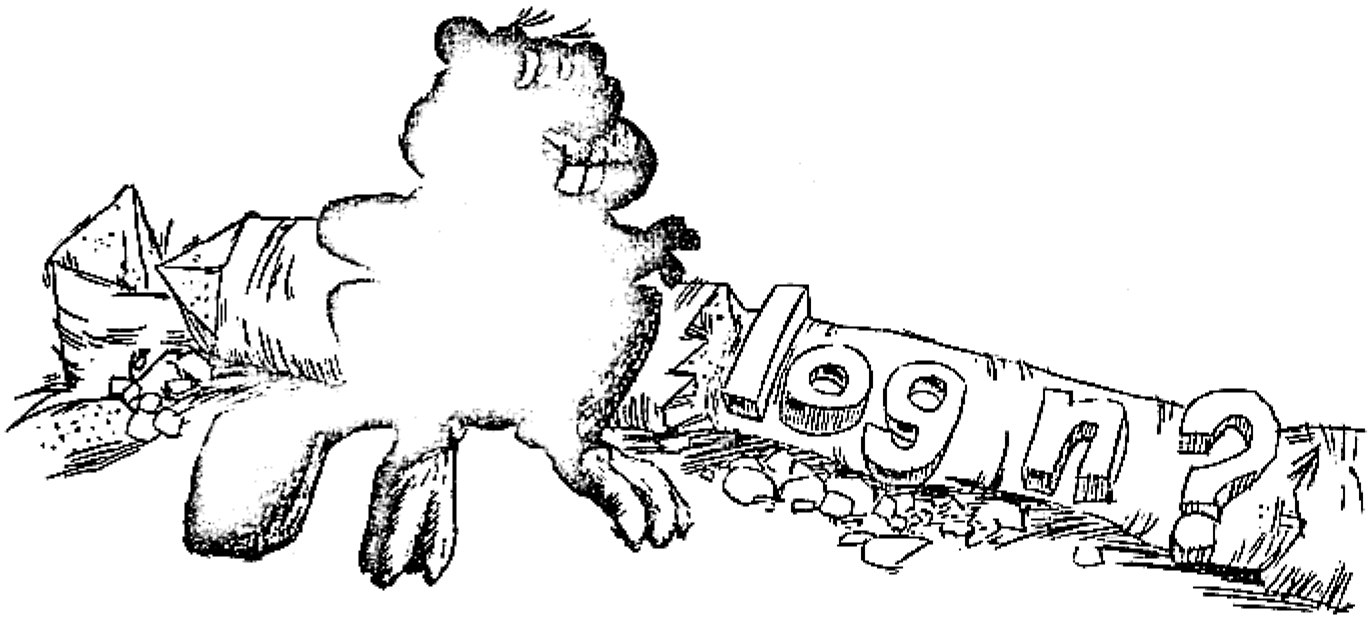
Proceedings of a Conference on Theoretical Computer Science

August 15-17, 1977
University of Waterloo
Waterloo, Ontario
Canada



Additional copies available, for $10, from:

Theory Conference, Computer Science Department,
University of Waterloo, Waterloo, Ontario N2L 3G1, Canada.

FAST ALGORITHMS FOR COMPOSITION AND REVERSION OF MULTIVARIATE POWER SERIES

(Preliminary Version)

R. P. Brent
Computer Centre
Australian National University
Canberra, Australia

H. T. Kung
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pennsylvania USA

1. INTRODUCTION

In our earlier paper [2], we gave fast algorithms for manipulating univariate power series. In this paper, fast algorithms for composition and reversion of multivariate power series are presented. The new algorithms requires substantially less operations than the best previously known counterparts. The amount of reduction in the number of required operations increases rapidly as the number of variables in multivariate power series increases. (See Table 5.1.)

In Section 2 we first define formally the general reversion problem for multivariate power series. We then show that every reversion problem is associated with a composition problem, in the sense that if the composition problem can be solved fast so can the reversion problem.

Section 3 deals with the case when the power series we want to compute are univariate, though they may result from compositions or reversions involving multivariate power series. The bivariate case is treated in Section 4. In Section 5, we state our results for more than two variables and give a table summarizing the new results.

This work has been supported in part by the National Science Foundation under Grant MCS 75-222-55 and the Office of Naval Research under Contract N00014-76-C-0370, NR 044-422.

2. THE COMPOSITION AND REVERSION PROBLEMS

Notation

We deal with power series over an algebraically closed field K, and count operations defined by the field. Let $Q(s_1,\ldots,s_k) = \sum_{i_j \geq 0} q_{i_1,\ldots,i_h} \cdot s_1^{i_1} \cdots s_h^{i_h}$ be a power series. We define the degree of the term $q_{i_1,\ldots,i_h} \cdot s_1^{i_1} \cdots s_h^{i_h}$ to be $i_1+\ldots+i_h$. Define $\deg Q$ (or $\operatorname{ord} Q$) to be the degree of the maximal (or minimal, respectively) degree terms in $Q(s_1,\ldots,s_k)$. $Q(s_1,\ldots,s_k) \bmod s^{n+1}$ is defined to be the polynomial consisting of all terms in $Q(s_1,\ldots,s_k)$ which have degrees $\leq n$. By computing a polynomial, we mean computing all the coefficients in the polynomial.

