

Coevolution:  
Studying the Dynamics of Competition  
*and*  
Gauss: Scenario Translation

*Mary McDonald*

*Science Applications International Corporation*

# What is “coevolution”?

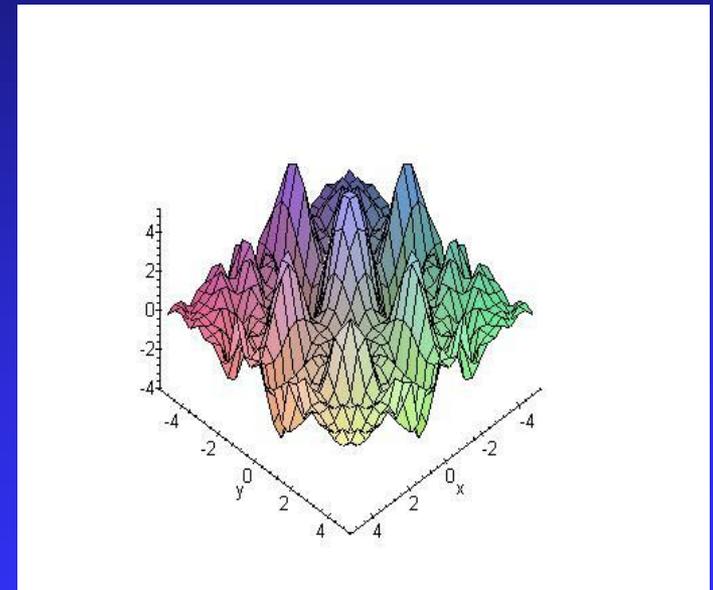
- A term originally from Evolutionary Biology
- Describes the phenomenon of two species evolving over time, *in direct relation to*, each other
  - ◆ e.g. predator and prey
  - ◆ ... like an arms race

# So why bring a concept from Evolutionary Biology into Operations Research?

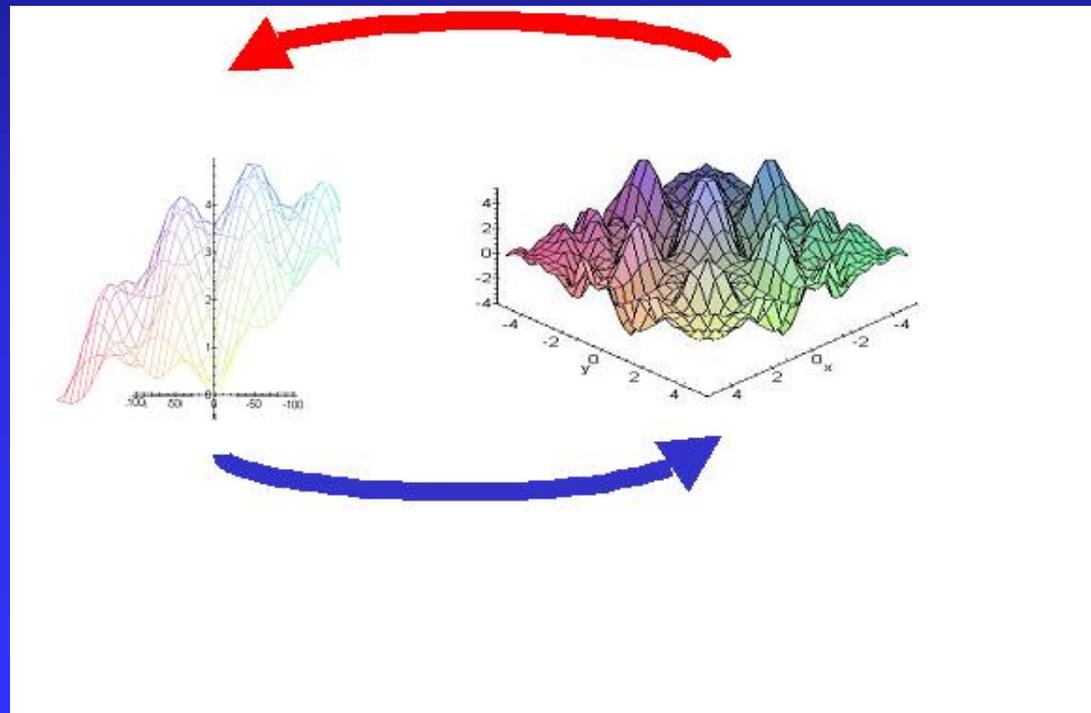
- Because it describes what actually happens in the real world!
- Perhaps a better understanding of the dynamics of this situation enables us to develop better, more effective command and control

# The Challenge of Coevolving Decision Landscapes

- Although the classical approach is to try to “optimize” over a (perceived fixed) decision making landscape... the reality is that the landscape is **dynamic**, and may be “**coupled**” (e.g. with an adversary’s landscape)
- Need to understand how your (perceived) landscape (and your adversary’s) changes as a function of:
  - ◆ What you do
  - ◆ What the other does
  - ◆ What you think the other might do
  - ◆ What the other thinks you might do
  - ◆ ...



Coevolving entities can turn each other's peaks into valleys!



# Why Study Coevolution?

- Hopefully gain insight on how to “win the coevolutionary race”
- Learn more about how to find a robust heuristic for decision-making strategies in a coevolutionary environment, e.g. a *changing environment with feedback loops*

## 2 aspects of “winning” the coevolutionary race:

### ■ *Deceiving (Out-thinking)*

- ◆ I bluff or fake you into a move, which I then exploit (*winning the “I think he thinks” game*)
- ◆ Can act on more than one “level” (e.g. but you don’t know that I know you know!!)

