$O((n \log n)^{3/2})$ ALGORITHMS FOR COMPOSITION AND REVERSION OF POWER SERIES

Richard P. Brent
Computer Centre
Australian National University

H. T. Kung
Department of Computer Science
Carnegie-Mellon University

ABSTRACT

Let $P(s) = p_1 s + p_2 s^2 + \ldots$ and $Q(t) = q_0 + q_1 t + \ldots$ be formal power series.

The <u>composition</u> of Q and P is the power series $R(s) = r_0 + r_1 s + \ldots$ such that $R(s) = Q(P(s))$. The <u>composition problem</u> is to compute $r_0, \ldots, r_n$, given $p_1, \ldots, p_n$ and $q_0, \ldots, q_n$.

The <u>functional inverse</u> of P is the power series $V(t) = v_1 t + v_2 t^2 + \ldots$ such that $P(V(t)) = t$ or $V(P(s)) = s$. The <u>reversion problem</u> is to compute $v_1, \ldots, v_n$, given given $p_1, \ldots, p_n$.

The classical algorithms for both the composition and reversion problems require $O(n^3)$ operations (see, e.g., Knuth, Vol. 2). In this paper we describe algorithms which can solve both problems in $O((n \log n)^{3/2})$ operations. The techniques used to obtain our results are applicable to several other problems.

1. INTRODUCTION

Let k be a field, which contains an nth root of unity for every positive integer n. (For example, k could be the field of complex numbers.) Let p_i, q_i, $i = 0,1,\dots$, be indeterminates over k, A the extension field $k(p_0,q_0,p_1,q_1,\dots)$, and s, t indeterminates over A. Suppose that E and F are finite subsets of A and that we perform computations in the field A. Let L(E mod F) denote the number of operations necessary to compute E starting from $k \cup F$.

Let $P(s) = p_1 s + p_2 s^2 + p_3 s^3 + \dots$ and $Q(t) = q_0 + q_1 t + q_2 t^2 + \dots$ be formal power series over A. The <u>composition</u> of Q and P is the power series $R(s) = r_0 + r_1 s + r_2 s^2 + \dots$ such that $R(s) = Q(P(s))$ is a formal identity. The <u>composition problem</u> is to compute $r_0,\dots,r_n$, given $\{p_1,\dots,p_n,q_0,\dots,q_n\} \cup k$. Define

$$\mathrm{COMP}(n) = L(r_0,\dots,r_n \bmod p_1,\dots,p_n,q_0,\dots,q_n)$$

Let $P(s) = p_1 s + p_2 s^2 + p_3 s^3 + \dots$ be a formal power series over A. The <u>functional inverse</u> of P is the power series $V(t) = v_1 t + v_2 t^2 + v_3 t^3 + \dots$ over A such that $P(V(t)) = t$ or $V(P(s)) = s$ is a formal identity. The <u>reversion problem</u> is to compute $v_1,\dots,v_n$, given $\{p_1,\dots,p_n\} \cup k$. Define

$$\mathrm{REV}(n) = L(v_1,\dots,v_n \bmod p_1,\dots,p_n).$$

The classical algorithms for both the composition and reversion problems require $O(n^3)$ operations (see, e.g., Knuth [71]), or $O(n^2 \log n)$ operations if the fast Fourier transform is used for polynomial multiplication as pointed out in Kung and Traub [74, Section 4]. In this paper we describe

algorithms which can solve both problems in $O((n \log n)^{3/2})$ operations.

In another paper, Brent and Kung [75], we shall give a complete treatment of the subject, which will include the following:

(i) The proof that the composition and reversion problems are equivalent (up to constant factors) if $\mathrm{MULT}(n) = O(\mathrm{REV}(n))$, where MULT(n) is the number of operations needed to multiply two nth degree polynomials.

(ii) Other algorithms requiring, e.g., $O(n^2)$ and $O(n^{1.9037})$ operations which do not use the fast Fourier transform and are faster for small n.

(iii) An algorithm which can evaluate the truncated functional inverse, i.e., $V_n(t) = v_1 t + v_2 t^2 + \dots + v_n t^n$, at one point in $O(n \log n)$ operations, and its application to the root-finding problem.

