UDC 621.3

G.F. KRIVOULYA, O.S. KOROBKO, A.I. LYPCHANSKIY, D.E. SHUKLIN

Kharkov National University of Radioelectronics, Ukraine

THE STRUCTURE AND CONSTRUCTION FEATURES OF OPERATIONAL SYSTEM FOR SEMANTIC NEURON NET MODEL

The description of multilevel system, which allows realizing computing system on the basis of neuron net, is offered in the given work. The description of each level of operational system is proposed. The collection of objects representing the interaction of system levels is considered.

Expert system, neuron, neuron net, virtual machine, memory block, operational system interface level, memory management level, frame repository, frame caching interface

Introduction

Nowadays a great variety of methods and instrumental means of system and net monitoring and diagnostics exist. It often happens that the problems of diagnostics have to be solved in the conditions of lack of information, scarce efficiency of experts or in the dangerous or harmful conditions for men. To overcome these problems expert systems (ES) are used.

The process of ES creation on the basis of semantic neuron net (NN) is considered in the given work.

A large number of various models of induced neuron nets exist today. All of them differ by various algorithms of behavior of separate neurons and connection topology construction principles. In spite of a great number of offered neuron net mathematical models presence, the scarce of these models realization on the sequential and parallel computing systems takes place.

The majority of NN models possess considerable similarity. The neuron represents some object which is connected with the other objects. Some data is transmitted over the connections between the neurons. The neurons process these data and deliver it to the other neurons. The NN can be considered as distributed computing system from this point of view. The neurons correspond to separate processors and the connections between the neurons correspond to data transmission channels between these processors in such a system [1].

Publication analysis. In the work [1] the semantic neuron net is considered. The neuron net as a formal language allows to process text senses as some algebra function. The functions are constructed from the separate neurons, fulfilling disjunction, conjunction and negation. The neuron net structure determines the order of basic algebra operations application processing incoming data. The separate neuron denominates an elementary notion of the analyzed language.

In the work [2] semantic neuron net structure is considered. The realization of parallel calculations on the basis of sequential computing system in the semantic neuron net is suggested. Decision taking time optimization and regard management is analyzed in the expert system ELEX-4, which was realized on the basis of semantic neuron net.

In the work [3] virtual machine transformation principle is considered. The virtual machine, providing semantic neuron net functioning, is realized on the basis of this principle. The inner structure of the neuron and neuron data depository organization is considered.

Problem statement. In the connection with the wide propagation of Intel architecture, the authors consider that it's appropriate to develop the universal virtual machine, which allows simulating different NN in the sequential computing system environment. Nowadays object oriented methods of software development got a wide spreading. In view of this fact, it seems rational to realize neuron net level as object-oriented database. The neurons will realized as objects and the connections between the neurons will be realized as object references. It is necessary to isolate separate neu-

rons behavior realization details from virtual machine in various models of neuron nets. To provide the maximum flexibility of system realignment the virtual machine has to be realized as several levels connected by legibly determined interfaces.

The main aim of the given project is to create the virtual machine maintaining the neuron net with free topology and a number of neurons about two milliard items in one depository. This possibility is provided by management system of net object-oriented database. Therefore only the part of objects is concentrated in the working memory at each instant of time. The most part of objects is contained in the file depository.

The management system of net object-oriented database has to possess the following possibilities:

– to keep the current state of object graph or neuron net between the séances of user interaction. To keep the current topology of object net. The object net has not to be created anew if the application is launched again.

– to limit memory resources used by object graph or NN in the case of a large number of object instances. The most frequently used objects remain in the working memory, while the rest of instances is replaced to the file depository and is loaded to the memory as required. In the process of instance loading to the main memory, the current instance displaces the seldom used items.

The use of management system of net object-oriented database doesn't impose limitations on used business logic and mathematical neuron model, which can be realized as methods of objects, situated in database. The main requirement is to organize the connections between the objects in net not with the heap of pointers, but with the application of object identifiers. An object pointer is to be obtained by using of Application Programming Interface (API) of management system of net object-oriented database.

