Use of Surrogate-based Model Search for Parallel Blackbox Optimization

Optimization Days 2019
Blackbox optimization session (TD6)
May 14, 2019

Stéphane Alarie
Institut de recherche d'Hydro-Québec

Hydro Québec

Outline

> **Problem context**
> **Proposed solution**
> **Numerical results**
> **Conclusion**

This is a joint work with **Bastien Talgorn** and **Michael Kokkolaras** from McGill University

Problem context

Problem

> **Let (P) be**

$$\min_{x\in\Omega} f(x)$$
$$\text{s.t. } c_j(x) \le 0, \quad j = 1, 2, \ldots, m$$

> **We assume that $f(x)$ and $c_j(x)$ are evaluated by simulators** (or others)

> **(P) is solved by blackbox approaches here, in particular, with NOMAD (MADS)**

MADS

Algorithm 1: High-level presentation of MADS

Initialization: Let $x_0 \in \mathbb{R}^n$ be an initial point and set the iteration counter $k \leftarrow 0$

Main loop:

repeat

- SEARCH on the mesh to find a better solution than x_k
- **if** *the* SEARCH *failed* **then**
  - POLL on the mesh to find a better solution than x_k
- **if** *a better solution than* x_k *was found by either the* SEARCH *or the* POLL **then**
  - call it x_{k+1} and coarsen the mesh
- **else**
  - set $x_{k+1} = x_k$ and refine the mesh
- Update parameters and set $k \leftarrow k + 1$

until *Stopping criteria is satisfied*;

* S. Le Digabel, C. Tribes, V. Rochon Montplaisir and C. Audet, *NOMAD User Guide Version 3.9.1*, 2018.

Evaluation distribution with Blocks

> **POLL**
  - natively generate 2N points at each iteration (excepted if some already in cache)

> **SEARCH**
  - mix of search strategies (speculative, quadratic, VNS, Nelder-Mead, LHS)
  - usually one point, one search after the other (mostly sequential $\Rightarrow$ bad use of CPUs + lost of time)
  - except LHS, which can generate 2N points (good for CPUs, but quickly useless $\Rightarrow$ lost of time)

What is wanted…

- > **CPUs are unlimited**
- > **The SEARCH should be as effective as the POOL in the use of CPUs**
- > **But this, in a smart manner**
  - not randomly (*cf.* LHS)
  - want to identify new potential optima and/or accelerate the overall search

New model search for parallel computing

Proposed SEARCH

> **Let’s be (S), a surrogate of (P)**

> **Let’s solve (S) by blackbox techniques to get a large number of points** ($|S| > 5000$)
  - LHS to get points widely spread in Ω
  - VNS to explore multiple local optima

> **Let’s take q diversified points** (2N) **from S and evaluate them** (simulators or others)

Resulting MADS

Algorithm 2 : The MADS algorithm with model searches.

[1] **Initialization**
$t \leftarrow 0$
Set initial poll and mesh sizes $\Delta_0^p \geq \Delta_0^m > 0$
Initialize $\mathbf{X}_0$ with starting points
Evaluate $\{f(\mathbf{x}), c_1(\mathbf{x}), c_2(\mathbf{x}), ..., c_m(\mathbf{x})\}\ \forall \mathbf{x} \in \mathbf{X}_0$

[2] **Model SEARCH**
Use $\mathbf{X}_t$ to build $\hat{f}$ and $\{\hat{c}_j\}_{j \in J}$
Solve surrogate problem **(S)**
$\mathbf{S}_t \leftarrow$ Set of points evaluated with surrogate model while solving **(S)**
$\mathbf{T}_t \leftarrow$ Selection of q points in $\mathbf{S}_t$
$\mathbf{T}_t \leftarrow$ Projection of the points of $\mathbf{T}_t$ onto mesh $\mathcal{M}_t$
Parallel evaluation of $\{f(\mathbf{x}), c_1(\mathbf{x}), c_2(\mathbf{x}), ..., c_m(\mathbf{x})\}\ \forall \mathbf{x} \in \mathbf{T}_t$
If success, `goto` [4]

