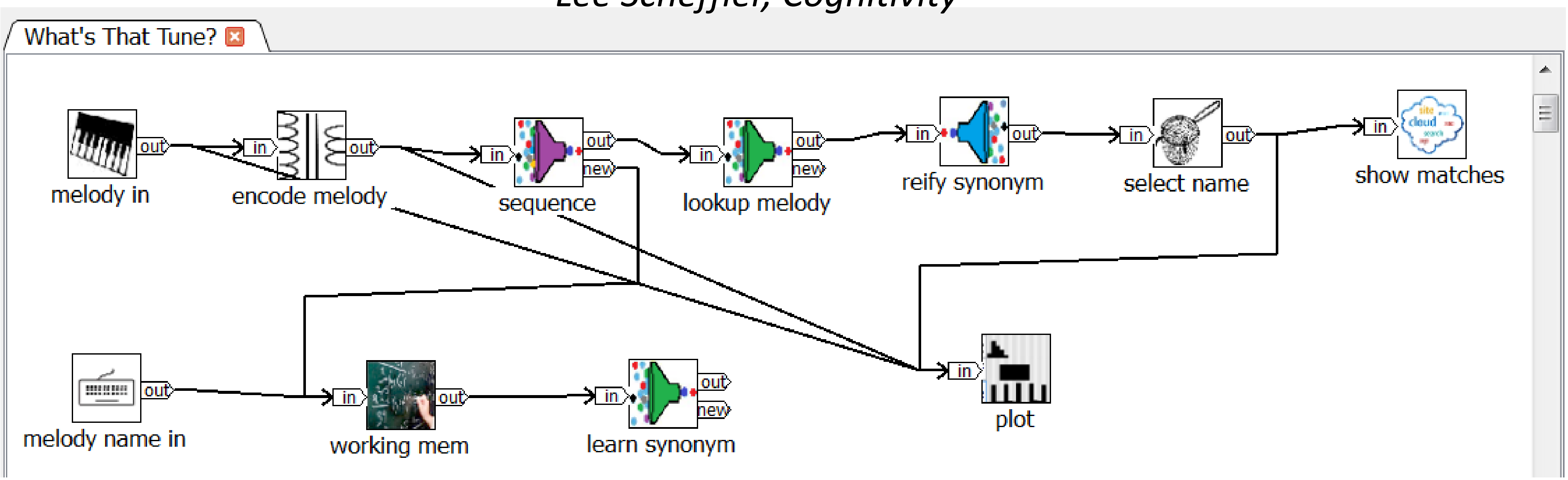
BRAIN LEGOS

NeurOS® and NeuroBlocks®

A Neural Operating System and Cognitive Building Blocks

Lee Scheffler, Cognitivity



Rapidly build/run { portable / scalable / embeddable / extensible } cognitive systems

Cognitivity®
www.cognitivity.technology

NeurOS and NeuroBlocks in a Nutshell

Build **cognitive systems**…
… by linking **reusable modules** …
… into **directed neural graphs/circuits**

Broad applicability

Rapid iterative visual flow development

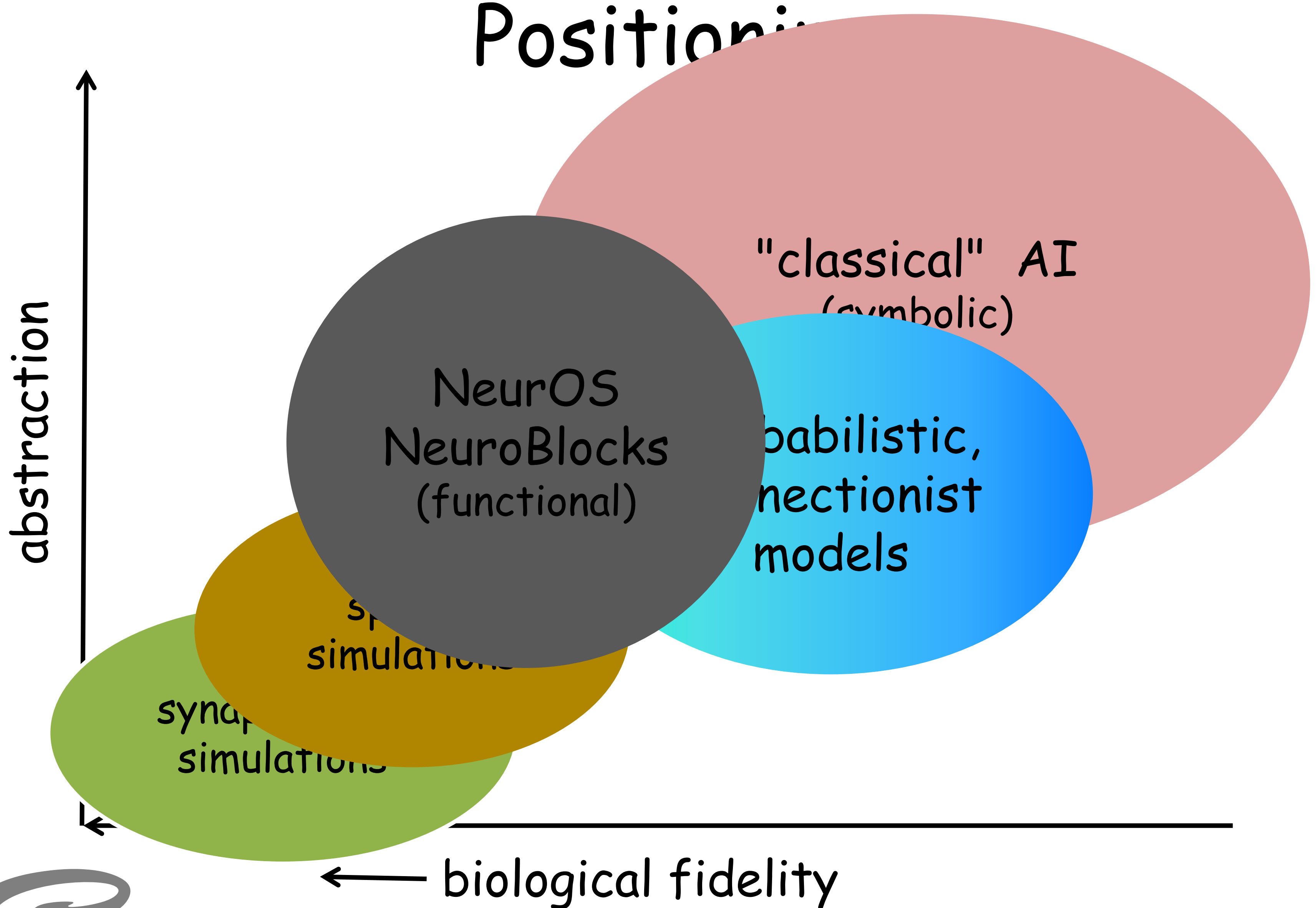
Open, extensible, embeddable, portable, scalable, …

Me

- Software architect/CTO career
  - Multics, security (MIT, Honeywell)
  - networking, office automation, workstations (Prime)
  - data access (Constellation - startup)
  - data integration – **DataStage** (VMark/Ardent/Informix/Ascential/IBM)
  - cognitive systems (Cognitivity)

- Maxims: Be Useful, Make Stuff Work

Positioni



abstraction
"classical" AI
(symbolic)
NeurOS
NeuroBlocks
(functional)
models
biological fidelity

Architecture Layers

- Cognitive system architecture

"application programs"

