Visualizing the Performance of Parallel Programs

Michael Heath *and* Jennifer Etheridge

Presented by **Chau-Wen Tseng**

Department of Computer Science
University of Maryland, College Park

Motivation

- Parallel program performance
  - often lower than expected
  - analysis complex
  - lots of performance data
- Graphical visualization
  - aids comprehension
  - insight into performance
  - help find bottlenecks

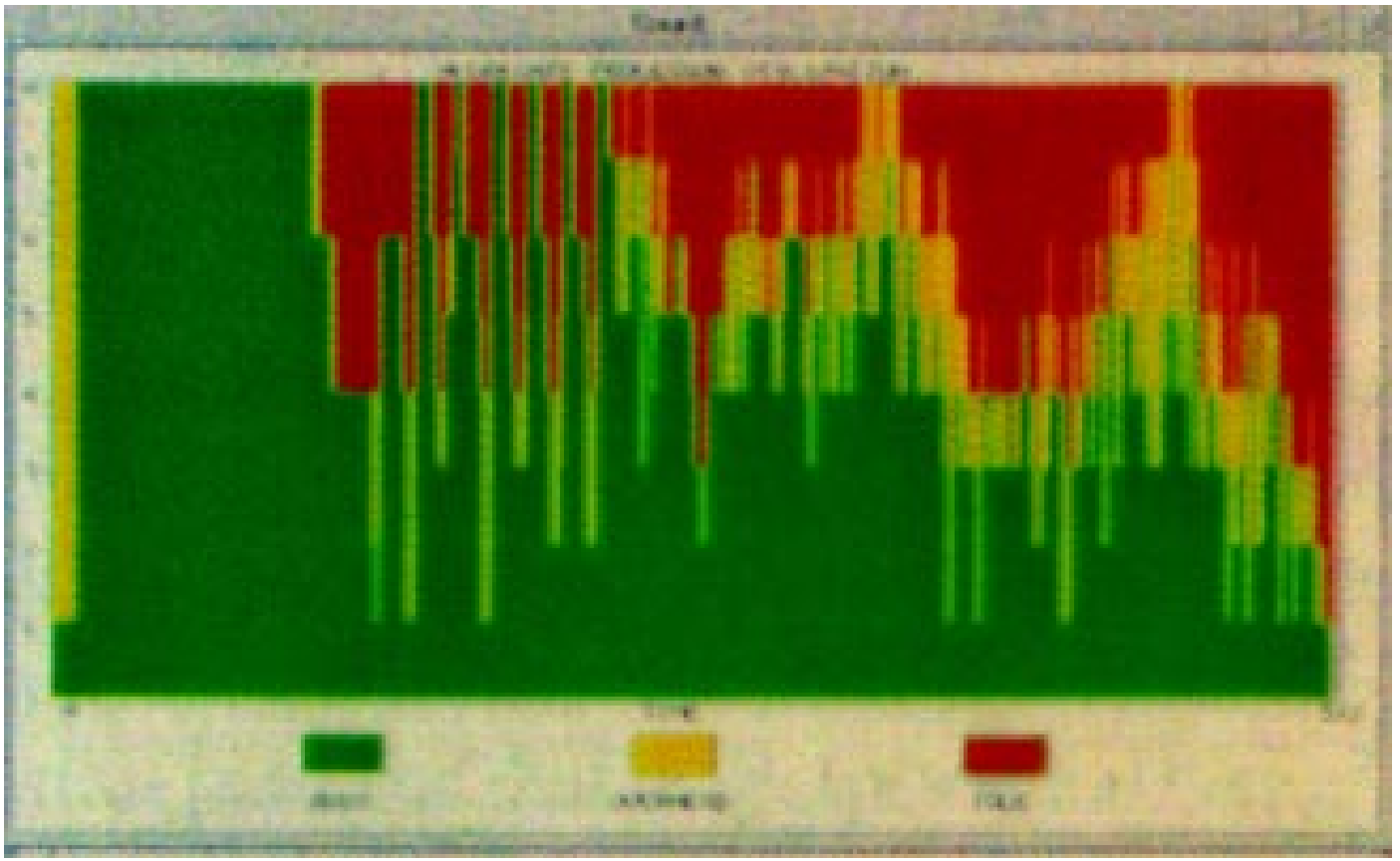
Visualization Issues

- High-dimensional data
  - multiple metrics
  - many interacting processors
  - varying rates of change
- Multi-level semantic correlation
  - high level user program
  - low level compiler transformations
- Mixed data
  - numerical data (discrete, continuous)
  - categorical data (states, classifications)

ParaGraph

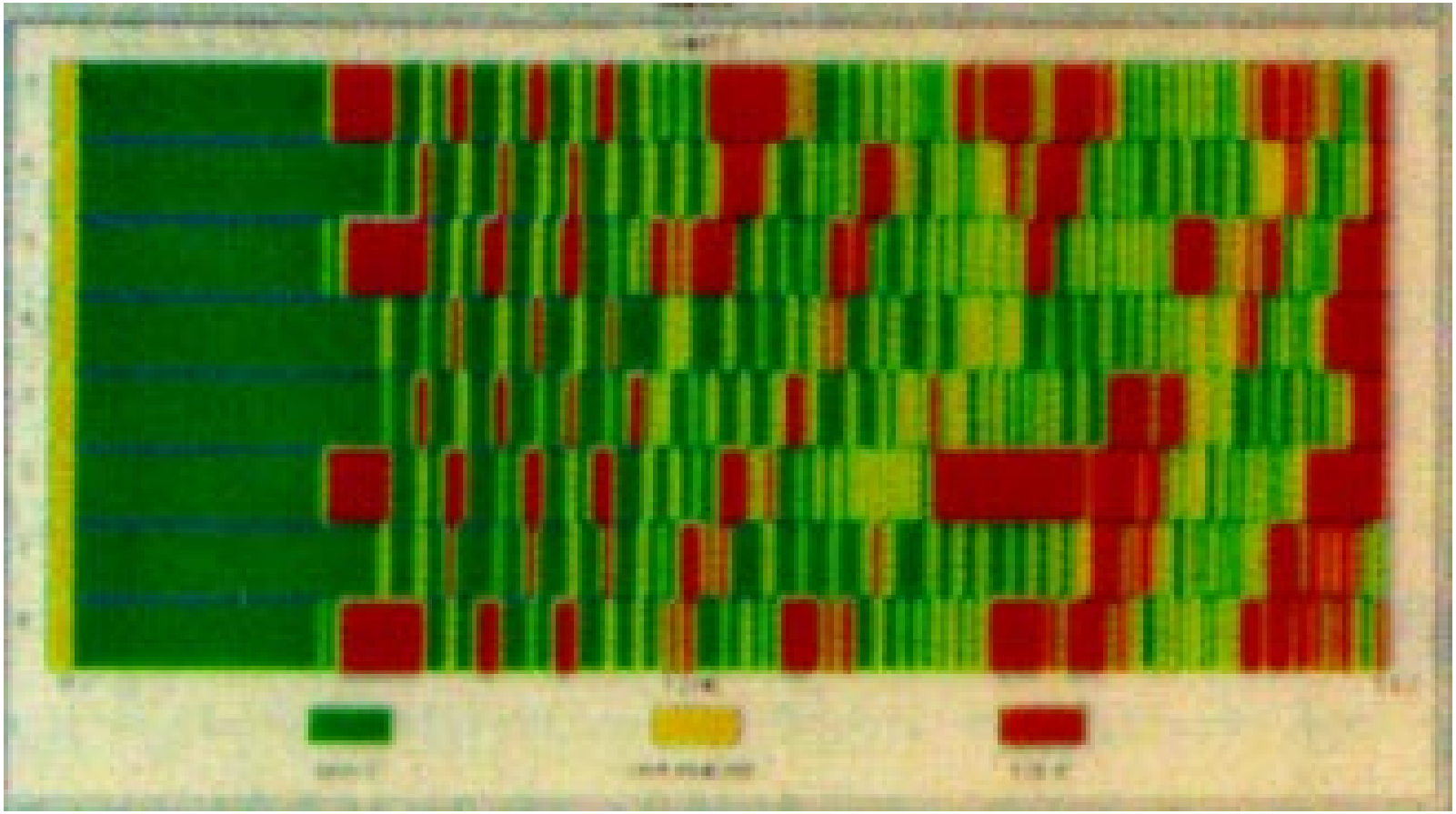
- Approach
  - instrument PICL message library
  - gather trace data, use timestamps
  - limited to message-passing machines
- Design
  - interactive, event-based
  - multiple displays
  - static displays & dynamic animations
- Experiments
  - sparse Cholesky factorization on iPSC/2
  - utilization displays, communication displays

