Importance of optimization techniques for the social sciences

Authors: **Linas Stripinis**, dr. Remigijus Paulavičius, prof. dr. (HP) Julius Žilinskas

Modelling and Simulation of Social-Behavioural Phenomena in Creative Societies
September 18–20, 2019
Vilnius, Lithuania

Motivation

Algorithm: DIRECT (DIviding RECTangles)

Jones, D. R., Perttunen, C. D., and Stuckman, B. E. (1993).
Lipschitzian optimization without the Lipschitz constant.
Journal of Optimization Theory and Applications, 79(1):157–181.

Popular Google scholar: 1918 citations (2019/09/08)

Motivation

Algorithm: DIRECT (DIviding RECTangles)

Jones, D. R., Perttunen, C. D., and Stuckman, B. E. (1993).
Lipschitzian optimization without the Lipschitz constant.
Journal of Optimization Theory and Applications, 79(1):157–181.

Popular Google scholar: 1918 citations (2019/09/08)

Simple and has only one algorithmic parameter ε that needs to be adjusted to make the algorithm work well on a given problem.

Motivation

Algorithm: DIRECT (DIviding RECTangles)

Jones, D. R., Perttunen, C. D., and Stuckman, B. E. (1993).
Lipschitzian optimization without the Lipschitz constant.
Journal of Optimization Theory and Applications, 79(1):157–181.

Popular Google scholar: 1918 citations (2019/09/08)

Simple and has only one algorithmic parameter ε that needs to be adjusted to make the algorithm work well on a given problem.

Converge to the global optimum, if the objective function is continuous or at least continuous in the neighbourhood of a global optimum.

Motivation

Algorithm: DIRECT (DIviding RECTangles)

Jones, D. R., Perttunen, C. D., and Stuckman, B. E. (1993).
Lipschitzian optimization without the Lipschitz constant.
Journal of Optimization Theory and Applications, 79(1):157–181.

Popular Google scholar: 1918 citations (2019/09/08)

Simple and has only one algorithmic parameter ε that needs to be adjusted to make the algorithm work well on a given problem.

Converge to the global optimum, if the objective function is continuous or at least continuous in the neighbourhood of a global optimum.

Suitable for black-box optimization.

1. Box constraints.
2. General constrains.
3. Hidden constraints.

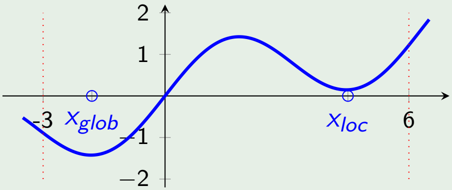
Outline

Global (Lipschitz) Optimization Problem (GOP)

We consider a box-constrained (GOP) of the form:

$$\begin{aligned} &\min && f(\mathbf{x}), \quad f : \mathbb{R}^n \to \mathbb{R} \\ \text{s.t. } &\mathbf{x} \in \mathbb{D} : && \mathbf{l} \le \mathbf{x} \le \mathbf{u}, \\ & && \mathbf{x}, \mathbf{l}, \mathbf{u} \in \mathbb{R}^n. \end{aligned}$$

$$\min_{\mathbf{x} \in [-3,6]} \quad 0.25\mathbf{x} + sin(\mathbf{x})$$



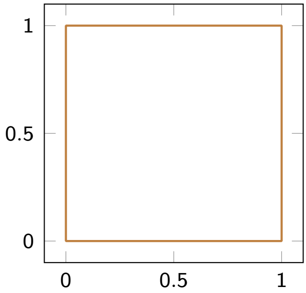
Objective function f satisfies the *Lipschitz condition*

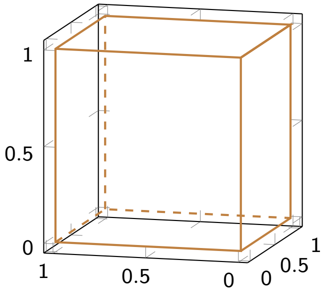
$$|f(\mathbf{x}) - f(\mathbf{y})| \le L\|\mathbf{x} - \mathbf{y}\|, \quad \mathbf{x}, \mathbf{y} \in \mathbb{D}, \quad 0 < L < \infty,$$

with unknown *Lipschitz constant* L.

DIRECT algorithm: main steps

1 step: *Scale* $\mathbb{D} \rightarrow \overline{\mathbb{D}} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq x_i \leq 1,\ i = 1, \dots, n\}$

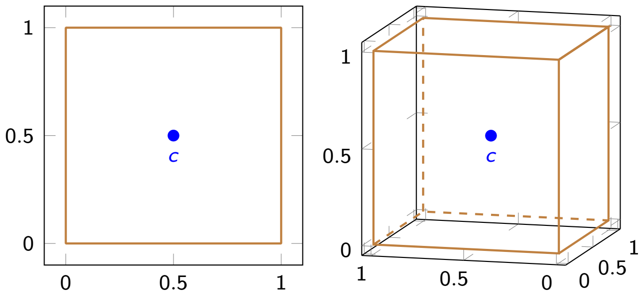




DIRECT algorithm: main steps

1 step: *Scale* $\mathbb{D} \rightarrow \overline{\mathbb{D}} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq x_i \leq 1, \ i = 1, \ldots, n\}$

2 step: *Evaluate f* at ● - *the center point $c \in \overline{\mathbb{D}}$*

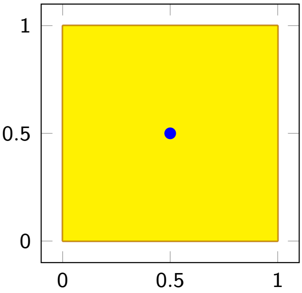


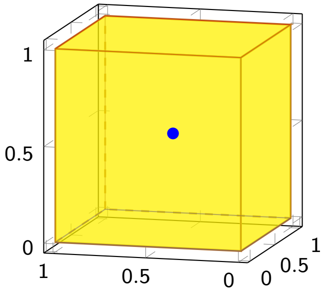
DIRECT algorithm: main steps

1 step: *Scale* $\mathbb{D} \rightarrow \overline{\mathbb{D}} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq x_i \leq 1,\ i = 1, \dots, n\}$

2 step: *Evaluate* f at ● - *the center point* $c \in \overline{\mathbb{D}}$

3 step: *Identify* (and *select*) ■ - *potentially optimal hyper-rectangles*





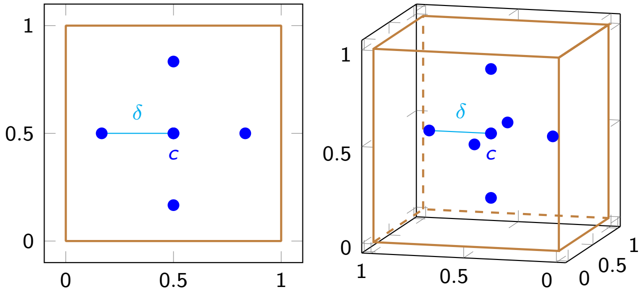
DIRECT algorithm: main steps

1 step: *Scale* $\mathbb{D} \to \overline{\mathbb{D}} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq x_i \leq 1,\ i = 1, \dots, n\}$

2 step: *Evaluate f* at ● - *the center point* $c \in \overline{\mathbb{D}}$

3 step: *Identify* (and *select*) ■ - *potentially optimal hyper-rectangles*

4 step: *Sample f* at the points ● - $c \pm \delta$ (where $\delta = 1/3$ maximal side length)



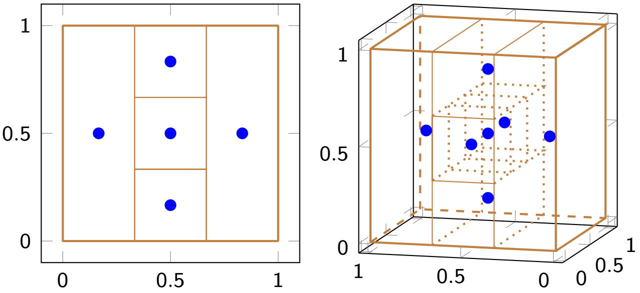
DIRECT algorithm: main steps

1 step: *Scale* $\mathbb{D} \to \overline{\mathbb{D}} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq x_i \leq 1,\ i = 1, \dots, n\}$

2 step: *Evaluate* f at ● - *the center point* $c \in \overline{\mathbb{D}}$

3 step: *Identify* (and *select*) ■ - *potentially optimal hyper-rectangles*

4 step: *Sample* f at the points ● - $c \pm \delta$ (where $\delta = 1/3$ maximal side length) and *divide* (trisect) along all dimensions with the maximum side length.



