Simplification of Symbolic-Numerical Interval Expressions*

E. D. Popova
Inst. of Mathematics & Comp. Sci.
Bulgarian Academy of Sciences
Acad. G. Bonchev str., block 8
BG-1113 Sofia, Bulgaria
epopova@iph.bio.bas.bg

C. P. Ullrich
Institute for Informatics
University of Basel
Mittlere str. 142
CH-4056 Basel, Switzerland
ullrich@ifi.unibas.ch

Abstract

Although interval arithmetic is increasingly used in combination with computer algebra and other methods, both approaches — symbolic-algebraic and interval-arithmetic — are used separately. Implementing symbolic interval arithmetic seems not suitable due to the exponential growth in the "size" of the end-points. In this paper we propose a methodology for "true" symbolic-algebraic manipulations on symbolic-numerical interval expressions involving interval variables instead of symbolic intervals. Due to the better algebraic properties, resembling to classical analysis, and the containment of classical interval arithmetic as a special case, we consider the algebraic extension of conventional interval arithmetic as an appropriate environment for solving interval algebraic problems. Based on the distributivity relations, a general framework for simplification of symbolic-numerical expressions involving intervals is given and some of the wider implications of the theory pertaining to interval algebraic problems are discussed.

1 Introduction

Interval arithmetic [1], [16] is widely recognized nowadays as a valuable computing technique. It is increasingly used in combination with symbolic, algebraic and other methods. Answering the objective requirements for controlling round-off errors and handling uncertain input data, the general-purpose computer algebra systems Reduce, Maple, *Mathematica* supply interval arithmetic [6], [12]. The usage of validated computations at critical points of some algebraic algorithms improves the stability of the complete solution [23]. Several hybrid algorithms [4], [8], [14], using floating-point and/or interval arithmetic in intermediate computations, combine the speed of numerical computations with the exactness of symbolic methods providing still guaranteed correct results and a dramatic speed up of the corresponding algebraic algorithm.

Several constraint satisfaction systems, based on combination and/or cooperation of different (including interval) methods, have been proposed during the last years [2], [5], [9]. Constraint solvers act either independently, dynamically sharing their results, or some solvers perform some preprocessing for the others. Most of the systems combine symbolic computations to transform the initial constraint system and interval-based techniques to compute the solutions. It is a common practice that interval arithmetic packages, even developed as part of a computer algebra system, support only numerical interval computations. The interval arithmetic package `INTPACK` of the Maple share library allows computations on intervals involving only floating-point numbers, `infinity` and `FAIL` [6]. The *Mathematica* kernel function `Interval` allows combined usage of all types of exact numbers, mathematical constants and/or exact singletons with inexact (approximate real) numbers at the interval end-points, providing that former are handled exactly and later rounded correctly by the interval arithmetic operations and functions [12]. Symbolic interval arithmetic, however, is not supported "because combinatorial explosion of expressions involving `Min` and `Max` functions would quickly render any symbolic result useless" [13]. The reason for a combined but independent usage of symbolic-algebraic and interval methods is that the well-known interval arithmetic is an incomplete algebraic structure inappropriate for both symbolic and algebraic computations.

Among several extensions of the classical interval arithmetic [1], [16] which have been proposed, we consider that one aiming at its algebraic completion[1]. First developed by H.-J. Ortolf [17] and E. Kaucher [11], further investigated by E. Gardenes et al. [7], S. Markov [15] and others, it is obtained as an extension of the set of conventional (proper) intervals by improper intervals and a corresponding extension of the definitions of all interval arithmetic operations and functions. The obtained extended interval arithmetic structure possesses group properties regarding addition and multiplication. Handling of norm and metric are very similar to norm and metric in linear spaces. An attractive goal is to make use of the algebraic completeness of extended interval arithmetic, embedding it in a computer algebra system, and investigating how the algebraic properties can be exploited for true symbolic-algebraic manipulations on interval expressions, automatic theorem proving, developing of explicit interval algorithms and effective solution of certain

*This work was partially supported by the Bulgarian National Science Fund under grant No. I-507.95.

Permission to make digital/hard copy of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication and its date appear, and notice is given that copying is by permission of ACM, Inc. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. ISSAC'98, Rostock, Germany. ©1997 ACM 0-89791-875-4/ 97/ 0007 $ 3.50

[1]completion to a group structure obtained by valid algebraic constructions [15]

interval algebraic problems.

Some basic formulae and algebraic properties are viewed in Section 2 in concise functional notations. In Section 3 we discuss the implementation of numerical extended intervals in computer-algebra system *Mathematica* [24] and which extra functionalities in comparison to conventional interval arithmetic can be achieved by working in the extended interval space. Section 4 is devoted to the conditionally distributive law on addition and multiplication of extended intervals. Two equivalent forms of the distributive relations are presented — one defining rules how to multiply out a sum of intervals and the other defining rules how to take a common variable out of brackets. Based on the distributivity relations for extended intervals, we give a general framework for simplification of symbolic-numerical expressions involving intervals. A discussion concerning possibilities and usefulness of what has been thought as impossible (or useless) by now — symbolic-algebraic manipulation of interval formulae — is followed by a summary of some of the wider implications of the theory pertaining to interval algebraic problems and some further implementation considerations.

2 Basic Formulae and Properties

The set of conventional (proper) intervals $\{[a^-, a^+] \mid a^- \leq a^+, a^-, a^+ \in R\}$ is extended by the set $\{[a^-, a^+] \mid a^- \geq a^+, a^-, a^+ \in R\}$ of improper intervals obtaining thus the set $D = \{[a^-, a^+] \mid a^-, a^+ \in R\} \cong R^2$ of all ordered couples of real numbers called extended (or directed) intervals. Extended intervals will be denoted by capital letters and $a^\lambda \in R$ with $\lambda \in \Lambda = \{+, -\}$ is the first or second endpoint of $A \in D$ depending on the value of λ. The binary variable λ will be sometimes expressed as a "product" of two or more binary variables, $\lambda = \mu\nu$, $\mu, \nu \in \Lambda$, defined by $++ = -- = +$, and $+- = -+ = -$. Degenerate (point) intervals are those for which $a^- = a^+$.

