Approaches to Dynamic Team Sizes

G. S. Nitschke
Department of Computer Science
University of Cape Town
Cape Town, South Africa
Email: gnitschke@cs.uct.ac.za

S. M. Tolkamp
Department of Computer Science
Vrije Universiteit Amsterdam
Amsterdam, Netherlands
Email: matthijstolkamp@gmail.com

***Abstract*—This paper elucidates the impact of comparative mechanisms for dynamically adapting team sizes in simulated robot teams that must accomplish a collective construction task. The first approach adapts team size using genotypic diversity of team behaviors in a population of behaviors. The second approach adapts team size as a function of behavioral success (degree of task accomplishment). Task complexity in the collective construction task is equated with the degree of cooperation required between robots for task accomplishment. Task complexity, and hence the most suitable team size, is not known *a priori*. Results indicate that adapting team size based on genotype diversity is beneficial for collective construction when the task requires a low degree of cooperation. However, when a relatively high degree of cooperation is required for task accomplishment, then dynamically adapting team size according to genotype diversity or behavioral efficacy yields comparable results.**

***Index Terms*—Neuro-Evolution, Collective Construction, Team Size Adaptation.**

I. Introduction

In multi-robot systems [1], it is often beneficial to dynamically adapt team size so as a suitable number of robots are assigned to the task [2], [3]. In cooperative multi-robot tasks, if team sizes are too small then sub-optimal task performance or failure often results [4]. Similarly, if a team is too large then sub-optimal task performance often results from physical interference between robots as they try to accomplish cooperative tasks [2]. Dynamic adaptation of team size is especially important in collective behavior tasks where a required degree of cooperation, and hence the optimal team size is not known *a priori* [3], [5]. Dynamic team size refers to adapting the number of robots (as well as behavior), during an *Evolutionary Robotics* [6] (ER) simulation. An end goal of ER is to transfer collective behaviors (and team sizes) evolved in simulation to a counter-part team of physical robots that accomplishes the same task in reality (outside this paper's scope).

Dynamic team size adaptation, in an ER context, contrasts with more conventional approaches that attempt to predetermine team size given a measure of task complexity [7], [8]. One approach of ER for evolving teams to solve collective behavior tasks, is to adapt team sizes as a function of task complexity. However, the most appropriate (task independent) method for adapting team sizes remains unclear. For example, the benefits of adapting team size as a function of only genotype diversity (where a genotype encodes a robot's behavior) versus task complexity has received little research attention.

In this study, the mechanism to adapt team size according to genotype diversity works within the *Neuro-evolution for Augmenting Topologies* (NEAT) method [9]. This approach is compared to a mechanism that adapts team size using the effectiveness of robot behaviors. This second mechanism works within the *Collective Neuro-Evolution* version 2 (CONE-2) method [10]. CONE (with which CONE-2 shares many similarities) has been applied to various collective behavior tasks [11], [12]. However, with the exception of NEAT extensions such as HyperNEAT, applied to specific multi-agent tasks [13], NEAT has not been applied to solve collective behavior tasks. NEAT was selected as a team behavior evolution method, given its *speciation* mechanism that maintains genotype diversity and protects controller innovation [14].

The efficacy of CONE-2 and NEAT for evolving team behaviors (and suitable team sizes) was tested in a collective construction task which required varying degrees of cooperative behavior. Task performance comparisons between CONE-2 and NEAT evolved teams were in terms of the number of blocks connected together in a specific order to form a larger structure. Varying degrees of cooperation were equated with the number of robots required to connect blocks. Either one (*no cooperation*), two (*low cooperation*), or three (*high cooperation*) robots were required to build connections.

A. Research Goal

To demonstrate two comparative methods for dynamically adapting team size that are beneficial for collective construction task performance. The task requires varying degrees of cooperation between robots (unknown to the team *a priori*). This research goal was formulated given previous ER research results [11], [12] elucidating the difficulty of predetermining suitable team sizes for collective (cooperative) behavior tasks.

B. Hypothesis

To test if adapting team size as function of genotype diversity is comparably effective to adapting team size according to behavioral success. For this study, a genotype encodes a robot behavior, that the NEAT *Neuro-Evolution* (NE) method evolves. This hypothesis was tested via comparing collective construction task performance of the fittest NEAT versus CONE-2 evolved teams, where both methods adapt team size and behavior. This hypothesis was formulated given the success of behaviorally heterogenous teams in cooperative tasks [8], [15]. In this study, genotype (controller encoding) diversity represents behavioral diversity at the team level.

C. Motivation

Previous research has focused on adapting team sizes as a function of task complexity (that is, controller output success) [7], [8]. However, to date, the most appropriate methods for dynamically adapting team sizes in any given cooperative task remains unclear. In this case, a comparison and analysis of *behavioral encoding* versus *behavioral success* mechanisms for adapting team size in cooperative tasks is tested. This comparison is conducted in context of NEAT and CONE-2. Both NE methods have been successfully applied as NE controller design methods in various tasks [16], [10].

NEAT is applied to evolve robot controllers and adapt team size using its *speciation* mechanism [16] (section III). In NEAT, the evolution of *N* controllers is represented by the fittest controllers selected from *N* species. Hence, NEAT controllers (genotypes) in evolving populations (species) only become as complex as the task requires, where team size equals the number of species being evolved.

