VLIV PARAMETRU Λ NA CHOD ALGORITMU PENALIZAČNÍHO LOKÁLNÍHO PROHLEDÁVÁNÍ

Jan Panuš
Univerzita Pardubice, Fakulta ekonomicko-správní, Ústav systémového inženýrství a informatiky

Abstract: *The combinatorial explosion problem prevents complete algorithms from solving many real-life optimization problems. In many situations heuristic search methods are needed. This paper deals with an aspect of Penalty Local Search algorithm that is combination of aspiration criterion and λ parameter. We show how this parameter can affect the running of Penalty Local Search algorithm.*

Key words: *Local Search Algorithm, λ parameter, optimization.*

Úvod

Algoritmus penalizačního lokálního prohledávání (PLP) je optimalizační technika využitelná pro celou řadu kombinatorických optimalizačních problémů. Tento algoritmus využívá výhod daných problémem a informacemi, které jsou spjaty při prohledání daného prostoru. Toto je umožněno zvyšováním účelové funkce daného problému pomocí nově vytvořené množiny poku[1]. Lokální vyhledávání je omezováno touto množinou pokut a během hledání optimálního (resp. suboptimálního) řešení je pak možné se soustředit na slibné oblasti prohledávaného prostoru. Algoritmus se provádí opakovaně a prohledává lokální okolí určeného bodu. Jakmile se algoritmus zastaví v lokálním minimu, množina pokut, jež je na začátku prázdná (hodnoty penalizací jsou nastaveny na nulu), je modifikována a algoritmus dále optimalizuje upravenou účelovou funkci.

Jak se penalizační faktory dále mění, jsou jednotlivá nalezená řešení upřesňována a postupně jsou lokalizována nová lokální minima. Lokální vyhledávání je pak generováno podle znalostí, jež jsou daná buď na počátku prohledávání, nebo se postupně objevují v průběhu prohledávání. Tyto znalosti se týkají především umístění a hodnotě lokálních minim a především se týkají hodnoty a umístění globálního minima. Chování tohoto algoritmu může být postaveno na podobném principu jako je metoda špatně postavené úlohy (ill-posed) [5], kdy se využívají jisté znalosti, které předchází řešení a slouží k vyřešení aproximačních problémů. Tyto konkrétní znalosti pak představují určitá omezení, která dále definují problém, tak že se redukuje počet všech možných kandidátů na vyřešení daného problému. Algoritmus také využívá dalších znalostí, které se naučí během prohledávání a to využitím dalších omezení, která se objeví na základě prohledávání daného prostoru. V podstatě se dá říci, že PLP založeno na meta-heuristice lokálního prohledávání.

Důležitým prvkem, který bude zkoumán v této práci a jež je součástí algoritmu penalizovaného lokálního prohledávání (PLP) [7], je aspirační kritérium. Aspirační kritérium se běžně využívá u algoritmu zakázaného prohledávání (TS) [1], [2], [3]. V této práci bude popsáno jakým způsobem se aspirační kritérium přidá do PLP a následně bude ukázáno, jak je chování algoritmu ovlivněno tímto prvkem. Současně bude naznačeno jakým způsobem se PLP chová při různých hodnotách parametru λ. Následně budou ukázány experimenty provedené na dvou testovacích funkcích.

[1] Pokutou se rozumí hodnota, která bude přiřazena k jistému argumentu účelové funkce a ovlivní tak další prohledávání v lokálním prostoru. Pokuta může být také nazývána penalizací nebo penalizačním faktorem.

Důvod využití aspiračního kritéria

PLP využívá tzv. penalizačních faktorů pro únik z lokálního minima pomocí penalizací hodnot argumentů prohledávané účelové funkce. Argumenty účelové funkce jsou pak penalizovány tak dlouho, než je algoritmu umožněno uniknout z lokálního minima. Aspirační kritérium v TS povoluje použití zakázané transformace v případě, že vzniklé sousední řešení poskytuje lepší hodnotu účelové funkce, než má dočasné nejlepší řešení. Toto kritérium je jedno z nejužívanějších forem aspiračního kritéria a je nazýváno vylepšené nejlepší aspirační kritérium [4]. Ve skutečnosti existuje mnoho forem tohoto kritéria jak je uvedeno v [4]. V této práci budou brány v úvahu pouze vylepšená nejlepší aspirační kritéria.

