Communication Optimization for Affine Recurrence Equations using Broadcast and Locality

Vincent Loechner ‡† and Catherine Mongenet ‡

‡ Université Louis Pasteur de Strasbourg, Laboratoire ICPS,
Bd. S. Brant, F-67400 Illkirch, France
`http://icps.u-strasbg.fr`

† Université de Metz, Laboratoire LRIM,
Ile du Saulcy, BP 80794, F-57012 Metz Cedex 1, France

October 15, 1999

Abstract

This paper deals with communication optimization which is a crucial issue in automatic parallelization. From a system of parameterized affine recurrence equations we propose a heuristic which determines a set of efficient space-time transformations. It focuses on distant communications removal using broadcast -including anticipated broadcast- and locality enforcement.

Keywords. automatic parallelization techniques, communication optimization, broadcast, system of recurrence equations.

1 Introduction

Among the many works dealing with automatic parallelization techniques, one of the important issues is concerned with communication optimization. Our paper focuses on this question. It discusses strategies of communication optimization for systems of parameterized affine recurrence equations [24] (PARE's) which formalize single assignment affine loop nests. Any Fortran-like loop nest composed of assignement statements in which the reference arrays are affine functions of the loop bounds, can be translated into an equivalent PARE where each equation corresponds to a statement. Several such loop nests can be expressed as a set of PARE's.

The parallelization of a PARE is classically based on an affine space-time transformation (see Lengauer's survey [13] for references). It assigns to each computation a virtual processor defined by an allocation function and an execution time defined by a schedule function. This framework of affine space-time transformation encompasses the traditional loop transformation techniques such as loop permutation, reversal, skewing, fusion, reindexing and scaling. In our execution model, we assume that each virtual processor has its own memory, and the communications between these processors result from the mapping on the processor space of the dependences of the PARE.

The objective of this paper is to propose a heuristic to automatically parallelize a PARE. Such a heuristic is guided by a cost function, which can optimize various criteria, such as latency or number of virtual processors. The cost function we take into account in this paper is the number of communications. We distinguish between three types of communications: local communications, broadcast communications, and distant communications *i.e.* communications between pairs of processors whose distance is not a given constant. Local and broadcast communications can be efficiently implemented on most target architectures. This is not always the case for distant communications, especially on massively parallel architectures. On this type of machine, the execution of a set of distant communications at a given time step may provoke bottlenecks at some nodes and increase the charge of the network. Therefore our strategy of communication optimization mainly focuses on dependences which generate distant communications in order to localize or broadcast them as much as possible. Localization and broadcast both require conditions to be satisfied by the allocation function and by the schedule function. Compared with other approaches which focus on allocation and/or alignment without paying attention to the schedule determination [5, 7, 26, 6] we deal with both questions simultaneously. Hence the allocation and the schedule functions are strongly related for the global space-time transformation to be valid as stated by Feautrier [11].

The communication optimization technique developed in this paper relies on the dependence information given by parameterized utilization sets and utilization vectors [22, 20]. This dependence modeling permits to classify the dependences according to the potential number of distant communications they may induce.

For a given dependence, the utilization points associated with a given data are all the computation points using this data. In the worst case, this dependence will result in as many communications (among which many distant ones) as there are utilization points. These distant communications can be removed using either a strong condition which maps all the utilization points onto the same virtual processor, or a weaker condition which distributes the computations on a set of neighboring processors. With such a distribution one may schedule the computations using a given set of data:

(1) either successively, by propagating the data among the neighboring processors using local communications,

(2) or simultaneously, by broadcasting the data among the neighboring processors.

Moreover one may consider an alternative solution, anticipated broadcast, which combines a scheduling of type (1) together with broadcast communications. Notice that such a solution requires memory to store the data that have been fetched by anticipation.

This paper is organized as follows. Section 2 recalls the basis notions of PARE parallelization. Communication optimization related to one dependence is discussed in section 3, where conditions to eliminate distant communications using either localization or broadcast are presented. In a program characterized by a set of dependences, the optimal solution for a given dependence may not be compatible with the optimal solution for another one. This is why we propose a heuristic to globally optimize communications using these techniques as described in section 4 and illustrated with an example in section 5. In section 6 we compare our approach with related works and propose different ways to improve the heuristic.

2 Systems of parameterized affine recurrence equations parallelization

A *parameterized affine recurrence equation* (PARE) is defined by:

$$X[z] = f(\ldots, Y[g(z,p)], \ldots) \qquad z \in \mathcal{D}(p) \qquad (Eq)$$

where X and Y are variable names. p is a vector of $\mathbb{Z}^q$ defining the problem parameters. $\mathcal{D}(p) \subset \mathbb{Z}^n$ is a parameterized convex bounded polyhedron called the *iteration domain*. It is defined by a set of k linear constraints of the form $\mathcal{D}(p) = \{z \in \mathbb{Z}^n \mid Cz + C'p + c \geq 0\}$, where C is a $k \times n$ integer matrix, C' a $k \times q$ integer matrix, and c an integer k-vector. The affine *index function* $g(z,p)$ is defined by $g(z,p) = Rz + Qp + r$, where R and Q are integer matrices, and r is an integer vector.

Example 1. The gaussian elimination problem consists in solving a linear system of equations $Ax = b$ where $A = (a_{ij})$ is an $N \times N$ matrix and b an N vector. It triangulates the extended matrix $\mathtt{A} = [A \mid b]$. It is defined by the following loop nest:

```
DO k = 1 TO N-1
   DO j = k+1 TO N+1
      DO i = k+1 TO N
         A[i,j] = A[i,j] - A[i,k] * A[k,j] / A[k,k]
      DONE
   DONE
DONE
```

Using classical techniques, this loop nest is transformed into the following equivalent single-assignment loop:

```
DO k = 1 TO N-1
   DO j = k+1 TO N+1
      DO i = k+1 TO N
         A[i,j,k] = A[i,j,k-1] - A[i,k,k-1] * A[k,j,k-1] / A[k,k,k-1]
      DONE
   DONE
DONE
```

The corresponding PARE is:

$$A[i,j,k] = A[i,j,k-1] - A[i,k,k-1] * A[k,j,k-1] / A[k,k,k-1] \qquad \begin{pmatrix} i \\ j \\ k \end{pmatrix} \in \mathcal{D}(N)$$

with

$$\begin{aligned} \mathcal{D}(N) &= \left\{ \begin{pmatrix} i \\ j \\ k \end{pmatrix} \middle| \begin{array}{l} k+1 \leq i \leq N \\ k+1 \leq j \leq N+1 \\ 1 \leq k \leq N-1 \end{array} \right\} \\ &= \left\{ \begin{pmatrix} i \\ j \\ k \end{pmatrix} \middle| \begin{pmatrix} 1 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} i \\ j \\ k \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} (N) + \begin{pmatrix} -1 \\ 0 \\ -1 \\ 1 \\ -1 \\ -1 \end{pmatrix} \geq 0 \right\} \end{aligned}$$

2.1 Dependence modeling

The dependence information extracted from such a PARE is characterized by its utilization and emission sets [22]. The *utilization set* corresponding to occurrence $Y[z_0]$ in equation (Eq) is:

$$\mathrm{Util}_{Eq,Y}(z_0, p) = \{ z \in \mathcal{D}(p) \mid g(z,p) = z_0 \}.$$

It is a convex polyhedron parameterized by z_0 and p. Its vertices are denoted by v_i. They can be determined as functions of z_0 and p using the algorithm developed by Loechner and Wilde [20]: $v_i = A_i z_0 + B_i p + c_i$, where A_i and B_i are constant matrices, and c_i is a constant vector. This algorithm deals with rational polyhedra, and therefore determines the rational vertices of such polyhedra. Notice that the number of vertices may depend on symbolic parameters p and z_0. This is taken into account by splitting the problem in subdomains relative to the parameters [20]. Used in the context of utilization sets determination, this algorithm manipulates the rational convex hull of $\mathrm{Util}_{Eq,Y}$ and computes its rational vertices. Notice however, that in most practical problems, utilization sets are characterized by integer vertices. In the following, we develop methods to optimize communications relatively to the vertices of the utilization sets. These methods only apply on integer vertices.

Any utilization set $\mathrm{Util}_{Eq,Y}(z_0, p)$ is supported by a linear space of dimension $dim(\mathrm{Util}_{Eq,Y})$ whose basis vectors are called the *utilization vectors*. In the following we denote by U the column matrix of these vectors and call it the *utilization matrix*. The linear space supporting the utilization sets is contained in the null space of R, denoted by $\mathrm{Ker}(R)$. Therefore, the utilization matrix satisfies $R \cdot U = 0$. In the particular case $U = 0$ the utilization set is reduced to one single point. We call this a *degenerate* utilization set.

The dependence from Y to X in equation (Eq) is expressed by a set of *dependence vectors* of the form $d = z - z_0$, where $z \in \mathrm{Util}_{Eq,Y}(z_0, p)$. Since the utilization set is a convex polyhedron, this dependence is characterized by a set of *extremal dependence vectors*. For any integer vertex v_i of $\mathrm{Util}_{Eq,Y}(z_0, p)$ the corresponding extremal vector is denoted by $d_i = v_i - z_0$.

For a given dependence, the set of all points z_0 from which the dependence vectors originate is called the *emission set*:

$$\begin{aligned}
\mathrm{Emit}_{Eq,Y}(p) &= \{z_0 \in \mathbb{Z}^n \mid \exists z \in \mathcal{D}(p) \text{ such that } z_0 = g(z,p)\} \\
&= \{Rz + Qp + r \mid z \in \mathcal{D}(p)\} \\
&= g(\mathcal{D}(p), p)
\end{aligned}$$

It is contained in a linear space of dimension $dim(\mathrm{Emit}_{Eq,Y})$ whose basis vectors form a column matrix denoted by E and called the *emission matrix*.

The utilization and emission sets satisfy:

$$dim(\mathrm{Util}_{Eq,Y}) + dim(\mathrm{Emit}_{Eq,Y}) = dim(\mathcal{D}(p))$$

The utilization matrix U and the emission matrix E are computed from the Hermite normal form decomposition of index matrix R. Notice that this decomposition is not unique when index matrix R is not of full rank. The computation of matrices U and E depends on the geometrical dimension of $\mathcal{D}(p)$:

case 1. if $dim(\mathcal{D}(p)) = n$. The Hermite normal form decomposition of index matrix R is:

$$R \cdot [V|U] = [E|0]$$

where matrix $[E|0]$ is the Hermite normal form of R, and $[V|U]$ is an $n \times n$ unimodular matrix, *i.e.* a matrix whose determinant is equal to ± 1. Since $[V|U]$ is unimodular, its column vectors generate the whole space $\mathbb{Z}^n$. In such a case U defines a basis of $\mathrm{Ker}(R)$ and V a basis of the *range* of R, and $E = R \cdot V$. Therefore we have $dim(\mathrm{Emit}_{Eq,Y}) = \mathrm{Rank}(R)$, and $dim(\mathrm{Util}_{Eq,Y}) = n - \mathrm{Rank}(R)$.

case 2. if $dim(\mathcal{D}(p)) < n$. This situation is illustrated in example 2 hereunder. In this case a Hermite normal form decomposition of index matrix R is $R \cdot M = H$ where unimodular matrix M defines a set of n basis vectors generating the whole n-dimensional space. Since the iteration domain has a smaller dimension, some of the vectors of M are irrelevant to describe matrices U and V. These matrices are obtained by an intersection with the linear space supporting $\mathcal{D}(p)$ as follows.

Let a Hermite normal form decomposition of index matrix R be:

$$R \cdot [V'|U'] = [E'|0]$$

where $[V'|U']$ is an $n \times n$ unimodular matrix. Utilization matrix U is obtained from U' by intersecting the subspace generated by the column vectors of U' with the subspace supporting $\mathcal{D}(p)$. Matrix V is obtained from V' in the same way. The emission matrix is then $E = R \cdot V$.

In the following we call V the *complementary matrix* of U. A linear combination of the column vectors of V defines a translation vector from an affine subspace supporting a utilization set to another one.

Example 2: Study of a problem where $dim(\mathcal{D}(p)) < n$ and computation of the utilization and emission matrices related to a given dependence.

Let us consider the following dependence:

$$X\begin{bmatrix} i \\ j \\ k \end{bmatrix} = X\begin{bmatrix} i-1 \\ i-1 \\ i-1 \end{bmatrix}, \ \mathcal{D}(N) = \left\{ \begin{pmatrix} i \\ j \\ k \end{pmatrix} \in \mathbb{Z}^3 \middle| \, 0 \leq i \leq N, 0 \leq j \leq N, j = k \right\}$$

The iteration domain is contained in the plane of equation $j = k$ in $\mathbb{Z}^3$, characterized by two basis vectors $\mathcal{B} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$. Both the emission and the utilization sets have dimension 1 (cf. figure 1).

This dependence may be expressed by different index matrices, for instance $R_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$ or $R_2 = \begin{pmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$.

The utilization and the emission matrices are computed from the Hermite normal form of the index matrix. Let us successively consider both index matrices:

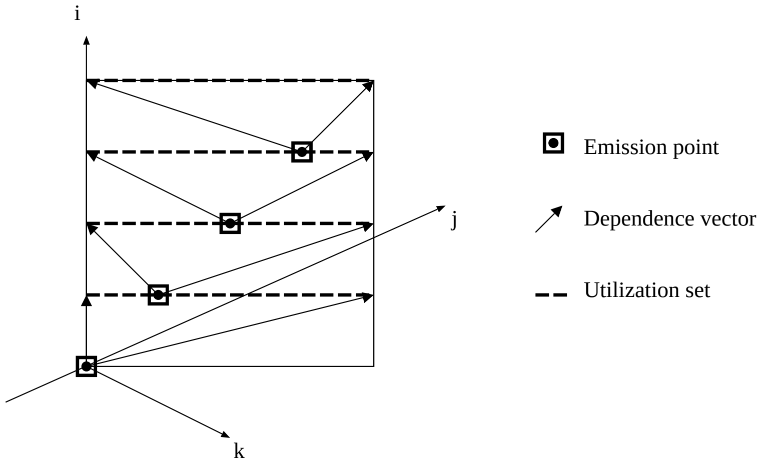


Figure 1: Dependence modeling of the PARE defined in example 2.

- $R_1 \cdot \begin{array}{c} \begin{array}{ccc} V' & U' & \end{array} \\ \left[\begin{array}{c|cc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \end{array} = \begin{array}{c} \begin{array}{ccc} E' & & \end{array} \\ \left[\begin{array}{c|cc} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{array}\right] \end{array}$. By intersecting the spaces generated by V' and by U' with the space generated by $\mathcal{B}$, we get: $V = V' = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$, $E = R \cdot V = E' = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$, and $U = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$.

- $R_2 \cdot \begin{array}{c} \begin{array}{ccc} V' & & U' \end{array} \\ \left[\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 1 \end{array}\right] \end{array} = \begin{array}{c} \begin{array}{ccc} E' & & \end{array} \\ \left[\begin{array}{cc|c} 1 & -1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{array}\right] \end{array}$. By intersecting the spaces generated by V' and by U' with the space generated by $\mathcal{B}$, we get: $V = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$, $E = R \cdot V = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$, and $U = U' = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$.

For all possible values of index matrix, we get the same matrices U, V and E.

2.2 Affine space-time transformation

In order to find a parallel solution to a set of PARE an *affine space-time transformation* has to be determined. Such a transformation is defined by two functions: an affine schedule function and an affine allocation function. In order to efficiently optimize the communications, we deal with *per variable* transformations. Such a transformation is expressed, for each variable X, by a full rank $n \times n$ transformation matrix $T_X = \begin{pmatrix} \lambda_X \\ \sigma_X \end{pmatrix}$ and two constant vectors α_X and γ_X where:

- λ_X is a $d_\lambda \times n$ integer matrix defining the *multi-dimensional schedule*: $t_X(z) = \lambda_X z + \alpha_X$. The row vectors of λ_X are called the *schedule vectors*. They are orthogonal to a set of affine subspaces called *isotemporal spaces*. Each isotemporal space contains all the computation points executed at a given time step.

