On the Optimality of Feautrier's Scheduling Algorithm

Frédéric Vivien

ICPS-LSIIT, Université Louis Pasteur, Strasbourg, Pôle Api, F-67400 Illkirch, France.

Abstract. Feautrier's scheduling algorithm is the most powerful existing algorithm for parallelism detection and extraction. But it has always been known to be suboptimal. However, the question whether it may miss some parallelism because of its design was still open. We show that this is not the case. Therefore, to find more parallelism than this algorithm does, one needs to get rid of some of the hypotheses underlying its framework.

1 Introduction

One of the fundamental steps of automatic parallelization is the detection and extraction of parallelism. This extraction can be done in very different ways, from the try and test of *ad hoc* techniques to the use of powerful scheduling algorithms. In the field of dense matrix code parallelization, lots of algorithms have been proposed along the years. Among the main ones, we have the algorithms proposed by Lamport [10], Allen and Kennedy [2], Wolf and Lam [14], Feautrier [7, 8], and Darte and Vivien [5]. This collection of algorithm spans a large domain of techniques (loop distribution, unimodular transformations, linear programming, etc.) and a large domain of dependence representations (dependence levels, direction vectors, affine dependences, dependence polyhedra). One may wonder which algorithm to chose from such a collection. Fortunately, we have some theoretical comparative results on these algorithms, as well as some optimality results.

Allen and Kennedy's, Wolf and Lam's, and Darte and Vivien's algorithms are optimal for the representation of the dependences they respectively take as input [4]. This means that each of these algorithms extracts all the parallelism contained in its input (some representation of the code dependences). Wolf and Lam's algorithm is a generalization of Lamport's; Darte and Vivien's algorithm is a generalization of those of Allen and Kennedy, and of Wolf and Lam, and is generalized by Feautrier's [4]. Finally, Feautrier's algorithm can handle any of the dependence representations used by the other algorithms [4].

It appears from these results that Feautrier's algorithm is the most powerful algorithm we have at hand. Although this algorithm has always be known to be suboptimal, its exact efficiency was so far unknown. Hence the questions we address in this paper: What are its weaknesses? Is its suboptimality only due to its framework or also to its design? What can be done to improve this algorithm? How can we build a more powerful algorithm?

In Section 2 we briefly recall Feautrier's algorithm. Then we discuss its weaknesses in Section 3. In Section 4 we present what seems to be a "better" algorithm. Section 5 presents the major new result of this paper: to find "more" parallelism than Feautrier's algorithm one needs to use far more powerful techniques.

2 The Algorithm

Feautrier uses schedules to detect and extract parallelism. This section gives an overview of his algorithm. The missing details can be found either in [7, 8] or [4].

Framework: Static Control Programs. To enable an exact dependence analysis, the control-flow must be predictable at compile time. The necessary restrictions define the class of the *static control programs.* These are the programs:

- whose only data structures are integers, floats, arrays of integers, and arrays of floats, with no pointers or pointer-like mechanisms;
- whose elementary statements are assignments of scalars or array elements;
- whose only control structure are sequences and do loops with constant steps;
- where the array subscripts and the loop bounds are affine functions of surrounding loop indices and structural parameters.

Static control programs are mainly sets of nested loops. Figure 1 presents an example of such a program. Let S be any statement. The iteration domain of S, denoted $\mathcal{D}_S$, is the set of all possible values of the vector of the indices (the *iteration vector*) of the loops surrounding S: in Example 1, $\mathcal{D}_S = \{(i,j) \mid 1 \leq i \leq N, 1 \leq j \leq i\}$. An iteration domain is always a polyhedron. In other words, there always exist a matrix A and a vector b such that: $\mathcal{D}_S = \{x \mid A.x \leq b\}$.

```
DO i=1, N
 DO j=1, i
 S: a(i,i+j+1) = a(i-1,2*i-1) + a(j,2*j)
 ENDDO
ENDDO
```

Fig. 1. Example 1.

e_1: $S(i\text{–}1, i\text{–}1) \rightarrow S(i,j), h_{e_1}(i,j) = (i\text{–}1, i\text{–}1)$
$\mathcal{D}_{e_1} = \{(i,j) \mid 2 \leq i \leq N, 1 \leq j \leq i\}$

e_2: $S(j, j\text{–}1) \rightarrow S(i,j), h_{e_2}(i,j) = (j, j\text{–}1)$
$\mathcal{D}_{e_2} = \{(i,j) \mid 1 \leq i \leq N, 2 \leq j \leq i\}$

Fig. 2. Dependences for Example 1.

Dependence Representation. In the framework of static control programs, an exact dependence analysis is feasible [6] and each exact *dependence relation* e from statement S_e to statement T_e is defined by a polyhedron $\mathcal{D}_e$, the domain of existence of the dependence relation, and a quasi-affine [1] function h_e as follows:

[1] See the original paper [6] for more details.

for any value $j \in \mathcal{D}_e$, operation $T_e(j)$ depends on operation $S_e(h_e(j, N))$:

$$j \in \mathcal{D}_e \quad \Rightarrow \quad S_e(h_e(j, N)) \rightarrow T_e(j)$$

where N is the vector of structural parameters. Obviously, the description of the exact dependences between two statements may involve the union of many such dependence relations. A dependence relation e describes for any value $j \in \mathcal{D}_e$ a dependence between the two operations $S_e(h_e(j, N))$ and $T_e(j)$, what we call an *operation to operation dependence*. In other words, a *dependence relation* is a set of elementary *operation to operation dependences*. Figure 2 presents the dependence relations for Example 1.

Following Feautrier [7], we suppose that all the quasi-affine functions we have to handle are in fact affine functions (at the possible cost of a conservative approximation of the dependences).

Searched Schedules. Feautrier does not look for any type of functions to schedule affine dependences. He only considers nonnegative functions, with rational values, that are affine functions in the iteration vector and in the vector of structural parameters. Therefore he only handles (affine) schedules of the form:

$$\Theta(S, j, N) = X_S.j + Y_S.N + \rho_S \tag{1}$$

where X_S and Y_S are non-parameterized rational vectors and ρ_S is a rational constant. The hypothesis of nonnegativity of the schedules is not restrictive as all schedules must be lower bounded.

Problem Statement. Once chosen the form of the schedules, the scheduling problem *seems* to be simple. For a schedule to be valid, it must (and only has to) satisfy the dependences. For example, if operation $T(j)$ depends on operation $S(i)$, $T(j)$ must be scheduled *after* $S(i)$: $\Theta(T, j, N) > \Theta(S, i, N)$. Therefore, for each statement S, we just have to find a vector X_S, a vector Y_S, and a constant ρ_S such that, for each dependence relation e, the schedule satisfies: [2]

$$j \in \mathcal{D}_e \quad \Rightarrow \quad \Theta(S_e, h_e(j, N), N) + 1 \ \leq \ \Theta(T_e, j, N). \tag{2}$$

The set of constraints is linear, and one can imagine using linear system solvers to find a solution. Actually, there are now two difficulties to overcome:

1. Equation (2) must be satisfied for any possible value of the structural parameters. If polyhedron $\mathcal{D}_e$ is parameterized, Equation (2) may correspond to an infinite set of constraints, which cannot be enumerated. There are two means to overcome this problem: the polyhedron vertices (cf. Section 4) and the affine form of Farkas' lemma (see below). Feautrier uses the latter.
2. There does not always exist a solution for such a set of constraints. We will see how the use of multidimensional schedules can overcome this problem.

