Programmable Smart Machines: A Hybrid Neuromorphic approach to General Purpose Computation

Speaker: Jonathan Appavoo

Amos Waterland, Katherine Zhao, Schuyler Eldridge, Ajay Joshi, Steve Homer, Margo Seltzer

PSM: Hybrid computing systems that behave as programmed but transparently learn and automatically improve their operation

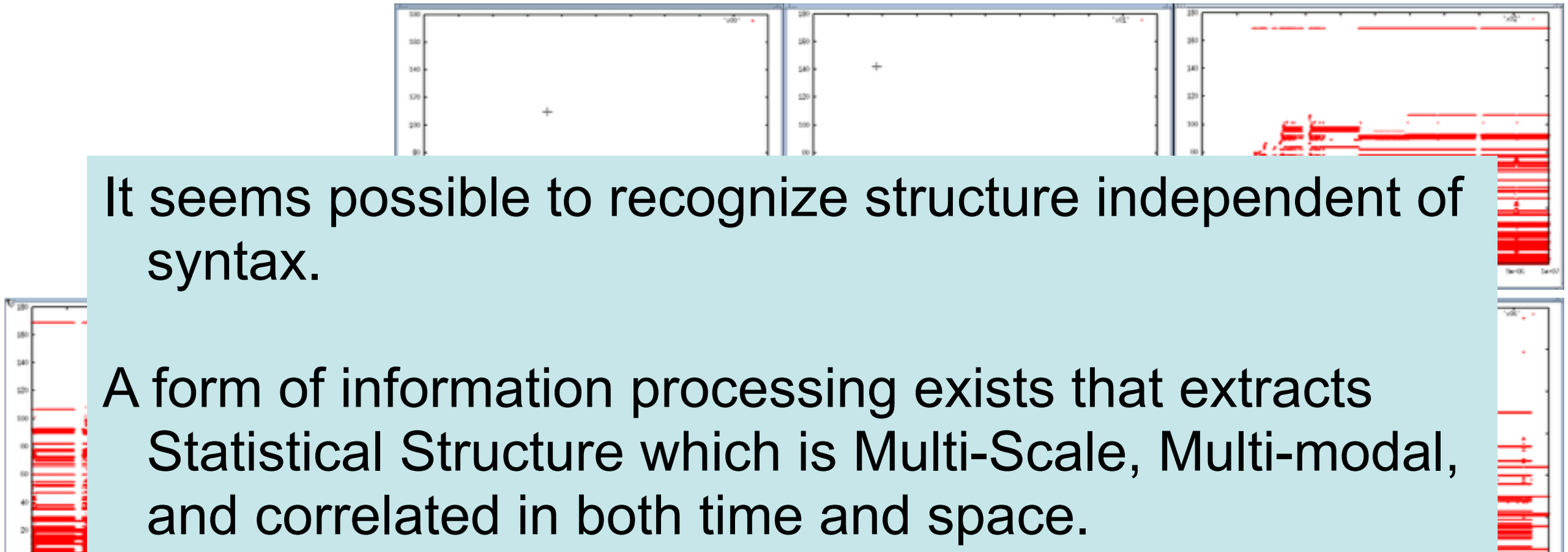
PSML
Programmable Smart Machines Lab

A Unification Principle of Information Processing Systems "Programmable Smart Machines"

DSRC/DARPA 2008 Summer Conference

Jonathan Appavoo & Amos Waterland
IBM Research – T.J. Watson Research Center NY

PSML
Programmable
Smart Machines
Lab



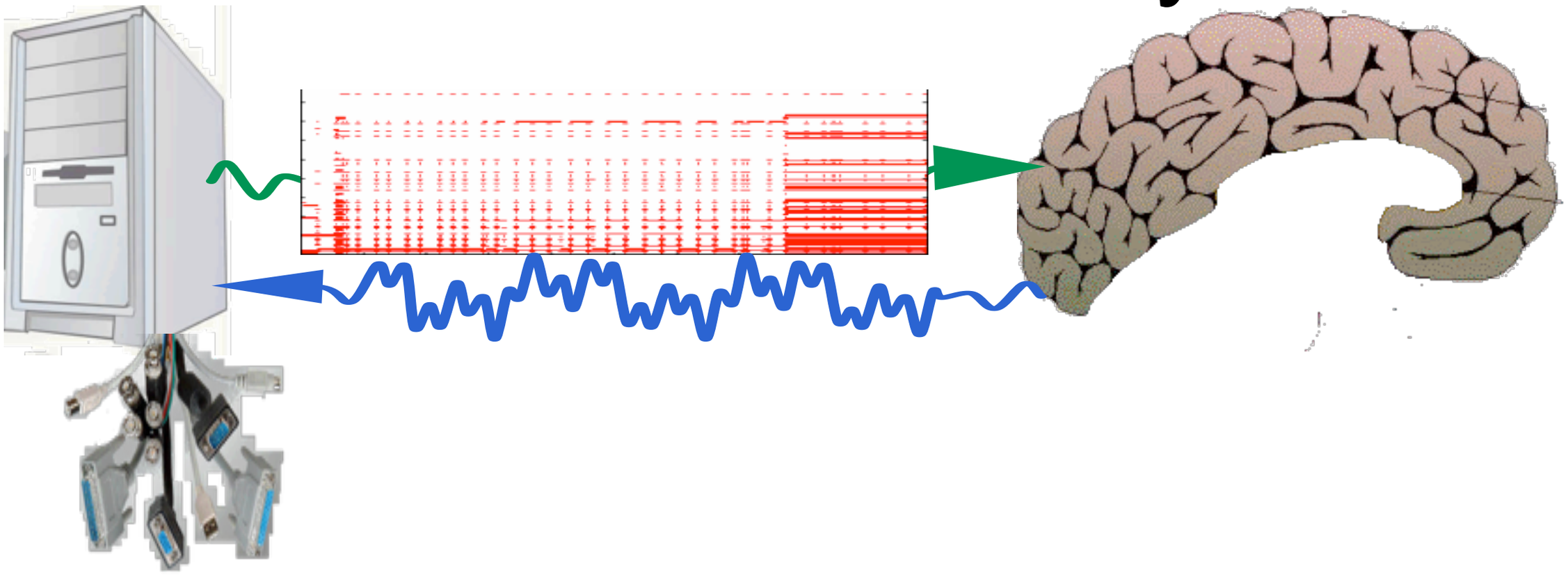
It seems possible to recognize structure independent of syntax.

A form of information processing exists that extracts Statistical Structure which is Multi-Scale, Multi-modal, and correlated in both time and space.

Our Goal:

Explore exploitation of structure in execution (in an ISA independent way), which is correlated over time in a consistent, way in order accelerate the programmed function and potentially enable new un-programmed function.