DIRECT algorithm: main steps

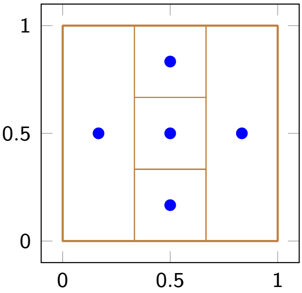
1 step: *Scale* $\mathbb{D} \to \overline{\mathbb{D}} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq x_i \leq 1,\ i = 1, \dots, n\}$

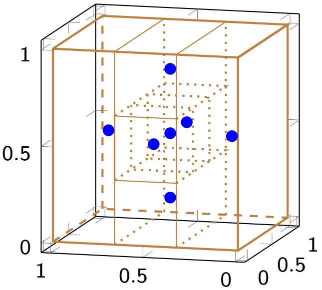
2 step: *Evaluate* f at ● - *the center point* $c \in \overline{\mathbb{D}}$

3 step: *Identify* (and *select*) ■ - *potentially optimal hyper-rectangles*

4 step: *Sample* f at the points ● - $c \pm \delta$ (where $\delta = 1/3$ maximal side length) and *divide* (trisect) along all dimensions with the maximum side length.

Repeat: *3-4* Steps until satisfied some *stopping criteria*.





Schemes for selection of potentially optimal hyper-rectangles

Potentially optimal hyper-rectangles (POH) in **DIRECT**

Definition 1 (Potentially optimal hyper-rectangles)

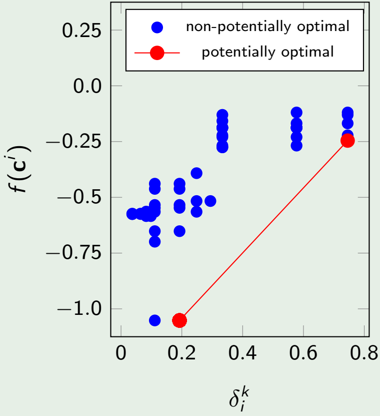
Let $\mathbf{c}^i$ denote the center sampling point and δ_i be a measure (distance, size) of the hyper-rectangle D_k^i. Let $\varepsilon > 0$ be a positive constant and $f_{\min}$ be the best currently known value of the objective function. A hyper-rectangle $D_k^j, j \in \mathbb{I}_k$ is said to be potentially optimal if there exists some rate-of-change (Lipschitz) constant $\tilde{L} > 0$ such that

$$
\begin{aligned}
f(\mathbf{c}^j) - \tilde{L}\delta_j &\leq f(\mathbf{c}^i) - \tilde{L}\delta_i, \quad \forall i \in \mathbb{I}_k, && (1)\\
f(\mathbf{c}^j) - \tilde{L}\delta_j &\leq f_{\min} - \varepsilon|f_{\min}|, && (2)
\end{aligned}
$$

where the measure of the hyper-rectangle is

$$\delta_i = \frac{1}{2}\|\mathbf{b}^i - \mathbf{a}^i\|_2. \quad (3)$$

Geometric interpretation on Shekel 5 test problem in the fifth iteration



New selection strategy for POH in proposed **DIRECT-G**

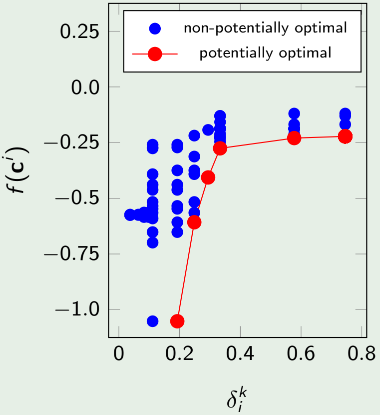
Definition 2 (Potentially optimal hyper-rectangles)

- **Step 1** Find an index $j \in \mathbb{I}_k$ and a corresponding hyper-rectangle D_k^j, such that

$$D_k^j = \arg\max_j \{ l_k^j : j = \underset{i \in \mathbb{I}_k :\ l_k^{\min} \leq l_k^i \leq l_k^{\max}}{\arg\min} \{ f(\mathbf{c}^i) \} \}. \tag{4}$$

- **Step 2** Set $l_k^{\min} = l_k^j + 1$. If $l_k^j \leq l_k^{\max}$ **repeat** from Step 1; otherwise **terminate**.

Geometric interpretation on Shekel 5 test problem in the fifth iteration



New selection strategy for POH in proposed **DIRECT-L**

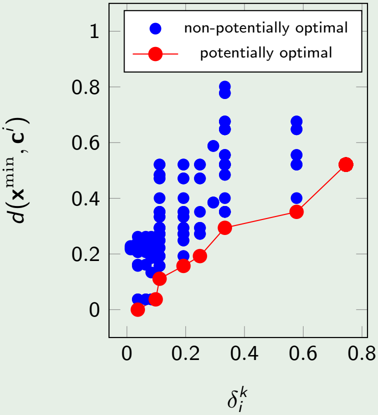
Definition 3 (Potentially optimal hyper-rectangles)

- **Step 1** At each iteration k, evaluate the Euclidean distance from the current minimum point ($\mathbf{x}^{\min}$) to other sampled points:

$$d(\mathbf{x}^{\min}, \mathbf{c}^i) = \sqrt{\sum_{j=1}^{n} (x_j^{\min} - c_j^i)^2} \qquad (5)$$

- **Step 2** Apply the procedure described in **Definition 3** in (4) using distances $d(\mathbf{x}^{\min}, \mathbf{c}^i)$ instead of objective function values.

Geometric interpretation on Shekel 5 test problem in the fifth iteration



Pseudo code of the **DIRECT-GL** algorithm

1 step: Scale $\mathbb{D} \to \overline{\mathbb{D}} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq x_i \leq 1, \ i = 1, \dots, n\}$.

2 step: Sample the center point $c_1 \in \overline{\mathbb{D}}$ of first hyper-rectangle and evaluate $f(c_1)$. Initialize $f_{\min} = f(c_1)$, $\mathbf{x}_{\min} c_1$ set (function evaluation counter) $m = 1$ and (iteration counter) $k = 1$. Set $\mathbb{J}_1 = 1$.

3 step: Identify the index set $\mathbb{J}_k^1 \subseteq \mathbb{I}_k$ of potentially optimal hyper-rectangles using **Definition 2**. (**DIRECT-G** selection procedure)

4 step: Subdivide (trisect) all hyper-rectangles from $\mathbb{J}_k^1$ and update $\mathbb{I}_{\mathrm{k}}$. Evaluate f at the centers of new hyper-rectangles. Update $f_{\min}$, $\mathbf{x}_{\min}$ and m.

5 step: Identify the index set $\mathbb{J}_k^1 \subseteq \mathbb{I}_k$ of potentially optimal hyper-rectangles using **Definition 3**. (**DIRECT-L** selection procedure)

6 step: Subdivide (trisect) all hyper-rectangles from $\mathbb{J}_k^1$ and update $\mathbb{I}_{\mathrm{k}}$. Evaluate f at the centers of new hyper-rectangles. Update $f_{\min}$, $\mathbf{x}_{\min}$ and m.

Repeat: Set $k = k + 1$. If stopping criteria are met, stop; otherwise go to **step 3**.

Stripinis, L., Paulavičius, R., Žilinskas J. (2017).
Improved scheme for selection of potentially optimal hyper-rectangles in DIRECT.
Optimization Letters 12(7), 1699–1712. DOI 10.1007/s11590-017-1228-4

Test problems and stopping conditions

- Since all the global minima f^* are known for all Hedar test problems in advance, investigated algorithms were stopped either when the point $\bar{\mathbf{x}}$ was generated such that the percent error

$$pe = 100\% \times \begin{cases} \frac{f(\bar{\mathbf{x}}) - f^*}{|f^*|}, & f^* \neq 0, \\ f(\bar{\mathbf{x}}), & f^* = 0, \end{cases} \tag{6}$$

  is smaller than the tolerance value $\varepsilon_{\mathrm{pe}}$, or when the number of function evaluations exceeds the prescribed limit of 10^6.
- In our investigation, four different values for $\varepsilon_{\mathrm{pe}}$ were considered: 10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}.

