

Datum: 2008-10-20

Laboration 2: "*Sökning*"

Kurs: Artificiell intelligens, KV, HT-08
(5DV063)

Kursansvarig: Christina Olsén

Handledare: Dennis Olsson

Studenter: Oskar Mothander (dit06omr)
Alan Larsson (dit06omr)

Innehållsförteckning

PROBLEMSPECIFIKATION.....	1
ÅTKOMST OCH ANVÄNDARHANDLEDNING.....	1
ALGORITMBESKRIVNING.....	2
LÖSNINGENS BEGRÄNSNINGAR.....	2
PROBLEM OCH REFLEKTIONER.....	3
<i>Flygplatsen – Begbilar</i>	4
<i>MIT – Flygplatsen</i>	5
<i>Nydala - Gamlia</i>	5
<i>Teg - Foa</i>	6

Problemspecifikation

Laborationens uppgift går ut på att programmera 4 sökalgoritmer som ska kunna hitta en väg i ett system uppbyggt av noder och kopplingar med hjälp av en xml-filsom tilldelats. Hur denna väg ska väljas ut beror på den typ av sökalgoritm som används. De 4 olika typer av sökalgoritmer som ska implementeras är Bredden-Först , Djupet-Först, A* och Greedy. För att visa vägen de olika algoritmerna tar så ska detta representeras som en sträng med namnet på de noder som passeras för att anlända till målet. Om algoritmen besöker andra noder som inte tas upp i den slutgiltiga vägen så ska även dessa representeras på något sätt.

Originalspecifikationen kan hämtas på

<http://www.cs.umu.se/kurser/5DV063/HT08/assignments/lab2-spec-ht08.pdf>

Åtkomst och användarhandledning

Programmet kan hittas på `dit06omr\edu\ai\lab2\`

Tredje parametern hänvisar till en xml-fil som representerar en karta.

Fjärde parametern hänvisar till sökalgoritmen som ska användas; `[B | D | G | A]` . Där B är bredden-Först, D: Djupet-Först, G: Greedy, A: A* .

Femte parametern hänvisar till ursprungsplatsen och sjätte till destinationen.

Körkod:

```
java SearchInterface SearchEngine umea_map.xml B Teg Begbilar
```

OBS: Bifogad SearchInterface måste användas vid testning eftersom setMap() måste köras mellan A* kortast och snabbast.

För mer information om programmet och dess struktur så kan programmets javadoc-sida hänvisa:

http://www.cs.umu.se/~dit06omr/ai_lab2_javadoc/

Algoritmbeskrivning

Bredden-Först:

Bredden-först bygger upp en trädliknande struktur med hjälp av en vector. Den börjar med att lägga till startnoden i en vector som håller reda på hittade noder. Sedan påbörjar den for-loopen som används för att söka reda på målnoden. Eftersom startnoden är den enda noden i vectorn för tillfället kommer den finnas på index 0. Algoritmen tar den nod den befinner sig på just vid tillfället och lägger till dess barn/grannar om dessa inte redan finns i vectorn. Praktiskt sett så kommer startnodens grannar hamna på index 1 och framåt. Den första av startnodens grannar kommer sedan lägga till sina grannar efter startnodens sista granne i vectorn. Detta gör att algoritmen kommer få ett bredden-först beteende när den besöker noderna en efter en.

Djupet-Först:

Algoritmen använder sig av en rekursiv metod som hela tiden anropar sig själv med nya parametrar om noden inte hittats. Varje gång den expanderar en nod så läggs den till i en vector med redan besökta noder. Detta förhindrar den från att besöka en granne som redan expanderats. När noden hittats så returnerar metoden en sträng som returneras längst med hela den rekursiva kedjan tills den anlant till början där den skriver ut resultatet.

A* kortaste vägen:

Vid varje nod kontrolleras avstånd till barn/grannar plus fågelvägen från barnet till målnoden. Detta jämförs dels med de andra barnen till noden men även med tidigare Oexpanderade noder. När en nod har besökts/expanderats (kontrollerat dess barn) så kontrolleras denna inte igen i nästa besökta nod.

A* snabbaste vägen:

Samma som kortaste vägen men istället för att jämföra avståndet så mäts tiden det tar att resa mellan noderna. Fågelvägen antas åka i 110 (hårdkodat).

Lösningens begränsningar

Mellan alla (förutom Bredden-först) implementerade algoritmer måste setMap(...) anropas eftersom t.ex. grafen har satt noder som besökta vilket vi inte återställs.

I A* kortaste vägen så antas fågelvägen åka i 110 km/h vilket vi har hårdkodat. Egentligen borde detta vara snabbaste vägen på kartan vilket kan skilja sig kartor i mellan. Vi implementerade emellertid inte detta p.g.a. tidsbrist.

Problem och reflektioner

Labbspecifikationen var väldigt kortfattad. Vi hade uppskattat om den hade funnits i html-format eller att man åtminstone kunde kopiera text från filen.

