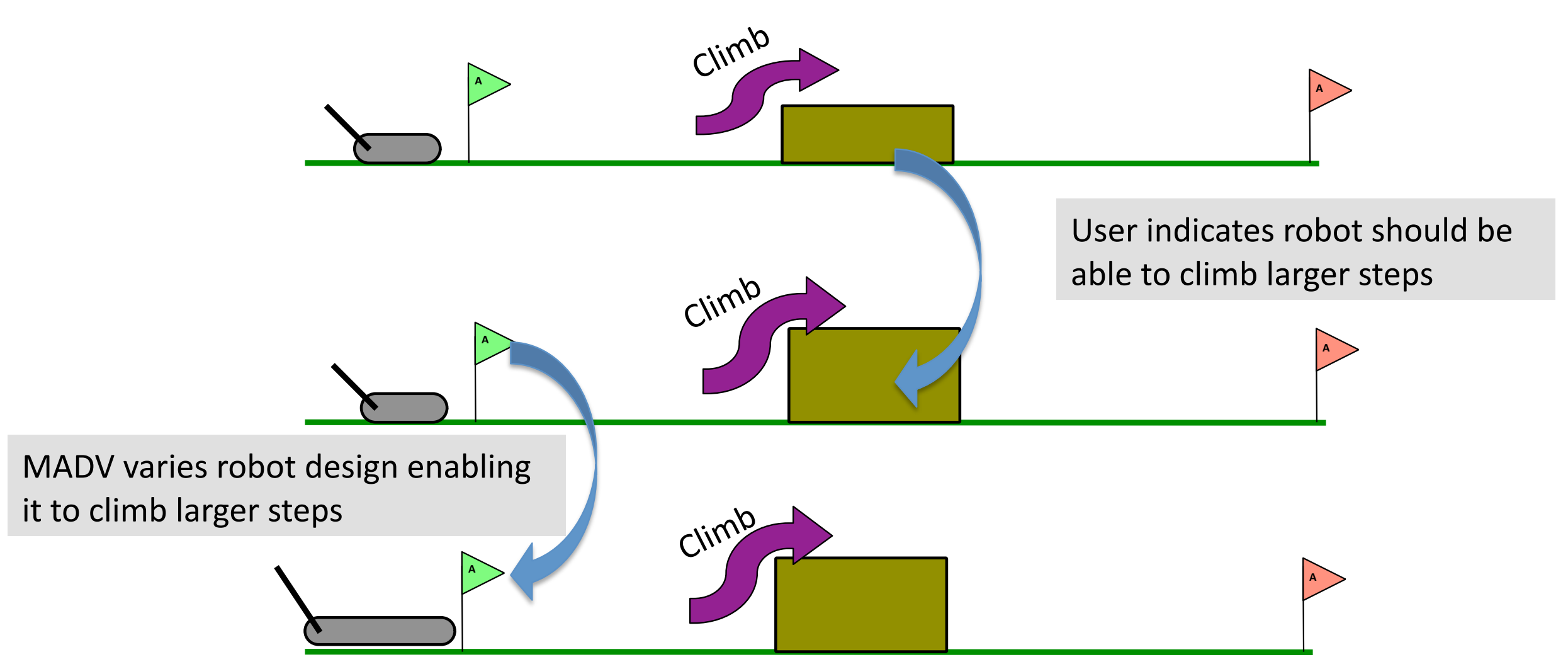
Morphogenetically Assisted Design Variation

Raytheon BBN Technologies · iRobot

Aaron Adler, Fusun Yaman, Jeffrey Cleveland, Annan Mozeika, and Jacob Beal

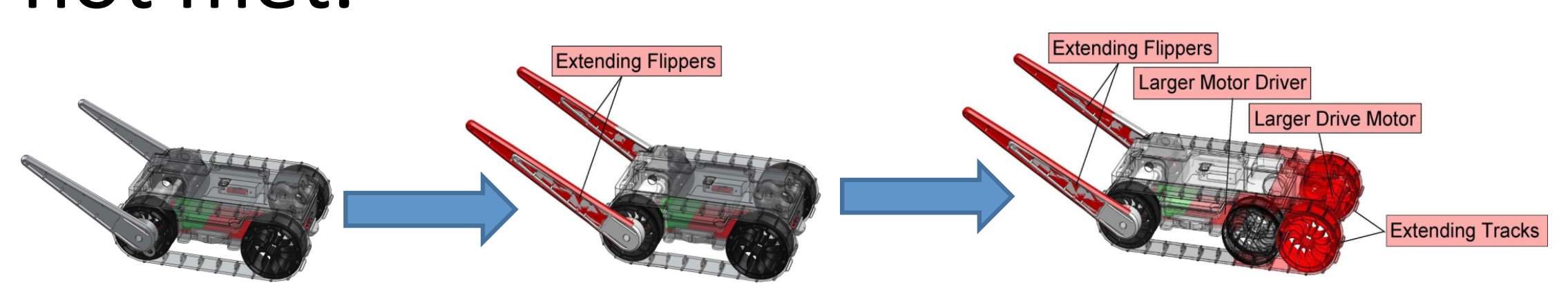
madv-team@bbn.com http://madv.bbn.com

Objective: MADV is attempting to enable the rapid customization of robots and other vehicles.


Climb
Climb
Climb
User indicates robot should be able to climb larger steps
MADV varies robot design enabling it to climb larger steps

A novice designer **indirectly specifies robot variations** through the modification of operating environments (e.g., obstacle courses) or key design elements (e.g., flipper size).