[2] The transformation of the inequality, from $a > b$ to $a \geq 1+b$, is obvious for schedules with integral values and classical for schedules with rational values [12].

The Affine Form of Farkas' Lemma and Its Use. This lemma [7, 13] predicts the shape of certain affine forms.

Theorem 1 (Affine Form of Farkas' Lemma). *Let $\mathcal{D}$ be a nonempty polyhedron defined by p inequalities: $a_k x + b_k \geq 0$, for any $k \in [1, p]$. An affine form Φ is nonnegative over $\mathcal{D}$ if and only if it is a nonnegative affine combination of the affine forms used to define $\mathcal{D}$:*

$$\Phi(x) \equiv \lambda_0 + \sum_{k=1}^{p} \lambda_k (a_k x + b_k), \ \textit{ with } \lambda_k \geq 0 \textit{ for any } k \in [0, p].$$

This theorem is useful as, in static control programs, all the important sets are polyhedra: iteration domains, dependence existence domains [6], etc. Feautrier uses it to predict the shape of the schedules and to simplify the set of constraints.

Schedules. By hypothesis, the schedule $\Theta(S, j, N)$ is a nonnegative affine form defined on a polyhedron $\mathcal{D}_S$: the iteration domain of statement S. Therefore, the affine form of Farkas' lemma states that $\Theta(S, j, N)$ is a nonnegative affine combination of the affine forms used to define $\mathcal{D}_S$. Let $\mathcal{D}_S = \{x \mid \forall i \in [1, p_S],\ A_{S,i}.x + B_{S,i}.N + c_{S,i} \geq 0\}$ ($\mathcal{D}_S$ is thus defined by p_S inequalities). Then Theorem 1 states that there exist some nonnegative values $\mu_{S,0}$, ..., μ_{S,p_S} such that:

$$\Theta(S, j, N) \equiv \mu_{S,0} + \sum_{i=1}^{p_S} \mu_{S,i} (A_{S,i}.j + B_{S,i}.N + c_{S,i}). \tag{3}$$

Dependence Constraints. Equation (2) can be rewritten as an affine function that is nonnegative over a polyhedron because the schedules and the function h_e are affine functions:

$$j \in \mathcal{D}_e \quad \Rightarrow \quad \Theta(T_e, j, N) - \Theta(S_e, h_e(j, N), N) - 1 \geq 0.$$

Once again we can apply the affine form of Farkas' lemma. Let $\mathcal{D}_e = \{x \mid \forall i \in [1, p_e],\ A_{e,i}.x + B_{e,i}.N + c_{e,i} \geq 0\}$ ($\mathcal{D}_e$ is thus defined by p_e inequalities). Theorem 1 states that there exist some nonnegative values $\lambda_{e,0}, ..., \lambda_{e,p_e}$ such that:

$$\Theta(T_e, j, N) - \Theta(S_e, h_e(j, N), N) - 1 \equiv \lambda_{e,0} + \sum_{i=1}^{p_e} \lambda_{e,i} (A_{e,i}.j + B_{e,i}.N + c_{e,i}).$$

Using Equation (3), we rewrite the left-hand side of this equation:

$$\begin{aligned} &\left(\mu_{T_e,0} + \sum_{i=1}^{p_{T_e}} \mu_{T_e,i} (A_{T_e,i}.j + B_{T_e,i}.N + c_{T_e,i}) \right) \\ &\qquad - \left(\mu_{S_e,0} + \sum_{i=1}^{p_{S_e}} \mu_{S_e,i} (A_{S_e,i}.h_e(j, N) + B_{S_e,i}.N + c_{S_e,i}) \right) - 1 \\ &\qquad\qquad \equiv \lambda_{e,0} + \sum_{i=1}^{p_e} \lambda_{e,i} (A_{e,i}.j + B_{e,i}.N + c_{e,i}). \end{aligned} \tag{4}$$

Equation 4 is a formal equality ($\equiv$). Thus, the coefficients of a given component of either of the vectors j and N must be the same on both sides. The constant terms on both sides of this equation must also be equal. This identification process leads to a set of $(n+q+1)$ equations, equivalent to Equation (4), where n is the size of the iteration vector j, and q the size of the parameter vector N.

The way Feautrier uses the affine form of Farkas' lemma enables him to obtain a finite set of linear equations and inequations, equivalent to the original scheduling problem, and that can be solved using any solver of linear systems.

Extension to Multidimensional Scheduling. There exist some static control programs that cannot be scheduled with (monodimensional) affine schedules (e.g. Example 1, cf. Section 4). Hence the need for multidimensional schedules, i.e. schedules whose values are not rationals but rational *vectors* (ordered by lexicographic ordering). The solution proposed by Feautrier is simple and greedy. For the first dimension of the schedules one looks for affine functions that 1) respect all the dependences; 2) satisfy as many dependence relations as possible. The algorithm is then recursively called on the unsatisfied dependence relations. This, plus a strongly connected component distribution[3] that reminds us of Allen and Kennedy's algorithm, defines the algorithm below. G denotes the multigraph defined by the statements and the dependence relations. The multidimensional schedules built satisfy the dependences according to the lexicographic order [4].

FEAUTRIER(G)

1. Compute the strongly connected components of G.
2. **For each** strongly connected component G_i of G **do** in topological order:
   (a) Find, using the method exposed above, an affine function that satisfies
   $$\forall e,\ j \in \mathcal{D}_e \ \Rightarrow\ \Theta(S_e, h_e(j,N), N) + z_e \leq \Theta(T_e, j, N) \text{ with } 0 \leq z_e \leq 1 \quad (5)$$
   and which maximizes the sum $\sum_e z_e$.
   (b) Build the subgraph G'_i generated by the unsatisfied dependences. If G'_i is not empty, recursively call FEAUTRIER(G'_i).

3 The Algorithm's Weaknesses

Definitions of Optimality. Depending on the definition one uses, an algorithm extracting parallelism is optimal if it finds all the parallelism: 1) that can be extracted in its framework (only certain program transformations are allowed, etc.); 2) that is contained in the representation of the dependences it handles; 3) that is contained in the program to be parallelized (not taking into account the dependence representation used nor the transformations allowed). For example, Allen, Callahan, and Kennedy uses the first definition [1], Darte and Vivien the second [5], and Feautrier the third [8]. We now recall that FEAUTRIER is not optimal under any of the last two definitions.

[3] This distribution is rather esthetic as the exact same result can be achieved without using it. This distribution is intuitive and ease the computations.

The Classical Counter-Example to Optimality. Feautrier proved in his original article [7] that his algorithm was not optimal for parallelism detection in static control programs. In his counterexample (Example 2, Figure 3) the source of any dependence is in the first half of the iteration domain and the sink in the second half. Cutting the iteration domain "in the middle" enables a trivial parallelization (Figure 4). The only loop in Example 2 contains some dependences. Thus, Feautrier's schedules must be of dimension at least one (hence at least one sequential loop after parallelization), and FEAUTRIER finds no parallelism.

```
DO i=0, 2n
   x(i) = x(2n-i)
ENDDO
```

Fig. 3. Example 2.

```
DOPAR i=0, n
   x(i) = x(2n-i)
ENDDOPAR
DOPAR i=n+1, 2n
   x(i) = x(2n-i)
ENDDOPAR
```

Fig. 4. Parallelized version of Example 2.

