UiT Norges arktiske universitet

Næringsøkonomi og konkurransestrategi

Oppsummering

Anita Michalsen

Hovedtemaer

1. Introduksjon
   - Grunnleggende mikroøkonomi
2. Monopol
   - Prisdiskriminering, produktvalg og kvalitet
3. Oligopolmodeller
   - Basismodeller for pris- og kvantumskonkurranse; Cournot, Bertrand og Stackelberg
4. Konkurranseskadelige strategier
   - Prissamarbeid og kartell
5. Relasjoner mellom bedrifter
   - Fusjoner og oppkjøp
   - Vertikale relasjoner

Bertrand-modell

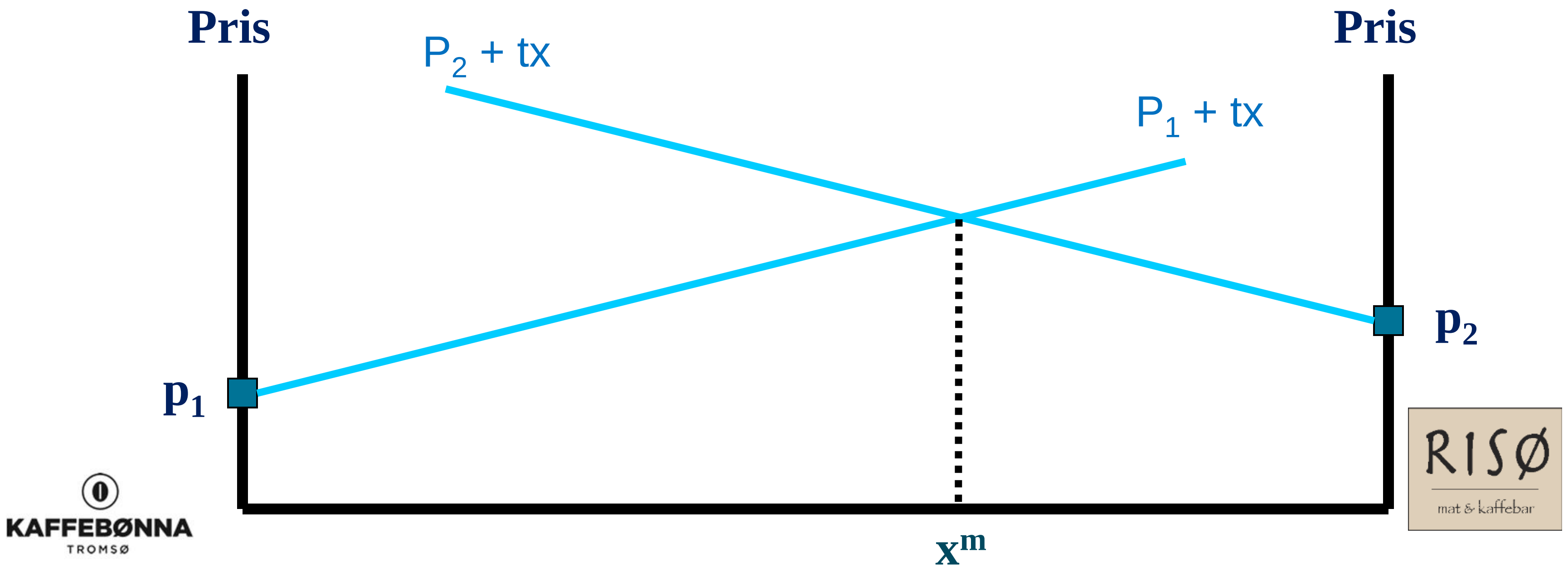
Pris er bedriftens handlingsvariabel, og bedriftene velger pris simultant

Under forutsetning om at hver bedrift alene kan betjene hele markedet får vi følgende profitt:

$$\pi_i = \begin{cases} (p_i - c)D(p_i) & hvis\ \ p_i < p_j \\ \dfrac{(p_i - c)D(p_i)}{2} & hvis\ \ p_i = p_j \\ 0 & hvis\ \ p_i > p_j \end{cases}$$

Nash-likevekt: $p_i^* = p_j^* = c$ => Bertrand paradoks

Bertrand-konkurranse og lokaliseringsbasert differensiering - Hotelling modell med 2 bedrifter



Kunden er indifferent når: $P_1 + tx = P_2 + t(1\text{-}x)$

Etterspørsel $x^m = \frac{P_2 - P_1 + t}{2t}$

Bertrand konkurranse og differensiering

```
from sympy import *
import sympy as sp
from matplotlib import pyplot as plt
import numpy as np
```

```
P1, P2,c, t,N=symbols('P1 P2 c t N')
```

Etterspørsel bedrift 1:

```
def demand1(P1,t,P2,N):
    return (N*(P2-P1+t))/(2*t)
demand1(P1,t,P2,N)
```

$$\frac{N\left(-P_1+P_2+t\right)}{2t}$$

Etterspørsel bedrift 2:

```
def demand2(P1,t,P2,N):
    return (N*(P1-P2+t))/(2*t)
demand2(P1,t,P2,N)
```

$$\frac{N\left(P_1-P_2+t\right)}{2t}$$

Profitt for bedrift 1:

```
def profit1(P1,t,P2,c,N):
    return ((P1-c)*demand1(P1,t,P2,N))
profit1(P1,t,P2,c,N)
```

$$\frac{N\left(P_1-c\right)\left(-P_1+P_2+t\right)}{2t}$$

Profitt for bedrift 2:

```
def profit2(P1,t,P2,c,N):
    return ((P2-c)*demand2(P1,t,P2,N))
profit2(P1,t,P2,c,N)
```

$$\frac{N\left(P_2-c\right)\left(P_1-P_2+t\right)}{2t}$$

For å finne optimal løsning så deriverer vi profitt funksjonene mhp P

```
# den deriverte av profittfunksjonen mhp P1 og P2
d_profit1=diff(profit1(P1,t,P2,c,N),P1)
d_profit2=diff(profit2(P1,t,P2,c,N),P2)

display(d_profit1)
display(d_profit2)
```

$$-\frac{N(P_1-c)}{2t}+\frac{N(-P_1+P_2+t)}{2t}$$

$$-\frac{N(P_2-c)}{2t}+\frac{N(P_1-P_2+t)}{2t}$$

Finner reaksjonsfunksjon til bedrift 1 ved å sette den deriverte lik 0 og løse for P1

```
#reaksjonsfunksjon til bedrift 1
P1_equ=sp.solve(d_profit1,P1)[0]
P1_equ
```

$$\frac{P_2}{2}+\frac{c}{2}+\frac{t}{2}$$

Tilsvarende for reaksjonsfunksjon til bedrift 2:

```
#reaksjonsfunksjon til bedrift 2
P2_equ=sp.solve(d_profit2,P2)[0]
P2_equ
```

$$\frac{P_1}{2}+\frac{c}{2}+\frac{t}{2}$$

Setter RF2 inn i RF1 og finner optimal pris

```
# Optimal pris for bedrift 1 og 2
sol=solve([d_profit1,d_profit2],[P1,P2])

display(sol[P1])
display(sol[P2])
```

$$c+t$$

$$c+t$$

Bertrand-konkurranse og reaksjonsfunksjon

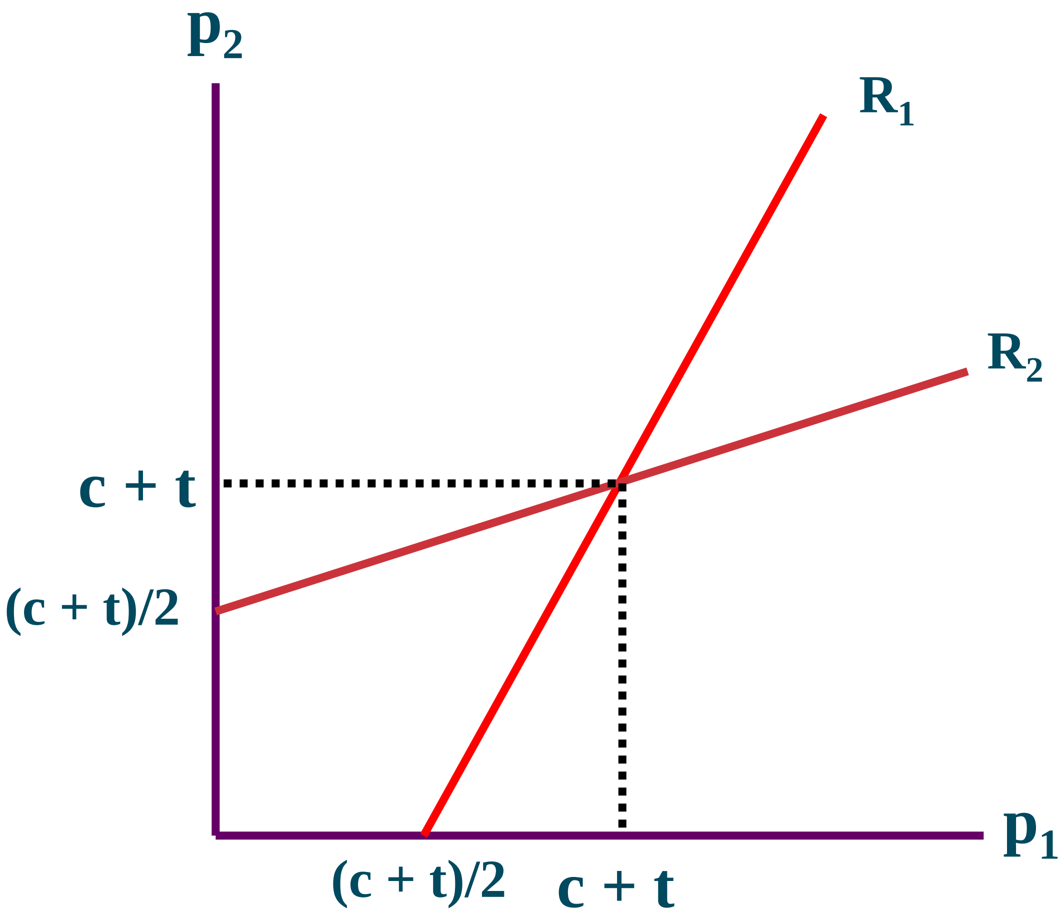
$$max\ \pi_1 = (P_1 - c)(\frac{P_2 - P_1 + t}{2t})$$