The Crux → Novelty



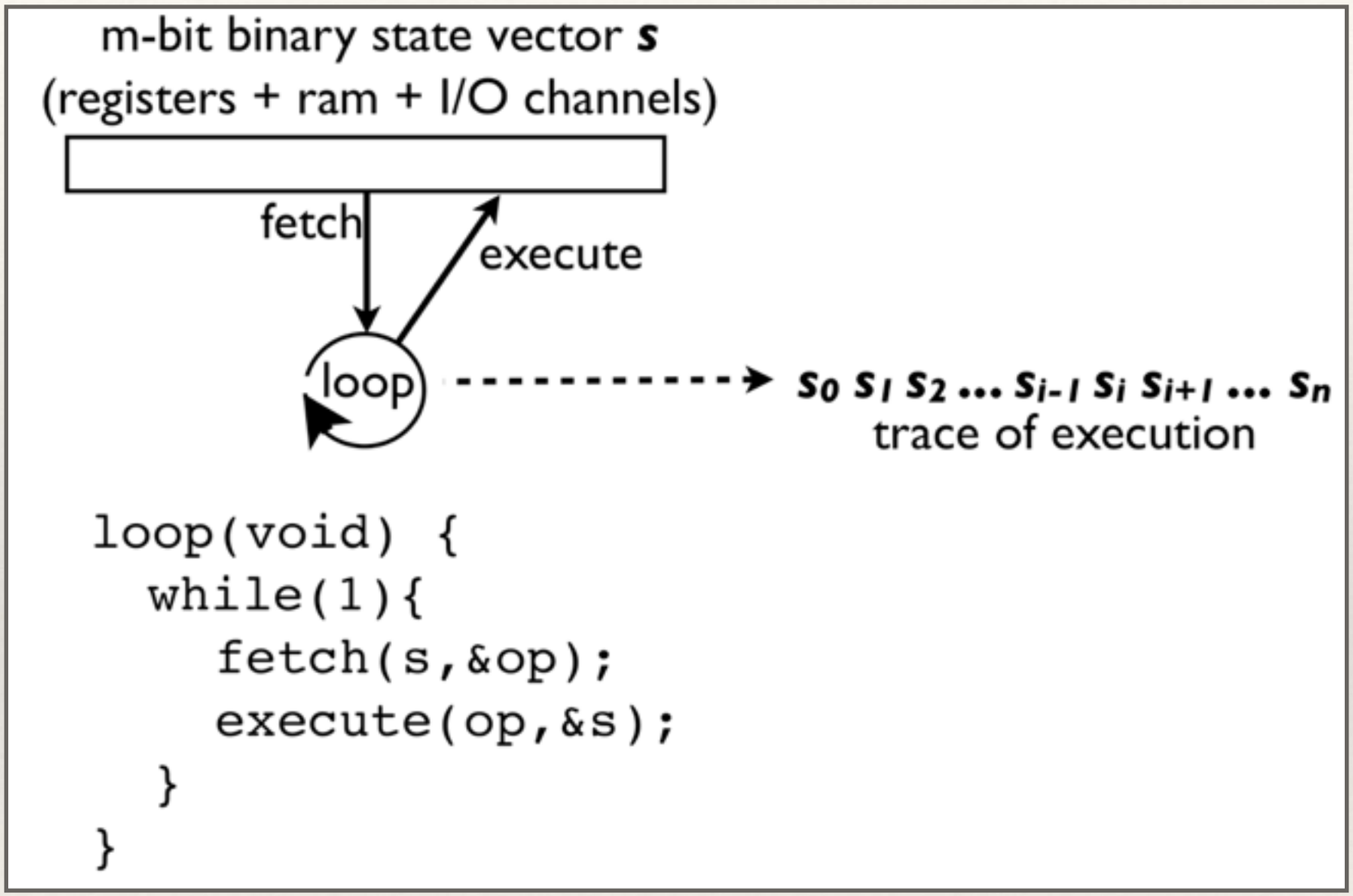
1. Convert ENTIRE operation of a Modern Computer into an Execution Signal.
2. Extract, Represent and Express Structure in Signal.
3. Exploit Structure to potentially:
    1. Accelerate Future Computation,
    2. Expose Unknown/Un-Expected Correlations and
    3. Enable Extrapolation to generate new Behaviour.

ASC: Automatically Scaling Computation (A. Waterland et. al, ASPLOS 2014)

