An Approach to Checking Link Conflicts in the Mapping of Uniform Dependence Algorithms into Lower Dimensional Processor Arrays

Jenn-Yang Ke and Jong-Chuang Tsay

Abstract—in this paper, we propose an enumeration method to check link conflicts in the mapping of \( n \)-dimensional uniform dependence algorithms with arbitrary convex index sets into \( k \)-dimensional processor arrays. Previous methods on checking the link conflicts had to examine either the whole index set or the I/O spaces whose size are \( O(N^n) \) or \( O(N^{n-1}) \), respectively, where \( N \) is the problem size of the \( n \)-dimensional uniform dependence algorithm. In our approach, checking the link conflicts is done by enumerating integer solutions of a mixed integer linear program. In order to enumerate integer solutions efficiently, a representation of the integer solutions is devised so that the size of the space enumerated is \( O(2Nn^{n-1}) \). Thus, our approach to checking link conflicts has better performance than previous methods, especially for larger \( k \). For the special case \( k = n - 2 \), we show that link conflicts can be checked by solving two linear programs in one variable.

Index Terms—Uniform dependence algorithms, lower dimensional arrays, space-time mapping, link conflict, mixed integer linear programming, Hermite normal form, Smith normal form.

1 INTRODUCTION

REGULAR processor arrays, such as systolic arrays introduced by Kung [1], which have regularly and locally connected connections via data links between processing elements (PEs) are very suitable for implementation on VLSI chips. This type of processor arrays supports the parallel implementation of algorithms, especially the uniform dependence algorithms, from signal or image processing and scientific computation applications [2].

Uniform dependence algorithms, termed by Shang and Fortes [3], include those described by single uniform recurrence equations [2], [4], [5], [6], [7], [8], [9], [10] and those described by programs with nested loops [11]. They are characterized by uniform data dependencies and unit-time computation. Informally, a uniform dependence algorithm is represented by a subset (called index set) of multidimensional integer points (called index points) and a finite set of data dependence vectors. The index set of an algorithm is a finite convex subset of \( \mathbb{Z}^n \) [12]. The minimal convex polytope or convex hull \( R \) bounding the index set is usually a nondegenerated convex polytope in \( \mathbb{R}^n \). An index set is a hyperparallelepiped index set if \( R \) is a hyperparallelepiped. We call \( n \) the dimension of the uniform dependence algorithm.

Most researches on synthesizing [13], [14], [15], [16], [17], [18] processor arrays from the uniform dependence algorithms focus on finding a space-time mapping (linear transformation) to the algorithm such that the transformed algorithm represents a regular processor array. The space-time mapping is, in general, represented as a transformation matrix. The first row and the rest of the transformation matrix are the time mapping vector, called linear schedule vector, and the space mapping matrix, called allocation matrix, respectively. In other words, the space-time mapping transforms an index point of the uniform dependence algorithm into a time step and a PE location in the processor array. Thus, a processor array is a \( k \)-dimensional array if the transformation matrix is \( (k+1) \times n \) integer matrix. For mapping a uniform dependence algorithm into a \( k \)-dimensional processor array, three kinds of conflict-free mapping conditions must be satisfied. They are precedence, computation, and data link conflict-free conditions.

We say that a computational conflict occurs if more than one computation of a uniform dependence algorithm are mapped to the same processor and the same time step, and that a link conflict occurs if more than one datum are mapped such that they travel along the same data link at the same time step. In this paper, we address the problem of checking link conflicts in the mapping of \( n \)-dimensional uniform dependence algorithms into \( k \)-dimensional processor arrays with \( 0 < k < n - 1 \).

There have been several attempts on the problem of checking the link conflicts [13], [14], [16], [17], [18], [19], [20]. One of these methods [13], [14] has to examine the whole index set of size \( O(N^n) \), where \( N \) is the problem size. Other methods [16], [17], [18], [19], [20] use the I/O spaces concept to check link conflicts. Each I/O space is associated with a data dependence vector \( d \). An I/O space can be an input space or an output space and is defined as the set \( \{ \tilde{t} - \tilde{d}_i \mid \tilde{t} \in J \text{ and } \tilde{d}_i \notin J \} \) and \( \{ \tilde{t} \mid \tilde{t} \in J \text{ and } \tilde{d}_i \notin J \} \), respectively, where \( J \) is the index set. However, an exact link conflict checking cannot be obtained by their methods if the I/O space is a nonconvex one. In [17], [18], a procedure was proposed to map nonconvex I/O space into convex one. But, the projection of nonconvex I/O space may introduce superfluous points which are not needed in the computation of the algorithm, into the projected I/O space. An example of such projection of nonconvex I/O space is shown in Fig. 1. Thus, to check link conflict exactly, enumeration of the I/O space of size \( O(N^{n-1}) \) is required.

