Duplication of Modules Facilitates the Evolution of Functional Specialization

Raffaele Calabretta,[1,3]
Stefano Nolfi,[1]
Domenico Parisi,[1] and
Günter P. Wagner[2,3]

[1] Department of Neural Systems and Artificial Life
Institute of Psychology, C.N.R.
Rome, Italy
rcalabretta@ip.rm.cnr.it

[2] Department of Ecology and Evolutionary Biology
Yale University
New Haven, CT, 06520, U.S.A.
and

[3] Center for Computational Ecology
Yale University
New Haven, CT, 06520, U.S.A.

Abstract The evolution of simulated robots with three different architectures is studied in this article. We compare a nonmodular feed-forward network, a hardwired modular, and a duplication-based modular motor control network. We conclude that both modular architectures outperform the non-modular architecture, both in terms of rate of adaptation as well as the level of adaptation achieved. The main difference between the hardwired and duplication-based modular architectures is that in the latter the modules reached a much higher degree of functional specialization of their motor control units with regard to high-level behavioral functions. The hardwired architectures reach the same level of performance, but have a more distributed assignment of functional tasks to the motor control units. We conclude that the mechanism through which functional specialization is achieved is similar to the mechanism proposed for the evolution of duplicated genes. It is found that the duplication of multifunctional modules first leads to a change in the regulation of the module, leading to a differentiation of the functional context in which the module is used. Then the module adapts to the new functional context. After this second step the system is locked into a functionally specialized state. We suggest that functional specialization may be an evolutionary absorption state.

Keywor ds
modularity, genetic duplication, neural networks, genetic algorithms, adaptive behavior

1 Introduction

Mathematical models have been rather successful in representing the population genetic mechanisms of adaptation, molecular evolution, and speciation [4, 5, 10]. One major class of evolutionary processes, however, has received relatively little attention from theorists, that is, evolutionary innovation. Innovation is defined here as the origin of new body parts and/or new body plans [16]. The process of innovation poses particular challenges for mathematical modeling, because it involves the origin of new units that are usually assumed to be invariant in the classical mathematical models of evolutionary processes [21]. Here we demonstrate that the Artificial Life modeling approach can be a powerful tool to investigate innovation processes because of the openness of Artificial Life models.

Higher-level multicellular organisms are characterized by a high degree of differentiation, where quasi-autonomous parts of the body are often dedicated to one or a few major functions [22]. These parts have been called organs or homologues. Hence, one of the most obvious trends in organismal evolution is the increase in the maximal

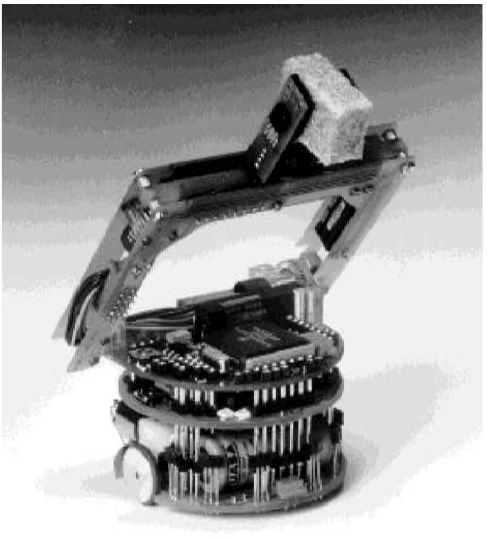

Figure 1. The Khepera robot.

complexity of a clade [12]. Little is known, however, about the mechanisms that lead to the origin of functionally specialized body parts.

In already published work [2, 3] we have shown that the duplication of functional units leads to the evolutionary specialization of the duplicated units, similar to the functional specialization of duplicated genes. This is in contrast to the evolution of units which do not arise by duplication [17]. In this case the functional tasks tend to be distributed among redundant units without any obvious division of function. However, important questions which remain to be answered are why or how duplication leads to functional specialization, and why the evolution of redundant units that are hardwired in the system and that do not arise because of genetic duplication does not lead to specialization. In the present article we review our previous results and we specifically address this question.

The Artificial Life literature does not contain much work which addresses these kinds of questions. However, relevant work includes Koza [11], who has used gene duplication in genetic programming, and Gruau [7], who has proposed a genetic encoding scheme for neural networks based on the cellular duplication and differentiation process. For an interesting discussion on how gene duplication supports modularity see also Rotaru-Varga [19].

2 The Model

For a detailed description of the experimental setup we refer the reader to Calabretta and colleagues [2, 3]. Here we summarize the model used in the simulations.

A population of neural networks [20] is evolutionarily trained to control a mobile robot designed to keep an arena clear by picking up trash objects and releasing them outside the arena. The "organism" is a miniature mobile robot (Khepera [15]; see Figure 1), which is supported by two wheels that allow it to move in various directions by regulating the speed of each wheel. In addition, the robot is provided with a gripper

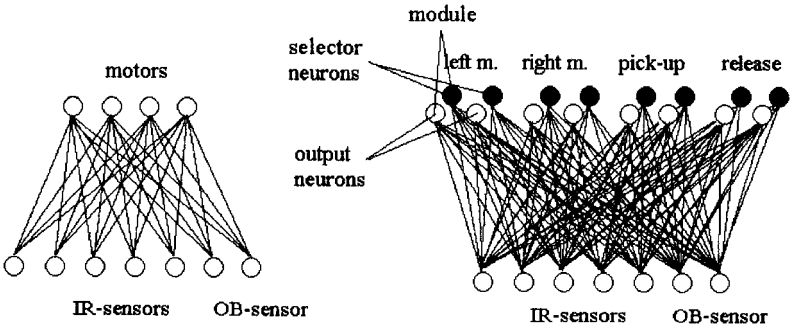

Figure 2. Architectures (a) and (b) are shown on the left and right sides, respectively. Architecture (a) is used in the nonmodular population. Architecture (b) is the basic architecture used in the two modular populations (i.e., in both the hardwired and duplication-based modular populations). The two populations differ in the type of modularity that is added to the basic architecture. In the hardwired modular population two modules compete to gain control of each of the four actuators in all individuals from the beginning of evolution. In the duplication-based modular population the individuals of the initial generation have only one module for each motor, that is, they initially have architecture (a). A second competing module may be added in individuals of successive generations as a result of the duplication operator. Another difference is that in the first modular population, competing modules have different random weights at the beginning, while in the second modular population, when a second competing module is generated, the two competing modules have identical weights.