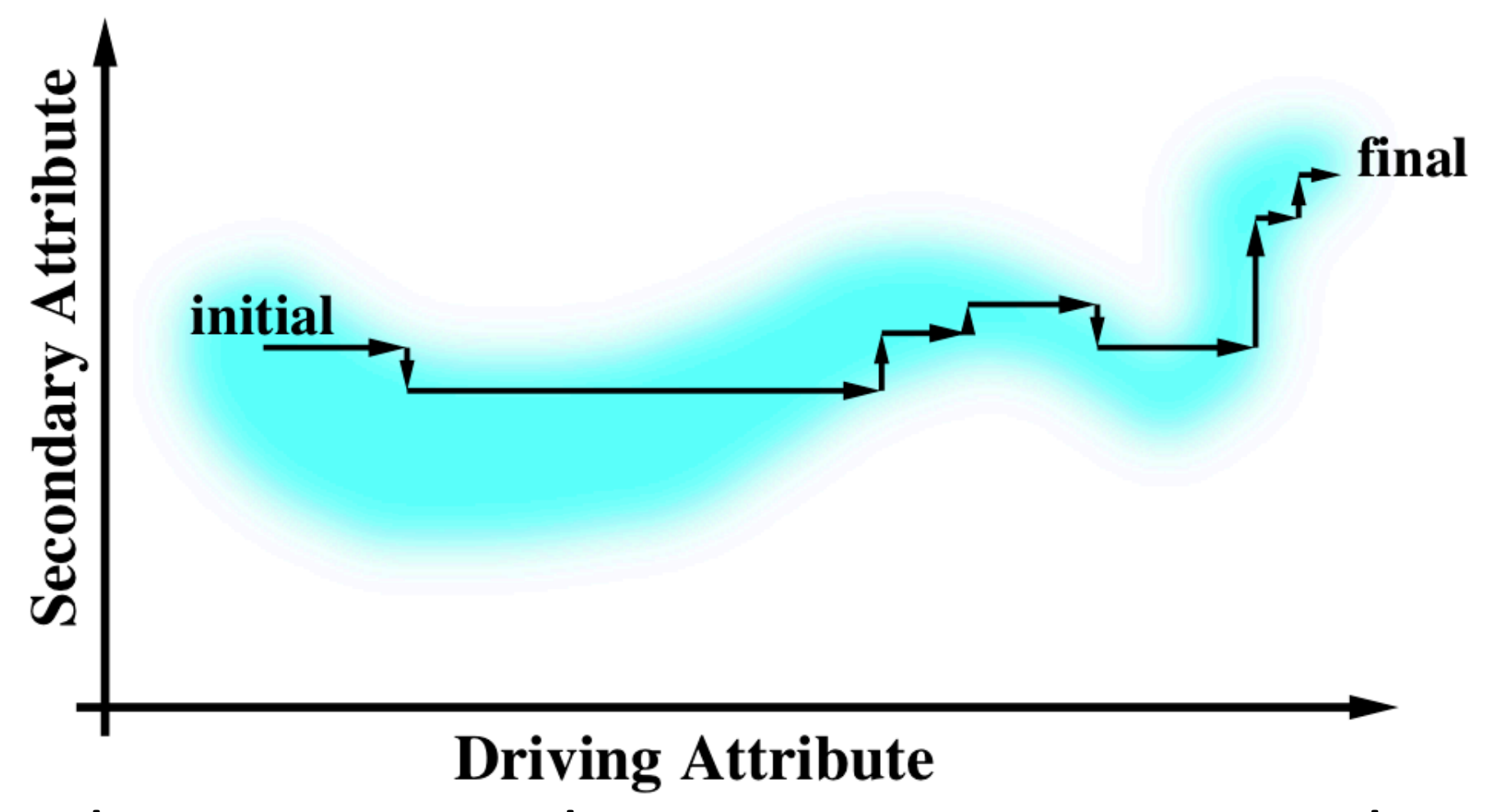
Approach: Functional blueprints are used to specify *design as behavioral goals* and a method for adjusting the physical structure when those goals are not met.