Weaknesses. The weaknesses in Feautrier's algorithm are either a consequence of the algorithm framework, or of the algorithm design.

Framework. Given a program, we extract its implicit parallelism and then we rewrite it. The new order of the computations must be rather regular to enable the code generation. Hence the restriction on the schedule shape: affine functions. The parallel version of Example 2 presented Figure 4 can be expressed by a non affine schedule, but not by an affine schedule. The restriction on the schedule shape is thus a cause of inefficiency. Another problem with Example 2 is that Feautrier looks for a transformation conservative in the number of loops. Breaking a loop into several loops, i.e., cutting the iteration domain into several subdomains, can enable to find more parallelism (even with affine schedules). The limitation here comes from the hypothesis that all instances of a statement are scheduled the same way, i.e., with the same affine function. (Note that this hypothesis is almost always made [10, 2, 14, 5], [9] being the exception.)

Some of the weaknesses of FEAUTRIER are thus due to its framework. Before thinking of changing this framework, we must check whether one can design a more powerful algorithm, or even improve FEAUTRIER, in Feautrier's framework.

Algorithm design. FEAUTRIER is a greedy algorithm which builds multidimensional schedules whose first dimension satisfies as many *dependence relations* as possible, and not as many *operation to operation dependences* as possible. We may wonder with Darte [3, p. 80] whether this can be the cause of a loss of parallelism. We illustrate this possible problem with Example 1.

The first dimension of the schedule must satisfy Equation (5) for both dependence relations e_1 and e_2. This gives us respectively Equations (6) and (7):

$$X_S\begin{vmatrix} i{-}1 \\ i{-}1 \end{vmatrix} + z_{e_1} \leq X_S \begin{vmatrix} i \\ j \end{vmatrix} \Leftrightarrow z_{e_1} \leq X_S \begin{vmatrix} 1 \\ j{-}i{+}1 \end{vmatrix} \Leftrightarrow z_{e_1} \leq \alpha + \beta(j-i+1) \text{ with } \begin{matrix} 2 \leq i \leq N \\ 1 \leq j \leq i \end{matrix} \quad (6)$$

$$X_S\begin{vmatrix} j \\ j{-}1 \end{vmatrix} + z_{e_2} \leq X_S \begin{vmatrix} i \\ j \end{vmatrix} \Leftrightarrow z_{e_2} \leq X_S \begin{vmatrix} i{-}j \\ 1 \end{vmatrix} \Leftrightarrow z_{e_2} \leq \alpha(i-j) + \beta \text{ with } \begin{matrix} 1 \leq i \leq N \\ 2 \leq j \leq i \end{matrix} \quad (7)$$

if we note $X_S = (\alpha, \beta)$ [4]. Equation (6) with $i = N$ and $j = 1$ is equivalent to $z_{e_1} \leq \alpha + \beta(2 - N)$. The schedule must be valid for any (nonnegative) value of the structural parameter N, this implies $\beta \leq 0$. Equation (7) with $i = j$ is equivalent to $z_{e_2} \leq \beta$. Hence $z_{e_2} \leq 0$. As z_{e_2} must be nonnegative $z_{e_2} = 0$ (cf. Equation (5)). This means that the first dimension of any affine schedule cannot satisfy the dependence relation e_2.

The dependence relation e_1 can be satisfied, a solution being $X_S = (1, 0)$ ($\alpha = 1$, $\beta = 0$). Therefore, FEAUTRIER **is called recursively on the whole dependence relation** e_2. However, most of the dependences described by e_2 are satisfied by the schedule $\Theta(S, (i, j), N) = i$ (defined by $X_S = (1, 0)$). Indeed, Equation (6) is then satisfied for any value $(i, j) \in \mathcal{D}_{e_2}$ except when $i{=}j$. Thus, **one only needed to call recursively** FEAUTRIER **on the dependence relation** e_2': $S(j, j{-}1) \rightarrow S(i, j)$, $h_{e_2}(i, j) = (j, j{-}1)$, $\mathcal{D}_{e_2'} = \{(i, j) \mid 2{\leq}i{\leq}N, i = j\}$. The search for the schedules in FEAUTRIER is thus overconstrained by design.

We may now wonder whether this overconstraining may lead FEAUTRIER to build some affine schedules of non minimal dimensions and thus to miss some parallelism. We first present an algorithm which gets rid of this potential problem. Later we will show that no parallelism is lost because of this design particularity.

4 A Greedier Algorithm

The Vertex Method. A polyhedron can always be decomposed as the sum of a polytope (i.e. a bounded polyhedron) and a polyhedral cone, called the characteristic cone (see [13] for details). A polytope is defined by its vertices, and any point of the polytope is a nonnegative convex combination of the polytope vertices. A polyhedral cone is finitely generated and is defined by its rays and lines. Any point of a polyhedral cone is the sum of a nonnegative combination of its rays and any combination of its lines. Therefore, a polyhedron $\mathcal{D}$ can be equivalently defined by a set of *vertices*, $\{v_1, \ldots, v_\omega\}$, a set of *rays*, $\{r_1, \ldots, r_\rho\}$, and a set of *lines*, $\{l_1, \ldots, l_\lambda\}$. Then $\mathcal{D}$ is the set of all vectors p such that

$$p = \sum_{i=1}^{\omega} \mu_i v_i + \sum_{i=1}^{\rho} \nu_i r_i + \sum_{i=1}^{\lambda} \xi_i l_i \quad (8)$$

[4] Example 1 contains a single statement S. Therefore, the components Y_S and ρ_S of Θ (cf. Equation (1)) have no influence here on Equation (5) which is equivalent to: $(X_S.h_e(j, N) + Y_S.N + \rho_S) + z_e \leq (X_S.j + Y_S.N + \rho_S) \Leftrightarrow X_S.h_e(j, N) + z_e \leq X_S.j$.

with $\mu_i \in \mathbb{Q}^+$, $\nu_i \in \mathbb{Q}^+$, $\xi_i \in \mathbb{Q}$, and $\sum_{i=1}^{\omega} \mu_i = 1$. As we have already stated, all the important sets in static control programs are polyhedra, and any nonempty polyhedron is fully defined by its vertices, rays, and lines, which can be computed even for parameterized polyhedra [11]. The vertex method [12] explains how we can use the vertices, rays, and lines to simplify set of constraints.

Theorem 2 (The Vertex Method). *Let $\mathcal{D}$ be a nonempty polyhedron defined by a set of vertices, $\{v_1, \ldots, v_\omega\}$, a set of rays, $\{r_1, \ldots, r_\rho\}$, and a set of lines, $\{l_1, \ldots, l_\lambda\}$). Let Φ be an affine form of linear part A and constant part b ($\Phi(x) = A.x + b$). Then the affine form Φ is nonnegative over $\mathcal{D}$ if and only if 1) Φ is nonnegative on each of the vertices of $\mathcal{D}$ and 2) the linear part of Φ is nonnegative (respectively null) on the rays (resp. lines) of $\mathcal{D}$. This can be written:*

$$\forall p \in \mathcal{D},\ A.p + b \geq 0 \quad \Leftrightarrow$$
$$\forall i \in [1, \omega],\ A.v_i + b \geq 0,\ \forall i \in [1, \rho],\ A.r_i \geq 0,\ \textit{and}\ \ \forall i \in [1, \lambda],\ A.l_i = 0.$$

The polyhedra produced by the dependence analysis of programs are in fact polytopes. Then, according to Theorem 2, an affine form is nonnegative on a polytope if and only if it is nonnegative on the vertices of this polytope. We use this property to simplify Equation (2) and define a new scheduling algorithm.