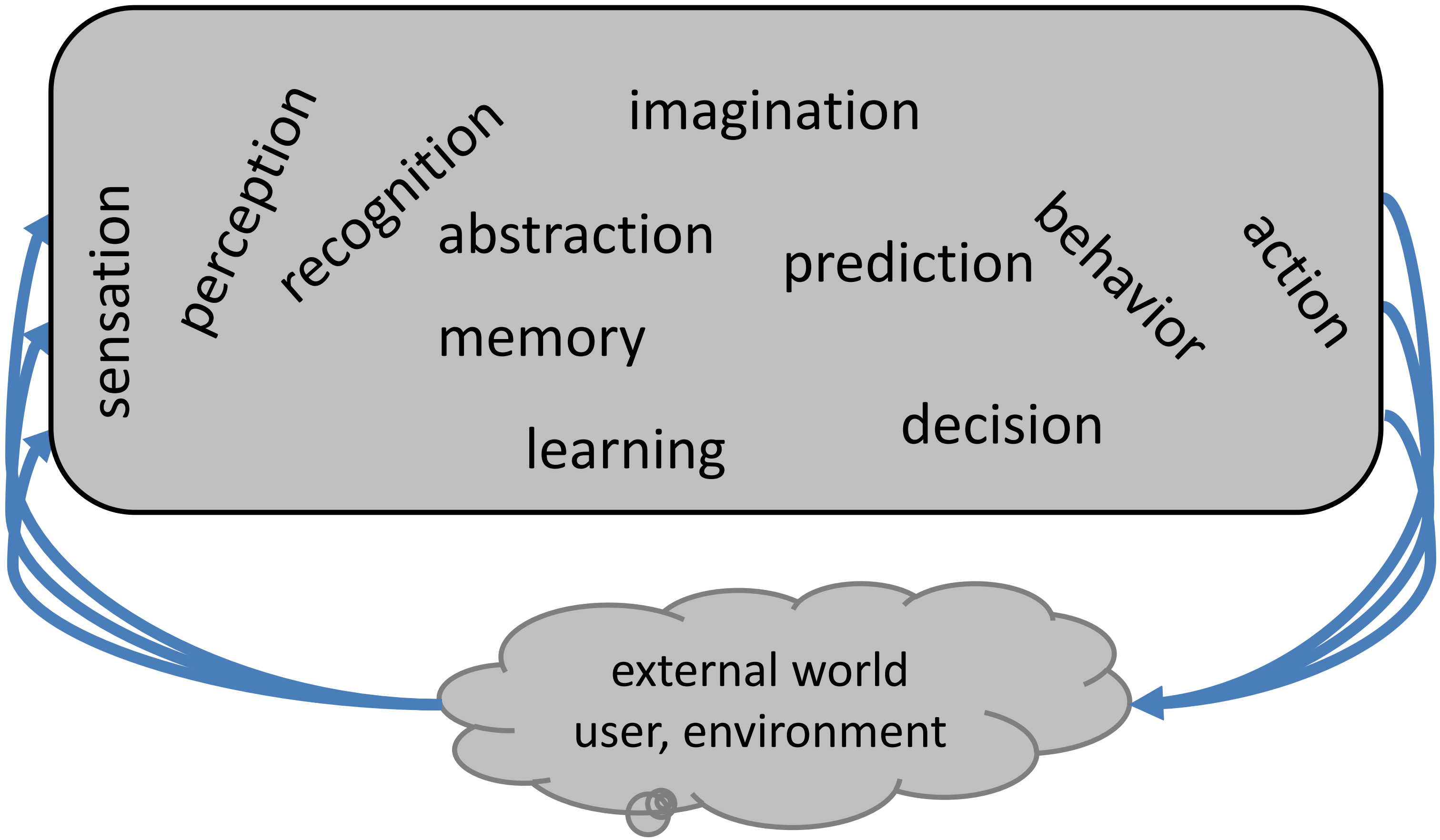
- Neural processing architecture
  - biologically inspired non-von Neuman non-procedural dataflow system

"programming language"

- Implementation architecture
  - conventional computers, networks

"virtual machine"

Cognitive System Challenge

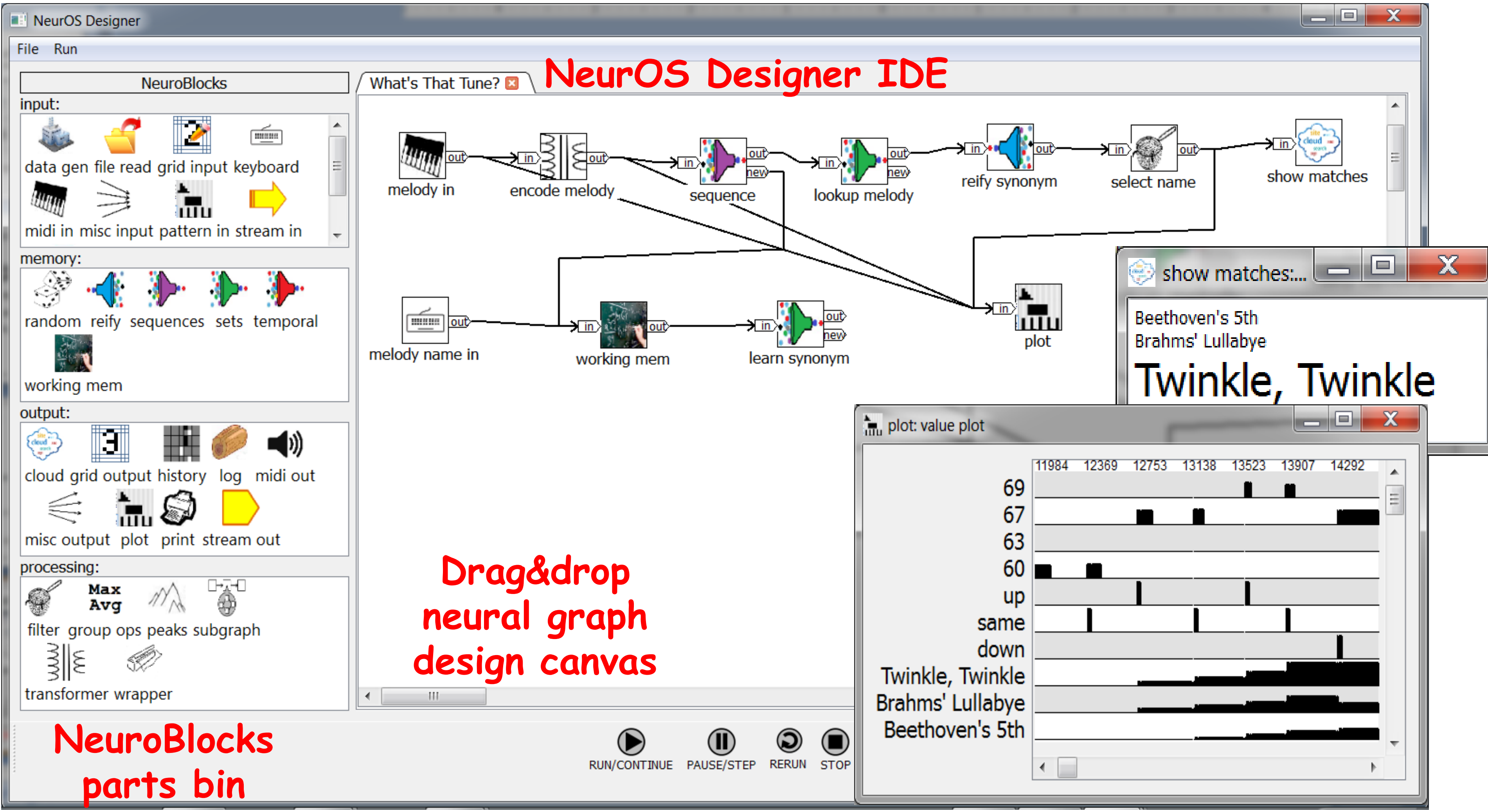


ognitivity®
www.cognitivity.technology

Inspirations/Influences

- Modular component systems
  - **Erector set**, MIDI, Legos, ...
- Circuit design
- Shell pipelines
- Spreadsheets
- Dataflow systems
- **Braitenberg: "Vehicles"**
- Signal processing
- Analog computing
- **Biological brains**
  - Jeff Hawkins: "On Intelligence"