The inclusion order relation between normal intervals is extended for $A, B \in D$ by

$$A \subseteq B \iff (b^- \leq a^-) \text{ and } (a^+ \leq b^+). \quad (1)$$

Several functionals are used extensively for characterizing extended intervals. For an interval $A \in D$ "direction" $\tau : D \to \Lambda$ is defined by

$$\tau(A) = \begin{cases} +, & \text{if } a^- \leq a^+, \\ -, & \text{otherwise} . \end{cases} \quad (2)$$

An extended interval A is called proper, if $\tau(A) = +$ and improper otherwise. Improper intervals should not be confused with external (Kahan's [10]) intervals, obtained as a result of division by zero containing normal intervals. An extended interval can be considered as a set of values between two real numbers which is equipped with a direction of tracing this set. An interpretation of extended intervals by ranges of monotonous functions leads to some valuable applications. Let $f(x)$ be a continuous, monotonous function over an interval $X = [x^-, x^+]$. $f[X] = [f(x^-), f(x^+)]$ is called "directed range" of f over X. For a proper interval X, the extended interval $f[X]$ is proper, if f is monotonously increasing and $f[X]$ is improper, if f is monotonously decreasing over X (see Figure 1). This way, an extended interval contains information about not only a set of values but also about which direction this set is traced (that is the monotonicity type of a function). Another interpretation of

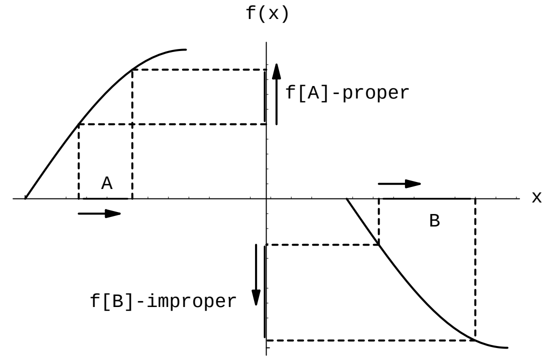

Figure 1: Interpretation of extended (directed) intervals as directed ranges of monotonous functions.

proper and improper intervals as tolerance or control sets can be found in [7]. From an algebraic point of view, improper intervals play the same role in interval arithmetic which negative numbers play in real arithmetic (remember Diophantus equations) [15].

Denote $\mathcal{T} = \{A \in D \mid a^-a^+ \leq 0\}$. For an interval $A \in D \setminus \mathcal{T}$, "sign" $\sigma : D \setminus \mathcal{T} \to \Lambda$ is defined by

$$\sigma(A) = \begin{cases} +, & \text{if } a^{-\tau(A)} > 0, \\ -, & \text{if } a^{\tau(A)} < 0. \end{cases} \quad (3)$$

In particular, σ is well defined over $R \setminus 0$.

Functional $\chi : D \to [-1, 1]$ is defined by

$$\chi_A = \begin{cases} -1, & \text{if } A = [0, 0] \\ a^{-\nu_A}/a^{\nu_A}, & \text{otherwise} , \end{cases} \quad (4)$$

where $\nu_A = \{+, \text{if } |a^+| = |a^-|;\ \ \sigma(|a^+| - |a^-|), \text{otherwise}\}$. Thus $a^{\nu_A} = \{a^+, \text{if } |a^+| \geq |a^-|;\ \ a^-, \text{otherwise}\}$. Functional χ admits the geometric interpretation that A is more symmetric than B iff $\chi_A \leq \chi_B$.

Dual is an important operator that reverses the endpoints of the intervals and expresses an element-to-element symmetry between proper and improper intervals. For $A = [a^-, a^+] \in D$, "dual" is defined by

$$\texttt{Dual}[A] = A_- = [a^+, a^-]. \quad (5)$$

We shall also use the functional notation A_λ with $\lambda \in \Lambda$ and $A_+ = A$.

The arithmetic operations $+$ and $\times$ are extended from the familiar set of normal intervals to D.

$$A + B \;=\; [a^- + b^-, a^+ + b^+], \text{ for } A, B \in D;$$

$$A \times B \;=\; \begin{cases} [a^{-\sigma(B)}b^{-\sigma(A)}, a^{\sigma(B)}b^{\sigma(A)}], & \text{if } A, B \in D \setminus \mathcal{T} \\ [a^{\sigma(A)\tau(B)}b^{-\sigma(A)}, a^{\sigma(A)\tau(B)}b^{\sigma(A)}], & \text{if } A \in D \setminus \mathcal{T},\ B \in \mathcal{T} \\ [a^{-\sigma(B)}b^{\sigma(B)\tau(A)}, a^{\sigma(B)}b^{\sigma(B)\tau(A)}], & \text{if } A \in \mathcal{T},\ B \in D \setminus \mathcal{T} \\ [\min\{a^-b^+, a^+b^-\}, \max\{a^-b^-, a^+b^+\}]_{\tau(A)}, & \text{if } A, B \in \mathcal{T}, \tau(A) = \tau(B) \\ 0, & \text{if } A, B \in \mathcal{T}, \tau(A) = -\tau(B) \end{cases}$$

Interval subtraction and division can be expressed as composite operations $A - B = A + (-1) \times B$ and $A/B =$

$A \times (1/B)$, where $1/B = [1/b^+, 1/b^-]$ if $B \in D \setminus \mathcal{T}$.

$$A - B = [a^- - b^+, a^+ - b^-], \qquad A, B \in D;$$
$$A/B = \begin{cases} [a^{-\sigma(B)}/b^{\sigma(A)}, a^{\sigma(B)}/b^{-\sigma(A)}], \\ \hfill \text{if } A, B \in D \setminus \mathcal{T} \\ [a^{-\sigma(B)}/b^{-\sigma(B)\tau(A)}, a^{\sigma(B)}/b^{-\sigma(B)\tau(A)}], \\ \hfill \text{if } A \in \mathcal{T},\ B \in D \setminus \mathcal{T}. \end{cases}$$

The restrictions of the arithmetic operations to proper intervals produce the familiar operations in the conventional interval space.