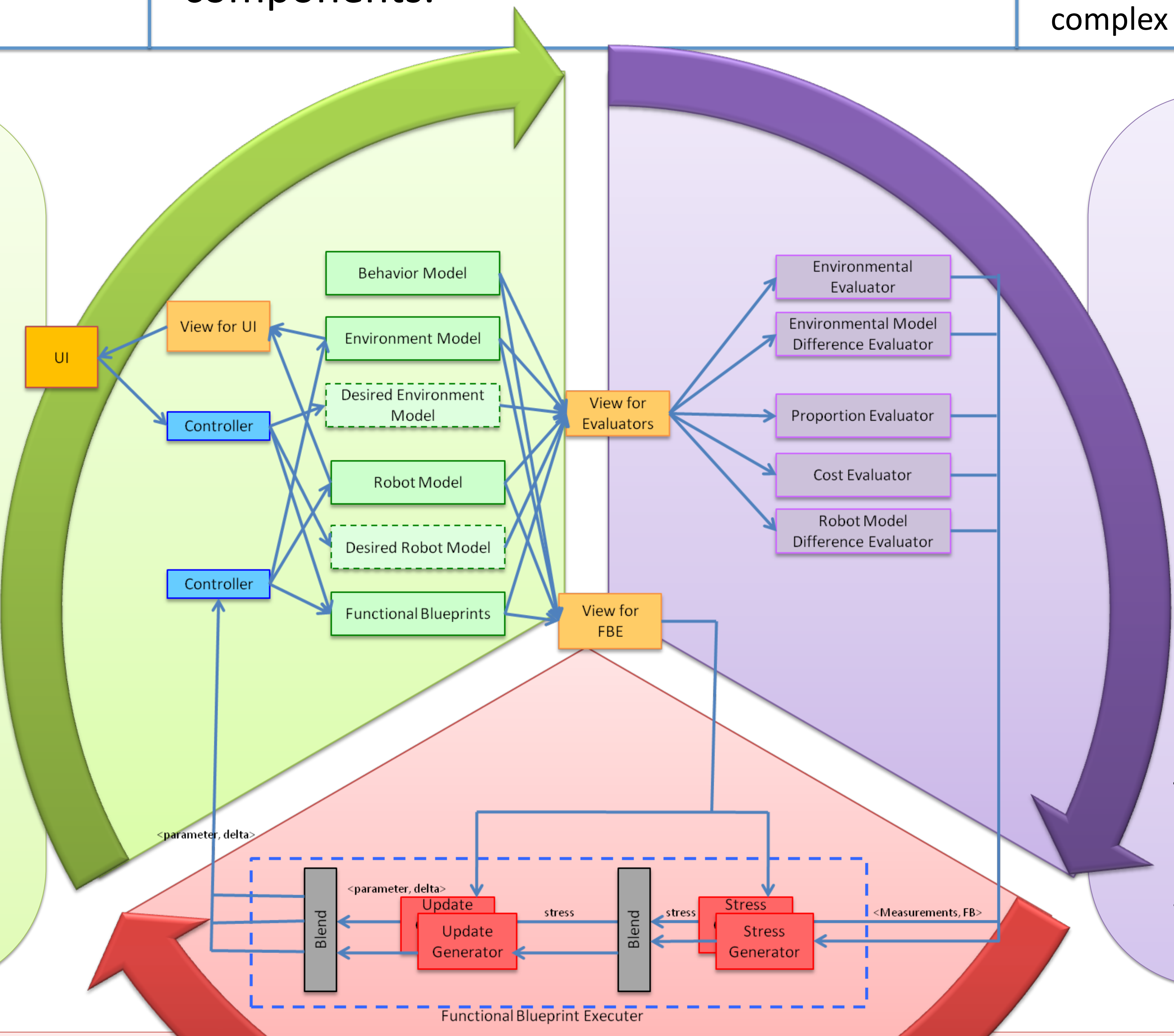
By encoding the relationships of structure and function, functional blueprints implicitly **capture system interdependencies** enabling the system to adapt to changes made on individual components.

Definition: Functional blueprints consists of four things:

1. A system behavior that degrades gracefully
2. A stress metric quantifying the degree and direction of stress on the system
3. An incremental program that relieves stress through growth, shrinkage, or other structural change
4. A program to construct an initial viable system



In this abstract example, a growing system with two attributes uses stress tolerance to navigate through a complex viability envelope (blue).

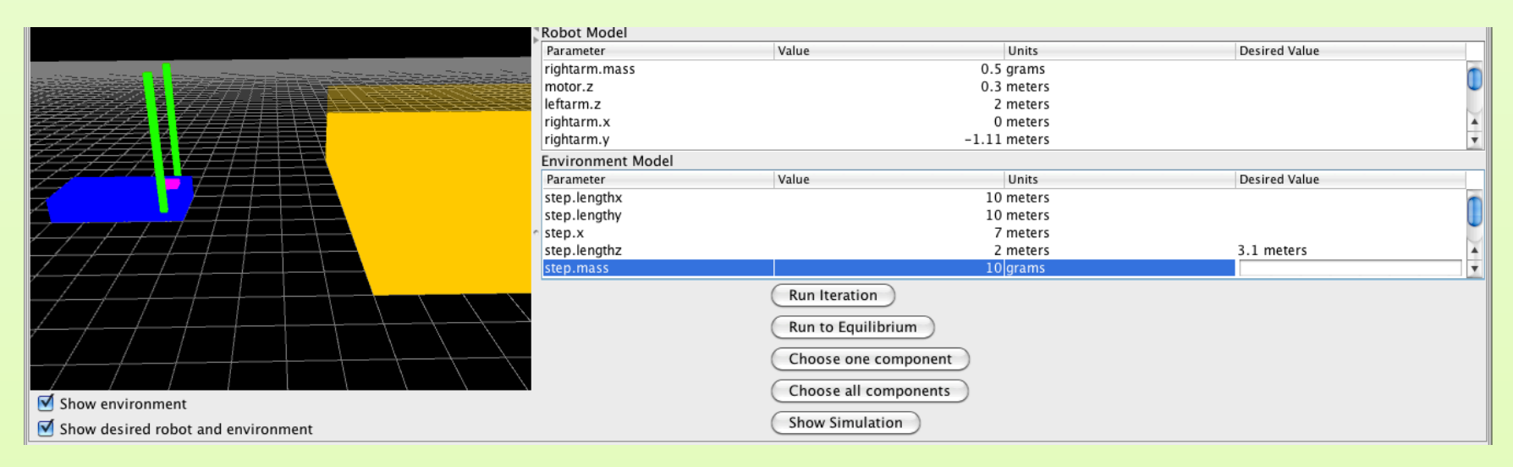


1.) Model the system

An initial model of the robot is specified based on an existing viable robot design, e.g., the iRobot miniDroid.

A model consists of parameters corresponding to physical features of the robot, e.g., dimensions of flippers, body length, motor strength.

A user is able to specify changes to the operating environment by changing parameters in the desired environment model.