Det känns som viktig information som inte har tagits upp på lektioner eller i boken saknas, t.ex. att i A* snabbaste vägen så ska man använda sig utav snabbaste vägen på kartan. Den informationen fick vi reda på genom labbkamrater som i sin tur frågat handledaren. Svar på sådana frågor borde kanske skickas ut till alla via kursmailinglistan.

Ett stort problem som kvarstod genom hela arbetet var att arbeta via SVN. Då olika versioner av eclipse och svn användes mellan hem (nyare) och universitets labbsalar så skapades konflikter mellan versionerna som hindrade uppdatering av kod.

Diskussion

Den största nackdelen med Bredden-Först och Djupet-Först är det faktum att de inte genererar en optimal väg medräknat kostnader och vikter mellan noder. Dessa 2 sökalgoritmer returnerar bara första hittade vägen från start till mål.

Djupet-Först är dock en väldigt ineffektiv sökalgoritm för avancerade system då noder kan bilda ”öglor”. Detta leder till att vägen till en nod kan gå en onödig omväg.

En nackdel med Greedy är att den bara väljer vägen till noden som är närmast målet. Detta kan leda till att den hamnar på en omständlig väg t.ex. Mit – Flygplatsen (se bilaga testkörningar). Greedy kan välja en väg som den anser kortast till målet på kort sikt men som egentligen leder till att en längre väg måste tas efteråt. Man kan lätt hamna i ”virvlar” men dock inte i cirklar eftersom man inte får passera samma plats flera gånger. Den väljer endast en ny väg om den kommer till en återvändsgränd.

Vi anser att A* är den smartaste algoritmen av de implementerade, dock var den svårast att implementera. Den gav emellertid inte alltid den optimala vägen (t.ex. Teg – Foa) där greedy och bredden-först faktiskt gav en bättre väg.

Tegsbron - Nydala

SearchEngine söker med bredden först...

Expanderar:

Tegsbron, Teg, I20, Rondellen, Obs, Station, Kyrkan, TreKorvar, Begbilar, Foa, Gamlia, NUS, On, Flygplatsen, Mariehem, Berghem, Sofiehem, Nydala klar.

Resa: Tegsbron, I20, Obs, Foa, Mariehem, Nydala

SearchEngine söker med djupet först...

Expanderar: Tegsbron, Teg, Rondellen, Kyrkan, Station, I20, Obs, Begbilar, Foa, Mariehem, Berghem, Gamlia, NUS, Sofiehem, GimonAs, Ok, Alidhem, MIT, klar.

Resa: Tegsbron, Teg, Rondellen, Kyrkan, Station, I20, Obs, Begbilar, Foa, Mariehem, Berghem, Gamlia, NUS, Sofiehem, GimonAs, Ok, Alidhem, MIT, Nydala

SearchEngine söker med A* (kortaste vägen)...

Expanderade noder: Tegsbron, I20, Station, Gamlia, Berghem, NUS, MIT, klar.

Tegsbron, I20, Station, Gamlia, Berghem, MIT, Nydala

söker med A* (snabbaste vägen)...

Expanderade noder: Tegsbron, I20, Station, Gamlia, NUS, Sofiehem, Alidhem, MIT, klar.

Tegsbron, I20, Station, Gamlia, NUS, Sofiehem, Alidhem, MIT, Nydala

SearchEngine söker med Greedy Search...

Expanderade noder: Tegsbron, I20, Station, Gamlia, Berghem, MIT, klar.

Tegsbron, I20, Station, Gamlia, Berghem, MIT, Nydala

Flygplatsen – Begbilar

SearchEngine söker med bredden först...

Expanderar:

Flygplatsen, TreKorvar, Rondellen, On, Teg, Kyrkan, Tegsbron, Station, NUS, I20, Obs, Gamlia, Sofiehem, Begbilar klar.

Resa: Flygplatsen, TreKorvar, Rondellen, Kyrkan, Station, Obs, Begbilar

SearchEngine söker med djupet först...

Expanderar: Flygplatsen, TreKorvar, Rondellen, Teg, Tegsbron, I20, Obs, klar.

Resa: Flygplatsen, TreKorvar, Rondellen, Teg, Tegsbron, I20, Obs, Begbilar

SearchEngine söker med A* (kortaste vägen)...

Expanderade noder: Flygplatsen, TreKorvar, On, Rondellen, Kyrkan, Station, Flygplatsen, NUS, Gamlia, Obs, klar.

Flygplatsen, TreKorvar, Rondellen, Kyrkan, Station, Obs, Begbilar

söker med A* (snabbaste vägen)...

Expanderade noder: Flygplatsen, TreKorvar, Flygplatsen, On, Rondellen, Kyrkan, NUS, Sofiehem, Alidhem, MIT, Nydala, Mariehem, Foa, klar.

Flygplatsen, TreKorvar, Rondellen, Kyrkan, NUS, Sofiehem, Alidhem, MIT, Nydala, Mariehem, Foa, Begbilar

SearchEngine söker med Greedy Search...

Expanderade noder: Flygplatsen, TreKorvar, On, TreKorvar, Rondellen, Kyrkan, Station, Obs, klar.

Flygplatsen, TreKorvar, Rondellen, Kyrkan, Station, Obs, Begbilar

MIT – Flygplatsen

SearchEngine söker med bredden först...

