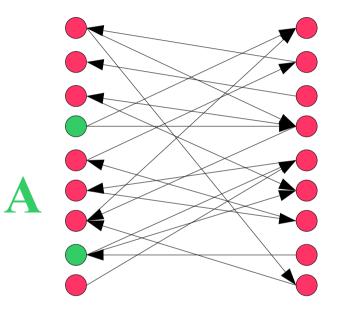
# Implementing Valiant's Neuro-Logic

Jake Beal December, 2005

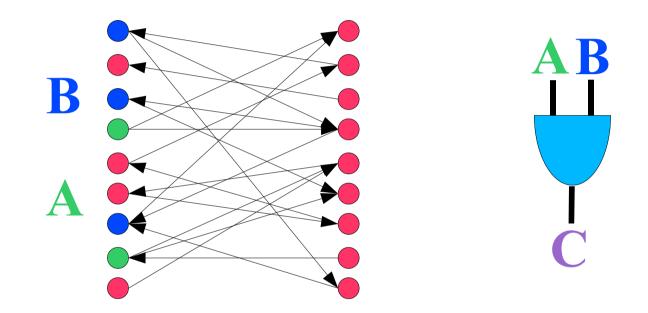
# How could neural circuits work?

- Many components (10^10 neurons)
- Large fan-in (24K-80K neighbors)
- Slow cycling (1-10 milliseconds)
- Low precision wiring
- Low synapse strength (0.3-20% of threshold)

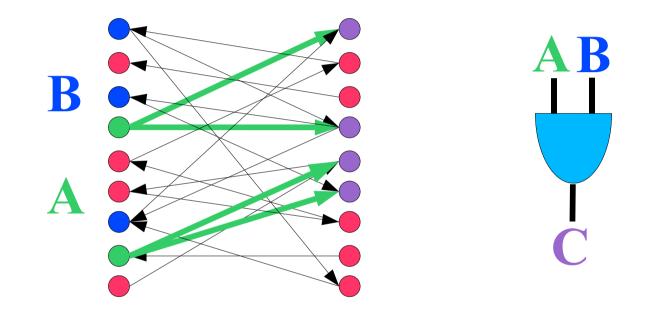
### Valiant's Approach



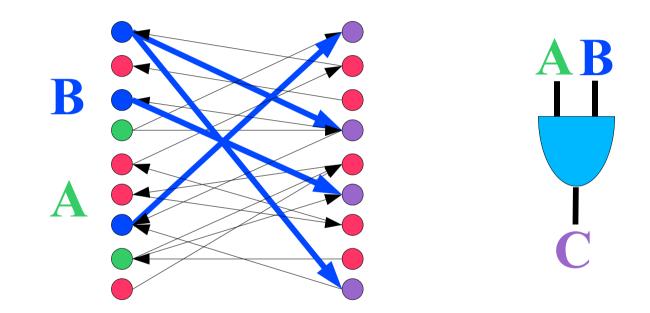
- Random network with sparse connections
- Symbols are sparse sets of nodes
- Implement circuits by discovering connections



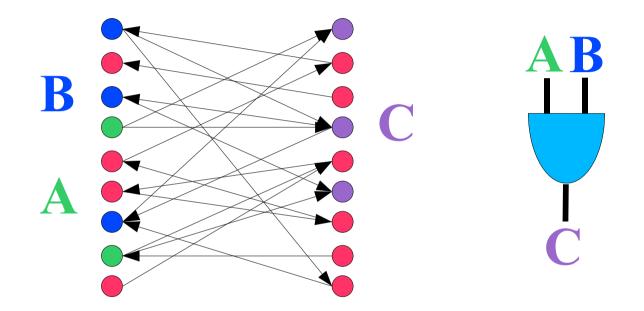
- Activate first A, then B
- Nodes activated by both are a new symbol C
- Lower synapse strength so  $C = A^*B$



- Activate first A, then B
- Nodes activated by both are a new symbol C
- Lower synapse strength so  $C = A^*B$

