SIAM J. COMPUT.
Vol. 6, No. 3, September 1977

FAST ALGORITHMS FOR PARTIAL FRACTION DECOMPOSITION*

H. T. KUNG† AND D. M. TONG‡

Abstract. The partial fraction decomposition of a proper rational function whose denominator has degree n and is given in general factored form can be done in $O(n \log^2 n)$ operations in the worst case. Previous algorithms require $O(n^3)$ operations, and $O(n \log^2 n)$ operations for the special case where the factors appearing in the denominator are all linear.

Key words. fast algorithms, partial fraction decomposition, computational complexity

1. Introduction. Let

$$\frac{P(x)}{\prod_{i=1}^{k} Q_i^{l_i}(x)}$$

be a given fraction, where the P, Q_i are polynomials and the l_i are positive integral exponents such that

1. $\deg P < \sum_{i=1}^{k} l_i \cdot \deg Q_i = n$, and
2. $Q_1, \cdots, Q_k$ are relatively prime.

The *general partial fraction decomposition problem* (*general* PF *problem*) is to compute the coefficients of the polynomials $C_{i,j}$ for $i = 1, \cdots, k$ and $j = 1, \cdots, l_i$ such that

$$\frac{P(x)}{\prod_{i=1}^{k} Q_i^{l_i}(x)} = \sum_{i=1}^{k} \sum_{j=1}^{l_i} \frac{C_{i,j}(x)}{Q_i^j(x)}$$

with $\deg C_{i,j} < \deg Q_i$ for all i, j. The existence and uniqueness of the polynomials $C_{i,j}$ are well known (see, e.g., van der Waerden (1953, § 29). There are enormous applications of partial fractions in applied mathematics and in network theory (see, e.g., Henrici (1974, Chap. 7) and Weinberg (1962)). This paper gives fast algorithms for solving the general partial fraction decomposition problem when n is large.

Previous algorithms for the problem usually involve solving systems of linear equations (see Henrici (1974, Chap. 7) for a nice summary). Hence they take $O(n^3)$ arithmetic operations, or $O(n^{2.81})$ operations if Strassen's method (Strassen (1969)) is used. For the special case that the Q_i have either degree one or two, many algorithms were known: see, e.g., Schwatt (1924, Chap. VIII), Turnbull (1927), Hazony and Riley (1959), Pottle (1964), Pessen (1965), Brugia (1965), Moad (1966), Valentine (1967), Wehrhahn (1967), Karni (1969) and Linnér (1974). But these algorithms still take $O(n^2)$ or more operations.

Recently Chin and Ullman (1975) showed that in case that all the Q_i have degree one the problem can be done in $O((n \log n)^{3/2})$ operations. This bound

* Received by the editors February 19, 1976. This work was supported in part by the National Science Foundation under Grant MCS75-222-55 and the Office of Naval Research under Contract N00014-76-C-0370, NR044-422.

† Department of Computer Science, Carnegie–Mellon University, Pittsburgh, Pennsylvania 15213.

‡ Department of Mathematics, Kalamazoo College, Kalamazoo, Michigan 49001.

was further improved by Chin in his thesis (Chin (1975)). He showed that if the Q_i are all linear, then the problem can be done in $O((\log k)\cdot(n \log n))$ operations. However, the assumption that the Q_i are all linear factors is crucial in his methods. Hence the problem of solving the general PF problem (without assuming that the Q_i are linear) in $O(n^2)$ operations is stated as an unsolved problem in his thesis. Note that the general PF problem does occur frequently in practice. For example, if we work over the field of real numbers, then the factors Q_i certainly can have either degree one or two. (See also Grau (1971) and Henrici (1971) for more examples.) *In this paper, we show that the general* PF *problem can be done in* $O((\log n)\cdot M(n))$ *operations in the worst case.* $M(n)$ *is any upper bound on the number of operations needed to multiply two* n*-th degree polynomials, which satisfies some mild regularity condition* (see § 2). In particular, if an FFT algorithm is used for polynomial multiplication (see, e.g., Knuth (1969, § 4.6.4), Borodin and Munro (1975)), then we have $M(n)=O(n \log n)$, which satisfies the regularity condition, and hence the general PF problem can be done in $O(n \log^2 n)$ operations. Moreover, we note that for the special case where the Q_i are all linear, our approach will lead to Chin's $O((\log k)\cdot(n \log n))$ algorithm.

Basic assumptions and preliminary lemmas used in this paper are introduced in § 2. In § 3, the solution of the general PF problem is reduced to the solution of two simpler problems, Problem P1 and Problem P2, and precise statements of the main results of the paper are given. An algorithm, based on a *new theorem* (Theorem 4.1), for solving Problem P1 is presented in § 4. Section 5 contains an algorithm for solving Problem P2. Finally, an important special case of Problem P2 is solved in § 6.

