Software Visualization
Reflections and Future Directions

John Stasko

Information Interfaces Research Group
College of Computing and GVU Center
Georgia Tech

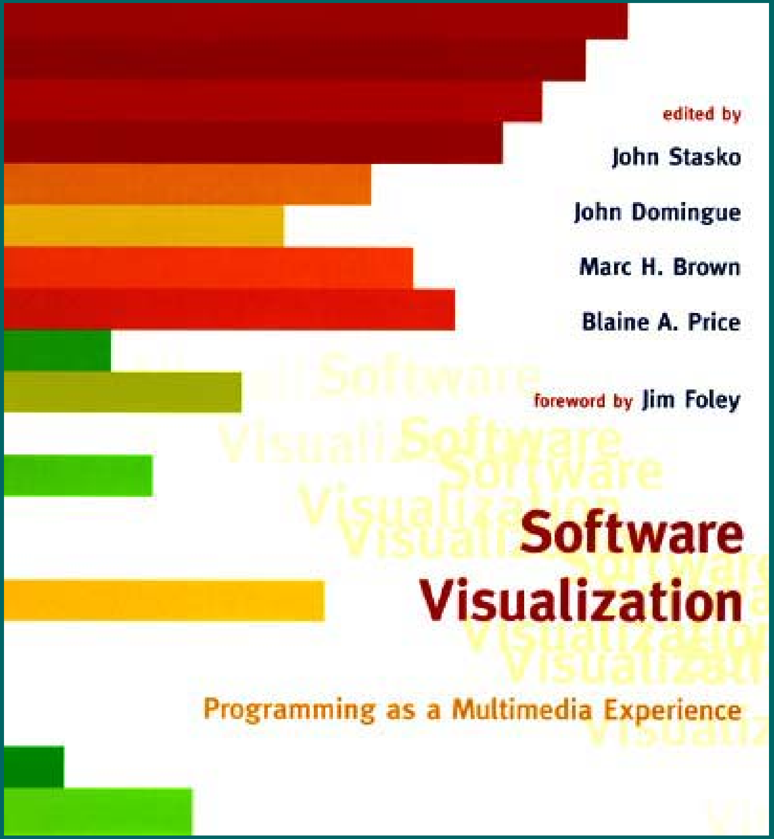
stasko@cc.gatech.edu
www.cc.gatech.edu/~john.stasko

Software Visualization

- Definition

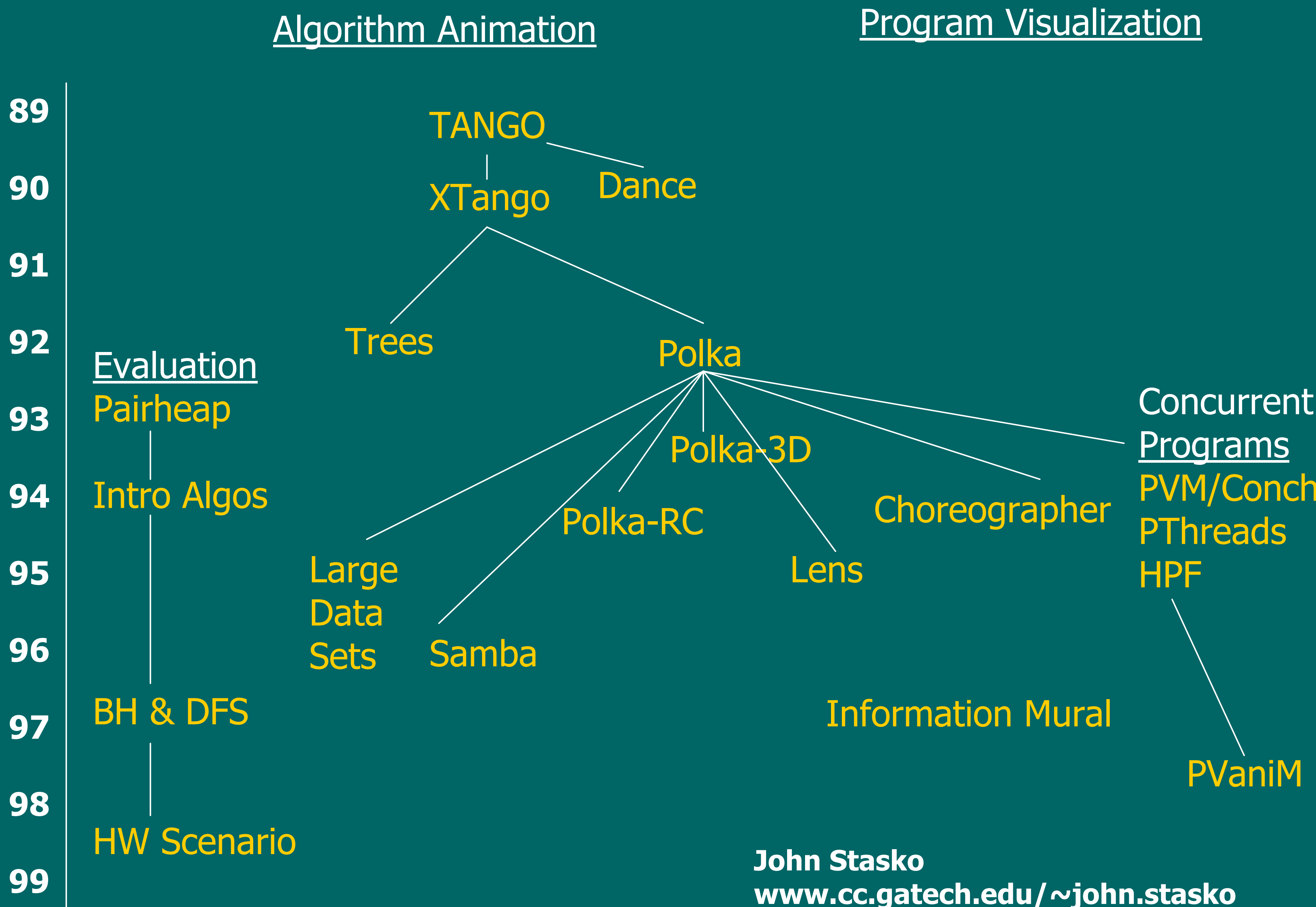
"The use of the crafts of typography, graphic design, animation, and cinematography with modern human-computer interaction and computer graphics technology to facilitate both the human understanding and effective use of computer software."

Price, Baecker and Small, '98



Software Visualization Areas

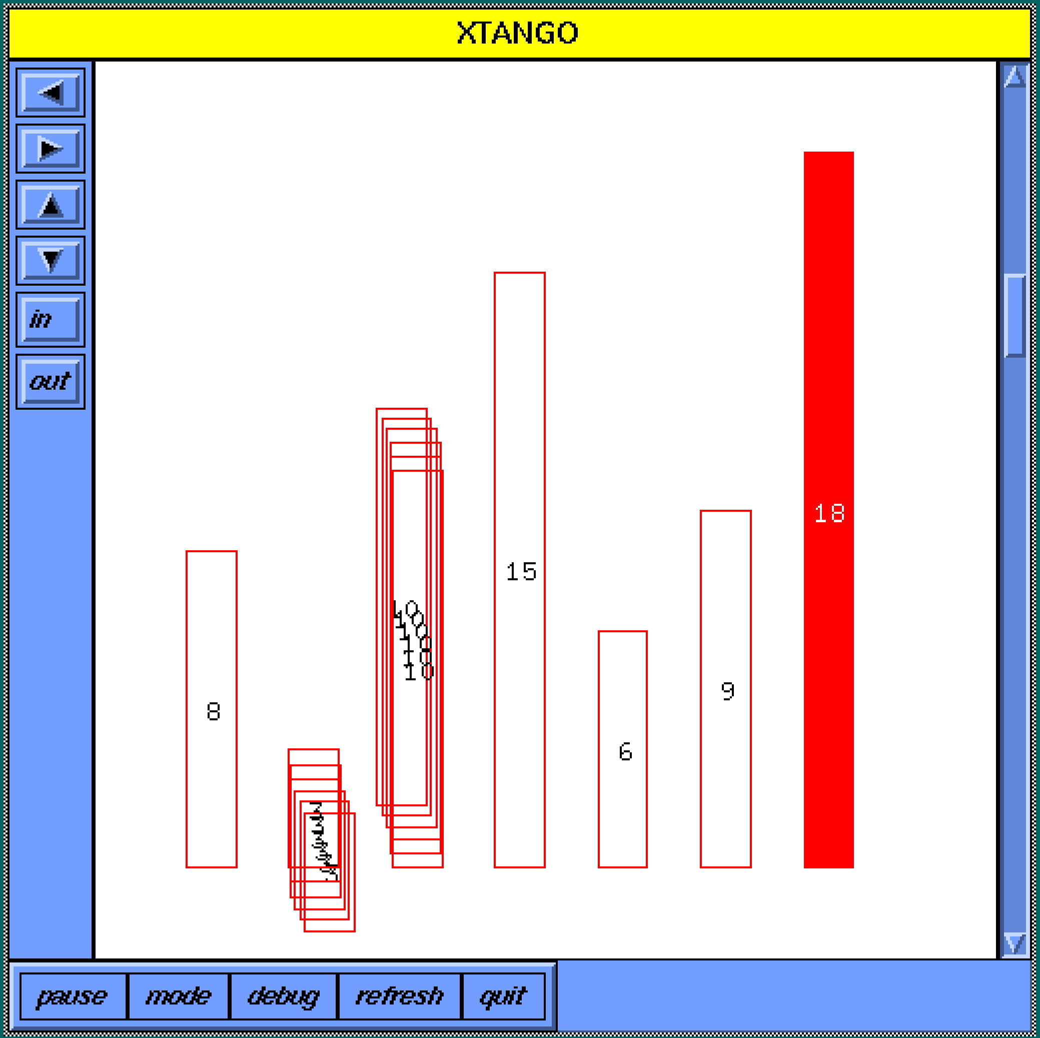
- Algorithm Visualization
  - Pedagogy
  - Systems
  - Use in classroom
  - Empirical study
- Program Visualization
  - Software engineering
  - Debugging
  - Program analysis
  - Systems