- σ_X is a $d_\sigma \times n$ integer matrix representing the *allocation* on a virtual processor space of dimension d_σ. The row vectors of σ_X define the basis vectors of the virtual processor space. The allocation function is defined by $\mathrm{alloc}_X(z) = \sigma_X z + \gamma_X$.

As mentioned above, space-time transformation matrices T_X must be of full rank. This condition is necessary for the transformation to be valid. Hence, it guarantees that no two computations are executed at the same instant by the same processor. Of course, not every full rank matrix is valid. Other conditions must or may be enforced. The first conditions that *must always* be satisfied are the *causal constraints*. These constraints state that a data has to be computed before its use. They are classically expressed for any dependence from Y to X by

$$t_X(z) \succ t_Y(z_0), \quad \forall z \in \mathrm{Util}_{Eq,Y}(z_0, p)$$

where $\succ$ is the lexicographical *greater than* operator. A set of affine constraints on schedule matrices λ_X and λ_Y, and on vectors α_X and α_Y is found by applying the Farkas Lemma [8, 9] or the vertex method [24] to this system.

Other conditions *may be* considered depending on the characteristics we want the final solution to have. In this paper, we focus on conditions related to communication optimization. We use the term *communication vectors* to refer to the image of the dependences in the processor space by the transformation matrices. Communication optimization consists in reducing the number of non-null communication vectors as much as possible, especially the distant ones.

3 Communication optimization for a given dependence

In order to develop our communication optimization techniques, we focus in this section on a given dependence from Y to X of the form $X[z] = f(..., Y[Rz +$

$Qp + r], \ldots)$ with $z \in \mathcal{D}(p)$. We simplify the notation by calling $\mathrm{Util}(z_0, p)$ the associated utilization set $\mathrm{Util}_{Eq,Y}(z_0, p)$, and by calling $\mathrm{Emit}(p)$ the associated emission set $\mathrm{Emit}_{Eq,Y}(p)$. We denote $dim(\mathrm{Util}_{Eq,Y})$ by dim_{Util} and $dim(\mathrm{Emit}_{Eq,Y})$ by dim_{Emit}. Focusing on such a dependence, we determine the constraints to be satisfied in order to optimize the communications. These constraints may be related to both the schedule and the allocation functions. This is why, compared to other approaches where conditions on the schedule and conditions on the allocation are examined independently, we consider the whole problem globally.

At a given time step for a given dependence, a subset of computations is executed, and we are concerned with the non-null communication vectors associated with these computations. These vectors may correspond to three types of communications:

- local communications, between pairs of processors at a constant distance,
- distant communications, between pairs of processors whose distance is not constant,
- broadcast communications, from a set of emission processors to a set of utilization processors. Depending on the constraints on the schedule, this may correspond to a "true" broadcast or to an anticipated broadcast.

Local and broadcast communications are efficiently implemented on most target architectures. This is not the case for distant communications, especially on massively parallel architectures, where bottlenecks may appear at some nodes and the charge of the network globally increases.

3.1 How to eliminate distant communications

Since distant communications are costlier than the other types of communications, we try first to eliminate them. A communication results from the image by the allocation functions of a dependence vector on two different virtual processors. For a dependence from Y to X, the resulting communication vector is: $\mathrm{alloc}_X(z) - \mathrm{alloc}_Y(z_0)$. Using equality $z_0 = Rz + Qp + r$ with $z \in \mathrm{Util}(z_0, p)$, we have:

$$\begin{aligned}\mathrm{alloc}_X(z) - \mathrm{alloc}_Y(z_0) &= \mathrm{alloc}_X(z) - \mathrm{alloc}_Y(Rz + Qp + r)\\ &= \sigma_X z + \gamma_X - \sigma_Y \cdot Rz - \sigma_Y(Qp + r) - \gamma_Y\\ &= (\sigma_X - \sigma_Y \cdot R)z - \sigma_Y(Qp + r) - \gamma_Y + \gamma_X\end{aligned}$$

Let us rewrite this equation on the simplified form

$$\mathrm{alloc}_X(z) - \mathrm{alloc}_Y(z_0) = (\sigma_X - \sigma_Y \cdot R)z + \rho(p) \qquad (1)$$

where $\rho(p) = -\sigma_Y(Qp+r) - \gamma_Y + \gamma_X$ does not depend on z and can therefore be seen as a constant term. From this equation we can deduce a sufficient condition to eliminate distant communications [1, 10, 7, 21].

Theorem 1 [Strong condition] *The dependence from Y to X does not induce any distant communication if there exists a constant c such that*

$$\begin{array}{ll} & (\sigma_X - \sigma_Y \cdot R)z = c \qquad \forall z \in \mathcal{D}(p) \\ \Leftrightarrow & (\sigma_X - \sigma_Y \cdot R) \cdot \mathcal{B} = 0 \qquad \textit{where } \mathcal{B} \textit{ is a basis of } \mathcal{D}(p) \end{array}$$

When this theorem is satisfied all the utilization points are mapped onto the same virtual processor. A consequence of such an allocation, is that the utilization points have to be scheduled at successive time steps. However this condition is already expressed by the fact that the transformation matrices are of full rank and there are no new constraints on the schedule in this case.

The distance between the emission processor and the corresponding utilization processor is $c + \rho(p)$. The value of c can be computed using one of the vertices of $\mathcal{D}(p)$. This distance can be zero, and in this case there is no communication at all for this dependence, as stated by the following corollary.

Corollary 1 *The dependence from Y to X does not induce any communication if theorem 1 is satisfied and $c + \rho(p) = 0$.*

One of the results of this paper is to show that the condition given in theorem 1 is often too strong. In the case of 0-dimensional utilization sets related to n-dimensional emission sets, R is of full rank and it has been proved that this condition *is necessary* [21]. However it is sufficient but not necessary in the case of non-degenerate utilization sets, *i.e.* one or multi-dimensional sets. One of our objectives is to propose for such utilization sets a weaker condition to eliminate distant communications.

In the case of a non-degenerate utilization set, the data computed at a given emission point is used by a convex polyhedron: the utilization set. There are two efficient ways to transmit this data to the whole utilization set.

- The first one, called *propagation*, sends the data along one of the extremal dependence vectors, and then sends it from neighbor to neighbor using the utilization vectors. Since the utilization vectors are constant, these latter communications are local. Therefore the only distant communications that may occur are the ones induced by the chosen extremal dependence vector d_i (cf. figure 2). If these communications are local (resp. distant), we call this a *local propagation* (resp. *distant propagation*). Notice that a propagation imposes new constraints on the schedule as discussed in section 3.2.

- The second one broadcasts the data from the emission processors simultaneously to all the utilization processors (cf. figure 3). This solution imposes new constraints on the schedule, as discussed in section 3.3.1. Moreover, we may constrain the allocation to guarantee that there is a local communication between the emission processor and one of the utilization processors. Such a situation may be particularly interesting in the case of hardware design. We call it a *local broadcast*.

Let us now study successively these two communication schemes.

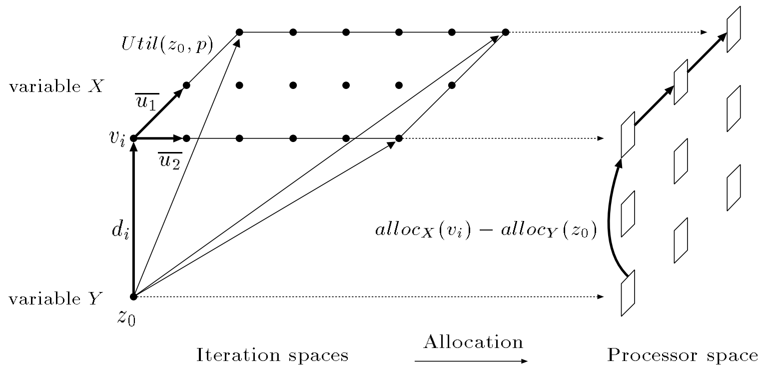


Figure 2: Example of the communications induced by a non-degenerate utilization set in case of propagation.

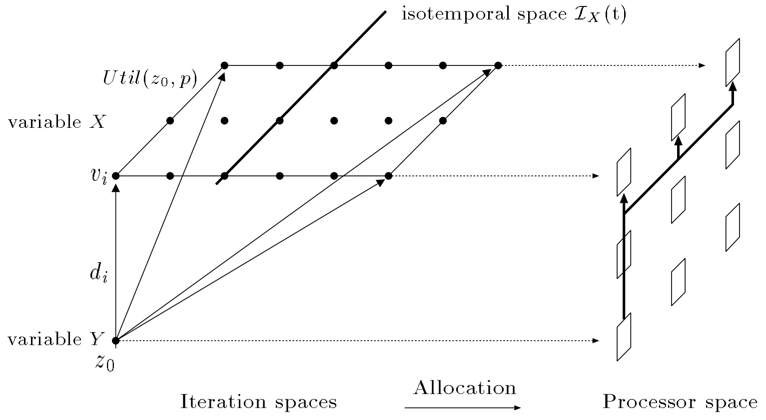


Figure 3: Example of the communications induced by a non-degenerate utilization set in case of broadcast.

3.2 Local propagation

When we want to apply propagation, to a non-degenerate utilization set, an integer vertex v_i of the utilization set is chosen to initiate the propagation, as described in figure 2. Notice that not all the vertices are good candidates for propagation due to incompatibilities with the schedule constraints. This point will be discussed in section 4. As mentioned above, the only distant communication that may occur is the one related to the chosen integer vertex v_i. It is characterized by vector: $\mathrm{alloc}_X(v_i) - \mathrm{alloc}_Y(z_0)$. The local propagation condition must guarantee that the mapping of this vector on the processor space does not depend on z_0. This yields the following theorem [18].

Theorem 2 [Local propagation, condition on the allocation] *Let $v_i = A_i z_0 + B_i p + c_i$ be the vertex chosen for propagation. The dependence from Y to X does not induce any distant communication if and only if:*

$$(\sigma_X \cdot A_i - \sigma_Y) \cdot E = 0$$

where E is the emission matrix.

Proof. Let $v_i = A_i z_0 + B_i p + c_i$ be the chosen vertex. The distant communication that may occur is characterized by vector:

$$\begin{aligned}
\mathrm{alloc}_X(v_i) - \mathrm{alloc}_Y(z_0) &= \mathrm{alloc}_X(A_i z_0 + B_i p + c_i) - \mathrm{alloc}_Y(z_0) \\
&= \sigma_X \cdot A_i z_0 + \sigma_X \cdot B_i p + \sigma_X c_i + \gamma_X - \sigma_Y z_0 - \gamma_Y \\
&= (\sigma_X \cdot A_i - \sigma_Y) z_0 + \sigma_X \cdot B_i p + \sigma_X c_i + \gamma_X - \gamma_Y \\
&= (\sigma_X \cdot A_i - \sigma_Y) z_0 + \rho'(p) \qquad (2)
\end{aligned}$$

where $\rho'(p) = \sigma_X \cdot B_i p + \sigma_X c_i + \gamma_X - \gamma_Y$ is a constant function according to z_0.

Let $z_0, z_1 \in \mathrm{Emit}(p)$. We have $z_0 = z_1 + \sum_{j=1}^{d} (\alpha_j \, \varepsilon_j)$ with $\alpha_j \in \mathbb{Z}$ and ε_j the column vectors of E, *i.e.* the basis vectors of the affine space supporting $\mathrm{Emit}(p)$. Constant d denotes dim_{Emit}. From equation (2) we deduce that there is no distant communication if and only if $(\sigma_X \cdot A_i - \sigma_Y) z_0$ is constant $\forall z_0 \in \mathrm{Emit}(p)$. Therefore we have:

$$\begin{aligned}
& (\sigma_X \cdot A_i - \sigma_Y)(z_0 - z_1) = 0 \\
\Leftrightarrow \quad & (\sigma_X \cdot A_i - \sigma_Y) \sum_{j=1}^{d} (\alpha_j \, \varepsilon_j) = 0 \quad \forall \alpha_j \in \mathbb{Z} \\
\Leftrightarrow \quad & (\sigma_X \cdot A_i - \sigma_Y) \, \varepsilon_j = 0 \qquad \forall j \in [1, d]
\end{aligned}$$

□

This theorem states that for any emission point, the distance between the emission processor and the utilization processor related to the chosen vertex is constant. It does not constrain the mapping on the processor space of the utilization sets. In particular, if $\sigma_X \cdot U = 0$ holds, all the utilization points are mapped on a single processor, and in this case, it is equivalent to the strong condition. This corresponds to a *degenerate case of propagation.*

The dimension of the utilization set mapped on the processor space is: $\mathrm{Rank}(\sigma_X \cdot U)$. If $\mathrm{Rank}(\sigma_X \cdot U) = \mathrm{Rank}(U)$, any two utilization points are mapped onto different processors. Conversely, when $\mathrm{Rank}(\sigma_X \cdot U) < \mathrm{Rank}(U)$, there is a set of utilization points mapped on each utilization processor.

The propagation (when non-degenerate) requires new constraints on the schedule that have to be added to the initial causal constraints. Recall that the propagation sends the data from the virtual processor associated with z_0 to the one associated with chosen vertex v_i and then to all the processors related to the other utilization points. Therefore the execution time step related to vertex v_i must be less than or equal to the one of the other utilization points. This results in theorem 3.

To express this condition, we define a set of vectors $\overline{u_j}$ generating the smallest cone whose origin is v_i and which contains the utilization set (cf. figure 2). In the following, we denote by $\overline{U}$ the matrix whose columns are vectors $\overline{u_j}$. Notice that vectors $\overline{u_j}$ are director vectors of the edges of $\mathrm{Util}(z_0, p)$ at vertex v_i. Since the utilization sets depend *linearly* on the parameters z_0 and p, these vectors are constant (cf. [20], Theorem 3). For one-dimensional utilization sets there is only one vector $\overline{u}$ which is equal either to the utilization vector u or to $-u$. A two-dimensional utilization set is characterized by two utilization vectors and two vectors $\overline{u}$. A utilization set of dimension $dim_{\mathrm{Util}} > 2$ has dim_{Util} utilization vectors but it has any number x of vectors $\overline{u}$ ($x \geq dim_{\mathrm{Util}}$). Cones of dimension 3 for instance may have more than 3 generating vectors.

Theorem 3 [Local propagation, condition on the schedule] *The propagation related to vertex v_i of $\mathrm{Util}(z_0, p)$ can be applied if and only if schedule function λ_X satisfies:*

$$\lambda_X \quad \overline{u_j} \succ 0 \qquad \textit{for all } \overline{u_j} \textit{ column vectors of } \overline{U}.$$

For some specific dependences, the local propagation conditions (theorem 2 and 3) are equivalent to the strong condition (theorem 1), as discussed hereunder.

3.2.1 Full rank index matrix.

When index matrix R is of full rank the dependence is degenerate and there is only one utilization point z related to a given emission point $z_0 = Rz + Qp + r$. Therefore point z is vertex $v_1 = A_1 z_0 + B_1 p + c_1$. Since R is of full rank we have $z = R^{-1} z_0 - R^{-1} \cdot Qp - R^{-1} r = A_1 z_0 + B_1 p + c_1$ for any emission point z_0. This implies $A_1 = R^{-1}$. In this case theorem 2 can be rewritten as:

$$(\sigma_X \cdot R^{-1} - \sigma_Y) \cdot E = 0$$

Since R is of full rank, $\mathcal{B}$ can be chosen such that $\mathcal{B} = R^{-1} \cdot E$. Therefore the above constraint is equivalent to $(\sigma_X - \sigma_Y \cdot R) \cdot \mathcal{B} = 0$ and it proves that, in this particular case, the strong condition is *necessary.*

3.2.2 Uniform dependence.

A uniform dependence is characterized by $R = Id$. In this case, we have $A_i = R = Id$, and theorem 2 can be simplified into $(\sigma_X - \sigma_Y) \cdot E = 0$. Moreover $\mathcal{B}$ can be chosen such that $E = \mathcal{B}$, and it follows that theorem 2 is equivalent to theorem 1.