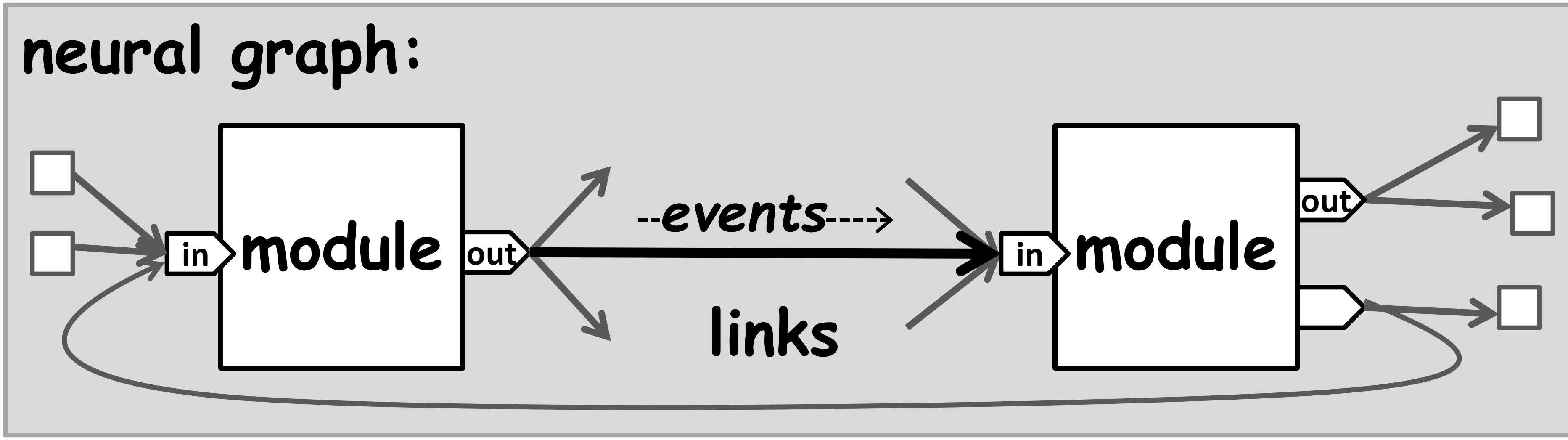
Example: What's That Tune?



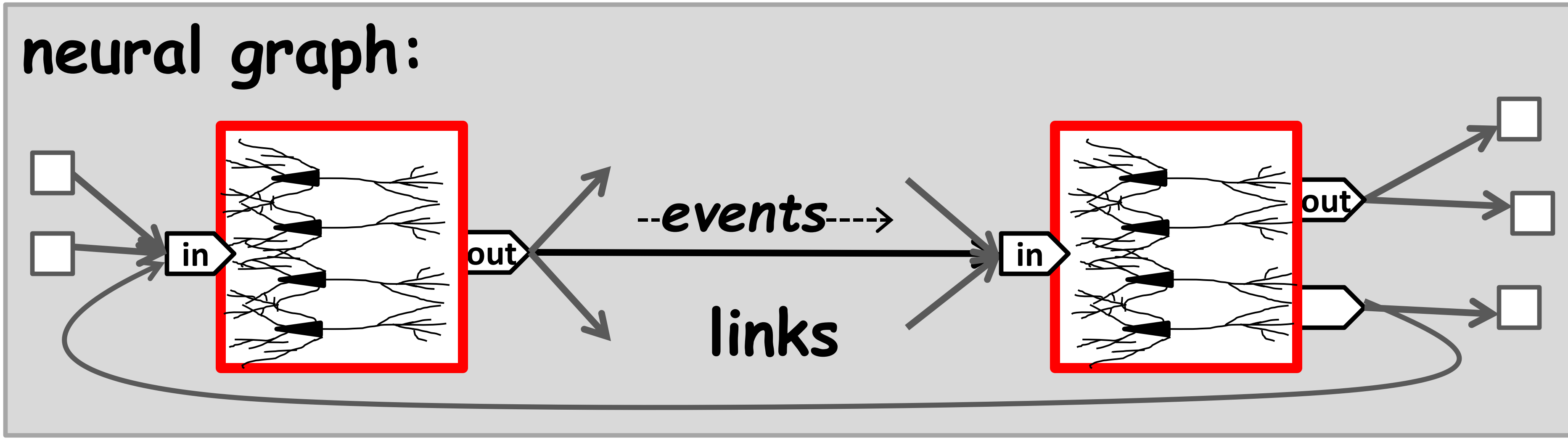
Output module displays

ognitivity®
www.cognitivity.technology

Basics

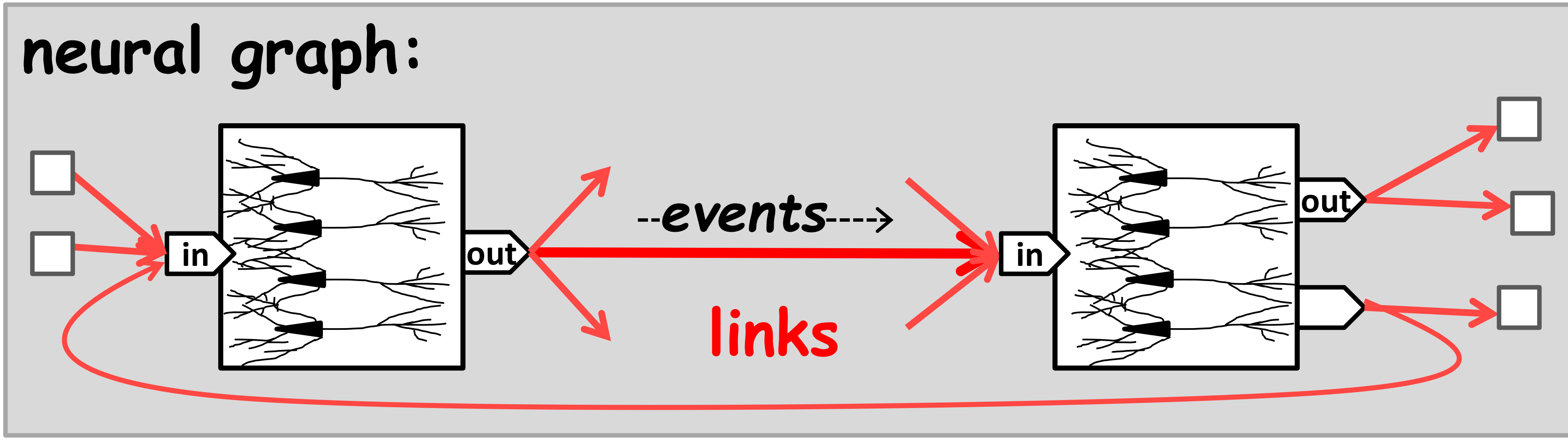


Basics



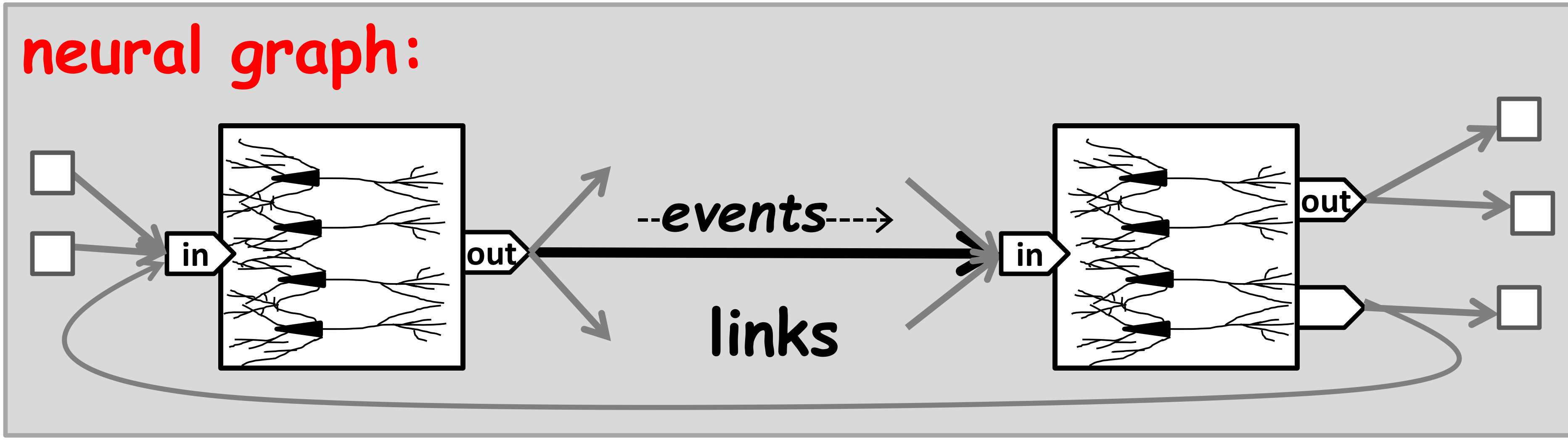
module	Group/layer of neurons or dendritic branches with similar function; state
link	Mulitiplexed event signal path: axons of multiple neurons
neural graph	Directed flow, loops, nestable sub-graphs
event	New spiking rate of a neuron

