

Generelle optimeringsheuristikker

Også kaldet *metaheuristikker*. I dag gennemgås:

- Indplacering
- Lokal søgning
- Simuleret udglødning
- Tabusøgning
- Genetiske algoritmer

Løsningsmetoder for \mathcal{NP} -hårde opimeringsproblemer

Opdelingskriterier

- løsningskvalitet: optimal/ikke-optimal
- beregningstid: polynomiel/ikke-polynomiel

Approximationsalgoritme giver garanti for løsningskvalitet
Heuristik giver ingen garanti for løsningskvalitet

1

Generelle optimeringsheuristikker

Branch-and-bound

- helhedsbetragtninger til at finde grænseværdier
- global struktur holder styr på alle delproblemer

Konstruktionsheuristikker

- konstruerer en løsning fra grunden
- når først et valg er truffet er det låst

Lokalsøgning

- forbedrer løsninger, tidligere valg ændres
- omegn, validering, objektfunktion

Definitioner

Løsningsrum S

Objektfunktion $f : S \rightarrow \mathbb{R}$ skal minimeres

Omegn $N(s) \subset S$ for hver løsning $s \in S$

Omegn skal opfylde at alle løsninger kan nås

Skridt (*overgang*) betegner $s \rightarrow s'$

2

Simpleste lokalsøgning

algorithm lokalsøgning

vælg en initial løsning $s \in S$

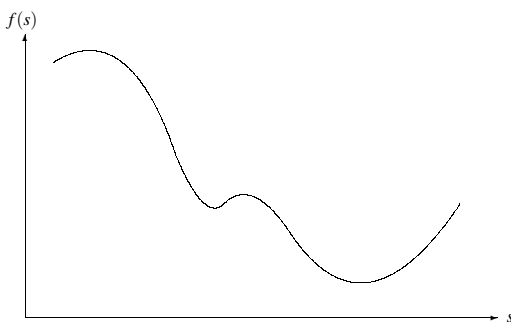
repeat

 vælg en løsning $s' \in N(s)$

if accepter_ny_løsning(s, s') **then** $s := s'$

until stopkriterium;

Acceptkriterium: f.eks. trinvis forbedring



Eksempel: TSP

3

Design af lokalsøgningsalgoritmer

Balanceakt mellem

- *Intensivering* (skal forfølge gode løsninger til bunds)
- *Diversifikation* (skal komme rundt i løsningsrummet)

Intensivering

- Grådigt paradigme
- Selvom havner i lokalt minimum, undersøges omegn

Diversifikation

- Genstart fra tilfældigt sted
- Accepter dårligere løsninger nogle gange

Undgå at besøge tidligere besøgte løsninger

- tilfældighed i valg af naboløsninger
- forbyd alle tidligere besøgte løsninger

4

Varianter af lokalsøgning

algorithm trinvis forbedring
vælg en initial løsning $s \in S$
repeat
 vælg en løsning $s' \in N(s)$
 if $f(s') < f(s)$ **then** $s := s'$
until s er et lokalt minimum
return s

Løsning s er lokalt minimum hvis $\forall s' \in N(s) : f(s') \geq f(s)$

- Trinvis forbedring: Vælg en vilkårlig $s' \in N(s)$ hvor $f(s') < f(s)$.
- Stejleste nedstigning: Vælg $s' \in N(s)$ så $f(s') < f(\hat{s})$ for alle $\hat{s} \in N(s)$

Genstart: Benyt forskellige initielle løsninger for trinvis forbedring eller stejleste nedstigning.

5

Ordbog

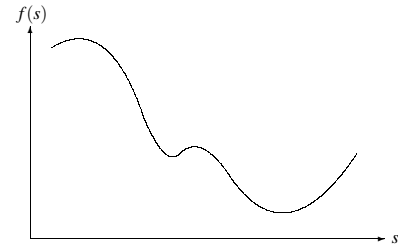
Dowland (1995) viser følgende sammenhæng mellem fysisk nedkøling og optimering

Thermodynamic Simulation	Combinatorial Optimisation
System States	Feasible Solutions
Energy	Cost
Change of State	Neighbouring Solutions
Temperature	Control Parameter
Frozen State	Heuristic Solution