[3] **POLL**
Build poll set $\mathbf{P}_t$
Sort $\mathbf{P}_t$ according to $\hat{f}$ and $\{\hat{c}_j\}_{j \in J}$
Parallel evaluation of $\{f(\mathbf{x}), c_1(\mathbf{x}), c_2(\mathbf{x}), ..., c_m(\mathbf{x})\}\ \forall \mathbf{x} \in \mathbf{P}_t$

[4] **Updates**
$t \leftarrow t + 1$
Update Δ_t^m, Δ_t^p, $\mathbf{x}_t^*$ and $\mathbf{X}_t$
If no stopping condition is met, `goto` [2]

Selection of interesting points $s \in S$

> **6 ways to select points** (not yet in $X \cup T$)

- Best point s
- Most distant point s from $X \cup T$
- Best point s at $d_{\min}$ from $X \cup T$
- Best feasible point s
- Most isolated point s
- Point s in the most populated area

> **If one fails, go to the next one**

> **Repeated until to get q points in T**

Preliminary definitions

> **Constraint penalty:** $h(x) = \sum_{j=1}^{m} \max\{0, c_j(x)\}^2$

> **Order:** $x \prec y \Leftrightarrow \begin{cases} h(x) < h(y) \text{ whatever } f(x) \text{ and } f(y) \\ h(x) = h(y) \text{ and } f(x) < f(y) \end{cases}$

> **Distance:** $d(X,Y) = \min_{x \in X} \min_{y \in Y} \| x - y \|_2$

$d(x,Y) > 0 \iff x \notin Y$

$d(X,\varnothing) = d(\varnothing,\varnothing) = +\infty$

Methods 1 and 2

> **Best point s**

$$\text{Find } s^* \text{ s.t. } s^* \prec s \quad \forall s \in S$$
$$\text{and } d(s^*, X \cup T) > 0$$

> **Most distant point s from $X \cup T$**

$$\text{Find } s^* \text{ s.t. } d(s^*, X \cup T) > d(s, X \cup T) \quad \forall s \in S$$

Method 3

> **Best point s at $d_{\min}$ from $X \cup T$**

$$\text{Find } s^* \text{ s.t. } s^* \prec s \text{ and } d(s^*, X \cup T) \geq d_{\min} \quad \forall s \in S$$

where $d_{\min} \leftarrow 0$ as initial value

and $d_{\min} \leftarrow d_{\min} + \Delta_{\text{mesh}}$ the next application

> **May fail if $d_{\min}$ becomes too large…**

Method 4

> **Best feasible point s**

$$\text{Find } s^* \text{ s.t. } c_{\max}(s^*) \le c_{\text{margin}} \text{ and } f(s^*) < f(s) \quad \forall s \in S$$

where $c_{\max}(s) = \max_{j=1,\ldots,m} c_j(s)$

and $c_{\text{margin}} \leftarrow \max_{\substack{s\in S \\ c_{\max}(s)<0}} c_{\max}(s)$ as initial value

and $c_{\text{margin}} \leftarrow 2c_{\max}(s^*)$ the next application

> **May fail if c_{margin} becomes too small…**

Method 5

> **Most isolated point s**

$$\text{Find } s^* \text{ s.t. } n_{\text{isolated}}(s^*) > n_{\text{isolated}}(s) \quad \forall s \in S$$

where $n_{\text{isolated}}(s) = \text{card}\{s' : s' \in S, d(s,s') < d_{\text{isolated}}(s)\}$

and $d_{\text{isolated}}(s) = \min_{\substack{s'\in S \\ s' \prec s}} d(s,s')$ where s' is the closest better point from s

Method 6

> **Point s in the most populated area**

$$\text{Find } s^* \text{ s.t. } n_{\text{density}}(s^*) > n_{\text{density}}(s) \quad \forall s \in S$$

where

$$n_{\text{density}}(s) = \text{card}\{s' : s' \in S, d(s,s') < d(s, X \cup T)\}$$

