On Some Results in Unmanned Aerial Vehicle Swarms

Chris Augeri*

Air Force Institute of Technology

Dept. of Electrical & Computer Engineering

*work performed as a Ph.D. candidate under the supervision of Barry Mullins, Rusty Baldwin, Dursun Bulutoglu, and Leemon Baird

This research is partially supported by the Air Force Communications Agency. The views expressed are those of the authors and do not reflect the official policy or position of the U.S. Air Force, Department of Defense, or U.S. Government.

Managing Data in UAV Swarms

Study Area	Key Result
1. Moving Data	JOCosim Co-Simulator
2. Compressing Data	XML Compression Study
3. Indexing Data	*k*-D Skip Graph
4. Conclusions	Summary & Future Work

1. Moving Data

The JOCosim Co-simulator*

*collaborative effort with Kevin Morris and Barry Mullins

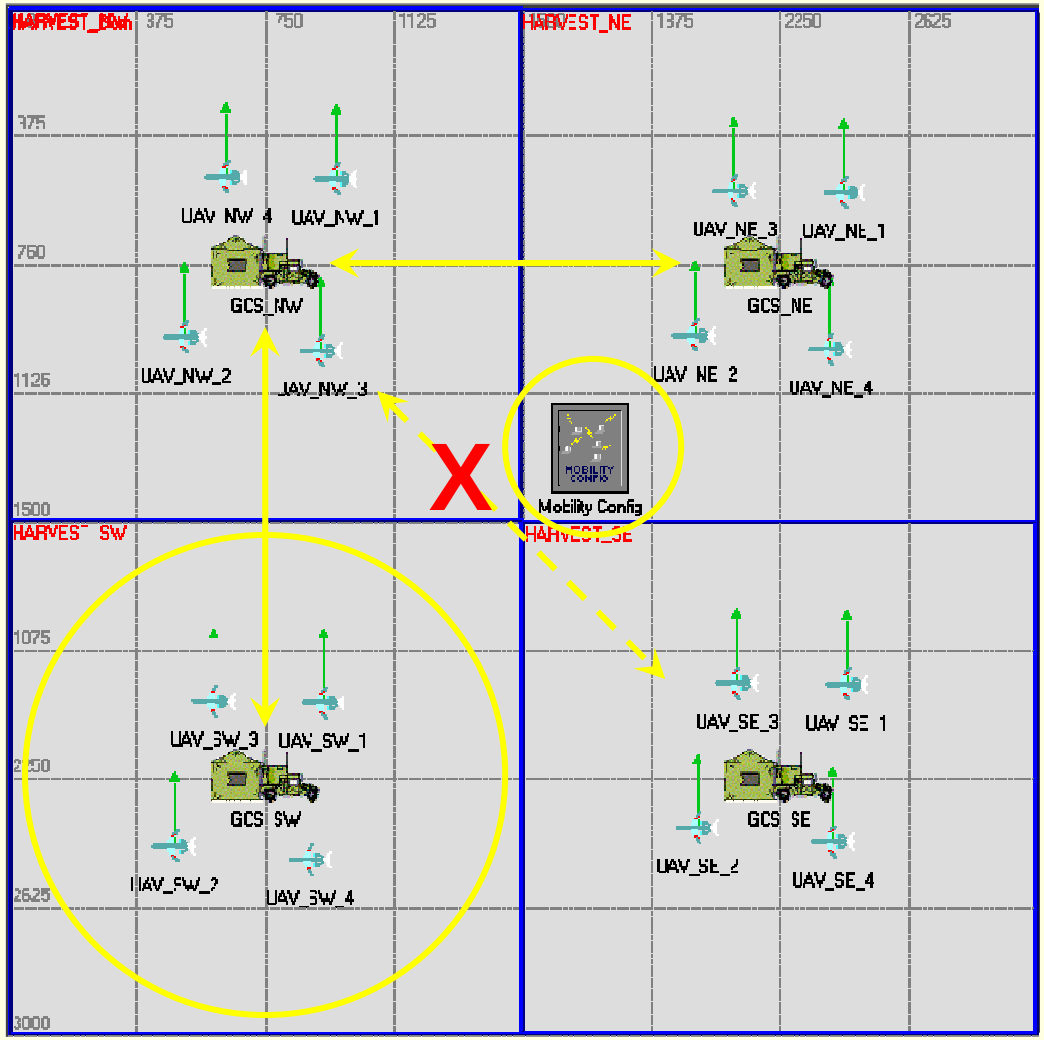
Desert Hawk UAV

- Launch Bungee
- Endurance 1 hour
- Range 3 mi
- Speed 55 mph
- Weight 7 lbs
- Wingspan 4 ft

•**References**
•http://www.af.mil/news/story_media.asp?storyID=123009770
•http://www.af.mil/news/airman/0105/unmanned2b.shtml

HARVEST: Swarm Paradigm

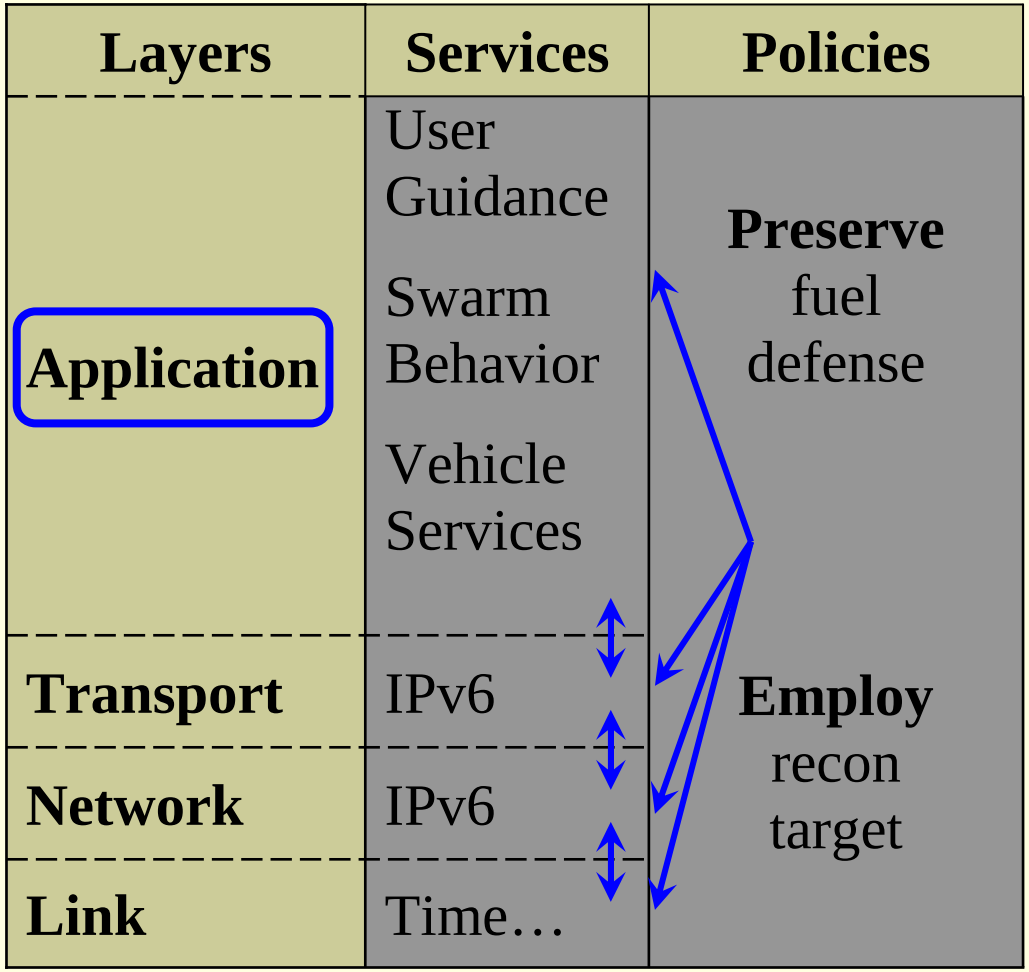
- **Configuration**
  - 4 GCSs
  - 4 UAVs / GCS
- **Cooperative Search**
  - Sniper trackers
  - Plume monitors
- **JOCosim**
  - I: OPNET → Java
  - II: Java → OPNET



