Optimization methods for process network synthesis — a case study

Tibor Csendes
József Attila University, Department of Applied Informatics,
H-6701 Szeged, P.O. Box 652, Hungary

April 9, 1998

Abstract

The paper investigates the possibilities of the optimal solution of process network design problems in a reliable and computationally tractable way. The capabilities of the solution methods are highlighted through example problems.

Keywords: Chemical network design, clustering algorithm, interval methods, penalty function.

AMS Subject Classification: 90C05, 65K30.

1 Introduction

Some simple process network design problems [6, 24] were studied to investigate the possible solution methods, their advantages and drawbacks. The examples have the most important characteristics of the related class of problems and in this way they are suitable to serve as case study problems.

The usual way of solution for such problems is either a trial-and-error approach, or in the better case a branch-and-bound scheme based on relaxation and on dual linear programming problems [29, 13]. The algorithm proposed in the paper utilizes the known expressions of the objective function and the constraints, and provide guaranteed reliability solutions using inclusion functions built up by the principle of interval arithmetic (see e.g. [1, 5, 6, 10, 26, 30]).

The basic idea of interval calculations is to extend the basic operations and the elementary functions from the real numbers to intervals. For the argument intervals $[a, b]$ and $[c, d]$ the following expressions hold:

$$[a, b] + [c, d] = [a + c, b + d]$$
$$[a, b] - [c, d] = [a - d, b - c]$$

$[a,b] \ * [c,d] = [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}]$
$[a,b] \ / \ [c,d] = [a,b] * [1/d, 1/c]$ if $0 \neq [c,d]$.

Denote the set of compact intervals by $\mathbb{I} := \{[a,b] \mid a \leq b,\ a,b \in \mathbb{R}\}$ and the set of n-dimensional intervals (also called simply intervals or boxes) by $\mathbb{I}^n$. We call a function $F : \mathbb{I}^n \to \mathbb{I}$ to be an *inclusion function* of $f : \mathbb{R}^n \to \mathbb{R}$ in X, if $x \in Y$ implies $f(x) \in F(Y)$ for each interval Y in X. In other words, $f(X) \subseteq F(X)$, where $f(X)$ is the range of $f(x)$ on X. It is assumed usually that the inclusion functions have the *isotonicity* property, i.e. $X \subseteq Y$ implies $F(X) \subseteq F(Y)$.

There are several ways to build an inclusion function for a given optimization problem (e.g. by using the Lipschitz constant). Interval arithmetic (see for a detailed introduction e.g. [1]) is a convenient tool for constructing the inclusion functions, and one can get those for almost all functions that can be calculated by a finite algorithm (i.e. not only for given expressions).

In the computational implementation, we coded the involved inclusion functions in Fortran-90 with an implemented interval arithmetic package handling the outside rounding necessary for the inclusion functions. The inclusion functions were also implemented by Fortran-77 subroutines in some of the numerical test. Interval extensions are available for the most programming languages (e.g. [17, 18, 22, 23]). The inclusion functions can be produced by the natural interval extension that provides isotone inclusions. In this straightforward way, to transform a subroutine calculating a real function to the interval version, one simply has to write a new statement to include the interval module, change the data types from real to interval, and rename some function calls.

The generality of the problem class and the modest requirement of the existence of the inclusion functions stress the importance of interval global optimization methods.

2 Example 1, the case of sharp separators

Consider a separation network that has two raw materials R_1 and R_2, each consisting of three components A, B and C. The network produces three products P_1, P_2 and P_3 consisting also of the same three components (with a possibly different distribution). There are merely two separators in the network, S_1 separates the component A, while S_2 separates the component C (see Figure 1). The f_j is the flow through separator S_j, i.e. $f_1 = 28 + 6x_1$ and $f_2 = 6(1 - x_1) + (1 - x_2)(18 + 5x_1) = 24 - x_1 - 18x_2 - 5x_1x_2$. The vector $x \in [0,1]^2$ to be optimized is the control vector of blending before the separators.

The problem is given by the above described separation network and by the distributions of the raw materials and the products. The actual distribution values can be found in Table 1.

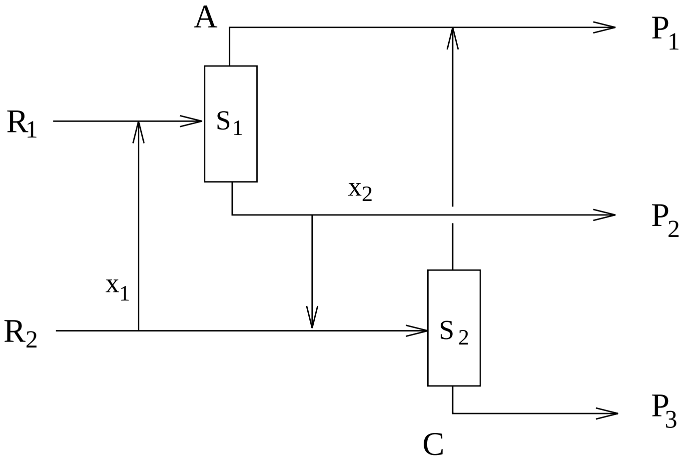

Figure 1: A simple separation network — Example 1

Table 1: Component flow rates of Example 1

	A	B	C
Feed stream R_1	10	9	9
Feed stream R_2	1	2	3
Product stream P_1	11	13/3	0
Product stream P_2	0	20/3	7
Product stream P_3	0	0	5

The design must be optimal in the sense that while fitting the above conditions on the distributions, the cost of the separation must be minimal:

$$\min f(x) = (D_1 f_1)^{0.6} + (D_2 f_2)^{0.6}, \tag{1}$$

where $D_j > 0$ is the difficulty of the separation on the separator number j. The conditions can be formulated as

$$\begin{aligned} 33 - 27x_2 - 6x_1x_2 &= 13 && (2)\\ 27x_2 + 6x_1x_2 &= 20 && (3)\\ 9x_2 + 3x_1x_2 &= 7 && (4)\\ 12 - 9x_2 - 3x_1x_2 &= 5. && (5) \end{aligned}$$

Notice that the problem belongs to the constrained nonlinear optimization problems, the constraints are bounds and equality constraints, the set of feasible points is convex and bounded, and the objective function differentiable and nonconvex. The flow functions f_1 and f_2 are now monotonous in the set $[0,1]^2$, f_1 is strictly monotonously increasing in x_1, and f_2 is strictly monotonously decreasing in x_1 and in x_2. The objective function can be both decreasing and increasing in x_1 according to the set separation costs values of D_1 and D_2, and is nondecreasing in x_2. The latter can easily be seen also in Figure 1, but it is studied in detail in the Appendix.

Notice that (2) and (3), and also (4) and (5) are equivalent, imposing the same constraints. With (3) $-$ 2 (4) we obtain immediately that $x_2 = 2/3$, and thus $x_1 = 1/2$. These values fit also the other equations, and this is the only feasible solution. Denote it with x^*, then

$$f^* = f(x^*) = (31D_1)^{0.6} + \left(9\frac{5}{6}D_2\right)^{0.6}.$$

As the set of feasibility is a single point, the separation cost coefficients D_1 and D_2 do not play any role now.

3 Some possible solution methods

The 'per hand' solution used in the previous section has the advantage that number representation problems like the one of 1/3 in the example can be solved in a straightforward way. Single point or zero measure feasibility sets does not mean much difficulty for this approach — they count better as advantages. Problems with positive measure feasibility sets cannot be solved in general by this procedure, only the recognition of suitable properties (monotonicity in a variable, simple partial derivatives, simplifyable objective function [9]) of the optimization problem can help. On the other hand, this method has obviously its limitations even for zero measure feasibility set problems due to the error prone manipulations, and hence only small complexity problems can be solved in this way.

