Førsteårsprojekt F2008
Flere grafalgoritmer, og visualisering

Peter Sestoft

2008-03-11*

SØGES

- 1-2 studerende til Åbent Hus
  torsdag 10. april kl 1700-1800
- Skal kunne
  - fortælle 5-10 minutter om hvad hvordan det er at være bachelorstuderende på ITU
  - svare på spørgsmål fra salen

I dag

- Deling af CVS-repository
  - Opret Unix-gruppe: skriv til sysadm@itu.dk
  - Opret fælles CVS-repository hos et gruppemedl.
- Korteste vej i en graf, Dijkstras algoritme
- Projekt: visualisering vejnet og ruter
  - visning med farver mv.
  - zoom
  - scroll
  - vejnavne
- Implementer visualisering
  - afleveres 1. april kl 1200

Delt CVS repository

- CVS er essentielt for gruppearbejde
- Opret et fælles, delt CVS repository sådan:
  - Send mail til sysadm@itu.dk om at I gerne vil have en Unix-gruppe
  - Oplys: ønskede medlemmer (jeres brugernavne), og gruppens udløbsdato
  - Opret et CVS repository i et gruppemedlems hjemmekatalog på ssh.itu.dk

```
# mkdir faapcvs
# chgrp -R f8grup1 faapcvs
# chmod g+srwx faapcvs
# cvs -d /import/home/sestoft/faapcvs init
```

Forklaring på besværgelserne

```
# mkdir faapcvs

# chgrp -R f8grup1 faapcvs

# chmod g+srwx faapcvs

# cvs -d /import/home/sestoft/faapcvs init
```

Opret mappe

Sæt mappens gruppe til f8grup1

Giv gruppen læse- og skriveadgang, også til undermapper

Giv gruppen læse- og skriveadgang

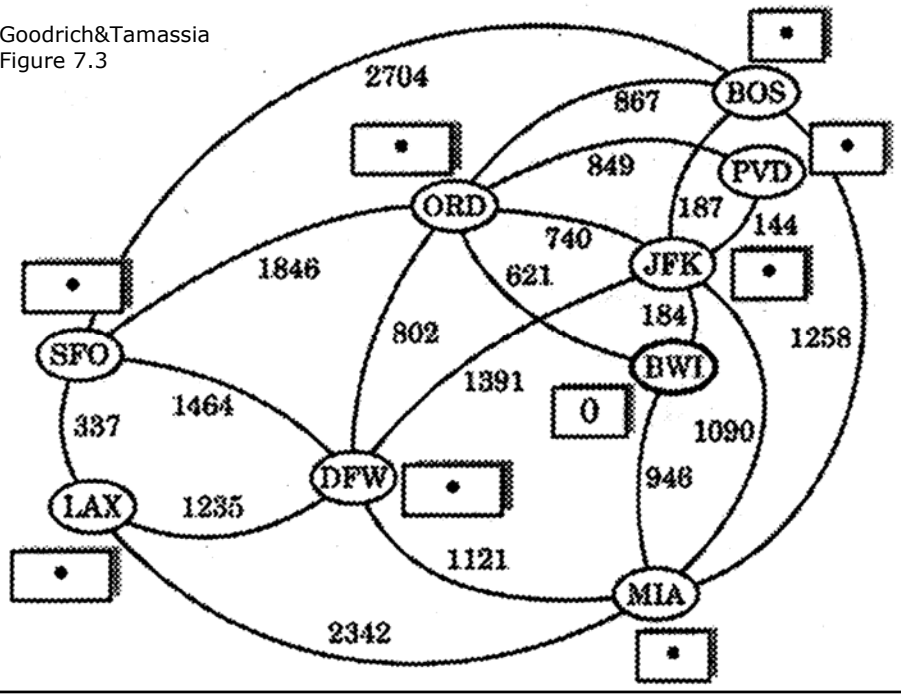
Hvordan *benytte* delt CVS?

- Repositoriet tilknyttes Eclipse som normalt
- Et gruppemedlem opretter et projekt fra Eclipse
- Alle andre gruppemedlemmer importerer det
- Når du skal redigere noget:
  - Udfør **`Team > Update`** i Eclipse
  - Lav dine ændringer
  - Udfør **`Team > Commit`** i Eclipse
- Eller brug **`Team > Synchronize with repository`**

Algoritmer til korteste vej i graf

- "Single source shortest path"
- Givet en startknude s ("source"),
  - find den korteste vej til alle andre knuder;
  - eller til en bestemt knude t ("target")
- Dijkstras algoritme, 1959
  - Virker når alle kantlængder >= 0
  - Hvis grafen er tynd (få kanter ift knuder) er det effektivt at bruge en prioritetskø
- Eksempel fra Goodrich & Tamassia fig 7.3-4

Goodrich&Tamassia
Figure 7.3



Dijkstra pseudo-kode

- Sæt D[v]=∞ for alle knuder undt. D[s]=0
- Opret prioritetskø pq med D[v] som nøgle
- Så længe pq ikke-tom
  - udtag fra pq en knude v med minimal nøgle
  - for hver kant (v,w) ud fra v,
  - hvis D[v] + len(v,w) < D[w], så sæt D[w] = D[v] + len(v,w)

Fandt en kortere vej fra s til w

- Når pq er tom så er D[t] = korteste afstand fra s til t
- Faktisk gælder dette så snart t er blevet udtaget fra prioritetskøen

