Algorithms for computing primary ideal decomposition without producing intermediate redundant components

Taro Kawazoe[1]

Department of Mathematics
Graduate school of Science, Kobe University
1-1 Rokkodai, Nada-ku, 657-8501, Kobe, Japan

Masayuki Noro

Department of Mathematics
Graduate school of Science, Kobe University
1-1 Rokkodai, Nada-ku, 657-8501, Kobe, Japan
JST CREST

Abstract

In Noro (2010) we proposed an algorithm for computing primary ideal decomposition by using the notion of separating ideal and showed that it can efficiently decompose several examples which are hard to decompose by existing algorithms. In particular the number of redundant components produced in the algorithm is zero or very small in many examples, but no theoretical explanation for the efficiency was made.

In this paper we define a more sophisticated class of separating ideals: *saturated separating ideal*. By using this notion we modify the algorithm in Noro (2010) so that it directly outputs a minimal primary decomposition without producing any intermediate redundant component.

By modifying the process of extraction of a primary component via pseudo-primary decomposition proposed in Shimoyama, Yokoyama (1996), we find a method for intermediate decomposition of an ideal and propose a variant of the new primary decomposition algorithm based on this intermediate decomposition. Our experiment shows that this variant efficiently decomposes many examples which are still hard to decompose even if we apply the original version of the new algorithm. Furthermore, in this algorithm we can bypass the computation of primary components and obtain directly the set of all associated primes of an ideal.

Key words: primary ideal decomposition, ideal quotient, Gröbner basis computation

Email addresses: `tkawazoe@math.kobe-u.ac.jp` (Taro Kawazoe), `noro@math.kobe-u.ac.jp` (Masayuki Noro).

[1] The current address is: SOFTBANK MOBILE Corp., 1-9-1 Higashi-shimbashi, Minato-ku, Tokyo 105-7317, `taro.kawazoe@mb.softbank.co.jp`.

1. Introduction

There are two well-known algorithms for computing primary ideal decomposition based on zero-dimensional decomposition: the GTZ algorithm (Gianni et al., 1988) and the SY algorithm (Shimoyama, Yokoyama, 1996). Let I be an ideal in a polynomial ring $k[X] = k[x_1, \ldots, x_n]$ over a field k. The GTZ algorithm extracts some of primary components $Q_1, \ldots, Q_k$ of I via a reduction to a zero-dimensional case. As a by-product of this operation, one obtains an element $f^s \notin I$ such that

$$I = (I : f^s) \cap (I + \langle f^s \rangle), \quad I : f^s = I : f^\infty = Q_1 \cap \cdots \cap Q_k. \tag{1}$$

Then this procedure is applied to $I + \langle f^s \rangle$ to obtain a primary decomposition of I. The SY algorithm first computes the set of all minimal associated primes $\{P_1, \ldots, P_l\}$ of I. By using them, ideals $\tilde{Q}_1 \ldots, \tilde{Q}_l$ and elements $f_1, \ldots, f_l$ satisfying $\sqrt{\tilde{Q}_i} = P_i$ and

$$I = (\tilde{Q}_1 \cap \cdots \cap \tilde{Q}_l) \cap (I + \langle f_1, \ldots, f_l \rangle), \quad \dim(I + \langle f_1, \ldots, f_l \rangle) < \dim I \tag{2}$$

are computed. Each $\tilde{Q}_i$ contains only one isolated primary component Q_i of I and we can compute an ideal I' such that $\tilde{Q}_i = Q_i \cap I'$ and $\dim I' < \dim I$. Then this procedure is applied to I' and $I'' = I + \langle f_1, \ldots, f_l \rangle$ to obtain a primary decomposition of I. Each $\tilde{Q}_i$ is called a *pseudo primary component* and (2) is called a *pseudo primary decomposition* of I.

These algorithms have the following common structure: Let Q be the intersection of known primary components of I. Then these algorithms find an ideal J satisfying $I = Q \cap J$. J is called a remaining ideal. In general a remaining ideal contains components which do not appear in the final minimal decomposition of I. Although these components are removed after or during the decomposition procedure, there are cases that the number of these useless components is very large. SY contains a mechanism for detecting a redundant component soon after it is produced and SY works efficiently for a wide range of input ideals. However there are cases that SY produces an intermediate component which is very hard to decompose because of redundant components included in the intermediate component.

In order to efficiently decompose such examples we proposed the following algorithm in Noro (2010).

Algorithm 1.

```
Input : an ideal $I_{in} \subset R$
Output : a minimal primary decomposition of $I_{in}$
$QL_{in} \leftarrow \emptyset$; $Q_{in} \leftarrow R$; $I_t \leftarrow I_{in}$
RESTART: $Q \leftarrow R$; $I \leftarrow I_t$; $C = \{0\}$
do
    if $I_t = R$ goto LAST
    $PL_t \leftarrow MinimalAssociatedPrimes(I_t)$
    $QL_t \leftarrow IsolatedPrimaryComponents(I_t, PL_t)$
    $Q_t \leftarrow \bigcap_{J \in QL_t} J$
    if $Q \subset Q_t$ goto RESTART else $Q \leftarrow Q \cap Q_t$
    if $Q_{in} \not\subset Q_t$ then $\{Q_{in} \leftarrow Q_{in} \cap Q_t;\ QL_{in} \leftarrow QL_{in} \cup QL_t\ \}$
    if $Q_t = I_t$ or $Q = I$ or $Q_{in} = I_{in}$ goto LAST else $C_t \leftarrow I : Q$
    if $C_t = C$ goto RESTART else $C \leftarrow C_t$
    $I_t \leftarrow I + SeparatingIdeal(I, Q, C)$
end do
LAST: $QL_{in} \leftarrow RemoveRedundancy(QL_{in})$
return $QL_{in}$
```

In this algorithm, $MinimalAssociatedPrimes(I)$ returns the set of all minimal associated primes of an ideal I. $IsolatedPrimaryComponents(I, PL)$ ($PL = \{P_1, \dots, P_k\}$) computes the set of all isolated primary components $\{Q_1, \dots, Q_k\}$ of an ideal I, where PL is the set of all minimal associated primes of I and P_i is the associated prime of Q_i. $SeparatingIdeal(I, Q, C)$ ($C = I : Q$) finds a separating ideal J for (I, Q), that is an ideal J which gives a non trivial decomposition $I = Q \cap (I + J)$. Finally $RemoveRedundancy(QL)$ combines primary components with the same associated prime and removes unnecessary components. Compared with GTZ and SY, Algorithm 1 differs in the following points:

- While GTZ and SY simply try to decompose remaining ideals, Algorithm 1 keeps the target ideal I as long as possible and I_t is used only for extracting its isolated primary components.
- Algorithm 1 constructs a "large" separating ideal J to make I_t large. In GTZ and SY, a remaining ideal is also constructed by adding a generator set to I. However the set are chosen so as to satisfy (1) or (2) and the "size" of the set has not been considered.