Utilization Processor Count



- **Y-axis** **# procs in (busy/overhead/idle) state**
- **X-axis** **elapsed time**

Utilization Gantt Chart

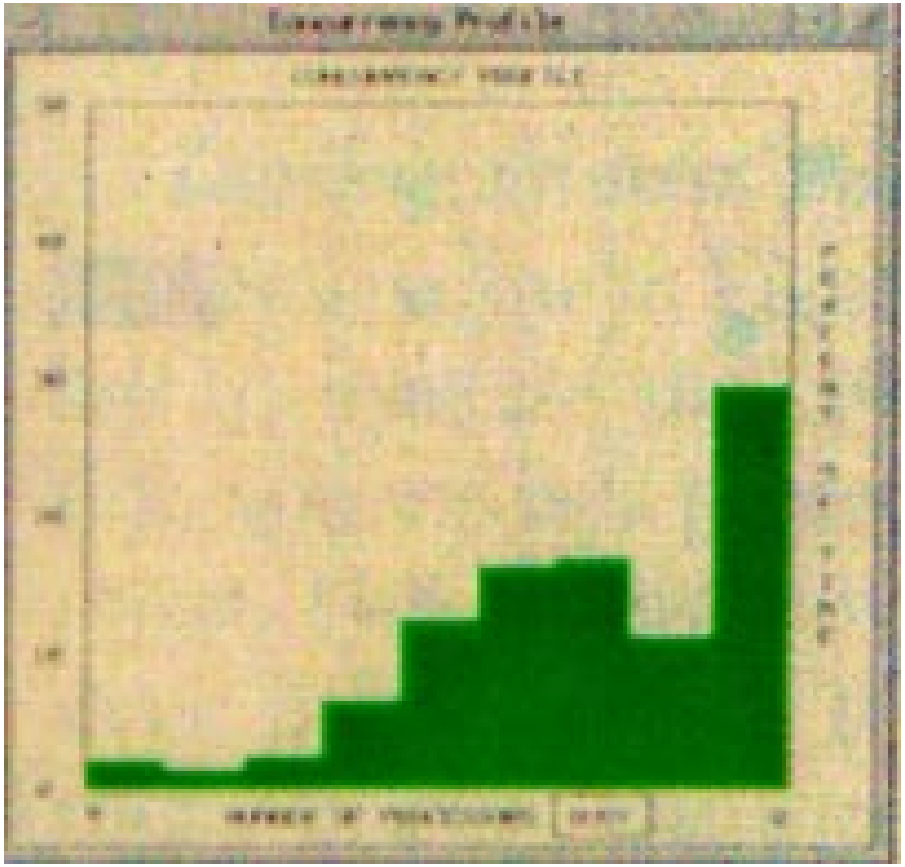


- **Y-axis** each proc in (busy/overhead/idle) state
- **X-axis** elapsed time

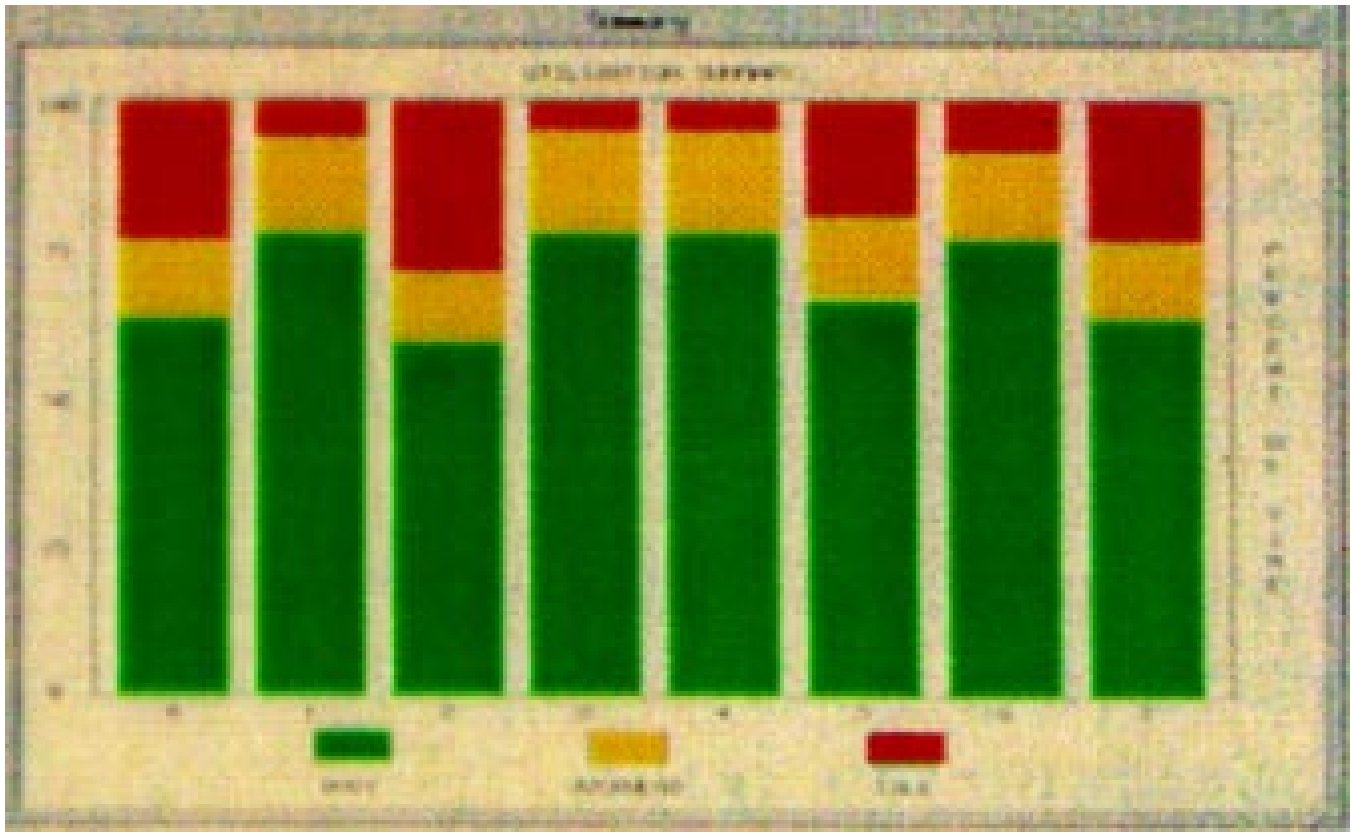
Concurrency Profile

- Y-axis % time
- X-axis # procs