Computing system realization

The virtual machine, effecting the transformation of NN abstractions into operational system abstractions is realized as multilevel system [4]. Each level is represented by the set of API functions. The collection of API of all levels of virtual machine is called Virtual Neural Programming Interface (VNPI). VNPI core has four levels: System Resource Manager (SRM) – operational system interface level, Virtual Memory Manager (VMM) – memory management level and garbage assembler, Virtual Cluster Manager (VCM) – the level of management interfaces of frame depository file, Virtual Frame Manager (VFM) – frame caching interface level.

The operational system represents the program fulfilling the distribution of computing system resources among different processes. SRM-level provides the adaptation of operational system API functions to the needs of virtual machine, which realizes semantic neuron net. VMM-level provides memory management. Memory operation possesses an important particularity in the described system. All memory blocks are situated in the list of most recently used memory blocks. In the lack of system resources the list is scanned to find the least recently used blocks. If the CALLBACK-function of garbage assembler is determined for the allocated memory block, the given can be destroyed by memory management subsystem. The CALLBACK-function of garbage assembler is previously called for the present block in that case. Memory block and the reason of the call are indicated. This mechanism serves to deallocate the least recently used by the system neurons. The CALLBACK-function of garbage assembler preserves neuron body data in the neuron depository (VCM) before it destructs the neuron body.

VCM-level provides the functioning of frame depository. Data depository represents long duration data structure, which contains neuron state describing data. Data depository represents the common file by itself on the operational system level. The depository structure is organized in such way, that the neuron state is kept intact whether the copies of the given neurons are loaded to the computer memory or not. This is achieved by the exclusion of all context-depended data, such as file pointers or memory pointers from the depository. While creating new neuron, the search of some free cluster carried out in the neuron depository. The address of this cluster is returned as unique neuron identifier. If the neuron enlarges its sizes and exceeds the bounds of its cluster during the process of its existence, depository management

level finds one more free cluster and adds it to the end of the used by the neuron cluster list. This mechanism provides the possibility of preservation of neurons of any necessary size. In the given realization the limitation of the neuron size is The possibility of keeping of the neurons of any required size is provided. In the given realization, the limitation on one neuron size is up to 2^{31} bytes – the restriction of the signed integer value size, which can be placed in the register of x86 processor.

Physically the neuron is realized as data stream in the depository. Due to this every neuron can be regarded as some file. As a result, neuron processing fulfils by means of file processing functions, such as reading and writing of data block, file pointer relocation.

An identification, search and neuron processing in the depository are provided for neuron identifiers. Neuron identifier is an important notion of data repository. This identifier constitutes the integer. While being created, each neuron gets an identifier, which stays permanent in the course of neuron's life. After the neuron was destructed, the identifier can be appointed to another neuron. There can't be two different neurons with the same identifier in the depository. If a 32-bit number is taken as an identifier, the depository can contain up to 2^{31}= 2'147'483'648 different neurons. All the operations under the neurons are carried out with the indication of their identifier. It's convenient to set the special NULL identifier, which has a null value. No neuron can have a NULL identifier. The replacement of the identifier by NULL value allows signalizing that the identifier is not connected with any concrete object without additional expenses.

The VFM-level provides the functioning of API-functions of frame caching. On this level, the neuron bodies represent the frames, situated in the repository. The VFM-level loads neuron data from repository to the operational system memory. The main memory block is picked out by operational system level on request of neuron body management level. Then, the depository fills this memory block by the concrete neuron data upon request of body management level. As a result, two copies of neuron data appear: the obsolete copy in the depository and the actual one in the working memory. After finishing neuron processing neuron body management level retains neuron body data in the depository and removes the memory block from the working memory. On the body management level each neuron can be considered as a file or a data flow, which is reflected to the computing system memory. The neuron belongs to its depository and it has an identifier, which stays stationary during the whole life cycle of the file. Frame management level considers the neuron to be some collection of data without inner structure.

