Linköpings Tekniska Högskola
Matematiska institutionen/Optimeringslära
Kaj Holmberg

2019–11–27

Laborationsinformation

TAOP33 Kombinatorisk optimering, HT19

Projekt 4: Lösning av snöröjningsproblemet

1 Uppgift

Uppgiften i projekt 4 är att lösa (en förenklad variant av) snöröjningsproblemet. Uppdelningen av uppgifter mellan fordon ska göras för hand och/eller med heuristiker, implementerade i koden SNOWPLAN. Kinesiska brevbärarproblem och lantbrevbärarproblem löses med en kod i Vineopt.

Fiktivt scenario: Ni är inhyrda av en svensk kommun för att planera snöröjningen. En konsultfirma har levererat ett program, SNOWPLAN, som påstås göra en bra planering. Er uppgift är att utvärdera SNOWPLAN, samt att ta fram bra planer för snöröjningen. Man kan både tänka sig att låta programmen konkurrera och att låta dem samarbeta, genom att låta ett program förbättra lösningen från det andra, i båda riktningarna. Vad kommunen är intresserad av är följande:

- Bra planer för snöröjningen samt jämförelser avseende de olika vikterna på tid.
- En jämförelse mellan SNOWPLAN och vad man kan få fram för hand med Vineopt.
- Diskussion om brister/potential hos SNOWPLAN.
- Diskussion om möjligheterna att kombinera Vineopt och SNOWPLAN.
- Ett anbud på snöröjningen i Vadstena.

2 Att göra

1. Studera beskrivningen av snöröjningsproblemet.
2. Studera beskrivningen av SNOWPLAN.
3. Lär känna relevanta delar av Vineopt.
4. Läs på om lokalsökning och metaheuristiker, t.ex. simulerad kylning.
5. Lös problemen som beskrives senare.
6. Redovisa genom att skriva en rapport om resultaten som svarar på alla frågor och kan läsas av både en som inte kan optimering och en som kan det och är nyfiken på hur ni gjorde. Inkludera gärna bilder på bra uppdelningar mellan fordon.
7. Ev: Förbered en redovisning av resultaten/rapporten.

3 Brevbärarproblem

Det kinesiska brevbärarproblemet beskrives i Holmberg (2010/2018), kapitel 10.6. Där ges en matematisk modell samt en optimerande metod för problemet. (Denna metod finns implementerad i Vineopt.) Problemet handlar helt enkelt om att finna billigaste sättet att gå i varje båge i grafen minst en gång.

Lantbrevbärarproblemet beskrives också kort i samma kapitel. I detta problem behöver inte alla bågar genomgås. Istället finns en given delmängd av bågar som ska gås igenom minst en gång. Om denna mängd med "nödvändiga" bågar är sammanhängande, kan lantbrevbärarproblemet lösas till optimalitet på samma sätt som det kinesiska brevbärarproblemet. (Man fyller helt enkelt på grafen på billigaste sätt, så att alla noder får jämn valens.)

Om de "nödvändiga" bågarna inte bildar en sammanhängande mängd är problemet svårt att lösa. Dock finns relativt effektiva heuristiker som oftast finner ganska bra lösningar. (En av dem är implementerad i Vineopt.)

Om man har flera brevbärare som delar på arbetet, blir problemet svårt. Man måste då först bestämma vilka bågar varje brevbärare ska ta. När man har gjort det, kan man lösa ett lantbrevbärarproblem för varje brevbärare. Frågan är hur man bestämmer fördelningen av bågarna på brevbärarna.

En möjlighet är att göra detta för hand i Vineopt. För ett verkligt problem innehåller bilden av nätverket mycket information. Det är ofta ganska lätt att göra en ganska vettig uppdelning av grafen i rätt antal delar genom att betrakta nätverket. (Det är så snöröjning planeras idag.)

Ett annat alternativ är att använda lokalsökning eller en metaheuristik, såsom simulerad kylning. Man bestämmer alltså en uppdelning av bågarna på brevbärarna, och evaluerar lösningen genom att lösa ett lantbrevbärarproblem för varje brevbärare och summera ihop kostnaderna.