### ■ *Acting (Out-doing)*

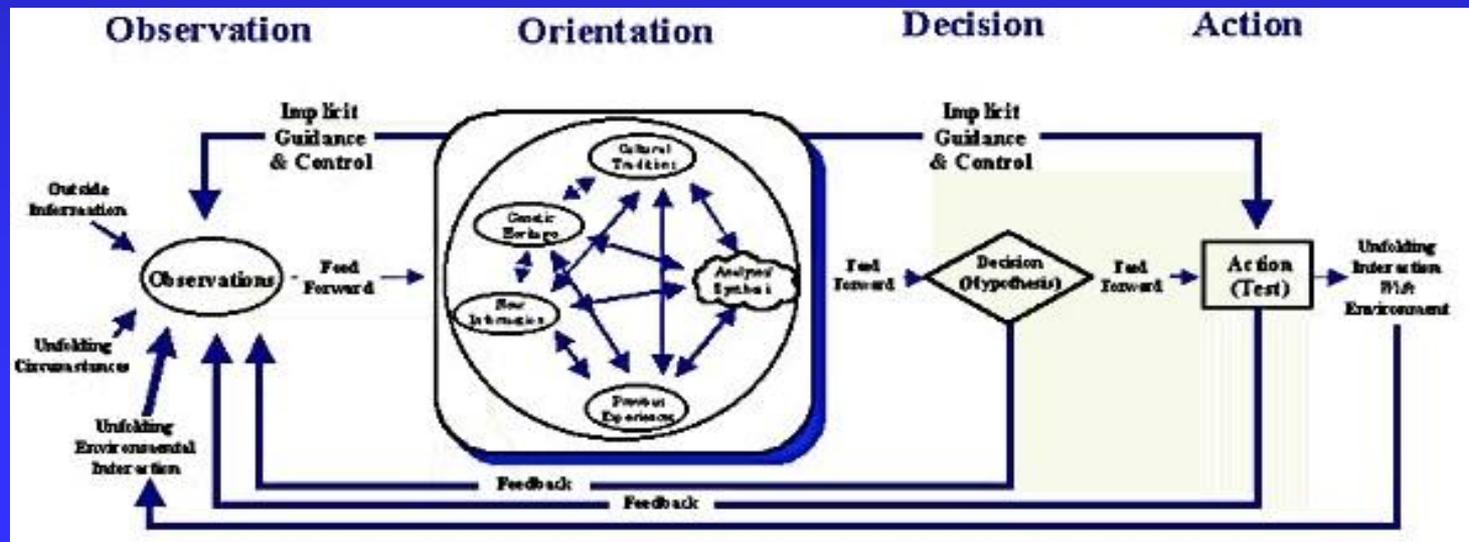
- ◆ I act **faster** or **more appropriately** than you, I exploit the predictability of your decision making (*winning the “one-up-man-ship” action game*)

# A Related Idea: Reflexive Control

- Studied by Russians and U.S. NGIC
- Characteristics of the theory:
  - ◆ I know that if you sense  $x$ , you'll do  $y$
  - ◆ I try to manipulate the environment so you'll sense  $x$
  - ◆ I formulate a move which will take advantage of  $y$

# Another Related Idea: The OODA loop and “Lock In”

- “...the victorious side will be the side that can observe-orient-decide-act more inconspicuously, more quickly, and with more irregularity as basis to keep or gain initiative as well as shape and shift main effort; to repeatedly and unexpectedly penetrate vulnerabilities and weaknesses exposed by that effort or other effort(s) that tie-up, divert, or drain away adversary attention (and strength) elsewhere”



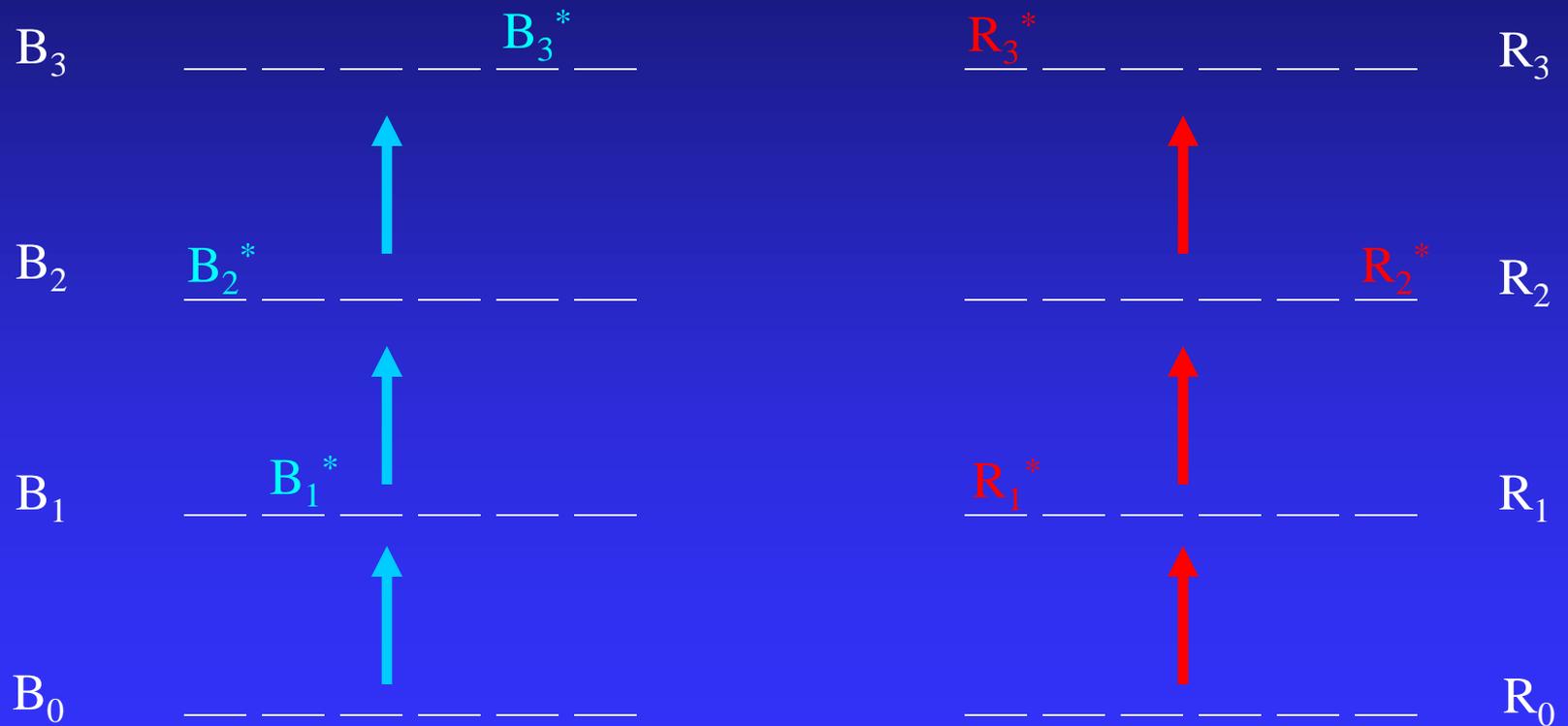
# Two ways to address the topic of coevolution in an analytic context:

- Simulate coevolving decision making within a single simulation run. (“recursive anticipation”)
  - ◆ How much better do the agents perform when they try to anticipate the enemy’s reaction?
  - ◆ What if the agents assume the enemy is doing the same to them???(levels of coevolution)
  - ◆ Exploring possible outcomes that the decision maker may not have previously thought of!
- Coevolve red and blue parameter sets. (evolutionary computation)
  - ◆ Goal is to understand the taxonomy of things that can happen, and why
    - competitive arms race
    - mediocre stable states
    - “red queen” dynamics
    - domination/extinction

# Evolutionary Computation

- Computational technique for finding a “good” solution to a problem with a vast, unknown, and or highly nonlinear search space
  - ◆ Not guaranteed to find optimal solution
  - ◆ Technique is inspired by the Darwinian concepts of natural selection, mutation, recombination
  - ◆ Standard Genetic Algorithm is an example
  - ◆ Usually start out with a population of randomly generated candidate solutions, and evolve them (e.g. run the algorithm) over a fixed number of generations

# COEVOLVING RED AND BLUE: A SCHEMA

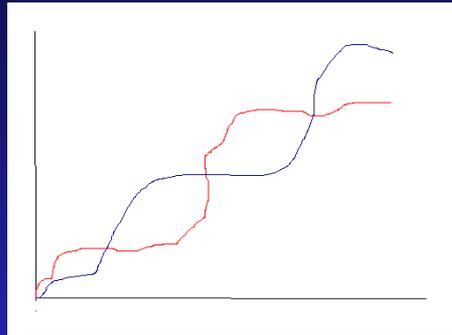


# Assumptions, Limitations, and Considerations:

- Blue and Red have knowledge of each other's "best move"
  - ◆ More realistic to model degraded knowledge to determine its effect?
- Each side evolves against one enemy threat
  - ◆ Perhaps more robust if it evolves against more than one (e.g. measure average fitness vs the top n adversary solutions)?
- Question: Are we getting better and more robust solution sets the further we let the co-evolution process continue?
- Like automating a wargame:
  - ◆ Who won and why?
  - ◆ How much did the result depend on initial conditions?
  - ◆ Achieving a good but robust solution is the goal

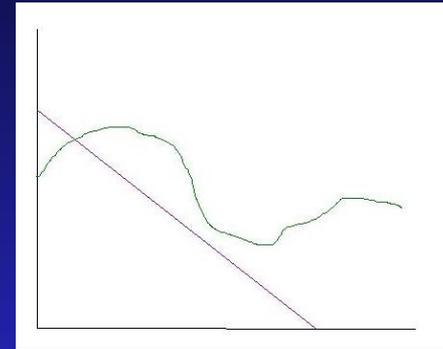
# A Few Ways to Visualize Coevolution

Normalized  
MOE



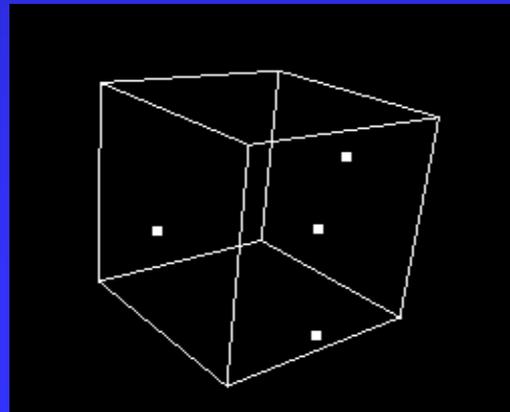
Time in Generations

Blue  
MOE



Red  
MOE

Blue Parameter  
Space



# Now We Need to Get Data!

- Using MITRE's GaN software
  - ◆ Few java classes needed to make evolutionary software run in coevolutionary mode
- Then...
  - ◆ Pick a model
  - ◆ Pick a base case
  - ◆ Pick parameters for Blue to evolve over
  - ◆ Pick parameters for Red to evolve over
  - ◆ Coevolve!

# One example of a 2-person coevolving decision making scenario...

It's "3 and 2" and the bases are loaded...

		Pitcher	
		strike zone	ball
Batter	swing	1,-1 	-2,2 
	don't swing	-1,1 	2,-2 

# Use of Evolutionary Game Theory to Study Coadaptation

- In EGT you adapt a population of batters against a population of pitchers over time, rewarding strategies that are working well, and punishing strategies that aren't.
- Not “evolving” the population, but adapting it using deterministic replicator dynamics