Exploit Learning of Structure to enable a form of Auto-Parallelization

PSML
Programmable Smart Machines Lab

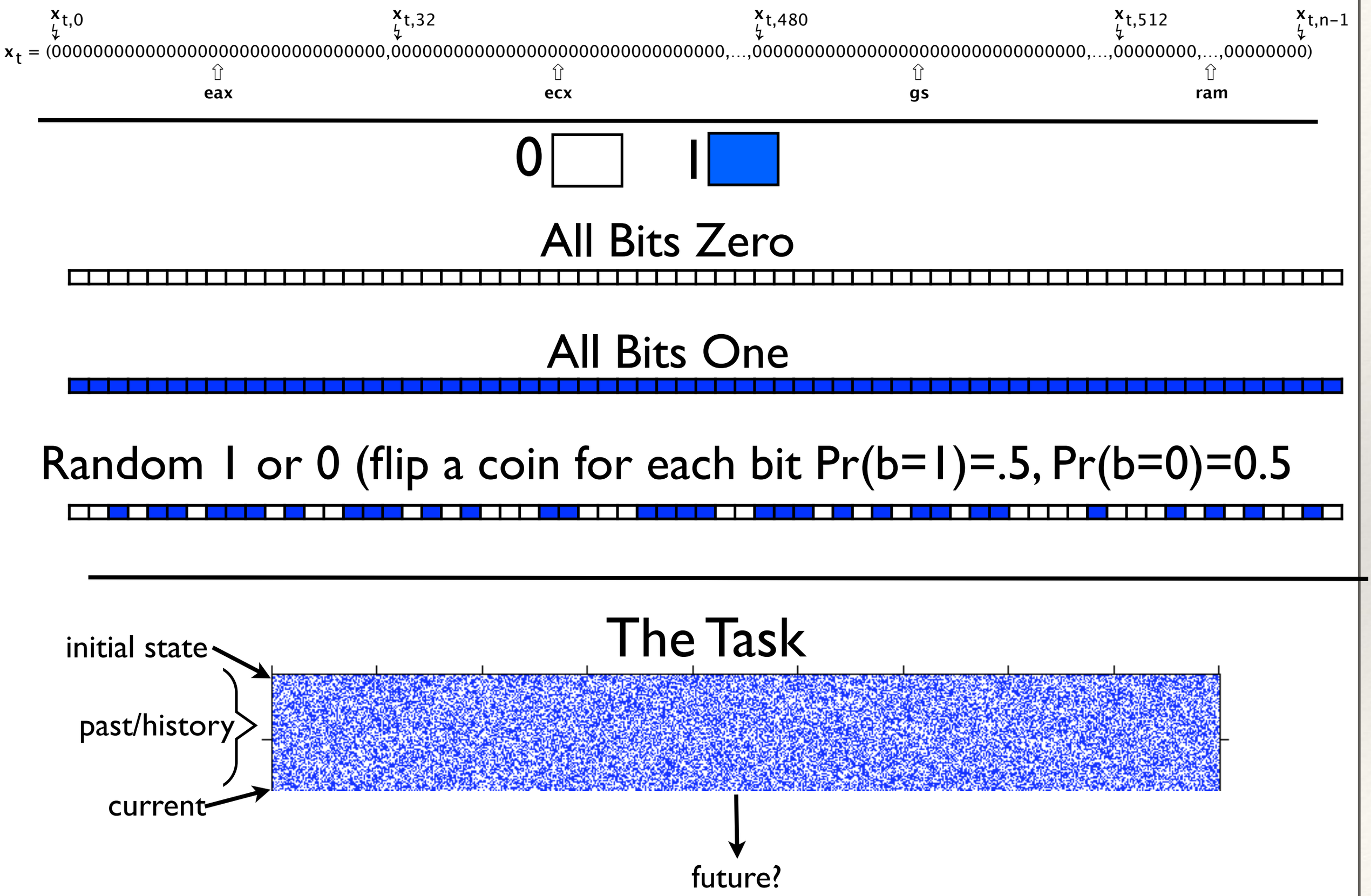
Execution Signal

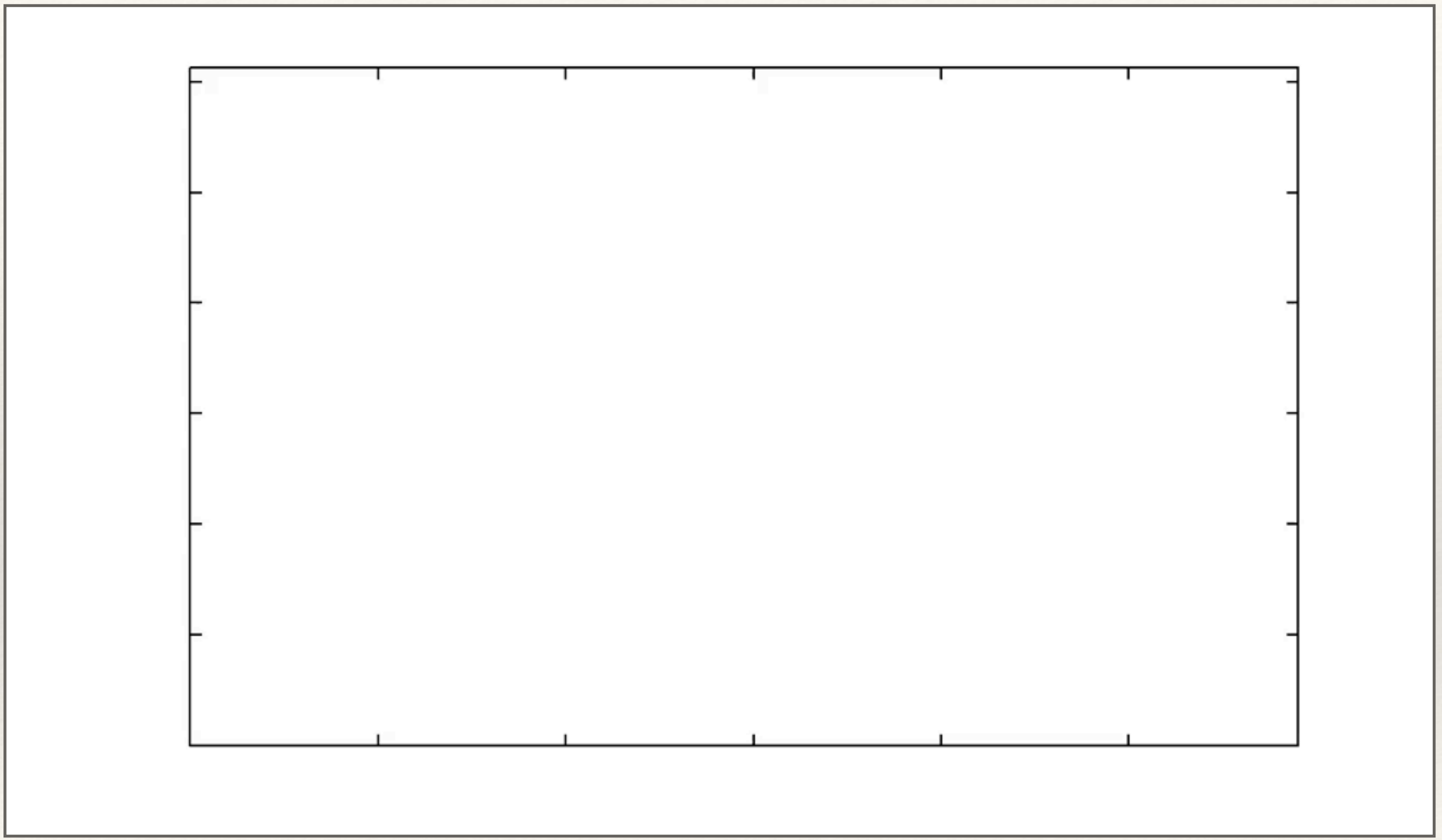


See our *HotPAR'12* Paper, "Parallelization by Simulated Tunneling", Waterland et al. For a more technical view of execution as state space traversal (Dynamical Systems Interpretation)