7

Simuleret udglødning

Fysiske systemer: metal, glas (Metropolis 1953)
Heuristik (Kirkpatrick 1983)

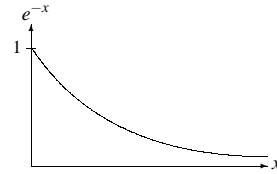


Heuristik har intet at gøre med fysiske systemer

- $s' \in N(s)$ vælges tilfældigt
- Intensivering: accepter altid $f(s') \leq f(s)$
- Diversifikation: accepter $f(s') > f(s)$ med sandsynlighed

$$P(s \rightarrow s') = e^{-\frac{f(s')-f(s)}{T}}$$

hvor T er aftagende kontrolparameter (kaldet *temperatur*)



6

Simuleret udglødning

algorithm simuleret udglødning
vælg en initial løsning $s \in S$
vælg en initial temperatur T
repeat
 vælg en løsning $s' \in N(s)$
 if $f(s') \leq f(s)$ **then** $s := s'$
 else
 $p := \text{rand}(0, 1)$
 if $p < e^{-\frac{f(s')-f(s)}{T}}$ **then** $s := s'$
 end
 $T := \text{ny temperatur}(T)$
until stopkriterium
return s

- Hvad skal initial temperatur T_0 være?
- Hvorledes skal T_{i+1} udregnes på basis af T_i ?
- Hvornår skal algoritmen slutte ?

8

Simuleret udglødning

Initial temperatur

- Bestemmes eksperimentelt
- Accept ratio $\chi(T) \approx 0.5$ ved T_0

Kølingsrate

- $T_{i+1} := \alpha T_i$, $0 \leq \alpha < 1$
- jo større α des længere køretid

Stopkriterie

- Temperaturen T kommer under en given grænse T_s
- Ingen forbedring sidste K iterationer

Der findes også en variant af simuleret udglødning hvor der foretages M skridt for hver temperatur, inden denne sænkes. Ingen signifikant forskel.

9

Simuleret udglødning

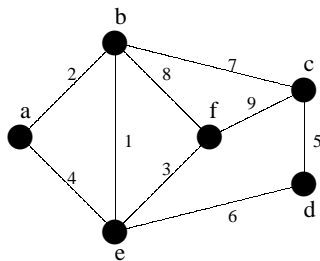
Laarhoven (1988) viste at hvis temperatur sænkes uendeligt langsomt, og der er uendeligt mange skridt til rådighed, så vil simuleret udglødning returnere optimal løsning.

$$\lim_{\substack{\infty \text{ langsom afkøling} \\ \infty \text{ mange iterationer}}} P(\text{optimal løsning}) \rightarrow 1$$

Bevis baseret på Markovkæder

10

Simuleret udglødning for grafdeling



Givet graf $G = (V, E)$ og omkostnings funktion $c : E \rightarrow \mathbb{N}$. Del V i to lige store dele V_1 og V_2 så omkostningen af kanter (u, v) hvor $u \in V_1$ og $v \in V_2$ minimeres.

Lad $x_v = 1$ hvis $v \in V_1$ mens $x_v = 0$ hvis $v \in V_2$. Kvadratisk minimeringsproblem:

$$\min \sum_{u,v \in V} c_{uv} x_u (1 - x_v)$$

hvor

$$\sum_{v \in V} x_v = |V|/2$$

11

Simuleret udglødning for grafdeling

- Objektfunktion:

$$f(x) = \sum_{u,v \in V} c_{uv} x_u (1 - x_v)$$

- Omegn: Ombyt to elementer k, ℓ hvor $k \in V_1$ og $\ell \in V_2$
- Udregning af objektfunktion:

Trivielt $O(V^2)$

Ved ombytning af knuder ændres objektfunktionen

$$\begin{aligned} \Delta f &= \sum_{v \in V} x_v c_{kv} - \sum_{v \in V} (1 - x_v) c_{kv} \\ &\quad + \sum_{v \in V} (1 - x_v) c_{\ell v} - \sum_{v \in V} x_v c_{\ell v} \end{aligned}$$

(idet $x_k = 1 \rightarrow x_k = 0$ og $x_\ell = 0 \rightarrow x_\ell = 1$). Kan beregnes i $O(V)$ tid.