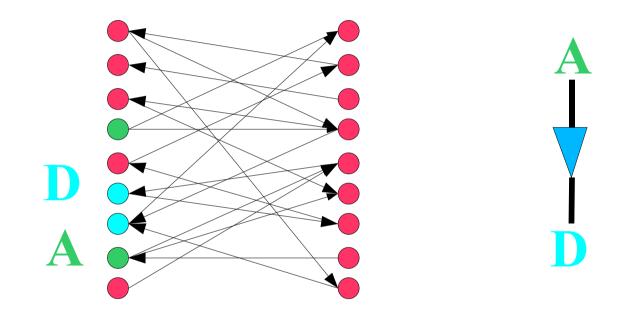


- Activate first A, then B
- Nodes activated by both are a new symbol C
- Lower synapse strength so  $C = A^*B$



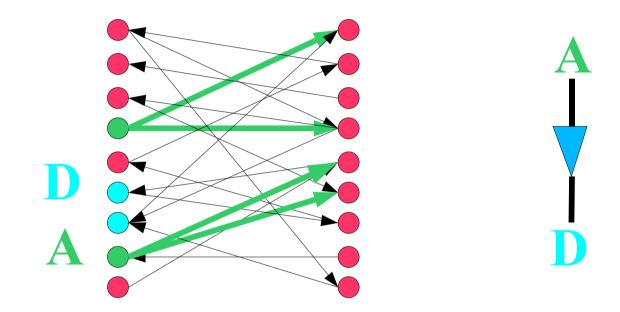
- Activate first A, then B
- Nodes activated by both are a new symbol C
- Lower synapse strength so  $C = A^*B$

# LINK(A,D)



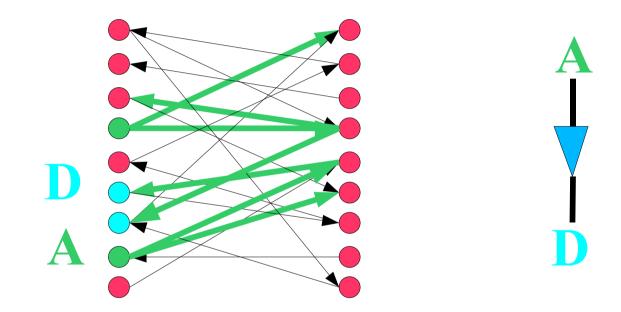
- Activate A
- Raise synapse strength on D's active inputs
- $A \rightarrow [intermediates] \rightarrow D$

# LINK(A,D)



- Activate A
- Raise synapse strength on D's active inputs
- $A \rightarrow [intermediates] \rightarrow D$

# LINK(A,D)



- Activate A
- Raise synapse strength on D's active inputs
- $A \rightarrow [intermediates] \rightarrow D$

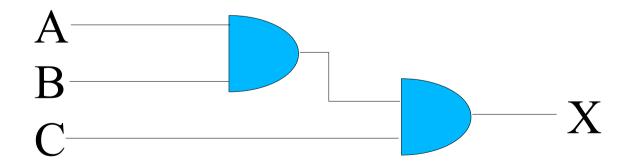
### But does it work?

- Between theory and practice lies engineering
  - Simulation feasiblity
  - Time
  - Device longevity
  - Device characterization
  - Coding Capacity
  - Universality

# Simulation Feasibility

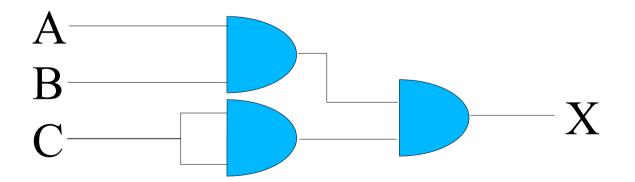
- Even the smallest networks are very large
  - 100,000 nodes, 256 links/node, 1981 nodes/symbol
- Bottleneck operations walk all links
- Some tricks I use:
  - Links implicitly represented by a random seed
    - Two bit-arrays store state
    - Generated and cached when necessary
    - An arbitrary set is permanently cached
  - Fast-fail on link-walk segments with known answers

- JOIN takes 1 cycle, LINK takes 2
- How can multi-layer logic be synchronized?