John Stasko
www.cc.gatech.edu/~john.stasko
stasko@.cc.gatech.edu

Tango

Multiple frames from bubblesort



IEEE Computer '90
JVLC '90

Tango Model

- Image, Location, Path, Transition

Sample code

```
Location    fromloc, toloc;
Image       ball;
Path        path1, path2;
Transition  mover;

ball = AssocRetrieve("ID", paramvalue);
fromloc = ImageLoc(ball, Center);
toloc = LocCreate(0.4, 0.7);
path1 = PathMakeType(Clockwise);
path2 = PathExample(fromloc, toloc, path1);
mover = TransCreate(Move, ball, path2);
TransPerform(mover);
```

Tango Contributions

- Importance of smooth animation
- Simplification of the design/programming process
- Formal model of the animation, the Path-Transition Paradigm

XTango

- Native X Windows version of Tango

Tango

BWE

X Windows

XTango

X Windows

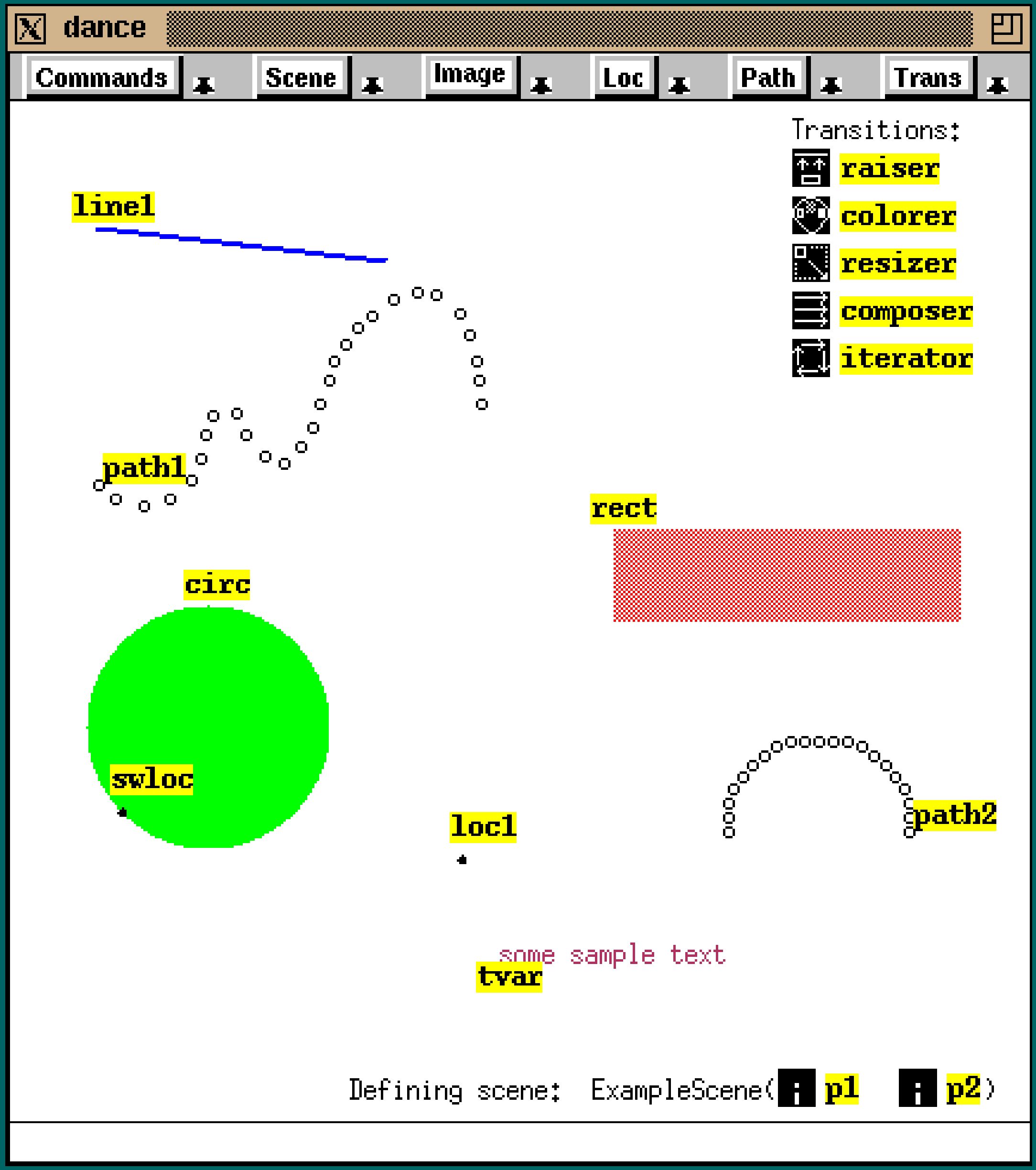
SIGACT News '92

D. Hayes

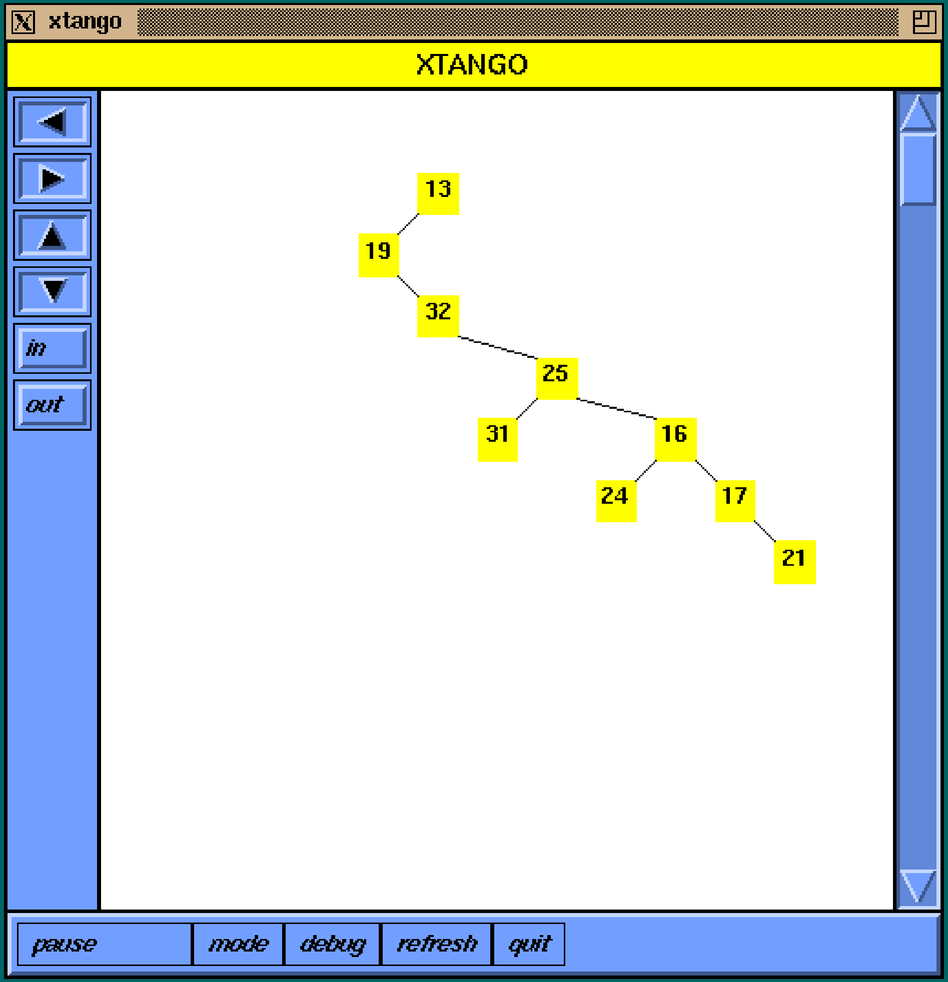
DANCE

- Why not design an algorithm animation visually?
- Direct manipulation environment for interactive, visual design of algorithm animations
  => Automatically generates Tango code