12

Simuleret udglødning, diskussion

Intensivering ↔ diversifikation

- Intensivering: Accepter altid forbedrende løsning
- Diversifikation: Accepter forværret løsning med lille sandsynlighed
- Algoritmen gennemløber kontinuum fra stærk diversifikation til intensiv søgning

Fordele ↔ ulemper

- + Robust algoritme uanset omegn $N(s)$
- + Få parametre (T_0, α), disse kan justeres automatisk
- Ikke videre grådige, meget tid bruges på at gætte

Hints

- Mest vellykket søgning “midt” i udglødning, derfor varianter med genopvarmning
- Udregn $f(s') := f(s) + \Delta f$
- Kan være hensigtsmæssigt at tillade ulovlige løsninger mod straf i objektfunktion
- Genstart

13

Tabusøgning

Tidligere algoritmer

- Trinvis forbedring
- Stejleste nedstigning
- Genstart
- Simuleret udglødning

Ingen af ovenstående har hukommelse

Tabusøgning

- opsamler information og bruger hukommelse til at styre søgning
- har ingen simpel definition
- er svær at analysere matematisk
- kræver mere tilpasning til hvert problem/omegn

14

Tabusøgning

Paradigmet anvendt i 1970'erne, nuværende form indført af Glover (1986)

- “... a framework that links the perspectives of artificial intelligence and operations research”
- “... effective strategies for combinatorial problems can require methods that formal theorems are unable to justify”
- “Methods that are 'intelligently flexible' lie in some yet-to-be-defined realm to which theorems do not apply, yet which embraces enough structure to exclude aimless wandering”

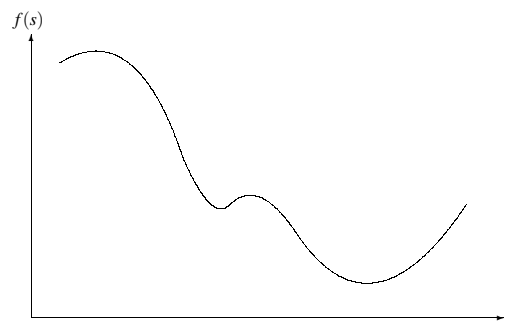
Vellykkede anvendelser og klogt valgt navn har gjort tabusøgning til standard metaheuristik

Ordbog

AI	ordinary meaning
tabu	forbidden (change of solution vector)
tabu list	table of previous solutions (compact)
aspiration level	objective function value
aspiration criterion	accept criterion
long time memory	statistics of good/bad solutions

15

Tabusøgning



Videreudvikling af stejleste nedstigning

algorithm tabusøgning

vælg en initial løsning $s \in S$

sæt $s^* := s$

initialiser tabuliste

repeat

 fi nd bedste løsning $s' \in N(s)$ som ikke er tabu

$s := s'$

 opdater tabuliste

if $f(s) \leq f(s^*)$ **then** $s^* := s$

until stopkriterium

return s^*

16

Tabusøgning

Grundlæggende strategi

- vælg mest forbedrende (eller mindst forværende) skridt i hver iteration
- undgå cykler ved at forbyde visse løsninger/skridt, såkaldte tabuløsninger

Implementation

- forbyd alle tidligere løsninger
- forbyd max k tidligere løsninger
- forbyd max k tidligere skridt
- forbyd løsninger med en given attribut

Tabuliste: hvis ændrer $x_i = a$ til $x_i = b$ tilføjes $x_i \neq a$

$x_3 \neq 1$	$x_4 \neq 7$	$x_5 \neq 2$
--------------	--------------	--------------

17

Tabusøgning

Tabulistens længde 5–10.

- for lille: cykler
- for stor: tidskrævende at kontrollere tabu alt for restriktiv

Med længde M garanteres cykler $\geq M + 1$

Aspirationskriterie

Giver mulighed for at bryde forbud i tabulisten

- hvis objektfunktion bedre end hidtil globalt bedste
- hvis objektfunktion bedre end daværende bedste