2. Basic assumptions and preliminary lemmas. We assume throughout the paper that polynomials are over some field K, are denoted by upper case letters, and are given in the form $P(x)=\sum p_i x^i$ where $p_i \in K$. To compute P or $P(x)$ means to find all the coefficients of P. We assume that $M(n)$ is an upper bound on the number of operations needed to multiply two nth degree polynomials. Given relatively prime polynomials A_1, A_2 with $\deg A_1$, $\deg A_2 \leqq n$, let $F(n)$ be an upper bound on the number of operations to find polynomials F_1, F_2 such that

$$F_2\cdot A_1+F_1\cdot A_2=1$$

with $\deg F_1<\deg A_1$ and $\deg F_2<\deg A_2$. The existence and uniqueness of F_1 and F_2 are well-known (see, e.g., van der Waerden (1953, § 29)).

Let Z^+ be the set of all nonnegative integers and let $G: Z^+\to Z^+$ be a nondecreasing function. We say G satisfies *Condition* C, if

$$G(n)=n\cdot H(n)$$

for some nondecreasing function $H: Z^+\to Z^+$. *We assume that M satisfies Condition* C. Similar regularity conditions are usually assumed (see, e.g., Aho, Hopcroft and Ullman (1974, p. 280), Brent and Kung (1976) and Moenck (1973b)). There are many algorithms for polynomial multiplication. For example, the classical algorithm gives $M(n)=c_1 n^2$, binary splitting multiplication gives $M(n)=c_2 n^{1.585}$, and if the field K is algebraically closed, then FFT multiplication gives $M(n)=c_3 n \log n$, where c_1, c_2, c_3 are positive constants (see e.g., Fateman (1974)). In all cases M satisfies Condition C. In fact all we need in this paper are some consequences of Condition C. Hence it is possible to weaken our assumption on M, if one wishes to do so.

Let $D(n)$ be the number of operations needed to divide a polynomial of degree $2n$ by a polynomial of degree n. Then using Newton's method and the fact that M satisfies Condition C, one can show the following lemma (see, e.g., Borodin and Munro (1975) and Kung (1974)).

LEMMA 2.1. $D(n) = O(M(n))$.

Using the algorithm EGCD in Moenck (1973a), which is a generalization of an algorithm due to Schöenhage (1971) for integer GCDs, one can show the following lemma.

LEMMA 2.2. $F(n) = O((\log n) \cdot M(n))$.

We shall assume that F satisfies the condition that

$$\sum F(n_i) \leqq F\left(\sum n_i\right)$$

for any $n_i \in Z^+$. Clearly, if $F(n) = c \cdot (\log n) \cdot M(n)$ for some positive constant c as in Lemma 2.2, then F satisfies the condition. In fact, the required condition in F is satisfied as long as F satisfies Condition C.

3. Problems P1, P2 and statement of results. Consider the following two instances of the general PF problem defined in § 1.

PROBLEM P1. (This is the general PF problem with $l_i = 1$ for all i.) Given the fraction $P/\prod_{i=1}^k R_i$ where the R_i are relatively prime and

$$\deg P < \sum_{i=1}^{k} \deg R_i = n,$$

compute the polynomials $C_1, \cdots, C_k$ such that

$$\frac{P(x)}{\prod_{i=1}^k R_i(x)} = \sum_{i=1}^{k} \frac{C_i(x)}{R_i(x)} \tag{3.1}$$

with $\deg C_i < \deg R_i$ for all i.

The decomposition (3.1) is called the incomplete partial fraction decomposition by Henrici (1971), (1974, Chap. 7). Note also that efficient algorithms for solving Problem P1 will furnish efficient procedures for factoring polynomials, as observed by Grau (1971).

PROBLEM P2. (This is the general PF problem with $k = 1$.) Given the fraction P/Q^l where $\deg P < l \cdot \deg Q$ compute the polynomials $C_1, \cdots, C_l$ such that

$$\frac{P(x)}{Q^l(x)} = \sum_{j=1}^{l} \frac{C_j(x)}{Q^j(x)}$$

with $\deg C_j < \deg Q$ for all j.

The following lemma essentially shows that fast algorithms for Problems P1 and P2 will lead to fast algorithms for the general PF problem. *Define* $T(k, n)$, $T_1(k, n)$ *and* $T_2(l, \deg Q)$ *to be the number of operations needed to solve the general* PF *problem, Problem* P1 *and Problem* P2, *respectively*.