The rest of the paper is organized as follows: In the next section, we define the algorithm model and the array model used in this paper. Section 3 is devoted to the formulation of the link conflict checking problem as a mixed integer linear program. In Section 4, a representation of the integer solutions of the mixed integer linear program is given. Based on this representation, we can enumerate the integer solutions in an efficient way. For the special case \( k = n - 2 \), we also show that the link conflicts can be checked by solving two linear programs in one variable. In Section 5, we estimate the size of the enumeration space and show that the time complexity for the representation is polynomial. Finally, a conclusion is given in Section 6.

2 ALGORITHM MODEL AND ARRAY MODEL

In this section, the models for the algorithms and processor arrays used in this paper are introduced. We briefly describe them as follows.

2.1 Algorithm Model

Algorithms under consideration are the uniform dependence algorithms. A uniform dependence algorithm is a single-assignment algorithm [2], which can be described as \( n \)-nested loops of the form:

\[
\text{for } i_1 \text{ from } l_1 \text{ to } u_1 \text{ by } 1 \text{ do}
\]

\[
\text{for } i_2 \text{ from } l_2 \text{ to } u_2 \text{ by } 1 \text{ do}
\]

... 

\[
\text{for } i_n \text{ from } l_n \text{ to } u_n \text{ by } 1 \text{ do}
\]

\[\text{stat}_1;\]

\[\text{stat}_2;\]

... 

\[\text{stat}_m;\]

endfor
Formulation of the Link Conflict Checking Problem

In this section, we discuss the formulation of the link conflict checking problem. A link conflict occurs if and only if two tokens of a variable arrive at the same PE at the same time and move together contending the same link. Given a uniform dependence algorithm with data dependence matrix $D$ and index set $J$, the formulation of the link conflict checking problem can be derived as follows:

Let $T = \begin{bmatrix} A \\ S \end{bmatrix} \in \mathbb{Z}^{(k+1) \times n}$ be the space-time mapping. Let variable $V$ be associated with data dependence vector $\vec{d}_V$. Since, for each data dependence vector $\vec{d}_V = (d_1, \ldots, d_n)$, we restricted that $\text{gcd}(d_1, \ldots, d_n) = 1$, there is no integer point between index point $\vec{i}$ and index point $\vec{i} + \vec{d}_V$. Let $g$ be the greatest common divisor of the entries of the vector $S\vec{d}_V$. Obviously, the value of $g$ may be greater than 1 for some allocation matrix $S$ and data dependence vector $\vec{d}_V$. Assume that $g > 1$. Then,
In this section, we give a method to enumerate the integer solutions \( \mathbf{y} \) of the MILP problem so as to check link conflicts for a variable \( V \) with its associated data dependence vector \( \mathbf{d}_V \). As \( \mathbf{y} = \mathbf{y} - q \mathbf{d}_V \) in the MILP problem where \( \mathbf{y} \in \mathbb{Z}^{m \times 1} \) and \( q \in \mathbb{Q} \), enumeration of the integer solutions \( \mathbf{y} \) is not a straightforward task. Thus, we derive a representation for the integer solutions \( \mathbf{y} \). Based on this representation, all integer solutions \( \mathbf{y} \in \text{diff}(J) \) can be enumerated systematically.

### 4 Checking Link Conflicts

In this section, we give a method to enumerate the integer solutions \( \mathbf{y} \) of the MILP problem so as to check link conflicts for a variable \( V \) with its associated data dependence vector \( \mathbf{d}_V \). As \( \mathbf{y} = \mathbf{y} - q \mathbf{d}_V \) in the MILP problem where \( \mathbf{y} \in \mathbb{Z}^{m \times 1} \) and \( q \in \mathbb{Q} \), enumeration of the integer solutions \( \mathbf{y} \) is not a straightforward task. Thus, we derive a representation for the integer solutions \( \mathbf{y} \). Based on this representation, all integer solutions \( \mathbf{y} \in \text{diff}(J) \) can be enumerated systematically.