Basics



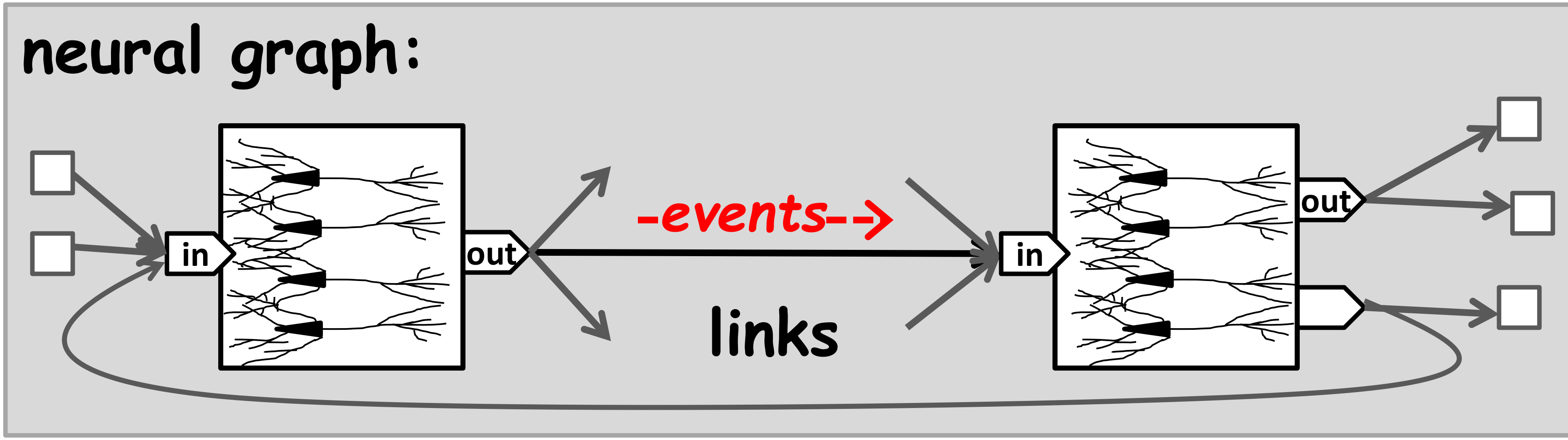
module	Group/layer of neurons or dendritic branches with similar function; state
link	Mulitiplexed event signal path: axons of multiple neurons
neural graph	Directed flow, loops, nestable sub-graphs
event	New spiking rate of a neuron

Basics



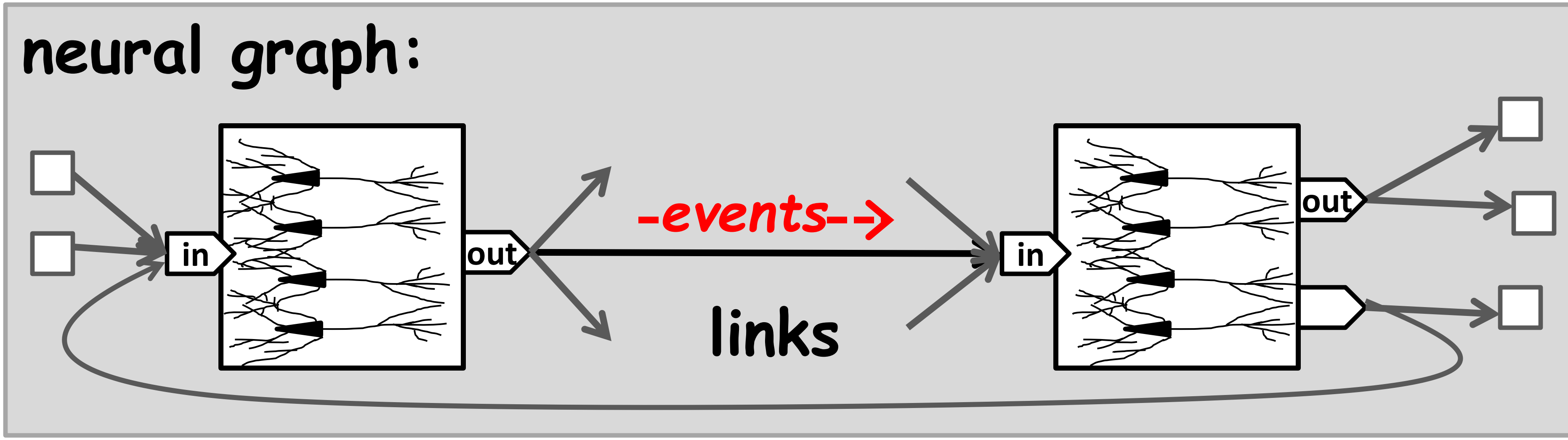
module	**Group/layer of neurons or dendritic branches with similar function; state**
link	**Mulitiplexed event signal path: axons of multiple neurons**
neural graph	**Directed flow, loops, nestable sub-graphs**
event	**New spiking rate of a neuron**

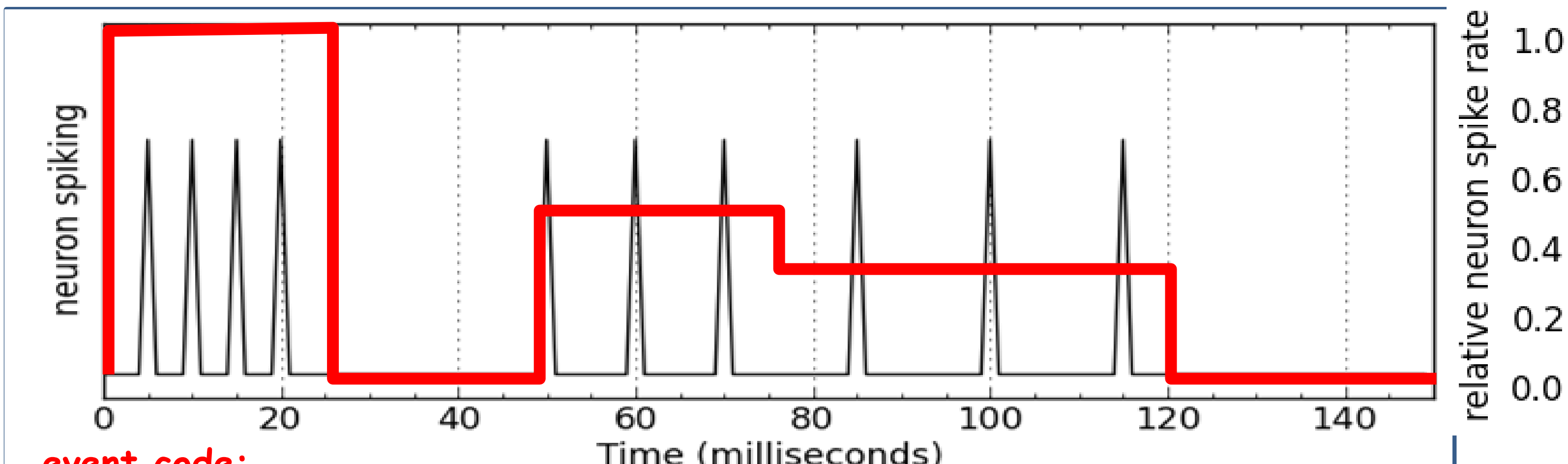
Basics



module	**Group/layer of neurons or dendritic branches with similar function; state**
link	**Mulitiplexed event signal path: axons of multiple neurons**
neural graph	**Directed flow, loops, nestable sub-graphs**
event	**New spiking rate of a neuron**