% of time certain # procs were in busy state

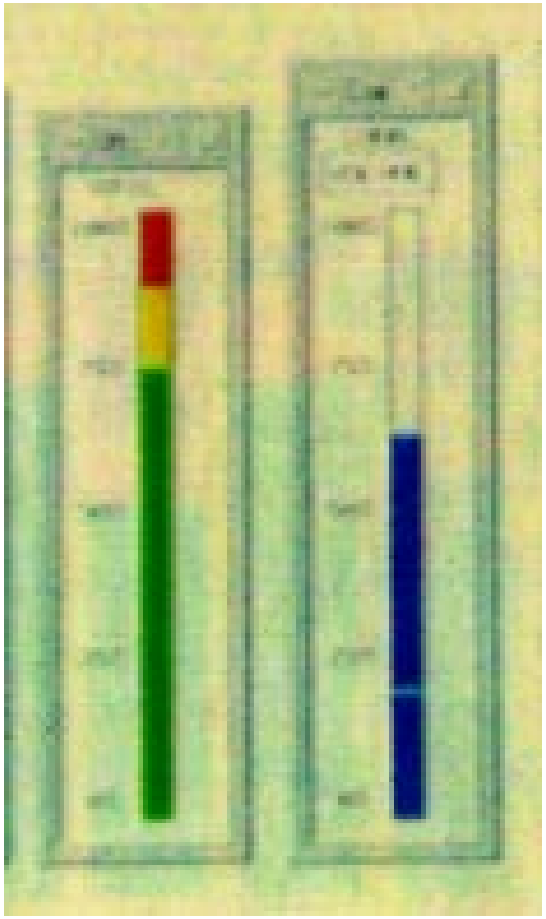


Utilization Summary



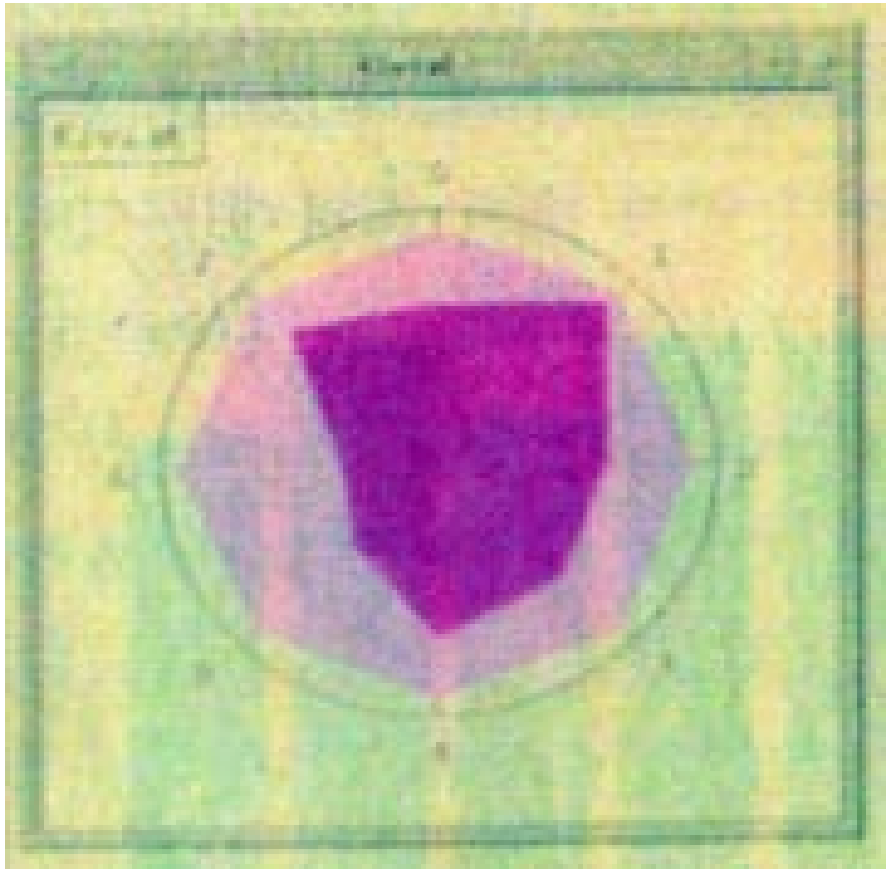
- **Y-axis** % time in (busy/overhead/idle) state
- **X-axis** each proc

Utilization Meters



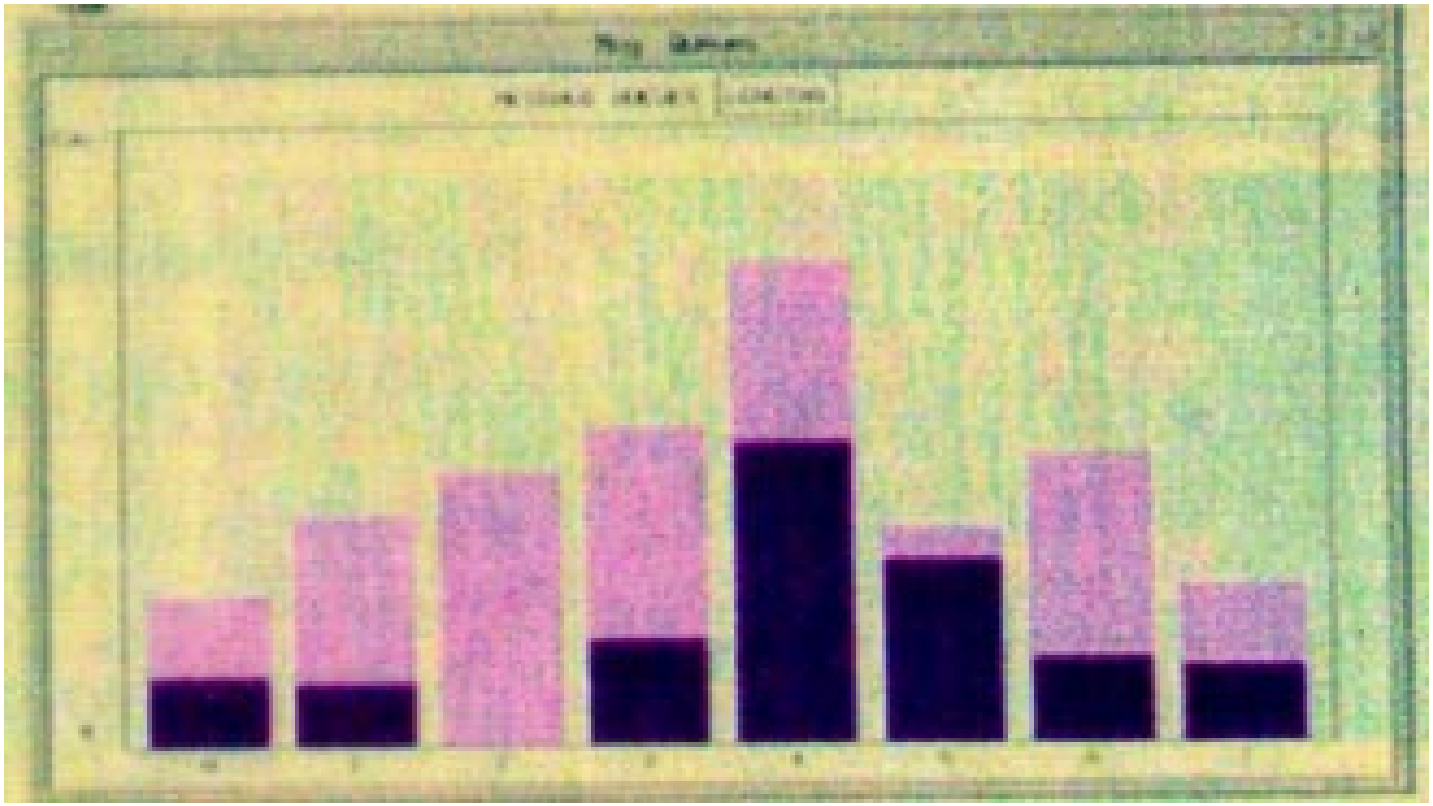
- **Y-axis 1 % procs in (busy/overhead/idle) state**
- **Y-axis 2 % communication volume**