D is a conditionally complete lattice regarding $\subseteq$ with the following lattice operations:

$$\inf_{\subseteq}(A, B) = [\max\{a^-, b^-\}, \min\{a^+, b^+\}],$$
$$\sup_{\subseteq}(A, B) = [\min\{a^-, b^-\}, \max\{a^+, b^+\}].$$

Some basic properties of the extended interval arithmetic structure $(D, +, \times, \subseteq)$ are:

1. The operations $\circ \in \{+, \times\}$ are commutative and associative in D.
2. $X = 0$ and $Y = 1$ are the unique neutral elements with respect to $+$ and $\times$ operations. That is for all $A \in D$
   $$A = X + A \Leftrightarrow X = [0, 0]; A = Y \times A \Leftrightarrow Y = [1, 1].$$
3. The substructures $(D, +, \subseteq)$ and $(D \setminus \mathcal{T}, \times, \subseteq)$ are isotone groups [11]. Hence, there exist unique inverse elements $-(A_-)$ and $1/(B_-)$ with respect to the operations $+$ and $\times$ such that
   $$A - A_- = 0 \quad \text{and} \quad B/B_- = 1. \tag{6}$$
4. A conditionally distributive law holds true [18]. Full characterization of the distributive relations will be given in Section 4.1.
5. $A \subseteq B \Longleftrightarrow A_- \supseteq B_-$;
   $$(A \circ B)_- = A_- \circ B_- \quad \text{for} \quad \circ \in \{+, -, \times, /\}. \tag{7}$$

Definition of norm and metric, as well as many topological properties of $(D, +, \times, \subseteq)$ are given in [11]. Some other properties, references and applications of the extended interval arithmetic can be found in [7], [20]. In what follows we shall be concerned with computer algebra implications of the extended interval space.

3 Numerical Directed Intervals

A *Mathematica* [24] package for extended interval arithmetic (`directed.m`) [20] was designed as an experimental demonstrative package intended to provide functionality that can not be obtained by conventional interval arithmetic. At a first stage, *Mathematica* interval capabilities were extended by the definitions of a new data object `Directed` and a number of functions handling numerical extended intervals.

Usually, the symbolic-processing systems do not consider mathematical constants and exact singletons as numerical objects. Some conditional functions also remain unevaluated on these arguments. Due to the many conditionals involved in the interval formulae, a correct handling of intervals involving such quantities at the end-points requires that numerical evaluation takes substantial part of interval arithmetic operations and functions.

Definition 1 *Numerical expression is called any expression whose numerical approximation gives an approximate real number or infinity.*

Definition 2 *Numerical interval is called any interval whose end-points are numerical expressions.*

Data object `Directed` supports numerical extended intervals considered in Section 2. Being an extension of normal intervals, `Directed` intervals contain latter as a special case. Some applications (see e. g. [22]) require handling sets of a finite number of disjoint proper intervals, called Kahan's intervals [10]. `Directed` multi-intervals, supported by `directed.m`, generalize Kahan's intervals but in what follows we shall be concerned only with single directed intervals.

Exact numbers, mathematical constants and exact singletons participating in intervals are handled exactly, while the approximate real numbers are rounded in the corresponding direction according to the strict definitions for outwardly rounded computer operations [21], providing that the resulting interval always encloses the true result according to (1). The outward rounding is performed *a posteriori* rather than as implicit rounding in hardware.

Sometimes, an inner inclusion of the true interval solution can be very useful giving an estimation of the tightness of the obtained outer interval solution. An inner inclusion interval is an interval which is contained in the true solution interval. Some safety problems also search for a minimum set of the solutions instead of an inclusion. Inner inclusions in conventional interval arithmetic can be obtained only if inwardly rounded interval operations are implemented in addition to the outwardly rounded ones which requires an extension of the set of operation symbols. An important property of the extended interval arithmetic is that inner inclusions can be obtained only by outwardly rounded operations and the corresponding dual of the input interval expression [7]. Roundings $\bigcirc, \Diamond : D \longrightarrow RD$ (where RD is the set of computer representable extended intervals) are defined by $\bigcirc A = [\triangle a^-, \triangledown a^+]$ (inward rounding), and $\Diamond A = [\triangledown a^-, \triangle a^+]$ (outward rounding); $\triangledown$, $\triangle$ are the floating-point directed roundings toward $-\infty$ and $+\infty$, respectively. For $A \in D$ we have [7]

$$\texttt{Dual}[\Diamond\texttt{Dual}[A]] = \bigcirc A \subseteq A \subseteq \Diamond A = \texttt{Dual}[\bigcirc\texttt{Dual}[A]].$$

If $\circ \in \{+, -, \times, /\}$ is an operation in D, the properties

$$(\bigcirc A) \circledcirc (\bigcirc B) \subseteq A \circ B \subseteq (\Diamond`A) \mathbin{\Diamond\!\!\!\cdot\,} (\Diamond B)$$
$$\texttt{Dual}[(\Diamond\texttt{Dual}[A]) \mathbin{\Diamond\!\!\!\cdot\,} (\Diamond\texttt{Dual}[B])] = (\bigcirc A) \circledcirc (\bigcirc B)$$

are extended for rational expressions to facilitate obtaining an inner inclusion. In order to give the user the opportunity for both outward and inward rounding of an extended interval involving inexact numbers, an optional parameter `Round`, specifying outward rounding for the intervals, was included in the syntax of the data object `Directed`. A function `R` is defined to give the approximate real interval including the directed interval argument.

Example 1 *Find an interval F, such that*

$$F \subseteq \{(2.3 + b)/c \mid b \in B, c \in C\}, \tag{8}$$

wherein $B, C \in D$ are proper intervals.

In outwardly rounded conventional interval arithmetic we can obtain only

$$(2.3 + B)/C \supseteq \{(2.3 + b)/c \mid b \in B, c \in C\},$$

while in extended interval arithmetic we get

```
F = (2.3 + Dual[R[Dual[B]]]) / Dual[R[Dual[C]]]
```

satisfying (8).

Basic arithmetic on extended intervals is automatic, performed in machine or user-specified precision. For the sake of efficiency interval arithmetic operations and functions are implemented by giving *upvalues* for (definitions associated with) `Directed`. Properties (6), implemented as corresponding rewrite rules for the interval operations provide no blowing-up of the interval result if the arguments involve approximate real numbers.

Symbolic manipulation proved to be an efficient tool for detection and removal of dependency relations between variables and the reduction of the number of occurrences of variables in range computation of interval functions [3]. However, the limited possibilities for reduction and the varying character of expressions mean that we never can be sure to have been producing the best computable form for an expression (if existing) but only a more suitable one. Reduction of the dependency problem in range computation of rational interval functions can be simply achieved working in the extended interval space [7].

