Evaluation of expressions involving higher order derivations

Robert Grossman[*]
University of Illinois at Chicago

November, 1989

This is a draft of a paper which later appeared in the Journal of Mathematical Systems, Estimation, and Control, Vol. 1, 1991, pp. 91-106.

Abstract

Let R be a commutative ring and $E_1, \ldots, E_M$ be derivations of R which are defined in terms of derivations $D_1, \ldots, D_N$ which commute with each other. In this paper we examine the problem of rewriting expressions involving the $E_1, \ldots, E_M$ in terms of the $D_1, \ldots, D_N$ in such a way as to handle efficiently any cancellation occuring due to the commuting of the $D_1, \ldots, D_N$. Roughly speaking we introduce a data structure which allows us to organize the computation in such a way as to take advantage of the symmetries in the expression and reduce the operation count.

1 Introduction

Let R be a commutative ring with a unit element. A derivation of R is a map D of R to itself satisfying

$$\begin{aligned} D(a+b) &= D(a) + D(b) \\ D(ab) &= aD(b) + bD(a) \qquad \text{for all } a, b \in R. \end{aligned}$$

Let $D_1, \ldots, D_N$ be N commuting derivations of R; that is for $i, j = 1, \ldots, N$,

$$D_i D_j a = D_j D_i a, \qquad \text{for all } a \in R.$$

[*]This research was supported in part by NASA grant NAG 2-513.

Suppose that we are also given M derivations $E_1, \ldots, E_M$ of R which can be expressed as $R-$linear combinations of the former derivations; that is for $j = 1, \ldots, M$,

$$E_j = \sum_{\mu=1}^{N} a_j^\mu D_\mu, \qquad \text{where } a_j^\mu \in R.$$

We are interested in performing efficient computations involving the higher order derivations generated by the $E_1, \ldots, E_M$. Such higher order derivations can be expressed in terms of the commuting derivations $D_1, \ldots, D_N$, but often times this rewriting involves cancellation. In this paper we examine the problem of rewriting expressions involving the $E_1, \ldots, E_M$ in terms of the $D_1, \ldots, D_N$ in such a way as to efficiently handle any cancellation occuring due to the commuting of the $D_1, \ldots, D_N$. Roughly speaking we introduce a data structure which allows us to organize the computation in such a way as to take advantage of the symmetries in the higher order derivation and reduce the operation count.

An important example is when $R = C^\infty(\mathbf{R}^N)$,

$$D_1 = \partial/\partial x_1, \ldots, D_N = \partial/\partial x_N,$$

and $E_1, \ldots, E_M$ are first order differential operators, viewed as derivations of R. In this case a higher order derivation generated by $E_1, \ldots, E_M$ is simply a differential operator generated by the vector fields $E_1, \ldots, E_M$. More generally, let $E_1, \ldots, E_M$ be vector fields defined on a manifold M, viewed as derivations of the ring of smooth functions $R = C^\infty(M)$ on M, and let

$$D_1 = \partial/\partial x_1, \ldots, D_N = \partial/\partial x_N,$$

where $x_1, \ldots, x_N$ are a system of local coordinates. The study of such differential operators began with [2].

In the remainder of this section we give a more precise statement of the problem. This requires some additional notation. Let K denote a subring of R with the property

$$D_1 a = 0, \ldots, D_N a = 0, \qquad \text{for all } a \in K,$$

and let

$$K < E_1, \ldots, E_M >$$

denote the free associative K-algebra over the set consisting of $E_1, \ldots, E_M$. If $p \in K < E_1, \ldots, E_M >$, then

$$L = p(E_1, \ldots, E_M)$$

is a $K-$linear map of R. This map may be thought of as a higher order derivation of R generated by the $E_1, \ldots, E_M$. Let **Diff** $(D_1, \ldots, D_N; R)$ denote the space of formal linear differential operators with coefficients from R; that is **Diff** $(D_1, \ldots, D_N; R)$ consists of all formal expressions

