JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 14, 743-763 (1998)

Finding Space-Optimal Linear Array for Uniform Dependence Algorithms with Arbitrary Convex Index Sets[†]

JENN-YANG KE AND JONG-CHUANG TSAY
Department of Computer Science and Information Engineering
National Chiao Tung University
Hsinchu, Taiwan 300, R.O.C.
E-mail: jctsay@csie.nctu.edu.tw

The mapping of an *n*-dimensional uniform dependence algorithm onto a linear processor array can be considered as a linear transformation problem. However, to find a linear space-optimal transformation is difficult because the conditions for checking a correct mapping and the space cost function do not have closed-form expressions, especially when the index set *J* of an *n*-dimensional algorithm is of an arbitrary bounded convex index set. In this paper, we propose an enumeration method to find a space-optimal PE allocation vector for mapping an *n*-dimensional uniform dependence algorithm with an arbitrary bounded convex index set onto a linear processor array, assuming that a linear schedule is given a priori.

Keywords: uniform dependence algorithms, linear schedule, allocation vector, norm, space optimal.

1. INTRODUCTION

Usually the performance of an algorithm on a processor array is measured by two cost parameters: the *space* cost and the *time* cost. The space cost is the number of processing elements (PEs) used to implement the algorithm on the processor array, and the time cost is the total execution time needed to execute the algorithm on the processor array. The space cost is directly determined by the *PE allocation function*, and the time cost is determined by the *scheduling function*. In this paper, we address the space-optimal mapping problem of uniform dependence algorithms onto linear processor arrays.

Uniform dependence algorithms [1] 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 constant data dependence vectors. The index set of an algorithm is a finite convex subset of Z^n [2]. The minimal convex polytope or convex hull R bounding the index set is usually a nondegenerated convex polytope in R^n. We call n the dimension of the index set and the algorithm an n-dimensional uniform dependence algorithm.

Received October 4, 1996; accepted May 3, 1997
Communicated by Jang-Ping Sheu.
[†] This research was supported by the National Science Council of the Republic of China under Grant No. NSC86-2213-E-009-017.

Previous research on space-optimal mapping of n-dimensional uniform dependence algorithms onto k-dimension processor arrays can be divided into two categories. The first category focuses on the mapping of n-dimensional algorithms onto $(n-1)$-dimensional processor arrays. There are two types of space-optimal designs for this group: nonlinear space-optimal design and linear space-optimal design. A nonlinear space-optimal design uses nonlinear processor assignment such that the number of PEs used is equal to the maximum number of simultaneously executable index points allowed by the given linear schedule [3-8]. A linear space-optimal design uses linear processor assignment such that the number of PEs used is equal to the minimum number of PEs that such a linear projection function can give [9,10].

The second category focuses on the mapping of n-dimensional uniform dependence algorithms onto k-dimensional processor arrays, $k < n-1$. Most works dealing with optimal lower dimensional processor array design have focused on linear time-optimal and linear space-optimal designs. In [11, 12], methods to design asymptotically linear space-optimal arrays were proposed. In [13, 14], Ganapathy and Wah proposed a generalized parameter method (GPM) to synthesize lower dimensional systolic arrays from n-dimensional uniform recurrences. They proposed an enumeration search procedure to find an optimal design which is either linear time-optimal, linear space-optimal, or the optimal of the product of space and time. However, there are some drawbacks in using the GPM to find an optimal design for mapping an n-dimensional uniform dependence algorithm with an arbitrary bounded convex index set into a processor array. In the GPM method, the objective function must be derived by the user. In most cases, objective functions expressed in terms of the parameters of the GPM are difficult to derive since this depends on the size and the shape of the index set.

In this paper, we formulate the space-optimal objective function directly in terms of the PE allocation vector and propose an enumeration procedure for linear mapping of an n-dimensional uniform dependence algorithm with an arbitrary bounded convex index set onto a linear processor array. The enumeration procedure finds a space-optimal PE allocation vector for the mapping, assuming that a valid linear schedule has been given a prior. Based on this approach, a tool called SODTLA (Space-Optimal Design Tool for Linear Array) was developed to find a space-optimal PE allocation vector without intervention by the user. Given the specifications of the index set, the data dependence vectors, the linear schedule, and I/O space [15] for each variable of the algorithm, SODTLA automatically finds a linear space-optimal array with a minimum number of PEs used for a uniform dependence algorithm with an arbitrary bounded convex index set. In our approach, the problem of minimizing the number of PEs used is reduced to the problem of minimizing the norm (length) of the PE allocation vector S, where the norm is defined with respect to a dual body R^*, and R^* is derived from the index set J of the n-dimensional uniform dependence algorithm. Two lower bounds on the norm of PE allocation vectors, one for an arbitrary convex index set and the other one for a constant-bounded index set [16], are also derived for use in the enumeration procedure.

In the next section, we will formulate the space-optimal linear array problem and describe the target linear array model. In Section 3, we will show that the number of PEs used for a PE allocation vector S is related to the norm of S. In Section 4, we will propose an enumeration procedure to find the space-optimal allocation vector. We also will derive some lower bounds on the norm of PE allocation vectors in this section. In addition, the maximum bounding box which bounds the values of the entries of all feasible PE allocation vectors will also be derived here. Finally, some concluding remarks will be given in Section 5.

2. FORMULATION OF THE SPACE-OPTIMAL LINEAR ARRAY PROBLEM

The mapping of an n-dimensional uniform dependence algorithm onto a 1-dimensional processor array can be considered as a linear transformation problem. The linear transformation is represented by a $2 \times n$ matrix, the first row of which denotes the *linear* scheduling vector (Λ), and the second row of which constitutes the *linear* PE allocation vector (S). This linear transformation maps the index set of a uniform dependence algorithm to a 1-dimensional time domain and a 1-dimensional space domain. The computation of an index pointin $\bar{x}$ in index set J is scheduled for execution on the PE at location $S\bar{x}$ at time $\Lambda\bar{x}$.

The linear array model used in this paper is depicted as follows. We assume that the linear array model has enough processors and data links required to implement the n-dimensional uniform dependence algorithm. For each data dependence vector $\bar{d}$ in an n-dimensional uniform dependence algorithm, there exists a directed data link between any two adjacent PEs at location p and at location $p + 1$. Assume that the evaluation of a computation by a PE takes unit time. Then, the number of buffers on the link(s) of length $|S\bar{d}|$ is $\Lambda\bar{d} - 1$. This linear array model is the same as that in [11, 13-15, 17, 18], which represents the typical properties of VLSI processor arrays, namely, simple and regular interconnection patterns between PEs. Since this model does not allow data broadcasting, the distance traveled by a data token along data link(s) must be less than the number of buffers on the data link(s), i.e., $|S\bar{d}| \leq \Lambda\bar{d}$ for each data dependence vector $\bar{d}$.

Observe that the PE allocation vector S is the normal vector of the parallel hyperplanes covering the index set J, and that $S\bar{x}$ represents an index of a PE executing the computation of the index point $\bar{x} \in J$. Given the PE allocation vector S, to count the number of parallel hyperplanes covering J, the greatest common divisor of the entries of S must be equal to one. Also, notice that the number of PEs used to implement an n-dimensional uniform dependence algorithm for a given PE allocation vector S is equal to $p(S) = 1 + \max_{\bar{x}_1, \bar{x}_2 \in J} S(\bar{x}_1 - \bar{x}_2)$.

When mapping an n-dimensional uniform dependence algorithm onto a linear array, a *correct* linear transformation $\begin{bmatrix} \Lambda \\ S \end{bmatrix}$ must satisfy the following conditions:

Precedence Constraint: For all data dependence vectors $\bar{d}$, $\Lambda\bar{d} \geq 1$. This means that a computation is executed only after all the computations on which it depends have been executed.

Computation constraint: For any two index points $\bar{x}_1, \bar{x}_2 \in J$, $\bar{x}_1 \neq \bar{x}_2$ if and only if $\Lambda\bar{x}_1 \neq \Lambda\bar{x}_2$ or $S\bar{x}_1 \neq S\bar{x}_2$. This means that no two computations are executed at the same PE at the same time step.

Data Link Constraint: There is no data collision in the data links of the linear processor array.