Solution with robust random sampling or random walk algorithms has the necessary convergence properties in limit, yet several theoretical results stress [3, 16] that the convergence can be arbitrarily slow if the feasibility set is small relative to the search domain.

More sophisticated clustering stochastic methods [4, 8, 31, 32, 33] can improve the efficiency, yet with small feasibility sets and on limited computational facilities (in terms of memory size and CPU speed), the solution is not guaranteed. The source codes of two nonderivative implementations (in C and Fortran) are available for academic purposes per anonymous ftp at ftp.jate.u-szeged.hu in the /pub/math/optimization directory.

In [6] Csallner introduced interval methods for optimizing separation networks. The algorithm was based on minimalizing the number of variables to be optimized utilizing the equality constraints. This technique is straightforward if many equality constraints are available. The effectivity of the interval method was demonstrated by the solution of example problems.

The present study aims to provide an overview on the numerical properties of interval nonlinear optimization methods solving process network analysis problems and to compare them to traditional techniques.

4 Interval method for constrained optimization

The interval method suggested for the studied problem class:

ALGORITHM 1 GlobalOptimize $(f, X, \varepsilon, Y, F^*, L)$

1. $Y := X;\quad L := \{\,\};\quad \widetilde{f} := \overline{F(X)};$ $\{$Upper bound for $f^*\}$
2. $k :=$ OptimalComponent (Y); $\{$Direction Selection$\}$
3. Bisection (Y, k, U^1, U^2);
4. **for** $i := 1$ **to** 2 **do**
   (a) $F_U := F(U^i)$; **if** $\widetilde{f} < \underline{F_U}$ **then next**$_i$;
   (b) **if** $0 \notin G_j(U^i)$ for a $j = 1, \ldots, m$, **then next**$_i$;
   (c) $L := L + (U^i, \underline{F_U})$; $\{$Store U^i and $\underline{F_U}$ in $L\}$
5. **if** $L = \{\,\}$ **then return**;
6. $(Y, \underline{F_Y}) :=$ Head (L); $\quad L := L - (Y, \underline{F_Y})$;
7. $\widetilde{f} := \min\{\widetilde{f}, \overline{F(Y)}\}$; $\{$Improve upper bound for $f^*\}$
8. $L :=$ CutOffTest $(L, \widetilde{f})$;
9. **if** $w(F(Y)) \le \varepsilon$ **then**
   $F^* := [\underline{F_Y}, \widetilde{f}]$; $\{$Best possible inclusion of $f^*\}$
   return Y, F^*, L;
10. **goto** 2.

For Example 1, the inclusion function values $F_1(X)$ and $F_2(X)$ of the flows on the initial interval $X = ([0,1],[0,1])$ are:

$$\begin{aligned} F_1(X) &= 28 + 6[0,1] = [28,28] + [0,6] \\ &= [28,34], \\ F_2(X) &= 34 + 6[0,1] - 34[0,1] - 6[0,1][0,1] \\ &= [34,34] + [0,6] - [0,34] - [0,6] \\ &= [-6,40]. \end{aligned}$$

The negative lower estimate on the f_2 can be improved on the basis that by obvious reasons, the flows are always nonnegative. The inclusions of the constraint functions are for X:

$$\begin{aligned} G_2(X) &= 27[0,1] + 6[0,1][0,1] - 20 = [-20,13] \\ G_3(X) &= 9[0,1] + 3[0,1][0,1] - 7 = [-7,5] \end{aligned}$$

Each $G_j(X)$ $(j = 1, \ldots, 4)$ contains zero, thus Algorithm 1 subdivides X. Consider the subinterval $Y = ([0, 0.5], [0, 0.5])$, for this the inclusion functions for the constraints are:

$$\begin{aligned} G_2(X) &= 27[0,0.5] + 6[0,0.5][0,0.5] - 20 \\ &= [-20,-5] \\ G_3(X) &= 9[0,0.5] + 3[0,0.5][0,0.5] - 7 \\ &= [-7,-1.75] \end{aligned}$$

Thus, the subinterval cannot contain any feasible solution of Example 1, and hence it is to be deleted. It is easy to see that the constraint functions are monotonous functions of the decision variables x_1 and x_2, and thus the natural interval extension [1] conveys exact bounds. These can be made looser (to an extremely small extent, however) by the outside rounding necessary in computer implementation. As a consequence, the branch-and-bound algorithm presented can produce a convergence rate close to the quickest possible. In the case of our problem, a guaranteed inclusion of the feasible optimum point with a tolerance of 10^{-14} for each decision variable was achieved at the cost of merely 3665 objective function and constraint function evaluations. The detailed results are comprised in Table 2.

In Table 2, ε is the stopping condition parameter reflecting the aimed width of the objective function inclusion, NFE stands for the number of (objective and constraint) function evaluations, CPU is the required computation time in seconds, MAXIND is the necessary length of list L, and $w(Y)$ gives the width of the result interval Y. According to these numerical results, the computational and space complexity increase just moderately with the decrease of the user set tolerance ε. Thus, it is straightforward to require the highest possible precision.

For $\epsilon = 10^{-9}$, the result interval found was $Y =$

$$\begin{array}{c} [0.499999998486600816, 0.499999998603016138] \\ [0.666666666860692203, 0.666666666977107525] \end{array}$$

with objective function inclusion of $F(Y) =$

$$[11.7901758082679233, 11.7901758090692592].$$

It may be surprising that the subinterval Y does not contain the single feasible point (1/2, 2/3) discussed in Section 1. The latter point is represented by a

Table 2: Numerical results for Example 1

tolerance	results			
ε	NFE	CPU	MAXIND	$w(Y)$
10^{-1}	159	0.3	17	0.015625
10^{-5}	1335	2.5	68	10^{-6}
10^{-9}	2511	4.9	125	10^{-10}
10^{-13}	3665	7.2	173	10^{-14}

subinterval stored in the list L, and the given result interval Y is such that it meets the stopping criterion while having the smallest possible lower bound on the objective function. In this way, if we set ε below the difference of the machine representable numbers, then (with an additional safeguard stopping condition inhibiting infinite cycling) our algorithm would give the smallest possible positive width subinterval that may contain feasible points and which has the smallest lower bound on the inclusion of the objective function. It is definitely the most one may want from a computer program handling finite precision data.

5 Example 2 — the case of sharp separators

The second example is a more complex problem containing 2 input streams or raw materials, 3 output streams, 4 separators and 4 blending controls. The feed streams R_1 and R_2 are in general both mixtures of the components A, B and C. The network produces again three products P_1, P_2 and P_3 consisting also of the same three components (with a possibly different distribution). It is illustrated by Figure 2.