$$L = \sum_{\mu_1=1}^{N} a_{\mu_1} D_{\mu_1} \quad + \quad \sum_{\mu_1,\mu_2=1}^{N} a_{\mu_1,\mu_2} D_{\mu_2} D_{\mu_1} \\ \quad + \quad \ldots + \sum_{\mu_1,\ldots,\mu_k=1}^{N} a_{\mu_1,\ldots,\mu_k} D_{\mu_k} \cdots D_{\mu_1},$$

where $a_{\mu_1}, a_{\mu_1,\mu_2}, \ldots, a_{\mu_1,\ldots,\mu_k} \in R$. We say that $L \in$ **Diff** $(D_1, \ldots, D_N; R)$ and $L' \in$ **Diff** $(D_1, \ldots, D_N; R)$ *agree* if

$$L(a) = L'(a), \qquad \text{for all } a \in R.$$

Fix an element $p \in K{<}E_1, \ldots, E_M{>}$ and suppose that p contains l terms, each of which is homogeneous of degree m. If we make the substitution

$$E_j = \sum_{\mu=1}^{N} a_j^{\mu} D_{\mu}, \qquad \text{where } a_j^{\mu} \in R,$$

in the expression p, and use the fact that $D_1, \ldots, D_N$ are derivations of R, we get a differential operator $L^{\#} \in$ **Diff** $(D_1, \ldots, D_N; R)$. It is easy to see that $L^{\#}$ contains $lm!\,N^m$ terms involving $D_1, \ldots, D_N$. In this paper we discuss the question whether we can find an operator $L \in$ **Diff** $(D_1, \ldots, D_N; R)$ which agrees with $L^{\#}$, but which requires fewer operations involving the derivations $D_1, \ldots, D_N$. We show that such an L does indeed exist in case p has a certain symmetry, which we call a symmetry decomposition. In §5 we give the exact definitions and prove the following theorem.

Theorem 1.1 *Fix an expression $p \in K{<}E_1, \ldots, E_M{>}$ which is homogeneous of degree m. Let $L^{\#} \in$ **Diff** $(D_1, \ldots, D_N; R)$ denote the differential operator obtained by the substitution*

$$E_j = \sum_{\mu=1}^{N} a_j^{\mu} D_{\mu}.$$

Assume that p has a symmetry decomposition. Then there exists a differential operator L such that (i) L involves

$$c\,N^m$$

fewer occurrences of terms containing $D_1, \ldots, D_N$ than does $L^{\#}$ (ii) the higher order derivations $L^{\#}$ and L agree. Here c is a constant depending upon the symmetry decomposition.

We end this section with some remarks.

1. In §5 we give an algorithm to compute L. It is important to note that this algorithm does not require the explicit identification of the symmetry decomposition; rather, if such a symmetry decomposition is present the differential operator L produced by the algorithm will contain fewer terms than the differential operator $L^{\#}$.

2. The table below details the reduction in the number of terms for three different expressions. The last column lists the number of terms which either cancel or combine: these terms need not be computed. We hope to return in a later paper to the classification of those expressions in $E_1, \ldots, E_M$ which result in cancellation or combination when they are written in terms of the $D_1, \ldots, D_N$. In this paper we simply point out that Lie brackets are expessions of this type and that is easy to write down other such expressions as in the table.

expression	$\lvert L^{\#}\rvert$	$\lvert L\rvert$	$\lvert L^{\#}\rvert - \lvert L\rvert$
$E_2E_1 - E_1E_2$	$4N^2$	$2N^2$	$2N^2$
$E_3E_2E_1 + E_3E_1E_2$	$12N^3$	$6N^3$	$6N^3$
$E_3E_2E_1 - E_3E_1E_2 - E_2E_1E_3 + E_1E_2E_3$	$24N^3$	$6N^3$	$18N^3$

Here $|L|$ denotes the number of terms comprising the differential operator L.

3. We anticipate that the ideas in this paper may be applied to symbolic computations involving differential operators defined in terms of vector fields, such as in [2]; to symbolic calculations involving Lie brackets, such as occur in control theory [1]; and to symbolic calculation of Poisson brackets, such as occur in mechanics [3].