Husk: breddeførst og dybdeførst

- Analogier:
  - DFS: Vælg næste knude fra en stak
  - BFS: Vælg næste knude fra en kø (og hold eventuelt regnskab med level)
  - Dijkstra: Vælg næste knude fra en prioritetskø (og hold regnskab med afstand fra source)
- BFS finder korteste afstand, i antal kanter, fra source til andre knuder
- Dijkstra finder korteste afstand, i kantlængde, fra source til andre knuder

Dijkstras algoritme i Java

```
Map<KrakNode,ILocator<KrakNode>> locators
  = new HashMap<KrakNode, ILocator<KrakNode>>();
IPriorityQueue<KrakNode> pq = new PriorityQueue<KrakNode>();
for (KrakNode node : graph.nodes)
  if (node != null && node != source) {
    Ilocator<KrakNode> loc =
      pq.insert(node,Double.POSITIVE_INFINITY));
    locators.put(node, loc);
  }
Ilocator<KrakNode> loc = pq.insert(source, 0.0);
locators.put(source, loc);
// Slut på initialisering, her begynder selve algoritmen
while (!pq.isEmpty()) {
  ILocator<KrakNode> v = pq.removeMin();
  for (KrakEdge edge : graph.edges.get(v.getItem().index)) {
    KrakNode w = edge.getOtherEnd(v.getItem());
    if (v.getKey() + edge.length < locators.get(w).getKey())
      locators.get(w).replaceKey(v.getKey() + edge.length);
  }
}
```

Prioritetskø med justerbar nøgle

- Typisk prioritetskø (fx fra java.util)

```
interface IPriorityQueue<T> {
  public void insert(T item, double key);
  public T removeMin();
}
```

Mangler en måde at justere nøgle

- Bedre prioritetskø

```
interface IPriorityQueue<T> {
  public ILocator<T> insert(T item, double key);
  public Ilocator<T> removeMin();
  public boolean isEmpty();
}
interface ILocator<T> {
  public void replaceKey(double key);
  public T getItem();
  public double getKey();
}
```

Insert giver et "håndtag"

Håndtaget bruges ved justering af key

Hvordan finde korteste *vej*?

- Dijkstra finder korteste *afstand*
- Den korteste *vej* findes ved at knytte en forgænger til hver knude
- Så længe pq ikke-tom
  - udtag fra pq en knude v med minimal nøgle
  - for hver kant (v,w) ud fra v,
  - hvis D[v] + len(v,w) < D[w],
    så sæt D[w] = D[v] + len(v,w)
    og sæt w.previous = v
- Vejen (baglæns) findes ved at gå fra t til t.previous osv. til man når s

Dijkstra uegnet til punkt-til-punkt

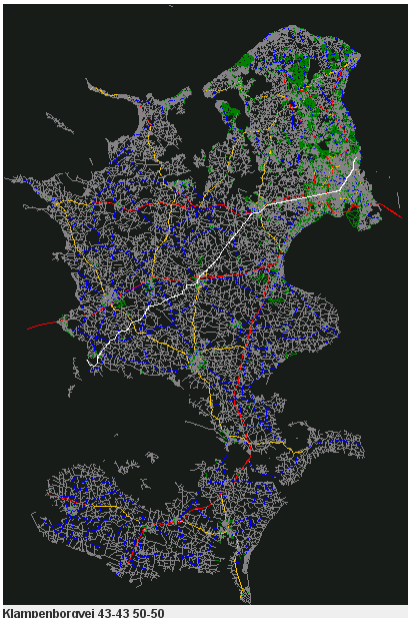
- Ex: Find korteste vej fra s=ITU til t=Nakskov
- Dijkstra finder først afstanden til alle steder der er nærmere end Nakskov
- Dvs næsten alle steder på Sjælland, Møn, ...
- Det er spildt arbejde ...
- En simpel forbedring: Søg fra begge ender samtidig – hjælper noget men ikke meget
- Algoritme A* gør det lidt bedre ved kun at søge "i den geografisk rigtige retning"

Algoritmiske forbedringer

- A* algoritmen mm præsenteres i Algoritmer og Datastrukturer
- Eksperimentelle resultater, se *Sanders & Schultes: Engineering fast route planning algorithms, Workshop on Experimental Algorithmics 2007*
- En hurtig, ikke afsindig indviklet algoritme, se *Sanders & Schultes: Highway hierarchies hasten exact shortest path queries, European Symposium on Algorithms 2005*
- Men måske stadig rigelig kompliceret ...

Visualisering af vejnettet

- 1. Tegn veje på fast størrelse kort
- 2. Tegn med forskellige farver
- 3. Tilpas til vinduets størrelse
- 4. Tillad zoom ind
- 5. Tillad zoom ud
- 6. Tillad scroll op, ned, venstre, højre
- 7. Brug mus til at vise navn på nærmeste vej
- ... og 1000 andre ting



Koordinattransformationer

- Vejdataenes koordinatsystem er UTM32, men (øst,nord) koordinater i meter
- Skærmens koordinatsystem er i pixler (venstre,ned)
- Man skal kunne regne frem og tilbage
  - fra meter til pixel: for at tegne en vej
  - fra pixel til meter: for at vide hvad musen peger på
- Transformationerne er ret simple ...
- ... men skal indpakkes i en klasse, ellers bliver det umådelig rodet

Koordinattransformationsklasse

- Fra meter til pixel:

```
public int gFromX(double x);
public int gFromY(double y);
```

- Fra pixel til meter:

```
public double xFromG(int gX);
public double yFromG(int gY);
```

Opgaver i relation til projekt

- Interaktiv visualisering af vejnettet
- Tirsdag 1. april:
  - Aflevering af program og designdokumenter
- Tirsdag 8. april:
  - Hver gruppe præsenterer sin visualisering