LEMMA 3.1.

$$T(k, n) \leqq T_1(k, n) + \sum_{i=1}^{k} [T_2(l_i, \deg Q_i) + O(M(l_i \cdot \deg Q_i))].$$

Proof. The result follows from the observation that general PF problem can be solved in the following way:

1. Multiply $Q_i^{l_i}(x)$ out for $i = 1, \cdots, k$. Let the expansion of $Q_i^{l_i}(x)$ be $R_i(x)$ for all i.

2. Solve Problem P1 for the fraction $P/\prod_{i=1}^{k} R_i$ and obtain the polynomials C_i satisfying (3.1).

3. Solve Problem P2 for the fractions $C_i/Q_i^{l_i}$, $i = 1, \cdots, k$.

Note that each $Q_i^{l_i}(x)$ can be computed in $O(M(l_i \cdot \deg Q_i))$ operations by an algorithm in Brent (1976, § 13). □

We summarize our results on $T_1(k, n)$ and $T_2(l, \deg Q)$ in the following:

(i) $T_1(k, n) \leqq F(n) + O((\log k) \cdot M(n))$. (Theorem 4.2)

(ii) $T_1(k, n) = O((\log k) \cdot (n \log n))$, when the $R_i(x)$ is given in the form $(x - z_i)^{m_i}$ for all i. (Theorem 4.3)

(iii) $T_2(l, \deg Q) = O((\log l) \cdot M(l \cdot \deg Q))$. (Theorem 5.1)

(iv) $T_2(l, \deg Q) = O(l \log l)$, when $\deg Q \leqq 2$. (Theorems 6.1 and 6.2)

We have the following results for the general partial fraction decomposition problem.

THEOREM 3.1. *The general* PF *problem can be done in* $F(n) + O((\log k) \cdot M(n)) + O((\log l) \cdot M(n))$ *operations, where* $l = \max(l_1, \cdots, l_k)$.

Proof. Note that

$$\begin{aligned}\sum_{i=1}^{k} (\log l_i) \cdot M(l_i \cdot \deg Q_i) &\leqq (\log l) \sum_{i=1}^{k} l_i \cdot \deg Q_i \cdot H(l_i \cdot \deg Q_i) \\ &\leqq (\log l) \cdot n \cdot H(n) = (\log l) \cdot M(n).\end{aligned}$$

The result follows from (i), (iii) and Lemma 3.1. □

COROLLARY 3.1. *The general* PF *problem can be done in* $O(n \log^2 n)$ *operations.*

Proof. Note that in Theorem 3.1, $k \leqq n$ and $l \leqq n$. The result follows from the theorem and Lemma 2.2 by letting $M(n) = O(n \log n)$. □

$O(n \log^2 n)$ is the best asymptotic bound known for the general PF problem.

THEOREM 3.2. *The general* PF *problem can be done in* $O((\log k) \cdot (n \log n))$ *operations, if* $Q_i(x) = x - z_i$, *for* $i = 1, \cdots, k$.

Proof. The result follows from (ii), (iv) and Lemma 3.1. □

The bound in Theorem 3.2 was obtained previously by Chin (1975). We include it here just to show that his result will emerge as a special case in our general approach. See the remarks at the end of § 4.

4. An algorithm for problem P1. We first assume that $P(x) \equiv 1$ in Problem P1. Thus we want to find $A_1, \cdots, A_k$ such that

$$\frac{1}{\prod_{i=1}^{k} R_i(x)} = \sum_{i=1}^{k} \frac{A_i(x)}{R_i(x)}$$

with $\deg A_i < \deg R_i$ for all i. Note that

$$1 = \sum_{i=1}^{k} \left[A_i(x) \prod_{\substack{j=1 \\ j \neq i}}^{k} R_j(x) \right]. \tag{4.1}$$

Define

$$R(x) = \sum_{i=1}^{k} \left(\prod_{\substack{j=1 \\ j \neq i}}^{k} R_j(x) \right),$$

and for each $i = 1, \cdots, k$, define B_i, D_i by

$$R(x) = B_i(x) R_i(x) + D_i(x) \tag{4.2}$$

where $\deg D_i < \deg R_i$. Note that $D_i(x) \not\equiv 0$, since the R_i are relatively prime. Suppose that $\deg D_i \geqq 1$, i.e., $D_i(x)$ is not a constant. Then (4.2) implies that D_i and R_i are relatively prime, since R_i and R are relatively prime. Hence there exist unique polynomials $\tilde{A}_i$ and E_i such that

$$\tilde{A}_i(x) D_i(x) + E_i(x) R_i(x) = 1 \tag{4.3}$$

with $\deg \tilde{A}_i < \deg R_i$ and $\deg E_i < \deg D_i$. The following theorem appears to be new.