DANCE


dance
Commands
Scene
Image
Loc
Path
Trans
Transitions:
raiser
colorer
resizer
composer
iterator
line1
path1
rect
circ
swloc
loc1
path2
some sample text
tvar
Defining scene: ExampleScene(p1 p2)

Animating Tree Algorithms



Binary representation of a pairing heap data structure

VL '92

C. Turner

Concurrent Programs

- Understanding parallel programs is even more difficult than serial
- Visualization and animation seem naturals for illustrating concurrency
- Temporal mapping of program execution to animation becomes critical

Tango Insufficiencies

- Simulated object-oriented
- One animation window
- Transition model

Must compose all transitions into one "super" transition, then perform it

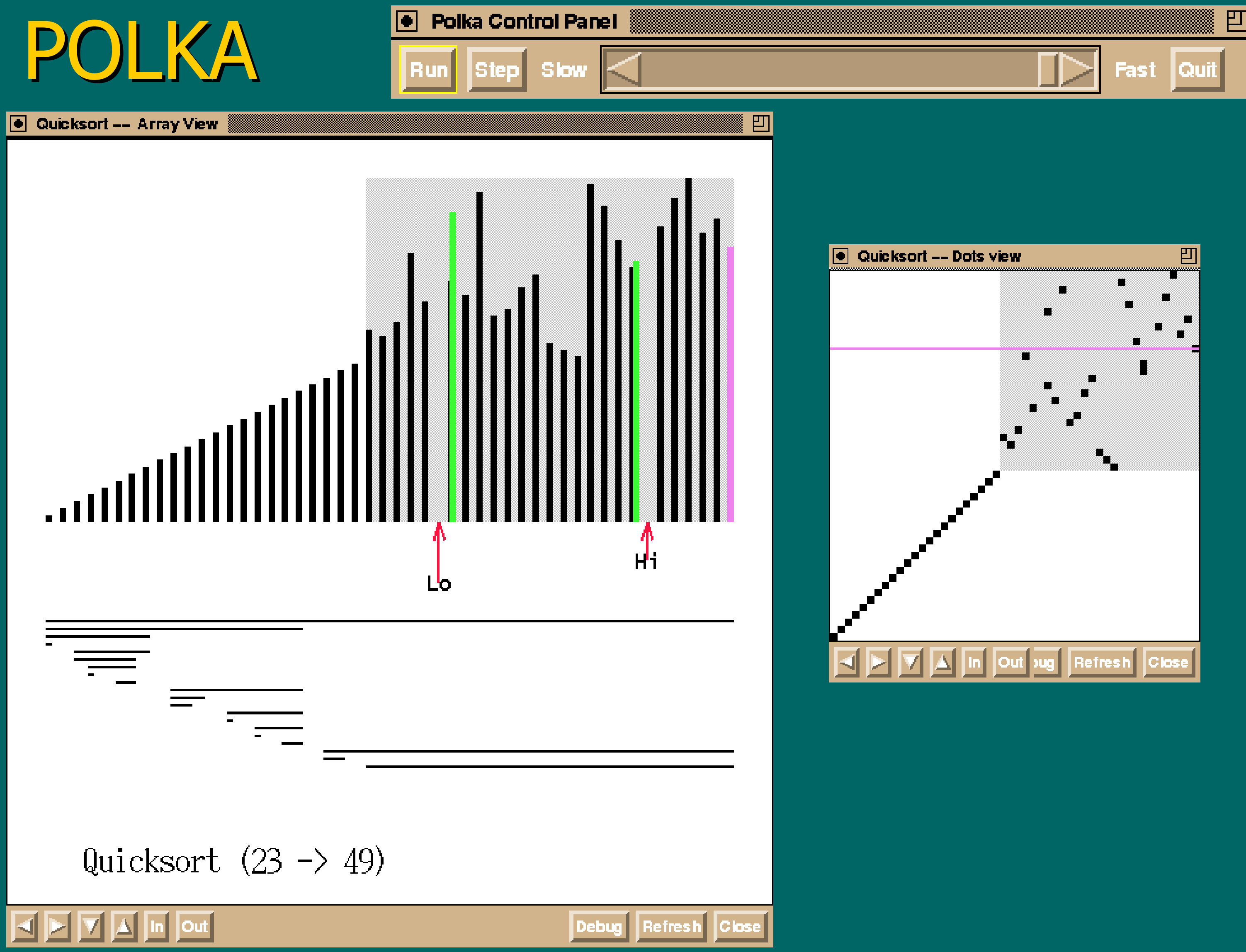
POLKA

- Improved animation design model
- Object-oriented paradigm
- Multiple animation windows
- Much richer visualization/animation capabilities

E. Kraemer

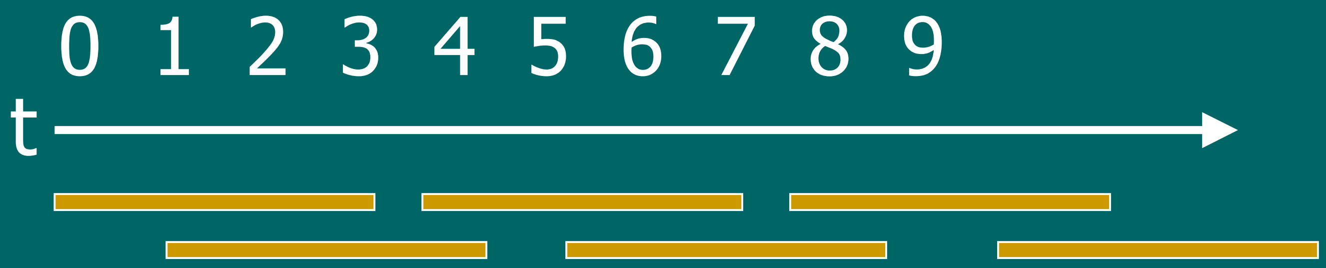
JPDC '93

POLKA


Polka Control Panel
Run
Step
Slow
Fast
Quit
Quicksort -- Array View
Lo
Hi
Quicksort (23 -> 49)
In
Out
Debug
Refresh
Close
Quicksort -- Dots view
In
Out
Refresh
Close

POLKA Model

- Location, AnimObject, Action
- Introduce explicit animation time (frame)