3.2.3 Self-dependence.

A self-dependence is a dependence from a given variable X to itself. In such a case, theorem 2 simplifies into: $\sigma_X \cdot (A_i - Id) \cdot E = 0$,

Notice that in the case of a uniform self-dependence, $A_i = R = Id$ and there is no distant communication, since any matrix σ_X satisfies theorem 2.

3.3 Broadcast communications

Let us now consider an alternative to propagation, using broadcast communications. We focus first on a "true" broadcast, and then on anticipated broadcast. By true broadcast, we mean that if a data is broadcast at time step t_0, then it is *first* used at time step $t_0 + \delta$ by *more than one* utilization processor, where δ is an integer constant not depending on the problem parameters. Conversely in an anticipated broadcast, the broadcast data is used at successive time steps, and must therefore be stored in memory. The broadcast data is stored on the utilization processors. This is an arbitrary choice; another solution could consist in storing it on the emission processor, and re-issue the broadcast for each isotemporal space in the utilization set. A third solution could be to choose one utilization processor per isotemporal space, and then broadcast the data from the chosen processor to all the utilization processors in the next isotemporal space.

Recall that we consider each dependence from Y to X characterized by $\mathrm{Util}(z_0, p)$ and that U is the column matrix of its utilization vectors. Notice that if the dependence is 0-dimensional, there is no utilization vector, and this section does not apply.

3.3.1 True broadcast

True broadcast imposes new constraints on the schedule, to be added to the initial causal constraints. In the simple case of one-dimensional utilization sets, the problem has been solved by Mongenet Clauss and Perrin [22]. For multi-dimensional utilization sets, it is possible to schedule several utilization points at the same time step without broadcasting the used data. Consider for instance the problem pictured in figure 4. Case (a) and (b) represent two different schedule functions expressed on the figure by their isotemporal spaces. Regardless the allocation functions, a broadcast is required in case (a), but is not necessary in case (b). In any case, the utilization points will be mapped onto neighboring utilization processors. In case (a), the emission point is executed at time step t, and there is a set of utilization processors to be activated at time step $t + 1$.

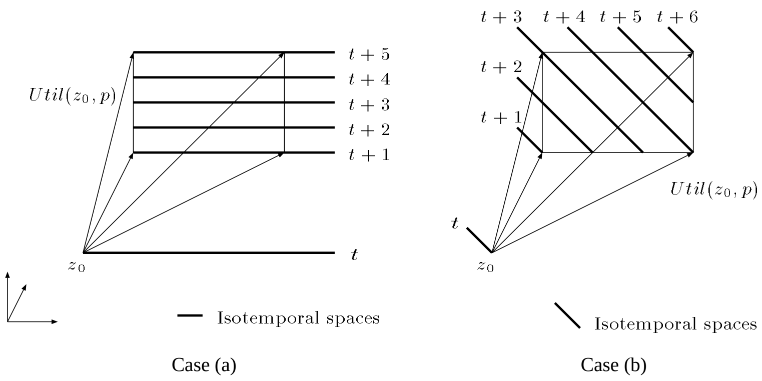


Figure 4: Two different schedule functions and their corresponding isotemporal spaces on the same multi-dimensional utilization set.

The data must therefore be broadcast. In case (b) there is only one utilization processor activated at time step $t+1$, and one more at each further time step. Therefore, broadcast is not necessary. The data can be propagated from neighbor to neighbor along the utilization processors. In such a situation anticipated broadcast is of course possible.

The following theorem states the condition for true broadcast in the general case of multi-dimensional utilization sets.

Theorem 4 [True broadcast, condition on the schedule] *The dependence from Y to X induces a true broadcast if and only if schedule matrix λ_X satisfies the following conditions. There exists a vertex v_i of* $\mathrm{Util}(z_0, p)$*, whose corresponding matrix $\overline{U}$ can be decomposed into two subcolumn matrices $\overline{U_1}$ and $\overline{U_2}$ such that:*

$$\begin{cases} \lambda_X \ \overline{u_{1_j}} = 0 & \textit{for all } \overline{u_{1_j}} \textit{ column vectors of } \overline{U_1} \\ \lambda_X \ \overline{u_{2_j}} \succ 0 & \textit{for all } \overline{u_{2_j}} \textit{ column vectors of } \overline{U_2} \end{cases}$$

The dimension of the broadcast is $\mathrm{Rank}(\overline{U_1})$*. The dimension of remaining (local) communications is* $\mathrm{Rank}(\overline{U_2})$*.*

Proof. Since $\lambda_X \overline{u_{1_j}} = 0$, there exists a vertex $v' = v_i + \tau \overline{u_{1_j}}$ of $\mathrm{Util}(z_0, p)$, such that $t_X(v_i) = t_X(v')$ where τ is a positive integer. Since the schedule function is affine, all the points on the segment between v_i and v' are also scheduled at the same time step. These points are the first active points of the utilization set, and a broadcast is required to validate these simultaneous computations. If there is only one $\overline{u_{1_j}}$ such that $\lambda_X \overline{u_{1_j}} = 0$ then the broadcast is one-dimensional. Conversely, the number of independent such vectors defines the dimension of the broadcast. This number is the rank of $\overline{U_1}$.

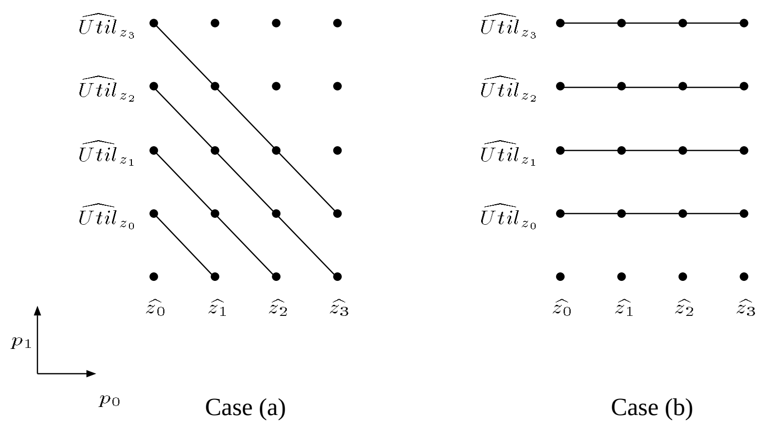


Figure 5: Two different allocations and their corresponding images of the utilization sets in the processor space.

In the same way, the number of independent vectors $\overline{u_{2_j}}$ such that $\lambda_X\ \overline{u_{2_j}} \succ 0$, defines the dimension of the remaining non-broadcast communications. This is the rank of $\overline{U_2}$.

□

The theorem easily simplifies for one-dimensional true broadcast. If the target architecture only allows one-dimensional broadcast, then the simpler version of the theorem should be used [22].

Corollary 2 *The dependence does not induce a true broadcast if schedule matrix λ_X is such that $\lambda_X \cdot U$ is of full column rank.*

Proof. The column vectors of U define a basis of the utilization sets. Since λ_X is the schedule matrix, $\lambda_X \cdot U$ represents a set of column vectors supporting the image by the schedule function of the utilization sets. If all these vectors are linearly independent, *i.e.* matrix $\lambda_X \cdot U$ is of full column rank, the dimension of the image of $\mathrm{Util}(z_0, p)$ in the schedule space is equal to dim_{Util}. In this case, the utilization points are scheduled at successive time steps, and there is no true broadcast.

□

3.3.2 Local broadcast

In specific cases such as hardware design, it may be interesting to guarantee that all the processors initiating the broadcast are at a constant distance from one of the utilization processors.

Consider the example described in figure 5. We call $\widehat{z_i}$ the image of z_i by the allocation function: $\widehat{z_i} = \mathrm{alloc}_X(z_i)$. Vector $\widehat{z_i}$ defines the coordinates of the

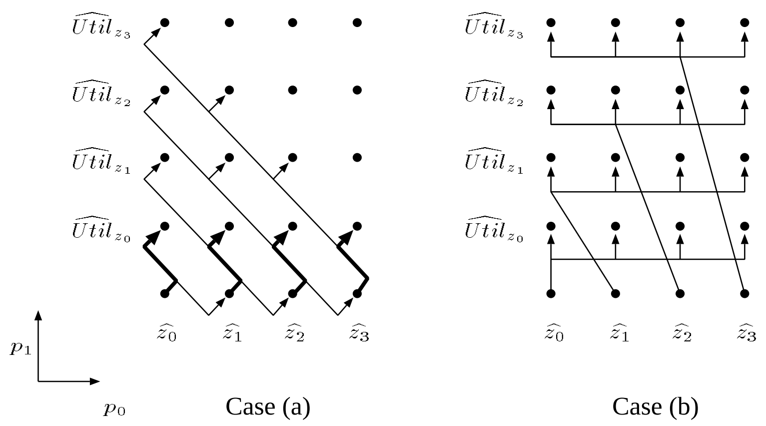


Figure 6: Communications in the processor space for both allocations.

processor executing computation z_i. In both cases, the image of the utilization sets in the allocation space are segments.

In case (a) for any processor $\widehat{z_i}$, the processor $\widehat{z_i} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ is a utilization processor. Therefore constant vector $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ characterizes the above mentioned constant distance. As a result, the broadcast comunications are regular, as shown figure 6 case (a). We call such a case *local broadcast.*

In case (b), for any processor $\widehat{z_i}$, the utilization processors are $\widehat{z_i} + \begin{pmatrix} j \\ i+1 \end{pmatrix}$ for $0 \le j \le 3$. Therefore there is no constant distance vector between $\widehat{z_i}$ and any of the corresponding utilization processors. As a result, the broadcast comunications are irregular, as shown figure 6 case (b).

In order to state the condition on the allocation for local broadcast we introduce the following notations. Let $\mathcal{S}(\mathrm{Util}(z_0, p))$ be the affine subspace supporting utilization set $\mathrm{Util}(z_0, p)$.

Let V be the complementary matrix of utilization matrix U. Recall that a linear combination of the column vectors of V defines a translation vector from an affine subspace $\mathcal{S}(\mathrm{Util}(z_0, p))$ supporting a utilization set to another one. In the case of a single vector V, it characterizes a constant distance vector between two adjacent affine subspaces supporting utilization sets.

Let us consider a linear combination Ex of the column vectors of E such that $z_1 = z_0 + Ex$ is another emission point. The following property states that when adding Vx to $\mathcal{S}(\mathrm{Util}(z_0, p))$, the subspace supporting the utilization set of z_0, we get $\mathcal{S}(\mathrm{Util}(z_1, p))$, the subspace supporting the utilization set of z_1.

Property 1 *Let z_0 be an emission point and $\mathcal{S}(\mathrm{Util}(z_0, p))$ be its corresponding subspace. Let z_1 be another emission point such as $z_1 = z_0 + Ex$, where x a constant column vector. Then we have $\mathcal{S}(\mathrm{Util}(z_1, p)) = \mathcal{S}(\mathrm{Util}(z_0, p)) + Vx$.*

Proof. Let $z \in \mathcal{S}(\mathrm{Util}(z_0, p))$, and $z' = z + Vx$. We have to show that $z' \in \mathcal{S}(\mathrm{Util}(z_1, p))$.

$$\begin{aligned} z_1 &= z_0 + Ex \\ &= Rz + Qp + r + Ex \\ &= Rz + Qp + r + R \cdot Vx \\ &= R(z + Vx) + Qp + r \\ &= g(z + Vx, p) \\ &= g(z', p) \end{aligned}$$

Therefore we have $z' \in \mathcal{S}(\mathrm{Util}(z_1, p))$.
□

Using the notions introduced hereabove, the condition on the allocation for local broadcast is stated as follows.

Theorem 5 [Local broadcast, condition on the allocation] *The dependence from Y to X induces a local broadcast if and only if there exists a constant integer matrix M of dimension $dim_{\mathrm{Util}} \times dim_{\mathrm{Emit}}$ such that:*

$$\sigma_X \cdot (V + U \cdot M) = \sigma_Y \cdot E$$

Proof. Let $\widehat{E} = \sigma_Y \cdot E$, $\widehat{U} = \sigma_X \cdot U$, and $\widehat{V} = \sigma_X \cdot V$. Let ϑ be a column vector of matrix V, and $\widehat{\vartheta} = \sigma_X \vartheta$ be the corresponding column vector in matrix $\widehat{V}$. Let $e = R\vartheta$. Then $\widehat{e} = \sigma_Y e$ is the image on the allocation space of the emission vector associated with vector $\widehat{\vartheta}$.

Let z_0 and z_0' be two emission points such that $z_0' = z_0 + e$. Let $z \in \mathrm{Util}(z_0, p)$ and $z' \in \mathrm{Util}(z_0', p)$. From property 1, we have:

$$z + \vartheta \in \mathrm{Util}(z_0', p)$$

Since $z' \in \mathrm{Util}(z_0', p)$ there exists a vector μ such that $z' = z + \vartheta + U\mu$

$$\begin{aligned} &\Rightarrow \quad z' - z = \vartheta + U\mu \\ &\Rightarrow \quad \widehat{z'} - \widehat{z} = \widehat{\vartheta} + \widehat{U}\widehat{\mu} \text{ by mapping on the processor space.} \end{aligned}$$

Let us now find a necessary and sufficient condition for both distances $\widehat{z} - \widehat{z_0}$ and $\widehat{z'} - \widehat{z_0'}$ to be equal.

$$\begin{aligned} & \widehat{z} - \widehat{z_0} = \widehat{z'} - \widehat{z_0'} \\ \Leftrightarrow \quad & \widehat{z'} - \widehat{z} = \widehat{z_0'} - \widehat{z_0} \\ \Leftrightarrow \quad & \widehat{z'} - \widehat{z} = \widehat{e} \\ \Leftrightarrow \quad & \widehat{\vartheta} + \widehat{U}\widehat{\mu} = \widehat{e} \text{ using the above equality.} \end{aligned}$$

When extending to the whole matrix V, we get the result where matrix M is composed of all the column vectors μ.
□

3.3.3 Anticipated broadcast

Anticipated broadcast corresponds to the use of broadcast communications even though the broadcast data are used at successive time steps. It is useful in the following situation. When considering the set of all dependences together, there may be incompatibilities between them in terms of communication optimization. For instance, one may get from one dependence a true broadcast, but the corresponding schedule may not be valid due to constraints related to the other dependences. In this case, if the architecture efficiently implements broadcast one may wish to realize this dependence using anticipated broadcast.

Since the broadcast data are used at successive time steps by the utilization processors, there are no new constraints on the schedule. The validity of the schedule is ensured by the fact that the space-time transformation matrices are of full rank and by the causal contraints.

Anticipated broadcast of course requires more memory space on each utilization processor. Hence, each processor must memorize a data from the time step it receives it, until the time step it uses it for the last time. Moreover, such a processor may receive several data coming from several emission processors that will have to be stored in the same way. Therefore, the number of memory locations required on one processor, relatively to one dependence, is bounded by the number of utilization sets that have at least one point mapped on the processor. This result is stated by the following theorem, where we use the notation alloc_X^{-1} to denote the preimage, *i.e.* the inverse operation of image [27]. This does not imply that alloc_X is invertible.

Theorem 6 [Memory cost for anticipated broadcast] *The dependence from Y to X may be implemented as an anticipated broadcast if $\sigma_X \cdot U \neq 0$. The number of memory locations required to store the corresponding data on a given processor P is bounded by the number of points of $g(\mathrm{alloc}_X^{-1}(P) \cap \mathcal{D}(p))$, where $\mathrm{alloc}_X^{-1}(P)$ defines the set of iteration points of X whose image by function alloc_X is P.*

Proof. For broadcast to apply, not all the utilization points must be mapped onto the same utilization processor. This is expressed by constraint $\sigma_X \cdot U \neq 0$.

The set of points in the iteration domain of X that are mapped on processor P is given by $\mathrm{alloc}_X^{-1}(P) \cap \mathcal{D}(p)$.

By applying index function g to $\mathrm{alloc}_X^{-1}(P) \cap \mathcal{D}(p)$, we get the corresponding emission points. The number of such points corresponds to the number of memory locations. Notice however that they are not necessarily active at the same time steps. This is why this value is an upper bound of the number of memory locations.