THEOREM 4.1. *For $i = 1, \cdots, k$, if $D_i(x) \equiv d_i$ for some constant d_i, then A_i is the constant $1/d_i$; else $A_i = \tilde{A}_i$.*

Proof. We classify the zeros of R_i according to their multiplicities. Let Z_m be the set of zeros of R_i which have multiplicity m. (The zeros exist in an algebraically closed extension field of K.) Clearly, we have that

(4.4) $\sum m \cdot |Z_m| = \deg R_i$, where $|Z_m|$ is the number of elements in Z_m,

and that if $z \in Z_m$ then

$$R_i^{(h)}(z) = 0 \quad \text{for } h = 0, \cdots, m-1. \tag{4.5}$$

Taking derivatives of (4.1) and (4.3), and using (4.5), one can easily show that

$$\sum_{q=0}^{h} \binom{h}{q} A_i^{(q)}(z) \cdot \left(\prod_{\substack{j=1 \\ j \neq i}}^{k} R_j \right)^{(h-q)} (z) = \delta_{0,h}, \tag{4.6}$$

$$\sum_{q=0}^{h} \binom{h}{q} \tilde{A}_i^{(q)}(z) \cdot D_i^{(h-q)}(z) = \delta_{0,h} \tag{4.7}$$

for $z \in Z_m$ and $h = 0, \cdots, m-1$, where $\delta_{0,h} = 1$ if $h = 0$ and $\delta_{0,h} = 0$ otherwise. (Derivatives are represented by the superscripts.) Note that by (4.2) and (4.5),

$$\begin{aligned} D_i^{(h-q)}(z) &= R^{(h-q)}(z) \\ &= \left(\prod_{\substack{j=1 \\ j \neq i}}^{k} R_j \right)^{(h-q)} (z) \end{aligned} \tag{4.8}$$

for $z \in Z_m$, $h = 0, \cdots, m-1$ and $q = 0, \cdots, h$. Suppose that $D_i(x) \equiv d_i$ for some constant d_i. Then by (4.6) and (4.8),

$$A_i^{(h)}(z) \cdot d_i = \delta_{0,h} \tag{4.9}$$

for $z \in Z_m$, $h = 0, \cdots, m-1$. Since $d_i \neq 0$ and $\deg A_i < \deg R_i$, A_i is uniquely determined by the Hermite interpolation problem defined by (4.9). Hence $A_i(x) \equiv 1/d_i$. On the other hand, suppose that $\deg D_i \geqq 1$. Because the R_i are relatively prime,

$$D_i(z) = \left(\prod_{\substack{j=1 \\ j \neq i}}^{k} R_j \right)(z) \neq 0 \tag{4.10}$$

for $z \in Z_m$. By (4.6), (4.7), (4.8) and (4.10) it is easy to see that A_i and $\tilde{A}_i$ are deteremined by the *same* Hermite interpolation problem. This implies that $A_i = \tilde{A}_i$. □

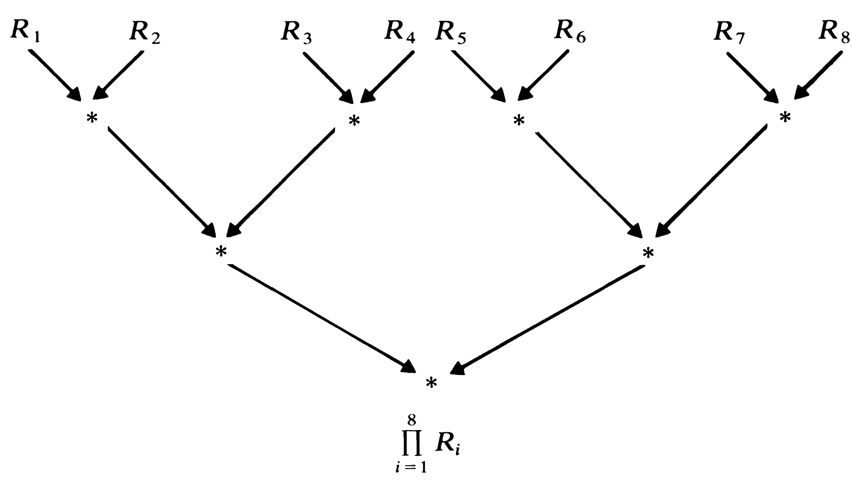
By Theorem 4.1 the following algorithm can be used for computing $A_i(x)$ for $i = 1, \cdots, k$.

ALGORITHM 4.1.

1. Compute $R(x)$.
2. Compute $D_i(x)$ for $i = 1, \cdots, k$.
3. For $i = 1, \cdots, k$, if $D_i(x) \equiv d_i$ for some constant d_i then set $A_i(x) \leftarrow 1/d_i$ else compute $A_i(x)$ by solving (4.3).

In the following we study the number of operations needed by the algorithm.