Let $A(s)$ and $B(s)$ be two given univariate power series and let $C = A \cdot B$. We denote by $M(n)$ the number of operations required to compute $C(s) \bmod s^{n+1}$. The classical multiplication algorithm gives $M(n) = O(n^2)$ and the FFT gives $M(n) = O(n \log n)$. For the convenience in describing our results, we shall assume that $M(n)$ satisfies the following properties:

$$M(n) + M(\lceil \tfrac{n}{2} \rceil) + M(\lceil \tfrac{n}{4} \rceil) + \ldots = O(M(n)),$$

$$M(n) + 2M(\lceil \tfrac{n}{2} \rceil) + 4M(\lceil \tfrac{n}{4} \rceil) + \ldots = O((\log n)M(n)),$$

$$M(n) + 2M(\lceil \tfrac{n}{2^2} \rceil) + 4M(\lceil \tfrac{n}{4^2} \rceil) + \ldots = O(M(n)),$$

$$nM(n) = O(M(n^2)).$$

Note that these properties are satisfied if $M(n) = cn^2$ or $cn \log n$ where c is any nonzero constant.

<u>The Reversion Problem</u>

The reversion problem for multivariate power series arises frequently in various branches of applied mathematics. (See e.g., [3,4,5] for some recent results on the problem.) The problem can be defined formally as follows: Suppose that we are given multivariate power series $Q_i(s_1,\ldots,s_h)$ and $C_i(s_1,\ldots,s_k)$ for $i = 1,\ldots,m$ where $h \geq m$ and $k \geq h - m$. Let $j = h - m$. The reversion problem defined by

$$\begin{aligned} &Q_1(s_1,\ldots,s_j, V_1(s_1,\ldots,s_k),\ldots, V_m(s_1,\ldots,s_k)) = C_1(s_1,\ldots,s_k),\\ (2.1)\quad &\vdots\\ &Q_m(s_1,\ldots,s_j, V_1(s_1,\ldots,s_k),\ldots, V_m(s_1,\ldots,s_k)) = C_m(s_1,\ldots,s_k), \end{aligned}$$

is to compute $V_i(s_1,\ldots,s_k) \bmod s^{n+1}$ for $i = 1,\ldots,m$. For briefness we shall often use vector notation, so (2.1) can be written as

$$(2.2)\quad \underset{\sim}{Q}(\underset{\sim}{s}';\underset{\sim}{V}(\underset{\sim}{s})) = \underset{\sim}{C}(\underset{\sim}{s})$$

where the $\underset{\sim}{Q}, \underset{\sim}{s}', \underset{\sim}{V}\ldots$ are vectors whose definitions should be evident from the context. To ensure the existence of $\underset{\sim}{V}(\underset{\sim}{s})$, it is necessary to impose some conditions on the system (2.1) or (2.2). The following theorem is an immediate generalization of Corollary 5.1 of Kung and Traub [7] or Theorem 3.1 of Lipson [8]. Its proof is omitted.

<u>Theorem 2.1.</u>

If

$$(2.3)\quad \underset{\sim}{Q}(\underset{\sim}{0};\underset{\sim}{0}) = \underset{\sim}{C}(\underset{\sim}{0}), \text{ and}$$

$$(2.4)\quad D_{\underset{\sim}{V}}\underset{\sim}{Q}(\underset{\sim}{0};\underset{\sim}{0}) \text{ is nonsingular,}$$

then there exists a unique $\underset{\sim}{V}(\underset{\sim}{s})$, $\underset{\sim}{V}(\underset{\sim}{0}) = \underset{\sim}{0}$, satisfying (2.2), and the iterates $\underset{\sim}{V}^{(i)}$ generated by the Newton-like iteration

$$(2.5)\quad \underset{\sim}{V}^{(i+1)}(\underset{\sim}{s}) = \underset{\sim}{V}^{(i)}(\underset{\sim}{s}) - D_{\underset{\sim}{V}}\underset{\sim}{Q}(\underset{\sim}{s}';\underset{\sim}{V}^{(i)}(\underset{\sim}{s}))^{-1}\underset{\sim}{Q}(\underset{\sim}{s}';\underset{\sim}{V}^{(i)}(\underset{\sim}{s})) \bmod s^{2^{i+1}}$$

are well defined and satisfy

$$\operatorname{ord}(\underset{\sim}{V}^{(i)}(\underset{\sim}{s}) - \underset{\sim}{V}(\underset{\sim}{s})) \geq 2^i.$$

Note that $D_{\underset{\sim}{V}}\underset{\sim}{Q}$ is defined by:

$$D_{\underset{\sim}{V}}\underset{\sim}{Q}(\underset{\sim}{s}';\underset{\sim}{V}) = \begin{bmatrix} \partial_{V_1}Q_1(\underset{\sim}{s}';\underset{\sim}{V}) \ldots \partial_{V_m}Q_1(\underset{\sim}{s}';\underset{\sim}{V}) \\ \vdots \qquad\qquad \vdots \\ \partial_{V_1}Q_m(\underset{\sim}{s}';\underset{\sim}{V}) \ldots \partial_{V_m}Q_m(\underset{\sim}{s}';\underset{\sim}{V}) \end{bmatrix}$$

where $\partial_{V_i}Q_j(\underset{\sim}{s}';\underset{\sim}{V})$ denotes the partial derivative of $Q_j(\underset{\sim}{s}';\underset{\sim}{V})$ with respect to V_i.

<u>The Composition Problem</u>

Observe that to compute $\underset{\sim}{V}^{(i+1)}(s)$ by (2.5) we need to compute $\underset{\sim}{Q}(\underset{\sim}{s}';\underset{\sim}{V}^{(i)}(\underset{\sim}{s}))$ and $D_{\underset{\sim}{V}}\underset{\sim}{Q}(\underset{\sim}{s}';\underset{\sim}{V}^{(i)}(\underset{\sim}{s}))$ $\bmod s^{2^{i+1}}$. This motivates the following definition for the composition problem: Suppose that we are given multivariate power series $Q(s_1,\dots,s_h)$ and $P_i(s_1,\dots,s_k)$, $i = 1,\dots,m$, where $h \geq m$, $k \geq h - m$ and $P_i(0,\dots,0) = 0$ for all i. Let $j = h - m$. The composition problem defined by

$$R(s_1,\dots,s_k) = Q(s_1,\dots,s_j, P_1(s_1,\dots,s_k),\dots,P_m(s_1,\dots,s_k))$$

is to compute $R(s_1,\dots,s_k) \bmod s^{n+1}$.