To identify a linear transformation without violating the computation constraint and the data link constraint, various closed form conditions were derived in [11, 13-19]. However, no general closed form conditions were derived for an arbitrary bounded convex index set in their papers. Thus, we use the unified approach proposed in [15] for the checking of the computation constraint and the data link constraint. It is the same as the one of checking the existence of non-zero integer solutions $\bar{x}$ to a generic problem:

$$B\bar{x} = 0$$

$$-\bar{a} < \bar{x} < \bar{a}\ . \tag{2.1}$$

In the case of checking a computation constraint, B is the linear transformation matrix T, and $-\bar{a} < \bar{x} < \bar{a}$ specifies the size of the index set J of the algorithm. In the case of checking a data link constraint, B is a matrix whose columns are the displacement (i.e., directional distances) of the data elements in the I/O space along each dimension, and $-\bar{a} < \bar{x} < \bar{a}$ specifies the size of the I/O space. To elaborate, we formulate the space-optimal linear array problem as follows.

Space-optimal linear array problem: Given an n-dimensional uniform dependence algorithm with index set J, dependence matrix $D = [\bar{d}_1, \ldots, \bar{d}_m]$, I/O space for each variable, and a linear schedule vector Λ, find a PE allocation vector $S \in Z^{1\times n}$ such that it minimizes

$$\max S(\bar{x}_1 - \bar{x}_2)$$

$$\text{subject to} \begin{cases} \Lambda D \geq 1 \\ \text{Solving the generic problem (2.1) to check} \\ \text{the computation constraint and the data link constraint} \\ \gcd(s_1, \ldots, s_n) = 1 \\ \left| S\bar{d}_i \right| \leq \Lambda\bar{d}_i,\ i = 1, \ldots, m \\ \bar{x}_1, \bar{x}_2 \in J \\ S \in Z^{1\times n} \end{cases}$$

where the index set $J = \{x \mid Ax \leq b, A \in Z^{a\times n}, b \in Z^a, x \in Z^n\}$ is a convex polytope.

In the following discussion, we will call a PE allocation vector a *feasible* allocation vector if $\begin{bmatrix} \Lambda \\ S \end{bmatrix}$ is a correct mapping for a given linear schedule Λ. In the next section, we will show that the number of PEs used for an allocation vector S is related to a norm defined with respect to a dual body R^*, which is derived from index set J.

3. DISTANCE FUNCTION AND PE ALLOCATION VECTOR

First, we will present the concept of a norm used in this paper. A norm is used to measure the length of a vector. Usually, a norm can be defined by the concept of *Distance function*. A distance function f of a vector $\bar{x}$ with respect to a convex, compact and symmetric set R^* is defined as follows [20, p. 110]:

$$f(\bar{x}) = \min_l\{l > 0 \mid \bar{x}/l \in R^*\} \ . \tag{3.1}$$

Geometrically, $f(\bar{x})$ is the minimum scaling factor for R^* to contain the vector $\bar{x}$. From this definition of $f(\bar{x})$, we see that the minimum value of l occurs when $\bar{x}/l$ is the point on the boundary of R^*. For example [21], the distance function $f(\bar{x})$ of a vector $\bar{x} = (x_1, \ldots, x_n)^T$ with respect to $R^* = \{(r_1, \ldots, r_n) | r_1| \ldots + | r_n \le 1\}$ in R^n is the l_1-norm $||x|| = \Sigma_{i=1}^{n} | x_i|$. See Fig. 1 for the case $n = 2$.

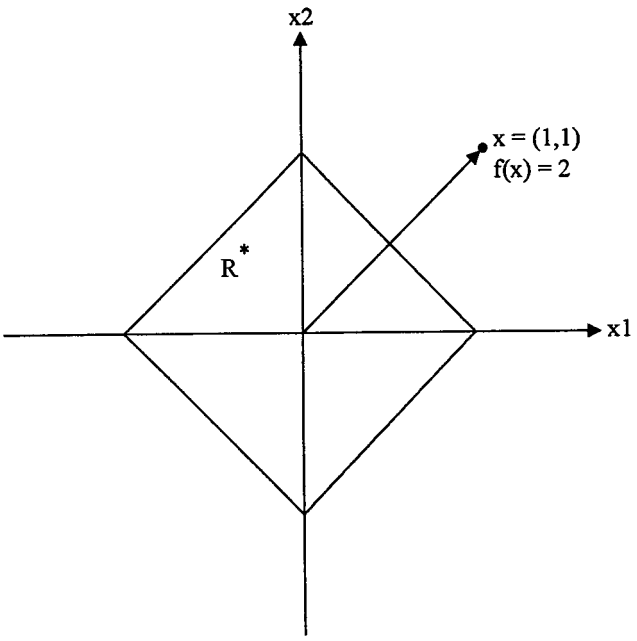

Fig. 1. The distance function $f((x_1, x_2)^T) = \Sigma_{i=1}^{2} | x_i|$ or the l_1-norm with respect to $R^* = \{(r_1, r_2) \mid -1 \le r_1 + r_2 \le 1, -1 \le r_1 - r_2 \le 1\}$.

Given the index set J, the R^* of the distance function $f(\bar{x})$ can be derived as follows. Let R be the minimal convex polytope bounding index set J of a uniform dependence algorithm. The *difference body* of R is defined as

$$(R - R) = \{(\bar{x}_1 - \bar{x}_2) \mid \bar{x}_1, \bar{x}_2 \in R\} . \tag{3.2}$$

The *dual body* R^* of $(R - R)$ is convex and symmetric about the origin and is defined as

$$R^* = \{\bar{\eta} \mid \bar{x}^T \bar{\eta} \leq 1, \ \forall \bar{x} \in (R - R)\} . \tag{3.3}$$

In [9], Wong and Delosme used $f(\bar{x})$ to derive the upper bound of the length of the optimal projection vector.

Observe that the number of PEs used to implement an n-dimensional uniform dependence algorithm for a given PE allocation vector S is equal to $p(S) = 1 + \max_{\bar{x}_1, \bar{x}_2 \in J} S(\bar{x}_1 - \bar{x}_2)$. In [22], we derived that $f(\Lambda) = \max_{\bar{x}_1, \bar{x}_2 \in J} \Lambda(\bar{x}_1 - \bar{x}_2)$ for a given linear schedule vector Λ. Thus, $f(S) = \max_{\bar{x}_1, \bar{x}_2 \in J} S(\bar{x}_1 - \bar{x}_2)$ for a given PE allocation vector S. Since S, $\bar{x}_1$, and $\bar{x}_2$ are integer vectors, $f(S)$ is an integer number. Therefore, $f(S) + 1$ is the number of PEs used for the PE allocation vector S, where the greatest common divisor of its entries is equal to one. We will give an example to illustrate the above argument.

Example 1: In the LU decomposition problem, a given matrix C is decomposed into $C = L \cdot U$, where L is a lower triangular, and U is an upper triangular matrix. This problem can be formulated as a 3-dimensional uniform dependence algorithm [23] with index set $J = \{(i, j, k) \mid 1 \leq i, j, k \leq N, 0 \leq i - k \leq N - 1, 0 \leq j - k \leq N - 1\}$ and dependence matrix $D = [\bar{d}_1 \ \bar{d}_2 \ \bar{d}_3]$, where $\bar{d}_1^T = (1, 0, 0)$, $\bar{d}_2^T = (0, 1, 0)$ and $\bar{d}_3^T = (0, 0, 1)$. In Fig. 2, we show the minimal convex polytope R bounding the index

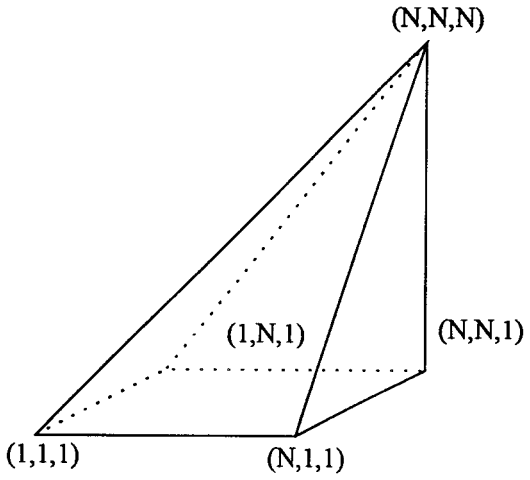

Fig. 2. The minimal convex polytope R bounding the index set J of the LU-decomposition problem.

set J. Notice that the index set J is the integer points in R. In Fig. 4, we show the difference body $(R - R)$ of R, and the dual body R^* is shown in Fig. 5. Let allocation vectors $S_1 = (1, 1, 1)$ and $S_2 = (2, 2, 2)$. From Fig. 3, we see that the number of hyperplanes (PEs) with normal vector S_1 covering the index points of index set J is equal to 9 + 1 = 10. Since $\bar{\xi}^T = (1/9.\ 1/9,\ 1/9)$ is the intersection point of vector S_1 and the boundary of R^*, we can find $f(S_1) = 9$ from Fig. 5 and check that $p(S_1) = 1 + f(S_1)$. For S_2, the number of hyperplanes is numbered in the sequence 0, 2, 4, 6, 8, 10, 12, 14, 16, 18. Therefore, $f(S_2) = 18$ and the number of hyperplanes covering the index points of index set J is 19, which is incorrect. Consequently, S_2 (whose components have a greatest common divisor not equal to one) will not be considered as a PE allocation vector. End of example.