PSML Programmable Smart Machines Lab

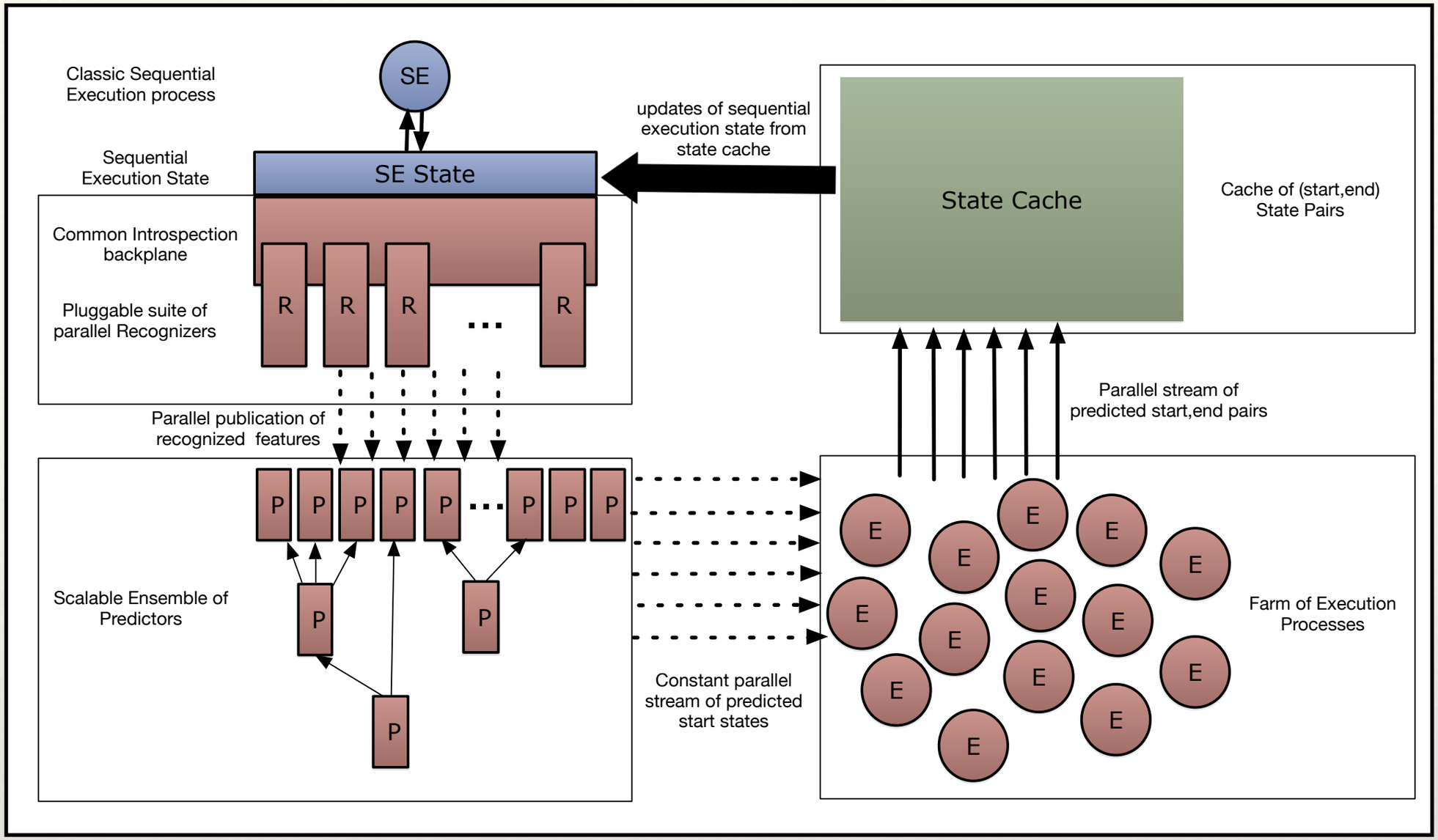
Computer As a State Vector

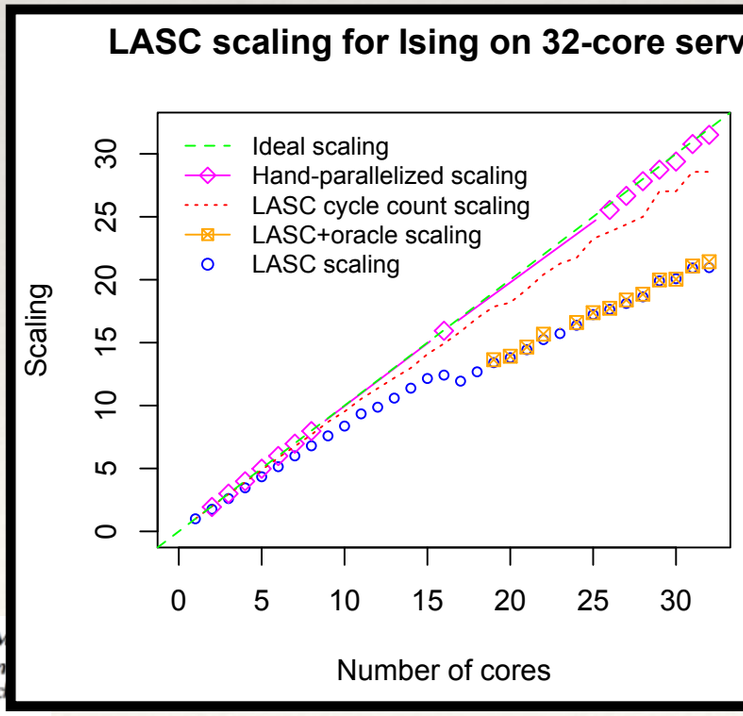


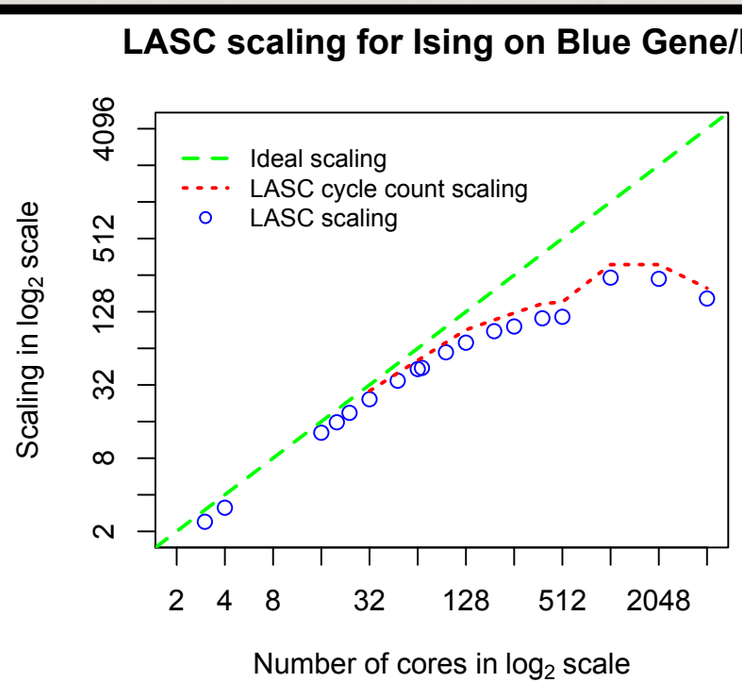


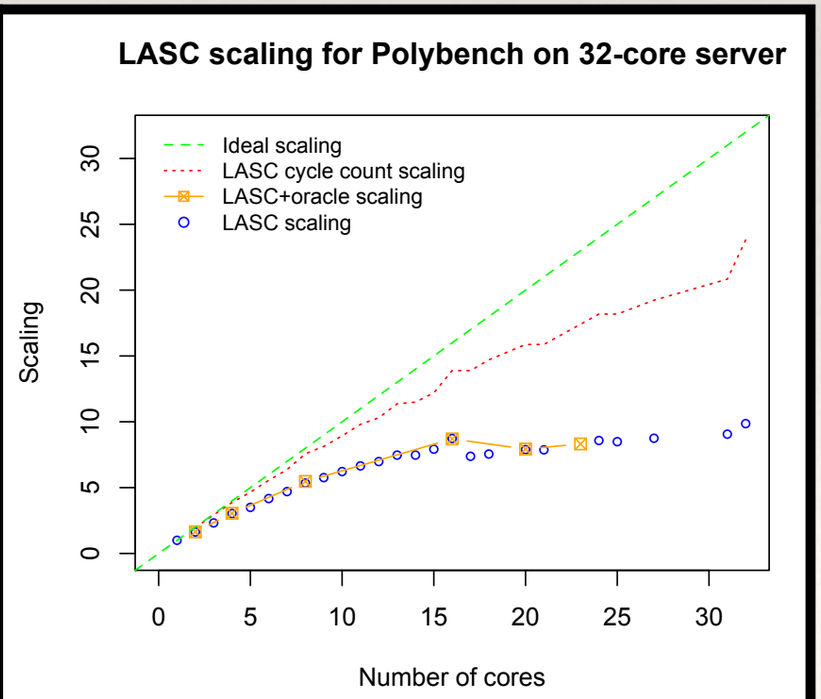
PSML
Programmable
Smart Machines
Lab

LASC

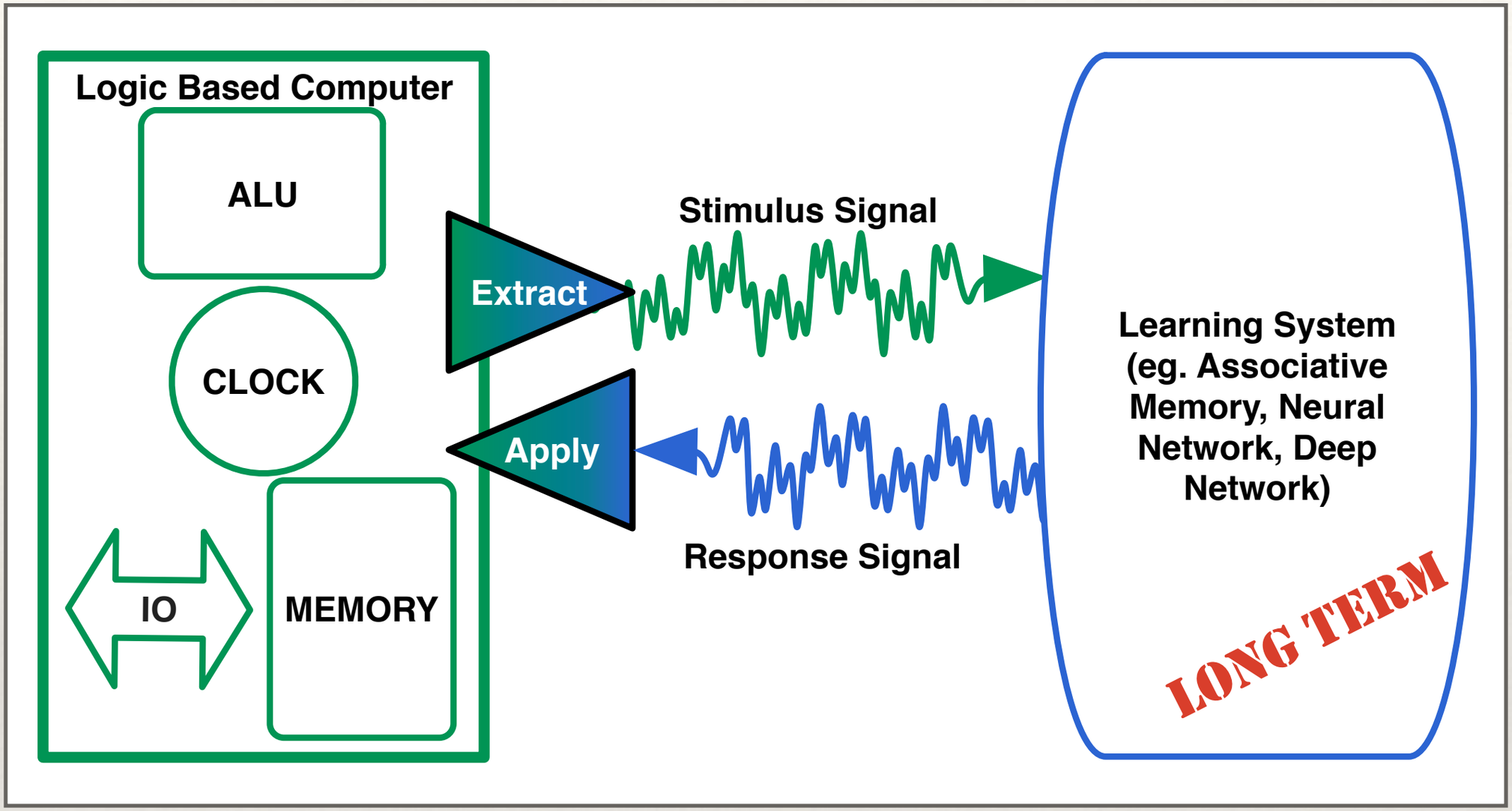

Classic Sequential Execution process
SE
Sequential Execution State
SE State
updates of sequential execution state from state cache
State Cache
Cache of (start,end) State Pairs
Common Introspection backplane
Pluggable suite of parallel Recognizers
R
R
R
R
Parallel publication of recognized features
Parallel stream of predicted start,end pairs
Scalable Ensemble of Predictors
P
Constant parallel stream of predicted start states
E
Farm of Execution Processes


LASC scaling for Ising on 32-core server
Ideal scaling
Hand-parallelized scaling
LASC cycle count scaling
LASC+oracle scaling
LASC scaling
Scaling
Number of cores


LASC scaling for Ising on Blue Gene/P
Ideal scaling
LASC cycle count scaling
LASC scaling
Scaling in log2 scale
Number of cores in log2 scale


LASC scaling for Polybench on 32-core server
Ideal scaling
LASC cycle count scaling
LASC+oracle scaling
LASC scaling
Scaling
Number of cores

PSM Neuromorphic Potential



A different role for Neuromorphic devices in a general purpose Computer Model

A Computation Cortex — A massive, power efficient, autoassociative memory of execution patterns

PSML
Programmable Smart Machines Lab

```
int inner(bool u[],bool v[],int d) {
  int i, E = 0;
  for (i = 0; i < d; i++) {
        E += u[i] * v[i];
  }
  return E;
}
```