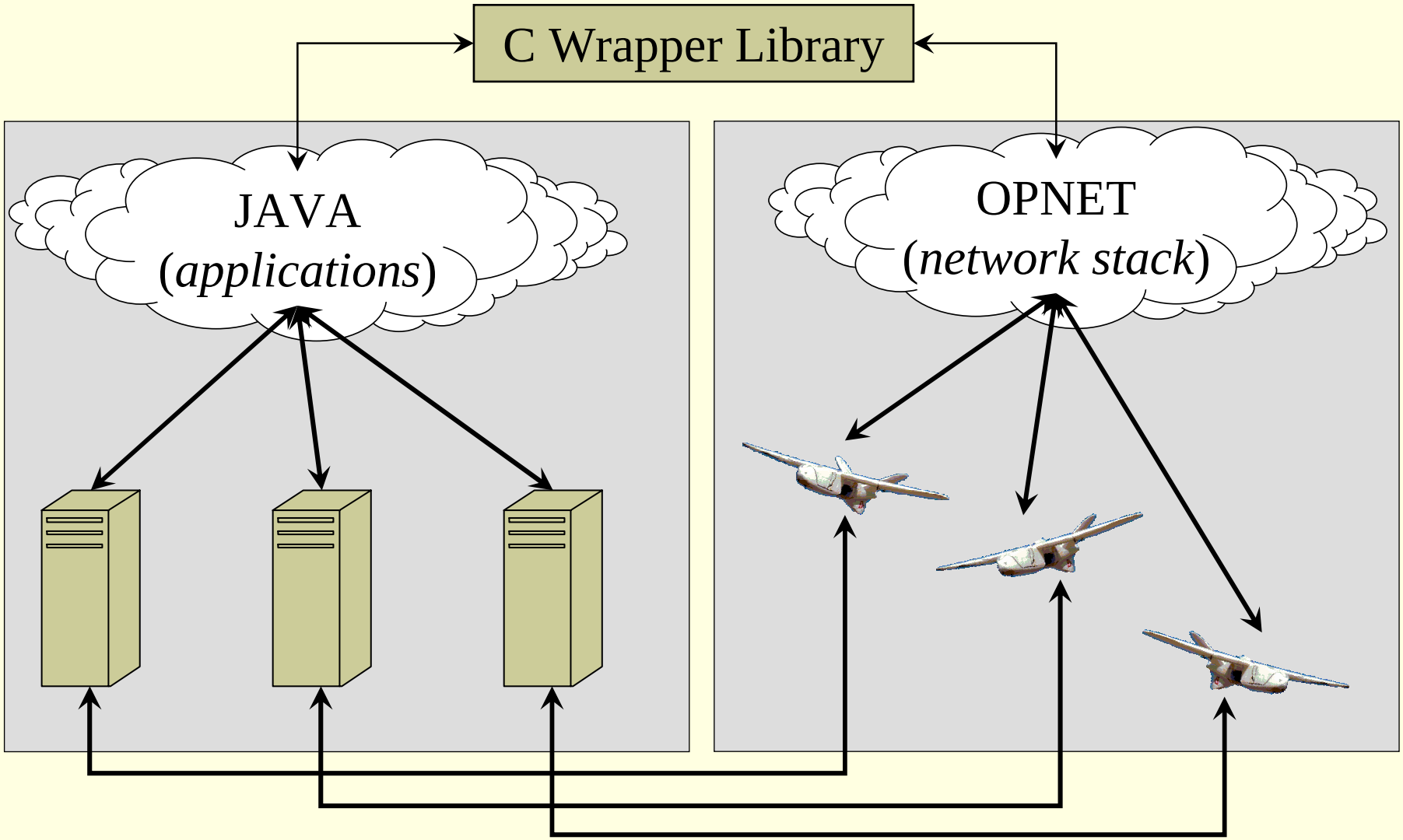
HARVEST: Protocol Stack

- **Cross-Layer Design**
  - Integrate vertically…
  - Pros: optimization
  - Cons: updating
- **Managing Policies**[1]
  - Command vertically…
  - Pros: scalability
  - Cons: speed?

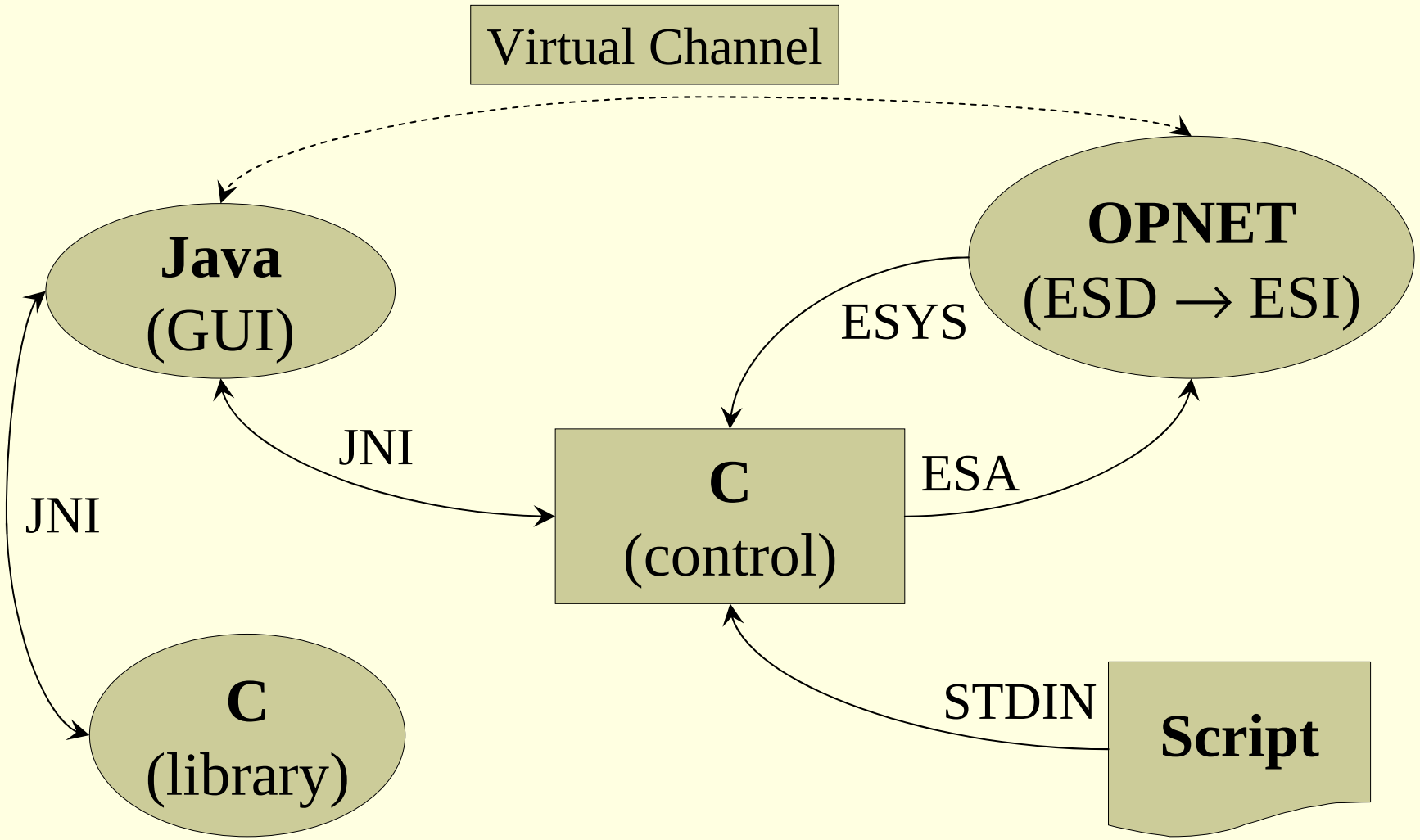
[1]I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. "A survey on sensor networks", *IEEE Communications*, 40(8):102–114, 2002.



