrierende Denkweise ohne Anwendung scheinbarer Veränderlichen mit unendlichem Ausdehnungsbereich."

Trinn 2 – posisjonssystemet

Det er flere problemer med det unære tallsystem:

- Tallene blir omtrent like store som det vi skal telle. Det vil være umulig å telle opp antall atomer i universet uten å bruke hele universet og vel så det.
- Med primitiv rekursjon kan vi definere funksjoner og relasjoner ved å angi en beregningsprosess, men vi trenger å begrunne at denne beregningsprosessen stopper opp og gir riktig resultat.

Det første problemet tok Arkimedes opp i en berømt artikkel ”Sandregneren”. Der viste han på 4 sider hvordan han kunne regne opp alle sandkorn i universet. I moderne språkdrakt viste han hvordan han kunne navngi tall så store som 10^{60} med et argument for at det tallet var stort nok til å telle opp alle sandkorn. Tilsvarende overveielser har vært gjort mange andre steder.

Våre tallord vitner om at det er ikke bare telleprosessen – det å legge til 1 – som er viktig. La oss amatørmessig nevne noen punkter:

- Navnet ti har sammenheng med to hender. En ser klarere rester av det på tysk ”Zehn”
- Navnene elve og tolv kommer av at en har hatt en eller to til overs (left) etter å ha brukt de to hendene.
- Så går en i tier-skritt videre opp til hundre – der vi igjen har rester etter de to hendene.
- Tusen er et stort hundre – tus-hundre.

Poenget er at det ikke er bare den rene telleprosessen, men også prosesser der vi tar større skritt enn bare å legge 1 til det foregående. Dette er godt innenfor en kantiansk oppfattelse av tallene.

Med det indiske posisjonssystemet løser en mange av problemene med det unære tallsystem. Nå kan vi angi tall som gir antall sandkorn i universet på en enkel måte – vi trenger mindre enn en linje til å skrive et slikt tall. La oss se nærmere på problemet med å vise at tallene er velfunderte. Til det lager jeg litt matematikk. Vi innfører et nytt symbol exy som kan tolkes slik

$$e\,xy = 2^x + y$$

Vi kan definere det ved primitiv rekursjon ved

$$e\,0\,y = sy$$
$$e\,sx\,y = e\,x\,e\,xy$$

Litt utskriving av den intenderte tolkingen gjør det klarere

$$e\,0\,y = 2^0 + y = 1 + y = sy$$
$$e\,sx\,y = 2^{x+1} + y = 2^x + 2^x + y = e\,x\,e\,xy$$

Ved bruk av operatorene e og s og konstanten 0 kan vi definere kjempestore tall på en enkel måte. Tallet

$$eeeeeee00000000$$

er et monstertall. Vi får

$$e00 = 2^{0}+0 = 1$$

$$ee000 = 2^{1}+0 = 2$$

$$eee0000 = 2^{2}+0 = 4$$

$$eeee00000 = 2^{4}+0 = 16$$

$$eeeee000000 = 2^{16}+0 = 65536$$

Så begynner det å bli veldig store tall. La oss bare ta for oss neste

$$eeeeee0000000 = 2^{65536}+0$$

Tallet kan skrives ut. Det begynner slik

2003529930406846464979072351560255750447825475569751419265016973710894059556311453089506130
0880933348101038234342907263181822949382118812668869506364761547029165041871916351587966347
2194429309279820843091048559905701593189596395248633723672030029169695921561087649488892540
9080591145703767520850020667156370236612635974714480711177481588091413574272096719015183628
2560618091458852699826141425030123391108273603843767876449043205960379124490905707560314035
0761625624760318637931264847037437829549756137709816046144133086921181024859591523801953310
3029216280016056867010565164675056803874152946384224484529253736144253361437372908830379460
1274724958414864915930647252015155693922628180691650796381064132275307267143998158508811292
6289011342377827055674210800700652839633221550778312142885516755540733451072131124273995629
8271976915005488390522380435704584819795639315785351001899200002414196370681355984046403947
2194016069517690156119726982337890017641517190051133466306898140219383481435426387306539552
9691848834211006047082390658052306380307651402004289099958080829891850866290201218520883474
…