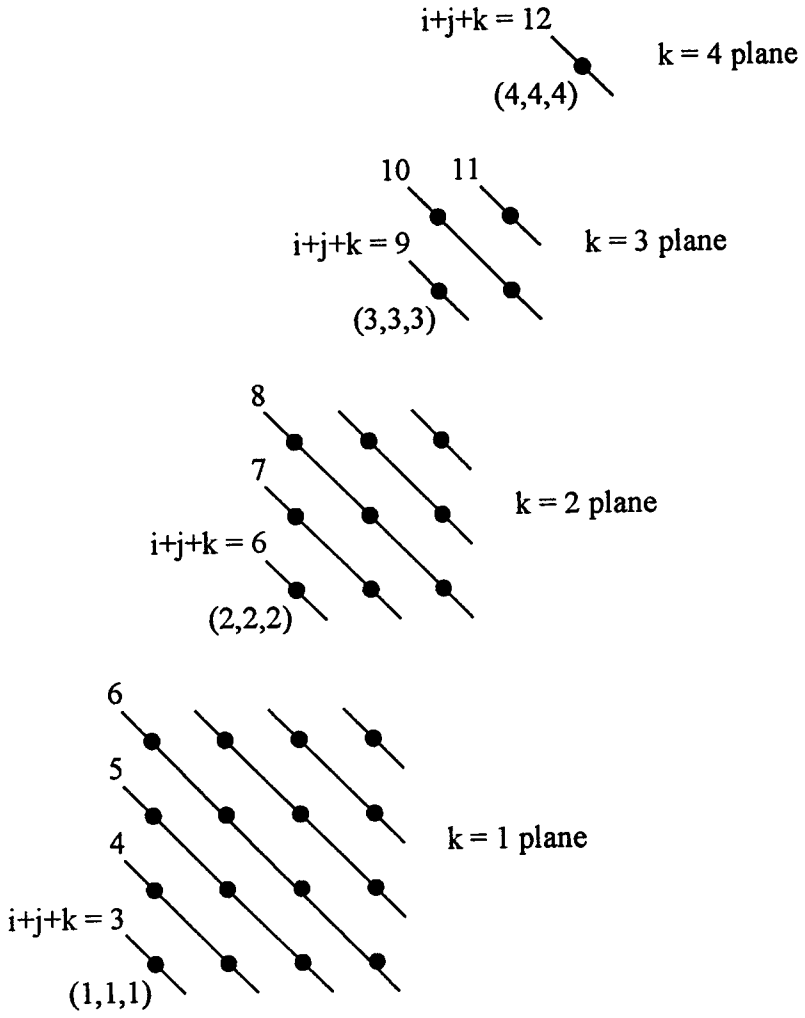

Fig. 3. The index set of the LU-decomposition problem ($N = 4$) and the hyperplanes with normal vector $S_1 = (1, 1, 1)$.

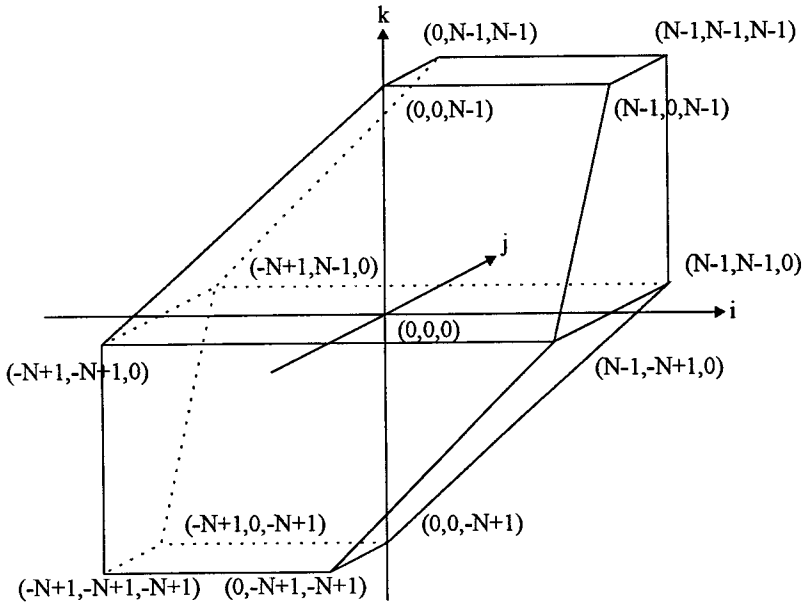

Fig. 4. The difference body ($R - R$) of the convex polytope R in Fig. 2.

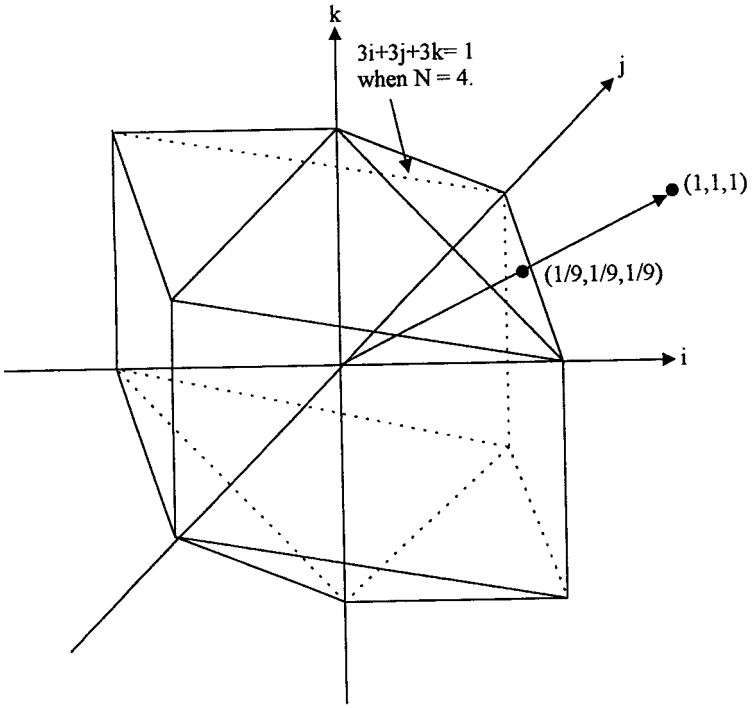

Fig. 5. The dual body R^* of the difference body ($R - R$).

Since computational conflict free and link conflict free checking does not have a closed linear form expression, we use an enumeration procedure to find the space-optimal PE allocation vector S. Notice that we only need to enumerate the PE allocation vectors S with the greatest common divisor of the entries being equal to one. In the next section, we will show how the PE allocation vectors for the n-dimensional uniform dependence algorithm can be enumerated.

4. FINDING AN OPTIMAL ALLOCATION VECTOR

In this section, we will propose an enumeration procedure to find a space-optimal allocation vector S. Since the number of PEs used for an allocation vector S is related to the norm of S with respect to R^*, we need to enumerate the allocation vectors S in an increasing order of their norms in order to find the space-optimal allocation vector. In the following discussion, we will give the method used to enumerate the PE allocation vectors S. To simplify the explanation, we use the following notation: if polyhedron $P = \{\bar{x} \mid A\bar{x} \le \bar{b}, \bar{x} \in R^n\}$, then $k \cdot P$ denotes the polyhedron $\{\bar{x} \mid A\bar{x} \le k\bar{b}, \bar{x} \in R^n\}$. Since $f(S)$ is the minimum scaling factor for dual body R^* to contain vector S, S is on the boundary of the convex polytope $f(S) \cdot R^*$. Therefore, an integer point S with norm $f(S) = k$ is on the boundary of the convex polytope $k \cdot R^*$.