2. PRELIMINARY LEMMAS

Let $P(s) = p_0 + p_1 s + \dots$, $Q(s) = q_0 + q_1 s + \dots$, $U(s) = u_0 + u_1 s + \dots$, etc. be formal power series over A.

<u>Lemma 2.1</u>

<u>If $U(s) = P(s)Q(s)$, then</u>

$$\underline{L(u_0,\dots,u_n \bmod p_0,\dots,p_n,q_0,\dots,q_n) = O(n \log n)}$$

Proof

Use the fast Fourier transform (see, e.g. Knuth [71, p. 441]). ■

Lemma 2.2

If $U(s) = P(s)/Q(s)$, then

$$L(u_0,\ldots,u_n \bmod p_0,\ldots,p_n,q_0,\ldots,q_n) = O(n \log n)$$

Proof

Use Lemma 2.1 and Newton's method as in Kung [74]. ■

Lemma 2.3

If $P(s) = p_1 s + p_2 s^2 + \ldots$, $R(s) = Q(P(s))$ and $D(s) = Q'(P(s))$, then

$$L(d_0,\ldots,d_n \bmod r_0,\ldots,r_n,p_1,\ldots,p_n) = O(n \log n).$$

(Here the prime denotes formal differentiation with respect to s.)

Proof

By chain rule, $R'(s) = Q'(P(s))\cdot P'(s)$. Hence $D(s) = R'(s)/P'(s)$, and the result follows from Lemma 2.2. ■

Lemma 2.4

If $P(s) = p_1 s + \ldots + p_m s^m$,

$Q(t) = q_0 + q_1 t + \ldots + q_j t^j$, where $m \le n$ and $j \le n$ and

$R(s) = Q(P(s)) = r_0 + r_1 s + \ldots$, then

$$L(r_0,\ldots,r_n \bmod p_1,\ldots,p_m, q_0,\ldots,q_j)$$

$$= O(jm(\log n)^2).$$

Proof

We may assume that j is a power of 2. Write $R = Q_1(P) + P^{j/2} \cdot Q_2(P)$, where Q_1 and Q_2 are polynomials of degree $j/2$. During the computation we always truncate terms of degree higher than n.

The proof is by induction, so we can assume that $P^{j/4}$ is known. Thus, $P^{j/2}$ can be computed with $O(jm \log jm) = O(jm \log n)$ additional operations, and multiplication by $Q_2(P)$ also requires $O(jm \log n)$ operations. If $T(j)$ operations are required to compute R and $P^{j/2}$, then Q_1 and Q_2 may each be computed in $T(j/2)$ operations. Thus,

$$T(j) \le 2T(j/2) + O(jm \log n),$$

so

$$T(j) = O(jm(\log n)(\log j)) = O(jm(\log n)^2).$$ ■

Lemma 2.4 can also be proved by using the fast evaluation and interpolation algorithms of Moenck and Borodin [72], but this method involves larger asymptotic constants and may have numerical stability problems.

3. THE COMPOSITION PROBLEM

Write $P(s) = P_h(s) + P_r(s)$, where $P_h(s) = p_1 s + p_2 s^2 + \ldots + p_m s^m$ and $P_r(s) = p_{m+1}s^{m+1} + p_{m+2}s^{m+2} + \ldots$, for $m = \left\lceil \sqrt{\frac{n}{\log n}} \right\rceil$. Then

$$Q(P) = Q(P_h + P_r)$$
$$= Q(P_h) + Q'(P_h)P_r + \frac{1}{2}Q''(P_h)(P_r)^2 + \dots .$$

Let $\ell = \lceil \frac{n}{m} \rceil$. Since the degree of any term in $(P_r)^{\ell+i}$ is $\geq n+1$ for any $i > 0$,

$$Q(P(s)) = Q(P_h) + Q'(P_h)P_r + \dots + \frac{1}{\ell!}Q^{(\ell)}(P_h)(P_r)^{\ell} + O(s^{n+1}).$$

This equality gives us the following algorithm for computing the first n coefficients of $R(s) = Q(P(s))$:

Step 1. Compute the first n coefficients of $W(s) = Q(P_h(s))$. By Lemma 2.4 with $j = n$ and m as above, this can be done in $O((n \log n)^{3/2})$ operations.