It is well known that $\prod_{i=1}^{k} R_i(x)$ can be computed by using a binary splitting scheme, which is illustrated as follows for the case $k = 8$:



LEMMA 4.1. *By using the binary splitting, $\prod_{i=1}^{k} R_i(x)$ and all the intermediate results such as $\prod_{i=1}^{2} R_i(x)$ and $\prod_{i=5}^{8} R_i(x)$ can be computed in $O((\log k) \cdot M(n))$ operations.*

Proof. Note that the sum of the degrees of all the polynomials at any level of the tree is n. Hence each level takes $M(n)$ operations, since M satisfies Condition C. The result then follows from the fact that the height of the tree is $\lceil \log_2 k \rceil$. □

LEMMA 4.2. *$R(x)$ can be computed in $O((\log k)\cdot M(n))$ operations.*

Proof. We shall again use the binary splitting technique. We may assume that k is a power of 2. It is easy to check that

$$\sum_{i=1}^{k}\left(\prod_{\substack{j=1\\ j\neq i}}^{k} R_j\right)=\left(\prod_{j=k/2+1}^{k} R_j\right)\cdot\sum_{i=1}^{k/2}\left(\prod_{\substack{j=1\\ j\neq i}}^{k/2} R_j\right)+\left(\prod_{j=1}^{k/2} R_j\right)\cdot\sum_{i=k/2+1}^{k}\left(\prod_{\substack{i=k/2+1\\ j\neq i}}^{k} R_j\right).$$

This gives us a recursive algorithm for computing R. By Lemma 4.1, we may assume that all the products such as $\prod_{j>k/2} R_j$ and $\prod_{j\leqq k/2} R_j$ needed by the algorithm have been precomputed. The result again follows from the fact that the sum of the degrees of all polynomials at any level of the associated binary tree is n. $\square$

LEMMA 4.3. *$D_1(x),\cdots,D_k(x)$ can be computed in $O((\log k)\cdot M(n))$ operations.*

Proof. We may assume that k is a power of 2. Note that if we use divisions to obtain V_1 and V_2 such that

$$R(x)=U_1(x)\cdot\prod_{j=1}^{k/2} R_i(x)+V_1(x),$$

$$R(x)=U_2(x)\cdot\prod_{j=k/2+1}^{k} R_i(x)+V_2(x),$$

where $\deg V_1<\deg\prod_{j=1}^{k/2} R_i$ and $\deg V_2<\deg\prod_{j=k/2+1}^{k} R_i$, then the problem of computing D_i from R for $i=1,\cdots,k$ is reduced to the problems of computing D_i from V_1 for $i=1,\cdots,k/2$ and computing D_i from V_2 for $i=k/2+1,\cdots,k$. This again gives us a recursive procedure. Using the fact that $D(n)=O(M(n))$ (Lemma 2.1), the lemma can be proved by the same argument as used in the proofs of Lemmas 4.1 and 4.2. $\square$

LEMMA 4.4. *$A_1(x),\cdots,A_k(x)$ can be computed in $F(n)$ operations.*

Proof. Since $\deg D_i<\deg R_i$, the $A_i(x)$ and $E_i(x)$ satisfying (4.3) can be computed in $F(\deg R_i)$ operations. Hence all the A_i can be computed in

$$\sum_{i=1}^{k} F(\deg R_i)\leqq F\left(\sum_{i=1}^{k}\deg R_i\right)=F(n)$$

operations. $\square$

By Lemmas 4.2, 4.3 and 4.4, we know that Algorithm 4.1 can be done in $F(n)+O((\log k)\cdot M(n))$ operations. After the A_i have been computed, we can solve Problem P1 without assuming $P(x)\equiv 1$ in $O((\log k)\cdot M(n))$ operations by the following method: For $i=1,\cdots,k$,

1. compute $K_i(x)$ such that

$$P(x)=J_i(x)R_i(x)+K_i(x)$$

with $\deg K_i<\deg R_i$, for some J_i,

2. compute $L_i(x)=K_i(x)\cdot A_i(x)$ and $C_i(x)$ such that

$$L_i(x)=N_i(x)R_i(x)+C_i(x)$$

with $\deg C_i<\deg R_i$, for some N_i.

Note that

$$\frac{P}{\prod R_i}=\sum\left(\frac{P}{R_i}\right)A_i$$
$$=\sum\left(J_i+\frac{K_i}{R_i}\right)A_i$$
$$=\sum J_iA_i+\sum\frac{L_i}{R_i}$$
$$=\sum J_iA_i+\sum N_i+\sum\frac{C_i}{R_i}$$

Since $P/\prod R_i$ is a proper fraction, $\sum J_iA_i+\sum N_i$ must be zero. Therefore the C_i are the desired solution. Since $\deg P<n$, by the same argument as used in the proof of Lemma 4.3, $K_i(x)$ for $i=1,\cdots,k$ can be computed in $O((\log k)\cdot M(n))$ operations. A_i and K_i have degree at most $\deg R_i$, so $C_i(x)$ can be computed in $O(M(\deg R_i))$ operations. This implies that $C_1(x),\cdots,C_k(x)$ can be computed in $O(M(n))$ operations. Therefore, we have shown the following

THEOREM 4.2. *Problem* P1 *can be done in*

$$F(n)+O((\log k)\cdot M(n))$$

operations.