Note that the sizes of both the composition and reversion problems are determined by (h,k,m) where $h - k \leq m \leq h$. We say a composition problem is associated with a reversion problem or vice-versa if they are defined by the same (h,k,m). With respect to any fixed (h,k,m), let REV(n) and COMP(n) be the number of operations required to solve a reversion problem and the associated composition problem, respectively. Then similar to Theorem 3.1 of Brent and Kung [2], one can obtain by (2.5) that

$$(2.6)\quad \mathrm{REV}(n) = \mathrm{REV}(\lceil \tfrac{n}{2} \rceil) + O(\mathrm{COMP}(n)).$$

(Here we used the fact that the $D_{\underset{\sim}{V}}\underset{\sim}{Q}(\underset{\sim}{s}';\underset{\sim}{V}^{(i)}(\underset{\sim}{s}))^{-1} \bmod s^{2^{i+1}}$ needed in (2.5) can be computed by the Newton-like iteration given $D_{\underset{\sim}{V}}\underset{\sim}{Q}(\underset{\sim}{s}';\underset{\sim}{V}^{(i)}(s)) \bmod s^{2^{i+1}}$, as in Kung [6].) (2.6) implies that a reversion problem can be solved fast if the associated composition problem can. Therefore, in the rest of the paper, we shall focus mainly on the derivation of fast composition algorithm. It turns out, however, that fast reversion algorithms may also help in the derivation of fast composition algorithms. This is rather surprising and will become evident in sections such as §3.3 and §4.3.

3. UNIVARIATE COMPOSITIONS AND REVERSIONS

In this section, we consider the case when the power series we want to compute are univariate (i.e., the case when $k = 1$ in the notation of Section 2.) In the rest of the paper, in the title of each subsection, the left part defines a composition problem and the right part defines the associated reversion problems.

3.1. $R(s) = Q(P(s))$, $Q(V(s)) = C(s)$

It has been shown in Brent and Kung [2] that

$$\mathrm{COMP}(n) = O((n \log n)^{\frac{1}{2}} M(n)),$$
$$\mathrm{REV}(n) = O((n \log n)^{\frac{1}{2}} M(n)).$$

(In fact, it is shown there that $\mathrm{COMP}(n) = O(\mathrm{REV}(n))$ and $\mathrm{REV}(n) = O(\mathrm{COMP}(n))$.)

3.2. $R(s) = Q(s,P(s))$, $Q(s,V(s)) = C(s)$

Using $R(s) = \Sigma\, Q_i(s)\cdot P(s)^i$, we have

$$\mathrm{COMP}(n) = O(nM(n)).$$

By (2.6) we have

$$\mathrm{REV}(n) = O(nM(n)).$$

3.3. $R(s) = Q(P_1(s),P_2(s))$, $\begin{cases} Q_1(V_1(s),V_2(s)) = C_1(s) \\ Q_2(V_1(s),V_2(s)) = C_2(s) \end{cases}$

Suppose that ord $P_1 = r$. Let $P_1(s) = p_r s^r + p_{r+1} s^{r+1} + \ldots, p_r \neq 0$. We make the change of variables $P_1(s) = p_r t^r$. Then

$$t = \overline{P}_1(s) \overset{\text{def}}{=} \left(\frac{P_1(s)}{p_r}\right)^{\frac{1}{r}} = s\left(1 + \frac{p_{r+1}}{rp_r} s + \ldots\right).$$

Hence there exists $\overline{V}_1(t)$ such that $\overline{V}_1(t) = s$. We have

$$R(\overline{V}_1(t)) = \overline{Q}(t) \overset{\text{def}}{=} Q(p_r t^r, P_2(\overline{V}_1(t)))$$

and

$$R(s) = R(\overline{V}_1(\overline{P}_1(s))) = \overline{Q}(\overline{P}_1(s)).$$

Therefore, $R(s) \bmod s^{n+1}$ can be computed by the following algorithm:

<u>Algorithm 3.1.</u>

1) Compute $\overline{P}_1(s) \bmod s^{n+1}$.
2) Compute $\overline{V}_1(t) \bmod t^{n+1}$.
3) Compute $P_2(\overline{V}_1(t)) \bmod t^{n+1}$.
4) Compute $\overline{Q}(t) \bmod t^{n+1}$.
5) Compute $\overline{Q}(\overline{P}_1(s)) \bmod s^{n+1}$.

