INF-MAT-5380

http://www.uio.no/studier/emner/matnat/ifi/INF-MAT5380/

Leksjon 5

Leksjon 4 - Oversikt

- Tabusøk

Tabusøk - Sammendrag

- Inspirert fra matematisk optimering og kunstig intelligens/kognitiv vitenskap
- Fokus på bruk av minne fremfor bruk av tilfeldighet
- Basert på “aggressivt” lokalsøk, aksepterer flytt til dårligere løsning
- Kandidatliste-strategier etc. for billigere evaluering av nabolag
- Bruk av korttidsminne
  - attributter
  - unngå reversering og repetisjon
  - tabukriterier, tabuliste
- Aspirasjonskriterier - tabuer er til for å brytes ...
- Diversifisering og intensifisering
- Path relinking
- Strategisk oscillering

- Gode resultater for mange, harde DOP

Leksjon 5 - Oversikt

- Styrt lokalsøk (Guided Local Search, GLS)
  - Martin Stølevik, SINTEF

Guided Local Search (GLS) - Bakgrunn

- GENET (neural network)
  - Prosjekt for løsing av 'Constraint Satisfaction' Problemer (CSP) Fra tidlig på 90-tallet
  - Tsang, Voudouris m.fl. (Davenport)
- Edward Tsang og Chris Voudouris 1994
  - videreutvikling av GENET fra CSP til 'Partial CSP' (nesten-løsninger når problemet er overbeskranket): satisfaction $\Rightarrow$ optimering
  - ledet til 'the Tunnelling Algorithm' (94) som igjen ledet til GLS (95)

Hovedidé

- Generell strategi, metaheuristikk, for å styre nabolagssøk/"indre" heuristikker
- Straffer uønskede egenskaper ('features') i løsninger
- Fokuserer på lovende deler av søkerommet
- Søker å unngå lokal optima ved å jobbe med en utvidet objektivfunksjon (original + straffe-ledd)
- Kjører lokalsøk til (lokalt) minimum, straffer egenskaper, lokalsøk til minimum, straffer, ….

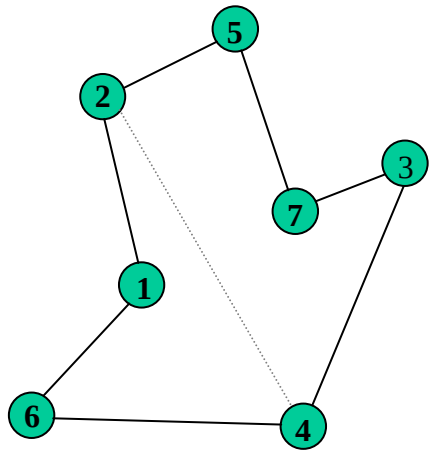
Egenskaper

- GLS fokuserer på karakteristiske (ikke-trivielle) egenskaper (features) ved løsninger
- Problemavhengige
- Egenskaper har en *kostnad.* Kostbare egenskaper prøver en å unngå.
  - Representerer direkte eller indirekte innvirkning fra en løsning på (utvidet) objektivfunksjon
  - Konstant eller variabel
- Indikatorfunksjon:

$$I_i(s) = \begin{cases} 1 \text{ , hvis løsning } s \text{ har feature } i \\ 0 \text{ , hvis løsning } s \text{ ikke har feature } i \end{cases} \quad , s \in S$$

Egenskap - eksempel: TSP

- En løsning (tur) er et antall kanter
- En kant er godt valg som egenskap:
  - Er enten med i en løsning eller ikke
  - egenskap-kostnad = kantlengden
- La f.eks. settet av alle kanter e_{ij} være egenskapene:



$$\mathbf{E}=\{e_{ij}\}, i=1...N,\ j=i+1...N, i \neq j$$

- Kostnad for en egenskap e_{ij} er gitt av d_{ij} i distanse-matrisen:

$$\mathbf{C} = [d_{ij}], i=1...N, j=1...N$$

Utvidet objektivfunksjon

- Sett av egenskaper E=*{ e_i }, i*=1…*G*
- Indikatorfunksjon:

$$I_i(s) = \begin{cases} 1 \text{ , hvis løsning } s \text{ inneholder e}_i \\ 0 \text{ , hvis løsning } s \text{ ikke inneholder e}_i \end{cases} , s \in S$$