2 Orchards

In this section we describe a data structure that is useful for computations involving higher order derivations. We will be concerned with sets of rooted trees. Recall that a graph consists of a set of vertices and a set of edges. A graph is undirected in case each edge consists of an unordered pair of distinct vertices. We say two vertices are adjacent in case they are connected by an edge. A path consists of a sequence of distinct, adjacent vertices, and a cycle is a path whose initial and final vertices are the same and which does not intersect itself. A tree is an undirected graph which is connected and does not contain any cycles. A tree is rooted in case there is a distinguished vertex called a root. We will call the vertices of a tree nodes. See [4] for a careful discussion of these concepts as well as any unexplained notation concerning trees.

Definition. We define H_m to be the set of all rooted trees h consisting of $m+1$ nodes together with a map

$$\kappa : \text{nodes } h \longrightarrow \{0, 1, \ldots, m\}$$

satisfying

1. κ is bijective
2. v a child of w implies $\kappa(v) > \kappa(w)$, where $v, w \in$ nodes h.

That is, H_m consists of all heaps on $m+1$ nodes with keys κ. Put

$$H = \bigcup_{m \geq 0} H_m.$$

From now on we identify a node v with its heap name $j \in \{0, \ldots, m\}$ via the map κ.

Let $E_1, \ldots, E_M$ be M arbitrary symbols. Given a heap $h \in H_m$, let

$$h(E_{\gamma_1}, \ldots, E_{\gamma_m})$$

denote the labeled heap whose node with heap name j also has the label

$$E_{\gamma_j} \in \{E_1, \ldots, E_M\},$$

for $j = 1, \ldots, m$. Denote the set of all such labeled heaps by

$$LH_m = LH_m(E_1, \ldots, E_M)$$

and put

$$LH = LH(E_1, \ldots, E_M) = \bigcup_{m \geq 0} LH_m(E_1, \ldots, E_M).$$

We say that $h \in LH(E_1, \ldots, E_M)$ is homogeneous of degree m in case $h \in LH_m(E_1, \ldots, E_M)$.

We now describe a way of putting an algebraic structure on a set of trees T. Usually T will consist of an infinite set of trees.

Definition. Let K be a commutative ring with a unit element. We define $\mathcal{O}(T)$ to be the set consisting of all maps

$$\sigma : T \longrightarrow K$$

with the property that $\sigma(t) = 0$, for all but a finite number of the $t \in T$. This is an $K-$module and its elements are called orchards. Let $|\sigma|$ denote the number nonzero coefficients $\sigma(t)$; that is

$$|\sigma| = \#\{t \in T : \sigma(t) \neq 0\}.$$

In this paper we will usually let $T = LH(E_1, \ldots, E_M)$.

We end this section by defining a multiplication in $\mathcal{O}(LH(E_1, \ldots, E_M))$. Fix labeled heaps

$$\begin{aligned} h_1(E_{\gamma_1}, \ldots, E_{\gamma_{m_1}}) &\in LH_{m_1}(E_1, \ldots, E_M) \\ h_2(E_{\eta_1}, \ldots, E_{\eta_{m_2}}) &\in LH_{m_2}(E_1, \ldots, E_M). \end{aligned}$$

Definition. The product

$$h_2 \circ h_1 \in \mathcal{O}(LH(E_1, \ldots, E_M)).$$

of two orchards is defined as follows

Step 1. Recall that each node of heap h_2 has a name $0, 1, \ldots, m_2$ and each node of heap h_1 has a name $0, 1, \ldots, m_1$. We rename the names of heap 2 as follows:

old name	1	2	$\cdots$	m_2
new name	$m_1 + 1$	$m_1 + 2$	$\cdots$	$m_1 + m_2$
label	E_{η_1}	E_{η_2}	$\cdots$	$E_{\eta_{m_2}}$

We keep the names and nodes of h_1 the same.

Step 2. Delete the root of h_2. This produces several subtrees $t_1, \ldots, t_l$ with roots $c_1, \ldots, c_l$. We write this as

$$\text{deleteroot } h_2 = \{t_1, \ldots, t_l\}.$$

Step 3. Choose l nodes of h_1, allowing repetition; that is, choose $n_1, \ldots, n_l \in (\text{nodes } h_1)^l$.