Pro PLP byla na argumenty účelové funkce uvalena množina penalizací na rozdíl od využití tabu seznamu jednotlivých řešení nebo tabu seznamu přesunů v prostoru. Zlepšujícím nejlepším aspiračním přesunem v algoritmu je pak takový přesun, který vygeneruje nejlepší nově nalezené řešení a zároveň tento přesun není vybrán lokálním hledáním za využití zvýšené účelové funkce. Nyní následuje pseudokód pro lokální hledání za využití aspiračního kritéria.

```
Lokální_Prohledání_S_Aspiračním_Kritériem (x,f,h,N)
{
     Do
     {
     If (Přesun_Pomocí_Aspirace (x,f,h,N,z))
       x=z
     else
          {
            y = y v N(x) takové, že h(y) je minimalizováno
            ∆h = h(y) – h(x)
            If (∆h <= 0) then x = y
            If (∆h > 0) then  iterace = iterace + 1
            Else              iterace = 0
          }
     If (f(x) < f(x*)) then x* = x
     }
     While (∆h <= 0) a (iterace < 2)
     Return x
}
Přesun_Pomocí_Aspirace (x,f,h,N,z)
{
     z = z v N(x) takové, že f(z) je minimalizováno
     if (f(z) < f(x*) a ((h(z) – h(x)) > 0 ))
          return true
     else
          return false
}
```

kde:

- *x, y* a z jsou hodnoty řešení,
- *f()* vrací cenu řešení vzhledem k originální účelové funkci,
- *h()* vrací zvýšenou cenu řešení,
- x^* je nejlepší nalezené řešení v průběhu běhu algoritmu,

- *N(x)* je funkce sousedství, která dá konkrétní řešení tohoto sousedství *x*.

Tento postup může být jednoduše implementován na různé typy problémů, pokud je uvažován nějaký přesun v sousedství, který přinese nové nejlepší řešení shodné s účelovou funkcí *f*. Sousední okolí tohoto řešení je poté prozkoumáno za použití zvýšené účelové funkce *h*, a to z toho důvodu, aby se poznalo, zda existuje jiný přesun, který může zredukovat zvýšenou cenou dalšího řešení. Jestliže nejlepší přesun v sousedství přinese nové lepší řešení takové, že cena originální účelové funkce je menší než ta cena, která by byla vybrána s ohledem na zvýšenou účelovou funkci, pak vybereme ten přesun, který vygeneruje nejnižší cenu nového nejlepšího řešení (což se dá považovat za aspirační přesun). Jinak vybereme takový přesun s nejnižší hodnotou zvýšené účelové funkce.

Parametr λ v algoritmu PLP

Omezení na argumentech funkce je vytvářeno zvyšováním účelové funkce *f* problému a to započítáním prvků z množiny penalizačních faktorů. Nová účelová funkce je nazvána zvýšenou účelovou funkcí a je definována následovně

$$f(x) = g(x) + l \times \sum_{i=1}^{M} \mathbf{p}_i \times U_i(x), \qquad \textbf{(1)}$$

kde *M* je počet argumentů stanovených před samotným řešením problému, p_i je penalizační parametr odpovídající argumentu f_i a l (lambda) je regulující parametr dané funkce. Penalizační parametr p_i předává hodnoty do těch argumentů účelové funkce f_i, které jsou zahrnuty v aktuálním řešení. Regulační parametr l reprezentuje poměrovou část penalizací a ovlivňuje výslednou velikost hodnoty účelové funkce. Jeho význam tkví v možnosti kontroly vlivu znalostí na procesu prohledávání prostoru. Algoritmus opakovaně používá lokálního prohledávání a modifikuje penalizační vektor ***p***, tak že každý parametr p_i je zvýšen o hodnotu 1 v případě uváznutí algoritmu v lokálním minimu prohledávané funkce. Parametr $U_i(x)$ určuje zda má být konkrétní argument účelové funkce penalizován či nikoliv. Tato hodnota je buď 0 a v takovém případě argument penalizován není anebo jeho hodnota je 1 a pak argument je penalizován. Tyto dvě hodnoty jsou určovány podle toho, zda daný argument účelové funkce