In Noro (2010) we showed that a careful selection of separating ideals makes the algorithm very efficient. In particular the number of redundant components produced in Algorithm 1 is zero or very small in many examples, which is realized by large separating ideals. However Algorithm 1 is still unsatisfactory because it often produces completely redundant set of primary components and we have to restart the computation in such a case. In this case the target ideal I is replaced by the current remaining ideal I_t, which tends to incorporate redundant components. Also there is no criterion of the size of a separating ideal. Here we regard a separating ideal sufficiently large if it does not produce any redundant component. In this paper we give a clear criterion of the size of a separating ideal and propose an algorithm for computing a minimal primary decomposition without producing any intermediate redundant components. In section 2 we define a more sophisticated class of separating ideals: *saturated separating ideal*. By using this notion we can modify Algorithm 1 so that it produces no redundant components and the obtained primary decomposition is a minimal primary decomposition. This explains the reason why the number of redundant components is small in Algorithm 1. That is, if a separating ideal is close to a saturated separating ideal, then we can expect that the number of redundant components is small. But the obtained algorithm (Algorithm 3) is not necessarily efficient because the computation of saturated separating ideals and isolated primary components are often very hard. In section 3 we propose an intermediate decomposition and apply it to reduce the costs of these computations. Let Q_{i-1} be the intersection of all primary components of an ideal I found before the i-th step in Algorithm 3. Then the algorithm finds the primary components $Q_{i1}, \dots, Q_{in_i}$ of I such that $\{\sqrt{Q_{i1}}, \dots, \sqrt{Q_{in_i}}\}$ coincides with the set of all prime components of $I : Q_{i-1}$. By analyzing the process of extraction of each Q_{ij} via pseudo-primary decomposition proposed in Shimoyama, Yokoyama (1996), we find that Q_i and $Q_{i-1} \cap Q_{ij}$ can be computed by applying the same extraction process without computing Q_{ij} itself. Then Q_{ij} can be computed as a component of $Q_{i-1} \cap Q_{ij}$ and we obtain Algorithm 5, which is a variant of Algorithm 3. At the same time we also obtain Algorithm 6 for computing the set of all associated primes of an ideal without computing primary decomposition because they can be computed if we know Q_i's. In section 4 we give some remarks on implementation. In section 5 experimental results are shown.

2. An algorithm for computing a minimal primary decomposition

In this section we modify Algorithm 1 so that it directly produces a minimal primary decomposition of an ideal in a Noetherian ring R. We assume that we have an algorithm $MinimalAssociatedPrimes(I)$ for computing the set of all minimal associated primes of I.

2.1. Saturated separating ideal

Definition 1. Let I, Q be ideals in R satisfying $I \subset Q$. An ideal J is called *a separating ideal* for (I, Q) if $I = Q \cap (I + J)$ holds. If a separating ideal for (I, Q) satisfies $\sqrt{I : Q} = \sqrt{I + J}$ then J is called *a saturated separating ideal* for (I, Q).

If J is a separating ideal for (I, Q) then $J \subset I + J \subset I : Q$ holds. Therefore a separating ideal J is a saturated separating ideal if and only if $\sqrt{I : Q} \subset \sqrt{I + J}$.

Example 2. There exists an integer m satisfying $I = Q \cap (I + (I : Q)^m)$ (c.f. Noro (2010)). For such m $(I : Q)^m$ is a saturated separating ideal. For the same m, $J = \langle f_1^m, \ldots, f_l^m \rangle \subset (I : Q)^m$ is also a saturated separating ideal, where $S = \{f_1, \ldots, f_l\}$ is any generating set of $I : Q$. However, from the viewpoint of efficiency it is desirable to find a saturated separating ideal $\langle f_1^{m_1}, \ldots, f_l^{m_l} \rangle$ with each m_i as small as possible.

We can remove elements in $\sqrt{I}$ from S to construct a saturated separating ideal of the above type.

Proposition 3. Suppose that $\sqrt{I : Q} = \sqrt{\langle S \rangle}$. If a separating ideal J for (I, Q) satisfies $(S \setminus \sqrt{I}) \subset \sqrt{J}$ then J is a saturated separating ideal for (I, Q).

Proof. $(S \setminus \sqrt{I}) \subset \sqrt{J}$ implies $S \subset \sqrt{I} + \sqrt{J}$ and thus $\sqrt{I : Q} = \sqrt{\langle S \rangle} \subset \sqrt{\sqrt{I} + \sqrt{J}} = \sqrt{I + J}$. □

The following theorem enables us to construct such a saturated separating ideal incrementally.

Theorem 4. Let J be a separating ideal for (I, Q). If $f \in \sqrt{I : Q}$ then there exists a positive integer m satisfying $I = Q \cap (I + J + \langle f^m \rangle)$.

Proof. $f \in \sqrt{I : Q}$ implies that there exists $k > 0$ such that $\langle f^k \rangle Q \subset I$. By Artin-Rees lemma there exists an integer c such that $\langle f^m \rangle \cap (I+J+Q) = \langle f^{m-c} \rangle (\langle f^c \rangle \cap (I+J+Q))$ for any integer $m > c$. We show that $I = Q \cap (I+J+\langle f^m \rangle)$ if $m > c+k$. If $m > c+k$ we have $\langle f^m \rangle \cap (I+J+Q) = \langle f^{m-c} \rangle (\langle f^c \rangle \cap (I+J+Q)) \subset \langle f^k \rangle (I+J+Q) \subset I+J+\langle f^k \rangle Q \subset I+J$. If $q \in Q \cap (I + J + \langle f^m \rangle)$, then there exist $a \in R$ and $j \in I + J$ satisfying $q = j + af^m$. Then $af^m = -j + q \in I + J + Q$ and $af^m \in \langle f^m \rangle \cap (I + J + Q) \subset I + J$. Thus $q = j + af^m \in I + J$ and we have $q \in Q \cap (I + J) = I$. □

Corollary 5. If $I = Q \cap (I+J)$ and $\sqrt{I+J} \neq \sqrt{I:Q}$ then $(I:Q) \setminus \sqrt{I+J} \neq \emptyset$. For any $f \in (I:Q) \setminus \sqrt{I+J}$, there exists a positive integer m satisfying $I = Q \cap (I + J + \langle f^m \rangle)$ and $I + J + \langle f^m \rangle \neq I + J$.

The following algorithm computes a saturated separating ideal incrementally.

Algorithm 2. *SaturatedSeparatingIdeal*(I, Q, C)

```
Input : ideals $I, Q, C \subset R$ satisfying $I \subset Q$ and $\sqrt{C} = \sqrt{I:Q}$
Output : a saturated separating ideal for $(I, Q)$
$S \leftarrow$ a generating set of $C$
$S_0 = \{f_1, \ldots, f_l\} \leftarrow S \setminus \sqrt{I}$
$J = \{0\}$
for $i = 1$ to $l$ do
    $j \leftarrow 0$
    do $j \leftarrow j+1$ while $Q \cap (I + J + \langle f_i^j \rangle) \neq I$
    $J \leftarrow J + \langle f_i^j \rangle$
end for
return $J$
```

Proposition 6. Algorithm 2 outputs a saturated separating ideal for (I, Q).

Proof. $Q \cap (I+J) = I$ and $f_i \in \sqrt{I:Q}$ imply that there exists an integer j satisfying $Q \cap (I + J + \langle f_i^j \rangle) = I$ by Theorem 4. The output J satisfies the condition of Proposition 3 and it is a saturated separating ideal for (I, Q). □

Theorem 7. Suppose that $I = Q \cap J$ and $\sqrt{J} = \sqrt{I:Q}$ for a proper ideal J. Let $Q_1, \ldots, Q_r$ be the set of all isolated primary components of J and we set $Q' = Q \cap \bigcap_{i=1}^{r} Q_i$. If $I = Q' \cap J'$ and $\sqrt{J'} = \sqrt{I:Q'}$ for a proper ideal J', then any minimal associated prime of J' is a non-minimal associated prime of J. In particular any minimal associated prime of J' properly contains a minimal associated prime of J.

Proof. In general if T is primary then $U \not\subset T$ implies $\sqrt{T:U} = \sqrt{T}$ and $U \subset T$ implies $T:U = \langle 1 \rangle$. Let

$$J = \bigcap_{i=1}^{r} Q_i \cap \bigcap_{i=1}^{s} S_i$$

be a minimal primary decomposition of J. S_i is an embedded primary component of J. Then

$$I : Q' = \bigcap_{i=1}^{s} (S_i : Q') = \bigcap_{Q' \not\subset S_i} (S_i : Q')$$

implies $\sqrt{J'} = \sqrt{I:Q'} = \bigcap_{Q' \not\subset S_i} \sqrt{S_i}$. The minimal prime decomposition of $\sqrt{J'}$ is obtained by removing redundant components from $\{\sqrt{S_i} \mid Q' \not\subset S_i\}$. Therefore all minimal associated primes of J' appear in $\sqrt{S_1}, \ldots, \sqrt{S_s}$. Since S_i is an embedded primary component of J, there exists a minimal associated prime of J which is properly contained in $\sqrt{S_i}$. □

Corollary 8. For J and J' in Theorem 7, $\sqrt{J'}$ properly contains $\sqrt{J}$.