Hedar, A. (2005).
Test functions for unconstrained global optimization.
http://www-optima.amp.i.kyoto-u.ac.jp/ member/student/hedar/Hedar files/TestGO.htm.

Unsolved test problems and average of function evaluations

Table: Unsolved test problems

Alg./$\varepsilon_{\mathrm{pe}}$	10^{-2}	10^{-4}	10^{-6}	10^{-8}	Total
DIRECT	9/54	11/54	18/54	26/54	64/216
DIRECT-G	8/54	9/54	10/54	12/54	39/216
DIRECT-L	11/54	13/54	13/54	15/54	52/216
DIRECT-GL	**4/54**	**4/54**	**6/54**	**6/54**	20/216

Table: Average of function evaluations

Alg./$\varepsilon_{\mathrm{pe}}$	10^{-2}	10^{-4}	10^{-6}	10^{-8}
DIRECT	184,591	236,891	369,800	493,577
DIRECT-G	199,253	211,822	235,896	263,322
DIRECT-L	226,023	265,436	277,382	298,068
DIRECT-GL	**114,887**	**150,622**	**170,131**	**186,799**

Average of function evaluations solving **unimodal** and **multimodal** test problems

Table: Average of function evaluations for **unimodal** test problems

Alg./ε_{pe}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
DIRECT	**115,099**	**126,330**	**170,648**	**176,480**
DIRECT-G	195,439	199,961	207,523	220,482
DIRECT-L	373,655	374,537	376,095	378,051
DIRECT-GL	194,300	202,406	214,502	228,328

Table: Average of function evaluations for **multimodal** test problems

Alg./ε_{pe}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
DIRECT	208,913	275,588	439,503	604,561
DIRECT-G	200,588	215,973	245,826	278,316
DIRECT-L	174,351	227,251	242,832	270,074
DIRECT-GL	**87,092**	**132,498**	**154,601**	**172,263**

Average of function evaluations solving $n \leq 3$ and $n \geq 4$ test problems

Table: Average of function evaluations for test problems with $n \leq 3$

Alg./$\varepsilon_{\mathrm{pe}}$	10^{-2}	10^{-4}	10^{-6}	10^{-8}
DIRECT	2,290	3,828	25,335	262,245
DIRECT-G	15,760	15,942	16,246	16,685
DIRECT-L	1,480	1,666	1,905	2,278
DIRECT-GL	**509**	**759**	**1,064**	**1,533**

Table: Average of function evaluations for test problems with $n \geq 4$

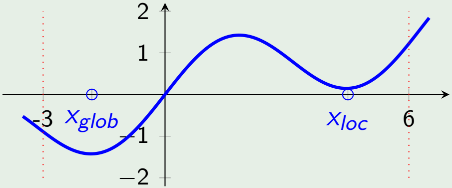
Alg./$\varepsilon_{\mathrm{pe}}$	10^{-2}	10^{-4}	10^{-6}	10^{-8}
DIRECT	319,846	409,809	625,371	665,211
DIRECT-G	335,394	357,192	398,862	446,311
DIRECT-L	392,619	461,136	481,767	517,525
DIRECT-GL	**199,748**	**261,812**	**295,568**	**324,254**

Global (Lipschitz) Optimization Problem (GOP)

We consider a general constrained (GOP) of the form:

$$\begin{aligned} &\min && f(\mathbf{x}), \quad f : \mathbb{R}^n \to \mathbb{R} \\ \text{s.t. } \mathbf{x} \in \mathbb{D}: & && g_i(\mathbf{x}) \le 0, i = 1...m \\ & && h_j(\mathbf{x}) = 0, j = 1...r \\ & && \mathbf{l} \le \mathbf{x} \le \mathbf{u}, \mathbf{x}, \mathbf{l}, \mathbf{u} \in \mathbb{R}^n. \end{aligned}$$

$$\min_{\mathbf{x} \in [-3,6]} \quad 0.25\mathbf{x} + sin(\mathbf{x})$$



All functions f, g_i, h_j satisfies the *Lipschitz condition*

$$|f(\mathbf{x}) - f(\mathbf{y})| \le L\|\mathbf{x} - \mathbf{y}\|, \quad \mathbf{x}, \mathbf{y} \in \mathbb{D}, \quad 0 < L < \infty,$$

with unknown *Lipschitz constant* L.

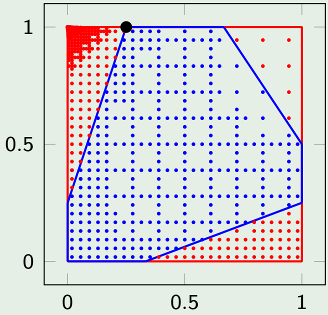
Exact L1 penalty function approach

An exact L1 penalty approach is a transformation of the original constrained problem to the form:

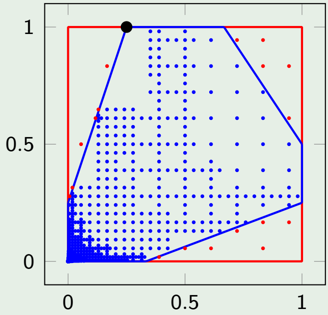
$$\min_{\mathbf{x} \in D} f(\mathbf{x}) + \sum_{i=1}^{m} \max\{p_i g_i(\mathbf{x}), 0\} + \sum_{i=1}^{r} p_{i+m} |h_i(\mathbf{x})|, \qquad (7)$$

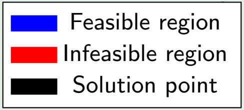
where $\mathrm{p_i}$ - penalty parameters.

Small penalty problem



Large penalty problem





Constrained test problems

Cagnina, L. C., Esquivel, S. C., and Coello Coello, C. A. (2008).
Solving engineering optimization problems with the simple constrained particle swarm optimizer.
Informatica (Ljubljana), 32(3), 319–326.

Finkel, D.E. (2005).
Global optimization with the Direct algorithm.
Ph.D. thesis, North Carolina State University.

Horst, R., Pardalos, P.M., Thoai, N.V. (1995).
Introduction to Global Optimization.
Nonconvex Optimization and Its Application. Kluwer Academic Publishers.

Na, J., Lim, Y., Han, C. (2017).
A modified direct algorithm for hidden constraints in an lng process optimization.
Energy p. 488500 . DOI 10.1016/j.energy.2017.03.047.

Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Y.P.Chen, Auger, A., Tiwari, S. (2005).
Problem definitions and evaluation criteria for the cec 2006 special session on constrained realparameter optimization.
KanGAL pp. 251–256.

Vaz, A., Vicente, L. (2009).
Pswarm: a hybrid solver for linearly constrained global derivative-free optimization.
Optimization Methods and Software 24(45), 669–685. DOI 10.1080/ 10556780902909948.

Stripinis, L., Paulavičius, R. (2018).
DIRECTLib - a library of global optimization problems for DIRECT-type methods, v1.1.
DOI 10.5281/zenodo.1403547. URL https: //doi.org/10.5281/zenodo.1403547

Comparison of Exact L1 penalty function using different POH: our scheme vs original

Stopping conditions:

1 cond.: When the point $\bar{\mathbf{x}}$ was generated such that the percent error ε_{pe} is smaller than the 10^{-2}.

2 cond.: When the number of function evaluations exceeds the prescribed limit of 10^6.