CONE-2 is applied to evolve controllers and adapt team size via adding a new genotype population (from which individual robot controllers are evolved) whenever a cooperative task cannot be accomplished by the current number of robots. In CONE-2, the evolution of *n* controllers is represented by *n* genotype populations (section II). This results in a new robot being added to the environment. The motivation for this approach is that a CONE-2 team only becomes as large as required (controller complexity is fixed). Thus, CONE-2 team size is regulated by the degree of cooperation required to accomplish the collective construction task.

II. CONE-2: Collective Neuro-Evolution 2

CONE-2 is a cooperative co-evolution NE method that extends CONE [11]. CONE-2's contribution is that it increases the number of populations (from which controllers are evolved) as a function of task complexity. CONE-2 increases the number of controllers until a team size suitable for task accomplishment is found [10]. One controller (fully connected fixed topology feed-forward ANN with one hidden layer) is evolved from each population. This research uses a simplified version of CONE-2 where genotypes directly encode complete controllers. That is, each genotype is a vector of floating point values that represents all input to hidden layer and hidden layer to output connection weights in an ANN.

CONE-2 starts with *one population* (P_0), consisting of *u* genotypes (controllers). Each genotype encodes 250 connection weights (220 input to hidden layer weights plus 30 hidden layer to output connection weights, as depicted in figure 3, left). Each genotype's gene is initialized to a random value in the range: [-1.0, 1.0]. The controller evolution process is the same as *Conventional Neuro-Evolution* (CNE) [17].

A. Team Representation and Size Adaptation:

In CONE-2, one population represents each robot in the environment. That is, each robot's controller is evolved from a separate population. For example, a team size of *N*=3 will use three populations to evolve three different controllers. Thus, CONE-2 evolved teams are *heterogenous*. *N* populations (where, $N \geq 2$) of controllers are cooperatively co-adapted based on how well a task is cooperatively solved by controllers selected from each population. An example of CONE-2 using three controllers (populations), and adding a new population is presented in figure 1. The process for evaluating controllers evolved from *N* populations is the same as used for Multi-Agent CNE described by Potter [18] and Nitschke [19].

CONE-2 team size adaptation is depicted in figure 1 (right).

1) *Adding Population P_{n+1} ($n \geq 0$):* If one robot (ANN_i) or two robots (ANN_i and ANN_j) are gripping a block in the *construction zone*, but cannot connect the block to others[1], then a *help signal* is emitted. If there is a second (in the case of ANN_i only) or third (in the case of ANN_i and ANN_j) robot in the environment, then ANN_i, or ANN_i and ANN_j wait *Z* iterations (table III). If another robot does not grip the block in this time, the current generation is stopped and a new population P_{n+1} is created. The next generation then starts, and a new controller ANN_k is evolved from population P_{n+1}.
2) *Population Initialization:* P_{n+1} is initialized with *u* genotypes. For each new population (P_{n+1}), *u* is same as for P_n. To encourage P_{n+1} to evolve a controller from a beneficial part of the solution space, genotypes of P_{n+1} are initialized based on one of the existing populations. This is P_m, from which ANN_m is evolved, where ANN_m is the controller (robot) that emitted the *help signal*. Each genotype in P_{n+1} is initialized with the genotypes of an existing population. Burst mutation with a *Cauchy distribution* [20] is applied to each gene of each genotype in P_{n+1} with probability *p* (table III). This P_{n+1} initialization procedure ensures that the new controller ANN_{n+1} is not too dissimilar to the current team of *N* controllers. Also, the time taken for ANN_{n+1} to evolve a beneficial behavior is minimized since ANN_{n+1} is based on an already functional controller.

B. CONE-2 Genetic Operators

- *Recombination:* Only occurs within populations (for each population). After all genotypes (in all populations) have been assigned a fitness and ranked, genotypes are recombined. For a given population, each genotype in the population's *elite portion* (table III) is systematically selected and paired with another genotype (randomly selected from the elite portion). Three-point crossover [21] is applied to each parent pair. Enough child genotypes are produced to replace the current population. This procedure is repeated for every population.
- *Mutation:* Burst mutation [20] is then applied to each genotype's gene with probability *p* (table III).

[1] At most three robots are needed to connect one block to another (table I).