Basics





event code:
(0, id,1.0) (25, id,0) (50,id, 0.5) (75, id,0.33) (120, id,0)

ognitivity®
www.cognitivity.technology

Module Types

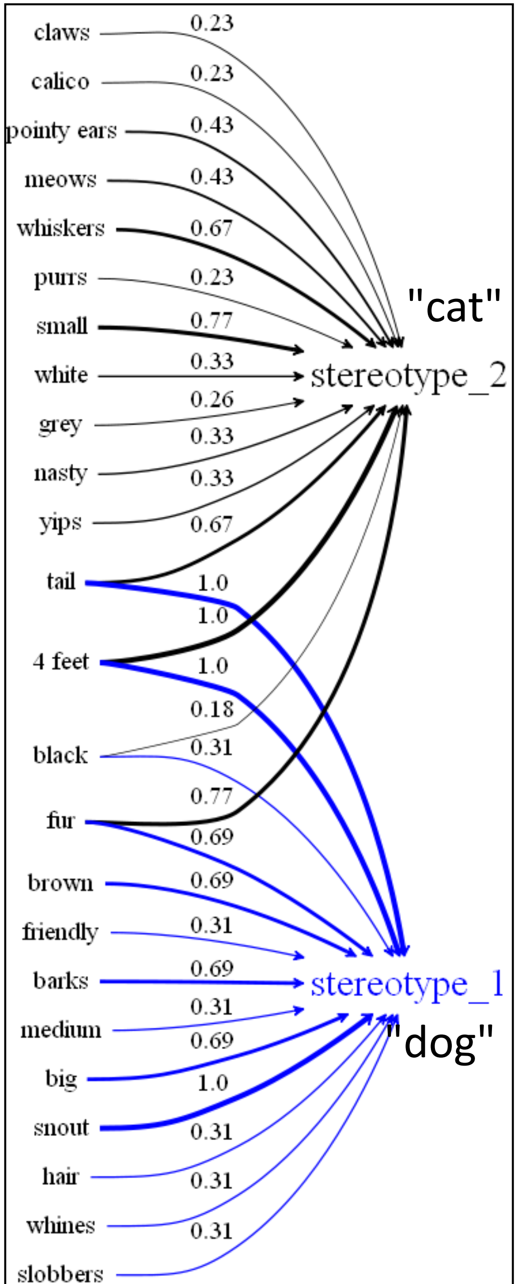
inputs (senses)	keyboard · temporal pattern in · grid in · data gen · stream in · MIDI in
processing	filter · transformer · group ops · wrapper · subgraph
memory	working memory · set patterns · sequence patterns · temporal patterns · pattern reify
outputs (actions, effectors)	print · plot · cloud · grid out · stream out · log · MIDI out

ognitivity®
www.cognitivity.technology

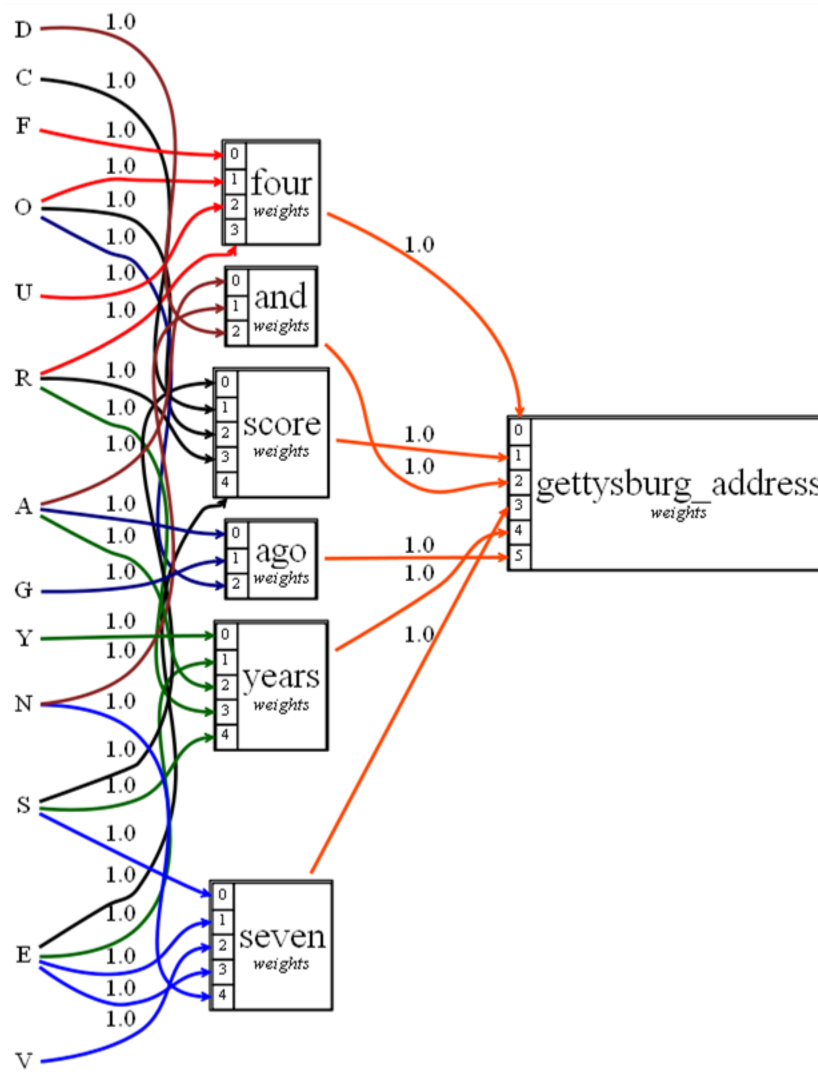
Memory Pattern Module Types

Sets	Concurrent feature collections in any order Semantic range: any/OR, a few, some, many, most, all/AND
Sequences	Time-independent sequences of features Parameters for non-exact sequence matching
Temporal Patterns	Time-relative sequences of multiple features Parameters for non-exact matching, speed range
Reify	Inverse: generate pattern features e.g., prediction, imagination, expectations, feedback

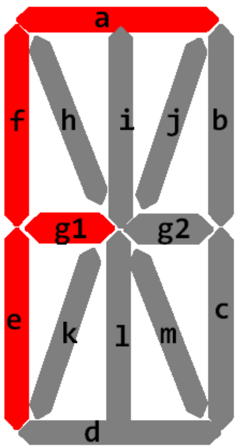
Pattern Examples



	stereotype_1	stereotype_2
brown	**0.69**	
friendly	0.31	
barks	**0.69**	
medium	0.31	
fur	**0.69**	**0.77**
big	**0.69**	
hair	0.31	
4 feet	**1.0**	**1.0**
slobbers	0.31	
black	0.31	0.18
whines	0.31	
snout	**1.0**	
tail	**1.0**	**0.67**
meows		0.43
yips		0.33
claws		0.23
pointy ears		0.43
calico		0.23
purrs		0.23
whiskers		**0.67**
small		**0.77**
nasty		0.33
white		0.33
grey		0.26