$$max\ \pi_2 = (P_2 - c)(\frac{P_1 - P_2 + t}{2t})$$

Reaksjonsfunksjon: $P_1 = \frac{P_2 + t + c}{2}$

Reaksjonsfunksjon: $P_2 = \frac{P_1 + t + c}{2}$

Optimal tilpasning : $P_1 = P_2 = t + c$



Cournot modell med symmetriske bedrifter

Vi har en etterspørsel som er lik P = A - BQ, hvor Q = q1 + q2. Marginalkostnader er lik c

```
import sympy as sp
from sympy import *
from matplotlib import pyplot as plt
import numpy as np
```

```
q1, q2,c, A, B=symbols('q1 q2 c A B')
```

```
def demand_1(q1):
        return (A-B*q1-B*q2)
```

```
def demand_2(q2):
        return (A-B*q1-B*q2)
```

```
def marginalrevenue_1(q1):
    return (A-2*B*q1-B*q2)
```

```
def marginalrevenue_2(q2):
    return (A-B*q1-2*B*q2)
```

Optimal tilpasning der MR = MC

```
q1=sp.symbols('q1', real=True, positive=True)
equ1=sp.Eq(marginalrevenue_1(q1),c)
equ1
```

$$A - 2Bq_1 - Bq_2 = c$$

```
#reaksjonsfunksjon til bedrift 1
q1_equ=sp.solve(equ1,q1)[0]
q1_equ
```

$$\frac{A - Bq_2 - c}{2B}$$

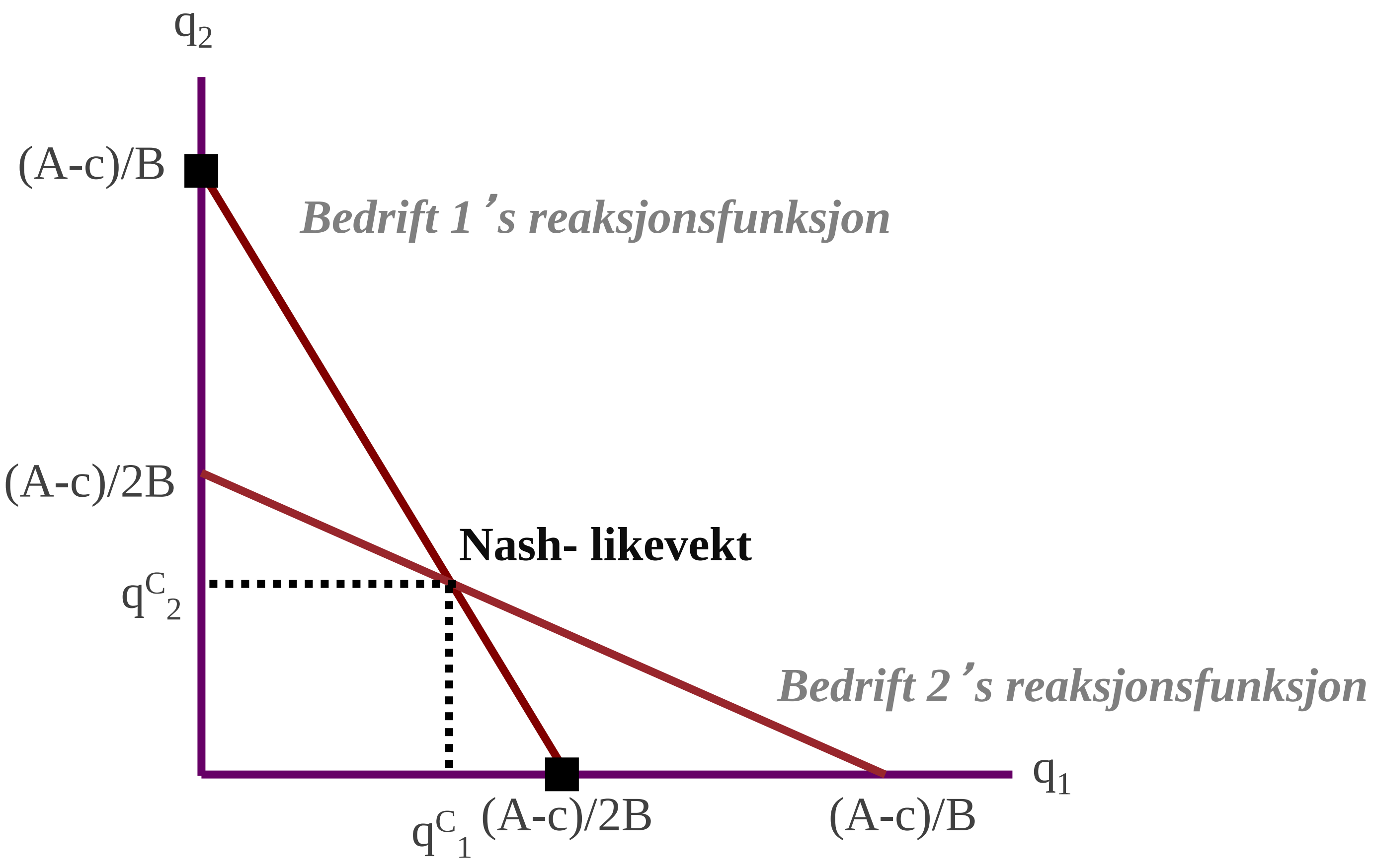
```
q2=sp.symbols('q2', real=True, positive=True)
equ2=sp.Eq(marginalrevenue_2(q2),c)
equ2
```

$$A - Bq_1 - 2Bq_2 = c$$

```
# reaksjonsfunksjon til bedrift 2
q2_equ=sp.solve(equ2,q2)[0]
q2_equ
```

$$\frac{A - Bq_1 - c}{2B}$$

Cournot modell



Tilpasning der MR = MC:

$A - 2Bq_1 - Bq_2 = c$

$A - Bq_1 - 2Bq_2 = c$

Reaksjonsfunksjon til bedrift 1 er

$$q^*_1 = = \frac{A-c}{2B} - \frac{q_2}{2}$$

Reaksjonsfunksjon til bedrift 2 er

$$q^*_2 = = \frac{A-c}{2B} - \frac{q_1}{2}$$

For å finne optimalt kvantum til bedrift 1 setter vi reaksjonsfunksjon til bedrift 2 inn i reaksjonsfunksjon til bedrift 1

```
# finner uttrykk for q1 og q2 der MR = MC
q1_eq = sp.solve(sp.Eq(marginalrevenue_1(q1), c), q1)[0]
q2_eq = sp.solve(sp.Eq(marginalrevenue_2(q2), c), q2)[0]
```

```
# setter uttrykk for q2 inn i q1
q1_unresv_eq = q1_eq.subs(q2, q2_eq)
```

```
# Løser for q1, og får optimalt kvantum for bedrift 1:
sp.solve(sp.Eq(q1_unresv_eq, q1), q1)[0]
```

$$\frac{A-c}{3B}$$

```
# setter uttrykk for q1 inn i q2
q2_unresv_eq = q2_eq.subs(q1, q1_eq)
```

```
# Løser for q2, og får optimalt kvantum for bedrift 1:
sp.solve(sp.Eq(q2_unresv_eq, q2), q2)[0]
```

$$\frac{A-c}{3B}$$

Cournot modell med asymmetriske bedrifter

Invers etterspørsel: P = A - BQ, marginalkostnad til bedrift 1 er lik c1 og bedrift 2 har marginalkostnader på c2

```
import sympy as sp
from matplotlib import pyplot as plt
import numpy as np
```

```
q1, q2,c1,c2, A, B=symbols('q1 q2 c1 c2 A B')
```

```
def demand_1(q1):
        return (A-B*q1-B*q2)
```

```
def marginalrevenue_1(q1):
    return (A-2*B*q1-B*q2)
```

```
def demand_2(q2):
        return (A-B*q1-B*q2)
```

```
def marginalrevenue_2(q2):
    return (A-B*q1-2*B*q2)
```

```
# Marginalinntekt for q1 og q2
mr_q1_eq = A - 2*B*q1 - B*q2
mr_q2_eq = A - B*q1 - 2*B*q2
```