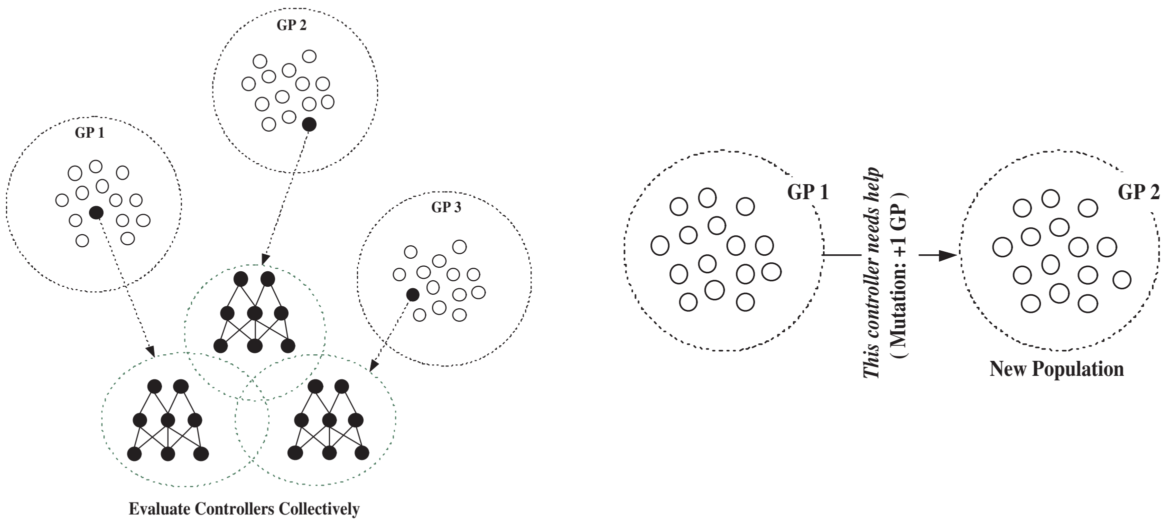


Fig. 1. *CONE-2 Example (Left):* Three ANN controllers are derived from three *Genotype Populations* (GP) and evaluated collectively. Recombination only occurs *within* populations. *Team Size Adaptation (Right):* Adding a genotype (controller) population (GP) and thus a new robot to the environment.

III. NEAT: Neuro-Evolution of Augmented Topologies

NEAT is a competitive co-evolution NE method that uses mechanisms for *historical gene marking*, *speciation*, and *complexification* [9], [14], [16] in its adaptation process.

Complexification is the incremental growth from minimal ANN controller topology. NEAT begins with one homogenous population of simple controllers (no hidden nodes) and adapts connection weights and topology as a function of task complexity. NEAT biases the search towards minimal dimensional spaces and only increases search space dimensionality (adding controller structure) if the task requires it.

Speciation in NEAT calculates if two controllers will be in the same or a new species (according to a genotype compatibility threshold) after controllers have been recombined and mutated every generation. Speciating the population means controllers will only compete within their given species. This protects new innovations in controller topology adaptation.

Historical gene markings allow NEAT to add new structure and recombine controllers with differing topologies, since gene markings are evidence of controller homology.

A. Team Representation and Size Adaptation:

The current fittest controller is selected from each of N species. NEAT evolves teams of N robots from N populations (species). Hence, NEAT teams are heterogenous, since each species evolves a different controller topology.

NEAT uses its speciation [16] mechanism to adapt team size. When a new species is added to NEAT's genotype population (resulting from sufficient genotype diversity), then a new robot (controller) is added to the environment.

B. NEAT: Genetic Operators

- *Mutation:* NEAT mutation adapts both connection weights and controller topology. To adapt controller weights, this application of NEAT uses burst mutation [20] to change each gene in each genotype with a given probability (table III). NEAT controller topology adaptation is the basis of complexification and works via adding genes to genotypes. New genes are new connections or controller nodes represented by a mutated genotype. These mutations are simultaneously applied to each genotype with a given probability. Added connection weights connect two previously unconnected nodes in a mutated controller. When a new node c is added, the existing connection between two existing nodes a and b is disabled. A new connection between nodes a and c (weight value = 1.0) is initialized. A second connection is initialized between node c and b, with the same weight of the previously connected nodes a and b. When a new gene is added, the new gene is assigned an incremented *global innovation number*.
- *Recombination:* Since NEAT tracks the historical origin of all genes using innovation numbers, this means that only homologous genes in two given controllers will be recombined. That is, NEAT only recombines genotypes (controllers) with ancestral genes in common. Matching genes are randomly selected for child genotypes. Disjoint and excess genes are inherited from the fitter parent, or at random in the case of equal fitness.

IV. Collective Construction Task and Experiments

The collective construction task requires a robot team to gather blocks (of type A, B, or C) and cooperatively build a pre-specified structure from the blocks in a *construction zone*. Task complexity is equated with the degree of cooperation (number of robots) required to connect blocks. *Construction rules* (table II) dictate how different block types connect together according to a *construction schema* (table I). Construction rules regulate the difficulty of the construction process via requiring varying degrees of cooperation. For example,

TABLE I
COLLECTIVE CONSTRUCTION TASK: BLOCK TYPE DISTRIBUTION FOR 10 CONSTRUCTION SCHEMAS.

Block Type Distribution and Construction Schema					
Schema Number	A Blocks	B Blocks	C Blocks	Construction Schema	Robots Required
1	5	5	0	A B A B A B A B A B	1
2	4	5	1	A B A B A B A B C B	2
3	3	5	2	A B A B A B C B C B	2
4	2	5	3	A B A B C B C B C B	2
5	1	5	4	A B C B C B C B C B	2
6	0	5	5	C B C B C B C B C B	2
7	0	6	4	C B C B C B C B B B	3
8	0	7	3	C B C B C B B B B B	3
9	0	8	2	C B C B B B B B B B	3
10	0	9	1	C B B B B B B B B B	3