The cost of the separator is proportional to the amount of materials flowing through the separators. These quantities are given below, a_{ij} denotes e.g. the amount of component A flowing from feed stream R_1 through separator S_1.

$$\begin{aligned}
a_{11} &= (1-x_1)*A_1+(1-x_2)*A_2+x_4*(x_1*A_1+x_2*A_2) \\
b_{11} &= \frac{(1-x_1)*B_1+(1-x_2)*B_2+x_4*x_1*B_1+x_4*x_2*B_2}{1-x_4*x_3} \\
c_{11} &= (1-x_1)*C_1+(1-x_2)*C_2 \\
a_{21} &= x_1*A_1+x_2*A_2 \\
b_{21} &= x_1*B_1+x_2*B_2+x_3*b_{11} \\
c_{21} &= x_1*C_1+x_2*C_2+x_3*c_{11} \\
a_{12} &= (1-x_4)*a_{21}
\end{aligned}$$

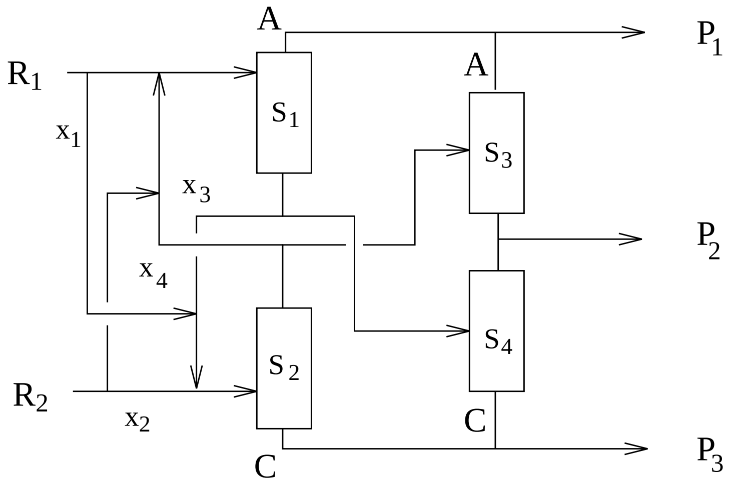

Figure 2: A more complex separation network — Example 2

$$\begin{aligned} b_{12} &= (1 - x_4) * b_{21} \\ b_{22} &= (1 - x_3) * b_{11} \\ c_{22} &= (1 - x_3) * c_{11} \end{aligned}$$

The cost of the separation is calculated then by

$$\begin{aligned} f(x) &= (D_1 * (a_{11} + b_{11} + c_{11}))^{0.6} + (D_2 * (a_{21} + b_{21} + c_{21}))^{0.6} + \\ & \quad (D_1 * (a_{12} + b_{12}))^{0.6} + (D_2 * (b_{22} + c_{22}))^{0.6}. \end{aligned}$$

Here $D_1 = D_2 = 1$ for all three subproblems. The controls x_i $(i = 1, \ldots, 4)$ denote the relative amounts of flow directed in a given direction, i.e.

$$0 \leq x_i \leq 1 \ \ (i = 1, \ldots, 4).$$

We have to disclose the possibility that both x_3 and x_4 are equal to 1, since otherwise no product flow would be produced (and also the above equations, e.g. that defining b_{11} are then well defined):

$$x_3 * x_4 \neq 1.$$

We investigate the subproblems given in Table 3. As we shall see, they have solutions with quite different nature. We also solved the subproblems discussed in [6], and after fixing the same variables as in that paper, we obtained closely the same results with our algorithm too.

Table 3: Component flow rates of Example 2

subproblem		A	B	C
1	Feed stream R_1	120	1	20
	Feed stream R_2	100	1	200
2	Feed stream R_1	100	6	15
	Feed stream R_2	10	1	97
3	Feed stream R_1	200	1	100
	Feed stream R_2	110	1	100

6 Various inclusion functions

The inclusion functions for the branch-and-bound algorithm are built by the so called natural interval extension [1, 30]. Due to the subdistributive law ($A(B+C) \subseteq AB+AC$) [1], the algebraically equivalent expressions may result in substantially different enclosures. If we have, on the other hand, a function $f(x)$ that is monotonous in all its variables, then it is possible to construct an inclusion function $F(X)$ such that it is basically the range of the function $f(X)$ on the given argument interval X. The only difference is caused by the machine dependent outside rounding necessary for the guaranteed inclusions.

Beyond the reformulation, we can also manipulate the inclusion functions by the redefinitions of the decision variables: from the point of the view of the results, it is allowed to pad all variables x_i by $(1-x_i)$. This seemingly unimportant change may also result in a better inclusion function as we shall see later.

Consider the defining equations of a_{11} to c_{22}. If a_{11} is computed in the showed order of operations, the addition $[(1-x_1)A_1 + (1-x_2)A_2] + [x_4(x_1A_1 + x_2A_2)]$ results in an unnecessary overestimation, since the upper bound of the first term is achieved when $x_1 = 0$, while the upper bound of the second term is reached when $x_1 = 1$. Thus the calculated upper bound for a_{11} reflect a never occurring case.

We can recognize immediately that c_{11}, a_{21}, a_{12} and c_{22} are monotonous in the control variables x_i, and this property is invariant for the $x_i \to (1-x_i)$ transformation. The monotonicity of b_{21} – c_{22} depend on other coefficients (as e.g. b_{21} on b_{11}). Although the monotonicity of c_{21} depends on c_{11} that is monotonous both in x_1 and x_2, yet (in the present form) it has the same property we discussed for a_{11}. It can also be computed according to the expression: $(x_1C_1 + x_2C_2)(1-x_3) + x_3(C_1+C_2)$. It is now monotonous in the variables x_1 and x_2, while it is not always monotonous in x_3.

We can conclude that although some of the inclusion functions of the coefficients can be improved, we cannot find such a transformation that would make

Table 4: Relative overestimations $(F_j(X^i) - F(X^i))/(w(F(X^i)))$ of different inclusion functions F_j of Example 2, subproblem 1 compared to the best computed inclusion $F(X^i)$.

i	$\frac{F_1(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_2(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_3(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_4(X^i)-F(X^i)}{w(F(X^i))}$
1	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]
3	[-5.5328%, 7.8820%]	[0.0000%, 0.0000%]	[0.0000%, 2.3949%]	[0.0000%, 1.4066%]
5	[-5.6916%, 6.2812%]	[0.0000%, 0.0000%]	[-0.5915%, 1.2279%]	[0.0000%, 0.3180%]
7	[-4.7421%, 4.8680%]	[0.0000%, 0.0000%]	[-0.6739%, 0.8000%]	[0.0000%, 0.0637%]
9	[-3.5359%, 3.5604%]	[0.0000%, 0.0000%]	[-0.5341%, 0.5574%]	[0.0000%, 0.0117%]
11	[-2.4276%, 2.4323%]	[0.0000%, 0.0000%]	[-0.3720%, 0.3760%]	[0.0000%, 0.0020%]
13	[-1.5675%, 1.5683%]	[0.0000%, 0.0000%]	[-0.2410%, 0.2417%]	[0.0000%, 0.0003%]
15	[-0.9690%, 0.9691%]	[0.0000%, 0.0000%]	[-0.1491%, 0.1492%]	[0.0000%, 0.0001%]
17	[-0.5820%, 0.5820%]	[0.0000%, 0.0000%]	[-0.0896%, 0.0896%]	[0.0000%, 0.0000%]
19	[-0.3433%, 0.3433%]	[0.0000%, 0.0000%]	[-0.0528%, 0.0528%]	[0.0000%, 0.0000%]
21	[-0.2002%, 0.2002%]	[0.0000%, 0.0000%]	[-0.0308%, 0.0308%]	[0.0000%, 0.0000%]
23	[-0.1161%, 0.1161%]	[0.0000%, 0.0000%]	[-0.0179%, 0.0179%]	[0.0000%, 0.0000%]
25	[-0.0670%, 0.0670%]	[0.0000%, 0.0000%]	[-0.0103%, 0.0103%]	[0.0000%, 0.0000%]
27	[-0.0386%, 0.0386%]	[0.0000%, 0.0000%]	[-0.0059%, 0.0059%]	[0.0000%, 0.0000%]
29	[-0.0222%, 0.0222%]	[0.0000%, 0.0000%]	[-0.0034%, 0.0034%]	[0.0000%, 0.0000%]
31	[-0.0128%, 0.0128%]	[0.0000%, 0.0000%]	[-0.0020%, 0.0020%]	[0.0000%, 0.0000%]
33	[-0.0073%, 0.0073%]	[0.0000%, 0.0000%]	[-0.0011%, 0.0011%]	[0.0000%, 0.0000%]
35	[-0.0042%, 0.0042%]	[0.0000%, 0.0000%]	[-0.0006%, 0.0006%]	[0.0000%, 0.0000%]
37	[-0.0024%, 0.0024%]	[0.0000%, 0.0000%]	[-0.0004%, 0.0004%]	[0.0000%, 0.0000%]
39	[-0.0014%, 0.0014%]	[0.0000%, 0.0000%]	[-0.0002%, 0.0002%]	[0.0000%, 0.0000%]
41	[-0.0008%, 0.0008%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0001%]	[0.0000%, 0.0000%]
43	[-0.0005%, 0.0005%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0001%]	[-0.0001%, 0.0001%]
45	[-0.0005%, 0.0005%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0001%]	[-0.0001%, 0.0001%]
47	[-0.0006%, 0.0006%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0001%]	[-0.0003%, 0.0003%]
49	[-0.0011%, 0.0011%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0001%]	[-0.0007%, 0.0007%]
51	[-0.0025%, 0.0025%]	[0.0000%, 0.0000%]	[-0.0002%, 0.0002%]	[-0.0016%, 0.0016%]
53	[-0.0055%, 0.0056%]	[0.0000%, 0.0000%]	[-0.0005%, 0.0004%]	[-0.0037%, 0.0036%]
55	[-5.2052%, 5.3053%]	[0.0000%, 0.0000%]	[-0.5005%, 0.7007%]	[-3.4034%, 3.7037%]
57	[-5.2052%, 5.3053%]	[0.0000%, 0.0000%]	[-0.5005%, 0.7007%]	[-3.4034%, 3.7037%]
59	[-5.2052%, 5.3053%]	[0.0000%, 0.0000%]	[-0.5005%, 0.7007%]	[-3.4034%, 3.7037%]