# Replicator Dynamics of EGT

$$G = \begin{pmatrix} 1 & -2 \\ -1 & 2 \end{pmatrix}$$

$$H = \begin{pmatrix} -1 & 1 \\ 2 & -2 \end{pmatrix}$$

$$p(0) = (.3, .7)$$

$$q(0) = (.2, .8)$$

$$F_1(p) = Gp + w_0^G$$

$$F_2(q) = Hq + w_0^H$$

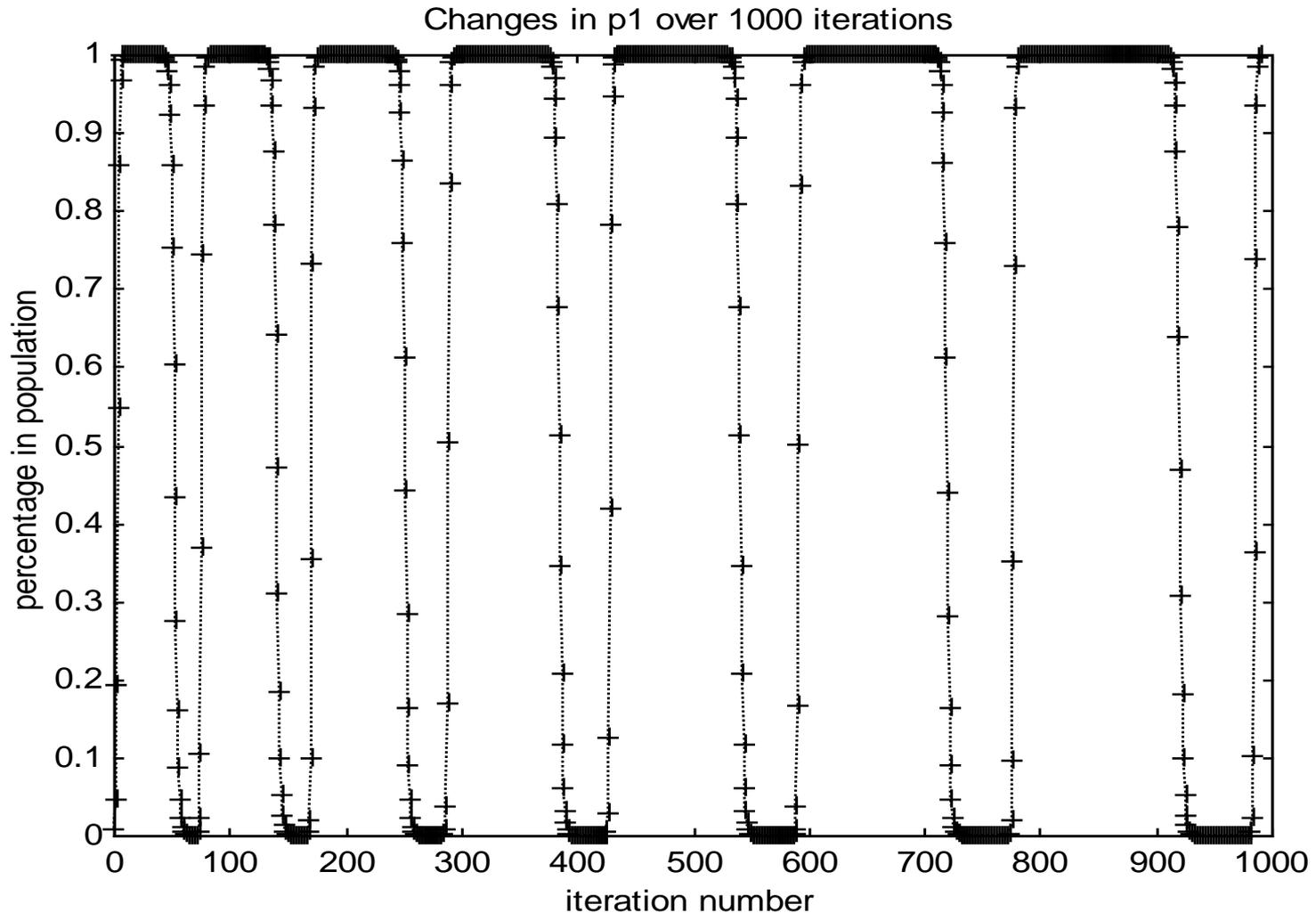
$$p(t+1) = p(t) \times \frac{F_2}{p \cdot F_2}$$

$$q(t+1) = q(t) \times \frac{F_1}{p \cdot F_1}$$

$$w_0^G = 3$$

$$w_0^H = 3$$

# What happens (to the effectiveness of swinging) over time as we coevolve the best batters and pitchers?



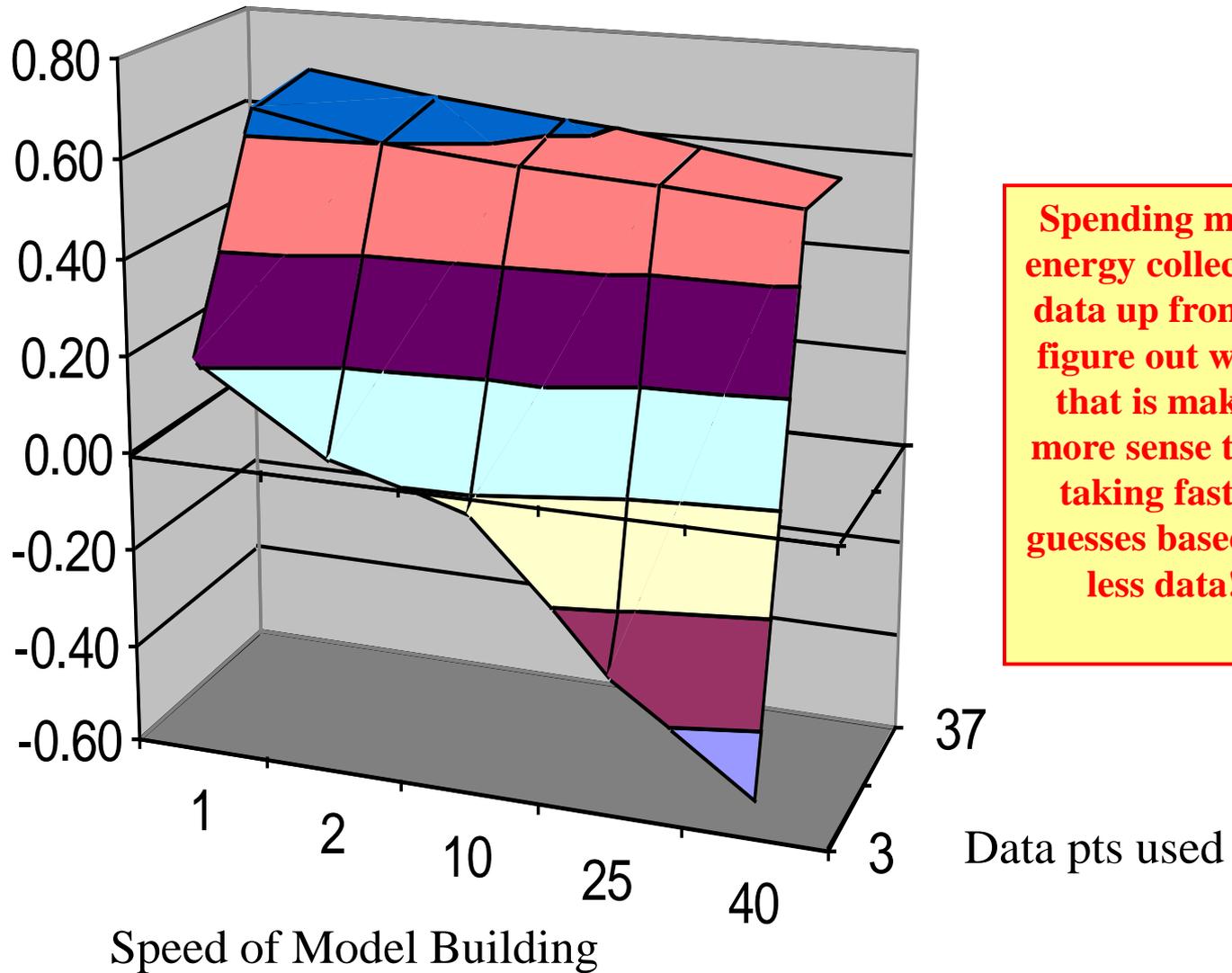
# What's Happening?