4.1 Minimum Bounding Box

Since an integer point S with norm $f(S) = k$ is on the boundary of the convex polytope $k \cdot R^*$, we need to enumerate all such integer points on the boundary of $k \cdot R^*$. However, enumerating those integer points on the boundary of convex polytope $k \cdot R^*$ is not an easy task. We will describe a method for enumerating these integer points as follows. Assume that R^* has h boundary hyperplanes $\bar{v}_o^T\bar{x} = 1$, $i = 1, \ldots, h$. Clearly, the integer points on the boundary of $k \cdot R^*$ are on the hyperplanes defined by equations $\bar{v}_o^T\bar{x} = k$, $i = 1, \ldots, h$, but only a small fraction of the integer points on the boundary hyperplane $v_o^T\bar{x} = k$ have norm equal to k. Therefore, we must limit the range of enumeration in order to prune most of the integer points with norm greater than k. Since $k \cdot R^*$ is a convex polytope, we can use a minimum bounding box $B = \{(x_1, \ldots, x_n) \mid l_i \le x_i \le u_i, i = 1, \ldots, n; u_i > l_i\}$ bounding the convex polytope $k \cdot R^*$ to limit the enumeration of all integer points on the boundary of $k \cdot R^*$. Let B_1 be the minimum bounding box of R^*. Since $k \cdot R^*$ can be obtained by enlarging R^* by a factor k, $k \cdot R^*$ has the minimum bounding box $k \cdot B_1$. The minimum bounding box B_1 of R^* can be easily obtained from the difference body $(R - R)$ due to the vertex-boundary relationship between the difference body $(R - R)$ and its dual body R^*. The vertex-boundary relationship can be derived by means of the following lemmas. These lemmas are stated here without proof; for details see [22]. Lemma 4.1 concerns the construction of the difference body $(R - R)$ from a convex polytope R.

Lemma 4.1: *Let* $R = \{\bar{x} \mid l_i \le \bar{a}_i^T\bar{x} \le u_i, \bar{a}_i \in Z^n, \bar{x} \in R^n, l_i, u_i \in Z, u_i > l_i, i = 1, \ldots, q\}$. Let $(R - R) = \{(\bar{x} - \bar{x}') \mid \bar{x}, \bar{x}' \in R\}$ *be the difference body of* R. *Then* $(R - R) = \{\bar{x} \mid l_i - u_i \le \bar{a}_i^T\bar{x} \le u_i - l_i, \bar{a}_i \in Z^n, \bar{x} \in R^n, l_i, u_i \in Z, u_i > l_i, i = 1, \ldots, q\}$.

Lemma 4.2 states that there is a one-to-one correspondence between a vertex of the difference body of $(R - R)$ and a boundary on the dual body R^* of $(R - R)$.

Lemma 4.2: *Let $(R - R)$ be the difference body of a convex set R. Let $\bar{v}_1, \ldots, \bar{v}_h$ be the h vertices of the convex body $(R - R)$.* Let $C = \{\bar{\eta} \mid \bar{v}_i^T \bar{\eta} \leq 1, i = 1, \ldots, h,$ *where $\bar{v}_i$ is a vertex of $(R - R)$*$\}$. *Let* $R^* = \{\bar{\eta} \mid \bar{x}^T \bar{\eta} \leq 1, \forall \bar{x} \in (R - R)\}$. Then $C = R^*$.

From Lemma 4.2, we see that the dual body R^* is the convex polytope defined by a set of linear inequalities, $\bar{v}_i^T \bar{\eta} \leq 1$, $i = 1, \ldots, h$. The coefficients $\bar{v}_i$ of the linear inequalities correspond to the vertices of the difference body $(R - R)$. Therefore, the dual body $k \cdot R^*$ is the convex polytope defined by a set of linear inequalities, $\bar{v}_i^T \bar{\eta} \leq k$, $i = 1, \ldots, h$. The following lemma states that the dual R^{**} of the dual body R^* is the difference body $(R - R)$ with respect to R^*.

Lemma 4.3: *Let $(R - R)$ be the difference body of a convex set R. Let* $R^* = \{\bar{\eta} \mid \bar{x}^T \bar{\eta} \leq 1, \forall \bar{x} \in (R - R)\}$ *be the dual body of $(R - R)$. Let* $R^{**} = \{\bar{\eta}' \mid \bar{\eta}^T \bar{\eta}' \leq 1, \forall \bar{\eta} \in R^*\}$ *be the dual body of R^*. Then $(R - R) = R^{**}$.*

From Lemma 4.3 and Lemma 4.2, we see that the difference body $(R - R)$ is a convex polytope defined by a set of linear inequalities, $\bar{w}_i^T \bar{x} \leq 1$, $i = 1, \ldots, a$, where the $\bar{w}_i$'s correspond to the vertices of the dual body R^*. Therefore, the minimum bounding box B_1 can be easily computed as follows. Let $u_i = \max_j\{w_{ij} \mid w_{ij}$ is the ith entry of vector $\bar{w}_j\}$, and let $l_i = \min_j\{w_{ij} \mid w_{ij}$ is the ith entry of vector $\bar{w}_j\}$. Observe that, from Lemma 4.1, there is an even number of linear inequalities $-1 \leq \bar{w}_i^T \bar{x} \leq 1$ defining $(R - R)$. Thus, $l_i = -u_i$. Therefore, the minimum bounding box B_1 of R^* is the set $\{(x_1, \ldots, x_n) \mid -u_i \leq x_i \leq u_i, i = 1, \ldots, n\}$, and the minimum bounding box $k \cdot B_1$ of $k \cdot R^*$ is the set $\{(x_1, \ldots, x_n) \mid -k \cdot u_i \leq x_i \leq k \cdot u_i, i = 1, \ldots, n\}$.

Using the concept of the minimum bounding box $k \cdot B_1$ of the dual body $k \cdot R^*$, the PE allocation vectors S can be enumerated as follows. The minimum bounding box $k \cdot B_1$ of $k \cdot R^*$ is constructed step by step with value k in an increasing order. First, an initial minimum bounding box $k_1 \cdot B_1$ is obtained, where k_1 is the lower bound of the number of PEs used for the n-dimensional uniform dependence algorithm. All the integer points within the bounding box $k_1 \cdot B_1$ are enumerated. If there does not exist a *feasible* integer point (feasible PE allocation vector), then another minimum bounding box $k_2 \cdot B_1$ of $k_2 \cdot R^*$ is constructed. Then, the integer points in the region $k_2 \cdot B_1 - k_1 \cdot B_1$ are enumerated. The enumeration of the integer points in $k_{i+1} \cdot B_1 - k_i \cdot B_1$, $i = 1, \ldots$ is repeated until a feasible integer point is found. When a feasible integer point S with the smallest norm $f(S)$ among the other feasible integer points is found in $k_{i+1} \cdot B_1 - k_i \cdot B_1$, the norm $f(S)$ is compared to k_{i+1}. If $k_i < f(S) \leq k_{i+1}$, then the optimal allocation vector S is found. Otherwise, a minimum bounding box $f(S) \cdot B_1$ is constructed, and all integer points S_1 in $f(S) \cdot B_1 - k_{i+1} \cdot B_1$ with norm $f(S_1)$ less than $f(S)$ are enumerated. This is due to the fact that the minimum bounding box $k_{i+1} \cdot B_1$ contains all integer points S_2 with $f(S_2) \leq k_{i+1}$ and may contain an integer point S_3 with norm $f(S_3) > k_{i+1}$. Therefore, an additional check is necessary to find the optimal allocation vector.

Since the integer points in region $k_{i+1} \cdot B_1 - k_i \cdot B_1$ need to be enumerated, we partition this bounded region into subregions as follows. Assume that $k_i \cdot B_1 = \{(x_1, \ldots, x_n) \mid l_i \le x_i \le u_i, i = 1, \ldots, n; u_i > l_i\}$ and $k_{i+1} \cdot B_1 = \{(x_1, \ldots, x_n) \mid l_i' \le x_i \le u_i', i = 1, \ldots, n; u_i' > l_i'\}$. Since the minimum bounding box $k_i \cdot B_1$ of $k_i \cdot R^*$ is a hyperparallelepiped, there are exactly $2n$ boundary hyperplanes $x_i = l_i$ *and* $x_i = u_i$, $i = 1, \ldots, n$ for $k_i \cdot B_1$. Therefore, the bounded region $k_{i+1} \cdot B_1 - k_i \cdot B_1$ can be partitioned into subregions SB_{2r-1} and SB_{2r}, $r = 1, \ldots, n$ according to the $2n$ boundary hyperplanes, where $SB_{2r-1} = \{(x_1, \ldots, x_j, \ldots, x_n) \mid l_j \le x_j \le u_j$, if $j < r$; $u_j + 1 \le x_j \le u_j'$, if $j = r$; $l_j' \le x_j \le u_j'$, if $j > r\}$ and $SB_{2r} = \{(x_1, \ldots, x_j, \ldots, x_n) \mid l_j \le x_j \le u_j$, if $j < r$; $l_j' \le x_j \le l_j - 1$, if $j = r$; $l_j' \le x_j \le u_j'$, if $j > r\}$, $r = 1, \ldots, n$. Obviously, $SB_p \cap SB_q = \emptyset$, $p \ne q$, and $p, q \in [1, 2n]$, and $k_i \cdot B_i \cap SB_p = \emptyset$, $p = 1, \ldots, 2n$. Furthermore, the union of $k_i \cdot B_i$ and SB_p, $p = 1, \ldots, 2n$ is equal to $k_{i+1} \cdot B_1$. Therefore, we can enumerate the integer points in $k_{i+1} \cdot B_1 - k_i \cdot B_1$ by enumerating the integer points in the $2n$ subregions SB_p, $p = 1, \ldots, 2n$. Since the linear arrays with respect to allocation vectors S and $-S$ are the same except that the directions of the data links and the numbering of the PE indices are reversed, only half of the integer points in $k_{i+1} \cdot B_1 - k_i \cdot B_1$ need to be enumerated. Since $l_i = -u_i$, $i = 1, \ldots, n$, enumerating the integer points in n subregions SB_{2i-1}, $i = 1, \ldots, n$, is sufficient.