module with two degrees of freedom. The robot is also provided with eight infrared proximity sensors and an optical barrier (OB) sensor on the gripper capable of detecting the presence of an object between the two arms of the gripper. The environment is a rectangular arena surrounded by walls containing 5 target cylindrical objects, which are positioned randomly inside the arena. The evolutionary process is conducted only in simulation in order to speed it up [13].

We compared the results obtained with modular and nonmodular neural network architectures (see Figure 2). In both cases the robot has 7 sensory neurons and 4 motor neurons. The first 6 sensory neurons are used to encode the activation level of the corresponding 6 frontal sensors of Khepera (the two back sensors are ignored) and the seventh sensory neuron is used to encode the OB light sensor on the gripper. On the motor side the 4 neurons respectively codify for the speed of the left and right wheels and for the triggering of the "object pick up" and "object release" procedures. The logistic function is used to determine the activation of the motor neurons.

The nonmodular architecture (Figure 2, left) is a simple feed-forward network with 7 input units encoding the state of the 7 sensors and four output units encoding the state of the 4 effectors. The input units are directly connected to the output units through 28 connection weights (plus 4 biases). This architecture is not divided into modules. The other two architectures are modular ones and differ in the type of modularity that enriches their architecture (Figure 2, right).

The architecture of the first modular population (*hardwired modular architecture*) has 16 output units, which, at every time step, give 4 output values controlling the 4 previously described effectors. Four pairs of output neurons (represented by empty circles) code for the speed of the left and right motors and for the triggering of the "object pick-up" and "object release" procedures, respectively, and four pairs of selector neurons (represented by full circles) determine which of the two competing output neurons have control over the corresponding effector at each time step (the competitor with the more highly activated selector neuron gains control). Each module is composed of two output neurons, the two corresponding biases, and 14 connections from sensory neurons. The first output neuron determines the motor output when the module has control, and the second output neuron (selector) competes with the selector

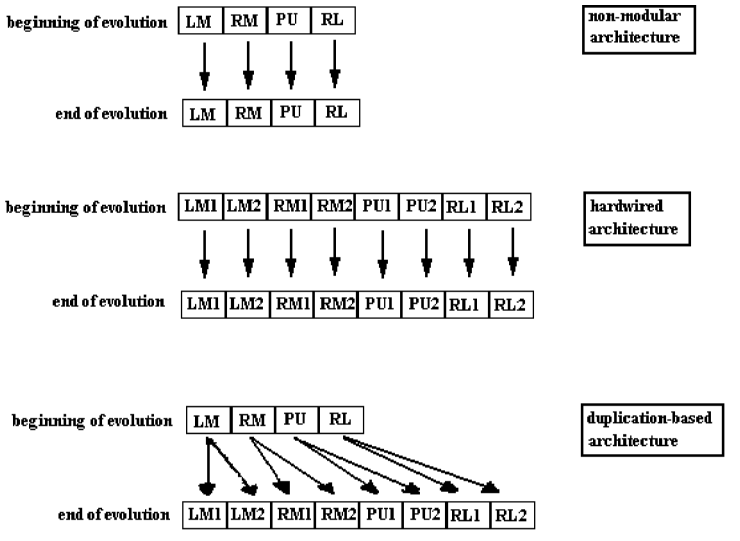

Figure 3. Schematic representation of the genomes of the three architectures. (LM = genetic encoding for the connection weights of the left motor; RM = right motor; PU= pick-up motor; RL = release motor. Genetic encoding for selectors is not indicated).

neuron of the other corresponding module to determine which of the two modules has control.

The architecture of the second modular population is called *duplication-based modular architecture* because, in this case, the modules are not hardwired in the architecture from the beginning of evolution, but can be added during the evolutionary process. Each module, as in the case of the hardwired architecture, consists of two output units (one motor output unit and one selector unit) which receive connections from the 7 sensors. At the beginning of the evolutionary process there is only one module for each of the four outputs, that is, the same module always controls the corresponding output. However, during reproduction, modules may be duplicated (see below). Duplicated modules, which are exactly the same when duplication takes place, can differentiate across generations because of genetic mutations.

A genetic algorithm [8] was used to evolve the connection weights of all the neural networks (Figure 3). In the nonmodular population the genotypes of the initial generation encode random values for the connection weights of the single modules of the basic architecture: 32 ($7 \times 4 = 28$ plus 4 biases) connections. Since each weight value is binarily encoded using 8 bits, the total genotype is a sequence of $32 \times 8 = 256$ bits. In the hardwired modular population the genotype encodes the values for all the connection weights of the modular architecture. Since each module includes 7×2 connections plus 2 biases and there are 8 modules, the total number of connection weights encoded in the genotype is 128. The total genotype is a sequence of $128 \times 8 = 1024$ bits. The individuals of the first generation are assigned random values for these 1024 bits and then the evolutionary process progressively finds better and better genotypes on the basis of the selective reproduction of the best individuals and the addition of random mutations to the inherited genotypes. Each generation includes 100 individuals.