Det som krävs för ett lyckat resultat är att definiera en lämplig omgivning för heuristiken. Det enklaste är att flytta en enstaka båge från en brevbärare till en annan. Man kan även tänka sig att byta bågar, om man inte vill ändra antalet bågar per brevbärare. Man kan även tänka sig mer avancerade byten, t.ex. genom om flytta hela understrukturer. En bra deltur (t.ex. en rundtur) kan flyttas i sin helhet mellan brevbärare. Det krångliga då är att finna dessa strukturer.

Man kan också tänka sig att starta med konstruktiva heuristiker som försöker bygga upp en tilldelning på ett smart sätt.

4 Snöröjning i städer

En generell beskrivning av snöröjningsproblemet ges separat. I detta projekt ska vi behandla ett förenklat problem, nämligen där man inte beaktar att det tar längre tid att vända än att köra rakt fram. Dessutom beaktas bara normala gator, ej gångvägar eller cykelvägar. Till sist bortser vi från detaljer avseende röjning av vändplatser och korsningar.

Vi räknar med en medelhastighet av 7.2 km/h. Tiden det tar att röja en gata uppskattas i medel till $p = 2$ gånger längden dividerat med hastigheten. Denna tid inkluderar två eller tre drag (ev. mitten, höger sida, vänster sida) samt avslutande röjning vid vändplats och/eller korsning. Allt detta ses alltså som en operation, som startar i ena änden av gatan och slutar i andra. Dessa siffror ger att det tar precis en sekund per meter, dvs. det tar l_j sekunder att röja gata j om den har längden l_j. Om fordonet kör gatan utan att röja (bara för att förflytta sig) blir $p = 1$, så tiden blir hälften så stor.

Dessa antaganden leder till att problemet är ett kinesiskt brevbärarproblem, om man bara har ett fordon som ska göra allt.

Uppgiften handlar om att jämföra resultatet av att använda olika antal (identiska) fordon. Självklart går det snabbare att låta flera fordon arbeta parallellt, men det kostar att ha flera fordon. Varje fordon som används ger en fast kostnad (som alltså inte beror på hur mycket man använder det, utan bara på om fordonent finns tillgängligt för röjning). Vi beaktar inte startpunkt, utan antar att fordonen kan transportera sig till valfri punkt innan vår planering startar.

Vilken rundtur som är bäst påverkas inte av att det går fortare att köra en redan röjd gata än att röja. Eftersom alla sträckor som jämförs i optimeringen körs lika fort, förändras inte den optimala lösningen av detta. Dock förändras målfunktionsvärdet.