Table: Results solving problems with general constraints

#	DIRECT-GL-L1 p=10	DIRECT-GL-L1 p=10^2	DIRECT-GL-L1 p=10^3	DIRECT-L1 p=10	DIRECT-L1 p=10^2	DIRECT-L1 p=10^3
Aver.(overall)	516,137	418,516	**312,676**	636,793	682,036	705,836
Aver.($n \leq 3$)	443,498	241,614	**42,175**	526,485	409,504	494,361
Aver.($n \geq 4$)	580,337	547,705	**520,763**	705,260	879,914	853,840
Aver.(lin.cons.)	398,612	308,194	**158,096**	528,508	612,280	650,601
Aver.(nonlin.cons.)	692,335	558,644	**520,906**	764,540	752,595	754,704
U.prob.(total)	28/56	21/56	**15/56**	34/56	37/56	38/56
U.prob.(infes.sol.)	19/28	11/21	**5/15**	17/34	12/37	7/38
U.prob.(exc.fun.eval.)	9/28	10/21	**10/15**	17/34	25/37	31/38

DIRECT-GLc: Case when initial point(s) is/are infeasible

Next extension for problems with constraints. Minimize sum of constraint violations:

$$\min_{\mathbf{x} \in D} \varphi(\mathbf{x}), \tag{8}$$

where

$$\varphi(\mathbf{x}) = \sum_{i=1}^{m} \max\{p_i g_i(\mathbf{x}), 0\} + \sum_{i=1}^{r} p_{i+m} |h_i(\mathbf{x})|, \tag{9}$$

until a feasible point $\mathbf{x} \in D$ is found, i.e., such that (9) is zero. Penalty parameters p_i are simply set to 1.

DIRECT-GLc: Find at least one feasible point

Table: The number of function evaluations needed by algorithms to find a feasible point

#	DIRECT-GL-L1 p=10	DIRECT-GL-L1 p=10^2	DIRECT-GL-L1 p=10^3	DIRECT-L1 p=10	DIRECT-L1 p=10^2	DIRECT-L1 p=10^3	**DIRECT-GLc**
G01	4,340	**4,036**	4,340	4,626	4,244	4,776	4,050
G03	4,037	3,393	1,413	$> 10^6$	$> 10^6$	$> 10^6$	**1,381**
G05	8,635	**5,507**	6,331	$> 10^6$	$> 10^6$	$> 10^6$	6,329
G06	1,431	575	122	1,521	547	112	**102**
G07	847	1,318	1,660	**449**	531	813	927
G10	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	**3,394**
NASA speed reducer design problem	$> 10^6$	$> 10^6$	5,019	99.515	77,051	5,561	**167**
P01	$> 10^6$	**2,373**	5,009	$> 10^6$	5,273	8,021	4,165
Reactor net-work design	5,506	5,484	5,484	45,558	33,026	34,968	**5,418**

DIRECT-GLc: Penalizing objective values obtained at infeasible points

By the second extension, we transform problem (7) to (10):

$$\min_{\mathbf{x}\in D} f(\mathbf{x}) + \xi(\mathbf{x}, f_{\min}^{\text{feas}}),$$
$$\xi(\mathbf{x}, f_{\min}^{\text{feas}}) = \begin{cases} 0, & \mathbf{x} \in D^{\text{feas}} \\ \varphi(\mathbf{x}) + \Delta, & \text{otherwise}, \end{cases} \tag{10}$$

where $f_{\min}^{\text{feas}}$ - current best feasible solution and $\mathbb{D}^{\text{feas}}$ is set of points which satisfies constraints. $\Delta = |f(\mathbf{x}) - f_{\min}^{\text{feas}}|$, which is equal to absolute value of the difference between the best feasible function value found so far $f_{\min}^{\text{feas}}$ and the objective value at an infeasible center point.

Comparison of **DIRECT-GLc** with previous versions

Stopping conditions:

1 cond.: When the point $\bar{\mathbf{x}}$ was generated such that the percent error $\varepsilon_{\mathrm{pe}}$ is smaller than the 10^{-2}.

2 cond.: When the number of function evaluations exceeds the prescribed limit of 10^6.

Table: Results solving problems with general constraints

#	DIRECT-GL-L1 p=10^3	DIRECT-L1 p=10	**DIRECT-GLc**
Aver.(overall)	312,676	636,793	**240,727**
Aver.($n \leq 3$)	42,175	526,485	**2,283**
Aver.($n \geq 4$)	520,763	705,260	**433,021**
Aver.(lin.cons.)	158,096	528,508	**125,135**
Aver.(nonlin.cons.)	520,906	764,540	**406,577**
U.prob.(total)	15/56	34/56	**12/56**
U.prob.(infes.sol.)	5/15	17/34	**0/12**
U.prob.(exc.fun.eval.)	10/15	17/34	**12/12**

DIRECT-GLce: adding small tolerance for constraints

By third extension we update problem (10) to form (11):

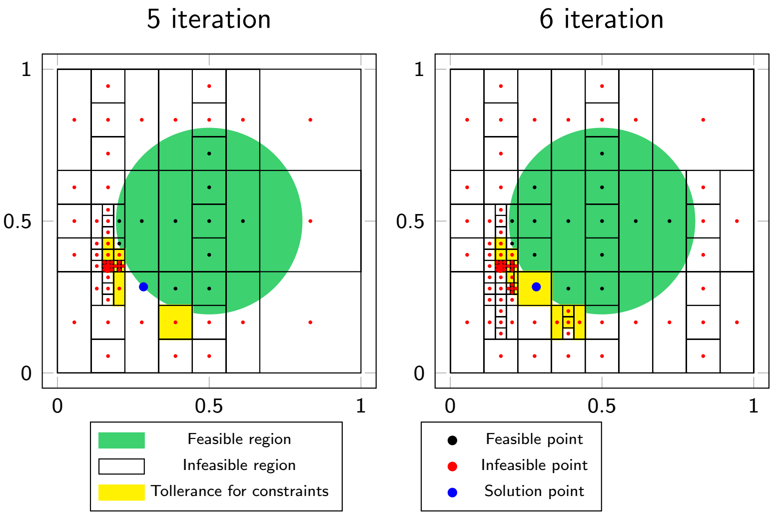
$$\min_{\mathbf{x}\in D} f(\mathbf{x}) + \tilde{\xi}(\mathbf{x}, f_{\min}^{\text{feas}}),$$

$$\tilde{\xi}(\mathbf{x}, f_{\min}^{\text{feas}}) = \begin{cases} 0, & \mathbf{x} \in D^{\text{feas}} \\ 0, & \mathbf{x} \in D^{\text{inf}}_{\varepsilon_{\text{cons}}} \\ \varphi(\mathbf{x}) + \Delta, & \text{otherwise}, \end{cases} \tag{11}$$

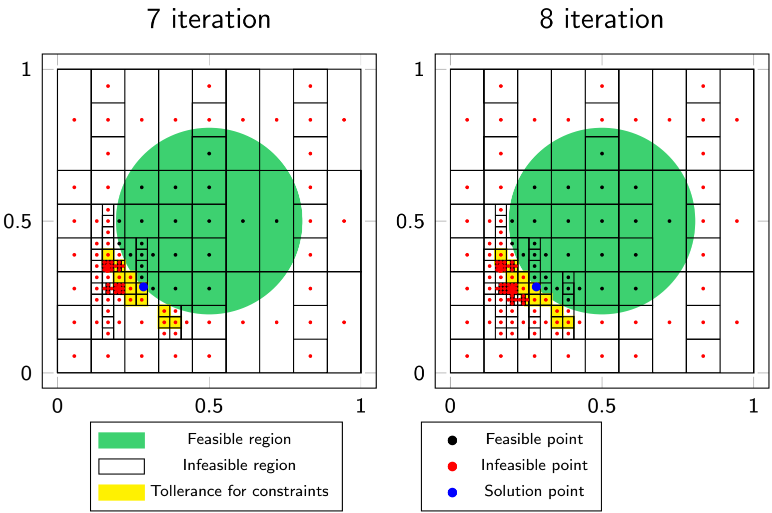
where $D^{\text{inf}}_{\varepsilon_{\text{cons}}} = \{\mathbf{x} : f(\mathbf{x}) \leq f_{\min}^{\text{feas}}, 0 < \sum_{i=1}^{m} \max\{g_i(\mathbf{x}), 0\} \leq \varepsilon_{\text{cons}}, \mathbf{x} \in D\}$
and $\varepsilon_{\text{cons}}$ is a small tolerance for constraint function sum, which automatically varies during the optimization process.