The Greediest Algorithm. Feautrier's algorithm is a greedy heuristic which maximizes the number of *dependence relations* satisfied by the first dimension of the schedule. The algorithm below is a greedy heuristic which maximizes the number of *operation to operation dependences* satisfied by the first dimension of the schedule, and then proceeds recursively. To achieve this goal, this algorithm greedily considers the vertices of the existence domain of the dependence relations. Let e_1, ..., e_n be the dependence relations in the studied program. For any $i \in [1, n]$, let $v_{i,1}, ..., v_{i,m_i}$ be the vertices of $\mathcal{D}_{e_i}$, and let, for any $j \in [1, m_i]$, $e_{i,j}$ be the operation to operation dependence from $S_{e_i}(h_{e_i}(v_{i,j}), N)$ to $T_{e_i}(v_{i,j})$. G denotes here the multigraph generated by the dependences $e_{i,j}$.

GREEDY(G)

1. Compute the strongly connected components of G.
2. **For each** strongly connected component G_k of G **do** in topological order:
   (a) Find an *integral* affine function Θ that satisfies
   $$\forall e_{i,j},\ \ \Theta(S_{e_i}, h_{e_i}(v_{i,j}, N), N) + z_{i,j} \ \leq\ \Theta(T_{e_i}, v_{i,j}, N)\ \text{ with }\ 0 \leq z_{i,j} \leq 1 \tag{9}$$
   and which maximizes the sum $\sum_{e_{i,j}} z_{i,j}$.
   (b) Build the subgraph G'_k generated by the unsatisfied dependences. If G'_k is not empty, recursively call GREEDY(G'_k).

Lemma 1 (Correctness and Maximum Greediness). *The output of algorithm* GREEDY *is a schedule and the first dimension of this schedule satisfies all the operation to operation dependences that can be satisfied by the first dimension of an affine schedule (of the form defined in Section 2).*

5 Schedules of Minimal Dimension

As GREEDY is greedier than FEAUTRIER, one could imagine that the former may sometimes build schedules of smaller dimension than the latter and thus may find more parallelism. The following theorem shows that this never happens.

Theorem 3 (The Dimension of Feautrier's Schedules is Minimal). *Let us consider a loop nest whose dependences are all affine, or are represented by affine functions. If we are only looking for one affine schedule per statement of the loop nest, then the dimension of the schedules built by* FEAUTRIER *is minimal, for each statement of the loop nest.*

Note that this theorem cannot be improved, as the study of Example 2 shows. The proof is direct (not using algorithm GREEDY), see Section A.3.

Principle of the proof. Let σ be an affine schedule whose dimension is minimal for each statement in the studied loop nest. Let e be a dependence relation, of existence domain $\mathcal{D}_e$. We suppose that e is not fully, but partially, satisfied by the first dimension of σ (otherwise there is no problem with e). The operation to operation dependences in e not satisfied by the first dimension of the schedule σ define a subpolyhedron $\mathcal{D}_e^1$ of $\mathcal{D}_e$: this is the subset of $\mathcal{D}_e$ on which the first dimension of σ induces a null delay. $\mathcal{D}_e^1$ is thus defined by the equations defining $\mathcal{D}_e$ and by the null delay equation involving the first dimension of σ ($\sigma_1(T_e, j, N) - \sigma_1(S_e, h_e(j, N), N) = 0$). The second dimension of σ must respect the dependences in $\mathcal{D}_e^1$, i.e., must induce a nonnegative delay over $\mathcal{D}_e^1$. Therefore, the second dimension of σ is an affine form nonnegative over a polyhedron. Using the affine form of Farkas' lemma, we obtain that the second dimension of σ is defined from the (null delay equation on the) first dimension of σ and from the equations defining $\mathcal{D}_e$. From the equations obtained using Farkas' lemma, we build a nonnegative linear combination of the first two dimensions of σ which induces a nonnegative delay over $\mathcal{D}_e$ (and not only on $\mathcal{D}_e^1$), and which satisfies all the operation to operation dependences in e satisfied by any of the first two dimensions of σ. This way we build a schedule *a la* Feautrier of same dimension than σ: a whole dependence relation is kept as long as all its operation to operation dependences are not satisfied by the same dimension of the schedule.

Consequences. First, a simple and important corollary of the previous theorem:

Corollary 1. FEAUTRIER *is well-defined: it always outputs a valid schedule when its input is the exact dependences of an existing program.*

The original proof relied on an assumption on the dependence relations that can be easily enforced but which is not always satisfied: all operation to operation dependences in a dependence relation are of the same dependence level. For example, dependence relation e_2 in Example 1 does not satisfy this property.

More important, Theorem 3 shows that Feautrier's algorithm can only miss some (significant amount of) parallelism because of the limitations of its framework, but not because of its design: as the dimension of the schedule is minimal, the magnitude of the schedule's makespan is minimal, for any statement.

6 Conclusion

Feautrier's scheduling algorithm is the most powerful existing algorithm for parallelism detection and extraction. But it has always been known to be suboptimal. We have shown that Feautrier's algorithm do not miss any significant amount of parallelism *because* of its design, even if one can design a greedier algorithm. Therefore, to improve Feautrier's algorithm or to build a more powerful algorithm, one must get rid of some of the restrictive hypotheses underlying its framework: affine schedules — but more general schedules will cause great problems for code generation — and one scheduling function by statement — Feautrier, Griebl, and Lengauer have already begun to get rid of this hypothesis by splitting the iteration domains [9].

What Feautrier historically introduced as a "greedy heuristic" is nothing but the most powerful algorithm in its class!

References

1. J. Allen, D. Callahan, and K. Kennedy. Automatic decomposition of scientific programs for parallel execution. In *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Programming Languages*, pages 63–76, Munich, Germany, Jan. 1987.
2. J. R. Allen and K. Kennedy. PFC: A program to convert Fortran to parallel form. Technical Report MASC-TR82-6, Rice University, Houston, TX, USA, 1982.
3. A. Darte. De l'organisation des calculs dans les codes répétitifs. *Habilitation thesis*, École normale supérieure de Lyon, 1999.
4. A. Darte, Y. Robert, and F. Vivien. *Scheduling and Automatic Parallelization.* Birkhäuser Boston, 2000. ISBN 0-8176-4149-1.
5. A. Darte and F. Vivien. Optimal Fine and Medium Grain Parallelism Detection in Polyhedral Reduced Dependence Graphs. *Int. J. of Parallel Programming*, 1997.
6. P. Feautrier. Dataflow analysis of array and scalar references. *International Journal of Parallel Programming*, 20(1):23–51, 1991.
7. P. Feautrier. Some efficient solutions to the affine scheduling problem, part I: One-dimensional time. *Int. J. Parallel Programming*, 21(5):313–348, Oct. 1992.
8. P. Feautrier. Some efficient solutions to the affine scheduling problem, part II: Multi-dimensional time. *Int. J. Parallel Programming*, 21(6):389–420, Dec. 1992.
9. M. Griebl, P. Feautrier, and C. Lengauer. Index set splitting. *International Journal of Parallel Programming*, 28(6):607–631, 2000.
10. L. Lamport. The parallel execution of DO loops. *Communications of the ACM*, 17(2):83–93, Feb. 1974.
11. V. Loechner and D. K. Wilde. Parameterized polyhedra and their vertices. *International Journal of Parallel Programming*, 25(6), Dec. 1997.
12. P. Quinton. *Automata Networks in Computer Science*, chapter The systematic design of systolic arrays. Manchester University Press, 1987.
13. A. Schrijver. *Theory of Linear and Integer Programming.* John Wiley & Sons, New York, 1986.
14. M. E. Wolf and M. S. Lam. A data locality optimizing algorithm. In *SIGPLAN Conference PLDI*, pages 30–44. ACM Press, 1991.