POLKA Programming Model

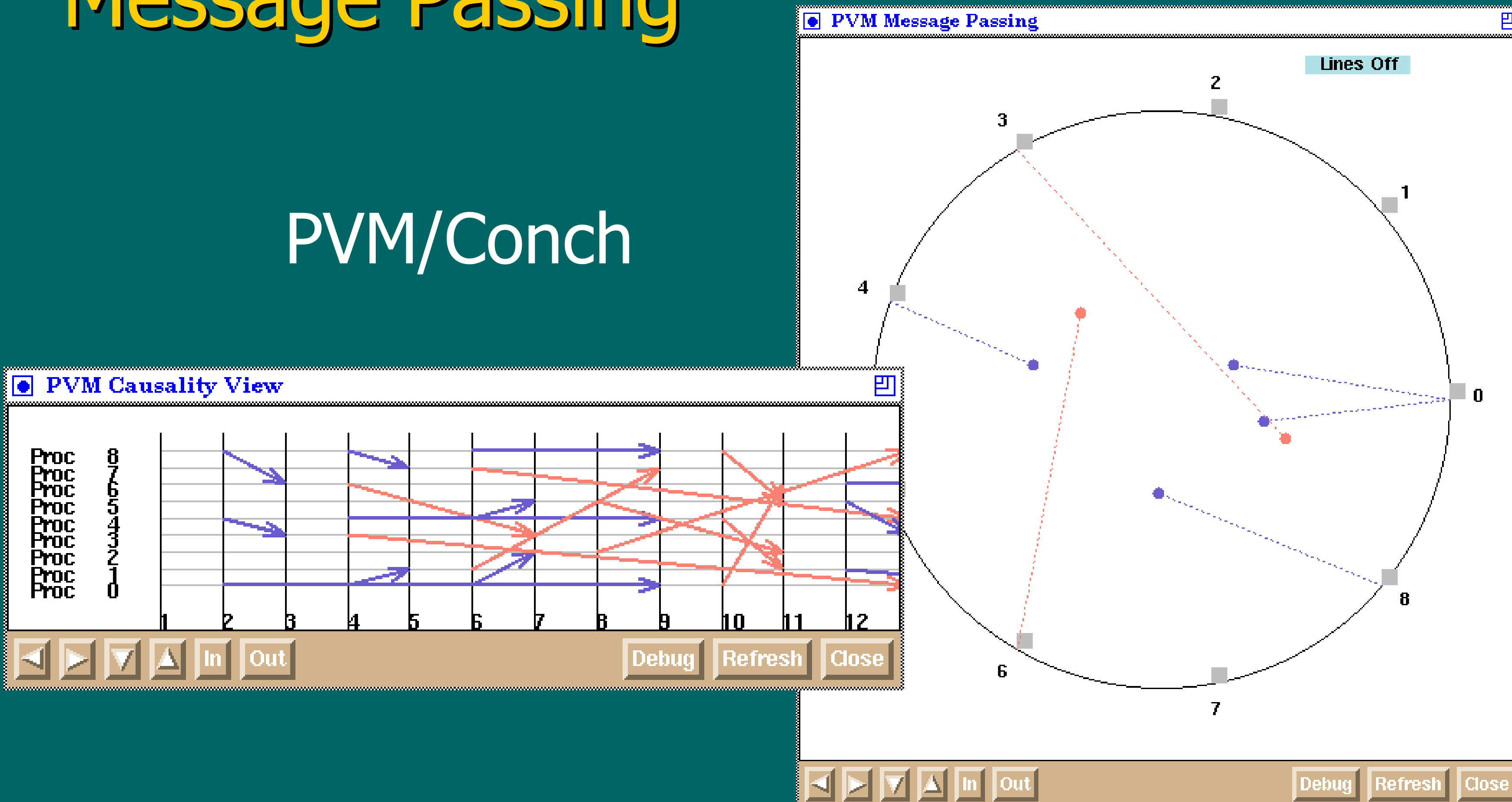
```
Circle *circ;
Loc *loc, *center;
Action *act;
int len;

circ = new Circle(this,1, 0.2,0.3, 0.1,"red", 1.0);
circ->Originate(time);

center = circ->Where(PART_C);
loc = new Loc(0.6, 0.5);

act = new Action("MOVE", center, loc, 20);
len = circ->Program(time, act);
time = Animate(time, len);
```

Concurrent Programs

- New model better, more flexible for illustrating concurrent program actions
- Polka used to build animation libraries for a variety of architectures/programming paradigms
  - message passing
  - shared memory
  - compiler-driven parallelism

