Lokális optimalizálók vonal menti keresés változatainak hatása a GLOBAL eljárásra

Mester Abigél, Bánhelyi Balázs, Pál László

SZTE-TTIK, Számítógépes Optimalizálás Tanszék

XXXII. Magyar Operációkutatás Konferencia
Cegléd, 2017. június 15.

Vázlat

A modulok összekötése
- Építsünk XML-ből

Motiváció

Lokális keresés
- Lokális keresési eljárások

Vonalmenti keresések
- Vonalmenti keresés funkciója
- Vonalmenti keresések

Eredmények
- Tesztelés menete
- Lokális keresések változatai

Publikációk
- Publikációk

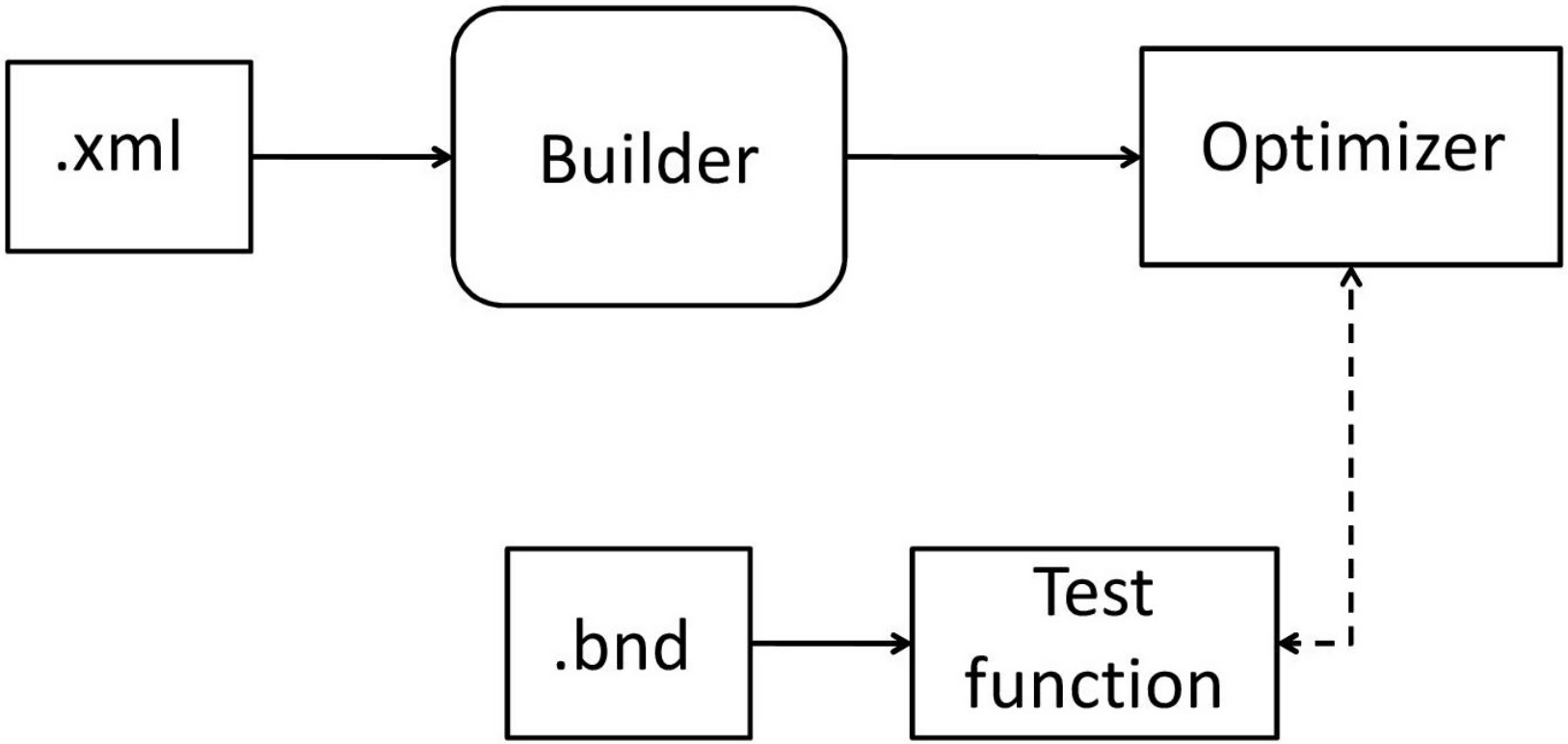
De hogyan is épül meg egy Optimizer?

```
// Creating the clusterizer
ClustrizerBuilder clustrizerBuilder = new ClustrizerBuilder();
clustrizerBuilder.setAlfa(0.01);
Clusterizer<Point> clusterizer = clustrizerBuilder.build();

// Creating the linesearch
LineSearchImpl.LineSearchBuilder lineSearchBuilder =
        new LineSearchImpl.LineSearchBuilder();
LineSearchImpl lineSearchOptimizer = lineSearchBuilder.build();

// Creating the local search
UnirandiCLS.UnirandiCLSBuilder unirandiBuilder =
        new UnirandiCLS.UnirandiCLSBuilder();
unirandiBuilder.setLineSearchOptimizer(lineSearchOptimizer);
unirandiBuilder.setRelativeConvergence(0.00000001);
unirandiBuilder.setMaxFunctionEvaluations(100000000);
LocalOptimizer<Vector> localOptimizer = unirandiBuilder.build();

//creating the global search
Global.GlobalBuilder globalBuilder = new Global.GlobalBuilder();
globalBuilder.setClusterizer(clusterizer);
globalBuilder.setLocalOptimizer(localOptimizer);
globalBuilder.setNewSampleSize(400);
globalBuilder.setSampleReducingFactor(0.03999);
Global global = globalBuilder.build();

...

global.run();
```