JOCosim I & II: Co-Simulation

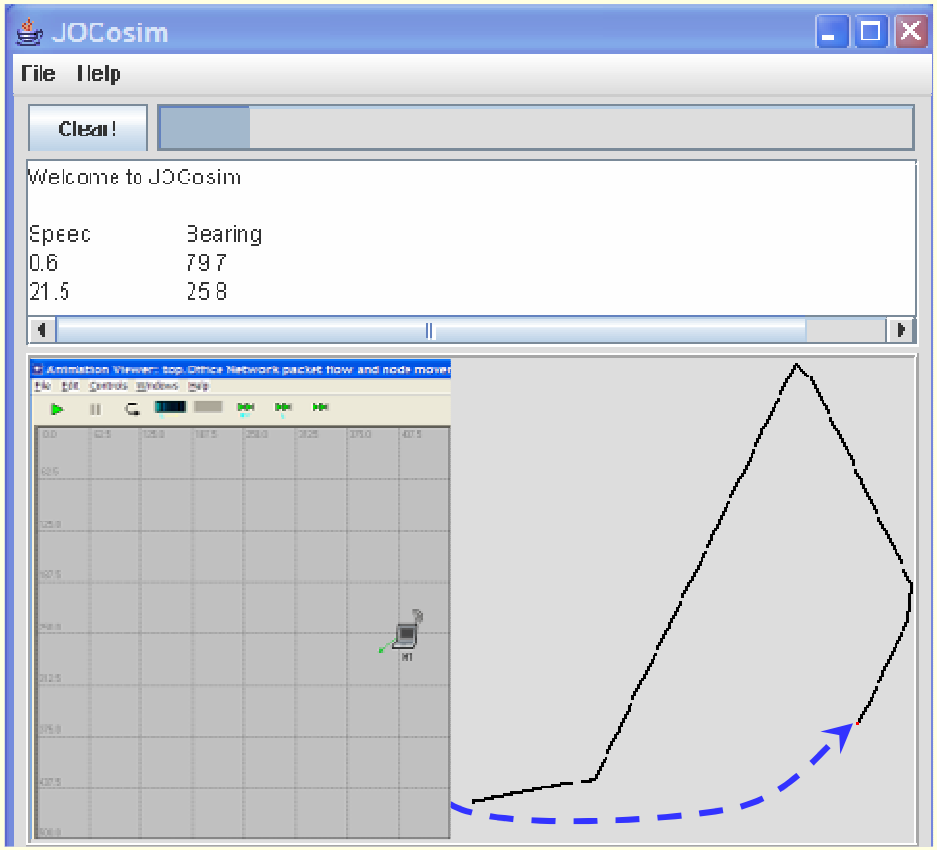


JOCosim I: Design

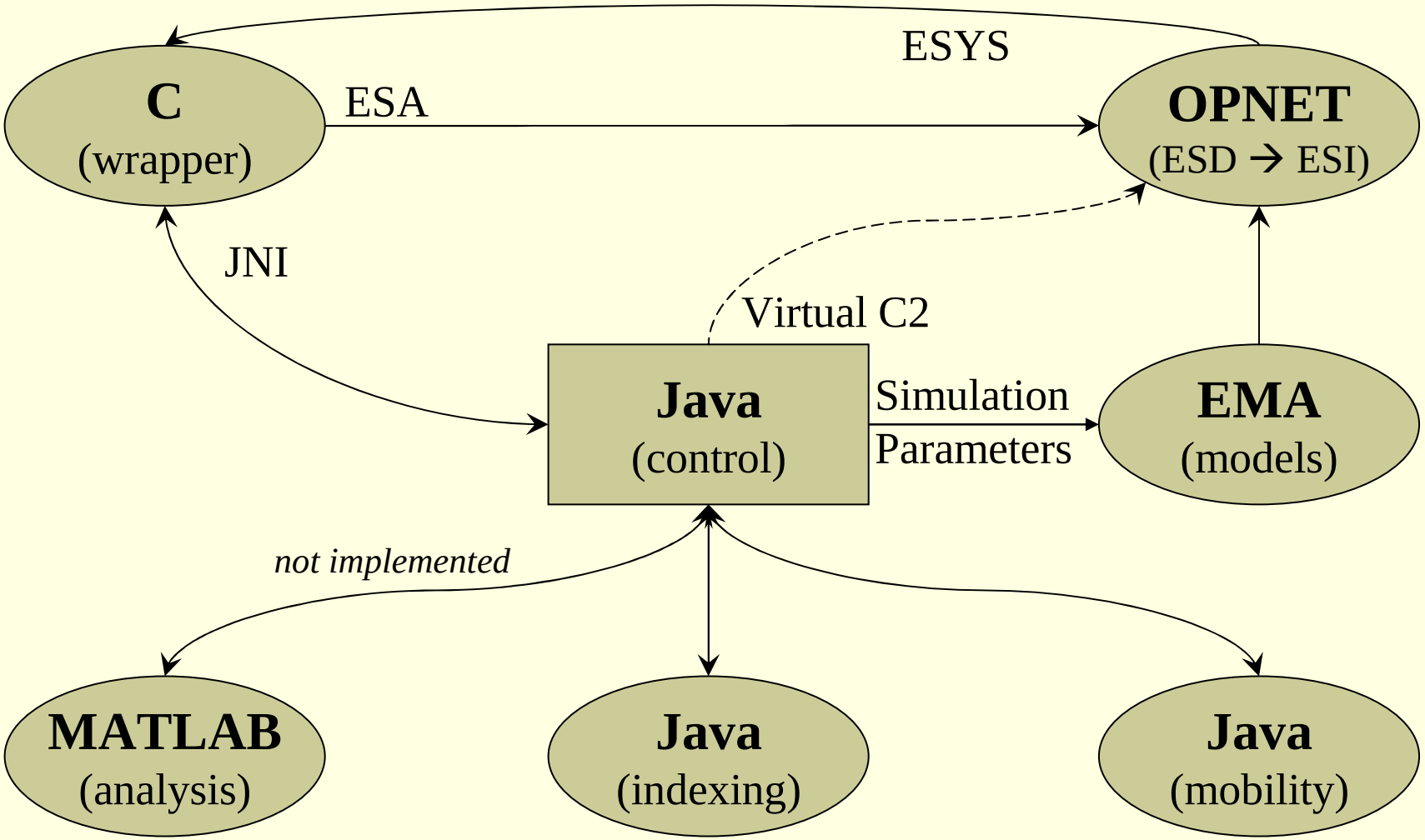


JOCosim I: Animation Viewer

- **First Prototype**
  - Script control (C)
  - JNI (C ↔ Java)
- **UAV Data**
  - Position → Java
  - Vector → OPNET
- **GUI Verifier**
  - Java: position tracker
  - OPNET: animation viewer