We now consider the special case where the $R_i(x)$ is given in the form $(x-z_i)^{m_i}$ for $i=1,\cdots,k$. In this case the A_i satisfying (4.3), i.e.,

$$A_i(x)D_i(x)+E_i(x)(x-z_i)^{m_i}=1,$$

can be computed easily in the following way. Let $\hat{A}_i(x)=A_i(x+z_i)$, $\hat{D}_i(x)=D_i(x+z_i)$, etc. Then

$$\hat{A}_i(x)\hat{D}_i(x)+\hat{E}_i(x)x^{m_i}=1.$$

This implies that

$$\hat{A}_i(x)\hat{D}_i(x)\equiv 1 \pmod{x^{m_i}}. \tag{4.11}$$

Hence we have the following algorithm for computing A_i:

ALGORITHM 4.2.

1. Compute $\hat{D}_i(x)$ such that $\hat{D}_i(x)=D_i(x+z_i)$.
2. Compute $\hat{A}_i(x)$ from (4.10).
3. Compute $A_i(x)$ such that $A_i(x)=\hat{A}_i(x-z_i)$.

Step 1 is equivalent to evaluating D_i and all its derivatives at z_i. Aho, Steiglitz and Ullman (1975) and Vari (1974) have independently shown that this can be done in $O(m_i\log m_i)$ operations. Similarly, step 3 can be done in $O(m_i\log m_i)$ operations. Step 2 involves a division, which is $O(m_i\log m_i)$ by Lemma 2.1. Since $\sum_{i=1}^k m_i\log m_i=O(n\log n)$, by Theorem 4.2 with $M(n)=O(n\log n)$ we have proved the following

THEOREM 4.3. *Problem* P1 *with* $R_i(x)$ *given by* $(x-z_i)^{m_i}$ *for* $i = 1, \cdots, k$ *can be done in*

$$O((\log k)\cdot(n \log n))$$

operations.

Suppose that we solve the general PF problem for $1/\prod_{i=1}^k (x-z_i)^{l_i}$ by solving Problem P1 for $1/\prod_{i=1}^k R_i(x)$ with $R_i(x) = (x-z_i)^{l_i}$. Then we need not perform step 3 of Algorithm 4.2 since the solution of the general PF problem is given by the coefficients of the $\hat{A}_i$. It turns out that this is exactly Chin's $O((\log k)\cdot(n \log n))$ algorithm for solving the general PF problem for $1/\prod_{i=1}^k (x-z_i)^{l_i}$. A similar observation can also be made for the case of solving the general PF problem for $P/\prod_{i=1}^k (x-z_i)^{l_i}$ with $P(x) \not\equiv 1$.

5. An algorithm for Problem P2. Note that using division, we have

$$\begin{aligned}\frac{P}{Q^l} &= \frac{1}{Q^{\lceil l/2\rceil}}\cdot\frac{P}{Q^{\lfloor l/2\rfloor}}\\ &= \frac{1}{Q^{\lceil l/2\rceil}}\cdot\left(P_1+\frac{P_2}{Q^{\lfloor l/2\rfloor}}\right)\\ &= \frac{P_1}{Q^{\lceil l/2\rceil}}+\frac{1}{Q^{\lceil l/2\rceil}}\cdot\left(\frac{P_2}{Q^{\lfloor l/2\rfloor}}\right).\end{aligned}$$

where $\deg P_1 < \lceil l/2\rceil \cdot \deg Q$ and $\deg P_2 < \lfloor l/2\rfloor \cdot \deg Q$. Thus, to solve Problem P2 for the fraction P/Q^l, it suffices to do the following:

1. Divide P by $Q^{\lfloor l/2\rfloor}$ and obtain the quotient P_1 and the remainder P_2.
2. Solve Problem P2 for the fractions $P_1/Q^{\lceil l/2\rceil}$ and $P_2/Q^{\lfloor l/2\rfloor}$.

This gives us a recursive prodedure for solving Problem P2. Assume that the expansion of the power such as $Q^{\lceil l/2\rceil}(x)$ and $Q^{\lfloor l/2\rfloor}(x)$ required by the recursive procedure have been precomputed. Let $X(l)$ be the number of operations needed to solve Problem P2. Then the recursive procedure gives

$$X(l) \leqq X(\lceil l/2\rceil)+X(\lfloor l/2\rfloor)+D(l\cdot\deg Q)$$

for $l > 1$ and $X(1) = 0$. Note that

$$\frac{1}{2} \leqq \frac{\lceil l/2\rceil}{l} \leqq \frac{2}{3}$$

for any integer $l \geqq 2$, and that by Lemma 2.1, $D(l\cdot\deg Q) = O(M(l\cdot\deg Q))$. We have

$$X(l) \leqq X(\alpha l)+X((1-\alpha)l)+O(M(l\cdot\deg Q))$$

where α is a variable with its values in $[1/2, 2/3]$. The expansion of the recurrence corresponds to a binary tree