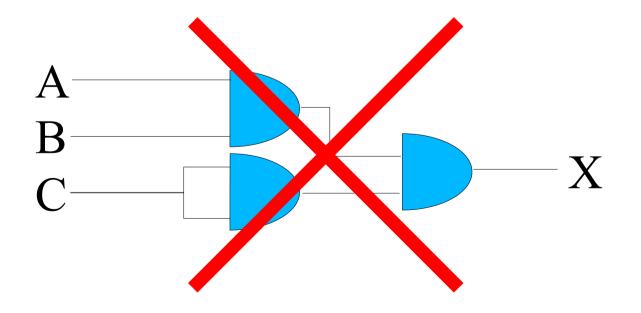


- JOIN takes 1 cycle, LINK takes 2
- How can multi-layer logic be synchronized?

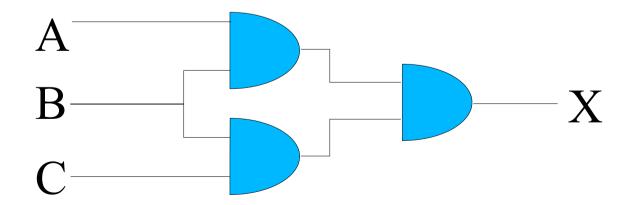
- Buffer C with a JOIN?



- JOIN takes 1 cycle, LINK takes 2
- How can multi-layer logic be synchronized?
  - Buffer C with a JOIN? JOIN(C,C) must fail!

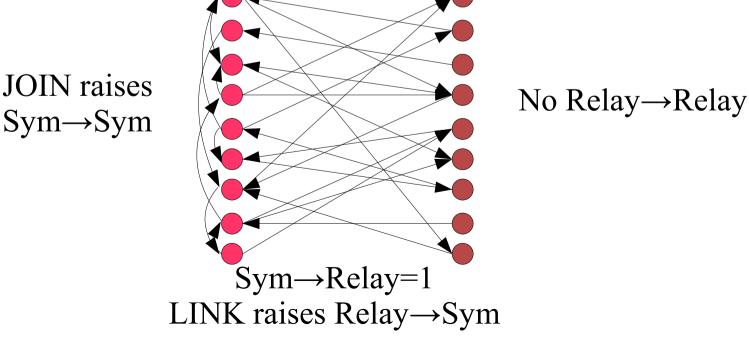


- JOIN takes 1 cycle, LINK takes 2
- How can multi-layer logic be synchronized?
  - Buffer C with a JOIN? JOIN(C,C) must fail!
  - Redundant use of inputs in *different* joins!



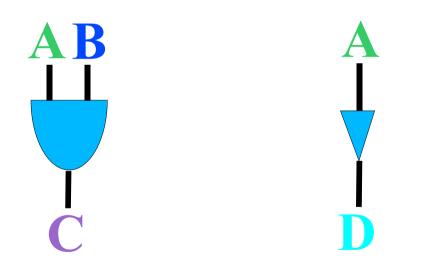
# Device Longevity

- JOIN lowers weights, LINK raises weights
  - After x JOINs, only  $(1-r/n)^x$  of a device is usable
    - 1,000,000,000 nodes = 700 JOINs to destroy a device
  - Solution: differentiate into Symbol and Relay nodes

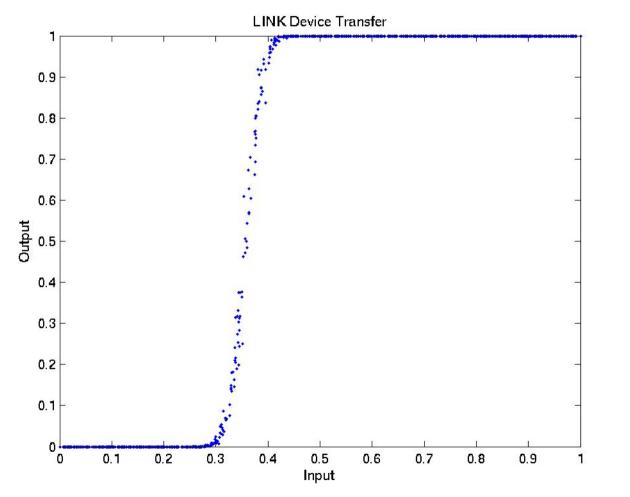


#### Device Characterization

• Are they really digital?