Stripinis, L., Paulavičius, R., Žilinskas J. (2019).
Penalty functions and two-step selection procedure based DIRECT-type algorithm for constrained global optimization.
Structural and Multidisciplinary Optimization 59(6),2155–2175. DOI 10.1007/s00158-018-2181-2

Geometric interpretation of **DIRECT-GLce** algorithm



Geometric interpretation of **DIRECT-GLce** algorithm



Comparison of **DIRECT-GLce** with previous versions

Stopping conditions:

1 cond.: When the point $\bar{\mathbf{x}}$ was generated such that the percent error $\varepsilon_{\mathrm{pe}}$ is smaller than the 10^{-2}.

2 cond.: When the number of function evaluations exceeds the prescribed limit of 10^6.

Table: Results solving problems with general constraints

#	DIRECT-GL-L1 $p=10^3$	DIRECT-L1 $p=10$	DIRECT-GLc	**DIRECT-GLce**
Aver.(overall)	312,676	636,793	240,727	**153,341**
Aver.($n \leq 3$)	42,175	526,485	**2,283**	4,331
Aver.($n \geq 4$)	520,763	705,260	433,021	**273,510**
Aver.(lin.cons.)	158,096	528,508	125,135	**78,962**
Aver.(nonlin.cons.)	520,906	764,540	406,577	**260,058**
U.prob.(total)	15/56	34/56	12/56	**3/56**
U.prob.(infes.sol.)	5/15	17/34	0/12	**0/3**
U.prob.(exc.fun.eval.)	10/15	17/34	12/12	**3/3**

Comparison with other DIRECT-type algorithms

Steps	DIRECT-L1	eDIRECTc	Filter Direct	DIRECT-GLce
Selection of potentially optimal hyper-rectangles (POH)	Original DIRECT strategy	Novel DIRECT-type constraint-handling technique that separately handles feasible and infeasible cells	Modified strategy, uses three sets: one from feasible, one from infeasible non-dominated and one from infeasible dominated points	Uses two step selection procedure from DIRECT-GL algorithm
Partitioning scheme	Original DIRECT trisection strategy	Based on Voronoi diagrams for partition the design space in Voronoi cells	Trisection strategy using the rules of "preference point" and "preference order" described in Definition 5 in their paper	Original DIRECT trisection strategy
Local minimization procedure	–	In `MATLAB` implementation uses `fmincon`	–	Only in the version: DIRECT-GLce+
Input parameters	Balance parameter ϵ, penalty parameters p_i	Balance parameter ϵ, allowed equality constraints violation ε_φ	Balance parameter ϵ, filter control parameters	Allowed equality constraints violation ε_φ

M. F. P. Costa and A. M. A. C. Rocha and E. M. G. P. Fernandes (2017).
Filter-based direct method for constrained global optimization.
Journal of Global Optimization in press (2017). DOI 10.1007/s10898-017-0596-8.

Liu, H., Xu, S., Chen, X., Wang, X., Ma, Q. (2017).
Constrained global optimization via a direct-type constraint-handling technique and an adaptive metamodeling strategy.
Structural and Multidisciplinary Optimization 55(1), 155–177 (2017). DOI 10.1007/s00158-016-1482-6 .

Comparative analysis with Filter DIRECT

Table: Comparison between algorithms on 20 test problems

	Filter DIRECT		DIRECT-GLce	
Label	f_{evl}	$f_{\min}$	f_{evl}	$f_{\min}$
P01	25,425	0.398895	117,367	0.029367
P02(a)	697,169	−22.44495	200,000	−397.147651
P02(b)	421,197	53.68674	200,000	−397.146850
P02(c)	724,337	−38.79484	200,000	−701.483390
P02(d)	16,715	−399.96613	54,769	−399.966130
P03(a)	1,109,995	−0.38317	117,665	−0.388737
P03(b)	347	−0.38888	985	−0.388736
P04	543	−6.66621	1,949	−6.666209
P05	1,009	201.159343	819	201.159319
P06	1,323	376.300244	1,791	376.306243
P07	1,417	−2.828227	2,705	−2.828227
P08	883	−118.700976	1,947	−118.689820
P09	2,203	−13.401764	8,271	−13.401411
P10	587	0.741833	2,455	0.741833
P11	5	−0.500000	11	−0.500000
P12	6665	−16.738797	23	−16.738069
P13	10,583	195.339906	41,431	189.357806
P14	1,967	−4.513963	9,409	−4.513921
P15	105	0.000000	181	0.000000
P16	151	0.705001	97	0.704964
Average		151,131		48,094
# of unsolved		5		3

Comparative analysis with eDIRECT-C

Table: Comparison of different algorithms for 13 test problems

#	Criteria	eDIRECT-C	DIRECT-GLce	DIRECT-GLce+min
G01	$f_{\min}$	-14.9998	-14.9991	$\mathbf{-15.0000}$
	f_{evl}	**147.4**	$787,405$	$4,153$
G02	$f_{\min}$	-0.2480	-0.2246	$\mathbf{-0.3148}$
	f_{evl}	$> 1,000.0$	$> 10^6$	$> 10^6$
G03	$f_{\min}$	-0.9989	$\mathbf{-1.0004}$	$\mathbf{-1.0004}$
	f_{evl}	145.4	**251, 547**	**251, 547**
G04	$f_{\min}$	$-30,665.5385$	$-30,663.5708$	$\mathbf{-30,665.5387}$
	f_{evl}	64.6	$21,355$	**25**
G05	$f_{\min}$	$5,145.8149$	5126.5089	**5, 126.4967**
	f_{evl}	412.80	$6,861$	**5, 629**
G06	$f_{\min}$	$-6,961.8137$	$-6,961.1798$	$\mathbf{-6,961.8139}$
	f_{evl}	**34.8**	$6,017$	129
G07	$f_{\min}$	**24.3062**	24.3332	**24.3062**
	f_{evl}	**152.4**	$> 10^6$	$1,161$
G08	$f_{\min}$	-0.095822	-0.095818	$\mathbf{-0.095825}$
	f_{evl}	154.2	$1,507$	**115**
G09	$f_{\min}$	785.6795	680.6928	**680.6301**
	f_{evl}	$> 1,000.0$	$89,301$	**41**
G10	$f_{\min}$	$7,049.2484$	$7,049.8749$	**7, 049.2480**
	f_{evl}	**104.8**	$561,857$	$3,607$
G11	$f_{\min}$	**0.7499**	**0.7499**	**0.7499**
	f_{evl}	**33.4**	$1,929$	447
G12	$f_{\min}$	$\mathbf{-1.0000}$	-0.9999	$\mathbf{-1.0000}$
	f_{evl}	52.0	85	**17**
G13	$f_{\min}$	0.6472	**0.05394**	**0.05394**
	f_{evl}	$> 1,000.0$	$458,239$	$100,171$
U.pr.		5	2	**1**

Comparative analysis with eDIRECT-C

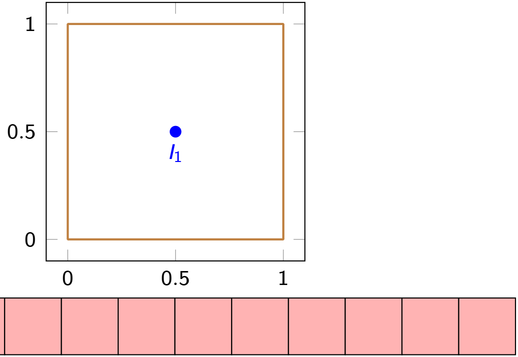
Table: Results of different algorithms for **engineering problems**