Og etter 3-4 sider med sifre ender tallet med

…1497016110987951336633771378434416194053121445291855180136575558667615019373029691932076
1200092550650815832755084993407687972523699870235679310268041367457189566414318526790547171
6996299036301554564509004480278905570196832831363071899769915316667920895876857229060091547
2919636381673596673959975710326015571920237348580521128117458610065152598883843114511894880
5521291457756991465775300413847171245779650481758563950728953375397558220877750607233944558
7895905719156736

Da har vi regnet ut eeeeee000000. Så skal vi gå enda et skritt videre og regne ut tallet med 7 e-er dvs eeeeeee00000000. Det er vanlig å regne med at antall atomer i universet er et tall på vel 80 sifre. Altså veldig mye mindre. Det samme med antall sekunder siden Big Bang. Det er mindre enn en halv trillion – tallet som starter med 5 etterfulgt av 17 sifre. Men hvordan kan vi begrunne at et slikt monstertall som eeeeeee00000000 er et skikkelig tall? Det nytter ikke å henvise til telleprosessen. Vi må henvise til noen mer kompliserte prosesser enn det å legge til 1. På den neste siden vil vi gjennomføre dette med alle detaljer. Deretter vil vi komme tilbake til de videre trinn med prosesser som svarer til Gentzens ufullstendighetsteorem. Den utålmodige leser kan hoppe over den matematiske turen under.

<u>Start – matematisk tur</u>.

La oss definere noen nye begrep:

$$\mathcal{N}_0 = \mathcal{N}$$

$$x : \mathcal{N}_{i+1} = \forall y : \mathcal{N}_i \, . \, e\, xy : \mathcal{N}_i$$

Observer at våre to aksiomer for $\mathcal{N}$ kommer nå fram som

$$0 : \mathcal{N}_0 = 0 : \mathcal{N}$$

$$0 : \mathcal{N}_1 = \forall y : \mathcal{N} .\ e\ 0y : \mathcal{N} = \forall y : \mathcal{N} .\ sy : \mathcal{N}$$

Vi skal så vise ved bruk av logikk at

$$0 : \mathcal{N}_{i+2}$$

eller med andre ord

$$\forall x : \mathcal{N}_{i+1} .\ sx : \mathcal{N}_{i+1}$$

Så anta at $x : \mathcal{N}_{i+1}$. Det vil si at

$$\forall y : \mathcal{N}_i .\ e\ xy : \mathcal{N}_i$$

Anta nå at $y : \mathcal{N}_i$. Da er $e\ xy : \mathcal{N}_i$ som vi igjen kan sette inn for y over og får

$$e\ x\ exy :\ \mathcal{N}_i = e\ sx\ y : \mathcal{N}_i$$

og har dermed vist

$$\forall y : \mathcal{N}_i .\ e\ sxy : \mathcal{N}_i = sx : \mathcal{N}_{i+1}$$

som skulle vises.

Med dette har vi et kort argument for at vårt store tall eeeeeee00000000 er med i $\mathcal{N}$.

$0 : \mathcal{N}_7$. $0 : \mathcal{N}_6$ gir $e00 : \mathcal{N}_6$. $0 : \mathcal{N}_5$ gir $ee000 : \mathcal{N}_5$. $0 : \mathcal{N}_4$ gir $eee0000 : \mathcal{N}_4$. $0 : \mathcal{N}_3$ gir $eeee00000 : \mathcal{N}_3$. $0 : \mathcal{N}_2$ gir $eeeee000000 : \mathcal{N}_2$. $0 : \mathcal{N}_1$ gir $eeeeee0000000 : \mathcal{N}_1$. Og til slutt $0 : \mathcal{N}$ gir $eeeeeee00000000 : \mathcal{N}$.

Slutt. Matematisk tur.

Det er bemerkelsesverdig at et slikt kort argument kan gi et slikt stort tall. Når det eneste vi vet om $\mathcal{N}$ er at 0 er med og at den er lukket under det å legge til 1, så kunne man tro at et argument for at et tall er med i $\mathcal{N}$ trenger minst like mange trinn som tallet. Veien utenom det er ved å bruke hjelpebegrep og vise hjelpesetninger. Hjelpebegrepene våre er interessante. La oss se på $\mathcal{N}_1$. Vi har

$0 : \mathcal{N}_1$ $\mathcal{N}$ er lukket under 1-er skritt

$1 : \mathcal{N}_1$ $\mathcal{N}$ er lukket under 2-er skritt

$2 : \mathcal{N}_1$ $\mathcal{N}$ er lukket under 4-er skritt

$3 : \mathcal{N}_1$ $\mathcal{N}$ er lukket under 8-er skritt

$4 : \mathcal{N}_1$ $\mathcal{N}$ er lukket under 16-er skritt

og så videre. Vi legger merke til at argumentet for en linje får vi ved å gjenta argumentet for linja over 2 ganger. Tilsvarende får vi for $\mathcal{N}_2$ og videre oppover. De nødvendige detaljene er akkurat det som er skrevet ut på foregående side.