Message Passing

PVM/Conch

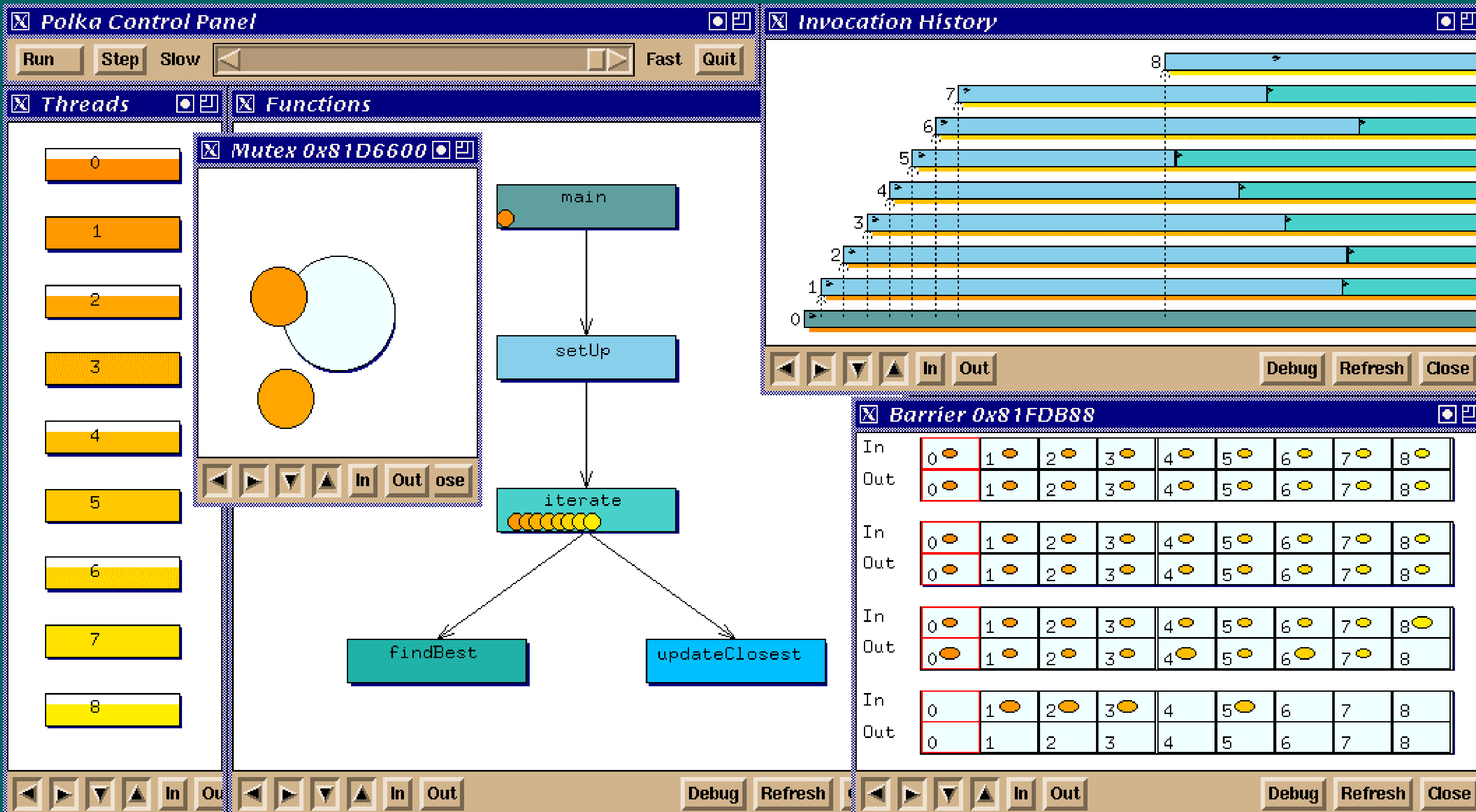


B. Topol
V. Sunderam

ICDCS '95
IJPDSN '98

Threads

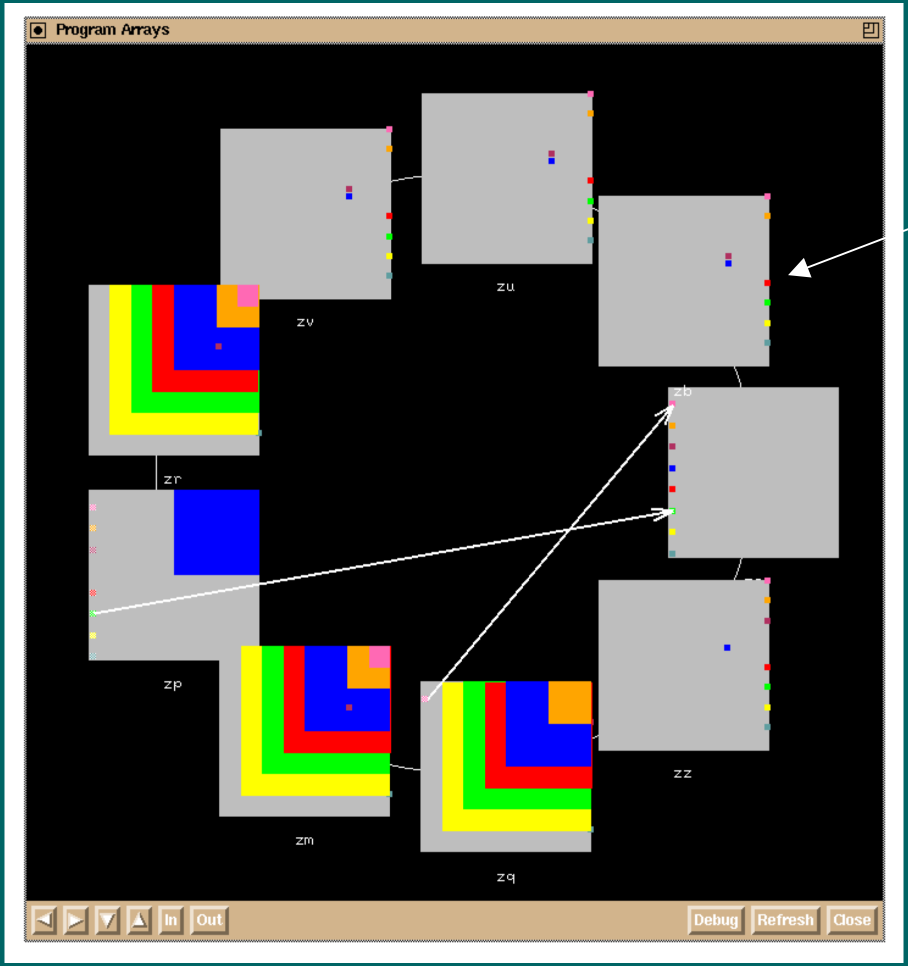
Pthreads



A. Zhao

TR

High Performance FORTRAN



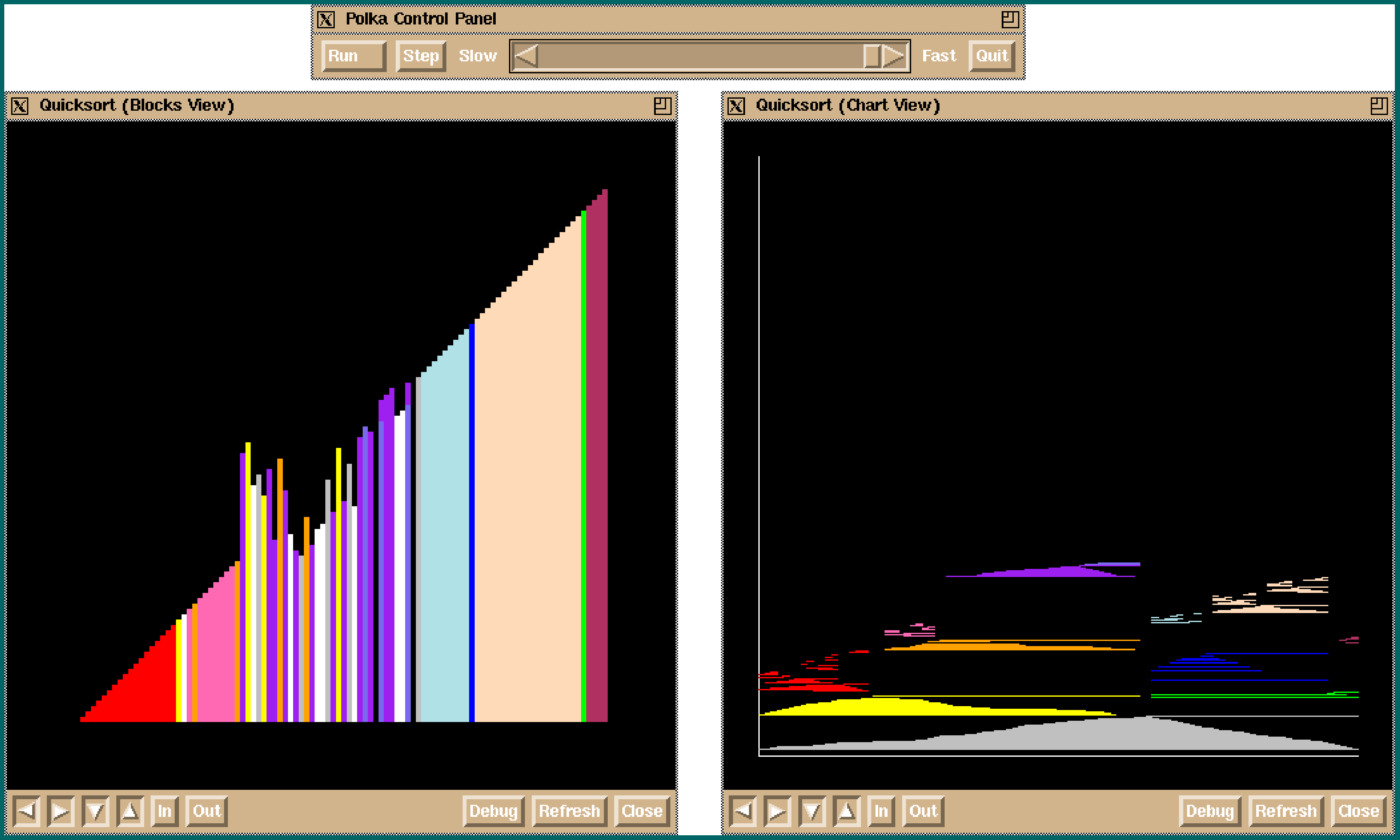
Arrays in program

Color represents process accessing that memory

W. Appelbe

Example Program Illustration

Parallel Quicksort



Array values view

History of exchanges view

Temporal Mapping

- Many temporal mappings exist from a concurrent program's execution to its animation
  - timestamp
  - serialized
  - maximum concurrency

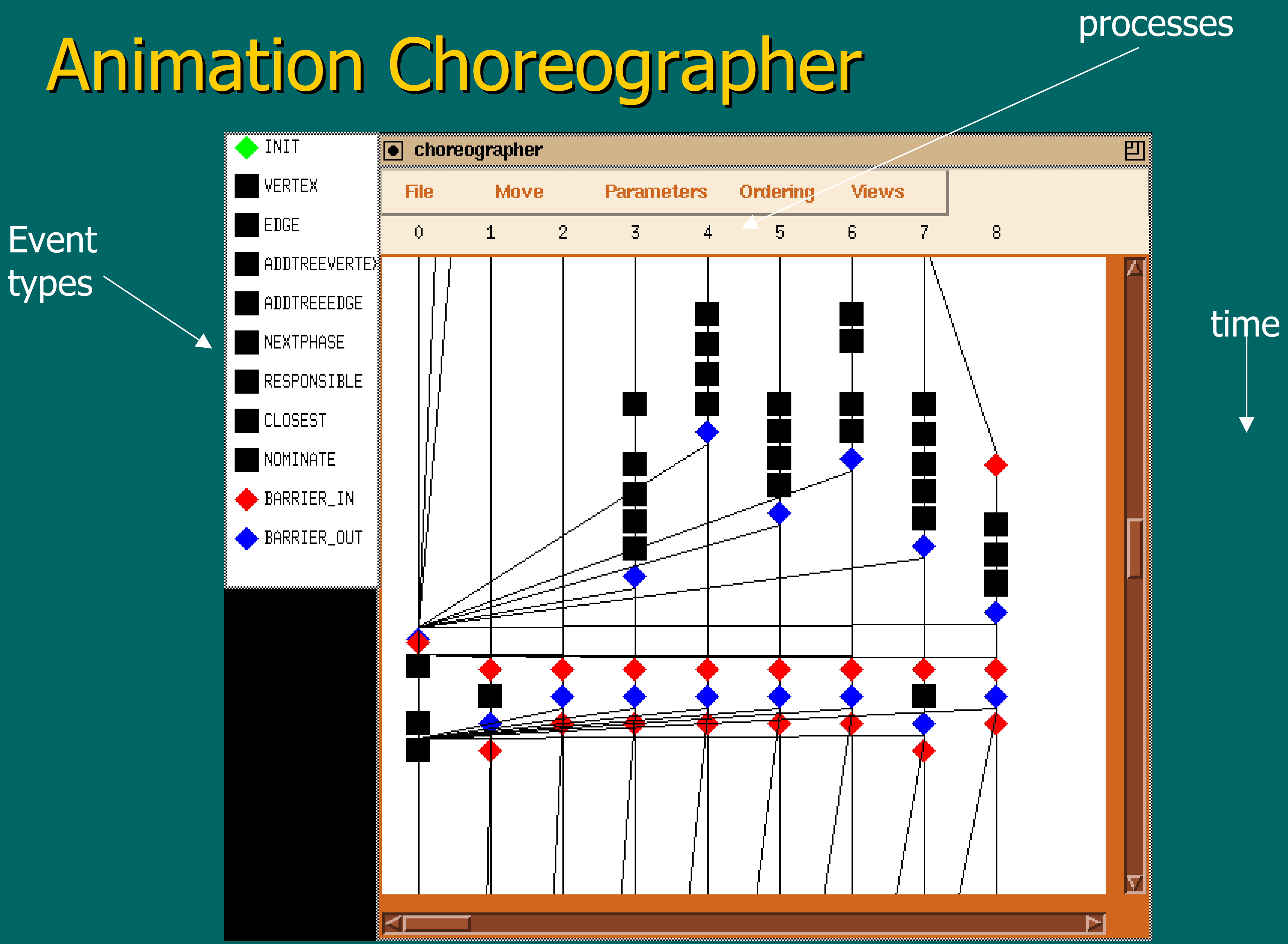
Animation Choreographer

- Visual depiction of program events and dependencies
- Allows viewer to manipulate events in time, then see animation that reflects that temporal order

