PARALLEL-LOOP-EXECUTION TECHNOLOGY FOR
IMPLEMENTATION ON VECTOR PROCESSOR

O. V. Lukinova

Vector and pipeline computer systems make it possible to run programs synchronously, but the program must be specially organized for this — data must be processed with the longest possible series of operations of the same type. As a rule, a conventional sequential program can be converted by the apparatus of synchronous parallel execution to a form that satisfies this principle.

It is clear that the greatest benefit in such conversion is obtained in the loops. Fairly thoroughly studied in this regard is the class of linear multivariate loops of the form

\[
\text{DO } \alpha \quad \bar{I} \in \Delta \\
Q(I) \alpha \text{ CONTINUE,}
\]

(1)

where \(Q\) is the loop body, which satisfies the necessary and sufficient conditions of parallel execution [1]; the index expressions of the elements of blocks that enter the allocation operators of body \(Q\) are arbitrary linear combinations of the loop parameters; and \(\Delta = \{(I_1, I_2, \ldots, I_n) : I_i^{\min} < I_i < I_i^{\max}, i = 1, 2, \ldots, n, \text{ where } n \text{ is the loop size}\}\) is the vector space of loop iterations.

It is precisely the space \(\Delta\) that must be the object of study, if we wish to understand how the parallel execution of loops is accomplished. The nature of the relationships between iterations of the vector space provides the key to understanding how to select a set of iterations that can be executed simultaneously.

Before analyzing the iteration space, we shall introduce some concepts.

**Definition 1.** We shall call two iterations \(\bar{I} = (I_1, I_2, \ldots, I_n) \in \Delta\) and \(\bar{J} = (J_1, J_2, \ldots, J_n) \in \Delta\) data-dependent if there exists a pair of block elements of the same name (at least one of which appears on the left side of the allocation operator) \(M(e_1, e_2, \ldots, e_n)\) and \(M(e_1', e_2', \ldots, e_n')\) such that \(e_1(I) = e_1'(J), e_2(I) = e_2'(J), \ldots, e_n(I) = e_n'(J)\), i.e., in these iterations, both elements identify the same memory cell, and they can only be executed sequentially.

We connect such iterations \(\bar{I}\) and \(\bar{J}\) by a line segment and indicate on it the direction from an iteration performed earlier for example, \(\bar{I}\) — to an iteration to be performed later — for example, \(\bar{J}\) (Fig. 1). Such a relationship between iterations can always be established, since a lexicographic order of loop-parameter variation is naturally introduced into \(\Delta\): if \(\bar{I}, \bar{J} \in \Delta, J \in \Delta, \text{ then } Q(I)\) is executed before \(Q(J)\). These directed lines are called data-dependence vectors.

The set of data-dependence vectors for iterations \(\bar{I}\) and \(\bar{J}\) for a pair of block elements of the same name, at least one of which is the left side of the allocation operator, is described analytically by the following system:

\[
\begin{align*}
A \bar{I} + \bar{B} &= G \bar{J} + \bar{D}, \\
K &= \bar{J} - \bar{I} > 0.
\end{align*}
\]

(2)

Here, on the right and left sides of the first equation, are the index expressions of the pair in question written in vector form; the second equation imposes constraints on the execution order: \(\bar{I}\) is executed before \(\bar{J}\).

The combined set of solutions of systems of type (2) for the vector \(K\) for all pairs of like-named block elements provides a characteristic set \(\{K\}\) of all data dependences of the loop. Such a set is shown in Fig. 2 for the bivariate case, where \(\{K\}\) is the union of sets \(\Delta K_1\) and \(\Delta K\), the first of which consists of vectors with a zero order coordinate \((K_1 = 0)\) while the second consists of all other vectors.

Now we must define the law according to which a loop should be executed in order that, firstly, independent iterations are processed in parallel and, secondly, the data relationships between dependent iterations are not violated. In the class of linear loops, this law is defined by a hyperplane \(P\), which we shall call the parallel-execution hyperplane.

Definition 2. Let \( I_1, I_2, \ldots, I_n \) be an orthogonal coordinate system in \( \Delta \). \( X \in \Delta \) is a data-dependence vector space, and \( P \) is a hyperplane in space \( \Delta \) such that the orthogonal projection \( Pr_{\Pi} \vec{n} > 0 \), where \( \vec{n} \) is the normal to it. Then we call \( P \) the parallel-execution hyperplane for set \( X \) if it has the property that \( (\forall \vec{x} \in X \mid (\vec{n}, \vec{x}) > 0) \). This means that in some fixed position \( P \) contains only the starting point of the same data-dependence vector and, therefore, iterations that belong to \( P \) can be executed simultaneously, i.e., the condition \( (\forall \vec{x} \in X \mid (\vec{n}, \vec{x}) > 0) \) is satisfied, where \( C = \sum_{i=1}^{n-1} C_{ij} \), \( i = 1, 2, j = 1, 2, n-1 \), is the matrix of coefficients of the parallel-execution hyperplane. Thus, \( P \) specifies a linear transformation of the coordinates of the iteration space that permits parallel execution of an initial loop.

Therefore, the possibility of parallel loop execution is determined by the possibility of finding a parallel-execution hyperplane \( P \). The method of orientation of \( P \) in the iteration space determines the technology of parallel loop execution, i.e., the choice of a method of parallel execution. The coordinate method [2] (heavy lines in Fig. 1) works if \( P \) is parallel to some coordinate axis and the hyperplane method [3] (light lines in Fig. 1) can be used if \( P \) is at an angle to the axes, but the linear-transformation method [4] allows the parallel-execution hyperplane to be either parallel to or at an angle to the axes. It is more universal and combines the two previous methods.

As a result of any of the above-mentioned methods, initial loop (1) is converted to the following parallel form:

$$
\begin{align*}
&\text{DO } \alpha \text{ PAR FOR ALL } \vec{t} \in \Delta, \\
&Q(\vec{t}) \\
&\text{CONTINUE.}
\end{align*}
$$

A situation can arise in the iteration space, however, in which the data relationships described by system (2) can be represented as the union of two sets \( V_1^1 \) and \( V_2^1 \), which are described by systems (3) and (4), respectively (their characteristic sets correspond to \( \Delta_{K_1} \) and \( \Delta_{K} \) in Fig. 2):

$$
\begin{cases}
A \vec{J} + B = G \vec{J} + D, \\
\vec{K} = \vec{J} - \vec{t} > 0, \\
K_1 = 0;
\end{cases}
$$

$$
\begin{cases}
A \vec{J} + B = G \vec{J} + D, \\
\vec{K} = \vec{J} - \vec{t} > 0, \\
K_1 > 1.
\end{cases}
$$

In this case, a hyperplane \( P \) with the required property cannot be found [5, 6]. Then we have the method of piecewise-linear parallel execution, which seeks \( P \) for sets \( V_1^1 \) and \( V_2^1 \) separately. Now, if set \( V_2^1 \), on the strength of the lexicographic properties of the space, has vectors \( \vec{K} : K_2 = 0 \) and \( \vec{K} : K_2 \geq 1 \), i.e., if systems of types (3) and (4) are solvable for the new older coordinate \( K_2 \), we obtain another pair of sets \( V_1^2 \) and \( V_2^2 \), for each of which the piecewise-linear method finds its own