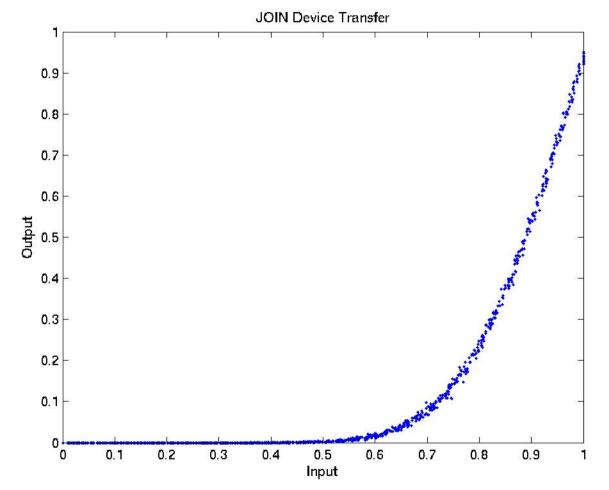


### Device Characterization: LINK



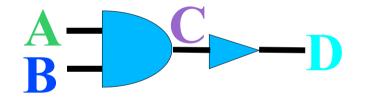
• Yes! (Static discipline  $V_{ol}=0.1 < V_{il}=0.2 < V_{ih}=0.5 < V_{oh}=0.9$ )

### Device Characterization: JOIN



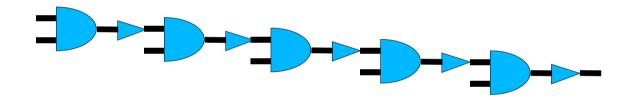
• No! ... and changing the bias turns it into an OR

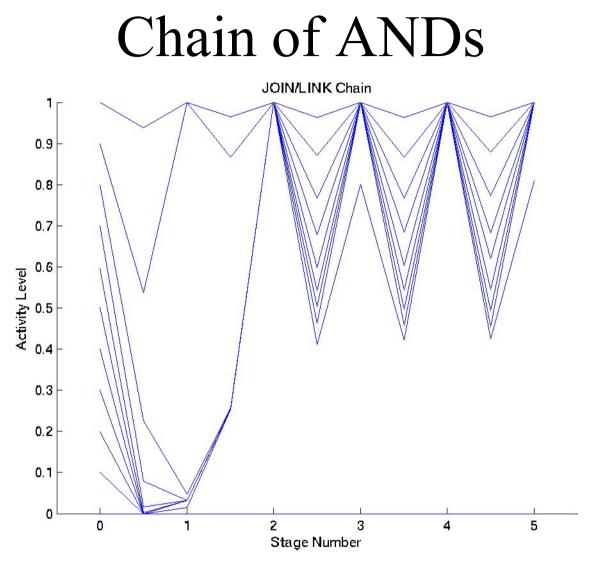
# Compound AND device



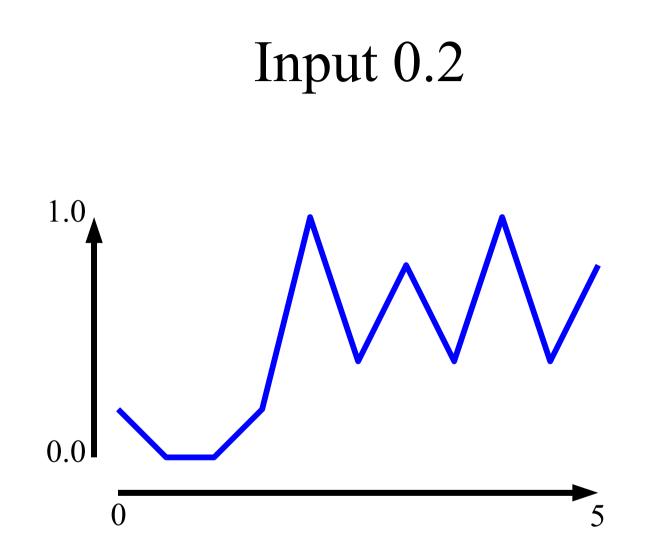
- LINK's nice transfer cleans JOIN's messy output
  - Should switch over around 0.85
- Side benefit: no symbol is derived from C

#### Chain of ANDs



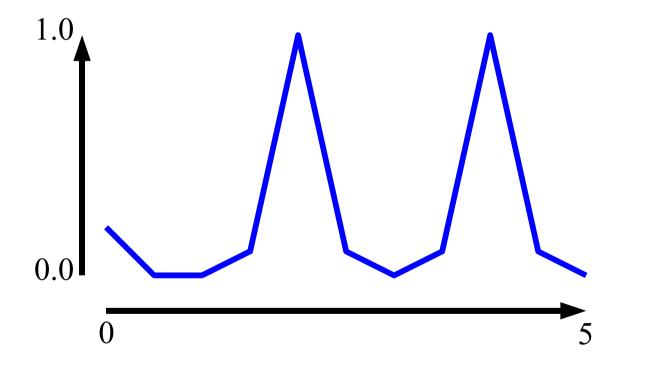


• What's going on here?