La oss så tenke oss dette som forsøk på å forstå telleprosesser. Da starter vi med en prosess **P** som skal gjentas svært mange ganger. Vi får det til ved å erstatte prosessen **P** med noen mer abstrakte prosesser men som skal gjentas langt færre ganger.

Logisk sett er argumentet fra foregående side interessant. Et direkte bevis for at et stort tall er med i $\mathcal{N}$ trenger minst like mange trinn som tallet. Men vi bruker hjelpebegrep og hjelpesetninger og lager et indirekte bevis. Man kan analysere dette nærmere og finner da ut at den ekstra $\forall$-kvantoren i hjelpebegrepet er helt nødvendig. Uten den ville vi ikke kunne ha lagd et slikt kort bevis.

Vi har (egentlig) brukt 2-tall systemet i argumentet. Vi kunne ha oversatt det til vårt vanlige 10-tallsystem uten problem. Det ville bare ha blitt litt flere detaljer.

Trinn 3 – abstrahere bort grunntall

Kant var opptatt av posisjonssystemet. Følgende sitat fra ”Kritik der Urteilskraft” og gjengitt i Kjosaviks artikkel er interessant:

> Denn in der Verstandesschätzung der Größen (der Arithmetik) kommt man eben so weit, ob man die Zusammenfassung der Einheiten bis zur Zahl 10 (in der Dekadik), oder nur bis 4 (in der Tetradik) treibt, die weitere Größerzeugung aber im Zusammensetzen, oder, wenn das Quantum in der Anschaung gegeben ist, im Auffassen, bloß progressiv (nicht komprehensiv) nach einem angenommenen Progressionsprinzip verrichtet.

Vi kan tenke oss at symbolene 0 og 1 finnes i alle tallsystem og tolkes som null og en. Men tallet 10 vil tolkes forskjellig. I 4-tallsystemet vil det være 4, i 1289-tallsystemet vil det være 1289. Verre blir det med 100. I 4-tallsystemet blir det 16, mens i 1289-tallsystemet vil det være 1661521. Hvis vi abstraherer bort grunntallet vil vi kunne tolke 10 som det første uendelige (ordinal)tall ω, mens 100 tolkes som ω^2. La oss fortsette denne leken litt til. Senere skal vi vise at dette vil kunne være nyttig og ikke går ut over kantianske anskuelser.

I grunntallet **g** kan vi se på tallene som

- Endelig sekvens av tall < **g**
- Sekvensen begynner med et tall > 0, så fremt ikke sekvensen bare består av 0 alene
- Sekvensene ordnes leksikografisk – akkurat som telefonkatalogen, men slik at sekvenser av lengde 1 kommer først, deretter de av lengde 2 og så videre

Vi abstraherer bort grunntallet g ved å bare stryke det over og får da

- Endelig sekvens av tall
- Sekvensen begynner med et tall > 0, så fremt ikke sekvensen bare består av 0 alene
- Sekvensene ordnes leksikografisk – akkurat som telefonkatalogen, men slik at sekvenser av lengde 1 kommer først, deretter de av lengde 2 og så videre