The unique neuron identifiers loaded to the system memory are kept as binary trees by caching level. This provides the possibility of finding memory block address of the neuron by its unique identifier. If the neuron hasn't been loaded to system memory, caching level turns to memory management subsystem and allocates the memory block of the necessary size. Then it loads neuron body data to this memory block. The CALLBACK-function of garbage collector is set for the memory block. If the necessity of clearing of the block appears, memory management subsystem turns to caching level with the help of CALLBACK-function. The caching level preserves neuron body data in the depository and liberates memory block to get it being used for the other purposes. So, the displacing algorithm of neuron caching is realized in the virtual machine.

The realization of neuron sections management may be fulfilled in such a way, that each new section with the section identifier can be correspondingly interpreted as some neuron body. As a result, the structure, shown by fig. 1, can be obtained. The number of taken up subsections is limited only by the physical restrictions of the computing system in this structure.

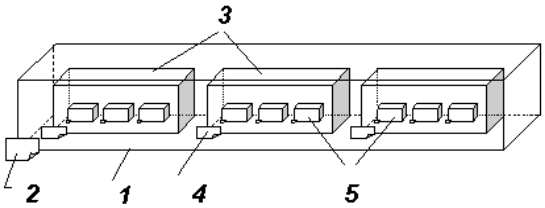

Fig. 1. The inner structure of the neuron:
1 – neuron body data; 2 – neuron identifier in the depository; 3 – section body data; 4 – section type identifier; 5 – other levels data, situated in the section body

Each neuron section can be considered as data flow or a file, which is reflected to the memory. It can be

interpreted by the other levels by any means. Section behavior logic level will consider each section as table structure, which consists of the unlimited number of records of N fields in each

The section fields are numerated by non-negative integer numbers with the step 1. Each field has its fixed type. Record arrangement order in the neuron section is called natural order of record disposition. Due to this organization of the separate neurons, the possibility of fulfillment of relational algebra operations, such as intersection, conjunction and others exits. These operations are used for neuron sections processing as indigested data. The possibility of keeping of rank-order data blocks (text strings as rank-order tables from one field of character type, for example) also appears. Both of possibilities are realized without additional computing machine productivity waste.

Section behavior logic level doesn't apply restrictions on usage of table structures, realized in neuron section. In future, the storage of neuron identifiers and section identifiers in the form of lists or relations will be the main application of such table structures.

Section behavior logic level determines the purpose of neuron attributes which have been defined earlier and the algorithms of their usage. The relations of some neuron with another neuron – neuron type descriptor – and the relations of some neuron with another neuron – section type descriptor – are pointed on fig. 2.

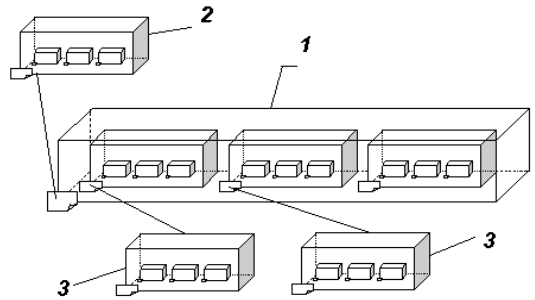

Fig. 2. Connection between the neurons realization example: 1 –some neuron; 2– neuron - neuron type descriptor; 3 – neuron - section type descriptor

Each neuron belongs to the type, from which it has been generated. Type descriptor is also a neuron and it has its identifier. On the analogy of object-oriented programming, the neurons – type descriptors can be called class types. Each neuron can play a role of the class. The neuron generation from some class can be called an inheritance. The unlimited number of the neurons can be generated from one class. Each neuron is subdivided on sections. Each section has its type. The section neuron is bearer of section type. Each section will have its numerical identifier, which is equal to the identifier of the section neuron.

Neuron depository organization

An expert system can contain the milliards of objects. Even if occupied by one project memory size is not so great, main memory capacity is not large enough to keep all objects. To overcome the mentioned inconvenience, the list of most recently used neurons is created. VMM level is charged with creating and keeping of this list. There exist restrictions on the number of objects, present in the list (handle limit) and on the used memory size (memory limit). The neurons which didn't get into the list are kept on the hard disk.

