On the Relationship between Environmental Noise and Modular Hierarchies

John Rieffel
Computational Synthesis Lab, Cornell University, Ithaca NY
rieffel@cornell.edu

Abstract

In the context of developmental representations, hierarchical modular composition is often cited as a requisite for scalable complexity, particularly in the presence of noise and error during development. In earlier work of ours (Rieffel & Pollack 2006), we demonstrated the ability of a modular representation to overcome errors due to noise in an Evolutionary Developmental system. In this paper, using the same model, we present some observations on the relationship between the level of noise in the environment and the structure of the hierarchies which emerge.

Introduction

One outstanding challenge in the open-ended evolution of formation lies in *scalable complexity* - that is, how to build increasingly large, increasingly complex objects in a managed fashion. Crucial to this is the process of hierarchical, modular assembly. In his landmark essay "The Architecture of Complexity", Herbert Simon (Simon 1962) makes the case that increasingly complex forms are nearly impossible to evolve without such hierarchical composition of *stable* modules. In the field of evolutionary developmental systems, our interest is in modular genetic *representations* which in turn lead to corresponding scalable hierarchical assembly of the phenotypic result.

While the value of modularity has been well studied in the Artificial Developmental Systems community (Angeline & Pollack 1994; Hornby 2005; Bongard 2002; Rosca & Ballard 1996; de Jong 2003; Watson 2002) relatively little attention has been paid to the utility of modularity in *stochastic* environments. In earlier work of ours (Rieffel & Pollack 2006) we began to explore this issue. Using a model of modular encapsulation which added subassemblies to the language of represenation only if they produced reliable results in the presence of noise, we were able to demonstrate the system's ability to overcome noise to build increasingly large and complex structures.

That work raised an interesting question: *in an evolutionary modular representation which chooses modules based upon their robustness to noise, what is the effect of varying degrees of environmental noise on the size and shape of the hierarchies which emerge?* In this paper, using our framework from (Rieffel & Pollack 2006), we explore this question in more detail.

Adaptive Modular Representations

From the perspective of evolution, the importance of *representational* modularity lies in its ability to couple functionally related portions of the genotype while simultaneously decoupling them from functionally unrelated portions (Wagner & Altenberg 1996). Changes to a representational module have few side effects in the remaining genome, and changes outside a module have few effects upon the module. Wagner and Altenberg (Wagner & Altenberg 1996) persuasively argue that the evolvability of a system is highly contingent upon its representation's ability to adapt by discovering and incorporating evolutionary modules.

Several models of representational modularity in evolutionary computation exist. Many systems, such as Hornby's L-systems (Hornby 2005) and Bongard's Gene Regulatory Networks (Bongard 2002) feature representations which are implicitly modular.

Of those which provide for the *explicit* encapsulation of modular components, the most common fall under the rubric of Hierarchical Genetic Programming (HGP), where encapsulated modules become new primitives in the language. Koza (Koza 1992) developed Automatically Defined Functions (ADFs), in which sub-functions are allowed to evolve their own function and terminal sets. Angeline expanded upon ADFs with *module acquisition* (MA), which co-evolve a representational genetic "library" of encapsulated primitives which are universally available to evolving programs(Angeline & Pollack 1994). Subsequently, Rosca and Ballard introduced Adaptive Representation through Learning (ARL), which replaced the randomness inherent in modular acquisition in ADFs and MA with a "usefulness" heuristic based upon fitness contribution and activation frequency of subtrees (Rosca & Ballard 1996).

More recently, de Jong co-evolved a representation and its corresponding population of genotypes (de Jong 2003). Candidate modules were chosen by finding the most frequent pair of alleles in the current population and, drawing from Watson's work on symbiotic composition (Watson 2002), were added to the language as primitives only if they their fitness contribution was at least as good as the fitness

contribution of all other possible pairs in a randomly chosen context.