Using a technique of Brent [1], step 1 can be done in $O(M(n))$ operations. By results of §3.1, steps 2, 3, and 5 can be done in $O((n \log n)^{1/2} M(n))$ operations. Using the technique of §3.2, step 4 can be done in $O(nM(n))$ operations. Thus, Algorithm 3.1 takes $O(nM(n))$ operations. We have

$$\text{COMP}(n) = O(nM(n)),$$

and by (2.6),

$$\text{REV}(n) = O(nM(n)).$$

<u>3.4. $R(s) = Q(P_1(S), \ldots, P_h(s))$, $\mathcal{Q}(\mathcal{V}(s)) = \mathcal{C}(s)$ an $h \times h$ System</u>

For $h = 3$, we write

$$R(s) = \Sigma\, Q_i(P_1(s), P_2(s)) \cdot P_3(s)^i.$$

By results of §3.3, for each i $Q_i(P_1(s), P_2(s)) \bmod s^{n+1}$ can be computed in $O(nM(n))$ operations. Hence the total work for $h = 3$ is $O(n^2 M(n))$ operations. For general h, using

$$R(s) = \Sigma\, Q_i(P_1(s), \ldots, P_{k-1}(s)) \cdot P_k(s)^i,$$

we obtain by induction that for $h \geq 3$,

$$\text{COMP}(n) = O(n^{h-1} M(n)),$$

and by (2.6),

$$\text{REV}(n) = O(n^{h-1} M(n)).$$

4. BIVARIATE COMPOSITIONS AND REVERSIONS

In this section, we consider the case when the power series we want to compute are bivraiate (i.e., the case when $k = 2$ in the notation of Section 2). To avoid using subscripts in this section, we shall write $s, t, \bmod (s+t)^{n+1}$ for $s_1, s_2, \bmod s^{n+1}$, respectively. The following two lemmas give some basic results concerning the multiplication and division of bivariate power series.

<u>Lemma 4.1.</u>

Let $A(s,t)$ and $B(s,t)$ be two given bivariate polynomials. If $\deg A \leq a$, $\deg B \leq b$ and the degree of B with respect to A is $\leq a$, then $A(s,t) \cdot B(s,t) \bmod (s+t)^{n+1}$ can be computed in $M_2(a,b;n)$ operations, where

$$M_2(a,b;n) = M(n+e\cdot\min(f,n)),$$
$$e = \min(a,n)+\min(b,n),$$
$$f = a+\min(a,b).$$

Proof

It suffices to consider $A_n(s,t) = A(s,t) \bmod (s+t)^{n+1}$ and $B_n(s,t) = B(s,t) \bmod (s+t)^{n+1}$. Let $C = A_n\cdot B_n$. Then $\deg C \le e$, and the degree of C with respect to t is $\le f'$, where $f' = \min(a,n) + \min(a,b,n)$. It is easy to see that the coefficients in $C(s,t)$ are one-to-one correspondent to those in the univariate polynomial $\overline{C}(s) \overset{\text{def}}{=} C(s,s^{e+1})$. Let cs^it^j be an arbitrary term in $C(s,t) \bmod (s+t)^{n+1}$. Then the degree of the correspondent term in $\overline{C}(s)$ satisfies:

$$\text{def}(cs^{i+(e+1)j}) = i+(e+1)j$$
$$\le n+e\cdot\min(f',n).$$

Hence it is sufficient to compute $\overline{C}(s) \bmod s^{n+e\cdot\min(f',n)+1}$, using $\overline{C}(s) = \overline{A}_n(s)\cdot\overline{B}_n(s)$ where $\overline{A}_n(s) = A_n(s,s^{e+1})$ and $\overline{B}_n(s) = B_n(s,s^{e+1})$. The lemma follows by noting that $\min(f',n) = \min(f,n)$. ■

Lemma 4.2.

Let $A(s,t)$ and $B(s,t)$ be two given bivariate power series. If ord $B = 0$, then $C = A/B$ exists and $C(s,t) \bmod (s+t)^{n+1}$ can be computed in $O(M(n^2))$ operations.

Proof

Since ord $B = 0$, the reciprocal D of B exists. Using the Newton-like iteration (Kung [6]), one can compute $D(s,t) \bmod (s+t)^{n+1}$ in $O(M_2(n,n;n) + M_2(n/2,n/2;n/2)+\dots)$ operations, i.e., in $O(M(n^2))$ operations by Lemma 4.1. $C(s,t) \bmod (s+t)^{n+1}$ can be obtained by multiplying $A(s,t) \bmod (s+t)^{n+1}$ and $D(s+t) \bmod (s+t)^{n+1}$ in $O(M(n^2))$ operations. ■

4.1. $R(s,t) = Q(P(s,t))$, $Q(V(s,t)) = C(s,t)$

Suppose that

(4.1) $P(s,t) = t^r\hat{P}(s,t)$ where $r \ge 1$ and ord $\hat{P} = 0$. We can assume that $r \le n$ or the composition problem would be trivial. In the following we shall generalize algorithm 2.2 of Brent and Kung [2] to compute the bivariate polynomial $R(s,t) \bmod (s+t)^{n+1}$. Write $P = P_m + P_r$ where P_m is a polynomial of degree m with $m \le n$ and P_r is a power series with ord $P_r \le m+1$. (The value of m will be determined later.) By the Taylor series expansion, we have

$$\begin{aligned} Q(P) &= Q(P_m+P_r) \\ &= Q(P_m) + Q'(P_m)\cdot P_r + \tfrac{1}{2}Q''(P_m)\cdot P_r^2 + \cdots . \end{aligned}$$

Let $\ell = \lceil \frac{n}{m} \rceil$. Since ord $(P_r)^{\ell+i} \ge n+1$ for any $i > 0$,

$$Q(P(s,t)) \bmod (s+t)^{n+1}$$
$$= [Q(P_m(s,t)) + \cdots + \tfrac{1}{\ell!}Q^{(\ell)}(P_m(s,t)) \cdot P_r(s,t)^{\ell}] \bmod (s+t)^{n+1}.$$

Therefore, the following algorithm computes $R(s,t) \bmod (s+t)^{n+1}$:

Algorithm 4.1.

```
1.  Compute $(Q(P_m(s,t)) \bmod (s+t)^{n+\ell r+1}$.
```

2. Compute $Q^{(i)}(P_m(s,t))$ mod $(s+t)^{n+(\ell-i)r+1}$, for $i=1, \cdots, \ell$.

3. Compute $P_r(s,t)^i$ mod $(s+t)^{n+1}$, for $i = 1, \cdots, \ell$.

4. Compute $\frac{1}{i!}Q^{(i)}(P_m(s,t)) \cdot P_r(s,t)^i$ mod $(s+t)^{n+1}$, for $i = 1, \cdots, \ell$.

5. Sum the results obtained from step 4.

It is easy to check that steps 3, 4 and 5 can all be done in

$$T_1 = O(\ell(M(n^2)) = O((^n/m)M(n^2))$$

operations. If $m < r$, then $P_m(s,t) \equiv 0$ and steps 1 and 2 become trivial. In the following we assume that $r \leq m$. Hence for $0 \leq i \leq \ell$,

(4.2) $\quad n+(\ell-i)r+1 \leq n+\ell m+1 \leq 3n+1.$

Lemma 4.3.

$Q(P_m(s,t))$ mod $(s+t)^{n+1}$ can be computed in $O(mM(n^2))$ operations.

Proof

Without loss of generality, we assume that Q is a polynomial of degree n and that n is a power of two. We may write

$$Q(P_m(s,t)) = Q_1(P_m(s,t)) + P_m(s,t)^{n/2} \cdot Q_2(P_m(s,t))$$

where Q_1 and Q_2 are polynomials of degree $n/2$. This relation gives a recursive procedure for computing $Q(P_m(s,t))$. Let $T(j)$ be the number of operations needed to compute both $P_m(s,t)^{j/2}$ mod $(s+t)^{n+1}$ and $Q(P_m(s,t))$ mod $(s+t)^{n+1}$ with $\deg Q = j$. Then by the recursive procedure and by Lemma 4.1, we have

$$\begin{aligned} T(j) &\leq 2T(j/2)+O(M_2(jm/2,jm/2;n)) \\ &\leq 2T(j/2)+O(M(n+\min(jm,2n)\cdot \min(jm,n))) \end{aligned}$$

Let q by the largest integer i such that $mn/2^i \geq n$. We have

$$\begin{aligned} T(n) &= O(M(2n^2+n)+\cdots+2^q \cdot M(2n^2+n))+2^{q+1}T(n/2^{q+1}) \\ &= O(mM(n^2))+2mT(n/2^{q+1}). \end{aligned}$$

Using our assumptions on $M(n)$ and the fact that $mn/2^{q+1} < n$, we have

$$\begin{aligned} T(n/2^{q+1}) &= O(M(n+(mn/2^{q+1})^2)+ 2M(n+(mn/2^{q+2})^2)+\cdots) \\ &= O(M(n+n^2)+2M(n+n^2/2^2)+ 4M(n+n^2/4^2)+\cdots) \\ &= O(M(n^2)+nM(n)) \\ &= O(M(n^2)). \end{aligned}$$

Hence $T(n) = O(mM(n^2))$. ■

By Lemma 4.3 and (4.2), step 1 of Algorithm 4.1 can be done in

$$T_2 = O(mM(n^2))$$

operations. Let $H(s,t) = Q(P_m(s,t))$. We have

$$\partial_t H(s,t) = Q'(P_m(s,t)) \cdot \partial_t P_m(s,t).$$

By (4.1), $\partial_t P_m(s,t) = t^{r-1}\hat{P}_m(s,t)$ where $\text{ord}\ \hat{P}_m(s,t) = 0$. Hence $\partial_t H(s,t)$ is divisible by t^{r-1}. Let $\partial_t H(s,t) = t^{r-1}\hat{H}(s,t)$. Then

$$\hat{H}(s,t) = Q'(P_m(s,t)) \cdot \hat{P}_m(s,t).$$

Hence by Lemma 4.2, $Q'(P_m(s,t))$ mod $(s+t)^{n+(\ell-1)r+1}$ can be computed in $O(M(n^2))$ operations from $\hat{H}(s,t)$ mod $(s+t)^{n+(\ell-1)r+1}$ and $\hat{P}_m(s,t)$ mod $(s+t)^{n+(\ell-1)r+1}$. The latter two polynomials can clearly be computed in $O(n^2)$ operations from the result of step 1 and from $P_m(s,t)$. Similarly, $Q''(P_m(s,t))$ mod $(s+t)^{n+(\ell-2)r+1}$ can be computed in $O(M(n^2))$ operations. Thus, step 2 takes $O(T_1)$ operations.

Taking $m \sim \sqrt{n}$, we conclude that if (4.1) holds then Algorithm 4.1 takes $O(\sqrt{n}M(n^2)$ operations.

Change of Variables

We now consider the case when (4.1) is not satisfied. Let $\operatorname{ord} P(s,t) = r$, $r \geq 1$. It is easy to see that there exists a $c \in K$ such that the minimal degree terms in $\bar{P}(\bar{s},t) \overset{\text{def}}{=} P(\bar{s}+ct,t)$ include $c_1 t^r$ for some non-zero $c_1 \in K$. Making another change of variables $\bar{s} = \bar{\bar{s}}t$, we define $\bar{\bar{P}}(\bar{\bar{s}},t) = \bar{P}(\bar{\bar{s}}t,t)$. Then $c_1 t^r$ is the unique minimal degree term in $\bar{\bar{P}}(\bar{\bar{s}},t)$ and all other terms in $\bar{\bar{P}}(\bar{\bar{s}},t)$ are divisible by t^r. Hence $\bar{\bar{P}}(\bar{\bar{s}},t)$ satisfies (4.1).