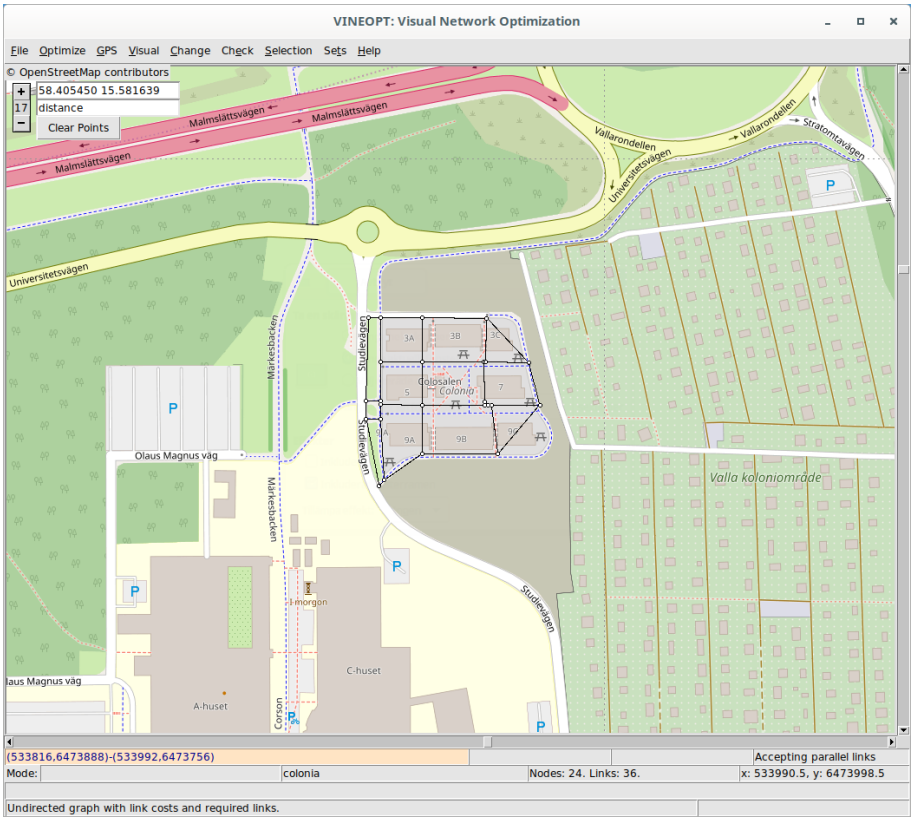
Man löser ett lantbrevbärarproblem för varje fordon. Totalkostnaden för detta blir summan av kostnaderna för turerna för varje fordon, plus en fast kostnad för varje fordon. Vi antar att den fasta kostnaden för varje fordon är 200, och att varje sekund kostar 1. Totalkostnaden för en lösning är alltså summan av kostnaderna som fås från lantbrevbärarproblemet för varje fordon plus antal fordon gånger 200.

Tiden för lösningen är lika med den maximala tiden (kostnaden) för något fordon. Vilken lösning som är bäst beror då på hur viktigt man tycker att tiden är jämfört med kostnaden. Många fordon ger hög kostnad men kort tid, medan få fordon ger lägre kostnad men längre tid. I detta projekt tittar vi på två scenarior, ett där kostnad och tid är värda lika mycket, dvs. vårt mått är totalkostnaden plus tiden, och ett där tiden är dubbelt så viktig som kostnaden, dvs. måttet är totalkostnaden plus två gånger tiden.