XML példa

```
<?xml version="1.0"?>
<Global class="org.uszeged.inf.optimization.algorithm.optimizer.global.Global">
    <NewSampleSize type="long"> 400 </NewSampleSize>
    <SampleReducingFactor type="double"> 0.03999 </SampleReducingFactor>
    <LocalOptimizer class="optimizer.local.parallel.UnirandiCLS">
        <MaxFunctionEvaluations type="long"> 100000000 </MaxFunctionEvaluations>
        <RelativeConvergence type="double"> 0.00000001 </RelativeConvergence>
        <LineSearchFunction class="optimizer.line.parallel.Fminbnd">
        </LineSearchFunction>
    </LocalOptimizer>
    <Clusterizer class="clustering.GlobalSingleLinkageClusterizer">
        <Alpha type="double"> 0.01 </Alpha>
    </Clusterizer>
</Global>
```

A modulok összekötése



Motiváció

- A lokális keresés működése:
  - Véletlenszerű pontok $\rightarrow$ függvénykiértékelések
  - Klaszterek a mintapontokból
  - Nem klaszterezhető pontok $\rightarrow$ lokális keresés
- Lokális keresés költséges
- Modulokkal új változatok gyors tesztelése

Unirandi

- LineSearch algoritmusnak generál irányt.
- Sikertelenség esetén ellenkező irány vizsgálata.
- Még párszor próbálkozik a korábbi lépésközzel, hogy talál-e esetleg más jobb irányt (eredeti változatban 2-szer).
- Ha eddig sikertelen a lépegetés, akkor a lépéstávolságot csökkentjük.
- Megállási feltétel:
  megfelelően kicsi függvényérték változás.

Rosenbrock

- Megszünteti a keresési irány korlátozottságát - tengelyek forgatása.
- Az egyik tengelyt a legkedvezőbb irányba forgatja és így keres tovább.
- Keresés:
  - ha sikeres a lépés: növeli a lépés nagyságát
  - ha sikertelen a lépés: csökkenti a nagyságot, ellenkező irányba fordul
  - megállási feltétel: legalább 1 sikertelen függvénykiértékelés mindegyik koordinátairányban
- Ezután ismét a tengelyeket forgatja.
- A megállási feltételt a transzformáció után vizsgálja meg.

NUnirandi

- Javítás rosszul kondicionált problémákra.
- Jónak tűnő irányt követi - utolsó 2 optimumot tárolja.
- 2 újabb vonalmenti keresés.
- Egyébként Unirandi működését követi.

Vonalmenti keresés szerepe

- LineSearch metódus kiemelése.
- Lehetőség új módszerekkel való kísérletezgetésre.
- Linesearch a lokális keresőktől kapja meg az indulási pontot és a kiértékelendő irányt.
- A futása végén ő maga is értékekkel tér vissza a lokális keresők számára.