Layers of Patterns

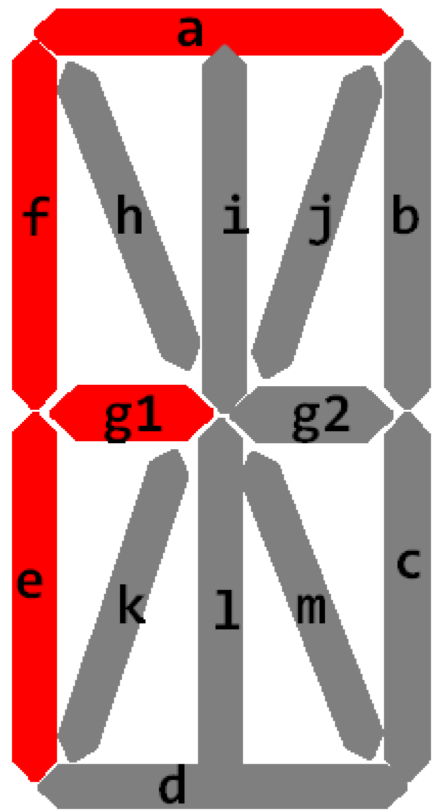


F = {a,e,f,g1}

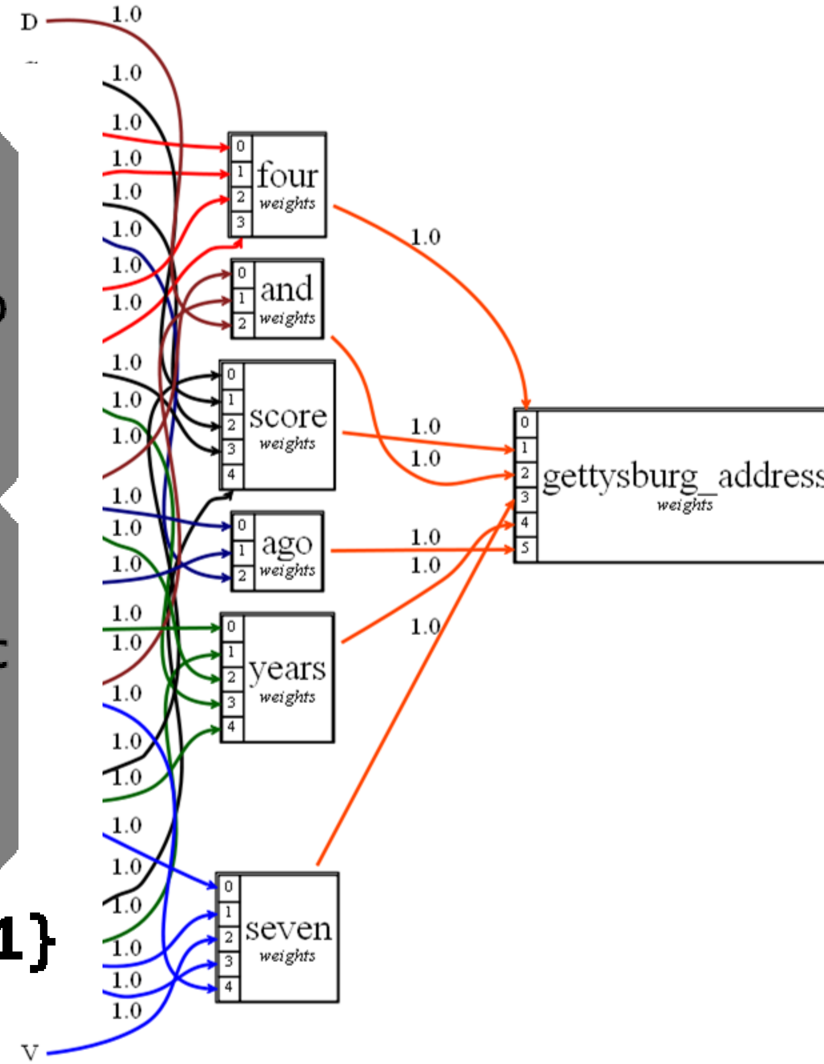
letter

LED segment

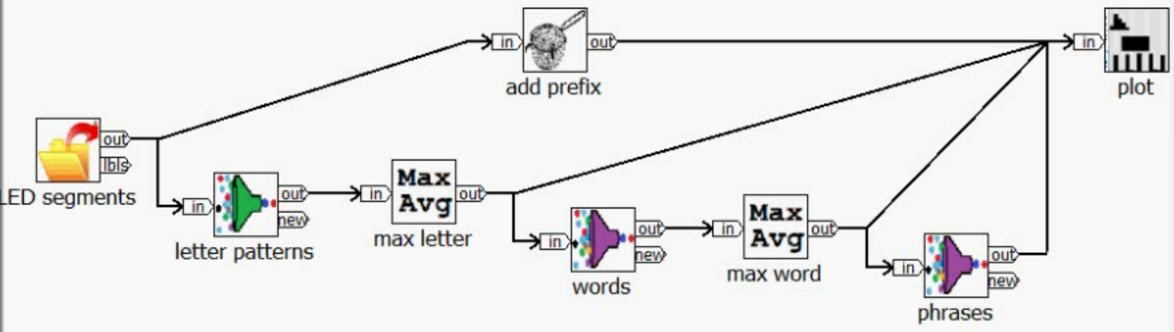
	A	C	D	E	F	G	N		
c	1.0		1.0			1.0	1.0	1	
b	1.0		1.0				1.0	1	
a	1.0	1.0	1.0	1.0	1.0	1.0		1	
f	1.0	1.0		1.0	1.0	1.0	1.0	1	
e	1.0	1.0		1.0	1.0	1.0	1.0	1	
g2	1.0			1.0		1.0			
g1	1.0			1.0	1.0				
d		1.0	1.0	1.0		1.0		1	
i			1.0						
l			1.0						
h							1.0		
m							1.0		
k									1.0
j									1.0