The set $g(\mathrm{alloc}_X^{-1}(P) \cap \mathcal{D}(p))$ is the image by an affine function of a parameterized lattice polytope, and its number of points is given by the Ehrhart-Clauss algorithm [2].

□

If one wants the anticipated broadcast to be localized, it is sufficient to satisfy the constraints on the allocation given by theorem 5.

This theorem gives an upper bound of the memory cost of anticipated broadcast. Evaluation of the exact memory cost is possible using a tighter modeling and a more complex algorithm. This is out of scope of this paper, and it has been done recently [19].

3.3.4 Normal broadcast

In both cases of true broadcast and anticipated broadcast, one may further wish to restrict the solutions to broadcast along the normal directions of the processor space. We call it a *normal broadcast.* It is expressed by constraints on the allocation matrix σ_X as stated in the following theorem.

Theorem 7 [Normal broadcast] *The dependence from Y to X induces a broadcast along the normal directions of the processor space if and only if:*

$$r_1 = \mathrm{Rank}(\sigma_X \cdot U)$$

where r_1 is the number of non-null rows of matrix $\sigma_X \cdot U$.

Proof. The columns of matrix $\sigma_X \cdot U$ define a basis of the image of the utilization set on the processor space. Therefore, $\mathrm{Rank}(\sigma_X \cdot U)$ represents the dimension of this set, and the maximal dimension of broadcast. In the case of an anticipated broadcast, $\mathrm{Rank}(\sigma_X \cdot U)$ *is* the dimension of the broadcast. However, for a true broadcast, the dimension may be smaller depending on the schedule function (cf. figure 4).

Recall that the row vectors of σ_X define the basis vectors of the virtual processor space. Therefore a null-row in matrix $\sigma_X \cdot U$ implies that the utilization set is orthogonal to the corresponding processor space basis vector. There is no communication at all along this direction. All the other directions are available to implement the broadcast. There are $r_1 = d_\sigma - r_0$ such directions, where r_0 is the number of null rows of $\sigma_X \cdot U$. If r_1 is equal to the dimension of the mapped utilization set *i.e.* $\mathrm{Rank}(\sigma_X \cdot U)$, then all the orthogonal directions can be used to broadcast. If it is smaller, then non-orthogonal directions have to be used to broadcast. Notice that it can of course not be larger.

□

3.4 Example

Let us consider the following example defined in $\mathbb{Z}^3$, where we focus on only one self-dependence:

$$X[i,j,k] = f(X[i-1,0,0]) \quad \left\{ \begin{pmatrix} i \\ j \\ k \end{pmatrix} \middle| \begin{array}{l} 0 \le i \le N, \\ 0 \le k \le N-i, \\ -N+i+k \le j \le N-i-k \end{array} \right\}$$

The utilization set is:

$$\mathrm{Util}\left(\begin{pmatrix} i_0 \\ 0 \\ 0 \end{pmatrix}, N \right) = \left\{ \begin{pmatrix} i_0+1 \\ j \\ k \end{pmatrix} \middle| \begin{array}{l} 0 \le k \le N-i_0-1, \\ -N+1+i_0+k \le j \le N-1-i_0-k \end{array} \right\}$$

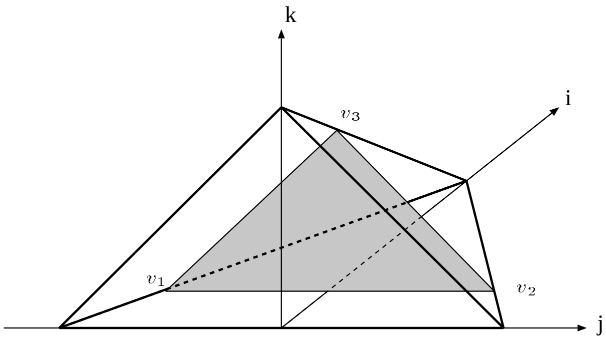


Figure 7: A utilization set in the iteration domain.

Since the dimension of the domain is 3, from the Hermite normal form decomposition of matrix R, we have $U = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$, $E = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$, and $V = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$.

Figure 7 shows the iteration domain and one instance of the utilization set (the grey triangle). Each utilization set related to an emission point $\begin{pmatrix} i_0 \\ 0 \\ 0 \end{pmatrix}$ is characterized by three vertices: $v_1 = \begin{pmatrix} i_0 + 1 \\ i_0 + 1 - N \\ 0 \end{pmatrix}$, $v_2 = \begin{pmatrix} i_0 + 1 \\ -i_0 - 1 + N \\ 0 \end{pmatrix}$ and $v_3 = \begin{pmatrix} i_0 + 1 \\ 0 \\ -i_0 - 1 + N \end{pmatrix}$.

Let us focus on a parallel solution characterized by a one-dimensional schedule $\lambda_X = (\lambda_1, \lambda_2, \lambda_3)$ and a two-dimensional allocation $\sigma_X = \begin{pmatrix} \sigma_1 & \sigma_2 & \sigma_3 \\ \sigma'_1 & \sigma'_2 & \sigma'_3 \end{pmatrix}$. From the vertices, it is easy to compute the causal constraints. Since the schedule is one-dimensional, lexicographical operator $\succ$ is equivalent to operator $>$.

$$\begin{cases} \lambda_1 > 0 \\ \lambda_2 = 0 \\ \lambda_3 \geq 0 \end{cases}$$

One can easily verify that the strong condition yields no valid solution. We show in the following that the local propagation can not be applied either, but that the broadcast is successful.

3.4.1 Propagation

Distant communications are removed using propagation if both theorems 2 and 3 can be satisfied. Since we consider a self-dependence, theorem 2 simplifies in:

$$\sigma_X \cdot (A_i - Id) \cdot E = 0$$

Vertex v_1 is characterized by matrix $A_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$. When applying theorem 2, we get $\sigma_2 = \sigma'_2 = 0$. In this case the transformation matrix would be: $T = \begin{pmatrix} \lambda_1 & 0 & \lambda_3 \\ \sigma_1 & 0 & \sigma_3 \\ \sigma'_1 & 0 & \sigma'_3 \end{pmatrix}$. Since this matrix is not of full rank, it is not valid.

We get exactly the same result with vertex v_2.

Vertex v_3 is characterized by matrix $A_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}$. When applying theorem 2, we get $\sigma_3 = \sigma'_3 = 0$, which does not induce any incompatibility yet. Let us now apply theorem 3. The generating vectors of the cone related to vertex v_3 are given by matrix $\overline{U} = \begin{pmatrix} 0 & 0 \\ 1 & -1 \\ -1 & -1 \end{pmatrix}$. Theorem 3 states that this propagation is possible only if $\lambda_X \overline{u_j} \succ 0$ for both column vectors of $\overline{U}$. This constraint implies $\lambda_3 < 0$, which is incompatible with the causal constraints.

To conclude, removing distant communications using propagation is not possible in this problem, since none of the vertices allows the determination of a valid space-time transformation according to theorem 2 and 3.

3.4.2 True broadcast

Let us try to find a solution with a two-dimensional broadcast. As stated by theorem 4, the constraints on the schedule function simplify into: there exists a vertex v_i of Util $\left(\begin{pmatrix} i_0 \\ 0 \\ 0 \end{pmatrix}, N \right)$, whose corresponding matrix $\overline{U}$ is such that: $\lambda_X \cdot \overline{U} = 0$. With vertex v_1, we have $\overline{U} = \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix}$. It results in $\lambda_2 = \lambda_3 = 0$. It can easily be verified that choosing the other vertices v_2 or v_3 results in the same constraints.

Let us now verify if it is possible to localize this broadcast, by applying theorem 5. This theorem states that the allocation matrix must satisfy: $\sigma_X \cdot V + \sigma_X \cdot U \cdot M = \sigma_X \cdot E$, where M is a constant vector, that is:

$$\sigma_X \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \sigma_X \cdot \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot M = \sigma_X \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

By choosing $M = 0$, this condition holds, without any new constraints on σ_X.

Therefore, a two-dimensional true local broadcast is obtained with any full rank transformation matrix $T = \begin{pmatrix} \lambda_1 & 0 & 0 \\ \sigma_1 & \sigma_2 & \sigma_3 \\ \sigma'_1 & \sigma'_2 & \sigma'_3 \end{pmatrix}$.

Normal broadcast. The normal broadcast condition, given by theorem 7, focus on matrix $\sigma_X \cdot U$ which is equal to $\begin{pmatrix} \sigma_2 & \sigma_3 \\ \sigma'_2 & \sigma'_3 \end{pmatrix}$. Since matrix T must be of full rank, submatrix $\begin{pmatrix} \sigma_2 & \sigma_3 \\ \sigma'_2 & \sigma'_3 \end{pmatrix}$ must also be of full rank. Therefore, $\text{Rank}(\sigma_X \cdot U) = 2$, and it has 2 non-null rows. Theorem 7 applies: the broadcast is realized along the orthogonal directions of the two-dimensional processor space.

3.4.3 Anticipated broadcast

The condition for true broadcast is $\lambda_3 = 0$, added to the causal constraints $\lambda_1 > 0$ and $\lambda_2 = 0$. If due to other dependences, the schedule vector can not meet the true broadcast constraint, for instance if $\lambda_3 > 0$, then anticipated broadcast may apply.

The transformation matrix is $T = \begin{pmatrix} \lambda_1 & 0 & \lambda_3 \\ \sigma_1 & \sigma_2 & \sigma_3 \\ \sigma'_1 & \sigma'_2 & \sigma'_3 \end{pmatrix}$. The directions of broadcast are determined according to submatrix $\sigma_X \cdot U = \begin{pmatrix} \sigma_2 & \sigma_3 \\ \sigma'_2 & \sigma'_3 \end{pmatrix}$ as stated by theorem 7.

Several cases may occur depending on the rank of $\sigma_X \cdot U$.

- if $\text{Rank}(\sigma_X \cdot U) = 0$, then transformation matrix T is not of full rank, and it is not valid.
- if $\text{Rank}(\sigma_X \cdot U) = 1$ and there are no null rows in $\sigma_X \cdot U$, then there is one-dimensional broadcast, but it is not directed by the basis vectors of the processor space, because the condition given by theorem 7 does not hold. A valid transformation matrix in such a case is for instance $T = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$.
- if $\text{Rank}(\sigma_X \cdot U) = 1$ and there is one null-row in $\sigma_X \cdot U$, then the one-dimensional broadcast is directed along one axis of the processor grid. A valid transformation matrix in such a case is for instance $T = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$.
- if $\text{Rank}(\sigma_X \cdot U) = 2$, then matrix $\sigma_X \cdot U$ does not have any null-row, and theorem 7 applies. There is a two-dimensional broadcast parallel to the

processor grid directions. A valid transformation matrix in such a case is for instance $T = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

4 A heuristic for communication optimization

A system of affine recurrence equations is of course characterized by several dependences which most of the time are incompatible in terms of communication elimination. In many practical cases, communication-free solutions result in only one virtual processor. Therefore, our objective is to propose a heuristic which eliminates as many distant communications as possible.

4.1 Communication minimization strategies

As discussed in this paper, we have proposed three different ways to eliminate distant communications relatively to one dependence:

- the strong condition (theorem 1). It maps all the points of one utilization set on one processor located at a constant distance of the related emission processor. This condition may further be strengthened by zeroing the remaining communication vector (corollary 1).
- the local propagation conditions (theorem 2 and 3). They only apply on non-degenerate utilization sets. Hence the local propagation conditions applied to the degenerate case are equivalent to the strong condition.
- the broadcast conditions. They also apply on non-degenerate utilization sets. They state constraints first on the schedule matrix in order to ensure true broadcast (theorem 4), and then -if requisite- on the allocation matrix to guarantee local broadcast (theorem 5). If these conditions can not be satisfied, alternative solutions using anticipated broadcast may be considered, and in this case memory cost must be evaluated (theorem 6). In both cases, we may further constrain the broadcast, along the directions of the processor grid (theorem 7).

All these conditions state constraints on the allocation functions and/or on the schedule functions. The dependences will be sucessively analyzed, and their related constraints on the transformation functions will be intersected.

Depending on the dimension of the utilization sets, different strategies to eliminate distant communications may be applied. If $dim(\mathrm{Util}_{Eq,Y}) = 0$, only the *strong-removal strategy*, corresponding to the strong condition applies. If $dim(\mathrm{Util}_{Eq,Y}) \geq 1$, there are three possibilities: the strong-removal strategy, the local propagation strategy, and the broadcast strategy. In the following, we consider that broadcast solutions are possible in the target architecture. If our heuristic has to be used where broadcast is prohibited, then it has to be simplified accordingly. Recall that normal broadcast, *i.e.* along the directions of the processor grid, or local broadcast, may be interesting in some specific cases. The heuristic we present in this paper includes these communication strategies, but they can of course be removed if such solutions are irrelevant.

Considering that broadcast -when allowed- is always an efficient solution, the order in which we apply the different strategies for a given non-degenerate dependence is the following:

1. true broadcast strategy, successively applied to all the vertices of the utilization set.
2. strong-removal strategy,
3. local propagation strategy, successively applied to all the vertices of the utilization set. The condition on the schedule given by theorem 3 can not be satisfied for many vertices. Hence they would either be incompatible with the causal constraints, or induce parameterized schedule matrices. Therefore, a first step of the algorithm consists in eliminating all the invalid vertices relatively to local propagation.

If broadcast is available, an alternative solution to local propagation is to consider anticipated broadcast, which requires memory as stated by theorem 6. Once the transformations will be eventually determined, the effective memory cost will be computed and the anticipated broadcast will be retained if it meets the target architecture ressources.

4.2 Building a tree of valid space-time transformations

Our heuristic successively analyzes the dependences. They are ordered so as to deal first with the dependences which could result in the largest number of distant communications. Recall that uniform self-dependences always result in local communications (cf. 3.2.3) and are therefore taken into account after all the affine and cross dependences.

The number of distant communications accountable to one dependence is in the worst case equal to $\mathrm{Card}(\mathrm{Emit}_{Eq,Y}(p))$. This occurs when neither the strong condition nor the local propagation condition can be satisfied. In this case, there is one distant communication for each emission point, the propagation along the utilization processors being always local. Therefore, our heuristic orders the dependences according to decreasing values of $dim(\mathrm{Emit}_{Eq,Y})$.

For several dependences with the same dimension of their emission sets, we will analyze them according to increasing values of their utilization set dimensions. Hence, for 0-dimensional utilization sets, only the strong condition may be satisfied, whereas for 1 or multi-dimensional utilization sets, the three types of conditions may be satisfied. Moreover, the larger the dimension of the utilization set is, the larger the number of choices is: different vertices to apply the local propagation conditions, different possible dimensions and directions for the broadcast conditions.

The algorithm we propose consists in building a tree of solutions. At each node of the tree we store a set of constraints on the schedule and on the allocation functions. The constraints associated with the root of the tree are the causal constraints. Each level of the tree is associated with one dependence. The levels are successively built, according to the dependence ordering given hereabove. For a given node, its direct descendants are built as follows:

- each descendant corresponds to one strategy of communication minimization. Among these strategies, we include a "do-nothing" strategy stating no new constraints for this dependence. This latter strategy ensures that there is at least one valid solution to the whole problem. In the worst case, this solution only meets the causal constraints.
- the constraints related to the descendant result from the intersection of its direct ancestor constraints and the ones given by the applied strategy. If this new set of constraints has no solution, the node is not added to the tree.

Due to the "do-nothing" strategy, all the paths in the tree have the same length. The leaves represent the various parallel solutions to the PARE.

4.3 Estimation of the efficiency of a solution

In order to estimate the efficiency of the various solutions, the algorithm determines for each node the *volume* of distant and local communications. This volume is expressed in terms of the dimensions of the utilization and emission sets. A similar approach based on communication volume is used by Feautrier [10]. The volume is stored in three arrays, called the *communication-cost arrays*, one for the distant communications, another one for the local communications, a third one for the broadcast communications. This third array stores independently the communication volume for each type of broadcast, when local or normal broadcasts are considered. Each entry d in these arrays corresponds to the number of d-dimensional communications.