While they vary in their details, each of these models of modular encapsulation involve incorporating *genotypic* sequences, thereby protecting them from the deleterious effects of mutation and crossover, and then adding them to the language of representation. As such, encapsulated modules are simply shorthand for the genetic sequence they represent - one can be substituted for the other without consequence. Below, we will motivate a system of encapsulation which, by contrast, involves incorporating *phenotypic* results into the language or representation.

Addressing Context Sensitivity

Because of their prescriptive nature, developmental representations display a measure of context dependency: the same sequence of operations can have vastly different results depending on where in the process it occurs. Consider a developmental representation as a *recipe*. The set of instructions which produce egg whites in a souffle recipe (separate yolks and whites, place whites in a bowl, whip into soft peaks) would produce a mess (if anything at all) if they occurred later in the recipe or, for that matter, in an omelet recipe.

To make matters worse, in developmental systems such as the one we describe below, a contiguous portion of the phenotype which we might want to modularize may have been produced by disjoint portions of the genotype. Similarly, a contiguous portion of the genotype may produce disjoint phenotypic results.

These factors can therefore stymie adaptive models such as HGP, which generate modules by extracting and compressing favorable genetic sequences. A genetic sequence which produces a favorable trait in one context will not necessarily preserve that result when transferred to another context. Furthermore, favorable phenotypic traits may not be attributable to modularizable portions of the genotype, and modularizable portions of the genotype may not produce useful phenotypic modules.

The challenge of modular acquisition in developmental representations, then, lies in preserving not the syntax, but rather the *meaning* of a desired phenotypic result. *Endosymbiosis*, the encapsulation of an entire organism by its host, is the model which we propose for this.

In our model of endosymbiotic encapsulation (See Figure 1), complete organisms, not just specific portions of their phenotype, are used to form modules. As such, endosymbionts become *precompiled* phenotypes, and join the set of primitives available to the representation. To continue the metaphor of the recipe, endosymbiosis is analogous to the parallel creation of a *sous chef*, who specializes in producing that one particular higher-order ingredient, such as stiffened egg whites.

When referenced during the course of development, it is the phenotypic result – the complete endosymbiont – rather than the genetic sequence responsible for creating the endosymbiont, that is used by the developmental process. In this manner, the meaning, rather than merely the syntax, of a module is preserved, and can be applied consistently across contexts.

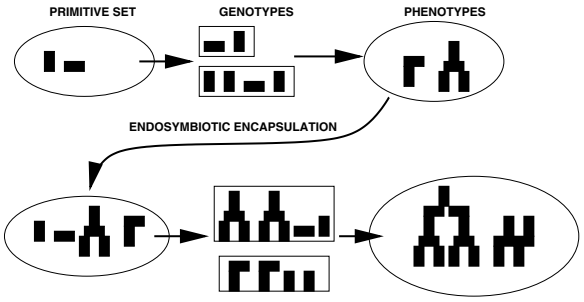

Figure 1: In the endosymbiotic model of module acquisition, only complete phenotypes, rather than genetic samples, are added to the set of primitives.

Modular Evolutionary Fabrication

Our Artificial Developmental System takes the form of *Evolutionary Fabrication*: the prescriptive genotype is a linear set of instructions to an external assembly mechanism. In this case the mechanism is a LOGO-like turtle, capable of movement in the X-Z plane, and of depositing objects in the environment. When strung into a sequence, commands to the turtle (move, rotate, put object, take object) form an *assembly plan*. Commands which would cause the turtle to move outside the target area, or place a brick where a brick already exists, are ignored.

The "put" command takes as an argument a unique identifier corresponding to an object in the library of encapsulated modules. Initially, the only objects available to the "put" command are primitive $2 \times 1 \times 1$ bricks. As new modules are encapsulated, they are inserted into the library as new objects and can be referenced by the "put" command (for instance put(brick) or put(module15).) As the modular library grows, the mutation operator selects from any of the modules currently available.

Assembly occurs within a realistic physics environment based upon the Open Dynamics Engine (ODE) [1] the widely used open-source physics engine, which provides high-performance simulations of 3D rigid body dynamics.