A Proofs

A.1 Correctness and Maximum Greedyness of Algorithm Greedy

Definition 1 (Delay of an affine function). *Let $S(i)$ and $T(j)$ be two operations of the loop nest such that $T(j)$ depends on $S(i)$: $S(i) \to T(j)$. Let $\tau(U, i, N)$ be any affine function (whose first parameter is a statement, the function being affine in its last two parameters). We call delay induced by τ on the dependence $S(i) \to T(j)$ the difference of the "execution times" of $T(j)$ and $S(i)$ defined by τ:*

$$\tau(T, j, N) - \tau(S, i, N).$$

Proof of lemma 1

Let $S(i)$ and $T(j)$ be two operations of the loop nest such that $T(j)$ depends on $S(i)$: $S(i) \to T(j)$. In the graph G there exists a dependence relation e which includes the operation to operation dependence $S(i) \to T(j)$. Let v_1, ..., v_p be the vertices of $\mathcal{D}_e$. By definition of these vertices, there exist some nonnegative rationals μ_1, ..., μ_p such that: $j = \sum_{k=1}^{p} \mu_k v_k$ and $\sum_{k=1}^{p} \mu_k = 1$.

1. Before all, we must show that this algorithm effectively builds a schedule. Equation (9) insures us that, for any $k \in [1, p]$:

$$\Theta(S, h_e(v_k, N), N) \leq \Theta(T, v_k, N).$$

Therefore:

$$\begin{aligned}
\sum_{k=1}^{p} \mu_k \Theta\left(S, h_e\left(v_k, N\right), N\right) \leq \sum_{k=1}^{p} \mu_k \Theta\left(T, v_k, N\right) \quad &\Leftrightarrow \\
\Theta\left(S, \sum_{k=1}^{p} \mu_k h_e\left(v_k, N\right), N\right) \leq \Theta\left(T, \sum_{k=1}^{p} \mu_k v_k, N\right) \quad &\Leftrightarrow \\
\Theta\left(S, h_e\left(\sum_{k=1}^{p} \mu_k v_k, N\right), N\right) \leq \Theta\left(T, \sum_{k=1}^{p} \mu_k v_k, N\right) \quad &\Leftrightarrow \\
\Theta\left(S, h_e\left(j, N\right), N\right) \leq \Theta\left(T, j, N\right) \quad &\Leftrightarrow \\
\Theta\left(S, i, N\right) \leq \Theta\left(T, j, N\right) &
\end{aligned}$$

The equivalence between the first and the second inequations (resp. between the second and the third) comes from the fact that Θ (resp. h_e) is an affine function.

From what precedes, $\Theta(S, i, N) \leq \Theta(T, j, N)$, and Θ (the first dimension of the schedule) respects the dependence between $S(i)$ and $T(j)$. Therefore, the schedule built by Greedy respects all the dependences.

We still have to show that Greedy *satisfies* any operation to operation dependence. This is a consequence of the next point.

2. We suppose that the operation to operation dependence $S(i) \rightarrow T(j)$ can be satisfied by the first dimension of an affine schedule. Let σ be the first dimension of any such schedule: $\sigma(S,i,N)+1 \leq \sigma(T,j,N)$. We suppose that σ is an integral function. If this is not the case, we just scale up σ by the least common multiple of the denominator of its rational coefficients.

$$\begin{aligned}
\sigma(S,i,N)+1 \leq \sigma(T,j,N) \quad & \Leftrightarrow \\
\sigma(S,h_e(j),N)+1 \leq \sigma(T,j,N) \quad & \Leftrightarrow \\
\sigma\left(S, h_e\left(\sum_{k=1}^{p} \mu_k v_k\right), N\right) + 1 \leq \sigma\left(T, \sum_{k=1}^{p} \mu_k v_k, N\right) \quad & \Leftrightarrow \\
\sigma\left(S, \sum_{k=1}^{p} \mu_k h_e(v_k), N\right) + 1 \leq \sigma\left(T, \sum_{k=1}^{p} \mu_k v_k, N\right) \quad & \Leftrightarrow \\
1 + \sum_{k=1}^{p} \mu_k \sigma(S, h_e(v_k), N) \leq \sum_{k=1}^{p} \mu_k \sigma(T, v_k, N) &
\end{aligned}$$

As σ is the first dimension of a schedule, it respects any dependence and in particular those corresponding to the vertices of $\mathcal{D}_e$: for any $k \in [1,p]$, $\sigma(S,h_e(v_k),N) \leq \sigma(T,v_k,N)$. Then, from the inequation:

$$1 + \sum_{k=1}^{p} \mu_k \sigma(S, h_e(v_k), N) \leq \sum_{k=1}^{p} \mu_k \sigma(T, v_k, N)$$

we conclude that there exists a vertex v_q such that $\sigma(S,h_e(v_q),N) < \sigma(T,v_q,N)$ with $\mu_q > 0$'. Thus there exists a vertex v_q such that $1 + \sigma(S,h_e(v_q),N) \leq \sigma(T,v_q,N)$ with $\mu_q > 0$ (as σ is integral). Therefore, the first dimension, Θ, of the schedule built by GREEDY satisfies the dependence corresponding to v_q. Otherwise $(\Theta + \sigma)$ would be an affine function inducing:

(a) a nonnegative delay on any vertex (as the sum of two such affine functions);

(b) a delay greater than or equal to one on all the vertices on which either Θ or σ induces a delay greater than or equal to one; therefore $(\Theta+\sigma)$ would induce a delay greater than or equal to one on v_q and therefore on at least one more vertex than Θ, which is impossible by definition of Θ.

As Θ induces a delay greater than or equal to one on v_q, it induces a strictly positive delay on $S(i) \rightarrow T(j)$. Indeed, from the inequations:

$$1 + \sigma(S,h_e(v_q),N) \leq \sigma(T,v_q,N)$$

and

$$\forall k \in [1,p],\ \sigma(S,h_e(v_k),N) \leq \sigma(T,v_k,N)$$

we can obtain (using the same transformations than previously):

$$\Theta(S,i,N) + \mu_q \leq \Theta(T,j,N) \quad \Rightarrow \quad \Theta(S,i,N) < \Theta(T,j,N).$$

As Θ is by construction an integral schedule, the strictly positive delay $\Theta(T, j, N) - \Theta(S, i, N)$ is greater than or equal to one, and Θ satisfies the operation to operation dependence $S(i) \rightarrow T(j)$.

The first point shows that the multidimensional affine functions built by GREEDY respect all the dependences. The second point show the maximum greediness: the first dimension of Θ satisfies all the operation to operation dependences that can be satisfied by an affine schedule. Thus, if there exists one multidimensional affine schedule for the studied set of dependences, GREEDY builds such a schedule. ■

A.2 Schedules of Minimal Dimension for all Statements

The existence of such schedules is needed by Theorem 3.

