



Evidence Based Research, Inc.

# Distributed Algorithms for Resource Allocation Problems

Mr. Samuel W. Brett

Dr. Jeffrey P. Ridder

Dr. David T. Signori Jr

20 June 2012



# Outline

- Survey of Literature
  - Nature of resource allocation problem
  - Comparison of Distributed Solution Algorithms
- ANACONDA
  - Key concepts & applications
  - An illustrative application
- Summary



# Major Challenge Faced by Decision Makers Allocating Resources

- Answering difficult questions
  - What and how many resources do I need to accomplish my objectives?
  - How do I ensure that these resources are available when I need them?
  - How do I allocate or schedule my resources?
- In complex environments
  - High Dimensionality
  - Dynamic
  - Non cooperative
  - Uncertain

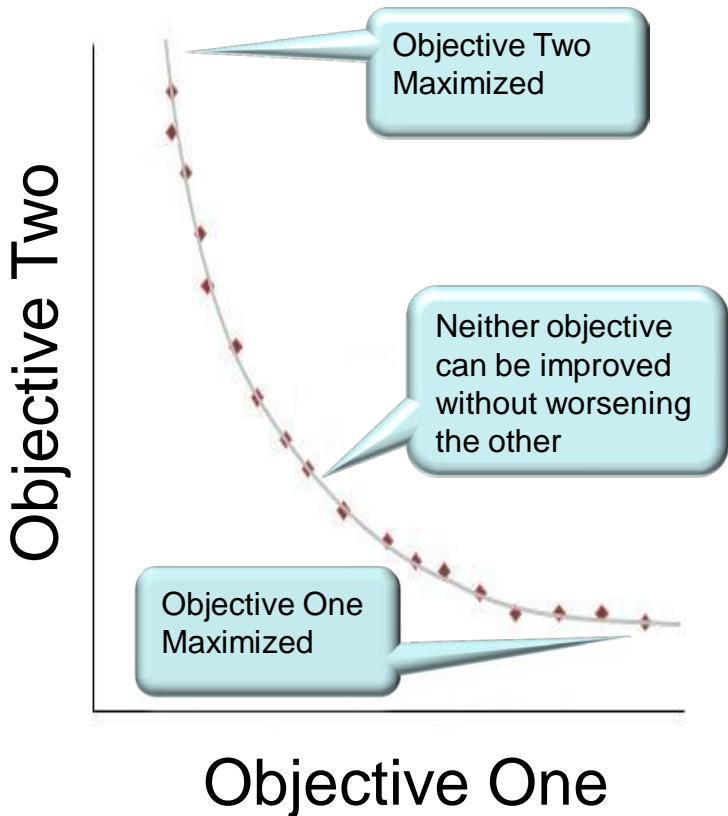


# Typical Resource Allocation Problems

- Scheduling patients in hospital
  - Minimize wait time with changes due to unpredictability of specialist availability
- Scheduling jobs to machines
  - Assign jobs to machines when new jobs can arise or machines can break at any time
- Supply chain management
  - Manage inventory and routing in an uncertain environment
- Inter-unit demand estimation
  - Determine node-to-node data flows subject to constraints on loads at nodes and relative loads for links

These problems have many military C2 counterparts

# Nature of the Resource Allocation Optimization Problem



- **Multi-objective**
- **Over constrained**
- **Tradeoffs required**



# The Algorithmic Challenge & A Pragmatic Solution Strategy

- Challenge: no existing method to compute an optimal solution in a reasonable amount of time for even modestly sized problems.
  - Significantly exacerbated by change and uncertainty
- Solution Strategy:
  - Recognize that in real world applications optimality is rarely needed
  - Accept solutions that are better than those that are manually produced

Historically Analysts have often been forced to manual solution methods



# The Distributed Constraint Optimization Problem (DCOP)

- Problem: multi-objective optimization that constrains feasible region of full solution.
- Approach: partition problem into agents; each having a set of variables and constraints to manage as well as local optimization criterion.
- Goal: find a feasible solution with the highest ranking by all agents determined by some solution ordering.

Overcome the limitations of centralized algorithms when dealing with large problem spaces: lack of scalability and poor local solutions



# Comparison of DCOP Algorithms

## Using Completeness and Complexity Attributes

Algorithm	Completeness	Time Complexity	Memory Complexity	Size of messages	Number of Messages
Adopt	Complete	Exponential	Polynomial (or any space)	Constant	Exponential
Distributed Breakout	Incomplete	Polynomial per timestep	Polynomial	Constant	Polynomial per timestep
DPOP	Complete	Exponential	Exponential	Exponential	Polynomial
NCBB	Complete	Exponential	Polynomial (or any space)	Constant	Exponential
ANACONDA	Incomplete	Polynomial per timestep	Polynomial	Constant	Polynomial per timestep

**Completeness:** will find solution or determine no solution exists  
**Complexity:** Constant << Polynomial Rate << Exponential Rate



# Advantages of ANACONDA\*

- **Computationally inexpensive**
  - It requires memory proportional to the number of variables and constraints and sends messages of minimal size.
- **Solves problems with an inordinate number of variables, often on the order of hundreds of thousands, in minutes as opposed to hours or days.**
  - Has been successfully applied in inter-unit demand estimation for large realistic forces.
- **When combined with user friendly HCI, it enables users to explore solutions along a Pareto front and find appropriate solution**
  - With constraints categorized by differing objectives, users can adjust the importance of a given objective in real time.
- **Ability to handle continuous as opposed to discrete variables.**
  - Constraints and errors calculated by agents using variables on continuous domains.