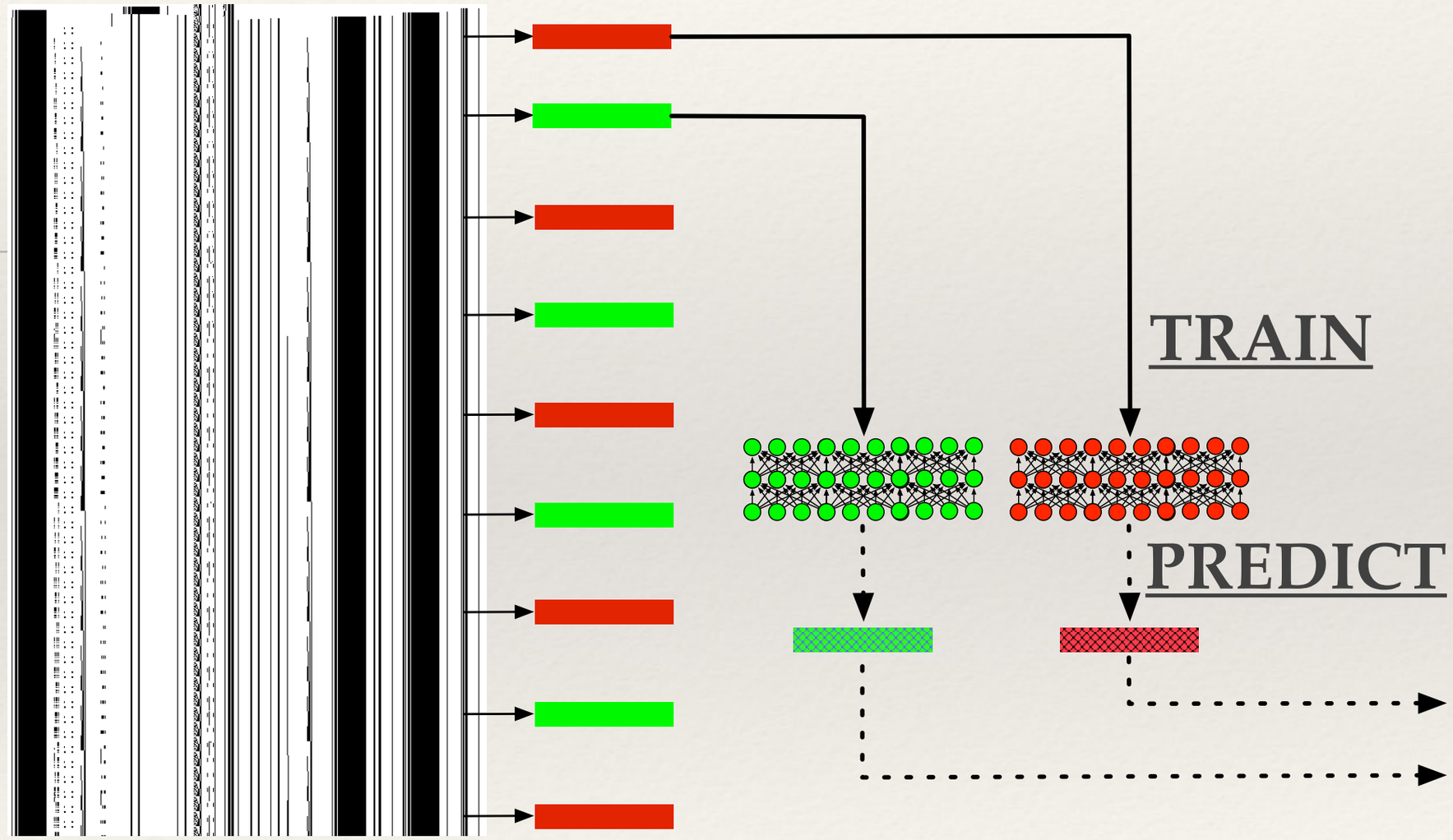
u[8192]

v[8192]