- As swinging gets more attractive for Batter, throwing “outside” becomes more attractive for Pitcher
- As throwing outside becomes more effective for Pitcher, swinging becomes less attractive for Batter
- The ‘cycle of coevolution’ continues indefinitely
- The transition happens quickly!
- Food for Thought: Can I (the batter) beat him to a transition??

# Adaptive Mental Model Building: Can the Batter “win” over time by learning and exploiting the regularities of the Pitcher’s behavior?

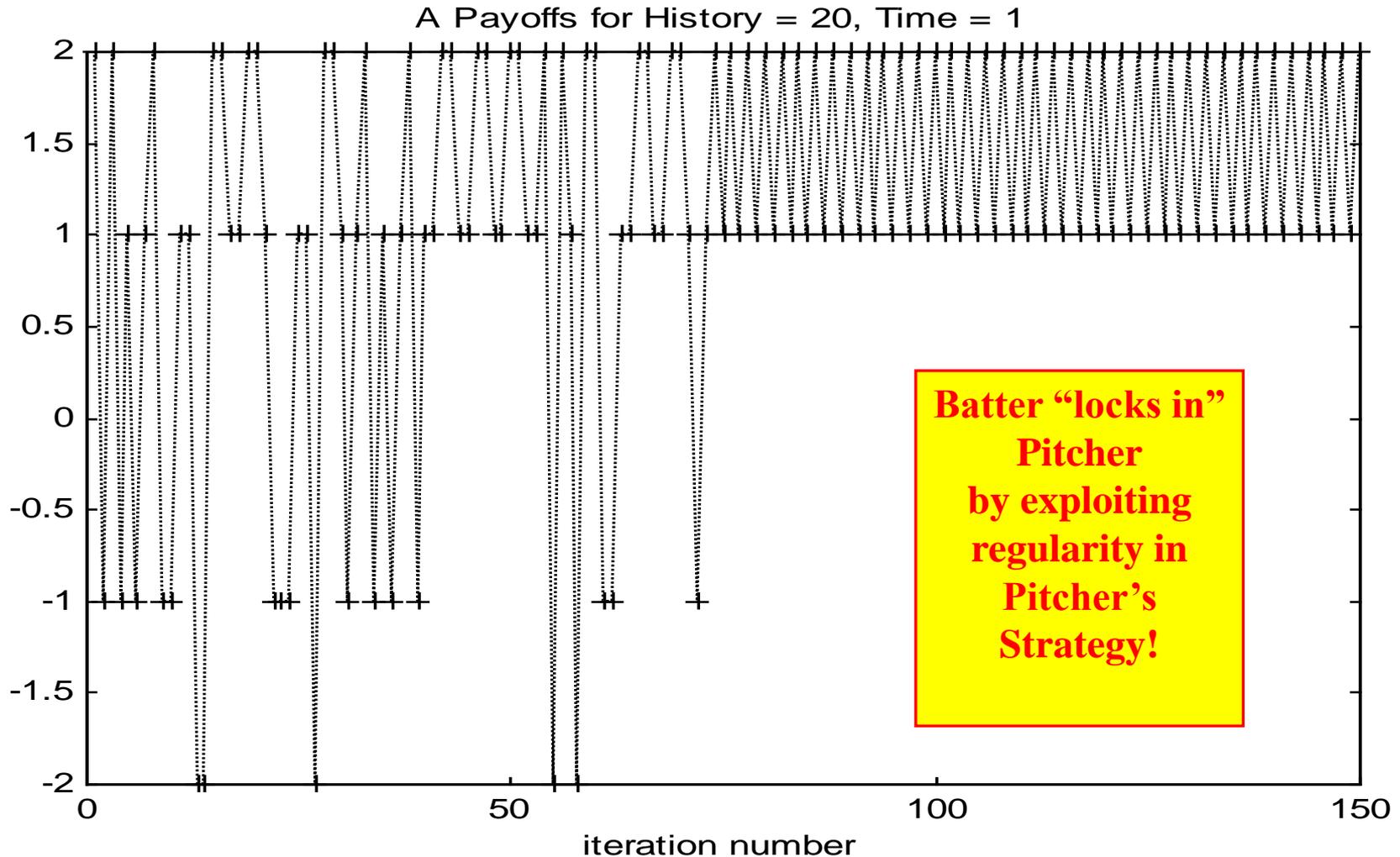
- Pitcher used a 3-history “majority rule”
  - ◆ Pitcher predicted that Batter’s next move would be the *majority* of whatever its last 3 moves were
- Batter’s mental model was a 3-layer, 3-input-node neural network
  - ◆ **Speed** variable: how often the net was retrained using the latest data (note: A uses the ‘last’ net to make predictions when not retraining)
  - ◆ **History** variable: how many data points were used to train the net
    - ◆ A data point consists of:
      - Input: Batter(n-4), Batter(n-3), Batter(n-2)
      - Target: Pitcher(n-1)
  - ◆ Note: we are endowing Batter with insight here... e.g. it knows that Pitcher is using a model which takes into account Batter’s last 3 moves... and it is using the neural net to ‘discover’ what that mental model is

# Average Batter Payoff (Normalized)



**Spending more energy collecting data up front to figure out what that is makes more sense than taking faster guesses based on less data!**

# What are the dynamics of a ‘good’ run for Batter? (Batter’s Payoffs)



# What's the Moral of the Baseball Story?

- The concept of coevolution presented here describes the dynamics of the race to win the coevolutionary race (by outthinking and/or outdoing)
- Whoever adapts quicker and better wins
  - ◆ Effectiveness of mental model building: **how can we build software that helps us learn from past history??**
  - ◆ **Is there such a thing as a “too fast” OODA loop??**
  - ◆ **Tradeoff between amount of information being collected (through the sensors) and speed of decision??**
- Situational Awareness/Feedback is important... it's what enables you to adapt to the environment (You can't adapt to what you can't perceive!)