Optimal tilpasning der MR = MC

```
q1=sp.symbols('q1', real=True, positive=True)
equ1=sp.Eq(marginalrevenue_1(q1),c1)
equ1
```

$$A - 2Bq_1 - Bq_2 = c_1$$

```
q2=sp.symbols('q2', real=True, positive=True)
equ2=sp.Eq(marginalrevenue_2(q2),c2)
equ2
```

$$A - Bq_1 - 2Bq_2 = c_2$$

```
#reaksjonsfunksjon til bedrift 1
q1_equ1=sp.solve(equ1,q1)[0]
q1_equ1
```

$$\frac{A - Bq_2 - c_1}{2B}$$

```
#reaksjonsfunksjon til bedrift 2
q2_equ2=sp.solve(equ2,q2)[0]
q2_equ2
```

$$\frac{A - Bq_1 - c_2}{2B}$$

Optimalt kvantum til bedrift 2 finner vi ved å sette RF1 inn i RF2:

```
# finner uttrykk for q1 og q2 der MR = MC
q1_eq = sp.solve(sp.Eq(marginalrevenue_1(q1), c1), q1)[0]
q2_eq = sp.solve(sp.Eq(marginalrevenue_2(q2), c2), q2)[0]
```

```
# setter uttrykk for q1 inn i q2
q2_unresv_eq = q2_eq.subs(q1, q1_eq)
```

```
# setter uttrykk for q2 inn i q1
q1_unresv_eq = q1_eq.subs(q2, q2_eq)
```

Finner optimalt kvantum for bedrift 2:

```
# Løser q2
sp.solve(sp.Eq(q2_unresv_eq, q2), q2)[0]
```

$$\frac{A + c_1 - 2c_2}{3B}$$

Tilsvare for bedrift 1, optimalt kvantum blir:

```
# Løser q1
sp.solve(sp.Eq(q1_unresv_eq, q1), q1)[0]
```

$$\frac{A - 2c_1 + c_2}{3B}$$

Cournot modell

- Kvantum er bedriftens handlingsvariabel og velges simultant av bedriftene

- Nash-likevekt:
  - Asymmetrisk Cournot $(c_i \neq c_j)$: $q_i = \frac{A-2c_i+c_j}{3B}$ $P^C = \frac{A+c_i+c_j}{3}$
  - Symmetrisk Cournot $(c_i = c_j)$: $q_i = \frac{A-c}{3B}$ $P^C = \frac{A+2c}{3}$
  - n – bedrifter $(n \geq 3)$: $q_i = \frac{A-c}{B(n+1)}$ $P^C = \frac{A+nc}{n+1}$

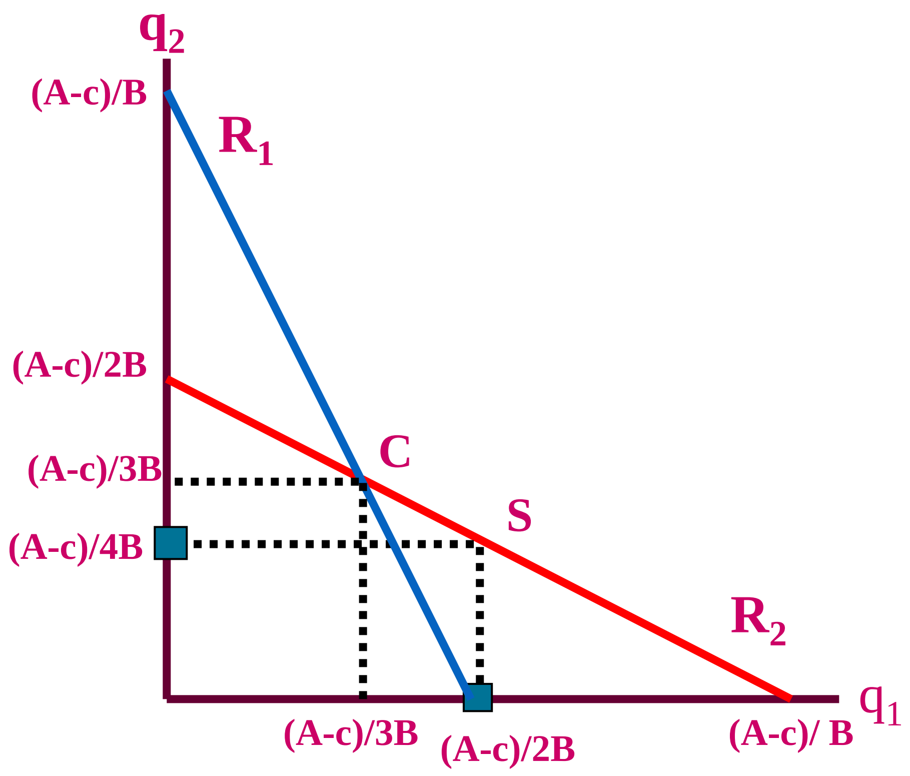
Stackelberg likevekt

Optimal kvantum og pris:

$$q_1^s = \frac{A - c}{2B}$$

$$q_2^s = \frac{A - c}{4B}$$

$$P^s = \frac{A + 3c}{4}$$



Stackelberg

Ved Stackelberg modell er kvantum bedriftens handlingsvariable og bedriftene gjør sine valg sekvensielt. Lederbedriften velger først hvor mye den vil produser og deretter velger følgerbedriften sitt volum. Prisen er gitt ved etterspørselsfunksjonen.

Vi antar at vi har to bedrifter som konkurrer med kvantum som strategisk variabel. Invers ettersprøsel er gitt ved: P(Q) = a-bQ = a-bq1 -bq2. Vi antar at bedriften har identiske marginalkostnader lik c.

Kode her er basert på notebook fra seminar sok-2030 vår 2022 (Espen Sirnes).

```
import sympy as sp
from sympy import *
from matplotlib import pyplot as plt
import numpy as np
```

```
q1, q2,c, a, b, pi,i=symbols('q1 q2 c a b pi i')

def P_demand(Q,a,b):
    return a-b*Q

def profit(q1,q2,c,a,b):
    return (P_demand(q1+q2,a,b)-c)*q1
```

Modellen løses i to trinn. På trinn 1 velger lederbedriften sitt kvantum og på trinn 2 velger følgerbedriften sitt optimale kvantum. Modellen løses ved baklengs induksjon, dvs av vi starter på trinn 2

Vi dereiverer profittfunksjon til bedrift 2 mhp q2: π2 = (P-c)*q2=(a-b(q1+q2)-c)*q2

```
d_profit2_Q=diff(profit(q2,q1,c,a,b),q2)
d_profit2_Q
```

$$a - bq_2 - b(q_1 + q_2) - c$$

Setter den derivert lik 0 og finner reaksjonsfunksjon til bedrift 2

```
Q2_sol1=solve(d_profit2_Q,q2)[0]
Q2_sol1
```

$$\frac{a - bq_1 - c}{2b}$$

På trinn 1 sette vi reaksjonsfunksjonene til bedrift 2 inn i bedrift 1 sin profittfunksjon, og deriverer dette utrykket mhp q1. Maks π1 =(a-b(q1+q2)-c)*q1 gitt q2=(a-bq1-c)/2b

```
d_profit1_Q=diff(profit(q1,Q2_sol1,c,a,b),q1)
d_profit1_Q
```

$$a - \frac{bq_1}{2} - b\left(q_1 + \frac{a - bq_1 - c}{2b}\right) - c$$

For å finne optimalt kvantum til lederbedriften setter vi uttrykket over lik 0

```
Q1_sol=solve(d_profit1_Q,q1)[0]
Q1_sol
```

$$\frac{a - c}{2b}$$

Vi setter så optimalt valg av q1 inn i reaksjonsfunksjonen til bedrift 2

```
Q2_sol2=Q2_sol1.subs({q1:Q1_sol})
Q2_sol2
```

$$\frac{\frac{a}{2} - \frac{c}{2}}{2b}$$

```
def P_demand(q1,q2):
    return a-b*(q1+q2)
```

```
# Optimal kvantum i sluttmarkedet:
P_opt=P_demand(q1,q2).subs({q1:Q1_sol,q2:Q2_sol2})
sp.simplify(P_opt)
```

$$\frac{a}{4} + \frac{3c}{4}$$

```
# profitt for lederbedrift:
def profitt(q1):
    return (P_opt-c)*Q1_sol

sp.simplify(profitt(q1))
```

$$\frac{(-a + c)^2}{8b}$$

```
# profitt for følgerbedrift:
def profitt(q2):
    return (P_opt-c)*Q2_sol2

sp.simplify(profitt(q2))
```

$$\frac{(-a + c)^2}{16b}$$

```
stackelberg=lambdify(
    (a,b,c),
    (Q1_sol,Q2_sol2)
)
```

```
stackelberg(200,1,60)
```

```
(70.0, 35.0)
```

Bertrand-konkurranse, differensiering og sekvensiell konkurranse - Python 11.2 b)

Sekvensiell priskonkurranse

Ved sekvensiell priskonkurranse er pris bedriftens handlingsvariable og bedriftene gjør sine valg sekvensielt. Lederbedriften velger først sin pris og deretter velger følgerbedriften sin pris.

Vi har to bedrifter som konkurrer med pris som strategisk variabel. Vi antar at bedriften har identiske marginalkostnader lik c.