\* AutoNomous Agent Constraint OptimizationN Distribution Algorithm (ANACONDA)



# Outline

- Survey of Literature
  - Nature of resource allocation problem
  - Comparison of Distributed Solution Algorithms
- ANACONDA
  - Key concepts & applications
  - An illustrative application
- Summary



# Highlights of ANACONDA

- **Based on similar premises as particle swarm optimization**
- **Uses agents to calculate constraints and errors**
- **Agents act to minimize this error**
- **Anaconda proceeds as a simulation**
  - Agent variables initialized to starting values then iteratively changed
- **At each time step:**
  - Constraint agents recalculate constraint errors
  - Variable agents change their value to minimize errors on constraints affecting them

Inputs	Outputs
Value of demand variable	Value minimizing errors
Error constraints	Magnitude of errors

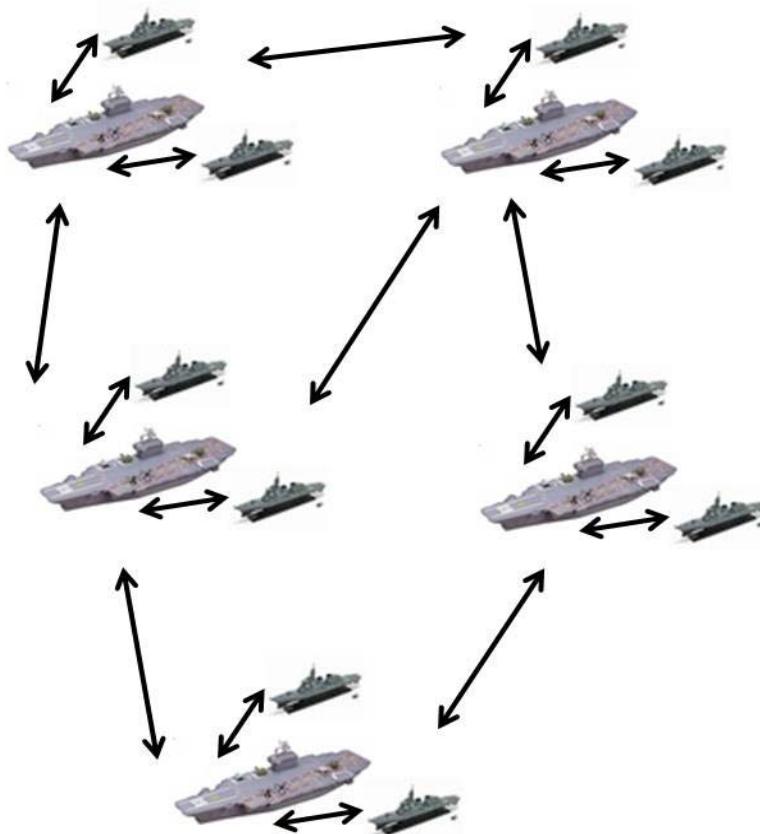


# Applications of ANACONDA

Application	Agents	Objectives	Constraints
Hospital Scheduling	<ul style="list-style-type: none"><li>• Doctors</li><li>• Patients</li></ul>	<ul style="list-style-type: none"><li>• Most efficient time</li><li>• Most Convenient time</li></ul>	<ul style="list-style-type: none"><li>• Availability</li></ul>
Job Shop Scheduling	<ul style="list-style-type: none"><li>• Machine</li><li>• Factory Managers</li></ul>	<ul style="list-style-type: none"><li>• Max productivity</li><li>• Min wait time given uncertainties</li></ul>	<ul style="list-style-type: none"><li>• Machine Function</li><li>• Time</li><li>• Availability</li></ul>
Supply Chain Management	<ul style="list-style-type: none"><li>• Routing</li><li>• Warehouse</li><li>• Customers</li></ul>	<ul style="list-style-type: none"><li>• Best Routes</li><li>• Uncertain demand</li><li>• Timely Delivery</li></ul>	<ul style="list-style-type: none"><li>• Route Availability</li><li>• Info Availability</li><li>• Warehse capacity</li><li>• No. of customers</li></ul>
Inter-unit Demand Estimation	<ul style="list-style-type: none"><li>• Flow agents for different types of demand</li></ul>	<ul style="list-style-type: none"><li>• Balance errors in meeting different constraints</li></ul>	<ul style="list-style-type: none"><li>• Flow conservation</li><li>• Operational Context</li></ul>

These Applications have many military C2 counterparts

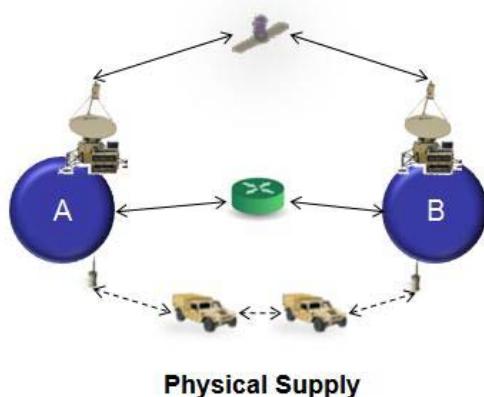
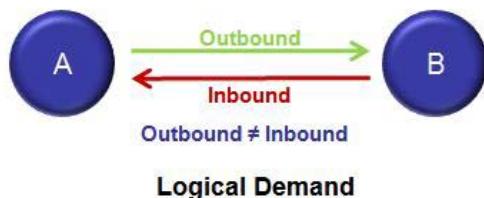
# Parsing Unit Demand To Determine Inter-unit Demand



- Estimating network demand is important
  - Build a supply architecture
  - Perform Mission Risk Assessment
  - Support acquisition decision makers
- Demand estimations include
  - Aggregate demand for each unit
  - Patterns of interaction
- There are different ways to approach this problem
  - Historical patterns (IER approach)
  - **Device based approach plus mission constraints on logical interactions**

