Föreläsning 23 i ADK

Oavgörbarhet

Stefan Nilsson

KTH

Avgörbart och oavgörbart

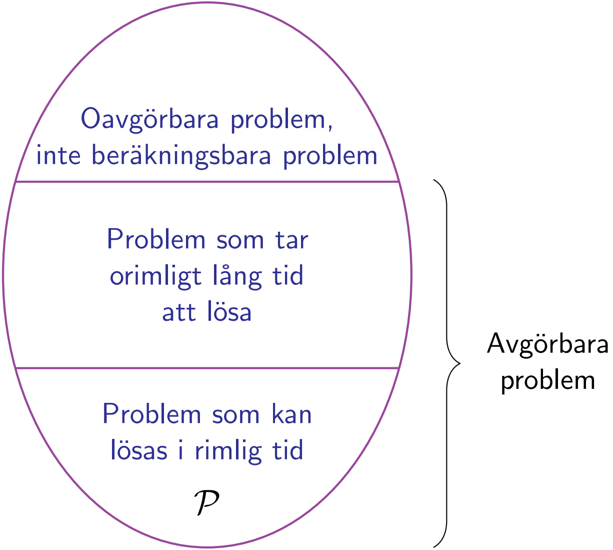
- Ett problem som har en algoritm som för alla instanser kan hitta en lösning i ändlig tid kallas **avgörbart**
- Ett problem som inte kan lösas i ändlig tid av någon algoritm kallas **oavgörbart**

Varje problem som bara har ett ändligt antal probleminstanser är avgörbart

Förutsatt att varje lösning kan skrivas ner i ändlig tid

Avgörbarhet/oavgörbarhet används i första hand om beslutsproblem
Annars talar man om **beräkningsbarhet**

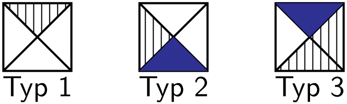
Avgörbart och oavgörbart



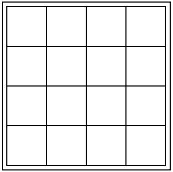
Oavgörbart problem 1: Kakelläggarproblemet

- **Instans:** Ett antal typer av kvadratiska kakelplattor, där en viss kant ska vara nedåt
- **Fråga:** Går det att kakla varje $m \times m$-ruta med kakelplattor av endast de givna typerna så att mönstret stämmer överallt?

Exempel:



4×4:



Oavgörbart problem 2: Ordöverensstämmelse

- **Instans:** Mängd ordpar $\{(x_i, y_i)\}$
- **Fråga:** Finns det någon ändlig talföljd $a_1, a_2, \dots$ så att $x_{a_1} x_{a_2} x_{a_3} \cdots = y_{a_1} y_{a_2} y_{a_3} \dots$

Exempel 1:

- $\{$(abb,bbab), (a,aa), (bab,ab), (bab,ab), (baba,aa), (aba,a)$\}$
- Överensstämmelse fås med 2, 1, 1, 4, 1, 5:

$$\overbrace{\text{a}}^{2}\overbrace{\text{abb}}^{1}\overbrace{\text{abb}}^{1}\overbrace{\text{baba}}^{4}\overbrace{\text{abb}}^{1}\overbrace{\text{aba}}^{5}$$

$$\underbrace{\text{aa}}_{2}\underbrace{\text{bbab}}_{1}\underbrace{\text{bbab}}_{1}\underbrace{\text{aa}}_{4}\underbrace{\text{bbab}}_{1}\underbrace{\text{a}}_{5}$$

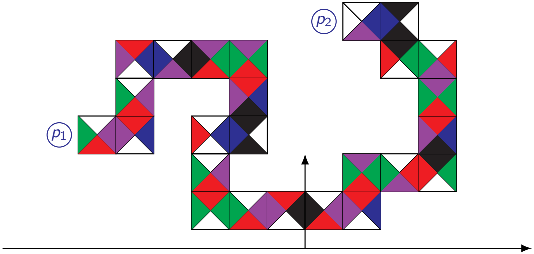
Exempel 2:

- $\{$(bb,bab), (a,aa), (bab,ab), (bab,ab), (baba,aa), (aba,a)$\}$
- Ingen överensstämmelse kan fås!

Oavgörbart problem 3: Orm i kakel

- **Instans:** Mängd typer av kakelplattor, två punkter, p_1 och p_2, i planet
- **Fråga:** Går det att kakla en "orm" som börjar i p_1 och slutar i p_2, där mönstret passar överallt och där ormen håller sig i övre halvplanet?