Významným činitelem na kvalitu provedených výpočtů má i parametr l, který ovlivňuje velikost finální hodnoty penalizace, jež je započítána do zvýšené účelové funkce. V následující části bude popsáno jakým způsobem byly prováděny experimenty s pomocí tohoto parametru.

Provedené experimenty

Pro otestování kvality algoritmu s aspiračním kritériem byl tento algoritmus otestován na vybraných testovacích funkcích [8]. Ve všech těchto případech byl porovnán jak základní algoritmu penalizovaného prohledávání bez aspiračního kritéria tak i upravený algoritmu s aspiračním kritériem. Experimenty byly spouštěny na osobním počítači P4 CPU 2,80 GHz a navržený algoritmus byl implementován v programovacím jazyku Visual Basic.NET.

Ve všech těchto případech byla také zkoumána velikost parametru λ na kvalitu řešení daného problému. Velikosti hodnot parametru jsou vyjádřeny v (2).

$$\lambda = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\} \quad \textbf{(2)}$$

Z celé rodiny testovacích funkcí byly vybrány 2 funkce a to 1st de Jongova funce a Rastriginova funkce. V průběhu měření byly zaznamenány různé hodnoty, jež vyjadřují kvalitu algoritmu a také vyjadřují, co se děje v průběhu prohledávání.

a) počet iterací, které jsou nutné k dosažení optimálního (nebo alespoň suboptimálního) řešení,

b) nejlepší nalezené řešení,

c) počet změn nejlepšího nalezeného řešení – hodnota vyjadřující kolikrát bylo změněno nejlepší nalezené řešení za další lepší nalezené v průběhu prohledávání daného prostoru,

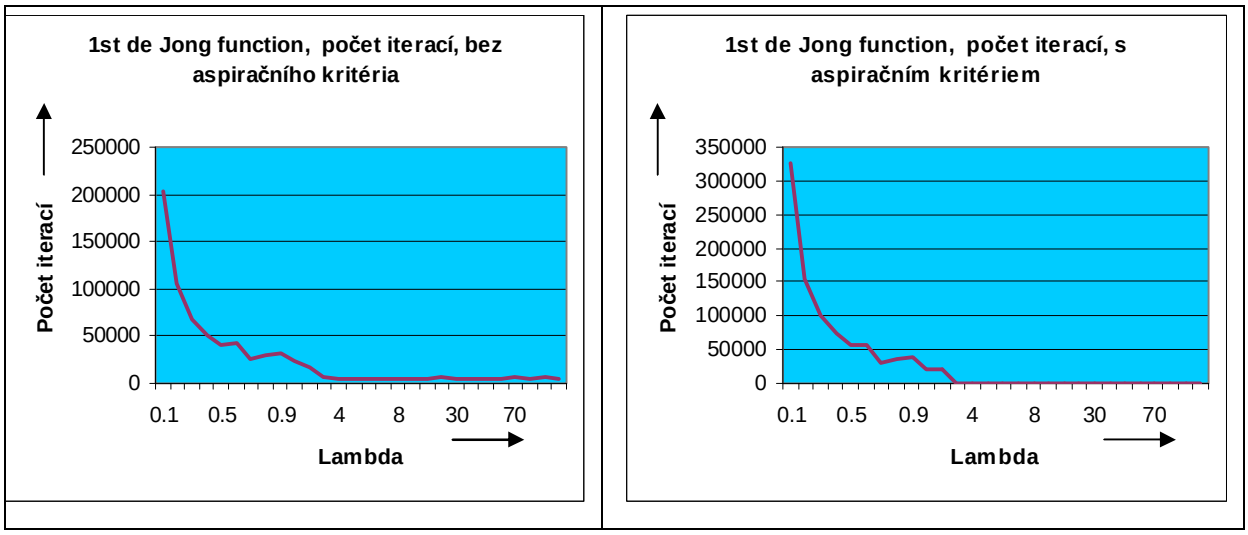
d) časová náročnost výpočtu.