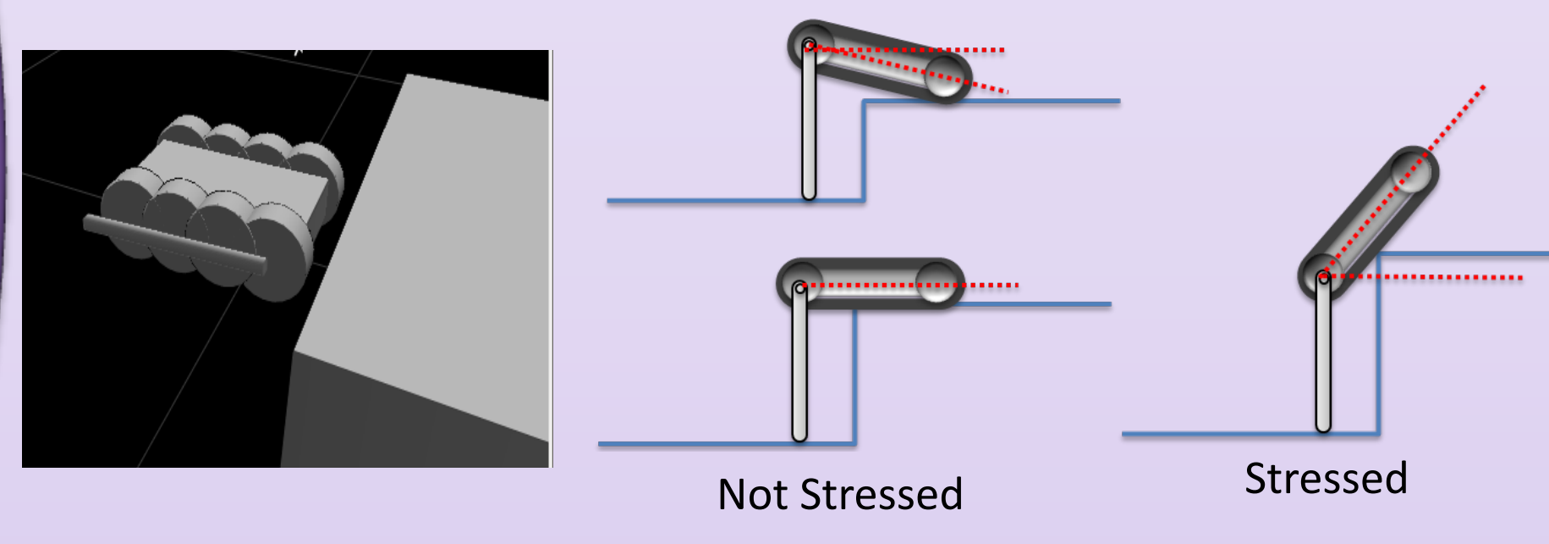


These are linked to other elements of the design such as the FBs and behavior models.

2.) Evaluate the model

Evaluators analyze the functionality of the current design, returning a set of metrics.

For example, the environmental evaluator simulates the current robot model operating (as defined by the static behavior model) in a set of environment models and returns metrics such as time to climb over an obstacle, or posture while doing so.



The robot and environmental model difference evaluator return a metric reflecting the difference between the desired models and the current model

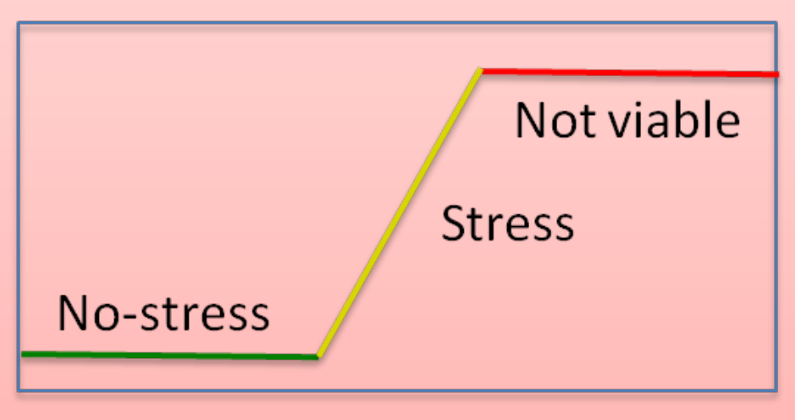
3.) Update the model using Functional Blueprints

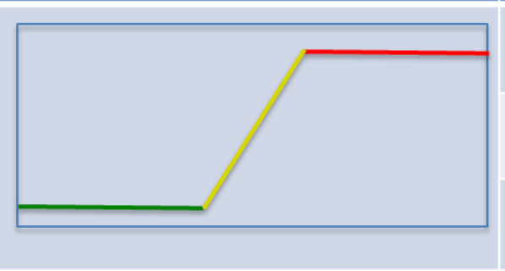
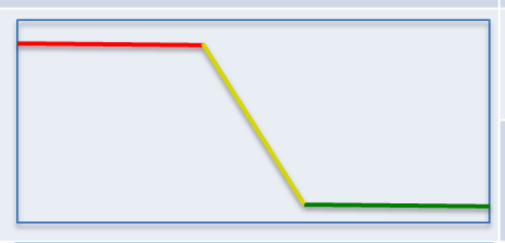
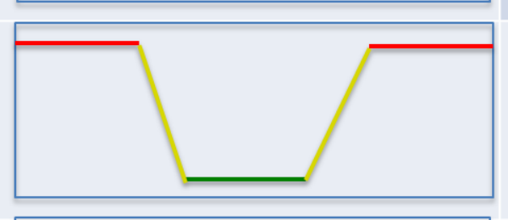
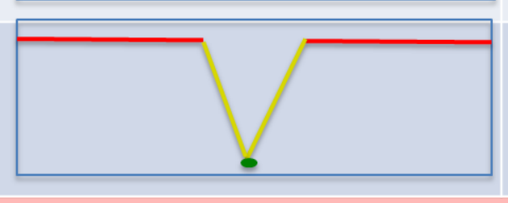
Metrics from the evaluators generate stresses on given functional blueprints, these stresses are blended together using the following equation:

$$V = \text{sign}\left(\sum_i V_i\right) \cdot \max_i\left(V_i \frac{\sum_i V_i}{\sum_i |V_i|}\right)$$

The functional blueprints then create deltas for parameters of the robot model based on the blended stresses.