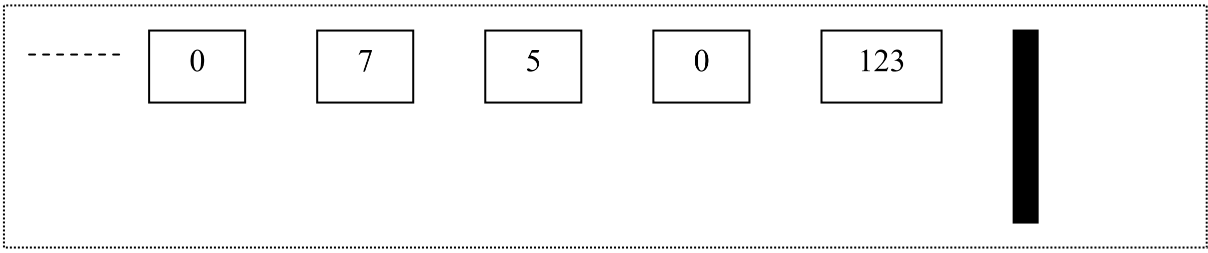
Vi må ha med den siste betingelsen med ordning av sekvenser. Ellers kunne vi ha følgende uendelig nedadstigende kjede

anderaa > aanderaa > aaanderaa > aaaanderaa > aaaaanderaa > ……

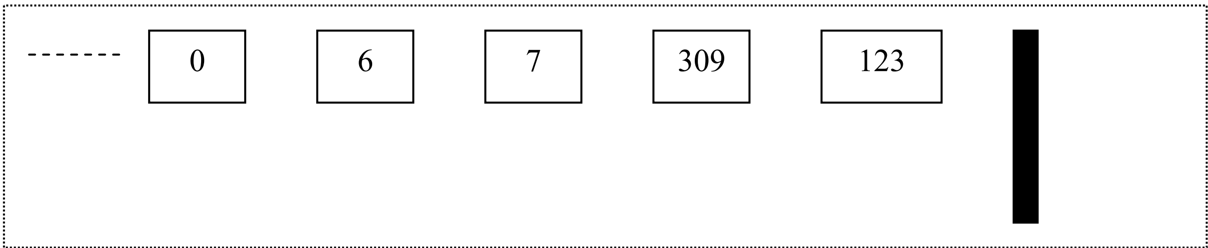
Slik det nå er vil det ikke finnes noen uendelig nedadstigende kjede av endelige sekvenser av tall – ordningen er det som kalles en velordning.

Vi skal lage en prosess der vi kan bruke slike sekvenser. Tenk deg at du har en stor samling av bokser – ordnet i rekke og som ender til høyre i en vegg. I hver boks kan det være et endelig antall kuler. Selve prosessen foregår ved at du tar en kule opp fra en boks og legger ned så mange kuler du vil i bokser til høyre for boksen. Alt her er endelig – det er bare et endelig antall bokser med kuler, til enhver boks er det endelig antall bokser til høyre for den, i hver boks er det bare et endelig antall kuler. Prosessen stopper opp når det ikke er flere kuler å ta opp.

Et bilde kan kanskje gjøre det klarere



Her har vi flere bokser som er på rekke og støter mot en vegg til høyre. Mot venstre kan det være flere bokser men de er alle tomme. Tallet inne i boksene angir hvor mange kuler vi har i boksene. Vi har altså en boks med 7 kuler, en med 5 kuler og en med 123 kuler. Resten er tomme. Et skritt i prosessen foregår ved at vi tar vekk en kule fra en boks og legger til kuler i bokser til høyre for den. Det hele slutter når det ikke er flere kuler igjen i noen av boksene. Et mulig skritt kan da lede til følgende situasjon:



At denne prosessen vil før eller siden stoppe opp er ikke så vanskelig å se. Vi kan representere situasjonene ved endelige sekvenser av naturlige tall. Den første blir representert ved

$$[\,7 , 5 , 0 , 123\,]$$

mens den andre blir representert ved

$$[\,6 , 7 , 309 , 123\,]$$

og vi ser at hvert skritt i prosessen vil minske den tilhørende endelige sekvensen. Nå er de endelige sekvensene velordnet – og etter et endelig antall skritt vil vi få

$$[\,0\,]$$

og det er ikke flere kuler igjen i boksene.

Et nøkkelbegrep her er at en ordning er velordnet. Det betyr at det ikke fins noen uendelig nedadstigende kjede i ordningen – og det er typisk det vi bruker for å vise at en beregning stopper opp etter et endelig antall trinn.

Hva har så dette med kantianske anskuelser å gjøre? Vi forsto overgangen fra det unære tallsystem til posisjonssystemet ved at vi kunne vi fra en utgangsprosess lage nye prosesser der vi tar lengre og lengre skritt. Men her har vi flere nye ting:

- Vi er interessert i prosesser som stopper opp.