Pro testování optimalizačních algoritmů se používají dva rozdílné postupy. V prvním z nich se obvykle vychází z již dříve řešených příkladů, avšak pomocí jiných algoritmů. Výsledky z právě testovaného algoritmu lze pak porovnat s výsledky, které již existují. Druhý způsob spočívá ve využití množiny testovacích funkcí, které v sobě zahrnují různé vlastnosti, jako je nelinearita, různé patologie typu rovina okolo extrému apod. V této kapitole bude představeno několik funkcí, na kterých byly upravené algoritmy testovány. Pro každou funkci je zobrazen její analytický zápis, její vzhled. Názvy funkcí jsou ponechány v jejich originálních názvech z důvodů foneticko-jazykových. Vzhledem k tomu, že jsou známy analytické vztahy, je u většiny z nich velmi jednoduché vypočítat hodnotu a pozici globálního extrému pro libovolnou dimenzi.

Známe-li hodnotu extrému dané funkce v jednorozměrném prostoru, je možno z ní odvodit další znalosti. Například u Schwefelovy funkce [8] je tato hodnota pro souřadnici x = 420.97 rovna $f(x)$ = -418,9829. Pokud je nutno vypočítat hodnotu pro 20D, stačí vynásobit hodnotu extrému v 1D číslem dimenze, tedy 20. V tomto případě je pak hodnota globálního extrému pro Schwefelovu funkci v 20D rovna $f_{20}(x_1,x_2,\ldots,x20)$ = 20 *-418,9829, tedy 8379,658. Tentýž postup samozřejmě platí i pro ostatní funkce.

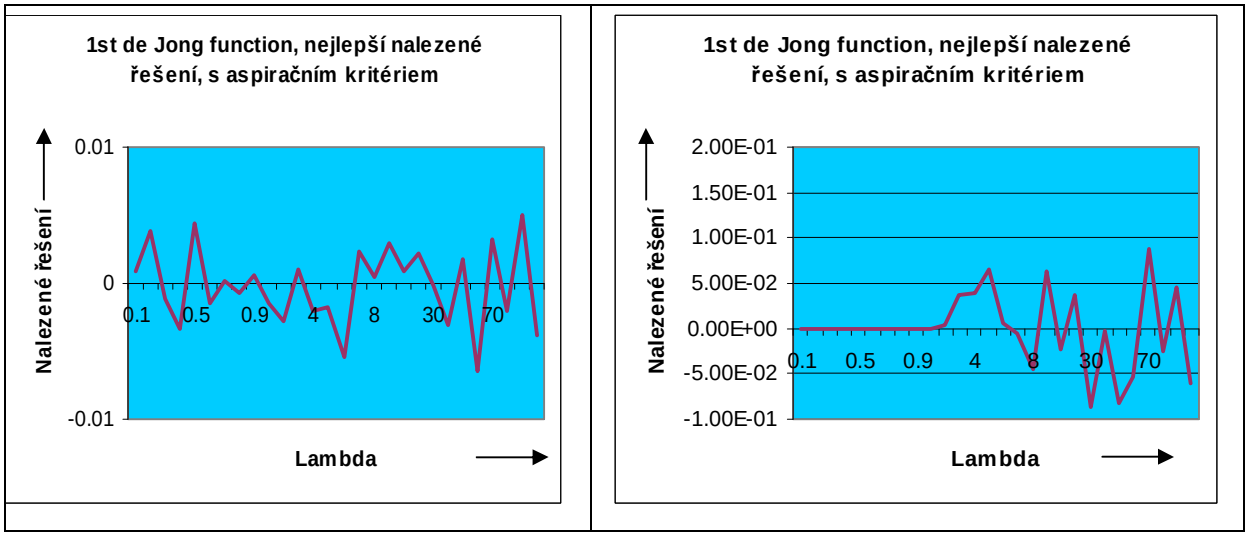
Celkem bylo spuštěno 100 testů a každá funkce byla definována pro 100D, tedy každá funkce měla 100 argumentů a jejich hodnota byla hledána tak, aby minimalizovala účelovou funkci.

