Om obligatoriske oppgave 2

Bakgrunn og tips

Forelesning 31/10, 2005

MAT-INF1100

Oblig 2

Hovedelementer

1. Dekomponering og interpolasjon
2. Kompresjon

Siktemål

- Gi eksempler på manipulering av tidsserie; med lyd som eksempel
- Koding av en algoritme med arrayer og indekshåndtering
- Testing av kode på kjent problem
- Inpassing av egen kodebit i et eksisterende rammeverk

Kompresjon

Viktig for lagring og sending av data.
To hovedtyper:

Tapsløs (lossless)
Ingen informasjon går tapt eller fordreies – orginale data kan rekonstrueres "eksakt" fra de komprimerte.
Brukt for tekst, programmer, vitenskapelige data etc. (zip, gzip)
Også for feks. bilder (gif,...)

Med tap (lossy)
Informasjon går tapt, men mer effektiv enn "lossless"
Brukt for lyd, bilder og film (jpeg, mpeg,...)

Kan ikke brukes for tekst, programmer etc.

Tapsløs kompresjon; gzip

Brukes mye for å komprimere filer som ikke er i bruk.
Bygger på gjenkjenning og effektiv representasjon av mønstre.
Eksempel med to filer:

null.dat	ver.dat
0.00000000	0.41231056
0.00000000	0.58309519
0.00000000	0.71414284
0.00000000	0.82462113
0.00000000	0.92195445
...	...

Effekt av gzip

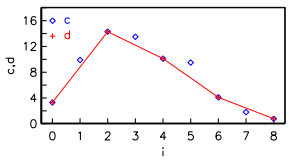
Filer med 10 linjer (bytes med rødt):

```
linux>ls -l -h null.dat ver.dat
-rw-r--r--   1 geirkp geirkp   110 Oct 29 22:36 null.dat
-rw-r--r--   1 geirkp geirkp   110 Oct 29 22:36 ver.dat
linux>gzip null.dat ver.dat
linux>ls -l -h null.dat.gz ver.dat.gz
-rw-r--r--   1 geirkp geirkp    37 Oct 29 22:36 null.dat.gz
-rw-r--r--   1 geirkp geirkp    95 Oct 29 22:36 ver.dat.gz
```

Med 1000 linjer (størrelse i bytes):
ukomprimert: 11000, null.dat.gz: 94, ver.dat.gz: 5000
Det er gjentagelser (og ikke at det er nuller) som gjør forskjellen.

gzip er en vesentlig del av kompresjonen i obligen

Flerskala-dekomponering



Opprinnelig sample $c_0, ...c_{2n}$, grovere sample $d_0, ..., d_n$:

$$d_i = c_{2i}, \quad i = 0, n$$

Avvik fra lineær interpolasjon i mellomnoder:

$$w_i = c_{2i+1} - \frac{1}{2}(d_{i+1} + d_i), \quad i = 0, n-1$$

$2n+1$ c'er erstattet av totalt like mange d'er or w'er.

Flerskala-dekomponering;egenskaper

- I utgangspunktet: intet tap, ingen kompresjon. Bare en reorganisering av informasjonen
- Kompresjon (lossy!) kommer inn ved at $|w_i| <$toleranse $\Rightarrow w_i$ settes til null.
  Totalt vil mange av w'ene da kunne bli null og kompresjon av typen gzip blir effektiv
- Dekomponeringen gjentas; $c \to d^1,\ w^1$, $d^1 \to d^2,\ w^2$,....$d^{m-1} \to d^m,\ w^m$.
  Totalt $c \Rightarrow d^m,\ w^1 ... w^m$
  (c, d^i, w^i betegner hele tidsserier (sampler))
- Rekonstruksjon: definisjonsformler for d'er og w'er brukes baklengs
- For lyd: alle samplene er 16 bits hetall; **short** i Java

Problem: antall sampler i c er like

Vi har $c_0, ...c_{2n-1}$, mulig metode:

1. $c_0, ...c_{2n-2}$ behandles som før og gir $d_0, .., d_{n-1}$, $w_0, ...w_{n-2}$
2. c_{2n-1} spesialbehandles ved innføring av w_{n-1}: $w_{n-1} = c_{2n-1} - d_{n-1}$

Dekomponering av odde sample: en flere av d enn w
Dekomponering av like sample: like mange d og w

Mye er likt for de to tilfellene; dette bør gjenspeile seg i programmeringen

Obligatorisk oppgave

1. Litt leking med lyd. Komme i gang. Start med fila **Filters.java**
2. Håndregning av en enkel dekomponering. Lære algoritmen å kjenne og etablere et testseksempel med kjent løsning
3. Kode dekomponering, teste kode og prøve ut dekomponering på en lyd. Valgfri del med multippel dekomponering.

I dag snakker vi bare om siste del.

Nedlasting, filflytting

Anbefaling: lag en mappe og flytt deg dit

mkdir -p kurs/mat-inf1100/oblig2
cd kurs/mat-inf1100/oblig2
wget ....
mkdir oppg3a
cp *.java oppg3a
rm ....

mkdir lager mappe(r), opsjon **-p** oppretter undermapper
cd bytter arbeidsmappe, **wget** henter filer fra nettet (adresse gitt på WEB)
cp kopierer de filer man trenger, men **rm** fjerner filer (det har fulgt en del "skrot" med **wget**
Vi bør arbeide med 3a og 3b i ulike mapper

Filhåndtering i Linux

flere filer håndteres samtidig med "joker"-notasjon (*)
rm *.htm
fjerner alle filer (i nåværende mappe) som har navn som ender på **.html**
Er siste argument i en kopi eller flytt kommando et mappenavn kopieres/flyttes alt til denne
cp *.java oppg3a

Fjerning av mapper
Først tømme mappen, deretter fjerne
rm maa/*; rmdir maa
(virker ikke om **maa** har undermapper)
Alternativ; rekursiv delete **rm -r maa**
Vær veldig forsiktig med **rm** kommandoen, spesielt kombinert med jokernotasjon (*)!

Javafiler i oppgave 3

- **AbstractCompressedSound.java** inneholder superklasse med samme navn
- **MatInf1100Sound.java** inneholder lydklassen vår
- I **Compressed1100Sound.java** skal vi legge inn en konstruktør som dekomponerer og en **public MatInf1100Sound decompress()**
  Utenfra ser disse like ut i 3a og 3b.
- **test_parabola.java** inneholder en **main** som tester koden i **Compressed1100Sound.java** vha. oppg. 2
- **CompressionTester.java** utfører dekomp., rekonstruksjon av strenglyd og skriver filer
  - **original.obj** opprinnelig generert signal
  - **Compressed1100Sound.obj** dekomp./rekons.

Testing av kode i 3a
>javac test_parabola.java
>java test_parabola
d-array:
...
w-array:
...

Testing av kompresjon i 3a
rm -f *.obj *.obj.gz
javac CompressionTester.java ; java CompressionTester
ls -l -h *.obj
gzip *.obj
ls -l -h *.obj.gz

Effektivisering; for spesielt interesserte

Testing av kompresjon med endringer i kode kan bli mye gjentagelser av kommandoer – slitsomt og gir lett feil. Lag en fil, feks. **run**, med linjene

rm -f *.obj *.obj.gz
javac CompressionTester.java ; java CompressionTester
ls -l -h *.obj
gzip *.obj
ls -l -h *.obj.gz

Nå utføres disse med **tcsh run** eller **source run**.
Dette er et enkelt eksempel på "skripting".
Kommandoshellet i linux er faktiske et programmeringsspråk som også innholder if-setninger og mye mer.