Proof. Each prime component of $\sqrt{J'}$ properly contains a prime component of $\sqrt{J}$. Thus $\sqrt{J} \subset \sqrt{J'}$ holds but $\sqrt{J} = \sqrt{J'}$ cannot hold because of the uniqueness of prime components of a radical ideal. □

2.2. An algorithm for computing a minimal primary decomposition

We assume that we have an algorithm $IsolatedPrimaryComponents(I, PL)$ for computing the set of all isolated primary components of an ideal I from PL, the set of all minimal associated primes of I. By using the notion of saturated separating ideal, we propose an algorithm for computing primary ideal decomposition.

Algorithm 3. *SYC_PrimaryDecomposition*(I) [†]

```
Input : an ideal $I \subset R$
Output : a list of sets of primary components of $I$
$QL \leftarrow \emptyset$; $Q_0 \leftarrow R$; $I_1 \leftarrow I$; $C_i \leftarrow I$; $i \leftarrow 1$
do
    $PL_i \leftarrow MinimalAssociatedPrimes(C_i)$
    $QL_i \leftarrow IsolatedPrimaryComponents(I_i, PL_i)$
    $Q_i \leftarrow Q_{i-1} \cap \bigcap_{J \in QL_i} J$
    If $Q_i = I$ then return $(QL_1, \ldots, QL_i)$
    $C_{i+1} \leftarrow I : Q_i$
    $J_{i+1} \leftarrow SaturatedSeparatingIdeal(I, Q_i, C_{i+1})$
    $I_{i+1} \leftarrow I + J_{i+1}$
    $i \leftarrow i + 1$
end do
```

Remark 9. In Algorithm 3, we can take any ideal C_{i+1} such that $\sqrt{C_{i+1}} = \sqrt{I : Q_i}$ instead of $I : Q_i$. See Section 4.2 for details.

Theorem 10. (1) Algorithm 3 terminates.

(2) In Algorithm 3, all primary ideals in QL_i's are distinct and $\bigcup_i QL_i$ gives a minimal primary decomposition of I.

Proof. (1) If $i = 1$ then $\sqrt{C_1} = \sqrt{I_1}$. If $i \geq 2$ then J_i is a saturated separating ideal for (I, Q_{i-1}) and $\sqrt{C_i} = \sqrt{I : Q_{i-1}} = \sqrt{I_i}$ holds. Therefore PL_i consists of all minimal associated primes of I_i and is valid as the argument of $IsolatedPrimaryComponents$. We also have $I = Q_{i-1} \cap I_i = Q_i \cap I_{i+1}$ and $Q_i = Q_{i-1} \cap \bigcap_{J \in QL_i} J$. By Corollary 8, $\sqrt{I_{i+1}}$ properly contains $\sqrt{I_i}$. Therefore the algorithm terminates because R is a Noetherian ring.

[†] *SYC* stands for Shimoyama-Yokoyama with Colon ideal.

(2) Set $QL_i = \{Q_{i1}, \ldots, Q_{in_i}\}$. If the algorithm terminates in t steps, then we have $I = \bigcap_{i=1}^{t} \bigcap_{j=1}^{n_i} Q_{ij}$. We first show $P_{ij} = \sqrt{Q_{ij}}$ are all distinct. Suppose $P_{ij} = P_{kl}$. We may assume $i \le k$. If $i < k$ then there exists a strictly decreasing sequence

$$P_{kl} = P_{kj_k} \supset P_{k-1,j_{k-1}} \supset \cdots \supset P_{ij_i}$$

starting from P_{kl} by Theorem 7. Then P_{ij_i} is a proper subset of P_{ij}, which cannot happen because $\sqrt{I_i} = P_{i1} \cap \cdots \cap P_{in_i}$ is the minimal prime decomposition of $\sqrt{I_i}$. Thus $i = k$ and we have $j = l$ by the same reason. Suppose that Q_{kl} is redundant, then $I = \bigcap_{(i,j) \neq (k,l)} Q_{ij}$. In the algorithm $Q_{k-1} = \bigcap_{i=1}^{k-1} \bigcap_{j=1}^{n_i} Q_{ij}$, $I = Q_{k-1} \cap I_k$ and $\sqrt{I : Q_{k-1}} = \sqrt{I_k}$ hold. Then we have $I : Q_{k-1} = \bigcap_{i \ge k, (i,j) \neq (k,l)} (Q_{ij} : Q_{k-1})$ and

$$\sqrt{I : Q_{k-1}} = \bigcap_{i \ge k, (i,j) \neq (k,l)} \sqrt{Q_{ij} : Q_{k-1}} = \bigcap_{i \ge k, (i,j) \neq (k,l), Q_{k-1} \not\subset Q_{ij}} P_{ij}.$$

Thus P_{kl} does not appear in the prime decomposition of $\sqrt{I : Q_{k-1}} = \sqrt{I_k}$ because all the P_{ij}'s are distinct. But this contradicts to the fact that $\sqrt{I_k} = P_{k1} \cap \cdots \cap P_{kn_k}$ is the minimal prime decomposition of $\sqrt{I_k}$. $\square$

2.3. Computation of isolated primary components

In this subsection we set $R = k[x_1, \ldots, x_n]$, an n-variate polynomial ring over a field k. According to Shimoyama, Yokoyama (1996) we can compute isolated primary components of an ideal I via pseudo primary decomposition. For $Y \subset X = \{x_1, \ldots, x_n\}$, we set $R_Y = k(Y)[X \setminus Y]$.

Algorithm 4. $IsolatedPrimaryComponents(I, PL)$

```
Input : an ideal $I \subset R$; the set of all minimal associated primes of $I$: $PL = \{P_1, \ldots, P_m\}$
Output : the set of all isolated primary components of $I$
for $j = 1$ to $m$ do
    $f \leftarrow$ an element of $(\bigcap_{l \neq j} P_l) \setminus P_j$
    $Y \leftarrow$ a maximally independent set for $P_j$
    $T_j \leftarrow (I : f^\infty) R_Y \cap R$
end for
return $\{T_1, \ldots, T_m\}$
```

By the following theorem this algorithm computes the set of all isolated primary component of an ideal I.

Theorem 11. Let $\{T_1, \ldots, T_m\}$ be the set of all isolated primary components of an ideal I and $P_j = \sqrt{T_j}$. Let Y_j be a maximally independent set for P_j. If $f_j \in (\bigcap_{l \neq j} P_l) \setminus P_j$ then $T_j = (I : f_j^\infty) R_{Y_j} \cap R$.

Proof. Let $I = \bigcap_{k=1}^{m} T_k \cap \bigcap_{l=1}^{q} S_l$ be a primary decomposition of I such that S_l's are embedded primary components of I. Then we have

$$(I : f_j^{\infty})R_{Y_j} \cap R = \bigcap_{k=1}^{m}((T_k : f_j^{\infty})R_{Y_j} \cap R) \cap \bigcap_{l=1}^{q}((S_l : f_j^{\infty})R_{Y_j} \cap R).$$

Since $f_j \notin P_j$ and $f_j \in P_k$ $(k \neq j)$, $T_j : f_j^{\infty} = T_j$ and $T_k : f_j^{\infty} = R$ $(k \neq j)$ hold. Thus we have $\bigcap_{k=1}^{m}((T_k : f_j^{\infty})R_{Y_j} \cap R) = T_j$. For each S_l there exists T_{k_l} such that P_{k_l} is proper subset of $\sqrt{S_l}$. If $k_l \neq j$ then $f_j \in P_{k_l} \subset \sqrt{S_l}$ and $S_l : f_j^{\infty} = R$. If $k_l = j$ then $\sqrt{S_l}$ properly contains P_j and Y_j cannot be an independent set of S_l. Thus $k[Y_j] \cap \sqrt{S_l} \neq \{0\}$ and $(S_l : f_j^{\infty})R_{Y_j} \cap R = R$. Thus we have $\bigcap_{l=1}^{q}((S_l : f_j^{\infty})R_{Y_j} \cap R) = R$. Therefore we have $T_j = (I : f_j^{\infty})R_{Y_j} \cap R$. □

Remark 12. $I : f_j^{\infty}$ in Theorem 11 is called a pseudo primary component of I and it contains only one isolated primary component T_j whose associated prime is P_j.