Each individual was allowed to "live" for 15 epochs, each epoch consisting of 200 input-output cycles or actions. At the beginning of each epoch the robot and the target objects are randomly positioned in the arena. An epoch is terminated either after 200 actions or after the first object had been correctly released. Individuals were scored for

their ability to perform the complete sequence of correct behaviors, that is, for their ability to find and pick up objects, carry them to the edge of the arena, and release them so that they fall outside the arena. However, in order to facilitate the emergence of this ability individuals were also scored (although with a lower reward) for their ability to pick up objects. At the end of life the 20 best individuals are selected for reproduction and each of these individuals generates 5 offspring, that is, new individuals with the same genotype as their parent. Reproduction consists in generating copies of an individual's genotype encoding the network's connection weights (we are assuming nonsexual reproduction in haploid populations) with the addition of random changes to some of the bits of the genotype sequence (genetic mutations; we did not use genetic crossover) and, in the case of the duplication-based modular architecture, the duplication of a randomly selected neural module. Genetic mutations consist in changing the value of about 10 bits in each genotype (1% mutation rate). The $20 \times 5 = 100$ new individuals constitute the second generation. The process is repeated for 1000 generations.

In the duplication-based modular population the genotypes of the initial generation encode random values for the connection weights of the single modules of the basic architecture: 32 ($7 \times 4 = 28$ plus 4 biases) connections. However, since each of the 4 output units has associated with it a nonfunctional selector unit with its 7 connection weights, the total number of connection weights encoded in the genotypes of the initial generation is 64. Notice, however, that until the module happens to be duplicated this selector unit remains completely nonfunctional and its associated connection weights are subject to random drift only. The genotype of this second modular population has 4 additional "duplication genes" each associated with one of the 4 output units. When one of these duplication genes is turned on by some mutation the gene duplicates its corresponding module assigning to the duplicated module the same weight values as the original module. The duplication genes cause a duplication with some probability that we have varied in different simulations. We have used 3 different probabilities of duplication: 0.02%, 0.03%, and 0.04%. (We did not test higher duplication probabilities because with a 0.04% probability we already obtained performance levels comparable to those obtained with the hardwired architecture). In the generation in which the duplication of one of the modules takes place there is no possible change in behavior since both the original and the duplicated module have the same connection weights. However, subsequently random mutations acting on the modules' connections weights (both on those leading to the output unit and those leading to the selector unit of the module) can progressively differentiate the two alternate modules.

In the present model the maximum number of duplicated modules allowed in the case of the duplication-based modular architecture is one for each motor output, and no module-deletion operator was used. As a result, the hardwired modular architecture, already described in Nolfi [17], is the most complex architecture that can possibly evolve starting from architecture (a). However, the addition of modules during the course of evolution (instead of right from the beginning) that are initially identical to their competing module (instead of being completely unrelated) may produce qualitatively different results in the case of the hardwired and duplication-based modular architectures, respectively.

3 Results

We have conducted several sets of simulations in which we compare (a) a simple nonmodular feed-forward neural network, (b) the hardwired modular architecture (i.e., a modular architecture that is predesigned as modular right from the beginning of the simulation and remains fixed throughout the evolutionary process), and (c) the

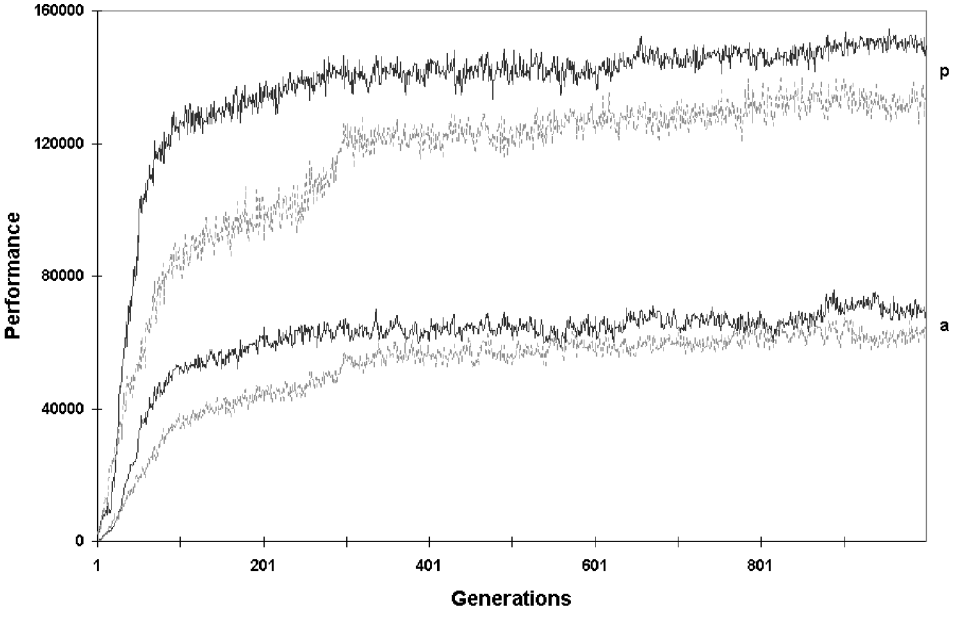

Figure 4. Average (a) and peak (p) performance of a population with **nonmodular architecture** (gray curve) and of a population with **hardwired modular architecture** (black curve). Average of 10 different runs.

duplication-based modular architecture (i.e., a modular architecture that evolves starting from a population of nonmodular ones as a result of gene duplication). In all simulations we used a mutation rate of 1%, that is, 2% of the bits of the genotype randomly selected were replaced by a new randomly selected value. We ran 10 simulations for each of the 3 different architectures described above. Each simulation started with populations of 100 networks with randomly assigned connection weights and lasted 1000 generations.

The hypothesis to be tested with these simulations is that *modular architectures which originate in genetic duplication favor the emergence of functional module specialization.* Moreover, if this prediction is confirmed, we would like to understand the mechanisms by means of which functional specialization is realized.

Nolfi [17] reported that hardwired modular architecture clearly outperformed nonmodular architecture in a garbage-collecting task. This is confirmed by the results shown in Figure 4 which gives the average and peak performance (respectively, the average performance and the performance of the best individual in each generation) for the nonmodular architecture and for the hardwired modular architecture. (Notice that there is less computational power, that is, number of neurons and connections, in the nonmodular architecture than in the modular architecture.)