Probability of "1" is Independent and Identically Distributed

A micro-scale Experiment

1. Filter
2. Train
3. Predict

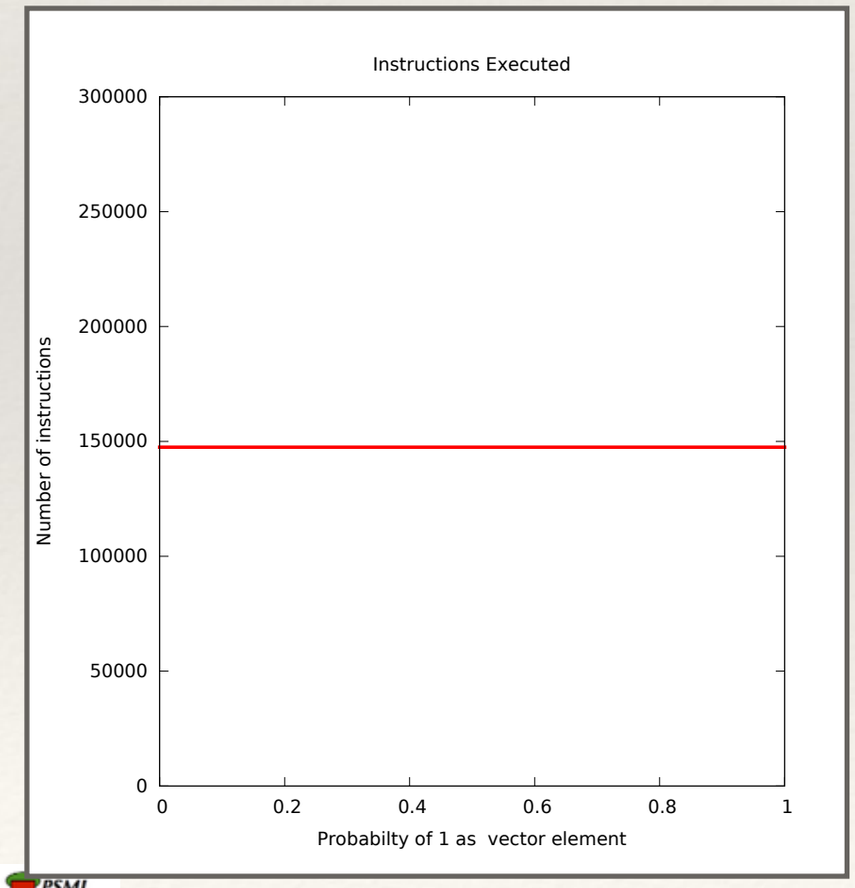


PSML
Programmable Smart Machines Lab

Filter

inner 0.1 Execution

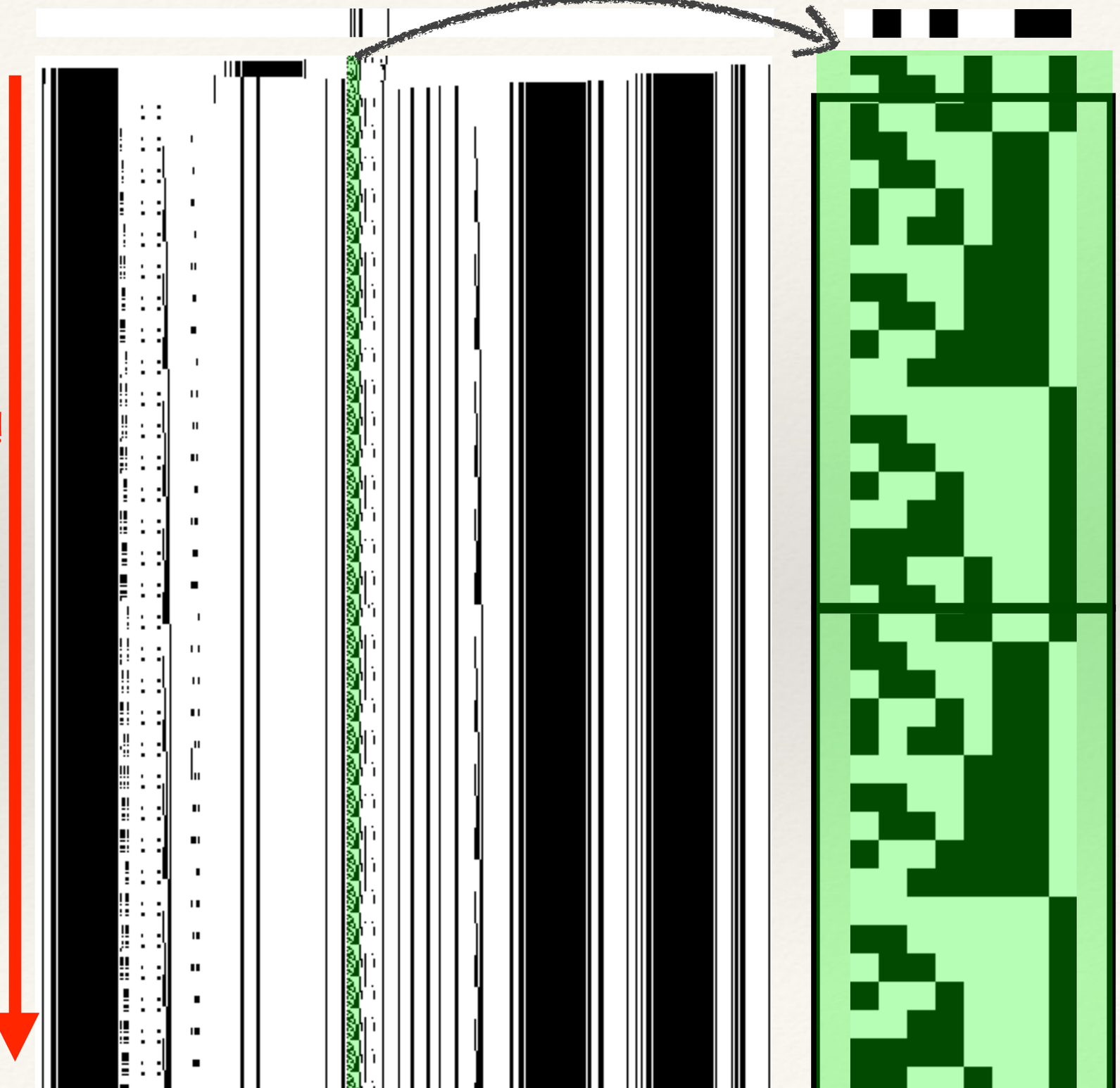
```
4001ca  4831ed              xor rbp, rbp
4001cd  5f                  pop rdi
4001ce  4889e6              mov rsi, rsp
4001d1  4883e4f0            and rsp, 0xfffffffffffffff0
4001d5  e8c6ffffff          call 0x4001a0
4001a0  55                  push rbp
4001a1  4889e5              mov rbp, rsp
4001a4  4883ec10            sub rsp, 0x10
4001a8  897dfc              mov [rbp-0x4], edi
4001ab  488975f0            mov [rbp-0x10], rsi
4001af  ba00200000          mov edx, 0x2000
4001b4  be00306000          mov esi, 0x603000
4001b9  bf00106000          mov edi, 0x601000
4001be  e881ffffff          call 0x400144
```



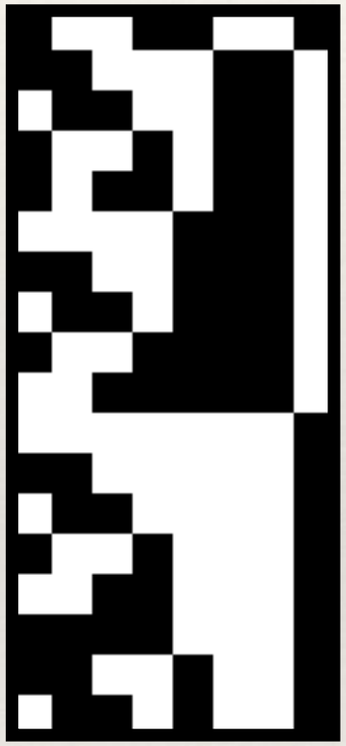
265 Changing Bits out of 1345216 State Bits