Example 2 *Compute the exact range of function*

$$f(t) \;=\; \frac{t+[1/5,2]}{t-[1/4,7/3]} \qquad over \;\; T=[3,36/5].$$

By conventional interval arithmetic we obtain interval $[64/139, 69/5]$ *for the range of* f. *Because* $f(t)$ *is monotonously decreasing on* t *over* T, *monotonously increasing on* t *in the numerator and monotonously decreasing on* t *in the denominator, we can apply a theorem from [7] to eliminate the effect of multi-incidence of variable* t. *According to this theorem*

$$f(T) \;=\; \frac{\texttt{Dual}[T]+[1/5,2]}{T-[1/4,7/3]},$$

obtaining thus $[148/139, 15/2]$*, which is the exact range.*

4 Symbolic-Algebraic Interval Computations

Having implemented numerical extended intervals we would like to utilize the algebraic extension of interval arithmetic to handle algebraic formulae including numerical intervals and/or interval variables. Algebraic manipulations involve simplifying rational expressions and finding algebraic solutions for several kinds of equations. Due to the existence of inverse additive and multiplicative elements in D, we can solve some simple kinds of interval algebraic equations by elementary algebraic transformations, which is not possible in conventional interval arithmetic. To be able to transform interval expressions into other interval expressions or to solve more complicated equations involving intervals we need rules how to multiply out interval sums and how to take a common variable out of parentheses in interval expressions. These rules are given by the distributive relations between extended intervals.

4.1 Interval Distributive Relations

In this section we present the conditionally distributive law for multiplication and addition of extended intervals.

For any $A \in D$, define $\mu(A) = \begin{cases} \sigma(A), & \text{if } A \in D\setminus\mathcal{T}; \\ \tau(A), & \text{if } A\in\mathcal{T} \end{cases}$

and distinguish between five assumptions which are used in the following two theorems.

Theorem 1 *Let* $A_i,\ i=1,\ldots,n$ *and* C *be extended intervals. Denote* $\sum_{i=1}^n A_i = S$. *The equality*

$$\sum_{i=1}^{n}\left(A_i\times C_{\mu(A_i)}\right) \;=\; \left(\sum_{i=1}^{n}A_i\right)\times C_{\mu(S)}, \qquad (9)$$

holds true iff exactly one of the assumptions i) to v) holds true.

i) $A_i, S \in D\setminus\mathcal{T},\ i=1,\ldots,n$ and $C\in D$;

ii) $A_i \in D\setminus\mathcal{T}, i=1,\ldots,n,\ \ S\in\mathcal{T},$ and

either $C=c\in R$,
or $C\in\mathcal{T},\ \ S=0$,
or $C\in\mathcal{T},\ \ \tau(C)=\{+, \text{if Th1}; \tau(S), \text{if Th2}\}$,
$\chi_C\le\chi_S,\ \ \nu_S=+$;

iii) $A_i, S\in\mathcal{T},\ i=1,\ldots,n$ and

either $C\in D\setminus\mathcal{T}$,
or $C\in\mathcal{T}, \{\tau(C)=-, \text{if Th1}; \tau(C)\ne\tau(S), \text{if Th2}\}$,
or $C\in\mathcal{T}, \tau(C)=\{+, \text{if Th1}; \tau(S), \text{if Th2}\}$, and
either $\forall i,j=1,..,n\ \ \tau(A_i)=\tau(A_j)$,
$\begin{cases}\chi_C\ge\chi_{A_i}, \text{ or}\\ \chi_C\le\chi_{A_i}, \nu_{A_i}=\nu_{A_j},\end{cases}$
or $\exists p,q$ so that $\tau(A_p)\ne\tau(A_q)$ and
$\chi_C\le\min\{\chi_{A_i},\chi_S\},\ \nu_{A_i}=\nu_S\ \forall i=1,...,n$;

iv) $A_i\in\mathcal{T},\ i=1,\ldots,n,\ \ S\in D\setminus\mathcal{T}$ and

either $C=c\in R$,
or $C\in\mathcal{T}, \{\tau(C)=-, \text{if Th1}; \tau(C)\ne\sigma(S), \text{if Th2}\}$,
$s^-=0$,
or $C\in\mathcal{T}, \tau(C)=\{+, \text{if Th1}; \sigma(S), \text{if Th2}\}$,
$\nu_{A_i}=+,\ \chi_C\le\chi_{A_i}, i=1,\ldots,n$;

v) there exist index sets $P,Q\ne\emptyset,\ P\cup Q=\{1,\ldots,n\}$, $P\cap Q=\emptyset$ such that $A_p\in D\setminus\mathcal{T}$ for $p\in P$, $A_q\in\mathcal{T}$ for $q\in Q$, and

either $C=c\in R$,
or $C\in\mathcal{T},\ \ \tau(C)=\{+, \text{if Th1};\ \mu(S), \text{if Th2}\}$,
$\chi_C\le\min_{q\in Q}\{\chi_{A_q}\},\ \ \nu_{A_q}=+$,
or $C\in\mathcal{T}, \{\tau(C)=-, \text{if Th1}; \tau(C)\ne\mu(S), \text{if Th2}\}$
and $\begin{cases}\sum_{q\in Q}a_q^-=0, & \text{if } S\in D\setminus\mathcal{T},\\ \sum_{p\in P}a_p^-=0, & \text{if } S\in\mathcal{T}.\end{cases}$

Theorem 1 gives rules how and when we can take a common multiplier out of brackets. Another equivalent distributive relation giving rules for multiplying out a sum of extended intervals is presented by the next theorem.

Theorem 2 *Let* $A_i,\ i=1,\ldots,n$ *and* C *be extended intervals. Denote* $\sum_{i=1}^n A_i=S$. *The equality*

$$\left(\sum_{i=1}^{n}A_i\right)\times C \;=\; \sum_{i=1}^{n}\left(A_i\times C_{\mu(A_i)\mu(S)}\right), \qquad (10)$$

holds true iff exactly one of the assumptions i) to v) holds true.