Vonalmenti keresés az XML-ben

```
<?xml version="1.0"?>
<Global class="org.uszeged.inf.optimization.algorithm.optimizer.global.Global">
    <NewSampleSize type="long"> 400 </NewSampleSize>
    <SampleReducingFactor type="double"> 0.03999 </SampleReducingFactor>
    <LocalOptimizer class="optimizer.local.parallel.UnirandiCLS">
        <MaxFunctionEvaluations type="long"> 100000000 </MaxFunctionEvaluations>
        <RelativeConvergence type="double"> 0.00000001 </RelativeConvergence>
        <LineSearchFunction class="optimizer.line.parallel.Fminbnd">
        </LineSearchFunction>
    </LocalOptimizer>
    <Clusterizer class="clustering.GlobalSingleLinkageClusterizer">
        <Alpha type="double"> 0.01 </Alpha>
    </Clusterizer>
</Global>
```

Duplázva lépegető

- Unirandiból emeltük ki, és fejlesztettük tovább.
- Megkapja: alappont, irány, lépéshossz.
- Új alappontot generál, ami ha jobb, ezt használja tovább.
- Sikeres lépésekben duplázza a lépésközt
  $\rightarrow$ innen a neve.
- Megállási feltétel: a kapott értékünk csökkenése megáll, ekkor optimumban vagyunk, vagy túlhaladtunk azon.

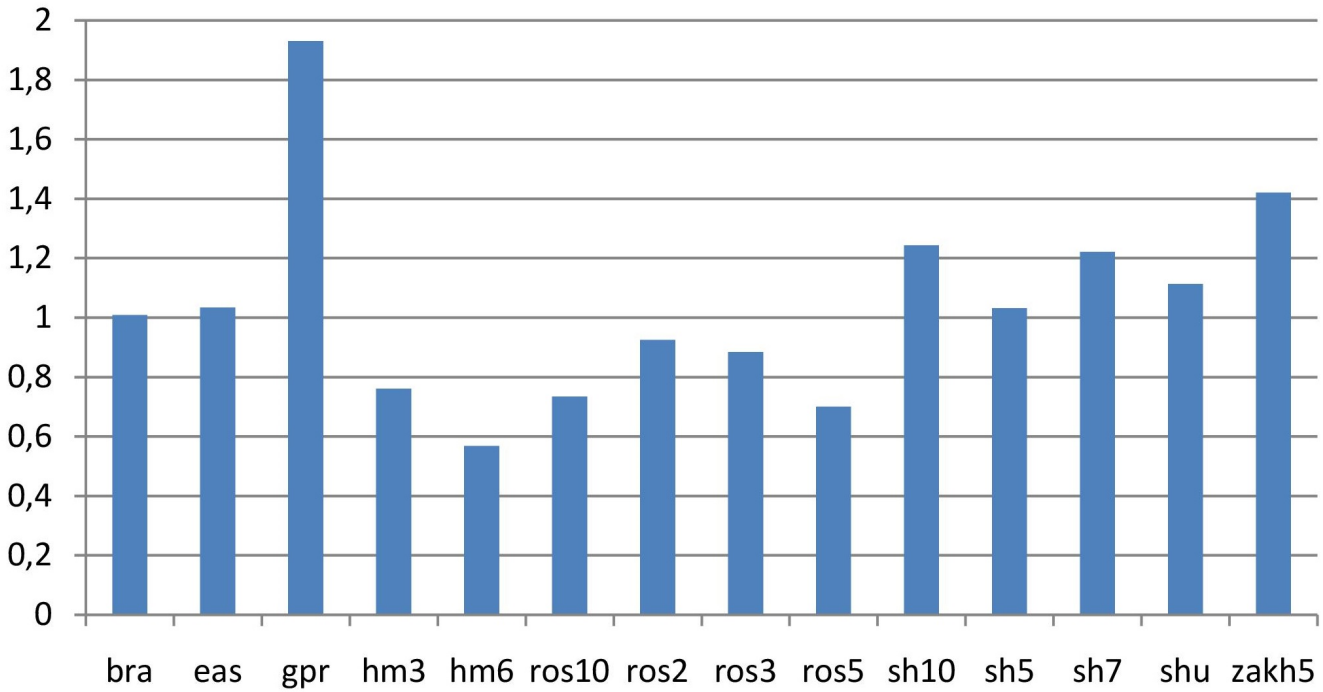
Függvény illesztés

- Ha a duplázva lépegető sikertelen
  $\rightarrow$ egyenes mentén további pontok.
- Legalább két/négy lépés után:
  - utolsó három/öt pontra illeszthetünk egy másodfokú/negyedfokú polinomot
  - lefedi a célfüggvény értékeket
- Minimumhelyen újabb kiértékelés. Ha sikeres, akkor azzal tér vissza a LineSearch metódusunk.
- Magasabb fokú polinomok minimumhelyeihez komolyabb matematika háttér szükséges.