- Prosessen er indeterministisk – vi tar ikke stilling til hvor mange nye kuler som legges i boksene til høyre, bare at det legges til nye kuler der og ikke i de andre boksene.
- Sekvensen [1 , 0] svarer til det første uendelige ordinaltall ω . Men ω spiller ikke her rollen som en aktuell uendelighet, men snarere som en markør for der vi senere kan foreta en beslutning om hvor mange kuler vi tar inn i den siste boksen til høyre.
- Vi kan sammenlikne to endelige sekvenser ved å betrakte dem som tall i et posisjonssystem med tilstrekkelig stort grunntall. Men hvor stort dette grunntallet må være er avhengig av vår prosess der vi ikke vet hvilke beslutninger som blir tatt etter hvert.

Frode Kjosavik argumenterer for at regnestykkene knyttet til funksjoner definert ved primitiv rekursjon følger kantianske anskuelser. Det har han helt rett i. Men vi må samtidig argumentere for at regnestykkene stopper opp og gir resultat. Det kan vi argumentere for ved å knytte regnestykkene an til vår prosess med endelige sekvenser. Det vil føre for langt å gjennomføre dette her. La meg bare redegjøre for hva i et regnestykke med primitiv rekursjon svarer til våre utsatte beslutninger i kulespillet.

Vi kan tenke oss at et regnestykke ved primitiv rekursjon foregår ved en lang prosess der vi flere ganger går gjennom løkker i prosessen. Når vi kommer til en løkke vet vi hvor mange ganger vi trenger gå gjennom løkka for å fortsette prosessen videre. Men – og det er det viktige – det er først når vi kommer til løkka at vi vet det. Det er vanlig i programmeringsteori å skille mellom egenskaper som er ”compile time” og de som er bare ”run time”. Noen egenskaper kan en finne ut av før programmet blir kjørt, mens andre blir først klare under kjøring. I våre primitivt rekursive beregninger betrakter vi ”antall ganger i en løkke” som en ”run time” egenskap – og i vår analyse av at beregninger stopper opp med et resultat betrakter vi slike ”run time” egenskaper som resultater av utsatte beslutninger.

I Skolems store 1923-artikkel innfører han ikke bare et programmeringsspråk – de primitivt rekursive funksjonene. Han innfører også en programmeringslogikk. Logikken består av

- Elementær aritmetikk
- Muligheten til ved primitiv rekursjon å definere nye relasjoner
- Bevis ved induksjon over utsagn som er kvantorfrie men kan inneholde frie variable

Skolem sier at han ble irritert ved å lese Russell og Whiteheads ”Principia Mathematicae”. De bruker der mye plass og stor innsats for å vise at enkle regnestykker som 1+1=2 eller 5+7=12 stemmer. I Arne Næss sin filosofihistorie blir dette tatt opp for å vise dybden i deres argumenter. Skolem ser det motsatt. Han viser at de enkle regnestykkene kan begrunnes enkelt og går ganske langt i sin artikkel med å vise hvordan aritmetikk kunne begrunnes. Selve begrunnelsen ved et bevis i Skolems logikk er akkurat så komplisert anskuelsesmessig som vi her har på trinn 3.

Trinn 4 – abstrahere bort posisjoner

Vi kan behandle endelige sekvenser av tall. Posisjonene til tallene blir regnet fra høyre mot venstre – posisjon 0 , posisjon 1, posisjon 2, posisjon 3 og så videre. I argumentene i trinn 3 var det vesentlige

- posisjonene er velordnet
- i bare et endelig antall posisjoner er det tall forskjellig fra 0

Men da kan vi bruke en annen velordning enn den til tallene for å navngi posisjoner. Vi har alt en godt egnet velordning – den til endelige sekvenser av tall. Vi er nå i ytterkant av hva vi kan beskrive uten å bringe inn mer notasjon og mer matematikk. Men la oss prøve. Vi skal beskrive velordninger av endelige sekvenser av tall der posisjonene er velordnet og bare et endelig antall posisjoner har tall forskjellig fra 0. Vi lager et endelig hierarki av slike endelige sekvenser:

- $\mathcal{S}_1$ er velordningen av endelige sekvenser av tall posisjonert med tallene $\mathcal{N}$
- $\mathcal{S}_2$ er velordningen av endelige sekvenser av tall posisjonert med sekvensene $\mathcal{S}_1$

Og så videre

- $\mathcal{S}_{i+1}$ er velordningen av endelige sekvenser av tall posisjonert med sekvensene $\mathcal{S}_i$

Nå er vi ved veis ende. Med $\mathcal{S}_i$ 'ene har vi beskrevet de velordningene som trengs for Gentzens analyse. Det berømte ordinaltall ε_0 er det minste ordinaltall som går utover $\mathcal{S}_i$ 'ene – og dette er beskrevet på en slik måte at vi kan knytte kantianske anskuelser til konstruksjonene.