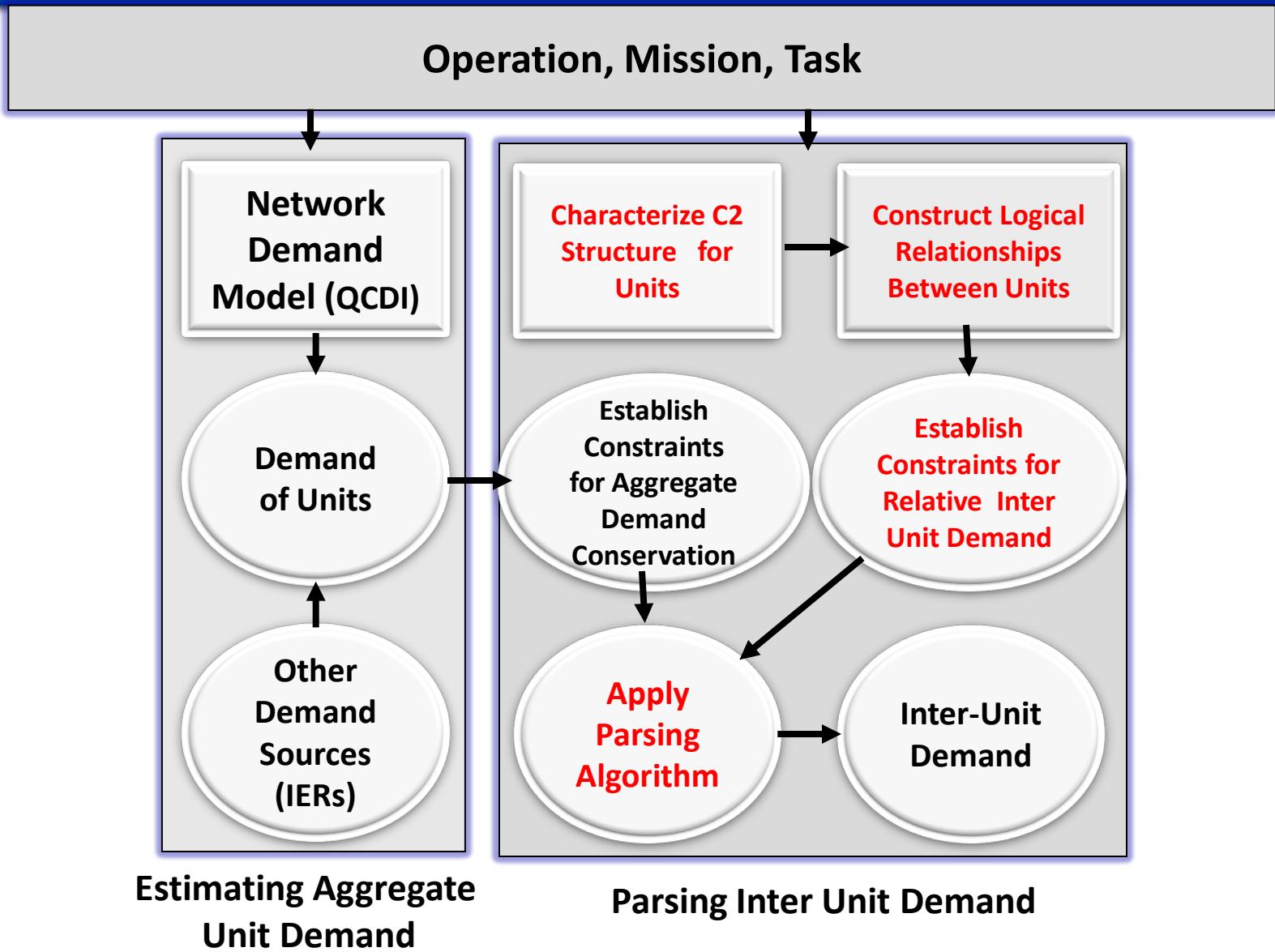
Device based approach computationally challenging

# Unit Demand vs. Inter-Unit Demand



- Unit demand provides:
  - Demand values at nodes A and B
- Parsing algorithm determines inter-unit demand:
  - A to B, B to A, etc. (inter-unit)
  - Network demand inside a unit, A to A, B to B, etc. (intra-unit)
- Parsing is supply architecture independent

# Scenario Based Demand Generation



# Unit to Inter-Unit Demand

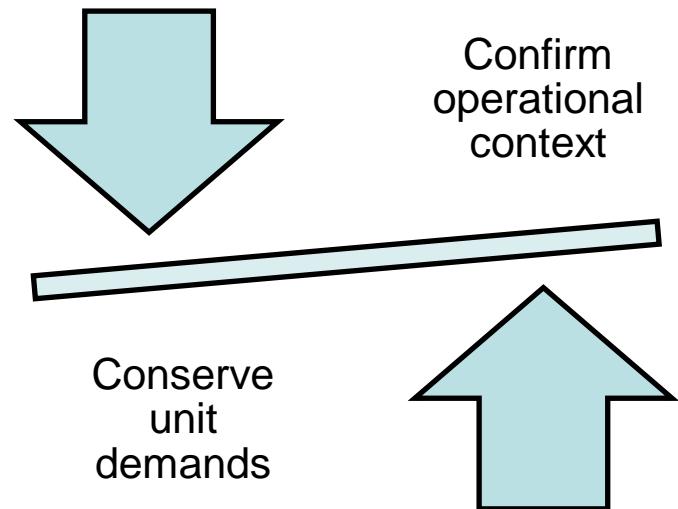
## The Mathematical Problem

### Given:

- Unit demand estimates for every unit
- Operational context for a given scenario

### Determine:

Inter-unit demands balancing:





# Role of Agents in Finding Solution

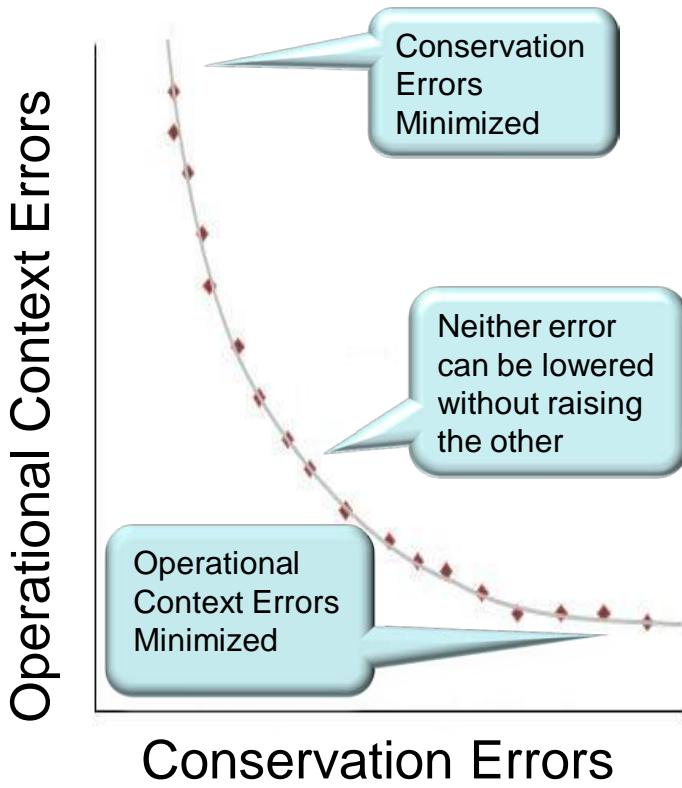
	Download Agent	Download Agent	Download Agent	Download Agent	
Upload Agent	Flow Agent	Flow Agent	Flow Agent	Flow Agent	To Other
Upload Agent	How do I change my flow value to lower my conservation errors?				To Other
Upload Agent	Flow Agent	Flow Agent	Flow Agent	Flow Agent	To Other
From Other					

Must consider current flow value, upload error from its row, and download error from its column