Tesztelés

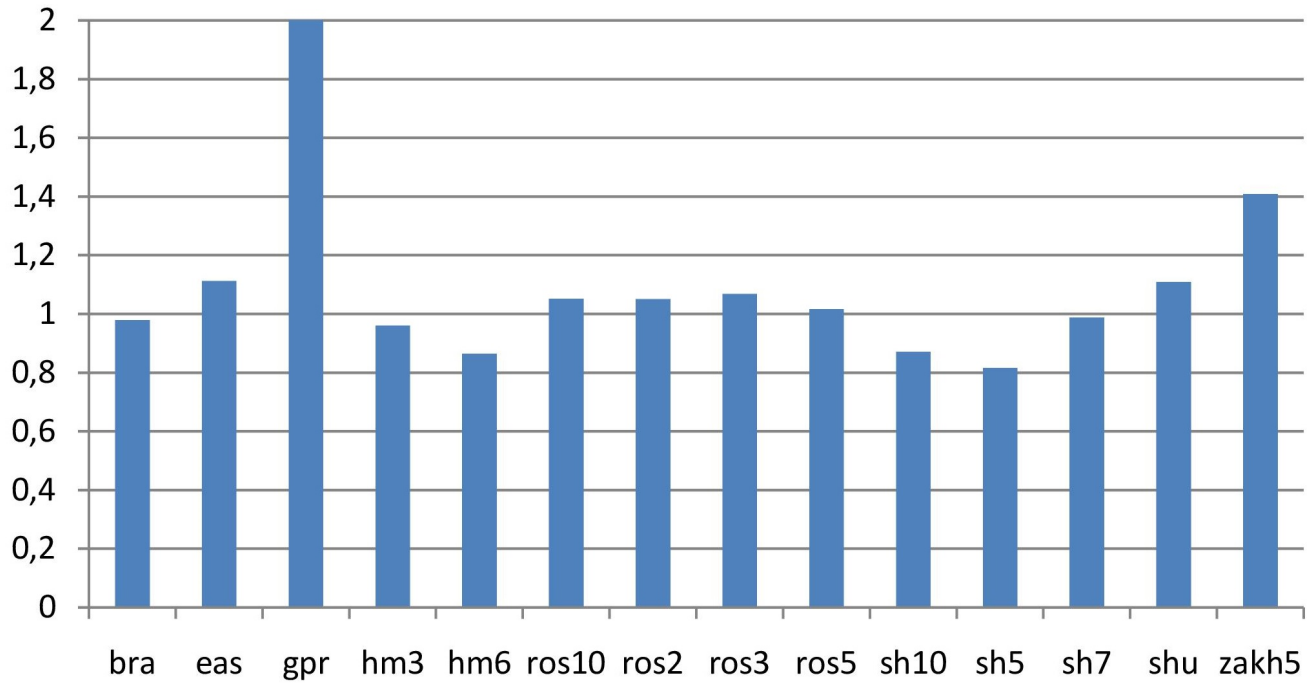
- Az optimalizálásban szokásos tesztfüggvények (bra; eas; gpr; hm3/6; ros2/3/5/10; sh5/10; zakh5).
- Kombinációkkal 6 algoritmus.
- 6 tizedes jegyes pontosság.
- Futásidő vizsgálat.
- 100 futás átlaga.
- Eredeti vonalmenti keresővel összehasonlítva:
  - eredeti (beépített vonalmenti kereső) lokális keresővel kapott eredményeket osztottuk
  - a módosított és lokális keresők és illesztések kombinációjával.