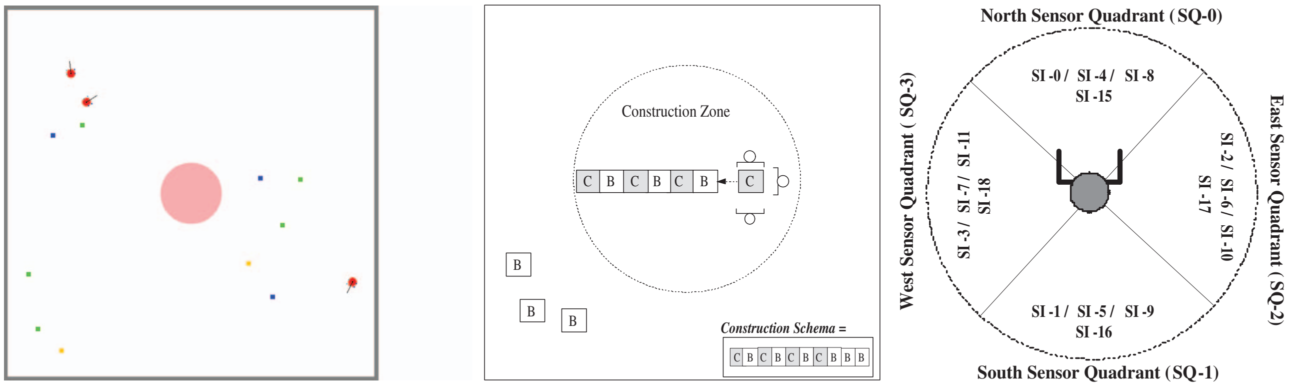


Fig. 2. *Example simulation environment (Left):* With three robots, five *type A* blocks (green), three *type B* blocks (blue), and two *type C* blocks (gold). *Construction schema example (center):* Six of 10 blocks are connected in the correct sequence (given by construction schema 7: table I). Three robots are cooperatively connecting a *type C* block to the partially built object. *Robot (Right):* Visualization of the robot's sensor quadrants, gripper range and sensor range. Robot sensor quadrants: SQ-1, SQ-2, SQ-3, SQ-4. Block demand sensors: SI-12, SI-13, SI-14 are not depicted since they work in all four quadrants.

the construction schema in figure 2 (center) mandates that the third and second last blocks in the construction sequence are type B blocks. Making a type B to type B block connection requires three robots to cooperate (table II). In this collective construction task, *low* and *high level cooperation* refers to two and three robots simultaneously gripping and pushing a block to connect to another block, respectively. Team fitness is the number of blocks connected during its lifetime.

Experiments test *n* (initially *n* = 1), robots in a bounded two dimensional continuous environment containing a distribution of type *A*, *B*, and *C* blocks (figure 2, left). The environment also contains a *construction zone*, where *N* gathered blocks are delivered and connected into a single object. Figure 2 (left, center, right) present an example of the simulation environment, a structure being built according to a construction schema, and a depiction of the robot's sensor quadrants and gripper range, respectively.

A. Fitness Evaluation

To drive the selection process of NEAT and CONE-2, the fitness function accounted for the distance and type of blocks moved by robots, even though the ultimate evaluation of team behavior was the number of blocks connected.

A robot's fitness was calculated based on the time taken (*T*) for it to move block *i* of a given type (BT_i) from an initial position in the environment, and to connect it to other blocks in the construction zone (V_η in equation 1).

TABLE II
BLOCK CONNECTION CONSTRUCTION RULES

Construction Rules			
Construction Schema	A Connects:	B Connects:	B Connects:
[1, 10] (table I)	B	C	B
Robots Required	1	2	3

$$V_\eta = \frac{1}{T} * BT_i \tag{1}$$

Block types *A*, *B*, and *C* yield different fitness values (table III) for being moved and connected in the construction zone. The varying fitness rewards for different block types reflect the degree of difficulty (cooperation) to connect given block types. If *n* robots cooperatively connected a block, then each of the *n* received the same fitness (V_η). For both CONE-2 and NEAT, fitness was assigned to each genotype representing each of the *n* robot controllers.

B. Simulation

An experiment applies CONE-2 or NEAT to evolve team behavior for 500 generations. A generation comprises three

TABLE III
SIMULATION AND NEURO-EVOLUTION PARAMETERS: COLLECTIVE CONSTRUCTION TASK.

Simulation and Neuro-Evolution Parameters	
Simulation runs (1 experiment) / Initial number of robots (Genotype populations)	10 (CONE-2 / NEAT) / 1 (CONE-2 / NEAT)
Maximum robot sensor / grip range	0.04
Robot size (diameter) / Maximum movement per iteration	0.02 / 0.01
Initial robot positions	Random (Excluding construction zone)
Simulation environment / Width x height	Continuous / 1.0 x 1.0
Construction zone size (Diameter) / Block size (width / height)	0.16 / 0.012 (Type A / B / C)
Block Type A / B / C Fitness reward	2 / 5 / 10
Generations / Epochs / Iterations per epoch (Robot lifetime)	500 / 3 (CONE-2 / NEAT) / 4000
Mutation (per gene) probability (*p*) / Mutation range	0.05 / [-1.0, +1.0]
Population elite portion	20%
Initial weight (gene) range	[-1.0, +1.0]
Genotype length (Connection weights)	250 (CONE-2) / Variable (NEAT)
Initial number of populations / Genotypes per population	1 (CONE-2 / NEAT) / 100 (CONE-2 / NEAT)
Wait (for cooperation) Z iterations	50
Sensory Input / Motor Output Nodes	22 / 3 (CONE-2 / NEAT)
Initial Sensory Input Nodes / Motor Output Nodes	4 / 3 (NEAT)
Survival threshold / Add node mutation probability	20% (NEAT)
Disjoint Coefficient / Excess Coefficient	2.0 (NEAT)
Weight Difference Coefficient	1.0 (NEAT)
Compatibility Threshold / Modifier	6.0 / 0.3 (NEAT)
Interspecies mutation rate / Recurrent link probability	5% (NEAT)

epochs. One epoch is 4000 simulation iterations and represents a task scenario that tests different robot starting positions, orientations, and block locations in an environment. CONE-2 and NEAT experiments use the same number of genotype evaluations (for each experiment) to ensure a fair comparison. The fitness of CONE-2 and NEAT teams is an average calculated over 20 simulation runs of a given experiment.