	Download Agent	Download Agent	Download Agent	Download Agent	
Upload Agent	Flow Agent	Flow Agent	Flow Agent	Flow Agent	To Other
Upload Agent	How do I change my flow value to lower my operational context errors?				To Other
Upload Agent	Flow Agent	Flow Agent	Flow Agent	Flow Agent	To Other
From Other					

Must consider current flow value, and flow value of all other agents on its row

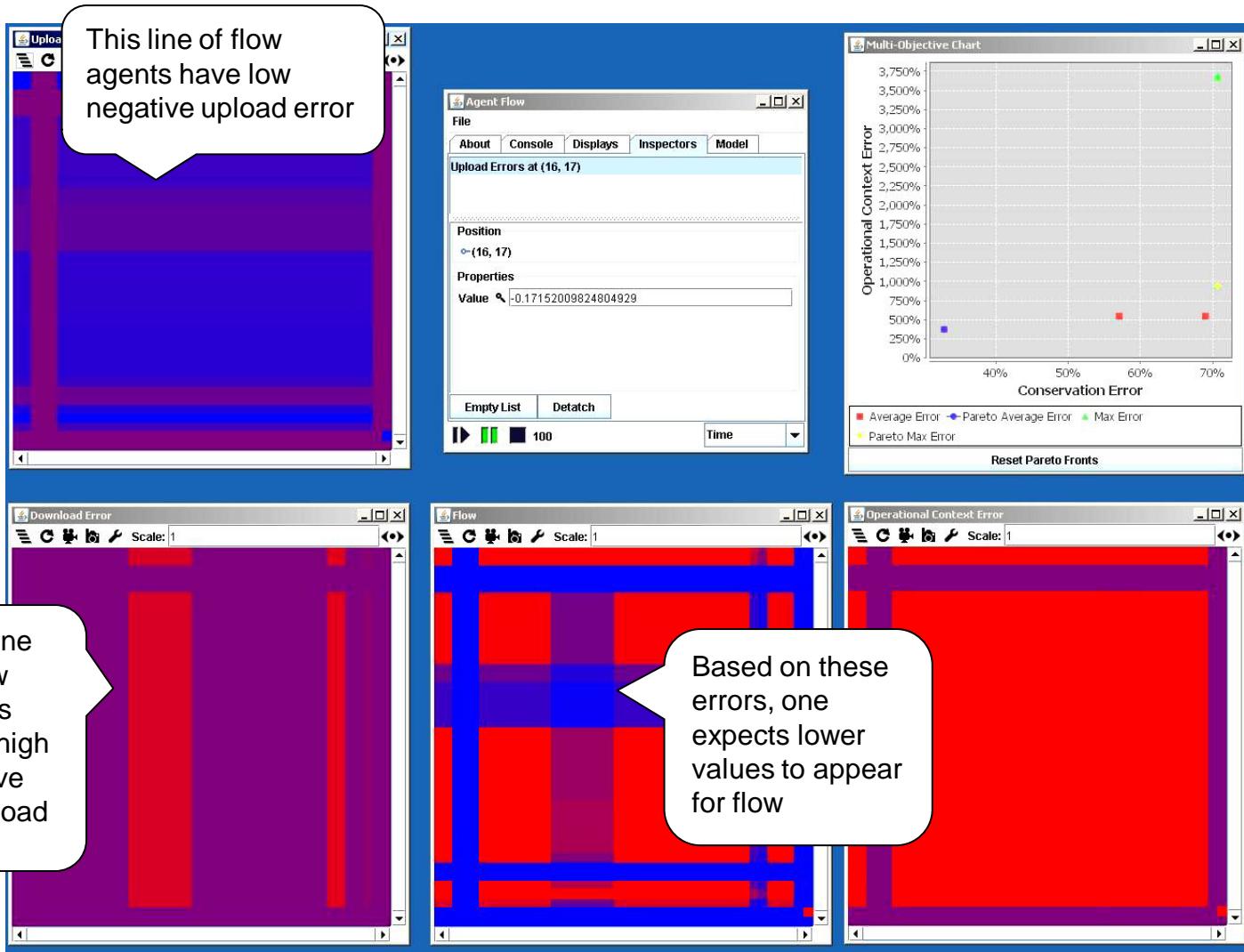
# Role of Users in Balancing Different Types of Errors



- In this application there are two objectives:
  - Flow conservation
  - Operational context
- The Pareto front can be explored interactively by users
  - balance objectives appropriately