Diskusjon

Vi startet med konstruksjoner av tall ved:

$$|\quad ||\quad |||\quad ||||\quad |||||\quad ||||||\quad |||||||\ \ldots$$

Dette er hva vi kaller ostensive konstruksjoner – vi kan peke ut og har en nokså direkte anskuelse av tallene. Vi kan også bruke dette til addisjonsstykker som det berømte 5 + 7 = 12

$$||||| + ||||||| = ||||||||||||$$

Tidsanskuelsen går igjen i telleprosessen og dens ubegrensethet er markert med de tre prikkene.

Disse tallene er uegnet til å regne med. Vi trenger symbolske konstruksjoner der vi også kan beskrive tall som er så store at vi ikke kan telle opp til dem. Med posisjonssystemet får vi til det.

Men med de symbolske konstruksjoner trenger vi ekstra argumenter for at våre symboler er notasjoner for tall. Et direkte argument nytter ikke – det har de samme svakheter som en ostensiv konstruksjon. Det direkte argumentet for at noe er et tall vil være større enn selve tallet. Meningen var noe annet – vi ønsker korte argumenter for at store tall er tall. Det oppnår vi ved å bruke hjelpebegrep og hjelpesatser.

De primitivt rekursive funksjonene gir interessante nye symbolske konstruksjoner, men argumentet for at beregningene av dem stopper opp og gir tall som resultat er mer komplisert. Skolem viste hvordan dette kunne gjøres ved kvantorfri induksjon. Vi har her sett på prosesser – som er lik beregningsprosessen for primitiv rekursjon – ved bruk av kuler i bokser. Dette er prosesser der vi har utsatte beslutninger.

Induksjonsprisippet er grunnleggende i aritmetikk. I matematisk notasjon kan vi skrive det som

Fra : $\mathcal{F}0$ og $\forall x\,(\mathcal{F}x \rightarrow \mathcal{F}\,sx)$

Kan vi slutte : $\forall x\,.\,\mathcal{F}x$

Vi begrunner induksjonsprinsippet ved at det reflekterer oppbyggingen av tall. Formelen $\mathcal{F}x$ er lagd i språket for aritmetikk. Vi har et skille mellom kvantorfrie formler og formler der vi tillater kvantorer. Med en kvantorfri formel trenger vi ikke i begrunnelsen for induksjon bruke at vi forstår tallene som et hele – vi viser $\forall x\,.\,\mathcal{F}x$ ved å ta for oss ett og ett tall om gangen. Men med kvantorer i formelen trenger vi å forstå tallene som et hele.

Skolems kvantorfrie induksjon kan begrunnes ut fra prosesser slik vi har dem på trinn 3 – ved bruk av kuler i bokser. Men for høyere aritmetikk trenger en induksjon der kvantorer kan brukes. Det er dette som Gentzen analyserer og hans analyse bruker igjen kuler i bokser men da med boksene i slike generaliserte posisjoner som vi har angitt på trinn 4.

La meg kort skissere Gentzens analyse. Fra Gödels teorem vet vi at det springende punktet er å vise konsistens av aritmetikk med et eller annet argument. Gentzen antar nå at det er et bevis i aritmetikk av en motsigelse som 0=1. Fra det beviset gir han en prosess der vi forenkler beviset ved å ta vekk hjelpesetninger, bruk av induksjon, bruk av kvantorer og ender opp med et bevis av 0=1 uten bruk av hjelpesetninger, induksjon eller kvantorer – noe som er umulig.