Algorithm	Solution point $\mathbf{x}^*$	f_{min}	f_{evl}
NASA speed reducer design problem			
eDIRECTc	3.50000, 0.70000, 17.00000, 7.30000, 7.71532, 3.35022, 5.28665	2994.4711[a]	118
DIRECT-GLce	3.50034, 0.70000, 17.00002, 7.30007, 7.80003, 3.35046, 5.28673	2996.5498	110,387
DIRECT-GLce+min	3.50000, 0.70000, 17.00000, 7.30000, 7.80000, 3.35021, 5.28668	**2996.3481**	**233**
Pressure vessel design problem			
eDIRECTc	1.00000, 0.62500, 51.81347, 84.57855	7006.7816[a]	412
DIRECT-GLce	1.10007, 0.62503, 56.99779, 50.99159	7164.3701	129,097
DIRECT-GLce+min	1.100000, 0.625000, 56.99481, 51.00125	**7163.7395**	**73**
Tension/compression spring design problem			
eDIRECTc	0.05169, 0.35674, 11.28819	0.012666[a]	292
DIRECT-GLce	0.05183279987, 0.36018518518, 11.1025880577	0.012679	20,845
DIRECT-GLce+min	0.05169590656, 0.35688327343, 11.2933789329	**0.012678**	**11**
Three-bar truss design problem			
eDIRECTc	0.78868, 0.40825	**263.8958**	26
DIRECT-GLce	0.78395, 0.42181	263.9158	1,331
DIRECT-GLce+min	0.78868, 0.40825	**263.8958**	**11**

a result is outside the feasible region

Static data structure (SDS) implementation in the DIRECT

0 Iter.: Evaluate hyper-rectangle at the center point and store information.

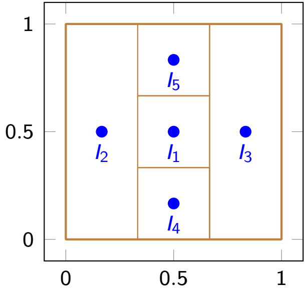


store the current state of the space partitioning

Static data structure (SDS) implementation in the DIRECT

0 Iter.: Evaluate hyper-rectangle at the center point and store information.

1 Iter.: Subdivide (trisect) potentially optimal hyper-rectangle $[I_1]$ and store information.



I_1	I_2	I_3	I_4	I_5								

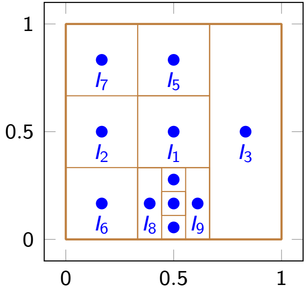
store the current state of the space partitioning

Static data structure (SDS) implementation in the DIRECT

0 Iter.: Evaluate hyper-rectangle at the center point and store information.

1 Iter.: Subdivide (trisect) potentially optimal hyper-rectangle $[I_1]$ and store information.

2 Iter.: Subdivide (trisect) potentially optimal hyper-rectangles $[I_2, I_4]$ and store information.



I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}		

store the current state of the space partitioning

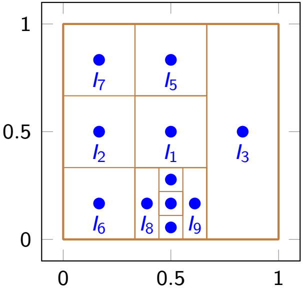
Static data structure (SDS) implementation in the DIRECT

0 Iter.: Evaluate hyper-rectangle at the center point and store information.

1 Iter.: Subdivide (trisect) potentially optimal hyper-rectangle $[I_1]$ and store information.

2 Iter.: Subdivide (trisect) potentially optimal hyper-rectangles $[I_2, I_4]$ and store information.

3 Iter.: ....

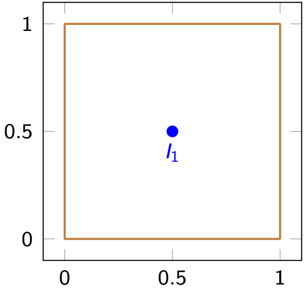


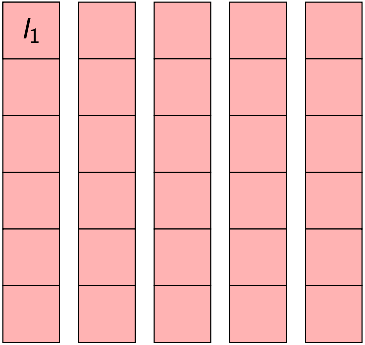
I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}		

store the current state of the space partitioning

Dynamic data structure (DDS) implementation in the DIRECT

0 Iter.: Evaluate hyper-rectangle at the center point and store information.

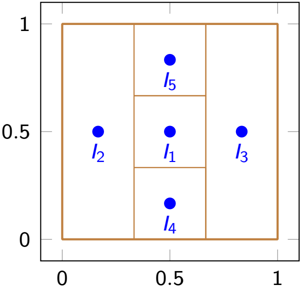


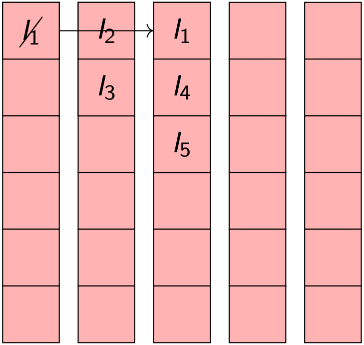


Dynamic data structure (DDS) implementation in the DIRECT

0 Iter.: Evaluate hyper-rectangle at the center point and store information.

1 Iter.: Subdivide (trisect) potentially optimal hyper-rectangle $[I_1]$ and store information.



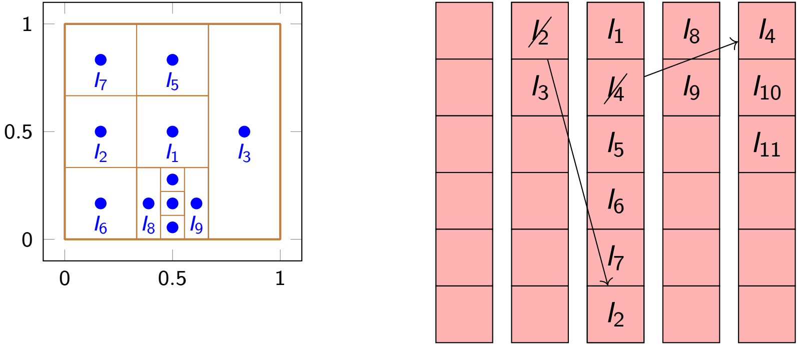


Dynamic data structure (DDS) implementation in the DIRECT

0 Iter.: Evaluate hyper-rectangle at the center point and store information.

1 Iter.: Subdivide (trisect) potentially optimal hyper-rectangle $[I_1]$ and store information.

2 Iter.: Subdivide (trisect) potentially optimal hyper-rectangles $[I_2, I_4]$ and store information.



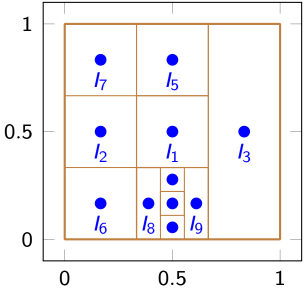
Dynamic data structure (DDS) implementation in the DIRECT

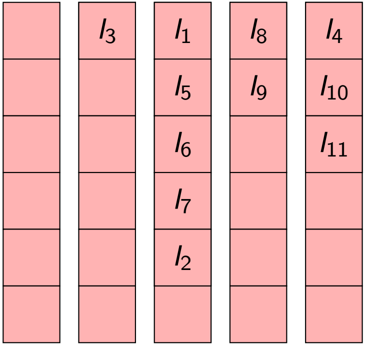
0 Iter.: Evaluate hyper-rectangle at the center point and store information.

1 Iter.: Subdivide (trisect) potentially optimal hyper-rectangle $[I_1]$ and store information.

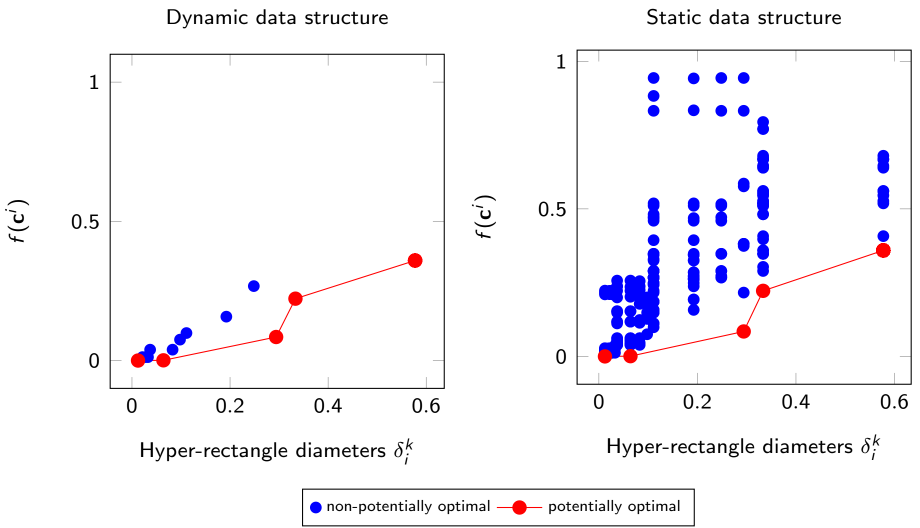
2 Iter.: Subdivide (trisect) potentially optimal hyper-rectangles $[I_2, I_4]$ and store information.