Exempel:



Avgörbar variant av Orm i kakel:

- Samma problem, men ormen behöver inte hålla sig i övre halvplanet

Fler oavgörbara problem

Oavgörbart problem 4: Stopproblemet

- **Instans:** Program P och inmatning X till programmet
- **Fråga:** Kommer programmet P någonsin att stanna om det startas med indata X?

Oavgörbart problem 5: Programverifiering

- **Instans:** Program P och specifikation S som beskriver vad P ska mata ut för varje indata.
- **Fråga:** Uppfyller programmet specifikationen?
- P uppfyller specifikationen om det för varje möjlig indata stannar och producerar den lösning som anges av S

Bevis av oavgörbarhet

- Reducera ett problem som du vet är oavgörbart till ditt problem
- Om reduktionen i sig är beräkningsbar så är också ditt problem oavgörbart

Exempel: Visa att detta problem är oavgörbart:

- **Instans:** Program P
- **Fråga:** Stannar P på alla indata?

Reduktion från stopproblemet:

```
function STOPP(P, X)
    function Q(Y)
        if X = Y then P(X)
        else stop
    return STOPPFÖRALLA(Q)
```

Stopp på blankt

- STOPP-PÅ-BLANKT(P) är problemet: “Stannar P någonsin om det startas med tomt indata?”
- Visa att STOPP-PÅ-BLANKT är oavgörbart genom att reducera vanliga stopproblemet!

Bevis av stopproblemets oavgörbarhet

- Anta att $\textsc{Stopp}(P, X)$ löser stopproblemet, dvs. att stopproblemet är avgörbart, och visa att detta leder till en motsägelse
- Konstruera programmet $\textsc{Meta}(P)$ på följande sätt:

  function $\textsc{Meta}(P)$
      if $\textsc{Stopp}(P, P)$ **then**
          while True **do** nothing
      else return

- Vad händer vid anropet $\textsc{Meta}(\textsc{Meta})$?
  1. $\textsc{Meta}(\textsc{Meta})$ går i oändlig slinga; då är $\textsc{Stopp}(\textsc{Meta}, \textsc{Meta})$ falskt och programmet stannar. Orimligt!
  2. $\textsc{Meta}(\textsc{Meta})$ stannar så småningom; då är $\textsc{Stopp}(\textsc{Meta}, \textsc{Meta})$ sant och programmet går in oändlig slinga. Orimligt!

Gemensam egenskap: Ändlig verifikation

En lösning till ett $\mathcal{NP}$-fullständigt problem kan verifieras i polynomisk tid.

På motsvarande sätt:

- En lösning till något av de beskrivna oavgörbara problemen kan verifieras i **ändlig** tid.

Exempel: Stopproblemet

Om P stannar på indata X så kan beräkningen verifieras i ändlig tid

Exempel: Ordöverensstämmelse

Om det går att få överensstämmelse med en viss talföljd så är det lätt att kolla att $x_{a_1}x_{a_2} = y_{a_1}y_{a_2}$

Exempel: Kakelläggning

Om det finns någon $m \times m$-ruta som inte kan kaklas så kan det (ganska) lätt verifieras: kolla alla tänkbara kaklingar av denna ruta.

Dubbel ändlig verifikation

- För kakelläggning kan <u>nej</u>-lösningar verifieras
- För ordöverensstämmelse kan <u>ja</u>-lösningar verifieras

Om man för ett problem både kan verifiera ja-lösningar och nej-lösningar så måste problemet vara avgörbart

Bevis:

- Gå igenom alla tänkbara lösningar i ordning (börja med alla lösningar av längd 1, sedan längd 2, osv.)
- Testa för var och en om den är en ja-lösning eller nej-lösning
- Avbryt så fort en verifiering lyckas
- Denna procedur är ändlig eftersom man förr eller senare testar en riktig lösning

Rekursiv och rekursivt uppräknelig

- **Rekursiv funktion** = beräkningsbar funktion = funktion som stannar för alla indata
- **Rekursiv mängd** = Språk som kan kännas igen av någon rekursiv funktion
- **Rekursivt uppräknelig funktion** = funktion med ändlig verifikation av ja-lösningar
- **Rekursivt uppräknelig mängd** = Språk som kan kännas igen av någon rekursivt uppräknelig funktion
- **R.E.-fullständiga problem** = De svåraste problemen som kan lösas med rekursivt uppräkneliga funktioner

Stopproblemet är R.E.-fullständigt