3. Application of intermediate decomposition to Algorithm 3

By introducing the notion of saturated separating ideal, we have obtained Algorithm 3 which directly outputs a minimal primary decomposition. However, if we execute the algorithm we observe that the cost for computing saturated separating ideals and isolated primary components are often very high. In this section we propose a variant of Algorithm 3 based on an intermediate decomposition. In that variant each saturated separating ideal is computed for extracting only one unknown primary component and it makes both the computation of the saturated separating ideal and the primary component easy.

3.1. An intermediate decomposition of an ideal

Algorithm 3 introduces a decomposition of the set of all associated primes of I: $\mathrm{Ass}(R/I) = PL_1 \cup \cdots \cup PL_t$. We first show that this decomposition has a definite meaning which depends on only I.

Definition 13. Let $A = \mathrm{Ass}(R/I)$ be the set of all associated primes of I. We define $A_i \subset A$ for $i \geq 1$ recursively:

$$A_i = \{P \in A \mid P \text{ is minimal with respect to the inclusion relation in } A \setminus \bigcup_{k=1}^{i-1} A_k\}.$$

An element of A_i is called an associated prime of level i.

Proposition 14. Let $I = \bigcap_{k=1}^{t}\bigcap_{l=1}^{n_k} Q_{kl}$ be a minimal primary decomposition such that $\{P_{i1}, \ldots, P_{in_i}\}$ coincides with A_i, the set of all associated primes of level i for $i = 1, \ldots, t$ and $P_{kl} = \sqrt{Q_{kl}}$. Set $Q_i = \bigcap_{k=1}^{i}\bigcap_{l=1}^{n_k} Q_{kl}$ and $S_i = R \setminus \bigcup_{k \leq i, P \in A_k} P$. Then $Q_i = IR_{S_i} \cap R$.

Proof. Although this proposition is a corollary of Lemma 2.19 in Shimoyama, Yokoyama (1996), we show a proof for convenience. From $I = \bigcap_{k=1}^{t} \bigcap_{l=1}^{n_k} Q_{kl}$ we have

$$IR_{S_i} \cap R = \bigcap_{k=1}^{t} \bigcap_{l=1}^{n_k} (Q_{kl} R_{S_i} \cap R).$$

If $k \le i$ then $P_{kl} \cap S_i = \emptyset$ and $Q_{kl} R_{S_i} \cap R = Q_{kl}$. Suppose $k > i$. We show $P_{kl} \cap S_i \ne \emptyset$. If $P_{kl} \cap S_i = \emptyset$ then $P_{kl} \subset \bigcup_{a \le i, P \in A_a} P$. By the prime avoidance $P_{kl} \subset P_{ab}$ for some $a \le i$ and b. Then P_{kl} is a proper subset of P_{ab} because $a \le i < k$. But this is a contradiction because P_{ab} is minimal in $A \setminus \bigcup_{i<a} A_i = \bigcup_{a \le i} A_i$ and $P_{kl} \in \bigcup_{a \le i} A_i$. Thus if $k > i$ then $P_{kl} \cap S_i \ne \emptyset$ and $Q_{kl} R_{S_i} \cap R = R$. Therefore

$$IR_{S_i} \cap R = \bigcap_{k=1}^{i} \bigcap_{l=1}^{n_k} Q_{kl} = Q_i. \qquad \square$$

Corollary 15. Let $I = \bigcap_k T_k$ be a minimal primary decomposition of an ideal I and Q_i the intersection of all primary components T_k such that the level of $\sqrt{T_k}$ is not greater than i. Then Q_i is independent of a minimal primary decomposition.

Theorem 16. PL_i in Algorithm 3 coincides with A_i. In particular PL_i is independent of saturated separating ideals.

Proof. If $i = 1$, then PL_1 is the set of all minimal associated primes of I and $PL_1 = A_1$. Assume $PL_k = A_k$ for all $k \le i - 1$. If $P \in A_i$ then P is minimal in $A \setminus \bigcup_{k=1}^{i-1} A_k = A \setminus \bigcup_{k=1}^{i-1} PL_k$. If $P \in PL_k$ for some $k > i$ then there exists $P' \in PL_i$ which is properly contained in P. Then $P' \notin \bigcup_{k=1}^{i-1} PL_k$ implies $P, P' \in A \setminus \bigcup_{k=1}^{i-1} PL_k$, which contradicts to the minimality of P. Thus $P \in PL_i$. Conversely suppose $P \in PL_i$. If $P \notin A_i$ then there exists $P' \in A \setminus \bigcup_{k=1}^{i-1} PL_k$ which is properly contained in P. If $P' \in PL_k$ $(k \ge i)$ then there exists $P'' \in PL_i$ satisfying $P'' \subset P'$. Then $P, P'' \in PL_i$ and P'' is a proper subset of P, which is a contradiction. Thus $P \in A_i$ and we have $PL_i = A_i$. By induction we have $PL_i = A_i$ $(i \ge 1)$. $\square$

Remark 17. Q_i in Algorithm 3 is independent of saturated separating ideals I_i. I_i only affects the shapes of primary components in QL_i.

In the rest of this section we set $R = k[X]$ for a field k and $X = \{x_1, \ldots, x_n\}$.

Theorem 18. Let I be an ideal in R. We retain the notations in Proposition 14. Let $Y_j \subset X$ be a maximally independent set for P_{ij}. Set $S_{i,j} = R \setminus (P_{ij} \cup \bigcup_{k \le i-1, P \in A_k} P)$. If $f_j \in (\bigcap_{l \ne j} P_{il}) \setminus P_{ij}$ then

(1) $Q_{i-1} \cap ((I : f_j^\infty)R_{Y_j} \cap R) = Q_{i-1} \cap Q_{ij}$,

(2) $Q_{i-1} \cap \bigcap_{j=1}^{n_i} ((I : f_j^\infty)R_{Y_j} \cap R) = Q_i$.

(3) $Q_{i-1} \cap Q_{ij} = IR_{S_{i,j}} \cap R$.

Proof. Let $I = \bigcap_{k=1}^{t} \bigcap_{l=1}^{n_k} Q_{kl}$ be a minimal primary decomposition such that $A_i = \{P_{i1}, \ldots, P_{in_i}\}$, $P_{ij} = \sqrt{Q_{ij}}$ for $i = 1, \ldots, t$. Then $Q_{i-1} = \bigcap_{k=1}^{i-1} \bigcap_{l=1}^{n_k} Q_{kl}$, $I = Q_{i-1} \cap \bigcap_{k=i}^{t} \bigcap_{l=1}^{n_k} Q_{kl}$ and $Q_{i1}, \ldots, Q_{in_i}$ are all isolated primary components of $\bigcap_{k=i}^{t} \bigcap_{l=1}^{n_k} Q_{kl}$. Thus

$$(I : f_j^\infty)R_{Y_j} \cap R = ((Q_{i-1} : f_j^\infty)R_{Y_j} \cap R) \cap Q_{ij}$$

by Theorem 11. Since $Q_{i-1} \subset (Q_{i-1} : f_j^\infty)R_{Y_j} \cap R$ we have

$$Q_{i-1} \cap ((I : f_j^\infty)R_{Y_j} \cap R) = Q_{i-1} \cap Q_{ij} \text{ and } Q_{i-1} \cap \bigcap_{j=1}^{n_i} ((I : f_j^\infty)R_{Y_j} \cap R) = Q_i.$$