all coefficients monotonous in all the control variables, and thus provide sharp bounding inclusion functions. We studied the overestimations of the inclusion functions for some promising transformations on some realistic argument intervals that can occur during the optimization (zooming in at the global minimum point).

We investigated four different versions of the possible transformed objective functions. We obtained $F_1(X)$ from the original function by substituting X_1 everywhere by $(1 - X_1)$ and X_2 everywhere by $(1 - X_2)$. The coefficient a_{11} is e.g. then

$$X_1A_1 + X_2A_2 + X_4((1 - X_1)A_1 + (1 - X_2)A_2),$$

that can be simplified to have a sharper inclusion function

$$(1 - X_4)(X_1A_1 + X_2A_2) + X_4(A_1 + A_2).$$

The original objective function is denoted by $F_2(X)$ (to have a systematic notion): here

$$\begin{aligned} a_{11} &= (1-X_1)A_1 + (1-X_2)A_2 + X_4(X_1A_1 + X_2A_2) \\ &= (X_1A_1 + X_2A_2)(X_4 - 1) + A_1 + A_2. \end{aligned}$$

Notice that the last form of a_{11} is monotonous in all the x_i variables. This property holds also for b_{11}.

The objective function $F_3(X)$ is obtained from $F_2(X)$ by substituting X_1 everywhere by $(1-X_1)$. a_{11} is then equal to

$$X_1A_1 + (1-X_2)A_2 + X_4((1-X_1)A_1 + X_2A_2)$$

that can be simplified to

$$(1-X_4)(X_1A_1 - X_2A_2) + A_2 + X_4A_1.$$

$F_4(X)$ is the objective function version that differs from F_2 in the variable X_2: it is everywhere substituted by $(1-X_2)$. That yields

$$\begin{aligned} a_{11} &= (1-X_1)A_1 + X_2A_2 + X_4(X_1A_1 + (1-X_2)A_2) \\ &= (1-X_4)(X_2A_2 - X_1A_1) + A_1 + X_4A_2. \end{aligned}$$

To demonstrate the differences in the inclusion functions, we calculated the best inclusion $F(X)$ that can be produced on the basis of the above four inclusion function versions: $F(X) = [\underline{F}(X), \overline{F}(X)]$, where

$$\underline{F}(X) = \max\{\underline{F}_1(X), \underline{F}_2(X), \underline{F}_3(X), \underline{F}_4(X)\},$$

$$\overline{F}(X) = \min\{\overline{F}_1(X), \overline{F}_2(X), \overline{F}_3(X), \overline{F}_4(X)\}.$$

The numerical results comprised in Tables 4, 5 and 6 indicate that the inclusion function F_2 was clearly the best. Its precision was attained in same cases by some other inclusion functions. For the starting box there was no difference between the inclusion functions. It is remarkable how the relative overestimation changed during the procedure of zooming in: the typical pattern was that the absolute value of the relative overestimation decreased gradually, and almost monotonously as the size of the argument interval decreased. This trend turned back then in the neighbourhood of the machine precision for obvious reasons.

The inclusion function improvements discussed in this section have substantial effects on the efficiency of the interval optimization algorithms, yet they cannot change the basic characteristics of the solutions. The CPU time, iteration number and number of function evaluation values may be different for different inclusion functions of the same objective function, yet the convergence properties remain the same.

In real life situations, and especially for larger computational complexity problems this kind of multiple modeling to achieve better inclusion functions is hard to accomplish, yet algorithmic generation of the mathematical models as described in [14] and [25] may utilise these findings.

Table 5: Relative overestimations $(F_j(X^i) - F(X^i))(w(F(X^i)))$ of different inclusion functions F_j of Example 2, subproblem 2 compared to the best computed inclusion $F(X^i)$.

