The efficiency of many of the existing data processing programs largely depends on the performance of the computer system and can be improved by better loading of the system, maximizing the utilization of expensive equipment, and saving material and labor resources in the production and operation of computer technology.

This paper is concerned with the semantics of the execution of a class of multimodule programs (MMP-programs) \([1]\) with external files (EF) on a multiprocessor computer (MPC) with various alternative methods of organization of calculations. Models of organization of the interaction of MMP-programs with EF buffering of exchanges with the external memory (EM), and computations based on the method of integrated multiprogramming (IMP), have been suggested in \([2-4]\), with special applications to the MMP-programs of this class. The IMP method provides parallel processing of large data bases and integrated data exchanges between the memory levels of MPC; it can be viewed as an implementation of the macroconveyer principle \([5]\) of EF processing. It is of interest, therefore, to examine the implementation of the IMP method on MPC, and to study the models of this method, evaluate the potential offered by MPC, and estimate the characteristics of this method for MPC and for the class of MMP-programs with EF.

Like in \([6]\), where the semantics of intermodule data exchanges in simple MMP-programs has been described, we operate in this paper with computation models based on digital dynamic systems \([7]\), including the interaction of MMP-programs with external memory of MPC.

Basic Concepts

An MMP-program of data processing is defined as a set of pairs \(G = ((K_1, P_1), \ldots, (K_N, P_N))\), where \(N \geq 1\), \(K_1, \ldots, K_N\) are the programs of these components; the latter consist of operators described below. For an abstract representation of these constituent programs, we will use, as in \([6]\), the algebra of algorithms over the memory \([8]\).

The programs \(P_1, \ldots, P_N\) of the components are expressions in the algebra of algorithms over the memory \(M = (U, Y, Z)\), constructed of the elements of the sets of elementary conditions

U = \bigcup_{i=1}^{N} U_i$, elementary operators $Y = \bigcup_{i=1}^{N} Y_i$, and operators of interaction with external informational environment $Z = \bigcup_{i=1}^{N} Z_i$; this interaction is accomplished by each component by means of operations of consecutive composition PR, branching $\alpha(F \lor R)$ and iteration $\alpha(F)$. An empty sequence of operations is an empty program. Elementary conditions and elementary operators act on internal informational environments of the programs; an internal informational environment of the program of the component $P_i$, $i \in \{1, \ldots, N\}$ is the set $B_i = \{b_i : X_i \rightarrow D\}$ of the memory states, i.e., the mappings of the set of variables $X_i$ of the program of $P_i$ into the data region $D$. We assume that the variables of the programs are localized within the components, i.e., the occurrence of a particular variable in the programs of different components is viewed as the occurrence of different variables. Each elementary operator $y \in Y_i$, $i \in \{1, \ldots, N\}$, defines a mapping $y : B_i \rightarrow B_i$; each elementary condition $\alpha \in U_i$, $i \in \{1, \ldots, N\}$ defines a predicate $\alpha : B_i \rightarrow \{0, 1\}$.

Denote by $M$ the set $BM$, and by $C_m$ the set of coordinates of the elements of the file $m \in M$. An element of a file will be subsequently denoted by a pair $(m, c)$, $m \in M$, $c \in C_m$. The notation $(M, C)$ denotes the set of all elements of all files. The set $B_0 = \{b_0 : (M, C) \rightarrow D\}$ of the mappings of the set of the elements of $BM$ into the data region $D$ is called the external information environment. In terms of the MAYAK language [9], we will examine the operators of interaction with an external information environment; there are four types of such operator: WRITE, READ, OCCUPY, and RELEASE (an element of $BM$). The writing operator $z' \in Z_i$, which writes the data into $BM$, defines the mapping $z' : B_i \rightarrow B_0$; the reading operator $z'' \in Z_i$ defines the mapping $z'' : B_0 \rightarrow B_i$, $i \in \{1, 2, \ldots, N\}$. For defining the operators occupying and releasing the elements of $BM$, we introduce a set of attributes of $BM$ engagement, which are a set of mappings $\Delta = \{\delta : (M, C) \rightarrow \{0, K_1, \ldots, K_N\}\}$, where zero is a symbol that does not coincide with any name of component.

The operator of occupation $z''' \in Z_i$ of an element of $BM$ is the mapping $z''' : \Delta \rightarrow \Delta'$, where $\Delta' \subset \Delta$ and $\Delta' = \{\delta : (M, C) \rightarrow \{k_i\}\}$, $i \in \{1, 2, \ldots, N\}$.

The operator of releasing $z'''' \in Z_i$ an element of $BM$ is a mapping $z'''' : \Delta \rightarrow \Delta''$, where $\Delta'' \subset \Delta$ and $\Delta'' = \{\delta : (M, C) \rightarrow \{0\}\}$.

The operators of interaction with $EF$ will be denoted as follows:

- writing operator $x \rightarrow (m, c)$,
- reading operator $(m, c) \rightarrow x$,
- occupation operator OCC $(m, c)$,
- release operator REL $(m, c)$

The operators READ (WRITE) carry out the READING (WRITING) of the elements of $EF$ from (into) $EF$, into (from) internal information environment. An application of the operator occupying an element of $EF$ in a component $K_i$ is the semantics of the MAYAK language means that only the program of that component can handle this operator. The release operator makes an element of $EF$ acceptable to the programs of other components. The semantics of the execution of MMP-programs of data processing will be described in terms of digital dynamic systems [7].

A Model of Natural Organization of Interaction

**A Model of Natural Organization of Interaction with $BM$ ($Sp$)**

A digital dynamic system is defined as a set of states $S$ and a system of transition rules $(\Rightarrow)$ defined on that state. A sequence $pr = s_0, s_1, \ldots, s_n, \ldots$, where $s_j \Rightarrow s_{j+1}$ for all $j = 0, 1, \ldots, n - 1, \ldots$, is called a process in this digital system; $s_0$ is its initial state. In the set of states $S$ a subset of final states $\{s^f\}$ is identified; by definition, transition rules are not applied to such states. The nonfinal states to which transition rules cannot be applied are deadlock states. A process that leads to a final state is called completable; a process leading into a deadlock is called a deadlock process.

We define the state of a digital dynamic system $Sp$ as the vector of the states of components of MMP-program, the states of the external information environment, and the states of occupation attributes of $EF$:

$$s_E = ((b_1, P_1), \ldots, (b_N, P_N), b_0, B_0).$$