We wanted, first of all, to know if a duplication-based modular architecture is just as efficient in outperforming a nonmodular architecture as a hardwired modular architecture. Figure 5 gives the average and peak performance measures for nonmodular architecture and for duplication-based architecture with a duplication rate of 0.04% (i.e., 0.04% of the modules were duplicated per replication). In both conditions the performance level increases until a plateau is reached. However, populations with modules achieve a higher terminal performance level and need less time (fewer generations) to reach it. More precisely, after about a hundred generations of overlapping performance

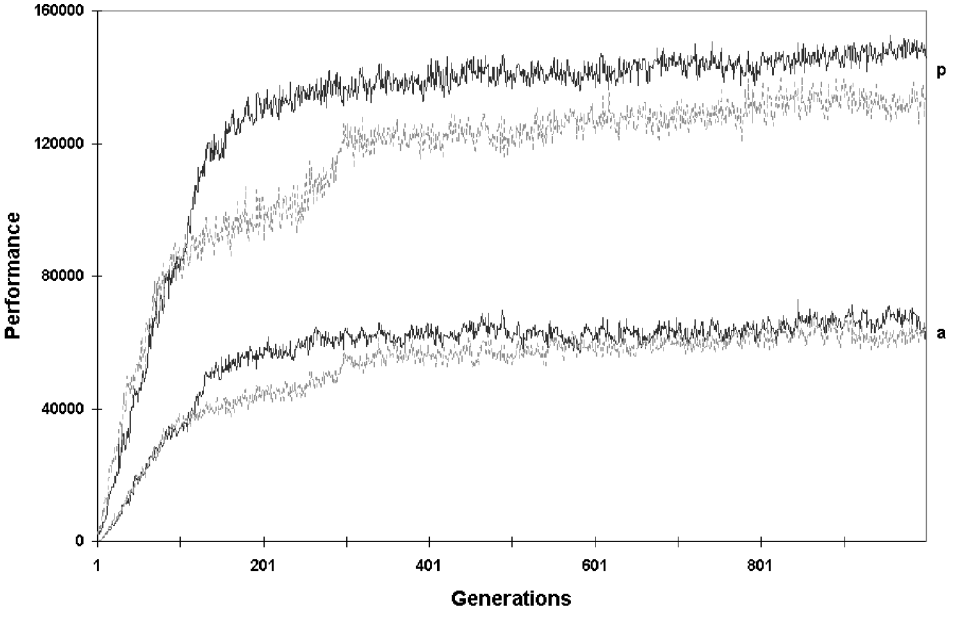

Figure 5. Average and peak performance of populations with **nonmodular architecture** (gray curve) and of populations with **duplication-based modularity** (black curve) with a duplication rate of 0.04%. Average of 10 different runs.

in the two conditions, populations with modules start to outperform populations without modules and this difference is maintained until the end of the evolutionary process, most obviously if we consider the performance of the best individual.

The hypothesis that modularity is implied in accomplishing this result can be indirectly tested by varying the duplication rate in duplication-based modular network simulations. Both average and peak performance decrease linearly with a decreased duplication rate (0.04%, 0.03%, and 0.02%) (results not shown). Figure 6 shows the results obtained with the duplication-based modular architecture for a duplication rate of 0.02% and compares them with those for a nonmodular architecture: The advantage of modular design is lost. This result shows the importance of the interaction between mutation and duplication rate.

If we compare the performance obtained with hardwired modular architecture with that obtained with duplication-based modular architecture, we see that the two populations do not differ in terms of overall performance except that performance growth is slightly slower in the population with duplication-based modules (see Figure 7). This difference can be explained by noting that in the case of duplication-based modular architecture, some generations have to pass before module duplication can take place and duplicated modules can differentiate between each other. Besides the comparison between the two modular architectures in terms of performance level, we were interested in understanding whether there were differences between the two modular architectures at other levels such as behavior (see [2]).

In his analysis of the role of neural modules in hardwired modular architecture, Nolfi [17] observes that it is impossible to find a direct correspondence between neural modules and resulting subbehaviors. In particular, by analyzing some evolved individuals he finds that both competing modules are used in all the phases of different subbehaviors: for instance, when the gripper is empty and the robot has to look for a target, or

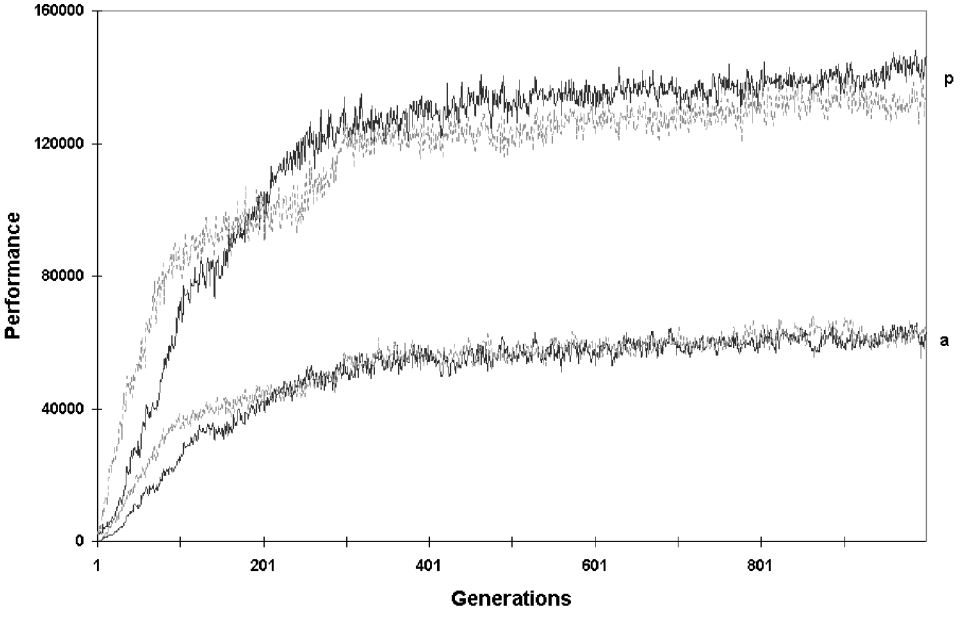

Figure 6. Average and peak performance measures of populations with **nonmodular architecture** (gray curve) and of populations with **duplication-based modular architecture** (black curve) with a duplication rate of 0.02%. Average of 10 different runs.

when the gripper is carrying a target and the robot has to look for a wall; when the robot is approaching a target, or when the robot is approaching a wall; when the robot perceives something and has to disambiguate between walls and targets, or when the robot does not perceive anything (see [17]).

