Modularity in Artificial Neural Networks

Ricardo Téllez
Technical University of Catalonia, Spain

Cecilio Angulo
Technical University of Catalonia, Spain

INTRODUCTION

The concept of modularity is a main concern for the generation of artificially intelligent systems. Modularity is an ubiquitous organization principle found everywhere in natural and artificial complex systems (Callebaut, 2005). Evidences from biological and philosophical points of view (Caelli and Wen, 1999) (Fodor, 1983), indicate that modularity is a requisite for complex intelligent behaviour. Besides, from an engineering point of view, modularity seems to be the only way for the construction of complex structures. Hence, whether complex neural programs for complex agents are desired, modularity is required.

This article introduces the concepts of modularity and module from a computational point of view, and how they apply to the generation of neural programs based on modules. Two levels, strategic and tactical, at which modularity can be implemented, are identified. How they work and how they can be combined for the generation of a completely modular controller for a neural network based agent is presented.

BACKGROUND

When designing a controller for an agent, there exists two main approaches: a single module contains all the agent's required behaviours (monolithic approach), or global behaviour is decomposed into a set of simpler sub-behaviours, each one implemented by one module (modular approach). Monolithic controllers implement on a single module all the required mappings between the agent's inputs and outputs. As an advantage, it is not required to identify required sub-behaviours nor relations between them. As a drawback, whether the complexity of the controller is high, it could be impossible at practice to design such a controller without obtaining large interferences between different parts of it. Instead, when a modular controller is used, the global controller is designed by a group of sub-controllers, so required sub-controllers and their interactions for generating the final global output must be defined.

Despite the disadvantages of the modular approach (Boers, 1992), complex behaviour cannot be achieved without some degree of modularity (Azam, 2000). Modular controllers allow the acquisition of new knowledge without forgetting previously acquired one, which represents a big problem for monolithic controllers when the number of required knowledge rules to be learned is large (De Jong et al., 2004). They also minimize the effects of the credit assignment problem, where the learning mechanism must provide a learning signal based on the current performance of the controller. This learning signal must be used to modify the controller parameters which will improve the controller behaviour. In large controllers, it becomes difficult finding changing parameters of the controller based on the global learning signal. Modularization helps to keep small the controllers' size, minimizing the effect of the credit assignment.

Modular approaches allow for a complexity reduction of the task to be solved (De Jong et al., 2004). While in a monolithic system the optimization of variables is performed at the same time, resulting in a large optimization space, in modular systems, optimization is performed independently for each module resulting on reduced searching spaces. Modular systems are scalable, in the sense that former modules can be used for the generation of new ones when problems are more complex, or just new modules can be added to the already existing ones. It also implies that modular systems are robust, since the damage on one module results in a loss of the abilities given by that module, but the whole system is partially kept functioning. Modularity can be a solution to the problem of neural interference (Di Ferdinando et al., 2000), which is encountered in monolithic networks. This phenomenon

is produced when an already trained network losses part of its knowledge when either, it is re-trained to perform a different task, called temporal cross-talk (Jacobs et al.,1991), or two or more different tasks at the same time, the effect being called spatial cross-talk (Jacobs,1990). Modular systems allow reusing modules in different activities, without re-implementation of the function represented on each different task (De Jong et al., 2004) (Garibay et al., 2004).

Modularity

From a computational point of view, modularity is understood as the property that some complex computational tasks have to be divided into simpler subtasks. Then, each of those simpler subtasks is performed by a specialized computational system called a module, generating the solution of the complex task from the solution of the simpler subtask modules (Azam, 2000). From a mathematical point of view, modularity is based on the idea of a system subset of variables which may be optimized independently of the other system variables (De Jong et al., 2004). In any case, the use of modularity implies that a structure exists in the problem to be solved.

In modular systems, each of the system modules operates primarily according to its own intrinsically determined principles. Modules within the whole system are tightly integrated but independent from other modules following their own implementations. They have either distinct or the same inputs, but they generate their own response. When the interactions between modules are weak and modules act independently from each other, the modular system is called nearly decomposable (Simon, 1969). Other authors have identified this type of modular systems as separable problems (Watson et al., 1998). This is by far one of the most studied types of modularity, and it can be found everywhere from business to biological systems. In nearly decomposable modular systems, the final optimal solution of a global task is obtained as a combination of the optimal solutions of the simpler ones (the modules).

However, the existence of decomposition for a problem doesn't imply that sub-problems are completely independent from each other. In fact, a system may be modular and still having interdependencies between modules. It is defined a decomposable problem as a problem that can be decomposed on other sub-problems, but the optimal solution of one of those problems depends on the optimal solution of some of the others (Watson, 2002). The resolution of such modular systems is more difficult than a typical separable modular system and it is usually treated as a monolithic one in the literature.

Module

Most of the works that use modularity, use the definition of module given by (Fodor, 1983), which is very similar to the concept of object in object-oriented programming: a module is a domain specific processing element, which is autonomous and cannot influence the internal working of other modules. A module can influence another only by its output, this is, the result of its computation. Modules do not know about a global problem to solve or global tasks to accomplish, and are specific stimulus driven. The final response of a modular system to the resolution of a global task, is given by the integration of the responses of the different modules by a especial unit. The global architecture of the system defines how this integration is performed. The integration unit must decide how to combine the outputs of the modules, to produce the final answer of the system, and it is not allowed to feed information back into the modules.

MODULAR NEURAL NETWORKS

When modularity is applied for the design of a modular neural network (MNN) based controller, three general steps are commonly observed: task decomposition, training and multi-module decision-making (Auda and Kamel, 1999). Task decomposition is about dividing the required controller into several sub-controllers, and assigning each sub-controller to one neural module. Modules should be trained either, in parallel, or in different processes following a sequence indicated by the modular design. Finally, when the modules have been prepared, a multi-module decision making strategy is implemented which indicates how all those modules should interact in order to generate the global controller response. This modularization approach can be seen as at the level of the task.

The previous general steps for modularity only apply for a modularization of nearly decomposable or separable problems. Decomposable problems, those