Step 2. Compute the first n coefficients of $Q'(P_h(s))$, $Q''(P_h(s))$, ..., $Q^{(\ell)}(P_h(s))$. By Lemma 2.3, it takes $O(n \log n)$ operations for each $Q^{(j)}(P_h(s))$. Hence the whole step can be done in $O(\ell\, n \log n) = O((n \log n)^{3/2})$ operations.

Step 3. Compute the first n coefficients of $P_r^2(s), P_r^3(s), \dots, P_r^{\ell}(s)$.

Step 4. Compute the first n coefficients of $Q'(P_h(s))P_r(s), \dots, \frac{1}{\ell!}Q^{(\ell)}(P_h(s))(P_r(s))^{\ell}$.

Step 5. Sum the results obtained from step 4.

It is clear that steps 3, 4 and 5 can be done in $O((n \log n)^{3/2})$ operations. Therefore, we have shown the following

Theorem 3.1

$COMP(n) = O((n \log n)^{3/2})$.

4. THE REVERSION PROBLEM

Define function $f: A(t) \to A(t)$ by $f(x) = P(x) - t$. Suppose that $V(t)$ is the functional inverse of P. Then $P(V(t)) = t$. Hence $V(t)$ is the zero of f, and the reversion problem can be viewed as a zero-finding problem. We shall use Newton's method to find the zero of f; other iterations can also be used successfully. (See Kung [74] for a similar technique for computing the reciprocals of power series and also Brent [75, Section 13].) The iteration function given by Newton's method is

$$(4.1)\quad \varphi(x) = x - \frac{f(x)}{f'(x)}$$
$$= x - \frac{P(x)-t}{P'(x)},$$

so we have

$$(4.2)\quad \varphi(x) - V(t)$$
$$= x - V(t) - \frac{(P(V(t)) + P'(V(t))(x-V(t)) + \dots) - t}{P'(V(t)) + P''(V(t))(x-V(t)) + \dots}$$
$$= \frac{P''(V(t))}{2P'(V(t))}(x - V(t))^2 + O(x - V(t))^3.$$

Suppose that the first n coefficients, $v_1, v_2, \dots, v_n$, of $V(t)$ have already been computed. Let x be taken to be $V_n(t) = v_1 t + v_2 t^2 + \dots + v_n t^n$. Then by (4.2)

$$\varphi(V_n(t)) = V(t) + O(t^{2n+2}).$$

Hence by computing the first 2n+1 coefficients of $\varphi(V_n(t))$ we obtain the first 2n+1 coefficents of V(t). Hence by (4.1) and Lemmas 2.2, 2.3, we have

$$(4.3) \quad REV(2n+1) \le REV(n) + COMP(2n+1) + O(n \log n).$$

Therefore, by (4.3) and Theorem 3.1 we have shown the following

<u>Theorem 4.1</u>

$\underline{REV(n) = O((n \log n)^{3/2})}$.

ACKNOWLEDGMENT

The authors want to thank J. F. Traub of Carnegie-Mellon University for his comments on the paper.

REFERENCES

Brent [75] Brent, R. P., "Multiple-Precision Zero-Finding Methods and the Complexity of Elementary Function Evaluation," these proceedings.

Brent and Kung [75] Brent, R. P. and H. T. Kung, to appear, 1975.

Knuth [71] Knuth, D. E., <u>The Art of Computer Programming</u>, Vol. 2, Addison-Wesley, Reading, Massachusetts, 1971.

Kung [74] Kung, H. T., "On Computing Reciprocals of Power Series," <u>Numer. Math.</u> 22, 1974, 341-348.

Kung and Traub [74] Kung, H. T. and J. F. Traub, "Computational Complexity of One-Point and Multipoint Iteration," in <u>Complexity of Computation</u>, edited by R. Karp, SIAM-AMS Proc., Vol. 7, American Mathematical Society, 1974, 149-160.

Moenck and Borodin [72] Moenck, R. and A. B. Borodin, "Fast Modular Transforms via Division, <u>Conf. Record IEEE 13th Annual Symposium on Switching and Automata</u>, 1972, 90-96.