A complete proof of the above two theorems is given in [18]. From the conditionally distributive law we can obtain the special cases of general distributivity:

Corollary 1 *For any $A_i \in D, i = 1, \ldots, n$ and $C = c \in R$*

$$A_1 \times c + A_2 \times c + \ldots + A_n \times c \;=\; \left(\sum_{i=1}^{n} A_i\right) \times c.$$

Corollary 2 *For any $C \in D$ and $A_i = a_i \in R$, $i = 1, \ldots, n$ and $s = \sum_{i=1}^{n} a_i$, it is*

$$a_1 \times C_{\sigma(a_1)} + \ldots + a_n \times C_{\sigma(a_n)} \;=\; \left(\sum_{i=1}^{n} a_i\right) \times C_{\sigma(s)}.$$

Corollary 3 *For $A_i, C \in D$ such that $\tau(C) = \tau(A_i) = +$, $i = 1, ..., n$, the equality*

$$C \times (A_1 + \ldots + A_n) = C \times A_1 + \ldots + C \times A_n \tag{11}$$

holds true iff exactly one of $\hat{i}$), $\hat{iii}$), $\hat{v}$) holds true.

- *$\hat{i}$) $A_i \in D \setminus \mathcal{T}, \sigma(A_i) = \sigma(A_j), i, j = 1, ..., n$, $C \in D$;*
- *$\hat{iii}$) $A_i \in \mathcal{T}$, $i = 1, \ldots, n$ and*

  either $C \in D \setminus \mathcal{T}$;
  or $C \in \mathcal{T}$, $\chi_C \geq \max_{i=1}^{n}\{\chi_{A_i}\}$;
  or $C \in \mathcal{T}$, $\chi_C \leq \min_{i=1}^{n}\{\chi_{A_i}\}$, $\nu(A_i) = \nu(A_j)$;
- *$\hat{v}$) $\exists P, Q \neq \emptyset$, $P \cup Q = \{1, \ldots, n\}, P \cap Q = \emptyset$ such that $A_p \in D \setminus \mathcal{T}$ for $p \in P$, $A_q \in \mathcal{T}$ for $q \in Q$, and*

  either $C = c \in R$;
  or $C \in \mathcal{T}$, $\chi_C \leq \min_{q \in Q}\{\chi_{A_q}\}$, $\sigma(A_p) = \nu(A_q) = +$.

For proper intervals, Theorem 2 and Theorem 1 give the same characterization of the equality (11) with the exception that in case $\hat{\text{i}}$) all additive terms must have positive signs in order to take a common multiplier out of brackets.

4.2 Applying the Distributive Laws

As intended to be appropriate for the broadest range of calculations some of the built-in computer algebra rules are not valid for extended intervals. For example,

```
In[1] := Directed[{1, 7}] x + Directed[{3, 2}] x
Out[1] = x Directed[{1, 7}] + x Directed[{3, 2}]

In[2] := 3 x + x^2 - 4 x
Out[2] = - x + x^2
```

Simplification of the expression `In[1]` is possible but has not been done because `Mathematica` automatically simplifies expressions involving only numbers. Simplification `Out[2]`, however, is not valid if the symbolic variable `x` represents a non-degenerate (extended) interval. To model the algebra of extended intervals we have to

- distinguish between symbols representing non-degenerate intervals and symbols representing point intervals or other objects for which built-in rules are valid;
- define new transformation rules corresponding to the specific algebra of extended intervals.

Any symbol (name of variable) can represent an extended interval if its type is explicitly specified as `Directed`. A symbol `symb` can be considered as directed interval in *Mathematica* by the explicit assignment

```
symb /: Head[symb] = Directed
```

where the kernel `Head` function identifies the type of the objects. This way we can use symbols representing extended intervals instead of symbolic data objects `Directed` (e. g. `Directed[{a, b}]`). Symbols without explicit type assignment are considered as degenerate (point) intervals for which the built-in algebraic rules are valid.

Definition 3 *An expression is called interval expression if it involves at least one numerical directed interval or symbol representing directed interval.*

The predicate `DirectedQ` returns true for any interval expression. The predicate `NumericQ` returns true for any numerical expression.

We consider symbolic-numerical expressions being finite interval sums involving two-terms products of a common symbolic multiplier and a coefficient which is either a numerical expression or a numerical directed interval. In what follows simplification of such expressions by taking the common variable out of parentheses is discussed.

Due to the associativity of interval addition the algorithm for simplification of a finite sum is reduced to a recursive execution of an algorithm for simplification of a two-terms symbolic-numerical interval sum. The following two corollaries of Theorem 1 specify how to take a common variable in such a sum out of parentheses.

Corollary 4 *Let A, B, T be directed intervals such that $\mu(A) = \mu(B)$ and $T \notin R$. The equality*

$$A \times T + B \times T \;=\; (A + B) \times T \tag{12}$$

holds true if and only if **a1***, or* **a2***, or* **a3***.*

a1. *$A, B \in D \setminus \mathcal{T}$ and $T \in D$;*

a2. *$A, B \in \mathcal{T}$ and*

either $T \in D \setminus \mathcal{T}$,
or $T \in \mathcal{T}$, $\tau(T) \neq \tau(A)$ and
$$\begin{cases} \chi_T \geq \max\{\chi_A, \chi_B\}, & \textit{or} \\ \chi_T \leq \min\{\chi_A, \chi_B\}, \quad \nu_A = \nu_B; \end{cases}$$

a3. *$A \in D \setminus \mathcal{T}$, $B \in \mathcal{T}$, $T \in \mathcal{T}$ and*
$$\begin{cases} \chi_T \leq \chi_B,\ \nu_B = +, & \textit{if } \tau(T) = \sigma(A) \\ a^- = 0, & \textit{otherwise.} \end{cases}$$