Unirandi másodfokú illesztővel



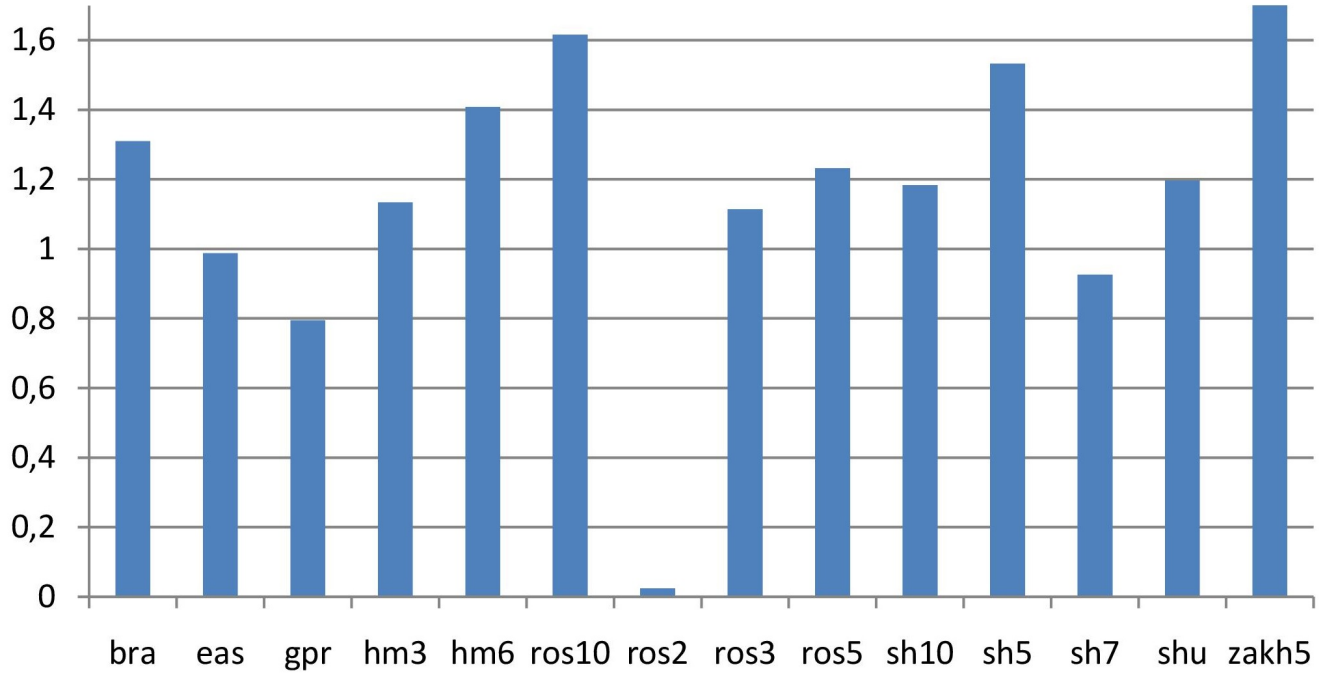
Az Unirandival 3 alappontra illesztve esetek több, mint felén javulást hoz

Unirandi negyedfokú illesztővel



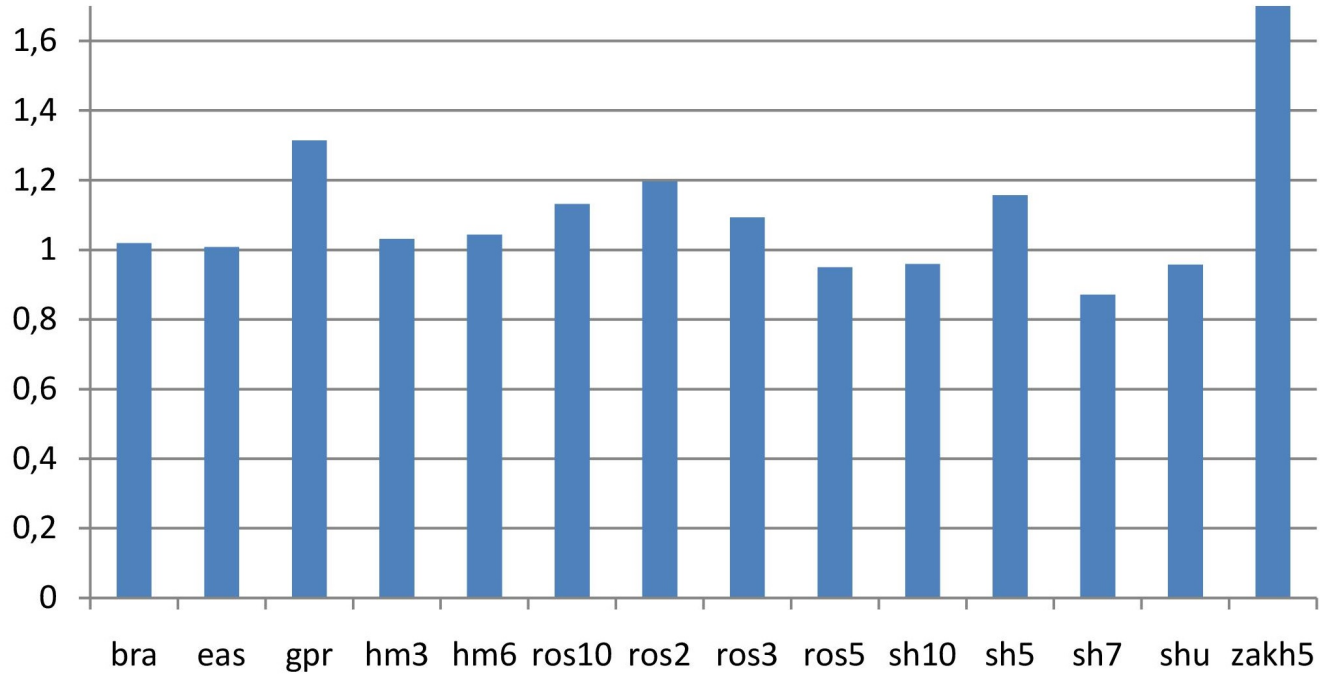
Az Unirandival 5 alappontra illesztve esetek több, mint felén javulást hoz