Define $\bar{R}(\bar{s},t) = R(\bar{s}+ct,t)$ and $\bar{\bar{R}}(\bar{\bar{s}},t) = \bar{R}(\bar{\bar{s}}t,t)$. Then $\bar{\bar{R}}(\bar{\bar{s}},t) = Q(\bar{\bar{P}}(\bar{\bar{s}},t))$. We have the following algorithm for computing $R(s,t) \bmod (s+t)^{n+1}$.

Algorithm 4.2.

1. Compute $\bar{P}(\bar{s},t) \bmod (\bar{s}+t)^{2n+1}$.
2. Compute $\bar{\bar{P}}(\bar{\bar{s}},t) \bmod (\bar{\bar{s}}+t)^{2n+1}$.
3. Compute $\bar{\bar{R}}(\bar{\bar{s}},t) \bmod (\bar{\bar{s}}+t)^{2n+1}$, using $\bar{\bar{R}}(\bar{\bar{s}},t) = Q(\bar{\bar{P}}(\bar{\bar{s}},t))$.
4. Compute $\bar{R}(\bar{s},t) \bmod (\bar{s}+t)^{n+1}$, using $\bar{R}(\bar{s},t) = \bar{\bar{R}}(\bar{s}/t,t)$.
5. Compute $R(s,t) \bmod (s+t)^{n+1}$, using $R(s,t) = \bar{R}(s-ct,t)$.

One can check that the algorithm is well-defined, namely, each step can be carried out after its previous step is finished. By Lemma 4.4 below with $m = 1$, steps 1 and 5 can be done in $O((\log n)M(n^2))$ operations. Steps 2 and 4 amount to changing indices of coefficients; they need no operations defined by the field K. Since $\bar{\bar{P}}(\bar{\bar{s}},t)$ satisfies (4.1), step 3 can be computed by Algorithm 4.1 and thus takes $O(\sqrt{n}M(n^2))$ operations. Therefore, Algorithm 4.2 takes $O(\sqrt{n}M(n^2))$ operations and we have the following theorem.

Theorem 4.1.

For the composition problem $R(s,t) = Q(P(s,t))$ and the associated reversion problem, we have

$$\mathrm{COMP}\ (n) = O(\sqrt{n}M(n^2)),$$
$$\mathrm{REV}\ (n) = O(\sqrt{n}M(n^2)).$$

4.2. $R(s,t) = Q(s,P(s,t))$, $Q(s,V(s,t)) = C(s,t)$

Suppose that

$$(4.3) \qquad P(s,t) = t^r \hat{P}(s,t)$$

where $r \geq 1$ and $\operatorname{ord} \hat{P} = 0$. Algorithm 4.1 can be used to compute $R(s,t) \bmod (s,t)$, except that coefficients of Q are now power series in s instead of constants. Corresponding to Lemma 4.3, we have the following lemma:

Lemma 4.4.

$Q(s,P_m(s,t)) \bmod (s+t)^{n+1}$ can be computed in $O(m(\log n)M(n^2))$ operations.

Proof

The recurrence for the cost of computing $Q(s,P_m(s,t))$ is:

$$T(j) \leq 2T(j/2)+O(M_2(jm/4,jm/4;n)+M_2(jm/2,n;n)).$$

This implies that $T(n) = O(m(\log n)M(n^2))$. ■

Thus, Algorithm 4.1 now takes $O(m(\log n)M(n^2))+O((n/m)M(n^2))$ operations. Taking $m \sim (n/\log n)^{\frac{1}{2}}$, we conclude that if (4.3) holds then

$R(s,t)$ mod $(s+t)^{n+1}$ can be done in $O((n\log n)^{\frac{1}{2}}(M(n^2))$ operations.

Change of Variables

We consider the case when (4.3) is not satisfied. The change of variables technique used in §4.1 can again be applied, except that we should now choose c to be nonzero and that we have

$$\bar{\bar{R}}(\bar{\bar{s}},t) = Q(\bar{\bar{s}}t+ct,\bar{\bar{P}}(\bar{\bar{s}},t)),$$

where $\bar{\bar{P}}(\bar{\bar{s}},t)$ satisfies (4.3). Making another change of variables $t = (c+\bar{\bar{s}})^{-1}\tilde{t}$, we define $\tilde{P}(\bar{\bar{s}},\tilde{t}) = \bar{\bar{P}}(\bar{\bar{s}},(c+\bar{\bar{s}})^{-1}\tilde{t})$ and $\tilde{R}(\bar{\bar{s}},\tilde{t}) = \bar{\bar{R}}(\bar{\bar{s}},(c+\bar{\bar{s}})^{-1}\tilde{t})$. Then

$$\tilde{R}(\bar{\bar{s}},\tilde{t}) = Q(\tilde{t},\tilde{P}(\bar{\bar{s}},\tilde{t})).$$

We can use the following algorithm to compute $R(s,t)$ mod $(s+t)^{n+1}$.

Algorithm 4.3.