Corollary 5 *Let A, B, T be directed intervals such that $\mu(A) \neq \mu(B)$ and $T \notin R$. The equality*

$$A \times T + B \times T_- \;=\; (A + B) \times T_{\mu(A+B)\mu(A)} \tag{13}$$

holds true iff exactly one of the assumptions **b1** *to* **b5** *holds true.*

b1. *$A, B, A + B \in D \setminus \mathcal{T}$, and $T \in D$;*

b2. *$A, B \in D \setminus \mathcal{T}$, $T \in \mathcal{T}$, $A + B \in \mathcal{T}$*
$$\begin{cases} A + B = 0, & \textit{or} \\ \tau(T) = \sigma(A),\ \chi_T \leq \chi_{A+B},\ \nu_{A+B} = +; \end{cases}$$

b3. $A, B, A+B \in \mathcal{T}$ *and*

either $T \in D \setminus \mathcal{T}$,

or $T \in \mathcal{T}$, $\tau(T) \neq \tau(A)$, $\chi_T \leq \min\{\chi_A, \chi_B, \chi_{A+B}\}$, $\nu_A = \nu_B = \nu_{A+B}$;

b4. $A, B \in \mathcal{T}$, $A+B \in D \setminus \mathcal{T}$, $T \in \mathcal{T}$ *and*

$$\begin{cases} a^- + b^- = 0, & \text{if } \tau(T) \neq \tau(A), \\ \nu_A = \nu_B = +,\ \chi_T \leq \min\{\chi_A, \chi_B\}, & \text{otherwise}; \end{cases}$$

b5. $A \in D \setminus \mathcal{T}$, $B \in \mathcal{T}$, $T \in \mathcal{T}$ *and*

$$\begin{cases} a^- = 0, & \text{if } \tau(T) \neq \sigma(A), \\ \nu_B = +,\ \chi_T \leq \chi_B, & \text{otherwise.} \end{cases}$$

The general implementation scheme is based on the mechanism of pattern-matched rewrite rules. The database of rewriting rules for simplification of symbolic-numerical interval expressions is built of three types of rewrite rules:

• For the special case of Corollary 1 define a rewrite rule simplifying a two-terms symbolic-numerical interval sum where the common multiplier is a non-interval symbolic expression.

```
a_. x_ + b_. x_ :=(a + b) x/; Not[DirectedQ[x]] &&
 (MatchQ[a, Directed[{_?NumericQ, _?NumericQ}]] ||
      NumericQ[a])  &&
 (MatchQ[b, Directed[{_?NumericQ, _?NumericQ}]] ||
      NumericQ[b]) ;
```

• For the special case of Corollary 2 define rewrite rules transforming every numerical expression, involved in a two-terms symbolic-numerical interval sum with interval common symbolic multiplier into numerical degenerate directed interval.

```
a_. x_?DirectedQ + b_. x_?DirectedQ :=
       Directed[a] x + Directed[b] x /;
       NumericQ[a] || NumericQ[b];

a_. x_?DirectedQ + b_. Dual[x_?DirectedQ] :=
       Directed[a] x + Directed[b] Dual[x] /;
       NumericQ[a] || NumericQ[b] ;
```

• According to Corollary 4.3 and Corollary 4.4, define rewrite rules simplifying a two-terms symbolic-numerical interval sum with interval common symbolic multiplier and interval numerical coefficients.

```
a_ x_?DirectedQ + b_ x_?DirectedQ := (a + b) x /;
MatchQ[a, Directed[{_?NumericQ, _?NumericQ}]] &&
MatchQ[b, Directed[{_?NumericQ, _?NumericQ}]] &&
    ( set of conditions a1.- a3. );

a_ x_?DirectedQ + b_ Dual[x_?DirectedQ] :=
   If[mu[a+b] === 1, (a+b) x, (a+b) Dual[x]] /;
MatchQ[a, Directed[{_?NumericQ, _?NumericQ}]] &&
MatchQ[b, Directed[{_?NumericQ, _?NumericQ}]] &&
    ( set of conditions b1.- b5. );
```

These rewrite rules are tried successively in the above order in which they are defined. Except for the first one, the above groups of rewrite rules are divided into two subgroups depending on the syntax of the patterns in the left-hand side of the rules:

• patterns describing the left side of equality (12), i.e. patterns involving the same common symbolic multiplier

```
a_ x_?DirectedQ + b_ x_?DirectedQ
```

• patterns describing the left side of equality (13), i.e. patterns involving two dual common symbolic multipliers

```
a_ x_?DirectedQ + b_ Dual[x_?DirectedQ]
```

Due to the commutativity of interval addition and multiplication operations just one pattern is enough to cover all possible cases of this subgroup.

The usage of a condition containing labels that appear in a pattern narrows down the requirements of the pattern match and a match is possible only if condition returns true. Simplifying expressions follows the general principle: Take the input expression and find those rewrite rules whose pattern matches part of the expression. That part is then replaced by the replacement text of that rule. Evaluation then proceeds by searching for further matching rules until no more are found. Due to commutativity and associativity of addition and multiplication the arguments of `Plus` and `Times` functions are rearranged and all possible orders of arguments are tested in trying to match patterns of the above distributive rules.

```
In[3] := x /: Head[x] = Directed;

In[4] := Directed[{2, 7}] x - x^2 + Directed[{3, 5}] x
Out[4] = -x^2 + x Directed[{5, 12}]
```

Note that cases **a2** and **a3** for $\tau(T) \neq \mu(A)$ in Corollary 4, respectively cases **b3, b4** and **b5** for $\tau(T) \neq \mu(A)$ in Corollary 5 result in an interval product equal to zero. An overloading of the multiplication operation to deliver zero on multiplication of a numerical and a symbolic extended intervals, appropriately specified, provides an *a priory* simplification of such products to zero. Thus there is no need to involve a condition $\tau(T) \neq \mu(A)$ in the rewrite rules corresponding to the above cases of Corollaries 4 and 5.

The requirements of Theorem 1 for taking a non-degenerate common multiplier out of brackets can be classified in three categories depending on the common multiplier:

s1. for any value of the common multiplier;

s2. for a common multiplier from $D \setminus \mathcal{T}$;

s3. for a common multiplier from $\mathcal{T}$ depending in addition on its direction and relations involving its χ value;

`Mathematica` functions `Sign`, `Direction` and `Chi` are defined to give the value of the corresponding functional σ, τ, χ for a numerical directed interval. `Sign` and `Direction` functions yield integer values 1 and -1 corresponding to "+" and "−", and 0 is returned for the sign of a directed interval from $\mathcal{T}$. Thus, the value of the `Sign` function is used to identify whether an interval is from $\mathcal{T}$ or from $D \setminus \mathcal{T}$. Beside a small number of cases for which simplification of interval expressions is possible for any value of the common multiplier, an explicit assignment to the values of `Sign`, `Direction` and `Chi` functions, associated with the common multiplier, is required for simplification of an interval expression. For a symbol `x`, representing directed interval, case $s2$. is characterized by `Sign[x]`$\neq 0$ independently of its value. If simplification of an interval expression is possible only for a common multiplier from $\mathcal{T}$, an explicit assignment of values to `Sign`, `Direction` and `Chi` functions associated with the common multiplier is required. That is, specification `x/:Sign[x]=0` requires also specification of `Direction[x]` and `Chi[x]`. Usually these values are known *a priori* from the context of the problem we are solving. Most frequently, conventional interval problems are solved by the extended interval arithmetic, so that proper intervals are sought and most practical problems seek for intervals not involving zero. If there is no *a priori* information about a common symbolic multiplier, the solution of the problem should be split into at most three subproblems ($s1$, $s2$ and $s3$).