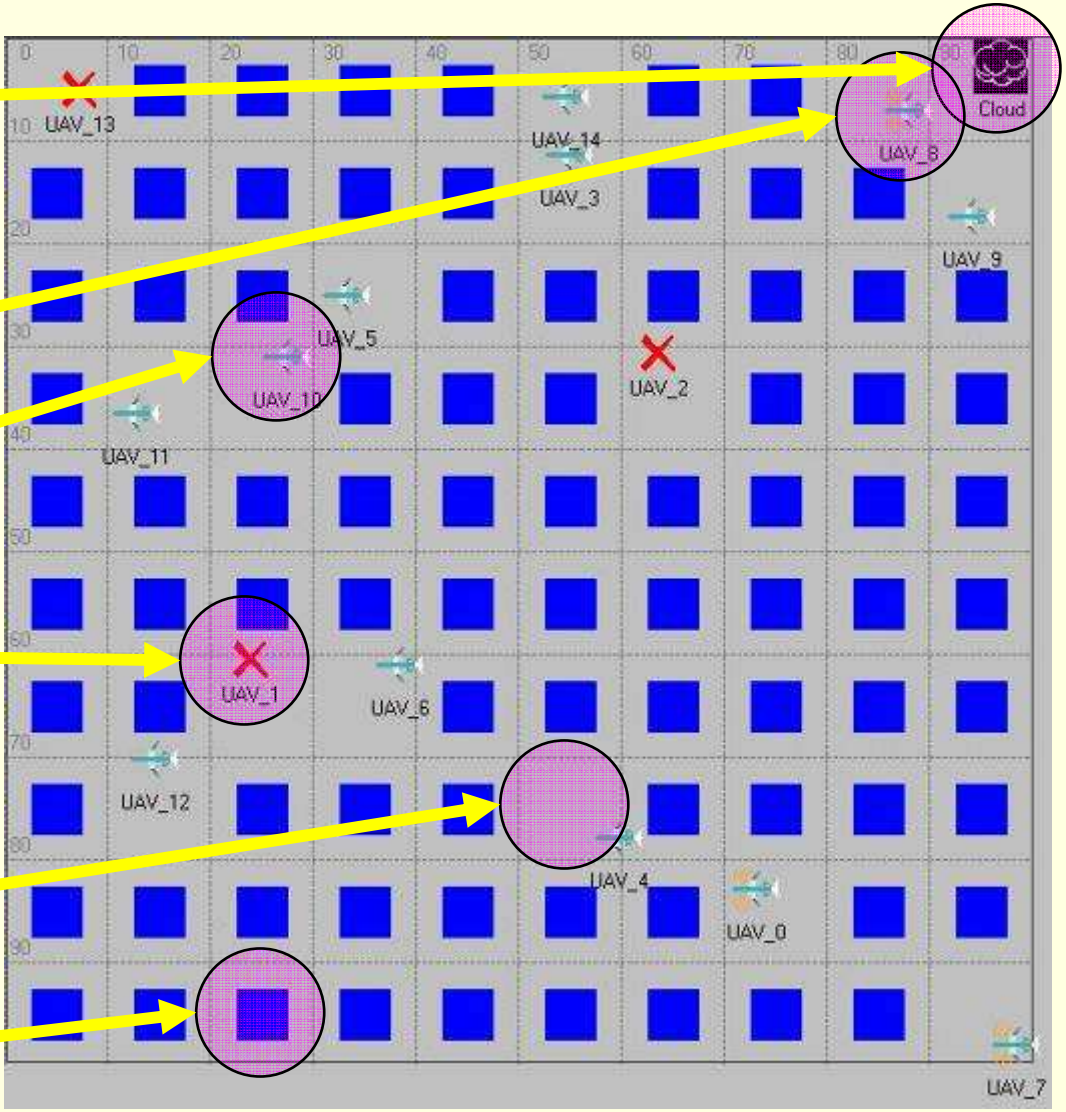


JOCosim II: Design



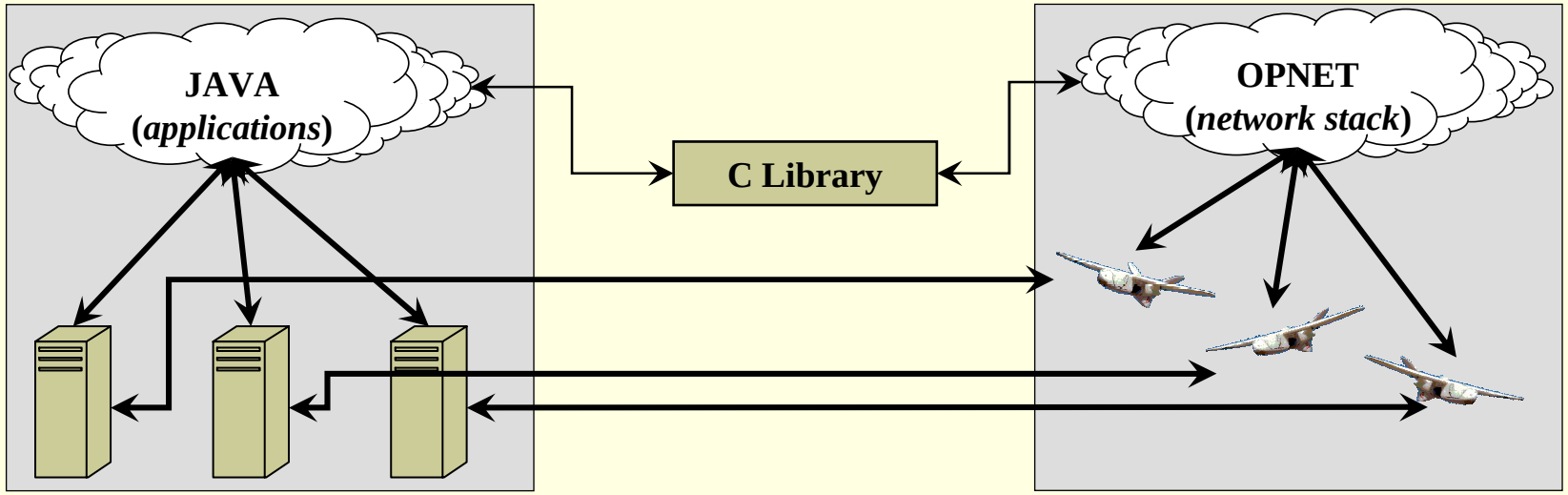
JOCosim II: Animation Viewer

- **Interface cloud**
- **UAVs**
  - Active
  - Reserve
  - Destroyed
- **Search Areas**
  - Searched
  - Not searched



Conclusions

- **JOCosim I**
  - Established Java & OPNET can communicate
  - Provides path for external application layer services
- **JOCosim II**
  - Established Java can control OPNET simulation
  - Provides path for external swarm services



2. Compressing Data

An XML Compression Study

XML: eXtensible Markup Language

```
1. <?xml version="1.0"?>
2. <Book><Title>Bestseller</Title>
3.    <Chapter><Title>The Night</Title>
4.       <Par>...dark & stormy...</Par>
5.    </Chapter>
6.    <Chapter><Title>The Day</Title>
7.       <Par>...enemy destroyed!</Par>
8.    </Chapter>
9. </Book>
```

Assessed Compressors

Compressor Class	Purpose	Compressor	Control?
Arithmetic	*"maximum" feasible compression*	CACM3	H_1
		PPMZ2	$E[H_\infty]$
		PAQ	
		PPMd (WinZip)	
Dictionary (*zip*)	*"industry-standard" compressors*	Bzip2	N/A
		Gzip	
		WinZip	
XML (*binary format*)	*publicly accessible XML-specific compressors*	Fast Infoset	
		WBXML	
		XBIS	
XML (*schema-aware*)		XGrind	
		XMill	
		XMLPPM	
		XML-ZIP	

Assembled Corpus

Category	# Files	Description	
Binary	7	• Baseball Statistics • DNA Sequence • European Weather	• GIS Data • Periodic Table
Conformance	3	• NIST Tests	
Databases	13	• DoD Per Diem • FCC Ham Radio • Oracle Transactions	• UN Catalog • Server Log
Documents	14	• CIA Factbook • Degree Listing • OpenDocument	• Shakespeare • US Congress • WSJ Linguistics
RSS Feeds	7	• Mainstream News • NOAA Weather	

Corpus Details (subset)

Source	Domain	Bytes	Lines	Chars	Tags	Depth	H_1	$E[H_\infty]$
Weather (2004)	Science	40042243	841765	85	88	10	3.987	0.008
		1171129	24652	83	85	10	3.989	0.011
		335	9	58	5	3	5.248	0.325
Baseball (1998)	Databases	904261	25965	76	43	6	4.373	0.020
OpenDoc	Documents	814397	17714	94	35	11	4.890	0.046
GIS Maps	Databases	1004047	18557	87	35	10	4.849	0.035