These deltas are blended in the same manner and then added to the model.



FB	Stress Function	Update Functions
Max Torque		Decrease torque
Flipping		Increase torque
Body Proportion		Decrement flipper length
Climb		Increase flipper and body length
Servo Viability		Decrease torque and increase motor mass
Servo Mass		Increase/decrease motor mass
Perturbation on X		Increase/decrease perturbed attribute X

Example functional blueprints

Convergence Dynamics Analysis

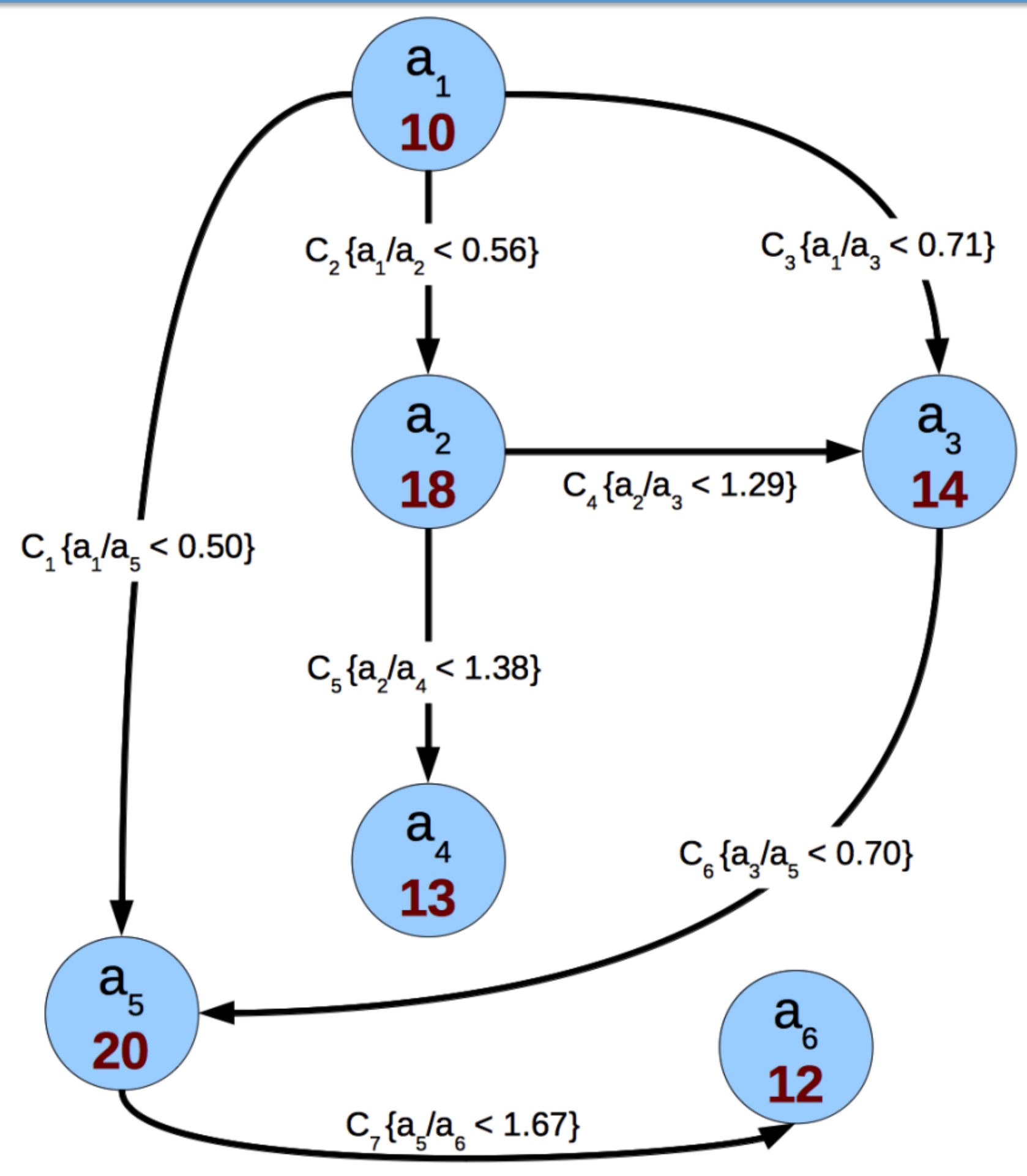
In order to evaluate the viability of this approach at a proof of concept level experiments were run on randomly generated acyclic constraint graphs representing a network of functional blueprints

N nodes (attributes): $a_1 \ldots a_N$
  Value range [10,20]
K edges (blueprints)
  Ratio constraints