Two functions `On/Off[IntervalSimplification]` are defined to facilitate the user as much as possible. These functions switch on/off printing prompt messages about possible simplification of any interval subexpression. Generating messages when *Mathematica* tries to simplify an expression is switched off by default.

```
In[5] := Directed[{2, 7}] x - Directed[{5, 3}] Dual[x]
Out[5] = x Directed[{2, 7}]+Directed[{-3, -5}] Dual[x]

In[6] := On[IntervalSimplification]

In[7] := Directed[{2, 7}] x - Directed[{5, 3}] Dual[x]
     IntervalSimplification::usage1:
      "Directed[{2, 7}] x + Directed[{-3, -5}] Dual[x]"
     will be simplified if Sign[x]=0,
     Direction[x]=1, Chi[x]<=-(1/2).
Out[7] = x Directed[{2, 7}]+Directed[{-3, -5}] Dual[x]

In[8] := x /: Direction[x] = 1;  x /: Sign[x] = 0;
     x /: Chi[x] = -2/3;

In[9] := In[16]
Out[9] = Directed[{-1, 2}] Dual[x]
```

If a common multiplier is not a single symbolic variable but an interval expression, the assignments should be done to the whole expression. Further research is necessary for the definition of functions `Sign`, `Direction` and `Chi`, so that they automatically determine the corresponding value for an arbitrary symbolic-numerical interval expression. A solution of this problem will allow the definition of a function `IntervalExpand` designed to disclose the parentheses around symbolic-numerical interval expressions according to Theorem 2.

Distributivity (7) of the `Dual` operator on the arithmetic operations is another key point of the knowledge database for symbolic manipulation of interval expressions. Function `ExpandDual[expr]` is defined to do all possible expansions of the `Dual` function around sums, products and powers. Actually this function transforms the `Dual` of a sum into a sum of dual terms, the `Dual` of a product into a product of dual terms and the `Dual` of a power into the power of a dual argument everywhere in an expression.

Now, we can turn back to the interval algebraic equations and show the application of distributivity relations for their solution.

Example 3 *Find a positive proper interval t (if exists) which is the algebraic solution to the equation*

$$\frac{(b + c \times t) \times \mathtt{Dual}[a]}{\sqrt{2} + a \times \mathtt{Dual}[t] + \mathtt{Dual}[t]/b} \quad = \quad b, \qquad (14)$$

where $a = [1/2, 3/5], b = [2, 3], c = [15/2, 19/2]$.

We specify in *Mathematica* that the symbol `t` represents a directed interval and input a symbolic-numerical expression specifying equation (14), where a, b, and c are replaced by their numerical values. The obtained equation

$$\frac{([2, 3] + [15/2, 19/2]\ t)\ [3/5, 1/2]}{\sqrt{2} + [5/6, 11/10]\ \mathtt{Dual}[t]} \quad = \quad [2, 3]$$

shows that due to the implemented rewrite rules *Mathematica* has automatically simplified the denominator in left-hand side of the equation. We solve this equation by applying elementary transformations, based on the algebraic identities (6). First divide both sides of the equation by dual of its right-hand side Then multiplying both sides of the equation by dual of the denominator in the left-hand side delivers the equivalent equation

$$([2, 3] + [\frac{15}{2}, \frac{19}{2}]\ t)\ [\frac{3}{10}, \frac{1}{6}] \quad = \quad \mathtt{Dual}[\sqrt{2} + [\frac{5}{6}, \frac{11}{10}]\ \mathtt{Dual}[t]]$$

Subtracting from both sides of the last equation dual of its right-hand side we obtain next equivalent equation

$$([2, 3] + [\frac{15}{2}, \frac{19}{2}]\ t)\ [\frac{3}{10}, \frac{1}{6}] - \sqrt{2} - [\frac{5}{6}, \frac{11}{10}]\ \mathtt{Dual}[t] \quad = \quad 0$$

To proceed later we need to disclose parentheses in the above equation which we can do because the requirements of Theorem 2 are fulfilled under the assumptions for `t`. By that, we obtain another equivalent equation

$$[\frac{3}{5} - \sqrt{2}, \frac{1}{2} - \sqrt{2}] \ + \ [\frac{23}{20}, \frac{3}{4}]\ t \quad = \quad 0$$

showing that another automatic simplification has been taken effect. Now, the sought solution $[(-12+20\sqrt{2})/23, (-2+4\sqrt{2})/3]$ is obtained as dual of the quotient of the negative intercept and the coefficient of `t`.

This example shows that the distributive law for extended intervals is an indispensable tool for the reduction interval algebraic equations, with multi-incidence of the unknown variable, to simpler ones. The general normal form of simplified interval algebraic equations is given in [19]. This is helpful for the explicit algebraic solution of some interval equations which are not linear in generall. For example, the interval equation

$$\frac{[7, -11] + [1, 5] \times X}{X} \quad = \quad [3, 2], \qquad 0 \notin X$$

is algebraically equivalent to the equation

$$[1, 5] \times X + [-3, -2] \times X \quad = \quad [-7, 11], \qquad 0 \notin X.$$

However in D, like in conventional interval arithmetic, there are only conditionally valid distributive relations and therefore these equations are not linear. Left-hand side of last equation cannot be further simplified and according to [19] the equation posesses four algebraic solutions: $X_1 = [2, 3]$ and $X_2 = [-15, -34]$. $X_3 = [-7/2, 11/2]$, $X_4 = [7, -11]$, the first two being algebraic solutions to the initial equation. Automatic simplification of symbolic-numerical interval expressions is also helpful for the reduction of the round-off errors (when rational arithmetic is not used) due to the reduced number of arithmetic operations in the simplified equation. The techniques applied above can be used in definition of function `IntervalSolve` giving all numerical and/or parametric solutions to certain kinds interval algebraic equations, providing thus facilities for true symbolic-algebraic computations.

