Föreläsning

Reduktioner

Kan användas till att

- visa positiva resultat (hitta algoritmer)
- visa negativa resultat (undre gränser)
- visa att ett problem är svårare än ett annat
- visa att två problem är lika svåra

Olika varianter av problem

Olika varianter av reduktioner

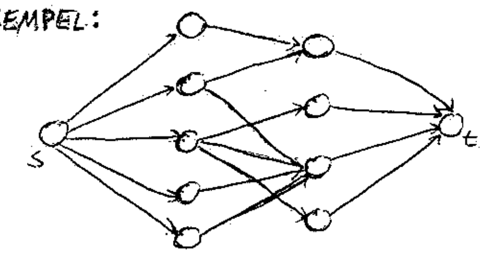
Bipartit matchning

Indata: bipartit graf $\langle U \cup V, E \rangle$

Utdata: en matchning $M \subseteq E$ av maximal storlek

M är en matchning om inga kanter i M har någon gemensam ändpunkt.

Exempel:



Reduktion av bipartit matchning till flöde:

Bipartite-Matching$(U, V, E) =$

- Konstruera graf $\langle V', E' \rangle$ som
  $V' = U \cup V \cup \{s, t\}$
  $E' = \{(u,v): u \in U, v \in V, (u,v) \in E\} \cup$
  $\cup \{(s,u): u \in U\} \cup$
  $\cup \{(v,t): v \in V\}$
  $c(e) = 1 \ \forall e \in E'$
- return (Ford-Fulkerson$(V', E') \cap E$)

Representantproblemet

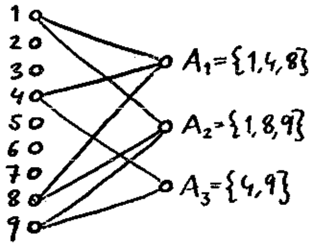
Indata: k stycken mängder $A_1, \ldots, A_k$ av heltal mellan 1 och k^2

Utdata: en mängd med k olika tal $a_1, \ldots, a_k$ så att $a_i \in A_i$ för $1 \le i \le k$

Algoritm: **Reducera problemet till bipartit matchning!**

```
Representant(A_1,...,A_k) =
  U ← {1,...,k²}
  V ← {A_1,...,A_k}
  E ← {(x,A): x∈U, A∈V, x∈A}
  M ← BipartitMatchning(U,V,E)
  return {x: (x,A)∈M för något A}
```

Exempel: $k=3$, $A_1=\{1,4,8\}$, $A_2=\{1,8,9\}$, $A_3=\{4,9\}$



GENERELL VIKTAD MATCHNING

DET FINNS EN POLYNOMISK ALGORITM SOM LÖSER MAXIMAL MAXMATCHNING:

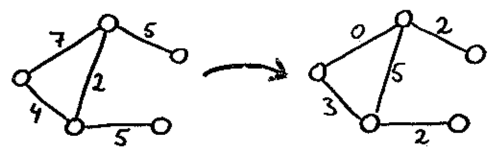
INDATA: GRAF $G=(V,E)$, KANTVIKTER $c: E \to \mathbb{N}$

UTDATA: MATCHNING $M \subseteq E$ SOM DELS INNEHÅLLER SÅ MÅNGA KANTER SOM MÖJLIGT OCH DELS HAR SÅ STOR SAMMANLAGD VIKT SOM MÖJLIGT.

BESKRIV EN ALGORITM FÖR PROBLEMET MINIMAL MAXMATCHNING DÄR SAMMANLAGDA VIKTEN AV KANTERNA I MAXMATCHNINGEN ÄR MINIMAL!

REDUCERA MINIMAL MAXMATCHNING TILL MAXIMAL MAXMATCHNING

```
MinimalMaxmatchning(G'=(V',E'), f: E'→ℕ) =
   w ← max f(e)
       e∈E'
   DEFINIERA c(e) SOM w−f(e)
   M ← MaximalMaxmatchning(G', c)
   RETURN M
```



VISA NEGATIVA RESULTAT MED REDUKTION

```
Sortering(A[1..n]) =                Inorder(T) =
   T ← ByggSökträd(A)                 IF T≠NIL THEN
   INDEX ← 1                              Inorder(T.left)
   Inorder(T)                             A[index] ← T.value
   RETURN A                               index ← index+1
                                          Inorder(T.right)
```

VI HAR REDUCERAT SORTERING TILL SÖKTRÄDSBYGGE.