$$\begin{array}{ccc} & O(M(l\cdot\deg Q)) & \\ \diagup & & \diagdown \\ O(M(\alpha l\cdot\deg Q)) & & O(M((1-\alpha)l\cdot\deg Q)) \\ \diagup \quad \diagdown & & \diagup \quad \diagdown \\ \cdots \qquad \cdots & & \cdots \qquad \cdots \end{array}$$

such that $X(l)$ is bounded above by the total value of the nodes inside the tree. Using the fact that M satisfies Condition C, one can easily show that the sum of the values of the nodes at each level is $O(M(l \cdot \deg Q))$. Since $\alpha \in [1/2, 2/3]$, the height of the tree is at most $\lceil \log_{3/2} l \rceil$. Hence

$$X(l) = O((\log l) \cdot M(l \cdot \deg Q)).$$

Now we examine how to compute all the required powers of Q. This can be done by using a recursion based on

$$Q^l = Q^{\lceil l/2 \rceil} \cdot Q^{\lfloor l/2 \rfloor}.$$

The number of operations needed here clearly satisfies the same recurrence as X, and hence is $O((\log l) \cdot M(l \cdot \deg Q))$. We have proved the following

THEOREM 5.1. *Problem* P2 *can be done in*

$$O((\log l) \cdot M(l \cdot \deg Q))$$

operations.

6. A special case for problem P2. The following theorem can be found in Chin and Ullman (1975).

THEOREM 6.1. *Problem* P2 *can be solved in* $O(l \log l)$ *operations if* $\deg Q = 1$.

In this section we extend the theorem to the case that $\deg Q = 2$. Our result is of interest when the underlying field K is the field of real numbers, for in this case irreducible factors can have either degree one or two. We may assume that Q is monic, since this will affect only $O(l)$ operations. Let

$$Q(x) = x^2 + ax + b.$$

By completing the square and letting $y = x + a/2$ and $c = b - a^2/4$, we have

$$\frac{P(x)}{Q^l(x)} = \frac{P(y - a/2)}{(y^2 + c)^l}.$$

Write

$$\begin{aligned} P\left(y - \frac{a}{2}\right) &= \sum_{i=0}^{2l-1} p_i y^i \\ &= (p_0 + p_2 y^2 + \cdots p_{2l-2} y^{2l-2}) \\ &\quad + y(p_1 + p_3 y^2 + \cdots + p_{2l-1} y^{2l-2}) \\ &= P_1(y^2) + y \cdot P_2(y^2), \end{aligned}$$

where $\deg P_1 \leqq l - 1$ and $\deg P_2 \leqq l - 1$. Then

$$\frac{P(x)}{Q^l(x)} = \frac{P_1(y^2)}{(y^2 + c)^l} + y \cdot \frac{P_2(y^2)}{(y^2 + c)^l}.$$

Hence we can solve Problem P2 for $P(x)/Q^l(x)$ by performing the following steps:

1. Compute $p_0, \cdots, p_{2l-1}$.
2. Form $P_1(z) = p_0 + p_2 z + \cdots + p_{2l-2} z^{l-1}$ and $P_2(z) = p_1 + p_3 z + \cdots + p_{2l-1} z^{l-1}$.

Solve Problem P2 for the fractions $P_1(z)/(z+c)^l$ and $P_2(z)/(z+c)^l$, and obtain

$$\frac{P_1(z)}{(z+c)^l}=\sum_{i=1}^{l}\frac{f_i}{(z+c)^i},\qquad \frac{P_2(z)}{(z+c)^l}=\sum_{i=1}^{l}\frac{e_i}{(z+c)^i}.$$

3. Since

$$\begin{aligned}\frac{P(x)}{Q^l(x)}&=\sum_{i=1}^{l}\frac{f_i}{Q^i(x)}+y\cdot\sum_{i=1}^{l}\frac{e_i}{Q^i(x)}\\&=\sum_{i=1}^{l}\frac{f_i}{Q^i(x)}+\left(x+\frac{a}{2}\right)\cdot\sum_{i=1}^{l}\frac{e_i}{Q^i(x)}\\&=\sum_{i=1}^{l}\frac{e_ix+f_i+ae_i/2}{Q^i(x)},\end{aligned}$$

we set $C_i(x)\leftarrow e_ix+f_i+\dfrac{ae_i}{2}$ for $i=1,\cdots,l$.

By the result of Aho, Steiglitz and Ullman (1975) and Vari (1974) step 1 can be done in $O(l\log l)$ operations. By Theorem 6.1, step 2 can be done in $O(l\log l)$ operations. Step 3 clearly uses $O(l)$ operations. Thus, we have shown the following theorem.

THEOREM 6.2. *Problem* P2 *can be solved in* $O(l\log l)$ *operations if* $\deg Q=2$.