The proof of (3) is similar to that of Proposition 14. □

3.2. An algorithm for computing primary decomposition via intermediate decomposition

Theorem 18 means that the intermediate components $Q_{i-1} \cap Q_{ij}$ and Q_i can be computed recursively without knowing individual Q_{ij}. Furthermore (3) of Theorem 18 means that $Q_{i-1} \cap Q_{ij}$ does not depend on a particular primary decomposition and is determined only by P_{ij}. Since Q_{ij} can be obtained as a component of $Q_{i-1} \cap Q_{ij}$, we have the following algorithm:

Algorithm 5. *SYCI_PrimaryDecomposition*(I)[‡]

```
Input : an ideal $I \subset R$
Output : a minimal primary decomposition of $I$
$i \leftarrow 1$; $R_0 \leftarrow R$
do
   $PL_i = \{P_{i1}, \ldots, P_{in_i}\} \leftarrow MinimalAssociatedPrimes(I : R_{i-1})$
   for $j = 1$ to $n_i$ do
      $Y_{ij} \leftarrow$ a maximally independent set for $P_{ij}$
      $f_{ij} \leftarrow$ an element in $(\bigcap_{k \ne j} P_{ik}) \setminus P_{ij}$
```

[‡] *SYCI* stands for Shimoyama-Yokoyama with Colon ideal and Intermediate decomposition.

```
        $R_{ij} \leftarrow R_{i-1} \cap ((I : f_{ij}^{\infty})R_{Y_{ij}} \cap R)$
        $C_{ij} \leftarrow$ an ideal satisfying $\sqrt{C_{ij}} = P_{ij}$
        $J_{ij} \leftarrow SaturatedSeparatingIdeal(R_{ij}, R_{i-1}, C_{ij})$
        $T_{ij} \leftarrow (R_{ij} + J_{ij})R_{Y_{ij}} \cap R$
    end for
    $QL_i = \{T_{i1}, \dots, T_{in_i}\}$
    $R_i \leftarrow R_{i1} \cap \cdots \cap R_{in_i}$
    If $R_i = I$ then return $(QL_1, \dots, QL_i)$
    $i \leftarrow i + 1$
end do
```

Theorem 19. Algorithm 5 outputs a minimal primary decomposition of I.

Proof. We fix a minimal primary decomposition of I as in the proof of Theorem 18. Then it is easy to see that $R_i = Q_i$, $R_{ij} = Q_{i-1} \cap Q_{ij}$ and $PL_i = A_i$, the set of all associated primes of level i, for all i, j. If J_{ij} is a saturated separating ideal for (R_{ij}, R_{i-1}) then $\sqrt{R_{ij} + J_{ij}} = \sqrt{Q_{ij} : Q_{i-1}} = P_{ij}$ and $T_{ij} = (R_{ij} + J_{ij})R_{Y_{ij}} \cap R$ is the unique isolated P_{ij}-primary component of $R_{ij} + J_{ij}$. Let $R_{ij} + J_{ij} = T_{ij} \cap \bigcap_{l=1}^{s} U_l$ be a minimal primary decomposition of $R_{ij} + J_{ij}$ such that U_l's are embedded components. Then $R_{ij} = \bigcap_{k=1}^{i-1} \bigcap_{l=1}^{n_k} Q_{kl} \cap T_{ij} \cap \bigcap_{l=1}^{s} U_l$ is a primary decomposition of R_{ij}. Since $\sqrt{U_l}$ properly contains $\sqrt{T_{ij}} = P_{ij}$ and $\mathrm{Ass}(R/R_{ij}) = \bigcup_{k=1}^{i-1} PL_k \cup \{P_{ij}\}$, $\sqrt{U_l}$ cannot be an associated prime of R_{ij}. Thus all U_l's are redundant and we have $R_{ij} = Q_{i-1} \cap T_{ij}$. Thus T_{ij} is valid as the P_{ij}-primary component of I. □

Remark 20. For computing a saturated separating ideal for (R_{ij}, R_{i-1}), we need $\sqrt{R_{ij}}$. Fortunately it coincides with $\sqrt{I}$ and it is sufficient to compute it only once after computing PL_1.

If we only want to know $\mathrm{Ass}(R/I)$ then Algorithm 5 is simplified as follows.

Algorithm 6. *SYCI_AssociatedPrimes*(I)

```
Input : an ideal $I \subset R$
Output : the set of all associated primes of $I$
$i \leftarrow 1$; $R_0 \leftarrow R$
do
    $PL_i = \{P_{i1}, \dots, P_{in_i}\} \leftarrow MinimalAssociatedPrimes(I : R_{i-1})$
    for $j = 1$ to $n_i$ do
        $Y_{ij} \leftarrow$ a maximally independent set for $P_{ij}$
        $f_{ij} \leftarrow$ an element in $(\bigcap_{k \neq j} P_{ik}) \setminus P_{ij}$
        $R_{ij} \leftarrow ((I : f_{ij}^{\infty})R_{Y_{ij}} \cap R)$
    end for
    $R_i \leftarrow R_{i-1} \cap R_{i1} \cap \cdots \cap R_{in_i}$
    If $R_i = I$ then return $(PL_1, \dots, PL_i)$
    $i \leftarrow i + 1$
end do
```

4. Efficient implementation of the new algorithms

In order to realize an efficient implementation of Algorithm 3, Algorithm 5 and Algorithm 6, we need efficient implementation for computing saturated separating ideals, ideal quotients and isolated primary components. In this section we propose several methods for each part. Again we set $R = k[x_1, \dots, x_n]$.

4.1. Computation of saturated separating ideals

If we apply Algorithm 2 for computing a saturated separating ideal, we often observe that the computation of j satisfying $Q \cap (I + J + \langle f_i^j \rangle) = I$ becomes harder as J becomes larger. From our experiments we guess that adding f_i^j with a large j makes the subsequent computation harder and we propose the following variant so that we can find $f_i^{s_i}$ with smaller s_i earlier.

Algorithm 7. *SaturatedSeparatingIdeal2*(I, Q, C)

```
Input : ideals $I, Q, C \subset R$ satisfying $I \subset Q$ and $\sqrt{C} = \sqrt{I : Q}$
Output : a saturated separating ideal for $(I, Q)$
$S \leftarrow$ a generating set of $C$
$S_0 = \{f_1, \dots, f_l\} \leftarrow S \setminus \sqrt{I}$
$J = \{0\}$; $U \leftarrow S_0$; $j \leftarrow 1$
while $U \neq \emptyset$ do
(*)   for each $f_i \in U$ if $Q \cap (I + J + \langle f_i^j \rangle) = I$ then $\{J \leftarrow J + \langle f_i^j \rangle; U \leftarrow U \setminus \{i\}\}$
      $j \leftarrow j + 1$
end while
return $J$
```

Since a large part of U satisfies the condition in $(*)$ as j becomes large, we may apply such a strategy that if $Q \cap (I + J + T) = I$ for a subset $T \subset U^j = \{f_i^j \mid j \in U\}$ then we try to find $T' \subset U^j \setminus T$ with $Q \cap ((I + J + T) + T') = I$ and $|T'| = 2|T|$.

In Algorithm 7, We have to compute many ideal intersections of $Q \cap (J + \langle f \rangle)$-type. Suppose that we have a Gröbner basis G of $tJ + (1 - t)Q$ with respect to an elimination order $<$ such that $\{t\} >> \{x_1, \dots, x_n\}$. We can apply the following methods for improving the efficiency.

- Incremental computation
  For computing $Q \cap (J + \langle f \rangle)$ it is sufficient to compute a Gröbner basis of $\langle G \cup \{tf\} \rangle$ with respect to $<$. Since G is a Gröbner basis, we don't have to consider S-polynomials constructed from G when we execute the Buchberger algorithm for $G \cup \{tf\}$.
- Early termination
  If $Q \cap J = I$ then G contains a Gröbner basis of I. Therefore if an element $g \in R$ is generated during an execution of the Buchberger algorithm to check whether $Q \cap (J + \langle f \rangle)$ or not, then it means that $g \notin I$ and thus $Q \cap (J + \langle f \rangle) \neq I$.

