Proceedings of TOGO 2010, pp. 23 – 26.

On interval Branch-and-Bound algorithm for additively separable functions with one common variable*

José L. Berenguel[1], L.G. Casado[1], E.M.T. Hendrix[2], Frédéric Messine[3] and I. García[2]

[1]*Department of Computer Architecture and Electronics, University of Almería, 04120, Spain.* jlberenguel@gmail.com, leo@ual.es

[2]*Department of Computer Architecture, University of Málaga, 29017, Spain.* Eligius.Hendrix@wur.nl, igarcia@uma.es

[3]*ENSEEIHT-IRIT, UMR-CNRS 5505, 2 rue Camichel BP 7122, 31071 Toulouse Cedex 7, France* Frederic.Messine@n7.fr

Abstract Interval Branch-and-Bound algorithms are powerful methods which aim for guaranteed solutions of Global Optimization problems. The computational effort to reach this aim increases exponentially with the problem dimension in the worst case. For separable functions this effort can be less as lower dimensional subproblems can be solved individually. We investigate possibilities to design specific methods for cases where the objective function can be considered separable, but common variables occur in the subproblems. As initial research we tackle the case where the problem can be decomposed in two additively separable functions with just one variable in common.

Keywords: Branch-and-Bound, Interval Arithmetic, Separable functions.

1. Introduction

Interval Branch-and-Bound methods are powerful methods which aim for guaranteed solutions of Global Optimization problems. Although these methods have the ability to handle constraints, we focus here on the generic box constrained global optimization problem, which is to find

$$f^* = \min_{x \in S} f(x) \tag{1}$$

where $S \in \mathbb{R}^n$. With increasing dimension n the computational effort of B&B interval methods increases drastically. In design problems, it is not unusual that the objective function f is composed of several functions. If this is the case in an additive way and the variables can be split into subgroups that do not overlap, we call the function completely additively separable (CASF). CASF can be solved by finding solutions of subfunctions independently and adding the obtained results. On the other hand, if subfunctions share variables we call the function additively shared separable (ASF). Here we study ASF with two subfunctions that share one variable. In general, solving (1) for ASF seems to be easier when separable character is taken into account. Here we study the adaption of an interval B&B algorithm to ASF. Section 2 describes the standard B&B algorithm we compare with. Section 3 describes the modifications in previous B&B algorithm to solve ASF problems. Results of experiments are presented in Section 4 and conclusion are shown in Section 5.

*This work has been funded by grants from the Spanish Ministry of Science and Innovation (TIN2008-01117), Junta de Andalucía (P08-TIC-3518), in part financed by the European Regional Development Fund (ERDF). Eligius Hendrix is a fellow of the Spanish "Ramon y Cajal" contract program, co-financed by the European Social Fund.

2. Interval B&B Algorithm

B&B algorithms can be determined by several rules, as it is shown in the following algorithm:

B&B (f,S):
Set the working list $L = \{S\}$ and the final list $Q = \emptyset$
While ($L \neq \emptyset$)
  Select an interval X from L — Selection rule
  if X cannot be eliminated — Elimination rule
    Divide X into X^k, $k = 1, \ldots, d$, subintervals — Division rule
    foreach X^k
      Compute a lower bound of $f(X^k)$ — Bounding rule
      if X^k satisfies the termination criterion — Termination rule
        Store X^k in Q
      else
        Store X^k in L

To solve (1) we use the following rules:

Bounding. An interval extension F of f is an inclusion function, i.e, $f(X) \subseteq F(X)$.
Selection. Interval $X \in L$ with smallest lower bound ($\underline{F}(X)$) is selected.
Division. The widest component of X is bisected. Two subintervals are generated.
Termination. If the length of the widest component of X is smaller or equal than ϵ, i.e. $w(X) \leq \epsilon$, interval X is stored in the final list Q.
Elimination. Two elimination tests are used:

- RangeUp test. Given $\overline{f^*}$ an upper bound of the global minimum f^*, an interval X does not contain a minimizer point if the lower bound of $f(X)$, $\underline{F}(X) > \overline{f^*}$. $\overline{f^*}$ is updated with the smallest value of f evaluated at the middle point of the selected intervals ($m(X)$).
- Monotonicity test. If $0 \notin F_i'(X)$ and X does not intersect the boundary of S, X can be rejected.

More elaborated rules can be found in literature but we use this simple algorithm as a basis to compare with [1–3].