Rosenborck másodfokú illesztővel



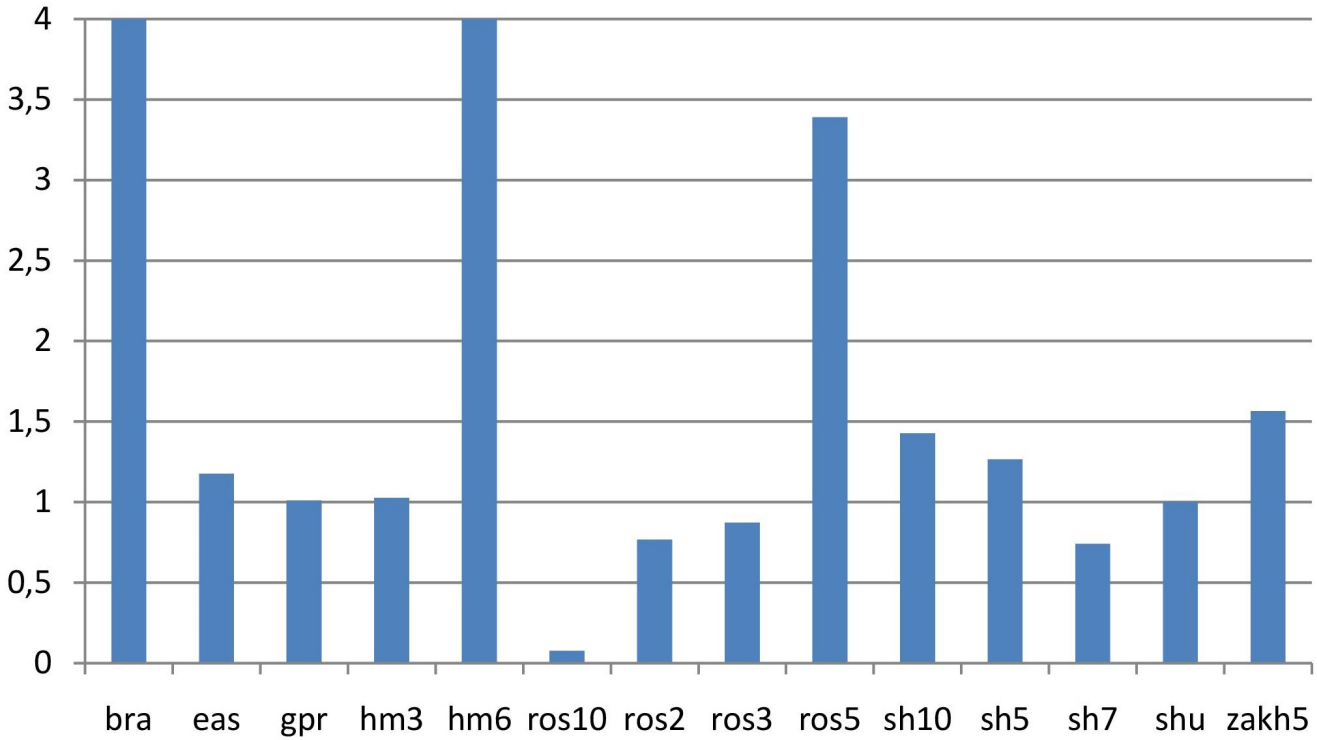
A Rosenbrockkal 3 alappontra illesztve esetek több, mint 70%-án javulást hoz