Creating a node requires three steps: updating the tree structure, labeling the node with its set of constraints, and initializing the communication-cost arrays. The strategies applied at each node to build its direct descendants related to a given dependence are:

1. true broadcast (thm. 4), successively applied to each valid integer vertex v_i of the utilization set.

   Theorem 4 decomposes matrix $\overline{U}$ into two submatrices $\overline{U_1}$ and $\overline{U_2}$, where only $\overline{U_1}$ carries the broadcast. In order to efficiently decompose $\overline{U}$, we proceed as follows. (1) Determine the director vectors $\overline{e}$ of the edges of the utilization set. These vectors correspond to vectors $\pm\overline{u_j}$ at a given vertex. (2) Intersect constraint $\lambda_X \overline{e} = 0$ with the direct ancestor constraints, and only retain vectors $\overline{e}$ satisfying this set of constraints. (3) Scan the vertices and for each of them, determine the possible $\overline{U_1}$ based on the intersections of the constraints related to vectors $\overline{e}$. Depending on the type of allowed broadcast, we may either determine the largest possible $\overline{U_1}$, or restrict to k-columns $\overline{U_1}$ for k-dimensional broadcast.

If $\overline{U_1}$ is not empty, a new node is created. The communication-cost arrays are updated with a new set of broadcast communications of dimension dim_{Emit}, and a new set of local communications of dimension $(dim_{\text{Emit}} + \text{Rank}(\overline{U_2}))$.

1b. If local broadcast is allowed, then add the condition given by theorem 5 to the previous constraints. If they can be satisfied, create a new node. There is a new set of local broadcast communications of dimension dim_{Emit} to be added to the communication-cost arrays.

1c. If normal broadcast is allowed, proceed in the same way using theorem 7.

2. strong-removal (thm. 1) and elimination of the remaining local communications (cor. 1). Add $(\sigma_X - \sigma_Y \cdot R) \cdot \mathcal{B} = 0$ and $\rho(p) + c = 0$ to the constraints of the direct ancestor. If this set of constraints has a solution, a new node is created. The resulting allocation functions project onto the same virtual processor all the utilization points and their corresponding emission points. There are therefore no new communications and no updating of the communication-cost arrays.

3. strong-removal (thm. 1). Add $(\sigma_X - \sigma_Y \cdot R) \cdot \mathcal{B} = 0$ to the constraints of the direct ancestor. If this set of constraints has a solution, create a new node. A new set of local communications of dimension dim_{Emit} due to $\rho(p)$ is added to the communication-cost arrays.

4. local propagation (thm. 2 and 3) successively applied to each valid integer vertex v_i of the utilization set. The new constraints on the allocation functions are given by theorem 2 and the new constraints on schedule function λ_X by theorem 3. If added to the initial ancestor constraints, they can all be satisfied, a new node is created. The communication-cost arrays are updated with two new sets of local communications: the first one due to $\rho'(p)$ of dimension dim_{Emit} and the second one of dimension $(dim_{\text{Emit}} + dim_{\text{Util}})$ resulting from the propagation along the utilization vectors. Notice that this evaluation gives an upper bound of the propagation cost. Hence, the dimension of the utilization sets *mapped* onto the processor space is the effective dimension of these communications. This dimension is $\text{Rank}(\sigma_X \cdot U)$. However at this stage of the process, σ_X is unknown, therefore $\text{Rank}(\sigma_X \cdot U)$ is unknown and the communication cost is upper bounded by $dim_{\text{Util}} = \text{Rank}(U)$.

5. do-nothing. Create a new node without new constraints. The communication-cost arrays are updated with a new set of distant communication of dimension dim_{Emit}, and a new set of local communication of dimension $(dim_{\text{Emit}} + dim_{\text{Util}})$.

The algorithm generating the descendants of a node is detailed in appendix A.

It is obviously an expensive algorithm since an integer linear program solver is called at each node creation. However, the height of the tree is equal to the number of dependences of a problem plus one, which is not very large for most practical problems (less than 10 for classical linear algebra problems). At a given level of the tree, the number of descendants of any node is bounded by $7 + nv$ where nv is the number of valid integer vertices of the corresponding utilization set ($4 + nv$ if we do not take into account local and normal broadcast).

4.4 Choosing an efficient solution

Once the tree is constructed, each leaf represents a valid space-time transformation as a set of constraints on the allocation and schedule functions. Each system of constraints is solved in order to find a numerical solution associated with each leaf.

In order to choose an efficient solution, several criteria may be taken into account: communication minimization, distant communication minimization, optimal latency, memory cost, number of virtual processors.

- If we are only concerned with communication optimization, then the communication cost arrays associated with each leaf must be compared to choose the best one. Recall that the local propagation communication costs may in some cases be upper bounded. Since at this stage the transformation matrices are computed, one can compute the effective local propagation communication cost $\mathrm{Rank}(\sigma_X \cdot U)$ (cf. item 4, section 4.3).

- If the latency is the main issue, we first compute the latency associated with each leaf. The latency is parameterized by the problem parameters, and is computed using the algorithm proposed by Loechner and Wilde [20, 17]. Notice that the solution corresponding to the best latency may yield more communications than other ones (in particular broadcast communications). If necessary, a trade-off must be found between latency optimization and communication optimization.

- If the number of virtual processors must be optimized, then we compute the parameterized number of processors required by a given solution using the Ehrhart-Clauss algorithm [3].

- In the special case where anticipated broadcast is allowed, memory costs must be evaluated. For each leaf, let us consider the dependences that have been realized with the do-nothing strategy. This strategy can be replaced by an anticipated broadcast. The memory cost is computed according to theorem 6. The costs related to all these anticipated broadcasts are added together, and if the final memory cost meets the target architecture ressources, the solution is kept. Notice that when a do-nothing strategy is replaced by an anticipated broadcast, the communication cost arrays related to the examined dependence have to be modified: the corresponding

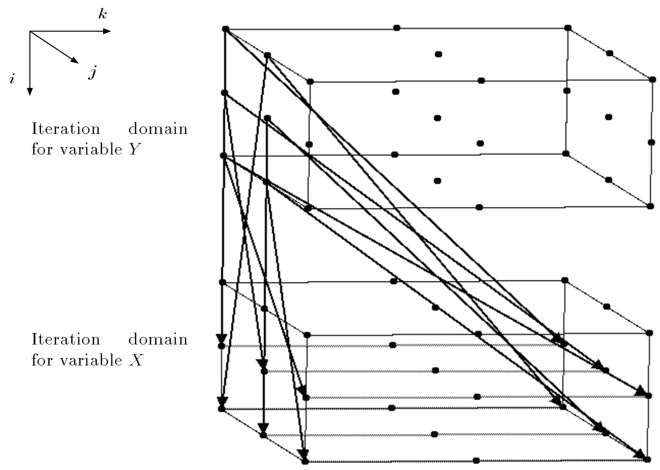


Figure 8: First dependence, from $Y[j, i-1, 0]$ to $X[i, j, k]$, for $N = 2$.

local and distant communications are zeroed and a new broadcast communication of dimension $dim(\mathrm{Emit}_{Eq,Y})$ is added. This computation is realized using a tree traversal from the leaf to the root.

5 An example

Consider the following PARE depending on one parameter N:

$$X[i,j,k] = Y[j, i-1, 0] + 2 \quad D_1 = \left\{ \begin{pmatrix} i \\ j \\ k \end{pmatrix} \middle| \begin{array}{l} 1 \le i \le N, \\ 0 \le j \le N, \\ 0 \le k \le N \end{array} \right\} \tag{1}$$

$$Y[i,j,k] = X[0, 0, k-1]/5 \quad D_2 = \left\{ \begin{pmatrix} i \\ j \\ k \end{pmatrix} \middle| \begin{array}{l} 0 \le i \le N, \\ 0 \le j \le N, \\ 1 \le k \le N \end{array} \right\} \tag{2}$$

In order to simplify the notations in the following, we call dependence (1) the dependence from Y to X defined in equation (1), and dependence (2) the dependence from X to Y defined in equation (2). The corresponding sets and

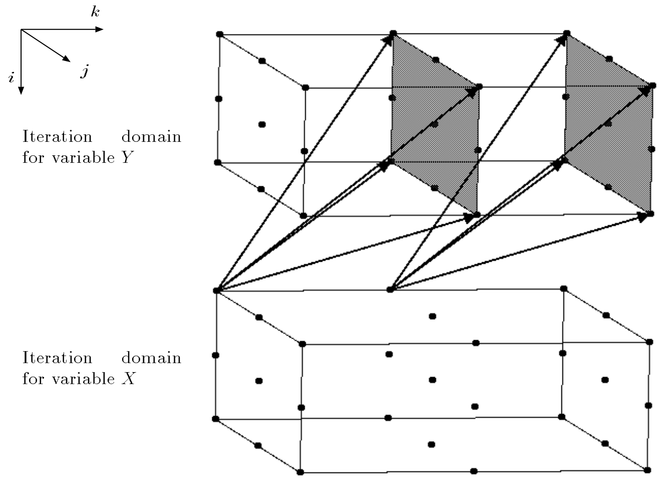


Figure 9: Second dependence, from $X[0,0,k-1]$ to $Y[i,j,k]$, for $N=2$.

vectors are simplified accordingly. The utilization sets are:

$$\begin{aligned}
\mathrm{Util}_1\left(\begin{pmatrix} i_0 \\ j_0 \\ k_0 \end{pmatrix}, N\right) &= \left\{ \begin{pmatrix} j_0+1 \\ i_0 \\ k \end{pmatrix} \middle| \, 0 \le k \le N \right\} \\
\mathrm{Util}_2\left(\begin{pmatrix} i_0 \\ j_0 \\ k_0 \end{pmatrix}, N\right) &= \left\{ \begin{pmatrix} i \\ j \\ k_0+1 \end{pmatrix} \middle| \, 0 \le i \le N, 0 \le j \le N \right\}
\end{aligned}$$

Util_1 has two vertices and is of dimension 1, Util_2 has four vertices, and is of dimension 2. These utilization sets are respectively represented figure 8 and 9, for $N=2$. The vertices coordinates, depending on $z_0=(i_0,j_0,k_0)$ and N, are given in the next table.

5.1 Processor array of dimension 2

Let us map this PARE on a two-dimensional array of virtual processors using one-dimensional schedule functions. Let the allocation matrices be:

$$\sigma_X = \begin{pmatrix} \sigma_{X1} & \sigma_{X2} & \sigma_{X3} \end{pmatrix}$$

$$\sigma_Y = \begin{pmatrix} \sigma_{Y1} & \sigma_{Y2} & \sigma_{Y3} \end{pmatrix}$$

where σ_{X*} and σ_{Y*} are 2-dimensional column vectors; allocation constants γ_X and γ_Y are 2-dimensional column vectors. Let the schedule matrices be $\lambda_X = \begin{pmatrix} \lambda_{X1} & \lambda_{X2} & \lambda_{X3} \end{pmatrix}$ and $\lambda_Y = \begin{pmatrix} \lambda_{Y1} & \lambda_{Y2} & \lambda_{Y3} \end{pmatrix}$ where λ_{X_*} and λ_{Y_*} are integer

dep.	vertex	coordinates	$\overline{U}$	new constraints on the schedule $\lambda \cdot \overline{u_j} \succ 0, \quad \forall \overline{u_j} \in \overline{U}$
(1)	v_1	$\begin{pmatrix} j_0+1 \\ i_0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$	$\lambda_{X3} > 0$
	v_2	$\begin{pmatrix} j_0+1 \\ i_0 \\ N \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$	$\lambda_{X3} < 0$ incompatible with (3)
(2)	v'_1	$\begin{pmatrix} 0 \\ 0 \\ k_0+1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$	$\begin{cases} \lambda_{Y1} > 0 \\ \lambda_{Y2} > 0 \end{cases}$
	v'_2	$\begin{pmatrix} 0 \\ N \\ k_0+1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \\ 0 & 0 \end{pmatrix}$	$\begin{cases} \lambda_{Y1} > 0 \\ \lambda_{Y2} < 0 \end{cases}$ incompatible with (3)
	v'_3	$\begin{pmatrix} N \\ 0 \\ k_0+1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$	$\begin{cases} \lambda_{Y1} < 0 \\ \lambda_{Y2} > 0 \end{cases}$ incompatible with (3)
	v'_4	$\begin{pmatrix} N \\ N \\ k_0+1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 0 & 0 \end{pmatrix}$	$\begin{cases} \lambda_{Y1} < 0 \\ \lambda_{Y2} < 0 \end{cases}$ incompatible with (3)

Table I: Constraints on the schedule for each vertex.

constants. Schedule constants α_X and α_Y are integers, and we impose in the following that $\alpha_X = \alpha_Y$.

The causal constraints whose computation is detailed in appendix B, are the following. Since the schedule functions are one-dimensional, the $\succ$ operator is equivalent to operator $>$.

$$\begin{cases} \alpha_X = \alpha_Y \\ \lambda_{X1} > 0 \\ \lambda_{Y3} > 0 \\ \lambda_{X1} \geq \lambda_{Y2} \geq 0 \\ \lambda_{X2} \geq \lambda_{Y1} \geq 0 \\ \lambda_{Y3} \geq \lambda_{X3} \geq 0 \end{cases} \tag{3}$$

Elimination of invalid vertices for local propagation. The local propagation, applied on a given vertex, imposes two types of constraints, one on the allocation (theorem 2), and one on the schedule (theorem 3). As explained section 4.1 the condition stated by theorem 3 together with the causal constraints, are not satisfied for many vertices. The constraints on the schedule corresponding to each vertex are summarized in table I.

According to the schedule constraints, the only vertices that can be used to apply the local propagation are v_1 for the first dependence and v'_1 for the second one. In the following, we consider local propagation along these two valid vertices.

Computation of the constraints associated with each strategy. Table II describes the new set of constraints associated with each of the communications optimization strategies. The detail of these computations is given in appendix B.

These constraints are combined to form pairs whose elements are respectively associated with the first and second dependence. One can verify that there are no incompatibilities in any of these possible pairs of constraints.

Computation of the tree of solutions.

case 1. Broadcast is allowed. In this case the construction of the tree is simplified since all the true broadcast conditions are compatible. It is therefore useless to build the rest of the tree. In the following, we choose to implement the largest possible broadcast. The intersection of the causal constraints together with all the true broadcast constraints results in:

$$\lambda_X = \begin{pmatrix}\lambda_{X1} & \lambda_{X2} & 0\end{pmatrix}$$
$$\lambda_Y = \begin{pmatrix}0 & 0 & \lambda_{Y3}\end{pmatrix}$$

where $\lambda_{X1} > 0$, $\lambda_{X2} \geq 0$ and $\lambda_{Y3} > 0$.

One can easily verify that all the local broadcast conditions can be moreover satisfied due to the large degree of liberty to choose constants m_1, m_2, m'_1, and m'_2 in table II.