Kiviat Diagram



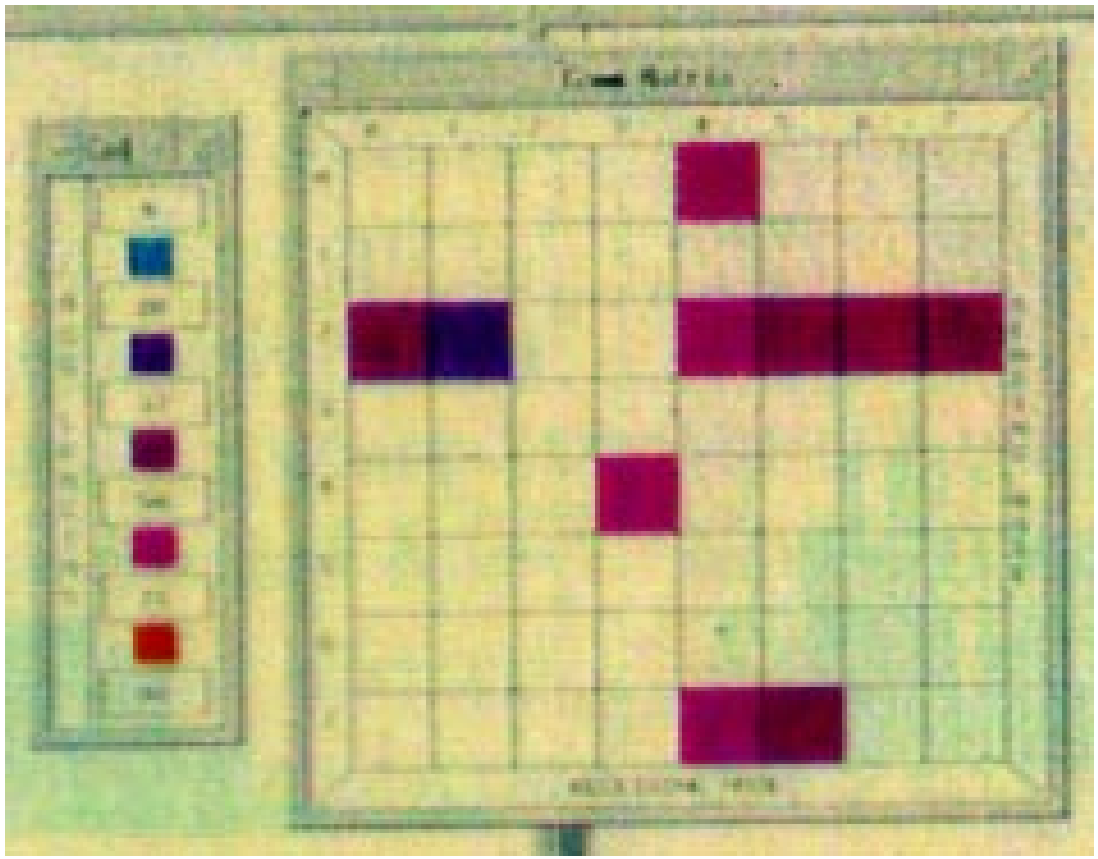
- Spoke — each processor
- Length of spoke — % load of processor

Message Queues



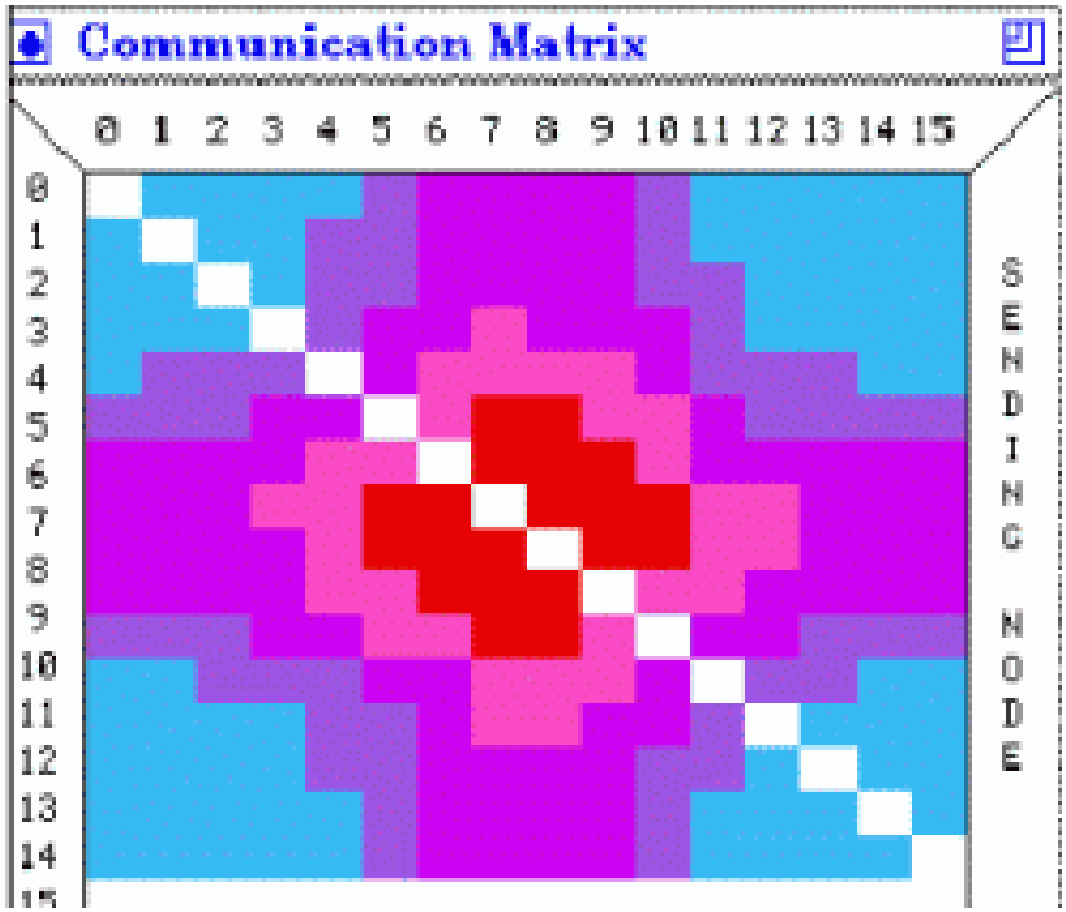
- **Y-axis** size of processor message queue
- **X-axis** each processor