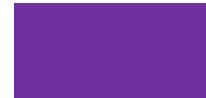


# Interactive Parsing



## Color Legend:

Moderate flow value/zero error



High flow value/ high positive error

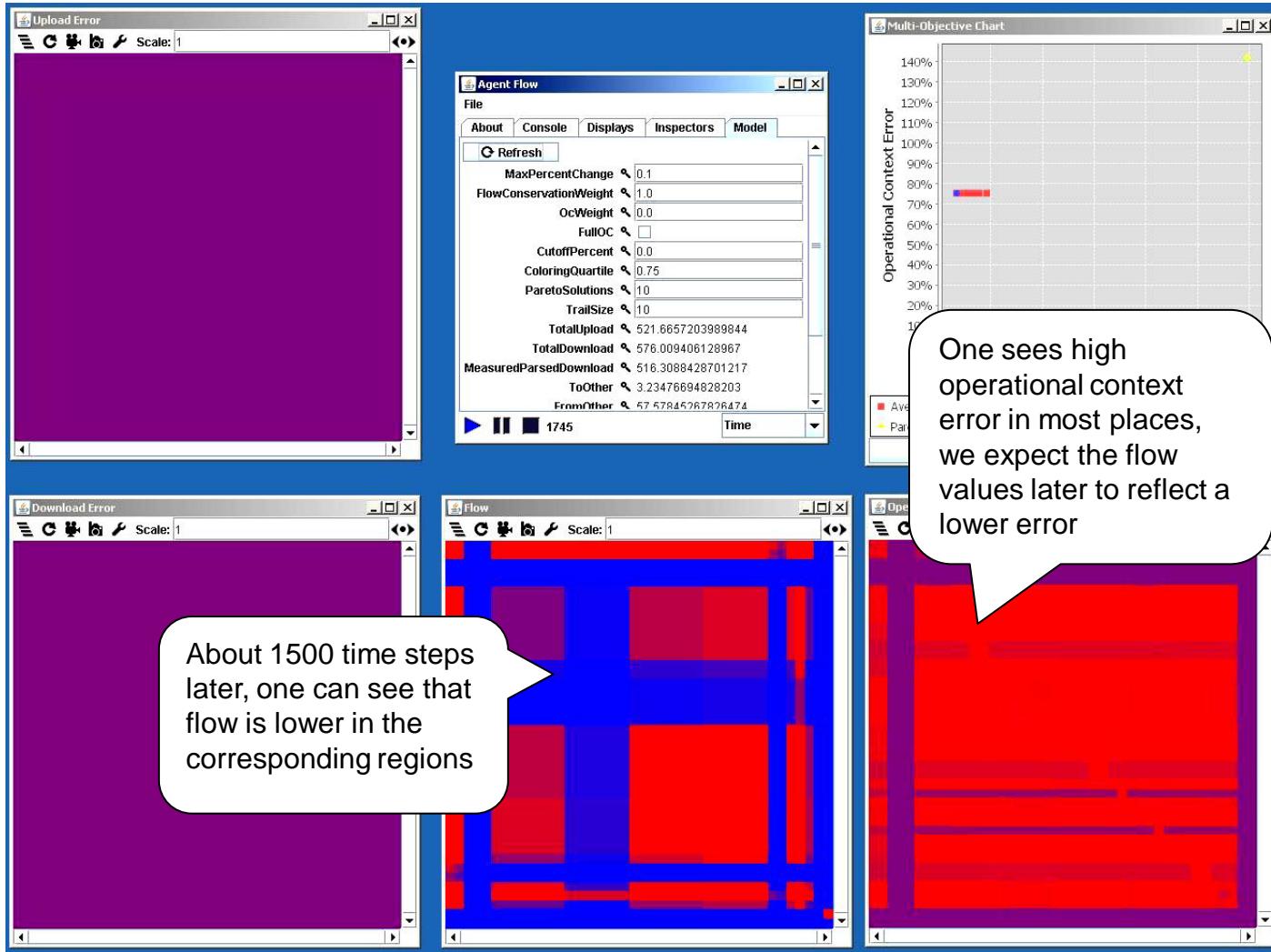


Low flow value/ high negative error





# Interactive Parsing



## Color Legend:

Moderate flow value/zero error



High flow value/ high positive error

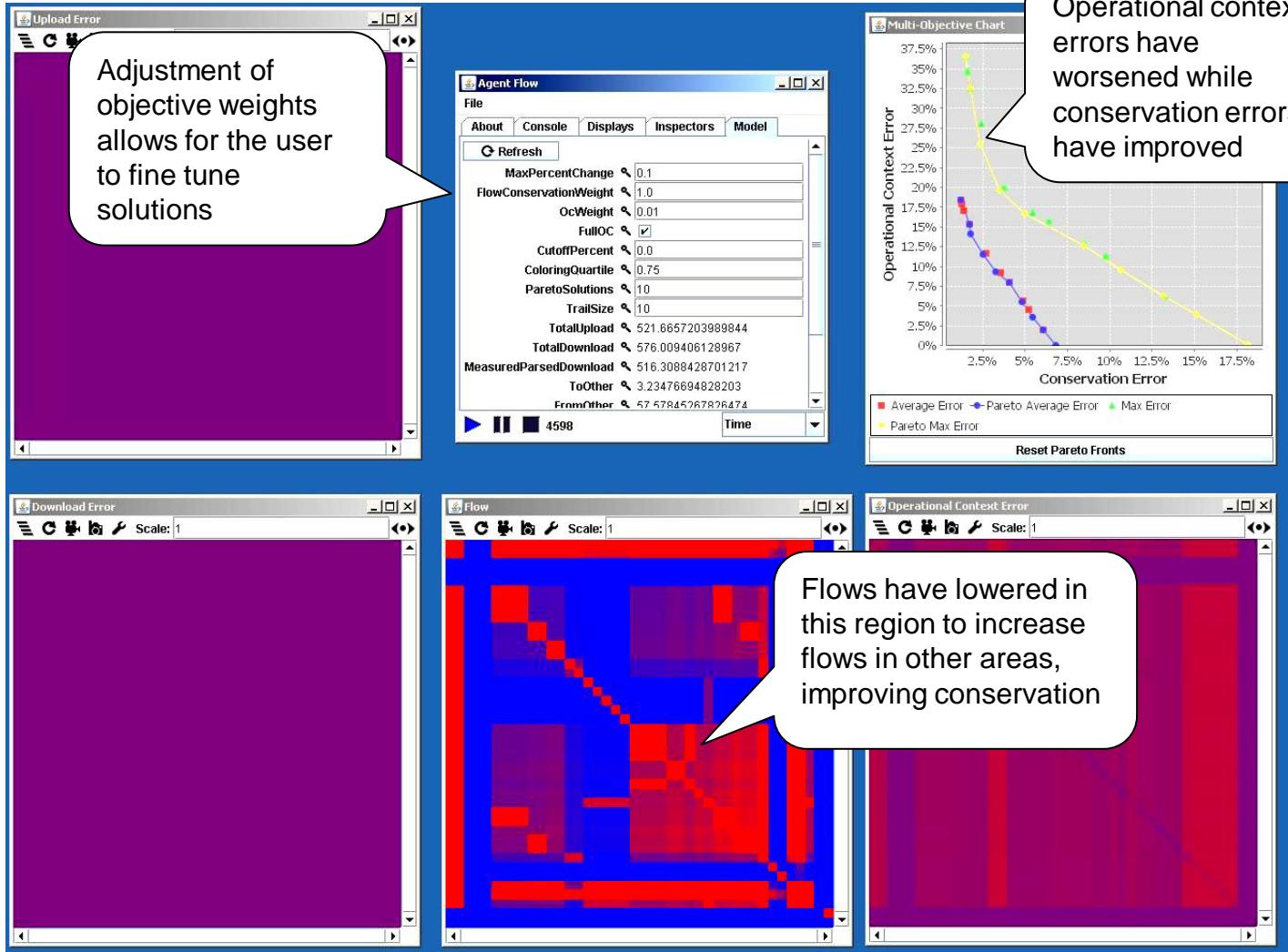


Low flow value/ high negative error





# Interactive Parsing



## Color Legend:

Moderate flow value/zero error



High flow value/ high positive error



Low flow value/ high negative error





# Scalability of Agent Based Parsing Tool

- Applied to a scenario involving almost 400 units
- Solved for about 140,000 flow variables in 10 minutes
  - Most analytical problems have fewer than 100 variables
- Used on a computer with 2 GB of RAM and 3 GHz Dual Core E8400 processor