i	$\frac{F_1(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_2(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_3(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_4(X^i)-F(X^i)}{w(F(X^i))}$
1	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]
3	[-9.6032%,11.1993%]	[0.0000%, 0.0000%]	[-8.1343%, 9.9538%]	[0.0000%, 0.3269%]
5	[-10.1652%,10.4983%]	[0.0000%, 0.0000%]	[-8.9337%, 9.3150%]	[0.0000%, 0.0751%]
7	[-9.2803%, 9.3772%]	[0.0000%, 0.0000%]	[-8.2113%, 8.3146%]	[0.0000%, 0.0167%]
9	[-7.7729%, 7.8041%]	[0.0000%, 0.0000%]	[-6.8884%, 6.9191%]	[0.0000%, 0.0035%]
11	[-6.0022%, 6.0121%]	[0.0000%, 0.0000%]	[-5.3213%, 5.3304%]	[0.0000%, 0.0007%]
13	[-4.2863%, 4.2892%]	[0.0000%, 0.0000%]	[-3.8004%, 3.8030%]	[0.0000%, 0.0001%]
15	[-2.8597%, 2.8605%]	[0.0000%, 0.0000%]	[-2.5356%, 2.5363%]	[0.0000%, 0.0000%]
17	[-1.8102%, 1.8104%]	[0.0000%, 0.0000%]	[-1.6050%, 1.6052%]	[0.0000%, 0.0000%]
19	[-1.1044%, 1.1045%]	[0.0000%, 0.0000%]	[-0.9792%, 0.9793%]	[0.0000%, 0.0000%]
21	[-0.6578%, 0.6578%]	[0.0000%, 0.0000%]	[-0.5833%, 0.5833%]	[0.0000%, 0.0000%]
23	[-0.3860%, 0.3860%]	[0.0000%, 0.0000%]	[-0.3423%, 0.3423%]	[0.0000%, 0.0000%]
25	[-0.2245%, 0.2245%]	[0.0000%, 0.0000%]	[-0.1991%, 0.1991%]	[0.0000%, 0.0000%]
27	[-0.1299%, 0.1299%]	[0.0000%, 0.0000%]	[-0.1152%, 0.1152%]	[0.0000%, 0.0000%]
29	[-0.0749%, 0.0749%]	[0.0000%, 0.0000%]	[-0.0664%, 0.0664%]	[0.0000%, 0.0000%]
31	[-0.0431%, 0.0431%]	[0.0000%, 0.0000%]	[-0.0382%, 0.0382%]	[0.0000%, 0.0000%]
33	[-0.0248%, 0.0248%]	[0.0000%, 0.0000%]	[-0.0220%, 0.0220%]	[0.0000%, 0.0000%]
35	[-0.0143%, 0.0143%]	[0.0000%, 0.0000%]	[-0.0127%, 0.0127%]	[0.0000%, 0.0000%]
37	[-0.0082%, 0.0082%]	[0.0000%, 0.0000%]	[-0.0073%, 0.0073%]	[0.0000%, 0.0000%]
39	[-0.0048%, 0.0048%]	[0.0000%, 0.0000%]	[-0.0042%, 0.0042%]	[0.0000%, 0.0000%]
41	[-0.0028%, 0.0028%]	[0.0000%, 0.0000%]	[-0.0025%, 0.0025%]	[0.0000%, 0.0000%]
43	[-0.0018%, 0.0018%]	[0.0000%, 0.0000%]	[-0.0016%, 0.0016%]	[-0.0001%, 0.0000%]
45	[-0.0014%, 0.0014%]	[0.0000%, 0.0000%]	[-0.0013%, 0.0013%]	[-0.0001%, 0.0001%]
47	[-0.0018%, 0.0018%]	[0.0000%, 0.0000%]	[-0.0016%, 0.0016%]	[-0.0003%, 0.0003%]
49	[-0.0032%, 0.0030%]	[0.0000%, 0.0000%]	[-0.0029%, 0.0028%]	[-0.0006%, 0.0005%]
51	[-0.0067%, 0.0067%]	[0.0000%, 0.0000%]	[-0.0063%, 0.0063%]	[-0.0015%, 0.0012%]
53	[-0.0152%, 0.0152%]	[0.0000%, 0.0000%]	[-0.0141%, 0.0141%]	[-0.0031%, 0.0031%]
55	[-7.4334%, 7.4334%]	[0.0000%, 0.0000%]	[-6.8724%, 6.8724%]	[-1.4025%, 1.5428%]
57	[-7.4334%, 7.4334%]	[0.0000%, 0.0000%]	[-6.8724%, 6.8724%]	[-1.4025%, 1.5428%]
59	[-7.4334%, 7.4334%]	[0.0000%, 0.0000%]	[-6.8724%, 6.8724%]	[-1.4025%, 1.5428%]

Table 6: Relative overestimations $(F_j(X^i) - F(X^i))(w(F(X^i)))$ of different inclusion functions F_j of Example 2, subproblem 3 compared to the best computed inclusion $F(X^i)$.

i	$\frac{F_1(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_2(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_3(X^i)-F(X^i)}{w(F(X^i))}$	$\frac{F_4(X^i)-F(X^i)}{w(F(X^i))}$
1	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]	[0.0000%, 0.0000%]
3	[-7.0868%, 8.1748%]	[0.0000%, 0.0000%]	[-3.6597%, 5.2973%]	[-0.9839%, 2.9820%]
5	[-5.2368%, 5.3999%]	[0.0000%, 0.0000%]	[-3.2330%, 3.4787%]	[-1.6367%, 1.9396%]
7	[-3.3841%, 3.4111%]	[0.0000%, 0.0000%]	[-2.1550%, 2.1940%]	[-1.1724%, 1.2201%]
9	[-2.0564%, 2.0609%]	[0.0000%, 0.0000%]	[-1.3190%, 1.3250%]	[-0.7289%, 0.7363%]
11	[-1.2171%, 1.2178%]	[0.0000%, 0.0000%]	[-0.7820%, 0.7829%]	[-0.4338%, 0.4349%]
13	[-0.7111%, 0.7112%]	[0.0000%, 0.0000%]	[-0.4571%, 0.4573%]	[-0.2538%, 0.2540%]
15	[-0.4125%, 0.4125%]	[0.0000%, 0.0000%]	[-0.2652%, 0.2652%]	[-0.1473%, 0.1473%]
17	[-0.2382%, 0.2382%]	[0.0000%, 0.0000%]	[-0.1532%, 0.1532%]	[-0.0851%, 0.0851%]
19	[-0.1373%, 0.1373%]	[0.0000%, 0.0000%]	[-0.0883%, 0.0883%]	[-0.0490%, 0.0490%]
21	[-0.0790%, 0.0790%]	[0.0000%, 0.0000%]	[-0.0508%, 0.0508%]	[-0.0282%, 0.0282%]
23	[-0.0454%, 0.0454%]	[0.0000%, 0.0000%]	[-0.0292%, 0.0292%]	[-0.0162%, 0.0162%]
25	[-0.0261%, 0.0261%]	[0.0000%, 0.0000%]	[-0.0168%, 0.0168%]	[-0.0093%, 0.0093%]
27	[-0.0150%, 0.0150%]	[0.0000%, 0.0000%]	[-0.0096%, 0.0096%]	[-0.0054%, 0.0054%]
29	[-0.0086%, 0.0086%]	[0.0000%, 0.0000%]	[-0.0055%, 0.0055%]	[-0.0031%, 0.0031%]
31	[-0.0049%, 0.0049%]	[0.0000%, 0.0000%]	[-0.0032%, 0.0032%]	[-0.0018%, 0.0018%]
33	[-0.0028%, 0.0028%]	[0.0000%, 0.0000%]	[-0.0018%, 0.0018%]	[-0.0010%, 0.0010%]
35	[-0.0016%, 0.0016%]	[0.0000%, 0.0000%]	[-0.0010%, 0.0010%]	[-0.0006%, 0.0006%]
37	[-0.0009%, 0.0009%]	[0.0000%, 0.0000%]	[-0.0006%, 0.0006%]	[-0.0003%, 0.0003%]
39	[-0.0005%, 0.0005%]	[0.0000%, 0.0000%]	[-0.0003%, 0.0003%]	[-0.0002%, 0.0002%]
41	[-0.0003%, 0.0003%]	[0.0000%, 0.0000%]	[-0.0002%, 0.0002%]	[-0.0001%, 0.0001%]
43	[-0.0002%, 0.0002%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0001%]	[-0.0001%, 0.0001%]
45	[-0.0001%, 0.0001%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0001%]	[0.0000%, 0.0000%]
47	[-0.0001%, 0.0001%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0001%]	[0.0000%, 0.0000%]
49	[-0.0001%, 0.0001%]	[0.0000%, 0.0000%]	[-0.0001%, 0.0002%]	[-0.0001%, 0.0001%]
51	[-0.0003%, 0.0003%]	[0.0000%, 0.0000%]	[-0.0003%, 0.0003%]	[-0.0001%, 0.0001%]
53	[-0.0006%, 0.0006%]	[0.0000%, 0.0000%]	[-0.0007%, 0.0007%]	[-0.0003%, 0.0003%]
55	[-0.0041%, 0.0041%]	[0.0000%, 0.0000%]	[-0.0048%, 0.0051%]	[-0.0019%, 0.0019%]
57	[-0.0094%, 0.0094%]	[0.0000%, 0.0000%]	[-0.0109%, 0.0109%]	[-0.0044%, 0.0051%]
59	[-0.0216%, 0.0216%]	[0.0000%, 0.0000%]	[-0.0249%, 0.0265%]	[-0.0100%, 0.0116%]

Table 7: Results of ten independent runs of a stochastic optimization algorithm on the first subproblem of Example 2

$f(x^*)$	x_1^*	x_2^*	x_3^*	x_4^*	NFE	CPU
62.550111	0.00296602	0.99875609	0.75149047	1.0000000	56 067	8.73
62.791640	0.01029729	0.99757082	0.84846761	1.0000000	27 232	4.45
62.851381	0.02461725	0.99482994	0.64821641	1.0000000	61 190	9.45
62.855458	0.00166749	0.99553307	0.86046189	1.0000000	51 263	8.13
62.836668	0.05248426	0.99983809	0.81663978	1.0000000	38 757	6.10
62.649733	0.00442050	0.99337292	0.76789393	1.0000000	46 144	7.36
62.790379	0.01450848	0.98804530	0.77031329	1.0000000	46 312	7.52
62.713777	0.02558191	0.99685280	0.71918822	1.0000000	61 224	9.45
62.657934	0.02220838	0.99788250	0.76932032	1.0000000	30 592	4.78
62.665794	0.02431616	0.99800826	0.75213013	1.0000000	56 082	8.73
average					47 486.3	7.470

7 Numerical results of Example 2

The numerical tests with the presented interval method were carried out on an IBM RISC 6000 computer, using Fortran subroutines to calculate the inclusion functions. The standard time unit [8] (the CPU time needed to calculate the Shekel-5 test function 1000 times at the point (4.0, 4.0, 4.0, 4.0)) is 0.0036 seconds.