Další problematikou, na kterou je zaměřena tato práce je nalezení takového parametru λ, který by mohl dávat dobré výsledky v rozumném čase. Z obrázků, které jsou zobrazeny dále v této práci je patrné, že dobré výsledky dává parametr, pokud se nachází v rozmezí <0.4, 0.9>, kdy jak časová náročnost klesá, tak i hodnoty nalezených extrémů nevykazují rozkolísanost okolo optimálních hodnot. Z toho hlediska lze uvažovat pro další výzkum a výpočty tento rozsah hodnot za přijatelný. Hodnoty, které jsou v rozmezí <1, 100> dávají také uspokojující řešení, ale pouze z hlediska časové náročnosti. Pokud by se měřila odchylka od globálního extrému, výsledky již tak dobré nejsou.



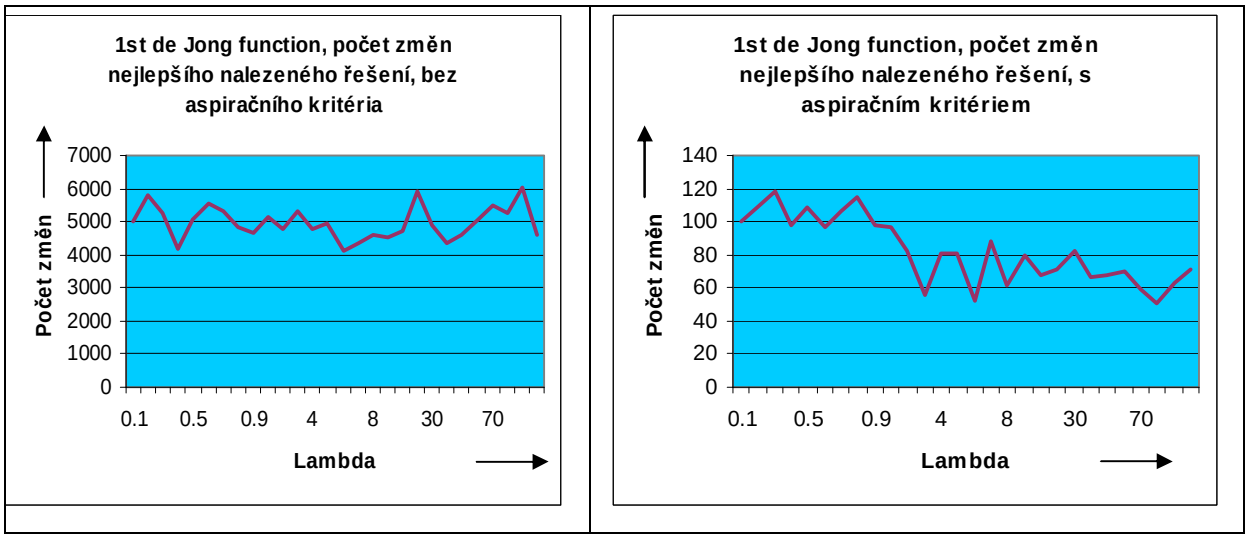
Obr. 1 - Porovnání počtu iterací u 1st de Jong funkce

Pro analýzu chování algoritmu na testovacích funkcích byly vybrány funkce 1st de Jongova (obr. 1 – obr. 3) a Rastriginova funkce (obr. 4 – obr. 5). Jak je patrné (obr. 1), algoritmus PLP bez aspiračního kritéria nalezne suboptimální řešení za pomoci menšího počtu iterací, než PLP s aspiračním kritériem. S hodnotami počtu iterací samozřejmě souvisí i celkový čas, který v této části analýzy nebude brán v potaz, avšak u celkového porovnání výsledku v příloze se nachází.