Step 4. Form the new tree obtained by linking each root c_j to the node n_j, for $j = 1, \ldots, l$. This is not merely a tree but also a heap, denoted

$$\text{link}(t_1, \ldots, t_l; n_1, \ldots, n_l).$$

Step 5. Form an orchard by summing over all possible choices of nodes $n_1, \ldots, n_l$ in step 3. This defines the circle product of the labeled heaps h_1 and h_2:

$$h_2 \circ h_1 = \bigoplus_{n_1, \ldots, n_l \in (\text{nodes } h_1)^l} \text{link}(t_1, \ldots, t_l; n_1, \ldots, n_l),$$

where deleteroot $h_2 = \{t_1, \ldots, t_l\}$.

Step 6. Finally we complete the definition by extending the operation $\circ$ to all of $\mathcal{O}(LH(E_1, \ldots, E_M))$ by $K-$linearity.

Lemma 2.1 *(i) The operation $\circ$ is associative. (ii) The space of orchards $\mathcal{O}(LH(E_1, \ldots, E_M))$is a $K-$algebra with respect to the operations $\circ$ and addition.*

Proof. Recall that a heap is characterized by a table listing the parents of the nodes. For example

node	parent
0	$\emptyset$
1	0
2	0
3	1
4	3
5	2

is an element of H_5. Notice that one of the nodes does not have a parent: this is denoted with an $\emptyset$.

Let $h_j \in H_{m_j}$, for $j = 1, 2, 3$. The product

$$h_3 \circ (h_2 \circ h_1)$$

contains heaps of the form

node	parent
0	$\emptyset$
1	$*$
$\vdots$	$\vdots$
m_1	$*$
$---------$	$---------$
(root)	$\emptyset$
m_1+1	$*$
$\vdots$	$\vdots$
m_1+m_2	$*$
$---------$	$---------$
(root)	$\emptyset$
m_1+m_2+1	$*$
$\vdots$	$\vdots$
$m_1+m_2+m_3$	$*$

This table uses a number of conventions which we now describe. We use a dashed line to indicate which heaps the nodes belonged to before the product was formed. We say that all the nodes above the first dashed line belong to the first layer; that all the nodes between the two dashed lines belong to the second layer, etc. For example all nodes from the third layer were originally elements of h_3. If a node is enclosed in parentheses, that indicates that it was a node of one of the heaps comprising the product, but was deleted during the formation of the product. As already remarked, a $\emptyset$ indicates that a node has no parent. Finally a * is used to indicate the name of any node higher up in the table. For example the first table says that node 3 is the parent of node 4. With this notation we could replace the 3 with a *, since 3 occurs above 4 in the first column of the table.

We will show that

$$h_3 \circ (h_2 \circ h_1) = (h_3 \circ h_2) \circ h_1.$$

Given a term $h \in h_3 \circ (h_2 \circ h_1)$, we will show that $h \in (h_3 \circ h_2) \circ h_1$ also. Notice that the table consists of three layers, corresponding to the three heaps h_1, h_2, h_3, and that the operation $\circ$ simply consists of replacing any * which is a 0 with the names of all the nodes in the layer(s) above it in the table and forming the corresponding heaps. For example in computing $h_3 \circ (h_2 \circ h_1)$, the *'s in h_3 which are 0 are replaced with the names of nodes in the two layers above. If the node is in the segment immediately above corresponding to h_2, then it is clear that such a heap is formed when computing $(h_3 \circ h_2) \circ h_1$. On the other hand if the node is in the segment corresponding to h_1, then this heap does not correspond to any term in the sum $h_3 \circ h_2$. In this case the * can first be replaced by 0, corresponding to the root of h_2, and then replaced by the proper node from h_1 when computing $(h_3 \circ h_2) \circ h_1$. This verifies the assertion made at the beginning of the paragraph. The opposite inclusion is proved in the same way. We have shown assertion (i) of the lemma; the other assertion now follows at once. ∎

3 From $K < E_1, \ldots, E_M >$ to orchards

In this section we show how computational problems involving higher order derivations can be translated into problems involving orchards. We begin by defining a map

$$\phi : K < E_1, \ldots, E_M > \longrightarrow \mathcal{O}(LH(E_1, \ldots, E_M)).$$