4.2 Lower Bounds on the Norm of PE Allocation Vectors

In the following discussion, we will derive lower bounds on the number of PEs used (or lower bounds on the norm of the PE allocation vectors) for an n-dimensional uniform dependence algorithm. First, we will derive a lower bound for the algorithm with an arbitrary bounded convex index set. Then, a better lower bound will be derived for the algorithm with a constant-bounded index set [16]. The constant-bounded index set is defined as

$$J = \{(j_1, \ldots, j_n) \mid 0 \le j_i \le \mu_i, j_i \in Z, \mu_i \in N^+, i = 1, \ldots, n\}.$$

We call $\mu = \min\{\mu_i \mid i = 1, \ldots, n\}$ the slenderness of the index set.

4.2.1 Lower Bound for Arbitrary Bounded Convex Index Set

In the following discussion, a lower bound for the algorithm with a arbitrary bounded convex index set will be derived. First, we will show that every integer point S on the boundary of $k \cdot R^*$ with equation $\Sigma_{j=1}^{n} \alpha_j x_j = k$, $\bar{\alpha} \in Z^n$ and $k \in Z$ can be expressed as a linear combination of some vectors. Clearly, this is the problem of finding all the integer solutions of the diophantine equation $\Sigma_{j=1}^{n} \alpha_j x_j = k$. Using the following lemma in [24, p.192], all the integer solutions of the diophantine equation $\Sigma_{j=1}^{n} \alpha_j x_j = k$ can be found as follows.

Lemma 4.4: *Let* $Sol = \{\bar{x} \in Z^n \mid \Sigma_{j=1}^{n} \alpha_j x_j = k, k \in Z\}$. *If* $gcd(\alpha_1, \ldots, \alpha_n) \mid k$, *then there exist n affinely independent points in Sol. Furthermore,* $Sol = \{\bar{x} \mid \bar{x} = \frac{k}{H} C_1 + C_2 \bar{w}, \bar{w} \in Z^{n-1}, H = \gcd(\alpha_1, \ldots, \alpha_n), C_1 \in Z^n, C_2 \in Z^{n \times (n-1)}\}$.

Concerning the property of the integer solutions of the diophantine equation, we rephrase a theorem [24, p, 191] for the set *Sol* in Lemma 4.4.

Theorem 4.5: $Sol \neq \emptyset$ if and only if $k/H \in Z$.

From Theorem 4.5 and Lemma 4.4, we can derive a lower bound on the norm of the PE allocation vectors (or a lower bound on the number of PEs used) for an n-dimensional uniform dependence algorithm as follows. Assume that $k \cdot R^*$ has h boundary hyperplanes defined by $\Sigma_{j=1}^{n} a_{ij}x_j = k$, a_{ij}, $k \in Z$, $i = 1, \ldots, h$. Using Lemma 4.4, all the integer solutions of $\Sigma_{j=1}^{n} a_{ij}x_j = k$ are the set $Sol = \{\bar{x} \mid \bar{x} = \frac{k}{H_i}C_1 + C_2\bar{w}, \bar{w} \in Z^{n-1}\}$, where $H_i = \gcd(a_{i1}, \ldots, a_{in})$. By Theorem 4.5, we have $Sol \neq \emptyset$ if and only if $\frac{k}{H_i} \in Z$. As a consequence, the dual body $k \cdot R^*$ may have integer points S with norm equal to k on the boundary if $\frac{k}{H_i} \in Z$ for one of the boundary hyperplane $\Sigma_{j=1}^{n} a_{ij}x_j = k$. Therefore, the norm of any PE allocation vector S is greater than or equal to the smallest positive integer k such that $\frac{k}{H_i} \in Z$. Notice that if $\gcd(s_1, \ldots, s_n) \neq 1$ for the PE allocation vector S, we can find a vector $\frac{S}{\gcd(s_1, \ldots, s_n)}$ the norm of which is smaller than that of S. This implies that an allocation vector S whose components have a greatest common divisor not equal to one will not be considered as a PE allocation vector. From the above discussion, we have the following theorem.

Theorem 4.6: *Let R^* be the dual body for index set J. Assume that R^* is defined by equations $\Sigma_{j=1}^{n} a_{ij}x_j \leq 1$, $i = 1, \ldots, h$. Let $H_i = \gcd(a_{i1}, \ldots, a_{in})$. Then, a lower bound on the number of PEs used for the n-dimensional uniform dependence algorithm is equal to* $\min\{H_1, \ldots, H_h\} + 1$.

Theorem 4.6 states that the lower bound of the number of PEs used for the n-dimensional uniform dependence algorithm can be found from the coefficient vectors of the boundary equations of its dual body R*. The following example will be used to show that the lower bound on the number of PEs used for the LU-decomposition algorithm is equal to N.

Example 2: The dual body R^* of the LU-decomposition algorithm in Example 1 is defined by the following six boundary equations: $(N-1, N-1, N-1)\bar{x} = 1$, $(0, N-1, N-1)\bar{x} = 1$, $(-N+1, N-1, 0)\bar{x} = 1$, $(N-1, N-1, 0)\bar{x} = 1$, $(0, 0, N-1)\bar{x} = 1$, $(N-1, 0, N-1)\bar{x} = 1$. Clearly, all of the greatest common divisors of the above coefficient vectors of the boundary equations are equal to $N-1$. Since $\min(N-1, \ldots, N-1) = N-1$, the lower bound on the PEs used for the LU-decomposition algorithm is equal to N. End of example.

4.2.2 Lower Bound for the Constant-Bounded Index Set

Since many n-dimensional uniform dependence algorithms (such as matrix multiplication, the transitive closure problem, etc.) have a constant-bounded index set, we will derive a better lower bound for these algorithms to reduce the time needed to search the optimal allocation vector S. Notice that a linear transformation $T = \begin{bmatrix} \Lambda \\ S \end{bmatrix}$ must satisfy the precedence constraint, computation constraint, and data link constraint when mapping an n-dimensional uniform dependence algorithm onto a linear array. Thus, we can make use of these constraints to derive a lower bound on the number of PEs used. In this paper, the computation constraint is used to help us derive a lower bound on the number of PEs used for those algorithms having a constant-bounded index set. First, a theorem is rephrased from [19] which is helpful in deriving the lower bound.

Lemma 4.7: Let $J = \{(j_1, \ldots, j_n) \mid 0 \le j_i \le \mu_i, j_i \in Z, \mu_i \in N^+, i = 1, \ldots, n\}$ *be the constant-bounded index set of a uniform dependence algorithm. Let* $T = \begin{bmatrix} \Lambda \\ S \end{bmatrix}$ *be the mapping matrix, where* $\Lambda = (\lambda_1, \ldots, \lambda_n) \in Z^{1\times n}$, *and* $S = (s_1, \ldots, s_n) \in Z^{1\times n}$. *If T does not violate the computation constraint, then the following inequality is true*:

$$\Sigma_{i=1}^{n} |s_i| \ge (\mu + 1)^{n-2} / \Sigma_{j=1}^{n} |\lambda_i|,$$

where $\mu = min\{\mu_i \mid i = 1, \ldots, n\}$.