$x_3 \neq 1$	$x_4 \neq 7$	$x_5 \neq 2$
17	12	14

18

Tabusøgning for grafdeling

- Objektfunktion:

$$f(x) = \sum_{u,v \in V} c_{uv} x_u (1 - x_v)$$

- Omegn: Ombyt to elementer k, ℓ hvor $k \in V_1$ og $\ell \in V_2$
- Mindre Omegn: Flyt et element fra V_1 til V_2 eller omvendt. Straf for ej overholdt begrænsning.
- Effektiv udregning af objektfunktion

$$f(s') := f(s) + \Delta f$$

- Tabuliste: Hvis knude k flyttes fra V_1 til V_2 ($x_k = 1 \rightarrow x_k = 0$)

variabel	k	ℓ	7	8	3
forbudt tilstand	1	0	1	0	0

- Hvis aspirationskriterie benyttes husker jeg gældende objektfunktion sammen med tabu listen:

variabel	k	ℓ	7	8	3
forbudt tilstand	1	0	1	0	0
objektfunktion	55	63	57	58	61

19

Tabu Søgning for Kvadratisk Tildeling

- n faciliteter med strøm a_{ij} fra facilitet i til j
- n lokaliteter med afstand b_{ij} fra lokalitet i til j
- find en permutation π der minimerer

$$f(\pi) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} b_{\pi(i)\pi(j)}$$

- π_i er facilitet i 's placering

lokalitet	1	2	3	4	5
facilitet	2	4	3	5	1

- Løsningsrum: alle mulige permutationer af n tal, $O(n!)$

20

Tabu Søgning for Kvadratisk Tildeling

- Omegn

$$N(\pi) = \{\pi' \mid \pi' \text{ opnåes ved at ombytte to elementer i } \pi\}$$

lokalitet	1	2	3	4	5
facilitet	2	1	3	5	4

- I tabu-listen gemmes følgende skridt

(i, j) : facilitet i og j bytter plads

(u, v) : lokalitet u og v bytter plads

- Dårlig repræsentation da

$i \ j \ k \ l$
 $i \ k \ j \ l$
 $l \ k \ j \ i$
 $l \ k \ i \ j$
 $k \ l \ i \ j$
 $i \ l \ k \ j$
 $i \ j \ k \ l$

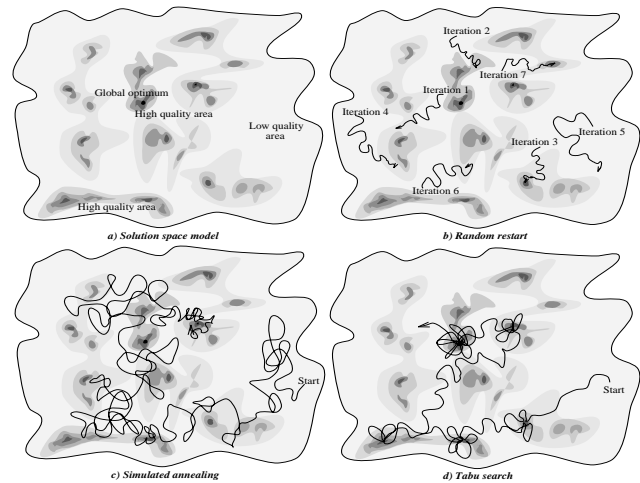
forhindrer ikke cyklisk opførsel

- I tabu-listen gemmes følgende skridt

$(i, \pi(i))$ og $(j, \pi(j))$

21

Sammenligning



22

Tabusøgning, diskussion

Intensivering ↔ diversifikation

- Intensivering: Bedste løsning i $N(s)$ vælges
- Diversifikation: Tabuliste, aspirationskriterier

Fordele ↔ ulemper

- + Meget grådig
- Følsom overfor omegns størrelse da hele $N(s)$ skal gennemses
- Flere parametre end simuleret udglødning
- Deterministisk, kan derfor cykle
- + Veldesignet algoritme giver gode løsninger

Hints

- Betragt kun en delmængde af $N(s)$ hvis $N(s)$ er stor
- Udregn $f(s') := f(s) + \Delta f$
- Kan være hensigtsmæssigt at tillade ulovlige løsninger mod straf i objektfunktion
- Reaktiv tabusøgning, langtidshukommelse, genstart

23

Genetiske algoritmer

Indført af Holland (1975) motiveret af Darwinisme

```
TCAAGCAGATCACTGTCCTTCTGCCATGGCCCTGT
GGATGCGCCTCCTGCCCTGCTGGCGCTGCTGGCC
```