Step 1. Fix a monomial $p \in K < E_1, \ldots, E_M >$ of the form

$$E_{\gamma_m} \cdots E_{\gamma_1}.$$

Define

$$\phi(E_{\gamma_m} \cdots E_{\gamma_1}) = \bigoplus_{h \in LH_m(E_1, \ldots, E_M)} h(E_{\gamma_1}, \ldots, E_{\gamma_m}).$$

Step 2. Extend the map ϕ to all of $K < E_1, \ldots, E_M >$ by $K-$linearity.

Lemma 3.1

$$\bigoplus_{h' \in H_m} h' = \bigoplus_{h \in H_{m-1}} \bigoplus_{v \in nodes\ h} attach(m, v)$$

where $attach(m, v)$ is the heap which arises when the node with name m is attached to node v of a heap in H_{m-1}.

Proof. Given a heap $h' \in H_m$, we obtain a heap $h \in H_{m-1}$ by removing the node labeled m. Since the heaps $h \in H_{m-1}$ are distinct so are the heaps we obtain by attaching the node labeled m to a node v of h. ∎

Theorem 3.1 *The map ϕ is a $K-$algebra homomorphism.*

Proof. Let $\gamma = (\gamma_1, \dots, \gamma_{m_1})$ and define

$$E_\gamma = E_{\gamma_{m_1}} \cdots E_{\gamma_1}.$$

Let $\delta = (\delta_1, \dots, \delta_{m_2})$. and define E_δ similarly.

It is clear that

$$\phi(E_\gamma + E_\delta) = \phi(E_\gamma) + \phi(E_\delta).$$

We need only prove that

$$\phi(E_\gamma \cdot E_\delta) = \phi(E_\gamma) \circ \phi(E_\delta).$$

We do this by induction on the length of the multi-index γ. If $m_1 = 1$, the assertion follows from lemma 1. Assume the assertion is true for $m = 1, \dots, m_1$. We compute

$$\begin{aligned} \phi(E_\gamma) \circ \phi(E_\delta) &= \phi(E_{\gamma_{m_1+1}} \cdots E_{\gamma_1}) \circ \phi(E_{\delta m_2} \cdots E_{\delta 1}) \\ &= \phi(E_{\gamma_{m_1+1}}) \circ \phi(E_{\gamma_{m_1}} \cdots E_{\gamma_1}) \circ \phi(E_{\delta m_2} \cdots E_{\delta 1}) \\ &= \phi(E_{\gamma_{m_1+1}}) \circ \phi(E_{\gamma_{m_1}} \cdots E_{\gamma_1} E_{\delta m_2} \cdots E_{\delta 1}) \\ &= \phi(E_{\gamma_{m_1+1}} \cdots E_{\gamma_1} E_{\delta m_2} \cdots E_{\delta 1}) \end{aligned}$$

showing that the assertion holds for $m = m_1 + 1$. ∎

4 From orchards to Diff $(D_1, \dots, D_N; R)$

In this section we define a map

$$\psi : \mathcal{O}(LH(E_1, \dots, E_M)) \longrightarrow \textbf{Diff}\ (D_1, \dots, D_N; R)$$

We do this in several steps.

Step 1. Let $h \in LH_m(E_1, \dots, E_M)$ and let $k \in$ nodes h, and suppose that $l, \dots, l'$ are the children of k. Fix $\mu_l, \dots, \mu_{l'}$ with

$$0 \leq \mu_l, \dots, \mu_{l'} \leq N$$

and define

$$\begin{aligned} R_h(k; \mu_l, \dots, \mu_{l'}) &= D_{\mu_l} \cdots D_{\mu_{l'}} a^{\mu_k}_{\gamma_k} \quad \text{if } k \text{ is not the root} \\ &= D_{\mu_l} \cdots D_{\mu_{l'}} \quad \text{if } k \text{ is the root .} \end{aligned}$$

We abbreviate this to $R_h(k)$ or $R(k)$.