Etterspørselsfunksjonen for bedrift 1 er gitt ved: D1(P1,P2) = (Xm((P1,P2)) = [(P2-P1+t)/2t]N

Etterspørselsfunksjonen for bedrift 2 er gitt ved: D2(P1,P2) = (1-Xm(P1,P2) = [(P1-P2+t)/2t]N

```
import sympy as sp
from sympy import *
from matplotlib import pyplot as plt
import numpy as np
```

Modellen løses i to trinn. På trinn 1 velger lederbedriften sin optimal pris og på trinn 2 velger følgerbedriften sin pris. Modellen løses ved baklengs induksjon, dvs av vi starter på trinn 2

```
P1, P2,c, N, t=symbols('P1 P2 c N t')

def demand_2(P2,P1,t,N):
    return (P1-P2+t)/(2*t)*N

def profit2(P2,P1,t,N):
    return (P2-c)*demand_2(P2,P1,t,N)
```

Vi dereiverer profittfunksjon til bedrift 2 mhp P2

```
d_profit2=diff(profit2(P2,P1,t,N),P2)
d_profit2
```

$$-\frac{N\left(P_2-c\right)}{2t}+\frac{N\left(P_1-P_2+t\right)}{2t}$$

Setter den derivert lik 0 og finner reaksjonsfunkjson til bedrifte 2

```
P2_sol1=solve(d_profit2,P2)[0]
P2_sol1
```

$$\frac{P_1}{2}+\frac{c}{2}+\frac{t}{2}$$

```
def demand_1(P2,P1,t,N):
    return (P2-P1+t)/(2*t)*N

def profit1(P2,P1,t,N):
    return (P1-c)*demand_1(P2,P1,t,N)
```

På trinn 1 sette vi reaksjonsfunksjonene til bedrift 2 inn i bedrift 1 sin profittfunksjon, og deriverer dette utrykket mhp P1.

```
d_profit1=diff(profit1(P2_sol1,P1,t,N),P1)
d_profit1
```

$$-\frac{N\left(P_1-c\right)}{4t}+\frac{N\left(-\frac{P_1}{2}+\frac{c}{2}+\frac{3t}{2}\right)}{2t}$$

For å finne optimalt pris til lederbedriften setter vi uttrykket over lik 0

```
P1_sol1=solve(d_profit1,P1)[0]
P1_sol1
```

$$c + \frac{3t}{2}$$

Vi setter så optimalt valg av P1 inn i reaksjonsfunksjonen til bedrift 2 og finner prisen til bedrift 2

```
P2_sol2=P2_sol1.subs({P1:P1_sol1})
P2_sol2
```

$$c + \frac{5t}{4}$$

Til pris lik c + 5t/4 vil bedrift 2 selge følgende kvantum:

```
demand_2(P2,P1,t,N).subs({P1:P1_sol1,P2:P2_sol2})
```

$$\frac{5N}{8}$$

og til pris lik c + 3t/2 vil bedrift 1 selge følgende kvantum:

```
demand_1(P2,P1,t,N).subs({P1:P1_sol1,P2:P2_sol2})
```

$$\frac{3N}{8}$$

Profitten til bedrift 2 blir:

```
profit2(P2,P1,t,N).subs({P1:P1_sol1,P2:P2_sol2})
```

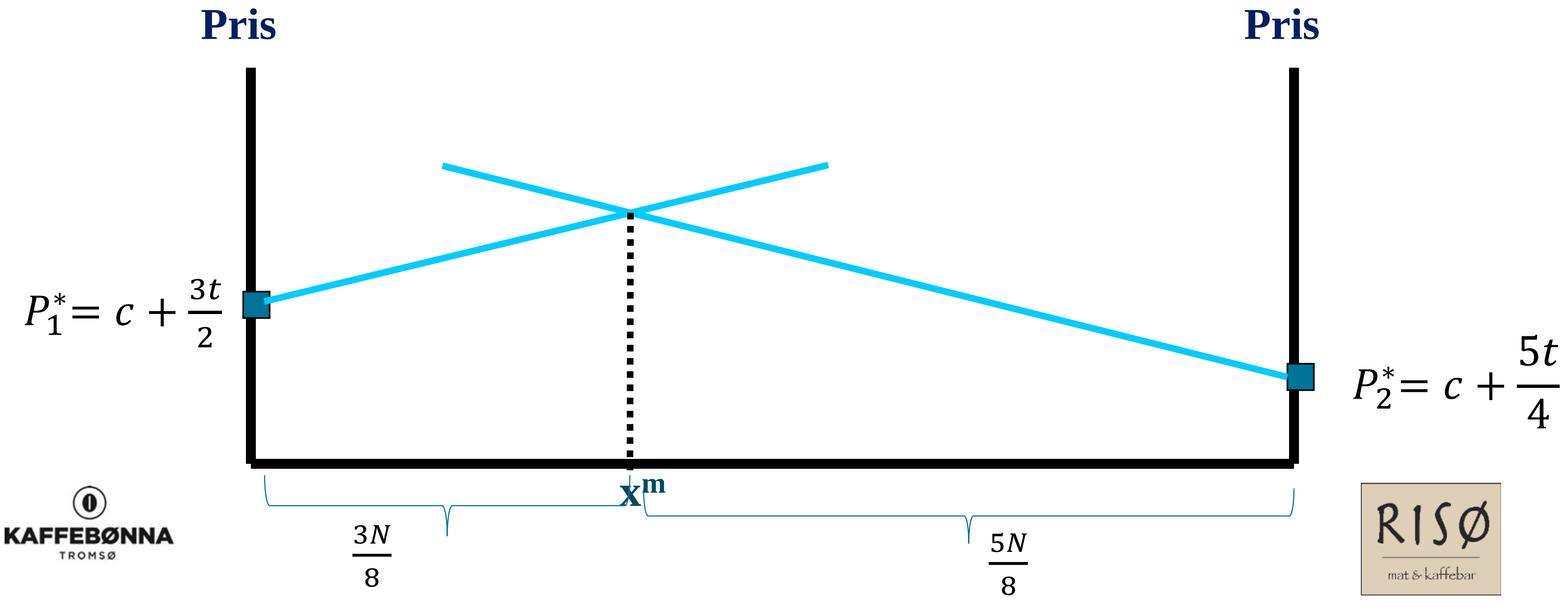
$$\frac{25Nt}{32}$$

og bedrift 1 får følgende profit:

```
profit1(P2,P1,t,N).subs({P1:P1_sol1,P2:P2_sol2})
```

$$\frac{9Nt}{16}$$

Bertrand-konkurranse, differensiering og sekvensiell konkurranse



$$\pi_1 = \frac{9NT}{16} \quad og \quad \pi_2 = \frac{25NT}{32}$$

Lederbedriften vil sett prisen høyere enn følgerbedriften, og vil da selge lavere kvantum enn bedrift 2

Del 4 Prissamarbeid

Cournot model og "fangens dilemma"

For å se om prissamarbeid vil lønnne seg må vi sammenligne ulike markedstilpasninger. Vi vil her se på optimal tilpasning ved Cournot konkurranse.

```
import sympy as sp
import numpy as np
from matplotlib import pyplot as plt
```

Anta følgende etterspørsel: P = 150 – Q og marginalkostnader: c = 30

```
def demand_1(q1):
        return (150-q1-q2)
```

```
def demand_2(q2):
        return (150-q1-q2)
```

```
def marginalrevenue_1(q1):
    return (150-2*q1-q2)
```

```
def marginalrevenue_2(q2):
    return (150-2*q2-q1)
```

Optimal tilpasning ved Cournot modell

```
# optimal tilpasning der MR = MC
q2=sp.symbols('q2', real=True, positive=True)
q1=sp.symbols('q1', real=True, positive=True)
equ=sp.Eq(marginalrevenue_2(q2),30)
equ
```

$$-q_1 - 2q_2 + 150 = 30$$

```
#reaksjonsfunksjon til bedrift 2
q2_equ=sp.solve(equ,q2)[0]
q2_equ
```

$$60 - \frac{q_1}{2}$$

```
equ=sp.Eq(marginalrevenue_1(q1),30)
equ
```

$$-2q_1 - q_2 + 150 = 30$$

```
#reaksjonsfunksjon til bedrift 1
q1_equ=sp.solve(equ,q1)[0]
q1_equ
```

$$60 - \frac{q_2}{2}$$

```
# setter reaksjonsfunskjon til bedrift 2 inn i reaksjonsfunksjon til bedrift 1
q1_unresv_eq = q1_equ.subs(q2, q2_equ)
```

```
# Løser for å finne optimalt kvantum for bedrift 1
q1_equ=sp.solve(sp.Eq(q1_unresv_eq, q1), q1)[0]
q1_equ
```

$$40$$

```
## setter optimal produksjonskvantum til bedrift 1 (q1=40) inn i reaksjonsfunksjon til bedrift 2
q2_unresv_eq = q2_equ.subs(q1, q1_equ)
```

```
# Løser for å finne optimalt kvantum for bedrift 1
q2_equ=sp.solve(sp.Eq(q2_unresv_eq, q2), q2)[0]
q2_equ
```

$$40$$

```
demand_1(q1).subs({q1:q1_equ,q2:q2_equ})
```

$$70$$

```
# Profitt for begge bedriftene er lik (P-c)q, hvor q er lik 40
def profitt(q1):
    return (demand_1(q1).subs({q1:q1_equ,q2:q2_equ})-30)*q1_equ
round(profitt(q1),2)
```

$$1600$$

Optimal tilpasning ved avvik når bedrift 1 produserer avtalt mengde på 30 og bedrift 2 velger å avvike fra avtalen om samarbeid

```
def demand(q1):
    return (150-q1-q2)
```

```
def demand(q2):
    return (150-q1-q2)
```

```
def marginalrevenue(q2):
    return (150-2*q2-q1)
```

```
def marginalrevenue(q1):
    return (150-2*q1-q2)
```

```
equ=sp.Eq(marginalrevenue_2(q2),30)
equ
```

$$-q_1 - 2q_2 + 150 = 30$$

```
#reaksjonsfunksjon til bedrift 2
q2equ=sp.solve(equ,q2)[0]
q2equ
```

$$60 - \frac{q_1}{2}$$

```
def prod(q1):
    return (30)
```

```
# Bedrift 2 vil produsere følgende kvantum:
q2equ.subs({q1:prod(q1)})
```