These results demonstrate that although modularity is useful in producing complex behaviors, one does not necessarily find a direct one-to-one correspondence between modules and simpler subbehaviors. This lack of direct one-to-one mapping is not just a matter of chance. By exploiting the interaction between the external environment and the robot's body and internal mechanisms, emergent forms of behavior can evolve which allow simple control systems to produce complex forms of behavior [1, 17].

The fact that there is not a one-to-one correspondence between internal modules and the various subbehaviors, however, does not necessarily imply that all internal modules contribute to all different subbehaviors in the same way. Although each module can contribute to the production of different overall behaviors, a single module or a group of modules may be mainly involved in only one or a few subbehaviors. In other words, modules can have a certain level of specialization. To illustrate this point, let us consider Figure 8. Although the phenotypical entities P1, P2, and P3 all contribute to the production of subbehavior B1, P2 has the main responsibility while P1 and P3 contribute in a less significant way. Similarly P1 and P3 have the main responsibility in producing subbehavior B2.

This kind of specialization may be an advantage, from an evolutionary point of view, if different subbehaviors have different functions (i.e., if a single subbehavior or a group of subbehaviors are primarily responsible for a single adaptive function as shown in Figure 8). Let us consider the case of our garbage-collecting robot. The performance of the robot depends on its ability to accomplish two sub-behaviors: collect objects and release objects outside the arena. These two subbehaviors correspond to two

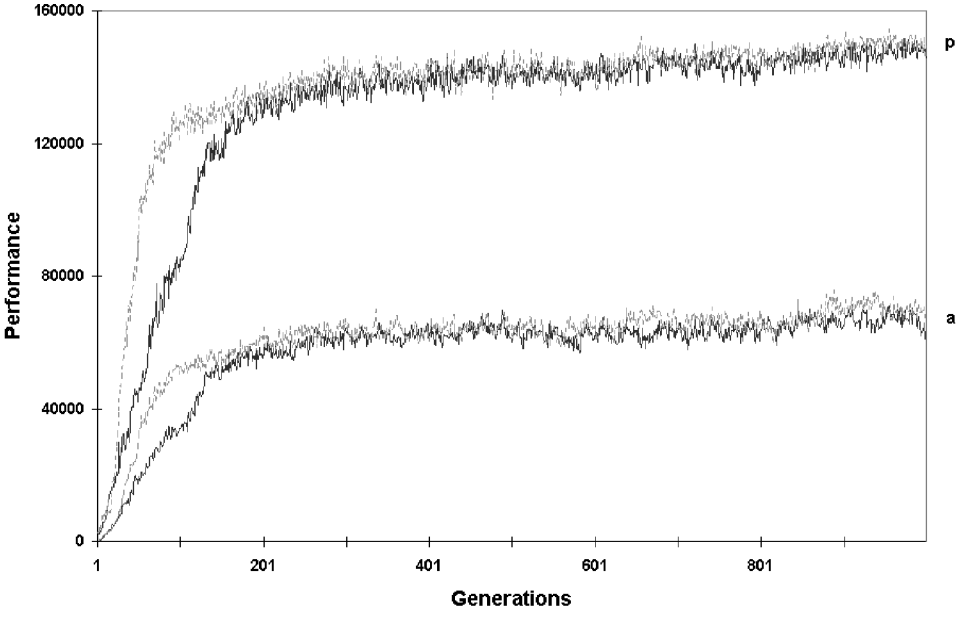

Figure 7. Average (a) and peak (p) performance of population with **hardwired modular architecture** (gray curve) and of population with **duplication-based modular architecture** (black curve) with a duplication rate of 0.04%. Average of 10 different runs.

different functions in that they contribute rather independently to the overall fitness of an individual. If internal structures (e.g., internal modules) are not specialized and each of the two subbehaviors is the result of all modules, changes affecting a single module will tend to affect all subbehaviors. On the other hand, if internal modules are specialized, changes affecting a single module will tend to affect primarily one of the two resulting behaviors. Once the population has converged to a local maximum for most of its characters, genetic operators tend to have negative effects, on the average. This means that changes in genes which affect different characters will produce negative effects on most of these characters. To produce an improvement, a variation of a single gene should positively affect at least a single phenotypical character, but not affect negatively all the other characters that are already optimized. As a consequence, the probability that a change affecting a gene will produce a positive effect is reduced with increased pleiotropy of that gene (i.e., by the number of phenotypical characters affected by that gene). A good mapping therefore should reduce pleiotropic effects among characters serving different functions. Independent functions, in other words, should be coded as independently as possible so that improvements of each function can be realized with minimal interference with other structures serving other functions.

In the case of evolved individuals with hardwired modular architecture, we can identify the level of specialization of internal modules by measuring the statistical relationship (i.e., chi-square value) between single neural modules or a combination of modules and individual subbehaviors (see [3]). The higher the chi-square value, the higher the level of specialization of the neural modules. Table 1 shows the results of such an analysis involving the following subbehaviors: (1) the ability to find and pick up a target while avoiding walls, and (2) the ability to find a wall and correctly release a target while avoiding other targets.