Obr. 2 - Porovnání nejlepších nalezených řešení u funkce 1st de Jong

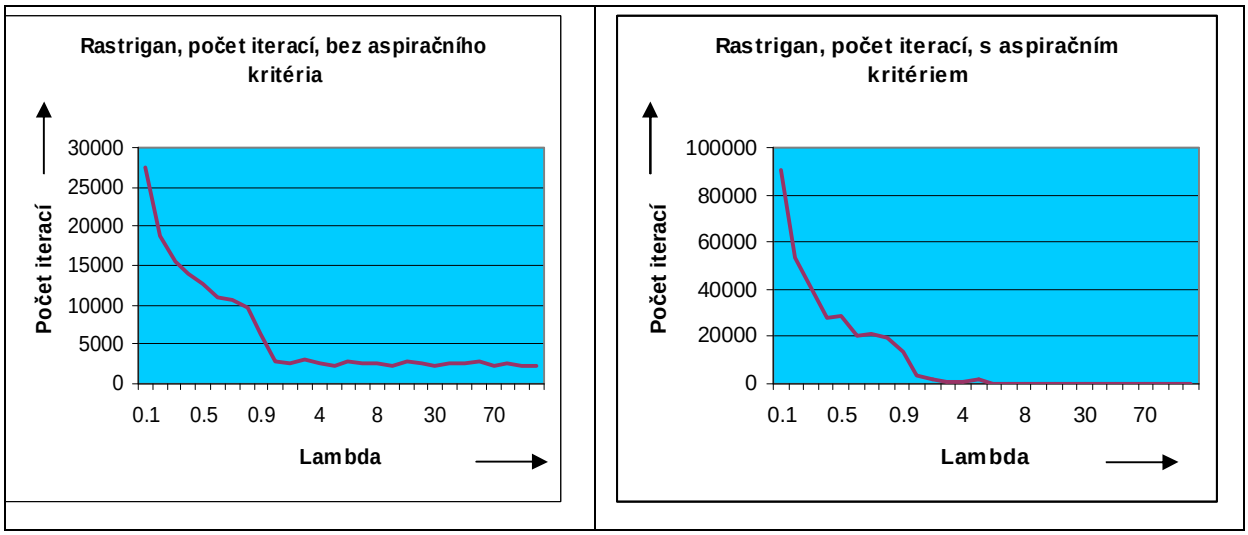
Pokud se týká nejlepšího nalezeného řešení v průběhu algoritmu, je patrné, že PLP s aspiračním kritériem dává lepší výsledky a to především u hodnot parametru λ v rozmezí <0.1,0.9>. Stanovení takové hodnoty parametru λ může být určující pro testování PLP v jiných optimalizačních problémech (např. problém obchodního cestujícího [6] atp.). Z obrázku 2 je patrné, že naměřené hodnoty u PLP s aspiračním kritériem nevykazují takovou rozkolísanost jako u PLP bez aspiračního kritéria. Tento stav u nalezených řešení lze vysvětlit právě nízkou hodnotou parametru λ, protože se penalizace nezvyšuje o tak veliké hodnoty jako u vyšších hodnot parametru, kde je tendence se rychle vymanit z lokálního minima a tudíž existuje reálná možnost toho, že se nenalezne dobré řešení.