If we further constrain the solutions to enforce normal broadcast, we get the following constraints (from theorem 7):

$$\text{Rank}\left(\sigma_X \cdot \begin{pmatrix}0\\0\\1\end{pmatrix}\right) = r_1\left(\sigma_X \cdot \begin{pmatrix}0\\0\\1\end{pmatrix}\right) \quad \text{for the first dependence,}$$
$$\text{Rank}\left(\sigma_Y \cdot \begin{pmatrix}1&0\\0&1\\0&0\end{pmatrix}\right) = r_1\left(\sigma_Y \cdot \begin{pmatrix}1&0\\0&1\\0&0\end{pmatrix}\right) \quad \text{for the second one,}$$

where $r_1(A)$ defines the number of non-null rows of matrix A.
Both sets of constraints can be satisfied. A possible numerical solution, among many others, is:

$$T_X = \begin{pmatrix}\lambda_X\\ \sigma_X\end{pmatrix} = \begin{pmatrix}1&0&0\\0&1&0\\1&0&1\end{pmatrix}$$
$$T_Y = \begin{pmatrix}\lambda_Y\\ \sigma_Y\end{pmatrix} = \begin{pmatrix}0&0&1\\1&0&0\\0&1&1\end{pmatrix}$$

and $\gamma_X = \begin{pmatrix}0\\0\end{pmatrix}$, $\gamma_Y = \begin{pmatrix}0\\0\end{pmatrix}$, $\alpha_X = \alpha_Y = 0$.

dep.	strategy	new constraints
(1)	true broadcast (case 1.)	$\lambda_{X3} = 0$
	local broadcast (case 1b.)	$\exists m_1 \in \mathbb{Z}, \sigma_{X1} + m_1 \sigma_{X3} = \sigma_{Y2}$ $\exists m_2 \in \mathbb{Z}, \sigma_{X2} + m_2 \sigma_{X3} = \sigma_{Y1}$
	strong-removal and elimination of all comm. (case 2.)	$\sigma_{X1} = \sigma_{Y2}$ $\sigma_{X2} = \sigma_{Y1}$ $\sigma_{X3} = 0$ $\gamma_Y = \gamma_X + \sigma_{Y2}$
	strong-removal (case 3.)	$\sigma_{X1} = \sigma_{Y2}$ $\sigma_{X2} = \sigma_{Y1}$ $\sigma_{X3} = 0$
	local propagation (v_1) (case 4.)	$\lambda_{X3} > 0$ $\sigma_{X1} = \sigma_{Y2}$ $\sigma_{X2} = \sigma_{Y1}$
(2)	true broadcast of dimension 2 (case 1.)	$\lambda_{Y1} = 0$ $\lambda_{Y2} = 0$
	true broadcast of dimension 1 (case 1.)	$\lambda_{Y1} = 0$ or $\lambda_{Y2} = 0$
	local broadcast (case 1b.)	$\exists m'_1, m'_2 \in \mathbb{Z}$, $\sigma_{Y3} + m'_1 \sigma_{Y1} + m'_2 \sigma_{Y2} = \sigma_{X3}$
	strong-removal (case 2. or 3.)	$\sigma_{Y1} = 0$ $\quad$ $\nexists\ \sigma_Y$ full row rank. $\sigma_{Y2} = 0$ $\quad \Rightarrow$ The strong condition $\sigma_{Y3} = \sigma_{X3}$ $\quad$ can not be satisfied
	local propagation (v'_1) (case 4.)	$\lambda_{Y1} > 0$ $\lambda_{Y2} > 0$ $\sigma_{X3} = \sigma_{Y3}$

Table II: New constraints on the schedule and the allocation for each strategy.

case 2. Broadcast is not allowed. The tree of solutions is represented figure 10. The first level of this tree is related to the dependence with the largest emission set, *i.e.* in this problem, dependence (1). The second level is related to dependence (2). Notice that there is no *strong-removal* branch at the second level, since the strong condition can not be satisfied, as mentioned in the previous table.

The volume of communications is computed at each node of the tree by two vectors representing respectively the dimensions (1, 2 or 3) of the sets of distant and local communications as described section 4.3 (the third array related to broadcast is not reported here).

Since broadcast is not allowed, the final step which consists in replacing local propagation by anticipated broadcast does not apply. In this case, the construction of this tree could be optimized in the following way. Once a leaf corresponding to a solution has been built, in a depth first construction, the corresponding volume of communications can be used as a reference to avoid the construction of nodes with worst communication costs. For this tree, the nodes corresponding to local propagation and do-nothing for the first dependence would have no children, since their communication vectors are greater than the communication vectors related to the leftmost leaf of the tree. With such an optimization, not all the paths of the tree have the same length.

The worst case (rightmost leaf) would result in one set of 1-dimensional distant communications, one set of 2-dimensional distant communications and two sets of 3-dimensional local communications. The retained solution is the leftmost leaf since it has the lowest communication cost. Since $\mathrm{Rank}(\sigma_Y \cdot U) = 2$ for the second dependence, the corresponding local propagation communication cost in the tree is not overestimated. It results in only two sets of local communications, one of dimension 3 the other of dimension 1.

It corresponds to the following set of constraints:

$$\begin{cases} \sigma_{X1} = \sigma_{Y2} \\ \sigma_{X2} = \sigma_{Y1} \\ \sigma_{X3} = 0 \\ \gamma_Y = \gamma_X + \sigma_{Y2} \\ \lambda_{Y1} > 0 \\ \lambda_{Y2} > 0 \\ \sigma_{X3} = \sigma_{Y3} \end{cases}$$

to be added to the causal constraints.

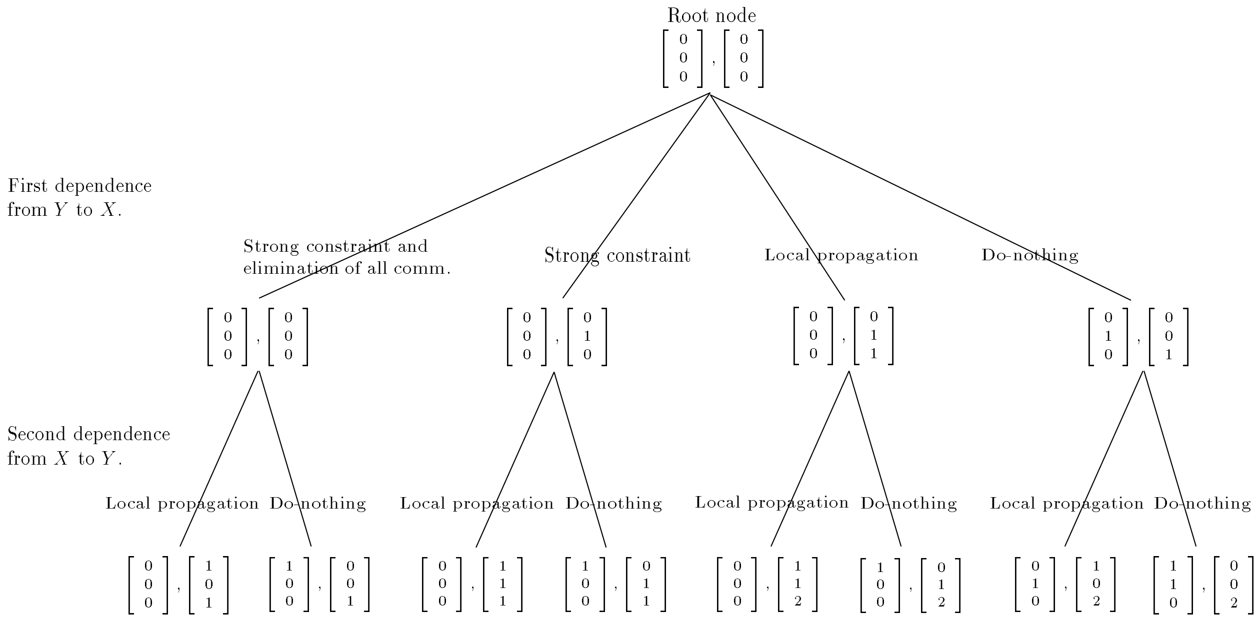


Figure 10: The tree of solutions when broadcast is not allowed.

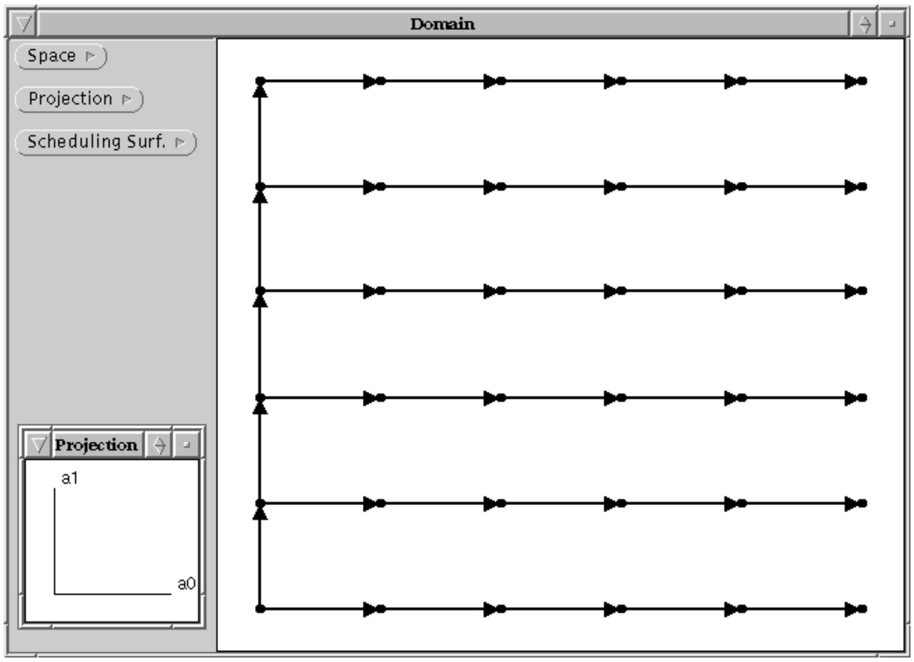


Figure 11: Communications on the processor array for the numerical solution given in case 2, for $N = 5$.

A possible numerical solution, among many others, is:

$$T_X = \begin{pmatrix} \lambda_X \\ \sigma_X \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

$$T_Y = \begin{pmatrix} \lambda_Y \\ \sigma_Y \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

and $\gamma_X = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $\gamma_Y = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $\alpha_X = \alpha_Y = 0$.

The two-dimensional processor array corresponding to this solution is represented figure 11, where the dots represent the processors, and the arrows between them symbolize the communications. All the commmunications are local and of distance 1.

dep.	vertex	coordinates	$\overline{U}$	new constraints on the schedule $\lambda \cdot \overline{u_j} \succ 0, \quad \forall \overline{u_j} \in \overline{U}$
(1)	v_1	$\begin{pmatrix} j_0+1 \\ i_0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$	$\lambda_{X3} \succ 0$
(2)	v'_1	$\begin{pmatrix} 0 \\ 0 \\ k_0+1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$	$\begin{cases} \lambda_{Y1} \succ 0 \\ \lambda_{Y2} \succ 0 \end{cases}$

Table III: Valid vertices for the local propagation.

5.2 Processor array of dimension 1

Let us map the same PARE on a one-dimensional array of virtual processors using two-dimensional schedule functions.

Let the allocation matrices be:

$$\sigma_X = \begin{pmatrix} \sigma_{X1} & \sigma_{X2} & \sigma_{X3} \end{pmatrix}$$

$$\sigma_Y = \begin{pmatrix} \sigma_{Y1} & \sigma_{Y2} & \sigma_{Y3} \end{pmatrix}$$

where σ_{X*} and σ_{Y*} are integer; Allocation constants γ_X and γ_Y are integer. Let the schedule matrices be $\lambda_X = \begin{pmatrix} \lambda_{X1} & \lambda_{X2} & \lambda_{X3} \end{pmatrix}$ and $\lambda_Y = \begin{pmatrix} \lambda_{Y1} & \lambda_{Y2} & \lambda_{Y3} \end{pmatrix}$ where λ_{X_*} and λ_{Y_*} are integer vectors of dimension 2. Schedule constants α_X and α_Y are integer vectors of dimension 2, and we impose in the following that $\alpha_X = \alpha_Y$.

The causal constraints whose computation is detailed in appendix B are:

$$\begin{cases} \alpha_X = \alpha_Y \\ \lambda_{X1} \succ 0 \\ \lambda_{Y3} \succ 0 \\ \lambda_{X1} \succeq \lambda_{Y2} \succeq 0 \\ \lambda_{X2} \succeq \lambda_{Y1} \succeq 0 \\ \lambda_{Y3} \succeq \lambda_{X3} \succeq 0 \end{cases} \tag{4}$$

Elimination of invalid vertices for local propagation. The local propagation conditions on the schedule (theorem 3) when intersected with the causal constraints, result in only two valid vertices given in table III.

Computation of the constraints associated with each strategy. Table IV describes the new set of constraints associated with each of the communications optimization strategies. Details regarding these computations is given in appendix B.

These constraints are combined to form pairs whose elements are respectively associated with the first and second dependence. Notice that there is an incompatibility between the two strong conditions. Since $\sigma_{X3} = 0$ for the

dep.	strategy	new constraints
(1)	true broadcast (case 1.)	$\lambda_{X3} = 0$
	local broadcast (case 1b.)	$\exists m_1 \in \mathbb{Z}, \sigma_{X1} + m_1 \sigma_{X3} = \sigma_{Y2}$ $\exists m_2 \in \mathbb{Z}, \sigma_{X2} + m_2 \sigma_{X3} = \sigma_{Y1}$
	strong-removal and elimination of all comm. (case 2.)	$\sigma_{X1} = \sigma_{Y2}$ $\sigma_{X2} = \sigma_{Y1}$ $\sigma_{X3} = 0$ $\gamma_Y = \gamma_X + \sigma_{Y2}$
	strong-removal (case 3.)	$\sigma_{X1} = \sigma_{Y2}$ $\sigma_{X2} = \sigma_{Y1}$ $\sigma_{X3} = 0$
	local propagation (v_1) (case 4.)	$\lambda_{X3} \succ 0$ $\sigma_{X1} = \sigma_{Y2}$ $\sigma_{X2} = \sigma_{Y1}$
(2)	true broadcast of dimension 2 (case 1.)	$\lambda_{Y1} = 0$ $\quad$ $\nexists\ \lambda_Y$ full row rank. and $\quad \Rightarrow$ Two-dimensional broadcast $\lambda_{Y2} = 0$ $\quad$ is impossible.
	true broadcast of dimension 1 (case 1.)	$\lambda_{Y1} = 0$ or $\lambda_{Y2} = 0$
	local broadcast (case 1b.)	$\exists m'_1, m'_2 \in \mathbb{Z},$ $\sigma_{Y3} + m'_1 \sigma_{Y1} + m'_2 \sigma_{Y2} = \sigma_{X3}$
	strong-removal and elimination of all comm. (case 2.)	$\sigma_{Y1} = 0$ $\sigma_{Y2} = 0$ $\sigma_{Y3} = \sigma_{X3}$ $\gamma_X = \gamma_Y + \sigma_{X3}$
	strong-removal (case 3.)	$\sigma_{Y1} = 0$ $\sigma_{Y2} = 0$ $\sigma_{Y3} = \sigma_{X3}$
	local propagation (v'_1) (case 4.)	$\lambda_{Y1} \succ 0$ $\lambda_{Y2} \succ 0$ $\sigma_{X3} = \sigma_{Y3}$

Table IV: New constraints associated with each strategy.

first dependence and $\sigma_{Y1} = \sigma_{Y2} = 0$, $\sigma_{Y3} = \sigma_{X3}$ for the second one, it results $\sigma_{Y3} = 0$, and allocation matrix σ_Y would be null. There are no other incompatibilities in any of these pairs of constraints.

Computation of the tree of solutions.

case 1. Broadcast is allowed. In the following, we choose to implement the largest possible broadcast. Recall that true broadcast of dimension 2 is impossible for the second dependence, as shown in the previous table. The construction of the tree is simplified since all the possible true broadcast conditions are compatible. It is therefore useless to build the rest of the tree.

There are two 1-dimensional broadcast solutions for dependence (2): the first one corresponding to constraint $\lambda_{Y1} = 0$, and the second one corresponding to $\lambda_{Y2} = 0$. The intersection of the causal constraints together with the true broadcast constraints results in:

- first broadcast solution: $\lambda_{Y1} = 0$.

$$\begin{cases} \lambda_{X3} = 0 \\ \lambda_{Y1} = 0 \\ \lambda_{X1} \succ 0 \\ \lambda_{Y3} \succ 0 \\ \lambda_{X2} \succeq 0 \\ \lambda_{X1} \succeq \lambda_{Y2} \succeq 0 \\ \alpha_X = \alpha_Y \end{cases}$$

  By choosing constants $m_* = 0$ the local broadcast constraints can be satisfied. A possible numerical solution, among many others, is:

$$T_X = \begin{pmatrix} \lambda_X \\ \sigma_X \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

$$T_Y = \begin{pmatrix} \lambda_Y \\ \sigma_Y \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

  and $\gamma_X = \gamma_Y = 0$, $\alpha_X = \alpha_Y = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

- second broadcast solution: $\lambda_{Y2} = 0$.

$$\begin{cases} \lambda_{X3} = 0 \\ \lambda_{Y2} = 0 \\ \lambda_{X1} \succ 0 \\ \lambda_{Y3} \succ 0 \\ \lambda_{X2} \succeq \lambda_{Y1} \succeq 0 \\ \alpha_X = \alpha_Y \end{cases}$$

By choosing constants $m_* = 0$ the local broadcast constraints can be satisfied. A possible numerical solution, among many others, is:

$$T_X = \begin{pmatrix} \lambda_X \\ \sigma_X \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

$$T_Y = \begin{pmatrix} \lambda_Y \\ \sigma_Y \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

and $\gamma_X = \gamma_Y = 0$, $\alpha_X = \alpha_Y = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

We may further check that these solutions satisfy the normal broadcast constraints (from theorem 7).

case 2. Broadcast is not allowed. The tree of solutions is represented figure 12.