E. Kraemer

IPPS '94
WPC '94
Concurrency '98

Animation Choreographer



PVaniM

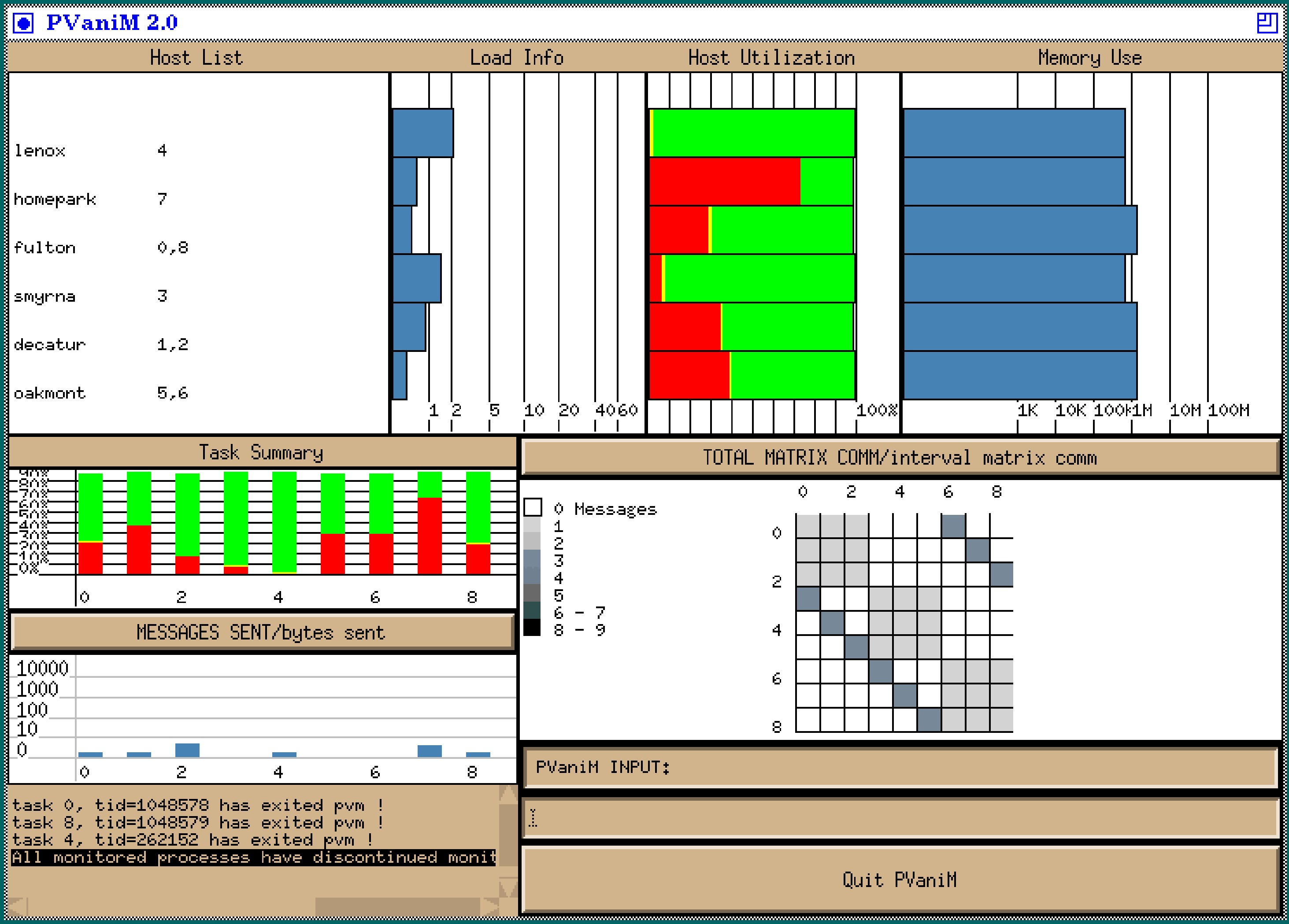
- Visualizing PVM programs on-line
- Must use sampling, not tracing due to sheer number of events
- Shows machine loads, host utilization, memory used, messages sent, communication patterns, etc.

B. Topol
V. Sunderam

Concurrency: P & E '98

PVaniM

Basic system UI



Polka-3D

- 3-D and VR version of Polka
- Same animation model
- Use third dimension to
  - Enhance visual aesthetics
  - Portray 3-D data
  - Encode more program attributes
- Not sure appropriate for algo anim

J. Wehrli

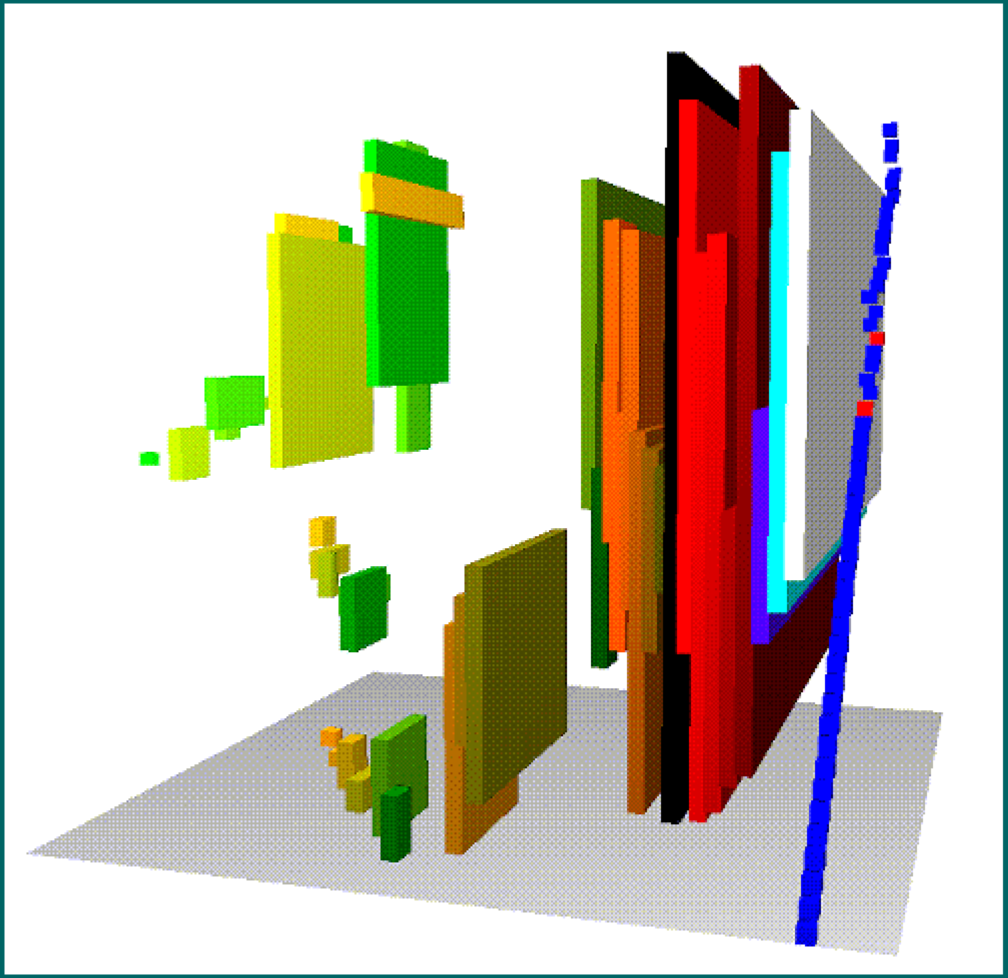
VL ‘93

POLKA-3D

Quicksort side view

Blue dots are as in 2-d view

Colored planes represent exhanges



Polka-RC

- Rather than time being animation frames, what if we use elapsed wall clock time?
  - Challenging under X Windows
  - Not clear if this is desirable for algorithm animations