Obr. 3 - Porovnání počtu změn nejlepšího nalezeného řešení u 1st de Jong

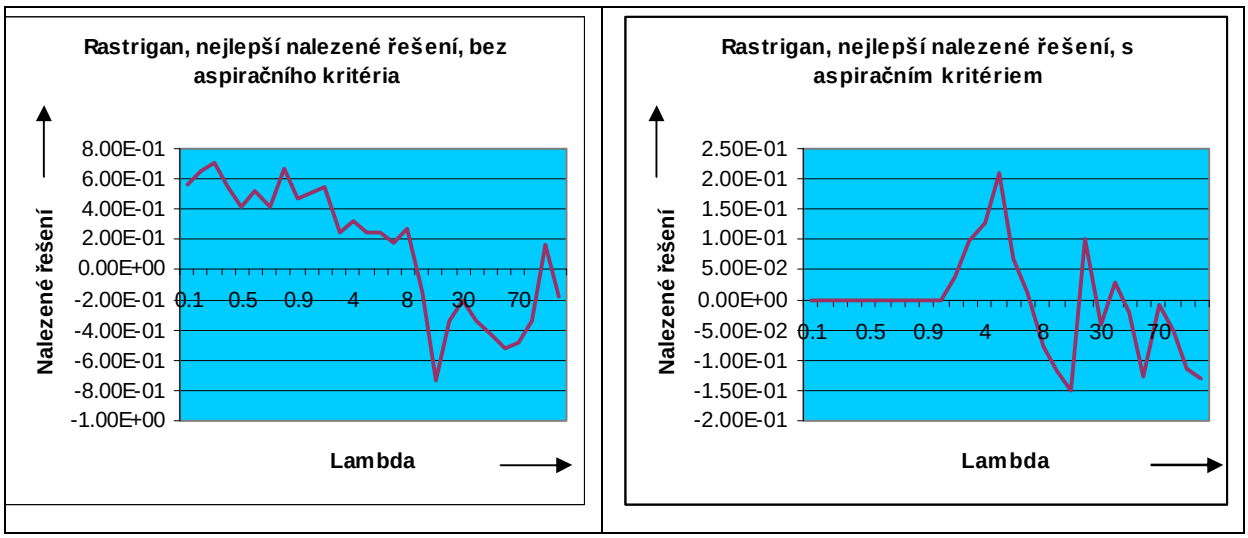
Pro určení kvality PLP s aspiračním kritériem je také vhodné porovnat počet změn nejlepšího nalezeného řešení. V průběhu algoritmu je zaznamenána hodnota nejlepšího nalezeného řešení, v momentu, kdy je nalezeno další jiné lepší řešení je to původní nahrazeno tímto novým a současně je navýšena o jednotku proměnná, která zaznamenává tuto hodnotu. Tato hodnota je zaznamenávána z toho důvodu, že je možné v momentu náhrady nejlepšího možného řešení provádět ještě jiné operace (jako je např. porovnávání hodnot s jinými naměřenými hodnotami a běh algoritmu pak dále upravovat) a tudíž se ušetří jak časová tak i paměťová náročnost algoritmu. Obrázek 3 zobrazuje porovnání těchto naměřených hodnot u obou typů porovnávaných algoritmů. Opět je patrné, že PLP s aspiračním kritériem dává lepší výsledky, a tudíž je možné jej označit za kvalitnější a paměťově méně náročnou variantu lokálního prohledávání.

V průběhu běhu algoritmu na Rastriginově funkce byl opět naměřen větší počet iterací u PLP s aspiračním algoritmem než u druhého algoritmu (obr. 4). U hodnot parametru λ vyšších než 1 se dramaticky snižuje počet iterací a to u obou algoritmů. V takových momentech je pak vhodné využít jeden ze sledovaných algoritmů, jak je patrné z následujících obrázků.



Obr. 4 - Porovnání počtu iterací u Rastriginovy funkce

V případě nejlepšího nalezeného řešení (obr. 5) v průběhu algoritmu i u této funkce je vidět, že PLP s aspiračním kritériem v porovnání s PLP bez aspiračního kritéria poskytuje výsledky, které jsou opět kvalitnější a to především u hodnot parametru λ <0.1,0.9>. Pokud bychom měli porovnat hodnoty počtu iterací a nejlepšího nalezeného řešení, pak i zde nejlépe vychází hodnota parametru λ přibližně na <0.4,0.8>, protože počet iterací je již snesitelný pro výpočet funkce a hodnoty naměřených nejlepších řešení jsou taktéž vhodné proto, abychom je mohli považovat za velmi blízké optimálnímu řešení.