It is an open problem whether Theorem 6.2 holds if $\deg Q>2$.

REFERENCES

A. V. AHO, J. E. HOPCROFT AND J. D. ULLMAN (1974), *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA.

A. V. AHO, K. STEIGLITZ AND J. D. ULLMAN (1975), *Evaluating polynomials at fixed sets of points*, this Journal, 4, pp. 533–539.

A. BORODIN AND I. MUNRO (1975), *The Computational Complexity of Algebraic and Numerical Problems*, American Elsevier, New York.

R. P. BRENT (1976), *Multiple-precision zero-finding methods and complexity of elementary function evaluation*, Analytic Computational Complexity, J. F. Traub, ed., Academic Press, New York, pp. 151–176.

R. P. BRENT AND H. T. KUNG (1976), *Fast algorithms for manipulating formal power series*, Rep., Computer Sci. Dept. Carnegie-Mellon Univ., Pittsburgh, PA.

O. BRUGIA (1965), *Noniterative method for the partial expansion of a rational function with high order poles*, SIAM Rev. 7, pp. 381–387.

F. Y. CHIN (1975), *Complexity of numerical algorithms for polynomials*, Ph.D. thesis, Dept. of Electrical Engineering, Princeton Univ., Princeton, NJ, October 1975.

F. Y. CHIN AND J. D. ULLMAN (1975), *Asymptotic complexity of partial fraction expansion*, Rep., Computer Sci. Laboratory, Dept. of Electrical Engineering, Princeton Univ., Princeton, NJ.

R. J. FATEMAN (1974), *Polynomial multiplication, powers and asymptotic analysis: Some comments*, this Journal, 3, pp. 196–213.

A. A. GRAU (1971), *The simultaneous Newton improvement of a complete set of approximate factors of a polynomial*, SIAM J. Numer. Anal. 8, pp. 425-438.

D. HAZONY AND J. RILEY (1959), *Evaluating residues and coefficients of high order poles*, IRE Trans. Automatic Control, AC-4, pp. 132–136.

P. HENRICI (1971), *An algorithm for the incomplete decomposition of a rational function into partial fraction*, Z. Angew. Math. Phys., 22, pp. 751–755.

——— (1974), *Applied and Computational Complex Analysis*, vol. 1, Wiley-Interscience, New York.

S. KARNI (1969), *Easy partial fraction expansion with multiple poles*, Proc. IEEE (letters), 57, pp. 231–232.

D. E. KNUTH (1969), *The art of Computer Programming*, vol. 2, Addison-Wesley, Reading, MA.

H. T. KUNG (1974), *On computing reciprocals of power series*, Numer. Math., 22, pp. 341–348.

L. J. P. LINNÉR (1974), *The computation of the kth derivative of polynomials and rational functions in factored form and related matters*, IEEE Trans. Circuits and Systems, CAS-21, pp. 233–236.

M. F. MOAD (1966), *On rational function expansion*, Proc. IEEE (letters), 54, pp. 899–900.

R. T. MOENCK (1973a), *Fast computation of GCDs*, Proc. 5th Annual ACM Symposium on Theory of Computing, May 1973, pp. 142–151.

——— (1973b), *Studies in fast algebraic algorithms*, Ph.D. thesis, Dept. of Computer Science, Univ. of Toronto.

D. W. PESSEN (1965), *Time-saving method for partial fraction expansion of function with one pair of conjugate complex roots*, Proc. IEEE (correspondence), 53, p. 1266.

C. POTTLE (1964), *On the partial fraction expansion of a rational function with multiple poles by digit computer*, IEEE Trans. Circuit Theory (correspondence), CT-11, pp. 161–162.

A. SCHÖNHAGE (1971), *Schnelle Berechnung von Kettenbruchentwicklugen*, Acta Informat., 1, pp. 139–144.

I. J. SCHWATT (1924), *An Introduction to Operations with Series*, Chelsea Publishing Co, New York.

V. STRASSEN (1969), *Gaussian elimination is not optimal*, Numer. Math., 13, pp. 354–356.

H. W. TURNBULL (1927), *Note on partial fractions and determinants*, Proc. Edinburgh Math. Soc. 1, no. 2, pp. 49–54.

C. W. VALENTINE (1967). *A method for partial fraction decomposition*, SIAM Rev., 9, pp. 232–233.

B. L. VAN DER WAERDEN (1953), *Modern Algebra*, vol. 1, Frederick Ungar, New York.

T. M. VARI (1974), *Some complexity results for a class of Toeplitz matrices*, Rep., Dept. of Computer Sci. and Math., York Univ., Toronto.

L. WEINBERG (1962), *Network Analysis and Synthesis*, McGraw-Hill, New York.

E. WEHRHAHN (1967), *On partial fraction expansion with high-order poles*, IEEE Trans. Circuit Theory (correspondence), CT-14, pp. 346–347.