Table III presents the simulation parameter settings. These were determined experimentally. Minor value changes produced similar results for both CONE-2 and NEAT evolved teams, and have functioned in related NE controller evolution experiments [12]. NEAT parameters not detailed in table III were given default values specified by Stanley [16].

V. ROBOT CONTROLLERS (CONE-2 AND NEAT)

A. Detection Sensors

A robot's sensory *Field of View* (FOV) is split into *north*, *south*, *east* and *west* sensor quadrants (SQ-0, SQ-1, SQ-2, and SQ-3, respectively, in figure 2, right).

A CONE-2 robot has 12 *block detection* ([SI-0, SI-11]), three *block demand* ([S-12, S-14]), and four *robot detection* ([S-15, SI-16, S-17]) sensors (figure 3, left). Sensor values are normalized to the range [0.0, 1.0].

1) Block Detection Sensors: Are constantly active for a robot's lifetime. Sensor *q* returns the closest block type in quadrant *q*, divided by the squared distance to *this* robot.

2) Robot Detection Sensors: Prevents collisions and enable cooperation. Sensor *q* returns the closest robot in sensor quadrant *q*, divided by the squared distance to *this* robot.

3) Block Demand Sensors: Are constantly active for a robot's lifetime. Each simulation iteration the construction zone broadcasts a signal that is received by each robot's block demand sensors. This signal indicates the block type with the highest demand (next required in the construction sequence). A 1.0 sensor value indicates the highest demand, 0.5 indicates second highest demand, and 0.0 indicates no demand.

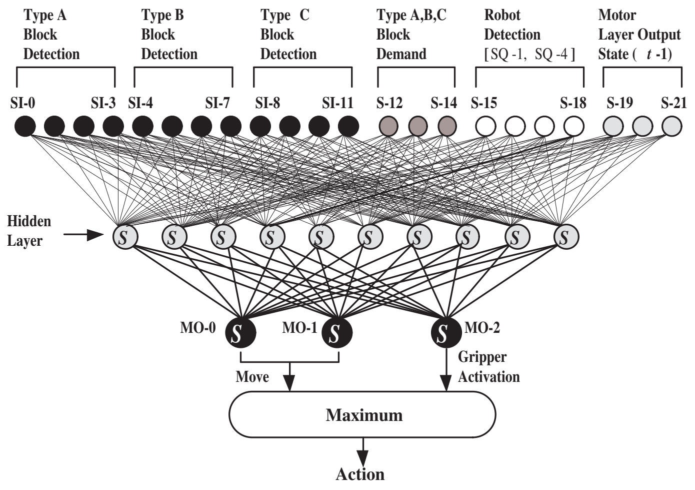
For example, block demand sensor S-12 receives a demand signal equal to 1.0 if block type *A* is the next type required in the construction sequence. Sensor S-13 receives a demand signal value of 0.5 if a type *B* block is the next type required after block type *A*. Sensor S-14 receives a signal of 0.0 if a type *C* block is no longer required. If block types *B* and *C* are both required after type *A* (for example, where type *B* and *C* blocks can be connected to either side of a type *A* block), then sensors S-13 and S-14 will receive a 0.5 signal value.

B. Movement Actuators

Two wheel motors control a robot's heading at a constant speed (table III). Wheel motors (MO-0 and MO-1 in figure 3) must be explicitly activated. Movement is calculated in terms of real valued vectors (*dx* and *dy* corresponding to the outputs of MO-0 and MO-1). A robot's heading is determined by normalizing and scaling its motor output values by the maximum distance a robot can traverse in one iteration, d_{max} (table III). Where, $dx = d_{max}(o_1 - 0.5)$, and $dy = d_{max}(o_2 - 0.5)$, and o_1 and o_2 are the MO-0 and MO-1 output values, respectively. To calculate the distance between *this* robot, and other robots and blocks in the environment, the squared Euclidean norm, bounded by a minimum observation distance is used [22].

C. Block Gripper

Each robot is equipped with a gripper to transport blocks to the construction zone. The gripper motor must be explicitly activated (MO-2 in figure 3). If no block is held, the robot grips the closest block within gripper range (table III). If the robot is already gripping a block, then the block is released.



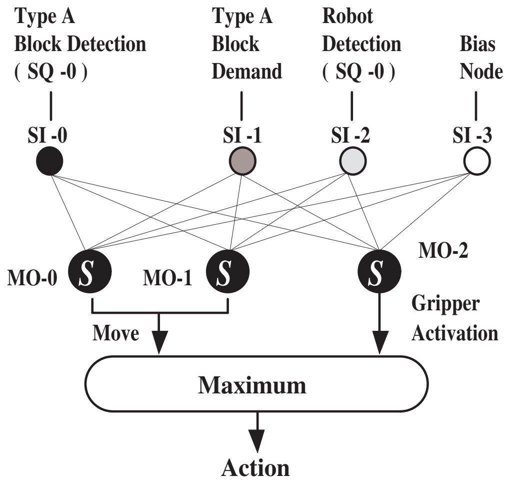


Fig. 3. *CONE-2 (Left) NEAT (Right) Controllers.* NEAT initializes a homogenous controller population subject to *complexification*. The NEAT controller illustrated is an example only. Controllers are initially random sensory-motor configurations (right). CONE-2 uses a controller with a fixed topology (left).