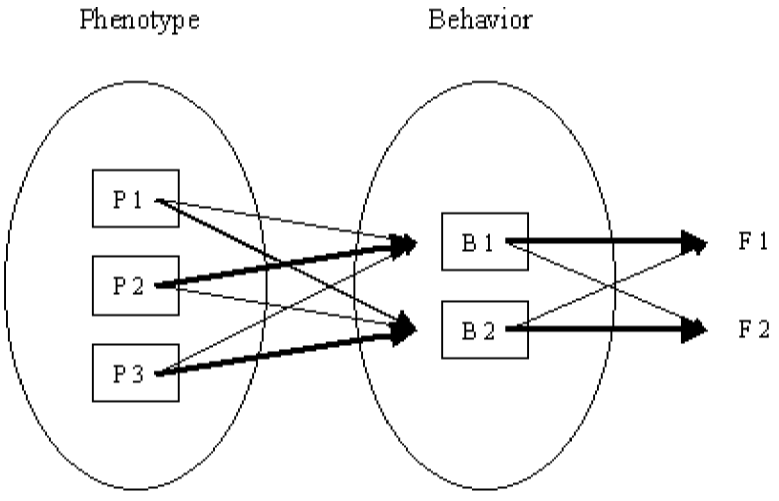

Figure 8. Left: Organization of a system at the level of the phenotype (P1, P2, and P3 represent different subcomponents of the phenotype, e.g. different modules of the control systems). Center: Organization of the corresponding behavior (B1 and B2 represents two different sub-behaviors). Right: Functions of the whole behavior. The thickness of the arrows indicates how important an entity is in determining another entity.

As can be seen in Table 1, there is a very high chi-square value between these subbehaviors and neural modules in only 2 out of 10 runs of the experiment and the relationship is statistically significant in only 5 out of 10 runs. This means that an evolutionary process based on selective reproduction and mutations does not necessarily tend to converge on solutions in which neural modules are specialized but on solutions in which all neural modules contribute to all subbehaviors.

Table 1. Evolved individuals of the experiments with the hardwired modular architecture. Chi-square values obtained by performing a linear regression between the current subbehavior (1 binary value) as the dependent variable and the result of the arbitration between modules (4 binary values) as the independent variable. Data obtained by testing each individual for 500 cycles. Values in bold represent individuals in which a significant correlation was found.

11.135
4.679
425.927
2.747
21.556
439.391
16.647
2.348
29.078
27.081

Table 2. Evolved individuals of the experiments with the duplication-based modular architecture. Chi-square values obtained by performing a linear regression between the current subbehavior (1 binary value) as the dependent variable and the result of the arbitration between modules (4 binary values) as the independent variable. Data obtained by testing each individual for 500 cycles. Values in bold represents individuals in which a significant correlation was found.

368.662
246.374
495.961
218.359
190.511
55.947
55.246
296.993
32.334
321.769

Table 2 shows the same analysis for the simulations with duplication-based modular architecture. A statistically significant relationship between neural modules and the two subbehaviors is observed in 10 out of 10 cases, that is, in all cases we see significant specialization of modules contributing to subbehaviors.

We might conclude that in duplication-based modular architecture, modules appear to be more specialized in the specific subbehaviors mentioned above, while this seems to be less true in hardwired modular architecture.

These results seem to support the model proposed by Hughes ([9]; see also [18]) which assumes that specialization might arise when genes serving multiple functions are duplicated. After gene duplication, in fact, the genes are released from conflicting functional demands and each copy can specialize in one of the different functions of the ancestral gene (for a more detailed discussion see [3]). It should be noted that gene duplication is only one of the factors that may lead to functional specialization (for a discussion see [22]).

These results show that the evolutionary process may lead to a certain level of specialization under certain conditions. It should be noted, once again, that this does not mean that there is a one-to-one correspondence between neural modules and subbehaviors serving different adaptive functions, but only that there is some correlation such that either a single internal entity or a group of internal entities is primarily responsible for a single subbehavior while other entities play a less important role.

Regarding the overall performance we did not observe significant differences after 1000 generations between individuals with the duplication-based architecture and individuals with the hardwired modular architecture. In other words, the functional specialization of internal modules did not result in a larger adaptation capability. This result appears to be in contrast with the assumption that individuals with specialized internal structures have a higher level of evolvability (i.e., a greater probability to obtain an improvement through random variations).

The fact that the two classes of individuals achieve about the same level of performance, however, can be explained by considering that nonspecialized individuals

already achieve close to optimal performances. Therefore, to determine if specialization may lead to higher adaptation levels additional studies should be conducted.

In order to understand the mechanisms by which duplication of structural units favors the specialization of modules we performed a winning lineage phylogenetic analysis [14]. To perform such an analysis, it is necessary to take a best individual of the last generation and trace back all the ancestors of this individual up to and including the first generation. In this way, the entire lineage of the best individual of the last generation can be reconstructed. In our simulation the lineage is constituted by a total of 1000 individuals, one for each generation.

Given our experimental setting, the individual representing the winning lineage in the first generation has a nonmodular architecture and random connection weights. The winning lineage individuals of succeeding generations will progressively change their architecture because of gene duplication and mutation.

We first focused on what happened in the generations immediately following module duplication. Specifically, we wanted to investigate how the occurrence of duplication facilitates the emergence of modular specialization. We expected to find that functional specialization emerges in generations soon after module duplication. We present the results of an analysis of the winning lineage in a typical simulation (i.e., the same simulation we considered in Calabretta et al. [2] for behavioral analysis), in which the results were particularly clear.

The best individual of the last generation (i.e., generation 1000) has all four output modules duplicated and specialized. The first module (i.e., PU) duplicated in generation 30, the second one (i.e., RL) in generation 52, the third one (i.e., RM) in generation 91, and the fourth one (i.e., LM) in generation 329. Below we report what happened after the last duplication, that is, the duplication of the LM module.