**Error Performance consistent with accuracy of unit demand inputs  
i.e. 80% solution**



# Sample Results for Illustrative Example Generic Air Campaign for Offensive Strike

- Upload Errors
  - Maximum: -6.15%
  - Average: -5.35%
- Download Errors
  - Maximum: 16.38%
  - Average: 5.33%
- Operational Context Errors
  - Maximum: 18.7%
  - Average: 6.80%



# Summary

- Anaconda provides a repeatable method of obtaining solutions which balance competing objectives
- The algorithm runs in a short amount of time with minimal memory and CPU requirements
- The ability to interactively select the location of a solution on the Pareto front gives an analyst increased flexibility
- It has been embedded in a user friendly tool that can be used to determine inter-unit communications demand for specific scenarios
- This algorithm can be applied to a wide range of C2 resource allocation problems to obtain timely and flexible solutions



# References

## Related to Anaconda & It's application

- Algorithm and Tools for Scenario Generated Demand
  - Ridder, J., Brett, S., Burris, C., McEver, J., O'Donnell, J., Signori, D., and Schoenborn, H., "Models and algorithms for determining inter-unit network demand," *Proceedings of the SPIE Defense, Security, and Sensing Symposium* (2012).
  - Jack E. O'Donnell, Ayanah S. George, Danielle M. Wynn, Samuel W. Brett, Jerey P. Ridder, David T. Signori, and Heather W. Schoenborn, "A Flexible Tool for Scenario Analysis of Network Demand", *Proceedings of the SPIE Defense, Security, and Sensing Symposium* (2012).
- Quantitative Capability Description Increment (QCDI)
  - Burris, C., Gonzales, D., McEver, J., Porche, I., Schoenborn, H., Signori, D., and Sudkamp, S., *Quantitative Capability Delivery Increments: A Novel Approach for Estimating and Assessing DoD Future Network Needs*, 15<sup>th</sup> ICCRTS
  - Burris, C., Gonzales, D., McEver, J., Signori, D., Smeltzer, M., Schoenborn, H., *Quantitative Capability Delivery Increments: A Novel Approach for Assessing DoD Network Capability*
- Network Mission Assurances NMA
  - McEver, J., Burris, C., Signori, D., and Schoenborn, H. (2010 June 23) *Network Mission Assurance: Estimating Operational Risk Associated with Network Performance*, 78<sup>th</sup> MORS Symposium: Quantico, VA.
  - Burris, C., McEver, J., Schoenborn, H., and Signori, D. (2010 May 21) *Steps Toward Improved Analysis for Network Mission Assurance*

**For more information contact :**

Jeffrey Ridder: Ridder@EBRInc.com

David Signori: Signori@EBRInc.com



# Back Up Slides

# Some DCOP Solution Algorithms

- **Asynchronous Distributed Optimization(ADOPT)**
  - Backtracking based on local info to approach opt
- **Distributed Breakout**
  - Communication only among neighboring agents to reduce time in achieving synchronization
- **Distributed Pseudo-tree Optimization Procedure(DPOP)**
  - Extension of the Sum-Product algorithm with the nodes as variables
- **No Commitment Branch and Bound(NCCB)**
  - Partitioning to enable asynchronous operation; uses greedy search for initialization and values based on logical ordering
- **Autonomous Agent Constraint Optimization Distribution Algorithm(Anaconda)**
  - Each variable computed by an agent trying to locally minimize its errors subject to different constraints.



# Addendum

## Theory with Key Equations



# ANACONDA Variables

- Variables were originally adapted for use with continuous, real variables
- The update equations for a variable's value is as follows:

$$v^{t+1} = v^t + r * \sum_{\substack{E \subseteq C \text{ s.t.} \\ \cap_{E \subseteq C} E = \emptyset}} w^E * \delta^E \quad \delta^E = -\text{sgn}\left(\sum_{c \in E} \varepsilon^c\right) * \sqrt{\frac{\sum_{c \in E} (\varepsilon^c)^2}{|E|}}$$

- Where:
  - $v^t$  is the variable value at time  $t$
  - $r$  is the relaxation rate or maximum amount of change of a value in the range  $[0,1]$
  - $C$  is the set of all constraints and each  $E$  is a type of error constraints acting on  $v$
  - $w^E$  is the user set weight of  $E$  in the range  $[0,1]$
  - $\delta^E$  is the amount by which  $v$  should change for set  $E$
  - $\varepsilon^c$  is the error on a constraint  $c$



# ANACONDA Constraints

- Constraints were originally adapted for continuous, real variables
- A constraint takes in variables from the problem and calculates some measure of error

Example Constraint	Calculation
Target Value	Calculate error by percent difference of constraint value from target value
Soft Max Value	Calculate error by percent different of constraint value from target value only if above the max value
Relative Value	Calculate error by percent difference of variable from relative variable value



# Definition of Terms For Agent Based Approach

- Flow of agent at time step n:
- Flow conservation and operational context weights:
- Change in flow due to flow conservation and operational context:
- Multiplier for flow conservation and operational context:
- Relaxation rate or max change on a time step:

$$\begin{array}{ll} f^n & \\ \omega^{FC} \quad \omega^{OC} & \\ \delta f^{FC} \quad \delta f^{OC} & \\ m^{FC} \quad m^{OC} & \\ r & \end{array}$$

# Formulas for Agent Error

- Flow conservation error for an inter-unit demand

$$\varepsilon_{ij}^{FC} = \frac{\varepsilon_i^U + \varepsilon_j^D}{2}$$

- Operational context error

for an inter-unit demand

$$\varepsilon_{ij}^{OC} = \begin{cases} \varepsilon_i^{Intra}, & \text{if } i = j \\ \sqrt{\frac{\sum_{kl \neq ij} \varepsilon_{ij,kl}^{Inter^2}}{M}}, & \text{if } i \neq j \end{cases}$$

- Average operational context demand

Assume there are M specified inter-unit demand ratios. Only those are summed if  $i \neq j$ . Notice this value is root mean squared of all inter-unit errors.

$$\widetilde{\varepsilon^{OC}} = \begin{cases} \varepsilon^{Intra}, & \text{if Intra} \\ \sum_{i=1}^M \varepsilon^{Inter}, & \text{if Inter} \end{cases}$$