- Straffevektor **p**=*[p_i], i*=1…*G*, antall ganger egenskap e_i er straffet til nå
- Straffefaktor eller reguleringsparameter *λ*

$$f'(s) = f(s) + \lambda \sum_{i=1}^{G} I_i(s) p_i$$

- Kostnadsvektor **c**=*[c_i], c_i > 0, i*=1…*G,* gir kostnad for e_i

Utvidet objektivfunksjon - kommentarer

$$f'(s) = f(s) + \lambda \sum_{i=1}^{G} I_i(s) p_i$$

- Reguleringsparameteren λ gir straffens innflytelse på den utvidede objektivfunksjonen.
  - Lav verdi: intensifisering
  - Høy verdi: diversifisering
- Initielt er p_i=0 $\forall$ i
- I lokale minima velges den (de) e_i som har høyest verdi ('utility'), $u_i(s_{min}, e_i) = I_i * c_i/(1+p_i)$. Denne egenskapen straffes ved å sette $p_i = p_i+1$.
  - ⇨ Diversifisering: ulike egenskaper straffes
  - ⇨ egenskaper med høy kostnad straffes oftere enn de med lav kostnad
- Merk: Bare egenskaper i lokale minima, straffes

Search_the_Neighborhood (current,N,f,strategy,incumbent)

```
best_neighbor:=current
neighbors=N(current)
for i in neighbors do
      if f(i) < f(best_neighbor) then best_neighbor:=i
       if f(i) < f(incumbent) then
           if strategy=”First Accept” then
             return (i,i,false) else
             incumbent:=i                  */ strategy is ”Best Accept”
           fi
      fi
od
return (best_neighbor,incumbent,best_neighbor=current)
```

Local_Search (S,f,N,strategy)

```
i=Init_Solution(S)
incumbent:=i                    */ best solution until now
local_optimum:=false
while not local_optimum do
    (i,incumbent,local_optimum):=
       Search_the_Neighborhood
       (i,N(i),f,strategy,incumbent)
    if local_optimum return incumbent
od
```

GLS(S, f, λ, I, c, G, stopCriterion, N)

```
{
    int k := 0; // number of GLS-iterations
     s*  := s_k := InitialSolution(S); // get initial solution
    set all p_i := 0; // set all penalties to zero
    while (stopCriterion not satisfied) do {
        f' = f + λ*∑(I_i *p_i);
        s_k+1 = Local_Search (s_k, f', N, "best accept"); // local minimum
        for (i=1 to G)
            u_i = I_i(s_k+1)* c_i /(1+p_i);
        for each i such that u_i is maximum do
             p_i := p_i + 1;
        k = k+1;
        if (f(s_k+1) < f(s*))
             s* = s_k+1; // save best solution found until now
    }
    return s*;
}
```

Kommentarer

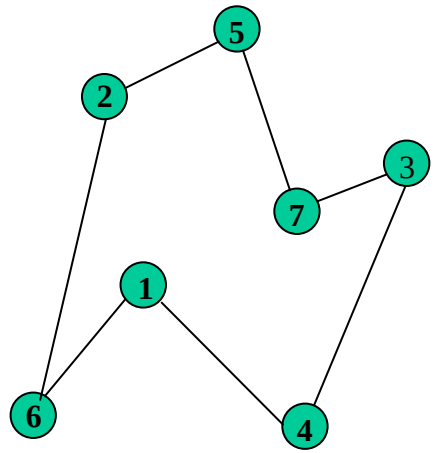
- Bruker utvidet objektivfunksjon i lokalsøket:
  - ⇨ s_{k+1} := Local_Search (s_k, f', **N**, "best accept");
- Lokalsøk-strategien kan også være "first accept", ..
- Funksjonen *Local_Search* kan velges for å løse problemet best mulig.
- Benytter den vanlige objektivfunksjonen til å finne beste løsning (naturlig nok…!):
  - ⇨ if ($f(s_{k+1}) < f(s^*)$)…
- Delta-evaluering av flytt må ta hensyn til egenskaper som endrer status
- Hvis alle egenskaper er straffet like mye, beskriver f'(s) samme "landskap" som f(s)