From Lemma 4.7, we can see that $\Sigma_{i=1}^{n} |s_i|$ is the l_1-norm. In terms of the concept of the distance function, it is known that the l_1-norm is defined with respect to a convex, compact, and symmetric set $R_1^* = \{(x_1, \ldots, x_n)^T \| x_1| + \ldots + |x_n| \le 1\}$ [21]. Notice that R_1^* is a convex polytope defined by 2^n linear inequalities. By choosing $l = ((\mu + 1)^{n-2} - 1)/\Sigma_{i=1}^{n} |\lambda_i|$, all the integer points in $l \cdot R_1^*$ are infeasible allocation vectors and need not be enumerated in our enumeration procedure. Let $(R - R)$ be the difference-body with respect to the constant-bounded index set $J = \{(j_1, \ldots, j_n) \mid 0 \le j_i \le \mu_i, j_i \in Z, \mu_i \in N^+, i = 1, \ldots, n\}$, and let R^* be the dual body of $(R - R)$. By Lemmas 4.1, 4.2, and 4.3, we can find the minimum bounding box B_1 of R^*, $B_1 = \{(x_1, \ldots, x_n) \mid -1/\mu_i \le x_i \le 1/\mu_i, i = 1, \ldots, n\}$. Since the integer points in $l \cdot R_1^*$ are infeasible allocation vectors, the lower bound k can be determined by finding a minimum bounding box $k \cdot B_1$ such that k is the maximum value for $k \cdot B_1$ contained in $l \cdot R_1^*$. Clearly, the 2^n vertices $\overline{v}$'s of the convex polytope B_1 are of the form $(\pm 1/\mu_1, \ldots, \pm 1/\mu_n)$. Observe that, when enlarging B_1 by a factor k to inscribe $l \cdot R_1^*$, all of the $k\overline{v}$'s are on the boundary of $l \cdot R_1^*$. Since the boundary of $l \cdot R_1^*$ can be denoted by equation $|x_1| + \ldots + |x_n| = l$, we have $k(\Sigma_{i=1}^{n} 1/\mu_i) = l$. Thus, the lower bound $k = l/(\Sigma_{i=1}^{n} 1/\mu_i)$. From the above discussion, we have the following theorem.

Theorem 4.8: Let $J = \{(j_1, \ldots, j_n) \mid 0 \le j_i \le \mu_i, j_i \in Z, \mu_i \in N^+, i = 1, \ldots, n\}$ *be the constant-bounded index set of a uniform dependence algorithm. Let* $T = \begin{bmatrix} \Lambda \\ S \end{bmatrix}$ *be the*

mapping matrix, where $\Lambda = (\lambda_1, \ldots, \lambda_n) \in Z^{1\times n}$, *and let* $S = (s_1, \ldots, s_n) \in Z^{1\times n}$. *The lower bound on the number of PEs used is equal to*

$$\left\lfloor ((\mu + 1)^{n-2} - 1) / ((\Sigma_{i=1}^{n} 1/\mu_i)(\Sigma_{i=1}^{n} |\lambda_i|)) \right\rfloor + 1,$$

where $\mu = min\{\mu_i \mid i = 1, \ldots, n\}$. *Furthermore, all integer points within* $k \cdot B_1$ *are infeasible allocation vectors, where* B_1 *is the boundary box of* R^* *with respect to the index set* J.

Notice that for $\mu_i = N$, $i = 1, \ldots, n$, the lower bound is equal to $\lfloor ((N + 1)^{n-2} - 1)(N/(n\Sigma_{i=1}^{n} |\lambda_i|)) \rfloor + 1$, which is of order $O(N^{n-1}/(n\Sigma_{i=1}^{n} |\lambda_i|))$. From Theorem 4.8, we see that the lower bound on the norm of the PE allocation vector will increase if the sum of the absolute values of the entries of the linear schedule Λ decreases. This means that the total execution time is expected to be shorter. Furthermore, given a linear schedule, the lower bound also increases if the dimension n of the index set increases. This means that the lower bound is bigger for mapping a higher dimensional uniform dependence algorithm onto a linear array.

4.3 Maximum Bounding Box of the Entries of PE Allocation Vectors

Since the array model does not allow data broadcasting, the distance traveled by a data token along data link(s) must be less than the number of buffers on the data link(s), i.e., $|S\bar{d}| \le \Lambda\bar{d}$ for each data dependence vector $\bar{d}$. Due to this property, there is a maximum bounding box B_{max} which bounds the values of the entries of the allocation vectors S. The bounding box can be obtained as follows. Let $\Lambda = (\lambda_1, \ldots, \lambda_n)$, $S = (s_1, \ldots, s_n)$, and dependence matrix $D = [\bar{d}_1 \ldots \bar{d}_m]$, where $\bar{d}_i^T = (d_{1i}, \ldots, d_{ni})$. For each $i = 1, \ldots, n$, let $u_i = max\{s_i \parallel S\bar{d}_j \mid \le \Lambda\bar{d}_j, j = 1, \ldots, m\}$, and let $l_i = min\{s_i \parallel S\bar{d}_j \mid \le \Lambda\bar{d}_j, j = 1, \ldots, m\}$. Then, the maximum bounding box B_{max} is the set $\{(x_1, \ldots, x_n) \mid l_i \le x_i \le u_i, i = 1, \ldots, n\}$.

4.4 The Enumeration Procedure

Now, we will present the enumeration procedure for finding a space-optimal allocation vector S as follows. In this procedure, the function *terminator*(x, y, B_{max}) is used to test whether there are feasible allocation vectors in the region $(y - x)$ which needs to be enumerated. If the function evaluates TRUE, then no allocation vectors need to be enumerated.

Procedure (Finding a space-optimal allocation vector *S*):

Input: Uniform dependence algorithm (J, D), where J is an arbitrary bounded convex index set; the I/O space of each variable; and a linear schedule vector $\Lambda \in Z^{1\times n}$.
Output: An integer row vector S which is the space-optimal allocation vector.
Step 1: Construct the dual body R^* from index set J (by Lemmas 4.1 and 4.2).
Step 2: Compute the minimum bounding box B_1 of R^*. Compute the maximum

bounding box B_{max} for the values of the entries of the allocation vectors.

Step 3: Compute the lower bound, $k_1 + 1$, on the number of PEs used by Theorem 4.6 for this n-dimensional uniform dependence algorithm.

Step 4: If *terminator*($NULL$, $k_1 \cdot B_1$, B_{max}) = TRUE, then go to Step 12; else for each integer point S in $k_1 \cdot B_1$ for $k_1 \cdot R^*$, check whether $\begin{bmatrix} \Lambda \\ S \end{bmatrix}$ is a correct mapping.

Step 5: If feasible allocation vectors are found in Step 4, then set $i = 0$ and go to Step 9.

Step 6: Set $i = 1$.

Step 7: Set $k_{i+1} = k_i + \Delta_k$, where $\Delta_k + 1$ is the lower bound computed by Theorem 4.6. Construct $k_{i+1} \cdot B_1$ for $k_{i+1} \cdot R^*$. If *terminator*($k_i \cdot B_1$, $k_{i+1} \cdot B_1$, B_{max}) = TRUE, then go to Step 12; else for each integer point in $k_{i+1} \cdot B_1 - k_i \cdot B_1$, check whether $\begin{bmatrix} \Lambda \\ S \end{bmatrix}$ is a correct mapping.

Step 8: If no feasible allocation vector S is found in Step 7, then set $i = i + 1$, and go to Step 7.

Step 9: Let S be the feasible allocation vector such that $f(S)$ is the smallest one of these feasible allocation vectors. If $f(S) \leq k_{i+1}$, then the optimal allocation vector is S and stop.

Step 10: Construct $f(S) \cdot B_1$ for $f(S) \cdot R^*$. For each integer point S_1 in $f(S) \cdot B_1 - k_{i+1} \cdot B_1$ with $f(S_1) < f(S)$, check whether $\begin{bmatrix} \Lambda \\ S \end{bmatrix}$ is a correct mapping.

Step 11: If no feasible allocation vector S_1 is found in Step 10, then the optimal allocation vector is S. Otherwise, the optimal allocation vector is S_2 with $f(S_2)$ being the smallest one among the feasible allocation vectors S_1 found in Step 10. Set $S = S_2$. Stop.

Step 12: No optimal allocation vector for this linear schedule Λ. Stop.

This enumeration procedure is applicable to a uniform dependence algorithm with an arbitrary bounded convex index set. For a uniform dependence algorithm with a constant-bounded index set, a better lower bound can be computed by using Theorem 4.8 and used in Step 3. Since by Theorem 4.8, we know that all the integer points in $k_1 \cdot B_1$ are infeasible, Step 4 and Step 5 can be omitted.