Step 2. Define

$$\psi(h) = \sum_{\mu_1, \ldots, \mu_m} R(m) \cdots R(1)R(0).$$

Step 3. Extend ψ to all $\mathcal{O}(LH(E_1, \ldots, E_M))$by $K-$linearity.

Lemma 4.1 *Suppose* $p(E_1, \ldots, E_M) \in K\!<\!E_1, \ldots, E_M\!>$ *and* $L^\# \in \mathbf{Diff}(D_1, \ldots, D_N; R)$ *is the corresponding derivation defined by the substitution*

$$E_j = \sum_{\mu=1}^{N} a_j^\mu D_\mu.$$

Then

$$L^\#(a) = (\psi \circ \phi(p(E_1, \ldots, E_M)))(a).$$

Proof. We need only prove the lemma when p is a monomial. Let

$$p = E_{\gamma_m} \cdots E_{\gamma_1}$$

Assume $m = 1$. Then

$$\phi(E_{\gamma_1}) =$$

and

$$\psi(\qquad) = \sum_{\mu_1=1}^{N} a_{\gamma_1}^{\mu_1} D_{\mu_1}$$

We now prove the lemma by induction. Assume the lemma holds for $m' = 1, \ldots, m-1$. Then we claim that

$$(\sum_{\mu_m=1}^{N} a_{\gamma_m}^{\mu_m} D_{\mu_m})(\sum_{\mu_{m-1}=1}^{N} a_{\gamma_{m-1}}^{\mu_{m-1}} D_{\mu_{m-1}}) \cdots (\sum_{\mu_1=1}^{N} a_{\gamma_1}^{\mu_1} D_{\mu_1})(a),$$

which is $L^\#(a)$, is equal to

$$\begin{aligned}
&= (\sum_{\mu_m=1}^{N} a_{\gamma_m}^{\mu_m} D_{\mu_m})(\psi \bigoplus_{h \in LH_{m-1}(E_1, \ldots, E_M)} h(E_{\gamma_1}, \ldots, E_{\gamma_{m-1}}))(a) \\
&= \sum_{\mu_m} a_{\gamma_m}^{\mu_m} D_{\mu_m}(\sum_{h \in LH_{m-1}(E_1, \ldots, E_M)} \psi(h(E_{\gamma_1}, \ldots, E_{\gamma_{m-1}})))(a) \\
&= \sum_{\mu_m} \sum_{h \in LH_{m-1}(E_1, \ldots, E_M)} a_{\gamma_m}^{\mu_m} D_{\mu_m}(R_h(m-1) \cdots R_h(0))(a)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{\mu_m} \sum_{h \in LH_{m-1}(E_1,\ldots,E_M)} a^{\mu_m}_{\gamma_m} (D_{\mu_m} R_h(m-1)) \cdots R_h(0))(a) \\
&+ \sum_{\mu_m} \sum_{h \in LH_{m-1}(E_1,\ldots,E_M)} a^{\mu_m}_{\gamma_m} R_h(m-1) \cdots (D_{\mu_m} R_h(0))(a) \\
&= \sum_{h' \in LH_m(E_1,\ldots,E_M)} R_{h'}(m) \cdots R_{h'}(0)(a) \\
&= \psi(\phi(E_{\gamma_m} \cdots E_{\gamma_1}))(a)
\end{aligned}
$$

Here $h' \in LH_m(E_1,\ldots,E_M)$ is obtained from $h \in LH_{m-1}(E_1,\ldots,E_M)$ by attaching the node labeled m to each node v of h', as in lemma 3.1. ▮

The map ψ from orchards to **Diff** $(D_1,\ldots,D_N;R)$ is not injective. This is because the derivations D_i commute. We conclude this section by describing some elements of the kernel. Let $LT_m(E_1,\ldots,E_M)$ denote the set of all rooted trees g consisting of $m+1$ labeled nodes. We will always label the nodes with the symbols $E_1,\ldots,E_M$; sometimes we write

$$g(E_{\gamma_1},\ldots,E_{\gamma_m})$$

to indicate which symbols label the nodes. Note that the notation does not indicate which node carries which label. Let $\mathcal{O}(LT_m(E_1,\ldots,E_M))$ denote the orchard defined on the set $LT_m(E_1,\ldots,E_M)$. Put

$$LT(E_1,\ldots,E_M) = \bigcup_{m\geq 0} LT_m(E_1,\ldots,E_M)$$

and let $\mathcal{O}(LT(E_1,\ldots,E_M))$ denote the corresponding orchard.