It is necessary creating data depository to provide the effective functioning of the VMM level. The information will be read from and written into the depository. VCM cluster manager was constructed for this purpose.

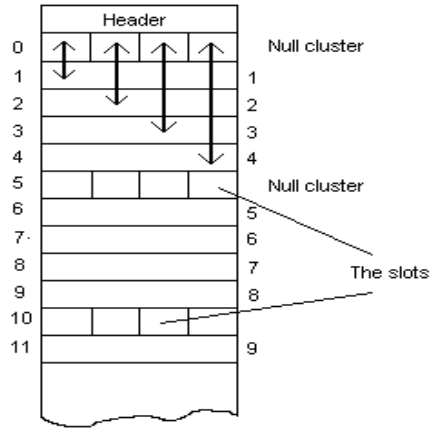

Fig. 3. Inner organization of the neuron depository

VCM works on the basis of file. One VCM base represents one file in the operational system. The given file contains the header defining file type, its parameters, the number of clusters and the number of the first free cluster.

The structure of one VCM file is represented by the following way: the file is subdivided into blocks, containing m+1 clusters in n bytes. Each of first clusters of the block is subdivided by m slots, and there's certain correspondence of clusters and slots.

VNPI characteristics

The represented structure of neuron net virtual machine was realized as multilevel system by the way of its description by the set of API functions. Virtual machine core, adjusted by the authors, possesses the following features.

1. Universality. The details of separate neurons behavior realization in the various models of NN are isolated of the virtual machine, which gives the opportunity of usage of the developed virtual machine for any model of neuron net.

2. Flexibility. The virtual machine is realized in the form of several levels, which are connected by legibly determined interfaces with each other. This allows providing maximal flexibility of system realignment for the required model of neuron net.

3. The effectiveness of resource management. Only the part of neuron objects is situated in the working memory at each point of time. The majority of the objects are kept in the file depository. In the case when the size of the object net is less than current free memory volume in the system, the whole net is situated in the working memory. This gives the possibility not to waste time on the loading of the neurons from the depository.

4. The effectiveness of connection between the objects organization. The connection between the objects in the net are realized not with the help of pointers, but with the usage of object identifiers, so we can apply no restrictions on the used mathematical neuron model.

5. The possibility of keeping of the net state between the séances of work with the users. While launching the application again, the current net topology remains, so the object bet has not to be created anew.

Conclusion

The architecture of virtual machine core was represented in the given work. The virtual machine was designed with the purpose of ES building on the basis of represented model. The core is realized in the component programming paradigm, which provides the possibility to change the core architecture dynamically without separate component recompilation. The represented virtual machine emulates parallel processing of the neurons in the net. VNPI levels consisting of the set of classes and the corresponding methods were described.

The later development of the expert system is proposed on the basis of the created program NN model. This ES is designed to solve non-formalized problems in the sphere of projection, diagnostics, computer system and net production, computer technology support. The following stage of ES development is the working-out and debugging of client application of ES and development and checkout of knowledgebase of computer complex fault diagnostics as well.

References

1. Shuklin D.E. The models of semantic neuron net and their application in the artificial intelligence systems. // PHD dissertation: – Kharkov, KNURE, 2003. – 196 p.

2. Shuklin D.E. Semantic neuron net application in the expert system, which transforms natural language text senses. // Radio electronics and informatics. – Kharkov, KNURE, 2001. – № 2. – P. 61-65.

3. Dudar Z.V., Shuklin D.E. The realization of neurons in the semantic neuron nets. // Radio electronics and informatics, Kharkov, KNURE, 2000. – № 4. – P. 89-96.

4. Shuklin D.E. The principles of construction of core component architecture of virtual machine, emulating semantic neuron net // The 8-th int. youth forum "The radioelectronics and the youth in the XXI century" P. 2: Forum material collection. – Kh., KNURE, 2004. – P. 101.

Надійшла до редакції 22.02.2006

Рецензент: д-р техн. наук. професор В.М. Ілюшко, Національний аерокосмічний університет ім. М.Є. Жуковського, Харків.