3. Interval B&B Algorithm for ASF (ASF-B&B)

The instances of ASF problem that we want to solve here can be formulated from (1) as:

$$f^* = \min_{x \in S} f(x) = \min_{x^{[1]} \in S^{[1]}, x^{[2]} \in S^{[2]}} f^{[1]}(x^{[1]}) + f^{[2]}(x^{[2]}), \tag{2}$$

with $S^{[1]} \cup S^{[2]} = S$, $x^{[1]} \in \mathbb{R}^{n^{[1]}}$, $x^{[2]} \in \mathbb{R}^{n^{[2]}}$, $n^{[1]} + n^{[2]} = n + 1$ and without loss of generality we take as common variable $x^{[1]}_{n^{[1]}} = x^{[2]}_{n^{[2]}}$. To solve (2) using a B&B algorithm, we define two working lists $L^{[1]}$, $L^{[2]}$ and two final lists $Q^{[1]}$, $Q^{[2]}$ for the corresponding subfunctions. Following, the rules that define ASF-B&B algorithm are described:

Bounding. Given an interval $X^{[i]}$, we store two lower bounds, one for $f^{[i]}$ and another for f. They are calculated as follows:

- $\underline{F}^{[i]}(X^{[i]})$ is a lower bound of $f^{[i]}(X^{[i]})$ due to interval arithmetic.
- We define $EX^{[i]} = \{E \subseteq S, E^{[i]} = X^{[i]}\}$. Without loss of generality, let us focus on $X^{[1]}$. Then,

$$\underline{F}(EX^{[1]}) = \underline{F}^{[1]}(X^{[1]}) + \underline{F}^{[2]}(Z^{[2]}), \text{ where} \tag{3}$$

$$Z^{[2]} = \arg \min_{X^{[2]} \in L^{[2]} \cup Q^{[2]}} \underline{F}^{[2]}(X^{[2]}), \text{ with } X^{[1]}_{n^{[1]}} \cap X^{[2]}_{n^{[2]}} \neq \emptyset. \quad (4)$$

The proof of $\underline{F}(EX^{[1]})$ as a correct lower bound of $f(EX^{[1]})$ will be given in the final article.

Selection. List $L^{[1]}$ and $L^{[2]}$ are visited using round robin. Interval $X^{[i]}$ with the smallest lower bound $\underline{F}(EX^{[i]})$ is selected (see eq. (3)).

Division. The widest component of $X^{[i]}$ is bisected. Two subintervals are generated.

Termination. If the length of the widest component of $X^{[i]}$ is smaller or equal than ϵ, i.e. $w(X^{[i]}) \leq \epsilon$, interval $X^{[i]}$ is stored in final list $Q^{[i]}$.

Elimination. Wlog, let us focus on $X^{[1]}$. Elimination tests are the following:

- Unshared value of common variable. $X^{[1]}$ can be removed if $\forall X^{[2]} \in L^{[2]} \cup Q^{[2]}$, $X^{[1]}_{n^{[1]}} \cap X^{[2]}_{n^{[2]}} = \emptyset$.
- Non common variables monotonicity. $X^{[1]}$ can be removed if $0 \notin F'_i(X^{[1]})$, $i \neq n^{[1]}$ and $X^{[1]}$ does not intersect the boundary of $S^{[1]}$.
- RangeUp. Given $\overline{f^*}$ an upper bound of the global minimum f^*, an interval $X^{[1]}$ does not contain a minimizer point if $\underline{F}(EX^{[1]}) > \overline{f^*}$ (see eq. (3)). $\overline{f^*}$ is updated with the smallest value of $f^{[1]} + f^{[2]}$ evaluated at the middle point of selected intervals $X^{[1]}$, $Z^{[2]}$ (see eq. (4)), but using as common variable $X^{[1]}_{n^{[1]}} = Z^{[2]}_{n^{[2]}} = X^{[1]}_{n^{[1]}} \cap Z^{[2]}_{n^{[2]}}$.
- Subfunction RangeUp. An interval $X^{[1]}$ does not contain a minimizer point if $\underline{F}^{[1]}(X^{[1]}) > \overline{g}^{[1]}(X^{[1]}_{n^{[1]}})$, where $\overline{g}^{[1]}(X^{[1]}_{n^{[1]}})$ is the lowest value for $f^{[1]}(m(Y^{[1]}))$ found so far with $m(Y^{[1]})_{n^{[1]}} \in X^{[1]}_{n^{[1]}}$.

4. Results

We designed several instances to measure the performance of the ASF-B&B algorithm compared with B&B algorithm. The design of these test functions has been done using well-known functions in Global Optimization. The process is the following: we select two functions, for example Levy-5 (2 dimensions) and Price (2 dimensions), and create the new separable function L5P sharing the last variable of both functions. Table 1 shows the list of separable functions with their search domain and the subfunctions used to create them.

Table 1. Separable Test Functions.