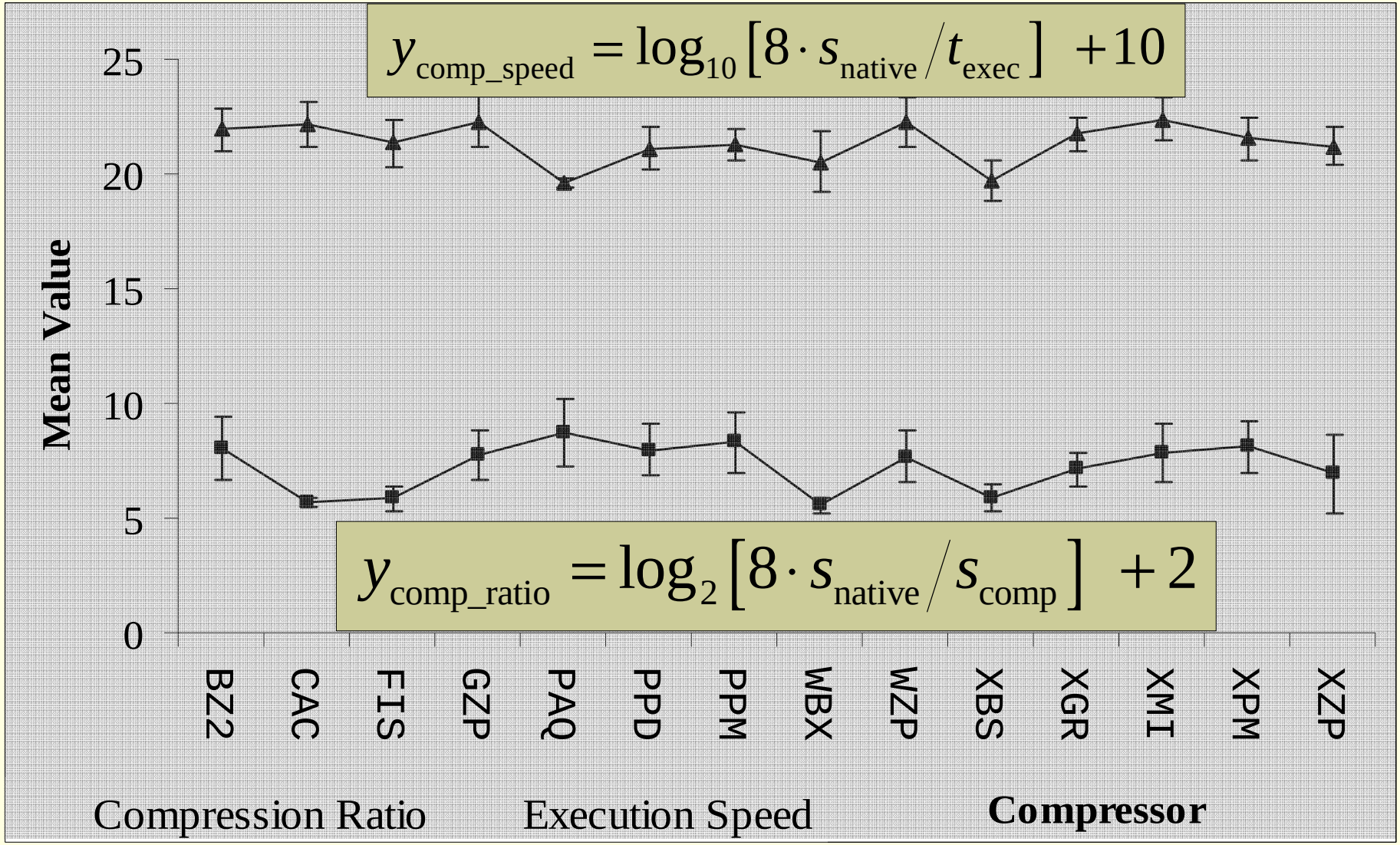
subjective groups

collected factors

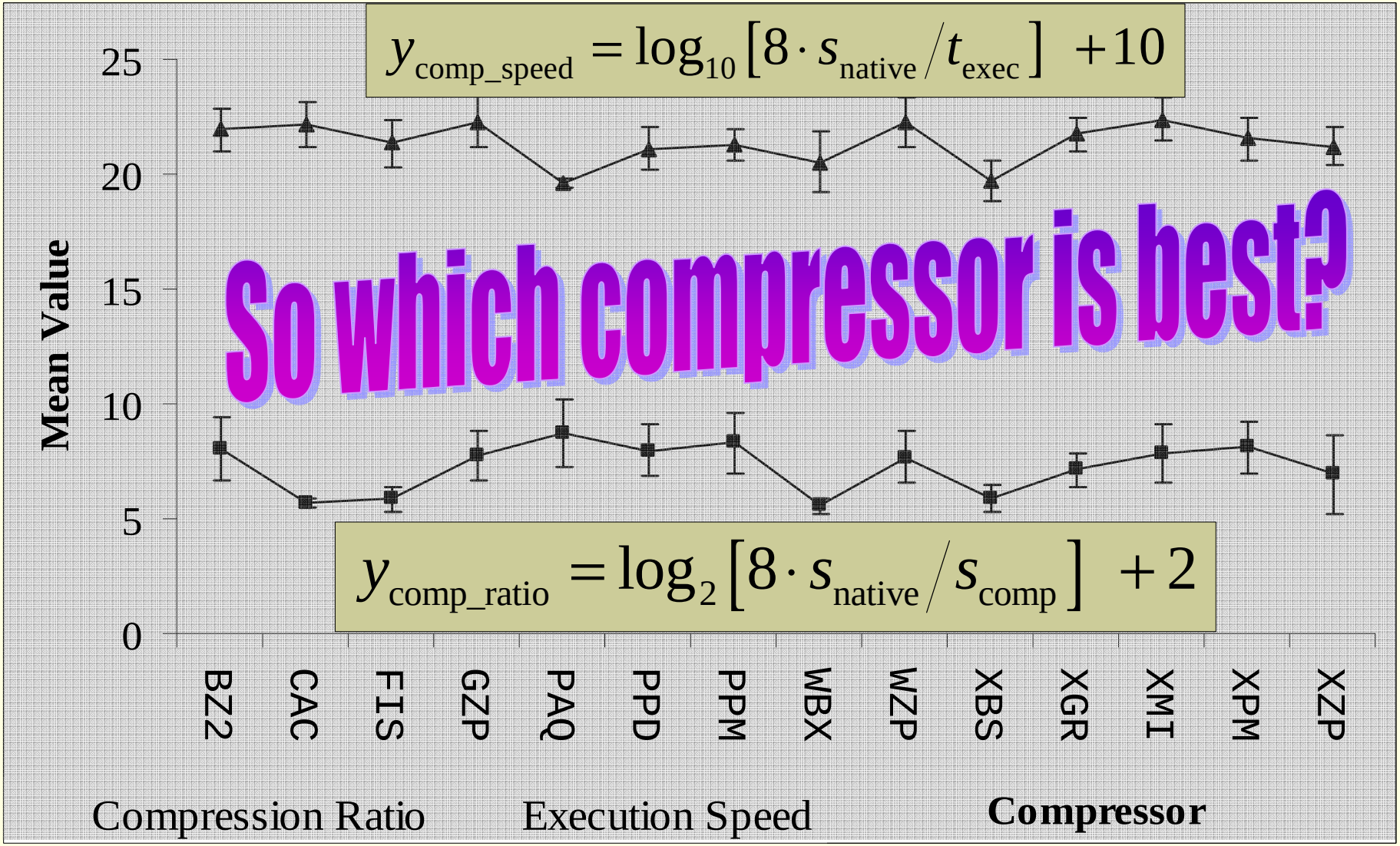
via CACM3

maximum compression

Compression Ratio & Speed?



Compression Ratio & Speed?



Combined Efficiency Metrics

Only Existing Efficiency Metric?

$$y_{\text{eff\_old}} = \log_{10}\left[2^{\left(s_{\text{comp}}/\min\left(s_{\text{comp}}\right)-1\right)} \cdot t_{\text{exec}}\right] + 10$$

Proposed Efficiency Metric

$$y_{\text{eff\_prop}} = \log_{10}\left[\frac{\left(\frac{s_{raw}^2}{\left(\min\left(s_i\right)\cdot s_x\right)}\right)}{t_x}\right] + 10$$

Pair-wise Means Comparison

Compressor	G_1	G_2	G_3	G_4	G_5	G_6	G_7	*Efficiency*
XMill	X							18.787
Gzip	X	X						18.644
WinZip	X	X						18.614
Bzip2	X	X	X					18.407
XMLPPM		X	X	X				18.003
CACM3			X	X				17.904
XGrind		X	X	X	X			17.880
PPMZ2			X	X	X			17.780
PPMd				X	X			17.543
XML-ZIP				X	X			17.350
Fast Infoset					X			17.130
WBXML						X		16.366
PAQ						X	X	16.272
XBIS							X	15.576

Only XMill is as efficient as zip!

Conclusions

- **Contributions**
  - Proposed XML Test Corpus
  - Methodology & Analysis
  - **Combined Efficiency Metric**
- **Results**
  - Binary format for small files
  - XMill may be beneficial in some instances
  - **Standard compressors best option for most users**
- **Future Research**
  - 2-stage compression in standard compressor
  - Decompression speed & time, streaming queries
  - **W3C's EXIWG & Efficient XML binary format**

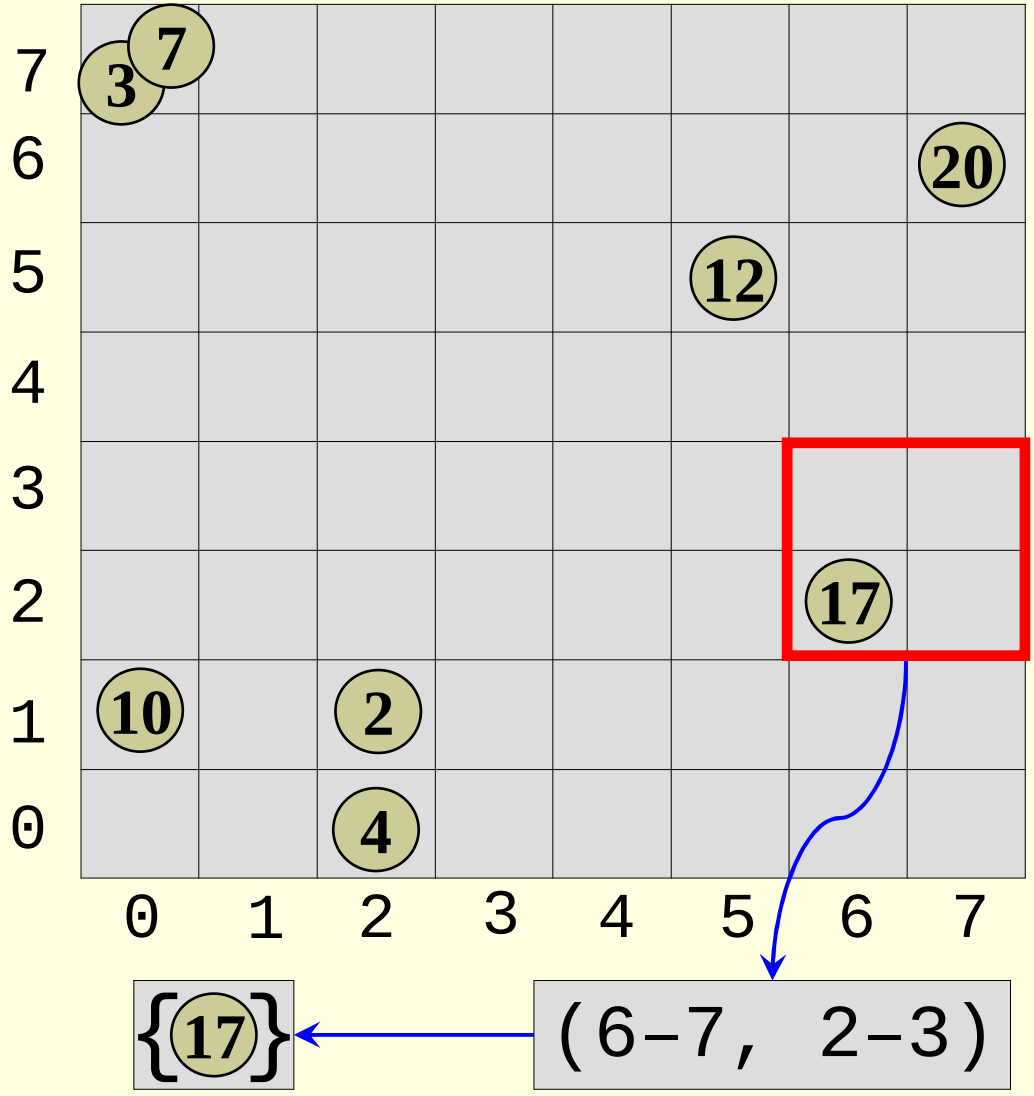
3. Indexing Data

k-Dimensional Skip Graphs*

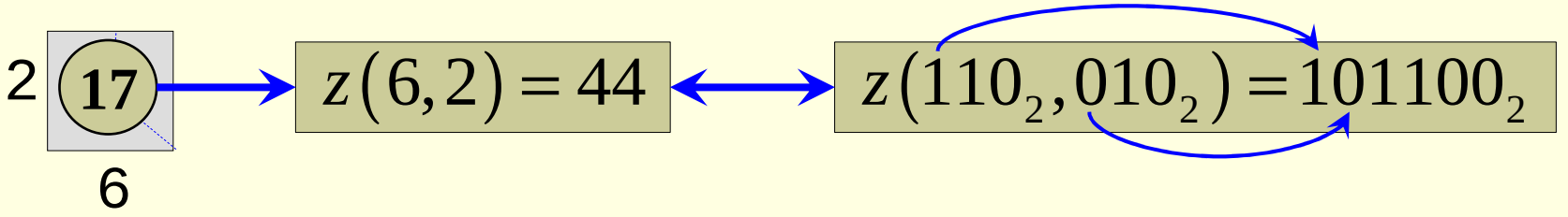
*collaborative effort with Greg Brault, Barry Mullins, Chris Mayer, and Rusty Baldwin