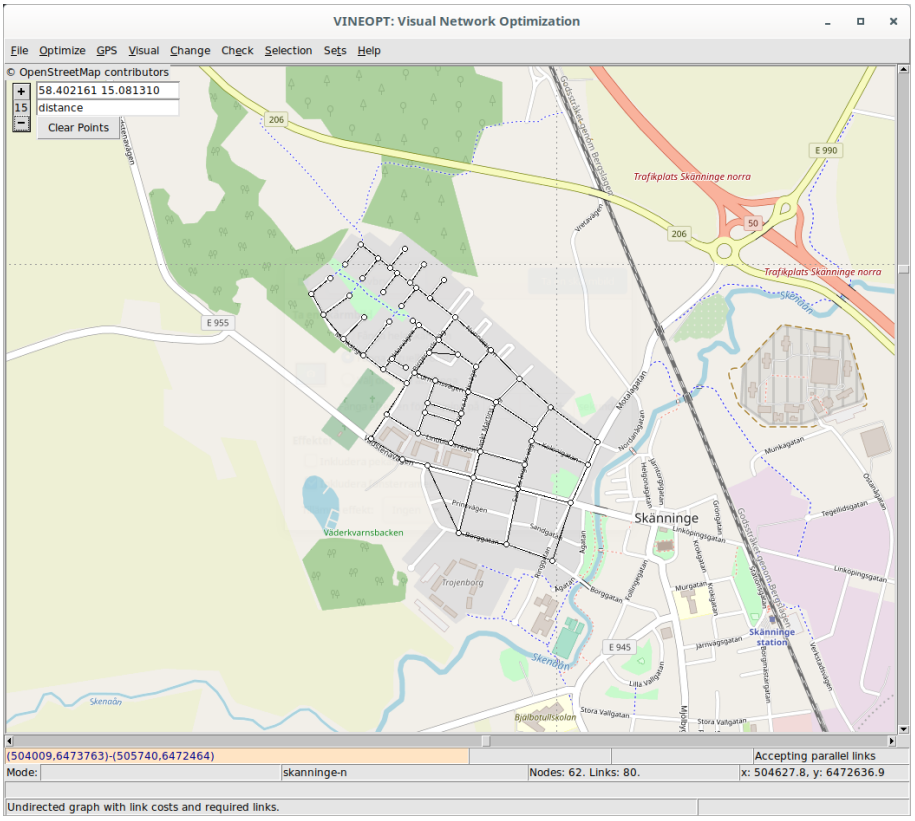
4.1 Indata

Gatunätverken finns tillgängliga i Vineopt-format (som lantbrevbärarproblem, oriktad graf med bågkostnader och möjlighet att definiera "nödvändiga" vågar). Bågkostnaderna är lika med vägsträckornas längd i meter.

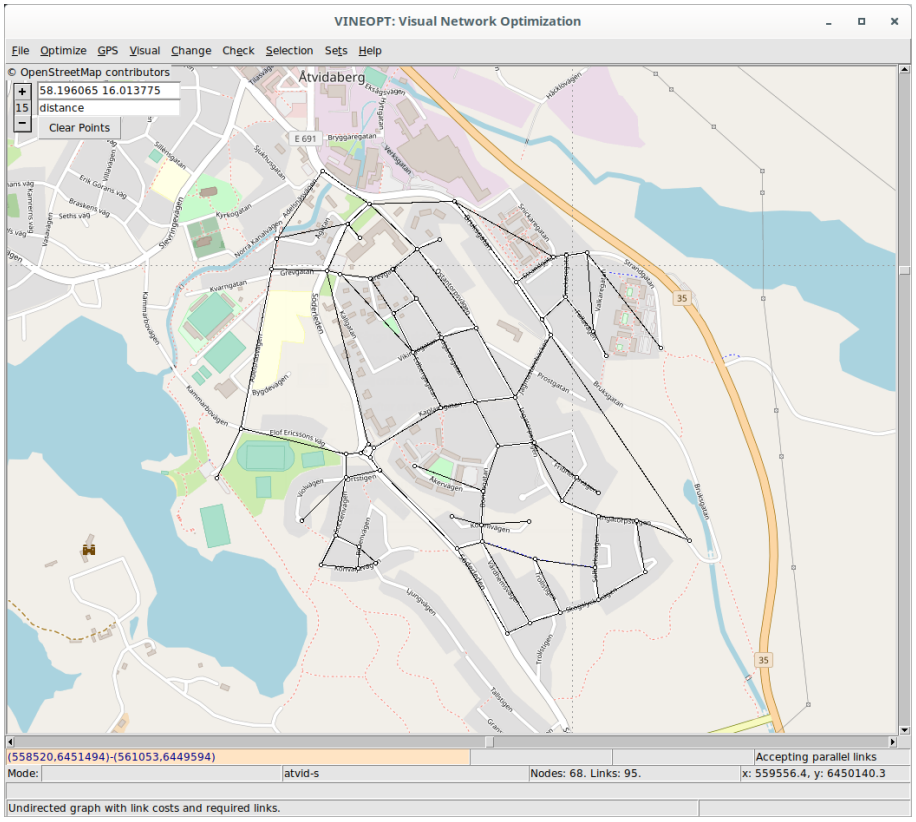
Det finns ett litet nätverk, *colonia*, (studentbostäder vid märkesbacken) med 24 noder och 36 länkar, två melllanstora, *skanninge-n* (Skänninge norra), med 62 noder och 80 länkar och *atvid-s* (Åtvidaberg södra), med 68 noder och 95 länkar, samt två större, *studentryd* (den del av Ryd där studenter bor), med 387 noder och 519 länkar, samt *vadstena* (liten stad med stort slott vid stor sjö), med 581 noder och 778 länkar.