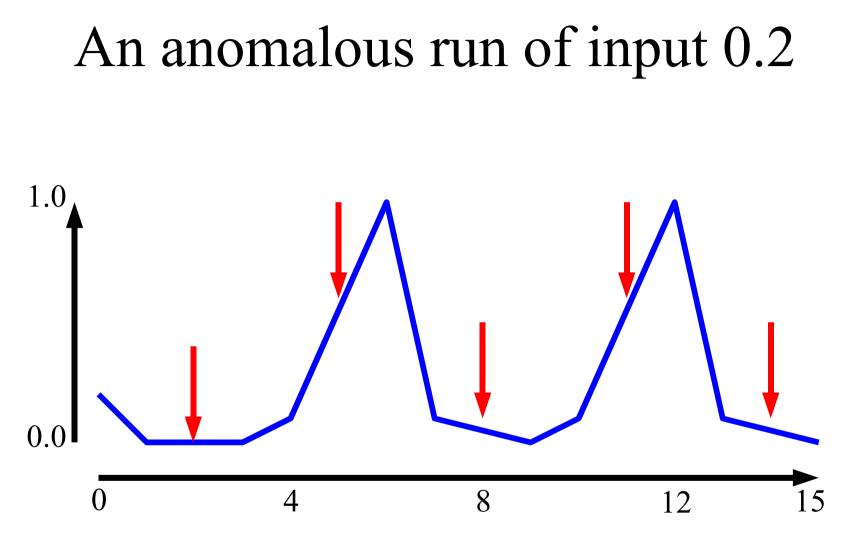


- Why is there a two-stage cycle?
- What are the individual runs doing?

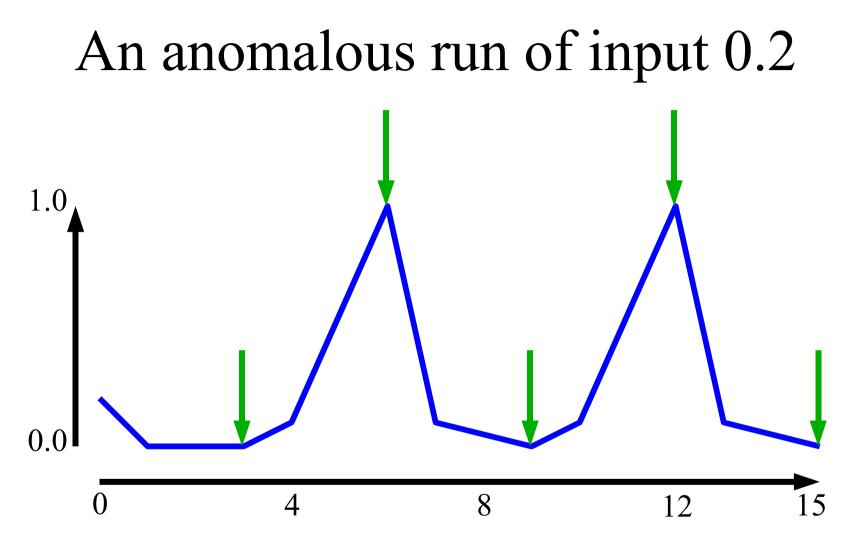
### An anomalous run of input 0.2



• But we're only seeing 2/3 of the time-steps...



• But we're only seeing 2/3 of the time-steps...

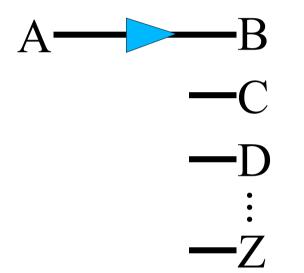


- Spikes at LINK outputs on even timesteps
  - Activity from time 0 is rebounding on the links!