Fitness Evaluation

In order to generate larger structures and correspondingly large modular hierarchies than in (Rieffel & Pollack 2006), the environment is now four times larger than the original, and a "leafy" fitness function (Figure 2 is used, which measures the number of horizontal surfaces crossed during vertical slices of the structure. The optimal structure is therefore a series of tightly packed horizontal branches.

The fitness objectives are listed below:

- Length Of Assembly Plan (minimizing)
- Mass of Objects in Environment (minimizing)
- Leafiness (maximizing)

[1] `www.ode.org`

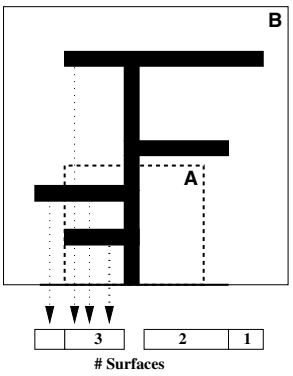

Figure 2: Illustration of Larger Environment and "Leafy" Fitness. The larger environment (Box B) is four times as large as the original (Box A). "Leafiness" is measured as the number of surfaces encountered along a vertical slice.

Noise Model Noise is injected into the system using a "shaky hand" model. When instructed to place an object, the turtle puts anywhere within some range $\triangle x$ around its current position, with uniform probability. Noise settings were given as percentages of a brick's width.

To measure the effect of developmental noise on evaluated genotypes, each assembly plan was interpreted 10 times, and average values over each objective were used for selection.

Module Discovery and Rejection

The key feature of this new framework is the ability to discover new modules *during the process of evolution* and then add them to the set of primitives available to evolving assembly plans, as shown in the schematic of Figure 1.

Every 10 generations, unused modules are removed from the library and new candidate modules are discovered and added to the library. The criteria for selection are as follows:

- **Fitness:** the selected module must be a member of the current pareto front. That is to say it must have some intrinsic fitness beyond these other criteria.
- **Unity:** the selected module must only consist of a single piece. Obviously if an assembly plan results in a structure with two distinct pieces, then it cannot be usefully encapsulated as a context-independent building block.
- **Reliability:** the assembly plan which produces the candidate module must have sufficiently low variance in fitness.

If a phenotype matches the criteria then it is added to the module library as a whole object, becoming available as an argument to the "put" instruction.

The *evolutionary viability* of the modules is determined by their reference count in the population of evolving assembly plans. Care must be taken in order to prevent spurious modules, those which are referenced by assembly plans but have no consequence (for instance those that are ignored because their placement would intersect with an existing object) from causing "bloat" in the set of primitives. Candidate modules are therefore tested for *necessity*. Each member of the pareto front which contains a reference to a module is built twice - once with and once without the referent module. If the results are the same then the module reference is permanently removed from the assembly plan. This is repeated for each module to which an assembly plan refers.

Once those unnecessary module references have been removed from the population of assembly plans, the reference count of each module in the library is measured. Whenever a module's reference count in the evolving population drops to zero it is deemed irrelevant, and removed from the object library.

Measuring Modularity

We can now investigate how varying levels of noise affect the *type* of hierarchies which emerge. Given the assembly plans for a structure and its corresponding modules we can generate a directed graph of the hierarchical assembly process by creating an adjacency matrix. (In fact, all of the graphs of hierarchical assembly shown in this work were produced automatically in this manner). We can then measure properties of the graph, such as the depth of the hierarchy, the number of modules (nodes), and the amount of reuse (edges). Using these measurements we can compare relative values between hierarchies evolved at differing noise levels.

Consider for instance the the matrix below, corresponding to the graph shown in Figure 3. The value at (i, j) corresponds to the number of times $module_i$ is used in $module_j$. In this hierarchy there are four nodes, four edges, and the depth is four.