In this enumeration procedure, Step 1 can be computed by finding all the vertices of the difference body $(R - R)$. This requires $O(\binom{a}{n} \cdot n^3)$ time complexity if the convex polytope R is defined by a linear inequalities. Furthermore, the difference body $(R - R)$ can be found by using Lemma 4.1 with $O(a)$ time complexity. In Step 2, the minimum bounding box B_1 of R^* can be found with $O(a)$ time complexity by using the vertex-boundary relationship between the difference body and dual body. Assume that there are h vertices of $(R - R)$. In Step 3, we need $h \cdot O(L)$ time complexity, where $O(L)$ is the time needed to compute the greatest common divisor of the entries of the coefficient vector of the boundary equation of R^* and is polynomial [24, p.193] for a uniform dependence algorithm with an arbitrary bounded convex index set. The construction of the bounding box $k_i \cdot B_1$ is easily

obtained by simply multiplying the bounds l_i, u_i of each axis x_i of the bounding box B_1 of R^* by k_i. This requires only $2n$ steps. Since Δ_k is the minimum enlarging factor k for $k \cdot R^*$ to contain integer points, the value k_{i+1} in Step 7 is set to $k_i + \Delta_k$ so that the procedure can enumerate the integer points in an area of $k_{i+1} \cdot B_1 - k_i \cdot B_1$. Since checking the feasibility of a mapping $\begin{bmatrix} \Lambda \\ S \end{bmatrix}$ is the most time consuming operation and computing the norm $f(S)$ of the allocation vector S requires solving a linear program, we compute $f(S)$ only when $\begin{bmatrix} \Lambda \\ S \end{bmatrix}$ is a feasible mapping, and the allocation vector S with the smallest $f(S)$ among those feasible allocation vectors is kept for use in Step 9. The same argument is also applied to Step 10, and all the integer points in $f(S) \cdot B_1 - k_{i+1} \cdot B_1$ are enumerated in order to find the vector S_1 with the smallest $f(S_1)$ among the feasible allocation vectors in $f(S) \cdot B_1 - k_{i+1} \cdot B_1$. Actually, only half the integer points in region $k_{i+1} \cdot B_1 - k_i \cdot B_1$ are enumerated. Since the linear array model does not allow data broadcasting, most of the integer points will be pruned away by the constraint $|S\vec{d}| \leq \Lambda\vec{d}$ for each data dependence vector $\vec{d}$ of the algorithm.

The method proposed by Ganapathy and Wah [13] is an enumeration method. The search space which they considered was defined on a coordinate system with dependence vectors as basis vectors. In their method, displacement vectors $\bar{\kappa}$ are enumerated. As $\bar{\kappa}^T = SD$ (see their paper [13, p.277] for details) where S is the allocation vector and D is the dependence matrix, it is possible to enumerate an integer vector $\bar{\kappa}$, which results in a rational allocation vector S. Notice that a rational vector S implies that some index points are mapped to PE with its index being a rational number, which is unreasonable. Our method is also an enumeration method. However, our search space is defined on the standard basis. Furthermore, the integer allocation vectors S are enumerated directly.

Our enumeration procedure was implemented as a design tool called SODTLA. We have applied SODTLA to algorithms for the transitive closure problem provided in paper [13] and to algorithms for LU-decomposition and band matrix-matrix multiplication problems. The results obtained by SODTLA are shown in the Appendix.

5. CONCLUSIONS

In this paper, we have proposed an enumeration method to find a space-optimal allocation vector to map an n-dimensional uniform dependence algorithm to a linear processor array. Previous work on solving the space-optimal k-dimensional processor array problem needs to derive the space-optimal objective function beforehand by users. For different problems, formulation of the space-optimal objective function is difficult and depends on the size and shape of the index set. For a space-optimal linear processor array problem, the space objective function is captured by the concept of a norm in our method. The norm is defined with respect to a convex polytope R^*, where R^* is derived from the index set J of the algorithm. This concept is applicable to an arbitrary bounded convex index set and can be

adopted in order to avoid derivation of the space-optimal objective function by the user.

For some real-time applications, execution time may be a critical factor in designing a linear array. Taking this into consideration, a linear schedule the execution time of which satisfies the time requirement must be given beforehand. In this situation, our approach can help users find a linear array with minimum hardware complexity.

For use in the enumeration procedure, a lower bound on the norm of PE allocation vectors for an arbitrary bounded convex index set has been derived. A better lower bound for a constant-bounded index set has also been derived. How to derive a better lower bound for an arbitrary bounded convex index set from the computation constraint needs further study. A possible solution is to find an inscribed constant-bounded set with maximum slenderness. Then, a better lower bound for the arbitrary bounded index set can be obtained by using the constant-bounded set as the new index set. However, finding such an inscribed constant-bounded set is not an easy task.

APPENDIX

In this appendix, we describe our application of SODTLA to finding optimal allocation vectors for 20 algorithms with three problems which varied in size. The first nine algorithms were for the transitive closure problem. Algorithms 10 to 14 were for the LU-decomposition problem. Algorithms 15 to 20 were for the band matrix-matrix multiplication problem. We present the three problems in Table 1. The results found by SODTLA are listed in Table 2. Listed in Table 1 are the dependence matrix D, the index set J, and the I/O space Φ and the dependence vector ϑ_V associated with each variable V of the algorithm. Also listed in Table 1 is the dual body R^* corresponding to the index set J. In the band matrix-matrix multiplication problem, the bandwith of matrix A was specified by the parameters $p1$, $p2$, while for matrix B it was specified by $q1$, $q2$ and for matrix C it was specified by $r1$, $r2$. The bandwidth of matrix A was equal to $p1 + p2 - 1$, where $p1$ ($p2$) is the number of band elements in each row (column) counted from the diagonal element. The same definitions were also applied to the bandwidths $q1$, $q2$, $r1$, and $r2$ of matrices B and C. The parameters $r1$ and $r2$ of matrix C could be determined to be $r1 = min(N3, p1 + q1 - 1)$ and $r2 = min(N1, p2 + q2 - 1)$. Notice that it is not simple to express the dual body R^* of the band matrix-matrix multiplication problem in terms of the problem size parameters. Table 2 shows the optimal allocation vector S and the optimal number of PEs found by SODTLA for the given problem size parameters and linear schedule Λ. The first nine algorithms of the transitive closure problem were taken from Table 1 (p. 283) in a paper [13] by Ganapathy and Wah. The results of SODTLA are the same as those of the enumeration procedure of the GPM method proposed by Ganapathy and Wah for these nine algorithms.

Table 1. Problem description.

problem	problem size parameters	dependence matrix D	index set J	variable V, I/O spaces Φ_V, dependence vector ϑ_V	dual body R*
transitive closure $A_{N\times N}$	N		$1 \le i, j, k \le N$	$V = A$, $\Phi_A = \{1 \le i, j \le N\}$, $\vartheta_A = (-1, -1, 1)$	$-1 \le (N-1)(i+j+k) \le 1$, $-1 \le (N-1)(i+j-k) \le 1$, $-1 \le (N-1)(i-j+k) \le 1$, $-1 \le (N-1)(i-j-k) \le 1$
LU – decomposition $C_{N\times N} = L_{N\times N} \cdot U_{N\times N}$	N		$1 \le i, j, k \le N$, $0 \le i-k \le N-1$, $0 \le j-k \le N-1$	$V = C$, $\Phi_C = \{1 \le i, j \le N\}$, $\vartheta_C = (0, 0, 1)$	$-1 \le (N-1)(i+j+k) \le 1$, $-1 \le (N-1)(i+j) \le 1$, $-1 \le (N-1)(i-j) \le 1$, $-1 \le (N-1)(j+k) \le 1$, $-1 \le (N-1)(j+k) \le 1$, $-1 \le (N-1)k \le 1$
band matrix – matrix multiplication $C_{N1\times N3} = A_{N1\times N2} \cdot B_{N2\times N3}$	$N1\ \ N2\ \ N3$ $p1\ \ p2\ \ q1$ $q2\ \ r1\ \ r2$		$1 \le i \le N1$, $1 \le j \le N3$, $1 \le k \le N2$, $1-q2 \le j-k \le q1-1$, $1-p1 \le i-k \le p2-1$	$V = A$, $\Phi_A = \begin{cases} 1 \le i \le N1, \\ 1 \le k \le N2, \\ -(p1-1) \le i-k \le p2-1 \end{cases}$, $\vartheta_A = (0, 1, 0)$ $V = B$, $\Phi_B = \begin{cases} 1 \le j \le N3, \\ 1 \le k \le N2, \\ -(q2-1) \le j-k \le q1-1 \end{cases}$, $\vartheta_B = (1, 0, 0)$ $V = C$, $\Phi_C = \begin{cases} 1 \le i \le N1, \\ 1 \le j \le N3, \\ -(r1-1) \le i-j \le r2-1 \end{cases}$, $\vartheta_C = (0, 0, 1)$	the coefficient vectors of the dual body R^* are the vertices of the difference body $(R - R) =$ $\begin{cases} 1-N1 \le i \le N1-1, \\ 1-N3 \le j \le N3-1, \\ 1-N2 \le k \le N2-1, \\ -(p1+p2-2) \le i-k \le p1+p2-2, \\ -(q1+q1-2) \le j-k \le q1+q2-2 \end{cases}$