S. McCrickard

Austral Comput Jrnl ‘95

Polka-RC

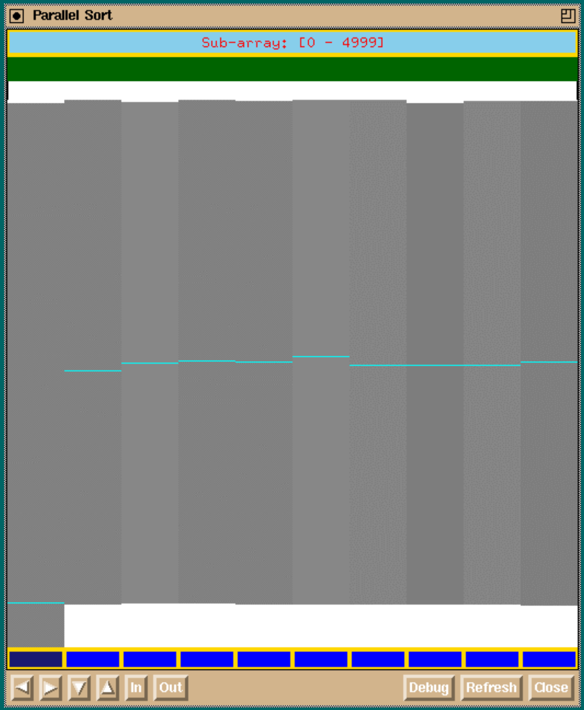
Programming model

```
Action a1("RESIZE", rect1,
         Traj(CLOCKWISE,0.2,-0.1,slowinout),
         START_AT, Now(),
         DURATION, Sec(1.5));

Action a2("MOVE", rect1,
         Traj(STRAIGHT,loc1,loc2,uniform),
         START_AFTER_END_OF, &a1, Sec(0.5),
         VELOCITY,50)

Action mov1("MOVE", elt[i],
          Traj(CLOCKWISE,from,to,uniform),
          START_AT, ASAP(), VELOCITY, 50);

Schedule(&a1);
Schedule(&a2);
Schedule(&mov1);
```

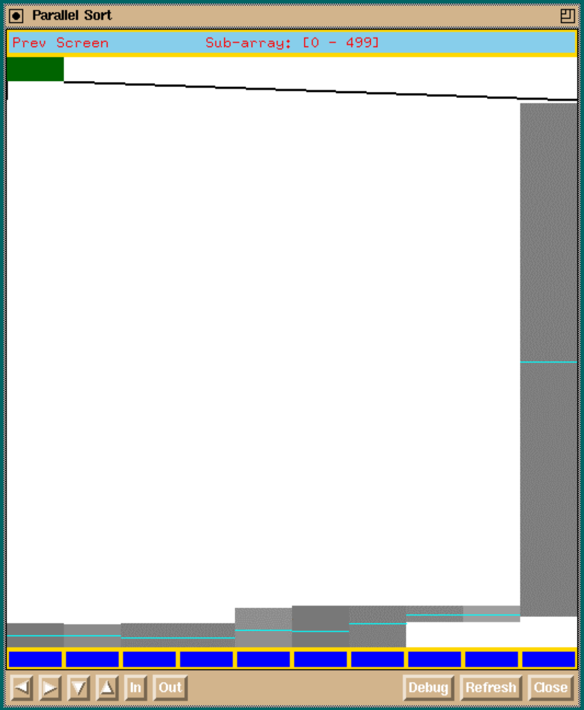
Visualizing Large Data Sets

Uses semantic zooming

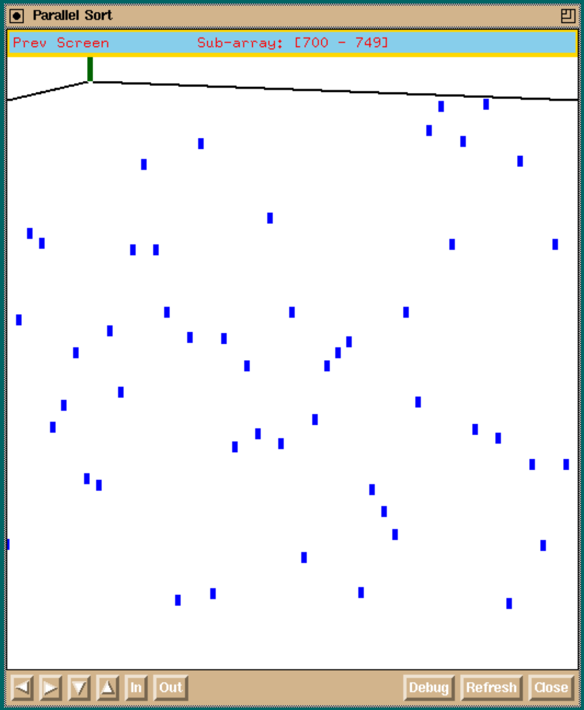
Sorting 5000 elements



View all data



Zoom to first 10%



Zoom further

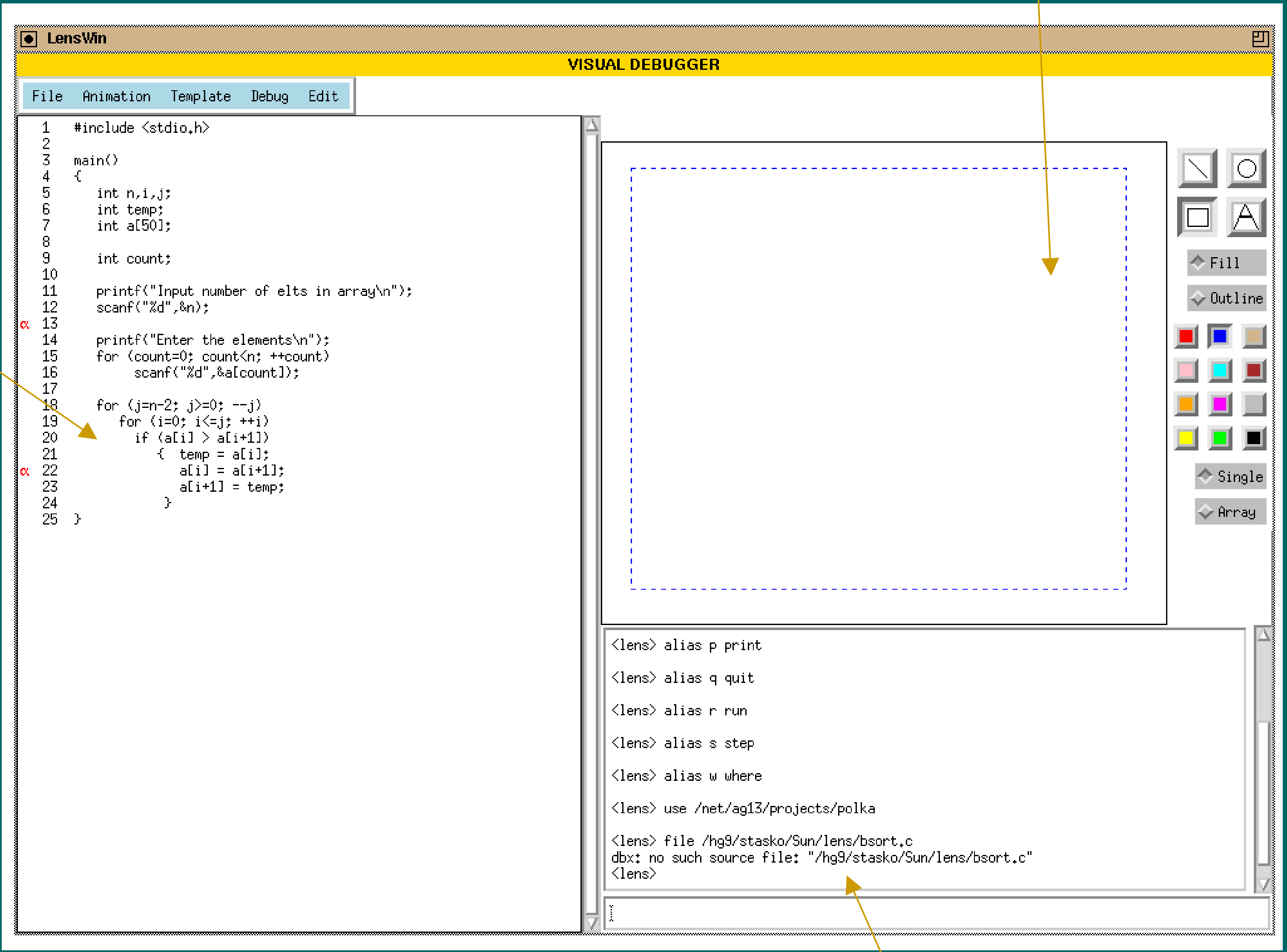
Visual Debugging

- Can we adapt algorithm animation capabilities to help programmers debug their code?
  – Want to go beyond data structure displays
  – Show semantics of program's domain
  – Easy specification by programmer