#### 4.1 A Representation of the Integer Solutions

In order to enumerate the integer solutions \( \mathbf{y} \), we represent \( \mathbf{y} \) as an integer linear combination of some vectors. First, we expressed \( \mathbf{y} = \mathbf{y} - q \mathbf{d}_V \) as a linear combination of some integer vectors as follows. Denote \( \text{NULL}(T) = \{ \tau \mid \tau \mathbf{x} = 0 \} \), the null space of the space-time mapping matrix \( T \). Since \( \tau \mathbf{x} = 0 \) in the MILP problem, we have \( \mathbf{y} \in \text{NULL}(T) \). Thus, \( \mathbf{y} \) can be expressed as a linear combination of the vectors of a basis spanning \( \text{NULL}(T) \). In order to find a basis of the \( \text{NULL}(T) \), we introduce the notion of the Hermite normal form.

**Definition 4.1.** A matrix is unimodular if and only if it is integral and the absolute value of its determinant is one.

**Theorem 4.1 (Hermite normal form) [21, p. 45].** Let \( T \in \mathbb{Z}^{(k+1) \times n} \) and \( \text{rank}(T) = k + 1 \). Then, there exists a unimodular matrix \( W \in \mathbb{Z}^{n \times n} \) such that \( TW = H = [L, 0] \) (0 denotes a zero matrix), where \( L \in \mathbb{Z}^{(k+1) \times (k+1)} \) is a nonsingular and lower triangular matrix. Matrix \( H \) is called the Hermite normal form of \( T \).

From Theorem 4.1, we have \( T = HW^{-1} \). Let \( W = [\overline{w}_1, \ldots, \overline{w}_n] \). Then, \( \tau \mathbf{y} = 0 \) can be rewritten as \( HW^{-1} \tau \mathbf{y} = 0 \). Let \( \overline{\mathbf{z}} = W^{-1} \tau = (\overline{z}_1, \ldots, \overline{z}_n)^T \). Then, we have the following lemma.

**Lemma 4.2 [15].**

\[
\tau = (\overline{w}_{k+2}, \ldots, \overline{w}_n)^T \cdot \begin{bmatrix} \overline{\zeta}_{k+2} \\ \vdots \\ \overline{\zeta}_n \end{bmatrix}
\]

where \( \overline{\zeta}_i \in \mathbb{Q}, i = k + 2, \ldots, n \).

**Proof.** After simple manipulation, the lemma follows. End of proof.

Let \( M = (\overline{d}_V, \overline{w}_{k+2}, \ldots, \overline{w}_n) \in \mathbb{Z}^{(n-k) \times (n-k)} \). By Lemma 4.2, \( \overline{w}_i \in \text{NULL}(T), i = k+2, k+3, \ldots, n \). In addition, since \( |\det(W)| = 1 \), \( \overline{w}_{k+2}, \ldots, \overline{w}_n \) are linear independent vectors. Moreover, since \( \overline{d}_V \neq 0 \), we have \( T \overline{d}_V \neq 0 \). Thus, \( \overline{d}_V \notin \text{NULL}(T) \). As a consequence, \( \text{rank}(M) = n - k \). Let

\[
\tau = (q, \overline{\zeta}_{k+2}, \ldots, \overline{\zeta}_n)^T \in \mathbb{Q}^{(n-k) \times 1}.
\]

We have \( \mathbf{y} = M\tau \) where \( M \) is an integer matrix and \( \tau \) a rational vector. Since \( \mathbf{y} \) is a rational linear combination of the columns of matrix \( M \), vectors \( \mathbf{y} \) cannot be enumerated. Thus, we must select another set of linear independent vectors such that vector \( \mathbf{y} \) can be expressed as an integer linear combination of those vectors. To achieve this, the notion of the Smith normal form is introduced.

**Theorem 4.3 (Smith normal form) [21, p. 50].** Given a matrix \( A \in \mathbb{Z}^{n \times m} \), there exist two unimodular matrices \( U \in \mathbb{Z}^{n \times n} \) and \( V \in \mathbb{Z}^{m \times m} \) such that

\[
UAV = S(A) = \begin{bmatrix}
s_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
0 & s_2 & \cdots & 0 & 0 & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & s_{m'} & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\end{bmatrix}
\]

\( S(A) \) is called the Smith normal form of matrix \( A, S(A) \) is unique, \( s_1, \ldots, s_{m'} \) are positive integers satisfying \( s_1 | s_2 | \cdots | s_{m'} \), and \( \Pi_{j=1}^{m'} s_j \), the greatest common divisor of subdeterminants of order \( k \) of the matrix \( A \), and \( m' \) is the rank of the matrix \( A \).