1. For the subproblem No. 1, the obtained bounds for the global optimum value were [62.496455, 62.697414]. It was achieved using 35 683 function evaluations, 10 000 memory units and 192.51 seconds. The best candidate interval to contain the global optimum found was:

$$\begin{array}{c}[0.00000000000, 0.00097656250]\\ [0.99804687500, 0.99902343750]\\ [0.71875000000, 0.72070312500]\\ [0.99804687500, 1.00000000000]\end{array}$$

with the function values in the interval

$$[62.4964558867970940, 62.9979748121370870].$$

The solution suggested by this result is demonstrated on Figure 3. The thick lines represent the blending decision variable setting $x = (0.0, 1.0, 0.72, 1.0)^T$. The objective function value (calculated with simple real arithmetic, double precision) at this point is 62.532910. Notice that this solution involves a recycling subgraph since both x_3 and x_4 are greater than zero. This finding supports again the statement of [24]: the long lasting folklore that optimal networks cannot contain a recycling subgraph is wrong.

We have solved the same problems by a clustering stochastic global optimization algorithm [8] containing a quasi-Newton and a random walk type [16]

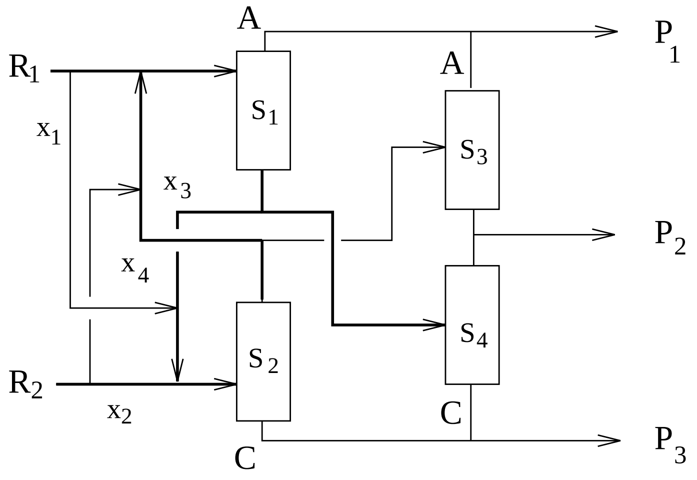

Figure 3: The found solution of Example 2 for subproblem 1.

local search phase. The results are given in Table 7. These are in accordance with the earlier reliable solutions, and they indicate that closely optimal result can be achieved by relatively simple tools too. It is remarkable, that in spite of the complex nature of the problem, this traditional method was always able to locate the global minimum.

2. For the second subproblem, the obtained bounds for the global optimum value was [41.706178, 41.728743]. It was achieved using 33 473 function evaluations, 9 162 memory units and 219.13 seconds. The best candidate interval to contain the global optimum found was:

$$\begin{array}{c}[0.00000000000000, 0.00006103515625]\\ [0.99993896484375, 1.00000000000000]\\ [0.99993896484375, 1.00000000000000]\\ [0.00000000000000, 0.00006103515625]\end{array}$$

with the function values in the interval

$$[41.7061780419035270, 41.7296852007469710].$$

This solution is demonstrated on Figure 4. The thick lines represent the blending decision variable setting $x = (0.0, 1.0, 1.0, 0.0)^T$. The objective function value in this point is 41.708090.

Table 8 shows the results of ten independent runs of the clustering algorithm on the second subproblem. Two equivalent global solutions were found, that represent symmetric settings of the network: either $x_3 = 1$ and $x_4 = 0$, or $x_3 = 0$ and $x_4 = 1$. Again, the found solution objective function values are

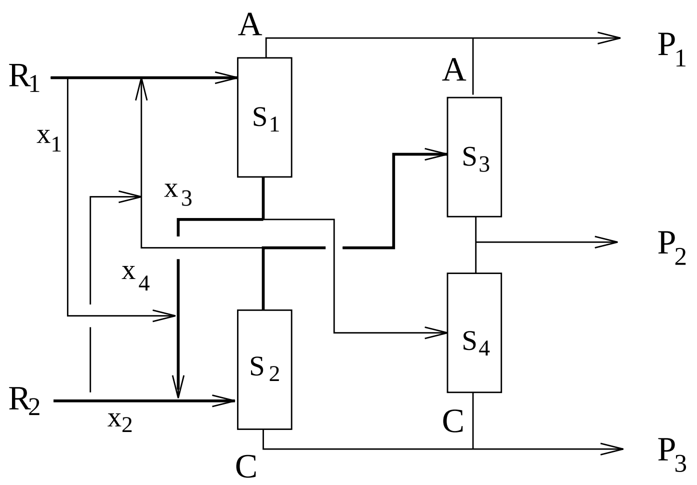

Figure 4: The found solution of Example 2 for subproblem 2.

Table 8: Results of ten independent runs of a stochastic optimization algorithm on the second subproblem of Example 2

$f(x^*)$	x_1^*	x_2^*	x_3^*	x_4^*	NFE	CPU
41.752882	0.00411032	0.99933612	0.06416711	1.00000000	63 237	9.67
41.758998	0.00008353	0.99315567	1.00000000	0.01898466	43 313	7.09
41.715977	0.00033055	0.99980205	0.03892092	1.00000000	61 882	9.34
41.712863	0.00009443	0.99978921	0.00456410	1.00000000	57 227	8.96
41.741688	0.00184536	0.99877479	0.02669378	1.00000000	48 546	7.86
41.752669	0.00494707	0.99950202	0.01486524	1.00000000	33 239	5.27
41.728559	0.00069627	0.99980878	0.10725010	1.00000000	59 292	9.34
41.739505	0.00074065	0.99998534	0.15366356	1.00000000	26 891	4.34
41.822413	0.01515444	0.99998251	0.06265361	1.00000000	13 231	2.14
41.778423	0.00342757	0.99736913	0.07210217	1.00000000	62 462	9.67
average					46 932.0	7.368

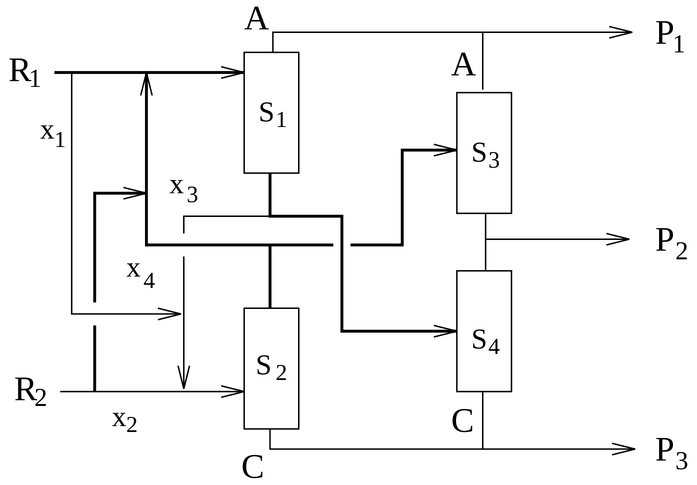

Figure 5: The found solution of Example 2 for subproblem 3.

within the bounds obtained by the interval method showing an acceptable, good approximate result.