D. CONE-2: ANN Controller

CONE-2 robots use a recurrent ANN controller [23], fully connecting 22 sensory input neurons to 10 hidden layer neurons to three motor output neurons (figure 3, left). Hidden and output neurons are sigmoidal [24] units. Sensory input neurons [SI-19, SI-21] have recurrent connections that accept the previous activation state of the output layer. At each simulation iteration of the robot's lifetime the motor output with the highest value is the action executed.

1) *MO-0, MO-1:* Calculate direction of movement from motor outputs MO-0 (*dx*) and MO-1 (*dy*).
2) *MO-2:* Activate gripper.

E. NEAT: Initial ANN Controller

NEAT robots begin with a minimalist yet functional ANN controller (figure 3, right), that is subject to NEAT's *complexification* process. The initial controller uses four sensory input neurons fully connected to three motor outputs. To ensure that NEAT controllers accomplish the collective construction task with some degree of success, motor outputs are kept the same as the CONE-2 controller (figure 3, left). At each iteration of the robot's lifetime the motor output with the highest value is the action executed. Initial sensory inputs are one *block type A detection sensor* (SI-0) using the robot's north sensor quadrant (SQ-0), one *block type A demand sensor* (SI-1) using all four sensor quadrants ([SQ-0, SQ-3]), one robot detection sensor (SI-2) using the robot's north sensor quadrant (SQ-0), and one bias node (SI-3). The bias node uses a constant weight value of 1.0. Connection weight values for this topology are randomly initialized in the range [-1.0, 1.0].

VI. Results and Discussion

For statistical pair-wise comparisons, unpaired two-sample t-tests [25] (α = 0.05) were used.

A. Blocks Connected

Figure 4 (left) presents the average *number of blocks* connected for the fittest CONE-2 and NEAT evolved teams. Fittest teams were selected from the final generation of 10 runs, executed for each of 10 construction schemas (table I), and the average task performance calculated. Statistical comparisons indicated that for construction schemas 1 to 5, where a *low degree* (two robots) or *no cooperation* was required, NEAT evolved teams outperformed CONE-2 evolved teams in terms of the average number of blocks connected.

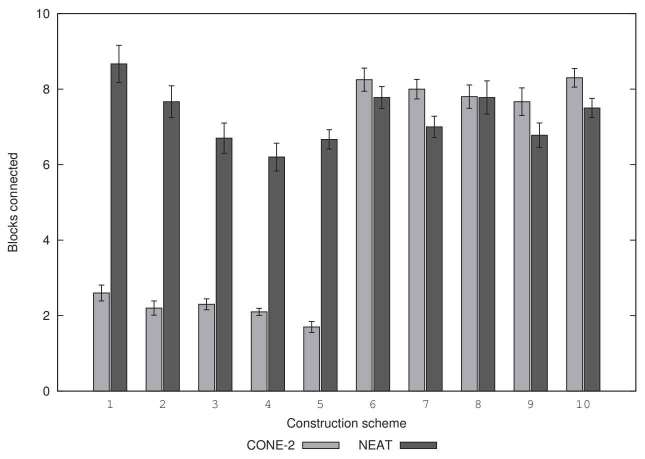
For construction schema 6 (low degree of cooperation), 8 and 9 (*high degree of cooperation*, three robots), CONE-2 and NEAT evolved teams yielded comparable task performances. This result indicates that both the NEAT and CONE-2 team size adaptation approaches are equally beneficial for schemas mandating low and high degrees of cooperation.

However, for construction schemas 7 and 10 (high cooperation), CONE-2 evolved teams yielded a significantly higher task performance, compared to NEAT evolved teams. This result indicates that there are cooperative task instances where CONE-2's team size adaptation mechanism is most suitable.

B. Team Size

Figure 4 (right) presents the average *team size*, for the fittest CONE-2 and NEAT evolved teams. For each construction schema, the fittest teams were selected from each of the 10 runs, and the average calculated. In figure 4 (right) note that when no error bars are shown, then team size does not vary.

For schemas 1 to 5, CONE-2 uses only one robot since most blocks could be connected by one agent. Also, one robot lifetime (4000 iterations) was not sufficient time to gather blocks that must be connected cooperatively (table II). Thus, most of a robot's lifetime was spent gathering blocks that could be individually connected, and an average of two blocks were connected for schemas 1 to 5 by the fittest CONE-2 teams. However, the fittest NEAT teams connected an average of 9



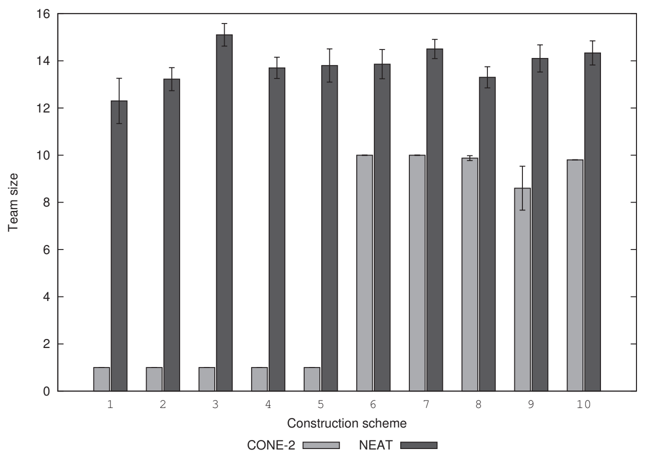


Fig. 4. *Left:* Average number of blocks connected as a structure by the fittest CONE-2 and NEAT evolved teams (for 10 construction schemas). *Right:* Average team size of the fittest CONE-2 and NEAT evolved teams (for 10 construction schemas).

to 7 blocks for schemas 1 to 5 (figure 4, left) as a result of comparatively large team sizes (figure 4, right).