Numerical Results

Tested algorithms

Algorithms	Type of Search	Selection Methods
MADS	Ortho 2N & Block(q)	N.A.
Multi-Start	q MADS & Block(1)	N.A.
LH Search	Latin Hypercube(q)	N.A.
Lowess-A	Model Search(q)	{0,1}
Lowess-B	Model Search(q)	{2,3,4,5}

Lowess[1] is taken from SGTELIB[2]:

```
TYPE Lowess DEGREE 1 RIDGE 0 SHAPE_COEFF OPTIM KERNEL_TYPE OPTIM
```

[1] B. Talgorn et al., Optimization and Engineering, 19(1), 2018.
[2] https://github.com/bastientalgorn/sgtelib.

Design optimization problems

Problems	# var.	# const.	Int. var.?	Inf. Bounds?
TCSD[1]	3	4	No	No
Vessel[1]	4	4	No	No
Welded[1]	4	6	No	No
Solar1[2]	9	5	Yes	Yes
Solar7[2]	7	6	Yes	Yes

TCSD: min spring weight under mechanical constraints

Welded: min cost of a beam under stress and constraints

Vessel: compressed air storage tank design

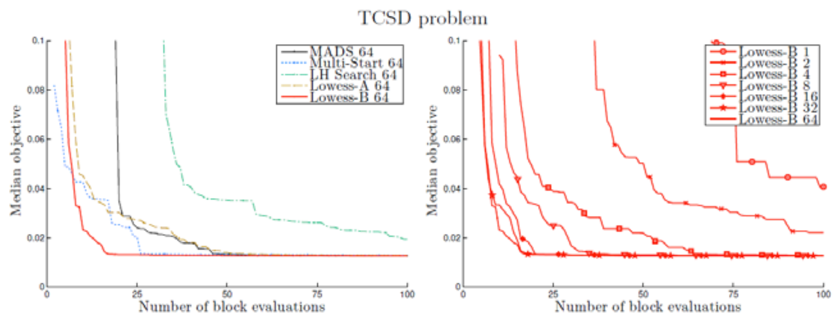
Solar1: max energy received

Solar7: max energy transfered to the molten salt

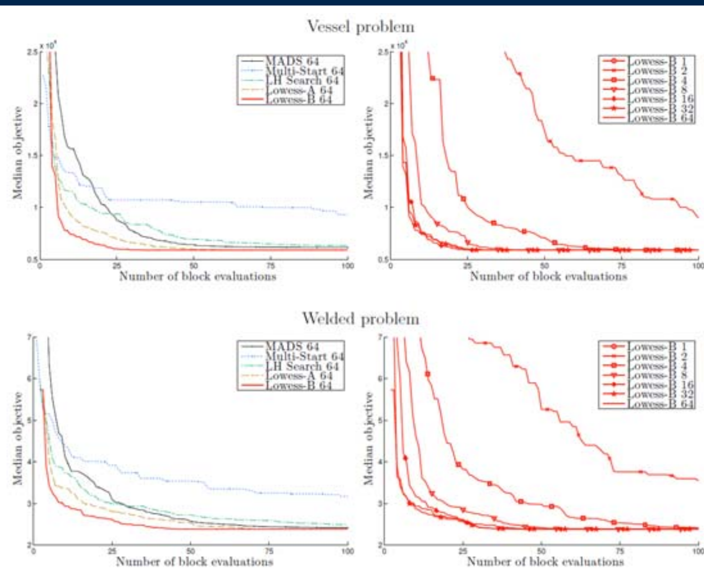
[1] H. Garg, Journal of Industrial and Management Optimization, 10(3), 2014.
[2] M.L. Garneau, Master's thesis, École Polytechnique de Montréal, 2015.