Now, let \( S(M) \in \mathbb{Z}^{n \times (n-k)} \) be the Smith normal form of the matrix \( M \in \mathbb{Z}^{(n-k) \times (n-k)} \). Then, there exist two unimodular matrices
$S \text{ is unimodular}$

$S = (A, S) = (2 1 2) -2$

Fig. 2. The index set $J$ and the dependence vectors $\bar{\alpha}_A, \bar{\alpha}_B$, and $\bar{\alpha}_C$ associated with variables $A, B$, and $C$, respectively.

**Theorem 4.4.**

$S(M)_{n \times (n-k)} = U_{n \times n} M_{n \times (n-k)} V_{(n-k) \times (n-k)}$

and

$\bar{\gamma} = M_{n \times (n-k)} \bar{\tau}_{(n-k) \times 1}$

we have

$U_{n \times n} \bar{\gamma}_{n \times 1} = S(M)_{n \times (n-k)} \bar{\tau}_{(n-k) \times 1}$

where

$\bar{\tau}_{(n-k) \times 1} = V_{(n-k) \times (n-k)} \bar{\tau}_{(n-k) \times 1}.$

Since $\bar{\gamma}_{n \times 1}$ must be an integer vector and $U_{n \times n}$ is a unimodular matrix, we have $U_{n \times n} \bar{\gamma}_{n \times 1} \in Z^{n \times 1}$. As a consequence, we have
In Theorem 4.4, we have expressed all integer vectors $\mathbf{R}$ and can be expressed as the set $M_s \in Z^{n \times 1}$. Remember that $\text{rank}(M) = n - k$. And, diagonal entries $s_i, i = 1, \ldots, n - k$ of the matrix $S(M)$ are positive integer numbers. This implies that $\beta_i = z_i/s_i, z_i \in Z, i = 1, \ldots, n - k$. Thus, after simple manipulation, $\gamma$ can be expressed as an integer linear combination of the first $n - k$ columns of matrix $U^{-1}$. This completes the proof.

Notice that we place the data dependence vector $\mathbf{d}_V$ in the first column of the matrix $M$. Let $g$ be the greatest common divisor of the entries of vector $\mathbf{d}_V$. By the restriction of the algorithm, $g$ is equal to one. Now, by the construction of Smith normal form $S(M)$ [21, p. 50], its (1, 1) entry $s_1$ should divide all entries of $\mathbf{d}_V$. Therefore, $s_1 = 1$. Since $\mathbf{d}_V$ is the first column of $M$ with $g = 1$, vector $\mathbf{d}_V$ can be brought into $\tau_1 = (1, 0, \ldots, 0)^T$ after suitable row operations applied on $M$. In other words, there exists a unimodular matrix $U$ such that the first column of $UMV$ is equal to $\tau_1$. Thus, to make the first column of $S(M) = UMV$ equal to $\tau_1$, we can choose a unimodular matrix $V$ such that its first column is equal to $\tau_1$. Now, since $U^{-1}S(M) = MV$, by comparing their first columns, we have the following theorem.

**Theorem 4.5.** There exist two unimodular matrices $U, V$ such that first columns of $UMV$ and $V$ are both equal to $\tau_1$; and the first column $\tau'_1$ of $U^{-1}$ is equal to $\mathbf{d}_V$.

### 4.2 Enumeration of the Integer Solutions

In Theorem 4.4, we have expressed all integer vectors $\gamma$ in the MILP problem as an integer linear combination of integer vectors. But, only integer vectors $\gamma \in \text{diff}(J)$ need to be checked for link conflicts of two tokens of a variable. Notice that the index set $J$ can be expressed as a set of linear inequalities, i.e.,

$$J = \{e | l_i \leq e_i \leq u_i, a_i \in Z^{n \times 1}, e \in Z^{n \times 1}, l_i, u_i \in Z, u_i > l_i, i = 1, \ldots, a\}.$$  

Since $\text{diff}(J) = \{e | l_i - u_i \leq e_i - l_i, a_i \in Z^{n \times 1}, e \in Z^{n \times 1}, l_i, u_i \in Z, u_i > l_i, i = 1, \ldots, a\}$, we have

$$\text{diff}(J) = \{e | l_i - u_i \leq e_i - l_i, a_i \in Z^{n \times 1}, e \in Z^{n \times 1}, l_i, u_i \in Z, u_i > l_i, i = 1, \ldots, a\}.$$  