45

```
# prisen i markedet blir:
demand(q2).subs({q1:prod(q1),q2:q2equ.subs({q1:prod(q1)})})
```

75

```
# profitt til bedrift 2:
round((demand(q2).subs({q1:prod(q1),q2:q2equ.subs({q1:prod(q1)})})-30)*(q2equ.subs({q1:prod(q1)})),3)
```

2025

```
# profitt til bedrift 1:
round((demand(q2).subs({q1:prod(q1),q2:q2equ.subs({q1:prod(q1)})})-30)*(prod(q1)),3)
```

1350

Prissamarbeid

Gjentatte spill – trigger strategi

Periode 1: Samarbeid

Periode 2: samarbeid hvis konkurrenten valgte samarbeid i forrige periode, hvis ikke velges avvik

	Samarbeid	Avvik
Samarbeid	1800,1800	1350, 2025
Avvik	2025, 1350	1600, 1600

Koordinert prissetting

Bedriftene tar hensyn til at de møtes flere ganger i markedet; har mulighet til å koordinere sin adferd og derigjennom oppnå høyre profitt

- To motstridende effekter av å bryte ut av prissamarbeid

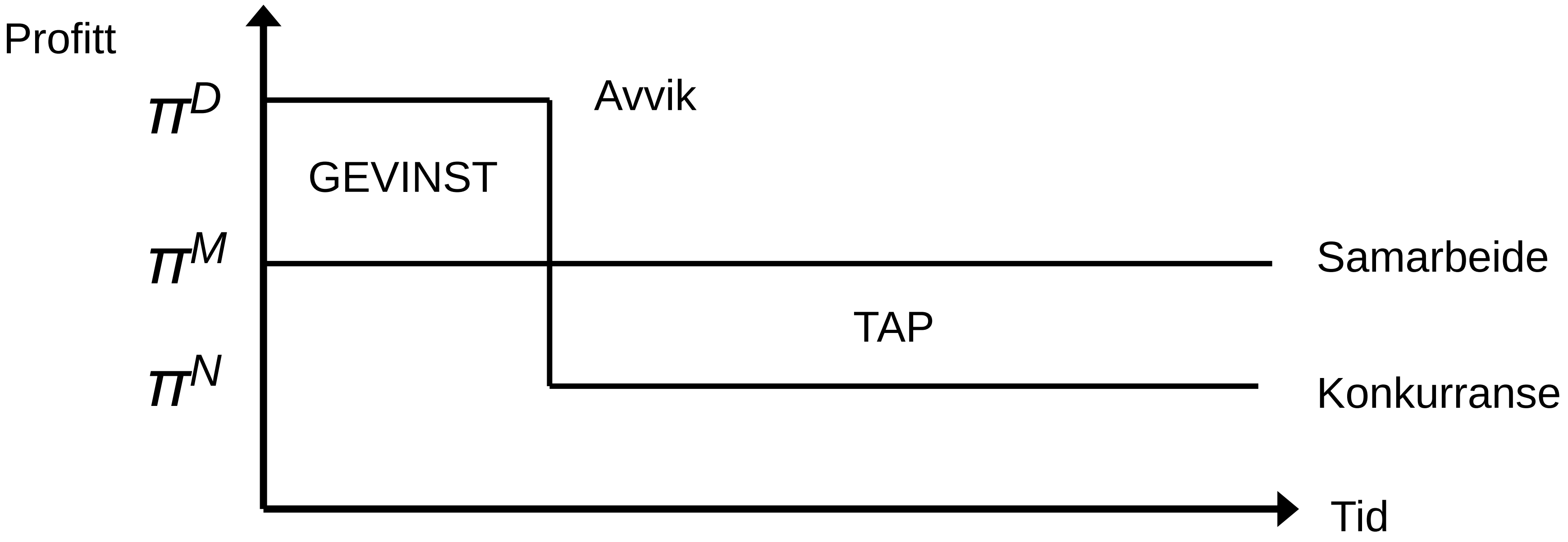
  → Setter pris under rivalens pris

  - *Kort sikt:* Økt profitt siden en tar markedsandeler fra de andre bedriftene

  - *Lang sikt:* Redusert profitt fordi 'bruddet' fører til hardere konkurranse i framtiden

Prissamarbeid?

- Sett monopolpris i denne periode hvis begge satt monopolpris i forrige periode.
- Hvis ikke, opptre som i statisk Nash-likevekt (Konkurranse)



Avveining: *Kortsiktig profitt* $\leftrightarrow$ *Langsiktig tap*

Når vil det lønne seg med samarbeid ved Cournot konkurranse?

Nåverdien av samarbeid > nåverdien ved avvik

```python
from sympy import Symbol
from sympy.solvers.inequalities import reduce_rational_inequalities
```

```python
x = Symbol('x', real=True)
```

```python
# Nåverdi ved sammarbeid > nåverdi ved avvik - med p=1
reduce_rational_inequalities([[1800/(1-x) > 2025+(x*1600)/(1-x)]], x)
```

$$\frac{9}{17} < x \wedge x < 1$$

```python
r = Symbol('r', real=True)
```

```python
reduce_rational_inequalities([[1/(1+r) > 9/17]], r)
```

$$-1.0 < r \wedge r < 0.888888888888889$$

```python
# Nåverdi ved sammarbeid > nåverdi ved avvik - med p=0.6
reduce_rational_inequalities([[1800/(1-(0.6*x)) > 2025+((0.6*x)*1600)/(1-(0.6*x))]], x)
```

$$0.882352941176471 < x \wedge x < 1.66666666666667$$

```python
reduce_rational_inequalities([[1/(1+r) > 0.88235]], r)
```

$$-1.0 < r \wedge r < 0.133337111123704$$

Når vil det lønne seg å sammarbeide under Bertrand konkurranse?

```
# Nåverdi ved sammarbeid > nåverdi ved avvik -
reduce_rational_inequalities([[1800/(1-x) > 3600]], x)
```

$$\frac{1}{2} < x \wedge x < 1$$

```
reduce_rational_inequalities([[1/(1+r) > 0.5]], r)
```

$$-1.0 < r \wedge r < 1.0$$

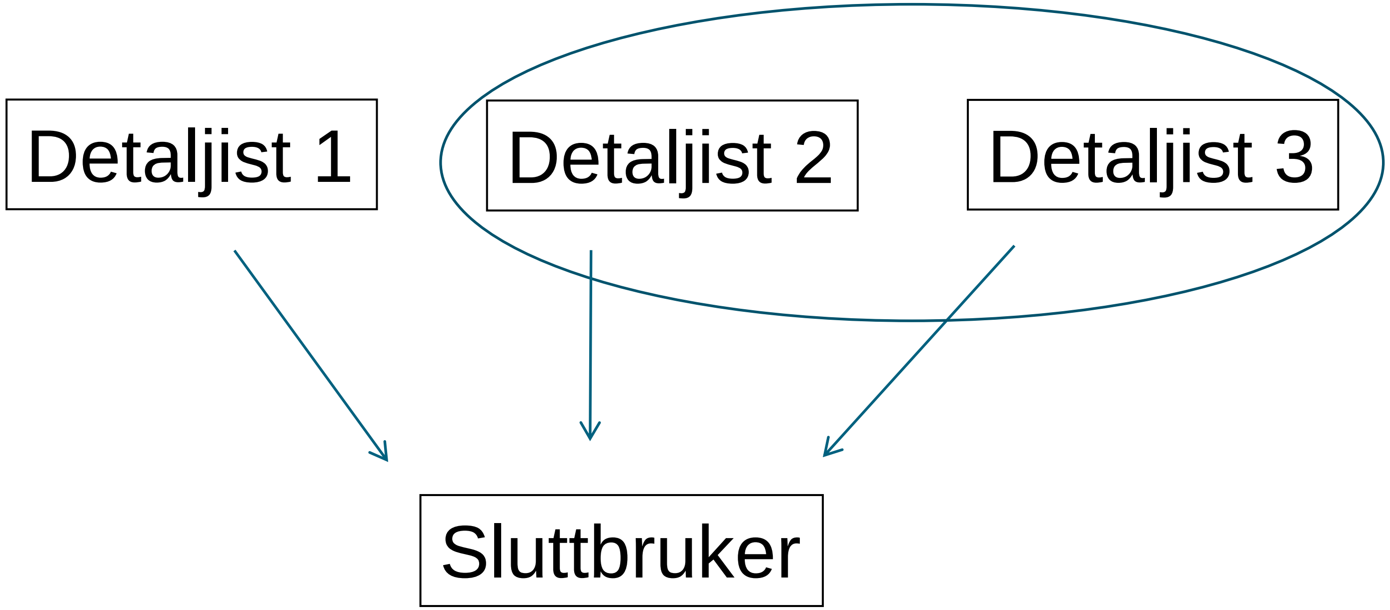
Når vil det lønne seg med samarbeid?