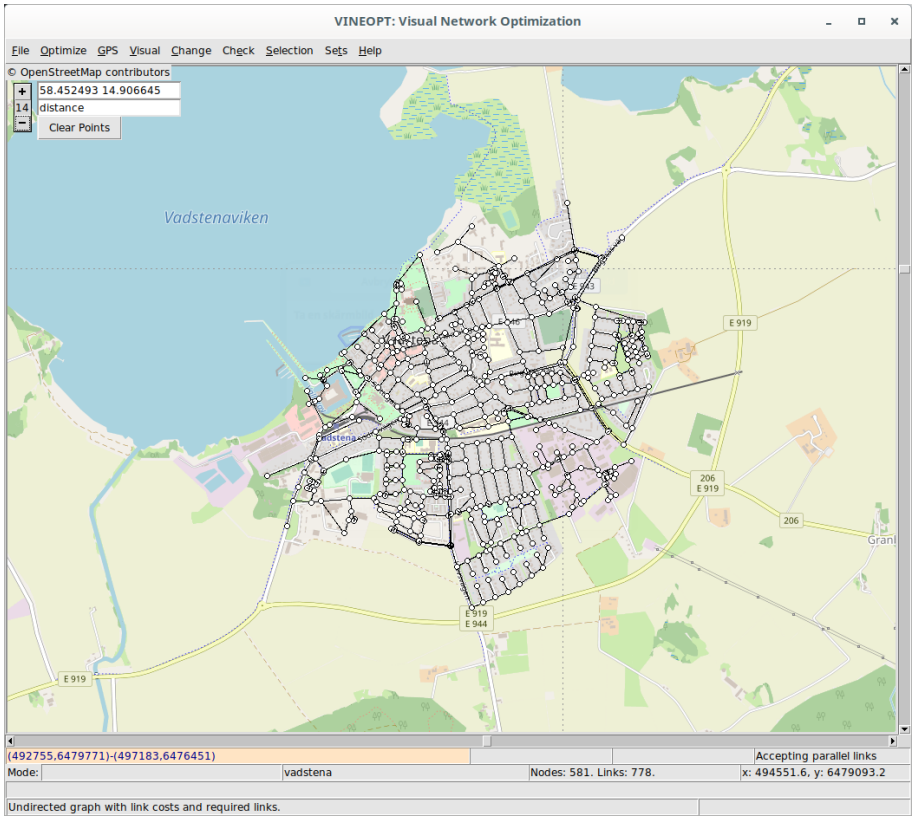
Figur 1: Colonia



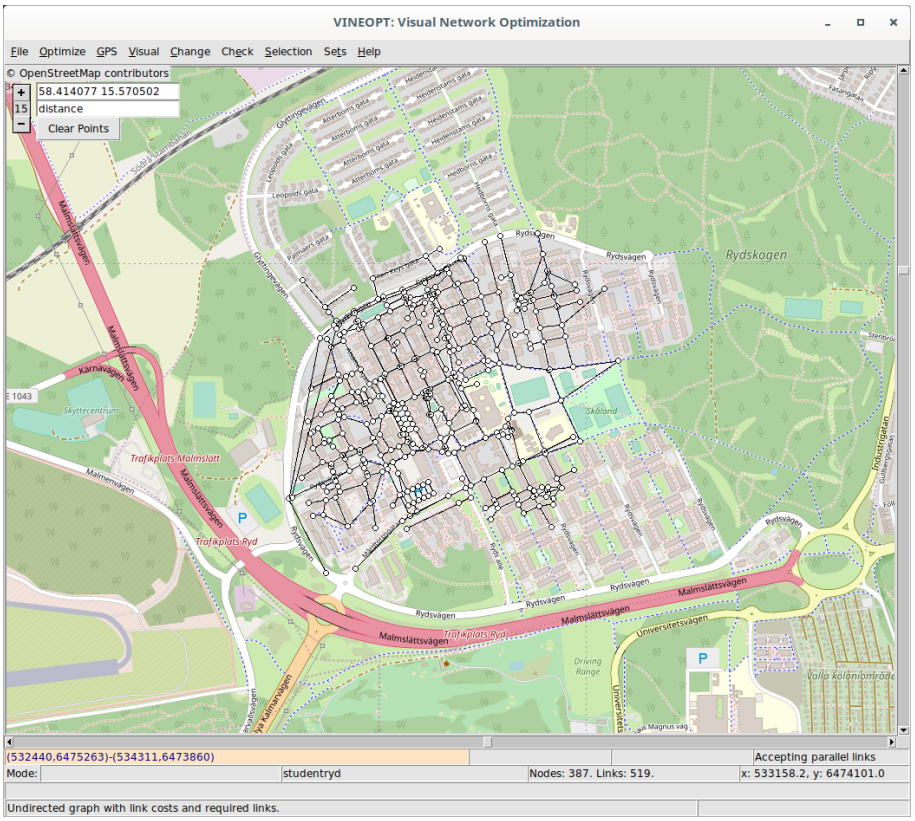
Figur 2: Skänninge Norra



Figur 3: Åtvidaberg Södra



Figur 4: Vadstena



Figur 5: Studentryd

4.2 Hjälpmedel

Vineopt startas genom att öppna ett terminalfönster och skriva `/courses/TAOP88/dine 5`, vilket ger en utökad version som behövs för detta projekt.

Om ett fordon ska göra allt, kan man direkt låta Vineopt finna optimal brevbärartur. Om arbetet ska delas upp mellan flera fordon, får man (för hand eller på annat sätt) bestämma vilka gator som varje fordon ska röja. Vineopt har verktyg för att definiera nod- och bågmängder. Under menyn `Select` finns flera olika sätt att välja ut mängder, t.ex. med en rektangulär form. Alternativet `toggle` betyder att för alla markerade enheter byts ovalda mot valda och valda mot ovalda.

De valda bågarna kan sedan göras till en bågmängd ("linkset"). Därefter väljer man en annan uppsättning bågar, och gör den till nästa bågmängd. Detta upprepas så att man får en bågmängd för varje fordon. (Se till att ingen länk glöms bort, utan att varje båge ingår i exakt en mängd.)

När man är färdig kan man se alla bågmängderna (i olika färger), eller markera dem en i taget. De markerade bågarna i grafen kan därefter användas som "nödvändiga" bågar (required arcs) i ett lantbrevbärarproblem. Man får alltså lösa ett sådant problem för varje fordon. Detta arbetssätt är dock ganska tidskrävande för större antal fordon.