F = {a,e,f,g1}



www.cognitivity.technology



Layers of Patterns

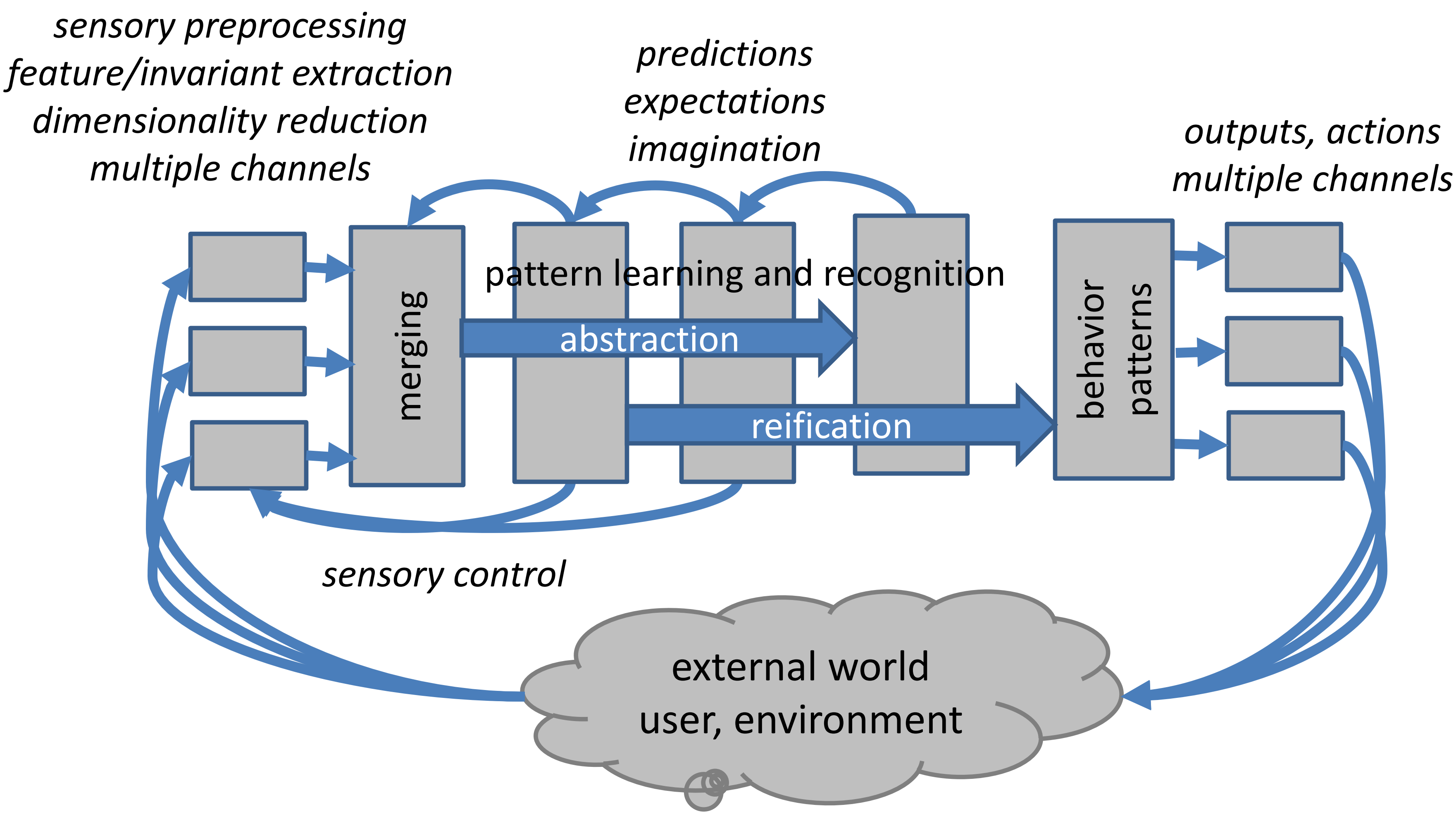
Pattern Learning

- Current inputs match an existing pattern well-enough?
  - novelty threshold parameter
  - **adjust feature weights** of best matching pattern(s); learning rate parameter, annealing
- Otherwise:
  - **create a new pattern** from the current inputs