3. For the third subproblem, the obtained bounds for the global optimum value was [66.378493, 66.800053]. It was achieved using 27 743 function evaluations, 10 000 memory units and 109.32 seconds. The best candidate interval to contain the global optimum found was:

$$\begin{array}{c}[0.000000000000, 0.000244140625]\\ [0.000000000000, 0.000244140625]\\ [0.000000000000, 0.000244140625]\\ [0.304687500000, 0.305175781250]\end{array}$$

with the function values in the interval

$$[66.3784930350121270, 66.9134166915420250].$$

This solution is demonstrated on Figure 5. The thick lines represent the blending decision variable setting $x = (0.0, 0.0, 0.0, 0.305)^T$. The objective function value in this point is 66.390600.

Table 9 comprises the results of ten independent runs of a stochastic global optimization algorithm on the third subproblem. The x_3 value for the second run was actually $0.93131840\ 10^{-6}$, and $x_2 = 0.3885780610^{-15}$ in the last run. In contrast to the earlier subproblems, where the stochastic solution cost was about 3% of that necessary for the guaranteed reliability method, this time the respective ratio was closely 7%.

Table 9: Results of ten independent runs of a stochastic optimization algorithm on the third subproblem of Example 2

$f(x^*)$	x_1^*	x_2^*	x_3^*	x_4^*	NFE	CPU
66.390599	0.00000000	0.00000000	0.00000000	0.82310751	60 789	9.56
66.396542	0.00000000	0.00000000	0.00000093	0.79924896	30 927	4.78
66.390599	0.00000000	0.00000000	0.00000000	0.98542356	46 144	7.69
66.390599	0.00000000	0.00000000	0.00000000	0.89319181	34 572	5.49
66.390599	0.00000000	0.00000000	0.00000000	0.92209975	49 497	7.96
66.390599	0.00000000	0.00000000	0.00000000	0.94439631	20 535	3.29
66.390599	0.00000000	0.00000000	0.00000000	0.95798284	56 890	9.17
66.390599	0.00000000	0.00000000	0.00000000	0.86388223	54 700	8.84
66.390599	0.00000000	0.00000000	0.00000000	0.99387717	50 375	8.18
66.390599	0.00000000	0.00000000	0.00000000	0.98693036	53 240	8.51
average					45 766.9	7.347

Summarizing the results of this section we can conclude that traditional optimization techniques may provide low cost approximate solutions, while the guaranteed reliability results can be given at a slightly increased computation time. If the reliability of the solution is important (as in this case proving an interesting new feature of process networks), interval arithmetic based techniques are useful.

8 Numerical results on Example 2 obtained by a sophisticated interval method

We solved the example problems no. 2 also by a sophisticated interval arithmetic based algorithm [15, 27] that included monotonicity, cut-off, and concavity tests together with an interval Newton step. The procedure utilized the new subdivision direction selection rules [11] and multisection techniques [7]. The program was run on 133 Mhz. Pentium PCs, and the supporting software environment was the C-XSC programming language [23] and the Numerical Toolbox for Verified Computing [15].

The model used was the most straightforward F_1 (cf. Section 6). The applied optimization algorithm variant was B/4, i.e. the subdivision direction selection method B was combined with a 4-multisection. For details see [7] and [11]. The tolerance was in each run 1.0 10^{-8}, and the search interval was for all three subproblems:

```
[-0.000000000000000E+000, 9.999999999999999E-001]
[-0.000000000000000E+000, 9.999999999999999E-001]
[-0.000000000000000E+000, 9.999999999999999E-001]
[-0.000000000000000E+000, 9.999999999999999E-001]
```

The printout of the program for the subproblem 1 (first the result inclusion of the optimizer points, then the inclusion of the optimum, followed by the efficiency measures CPUt, the CPU time in seconds, the maximal list length necessary MLL, the number of objective function, gradient, Hessian evaluations and iterations; together with the total number of result intervals):

```
[ 9.999847412109375E-001, 9.999999999999999E-001]
[ 9.999961853027342E-001, 9.999999999999999E-001]
[ 7.607116699218748E-001, 7.607650756835937E-001]
[ 9.999999999999994E-001, 9.999999999999999E-001]

contains the global minimizer,

[ 6.251229619001662E+001, 6.251229619999187E+001]

encloses the global minimum value!

B/4   CPUt:3254.29   MLL:1185   NFE:93618   NGE:37734   NHE:48
      IT NR:9409  total:2000
```

The printout of the program for the subproblem 2:

```
[ 9.999980926513671E-001, 9.999999999999999E-001]
[ 9.999923706054686E-001, 9.999999999999999E-001]
[ 9.999999999999994E-001, 9.999999999999999E-001]
[-0.000000000000000E+000, 1.525878906250000E-005]

contains the global minimizer,

[ 4.170808217195554E+001, 4.170808217963868E+001]

encloses the global minimum value!

B/4   CPUt:2230.06   MLL:1016   NFE:65167   NGE:26884   NHE:35
      IT NR:6703  total:1626
```

The printout of the program for the subproblem 3:

```
[ 9.999999998835846E-001, 9.999999999999999E-001]
[-0.000000000000000E+000, 1.164153218269349E-010]
[-0.000000000000000E+000, 2.328306436538697E-010]
[-0.000000000000000E+000, 9.999999999999999E-001]
```

```
contains the global minimizer,

[ 6.639058982692353E+001, 6.639058983626247E+001]

encloses the global minimum value!

Separation network 3-4dim

B/4   CPUt:3196.76   MLL:1212   NFE:92246   NGE:37114   NHE:68
      IT NR:9244  total:1808
```

In each case, the achieved accuracy was many order of magnitude better than those obtained with the basic interval optimization method, and also much better than any result obtained by the traditional stochastic clustering algorithm. Additionally, the interval Newton step was able to state that each result interval contained a unique global minimizer point. This impressive quality of result was provided at the cost of about just 10-20 times larger CPU time. The memory complexity proved to be manageable.

The present and future work is oriented to consider more complex process network synthesis problems like [28], and sophisticated constrained optimization techniques [19, 20, 21].

9 Summary and conclusions

This case study investigated possible solution approaches for process network synthesis problems. We studied a traditional stochastic algorithm, and two variants of interval arithmetic based procedures. Two modest size selected problems were solved by all the methods, and we were able to give a theoretical confirmation for the necessity of the recycling in optimal separation networks.

A traditional clustering stochastic optimization algorithm proved to be capable to provide good approximations of the global optimizer points and that of the optimum value. Interval methods are well suited to the needs of separation network design problems, they can give guaranteed reliability results indispensable in some sensitive cases. The efficiency of the improved interval algorithm was excellent regarding the high accuracy achieved. Further research is required for the special problems of real life larger complexity separator and heat exchanger networks.

Acknowledgements. The work has been supported by the Grants MKM 75/1996, OTKA T 016413, T 017241 and KDAW 960.4-90. The author acknowledges the programming and testing work made by Mihály Csaba Markót.

References

[1] Alefeld G. and Herzberger J. (1983), *Introduction to Interval Computations*, Academic Press, New York.