After duplication of the LM module takes place, the two copies of the LM module are exactly the same. As a consequence, there is no difference in performance as a function of whether one competing module or the other one controls the robot's left wheel. In our experimental setting, a single randomly chosen module always controls the left wheel, while the other one is nonfunctional. In the next generation, generation 330, only two connection weight values were mutated. Both mutations affect the regulatory part of the module (i.e., the selector; see Table 3), and, as a consequence of these two mutations, we find that the two competing LM modules almost always alternate depending on whether the robot is carrying an object or not. In other words, the two competing modules *seem* to be specialized in terms of the behavior they control. In fact, this specialization is cryptic because the structural part (i.e., the output unit) is exactly the same in the two competing modules. As a demonstration of that, it is sufficient to exchange the mutated connection weight values between the two competing modules. If one does this, exactly the same performance is obtained.

In the individual of the next generation, generation 331, something interesting happens. If we look at the connection weights of this individual we find that it carries one mutation on the structural part of the module. (There is also another mutation affecting the regulatory part of the module, but this mutation is neutral. See Table 3.) This mutation strongly affects the performance of the individual in the sense that individual performance increases as a result of the mutation. We performed two tests: (a) inserting the value of the preceding generation (i.e., 4.04, see Table 3) and (b) inserting the mutated value in the other competing module. In both cases, individual fitness decreases. We conclude that both the regulatory and the structural parts of the module are *really* specialized for the behavioral unit.

It is important to add that in the succeeding generation two other mutations take place in the regulatory part of the module. As a consequence, the specialization of

Table 3. Connection weights for the two competing modules controlling the LM motor in a typical winning lineage with duplication-based modular architecture (generations 329–332). Values which are mutated with respect to the previous generation are highlighted.

	generation	329	330	331	332
structural part of module	1st weight	2.78	2.78	2.78	2.8
		2.78	2.78	2.78	2.8
	2nd weight	2.39	2.39	2.39	2.4
		2.39	2.39	2.39	2.4
	3rd weight	3.1	3.1	3.1	3.1
		3.1	3.1	3.1	3.1
	4th weight	4.04	4.04	4.04	4.04
		4.04	4.04	**4.67**	4.7
	5th weight	2.71	2.71	2.71	2.7
		2.71	2.71	2.71	2.7
	6th weight	6.55	6.55	6.55	6.5
		6.55	6.55	6.55	6.5
	7th weight	−0.75	−0.75	−0.75	−0.7
		−0.75	−0.75	−0.75	−0.7
	bias	−2.78	−2.78	−2.78	−2.8
		−2.78	−2.78	−2.78	−2.8
regulatory part of module	1st weight	7.02	7.02	7.02	7
		7.02	7.02	7.02	7
	2nd weight	−0.2	−0.2	−0.2	−0.2
		−0.2	−0.2	−0.2	−0.2
	3rd weight	5.76	5.76	5.76	5.8
		5.76	**5.92**	5.92	5.9
	4th weight	−4.27	−4.27	**−3.96**	−4
		−4.27	**−9.29**	−9.29	−9.3
	5th weight	2.71	2.71	2.71	2.7
		2.71	2.71	2.71	2.7
	6th weight	4.75	4.75	4.75	4.7
		4.75	4.75	4.75	4.7
	7th weight	−9.06	−9.06	−9.06	−9.1
		−9.06	−9.06	−9.06	**1**
	bias	−8.27	−8.27	−8.27	−8.3
		−8.27	−8.27	−8.27	**−8.6**

the regulatory part of the module is completed. Now the switching between the two competing modules takes place *every time* the robot takes or releases an object.

These results suggest an evolutionary scenario that favors the emergence of functional modularity. If there are two structural units which compete for control over the motor output and if there is a mechanism which can switch between the competing structural units, the evolution of functional specialization is more likely. This situation arises readily if the two units originate by means of duplication.

In fact, a mutation affecting the regulatory part which determines the switching is neutral from a phenotypical point of view. But it can create an evolutionary context favorable to subsequent mutations of the structural part of the module that may result in fitness increases, that is, it can prime the subsequent adaptation.

In evolutionary biology parlance, the mutation affecting the regulatory part of one of the two competing modules can act as an *exaptation*, that is, a trait not built as an adaptation at all, but one allowing later adaptations for some function [6].

4 Discussion

In this paper we have compared three scenarios for the evolution of a somewhat complex behavior in a population of artificial organisms. The behavior of these "organisms" is controlled by simple feed-forward networks of three types: (a) nonmodular neural networks; (b) neural networks with an unchanging hardwired modularity; (c) neural networks in which modularity can evolve as a functional specialization of duplicated structural modules. Modules are portions of a neural network that specialize in controlling the network's output as a function of the particular input.

The results we have obtained can be summarized in the following way. (1) Modular architectures, both hardwired and duplication-based ones, produce better results in terms of both speed of evolution and steady-state final level of the behavioral performance than nonmodular architectures. (2) There is no difference between hardwired and duplication-based modular architectures in terms of overall performance. (3) There is some evidence that modular architectures which originated by duplication tend to favor functional module specialization more than hardwired modular architectures.

A possible explanation of this last result can be found in the different effects of mutations on the hardwired and the duplication-based modular architectures. In both cases mutations can fall either on the regulatory portion of the structural unit (controlling which module determines the network's output in response to some input) or on the structural portion (controlling the type of output a module generates in response to the input).

In the evolved duplication-based architecture, however, a mutation can also simply produce a duplication of a module. Initially the duplication does not produce any adaptive consequences because the two duplicated modules are perfectly identical. Then a new mutation falling on the regulatory portion of the genetic material can give one module control of the output for some inputs and the other module control of the output for other inputs. This second step is also neutral because the two structural units are still identical in their structural parts. Then, a further mutation falling on the structural part can lead to module specialization and, therefore, possibly to adaptive consequences.