Communication Matrix



- Coordinates (x,y) message from x to y
- color message size

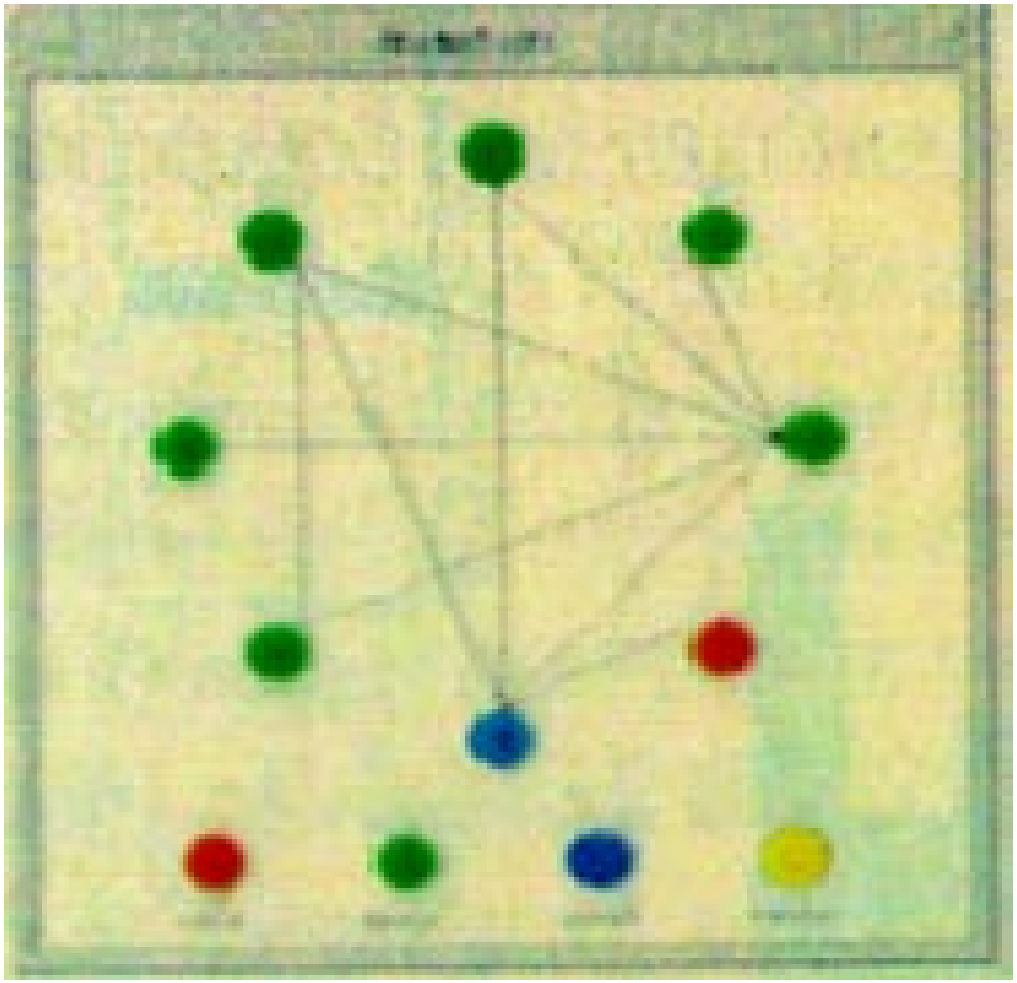
Communication Matrix



- **Coordinates (x,y)** **message from x to y**
- **color** **message size**

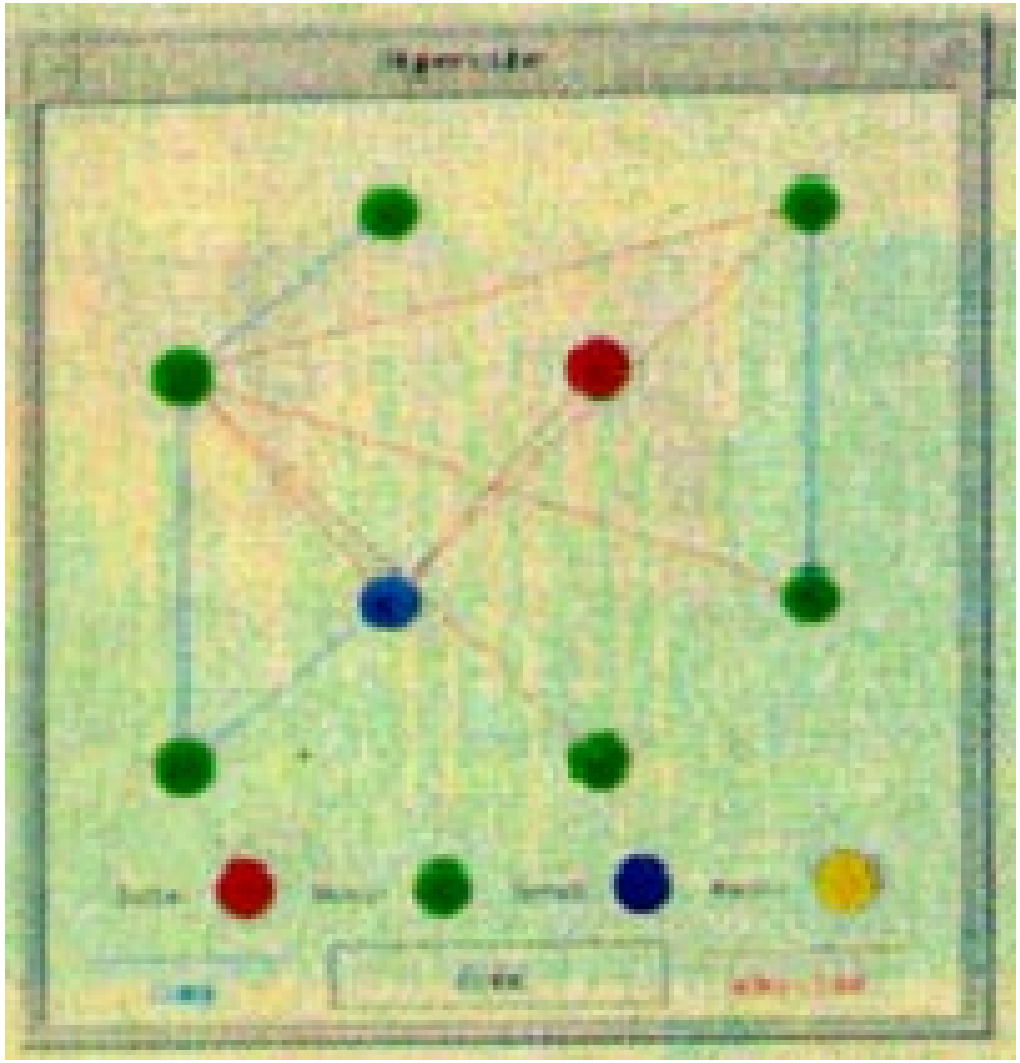
Multiprocessor Animation

- circles — each proc
- lines — messages
- color — state

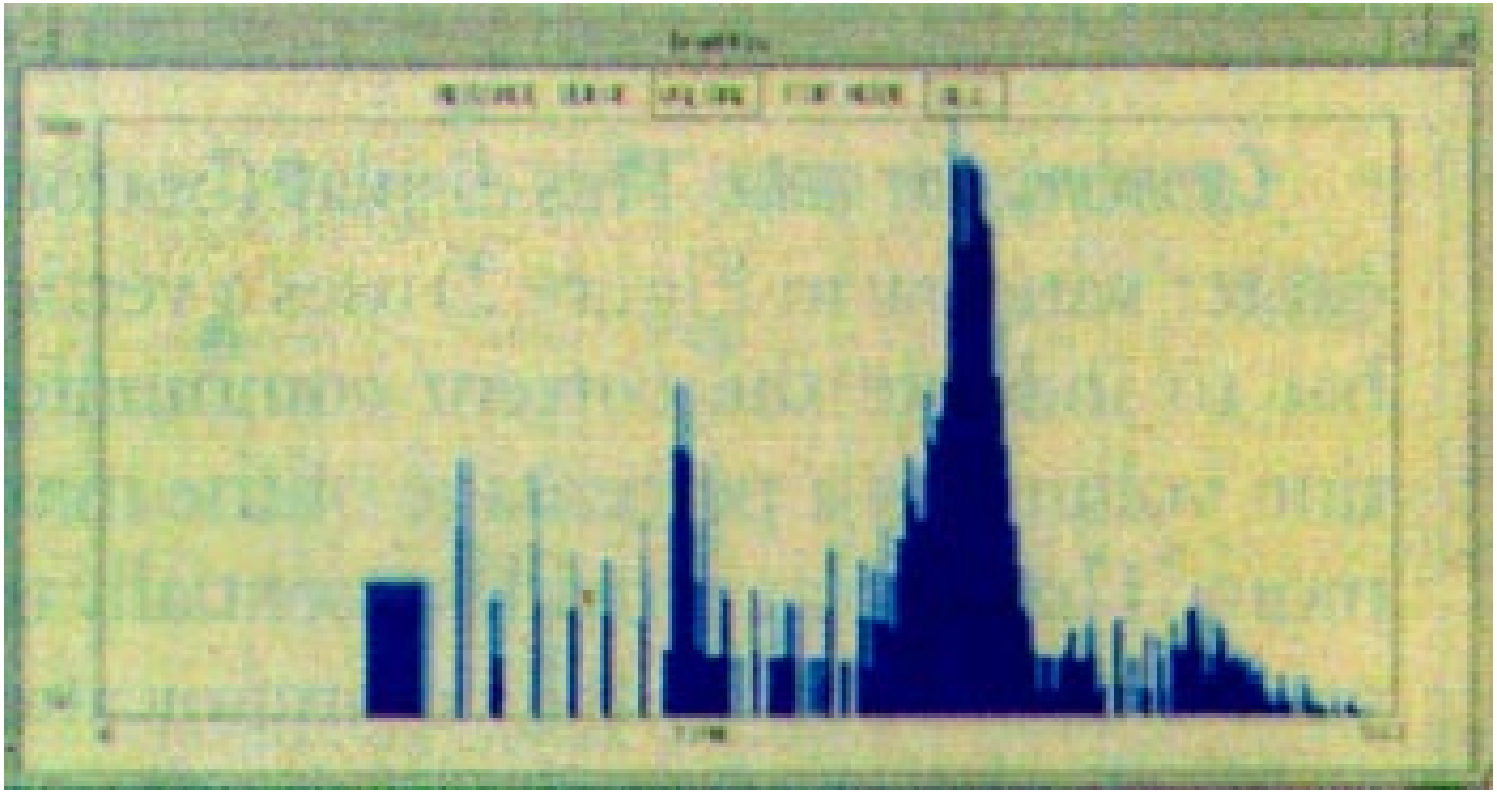


Multiprocessor Animation (Hypercube)