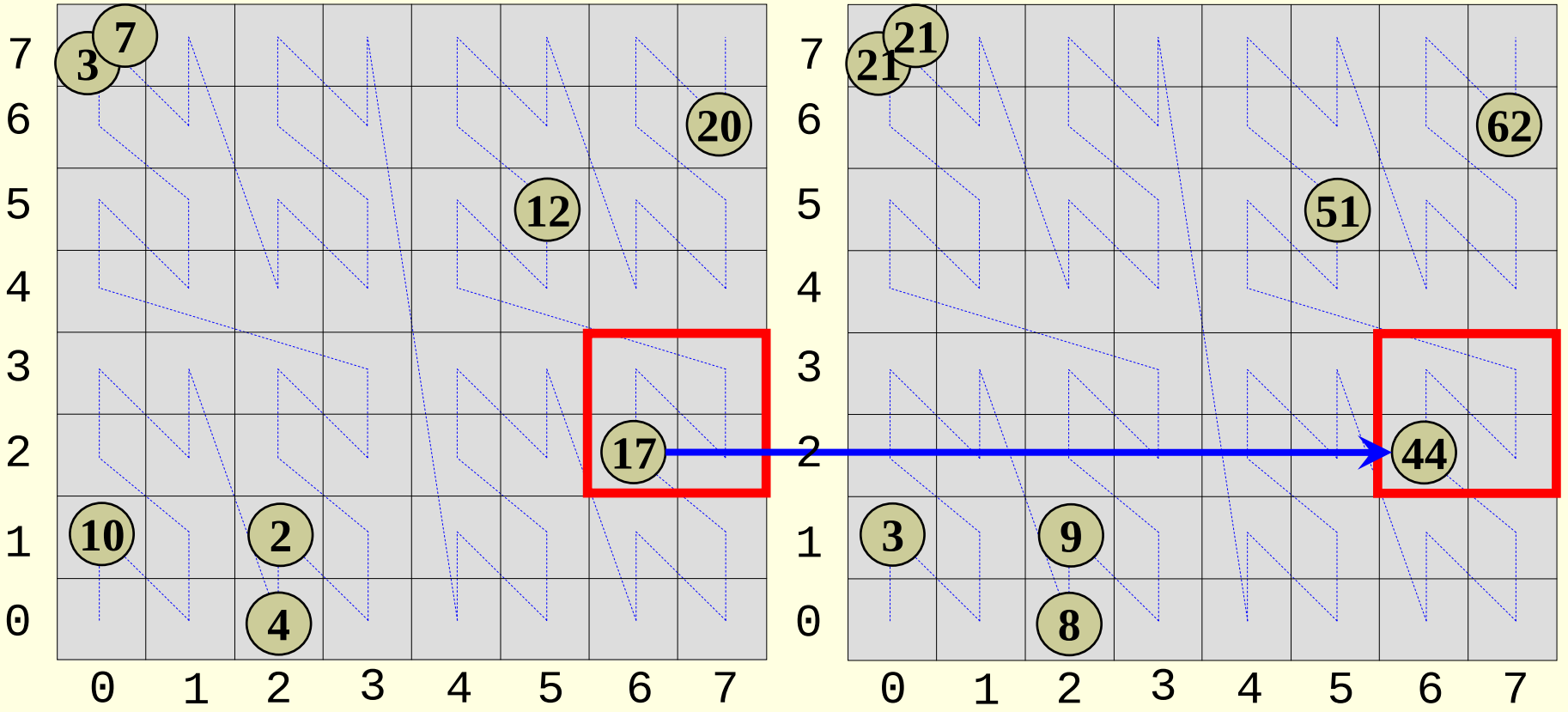
Distributed *k*-D Range Queries

- **k-D Range Query**
  - Geographic
  - Multi-Sensor
- **Distributed Indexing**
  - Hash Table (DHT)
  - Skip Graph
- ***k*-D Linearization**
  - Space-Filling Curve
  - Relative Importance