This is the sequence we have reconstructed in our simulations. In other words, in the duplication-based architecture the switching between competing modules is initially neutral from a phenotypic point of view but creates favorable conditions for (a) the continuing utilization of the two modules, (b) the selective retention of a favorable mutation falling on the structural part of one module, and (c) a larger probability of module specialization. In contrast, in Nolfi's hardwired architecture this type of

switching (i.e., one that is initially neutral from a phenotypic point of view) is less likely to occur because alternative modules are different from each other from the beginning. This may lead to constraints on the evolutionary accessibility of functionally specialized modules.

5 Conclusions

We conclude that, in our simulations, the duplication of partially adapted modules greatly facilitates evolution of functional specialization. The mechanism identified includes, first, the acquisition of a neutral change in the regulation of the duplicated modules and, second, the adaptation of the modules to the new functional context. This mechanism leads to a co-adaptation between the regulatory and the functional parts of the modules that lock the system into the functionally specialized state. It is thus possible that the prevalence of functional differentiation of body parts in higher organisms is caused by the fact that functional specialization of parts can be an evolutionary absorption state, as long as the function performed by the module contributes to fitness. We found no evidence that functionally specialized systems have inherently better performance or are more evolvable than nonspecialized modular systems in our simulations.

Acknowledgments

The financial support of the Italian National Research Council (National Advisory Committee for Biological and Medical Sciences: one-year sabbatical fellowship to RC and Bilateral Project 1998–2000 to RC and GPW) and of the Yale Institute for Biospheric Studies to GPW is gratefully acknowledged. The authors thank the referees for valuable suggestions on the manuscript. RC would also like to acknowledge useful discussions with the members of GPW's lab at Yale University during weekly meetings and with Andrea Di Ferdinando, Riccardo Galbiati, and the other members of the Research Group on Artificial Life (GRAL) in Rome. This is contribution #59 of the Yale Center for Computational Ecology.

References

1. Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation, 2*, 14–23.
2. Calabretta, R., Nolfi, S., Parisi, D., & Wagner, G. P. (1998a). Emergence of functional modularity in robots. In R. Pfeifer, B. Blumberg, J.-A. Meyer, & S. W. Wilson (Eds.) *From Animals to Animats 5* (pp. 497–504). Cambridge, MA: MIT Press.
3. Calabretta, R., Nolfi, S., Parisi, D., & Wagner, G. P. (1998b). A case study of the evolution of modularity: towards a bridge between evolutionary biology, artificial life, neuro- and cognitive science. In C. Adami, R. Belew, H. Kitano, & C. Taylor (Eds.), *Proceedings of the Sixth International Conference on Artificial Life* (pp. 275–284). Cambridge, MA: MIT Press.
4. Crow, J. F., & Kimura, M. (1970). *An introduction to population genetics theory*. Minneapolis, MN: Burgess Publishing Company.
5. Futuyma, D. J. (1998). *Evolutionary biology*. Sunderland, MA: Sinauer.
6. Gould, S. J., & Vrba, E. S. (1982). Exaptation—A missing term in the science of form. *Paleobiology, 8*(1), 4–15.
7. Gruau, F. (1995). Modular genetic neural networks for 6-legged locomotion. In J.-M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, & D. Snyers (Eds.) *Artificial Evolution, European Conference* (pp. 201–219). Berlin: Springer-Verlag.
8. Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: The University of Michigan Press. (Second edition: Cambridge, MA: MIT Press, 1992).

9. Hughes, A. L. (1994). The evolution of functionally novel proteins after gene duplication. *Proceedings of the Royal Society. Series B, 256*, 119–124.

10. Kimura, M. (1983). *The neutral theory of molecular evolution*. Cambridge, U.K.: Cambridge University Press.

11. Koza, J. R. (1995). Gene duplication to enable genetic programming to concurrently evolve both the architecture and work-performing steps of a computer program. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (pp. 695–717). San Francisco, CA: Morgan Kaufmann.

12. McShea, D. W. (1996). Metazoan complexity and evolution: Is there a trend? *Evolution, 50*, 477–492.

13. Miglino, O., Lund, H. H., & Nolfi, S. (1995). Evolving mobile robots in simulated and real environments. *Artificial Life, 4*, 417–434.

14. Miglino, O., Nolfi, S., & Parisi, D. (1996). Discontinuity in evolution: How different levels of organization imply pre-adaptation. In R. Belew, & M. Mitchell (Eds.), *Adaptive individuals in evolving populations*. Reading, MA: Addison-Wesley.

15. Mondada, F., Franzi, E., & Ienne, P. (1993). Mobile robot miniaturisation: A tool for investigation in control algorithms. In T. Yoshikawa & F. Miyazaki (Eds.), *Proceedings of the Third International Symposium on Experimental Robotics* (pp. 501–513).

16. Müller, G. B., & Wagner, G. P. (1991). Novelty in evolution: Restructuring the concept. *Annual Review of Ecology and Systematics, 22*, 229–256.

17. Nolfi, S. (1997). Using emergent modularity to develop control systems for mobile robots. *Adaptive Behavior, 5*, 343–363.

18. Ohno, S. (1970). *Evolution by gene duplication*. New York: Springer-Verlag.

19. Rotaru-Varga, A. (1999). Modularity in evolved artificial neural networks. In D. Floreano, J.-D. Nicoud, & F. Mondada (Eds.), *Proceedings of the Fifth European Conference on Artificial Life* (pp. 256–260). Berlin: Springer-Verlag.

20. Rumelhart, D., & McClelland, J. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Cambridge, MA: MIT Press.

21. Shpak, M., & Wagner, G. P. (2000). Asymmetry of configuration space induced by unequal crossover: Implications for a mathematical theory of evolutionary innovation. *Artificial Life, 6,* 25–44.

22. Wagner, G. P., & Altenberg, L. (1996). Complex adaptations and the evolution of evolvability. *Evolution, 50*, 967–976.