At denne prosessen terminerer vises ved å bruke slike ordninger som er gitt i trinn 4. Vi viser for hvert konkret bevis **b**

$$\neg \text{Bevis}(\mathbf{b} , 0=1)$$

men fra Gödels ufullstendighetsteorem kan vi ikke vise

$$\forall x . \neg \text{Bevis}(x , 0=1)$$

Ved å analysere det nærmere ser vi at problemet ligger i beviset av velordning av ordninger som er gitt i trinn 4. Hver konkret ordning kan vi gi et velordningsbevis for i aritmetikk, men vi kan ikke innenfor første ordens aritmetikk gi et slikt for den minste øvre grense til disse ordningene. Disse ordningene gir en analyse av hvilke anskuelser en kan fange inn innenfor første ordens aritmetikk og eksempel på anskuelser som så vidt ikke lar seg fange inn. Vi sier at kompleksiteten til første ordens aritmetikk er gitt ved ordinaltallet ε_0.

La oss se nærmere på de underliggende anskuelser.

I trinn 1 har vi *ostensive* konstruksjoner. Vi trenger da ikke noe ytterligere argument for at konstruksjonene er velfunderte.

I trinn 2 bruker vi symbolske konstruksjoner og da trengs det argumenter for at konstruksjonene er velfunderte. I et posisjonssystem behandler vi notasjoner som forstås slik eksempelet under viser:

$$2345 = 10^3 \text{ x } 2 + 10^2 \text{ x } 1 + 10^1 \text{ x } 4 + 10^0 \text{ x } 5$$

Og så har vi lært hvordan slik notasjoner skal behandles for addisjon og multiplikasjon og annet. Historisk var dette vanskelig å innarbeide i vår kultur – men etter mye streving har det gått. Trinn 2 går egentlig ut over det rene posisjonssystemet. Vi håndterer også notasjoner der eksponentene (dvs de som angir posisjonene) igjen kan skrives i systemet vårt og så videre. Denne generaliseringen av posisjonssystemet likner mye den generaliseringen vi også foretar fra trinn 3 til trinn 4. De argumentene vi bruker for at notasjonene er velfunderte er fra vanlig første ordens logikk der vi tillater bruk av hjelpebegreper og hjelpesetninger.

Trinn 3 behandler uendelighet. Vi innfører vårt kulespill for å anskueliggjøre den formen for uendelighet som brukes. Nøkkelbegrepet er ”endelige prosesser der en kan ha utsatte beslutninger”. Med det kan uendeligheten her behandles som en endelig prosess.

Trinn 4 går videre med kulespillet fra trinn 3. Vi bruker fortsatt bare endelige prosesser. Det er verdt å merke seg parallellen mellom overgangen fra posisjonssystemet til det fulle system i trinn 2 og overgangen fra trinn 3 til trinn 4. I begge tilfeller har vi et hierarki der vi på hvert nivå bruker notasjoner fra et nivå til å lage notasjon for nivået over.

Vi bruker konstruksjonene over for å gi notasjoner for velordninger. La oss til slutt gi ordinaltallene for de ulike nivåene i trinn 4. Vi bruker da

$$\omega_0 = \omega \qquad\qquad \omega_{i+1} = \omega^{(\omega_i)}$$

ω : ordinaltallet knyttet til $\mathcal{N}$ - velordningen av våre naturlige tall

$\omega_1 = \omega^{(\omega)}$: ordinaltallet knyttet til endelige sekvenser av naturlige tall = $\mathcal{S}_1$

ω_i : ordinaltallet knyttet til $\mathcal{S}_i$

Og grensen for disse konstruksjonene er $\sup_i \omega_i = \varepsilon_0$

Litteratur

Til denne artikkelen har jeg brukt som hovedreferanse

- Frode Kjosavik: ”Kants syn på aritmetikk” i festskrift til Johan Arnt Myrstad i anledning 50-års dagen utgitt på Tapir Forlag 1997. Her kunne en gå videre med Kjosaviks og Thor Sandmels doktoravhandliger fra henholdsvis 1999 og 2001 med diskusjoner og referanser fra dem.

Men så er det aller meste tatt som folklore-kunnskap fra det området i logikk som kalles bevisteori. En referanse her er

- Gaisi Takuti: ”Proof theory” North-Holland 1987

men det er en lang vei fra den til det som brukes i denne artikkelen – og på den veien bør en lese de grunnleggende artiklene av Skolem, Gödel og Gentzen. Disse finnes i samlingene

- Jean van Heijenoort: ”From Frege to Gödel” Harvard University Press 1967
- M.E.Szabo: “The collected papers of Gerhard Gentzen” North-Holland 1969