Nåverdien av samarbeid > nåverdien ved avvik

$$\underbrace{\frac{\pi^M}{1-\rho}}_{\text{Kartell}} > \underbrace{\pi^D}_{\text{Avvik}} + \underbrace{\frac{\rho\pi^N}{1-\rho}}_{\text{Konkurranse}}$$

Individuelt rasjonelt å opprettholde samarbeid dersom: $\boldsymbol{\rho} > \frac{\boldsymbol{\pi^D - \pi^M}}{\boldsymbol{\pi^D - \pi^N}}$

Horisontale fusjoner



- **Er det lønnsomt med fusjoner? For hvem er det lønnsomt?**

Python–koder: Cournot tilpasning før fusjon

```
[2]: q1, q2,q3,c, a, b=symbols('q1 q2 q3 c a b')
```

```
[3]: def P_demand(Q,a,b):
         return a-b*Q

     def profit(q1,q2,q3,c,a,b):
         return (P_demand(q1+q2+q3,a,b)-c)*q1
```

```
[4]: d_profit1_Q=diff(profit(q1,q2,q3,c,a,b),q1)
     d_profit2_Q=diff(profit(q2,q1,q3,c,a,b),q2)
     d_profit3_Q=diff(profit(q3,q2,q1,c,a,b),q3)

     display(d_profit1_Q)
     display(d_profit2_Q)
     display(d_profit3_Q)
```

$$a - bq_1 - b\left(q_1 + q_2 + q_3\right) - c$$

$$a - bq_2 - b\left(q_1 + q_2 + q_3\right) - c$$

$$a - bq_3 - b\left(q_1 + q_2 + q_3\right) - c$$

```
[5]: sol=solve([d_profit1_Q,d_profit2_Q,d_profit3_Q],[q1,q2,q3])

     display(sol[q1])
     display(sol[q2])
     display(sol[q3])
```

$$\frac{a-c}{4b}$$

$$\frac{a-c}{4b}$$

$$\frac{a-c}{4b}$$

Python–koder: Cournot tilpasning før fusjon

```
cournot=lambdify(
    (a,b,c),
    (sol[q1],sol[q2],sol[q3])
)
```

```
cournot(150,1,30)
```

```
(30.0, 30.0, 30.0)
```

```
a_value=150
b_value=1
c_value=30
```

```
q1sol, q2sol, q3sol=cournot(a_value,b_value,c_value)
print (P_demand(q1sol+q2sol+q3sol,a_value,b_value))
```

```
60.0
```

```
print(f"""Løsningen er at
bedriftene produserer {q1sol} enheter som gir profitt lik {profit(q1sol,q2sol,q2sol,c_value,a_value,b_value)} og
prisen i markedet blir {P_demand(q1sol+q2sol+q3sol,a_value,b_value)}""")
```

```
Løsningen er at
bedriftene produserer 30.0 enheter som gir profitt lik 900.0 og
prisen i markedet blir 60.0
```

Python–koder: Cournot tilpasning etter fusjon

Anta at 2 av bedriftene fusjonerer, slik at det nå kun er to bedrifter i markedet. Ny tilpasning blir da:

11]:
```
def P_demand1(Q,a,b):
    return a-b*Q

def profitF(q1,q2,c,a,b):
    return (P_demand(q1+q2,a,b)-c)*q1
```

12]:
```
d_profitF1_Q=diff(profitF(q1,q2,c,a,b),q1)
d_profitF2_Q=diff(profitF(q2,q1,c,a,b),q2)

display(d_profitF1_Q)
display(d_profitF2_Q)
```

$$a - bq_1 - b(q_1 + q_2) - c$$

$$a - bq_2 - b(q_1 + q_2) - c$$

13]:
```
sol=solve([d_profitF1_Q,d_profitF2_Q],[q1,q2])

display(sol[q1])
display(sol[q2])
```

$$\frac{a - c}{3b}$$

$$\frac{a - c}{3b}$$

Python–koder: Cournot tilpasning etter fusjon

```
4]: cournot=lambdify(
        (a,b,c),
        (sol[q1],sol[q2])
    )
```

```
5]: cournot(150,1,30)
```

```
5]: (40.0, 40.0)
```

```
6]: a_value=150
    b_value=1
    c_value=30
```

```
7]: q1sol, q2sol=cournot(a_value,b_value,c_value)
    print (P_demand(q1sol+q2sol,a_value,b_value))
```

```
70.0
```

```
8]: print(f"""Løsningen er at
    bedriftene produserer {q1sol} enheter som gir profitt lik {profitF(q1sol,q2sol,c_value,a_value,b_value)} og
    prisen i markedet blir {P_demand1(q1sol+q2sol,a_value,b_value)}""")
```

```
Løsningen er at
bedriftene produserer 40.0 enheter som gir profitt lik 1600.0 og
prisen i markedet blir 70.0
```

```
]:
```

Fusjonsparadokset

Eksempel med Cournot modell og 3 bedrifter

- Invers etterspørselsfunksjon: $P = 150 - (q_1 + q_2 + q_1)$ og marginalkostnad c = 30
- Antar at bedrift 2 og 3 fusjonerer

Markedskonsekvenser etter fusjon :

- Markedsprisen øker fra 60 til 70, solgt kvantum går ned fra 90 til 80
- Den bedriften som ikke er med i fusjonen tjener på fusjon: $\Delta\pi^{C}{}_{1} = 1600 - 900 = 700$
- De fusjonerte bedriftene taper på fusjon: $\Delta\pi^{C}{}_{23} = 1600 - (2 * 900) = -200$

Hvorfor skjer dette?

Horisontale fusjoner og kostnadssynergier

Vi antar at vi har tre bedrifter som konkurrer med kvantum som strategisk variabel. Invers etterspørsel er gitt ved: P(Q) = 150-Q og marginalkostnaden til to av bedriftene er på 30. Den tredje bedriften har potensielt høyere marginalkostnader; lik 30b, hvor b>1.

Markedslikevekt før fusjon

```
q1,q2,q3,c1,c2,c3, a, b=symbols('q1 q2 q3 c1 c2 c3 a b')
```

```
def P_demand(Q,a,b):
    return a-b*Q

def profit(q1,q2,q3,c,a,b):
    return (P_demand(q1+q2+q3,a,b)-c)*q1
```

```
d_profit1_Q=diff(profit(q1,q2,q3,c1,a,b),q1)
d_profit2_Q=diff(profit(q2,q1,q3,c2,a,b),q2)
d_profit3_Q=diff(profit(q3,q2,q1,c3,a,b),q3)
display(d_profit1_Q)
display(d_profit2_Q)
display(d_profit3_Q)
```

$$a - bq_1 - b(q_1 + q_2 + q_3) - c_1$$

$$a - bq_2 - b(q_1 + q_2 + q_3) - c_2$$

$$a - bq_3 - b(q_1 + q_2 + q_3) - c_3$$

```
sol=solve([d_profit1_Q,d_profit2_Q,d_profit3_Q],[q1,q2,q3])

display(sol[q1])
display(sol[q2])
display(sol[q3])
```

$$\frac{a - 3c_1 + c_2 + c_3}{4b}$$

$$\frac{a + c_1 - 3c_2 + c_3}{4b}$$

$$\frac{a + c_1 + c_2 - 3c_3}{4b}$$

Markedslikevekt før fusjon

```
cournot=lambdify(
    (a,b,c1,c2,c3),
    (sol[q1],sol[q2],sol[q3])
)
```

```
cournot(150,1,30,30,30*b)
```

```
(7.5*b + 22.5, 7.5*b + 22.5, 52.5 - 22.5*b)
```

```
a_value=150
b_value=1
c1_value=30
c2_value=30
c3_value=30*b
```

```
q1sol, q2sol, q3sol=cournot(a_value,b_value,c1_value,c2_value,c3_value)
print (P_demand(q1sol+q2sol+q3sol,a_value,b_value))
```

```
7.5*b + 52.5
```

```
print(f"""Løsningen er at
bedrift 1 produserer {q1sol} med profitt {profit(q1sol,q2sol,q3sol,c1_value,a_value,b_value)},
bedrift 2 produserer {q2sol} med profitt {profit(q2sol,q1sol,q3sol,c2_value,a_value,b_value)} og
bedrift 3 produserer {q3sol} med profitt {profit(q3sol,q1sol,q3sol,c3_value,a_value,b_value)}
prisen blir {P_demand(q1sol+q2sol+q3sol,a_value,b_value)}""")
```

```
Løsningen er at
bedrift 1 produserer 7.5*b + 22.5 med profitt 506.25*(0.333333333333333*b + 1)**2,
bedrift 2 produserer 7.5*b + 22.5 med profitt 506.25*(0.333333333333333*b + 1)**2 og
bedrift 3 produserer 52.5 - 22.5*b med profitt (52.5 - 22.5*b)*(7.5*b + 22.5)
prisen blir 7.5*b + 52.5
```

Markedslikevekt etter fusjon

Anta at 2 og 3 fusjonerer, og at all produksjon flyttes til bedrift 2. Ny tilpasning blir da:

```
def P_demand1(Q,a,b):
    return a-b*Q

def profitF(q1,q2,c,a,b):
    return (P_demand(q1+q2,a,b)-c)*q1
```

```
d_profitF1_Q=diff(profitF(q1,q2,c1,a,b),q1)
d_profitF2_Q=diff(profitF(q2,q1,c2,a,b),q2)

display(d_profitF1_Q)
display(d_profitF2_Q)
```

$$a - bq_1 - b\left(q_1 + q_2\right) - c_1$$

$$a - bq_2 - b\left(q_1 + q_2\right) - c_2$$

```
sol=solve([d_profitF1_Q,d_profitF2_Q],[q1,q2])

display(sol[q1])
display(sol[q2])
```

$$\frac{a - 2c_1 + c_2}{3b}$$

$$\frac{a + c_1 - 2c_2}{3b}$$

Markedslikevekt etter fusjon

```
cournot=lambdify(
    (a,b,c1,c2),
    (sol[q1],sol[q2])
)
```

```
cournot(150,1,30,30)
```

```
(40.0, 40.0)
```

```
a_value=150
b_value=1
c1_value=30
c2_value=30
```

```
q1sol, q2sol=cournot(a_value,b_value,c1_value,c2_value)
print (P_demand(q1sol+q2sol,a_value,b_value))
```

```
70.0
```

```
print(f"""Løsningen er at
bedriftene produserer {q1sol} enheter som gir profitt lik {profitF(q1sol,q2sol,c1_value,a_value,b_value)} og
prisen i markedet blir {P_demand1(q1sol+q2sol,a_value,b_value)}""")
```

```
Løsningen er at
bedriftene produserer 40.0 enheter som gir profitt lik 1600.0 og
prisen i markedet blir 70.0
```

Når er fusjon lønnsom?