Thus, we see that $\text{diff}(J)$ is symmetric with respect to the origin and can be expressed as the set

$$\{e | AE \leq \vec{b}, \vec{b} > 0, A \in Z^{2 \times n}, \vec{b} \in Z^n, e \in Z^{n \times 1}\}.$$  

Since $\gamma \in \text{diff}(J)$, we have $A\gamma \leq \vec{b}$. Let $U = (e, \ldots, e, a_i)$, and $\tau = (z_1, \ldots, z_n)^T$. By Theorem 4.4, we have $\gamma = U\tau$. By substituting $\gamma = U\tau$ in $A\gamma \leq \vec{b}$, the integer vectors $\tau$ can be enumerated by finding the bounds $z_{i_{\min}}$ and $z_{i_{\max}}$ of the values $z_i$, i.e., $z_{i_{\min}} \leq z_i \leq z_{i_{\max}}$ for $i = 1, \ldots, n - k$ such that $AU\tau \leq \vec{b}$. Since the convex polytope $R$ defined by $AU\tau \leq \vec{b}$ is symmetric with respect to the origin, $0 \in R$ and $z_{i_{\min}} = -z_{i_{\max}}$. Since $R$ is a convex polytope, the enumeration of the feasible $\tau$ (or the integer solutions $\tau = U\gamma$ of the MILP problem) terminates in a finite number of steps. The enumeration procedure is terminated as soon as a nonzero integer solution $\tau$ is found such that $\tau \neq z\mathbf{d}_V$ for any integer $\mathbf{d}_V$ since, in this case, a link conflict is found.

Notice that, as described in Section 3, $\gamma = z\mathbf{d}_V$ for some integers $\mathbf{d}_V$ are solutions of the MILP problem. The relation of these $\gamma$s and the link conflicts is stated in the following theorem.

**Theorem 4.6.** Let $T = \begin{bmatrix} A & \mathbf{1} \\ \mathbf{S} & \mathbf{B} \end{bmatrix}$ be a space-time mapping with $S\mathbf{d}_V \neq \mathbf{0}$. Let $\gamma = (\pi_1, \ldots, \pi_{n-k})^T(z_1, \ldots, z_{n-k})^T$ be a nonzero integer solution of the MILP problem. Then, there is no link conflict for the tokens of the variable $V$ if and only if all of $z_i$ for $i = 2, \ldots, n - k$ have zero values only.

**Proof.** Notice that $\gamma = z_1\mathbf{d}_V + (\pi_2, \ldots, \pi_{n-k})^T(z_2, \ldots, z_{n-k})^T$ by Theorems 4.4 and 4.5. Since token(0) and token(1) represent two different tokens or if only if $i \neq \bar{i} + \mathbf{d}_V$ for $r \in Z$, and $\gamma = \tau_i, \bar{i}$ represents link conflicts of two tokens at any two index points $\bar{i}$ and $\bar{i}$, the theorem follows. End of the proof.

From Theorem 4.6, we see that if there exists a nonzero integer vector $\tau = (z_1, z_2, \ldots, z_{n-k})^T$ which has a form other than $(0, 0, \ldots, 0)$, then there is a link conflict. Let $R$ be the convex polytope defined by $A\tau \leq \vec{b}$. To check whether there is an integer vector $\tau$ which has a form other than $(0, 0, \ldots, 0)$, we need to enumerate the integer vectors $\tau$ in $R$. Since $R$ is symmetric with respect to the origin, only half space of $R$ needs to be enumerated. To enumerate it, first, we construct a minimal hypercube $C$ containing the half space of $R$. The minimal hypercube $C$ is

$$C = \{(z_1, z_2, \ldots, z_{n-k}) | 0 \leq z_i - \xi_i \leq \xi_i - z_i, 2 \leq i \leq n - k\},$$

where $\xi_i = [z_{i_{\min}}]$ and $z_{i_{\max}}$ is found by solving the following linear program: $\max z_i | AU\tau \leq \vec{b}, \tau \in \mathcal{Q}$. Then, integer vectors $\tau \in C$ are enumerated in lexicographical order. For the special case $k = n - 2$, the enumeration of integer vectors is not necessary as stated by the following theorem.

**Theorem 4.7.** Link conflicts on mapping an $n$-dimensional uniform dependence algorithm into an $(n - 2)$-dimensional processor array can be checked by solving two linear programs in one variable.