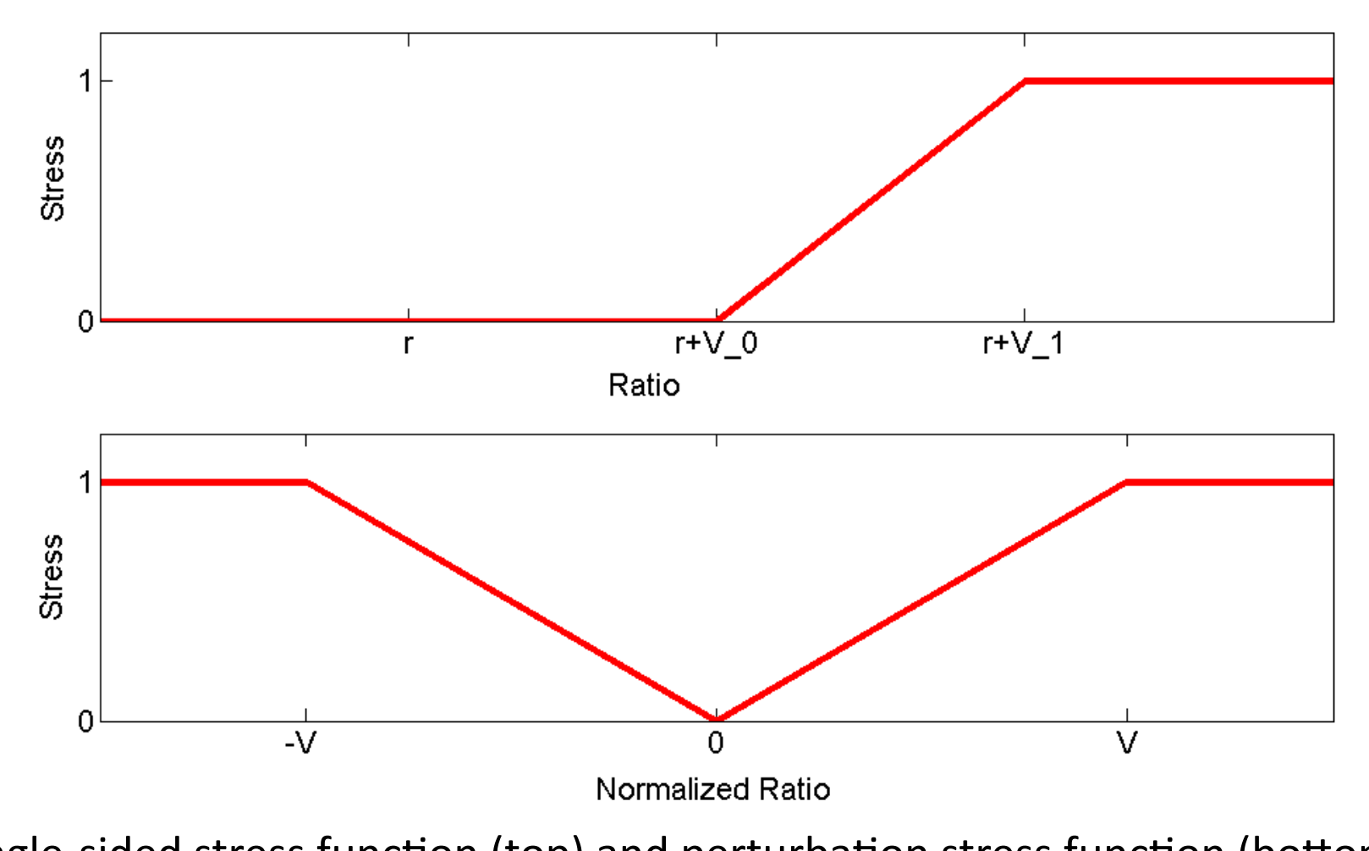
Single sided stress functions as seen below were used (absolute increments).

Once the network was created a perturbation was injected on one of the inputs, which would increase the value of the attribute by a factor of *P* at a drive rate of *d* per time step.

Conclusion: Networks of functional blueprints were able to disperse stress quickly and reliably converged to an appropriately adapted set of values.



Randomly generated network of functional blueprints.



Single-sided stress function (top) and perturbation stress function (bottom).

Experiment 1: Induce a 100% perturbation in one of the attributes

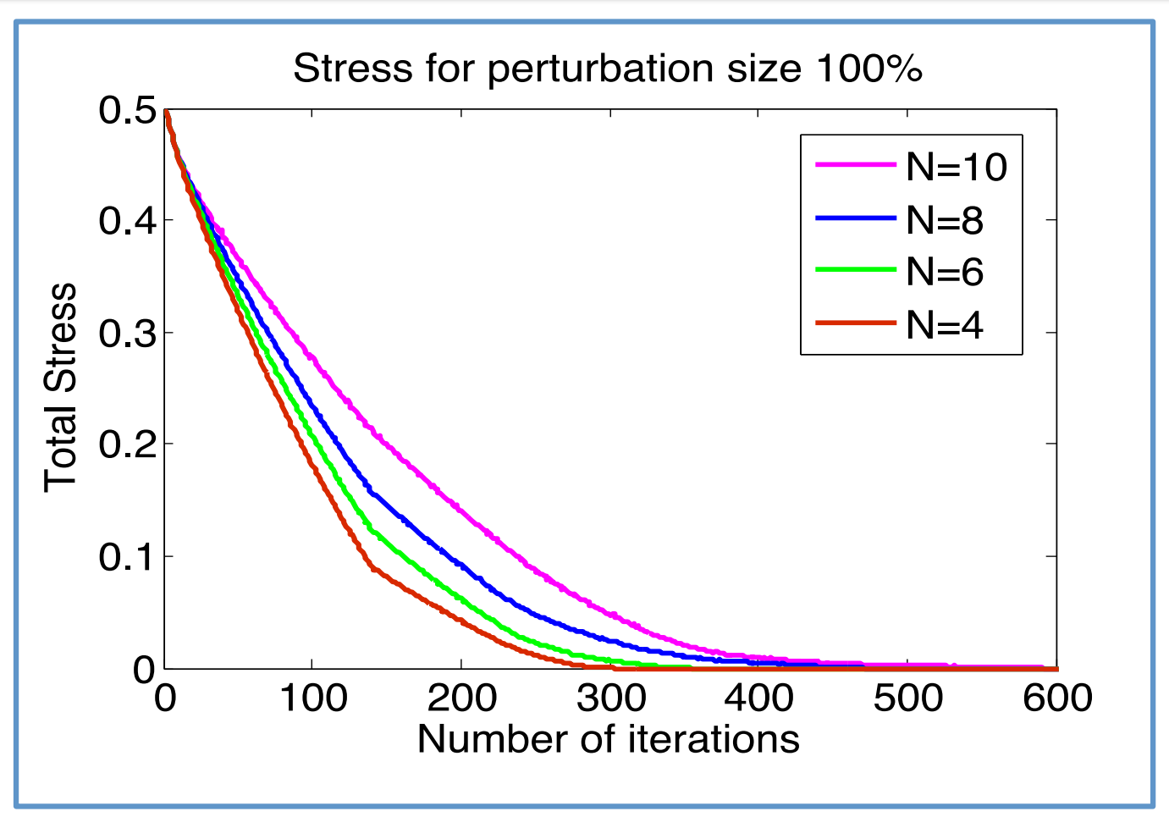
- For all random graphs tested, total stress decreases exponentially
- Majority of the stress is the user perturbation (TOP)
- The rest of the system disperses the stress efficiently (MIDDLE)