	A	B	C	D
A	0	0	0	0
B	1	0	0	0
C	0	1	0	0
D	1	0	1	0

Analysis

Using these metrics we can compare the properties of the hierarchies which emerged in the "leafy" fitness function at 0.1% noise with those that arose at 5.0% noise.

Figure 4 compares the progress of evolution between the two fitness regimes. As can be seen, even with noise at 5% of a brick width, modular evolution is still able to progress to produce highly fit individuals – in fact, at its best it does better than the average 1.0% run, and better than the worst 0.1% run. This leads us to believe that higher noise has a *retarding* effect upon progress, in the sense that it takes evolution longer to find reliable modules, but less so a *limiting effect*: by allowing the 5% run to continue we see that it catches up to the 1.0% run within the next 100 generations.

Figure 1 contains sample assembly hierarchies evolved at the 0.1% noise level, and figure 2 contains those evolved at the 5% noise level.

In each survey we measure the properties of all structures with fitness between 1200 and 1800. Taking into account the retarding effect that increased noise level has on the progress of evolution, in order to compare structures of equivalent fitnesses the 0.1% noise hierarchies were sampled at generation 280, and the 5.0% hierarchies were sampled at generation 500. The survey consists of 109 individuals across

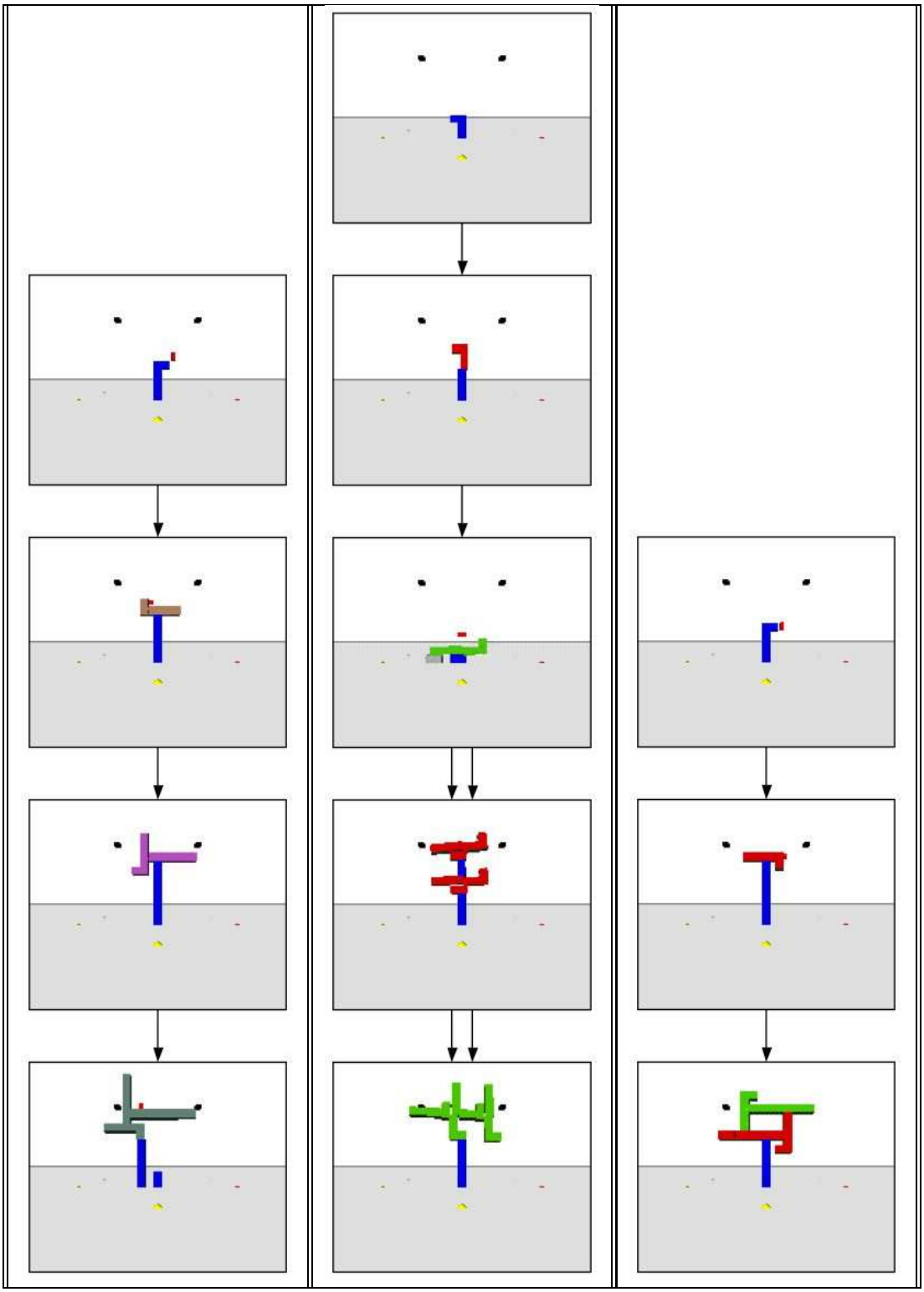

Table 1: Example Hierarchical Assemblies of Modules at 0.1% noise

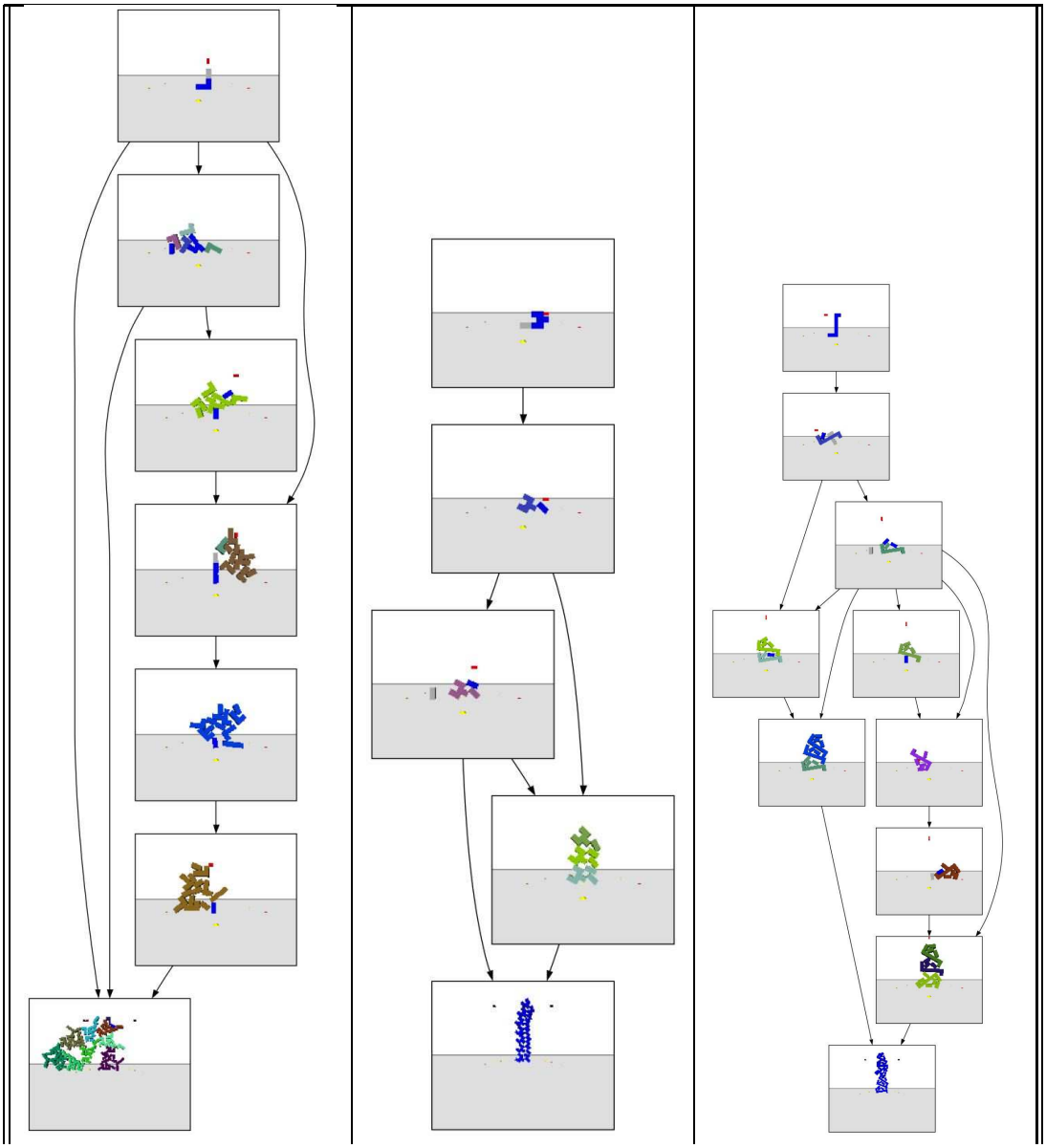

Table 2: Examples of hierarchical assembly of larger structures at 5% noise. Note the increased size and complexity of the assemblies.

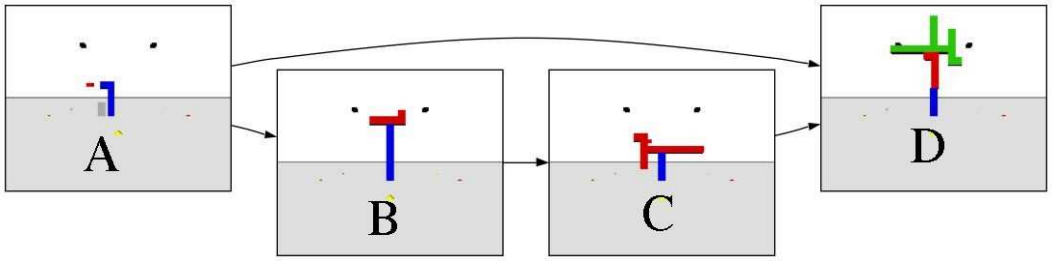

Figure 3: A robust hierarchical modular assembly from (Rieffel & Pollack 2006)