- circles — each proc
- lines — messages
- color — state

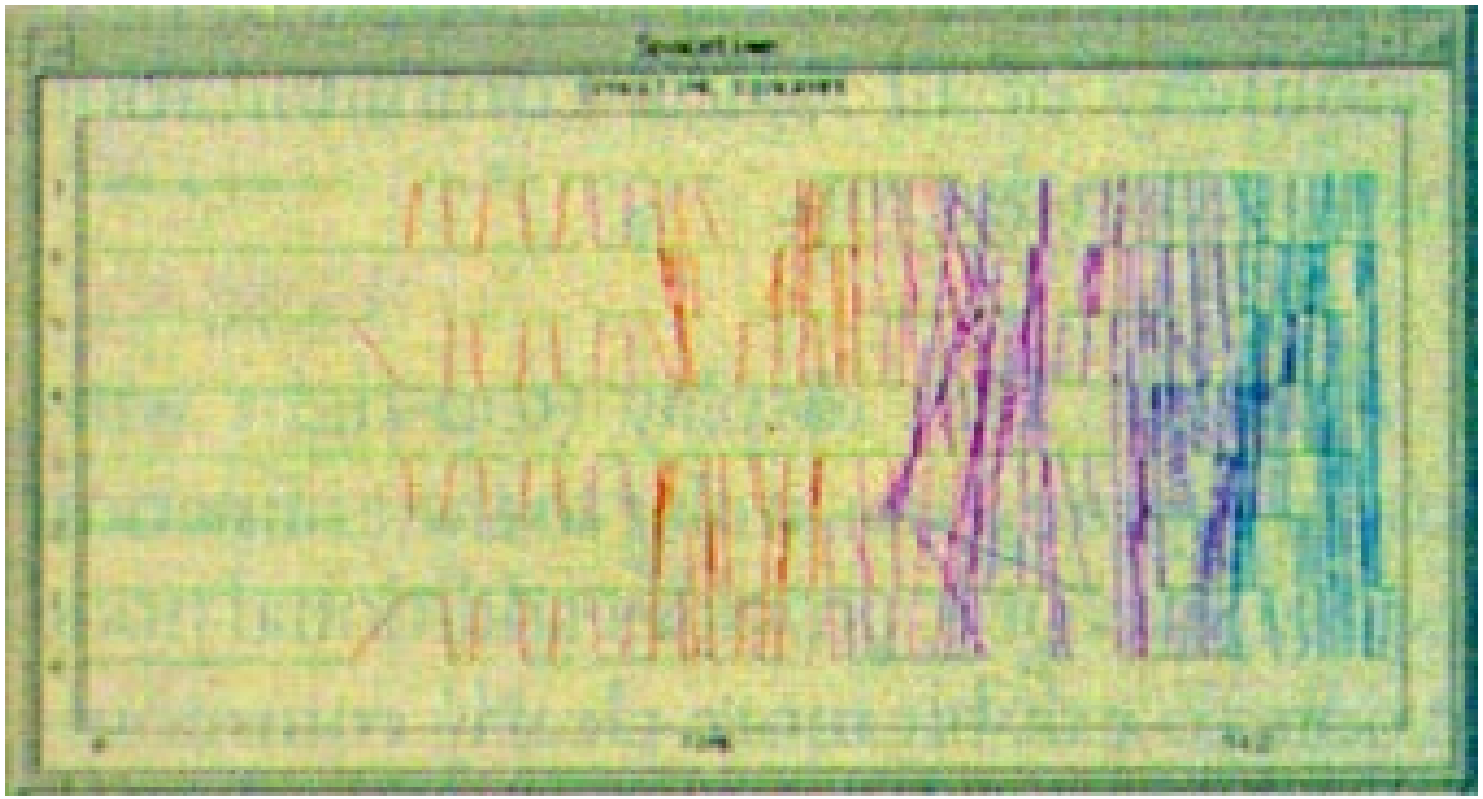


Communication Traffic



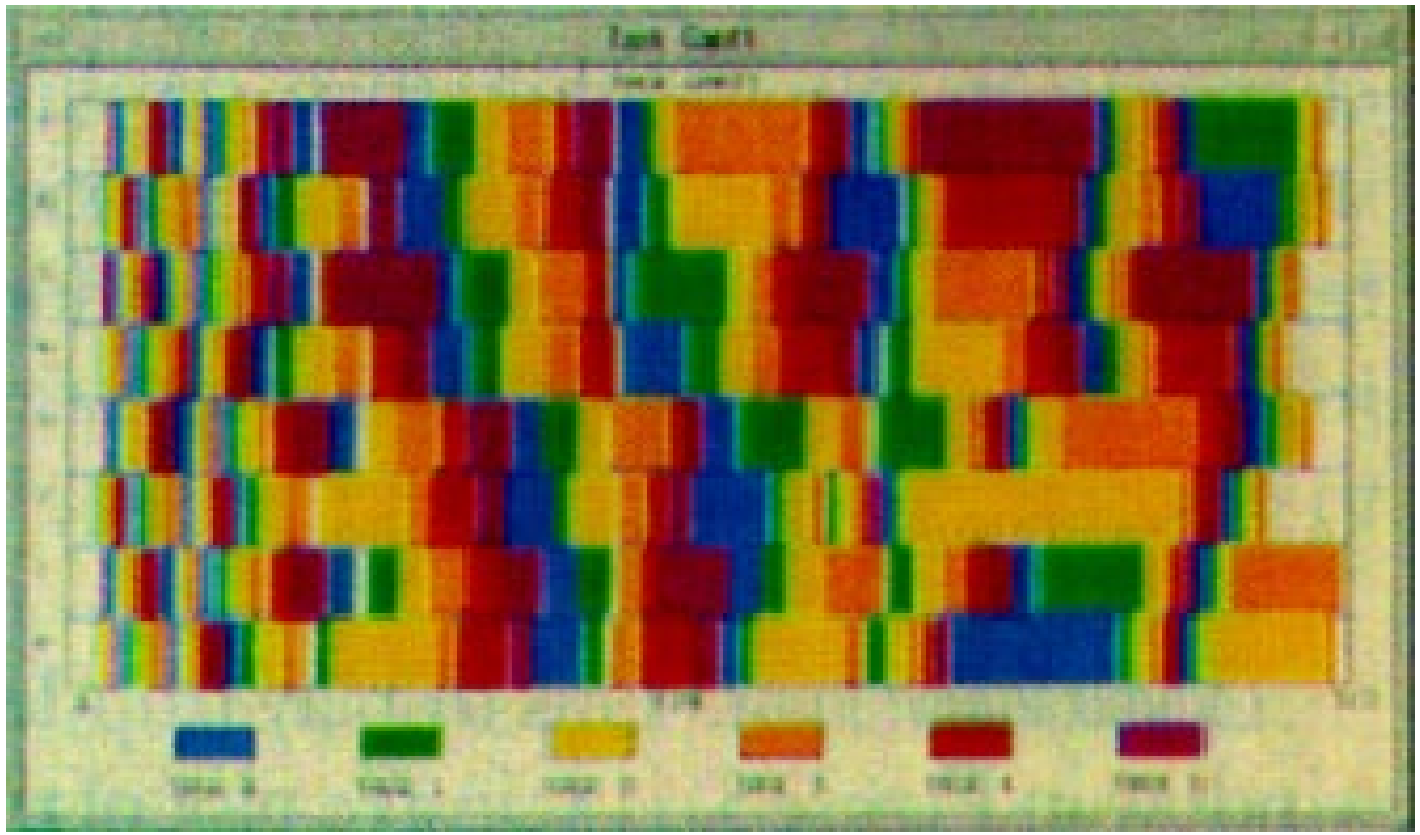
- **Y-axis** **total communication traffic**