Observera att faktorn p sätts i Vineopt med "Change req link factor" i menyn "Change". Alternativt kan man justera tiden för hand på följande sätt. Om v_1 är summan av kostnaden för alla nödvändiga bågar ("Calc req link cost") och v_2 totala kostnaden Vineopt rapporterar

(för $p = 1$), så är $v_3 = v_2 - v_1$ kostnaden för körning utan röjning. Den korrekta kostnaden blir då $v = v_1 + v_3/2 = v_1 + (v_2 - v_1)/2 = (v_1 + v_2)/2$.

Vineopt kan läsa och skriva bågmängder på fil, så man kan skapa en bra uppdelning och spara den för att sedan använda den som startlösning i SNOWPLAN. Man kan även läsa in en uppdelning som skapats i SNOWPLAN för se hur den ser ut och även kanske förbättra den manuellt.

Ni får tillgång till en implementering av SNOWPLAN i Python, och med den en föreslagen uppsättning värden på parametrarna. De kan dock behöva trimmas lite för vissa problem. Viktiga parametrar är bl.a.
1. Hur ska startlösningen konstrueras?
2. Hur ska ordning och allokering ändras mellan iterationerna?
3. Kylningsparametrarna. Hur snabbt ska temperaturen sänkas?
4. Vilken typ av förbättringar av lösningen ska man göra?

Alla detaljer återfinnes i dokumentationen om SNOWPLAN.

Det rekommenderas att man skapar en ny katalog och kör både Vineopt och SNOWPLAN där, och att sparar har alla problemfiler där. SNOWPLAN behöver filen `vno_dir.ini` i den katalog man kör. Den skapas automatiskt när man kör Vineopt (om den inte finns). (Behåll inga filer från förra kursen i den katalogen.)