Thus if a function for computing Gröbner basis allows incremental computation and early termination, we can reduce the cost for necessary Gröbner basis computations.

4.2. Computation of ideal quotients

Ideal quotients are used in two ways in Algorithm 3 and Algorithm 5.

(1) We need the set of all minimal associated primes of $I : Q_i$.

(2) We need a generating set of an ideal C satisfying $\sqrt{C} = \sqrt{I : Q}$ to compute a saturated separating ideal.

For (1) we can compute the required set via $\sqrt{I : Q_i} = \bigcap_{j=1}^{k} \sqrt{I; g_j}$, where $\{g_1, \ldots, g_k\}$ is a generating set of Q_i. However, for (2) it is not clear whether we can use a generating set of $\sqrt{I : Q}$ instead of that of $I : Q$ from a practical point of view. If we use the former one, then it is possible that the required exponent m in Theorem 4 is high, which may make the check of $I = Q \cap (I + J + \langle f^m \rangle)$ hard. In general, in Algorithm 3 the cost for computing ideal quotients is not dominant and it will be safe to set $C_{i+1} = I : Q_i$. However, in Algorithm 5 the cost for computing ideal quotients tends to occupy a large part of the whole computation. If we set $C_{ij} = \sqrt{R_{ij} : R_{i-1}} = P_{ij}$, then we can bypass the computation of $R_{ij} : R_{i-1}$. In our experiment the computations become faster by setting $C_{ij} = P_{ij}$ in Algorithm 5. Therefore we set $C_{ij} = P_{ij}$ in our implementation unless an option to set $C_{ij} = R_{ij} : R_{i-1}$ is specified.

4.3. Computation of isolated primary components

In Algorithm 4, $J = I : f^\infty$ and $JR_Y \cap R$ are often hard to compute. The following two remarks may be useful for improving Algorithm 4.

- Change of the order of two localizations
  The order of two localizations can be changeable, that is Q_i can be computed as $Q_i = (IR_Y \cap R) : f^\infty$.
- Computation of $JR_Y \cap R$
  This type of localization can be computed as follows: Let G_i be a Gröbner basis of JR_Y in $k(Y)[X \setminus Y]$. Set $h = \mathrm{LCM}(\text{the square free part of } \mathrm{LC}(g) \mid g \in G_i)$, where $\mathrm{LC}(g)$ is the leading coefficient of g as an element in $k(Y)[X \setminus Y]$. Then $JR_Y \cap R = \langle G_i \rangle : h^\infty$. G_i can be computed in two ways: direct computation in $k(Y)[X \setminus Y]$, or computation in $k[X]$ with respect to an elimination order. It often happens that the efficiency greatly differs between these two methods.

In both cases it is hard to predict which choice is faster than the other. Again in these cases a competitive computation will be useful.

5. Experiments

In this section we will show the performance of the new algorithms in Risa/Asir (Noro et al., 2011) and Singular (Decker et al., 2010). The implementation in Risa/Asir is an improved version of `noro_pd.rr` described in Noro (2010), which is contained in the OpenXM package (OpenXM committers, 2011). The implementation in Singular is an ongoing work and it uses a function for computing minimal associated primes in `primdec.lib`. The file `primdecSYCI.lib` is available from Noro (2011).

In order to measure the performance of Algorithm 3 and Algorithm 5, we need examples which have many embedded primary components and are hard to decompose by SY or GTZ. In addition to the examples used in Noro (2010), we use ideals T_i generated by

monomials, binomials and trinomials. We randomly generate such kind of ideals and use for our experiments. The input ideals are as follows:
$A_{k,m,n}$ denotes the ideal generated by adjacent $k \times k$ minors of $m \times n$ matrix X with indeterminate entries (Diaconis et al., 1998). I_2, I_3 are ideals related to local b-functions introduced in Noro (2010).

$$\begin{aligned}
T_1 = \langle & cdefghiz + cdefhjz + bcdeijz, 3cdfghz^3 + 4bdefghj + 4bdehjz^2, \\
& 2bfghijz + fhjz^3, 4bcefhz + cfgijz, cdjz, 3egjz^4 + bcdgij + 2cdhjz^2, \\
& 3defiz + 2defz^2 + 4bcei, 4bcefiz + 3dfhjz^2, cefhjz + bcfiz^2 + giz^4, \\
& 4ceghiz + bcejz\rangle \subset \mathbb{Q}[b, c, d, e, f, g, h, i, j, z] \\
T_2 = \langle & 3bcegz^2 + 4bcghi + 2bcez^2, bcez + 3dhi, cfgiz^3 + bcdegh, cfgz^4 + 3cdefgh, \\
& 2bcfgiz^2 + bcdegh + z^6, bchz + 4bcg, 4bcdgiz + 2cfhiz^2 + 3bdfhi, bdefhz + bz^4, \\
& 3befgiz + 2cefgz^2 + 4cfhz^2, 3bfh + 4fhi + bz^2\rangle \subset \mathbb{Q}[b, c, d, e, f, g, h, i, z] \\
T_3 = \langle & 4befjkmz^3 + 2bcdhijlm + cdegkmz^2, cdeghjlz, 2defghilz + 4jlz^6 + defjlz^2, \\
& beghjlmz + 4ceghiz^2 + bdeflz^2\rangle \subset \mathbb{Q}[b, c, d, e, f, g, h, i, j, k, l, m, z] \\
T_4 = \langle & 2cfhiz^2 + bdefh, bcfijz + 4bcghi, 2cdejz + 4cdfj + ijz^2, \\
& bcdfgijz + cdijz^3, 3bceijz + 3cgijz^2 + beiz^3, 4bchjz + cgiz^2, behj, \\
& 3cdefhiz + 2bdfgjz + 2bchjz^2\rangle \subset \mathbb{Q}[b, c, d, e, f, g, h, i, j, z] \\
T_5 = \langle & 4bc^2d^2e^2gh^2iz^2 + b^2ciz^9 + 2bceg^2hz^5, bcd^2e^2g^2h^2, bcfhz^5 + b^2dfg^2iz, \\
& 4bc^2e^2f^2h^2i^2z^2 + b^2c^2e^2fh^2iz^3, 2b^2de^2f^2hi^2z + 3b^2c^2e^2h^2i^2\rangle \\
& \subset \mathbb{Q}[b, c, d, e, f, g, h, i, z] \\
T_6 = \langle & 4bcdfghlz + 3bcfhlz^3, befhkl + defghz, 3bdefhijklz + 2cfhjkz^5 + bdehkz^4, \\
& 4befijkl + dgklz^3, bcdefghj + 2bcdegijz + 2bcdhjklz, \\
& cdegijz + 3bcdefk + 4fhklz^2, 2bdeghjkz + cdez^5 + 3eghjz^3, \\
& bcdghijz + cdfhklz + 2bcdhkz^2, 2bcdefi + bhijkl, eghjkz^5 + 2bcefghjkl, \\
& gilz^2 + 2beil, g, 3cdefijkl + 4bcdgjz^3, cdehijz + 4cegjz^3, bchkl, \\
& cdfghklz + befhilz + cdfgjlz, fiz^5 + 2cdfghk + bdfhiz, befijklz^2 + 3bcdghijl, \\
& 2bgijklz + 2bcghil + cefhjz, 2defghjz + 3cefhijz + 3bdghiz\rangle \\
& \subset \mathbb{Q}[b, c, d, e, f, g, h, i, j, k, l, z] \\
T_7 = \langle & cfghijklz + cdz^7, 3bdikz^7 + 3bcdefghikl + 4bfghkz^5, 3befghijkz + 2bcegijz^3, \\
& 3cfhjlz + dfhjlz + 4bdfkl, 3bejz^4 + bdfgjk + 2begjz^2, \\
& cdefgjkz + 3efgjlz^2 + 4elz^5, bcdefghjk, 4cehjlz^4 + 3ceghijkl, efghjklz, ik, \\
& 4beghijkz^3 + 3bdeghijkl, cdefkl + dgjklz, 2bghijlz + bcdgiz + 4eghjkz, \\
& bcehijklz + cdghijlz^2, 2bcdefglz + 2cfgijlz^2 + chz^6, \\
& 4bdefhjlz + bdhijlz + 2defgklz, 2cdgiklz + cehklz^2 + 4cghilz, chjkl, \\
& 2bcdhijlz + cgijz^4, bdfhijkz + 4bdijkz^3 + 2dhlz^4\rangle \subset \mathbb{Q}[b, c, d, e, f, g, h, i, j, k, l, z] \\
T_8 = \langle & 3bejz^4 + bdfgjk + 2begjz^2, cdefgjkz + 3efgjlz^2 + 4elz^5,
\end{aligned}$$