1. Compute $\bar{P}(\bar{s},t)$ mod $(\bar{s}+t)^{2n+1}$, using $\bar{P}(\bar{s},t) = P(\bar{s}+ct,t)$.
2. Compute $\bar{\bar{P}}(\bar{\bar{s}},t)$ mod $(\bar{\bar{s}}+t)^{2n+1}$, using $\bar{\bar{P}}(\bar{\bar{s}},t) = \bar{P}(\bar{\bar{s}}t,t)$.
3. Compute $\tilde{P}(\bar{\bar{s}},\tilde{t})$ mod $(\bar{\bar{s}}+\tilde{t})^{2n+1}$, using $\tilde{P}(\bar{\bar{s}},\tilde{t}) = \bar{\bar{P}}(\bar{\bar{s}},(c+\bar{\bar{s}})^{-1}\tilde{t})$.
4. Compute $\tilde{R}(\bar{\bar{s}},\tilde{t})$ mod $(\bar{\bar{s}}+\tilde{t})^{2n+1}$, using $\tilde{R}(\bar{\bar{s}},\tilde{t}) = Q(\tilde{t},\tilde{P}(\bar{\bar{s}},\tilde{t}))$.
5. Compute $\bar{\bar{R}}(\bar{\bar{s}},t)$ mod $(\bar{\bar{s}}+t)^{2n+1}$, using $\bar{\bar{R}}(\bar{\bar{s}},t) = \tilde{R}(\bar{\bar{s}},(c+\bar{\bar{s}})t)$.
6. Compute $\bar{R}(\bar{s},t)$ mod $(\bar{s}+t)^{n+1}$, using $\bar{R}(\bar{s},t) = \bar{\bar{R}}(\bar{s}/t,t)$.
7. Compute $R(s,t)$ mod $(s+t)^{n+1}$, using $R(s,t) = \bar{R}(s-ct,t)$.

Let $\bar{\bar{P}}(\bar{\bar{s}},t) = A_1(\bar{\bar{s}})t+A_2(\bar{\bar{s}})t^2+\cdots$. Then

$$\begin{aligned}\tilde{P}(\bar{\bar{s}},\tilde{t}) &= \bar{\bar{P}}(\bar{\bar{s}},(c+\bar{\bar{s}})^{-1}\tilde{t})\\ &= [A_1(\bar{\bar{s}})/(c+\bar{\bar{s}})]\tilde{t}+[A_2(\bar{\bar{s}})/(c+\bar{\bar{s}})^2]\tilde{t}^2+\cdots .\end{aligned}$$

Hence step 3 can be done in $O(nM(n))$ operations. Since $\tilde{P}(\bar{\bar{s}},\tilde{t})$ satisfies (4.3), step 4 can be done by Algorithm 4.1 in $O((n\log n)^{\frac{1}{2}}M(n^2))$ operations. let $\tilde{R}(\bar{\bar{s}},\tilde{t}) = B_0(\bar{\bar{s}})+B_1(\bar{\bar{s}})\tilde{t}+\cdots$. Then

$$\begin{aligned}\bar{\bar{R}}(\bar{\bar{s}},t) &= \tilde{R}(\bar{\bar{s}},(c+\bar{\bar{s}})t)\\ &= B_0(\bar{\bar{s}})+B_1(\bar{\bar{s}})(c+\bar{\bar{s}})t+\cdots .\end{aligned}$$

This implies that step 5 can be done in $O(nM(n))$ operations. By the arguments used in analyzing the cost of Algorithm 4.2, we see that steps 1, 2, 6 and 7 can be done in $O((\log n)M(n^2))$ operations. Therefore Algorithm 4.3 takes $O((n\log n)^{\frac{1}{2}}M(n^2))$ operations and we have the following theorem:

Theorem 4.2.

For the composition problem $R(s,t) = Q(s,P(s,t))$ and the associated reversion problem, we have

$$\begin{aligned}\text{COMP}\ (n) &= O((n\log n)^{\frac{1}{2}}M(n^2)),\\ \text{REV}\ (n) &= O((n\log n)^{\frac{1}{2}}M(n^2)).\end{aligned}$$

4.3. $R(s,t) = Q(P_1(s,t),\ P_2(s,t))$, $\begin{cases} Q_1(V_1(s,t),\ V_2(s,t)) = c_1(s,t),\\ Q_2(V_1(s,t),\ V_2(s,t)) = c_2(s,t)\end{cases}$

We shall reduce the present composition problem to that considered in §4.2, by the change of variables technique used in §3.3. Suppose that

(4.4) $\quad P_1(s,t) = t^r\hat{P}_1(s,t)$

where $r \geq 1$ and $\hat{P}_1$ has a nonzero constant term c. We make the change of variables $cu^r = P_1(s,t)$. Then

$$u = \bar{P}_1(s,t) \overset{\text{def}}{=} \left(\frac{P_1(s,t)}{c}\right)^{1/r} = t\left(\frac{\hat{P}_1(s,t)}{c}\right)^{1/r}.$$

One can easily check that the reversion problem $\overline{P}_1(s,V(s,u)) = u$ satisfies the hypotheses of Theorem 2.1. Hence there exists $V(s,u)$ such that $t = V(s,u)$. We have

$$R(s,V(s,u)) = \overline{Q}(s,u) \overset{\text{def}}{=} Q(cu^r,P_2(s,V(s,u))),$$

$$R(s,t) = R(s,V(s,\overline{P}_1(s,t))) = \overline{Q}(s,\overline{P}_1(s,t)).$$

Therefore, $R(s,t) \bmod (s+t)^{n+1}$ can be computed by the following algorithm:

Algorithm 4.4.

1. Compute $\overline{P}_1(s,t) \bmod (s+t)^{n+1}$.
2. Compute $V(s,u) \bmod (s+u)^{n+1}$.
3. Compute $P_2(s,V(s,u)) \bmod (s+u)^{n+1}$.
4. Compute $\overline{Q}(s,u) \bmod (s+u)^{n+1}$.
5. Compute $\overline{Q}(s,\overline{P}_1(s,t)) \bmod (s+t)^{n+1}$.

By Theorem 4.2 it is easy to see that the algorithm takes $O((n \log n)^{\frac{1}{2}}M(n^2))$ operations.

For the case when the condition (4.4) does not hold, the change of variables used in §4.2 can be applied and it takes $O((\log n)M(n^2))$ operations. Therefore we have established the following theorem:

Theorem 4.3.