[2] Androulakis I. P., Maranas C. D. and Floudas C. A. (1995), *αBB: A Global Optimization method for General Constrained Nonconvex Problems*, J. Global Optimization **7**, 337–363.

[3] Baritompa W. P., Baoping Z., Mladineo R. H., Wood G. R., Zabinsky Z. B. (1995), *Towards pure adaptive random search*, J. of Global Optimization **7**, 93–110.

[4] Boender C. G. E., Rinnooy Kan A. H. G., Timmer G. T. and Stougie L. (1982), *A stochastic method for global optimization*, Mathematical Programming **22**, 125–142.

[5] Byrne R.P. and Bogle I.D.L. (1996), Solving nonconvex process optimisation problems using interval subdivision algorithms. In: Grossmann I. E. (ed.): Global Optimization in Engineering Design, Kluwer, Dordrecht, 1996, 155–174.

[6] Csallner A. E. (1993), *Global optimization in separation network synthesis*, Hung. J. Ind. Chem. **21**, 303–308.

[7] Csallner A. E., Csendes T. and Markót M. C. (1997), Multisection in interval methods for global optimization, manuscript.

[8] Csendes T. (1988), *Nonlinear parameter estimation by global optimization — efficiency and reliability*, Acta Cybernetica **8**, 361–370.

[9] Csendes T. and Rapcsák T. (1993), *Nonlinear coordinate transformations for unconstrained optimization. I. Basic transformations*, J. of Global Optimization **3**, 213–221.

[10] Csendes T., Zabinsky Z. B. and Kristinsdottir B. P. (1995), *Constructing large feasible suboptimal intervals for constrained nonlinear optimization*, Annals of Operations Research **58**, 279–293.

[11] Csendes T. and Ratz D. (1997), Subdivision direction selection in interval methods for global optimization. SIAM J. Numerical Analysis **34**, 922–938.

[12] Floudas C. A., Ciric A. R., and Grossmann I. E. (1986), Automatic Synthesis of Optimum Heat Exchanger Network Configurations, AIChE J. **32**, 276.

[13] Floudas C. A. (1987), *Separation synthesis of multicomponent feed streams into multicomponent product stream*, AIChE J. **33**, 540–550.

[14] Friedler, F., B. Imreh, T. Csendes and L.T. Fan: Process Network Synthesis: Model Generation and Optimization, Kluwer, Dordrecht, in preparation.

[15] Hammer R., Hocks M., Kulisch U. and Ratz D. (1993), *Numerical Toolbox for Verified Computing I.*, Springer, Berlin.

[16] Järvi, T.: A random search optimizer with an application to a maxmin problem. Publications of the Institute for Applied Mathematics, University of Turku, No. 3, 1973

[17] Jüllig H.-P. (1992), *BIBINS/2.0 — C++ Bibliotheken für Vektoren und Matrizen über beliebigem skalaren Datentyp unter Berücksichtigung spezieller spärlicher Strukturen sowie Intervalldatentypen*, Bericht 92.6, Technical University of Hamburg-Harburg.

[18] Kearfott R. B. (1995), *A FORTRAN-90 Environment for Research and Prototyping of Enclosure Algorithms for Constrained and Unconstrained Nonlinear Equations*, ACM T. on Mathematical Software **21**, 63–78.

[19] Kearfott R. B. (1996), *A review of techniques in the verified solution of constrained global optimization problems.* In: Kearfott R. B. and Kreinovich V. (eds.): Applications of Interval Computations, Kluwer, Dordrecht, 1996, 23–60.

[20] Kearfott R. B. (1996), *Test results for an interval branch and bound algorithm for equality-constrained optimization.* In: Floudas C. A. and Pardalos P. M. (eds.): State of the Art in Global Optimization, Kluwer, Dordrecht, 1996, 181–199.

[21] Kearfott, R. B. (1996), Rigorous global search: continuous problems. Kluwer, Dordrecht.

[22] Klatte R., Kulisch U., Neaga M., Ratz D. and Ullrich Ch. (1992), *PASCAL–XSC – Language Reference with Examples*, Springer, New York.

[23] Klatte R., Kulisch U., Lawo C., Rauch M. and Wiethoff A. (1993), *C-XSC — A C++ Class Library for Extended Scientific Computing*, Springer, Heidelberg.

[24] Kovács Z., Friedler F. and Fan L. T. (1993), *Recycling in a separation process structure*, AIChE J. **39**, 1087–1089.

[25] Kovács Z., Friedler F. and Fan L. T. (1993), *Algorithmic Generation of the Mathematical Model for Separation Network Synthesis*, Proc. European Symposium on Computer Aided Process Engineering, Graz, Austria, July 5–7, 1993, Suppl., 7–11.

[26] Kristinsdottir B. P., Zabinsky Z. B., Csendes T. and Tuttle M. E. (1993), *Methodologies for tolerance intervals*, Interval Computations **3**, 133–147.

[27] Markót M.C.: Multisection in interval global optimization, B.Sc. Thesis, Szeged, 1997.

[28] Messine F., Lagouanelle J. L., Noailles J., Ghanem M., Floquet P. and Domenech S.: Global Optimization Method for Process Synthesis. CHISA'96, Prague, 25-30 August 1996, H5.4.

[29] Quesada I. and Grossmann I. E. (1995), *Global optimization of bilinear process networks with multicomponent flows*, Computers chem. Engng. **19**, 1219–1242.

[30] Ratschek H. and Rokne J. (1993), *Interval Methods*, In: Handbook of Global Optimization, Horst R. and Pardalos P. M. (eds.), Kluwer, Dordrecht, 751–828.

[31] Törn, A. (1978), A search clustering approach to global optimization. In: Towards Global Optimization 2., (ed. L.C.W. Dixon and G.P. Szegő), 49-62, North-Holland, Amsterdam.

[32] Törn, A., A. Žilinskas: Global Optimization. (Lecture Notes in Computer Science No. 350, G. Goos and J. Hartmanis, Eds.) Springer, Berlin, 1989.

[33] Törn, A., S. Viitanen (1992), Topological global optimization using presampled points. J. Global Optimization **5**, 267-276.

Appendix

Analytic study of the monotonicity properties of the objective function of Example 1

To study the monotonicity properties of the objective function, consider the derivatives

$$\begin{aligned}\partial f(x)/\partial x_1 \quad = \quad & \frac{3(D_1(x_1+28))^{0.6}}{5(x_1+28)} + \\ & \frac{3(5x_2+1)(-D_2(x_1(5x_2+1)+6(3x_2-4)))^{0.6}}{5(x_1(5x_2+1)+6(3x_2-4))}\end{aligned}$$

and

$$\partial f(x)/\partial x_2 = \frac{3(5x_1+18)(-D_2(x_1(5x_2+1)+6(3x_2-4)))^{0.6}}{5(x_1(5x_2+1)+6(3x_2-4))}.$$

The point $x_1 = x_2 = 1.0$ must be handled separately, since these control values mean that the separator S_2 is not needed, the flow through this unit is zero. The above partial derivatives give the following possible values on the set $\mathrm{A} := [0.0, 1.0] \setminus [1.0, 1.0]$:

$$\partial f(A)/\partial x_1 = \frac{3}{5}\left\{\frac{D_1^{0.6}}{[28.0, 29.0]^{0.4}} - \frac{D_2^{0.6}[1.0, 6.0]}{(0.0, 24.0]^{0.4}}\right\}$$

and

$$\partial f(A)/\partial x_2 = -\frac{3}{5}\left\{\frac{D_2^{0.6}[18.0, 23.0)}{(0.0, 24.0]^{0.4}}\right\}.$$

Now while $\partial f(x)/\partial x_2$ is positive for $x \in A$, $\partial f(x)/\partial x_1$ can be both negative and positive for certain D_1, D_2 values.