Lemma 2. *If there exists at least one valid affine schedule for the studied system, then there exists an affine schedule which is simultaneously of minimal dimension for all statements.*

Proof of lemma 2 Let Θ_1 and Θ_2 be two valid schedules. For each statement S, we define d_S as the minimum of the dimensions of $\Theta_1(S)$ and $\Theta_2(S)$. Then, we define a new (multidimensional) affine function Θ'_1 as follows: for each statement S, $\Theta'_1(S) = \Theta_1(S)_{|d_S}$, i.e., $\Theta'_1(S)$ is equal to the first d_S dimensions of $\Theta_1(S)$. As Θ_1 is a schedule, and as $\Theta_1(S)$ is of dimension greater than or equal to d_S, Θ'_1 induces a lexicographically nonnegative delay on all the dependences. We symmetrically define Θ'_2. Then $\Theta = \Theta'_1 + \Theta'_2$ is a schedule. Indeed, let us consider an operation to operation dependence between $S(i)$ and $T(j)$. Without any loss of generality, we suppose that $d_S \leq d_T$, and that d_S is the dimension of $\Theta_1(S)$. As Θ_1 is a schedule, it induces a lexicographically positive delay on the dependence between $S(i)$ and $T(j)$. Furthermore, as d_S is the dimension of $\Theta_1(S)$ and as $d_S \leq d_T$, Θ'_1 induces a lexicographically positive delay on the dependence between $S(i)$ and $T(j)$. As Θ'_2 induces a lexicographically nonnegative delay on all the dependences, $\Theta'_1 + \Theta'_2$ induces a lexicographically positive delay on the considered dependence.

We can take for each statement S a schedule of minimal dimension for S. If we apply the previous scheme to this set of schedules we end up with a schedule which is simultaneously of minimal dimension for all statements. ■

A.3 Minimality of the Dimension of Feautrier's Schedules

Proof of Theorem 3

Hypothesis

Let Θ be an affine schedule whose dimension is minimal for each statement. The existence of such a schedule is guaranteed by Lemma 2 on the hypothesis that there exist at least one (multidimensional) affine schedule satisfying all the dependences in the studied system. If there is no schedule, Theorem 3 obviously holds! From Θ, we build by induction an affine schedule P *a la* FEAUTRIER whose dimension is minimal for each statement.

We suppose that Θ is an integral function. If this is not the case, we just scale up Θ by the least common multiple of the denominator of its rational coefficients.

Notations
Let e_1, ..., e_p be the p dependence relations of the considered loop nest. Each of these dependence relations is a 4-uplet: $e_j = (S_j, T_j, \mathcal{D}_j, h_j)$, where S_j and T_j are two statements, $\mathcal{D}_j$ is a domain, and h_j an affine function such that e_j denotes the following set of dependences:

$$\forall i \in \mathcal{D}_j, T_j(i) \text{ depends on } S_j(h_j(i, N)). \tag{10}$$

Equation 10 describes, for any dependence e_j and any value i in $\mathcal{D}_j$, an *operation to operation dependence* between $T_j(i)$ and $S_j(h_j(i, N))$. In other words, a *dependence relation* is a set of elementary operation to operation dependences. A function is said to *fully satisfy* a dependence relation e_j if and only if it satisfies all the operation to operation dependences in e_j.

Principle of the proof
From Θ, we are going to build a schedule *a la* FEAUTRIER of same dimension, by combining the dimensions of Θ. Let us consider one dependence relation, say e_1. Suppose that no affine schedule can fully satisfy e_1 with its first dimension. The possible inefficiency comes from the fact that the first dimension of Θ can (perhaps) satisfy some of the operation to operation dependences in e_1. Therefore the dimensions of Θ of rank greater than one will no more have to satisfy these operation to operation dependences. Furthermore, the dimensions of Θ of rank greater than one will not be constrained by these operation to operation dependences. On the opposite, the dimensions of any schedule *a la* FEAUTRIER will still have to satisfy *all* the operation to operation dependences in e_j, even if some of these are satisfied by the first dimensions of the schedule.

Let us consider a single dependence relation, of existence domain $\mathcal{D}$. The operation to operation dependences not satisfied by the first dimension of the schedule Θ define a subpolyhedron $\mathcal{D}^1$ of $\mathcal{D}$. This subpolyhedron is **the** subset of $\mathcal{D}$ on which the first dimension of the schedule induces a null delay. $\mathcal{D}^1$ is thus defined by the equations defining $\mathcal{D}$ and by the null delay equation involving the first dimension of Θ. The second dimension of Θ must induce a nonnegative delay over $\mathcal{D}^1$. Using the affine form of Farkas' lemma, we obtain that the second dimension of Θ is defined from the first dimension and from the equations defining $\mathcal{D}$. Then, we can easily find a combination of the first two dimensions of Θ which induces a nonnegative delay over $\mathcal{D}$ (and not only on $\mathcal{D}^1$), and which satisfies all the (operation to operation) dependences in $\mathcal{D}$ satisfied by any of the first two dimensions of Θ.

The result is proved by induction.

Induction hypotheses for dimension i

1. The first i dimensions of P fully satisfy the dependence relations e_1, ..., e_{m_i}, but none of the dependence relations e_{1+m_i}, ..., e_p. Therefore, the latter must be satisfied by the dimensions of P of rank greater than or equal to $i+1$.
2. Dimension i of P fully satisfies as many of the dependence relations $e_{1+m_{i-1}}$, ..., e_p as possible.
3. The dimensions of Θ of rank greater than or equal to $i+1$ satisfy the dependence relations $e^i_{1+m_i}$, ..., e^i_p where, if $j \in [1+m_i, p]$, $e^i_j = (S_j, T_j, \mathcal{D}^i_j, h_j)$, with
$$\mathcal{D}^i_j = \{x \in \mathcal{D}_j \mid \forall k \in [1,i],\ \Theta_k(T_j, x, N) - \Theta_k(S_j, h_j(x,N), N) = 0\}.$$
The operation to operation dependences described by $e_1, ..., e_p$, but not included in $e^i_{1+m_i}$, ..., e^i_p, are satisfied by the first i dimensions of Θ. (Therefore, the first i dimensions of Θ fully satisfy the dependence relations e_1, ..., e_{m_i}.)
4. For any j in $[1,i]$, Θ_j is a linear combination of $P_1, ..., P_j$.
5. For any j in $[1,i]$, P_j satisfies all the operation to operation dependences which are satisfied by Θ_j.

Initialization of the induction: $i = 1$.
We first prove that the different induction hypotheses hold when $i = 1$.

The first dimension of Θ fully satisfies some of the dependence relations (possibly none). In other words, for some values of $j \in [1,p]$, the first dimension of Θ induces a delay greater than or equal to one on all the operation to operation dependences described by Equation (10). Without any loss of generality, we suppose that the first dimension of Θ fully satisfies the dependence relations e_1, ..., e_{m_1}, and no other dependence relations.

