



Heuristické optimalizačné procesy

Hybridné lokálne prehl'adávanie

prednáška 7

Ing. Ján Magyar, PhD.
ak. rok. 2022/2023 ZS

Hybridné algoritmy

kombinovanie jednoduchších stratégií do komplexnejších schém

pridanie triviálnych stratégií

- RII (II + URW)
- II s reštartmi (II + URP)

kombinovanie dvoch stratégií

- ILS
- GRASP
- AILS

Iteračné lokálne prehl'adávanie

nevyhýba sa lokálnemu optimu

dva typy krokov

- lokálne prehl'adávanie - dosiahnutie lokálneho optima
- perturbácia - štartovací bod pre nové lokálne prehl'adávanie

pohyb v priestore lokálnych optím - voľba optima pre pokračovanie

príklad: iteračný Lin-Kernighan (pre TSP)

Štruktúra ILS

input: π

output: $r \in S$ ☐

$s = \text{init}()$

$s = \text{localsearch}(s)$

$r = s$

while not $\text{term}(s)$

$s' = \text{perturb}(s)$

$s'' = \text{localsearch}(s')$

if($g(s'') > g(r)$) **then**

$r = s''$

endif

$s = \text{accept}(s, s'')$

endwhile

if($\text{valid}(r)$) **then**

return r

else

return ☐

endif

Lačné randomizované adaptívne hľadanie (GRASP)

prehl'adávanie začína z čo najlepšieho kandidáta

dva typy krokov

- tvorba štartovacieho kandidáta (konštrukčné prehl'adávanie)
- lokálne zlepšovanie kandidáta (perturbačné prehl'adávanie)

SGH - vynechanie perturbačnej fázy

Štruktúra GRASP

input: π

output: $r \in S$ ☐

$r = \square$, $g(r) = -\infty$

while not *term*(s)

$s = \text{construct}()$

$s' = \text{localsearch}(s)$

if($g(s') > g(r)$) **then**

$r = s'$

endif

endwhile

if(*valid*(r)) **then**

return r

else

return ☐

endif

GRASP - konštrukčná fáza

štartuje z prázdneho kandidáta

prechádza parciálnymi kandidátmi

- dopĺňa jednu zložku v každej iterácii

končí vytvorením úplného kandidáta

generuje rôznych úplných kandidátov

Obmedzený zoznam kandidátov

založený na počte

- pevne daný počet k
- náhodný alebo lačný výber

založený na ohodnotení (prah $k \in <0, 1>$)

$$g(z) \leq g(z_{\perp}) + k * (g(z_{\top}) - g(z_{\perp}))$$

GRASP *construct()* - greedy-random

construct()

$x = \square$

$V = \{ v_1, v_2, \dots, v_n \}$

while not *complete*(x)

$SC = \text{sort}(V, g)$

$RCL = \text{form}(SC, k)$

$v = \text{select}(RCL)$

$x = x + v$

$V = V - v$

endwhile

return x

endconstruct

GRASP *construct()* - random-greedy

construct()

$x = \square$

$V = \{ v_1, v_2, \dots, v_n \}$

while not *complete*(x)

$RCL = \text{sample}(V, k)$

$SRCL = \text{sort}(RCL, g)$

$v = \text{select-best}(SRCL)$

$x = x + v$


$V = V - v$

endwhile

return x

endconstruct

GRASP *construct()* - random+greedy

```
construct()  
  x =   
  V = { v1, v2, ..., vn }  
  for i=1, ..., k  
    v = select-random( V )  
    x = x + v  
    V = V - v  
  endfor  
  while not complete( x )  
    SV = sort( V, g )  
    v = select-best( SV )  
    x = x + v  
    V = V - v  
  endwhile  
  return x  
endconstruct
```

Adaptívne iteračné konštrukčné hľadanie (AICS)

spätná väzba

- forma súboru váh
- váhy sú adaptované v každej iterácii
 - podľa komponentov tvoriacich dosiahnutého kandidáta
 - podľa kvality dosiahnutého kandidáta

použiteľné aj bez perturbačného hľadania

príklad: SWO (squeaky wheel optimization)

Štruktúra ALCS

input: π

output: $r \in S \mid \square$

$r = \square$, $g(r) = -\infty$

$w = \text{initweights}()$

while not $\text{term}(s)$

$s = \text{construct}()$

$s' = \text{localsearch}(s)$

if($g(s') > g(r)$) **then**

$r = s'$

endif

$w = \text{adaptweights}(s', w)$

endwhile

if($\text{valid}(r)$) **then**

return r

else

return \square

endif

otázky?