- **X-axis** **elapsed time**

Space-time Diagram



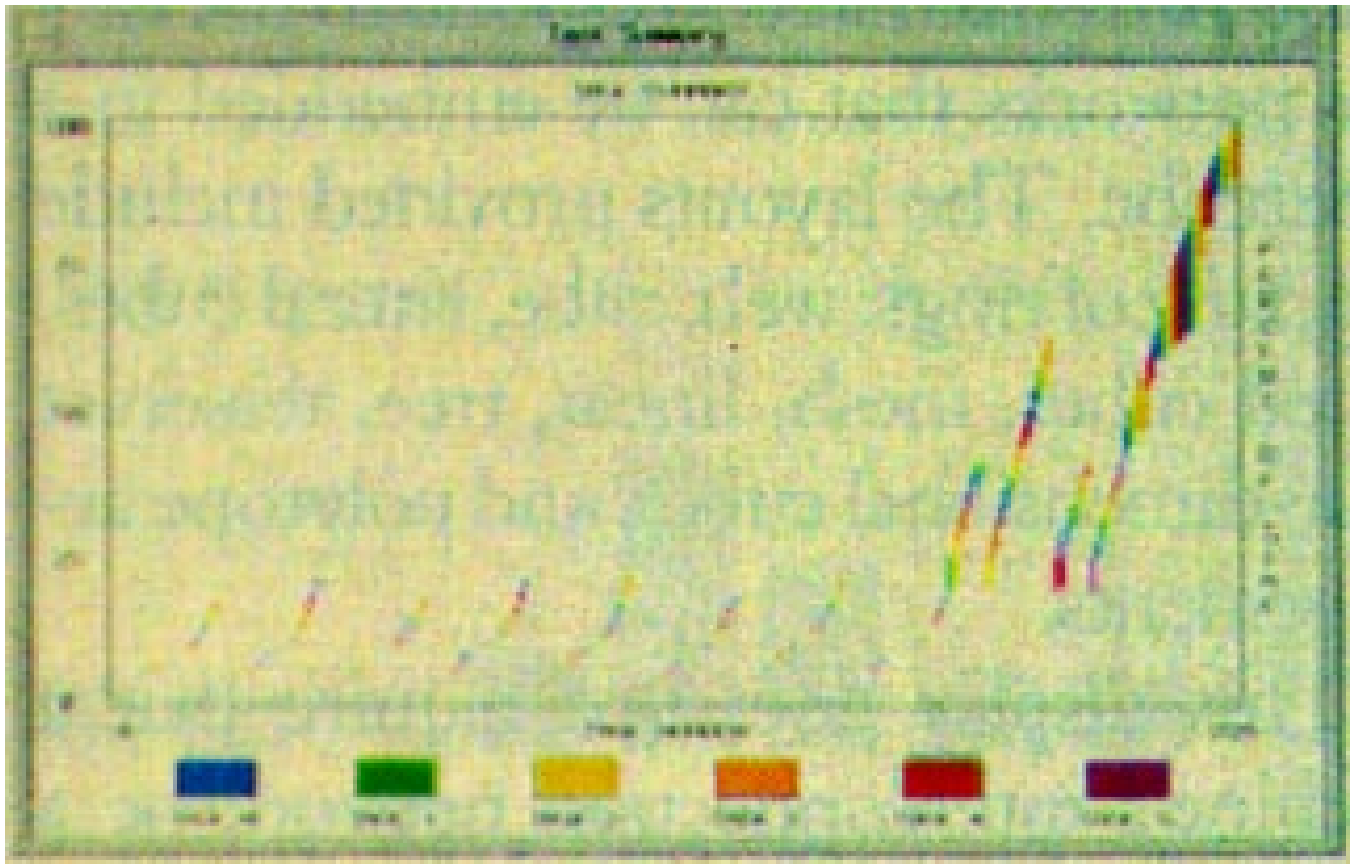
- **Y-axis** **each proc**
- **X-axis** **elapsed time**
- **lines** **messages**