Verdier på λ

- Kan være problematisk å finne verdier for λ selv om metoden er robust for dens verdi
- Generelt: brøkdel av lokalt minimum.
- Tsang og Voudouris:
  - TSP : $\lambda = a^{*}f(s_{min})/n$, $a \in [0,1]$ er problemavhengig
  - QAP: $\lambda = a^{*}f(s_{min})/n^2$, $a \in [0.2,1]$ er problemavhengig
  - For andre problemer rapporterer de absolutte verdier avhengig av problemstørrelse.
  - Eks: Minimering av brudd på føringer (PCSP): λ = straffen for å bryte en føring.

GLS - eksempel : TSP

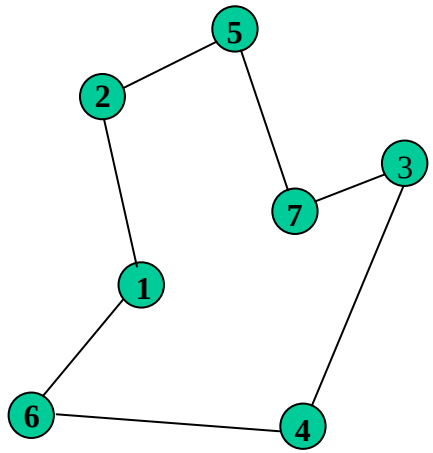
- Egenskap: kant
- Kostnad på egenskap: lengden
- Egenskap en assosiert med e_{26} vil bli straffet i løsningen til høyre:
- I neste runde med lokalsøk er objektivverdien som før (*f(s)*) bortsett fra hvis e_{26} er med, da er verdien $f(s)+\lambda$



	1	2	3	4	5	6	7
1		0	0	0	0	0	0
2			0	0	0	1	0
3				0	0	0	0
4					0	0	0
5						0	0
6							0

GLS - eksempel : TSP

- Etter neste lokalsøk blir e_{34} straffet.
- Deretter er objektivverdien som før (*f(s)*) bortsett fra hvis e_{26} eller e_{34} er med



	1	2	3	4	5	6	7
1		0	0	0	0	0	0
2			0	0	0	1	0
3				1	0	0	0
4					0	0	0
5						0	0
6							0

$$f'(s) = \begin{cases} f(s) & \text{, både } e_{26} \text{ og } e_{34} \notin s \\ f(s)+\lambda & \text{, } e_{26} \text{ eller } e_{34} \in s \\ f(s)+2\lambda & \text{, både } e_{26} \text{ og } e_{34} \in s \end{cases}$$

GLS vs. SA

- I SA er det vanskelig å finne riktig nedkjølingsskjema (problem-avhengig)
  - Høy temperatur gir dårlige løsninger
  - Lav temperatur gir konvergens til lokalt minimum
  - SA er ikke-deterministisk
- GLS besøker lokale minima, men kan komme unna.
  - Ikke tilfeldige ”oppover”-flytt som i SA
  - GLS er deterministisk
  - Konvergerer ikke i lokalt minimum; straff tillegges helt til man unnslipper

Minne - GLS vs. Tabusøk

- I tabusøk benyttes frekvensbasert (langtids)minne til å straffe egenskaper som opptrer ofte.
- GLS bruker sitt minne (p_i) gjennom hele søket, ikke bare i egne faser som i tabusøk.
- GLS straffer på bakgrunn av både kostnad og frekvens av en løsning.
- Tabusøk straffer bare på bakgrunn av frekvens og kan dermed søke å fjerne "gode" egenskaper.
- GLS unngår dette ved å benytte domenekunnskap (c_i)
- I GLS synker sannsynligheten for senere å bli straffet jo mer en egenskap er blitt straffet ($c_i/[1+p_i]$)