There is a natural map

$$\chi : \mathcal{O}(LH(E_1,\ldots,E_M)) \longrightarrow \mathcal{O}(LT(E_1,\ldots,E_M)),$$

which is defined as follows.

Step 1. If $h \in LH_m(E_1,\ldots,E_M)$, then define $\chi(h)$ to be the element of $LT_m(E_1,\ldots,E_M)$ obtained by ignoring the heap names of the nodes of h. That is, a labeled heap is a labeled tree together with heap names attached to the nodes and the map χ simply drops the heap names.

Step 2. Extend χ to all of $\mathcal{O}(LH(E_1,\ldots,E_M))$ by $K-$linearity.

Lemma 4.2 *Let $h, h' \in LH_m(E_1, \ldots, E_M)$ be two labeled heaps with the property $\chi(h) = \chi(h')$. Then $\psi(h) = \psi(h')$.*

Proof. We use the notation introduced at the beginning of the section and write

$$\begin{aligned} \psi(h(E_{\gamma_1}, \ldots, E_{\gamma_m})) &= \sum R(m; \mu_1, \ldots, \mu_m) \ldots R(0; \mu_1, \ldots, \mu_m) \\ \psi(h'(E_{\eta_1}, \ldots, E_{\eta_m})) &= \sum S(m; \nu_1, \ldots, \nu_m) \ldots R(0; \nu_1, \ldots, \nu_m). \end{aligned}$$

Recall that if a node k has children $l, \ldots, l'$, then

$$R(k; \mu_1, \ldots, \mu_m) = D_{\mu_l} \cdots D_{\mu_{l'}} a_{\gamma_k}^{\mu_k};$$

$S(k; \nu_1, \ldots, \nu_m)$ is defined similarly

$$S(k; \nu_1, \ldots, \nu_m) = D_{\nu_l} \cdots D_{\nu_{l'}} a_{\eta_k}^{\nu_k}.$$

Observe that the heap names of the nodes simply provide dummy indices – $\mu_1, \ldots, \mu_m$ for h and $\nu_1, \ldots, \nu_m$ for h' – which are used to write out the differential operators in **Diff** $(D_1, \ldots, D_N; R)$ corresponding to the heaps h and h'. Therefore if the underlying labeled trees are the same, then the differential operators in **Diff** $(D_1, \ldots, D_N; R)$ will be equal; that is $\chi(h) = \chi(h')$ implies $\psi(h) = \psi(h')$ as asserted. ∎

5 Symmetries of Orchards

Let $\sigma \in \mathcal{O}(LH_m(E_1, \ldots, E_M))$ be an orchard on labeled heaps which are homogeneous of degree m. We say that σ has a symmetry decomposition in case

1. $\sigma = \tau + \rho$
2. $\chi(\tau) = 0$.

In this section we show how symmetry decompositions can be used to reduce the operation counts of computations involving higher order derivations.

Consider the number of derivations D_μ which arise in the evaluation of a higher order derivation

$$p(E_1, \ldots, E_M), \qquad \text{where } p \in K\!<\!E_1, \ldots, E_M\!>.$$

Recall that substituting

$$E_j = \sum_{\mu=1}^{N} a_j^\mu D_\mu$$

into $p(E_1, \ldots, E_M)$ produces a higher order derivation $L^\# \in \mathbf{Diff}(D_1, \ldots, D_N; R)$. Because the derivations D_μ commute, the higher order derivation $L^\#$ will in general involve some cancellation, so that it makes sense to ask whether there is another higher order derivation L which agrees with $L^\#$ and involves fewer terms. We will show that this is the case under the assumption that there is a symmetry decomposition of the corresponding orchard. Recall that if the orchard τ can be written

$$\tau = \sum_h \tau(h)h, \quad \text{where } \tau(h) \in K, \quad \text{and } h \in LH_m(E_1, \ldots, E_M),$$

then $|\tau|$ denotes the number of heaps h such that the corresponding coefficient of the orchard $\tau(h)$ is nonzero.