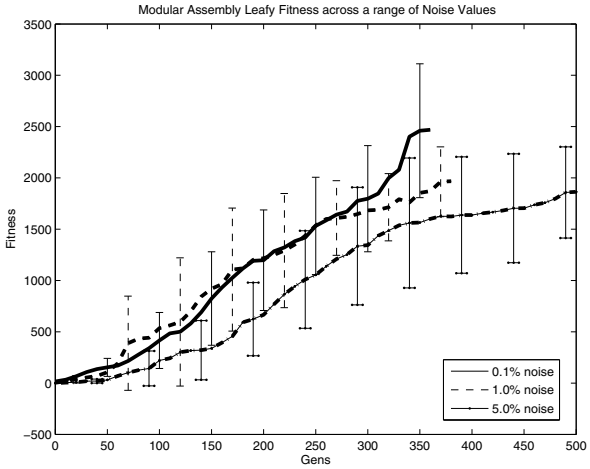

Figure 4: Evolutionary Fitness of the "Leafy" Fitness Function across a wider range of noise values. Higher noise has more of a *retarding* effect than it does a *limiting* effect.

10 runs at 0.1% noise and 212 individuals across 11 runs at 5.0% noise. In the bar graphs below, measurements for each hierarchy were normalized by the overall fitness of the hierarchy.

A two-sample T-test shows that the distributions of fitnesses between the two surveys are equivalent ($p = 0.778$), as are the normalized depths ($p = 0.9724$) and total number of edges ($p = 0.5743$). The distributions of *nodes* between the two samples, however is quite distinct ($p = 3.771e^{-8}$) - with the samples from the noisier environment having significantly fewer nodes. These data are presented in the bar graphs in Figures 6 and 7.