# Formulas for Updating an Agent

$$f^{n+1} = f^n + \omega^{FC} \cdot \delta f^{FC} + \omega^{OC} \cdot \delta f^{OC}$$

$$m^{FC} = \min(1, |\varepsilon^{FC}|)$$

$$\delta f^{FC} = \begin{cases} f^n \cdot (1 - m^{FC} \cdot r) - f^n & \text{if } \varepsilon^{FC} > 0 \\ \frac{f^n}{1 - m^{FC} \cdot r} - f^n & \text{if } \varepsilon^{FC} < 0 \end{cases}$$

$$m^{OC} = \min(1, \varepsilon^{OC})$$

$$\delta f^{OC} = \begin{cases} f^n \cdot (1 - m^{OC} \cdot r) - f^n & \text{if } \widetilde{\varepsilon^{OC}} > 0 \\ \frac{f^n}{1 - m^{OC} \cdot r} - f^n & \text{if } \widetilde{\varepsilon^{OC}} < 0 \end{cases}$$



# Addendum

## Potential Applications of ANACONDA



# Applications

- Original domain was in communication networks
- Other applicable network domains:
  - Supply chain networks
  - Transportation networks
  - Computer networks
- Can apply to other single objective or multi objective optimization problems such as resource allocation



# ANACONDA Appointment Scheduling

- Consider a hospital with two kinds of variable agents:
  - Doctor agents
  - Patient agents
- Doctor agents try to give themselves the best time slot while patients want the most convenient appointment
- Constraints include doctor availability, patient availability, etc

- Consider the factory to have several different kinds of variable agents:
  - Machine agents
  - Factory manager agents
- Machine agents try to maximize the amount of time they are productive
- Factory manager agents try to minimize wait time for products while accounting for uncertainties imposed by machines and new orders
- Constraints include which parts machines can produce, time constraints, machine availability/failure, etc.



# ANACONDA Supply Chain Management

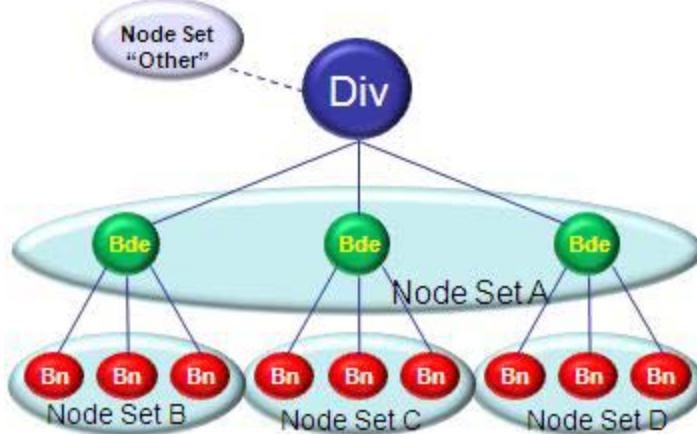
- Consider the supply chain to have several kinds of agents:
  - Routing agents
  - Warehouse agents
  - Customer agents
- Routing agents try to find optimal (distance, time, cost) routes for delivering products
- Warehouse agents try to manage their supply dealing with shipments in and out from routers along with uncertain demand
- Customer agents want products in a timely fashion or may switch to another competitor
- Constraints include available routes, information available to forecast demand, warehouse capacity, number of customers in the system, etc



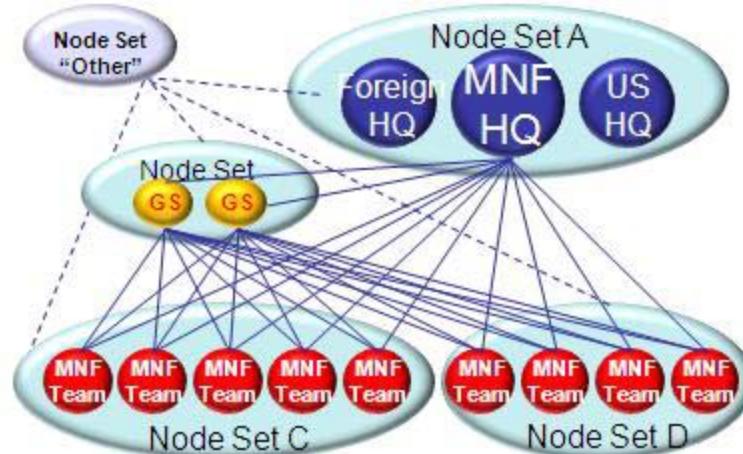
# ANACONDA Inter-Unit Demand Estimation

- Consider the inter-unit demand parser to have a single type of variable agent:
  - Flow agents
- There are two kinds of constraint agents:
  - Flow conservation constraints
  - Operational context constraints
- Flow agents try to balance their errors in the dimensions of flow conservation and operational context

# Node-Link Sets



Hierarchical Example

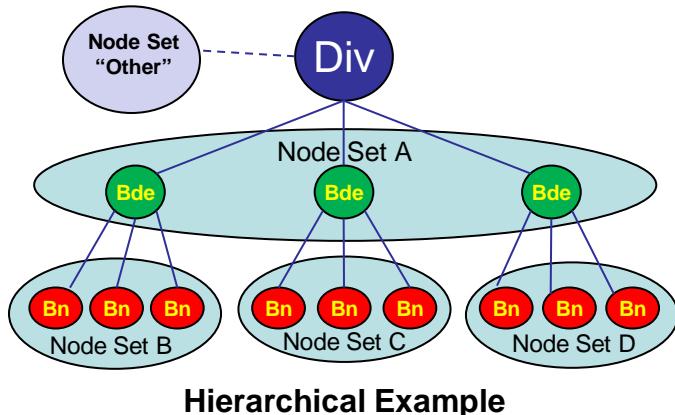


Functional/Geographical Example

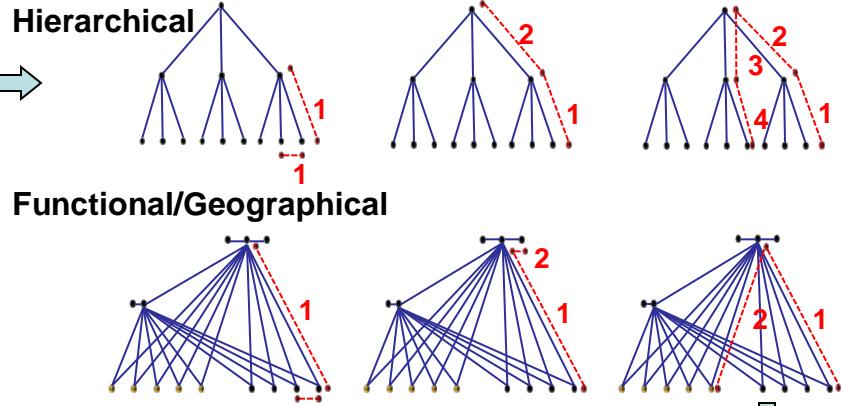
- Nodes defined by groups of units at a level of aggregation appropriate for operational and analytical context .
- Structure based on one or a combination of the following: function, location and hierarchy
  - The node-link sets influence the patterns of interaction among units