SNOWPLAN har möjligheten att köra igång Vineopt med aktuella bågmängder. När man gjort ändringar i Vineopt, sparar man bågmängderna ("Save link sets for Snowplan"), avslutar Vineopt och fortsätter med SNOWPLAN. Man läser först in bågmängderna från Vineopt och använder sedan startvar=0. Notera dock att Snowplan ev. gör flera försök att förbättra lösningen, och vissa av dessa ändringar medför inte ändring av allokeringen. Därför kan samma allokering ge sämre lösning i Vineopt än i Snowplan.

Er uppgift är att få SNOWPLAN att fungera så bra som möjligt, vilket kan innebära justeringar av vissa parametrar. Det kan vara intressant att dels testa med en dålig startlösning för se hur snabbt lösningarna förbättras, och dels testa med en bra startlösning för att se om den överhuvudtaget kan förbättras. Obsevera att SNOWPLAN måste köras i ett visst antal iterationer för att ha en chans att hitta bra lösningar, även om det tar tid.

5 Deluppgifter

1. Betrakta det lilla testproblemet *colonia*.

   a) Lös problemet för ett fordon. Notera lösningen, totalkostnaden samt vilken tid röjningen tar.

   b) Finn en lösning för två fordon genom att för hand i Vineopt ange vilka länkar som ska åtgärdas av varje fordon. Försök att hitta en bra uppdelning. Notera bästa lösningen, totalkostnaden samt vilken tid röjningen tar.

   c) Finn en lösning för tre fordon på samma sätt som i upppgift 1b.

   d) Antag att man ger lika vikt vid tid och kostnad. Vilken lösning är bäst?

   e) Antag att tiden är dubbelt så viktig som kostnaden, dvs. tiden får vikten två och kostnaden vikten ett. Vilken lösning är då bäst?

2. Betrakta problemet *colonia*.

   a) Lös problemet för två fordon med heuristiken SNOWPLAN. Jämför resultatet med uppgift 1b.

   b) Lös problemet för tre fordon med SNOWPLAN. Jämför med uppgift 1c.

   c) Besvara frågorna 1d och 1e för dessa lösningar.

3. Betrakta det lite större problemet *skanninge-n* (Skänninge norra). Gör samma sak som i uppgift 1 och 2.

4. Betrakta problemet *atvid-s* (Åtvidaberg södra). Finn lösningar för 1, 2, 3 och 4 fordon med SNOWPLAN. Notera målfunktionsvärde, totalkostnad och tid, samt lösningstid för Snowplan. Finn bästa antal fordon med lika vikt vid tid och kostnad, samt med tiden dubbelt så viktig som kostnaden. Studera den bästa lösningen/uppdelningen i Vineopt.

5. Betrakta det största problemet *vadstena*. Målet är att finna en mycket bra lösning för 6 fordon. Använd lika vikt på kostnad och tid. Kör först SNOWPLAN. Notera hur lång tid de olika fordonen tar, speciellt vilket fordon som tar längst tid. Läs sedan in lösningen i Vineopt, och försök förbättra den genom att flytta bågar från en mängd till en annan. (Observera att man kan illustrera alla bågmängder samtidigt i olika färger samt få mängdtillhörighet utskriven vid bågarna i grafen.) Gör några troliga förbättringar och spara dessa bågmängder på fil.

   Läs därefter in dem i SNOWPLAN. Om man bara gör en iteration, får man direkt veta målfunktionsvärdet, och kan se om man lyckats förbättra lösningen. Man kan även köra fler iterationer i SNOWPLAN, för att se om lösningen kan förbättras ytterligare.

   Välj den bästa av lösningarna och gör ytterligare försök att förbättra lösningen i Vineopt följt av SNOWPLAN.

   Lämna därefter in ert anbud, vilket består av målfunktionsvärde, tid för varje fordon, samt en fil med fordonsallokeringar (bågmängder). Lägst bud vinner, och vinnaren kommer att föräras ett diplom (samt det fiktiva uppdraget att snöröja Vadstena).

6. Frivillig uppgift: Gör något kul med problemet *studentryd*.