This above analysis shows a key quantitative difference between the hierarchies that emerge at the different noise levels. While the fitness, depths and edge counts in both sets of hierarchies remain comparable, those hierarchies which emerge in higher noise have fewer nodes overall. The number of nodes in the hierarchy correspond to the number of *distinct* modules that are found by evolution.

Discussion

On one hand, it makes sense that the hierarchies that emerged in the higher noise regime have fewer modules: as noise levels increase, the number of reliably buildable subassemblies decreases. And yet, while the number of nodes is different, the total number of edges between remains the same. In other words, the relative number of edges *per module* increases as noise increases.

This quantitative difference between properties of the emergent hierarchies at the two different noise levels leads us to conclude that *increased levels of noise lead to increased levels of modular reuse* in our evolved hierarchical assemblies (Figure 5).

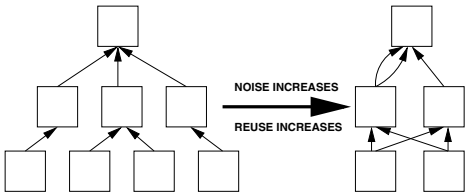

Figure 5: As the noise in the developmental system increases, the height of the trees remains essentially the same, as does the total number of edges. The number of modules, however, decreases, meaning that there are more edges *per module*. In other words, *increased levels of noise lead to increased levels of modular reuse* in our evolved hierarchical assemblies.

While these results are preliminary, and of course specific to our particular model of modular acquisition as well as our evolutionary task, we feel that this paper takes an important step in introducing methods by which the effect of environmental noise on developmental modularity can be measured. In our particular setup, when faced with a limited number of available modules due to increased noise, our evolutionary fabrication system, by using each available module more often, is nonetheless able to build structures equally fit as those produced under less noise. Our central analysis, that as environmental noise increases, modular reuse increases, can therefore be summarized with the phrase "where there's a will there's a way".

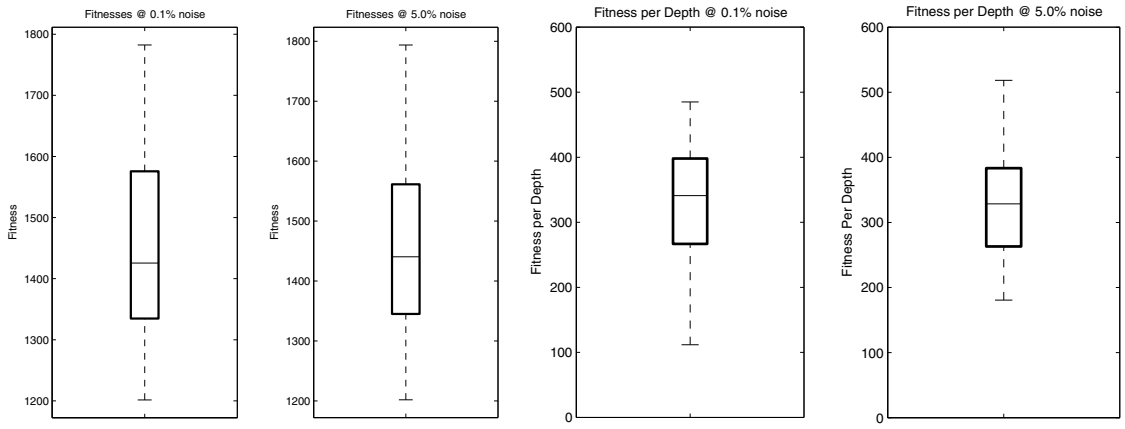

Figure 6: Comparing the effects on noise on the shape of the hierarchical assemblies which emerge. Increased noise does not seem to affect either the average fitness or the average depth of evolved hierarchies. The distribution of fitnesses sampled at 0.1% noise are indistinct from those sampled at 5.0% noise ($p = 0.7778$). The distributions of normalized depth between the two samples are also similar ($p = 0.9724$).