# “The Moral” cont.

- Want to exploit without being exploited
  - ◆ Maybe not use all the info you have??
  - ◆ Maybe not let the adversary know you know they broke your code??
  - ◆ Role of deception??
- Role of stochastic vs deterministic decision making
  - ◆ Tradeoff between the payoff of “going for the gold” and the risk involved (stochastic is safer, but deterministic offers the chance for the consistently higher payoff)??
  - ◆ e.g. *If you’re reasonably confident* that you can out-think or out-do the other guy, there’s a bigger payoff for you in deterministic decision making... but you’d be better off sticking to the mixed nash stochastic distribution *if you’re not*
  - ◆ Can use agent-based models to try to assess the risk and payoff

# Deterministic vs Stochastic Decision Making

<i>MOE = (red killed, blue killed)</i>				
	Det Value-Driven	Stoch (t1)	Stoch (t0andt1)	Stoch (all)
blue good and stable	24,3	17,24	19,22	18,6
blue good and unstable	23,17	18,15	23,19	8,3
blue bad and unstable	20,19	17,9	17,9	13,5
blue bad and stable	1,25	10,24	8,20	6,9

<i>MOE = (# blue alive) + (# red dead)</i>				
	Det Value-Driven	Stoch (t1)	Stoch (t0andt1)	Stoch (all)
blue good and stable	46	18	22	37
blue good and unstable	31	28	29	30
blue bad and unstable	26	33	33	33
blue bad and stable	1	11	13	22

VERY PRELIMINARY observations from only one scenario having been explored:

- doing well -> stick with deterministic “optimal” value-driven decision making
- doing poorly -> switch to stochastic decision making to try to “get out of the rut”
- deterministic decision making led to more kills, but also more friendlies dead while stochastic decision making led to killing less, but also being killed less

GAUSS

# What is Gaussing?

- Translate an input file for Model A into an input file compatible with Model B, while attempting to retain the *fundamental elements of behavior*
- The problem seems similar whether the two models are distillations or higher-resolution conventional simulations

# Why would we Gauss?

- Option 1: Doing a “verification check” using Model B on a scenario/study accomplished using Model A
  - ◆ Are the same (similar) parameters dominant?
  - ◆ Do similarly interesting behaviors occur?
  - ◆ If neither of the above is true... not necessarily bad... but can we figure out why? (insight!)
- Option 2: Creating a Federation of Models A and B (and maybe others)
- Option 3: Using Model B as a playback tool for a scenario developed and studied using Model A
- Other ideas??

# A Theory of “Gaussing”:

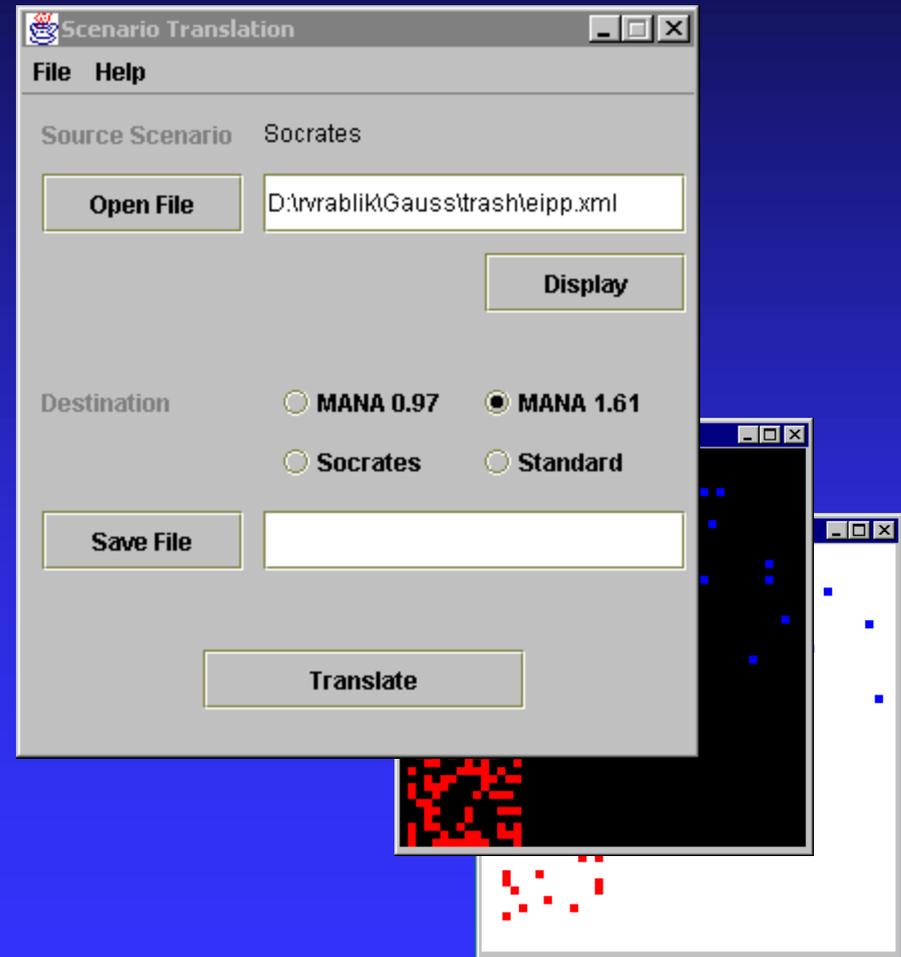
- Attempting behavior translation to the extent possible, vice “strict literal” translation
  - ◆ Being done on a case by case basis – detailed documentation provides “the how”
  - ◆ What are the fundamental elements of behavior??
    - ◆ Attack vs Defend
    - ◆ Control of an AO / Terrain Objective
    - ◆ Unit integrity
    - ◆ ...
  - ◆ If there are parameters in Model B that do not map to features in Model A
    - ◆ provide default values
    - ◆ ask the user to provide values