Theorem 5.1 *Fix an expression $p \in K{<}E_1, \ldots, E_M{>}$ which is homogeneous of degree m and let $\sigma = \phi(p)$ be the corresponding orchard. Let $L^\# \in \mathbf{Diff}\ (D_1, \ldots, D_N; R)$ be the differential operator obtained by the substitution*

$$E_j = \sum_{\mu=1}^{N} a_j^\mu D_\mu.$$

Assume that

$$\sigma = \tau + \rho$$

is a symmetry decomposition of σ and let $L = \psi(\rho)$. Then (i) L involves

$$|\tau|\, N^m$$

fewer occurrences of terms containing $D_1, \ldots, D_N$ than does $L^\#$ and (ii) the higher order derivations $L^\#$ and L agree.

Proof. All the work has already been done. Since

$$\sigma = \tau + \rho$$

is a symmetry decomposition, we have $\chi(\tau) = 0$. Hence by lemma 4.2, $\psi(\tau) = 0$, and there will be $|\tau|\, N^m$ fewer occurrences of terms containing $D_1, \ldots, D_N$. By lemma 4.1 $L^\#$ and L agree. This proves the theorem. ∎

Theorem 1.1 is an immediate corollary of this theorm.

6 Example

In this section we present a simple example of a computation of a second order derivation in terms of orchards. Fix two derivations

$$
\begin{aligned}
E_1 &= \sum_{\mu=1}^{N} a_1^{\mu} D_{\mu} \\
E_2 &= \sum_{\mu=1}^{N} a_2^{\mu} D_{\mu},
\end{aligned}
$$

where $a_j^{\mu} \in R$, for $j = 1, 2$.

Consider the higher order derivation $L = p(E_1, E_2)$, where the noncommuting polynomial p is of the form

$$p = E_1 E_2 - E_2 E_1.$$

Then $L = E_1 E_2 - E_2 E_1$. This may be written in terms of the commuting derivations D_{μ} as

$$
\begin{aligned}
L^{\#} &= \sum_{\mu_1,\mu_2} a_1^{\mu_2} a_2^{\mu_1} D_{\mu_2} D_{\mu_1} + \sum_{\mu_1,\mu_2} a_1^{\mu_2} (D_{\mu_2} a_2^{\mu_1}) D_{\mu_1} \\
&- \sum_{\mu_1,\mu_2} a_2^{\mu_2} a_1^{\mu_1} D_{\mu_2} D_{\mu_1} - \sum_{\mu_1,\mu_2} a_2^{\mu_2} (D_{\mu_2} a_1^{\mu_1}) D_{\mu_1}
\end{aligned}
$$

The orchard $\sigma = \phi(p)$ corresponding to p is

The orchard σ has a symmetry decomposition $\sigma = \tau + \rho$, where

Since $\chi(\tau) = 0$, the higher order derivation $L = \psi(\rho)$ agrees with L. The derivation L has the form

$$L = \sum_{\mu_1,\mu_2} a_1^{\mu_2}(D_{\mu_2} a_2^{\mu_1}) D_{\mu_1} - \sum_{\mu_1,\mu_2} a_2^{\mu_2}(D_{\mu_2} a_1^{\mu_1}) D_{\mu_1}$$

and contains $2N^2$ fewer products of the form $D_{\mu_2} D_{\mu_1}$ than does L.

References

[1] H. HERMES, *Nilpotent Approximations of Control Systems and Distributions,* SIAM J. Control Optim., 24 (1986), pp. 731-736.

[2] L. HÖRMANDER, *Hypoelliptic Second Order Differential Equations,* Acta Mathematica, 119 (1968), pp. 147 - 171.

[3] J. MARSDEN, ed., *Fluids and Plasmas: Geometry and Dynmaics,* American Math. Soc., Providence, 1984.

[4] R. E. TARJAN, *Data Structures and Network Algorithms,* SIAM, Philadelphia, 1983.