Expanderar:

MIT, Alidhem, Berghem, Nydala, Sofiehem, Ok, Gamlia, Mariehem, GimonAs, NUS, Station, Foa, Kyrkan, I20, Obs, Begbilar, Rondellen, Tegsbron, Teg, TreKorvar, On, Flygplatsen klar.

Resa: MIT, Alidhem, Sofiehem, NUS, Kyrkan, Rondellen, TreKorvar, Flygplatsen

SearchEngine söker med djupet först...

Expanderar: MIT, Alidhem, Sofiehem, GimonAs, Ok, Nydala, Mariehem, Berghem, Gamlia, Station, I20, Tegsbron, Teg, Rondellen, Kyrkan, NUS, TreKorvar, On, klar.

Resa: MIT, Alidhem, Sofiehem, GimonAs, Ok, Nydala, Mariehem, Berghem, Gamlia, Station, I20, Tegsbron, Teg, Rondellen, TreKorvar, Flygplatsen

SearchEngine söker med A* (kortaste vägen)...

Expanderade noder: MIT, Alidhem, Sofiehem, Berghem, Gamlia, MIT, GimonAs, NUS, Nydala, Ok, Station, Kyrkan, Rondellen, TreKorvar, klar.

MIT, Alidhem, Sofiehem, NUS, Kyrkan, Rondellen, TreKorvar, Flygplatsen söker med A* (snabbaste vägen)...

Expanderade noder: MIT, Nydala, Ok, Mariehem, Foa, Begbilar, Obs, I20, Station, Gamlia, NUS, Sofiehem, GimonAs, Kyrkan, Rondellen, TreKorvar, klar.

MIT, Nydala, Mariehem, Foa, Obs, I20, Station, Gamlia, NUS, Kyrkan, Rondellen, TreKorvar, Flygplatsen

SearchEngine söker med Greedy Search...

Expanderade noder: MIT, Alidhem, Sofiehem, GimonAs, Ok, Nydala, Mariehem, Berghem, Gamlia, NUS, Kyrkan, Rondellen, TreKorvar, klar.

MIT, Alidhem, Sofiehem, GimonAs, Ok, Nydala, Mariehem, Berghem, Gamlia, NUS, Kyrkan, Rondellen, TreKorvar, Flygplatsen

Nydala - Gamlia

SearchEngine söker med bredden först...

Expanderar:

Nydala, MIT, Mariehem, Ok, Alidhem, Berghem, Foa, Sofiehem, Gamlia klar.

Resa: Nydala, MIT, Berghem, Gamlia

SearchEngine söker med djupet först...

Expanderar: Nydala, MIT, Alidhem, Sofiehem, GimonAs, Ok, NUS, Kyrkan, Rondellen, Teg, Tegsbron, I20, Obs, Begbilar, Foa, Mariehem, Berghem, klar.

Resa: Nydala, MIT, Alidhem, Sofiehem, NUS, Kyrkan, Rondellen, Teg, Tegsbron, I20, Obs, Begbilar, Foa, Mariehem, Berghem, Gamlia

SearchEngine söker med A* (kortaste vägen)...

Expanderade noder: Nydala, MIT, Berghem, klar.

Nydala, MIT, Berghem, Gamlia

söker med A* (snabbaste vägen)...

Expanderade noder: Nydala, Ok, Nydala, Alidhem, Sofiehem, NUS, klar.

Nydala, Ok, Alidhem, Sofiehem, NUS, Gamlia

SearchEngine söker med Greedy Search...

Expanderade noder: Nydala, MIT, Berghem, klar.

Nydala, MIT, Berghem, Gamlia

Teg - Foa

SearchEngine söker med bredden först...

Expanderar:

Teg, Rondellen, Tegsbron, Kyrkan, TreKorvar, I20, Station, NUS, On, Flygplatsen, Obs, Gamlia, Sofiehem, Begbilar, Foa klar.

Resa: Teg, Tegsbron, I20, Obs, Foa

SearchEngine söker med djupet först...

Expanderar: Teg, Rondellen, Kyrkan, Station, I20, Tegsbron, Obs, Begbilar, klar.

Resa: Teg, Rondellen, Kyrkan, Station, I20, Obs, Begbilar, Foa

SearchEngine söker med A* (kortaste vägen)...

Expanderade noder: Teg, Rondellen, Kyrkan, Station, Tegsbron, NUS, Gamlia, Berghem, TreKorvar, On, Obs, klar.

Teg, Rondellen, Kyrkan, Station, Obs, Foa

söker med A* (snabbaste vägen)...

Expanderade noder: Teg, Tegsbron, I20, Station, Gamlia, NUS, Sofiehem, Alidhem, MIT, Nydala, Mariehem, klar.

Teg, Tegsbron, I20, Station, Gamlia, NUS, Sofiehem, Alidhem, MIT, Nydala, Mariehem, Foa

SearchEngine söker med Greedy Search...

Expanderade noder: Teg, Tegsbron, I20, Obs, klar.

Teg, Tegsbron, I20, Obs, Foa