3 Iter.: ....

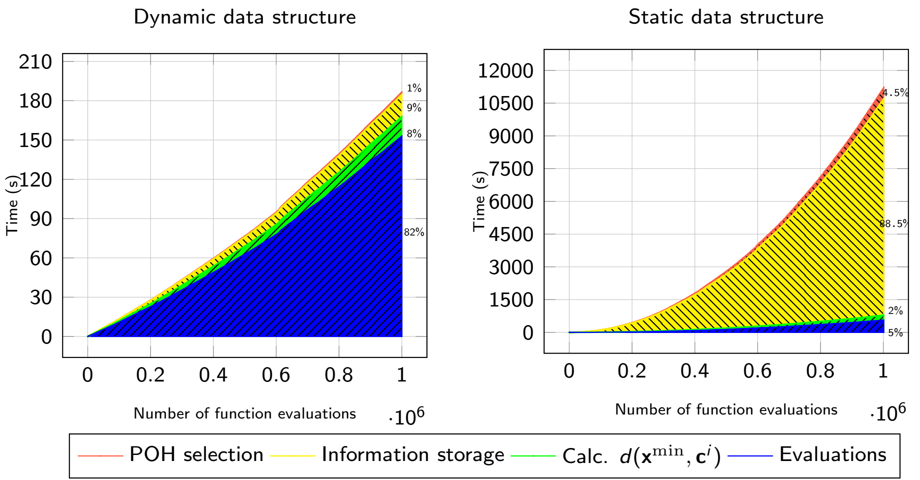




POH selection in SDS (left) and DDS (right) implementation DIRECT



Static data storage vs dynamic data storage



Problem	dimension	Type of function	Number of constrains	Type of constrains	p
G19	15	nonlinear	5	nonlinear inequality	33.4761%

p - estimated ratio between the feasible region and the search space

Design challenges of parallel DIRECT-type algorithms

Main steps of the DIRECT-GLCe algorithm

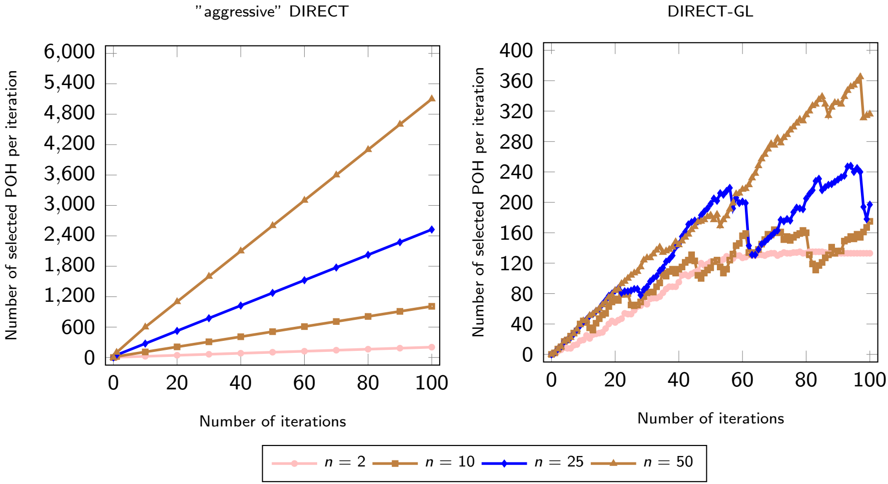
Initialization. Normalize the search space D to be the unit hyper-rectangle $\bar{D}$. Evaluate objective f (which also includes evaluation of an auxiliary function) at the center point $\mathbf{x}^c \in \bar{D}$. Set $f_{\min} = f(\mathbf{x}^c)$, $\mathbf{x}_{\min} = \mathbf{x}^c$, and initialize algorithmic performance measures and *stopping criteria*.

while *stopping criteria are not satisfied* **do**

> **Selection**. *Identify* the sets G and L of potentially optimal hyper-rectangles (subregions of $\bar{D}$) using enhanced global and local selection procedures accordingly, and take the unique union of these two sets $P = G \cup L$.
>
> **Sampling**. For each hyper-rectangle $j \in P$ *sample* it (using the same strategy as in original DIRECT) and *evaluate* objective (and auxiliary) functions at the centers of new hyper-rectangles. Update $f_{\min}, \mathbf{x}_{\min}$, and algorithmic performance measures.
>
> **Subdivision**. For each hyper-rectangle $j \in P$ subdivide (trisect), update partitioned search space information and *stopping criteria*.

Return $f_{\min}, \mathbf{x}_{\min}$, and algorithmic performance measures.

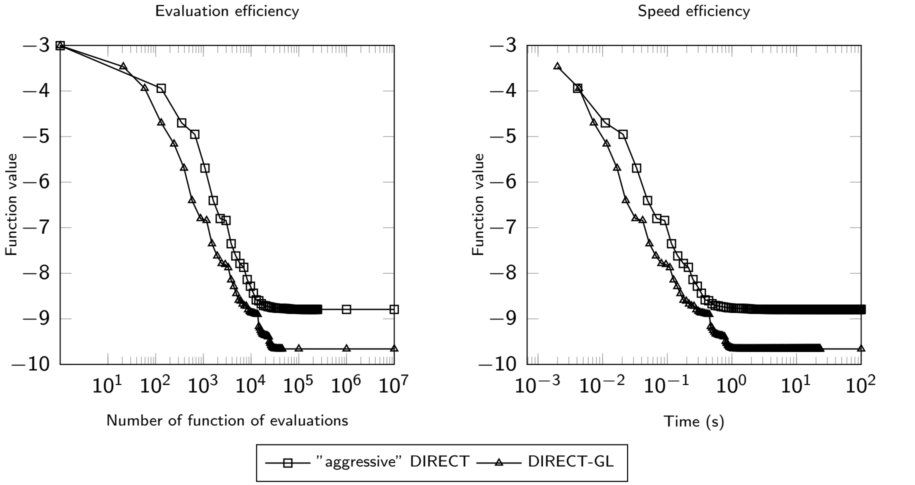
Design challenges of parallel DIRECT-type algorithms



Problem	dimension	Type of function	Number of constrains	Type of constrains	p
Rosenbrock	2, 10, 25, 50	-	-	box constraints	-

p - estimated ratio between the feasible region and the search space

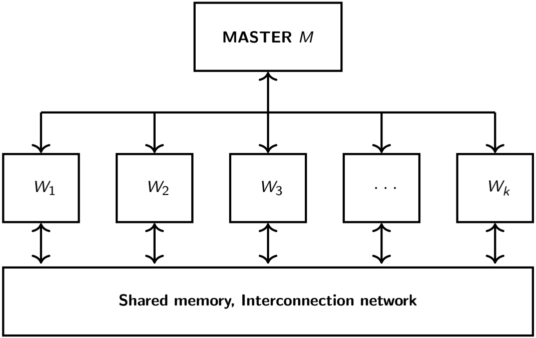
Design challenges of parallel DIRECT-type algorithms



Problem	dimension	Type of function	Number of constrains	Type of constrains	p
Michalewicz	10	-	-	box constraints	-

p - estimated ratio between the feasible region and the search space

Parallel scheme of pDIRECT-GLce



- Select POH
- Send instructions to the workers
- Collect results from the workers
- Check termination condition