$$
\begin{aligned}
&\quad bcdefghjk, 4cehjlz^4 + 3ceghijkl\rangle \subset \mathbb{Q}[b,c,d,e,f,g,h,i,j,k,l,z] \\
T_9 &= \langle 3hz^4 + 2cdfg, bdefgh + cfgz^3 + cgz^4, bcgz^2 + cdef + defz, \\
&\quad 3efgh + bcez + 2bfz^2, 3defh + 2cegh, dehz + 4cgz^2, 2cdefhz + chz^3, \\
&\quad 3cdefhz + 2cfghz, 3dfghz + 2efhz^2 + 2bcgz, bdhz + 2efz + 2bhz\rangle \\
&\quad \subset \mathbb{Q}[b,c,d,e,f,g,h,z] \\
T_{10} &= \langle 4cdfhjkz + 4efhijz^2 + cehiz^2, bcdfiz, 3bdefhj + 4cdeghz, cdegkz + bdiz^3, \\
&\quad bcdkz^2 + 2begjk, 2cdefhijz + 3cehijz^3 + bcdhz^4, efhjkz + 3bcfhz, \\
&\quad 2bcegiz + 3dghijz + 3fghiz, bdfjz + dfjkz, 4efhikz + 3befhi + 2dfghi, \\
&\quad cdhijz + 2efgkz^2, bcdgikz^2 + bcdfgik, 00, dfgikz, 2bcdghiz + bcegiz^2 + bdfijk, \\
&\quad cdefghijz, bcdegijkz + cdefkz^4, 4bdfghjz + bdgkz^3 + 2bcdeij, \\
&\quad cefghijkz + 4defgikz^3 + 4eghkz^4, bcdgijkz + ceghjkz^2 + 4cefghz^3\rangle \\
&\quad \subset \mathbb{Q}[b,c,d,e,f,g,h,i,j,k,z]
\end{aligned}
$$

As to the sub-algorithms called in each algorithm, we fix them according to preliminary experiments as follows:

- In Algorithm 3 we use Algorithm 7 for computing a saturated separating ideal because the number of generators of $I : Q_i$ tends to be large and the trick explained after Algorithm 7 often takes effect. In Asir the competitive computation explained in Section 4.3 is applied for computing a Gröbner basis with respect to an elimination order. This is realized by the OpenXM reset protocol described in Maekawa et al. (2001).
- In Algorithm 5 we use Algorithm 2 for computing a saturated separating ideal since the number of generators of $R_{ij} : R_{i-1}$ or its radical P_{ij} is relatively small. In Algorithm 2 we set $C_{ij} = P_{ij}$.
- Each extraction of Q_{ij} in Algorithm 3 or R_{ij} in Algorithm 5 is computed by $(IR_Y \cap R) : f^\infty$ in Asir and $(I : f^\infty)R_Y \cap R$ in Singular for appropriate I, Y and f (see Section 4.3).
- In Singular `minAssGTZ` is used for computing minimal associated primes.
- The incremental Gröbner basis computation and the early termination in Section 4.1 are used in both algorithms in Asir. These are available in Risa/Asir after version 20110112.

Two functions `noro_pd.syc_dec(Ideal,Vars)` and `noro_pd.syci_dec(Ideal,Vars)` are available in `noro_pd.rr`. These functions implement Algorithm 3 and 5 respectively. `Ideal` is a list of polynomials with variables `Vars`. a list $[L_1, \ldots, L_t]$ representing a minimal primary decomposition of `Ideal` is returned, where $L_i = [QP_{i1}, \ldots, QP_{in_i}]$ and each QP_{ij} is a pair $[Q_{ij}, P_{ij}]$ such that P_{ij} is an associated prime of level i and Q_{ij} is P_{ij}-primary component of `Ideal`. A function `primdecSYCI(ideal I)` is available in `primdecSYCI.lib`. This function implements Algorithm 5.

Timings were measured on a 64-bit Linux machine with Intel Xeon X5570, 2.93GHz. We show elapsed time in seconds. In Table 1 the time for Algorithm 5 by `noro_pd.syci_dec` in Asir is shown with the time for Algorithm 6 in parentheses. We also show the timings of Algorithm 5 by `primdecSYCI` and SY by `primdecSY` in Singular. The last column shows the number of components in each QL_i. In that table '—' means that the timing was not measured because it is expected that it takes very long time.

Ideal	Asir		Singular		$\lvert QL_1\rvert,\dots,\lvert QL_t\rvert$
	Alg. 3	Alg. 5 (Alg. 6)	SY	Alg. 5	
I_2	0.4	0.5 (0.4)	>1h	5.9	1,1,1
I_3	11	36 (16)	—	230	1,2,1,1
$A_{2,3,5}$	1.3	1.0 (0.6)	42000	1.9	10,5,3,1
$A_{2,3,6}$	12	5.4 (3.2)	—	8.9	18,12,8,2,1
$A_{2,3,7}$	330	33 (20)	—	54	32,26,20,7,2,1
$A_{2,3,8}$	17000	220(130)	—	390	57,56,46,19,7,2,1
$A_{2,3,9}$	—	1800(1100)	—	3500	102,116,103,46,21,8,5,1
$A_{2,3,10}$	—	14000(9200)	—	31000	182,236,224,110,55,26,19,2,1
$A_{2,4,4}$	5.5	3.8 (2.4)	>3h	5.6	15,12,4,1
$A_{2,4,5}$	1200	100 (67)	—	135	35,30,19,9,2,1
$A_{2,4,6}$	>100h	6400(4300)	—	6500	82,89,73,36,18,10,4,1
$A_{2,4,7}$	—	300h(260h)	—	—	193,254,236,136,74,63,35,16,1
$A_{2,5,5}$	—	38000(24000)	—	39000	100,107,80,61,35,32,18,4,1
T_1	950	48 (30)	1800	75	49,36,26,23,17,12,5,1
T_2	26	13 (12)	40	26	15,22,15,7,4
T_3	1900 ‡	80 (43)	>5h	140	46,68,64,19,3
T_4	57	25 (14)	2000	46 §	40,34,25,17,18,6,4,1
T_5	7500	34 (20)	>5h	67	14,28,30,27,7,2,1
T_6	4.5	3.6 (2.4)	210	5.0	48,42,18,8,4,2
T_7	580	180 (97)	>3h	330 §	55,58,66,62,56,44,37,12,1
T_8	1.5	1.3 (0.9)	>3h	2.0 §	37,16
T_9	8.0	4.6 (3.3)	4.4	8.6 §	15,14,12,10,4,1
T_{10}	1000	280 (160)	>1h	290	76,49,54,47,39,33,28,10,1

Table 1. Timing data for computing primary decomposition

I_2	I_3	$A_{2,3,5}$	$A_{2,3,6}$	$A_{2,3,7}$	$A_{2,3,8}$	$A_{2,4,4}$	$A_{2,4,5}$
0.9	17	5	133	3540	146h	31	12700

Table 2. Timing data of Algorithm 1 (from Noro (2010))

In Table 3 we show the timings for the examples in Decker et al. (1998) which do not necessarily have many embedded components but were taken from a wide range of sources. In this table the timings by Singular `primdecGTZ`, `primdecSY` and `primdecSYCI` for non zero-dimensional ideals are shown. All computations are done over $\mathbb{Q}$. Here we omit zero-dimensional ideals because `primdecSYCI` calls `primdecGTZ` for zero-dimensional ideals.