# Low Resolution Operational Constraints

## Node-Link Sets



## Organizational Distance



## Link Weighting Function

$$Weight = \begin{cases} \frac{d}{OD^2} + \frac{c}{OD} + \frac{s}{SPL} & \text{when } OD \leq SPL \\ \frac{d}{OD^2} + \frac{c}{OD} & \text{when } OD > SPL \end{cases}$$

Where:

$d$  = % Direction

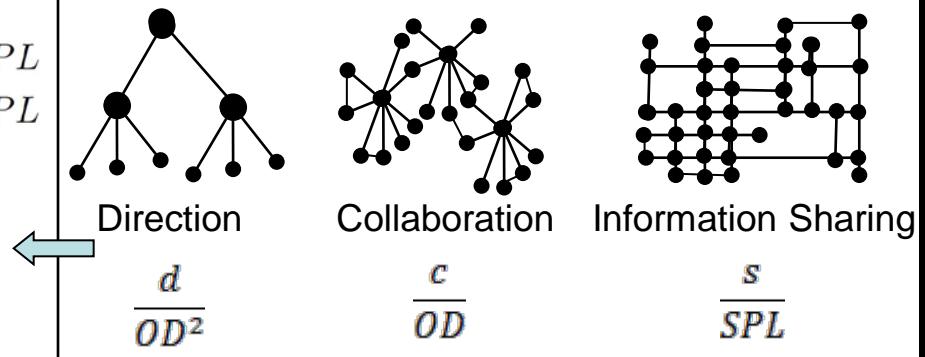
$c$  = % Collaboration

$s$  = % Information Sharing

$OD$  = Organizational Distance

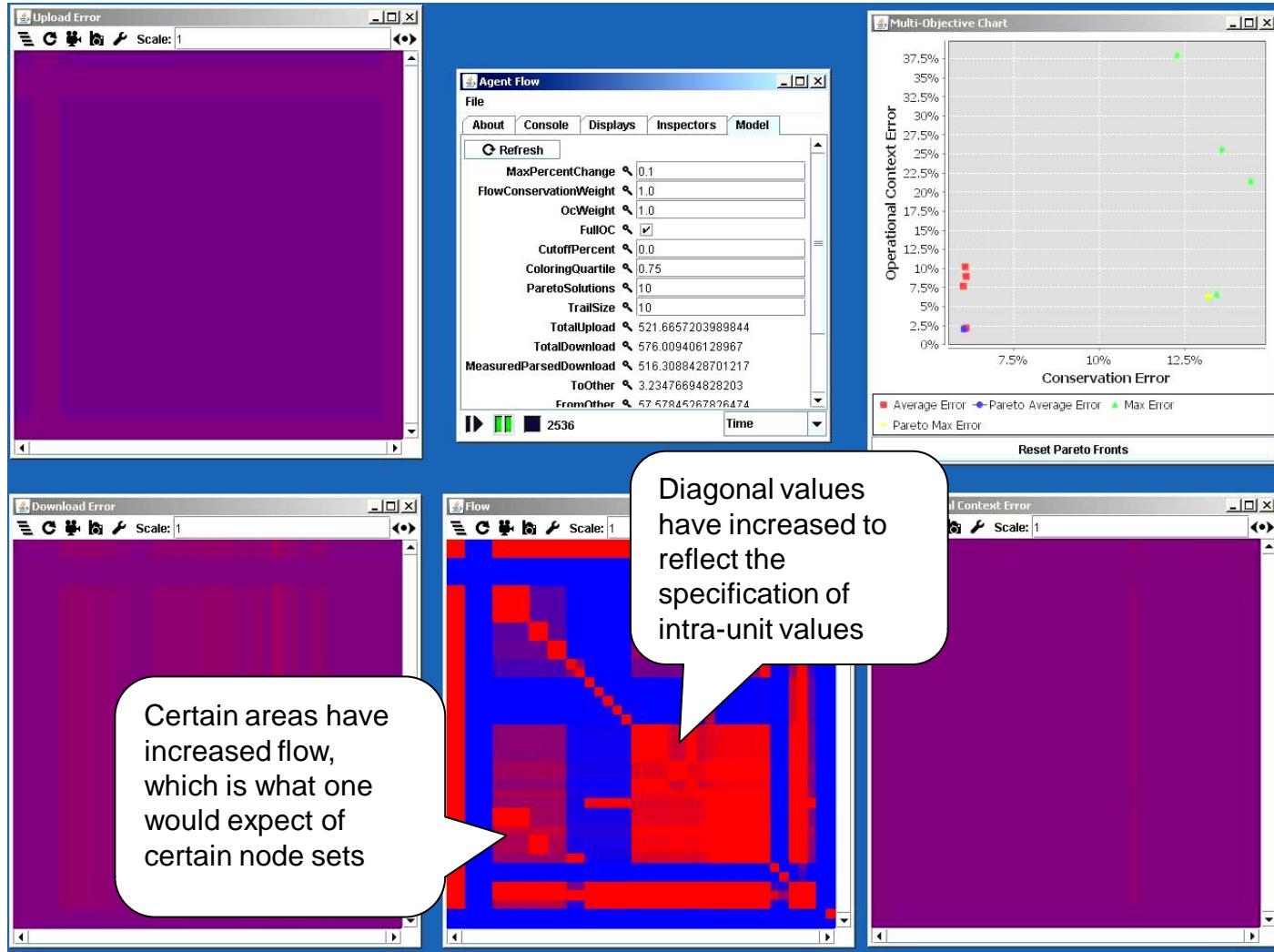
$SPL$  = Sharing path length

## Link Usage Patterns





# Interactive Parsing



## Color Legend:

Moderate flow value/zero error



High flow value/ high positive error



Low flow value/ high negative error