GLS vs. Tabusøk

	Tabusøk	**GLS**
Informasjon som brukes	Modifisert nabolag	Modifisert objektivfunksjon
Minne	Tabu liste, frekvensbasert minne	Straff av egenskaper
Når?	Hver iterasjon eller hver N'te iterasjon.	I lokalt minimum i utvidet objektivfunksjon
Søkets "natur"	- Unngå stopp i lokale minima og reverserende flytt - Diversifisering; straffe flytt som gjøres ofte eller attributter som opptrer ofte i løsninger	- Unnslippe lokale minima - Distribuere søke-innsatsen avhengig av kostnad på egenskaper
Intensifisering / diversifisering	< >	Parameteren λ
Reduksjon av nabolag	Kandidatliste	Fast Local Search - FLS

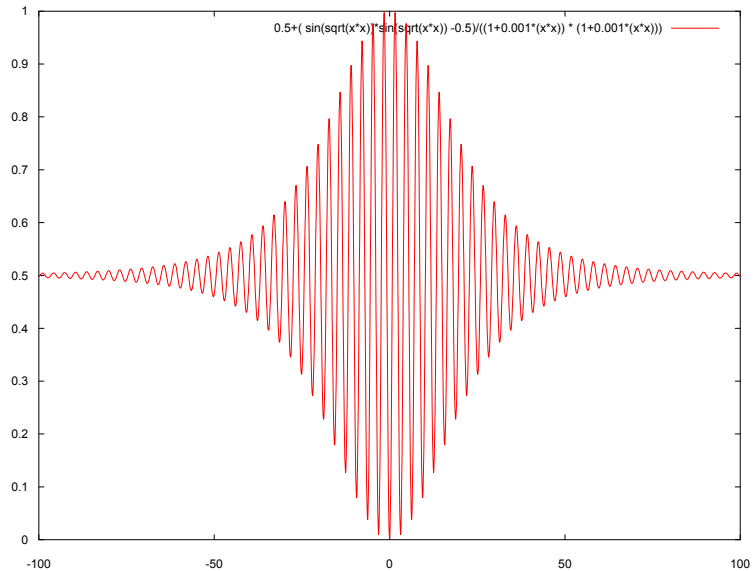
Fast Local Search

- Begrensning av søkerommet
- Ligner på kandidatliste-strategien i Tabusøk
- Idé
  - oppdeling av nabolaget (eg. søkerommet) i sub-nabolag
  - Status sub-nabolag: aktivt eller ikke aktivt (bit: ‘1’/’0’)
  - Søk bare i de aktive sub-nabolagene
  - Assosiere egenskaper til sub-nabolag
    - egenskap $\Leftrightarrow$ nabolag som endrer status for denne egenskapen
  - Ved flytt beholder man bit’ene og søker videre i aktive sub-nabolag
- Kan brukes for alle lokalsøk-algoritmer, men egner seg spesielt bra kombinert med GLS

GLS - bruksområder

- Workforce scheduling at BT (Tsang, Voudouris 97)
- TSP (Voudouris, Tsang 98)
- VRP (Kilby et. al. 97)
- Funksjonsoptimering (Voudouris 98)
- FAP (Voudouris, Tsang 98)

Muligheter og utvidelser



- Begrenset varighet på straff
- Avtakende straff
- Belønning, i tillegg til straff
- Automatisk regulering av λ
- Nye utility-funksjoner for å finne egenskaper å straffe

- Har vært benyttet til funksjonsoptimering, gode resultater på:

$$F6(x,y) = 0.5 + \frac{\sin^2\sqrt{x^2+y^2} - 0.5}{\left[1+0.001(x^2+y^2)\right]^2}$$

Leksjon 5 - Oppsummering

- Styrt lokalsøk (Guided Local Search, GLS)
  - Martin Stølevik, SINTEF

Leksjon 6 - Oversikt

- Genetiske algoritmer (GA)
- Øvrige populasjonsbaserte metoder
  - Memetiske algoritmer
  - Maurkolonioptimering