Princip

- population af løsninger
- udvælgelse
- krydsning
- mutation

algorithm genetisk optimering

vælg en initial population $P = \{s_i\}$, hvor $s_i \in S$

repeat

```
  evaluer population( $P$ );
  udvælg individer( $P$ , basis);
  kryds individer(basis,  $P'$ );
  muter individer( $P'$ );
   $P := P'$ 
```

until stopkriterium

return P

24

Genetiske algoritmer

Udvælgelse

- “survival of the fittest”
- objektfunktion $f(s) \rightarrow K(s)$ kvalitetsmål
- udvælgessandsynlighed

$$p_u(s) = \frac{K(s)}{\sum_{s' \in P} K(s')}$$

vælger probabilistisk $|P|$ gange

Krydsning: gentag til $|P|$ individer

- 1 udvælg en tilfældig løsning s_1 fra basis
- 2 overfør s_1 direkte til nye generation med sandsynlighed p_d
- 3 udvælg en anden løsning s_2 , kryds s_1 og s_2 overfør afkom til ny generation med sandsynlighed $(1 - p_d)$.

Bemærk: “gamle” individer kan overleve, typisk $p_d \approx \frac{1}{2}$

25

Genetiske algoritmer

Krydsning

- to-punkts krydsning

s_1 : 01101001 00101010 1110

s_2 : 01100111 11001101 0100

s'_1 : 01101001 11001101 1110

s'_2 : 01100111 00101010 0100

- bedste eller begge medtages i næste generation

Mutation

- ændrer et tegn

s_1 : 011010010 0 1010101110

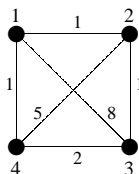
s'_1 : 011010010 1 1010101110

- sikre spredning i population da risiko for at alle ens
- sandsynlighed $p_m = \frac{1}{|P|}$
- krydsning skal være vigtigste operator

26

Genetiske algoritmer, TSP

$G = (V, E)$ med vægte d_{ij} , find korteste Hamilton-kreds



Permutation π af $\{1, 2, 3, \dots, n\}$ som minimierer

$$f(\pi) = \sum_{i=1}^n d_{\pi(i)\pi(i+1)}$$

Krydsning skal bevare at π er permutation

π_1 : 2 1 5 4 7 3 8 6

π_2 : 3 1 5 8 2 4 6 7

π'_1 : 2 1 5 8 2 4 8 6 (to-punkts)

π'_2 : 3 1 5 4 7 3 6 7

π'_1 : 2 1 5 3 8 4 6 7 (et-punkts)

π'_2 : 3 1 5 2 4 7 8 6

Mutation

π : 2 1 5 4 7 3 8 6

π' : 2 1 3 4 7 5 8 6 (ombyt to elementer)

27

Genetiske algoritmer, diskussion

Intensivering \leftrightarrow diversifikation

- Intensivering: Population af løsninger
- Diversifikation: Krydsning, mutation

Fordele \leftrightarrow ulemper

- + Returnerer familie af løsninger
- + Bevarer lokale egenskaber
- Dyrt at vedligeholde mange ens løsninger
- Svært at finde god kodning for mange problemer
- Temmelig mange parametre
- + Parametre kan indgå som del af strengen (mod øget beregning)

Hints

- Population $|P| \approx 100$
- Stopkriterium: uændret bedste løsning K gange

28

Litteratur

P.J.M. van Laarhoven (1988), "Theoretical and computational aspects of simulated annealing", Erasmus University, Rotterdam, PhD thesis.

N. Metropolis m.fl. (1953) "Equation of state calculations by fast computing machines", Journal of Chemical Physics, 21, 1087-1092.

S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi (1983) "Optimization by Simulated Annealing", Science 220, 671-680.

J.H. Holland (1975) "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Arbor.

Reeves (1995) "Modern Heuristic Techniques for Combinatorial Problems", McGraw-Hill.

F. Glover (1986) "Tabu Search, part I", ORSA Journal on Computing 1, 190-206.

Dowland (1995) "Variants of Simulated Annealing for Practical Problem Solving" in V. J. Rayward-Smith (Ed) Applications of Modern Heuristic Methods, Henley-on-Thames, Alfred Waller, 3-16.