Fusjon med asymmetrisk bedrifter – Cournot modell med 3 bedrifter, hvor en høykostnadsbedrift fusjonerer med en lavkostnadsbedrift.

$$\text{Fusjon er lønnsom hvis:} \ \pi^{C}{}_{23} > \pi^{C}{}_{2} + \pi^{C}{}_{3}$$

$$\Rightarrow 1600 > \left[\frac{90 + 30b}{4}\right]^2 + \left[\frac{210 - 90b}{4}\right]^2$$

$$\Rightarrow \text{Betingelse for lønnsom fusjon:} \quad b > \frac{19}{15}$$

En fusjon er lønnsom så lenge kostnadsulempen til høykostnadsbedriften er «stor nok»

Når er fusjon lønnsom?

Fusjon med asymmetrisk bedrifter og faste kostnader. Anta at bedrift 2 og 3 fusjonere, og de faste kostnadene for den fusjonerte bedriften reduseres til *af*, hvor 1 < a < 2

$$\text{Fusjon er lønnsom hvis: } \pi^{C}{}_{23} > \pi^{C}{}_{2} + \pi^{C}{}_{3}$$

$$\Rightarrow 1600 - af > 1800 - 2f$$

$$\Rightarrow \text{Betingelse for lønnsom fusjon: } \; a < 2 - \frac{200}{f}$$

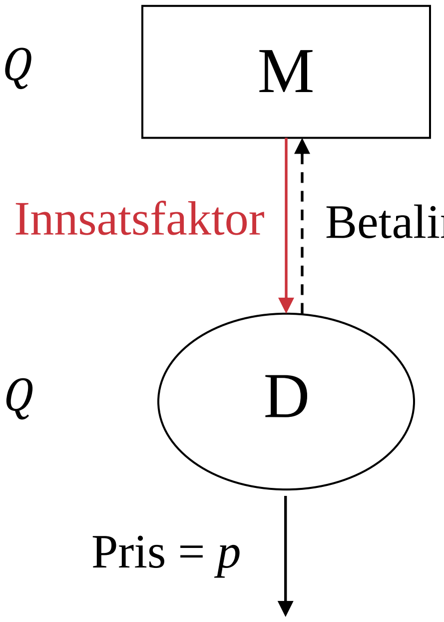
Sannsynligheten for en lønnsom fusjon er større når de faste kostnadene er relative høye slik at synergieffekten (sparte kostnader) er stor.

Vertikale relasjoner: Vertikal separasjon

Verdikjeden

$max\ \pi^M = (r - c)Q$

M produserer og selger et produkt som en annen bedrift benytter i sin produksjon



$max\ \pi^R = (P - r)Q$

D produserer for salg til sluttbrukere

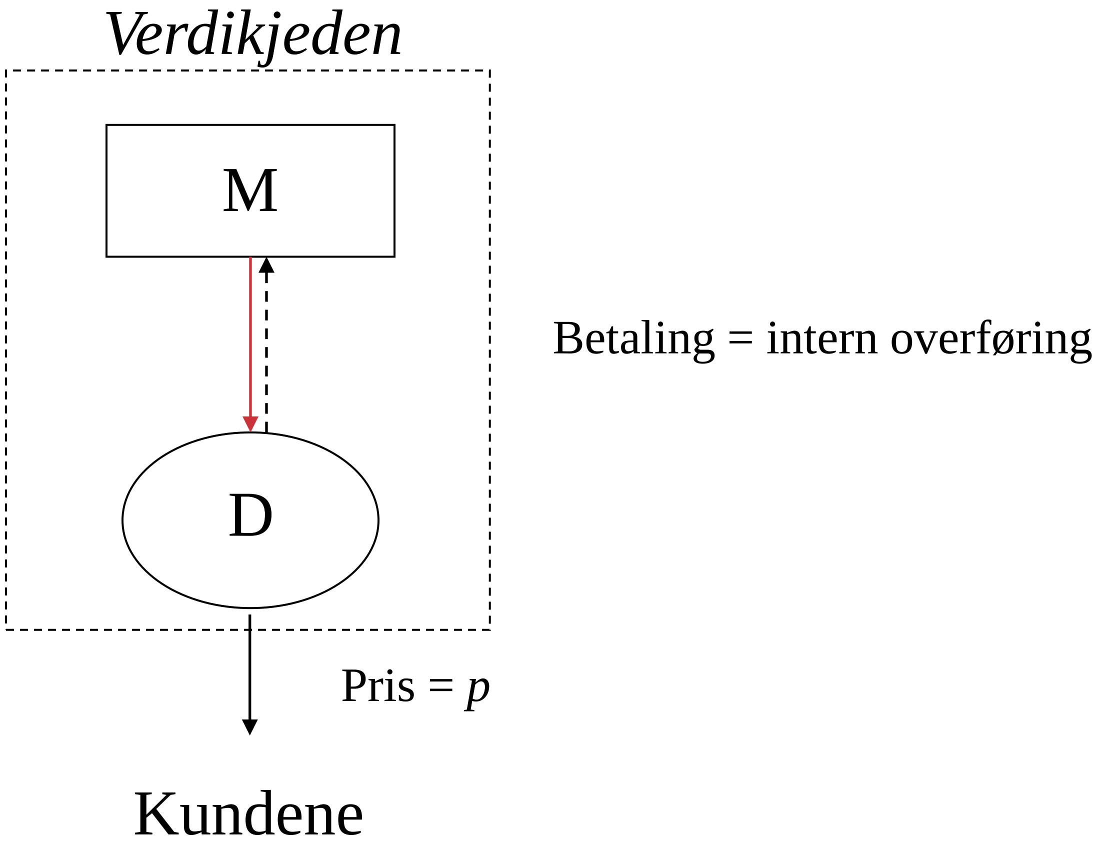
Kundene

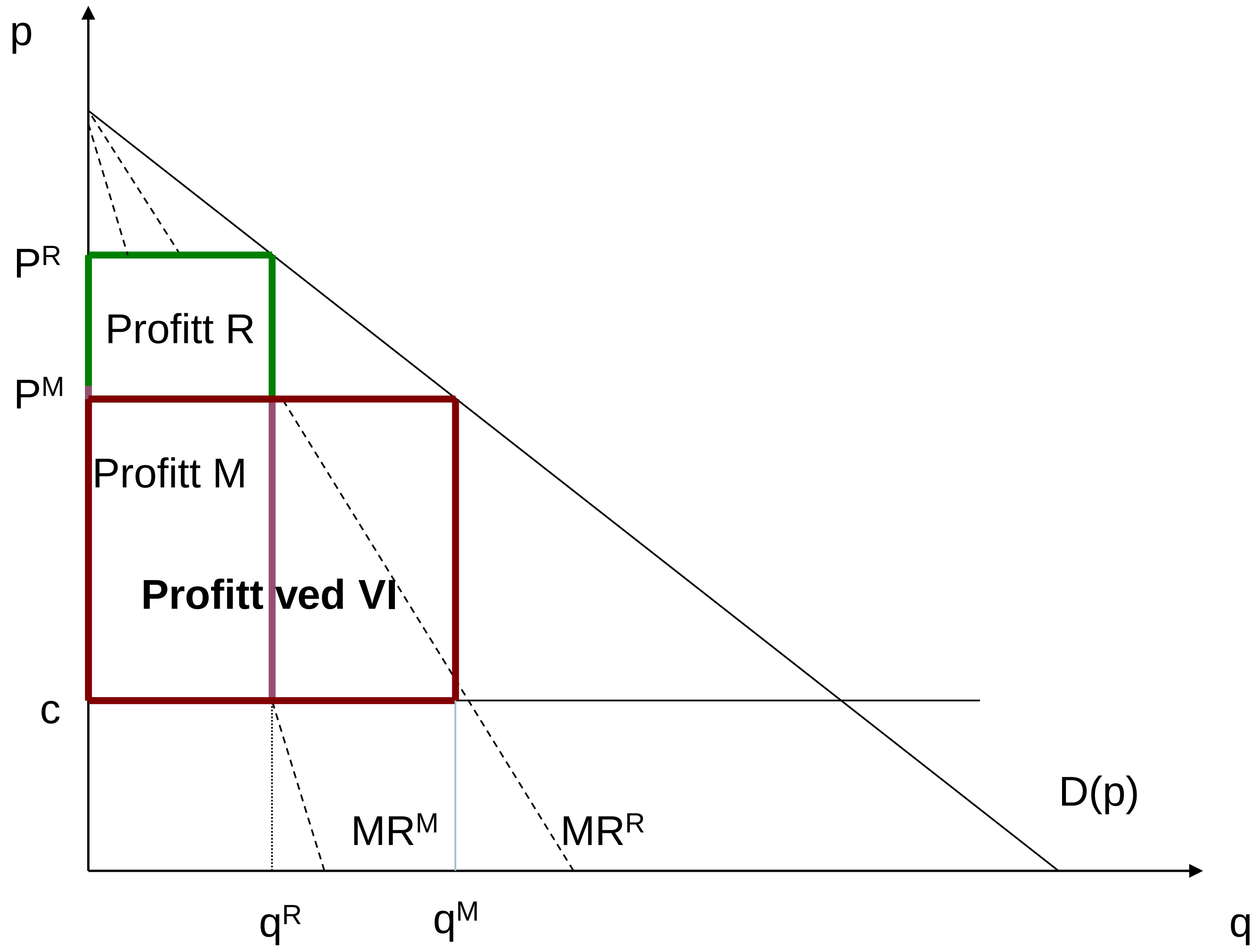
Dobbel- marginalisering $\Rightarrow Q^R = \dfrac{A - c}{4B}$ $\Rightarrow P^R = \dfrac{3A + c}{4}$

