Heuristické optimalizačné procesy

Hybridné lokálne prehľadávanie

prednáška 6

Ing. Ján Magyar, PhD.

ak. rok. 2025/2026 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 prehľadávanie

nevyhýba sa lokálnemu optimu

dva typy krokov

- lokálne prehľadávanie - dosiahnutie lokálneho optima
- perturbácia - štartovací bod pre nové lokálne prehľadávanie

pohyb v priestore lokálnych optím - voľba optima pre pokračovanie

príklad: iteračný Lin-Kernighan (pre TSP)

Štruktúra ILS

input: π
output: $r \in S$

s = *init()*
s = *localsearch*(s)
r = s
while not *term*(s)
    s' = *perturb*(s)
    s'' = *localsearch*(s')
    if(g(s'') > g(r)) **then**
        r = s''
    endif
    s = *accept*(s, s'')
endwhile

if(*valid*(r)) **then**
    return r
else
    return
endif

Lačné randomizované adaptívne hľadanie (GRASP)

prehľadávanie začína z čo najlepšieho kandidáta

dva typy krokov

- tvorba štartovacieho kandidáta (konštrukčné prehľadávanie)
- lokálne zlepšovanie kandidáta (perturbačné prehľadávanie)

SGH - vynechanie perturbačnej fázy

Štruktúra GRASP

input: π
output: $r \in S \mid$□

r = □ , g(r) = -∞
while not *term*(s)
    s = *construct*()
    s' = *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 (RCL)

založený na počte

- pevne daný počet *k*
- náhodný alebo lačný výber

založený na ohodnotení (prah $k \in \langle 0, 1 \rangle$)

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

GRASP *construct()* - greedy-random

construct()
    x = □
    V = { v_1, v_2, …, v_n }
    while not *complete*(x)
        SC = *sort*(V, g)
        RCL = *form*(SC, k)
        v = *select*(RCL)
        x = x + v
        V = V - v
    endwhile
    return x
endconstruct

GRASP *construct()* - random-greedy

construct()
    x = □
    V = { v_1, v_2, …, v_n }
    while not *complete*(x)
        RCL = *sample*(V, k)
        SRCL = *sort*(RCL, g)
        v = *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 AICS

input: π
output: $r \in S \mid$□

r = □ , g(r) = -∞
w = *initweights*()
while not *term*(s)
    s = *construct*()
    s' = *localsearch*(s)
    if(g(s') > g(r)) **then**
        r = s'
    endif
    w = *adaptweights*(s', w)
endwhile

if(*valid*(r)) **then**
    return r
else
    return □
endif

otázky?