Hypotheses 1 and 2
By definition, the first dimension of any schedule built by FEAUTRIER fully satisfies as many dependence relations as possible. Therefore, any schedule built by FEAUTRIER fully satisfies at least the dependence relations e_1, ..., e_{m_1}. Indeed, suppose this is not the case. Then take a schedule built by FEAUTRIER and not fully satisfying the dependence relations e_1, ..., e_{m_1}. We add to the first dimension of this schedule the first dimension of Θ. This way we obtain an affine schedule whose first dimension fully satisfies at least one more dependence relation than FEAUTRIER's schedules. This is impossible as it contradicts the maximization of the number of dependence relations fully satisfied (maximization of the sum $\sum_e z_e$ in Equation (5)).
We suppose that any schedule built by FEAUTRIER fully satisfies only the dependence relations e_1, ..., e_{m_1}. Is this is not the case, we take any schedule $\mathcal{S}$ built by FEAUTRIER. We then define a new schedule $\mathcal{T}$ equal to Θ except for its first dimension which is the sum of the first dimension of Θ and the first dimension of $\mathcal{S}$. We replace Θ by $\mathcal{T}$ and we restart our current construction from the beginning. This is valid as Θ and $\mathcal{T}$ have the same dimension for each statement.

The dimensions of P of rank strictly greater than one must satisfy the dependence relations e_{1+m_1}, ..., e_p.

Hypotheses 4 and 5

We define the first dimension of P to be equal to the first dimension of Θ: $P_1 = \Theta_1$. Then, Θ_1 is a linear combination of P_1 and P_1 satisfies all the operation to operation dependences which are satisfied by Θ_1.

Hypothesis 3

Let $j \in [1+m_1, p]$. Then e_j is a dependence not fully satisfied by the first dimension of Θ. As Θ is a schedule it induces a lexicographically nonnegative delay on all dependences. Thus its first dimension induces a nonnegative delay on all dependences. Furthermore, as Θ is integral by hypothesis, its first dimension induces on all dependences a delay either null or greater than one. Let $\mathcal{D}_j^1$ denote the subset of $\mathcal{D}_j$ corresponding to operation to operation dependences of e_j not satisfied by the first dimension of Θ. Then:

$$\mathcal{D}_j^1 = \{i \in \mathcal{D}_j \mid \Theta_1(T_j, i, N) - \Theta_1(S_j, h_j(i, N), N) = 0\}.$$

The dimensions of Θ of rank strictly greater than one must satisfy the dependence relations $e^1_{1+m_1}$, ..., e^1_p where, if $j \in [1+m_1, p]$, $e^1_j = (S_j, T_j, \mathcal{D}^1_j, h_j)$.

Induction: from dimension i to dimension $i+1$

We suppose that the induction hypotheses for i are satisfied. Then we build the $(i+1)$-th dimension of P while satisfying the induction hypotheses for $i+1$.

The delay induced by Θ_{i+1} is a nonnegative affine function over $\mathcal{D}^i_j$

By induction hypothesis 3, the dimensions of Θ of rank greater than or equal to $i+1$ must satisfy the dependence relations $e^i_{1+m_i}$, ..., e^i_p. Therefore, Θ_{i+1} induces a nonnegative delay on all the dependence relations $e^i_{1+m_i}$, ..., e^i_p. Let us take any value of j in $[1+m_i, p]$. $\Theta_{i+1}(T_j, x, N) - \Theta_{i+1}(S_j, h_j(x, N), N)$ is an affine function, as $\Theta_{i+1}(T_j)$, $\Theta_{i+1}(S_j)$, and h_j are affine functions by hypothesis. Furthermore this affine function is nonnegative over $\mathcal{D}^i_j$, the domain of the dependence relation e^i_j. $\mathcal{D}^i_j$ is obviously a polyhedron (cf. Equation (11)). Therefore, we can apply the affine form of Farkas' lemma (Theorem 1) to $\Theta_{i+1}(T_j, x, N) - \Theta_{i+1}(S_j, h_j(x, N), N)$ and $\mathcal{D}^i_j$.

Applying the affine form of Farkas' lemma

Let $f_1, ..., f_q$ be the affine functions defining polyhedron $\mathcal{D}_j$: $\mathcal{D}_j = \{x \mid \forall k \in [1, q],\ f_k(x, N) \geq 0\}$. Then, by induction hypothesis 3:

$$\mathcal{D}^i_j = \left\{ x \;\middle|\; \begin{array}{ll} \forall k \in [1, q], & f_k(x, N) \geq 0 \\ \forall k \in [1, i], & \Theta_k(T_j, x, N) - \Theta_k(S_j, h_j(x, N), N) \geq 0 \\ \forall k \in [1, i], & -\Theta_k(T_j, x, N) + \Theta_k(S_j, h_j(x, N), N) \geq 0 \end{array} \right\} \qquad (11)$$

Then the affine form of Farkas' lemma claims the existence of some nonnegative values α, $\beta_1, ..., \beta_q$, $\gamma_1, ..., \gamma_i$, and $\delta_1, ..., \delta_i$, such that:

$$\begin{aligned}\Theta_{i+1}(T_j, x, N) - \Theta_{i+1}(S_j, h_j(x, N), N) = \\ \alpha + \sum_{k=1}^{q} \beta_k f_k(x, N) \\ + \sum_{k=1}^{i} \gamma_k \Big(\Theta_k(T_j, x, N) - \Theta_k(S_j, h_j(x, N), N)\Big) \\ - \sum_{k=1}^{i} \delta_k \Big(\Theta_k(T_j, x, N) - \Theta_k(S_j, h_j(x, N), N)\Big).\end{aligned}$$

By induction hypothesis 4, for any value of l in $[1, i]$, Θ_l is a linear combination of $P_1, ..., P_l$. Thus, there exist some nonnegative values $\lambda_{j,1}, ..., \lambda_{j,i}$, and $\mu_{j,1}, ..., \mu_{j,i}$ such that:

$$\begin{aligned}\Theta_{i+1}(T_j, x, N) - \Theta_{i+1}(S_j, h_j(x, N), N) = \\ \alpha + \sum_{k=1}^{q} \beta_k f_k(x, N) + \sum_{k=1}^{i} (\lambda_{j,k} - \mu_{j,k}) \Big(P_k(T_j, x, N) - P_k(S_j, h_j(x, N), N)\Big)\end{aligned}$$

which is equivalent (after shifting some terms to the left-hand side) to:

$$\begin{aligned}\left(\Theta_{i+1}(T_j) + \sum_{k=1}^{i} \mu_{j,k} P_k(T_j)\right)(x, N) \\ - \left(\Theta_{i+1}(S_j) + \sum_{k=1}^{i} \mu_{j,k} P_k(S_j)\right)(h_j(x, N), N) = \\ \alpha + \sum_{k=1}^{q} \beta_k f_k(x, N) + \sum_{k=1}^{i} \lambda_{j,k} \Big(P_k(T_j, x, N) - P_k(S_j, h_j(x, N), N)\Big). \quad (12)\end{aligned}$$

The right-hand side of Equation (12) is nonnegative over $\mathcal{D}_j$
By induction hypothesis 1, none of the first i dimensions of P fully satisfies a dependence relation e_l if $l \in [1+m_i, p]$. Thus, for any value of k in $[1, i]$, P_k induces a nonnegative delay on the dependence relation e_l, if $l \in [1 + m_i, p]$. In other words, with $l = j$:

$$\forall x \in \mathcal{D}_j, \; P_k(T_j, x, N) - P_k(S_j, h_j(x, N), N) \geq 0. \quad (13)$$

Furthermore, the functions $f_1, ..., f_q$ are by definition nonnegative on $\mathcal{D}_j$, and the constants α, β_k, and $\lambda_{j,k}$ are nonnegative. Therefore, the right-hand side of Equation (12) is a nonnegative affine combination of functions which take nonnegative values on $\mathcal{D}_j$. Then, this right-hand side takes nonnegative values on $\mathcal{D}_j$, and thus on $\mathcal{D}_j^i$ ($\mathcal{D}_j^i \subset \mathcal{D}_j$).