Obr. 5 - Porovnání nejlepších nalezených řešení u Rastrinovy funkce

Z naměřených hodnot je patrné, že algoritmus PLP je silným nástrojem při hledání globálního extrému. A je i patrné, že lepší výsledky dává, pokud je spojený s aspiračním kritériem. PLP s aspiračním kritériem dává obecně stabilnější výsledky u nalezeného globálního extrému, hodnoty nalezeného extrému se více blíží optimu a nejeví takové známky rozkolísanosti jako u PLP bez aspiračního kritéria. Jistou nevýhodou je delší časová náročnost pro výpočet a i s tím spojený větší počet iterací, což je zapříčiněno kontrolou, zda se řešení nachází v aspiračním kritériu či nikoliv. Odměnou za to je však přesnější výsledek. Taktéž počet nahrazených nových lepších řešení je obecně nižší u PLP s aspiračním kritériem.

Závěr

Nemalý vliv na kvalitu nalezeného řešení v průběhu optimalizace různých problémů pomocí PLP má i λ parametr. Pro zjištění jaká velikost tohoto parametru má nejlepší účinek pro nalezení optimálního řešení byla jeho hodnota nastavena na různé hodnoty od hodnoty 0.1 až po hodnotu 100 (2). Ze zobrazených výsledků je patrné, že nejlepších výsledků dosahoval algoritmus v momentech, kdy hodnota tohoto parametru byla nastavena v rozmezí <0.4,1.0>. Velikost tohoto parametru má vliv na hladký průběh algoritmu, kdy při nižších hodnotách má algoritmus jemný průběh, tzn. je dosahováno poměrně stabilních výsledků. Avšak vysokou daní je vyšší časová náročnost. U vyšších hodnot parametru, je dosahováno již horších výsledků, ale v mnoha případech i tyto výsledky jsou kvalitní.

Na proběhnutých experimentech z oblasti testovacích funkcích je také patrné, že lepších výsledků dosahuje algoritmus s využitím aspiračního kritéria než bez něj. Z naměřených hodnot je také patrné, že úspěch aspiračního přesunu není jednoduše spojený pouze s penalizačními podmínkami (jež jsou základem penalizovaného lokálního prohledávání) v rozšířené účelové funkci.

Použitá literatura:

[1] Badeau, P., et al. A Parallel Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows. In Transportation Research - C 5, 1997. s. 109-122.
[2] Gass, S.I., Harris, C.M. Encyclopedia of Operations Research and Management Science centennial edition, druhé vydání, Dodrecht, Kluwer AP, 2004. ISBN 0-7923-7827-X
[3] Glover, F., Taillard, E., de Werra, D. a user's guide to tabu search. In Annals of Operations Research 41, 1993 s. 3 – 28.
[4] Glover, F., Laguna, M. Tabu Search. Norwelll, MA: Kluwer Academic Publishers, 1997. 382 s. ISBN 0-7923-8187-4.
[5]Hansen, P.C. Rank-deficient and discrete ill-posed problems: Numerical aspects of linear inversion, Philadelphia, PA: SIAM, 1998. ISBN 0-8987-1403-6.
[6] Johnson, D. Local Optimisation and the Travelling Salesman Problem. In Proceedings of the 17th Colloquium on Automata Languages and Programming, Lecture Notes in Computer Science, 443, Springer-Verlag, 1990. s. 446-461.
[7] Panuš, J. Using an aspiration criterion and Lambda parameter for local search algorithms.In 7th Annual Ph.D. Conference IMEA 2007. Seč, Chrudim 21st – 22nd May 2007. ISBN 978-80-7194-965-7.
[8] Zelinka, I. Umělá inteligence v problémech globální optimalizace. Praha: BEN – technická literatura, 2002. ISBN 80-7300-069-5.

Kontaktní adresa:

Ing. Jan Panuš
Ústav systémového inženýrství a informatiky
Fakulta ekonomicko-správní
Univerzita Pardubice
Studentská 84
53210 Pardubice
Email: jan.panus@upce.cz
tel.č.: 466036001

Tento článek byl vytvořen za podpory grantu Grantové agentury ČR č. 402/06/0084 Modelování a optimalizace rozhodovacích procesů v městské a regionální správě.