Name	S	$f^{[1]}$	$f^{[2]}$
Eligius	$[-10, 10]^3$	$x_1^2 + x_1 x_3 + \frac{1}{2} x_3^2 + x_3$	$x_2^2 - 2x_2 x_3$
L5P	$[-10, 10]^3$	Levy5	Price
GP3	$[-2, 2]^3$	Goldstein-Price	Goldstein-Price
SHCBL3	$[-10, 10]^3$	Six-Hump-Camel-Back	Levy3

Table 2 shows the execution results of the functions in Table 1 using B&B and ASF-B&B algorithms. From left to right, the columns of the table are the name of the function; the precision of the final boxes; the interval containing the minimum value; if FE is the number of functions evaluations and GE is the number of gradient evaluations, the effort is measured as: $Effort = FE + n \cdot GE$; the execution time; the number of final boxes for B&B; and the number of final boxes for ASF-B&B.

Values in column Q for ASF-B&B algorithm shows the number of boxes after post-processing boxes in $Q^{[1]}$ and $Q^{[2]}$. This post-processing is based on applying the B&B elimination

tests on combining separable final boxes in non-separable ones. The execution time for ASF-B&B is the algorithm running time plus the post-processing time. These results show that monotonicity test in common variable, which is not used in ASF-B&B, has importance when the precision increases.

The current ASF-B&B algorithm running time is in general worse than in B&B due to coding details of the data structures that needs more than one sorting index. The version in the final paper will be improved and then the execution times shoould be reduced. Focusing on Effort, ASF-B&B outperform B&B for low precisions. For higher precisions, ASF-B&B outperform B&B in two, out of four cases, and it is similar in one. The difference in dimension for separable and non separable functions in the experimentation is just one. This shows that it is interesting to investigate the improvements of the ASF-B&B algorithm to solve larger dimensional problems.

Table 2. Execution Results.

Name	ϵ	Minimum	Effort	Time	Q	$Q^{[1]}$	$Q^{[2]}$
Eligius-B&B	10^{-2}	$[-85.234328, -84.843821]$	2,227	0.01	4	-	-
Eligius-ASF-B&B		$[-85.244046, -84.921886]$	1,921	$0.04 + 0$	4	6	8
Eligius-B&B	10^{-4}	$[-85.001832, -84.998779]$	3,424	0.02	4	-	-
Eligius-ASF-B&B		$[-85.001908, -84.999389]$	2,929	$0.05 + 0.01$	4	6	8
L5P-B&B	10^{-2}	$[-174.6145, -172.2646]$	3,742	0.07	3	-	-
L5P-ASF-B&B		$[-174.6145, -172.2646]$	3,568	$0.09 + 0$	3	7	22
L5P-B&B	10^{-4}	$[-172.2961, -172.2769]$	4,363	0.08	3	-	-
L5P-ASF-B&B		$[-172.2961, -172.2769]$	14,894	$0.52 + 0.06$	3	77	239
GP3-B&B	10^{-1}	$[-66, 498.05, 65.118653]$	239,683	3.21	8,893	-	-
GP3-ASF-B&B		$[-66, 169.43, 65.118653]$	26,978	$2.09 + 1.18$	6,103	665	514
GP3-B&B	10^{-2}	$[-4, 280.754, 65.000372]$	2,681,671	35.45	68,017	-	-
GP3-ASF-B&B		$[-4, 280.412, 65.000372]$	287,367	$161.44 + 51.92$	52,838	5,013	4,073
SHCBL3-B&B	10^{-2}	$[-171.4726, -168.6490]$	40,846	0.73	48	-	-
SHCBL3-ASF-B&B		$[-171.4702, -168.6490]$	9,723	$0.41 + 0.02$	30	62	21
SHCBL3-B&B	10^{-4}	$[-168.6792, -168.6566]$	46,372	0.82	24	-	-
SHCBL3-ASF-B&B		$[-168.6792, -168.6566]$	46,767	$3.78 + 0.5$	24	635	246

5. Conclusions

A new B&B algorithm to solve additively separable functions has been presented. Numerical results show that the current version of the algorithm outperforms the classic one for low precision results. The causes of poor results for higher precisions are known and deserve additional research. Results of an improved ASF-B&B algorithm will be shown in the final article.

References

[1] E. R. Hansen and G. W. Walster. *Global optimization using interval analysis*. Marcel Dekker, 2nd edition, 2004.

[2] R. B. Kearfott. *Rigorous Global Search: Continuous Problems*. Kluwer Academic Publishers, Dordrecht, Holland, 1996.

[3] H. Ratschek and J. Rokne. *New Computer Methods for Global Optimization*. Ellis Horwood Ltd., Chichester, England, 1988.