Whereas, a single robot could make at least one connection in construction schemas 1 to 5, for schemas 6 to 10 at least two robots were required to connect any blocks. For construction schema 6, two robots were required to make all connections (tables I and II). Thus, for schema 6, it was necessary for CONE-2 to use more robots, compared to schemas 1 through 5 (figure 4, right). However, for schemas 7 to 10, three robots were required to build all connections (table I). Suitably, for schemas 7, 8 and 10, CONE-2 used 10 robots (and 9 robots for schema 9) for the fittest evolved teams.

Results also indicate the fittest NEAT teams, evolved for each construction schema, used larger team sizes, an average of 13 robots (figure 4, left). This resulted from NEAT increasing team size according to its speciation mechanism, which maintained genotype diversity and protected new controllers from being prematurely removed from the genotype population (by placing them in a new species) [16]. Increasing team sizes as a function of NEAT's speciation mechanism also resulted in comparatively larger teams. This resulted in a significantly higher performance (figure 4, left) for schemas 1 to 5 (requiring a low degree of cooperation), since more robots, on average, gathered and connected more blocks.

A notable result for the fittest CONE-2 and NEAT evolved teams (for schemas 6 to 10), was that labor was divided amongst team members. Large team sizes were beneficial in tasks mandating low to high cooperation since there were sufficient robots to concurrently gather and connect blocks.

C. Correlating Team Size and Task Performance

Notably for schemas 6 to 10, the fittest CONE-2 evolved teams were consistently smaller compared to the fittest NEAT evolved teams, yet CONE-2 teams achieved a comparable or significantly higher task performance (figure 4).

To test the correlation between team size and the number of blocks connected, the Pearson product-moment correlation coefficient [25] was applied. The coefficient was applied to construction schemas 6 to 10, since in these schemas the fittest CONE-2 and NEAT evolved teams yielded comparable task performances. The exception was schemas 7 and 10, where CONE-2 evolved teams yielded a significantly higher task performance, compared to the fittest NEAT evolved teams.

Results are presented in table IV. The correlation coefficient ranges from -1.0 to 1.0. A value of 1.0 implies that blocks connected increases as robots increase, -1.0 implies that blocks connected decreases as team size increases, and 0.0 implies no correlation between blocks connected and team size.

Results (table IV) indicate the correlation for the fittest NEAT evolved teams ranges from 0.54 to 0.69, highlighting a high positive correlation between team size and blocks constructed. However, for the fittest CONE-2 evolved teams (schemas 6 to 10), there is a very strong positive correlation between team size and blocks connected (0.99).

This very strong correlation indicates that the fittest CONE-2 evolved teams only adapt to become as large as required. That is, CONE-2 evolved teams become just large enough such that the task is accomplished efficiently. There are a sufficient number of robots to make all cooperative connections required by a schema, as well as a sufficient number for maximizing the number of blocks gathered during a team's lifetime. In the case of NEAT, the lower correlation indicates that larger team sizes were not necessarily beneficial in all schemas.

Comparatively large NEAT team sizes were used for all construction schemas. Whilst this was beneficial for schemas requiring no or low cooperation, for schemes requiring high cooperation, it is theorized that the larger team sizes were, in some instances, counter productive. That is, more than three robots would be concurrently attempting to cooperate, when only two or three were required to connect a block. Such physical interference between many robots concurrently attempting to grip a block delayed block connection times as robots had to try again. This led to lower task performance for NEAT teams in some schemas, compared to CONE-2 teams.

Thus the key result supports the hypothesis of this study

TABLE IV
CORRELATIONS BETWEEN NUMBER OF AGENTS PRESENT AND MAXIMUM BLOCKS CONSTRUCTED PER CONSTRUCTION SCHEMA.

Pearson Correlation Coefficients		
Construction Schema	NEAT	CONE-2
6	0.69	0.99
7	0.64	0.99
8	0.62	0.99
9	0.58	0.99
10	0.54	0.99

(section I-B). That is, that dynamically adapting team size as function of genotype diversity (NEAT) is comparably effective to adapting team size according to behavioral success (CONE-2). The caveat is that when the task requires no or occasional cooperation, then NEAT evolves more effective teams. However, when cooperation is always required then NEAT and CONE-2 evolved teams yield comparable task performances.

However, the mechanisms affecting relationships between team size and cooperative behavior for a given task and controller evolution type, are the subject of ongoing research.

VII. CONCLUSIONS

This study's objective was to elucidate the impact of two comparative methods for dynamically adapting team size in a collective construction task that required varying degrees of cooperation between robots. Team size adaptation was executed as part of the CONE-2 and NEAT methods (applied to controller design). CONE-2 team sizes were adapted as a function of cooperative behavior success. Whereas, NEAT team sizes were adapted as a function of genotypic (team behavior encoding) diversity. The comparison was motivated by the research question of whether a purely genotypic versus a behavioral mechanism is sufficient for dynamic team adaptation in cooperative tasks (in this study, collective construction).