Rosenborck negyedfokú illesztővel



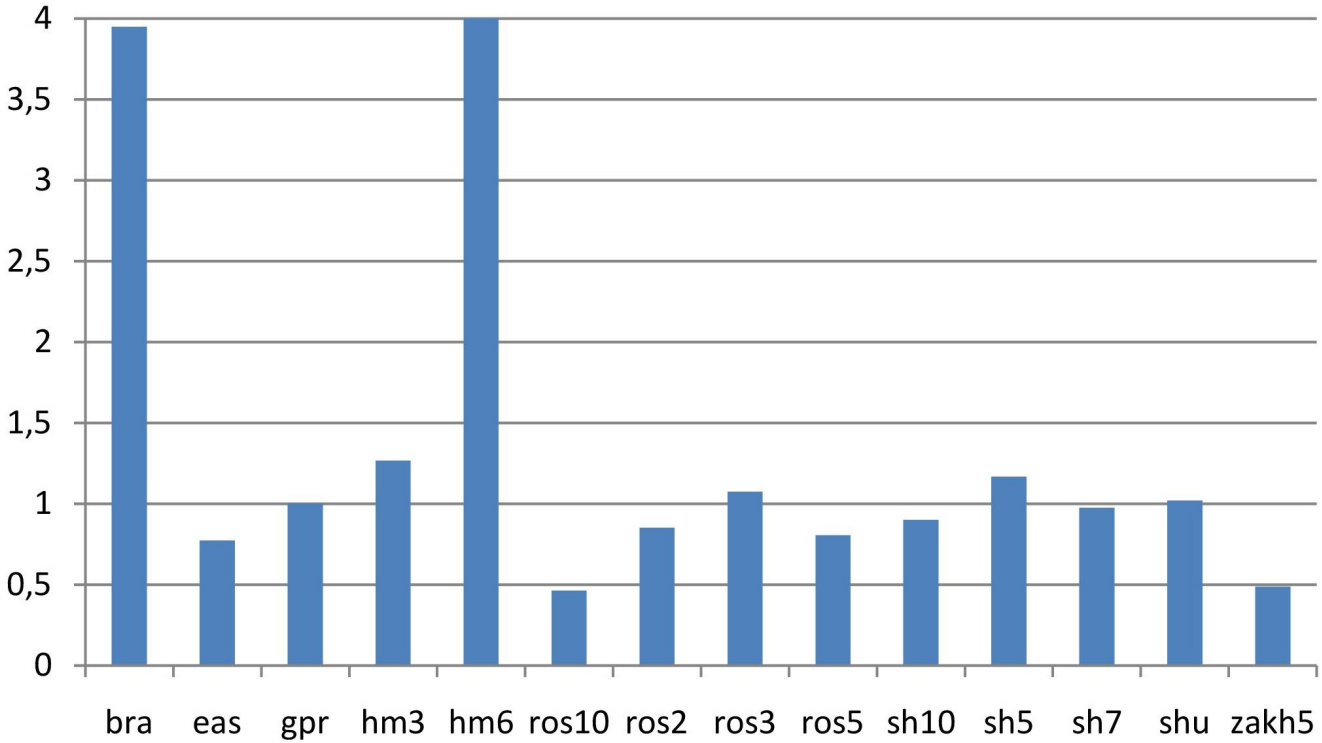
A Rosenbrockkal 5 alappontra illesztve esetek több, mint 75%-án javulást hoz