Share jobs equaly and execute

Numerical investigation

Table: Test problems

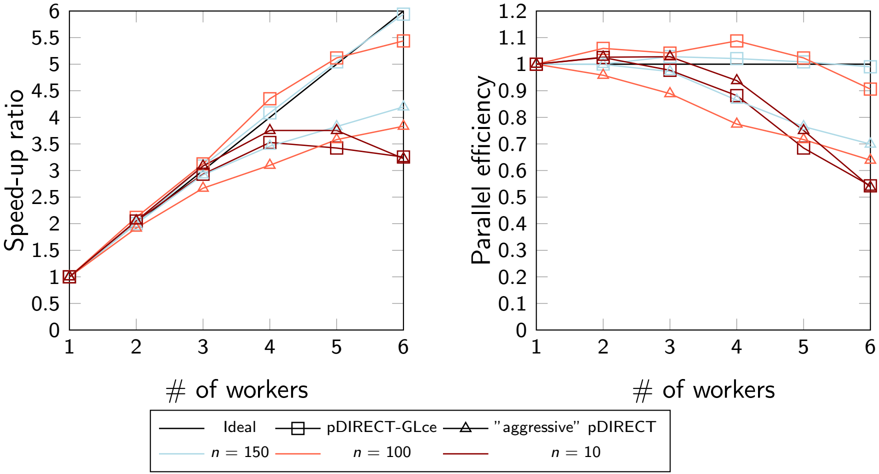
Problem	dimension	Type of function	Number of constrains	Type of constrains	p
G19	15	nonlinear	36	nonlinear inequality	0.0204%
G20	24	nonlinear	18	nonlinear inequality, nonlinear equality	0.0000%
G22	22	linear	20	nonlinear inequality, nonlinear equality	0.0204%
Michalewicz	10, 100, 150	-	-	box constraints	-

p - estimated ratio between the feasible region and the search space

Table: All the simulations were runed on 1 computer:

Product Collection	8th Generation Intel® Core™ i7 Processors
Processor Number	i7-8750H
# of Cores	6
# of Threads	12
Processor Base Frequency	2.20 GHz
Max Turbo Frequency	4.10 GHz
Memory	16 GB DDR4-2666 SDRAM

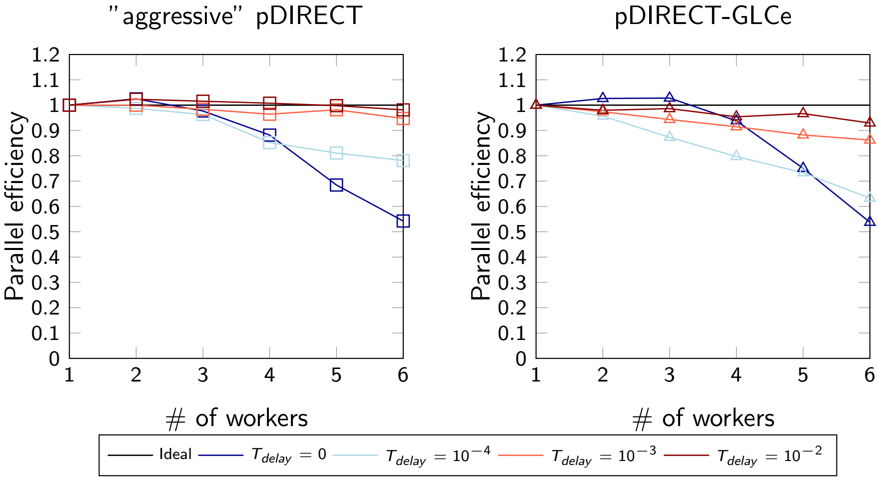
Speed-up ratio (*left*) and parallel efficiency (*right*) of parallel algorithms



Problem	dimension	Type of function	Number of constrains	Type of constrains	p
Michalewicz	10, 100, 150	-	-	box constraints	-

p - estimated ratio between the feasible region and the search space

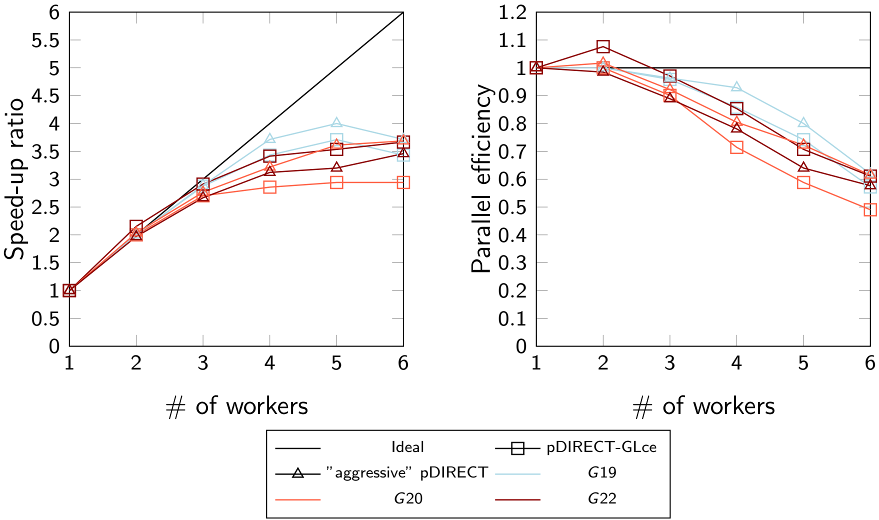
Speed-up ratio (*left*) and parallel efficiency (*right*) of parallel algorithms



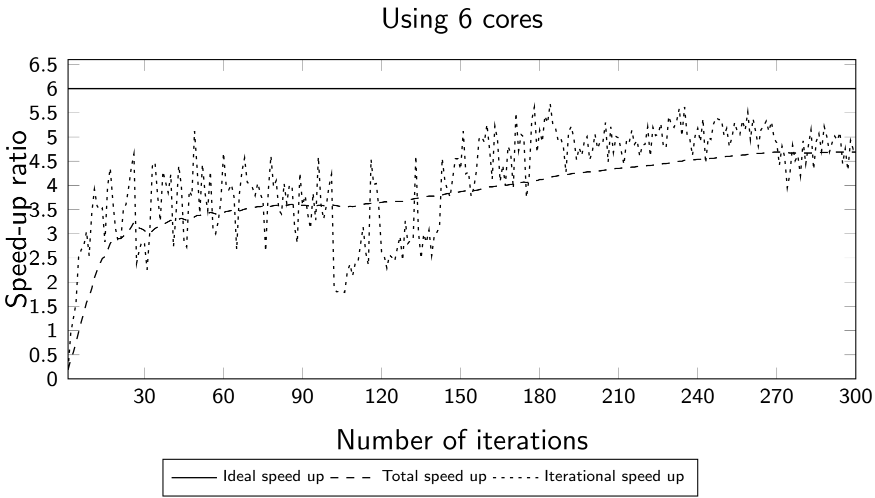
Problem	dimension	Type of function	Number of constrains	Type of constrains	p
Michalewicz	10	-	-	box constraints	-

p - estimated ratio between the feasible region and the search space

Speed-up ratio (*left*) and parallel efficiency (*right*) of parallel algorithms



Speedup ratio of pDIRECT-GLce in (1-300) iterations



Problem	dimension	Type of function	Number of constrains	Type of constrains	p
G19	15	nonlinear	36	nonlinear inequality	0.0204%

p - estimated ratio between the feasible region and the search space

Conclusions I

Modifications and improvements have been proposed for both DIRECT-type cases: for box and general constrained optimization problems:

- Improved potential optimal rectangles selection procedure. Comparing to original DIRECT, problems insolvability reduced by 39%, and the average number of function evaluations by 29%;
- Selection of potential optimal rectangles enlarged by candidates nearest to f_{min}. Comparing to DIRECT, problems unsolvability reduced by 69%, and number of function evaluations by 52%;
- New penalty function automatically generates the necessary penalty parameters to the DIRECT algorithm for problems with constraints;
- Added strategy for handling the cases with infeasible initial regions - situation when initial sampling points are infeasible and finding at least one feasible point is costly;
- Comparing to DIRECT-L1, problems unsolvability reduced by 82%, and number of function evaluations by 70%;

Conclusions II

- The DIRECT-GLc algorithm has the most wins, and it can solve about 50% of the problems with the highest efficiency;
- Solving more challenging problems (with nonlinear constraints and $n \geq 4$) DIRECT-GLce outperforms other algorithms, and the performance difference increases as the performance ratio increases;
- Implementation based on dynamic data structure requires 98.6 % less time compared to static versions;
- Proposed two versions: single-start based pDIRECT-GLce and multi-start based Aggressive pDIRECT;

Thank you!