5 Notes on the Applications

The right approach in applying interval distributive relations is not to restrict these relations to the set of normal intervals but to transform the initial problem in terms of extended intervals and to solve the new problem in an algebraically straightforward way. Thus, we obtain the opportunity to apply distributive relations which are valid in many more cases for extended intervals than for normal intervals (compare Corollary 3 and Theorem 2). Note, that the problems in Examples 1 and 2 were formulated in terms of normal intervals but solved efficiently by extended interval arithmetic. Due to the lack of space, the preliminary considerations that had led to equation (14) were omitted.

Of course, extended interval arithmetic is not an universal cure for all interval pains. However, in many cases extended interval arithmetic gives functionalities that cannot be achieved by conventional interval arithmetic. Examples 1, 2 and 3 give hints for some basic applications. Other class of applications is illustrated in [7]. More sophisticated practical applications can be found in the theory of quality control, interpolation and parameter identification, etc. (see also http://ima.udg.es/SIGLA/X/mod_interval).

6 Conclusion

The algebraic properties of extended interval arithmetic make it a powerful tool for explicit solution of some interval algebraic problems and the best environment for exploiting these properties is a computer algebra system. The implemented facilities for simplification of symbolic-numerical interval expressions allow not only an easy computation and exploration in the algebra of extended intervals. We made a first step in constructing a methodological framework and a knowledge database for performing true symbolic algebraic manipulation on interval expressions. A next step involves development of tools for automatic solution of classes interval equations.

Calculating with interval variables is a novel approach in combining symbolic and interval computations. Integrating the algebra of extended intervals and this approach in some general purpose systems would increase the efficiency of interval applications.

References

[1] Alefeld, G.; Herzberger, J.: *Introduction to Interval Computations.* Academic Press, 1983.

[2] Benhamou, F.; Mc Allester, D.; Van Hentenryck, P.: *CLP (Intervals) Revisited.* In Proceedings of ILPS'94, Ithaca, NY, USA, 1994, pp. 124–138.

[3] Caplat, G.: *Symbolic Preprocessing in Interval Function Computing.* In Goos, G.; Hartmanis J. (Eds.): *Symbolic and Algebraic Computation.* Lecture Notes in Computer Science **72**, Springer, 1979, pp. 369–382.

[4] Collins, G. E.; Krandick, W.: *A Hybrid Method for High Precision Calculation of Polynomial Real Roots.* In Bronstein, M. (Ed.): Proceedings of the 1993 International Symposium on Symbolic and Algebraic Computation, ACM Press, 1993, pp. 47–52.

[5] Colmerauer, A.: *Specifications of Prolog IV*, 1996.

[6] Connell, A. E.; Corless, R. M.: *An Experimental Interval Arithmetic Package in Maple.* Interval Computations, No. 2, 1993, pp. 120–134.

[7] Gardeñes, E.; Trepat, A.: *Fundamentals of SIGLA, an Interval Computing System over the Completed Set of Intervals.* Computing, **24**, 1980, pp. 161–179.

[8] Hong, H.: *An Efficient Method for Analyzing the Topology of Plane Real Algebraic Curves.* Mathematics and Computers in Simulations, **42**, 1996, pp. 571–582.

[9] Hyvönen, E.; De Pascale, S.; Lehtola, A.: *Interval Constraint Programming in C++.* In Mayoh, B.; Tyugu, E.; Penham, J. (Eds.): *Constraint Programming.* NATO Advanceed Science Institute, Series F, Springer, 1994.

[10] Kahan, W. M.: *A More Complete Interval Arithmetic.* Lecture Notes for a Summer Course at the University of Michigan, 1968.

[11] Kaucher, E.: *Interval Analysis in the Extended Interval Space IR.* Computing Suppl. **2**, 1980, pp. 33–49.

[12] Keiper, J. B.: *Interval Arithmetic in* Mathematica. Interval Computations, No. 3, 1993, pp. 76–87.

[13] Keiper, J. B.: *Interval Computation.* In *Major New Features in Mathematica Version 2.2.* Technical Report, Wolfram Research, 1993, pp. 20–23.

[14] Krandick, W.: *Isolierung reeller Nullstellen von Polynomen.* In Herzberger, J. (Ed.): *Wissenschaftliches Rechnen.* Akademie Verlag, 1995, pp. 105–154.

[15] Markov, S. M.: *On the Algebra of Intervals and Convex Bodies.* J. UCS **4**, 1, 1998, pp. 34–47. (http://www.iicm.edu/jucs_4_1)

[16] Moore, R. E.: *Interval Analysis.* Prentice Hall, Englewood Cliffs, N. J., 1966.

[17] Ortolf, H. -J.: *Eine Verallgemeinerung der Intervallarithmetik.* Gesellschaft für Mathematik und Datenverarbeitung, Bonn, **11**, 1969, pp. 1–71.

[18] Popova, E. D.: *Generalized Interval Distributive Relations.* Preprint No 2, Institute of Mathematics & Computer Science, BAS, February 1997, pp. 1–18.

[19] Popova, E. D.: *Algebraic Solutions to a Class of Interval Equations.* J. UCS **4**, 1, 1998, pp. 48–67.

[20] Popova, E. D.; Ullrich, C. P.: *Directed Interval Arithmetic in Mathematica: Implementation and Applications.* TR 96-3, Univ. Basel, 1996, pp. 1–56. (http://www.math.acad.bg/~epopova/directed.html)

[21] Popova, E. D.; Ullrich, C. P.: *Generalising BIAS Specification.* J. UCS **3**, 1, 1997, pp. 23–41.

[22] Shackell, J.: *Asymptotic Estimation of Oscillating Functions Using an Interval Calculus.* In Gianni, P. (Ed.): *Symbolic and Algebraic Computation.* LN in Computer Science **358**, Springer, 1989, pp. 481–489.

[23] Stetter, H. J.: *Analysis of Zero Clusters in Multivariate Polynomial Systems.* In Lakshman, Y. N. (Ed.): Poceedings of the 1996 International Symposium on Symbolic and Algebraic Computation, ACM Press, 1996, pp. 127–136.

[24] Wolfram, S.: Mathematica *A System for Doing Mathematics by Computer.* Addison-Wesley, 1991 (2nd ed.).