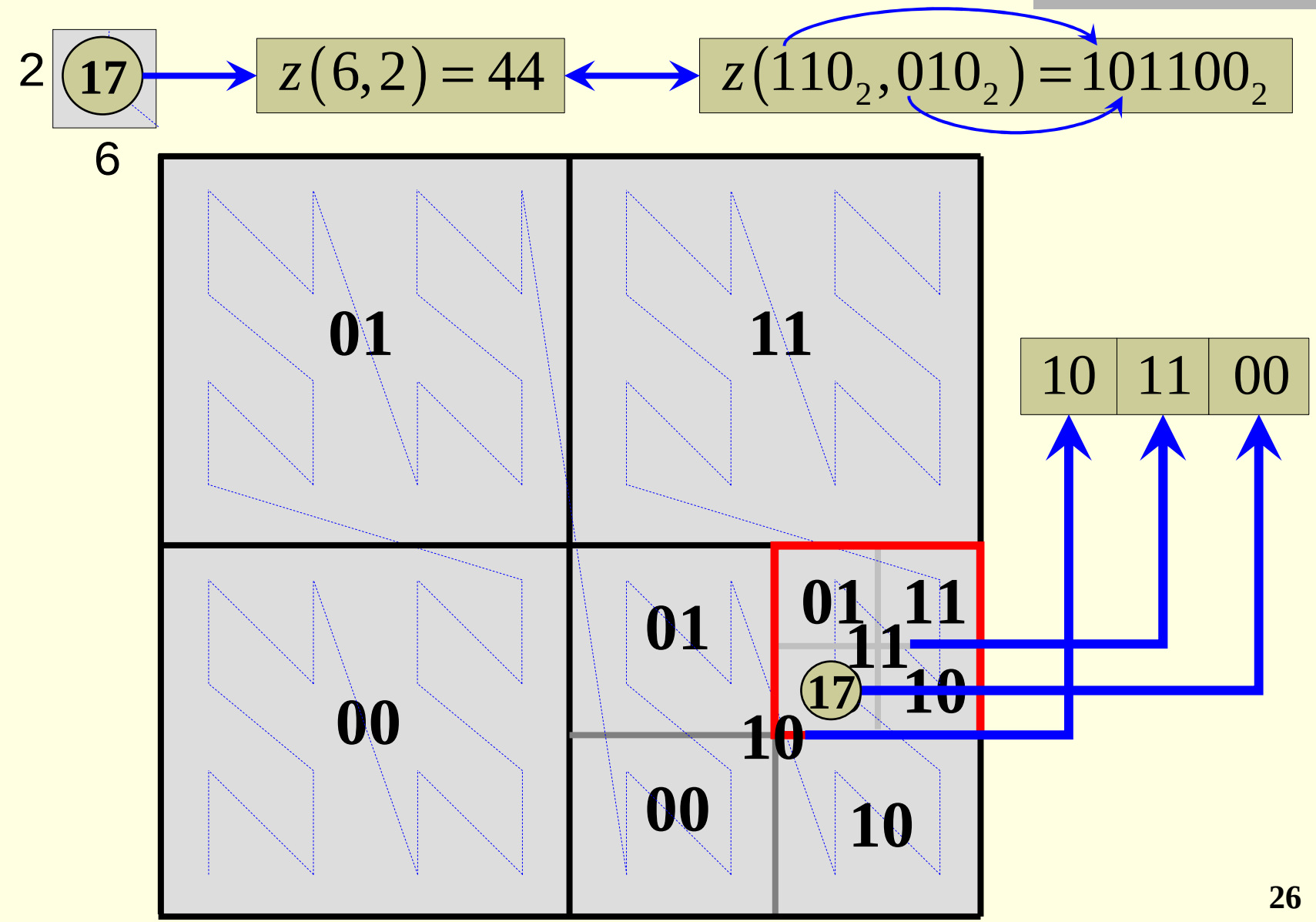


1-D → *k*-D: *z*-order SFC

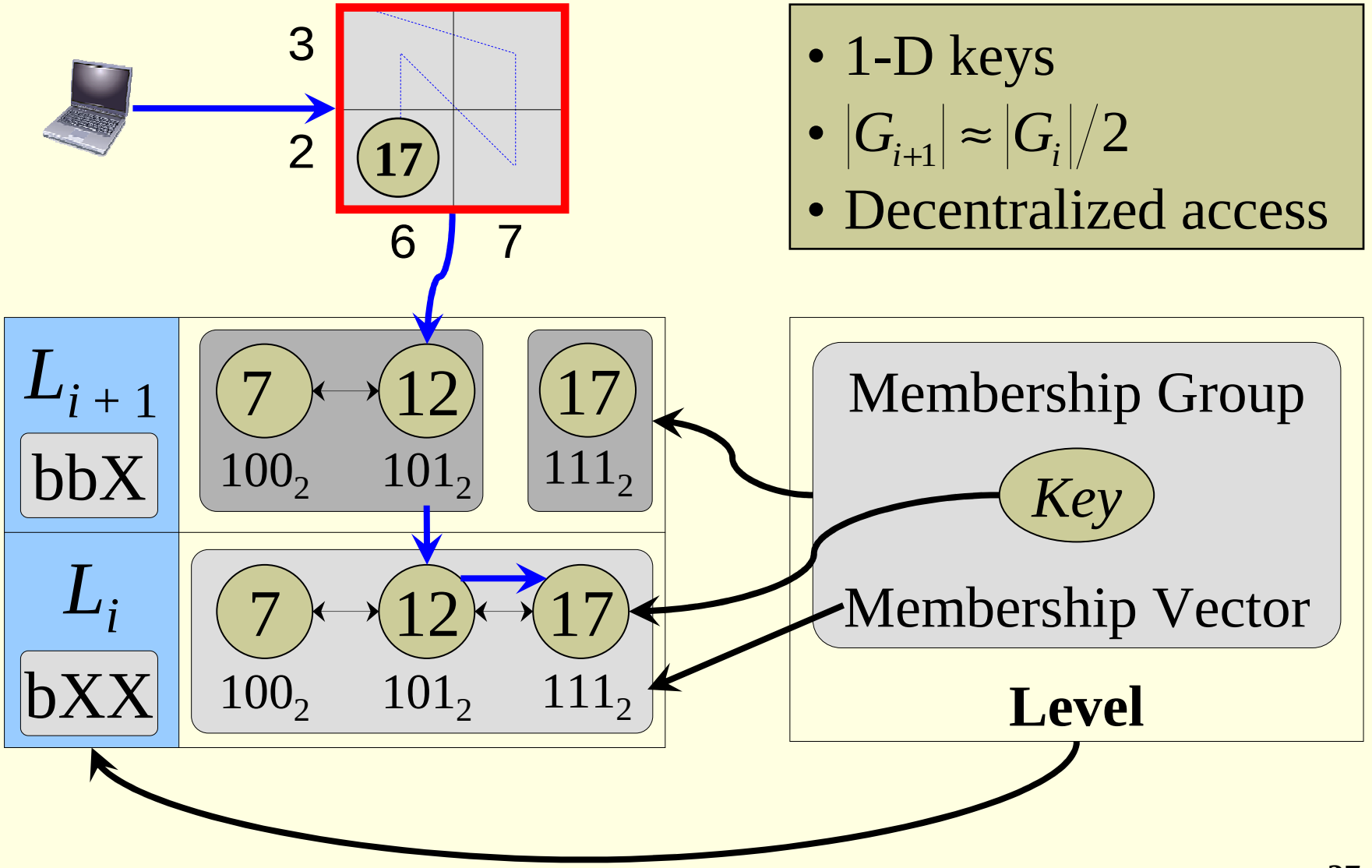




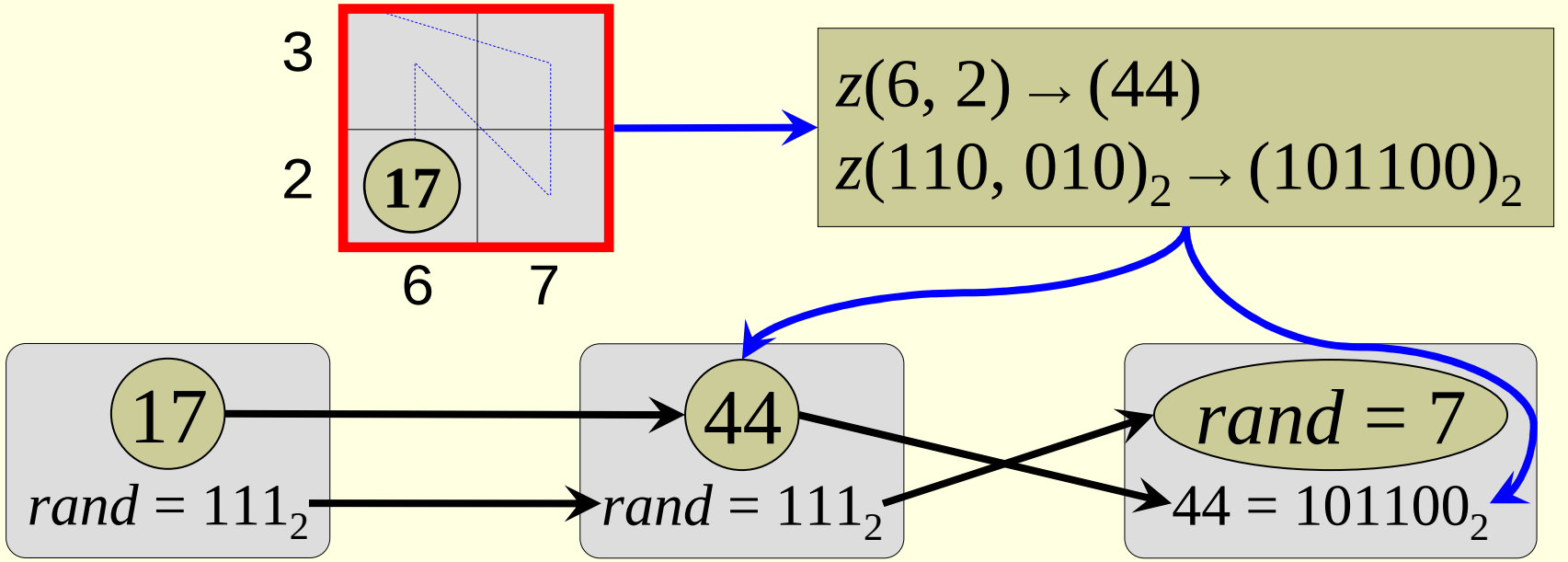
z-order SFC ↔ quad-tree



Skip Graph [Aspnes & Shah]



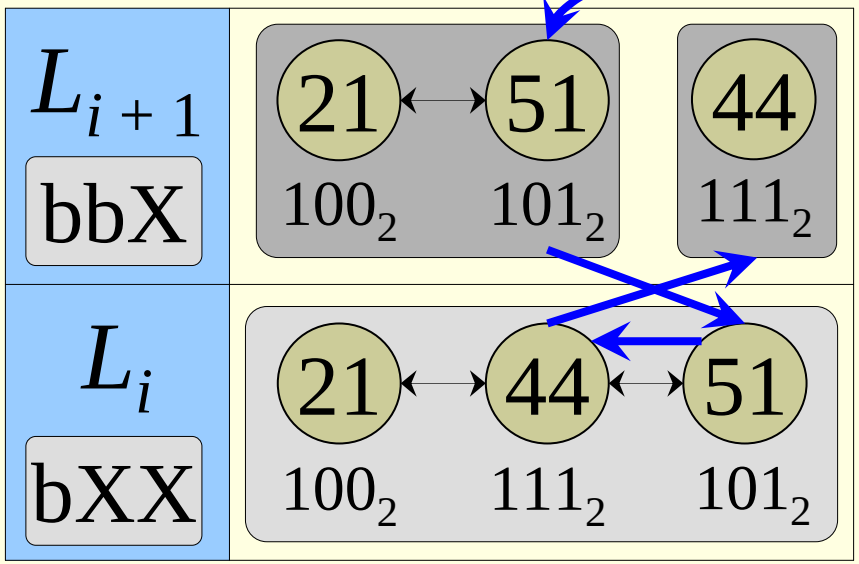
Skip Graphs: Variants

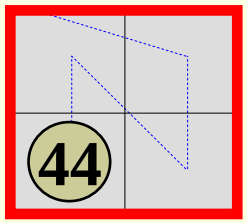


Key	1-D	k-D	random
Vector	random	random	k-D
Skip Graph (standard) [Aspnes & Shah]		**Skip Graph (standard k-D)** [Ganesan+, Epstein+]	**Skip Graph (inverted k-D)** [Brault & Augeri]

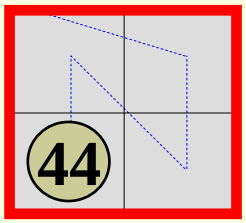
Skip Graphs: Executing Queries

Standard *k*-D (*top-down*)