NUnirandi másodfokú illesztővel



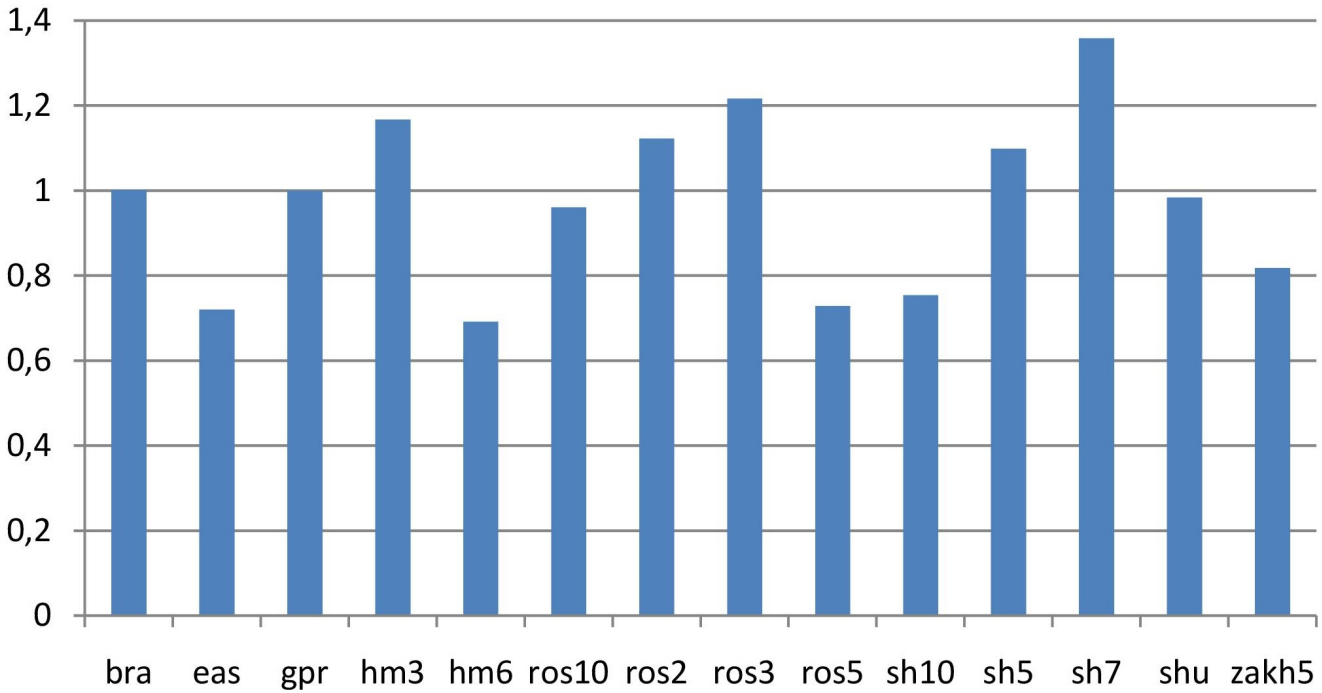
Az NUnirandival 3 alappontra illesztve esetek több, mint 70%-án javulást hoz

NUnirandi negyedfokú illesztővel



Az NUnirandival 5 alappontra illesztve esetek több, mint felén javulást hoz

Próbálkozás további illesztésekkel



Az NUnirandival tovább illesztve esetek felén javulást hoz
$\rightarrow$ nincs érdemi javulás

Konklúzió

- Olyan környezetet alkottunk, melyben könnyedén lehet tesztelni a különböző optimalizáló algoritmusokat (Builder, Factory, XML).
- Unirandi és Rosenbrock, valamint NUnirandi Globalhoz illeszkedő objektumorientált változatai, melyek a LineSearch metódust, mint beépített modul használják.
- Különféle vonalmenti keresők (duplázva lépegető, másod,-negyedfokú illesztő eljárások).
- Új kombinációk gyors tesztelése, ehhez pedig a variációkat hatékonyan lehet összeállítani.

Publikációk

José-Oscar H. Sendín, Julio R. Banga, and Tibor Csendes:

Extensions of a Multistart Clustering Algorithm for Constrained Global Optimization Problems.
Industrial & Engineering Chemistry Research **48(6)**, 3014–3023, 2009

Pál László:

Globális optimalizálási algoritmusok korlátos feladatokra.
Doktorandusz fórum, Csíkszereda, 120–127, 2010

Tibor Csendes, László Pál, J. Oscar H. Sendín, and Julio R. Banga:

The GLOBAL optimization Method Revisited.
Optimization Letters **2**, 445–454, 2008

László Pál and Tibor Csendes:

An improved stochastic local search method in a multistart framework.
Applied Computational Intelligence and Informatics (SACI), Timisoara, 2015, 117–120, 2015