The volume of communications is computed at each node of the tree by two vectors representing respectively the dimensions (1, 2 or 3) of the sets of distant and local communications as described section 4.3 (the third array related to broadcast is not reported here).

The leftmost leaf corresponds to the following set of constraints, to be added to the causal constraints:

$$\begin{cases} \sigma_{X1} = \sigma_{Y2} \\ \sigma_{X2} = \sigma_{Y1} \\ \sigma_{X3} = 0 \\ \gamma_Y = \gamma_X + \sigma_{Y2} \\ \lambda_{Y1} \succ 0 \\ \lambda_{Y2} \succ 0 \\ \sigma_{X3} = \sigma_{Y3} \end{cases}$$

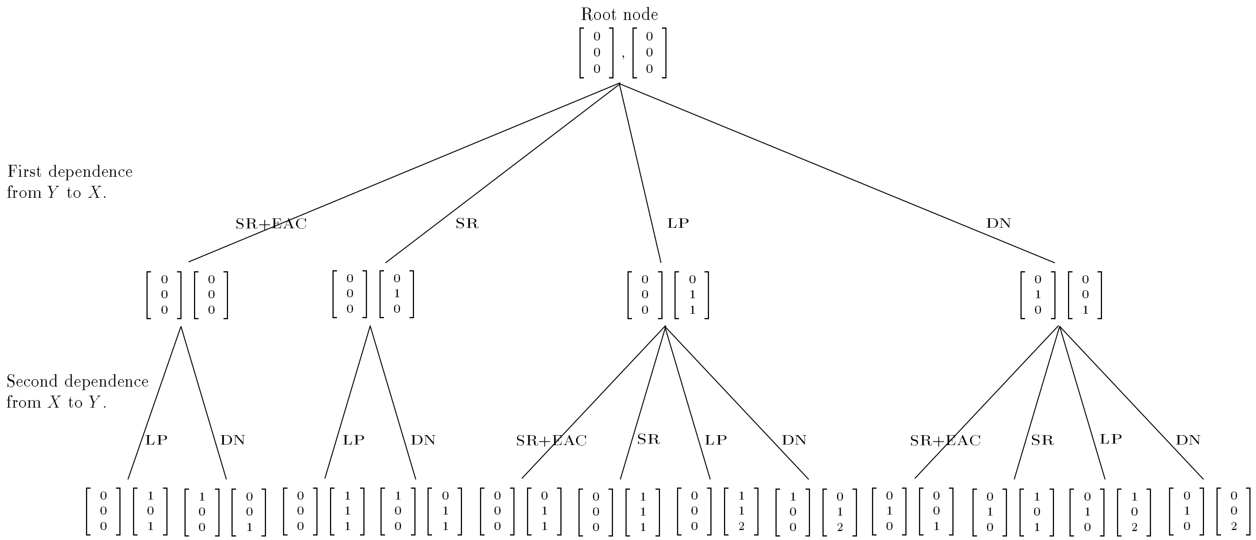
A possible numerical solution, among many others, is:

$$T_X = \begin{pmatrix} \lambda_X \\ \sigma_X \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

$$T_Y = \begin{pmatrix} \lambda_Y \\ \sigma_Y \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

and $\gamma_X = \gamma_Y = 0$, $\alpha_X = \alpha_Y = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

Notice that the local propagation cost for the second dependence is overestimated in the tree, since $\mathrm{Rank}(\sigma_X \cdot U) = 1 < dim(\mathrm{Util}_2) = 2$. The effective



SR+EAC = Strong-removal + elimination of all communications
SR = Strong-removal
LP = Local Propagation
DN = Do-nothing

Figure 12: The tree of solutions when broadcast is not allowed.

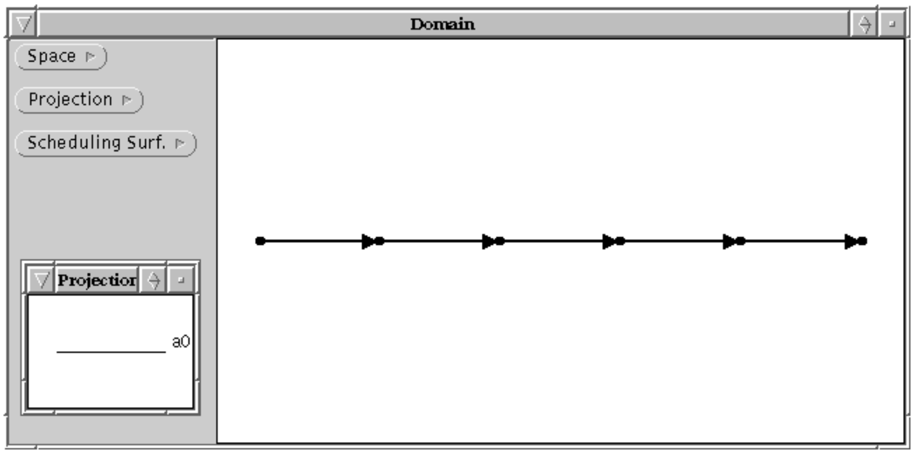


Figure 13: Communications on the processor array for the numerical solution given in case 2, for $N = 5$.

communication cost for this solution is given by vectors $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$. It corresponds to two sets of local communications, one of dimension 2, and the other of dimension 1. The same analysis for the other leaves of the tree shows that the best solution (with the lowest communication cost) is the leftmost leaf.

The one-dimensional processor array corresponding to this solution is represented figure 13. All the commmunications are local and of distance 1.

6 Conclusion

We have proposed a method to find efficient space-time transformations for a system of parameterized affine recurrence equations resulting in a set of constraints on the allocation and schedule functions. Besides the strong condition which is classical in other approaches, our approach includes a condition called local propagation to optimize communication cost [18]. The local propagation condition is based on an efficient mapping on the processor space of the utilization and emission sets. It is a necessary and sufficient condition to remove distant communications in the case of non-degenerate utilization sets.

Moreover, our approach takes advantage of broadcast capabilities when they are available. We consider in this paper two types of broadcasts. The first one, called true broadcast, guarantees that a set of utilization points is computed at the same time step. Therefore it imposes constraints on the schedule functions. If these constraints can not be satisfied, due to other constraints of the problem, then one may consider anticipated broadcast. It consists in broadcasting the data to a set of processors even though these processors do not use it immediately. They must therefore store it in memory. We have shown in this paper how to approximate the memory cost in this case. An exact computation of this cost has been realized in a further work [19]. Furthermore we have considered two specific cases of broadcast: local broadcast and normal broadcast, that can be useful for hardware design with regularity and good scalability.

Other works focus on communication optimization problems, through what is often called *placement*, *alignment* or *data distribution*. Since this problem has been proved to be NP-complete [7] various heuristics have been proposed. There are two classes of approaches based on the respect or not to the *owner computes rule*. When the owner computes rule is not respected, computations and data are allocated independently.

Let us consider various works conducted without respect to the owner computes rule. W. Kelly and W. Pugh [12] propose an algorithm that finds one degree of parallelism in loop nests, with the objective to minimize communications. The dependence analysis is based on extended direction/distance vectors. The algorithm examines successively all the transformations corresponding to loop permutations and loop reversals. Due to the fact that it verifies all possible candidates, it can not consider more general loop transformations such as skewing. It determines a computation distribution and assumes that this distribution phase will be followed by a computation reordering phase which is not actually included in their approach. As opposed to our approach, they only deal with the allocation part of the problem, without interaction with the schedule, and the possible transformations are more restricted. J. Anderson and M. Lam [1] propose an algorithm that aligns data and computations in order to optimize both parallelism and locality of a program. It is decomposed into three components: (1) partition which specifies the array elements and computations that are assigned to the same processor, (2) orientation which gives the correspondence between the data and computation dimensions and the processor dimensions, (3) displacement which specifies the offsets of the data and computation elements according to the processors. Besides the standard linear loop-nests transformations, they consider blocked decompositions (or tiling). Their algorithm produces parallel **forall** or **doacross** loops, without explicit communications. This work has been continued by A. Lim and M. Lam [15, 16, 14]. They introduce the concept of affine partition which is equivalent to our space-time transformation approach. The communication minimization is realized by opposing local communications to remote communications, without exploiting the advantage broadcast could bring as we do in this paper. Moreover, the communication cost is estimated in terms of the frequency of communication rather than the volume of communication which gives more precise information. The first step of their algorithm decomposes the initial program dependence graph into strongly connected components (SCC), each of them being parallelized partly independently. In our approach, we deal with a PARE considered as a whole, which could represent one of the SCC. In order to get the same generality as Lim and Lam's work, one can compute the reduced dependence graph of a whole problem and determine a PARE for each of its

SCC. A. Darte and Y. Robert [5] introduce the notion of communication graph and propose a method to optimize communications in uniform loop nests. This approach has been extended to affine loop nests by M. Dion and Y. Robert [7], using a more precise graph called the access graph. These last two approaches use a condition similar to the strong condition presented in this paper. To deal with residual communications they show how to optimize them using for instance broadcast or decomposing them into simpler communications [6].

Other heuristics are given with respect to the owner compute rule. J. Ramanujam and P. Sadayappan [25] address the problem of partitioning the data and allocating them so that the interprocessor communications are minimized. The problem is stated as a system of equations to be solved, using a matrix-based formulation. If the system has a solution then a communication-free partitioning does exist. When it has no solution the question of communication minimization is only discussed in the simple case of uniform dependences. P. Feautrier [10] proposes a greedy algorithm that cuts the edges of the dataflow graph according to their volume of communications. This approach is extended by A. Platonoff [23] where broadcasts and reductions are included in the process of placement determination. As mentioned in this paper, allocation and schedule are strongly related. This important concept is not taken into account by their approaches which deal exclusively with the allocation function.

The heuristic we propose can be refined in several ways. In the current version of the algorithm the volume of communications is estimated using the dimensions of the utilization and emission sets. Using the Ehrhart algorithm [4], the number of communications could be computed exactly as function of the parameters; an intermediate solution would be to compute only the highest degree coefficient of these functions. The comparison of these coefficients gives a good approximation of the communication cost. Finally, the local propagation developed in this approach only applies to integer vertices of the utilization sets. In many practical cases, these vertices are integer, but there are examples were the vertices are rational. Each rational vertex corresponds to a set of integer vertices, whose number depends on the number of facets intersecting at this rational vertex. For utilization sets of dimension greater than 2, they are not easy to compute, and further studies should be conducted to prove that the coordinates of these vertices are periodic numbers [4].

References

[1] J.M. Anderson and M.S. Lam. Global optimizations for parallelism and locality on scalable parallel machines. *ACM Sigplan Notices*, 28(6):112–125, 1993.

[2] Ph. Clauss. Counting solutions to linear and nonlinear constraints through Ehrhart polynomials: Applications to analyze and transform scientific programs. In *10th ACM Int. Conf. on Supercomputing, Philadelphia*, 1996.

[3] Ph. Clauss. Handling memory cache policy with integer points countings. In Lengauer, Griebl, and Gorlatch, editors, *Euro-Par'97, Passau*, pages 285–293. Springer-Verlag, LNCS 1300, August 1997.

[4] Ph. Clauss and V. Loechner. Parametric analysis of polyhedral iteration spaces. *Journal of VLSI Signal Processing*, 19(2):179–194, July 1998.

[5] A. Darte and Y. Robert. On the alignment problem. *Parallel Processing Letters*, 4(3):259–270, 1994.

[6] M. Dion, C. Randriamaro, and Y. Robert. How to optimize residual communications? Research Report 95-27, LIP, Ecole Normale Supérieure de Lyon, France, Octobre 1995.

[7] M. Dion and Y. Robert. Mapping affine loop nests : new results. In B. Hertzberger and G. Serazzi, editors, *Int. Conf. on High-Performance Computing and Networking*, pages 184–189. Springer Verlag, May 1995. Milan, Italy.

[8] P. Feautrier. Some efficient solutions to the affine scheduling problem, part 1 : one dimensional time. *Int. J. of Parallel Programming*, 21(5):313–348, October 1992.

[9] P. Feautrier. Some efficient solutions to the affine scheduling problem, part 2 : multidimensional time. *Int. J. of Parallel Programming*, 21(6), December 1992.

[10] P. Feautrier. Towards automatic distribution. *Parallel Processing Letters*, 4(3):233–244, 1994.

[11] P. Feautrier. Compiling for massively parallel architectures : a perspective. *Microprogramming and Microprocessors*, 41:425–439, 1995.

[12] W. Kelly and W. Pugh. Minimizing communication while preserving parallelism. In *10th ACM Int. Conf. on Supercomputing*, 1996. Philadelphia.

[13] C. Lengauer. Loop parallelization in the polytope model. In *CONCUR 93*, 1993.

[14] A.W. Lim, G.I. Cheong, and M.S. Lam. An affine partitioning algorithm to maximize parallelism and minimize communication. In *13th ACM Int. Conf. on Supercomputing*, 1999. Rhodes, Greece.

[15] A.W. Lim and M.S. Lam. Maximizing parallelism and minimizing synchronization with affine transforms. In *24th ACM SIGPLAN Symposium on Principles of Programming Languages*, 1997.

[16] A.W. Lim and M.S. Lam. Maximizing parallelism and minimizing synchronization with affine partitions. *Parallel Computing*, 24:445–475, 1998.

[17] V. Loechner. *Contribution à l'étude des polyèdres paramétrés et applications en parallélisation automatique*. PhD thesis, Université Louis Pasteur, Strasbourg, 1997. http://icps.u-strasbg.fr/pub-97/.

[18] V. Loechner and C. Mongenet. Solutions to the communication minimization problem for affine recurrence equations. In Lengauer, Griebl, and Gorlatch, editors, *Europar'97*, pages 328–337. Springer-Verlag, August 1997. LNCS 1300.

[19] V. Loechner and C. Mongenet. Memory cost due to anticipated broadcast. In *8th Workshop on Compilers for Parallel Computers, CPC*, January 2000. Aussois, France.

[20] V. Loechner and D. K. Wilde. Parameterized polyhedra and their vertices. *International Journal of Parallel Programming*, 25(6), December 1997.

[21] C. Mongenet. Affine dependence classification for communications minimization. *International Journal of Parallel Programming*, 25(6), December 1997.

[22] C. Mongenet, Ph. Clauss, and G.R. Perrin. A geometrical coding to compile affine recurrence equations on regular arrays. In *5th International Parallel Processing Symposium, IPPS'91*, 1991.

[23] A. Platonoff. Automatic data distribution for massively parallel computers. In *5th International Workshop on Compilers for Parallel Computers, Malaga University, Spain*, June 1995.

[24] P. Quinton and V. Van Dongen. The mapping of linear recurrence equations on regular arrays. *The Journal of VLSI Signal Processing*, 1:95–113, 1989.

[25] J. Ramanujam and P. Sadayappan. Compile-time techniques for data distribution in distributed memory machines. *IEEE Transactions on parallel and distributed systems*, 2(4):472–482, October 1991.

[26] W. Shang and Z. Shu. Data alignment of loop nests without nonlocal communications. In *IEEE Int. Conference on Application Specific Array Processors, ASAP'94*, pages 439–450. IEEE Computer Society Press, August 1994.

[27] D.K. Wilde. A library for doing polyhedral operations. Technical Report 785, IRISA, Rennes, France, 1993.

A Algorithm to build the tree of solutions

Generation of the descendants of a node

Input: index function, utilization and emission sets,
basis vectors related to the emission set $\varepsilon_{E,Y,j}$.