Task Gantt



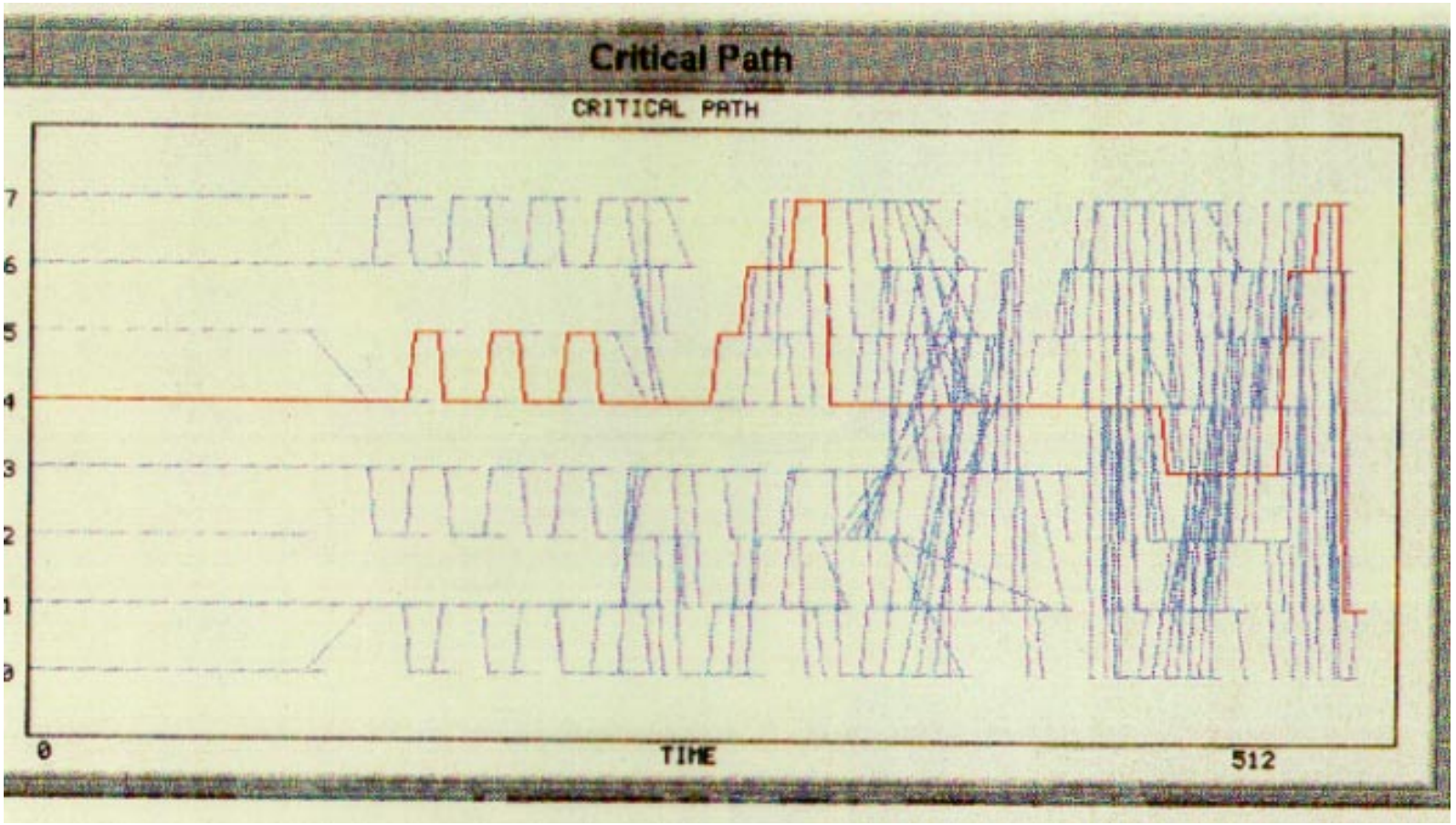
- **Y-axis** each proc in task (annotated by user)
- **X-axis** elapsed time

Task Summary



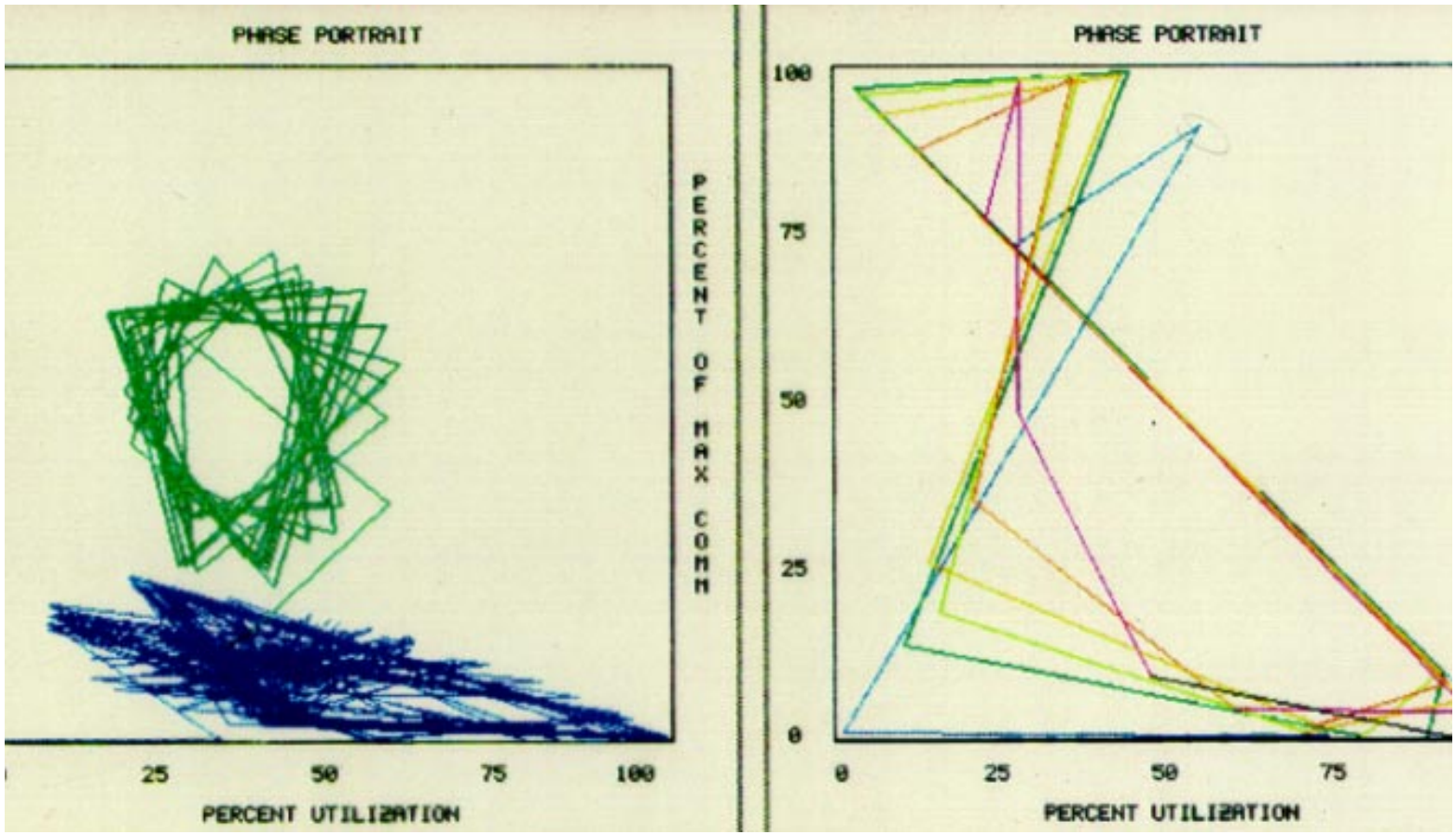
- **Y-axis** **% total execution time**
- **X-axis** **elapsed time**

Critical Path



- **Y-axis** **each proc**
- **X-axis** **elapsed time**

Phase Portrait



- **Coord (x,y)** — **x = % busy proc, y = % comm**
- **color** — **task**

Summary

- Performance visualization
  - many possibilities
  - static & dynamic
- Key question
  - how much does it help?