The left-hand side of Equation (12) is a delay formula
This left-hand side is obviously the formula of the delay induced by the affine function $\Theta_{i+1}+\sum_{k=1}^{i}\mu_{j,k}P_k$ on a dependence defined by the affine function h_j, e.g. e_j. As the right-hand side of Equation (12) takes nonnegative values on $\mathcal{D}_j$, $\Theta_{i+1}+\sum_{k=1}^{i}\mu_{j,k}P_k$ is an affine function which induces a nonnegative delay on e_j. Furthermore, this function induces a strictly positive delay on any points of $\mathcal{D}_j^i$ on which Θ_{i+1} induces a strictly positive delay (the rest of the delay expression is nonnegative, cf. Equation (13)).
Looking at dependence relations other than e_j
By induction hypothesis 1, none of the first i dimensions of P fully satisfies a dependence relation e_l if $l \in [1+m_i, p]$. Thus, for any value of k in $[1, i]$, P_k induces a nonnegative delay on $\mathcal{D}_l$, and thus on $\mathcal{D}_l^i$, if $l \in [1+m_i, p]$. By induction hypothesis 3, the components of Θ of rank greater than or equal to $i+1$ satisfy the dependence relations e_l^i, for $l \in [1+m_i, p]$. Thus, Θ_{i+1} induces a nonnegative delay on e_l^i, for $l \in [1+m_i, p]$. Therefore, $\Theta_{i+1}+\sum_{k=1}^{i}\mu_{j,k}P_k$ induces a nonnegative delay on e_l^i, for $l \in [1+m_i, p]$, as a nonnegative linear combination of functions inducing nonnegative delays.
Satisfying the conditions on all the e_k's
$\Theta_{i+1}+\sum_{k=1}^{i}\mu_{j,k}P_k$ is an affine function which induces:
1. A nonnegative delay on the dependence relation e_j;
2. A nonnegative delay on the dependence relations $e_{1+m_i}^i, ..., e_p^i$;
3. A strictly positive delay on the operation to operation dependences of e_j^i on which Θ_{i+1} induces a strictly positive delay.

Furthermore, a quick look to the above arguments show that these properties also hold for $\Theta_{i+1}+\sum_{k=1}^{i}\nu_k P_k$, if for any k in $[1, i]$, $\nu_k \geq \mu_{j,k}$. Thus, for any $k \in [1, i]$, let $\mu_k = \max_{j\in[1+m_i,p]}\mu_{j,k}$. Then, $\Theta_{i+1}+\sum_{k=1}^{i}\mu_k P_k$ is an affine function which induces:
1. A nonnegative delay on the dependence relations $e_{1+m_i}, ..., e_p$;
2. A strictly positive delay on the operation to operation dependences of e_l^i ($l \in [1+m_i, p]$) on which Θ_{i+1} induces a strictly positive delay.

Furthermore, a quick look to the above arguments show that these properties also hold for $\Theta_{i+1}+\sum_{k=1}^{i}\nu_k P_k$, if for any k in $[1, i]$, $\nu_k \geq \mu_k$.
P_{i+1} and induction hypotheses 4 and 5 for dimension $i+1$
We are now ready to define the $(i+1)$-th component of P: $P_{i+1} = \Theta_{i+1} + \sum_{k=1}^{i}(1+\mu_k)P_k$ (thus induction hypothesis 4 is satisfied for $i+1$). From what precedes, P_{i+1} induces a nonnegative delay on all the dependence relations $e_{1+m_i}, ..., e_p$, as required. Furthermore, P_{i+1} satisfies all the operation to operation dependences of e_k^i ($k \in [1+m_i, p]$) satisfied by Θ_{i+1}.
Let us consider any k in $[1+m_i, p]$, and any operation to operation dependence f in e_k. We have two cases to consider.
1. f belongs to e_k^i. We have already established that P_{i+1} induces a strictly positive delay on f, if Θ_{i+1} does so.
2. f does not belong to e_k^i. Then, by definition of the e_l^j's, this dependence is satisfied by one of the first $(i-1)$ dimensions of Θ, say the l-th dimension. Thus, Θ_l induces a strictly positive delay on f. Thus, because

of induction hypothesis 5 at depth l, P_l induces a strictly positive delay on f. Therefore, P_{i+1} satisfies the operation to operation dependence f, as the coefficient of P_l in P_{i+1} is $1+\mu_l$ and not simply μ_l.

Therefore, for any k in $[1+m_i, p]$, P_{i+1} satisfies all the operation to operation dependences in e_k satisfied by any of the $(i+1)$-th first dimension of Θ. Corollary: for any j in $[1+m_i, p]$, P_{i+1} fully satisfies the dependence relation e_j if e^i_j is fully satisfied by Θ_{i+1}.

Induction hypothesis 2 for dimension $i+1$

For P_{i+1} to be the $(i+1)$-th dimension of a schedule *a la* FEAUTRIER, we still have to check that P_{i+1} fully satisfies as many dependence relations as possible, among $e_{1+m_i}, ..., e_p$. If this is not the case, from what precedes, Θ_{i+1} does not fully satisfy all the fully satisfiable e^i_j's. Then, we add to the $(i+1)$-th dimension of Θ an affine function that induces a nonnegative delay on all the dependence relations $e^i_{1+m_i}, ..., e^i_p$, and which fully satisfies as many of them as possible. This way we obtain a new schedule of minimal dimension. We restart our recursive construction (directly at the $(i+1)$-th rank). This time P_{i+1} maximizes the number of the dependence relations it fully satisfies.

Induction hypothesis 1 for dimension $i+1$

Without any loss of generality, let the dependence relations fully satisfied by P_{i+1} be $e_{1+m_i}, ..., e_{m_{i+1}}$. Then, the first $(i+1)$ dimensions of P satisfy the dependence relations e_1, ..., $e_{m_{i+1}}$, but none of the dependence relations $e_{1+m_{i+1}}$, ..., e_p.

Induction hypothesis 3 for dimension $i+1$

As P_{i+1} only fully satisfies the dependence relations $e_{1+m_i}, ..., e_{m_{i+1}}$, then Θ_{i+1} only fully satisfies $e^i_{1+m_i}, ..., e^i_{m_{i+1}}$. Θ, and thus Θ_{i+1}, is integral by hypothesis. Then the dimensions of Θ of rank greater than or equal to $i+2$ must satisfy the dependence relations $e^{i+1}_{1+m_{i+1}}$, ..., e^{i+1}_p where, if $j \in [1+m_{i+1}, p]$, $e^{i+1}_j = (S_j, T_j, \mathcal{D}^{i+1}_j, h_j)$, with

$$\mathcal{D}^{i+1}_j = \{x \in \mathcal{D}^i_j \mid \Theta_k(T_{i+1}, x, N) - \Theta_{i+1}(S_j, h_j(x, N), N) = 0\}$$

which, by induction hypothesis 3, is equivalent to:

$$\mathcal{D}^{i+1}_j = \{x \in \mathcal{D}_j \mid \forall k \in [1, i+1], \Theta_k(T_j, x, N) - \Theta_k(S_j, h_j(x, N), N) = 0\}.$$

■