**Proof.** Since the processor array is $(n - 2)$-dimensional, solution space $R$ is a convex polytope in the space spanning by vectors $\mathbf{d}_V$ and $\tau_1$ by Theorem 4.4 and Theorem 4.5. Notice that $\tau = z_1\mathbf{d}_V + z_2\tau_1$, and $\gamma \in \text{diff}(J) = \{\tau_i \neq \bar{i}, \bar{i} \in J\}$. These observations imply that the convex polytope $R$ defined by $AU\tau \leq \vec{b}$ have at least three integer solutions $\tau = -\mathbf{d}_V, 0$, and $+\mathbf{d}_V$; otherwise, the computation in each index point in index set $J$ uses (generates) a value of the variable to be input (output) from (to) the host. Since $R$ is a convex polytope and symmetric with respect to the origin and contains three integer points $\{(-1, 0)^T, (0, 0)^T, (1, 0)^T\}$, $R$ must contains an integer point $(z_1, 1)$ if $R$ contains rational point $(z_1, z_2)$ such that $z_2 \geq 2$. Thus, link conflicts can be checked by testing whether $R$ contains an integer point $(z_1, 1)$. In other words, solve the following two linear programs: $\min z_1 | AU\tau \leq \vec{b}, \tau \leq \mathcal{Q}$ and $\max z_1 | AU\tau \leq \vec{b}, \tau \leq \mathcal{Q}$. If those linear programs have optimal solutions such that $[z_{i_{\min}}]$ and $[z_{i_{\max}}]$, there is a link conflict. Otherwise, there is no link conflict. This completes the proof.

### 5 Time Complexity Estimation

Now, we compare the time complexity of our method to that of Lee and Kedem [13], [14] and that of Xue [16], [18] or Ganapathy and Wah [19], [20]. Let $N$ be the problem size parameter for the $n$-dimensional uniform dependence algorithm. To check link conflicts, Lee and Kedem [13], [14] enumerated all pairs of index points in the index set $J$. Since $J$ is of size $O(N^n)$, the time complexity is $O(N^{2n})$. Xue [16], [18] (or Ganapathy and Wah [19], [20]) examined the I/O space instead of the whole index set. In order to check the link conflicts exactly, all index points in the I/O space need to be enumerated. Thus, the time complexity is $O(N^{2n-1})$. By our approach, as argued below, only $O((2N)^{n-k})$ integer vectors needed to be enumerated, where $k$ is the dimension of the processor array.
Recall that the convex polytope $R$ is defined by $A^T z \leq b$. Since each component of $b$ is of size $O(N)$ (see the first paragraph of Section 4.2) and $A^T$ is an integer matrix, the number of $z$s to be enumerated is $O(2N)$. Therefore, the total number of integer vectors to be enumerated is $O((2N)^{n-k})$. The additional cost that we pay for it is the time taken to find a representation of the integer solutions. To find the representation, we need to compute the Hermite normal form of space-time mapping matrix \( T = \begin{bmatrix} A \\ S \end{bmatrix} \) and the Smith normal form \( S(M) \) of matrix \( M \) corresponding to the data dependence vector \( d_\nu \). Using the algorithms in [22] and [23, appendix A], both forms can be computed in polynomial time. Consequently, the representation of the integer solutions can be found in polynomial time.

6 Conclusion

In this paper, a new formulation for the checking of link conflicts on mapping the uniform dependence algorithms into lower dimensional processor arrays with link connection between neighboring PEs is proposed. The formulation is a mixed integer linear program (MILP) and an integer solution of it can represent a neighboring PEs is proposed. The formulation is a mixed integer linear program (MILP) and an integer solution of it can represent a neighboring PEs is proposed. However, there is a main difference in implementation requirements between our method and that in [15]. First, they allow the data arrive before the time of their usage at the PE. Thus, a large bandwidth is required between neighboring PEs to support the necessary data movement. For our method, the data must arrive at the PE at the time of their usage. And, the buffer size between neighboring PEs can be derived when the design is complete. Second, if a large bandwidth is not possible, their method does not guarantee free of link conflicts for some designs. For more details, see [24]. But, all the designs produced by our method are free of link conflicts.

The applicability of our method can be seen by the following arguments: First, our method can be applied to uniform dependence algorithms with arbitrary convex index sets. Second, using the approach proposed in [13], our method can also be applied to the case that the components of the dependence vector are not relative-prime. Their approach is the use of a modified array model. For details, see their paper [13, pp. 68-69]. Third, since synthesizing processor arrays from algorithms is divided into two main steps, uniformization step [25] and mapping step, our method can be used in the mapping step after the algorithm is uniformized.

Acknowledgments

This research was supported by the National Science Council of the Republic of China under contract NSC88-2213-E-036-003.

References