# Sources of Crosstalk

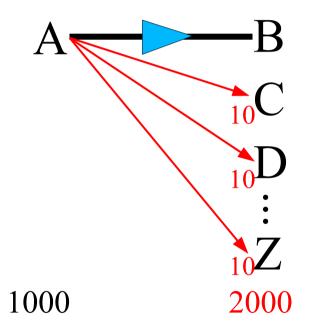
- Uniformity of network
- Strong relay activation, *sqrt(rn)* for a single link
  - 3% of 1,000,000,000 nodes
- Symbol node overlap
- Relay node overlap

#### Crosstalk is Cumulative



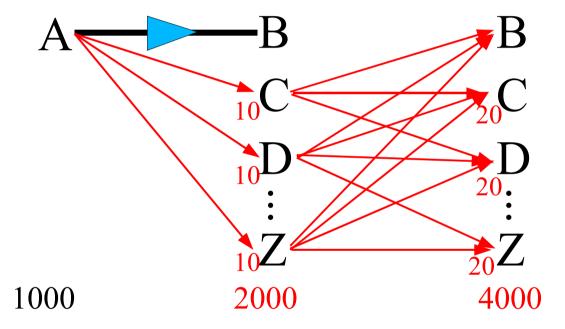
• Consider 200 independent links, 1% crosstalk The net doesn't know about our symbol abstraction!

#### Crosstalk is Cumulative



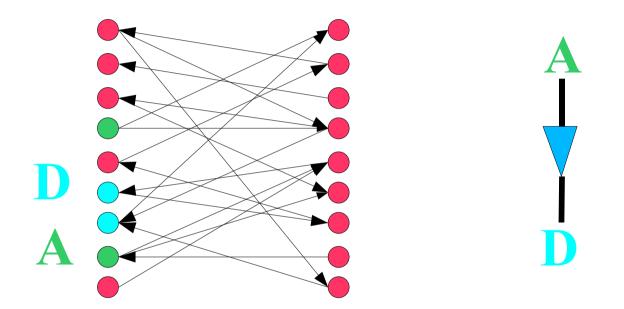
• Consider 200 independent links, 1% crosstalk The net doesn't know about our symbol abstraction!

### Crosstalk is Cumulative + Feedback!



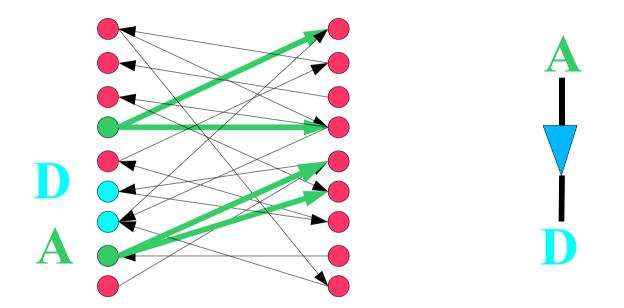
• Consider 200 independent links, 1% crosstalk The net doesn't know about our symbol abstraction!

# Proposed New LINK Algorithm



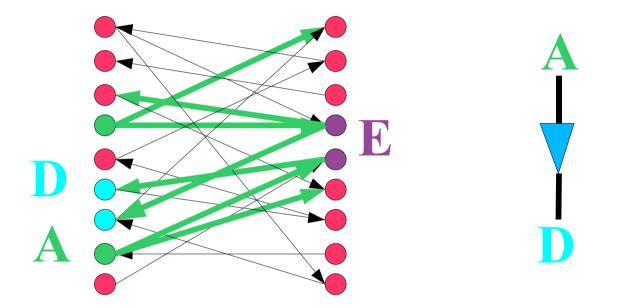
- Propagate A, marking used relay nodes
- Adjust relay strength as well as D input strength
- $A \rightarrow E \rightarrow D$

# Proposed New LINK Algorithm



- Propagate A, marking used relay nodes
- Adjust relay strength as well as D input strength
- $A \rightarrow E \rightarrow D$

# Proposed New LINK Algorithm



- Propagate A, marking used relay nodes
- Adjust relay strength as well as D input strength
- $A \rightarrow E \rightarrow D$

# **Remaining Problems**

- Coding Capacity
  - Better bounds analysis
  - Will the proposed fix work?
- Universality
  - Dual rail logic implementation
  - Efficient inverter, NAND, etc.
- Feasibility
  - Are there more efficient compound gates?