# Gauss Translation Tool

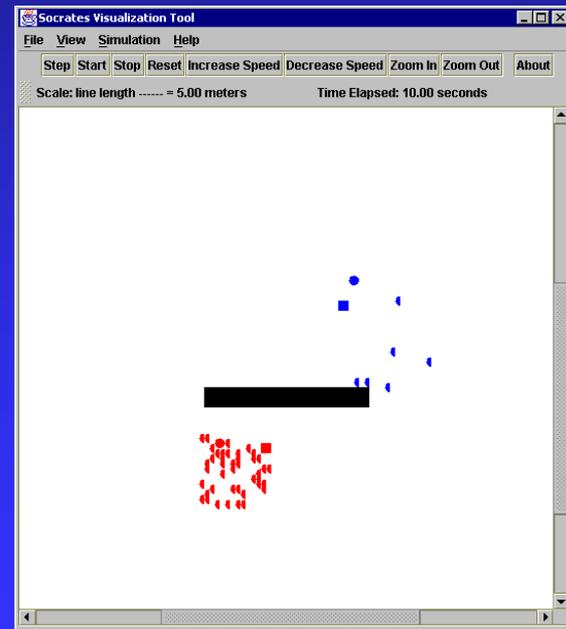
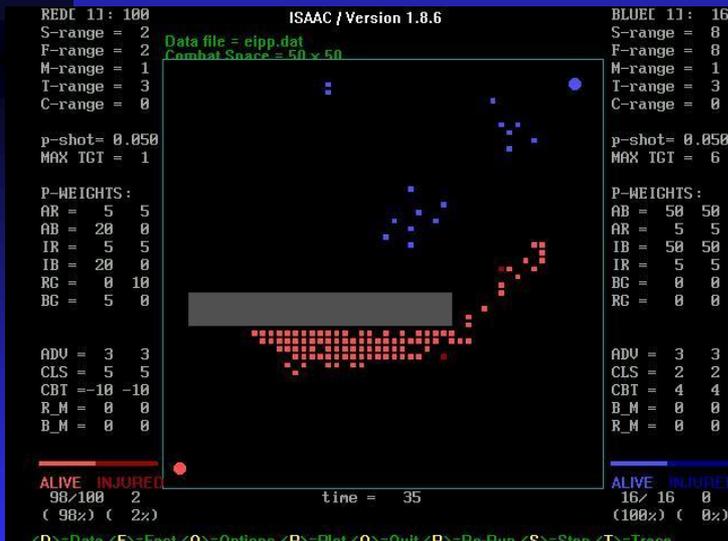
A Java based application designed to

- ◆ aid exploration of tactics and behaviors
- ◆ allow for verification of behaviors across levels of verisimilitude

by allowing translation of scenarios from one distillation/simulation format to another

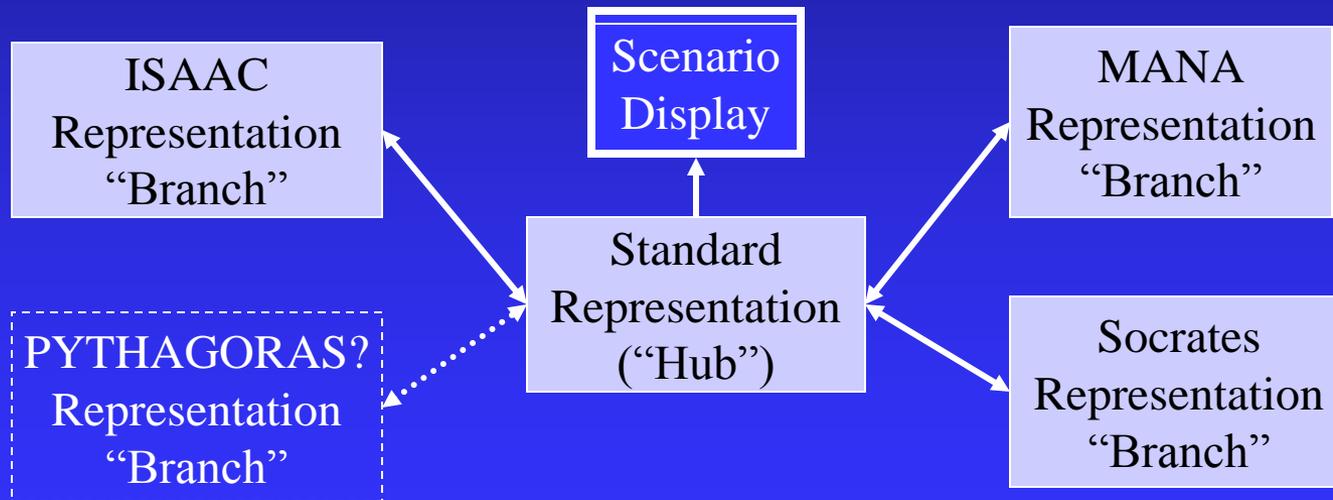


- Already have: ISAAC to MANA
- Working on: ISAAC to SOCRATES; SOCRATES TO ISAAC; MANA to ISAAC; (TBD) to PYTHAGORAS