Vertikal integrasjon

$$\max \pi^{VI} = (P - c)Q$$

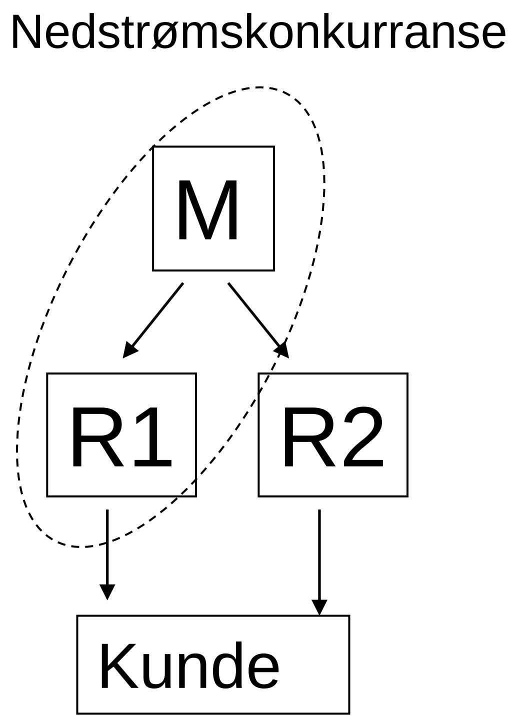
$$\Rightarrow Q^{VI} = \frac{A - c}{2B} \qquad \Rightarrow P^{R} = \frac{A + c}{2}$$

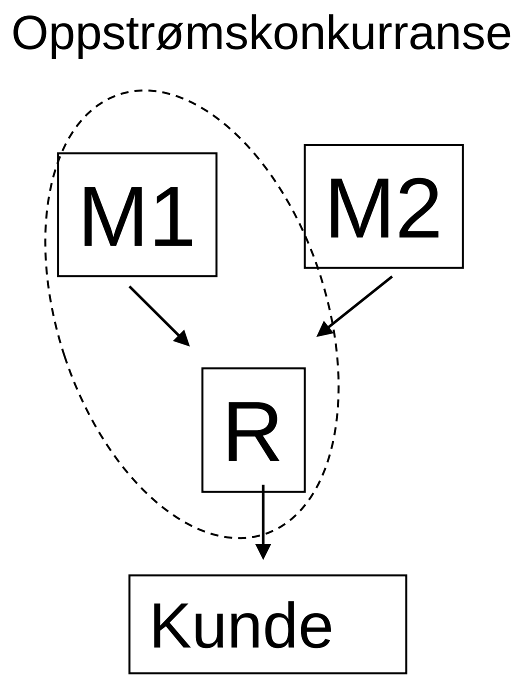




p
P^R
Profitt R
P^M
Profitt M
Profitt ved VI
c
D(p)
MR^M
MR^R
q^R
q^M
q

Vertikale markeder





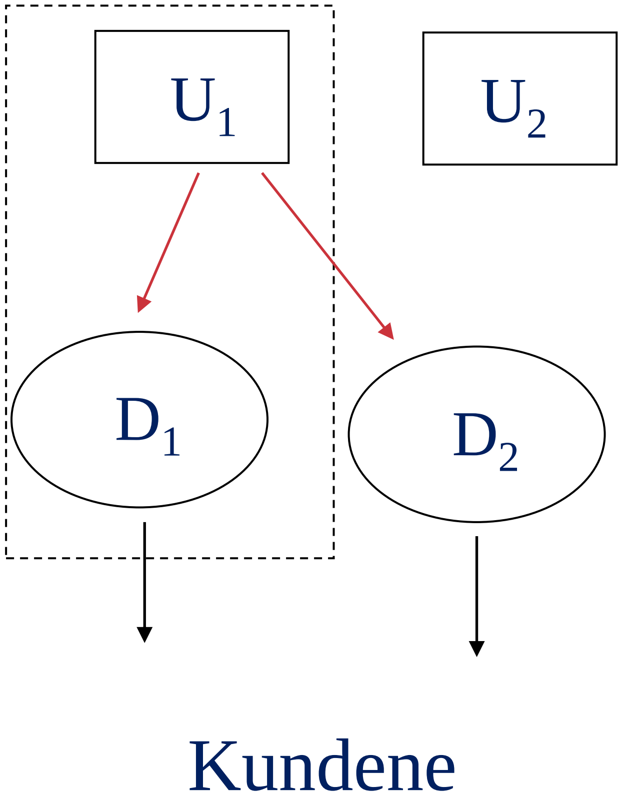
Løses ved 2-trinns spill:

Trinn 1: Produsent velger optimal pris på innsatsfaktor

Trinn 2: Detaljist velger Optimalt kvantum

Vertikal integrasjon og utestengelse i Cournot modell?

Vertikale fusjoner, oligopol og markedsutestengelse



Etterspørsel: $P = A - B(q_1 + q_2)$

Fast forhold mellom innsats og sluttprodukt

Pris på innsatsfaktor for D_1 og D_2 er r

Marginalkostnad for bedrift U_1 og U_2 er c^U

Marginalkostnad for bedrift D_1 og D_2 er c^D

To-trinns spill:

Trinn 1: U_1 og U_2 velger optimal engropris r

Trinn 2: D_1 og D_2 velger optimalt kvantum

Vertikale bindinger

Problemet ved dobbel-marginalisering kan løses ved ulike kontrakter:

1. To-delt tariff: $T = rq + f$
2. Maksimal sluttbrukerpris: $w = p^M$
3. RPM – bindende videresalgspris: $P_{min} = P^M$
4. Eksklusive områder (Eksklusive avtaler og eneforhandler)

Hjemmeeksamen

- ***Studentens evne til å reflektere, vurdere og analysere***
  - En godebesvarelse kjennetegnes ved:
    - god økonomisk forståelse koblet sammen med formell analyse (grafisk og/eller matematisk)
    - grundig redegjørelses av de økonomiske modeller og løsningskonsepter som brukes i besvarelsen
  - Python er verktøy for å analysere en problemstilling, men det er den økonomiske intuisjonen som er viktigst
  - Ta egne forutsetninger der dere finner det nødvendig

Plagiat

- ***Å plagiere er å presentere noen andres arbeid som sitt eget.***

- Plagiat handler ikke utelukkende om direkte gjenbruk av tekst, men om hvem som utførte arbeidet.

- Krav til kildehenvisninger:
  - Kode som du henter fra andre kilder må siteres - se MIT retningslinjer
  - Ved bruk av ChatGPT skal du levere et appendiks til besvarelsen som viser hvordan du har brukt dette hjelpemidlet

ChatGPT og eksamen

- Studenter som ønsker å bruke et KI-verktøy, f.eks. ChatGPT, må ha innsikt i hvordan dette verktøyet kan brukes lovlig for ikke å risikere å bli tatt i fusk. UiT sine retningslinjer for bruk av ChatGPT på eksamen sier følgende:
- «Dersom ChatGPT eller andre KI-verktøy brukes i arbeidet ditt, må du i besvarelsen beskrive hvordan du har brukt verktøyet. APA har beskrevet hvordan dette kan gjøres i henhold til eksisterende APA-stil. Her ser du hvordan du kan referere korrekt.
- Åpenhet omkring metode i arbeidet er alltid forventet av deg som student, og å levere inn tekst produsert av en annen kilde enn deg selv uten å oppgi referanse regnes som fusk. For eksempel innebærer dette at du ikke kan kopiere tekst fra ChatGPT og lime inn dette i besvarelsen din uten henvisning».
- Å kopiere inn en betydelig andel tekst fra et KI-verktøy, eller annen kilde, kan bli tolket som mangel på selvstendighet og kan medføre karaktertrekk, selv om teksten er referert korrekt. Verktøyet som brukes for å avdekke plagiering viser hvor stor andel av din besvarelse som kan spores til andre tekster.