Results indicated that if the task required cooperation (two or three robots), the fittest CONE-2 teams yielded comparable or significantly higher task performances, comparative to the fittest NEAT teams. However, CONE-2 teams were consistently smaller, indicating the evolution of a greater efficiency in collective construction task accomplishment.

This study concludes that both genotypic and behavioral approaches are well suited to dynamically adapt team sizes in the collective construction task. However, this was a preliminary study, simply aiming to elucidate a clear benefit of either team size adaptation approach, with respect to a cooperative task. The key result was that neither approach demonstrated clear benefits when cooperation was always required.

Thus as a next step, the benefits of either approach applied in a general set of collective behavior tasks (those mandating cooperation), will be examined. Furthermore, the benefits of combining genotypic and behavioral approaches as a means to dynamically adapt team sizes to suit a required degree of cooperation, is the subject of future research.

REFERENCES

[1] C. Schultz and L. Parker, in *Multi-robot Systems: From Swarms to Intelligent Automata*. Washington DC, USA: Kluwer, 2002.

[2] Y. Zhang and R. Vaughan, "Ganging up: Team-based aggression expands the population/performance envelope in a multi-robot system," in *Proceedings of the International Conference on Robotics and Automation*. Orlando, USA: IEEE Press, 2006, pp. 589–594.

[3] K. Cheng, P. Dasgupta, and B. Banerjee, "Adaptive multi-robot team reconfiguration using a policy-reuse reinforcement learning approach," in *Proceedings of the international conference on Advanced Agent Technology*. Berlin, Germany: Springer-Verlag, 2011, pp. 330–345.

[4] A. Kleiner, D. Sun, and D. Meyer-Delius, "Armo: Adaptive road map optimization for large robot teams," in *Proceedings of the International Conference on Intelligent Robots and Systems*. San Francisco, USA: IEEE Press, 2011, pp. 3276–3282.

[5] W. Agassounon and A. Martinoli, "A macroscopic model of an aggregation experiment using embodied agents in groups of time-varying sizes," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*. Pasadena, USA: IEEE Press, 2002, pp. 250–255.

[6] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines.* Cambridge, USA: MIT Press, 2000.

[7] L. Parker, "The effect of action recognition and robot awareness in cooperative robotic teams," in *Proceedings of the International Conference on Intelligent Robots and Systems*. IEEE Press, 1995, pp. 212–219.

[8] ——, "Adaptive heterogeneous multi-robot teams," *Neurocomputing*, vol. 28, no. 1, pp. 75–92, 1999.

[9] K. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation.*, vol. 10, no. 2, pp. 99–127, 2002.

[10] G. Nitschke, "Behavioral heterogeneity and collective construction," in *Proceedings of the Congress on Evolutionary Computation*. Brisbane, Australia: IEEE Press, 2012, pp. 387–394.

[11] G. Nitschke, M. Schut, and A. Eiben, "Collective neuro-evolution for evolving specialized sensor resolutions in a multi-rover task," *Evolutionary Intelligence*, vol. 3, no. 1, pp. 13–29, 2010.

[12] ——, "Evolving behavioral specialization in robot teams to solve a collective construction task," *Swarm and Evolutionary Computation*, vol. 2, no. 1, pp. 25–38, 2012.

[13] D. D'Ambrosio and K. Stanley, "Generative encoding for multiagent learning," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Atlanta, USA: ACM Press, 2008, pp. 1–8.

[14] K. Stanley and R. Miikkulainen, "Competitive coevolution through evolutionary complexification." *Journal of Artificial Intelligence Research*, vol. 21, no. 1, pp. 63–100, 2004.

[15] J. Rojas and R. Peters, "Analysis of autonomous cooperative assembly using coordination schemes by heterogeneous robots using a control basis approach," *Autonomous Robotics*, vol. 32(1), pp. 369–383, 2012.

[16] K. Stanley, *Efficient Evolution of Neural Networks Through Complexification. Ph. D. Dissertation*. Austin, USA: Department of Computer Sciences, The University of Texas, 2004.

[17] F. Gomez, J. Schmidhuber, and R. Miikkulainen, "Efficient non-linear control through neuroevolution," in *Machine Learning: ECML 2006*. Berlin, Germany: Springer, 2006, pp. 654–662.

[18] M. Potter and K. De Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary Computation*, vol. 8, no. 1, pp. 1–29, 2000.

[19] G. Nitschke, *Neuro-Evolution for Emergent Specialization in Collective Behavior Systems. PhD thesis*. Amsterdam, Netherlands: Computer Science Department, Vrije Universiteit, 2009.

[20] F. Gomez and R. Miikkulainen, "Incremental evolution of complex general behavior," *Adaptive Behavior*, vol. 5, no. 1, pp. 317–342, 1997.

[21] A. Eiben and J. Smith, *Introduction to Evolutionary Computing*. Berlin, Germany: Springer-Verlag, 2003.

[22] A. Agogino and K. Tumer, "Efficient evaluation functions for multi-rover systems," in *Proceedings of the Genetic and Evolutionary Computation Conference*. New York, USA: Springer, 2004, pp. 1–12.

[23] J. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 1, pp. 179–211, 1990.

[24] J. Hertz, A. Krogh, and R. Palmer, *Introduction to the Theory of Neural Computation*. Redwood City, USA: Addison-Wesley, 1991.

[25] B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes*. Cambridge, United Kingdom: Cambridge University Press, 1986.