Enables complex pattern heterarchies

- specific exemplars, stereotypes, ...
- layers of recombination of abstractions

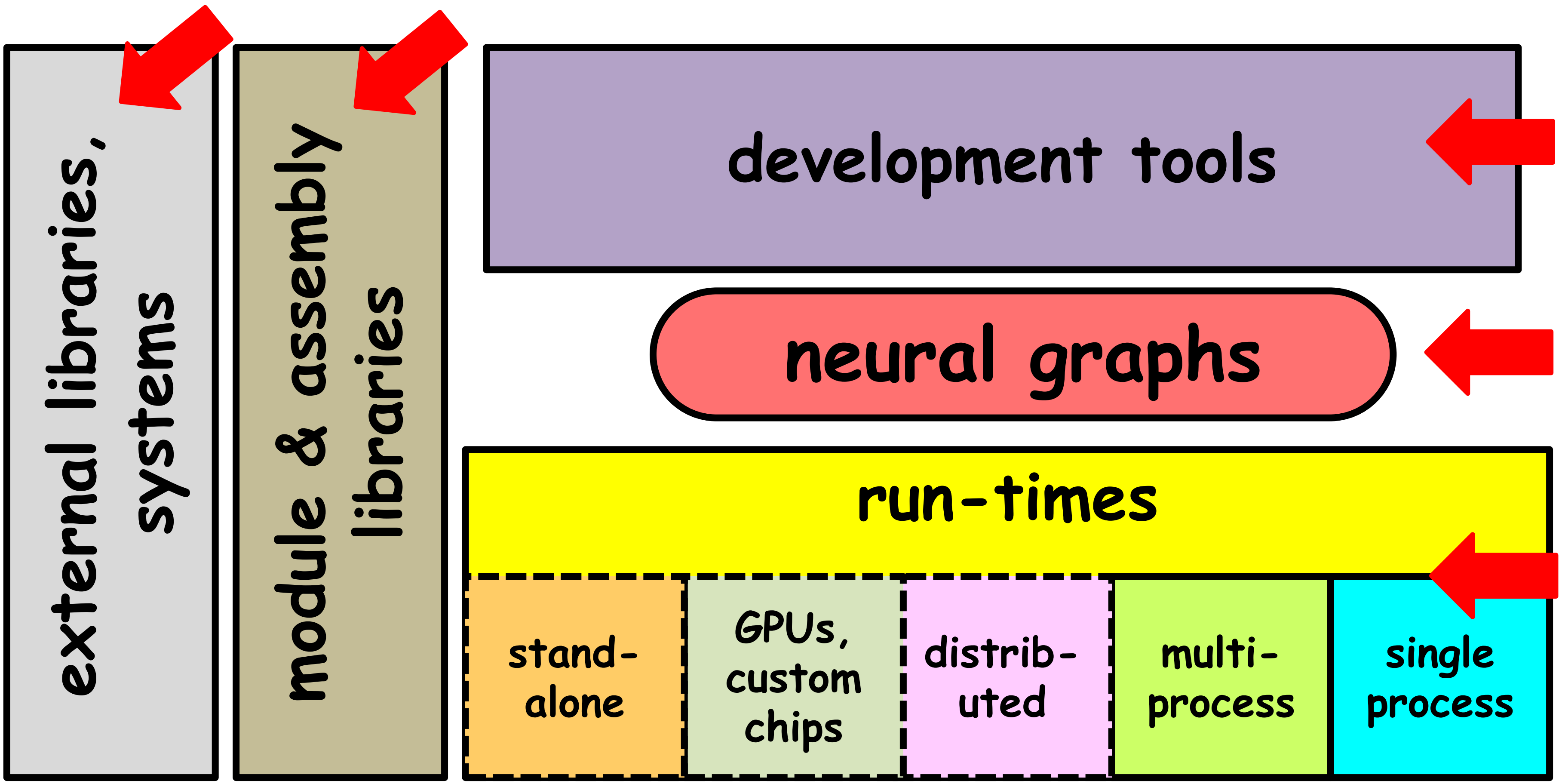
Typical Cognitive System Structure



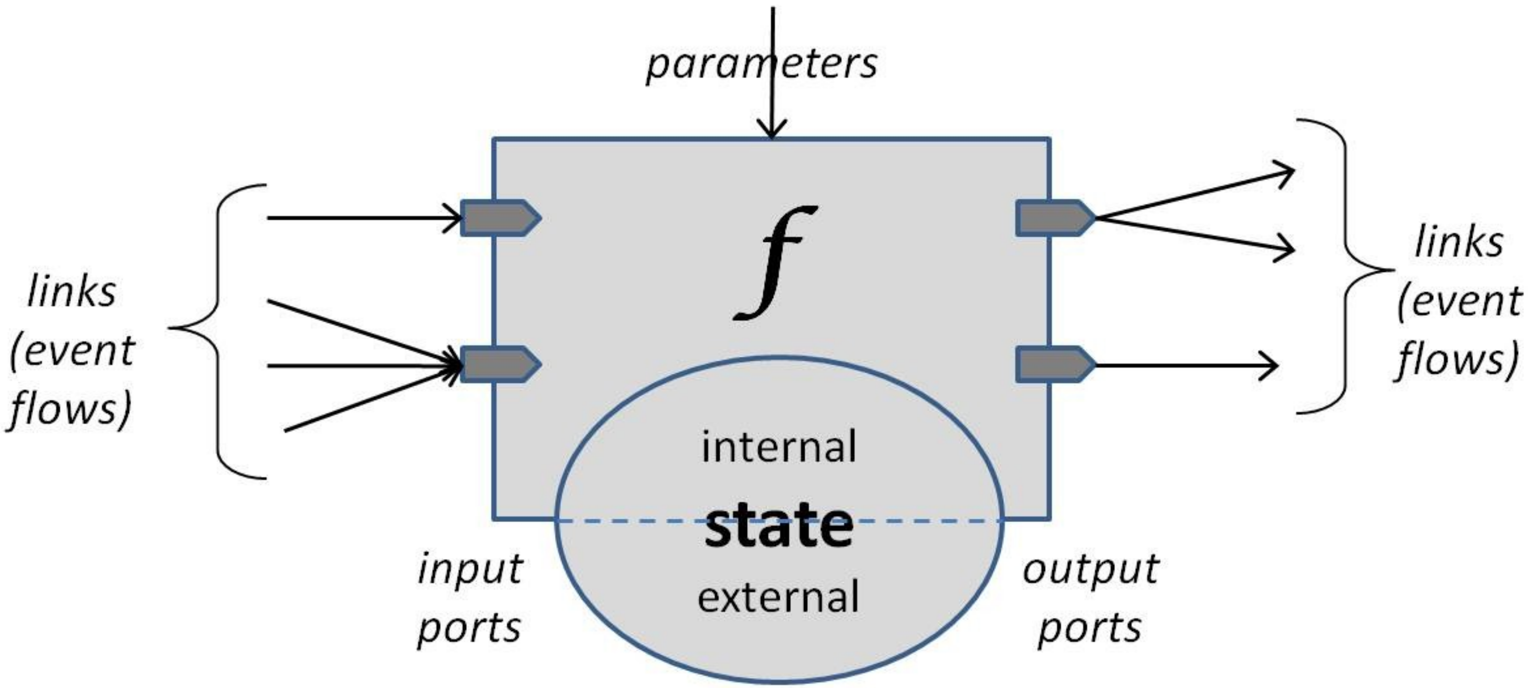
NeurOS Strengths

- One/few-shot learning
  - no huge training sets
- Continuous on-line learning
- Unsupervised, supervised ("teaching"), reinforcement learning
- Few configuration "hyper-parameters"

NeurOS Implementation Architecture



Module



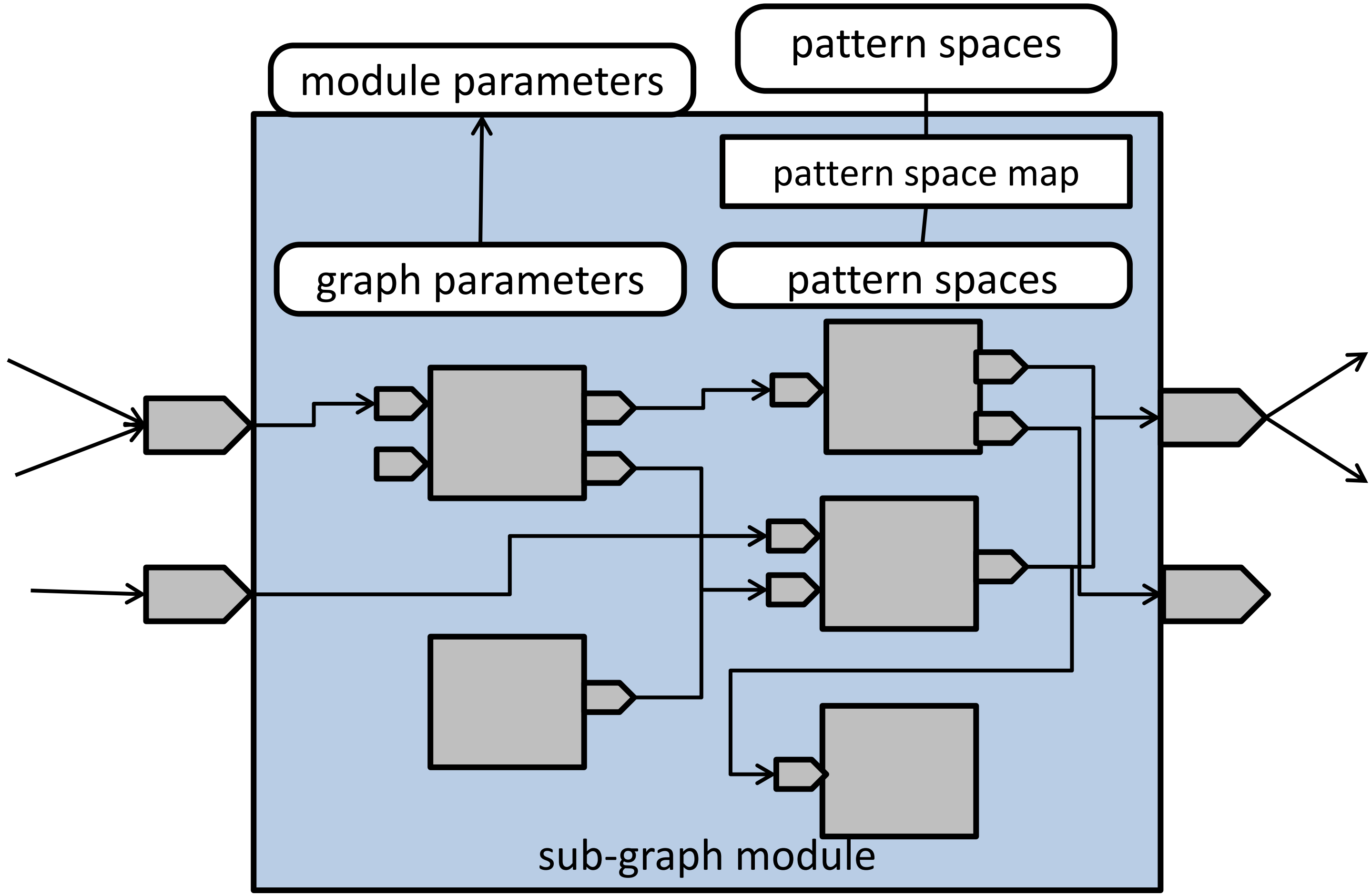
$$\{outputs_t, state_t\} = f\,(params, inputs_t, state_{t-1})$$

Module Life-Cycle

- **create** instance
- **start**(vtime)
  - initialize external resources, internal state
  - schedule self-events if needed (e.g., input polling modules)
- **run**(vtime)
  - access input signals
    - new events (changed signal values)
    - current signal values
  - recompute internal dynamic parameters (e.g., learning rate)
  - compute internal state changes
  - update shared global data
  - compute and send output signal events
    - non-zero virtual time delay
  - schedule self-events (input modules, decays, randomness)
- **finish**(vtime): finalize/close external resources

Recursion

- Sub-circuits recaptured as reusable modules



Virtual Time

- Always moves forward
  - non-zero module processing vtime
    - variable processing vtimes model biology
  - enables (feedback) loops – vital !
- Signals synchronized only where needed
  - enables parallelism
- Synchronized with real time at edges
  - typical compromises: miss inputs, stumble

Master Work Queue/Loop

(single process, multi-thread implementation)

- Event send:
  - post event message (vtime, signal id, new value) on each output link
  - add run(module, vtime) to vtime-sorted work queue

- Next iteration:
  - get run(module, vtime)s for lowest vtime and run
    - can be concurrent

Performance, Scalability

- Signal value changes only
  - don't run modules where there are no signal changes
  - good-enough approximations
- Patterns indexed by events
  - don't evaluate patterns that don't care about current signal changes
- Multi-threading
- Multi-process, distributed execution
- Partitioning, load balancing
- Custom hardware

Integrations, Extensions, Embeddings

- Configurable modules
  - formatted file I/O, database access, pipes/sockets, web services, etc.
- External library functions
- Wrap external library/program
  - stdin/stdout/stderr, APIs
  - no write-shared state among module instances; no synchronization guarantees
- Templated and custom modules

Active Projects

- Industrial machine vibration monitoring, alerting, predictive maintenance, diagnosis
- Adaptive doctor's user interfaces to EHR systems
- Virtual infant robot learning

- Redistributable dev kit

NeurOS and NeuroBlocks in a Nutshell

Build **cognitive systems**…
… by linking **reusable modules** …
… into **directed neural graphs/circuits**

Broad applicability

Rapid iterative visual flow development

Open, extensible, embeddable, portable, scalable, …

Contact

http://www.cognitivity.technology

(Sign up to be notified when NeurOS development kit is available)

lee@cognitivity.technology