Table 2. Results of SODTLA for the problems listed in Table 1.

i	problem size parameters	linear schedule A	optimal allocation vector S	#*PE*
1	$N = 3$	(1, 1, 4)	(–1, 0, 0)	2+1
2	$N = 4$	(1, 1, 5)	(–1, 0, 0)	3+1
3	$N = 8$	(1, 1, 7)	(–1, 0, 2)	21+1
4	$N = 16$	(2, 1, 8)	(–2, 0, 1)	45+1
5	$N = 32$	(3, 1, 10)	(–3, 0, 2)	155+1
6	$N = 64$	(5, 1, 13)	(–5, 0, 1)	378+1
7	$N = 100$	(5, 1, 17)	(–5, 0, 4)	891+1
8	$N = 200$	(8, 1, 22)	(–8, 1, 5)	2786+1
9	$N = 300$	(9, 1, 28)	(–9, 0, 8)	5083+1
10	$N = 4$	(1, 2, 1)	(0, 2, –1)	6+1
11	$N = 8$	(6, 5, 1)	(2, 0, –1)	14+1
12	$N = 100$	(5, 1, 27)	(4, 0, –1)	396+1
13	$N = 200$	(8, 1, 23)	(7, 0, –6)	1393+1
14	$N = 300$	(9, 1, 25)	(–9, 0, 11)	3289+1
15	$N1 = 5$ $N2 = 4$ $N3 = 3$ $p1 = 1$ $p2 = 5$ $q1 = 3$ $q2 = 1$ $r1 = 3$ $r2 = 5$	(1, 1, 4)	(–1, 1, –1)	6+1
16	$N1 = 4$ $N2 = 4$ $N3 = 4$ $p1 = 2$ $p2 = 2$ $q1 = 3$ $q2 = 2$ $r1 = 4$ $r2 = 3$	(1, 1, 4)	(1, –1, –1)	5+1
17	$N1 = 100$ $N2 = 100$ $N3 = 100$ $p1 = 2$ $p2 = 2$ $q1 = 3$ $q2 = 2$ $r1 = 4$ $r2 = 3$	(1, 2, 50)	(–1, –1, 2)	5+1
18	$N1 = 6$ $N2 = 4$ $N3 = 6$ $p1 = 2$ $p2 = 3$ $q1 = 3$ $q2 = 2$ $r1 = 4$ $r2 = 4$	(1, 2, 4)	(–1, –1, 2)	6+1
19	$N1 = 100$ $N2 = 100$ $N3 = 100$ $p1 = 25$ $p2 = 25$ $q1 = 10$ $q2 = 10$ $r1 = 34$ $r2 = 34$	(1, 3, 20)	(–1, –2, 7)	480+1
20	$N1 = 50$ $N2 = 60$ $N3 = 80$ $p1 = 5$ $p2 = 10$ $q1 = 15$ $q2 = 15$ $r1 = 19$ $r2 = 24$	(5, 1, 15)	(3, –1, –2)	67+1

REFERENCES

1. W. Shang and J. A. B. Fortes, "Time optimal linear schedule for algorithms with uniform dependencies," *IEEE Transactions on Computer*, Vol. 40, No. 6, 1991, pp. 723-742.
2. J. A. B. Fortes and F. Parisi-Presicce, "Optimal linear schedules for the parallel execution of algoeithms" in *Proceedings of International Conference on Parallel Processing*, 1984, pp. 322-329.
3. A. Benaini and Y. Robert, "Spacetime-minimal systolic architectures for gaussian elimination and the algebraic path problem," in *Proceedings of International Conference on Application Specific Array Processors*, 1990, pp. 747-757.
4. P. R. Cappello, "A processor-time-minimal systolic array for cubical mesh algorithms," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 3, No. 1, 1992, pp. 4-13.
5. P. R. Cappello, "A spacetime-minimal systolic array for matrix product," in *Proceedings of International Conference on Systolic Arrays*, 1989, pp.347-356.
6. P. Clauss, C. Mongenet and G. Perrin, "Calculus of space-optimal mapping of systolic algorithms on processor arrays" *Journal of VLSI Signal Processing*, Vol. 4, No. 1, 1992, pp. 27-36.
7. C. J. Scheiman and P. R. Cappello, "A processor-time-minimal systolic array for transitive closure," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 3, No. 5, 1992, pp. 257-269.
8. J. C. Tsay and P. Y. Chang, "Design of space-optimal regular arrays for algorithms with linear schedule," *IEEE Transactions on Computer*, Vol. 44, No. 5, 1995, pp. 683-694.
9. Y. Wong and J. M. Delosme, "Space-optimal linear processor allocation for systolic arrays synthesis," in *Proceedings of the Sixth International Parallel Processing Symposium*, 1992, pp. 275-282
10. X. Zhong and S. Pajopadhye, "Systematic generation of linear allocation functions in systolic arrays design," *Journal of VLSI Signal Processing*, Vol. 4, No. 4, 1992, pp. 279-293.
11. P. Z. Lee and Z. M. Kedem, "Synthesizing linear array algorithms from nested for loop algorithms," *IEEE Transactions on Computer*, Vol. 37, No. 12, 1988, pp. 1578-1598.
12. J. C. Tsay and P. Y. Chang, "Designing lower dimensional regular arrays for algorithms with uniform dependencies," *Journal of Parallel and Distributed Computing*, Vol. 33, No. 1, 1996, pp. 24-32.
13. K. N. Ganapathy and B. W. Wah, "Optimal synthesis of algorithm-specific lower-dimentional processor arrays," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 7, No. 3, 1996, pp. 274-287.
14. K. N. Ganapathy and B. W. Wah, "Synthesizing optimal lower dimensional processor arrays," in *Proceedings of International Conference on Parallel Processing*, 1992, pp. III. 96-III. 103.
15. J. Xue, "A unified approach to checking data link and computational conflicts in the design of algorithm-specific processor arrays," Technical Report. No. 94-100, Department of Mathematics, Statistics and Computing Science, the University of New England, 1994.

16. W. Shang and J. A. B. Fortes, "On time mapping of uniform dependence algorithms into lower dimensional processor arrays," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 3, No. 5, 1992, pp. 350-363.
17. P. Z. Lee and Z. M. Kedem, "Mapping nested loop algorithms into multidimensional systolic arrays," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 1, No. 1, 1990, pp. 64-76.
18. J. Xue, "Closed-form mapping conditions for the synthesis of linear processor arrays," *Journal of VLSI Signal Processing*, Vol. 10, No. 2, 1995, pp. 181-199.
19. Z. Yang, W. Shang and J. A. B. Fortes, "Conflict-free scheduling in nested loop algorithms on nested loop algorithms on lower dimensional processor arrays," in *Proceedings of the Sixth International Parallel Processing Symposium*, 1992, pp. 156-164.
20. J. W. S. Cassels, *An Introduction to the Geometry of Numbers*, Springer-Verlag, Berlin, 1959.
21. U. Dieter, "How to calculate shortest vectors in a lattice," *Mathematics of Computation*, Vol. 29, No. 131, 1975, pp. 827-833.
22. J. C. Tsay and J. Y. Ke, "A new approach to finding optimal linear scheduling for uniform dependence algorithms," *Parallel Algorithms and Applications*, Vol. 7, No. 1-2, 1995, pp. 73-86.
23. S. Y. Kung, *VLSI Array Processor*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.
24. G. Nemhauser and L. Wolsey, *Integer and Combinatorial Optimization*, Wiley, New York, 1988.

Jenn-Yang Ke（柯振揚）received the B. S. degree in computer science from Tamkang University in 1981 and the M.S. degree in information engineering from the Tatung Institute of Technology in 1987. In 1987, he was a lecturer in the Department of Information Engineering, Tatung Institute of Technology. He is now a Ph. D. candidate in the Department of Computer Science and Information Engineering at National Chiao Tung University. His current research interests include systolic arrays and parallel compiler.

Jong-Chuang Tsay（蔡中川）received the M. S. and Ph. D. degrees in computer science from National Chiao Tung University in 1968 and 1975, respectively. Since 1968, he has been with the faculty of the Department of Computer Engineering, National Chiao Tung University. Currently he is a professor in the Department of Computer Science and Information Engineering, National Chiao Tung University. His current research interests include systolic arrays and parallel computations.