Output: the set of descendants of the node. Each node is characterized
by a set of constraints and an estimation of the volume of communication.

```
procedure GenDescendants ( node : current node
                           R, Q, r : matrices defining the index function g(z, p)
                           Util : parameterized polyhedron Util_{E,Y}(z_0, p)
                           v_i, \overline{U_i} : vertices of Util_{E,Y}(z_0, p) and corresponding \overline{U_i}
                           ε_j : emission vectors ε_{E,Y,j}
                         )
begin
   vector e;                 /* vector supporting an edge of Util */
   set VE;                   /* set of valid vectors e */
   node n;                   /* temporary node */
   constraints C, C';        /* temporary sets of constraints */

   /*** CASE 1.  True broadcast ***/
   /* compute valid vectors \overline{e}, and put them in set VE */
   VE = ∅
   for all e in SupportingEdges(Util) do
      if node→constraints ∩ {λ_X e = 0} ≠ ∅ then
         VE = VE ∪ {e}
      endif
   done
   /* scan the vertices and the corresponding matrices \overline{U} */
   for all v_i do
      /* function ValidCombinationsOfColumns(\overline{U}, VE) returns the      */
      /* set of combinations of the column vectors of \overline{U}, such that   */
      /* for each vector u, we have ±u ∈ VE.                                   */
      for all cc = ValidCombinationsOfColumns(\overline{U}, VE) do
         C = node→constraints
         for all u ∈ cc do
            C = C ∩ {λ_X u = 0}
         done
         if C ≠ ∅ then
            AddNode( C, dim(cc), 0, 0 )
         endif
      done
   done

   /*** CASE 2.  Strong condition + elim.  of all communications ***/
   C = node→constraints ∩ {(σ_X − σ_Y · R) · B = 0}
   if C ≠ ∅ then
      v_0 = first_vertex(D(p))
      C' = C ∩ {c(v_0) + ρ(p) = 0}
      if C' ≠ ∅ then
         AddNode ( C', 0, 0, 0 )                /* no new communications */
```

```
    endif

    /*** CASE 3.  Strong condition ***/
    AddNode ( C, 0, 0, dim_Emit )                 /* dim_Emit new local comms */
  endif

  /*** CASE 4.  Local propagation condition:  scan the vertices v_i ***/
  for all integer v_i do
    /* Compute the constraints on the schedule (theorem 3) */
    U‾ = EdgeVectors ( v_i, Util )
                                /* U‾:  matrix of generating vectors u_j‾ */
    C = node→constraints ∩ {∩_k ∩_j λ_k · u_j‾ ≥ 0}
    if C ≠ ∅ then
      C' = C ∩ {∩_j(σ_X · (v_i→A) − σ_Y)ε_j = 0}
                              /* Local propagation condition (theorem 2) */
      if C' ≠ ∅ then
        /* no new distant comms.  but dim_Emit + dim_Util and */
        /* dim_Emit new local comms */
        AddNode ( C', 0, 0, (dim_Emit + dim_Util) and dim_Emit )
        endif
      endif
    endif
  done

  /*** CASE 5.  Do-nothing ***/
  /* new distant and local communications */
  AddNode( node→constraints, 0, dim_Emit, (dim_Emit + dim_Util) )
end
```

B Constraints computation for the example of section 5

B.1 Computation of the causal constraints

The schedule matrices are respectively $\lambda_X = \begin{pmatrix}\lambda_{X1} & \lambda_{X2} & \lambda_{X3}\end{pmatrix}$ and $\lambda_Y = \begin{pmatrix}\lambda_{Y1} & \lambda_{Y2} & \lambda_{Y3}\end{pmatrix}$, where each λ_{X*} and λ_{Y*} is either an integer constant (for one-dimensional schedule functions), or an integer column vector of dimension 2 (for two-dimensional schedule functions).

The causal constraints are:

$$\left\{\begin{array}{llll} \lambda_X \begin{pmatrix} i \\ j \\ k \end{pmatrix} + \alpha_X \succ \lambda_Y \begin{pmatrix} j \\ i-1 \\ 0 \end{pmatrix} + \alpha_Y & & \begin{pmatrix} i \\ j \\ k \end{pmatrix} \in D_1 \\ \lambda_Y \begin{pmatrix} i \\ j \\ k \end{pmatrix} + \alpha_Y \succ \lambda_X \begin{pmatrix} 0 \\ 0 \\ k-1 \end{pmatrix} + \alpha_X & & \begin{pmatrix} i \\ j \\ k \end{pmatrix} \in D_2 \end{array}\right.$$

$$\Leftrightarrow \left\{\begin{array}{l} \lambda_{X1} i + \lambda_{X2} j + \lambda_{X3} k + \alpha_X \succ \lambda_{Y1} j + \lambda_{Y2}(i-1) + \alpha_Y \\ \hfill \forall j,k \in [0,N], \forall i \in [1,N] \\ \lambda_{Y1} i + \lambda_{Y2} j + \lambda_{Y3} k + \alpha_Y \succ \lambda_{X3}(k-1) + \alpha_X \\ \hfill \forall i,j \in [0,N], \forall k \in [1,N] \end{array}\right.$$

$$\Leftrightarrow \left\{\begin{array}{l} (\lambda_{X1} - \lambda_{Y2}) i + (\lambda_{X2} - \lambda_{Y1}) j + \lambda_{X3} k + (\alpha_X - \alpha_Y + \lambda_{Y2}) \succ 0 \\ \hfill \forall j,k \in [0,N], \forall i \in [1,N] \\ \lambda_{Y1} i + \lambda_{Y2} j + (\lambda_{Y3} - \lambda_{X3}) k + (\alpha_Y - \alpha_X + \lambda_{X3}) \succ 0 \\ \hfill \forall i,j \in [0,N], \forall k \in [1,N] \end{array}\right.$$

Since the arguments λ and α of the schedule functions are constant (not depending on the problem parameter N), we obtain for any value of parameter N:

$$\left\{\begin{array}{l} \lambda_{X1} - \lambda_{Y2} \succeq 0 \\ \lambda_{X2} - \lambda_{Y1} \succeq 0 \\ \lambda_{X3} \succeq 0 \\ (\alpha_X - \alpha_Y + \lambda_{Y2}) + (\lambda_{X1} - \lambda_{Y2}) \succ 0 \\ \lambda_{Y1} \succeq 0 \\ \lambda_{Y2} \succeq 0 \\ \lambda_{Y3} - \lambda_{X3} \succeq 0 \\ (\alpha_Y - \alpha_X + \lambda_{X3}) + (\lambda_{Y3} - \lambda_{X3}) \succ 0 \end{array}\right.$$

In order to simplify the latter computations, we restrict the set of solutions by choosing $\alpha_X = \alpha_Y$.

$$\left\{\begin{array}{l} \alpha_X = \alpha_Y \\ \lambda_{X1} \succ 0 \\ \lambda_{Y3} \succ 0 \\ \lambda_{X1} \succeq \lambda_{Y2} \succeq 0 \\ \lambda_{X2} \succeq \lambda_{Y1} \succeq 0 \\ \lambda_{Y3} \succeq \lambda_{X3} \succeq 0 \end{array}\right. \tag{5}$$

B.2 Dependence (1). Hermite normal form decomposition of matrix R

This dependence is characterized by the following matrices: $R = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$, $Q = 0$ and $r = \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix}$. The Hermite normal form decomposition of R is:

$$R \cdot \left[\begin{array}{cc|c} \multicolumn{2}{c}{\scriptstyle V} & \scriptstyle U \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array} \right] = \left[\begin{array}{cc|c} \multicolumn{2}{c}{\scriptstyle E} & \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{array} \right]$$

B.3 Dependence (1). True broadcast (case 1.)

Since dependence (1) is one-dimensional is is characterized by a single vector $\overline{e} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$. For a true broadcast to apply, constraint $\lambda_X \overline{e} = 0$ must be compatible with the causal constraints.

$$\lambda_X \overline{e} = 0 \qquad \Leftrightarrow \qquad \lambda_{X3} = 0$$

B.4 Dependence (1). Local broadcast (case 1b.)

The local broadcast condition (theorem 5) states that there exists an integer matrix $M = \begin{pmatrix} m_1 & m_2 \end{pmatrix}$ such that:

$$\begin{aligned}
& \sigma_X \cdot V + \sigma_X \cdot U \cdot M = \sigma_Y \cdot E \\
\Rightarrow \quad & \begin{pmatrix} \sigma_{X1} & \sigma_{X2} & \sigma_{X3} \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} \sigma_{X1} & \sigma_{X2} & \sigma_{X3} \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot M \\
& = \begin{pmatrix} \sigma_{Y1} & \sigma_{Y2} & \sigma_{Y3} \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{pmatrix} \\
\Rightarrow \quad & \begin{pmatrix} \sigma_{X1} & \sigma_{X2} \end{pmatrix} + \begin{pmatrix} \sigma_{X3} \end{pmatrix} \cdot M = \begin{pmatrix} \sigma_{Y2} & \sigma_{Y1} \end{pmatrix} \\
\Rightarrow \quad & \left\{ \begin{array}{l} \sigma_{X1} + m_1 \sigma_{X3} = \sigma_{Y2} \\ \sigma_{X2} + m_2 \sigma_{X3} = \sigma_{Y1} \end{array} \right.
\end{aligned}$$

B.5 Dependence (1). Strong-removal (cases 2. and 3.)

The strong condition (theorem 1) is:

$$(\sigma_X - \sigma_Y \cdot R) \cdot \mathcal{B} = 0 \qquad \text{with } \mathcal{B} = Id$$

$$\Rightarrow \quad \begin{pmatrix}\sigma_{X1} & \sigma_{X2} & \sigma_{X3}\end{pmatrix} - \begin{pmatrix}\sigma_{Y1} & \sigma_{Y2} & \sigma_{Y3}\end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = 0$$

$$\Rightarrow \quad \begin{cases} \sigma_{X1} - \sigma_{Y2} = 0 \\ \sigma_{X2} - \sigma_{Y1} = 0 \\ \sigma_{X3} = 0 \end{cases}$$

The computation of c (realized using one of the vertices of $\mathcal{D}_1$) results in $c = 0$. Therefore the condition that eliminates all communications (corollary 1) simplifies in $\rho(p) = 0$:

$$\begin{aligned} \rho(p) = 0 \quad &\Leftrightarrow \quad -\sigma_Y \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} - \gamma_Y + \gamma_X = 0 \\ &\Leftrightarrow \quad \gamma_Y = \sigma_{Y2} + \gamma_X \end{aligned}$$

B.6 Dependence (1). Local propagation relatively to valid vertex v_1 (case 4.)

The local propagation condition on the allocation functions for vertex v_1 is (theorem 2):

$$\begin{aligned} & (\sigma_X \cdot A_1 - \sigma_Y) \cdot E = 0 \\ \Leftrightarrow \quad & (\sigma_X \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} - \sigma_Y) \cdot \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{pmatrix} = 0 \\ \Leftrightarrow \quad & \begin{cases} \sigma_{X1} - \sigma_{Y2} = 0 \\ \sigma_{X2} - \sigma_{Y1} = 0 \end{cases} \end{aligned}$$

B.7 Dependence (2). Hermite normal form decomposition of matrix R

This dependence is characterized by the following matrices:

$$R = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad Q = 0 \qquad r = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$$

The Hermite normal form decomposition of R is:

$$R \cdot \begin{array}{c} \begin{array}{ccc} V & U & \; \end{array} \\ \left[\begin{array}{c|cc} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{array}\right] \end{array} = \begin{array}{c} E \\ \left[\begin{array}{c|cc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{array}\right] \end{array}$$

B.8 Dependence (2). True broadcast (case 1.)

Two-dimensional dependence (2) is characterized by two vectors $\overline{e_1} = \begin{pmatrix}1\\0\\0\end{pmatrix}$ and $\overline{e_2} = \begin{pmatrix}0\\1\\0\end{pmatrix}$. For a true broadcast to apply, constraint $\lambda_Y \overline{e_*} = 0$ must be compatible with the causal constraints.

$$\begin{aligned}\lambda_Y \overline{e_1} = 0 \quad &\Leftrightarrow \quad \lambda_{Y1} = 0\\ \lambda_Y \overline{e_2} = 0 \quad &\Leftrightarrow \quad \lambda_{Y2} = 0\end{aligned}$$

Since the the schedule matrix λ_Y must be of full row rank, these two constraints can not simultaneously hold for multi-dimensional schedules. Such a case occurs in section 5.2, where only one of these constraints can be satisfied.

B.9 Dependence (2). Local broadcast (case 1b.)

The local broadcast condition (theorem 5) states that there exists an integer matrix $M' = \begin{pmatrix}m'_1\\m'_2\end{pmatrix}$ such that:

$$\begin{aligned}
& \sigma_Y \cdot V + \sigma_Y \cdot U \cdot M = \sigma_X \cdot E\\
\Rightarrow \quad & \begin{pmatrix}\sigma_{Y1} & \sigma_{Y2} & \sigma_{Y3}\end{pmatrix} \cdot \begin{pmatrix}0\\0\\1\end{pmatrix} + \begin{pmatrix}\sigma_{Y1} & \sigma_{Y2} & \sigma_{Y3}\end{pmatrix} \cdot \begin{pmatrix}1 & 0\\0 & 1\\0 & 0\end{pmatrix} \cdot M'\\
& = \begin{pmatrix}\sigma_{X1} & \sigma_{X2} & \sigma_{X3}\end{pmatrix} \cdot \begin{pmatrix}0\\0\\1\end{pmatrix}\\
\Rightarrow \quad & \begin{pmatrix}\sigma_{Y3}\end{pmatrix} + \begin{pmatrix}\sigma_{Y1} & \sigma_{Y2}\end{pmatrix} \cdot \begin{pmatrix}m'_1\\m'_2\end{pmatrix} = \begin{pmatrix}\sigma_{X3}\end{pmatrix}\\
\Rightarrow \quad & \sigma_{Y3} + m'_1\sigma_{Y1} + m'_2\sigma_{Y2} = \sigma_{X3}
\end{aligned}$$

B.10 Dependence (2). Strong-removal (cases 2. and 3.)

The strong condition (theorem 1) is:

$$\begin{aligned}
& (\sigma_Y - \sigma_X \cdot R) \cdot \mathcal{B} = 0 \qquad \text{with } \mathcal{B} = Id\\
\Rightarrow \quad & \begin{pmatrix}\sigma_{Y1} & \sigma_{Y2} & \sigma_{Y3}\end{pmatrix} - \begin{pmatrix}\sigma_{X1} & \sigma_{X2} & \sigma_{X3}\end{pmatrix} \cdot \begin{pmatrix}0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 1\end{pmatrix} = 0\\
\Rightarrow \quad & \begin{cases}\sigma_{Y1} = 0\\ \sigma_{Y2} = 0\\ \sigma_{Y3} - \sigma_{X3} = 0\end{cases}
\end{aligned}$$

Since allocation matrix σ_Y must be of full row rank, these constraints can not be satisfied if the processor space is not 1-dimensional, *i.e.* if $d_\sigma \geq 2$. This occurs in section 5.1 where the strong condition does not apply.
The computation of c (realized using one of the vertices of $\mathcal{D}_2$) results in $c = 0$. Therefore the condition that eliminates all communications (corollary 1) simplifies in $\rho(p) = 0$:

$$\begin{aligned} \rho(p) = 0 \quad &\Leftrightarrow \quad -\sigma_X \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} - \gamma_X + \gamma_Y = 0 \\ &\Leftrightarrow \quad \sigma_{X3} - \gamma_X + \gamma_Y = 0 \end{aligned}$$

B.11 Dependence (2). Local propagation relatively to valid vertex v_1' (case 4.)

The local propagation condition on the allocation functions for vertex v_1 is (theorem 2):

$$\begin{aligned} & (\sigma_Y \cdot A_1' - \sigma_X) \cdot E = 0 \\ \Leftrightarrow \quad & (\sigma_Y \cdot \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \sigma_X) \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 0 \\ \Leftrightarrow \quad & \sigma_{Y3} - \sigma_{X3} = 0 \end{aligned}$$