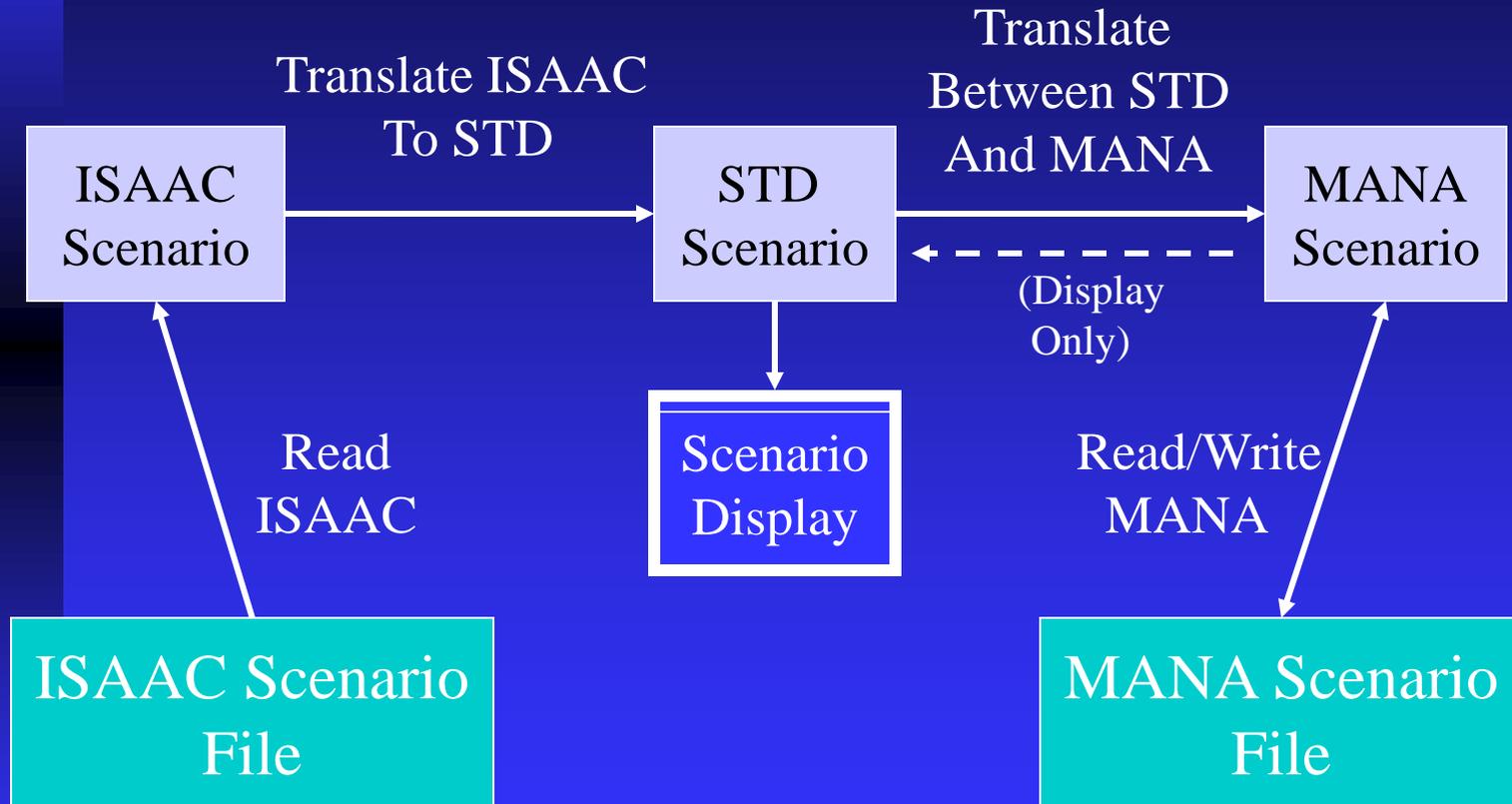


# “Hub” Architecture

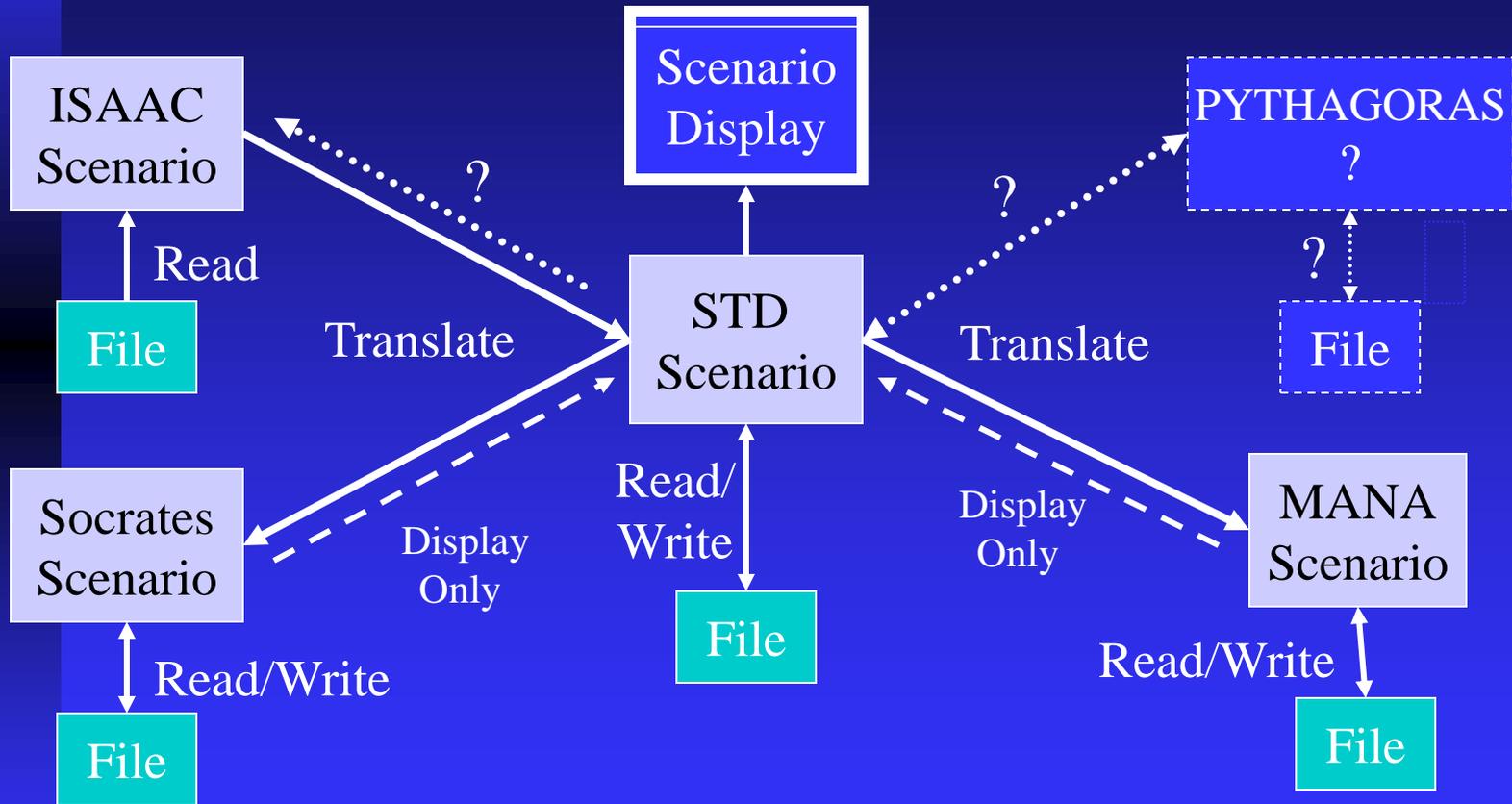
- All translations performed in two steps
  - ◆ Distillations translated to standard format
  - ◆ Standard format then translated to destination format



# Gauss Translation Tool, Version 1.0



# Proposed Translation Tool, Version 1.2



# QUESTIONS



[mmcdonald@sito.saic.com](mailto:mmcdonald@sito.saic.com)