Results

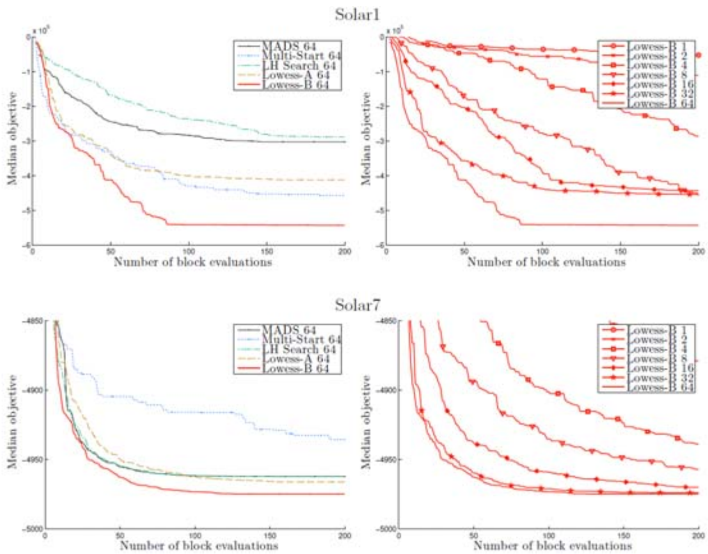
- For each prob / solver, 50 optimization runs from LH
- 100 blocks for TCSD, Vessel and Welded
  200 blocks for Solar 1 and Solar7
- Block size for all runs, $q \in \{1, 2, 4, 8, 16, 32, 64\}$



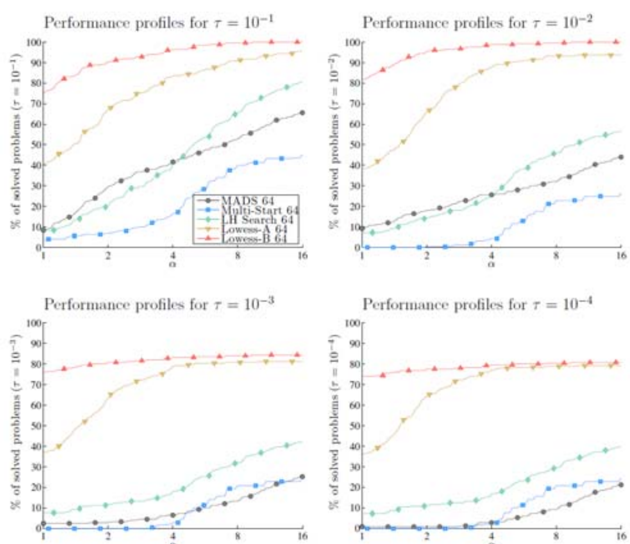
Results



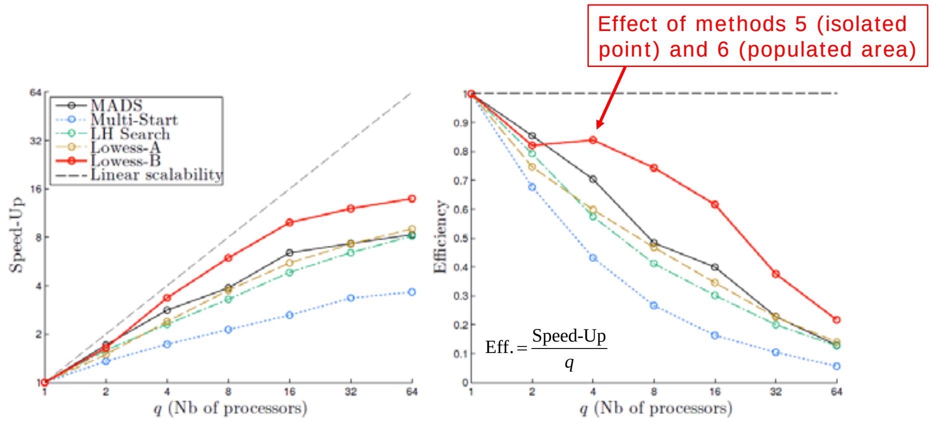
Results



Performance profiles



Scalability analysis



Conclusion

Conclusion

> **Lowess-B outperforms all other solvers**
  - seems to be due to methods 5 and/or 6
  - they are more about the points visited along the optimization than the surrogate itself
  - they are not about the value of $f(s)$ and $c_j(s)$, but how an optimization process spent time and effort to explore deeply some areas of (S)

> **This is new and deserves attention...**