No.	1	2	3	4	5	6	7	9	12	14	16	19	20
GTZ	0.03	1.3	0.1	0.4	7.4	1.2	0.2	0.1	0.3	0.03	0.2	0.1	0.1
SY	5.2	0.5	0.04	0.2	0.4	0.2	0.2	0.7 ¶	12 ¶	0.1	0.4	0.1 ¶	1.0
SYCI	0.2	0.3	0.1	0.1	0.8	4.0	0.1	0.6	3.6	0.1	1.3 §	0.1	0.1

No.	21	22	23	24	25	27	28	29	30	31	32	33
GTZ	0.1	0.3	0.2	0.3	1.1	0.01	0.03	55	6.6	0.04	1.0	0.04
SY	0.1	0.2	2	0.2	0.9 ¶	0.1	0.2	49 ¶	0.6	0.01	0.3	0.02
SYCI	0.2	0.4	0.7	0.1	0.8	0.1	0.1	2.3	0.7	0.1	0.6	0.1

Table 3. Timing data in Singular for examples in Decker et al. (1998)

6. Discussion

6.1. Evaluation of the new algorithms

Our initial purpose was to clarify the reason of the efficiency of Algorithm 1 and it has been satisfied by introducing the notion of saturated separating ideal. As a result we could propose Algorithm 3, which is simpler than Algorithm 1 and produces no intermediate redundant components. However, from a practical point of view we cannot expect a significant speed-up by Algorithm 3 because the number of redundant components is already zero or very small in Algorithm 1 and the most time-consuming part is the computation of separating ideals in both these two algorithms. Nevertheless Table 1 and Table 2 show that Algorithm 3 is more efficient than Algorithm 1. There are two possible reasons of this improvement: one is that we have introduced various techniques to speed up the computation of saturated separating ideals, and the other is that we compute the minimal associated primes of I_i from not I_i itself but $I : Q_{i-1}$.

Table 1 clearly shows that the performance of Algorithm 5 is remarkable for harder problems. Let us examine the results in Asir for $A_{2,3,8}$ for example. If we apply Algorithm 3 it takes 4.4 hours to compute saturated separating ideals. In Algorithm 5 each saturated separating ideal is constructed for extracting a single primary component and it is relatively easy to compute them. In fact it took only 90 seconds. Consequently the total time is greatly reduced.

‡ Primary components are computed as $(I : f^\infty)R_Y \cap R$.

§ `noFacstd` is specified in `minAssGTZ`.

¶ `minAssGTZ` is used in `primdecSY`.

6.2. Comparison with other methods

Table 1 shows that SY can decompose some of the input ideals. However the computing time of SY tends to be longer than that of the new algorithms. We guess that this is caused by a large number of redundant components. For example the number of embedded components of $A_{2,3,5}$ is 9. But Singular SY produces 411 redundant components during the execution. In SY we cannot predict how many redundant components will be produced. Our new algorithm not only produces no intermediate redundant components but also it produces components with a definite property in a definite order: in the i-th step all the primary components whose associated primes are of level i are exactly produced.

$A_{2,m,n}$'s are binomial ideals and we can apply a special algorithm based on cellular decomposition (Eisenbud, Sturmfels, 1996) to them. An implementation of the algorithm is available in Macaulay2 (Grayson, Stillman, 2011) but it took 85 seconds and 115 minutes to decompose $A_{2,3,5}$ and $A_{2,4,4}$ respectively and it could not decompose the other ones in reasonable time.

Table 3 shows that the performances of `primdecGTZ`, `primdecSY` and `primdecSYCI` are comparable except for a few examples. The reason is that the dominant part in these functions is often the zero-dimensional decomposition if the computation of embedded components is not hard and that subroutines in `primdec.lib` are commonly used in these functions. In particular GTZ performs best for zero-dimensional ideals and there is no reason to apply Algorithm 5 for such ideals. However, as shown in Table 1, Algorithm 5 can surely decompose some examples which are hard to decompose by GTZ and SY. Since Algorithm 5 is an improvement of SY, it is practical to choose either Algorithm 5 or GTZ depending on an input ideal.

6.3. Computation of associated primes

We also presented Algorithm 6 for computing all associated primes of an ideal I without computing primary decomposition. Since it does not contain the computation of saturated separating ideals, we expected at first that the computation would be much faster than Algorithm 5. However Table 1 shows that Algorithm 6 is not significantly faster than Algorithm 5. The reason is that the extraction of Q_{ij} from $Q_{i-1} \cap Q_{ij}$ is done very efficiently compared with the extraction of Q_{ij} from I_i in Algorithm 3.

An algorithm for computing the set of all associated primes of an ideal I has already been presented in Eisenbud et al. (1992). It is based on equidimensional decomposition and the radical of each equidimensional component is first computed by homological algebra. Then the associated primes are obtained by the prime decomposition of the radical. In our algorithm the radical decomposition of $I : Q_i$ is first computed and the algorithm proceeds with the knowledge of the prime components of $I : Q_i$. It is an interesting future work to compare two methods from a practical point of view.

6.4. parallel computation

Finally we mention an application of parallel computation. There are several parts where we can apply parallel computation: computation of the radical decomposition of an ideal quotient, extraction of isolated primary components in Algorithm 3 and extraction of Q_{ij} from $Q_{i-1} \cap Q_{ij}$ in Algorithm 5. If the cost for non-parallelizable parts is not large, we can expect that the parallelization of these parts reduce the total elapsed time.

Acknowledgements

We would like to thank Prof. Kazuhiro Yokoyama for valuable comments and suggestions. In particular we could improve the efficiency of our implementation of Algorithm 5 according to his suggestion on the choice of C_{ij} in that algorithm. We would also like to thank anonymous referees for their useful suggestions.

References

W. Decker, G.-M. Greuel, G. Pfister, H. Schönemann. Singular 3-1-2 — A computer algebra system for polynomial computations, `http://www.singular.uni-kl.de/`.

W. Decker, G.-M. Greuel, G. Pfister, 1998. Primary decomposition: Algorithms and comparisons, *Algorithmic algebra and number theory*, Springer, 187–220.

P. Diaconis, D. Eisenbud, B. Sturmfels, 1998. Lattice walks and primary decomposition, in B. Sagan, R. Stanley eds, Mathematical Essays in Honor of Gian-Carlo Rota, Birkhäuser, 173–194.

D. Eisenbud, C. Huneke, W. Vasconcelos, 1992. Direct methods for primary decomposition, *Invent. Math.* **110**, 207–235.

D. Eisenbud, B. Sturmfels, 1996. Binomial Ideals, *Duke Math. J.* **84**, 1–45.

P. Gianni, B. Trager, G. Zacharias, 1988. Gröbner basis and primary decomposition of polynomial ideals, *J. Symb. Comp.* **6**, 149–167.

D.R. Grayson, M.E. Stillman, 2011. Macaulay2, a software system for research in algebraic geometry, Available at `http://www.math.uiuc.edu/Macaulay2/`.

M. Maekawa, M. Noro, K. Ohara, N. Takayama, Y. Tamura, 2001. The Design and Implementation of OpenXM-RFC 100 and 101, *Proc. ASCM2001*, World Scientific, 102–111.

M. Noro, 2010. New algorithms for computing primary decomposition of polynomial ideals, in *Proc. ICMS 2010*, LNCS 6327, 233–244.

M. Noro, 2011. Packages for computing primary decomposition.
`http://www.math.kobe-u.ac.jp/HOME/noro/pd.html`.

M. Noro, N. Takayama, H. Nakayama, K. Nishiyama, K. Ohara, 2011. Risa/Asir : A computer algebra system. `http://www.math.kobe-u.ac.jp/Asir/asir.html`.

OpenXM committers, 2011. OpenXM, a project to integrate mathematical software systems, `http://www.openxm.org`.

T. Shimoyama, K. Yokoyama, 1996. Localization and Primary Decomposition of Polynomial ideals, *J. Symb. Comp.* **22**, 247–277.