For the composition problem $R(s,t) = Q(P_1(s,t), P_2(s,t))$ and the associated reversion problem, we have

$$\text{COMP}(n) = O((n \log n)^{\frac{1}{2}}M(n^2)),$$
$$\text{REV}(n) = O((n \log n)^{\frac{1}{2}}M(n^2)).$$

4.4. $R(s,t) = Q(P_1(s,t),\cdots,P_h(s,t))$, $\underline{Q}(s,t;\underline{V}(s,t)) = \underline{C}(s,t)$ an $h \times h$ System

By the change of variable technique used in §4.3, the composition problem $R(s,t) = Q(P_1(s,t),\cdots,P_h(s,t))$ can be reduced to $R(s,t) = Q(s,P_2(s,t),\cdots,P_h(s,t))$ and again to $R(s,t) = Q(s,t,P_3(s,t),\cdots,P_h(s,t))$. For $h = 3$, the last composition problem can clearly be done in $O(nM(n^2))$ operations, hence so can the first two problems. The $h = 4$, we can solve the last composition problem according to

$$R(s,t) = \Sigma Q_i(s,t,P_3(s,t))\cdot P_4(s,t)^i,$$

in $O(n^2M(n^2))$ operations, hence the first two can also be done in $O(n^2M(n^2))$ operations. In general, for $h \geq 3$, we have

$$\text{COMP}(n) = O(n^{h-2}M(n^2)),$$
$$\text{REV}(n) = O(n^{h-2}M(n^2)).$$

5. MULTIVARIATE COMPOSITIONS AND REVERSIONS - A SUMMARY

The techniques used in previous sections have been generalized to obtain results for more than two variables. In table 5.1 we give a summary of the new results together with the best previously known results, for some typical composition problems. The proofs of new results for more than two variables are not given in the paper. The reader who understands those algorithms used for the bivariate case should have no difficulty to generate his or her own proofs. Since each reversion problem is associated with a composition problem, results in Table 5.1 can be immediately translated into results on the corresponding reversion problems.

For the results in Table 5.1, $M(n)$ is taken to be $O(n \log n)$ and the lower bounds are obtained by counting coefficients in input power series.

Table 5.1

Bounds on the Number of Operations needed to Solve Composition Problems

Composition Problems	Best Previously known Upper Bounds	New Upper Bounds	Lower Bounds
$Q(P(s))$	$O(n^2 \log n)$	$O((n \log n)^{3/2})$†	$O(n)$
$Q(P_1(s),\ldots,P_h(s))$, $h>1$	$O(n^{h+1} \log n)$	$O(n^h \log n)$	$O(n^h)$
$Q(P(s_1,s_2))$	$O(n^3 \log n)$	$O(n^{2.5} \log n)$	$O(n^2)$
$Q(s_1,P(s_1,s_2))$	$O(n^3 \log n)$	$O(n^{2.5} \log^{1.5} n)$	$O(n^2)$
$Q(P_1(s_1,s_2),P_2(s_1,s_2))$	$O(n^4 \log n)$	$O(n^{2.5} \log^{1.5} n)$	$O(n^2)$
$Q(P_1(s_1,s_2),\ldots,P_h(s_1,s_2))$ $h>2$	$O(n^{h+2} \log n)$	$O(n^h \log n)$	$O(n^h)$
$Q(P(\underset{\sim}{s}))$	$O(n^{k+1} \log n)$	$O(n^{k+.5} \log n)$	$O(n^k)$
$Q(s_1,\ldots,s_j,P_{j+1}(\underset{\sim}{s}),\ldots,P_k(\underset{\sim}{s}))$ $0 \leq j \leq k-1$, $\underset{\sim}{s} = (s_1,\ldots,s_k)$	$O(n^{2k-j} \log n)$	$O(n^{k+.5} \log^{1.5} n)$	$O(n^k)$
$Q(P_1(\underset{\sim}{s}),\ldots,P_h(\underset{\sim}{s}))$ $h>k$, $\underset{\sim}{s} = (s_1,\ldots,s_k)$	$O(n^{h+k} \log n)$	$O(n^h \log n)$	$O(n^h)$

† This result was previously obtained in Brent and Kung [2].

REFERENCES

1. Brent, R. P., "Multiple-Precision Zero-Finding Methods and the Complexity of Elementary Function Evaluation," in Analytic Computational Complexity, ed. by J. F. Traub, Academic Press, New York, 1976, 151-176.

2. Brent, R. P. and Kung, H. T., "Fast Algorithms for Manipulating Formal Power Series," Computer Science Department Report, Carnegie-Mellon University, January 1976.

3. Estes, R. H. and Lancaster, E. R., "Some Generalized Power Series Inversions," SIAM J. Numer. Anal. 9 (1972), 241-247.

4. Goldstein, A. J. and Hall, A. D., "Solution to a Problem in Power Series Reversion," SIAM J. Math Anal. 6 (1975), 192-198.

5. Hall, A. D., "Solving a Problem in Eigenvalue Approximation with a Symbolic Algebra System," SIAM J. Comput. 4 (1975), 163-174.

6. Kung, H. T., "On Computing Reciprocals of Power Series," Numer. Math. 22 (1974), 341-348.

7. Kung, H. T. and Traub, J. F., "All Algebraic Functions Can Be Computed Fast," Computer Science Department Report, Carnegie-Mellon University, July 1976. To appear in J.ACM.

8. Lipson, J. D., "Newton's Method: A Great Algebraic Algorithm," Proc. 1976 ACM Symposium on Symbolic and Algebraic Computation, ed. by R. K. Jenks, 260-270.