Lens

Animation design palette

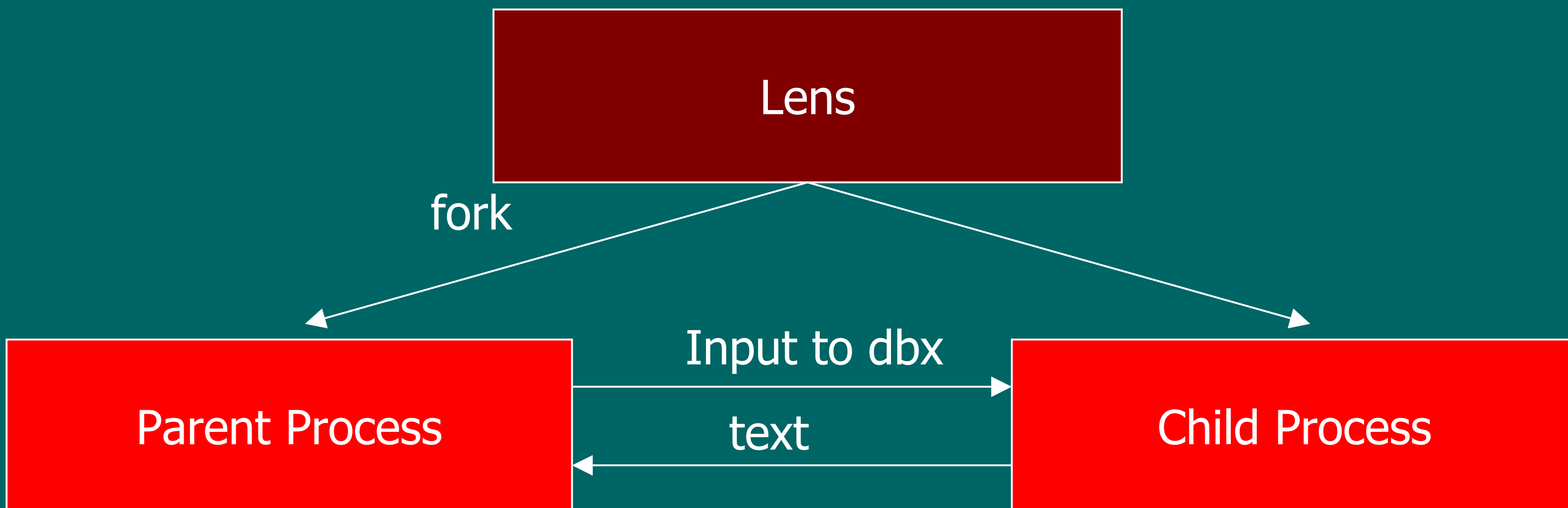
Source code



Debugger

Lens System

- Architectural model



Output from dbx

Runs animation, controls UI

Runs dbx, executes program

S. Mukherjea

ICSE ‘93
ToCHI ‘94

Empirical Evaluation of Algorithm Animations as Learning Aids

- Can we show that algorithm animations can help students learn?
- Compare learning with animation to learning without
- Measuring understanding is difficult
- Four main studies

1. Pairing Heaps

- Classical experimental design
- Just having animation doesn't make learning happen
- Difficult for student to leverage animation of complex algorithm when they don't understand algorithm and visual mapping yet

A. Badre
C. Lewis

InterCHI '93

2. Introductory Algorithms

- Sorting, graphs
- Interaction is the key
  - Students who enter their own data sets into the algorithm benefited from animation

A. Lawrence
A. Badre

VL ‘94

3. Binomial Heap & DFS

- Is animation like prediction?
- Does it help the student to anticipate what will happen next and learn from that?
- Mixed data, some support

M. Byrne
R. Catrambone

Computers & Ed '99

4. Homework Scenario

- Provide student with learning objectives/ questions up front
- Give unlimited work time
- Animations appear to help motivation
  - Make a complex algorithm less intimidating
  - Animation helped learning

C. Kehoe
A. Taylor

IJHCS ‘00

Student-Built Animations

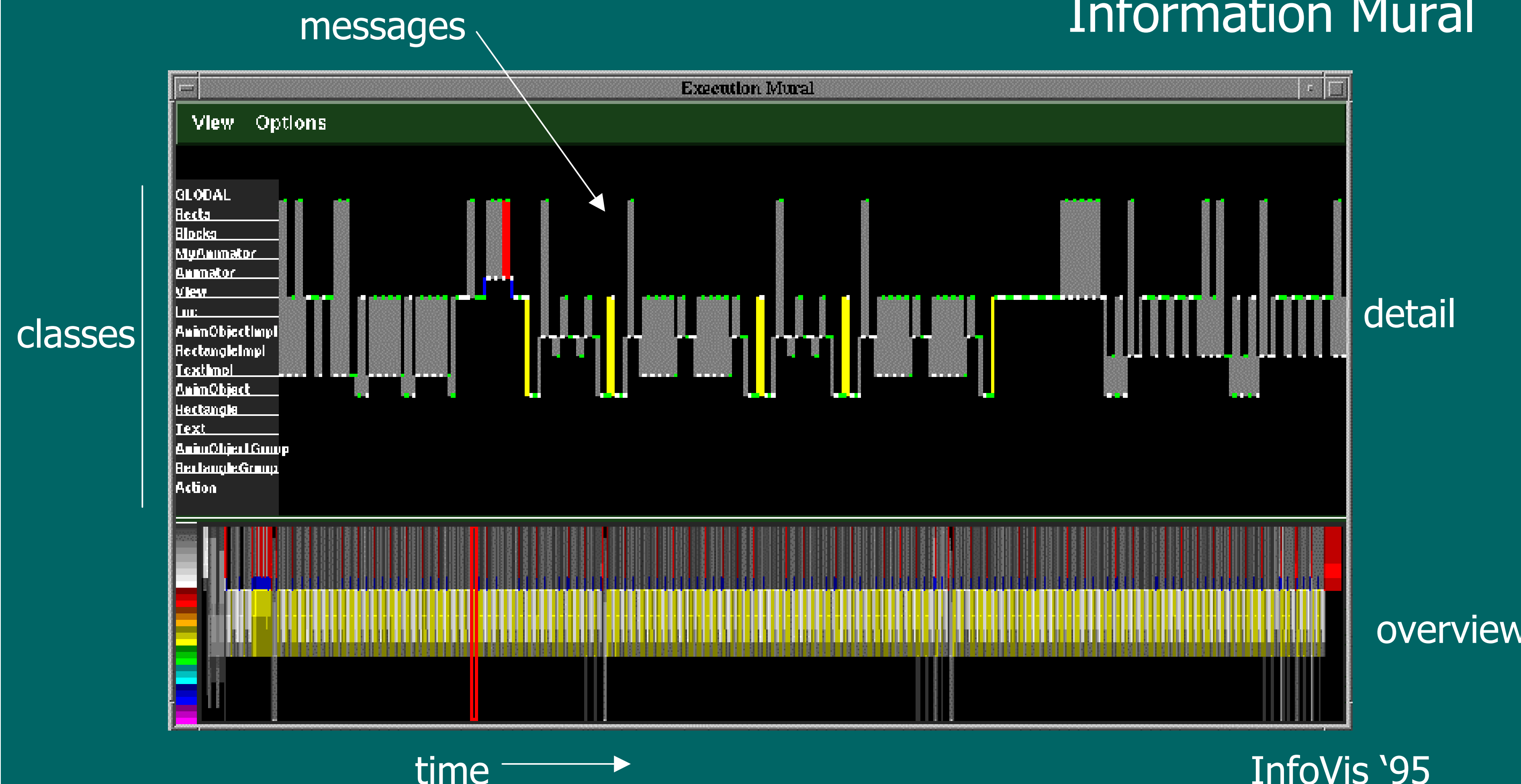
- Samba
  - Simple animation scripting language

```
circle 1 0.8 0.8 0.1 red half
line 2 0.1 0.1 0.2 0.2 green thin
rectangle 3 0.1 0.9 0.1 0.1 blue solid
text 4 0.0 0.0 0 black Hello
circle 6 0.3 0.3 0.2 wheat solid
triangle 7 0.5 1.0 0.6 0.8 0.4 0.9 cyan solid
bigtext 8 0.2 0.2 0 black Some Big Text
moveto 1 6
moverelative 3 0.05 -0.4
jumprelative 4 0.4 0.4
lower 1
color 6 blue
```

  - Embed print statements in any program to generate

Visualizing Large O-O Programs

Information Mural



D. Jerding

InfoVis '95
ICSE '97
ToVCG '98

Current State of Software Visualization

- Research continues…
- Some use of algorithm animations as pedagogical aids
- Program visualization trickling into commercial tools

What’s Needed? (AA)

- Focus on interactive tools
- Simpler animation construction
- Empirical validation of value

What's Needed? (PV)

- Better analysis of what software developers want and need
- Flexible displays providing overview and detail
- Improved tracing/monitoring/analysis capabilities

Acknowledgments

- Research supported by National Science Foundation, Sun, and the GVU Center

- More info
  - www.cc.gatech.edu/gvu/softviz
  - stasko@cc.gatech.edu