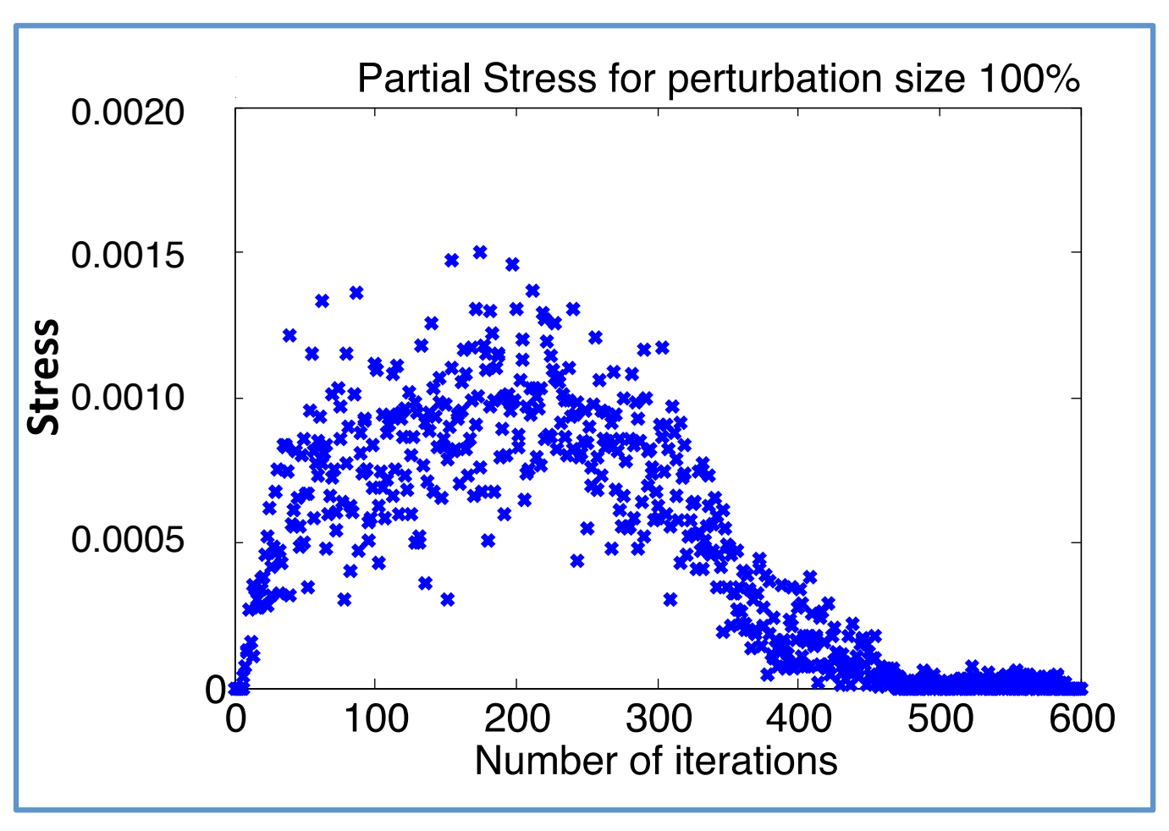
Experiment 2: Vary perturbation from 100% to 500%