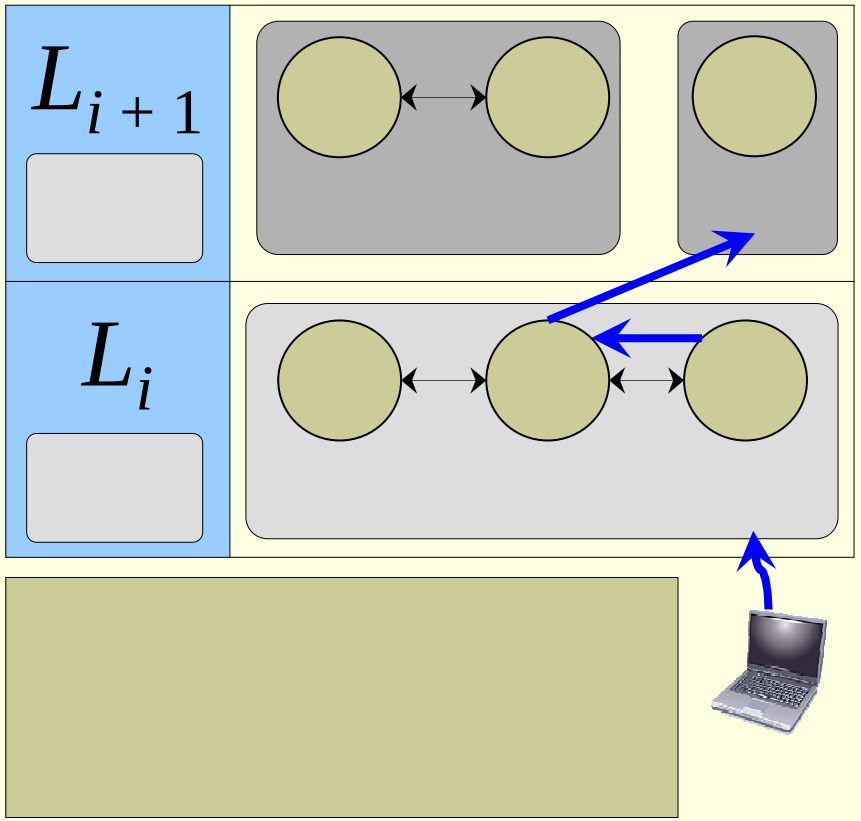




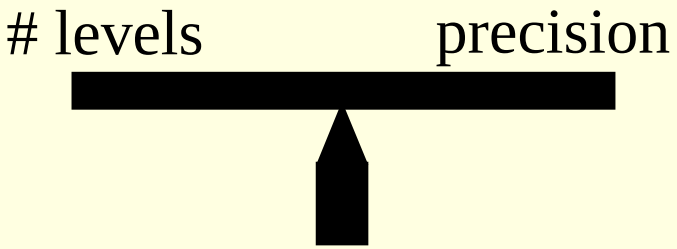
4X



11X



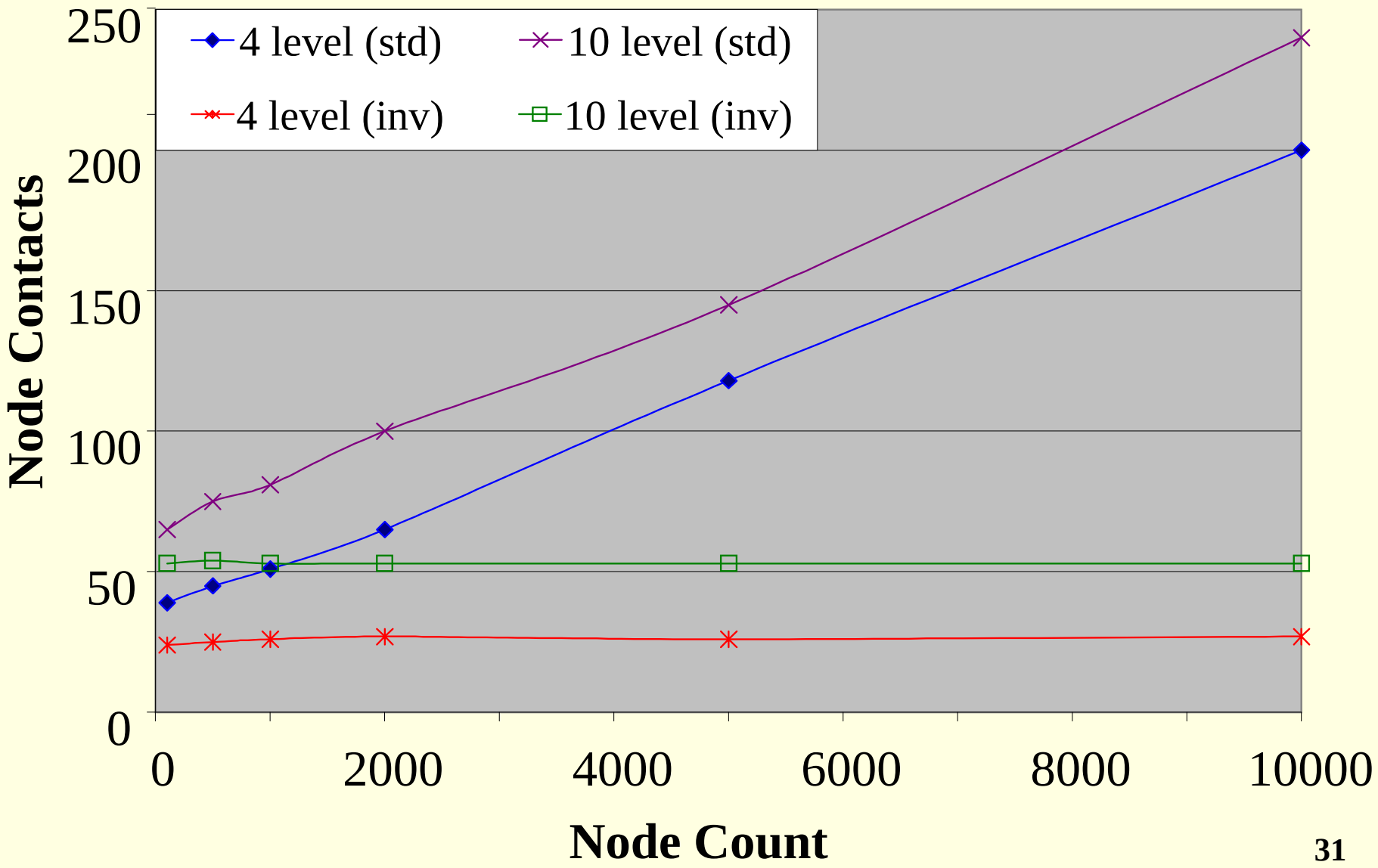
Skip Graph: Query Performance



As node count increases, inverted skip graphs generate less messages, that travel longer distances

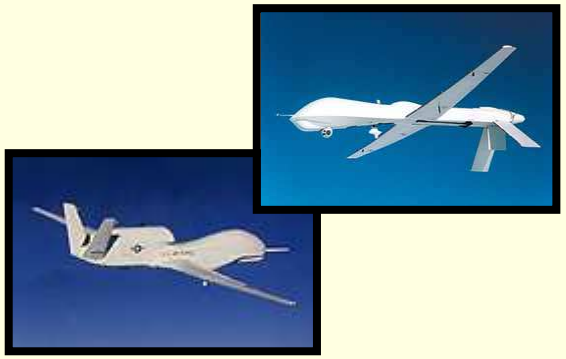
Query Precision	Number of Levels				
	4	5	6	8	10
10X	inv	-	-	-	-
1001X	inv	inv	-	-	-
100110X	inv	inv	-	-	-
10011000X	std	inv	inv	-	-
1001100011X	std	std	-	inv	inv
100110001101	std	std	-	inv	inv

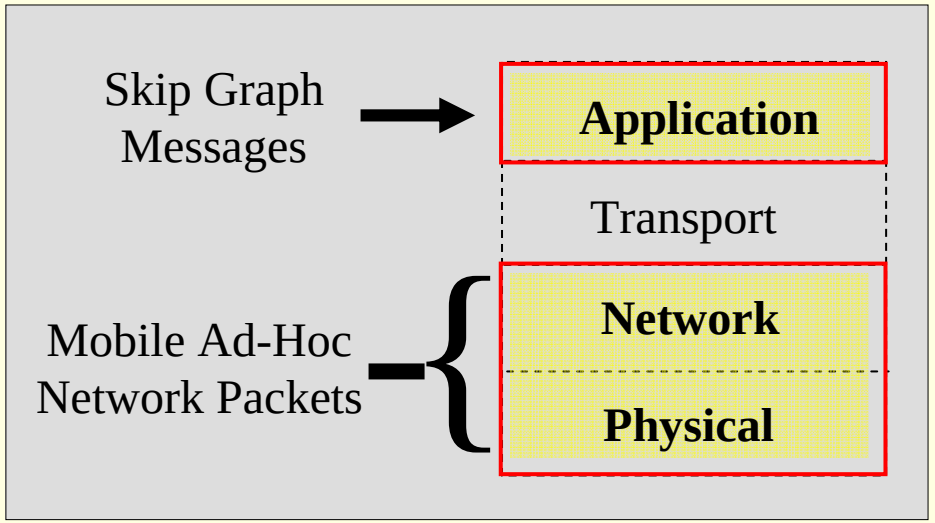
Skip Graph: Mobility Updates



Conclusions

- **Inverted Skip Graphs**
  - Invert keys & vectors
  - Bottom-up queries
- **Routing costs**
  - Estimate distances
  - Simulate in JOCosim
  - Assess mobility costs
- **Implement**
  - 3-D coordinates
  - “Hybrid” skip graph





4. Conclusions

Summary

Study Area	Key Result
1. Moving Data 	JOCosim Co-Simulator
2. Compressing Data	XML Compression Study
3. Indexing Data	*k*-D Skip Graph
4. Conclusions	Summary & Future Work