TIDEN FÖR SORTERING ÄR TIDEN FÖR BYGGSÖKTRÄD PLUS TIDEN FÖR INORDER (SOM ÄR $O(n)$).

OM BYGGSÖKTRÄD GICK SNABBARE ÄN $\Theta(n \log n)$ SKULLE DET ALLTSÅ GÅ ATT SORTERA SNABBARE ÄN $\Theta(n \log n)$.

REDUKTIONEN HAR FÖRT ÖVER DEN UNDRE GRÄNSEN PÅ $\Omega(n \log n)$ JÄMFÖRELSER FÖR SORTERING TILL SÖKTRÄDSBYGGE.

> OM P KAN REDUCERAS TILL Q
> OCH P ÄR SVÅRT
> SÅ ÄR Q OCKSÅ SVÅRT!

MAX KLICK

INDATA: GRAF G

UTDATA: ANTAL HÖRN I DEN STÖRSTA KLICKEN (FULLSTÄNDIGA DELGRAFEN) I G

MAX OBEROENDE MÄNGD

INDATA: GRAF G

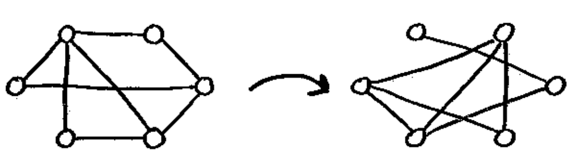
UTDATA: ANTAL HÖRN I DEN STÖRSTA OBEROENDE MÄNGDEN (HÖRN UTAN KANTER MELLAN) I G

REDUCERA MAX KLICK TILL MAX OBEROENDE MÄNGD!

```
MaxKlick(G) =
   Ḡ ← KOMPLEMENTGRAFEN TILL G
   RETURN MaxOberMängd(Ḡ)
```

REDUCERA MAX OBEROENDE MÄNGD TILL MAX KLICK!

```
MaxOberoendeMängd(G) =
   Ḡ ← KOMPLEMENTGRAFEN TILL G
   RETURN MaxKlick(Ḡ)
```



G $\qquad\qquad$ $\overline{G}$

MAX KLICK OCH MAX OBEROENDE MÄNGD ÄR LIKA SVÅRA!

OLIKA VARIANTER AV PROBLEM

1. **BESLUTSPROBLEM** (UTDATA ÄR SANT/FALSKT)
   INDATA: GRAF G, HELTAL K
   FRÅGA: FINNS DET EN KLICK AV STORLEK ≥K I G?

2. **OPTIMERINGSPROBLEM** (MAXIMERA EN MÅLFUNKTION)
   INDATA: GRAF G
   UTDATA: STORLEKEN AV STÖRSTA KLICKEN I G

3. **KONSTRUKTIONSPROBLEM**
   INDATA: GRAF G
   UTDATA: HÖRNEN I EN KLICK AV MAXSTORLEK I G

REDUKTIONER MELLAN PROBLEMVARIANTER

REDUCERA OPTIMERINGSPROBLEM TILL BESLUTSPROBLEM:

```
MaxKlick(G=(V,E)) =
  i←1; j←|V|
  WHILE i<j DO
    m← ⌈(i+j)/2⌉
    IF BeslutKlick(G,m) THEN i←m
      ELSE j←m-1
  RETURN i
```

REDUCERA KONSTRUKTIONSPROBLEM TILL OPTIMERINGSPROBLEM:

```
KonstruktivMaxKlick(G=(V,E)) =
  m←MaxKlick(G)
  S←V
  FOR EACH u∈V DO
    IF MaxKlick(S-{u})=m THEN
      S← S-{u}
  RETURN S
```

TURING- OCH KARPREDUKTIONER

EN TURINGREDUKTION AV PROBLEMET P TILL PROBLEMET Q ÄR EN ALGORITM FÖR P SOM ANROPAR EN ALGORITM FÖR Q EN ELLER FLERA GÅNGER.

```
P(x)=
  ⋮
  (HÄR ANROPAS ALGORITM FÖR Q)
  ⋮
  RETURN y
```

EN KARPREDUKTION AV P TILL Q ÄR DET SPECIALFALL AV TURINGREDUKTION DÄR P OCH Q ÄR BESLUTSPROBLEM, DÄR ALGORITMEN FÖR Q BARA ANROPAS EN GÅNG, OCH DÄR RESULTATET FRÅN DEN ALGORITMEN DIREKT RETURNERAS

```
P(x)=
  KONSTRUERA EN PROBLEM-
  INSTANS y TILL Q
  RETURN Q(y)
```