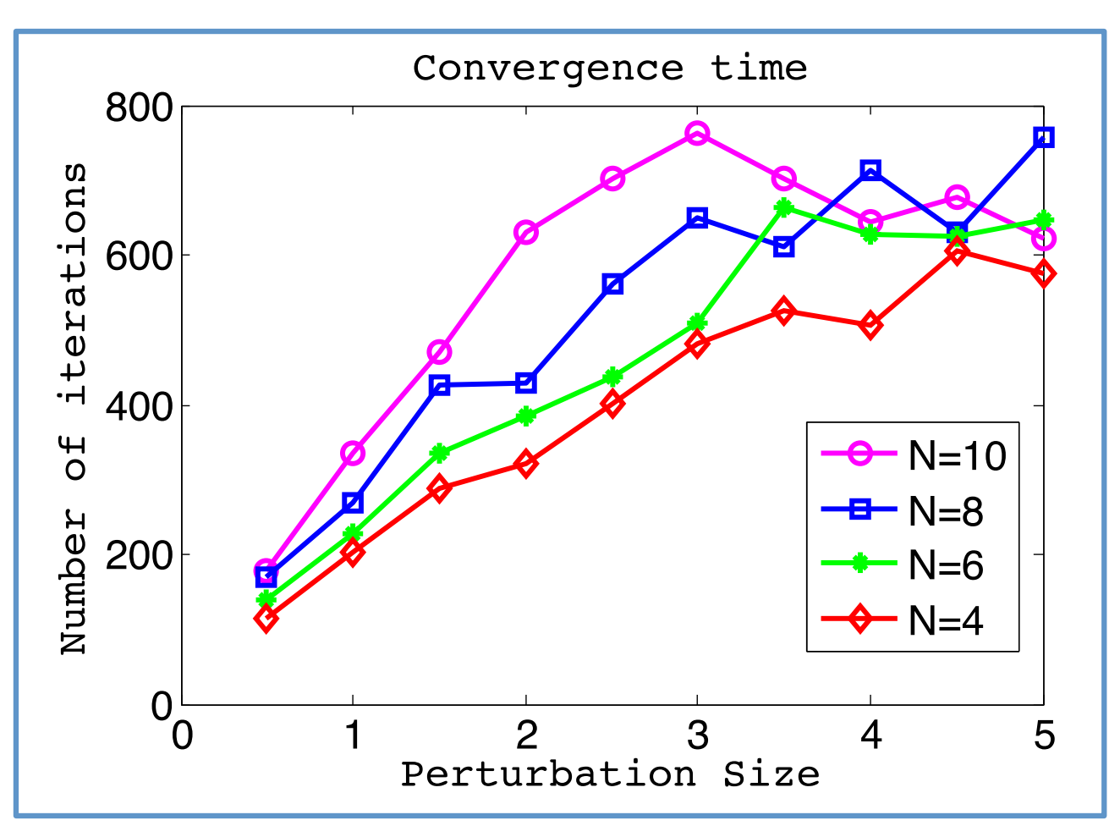
- Regardless of the size of the constraint graph, linear relationship between the perturbation size and convergence time (BOTTOM LEFT)

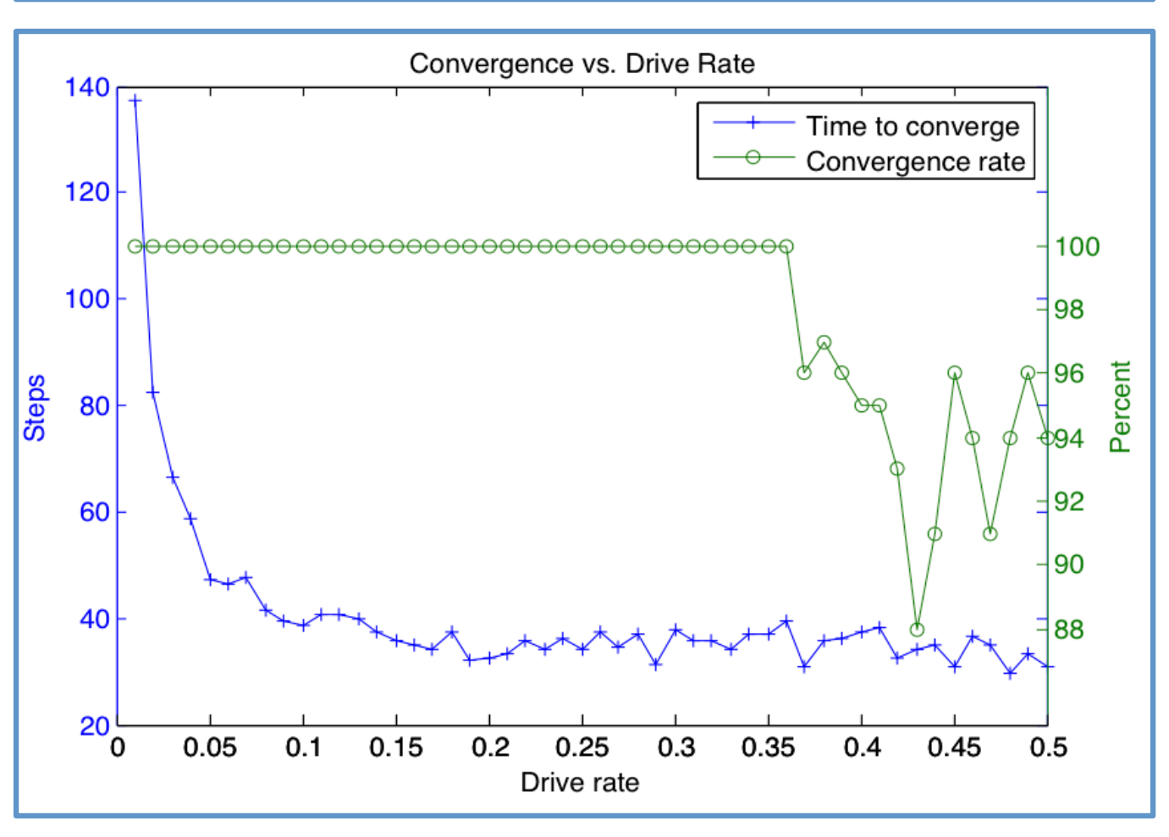
Experiment 3: Vary drive rate from 0.01% to 50%

- Faster drive rates lead to faster convergence
- When increment rate is too high the system becomes non-viable (BOTTOM RIGHT)









Work sponsored by DARPA DSO under contract W91CRB-11-C-0052;
The views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.