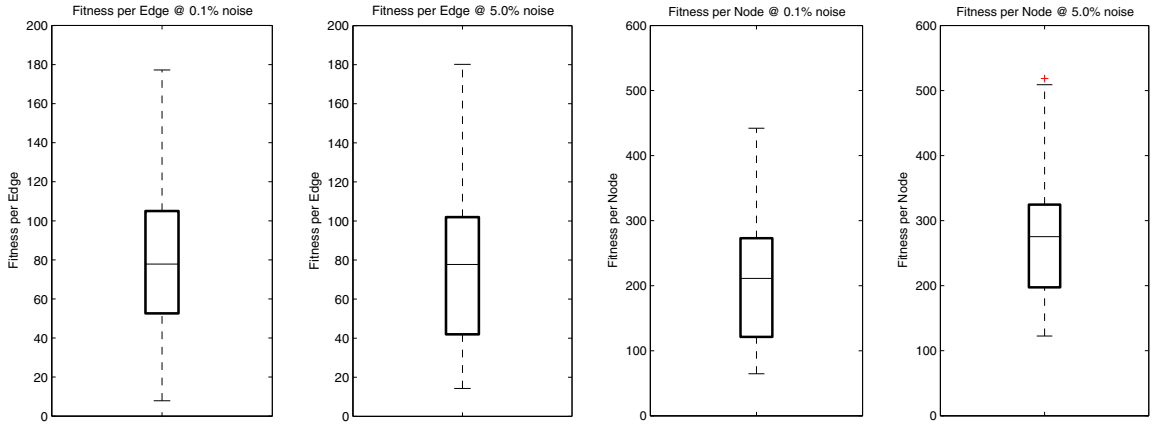

Figure 7: Comparing the effects on noise on the shape of the hierarchical assemblies which emerge. As noise increases, the amount of modular *reuse* increases. While the distribution of normalized edges in each sample is comparable ($p = 0.5743$), those hierarchies evolved in the higher noise regime have significantly fewer nodes ($p = 3.771e^{-8}$).

References

Angeline, P. J., and Pollack, J. B. 1994. Coevolving high-level representations. In Langton, C. G., ed., *Artificial Life III*, volume XVII, 55–71. Santa Fe, New Mexico: Addison-Wesley.

Bongard, J. C. 2002. Evolving modular genetic regulatory networks. In *Proceedings of the 2002 IEEE Conference on Evolutionary Computation*.

de Jong, E. D. 2003. Representation development from pareto-coevolution. In *Proceedings of the 2003 Genetic and Evolutionary Computation Conference*, 262–273.

Hornby, G. S. 2005. Measuring, enabling and comparing modularity, regularity and hierarchy in evolutionary design. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, 1729–1736. New York, NY, USA: ACM Press.

Koza, J. R. 1992. *Genetic Programming: on the Programming of Computers by Means of Natural Selection*. MIT Press: Cambridge, MA.

Rieffel, J., and Pollack, J. 2006. An endosymbiotic model for modular acquisition in stochastic developmental systems. In et al, L. R., ed., *Proceedings of the 10th International Conference on the Simulation and Synthesis of Living Systems (ALIFE X)*. MIT Press.

Rosca, J. P., and Ballard, D. H. 1996. Discovery of subroutines in genetic programming. In Angeline, P. J., and Kinnear, Jr., K. E., eds., *Advances in Genetic Programming 2*. Cambridge, MA, USA: MIT Press. 177–202.

Simon, H. A. 1962. The architecture of complexity. *Proceedings of the American Philosophical Society* 106(6):467–482.

Wagner, G. P., and Altenberg, L. 1996. Complex adaptations and the evolution of evolvability. *Evolution*.

Watson, R. 2002. *Compositional Evolution: Interdisciplinary Investigations in Evolvability, Modularity, and Symbiosis.* Ph.D. Dissertation, Brandeis University, Dept. of Computer Science, Boston, MA, USA.