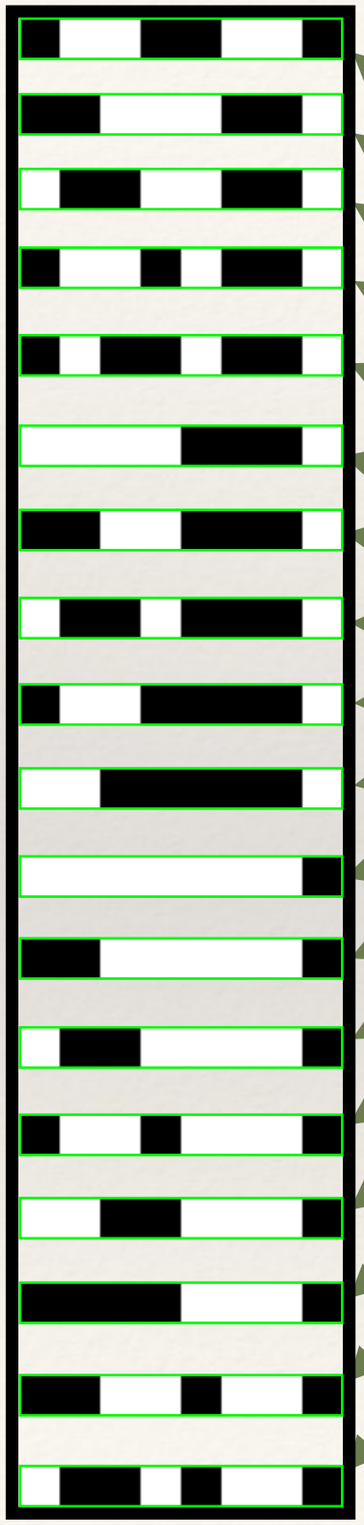
Byte 128

Time



PSML
Programmable
Smart Machines
Lab

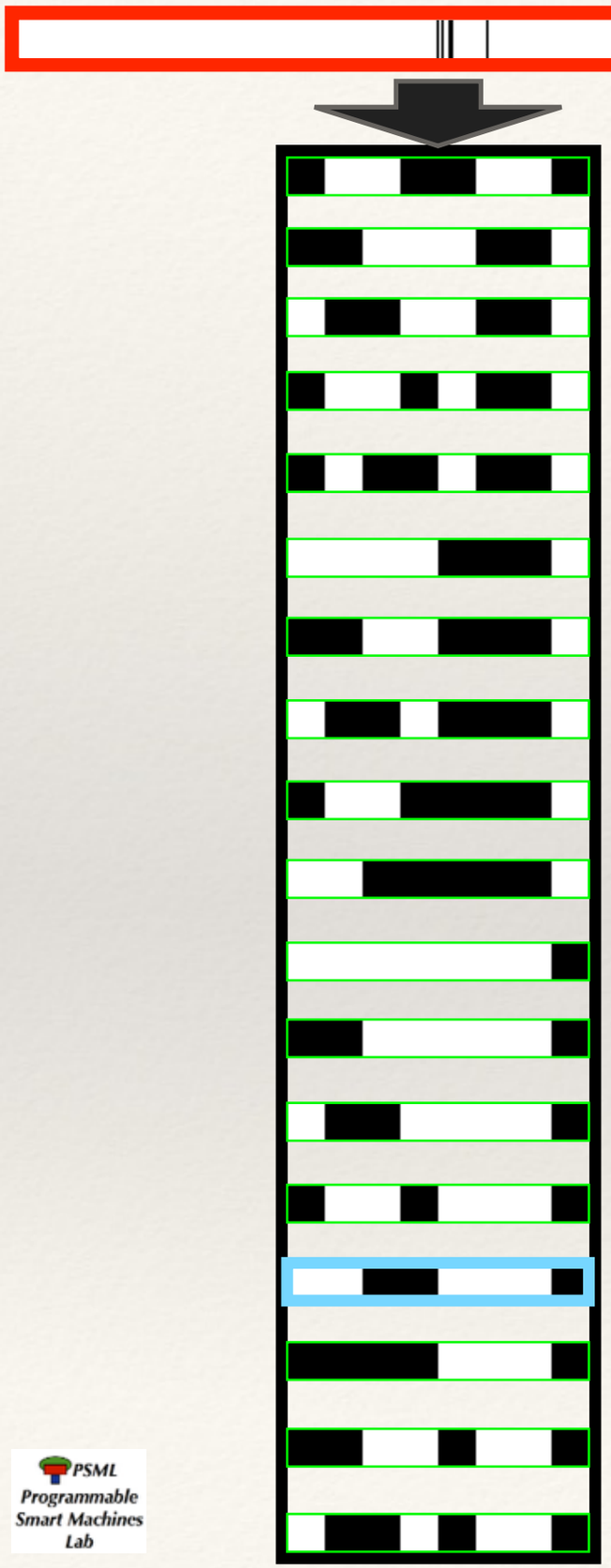




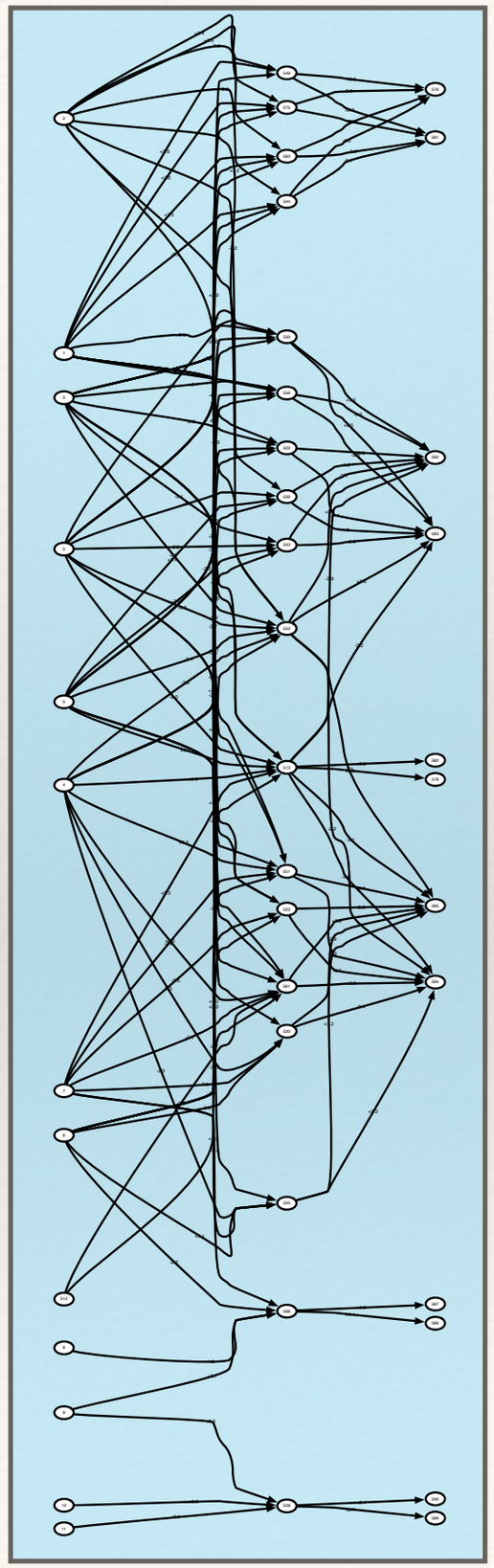
```
.0000000000400144 <inner>:
  400144:  55                    push   %rbp
  400145:  48 89 e5              mov    %rsp,%rbp
  400148:  48 89 7d e8           mov    %rdi,-0x18(%rbp)
  40014c:  48 89 75 e0           mov    %rsi,-0x20(%rbp)
  400150:  89 55 dc              mov    %edx,-0x24(%rbp)
  400153:  c7 45 f8 00 00 00 00  movl   $0x0,-0x8(%rbp)
  40015a:  c7 45 fc 00 00 00 00  movl   $0x0,-0x4(%rbp)
  400161:  eb 30                 jmp    400193 <inner+0x4f>
  400163:  8b 45 fc              mov    -0x4(%rbp),%eax
  400166:  48 63 d0              movslq %eax,%rdx
  400169:  48 8b 45 e8           mov    -0x18(%rbp),%rax
  40016d:  48 01 d0              add    %rdx,%rax
  400170:  0f b6 00              movzbl (%rax),%eax
  400173:  0f b6 d0              movzbl %al,%edx
  400176:  8b 45 fc              mov    -0x4(%rbp),%eax
  400179:  48 63 c8              movslq %eax,%rcx
  40017c:  48 8b 45 e0           mov    -0x20(%rbp),%rax
  400180:  48 01 c8              add    %rcx,%rax
  400183:  0f b6 00              movzbl (%rax),%eax
  400186:  0f b6 c0              movzbl %al,%eax
  400189:  0f af c2              imul   %edx,%eax
  40018c:  01 45 f8              add    %eax,-0x8(%rbp)
  40018f:  83 45 fc 01           addl   $0x1,-0x4(%rbp)
  400193:  8b 45 fc              mov    -0x4(%rbp),%eax
  400196:  3b 45 dc              cmp    -0x24(%rbp),%eax
  400199:  7c c8                 jl     400163 <inner+0x1f>
  40019b:  8b 45 f8              mov    -0x8(%rbp),%eax
  40019e:  5d                    pop    %rbp
  40019f:  c3                    retq
```

PSML
Programmable Smart Machines Lab

Train – Online Update



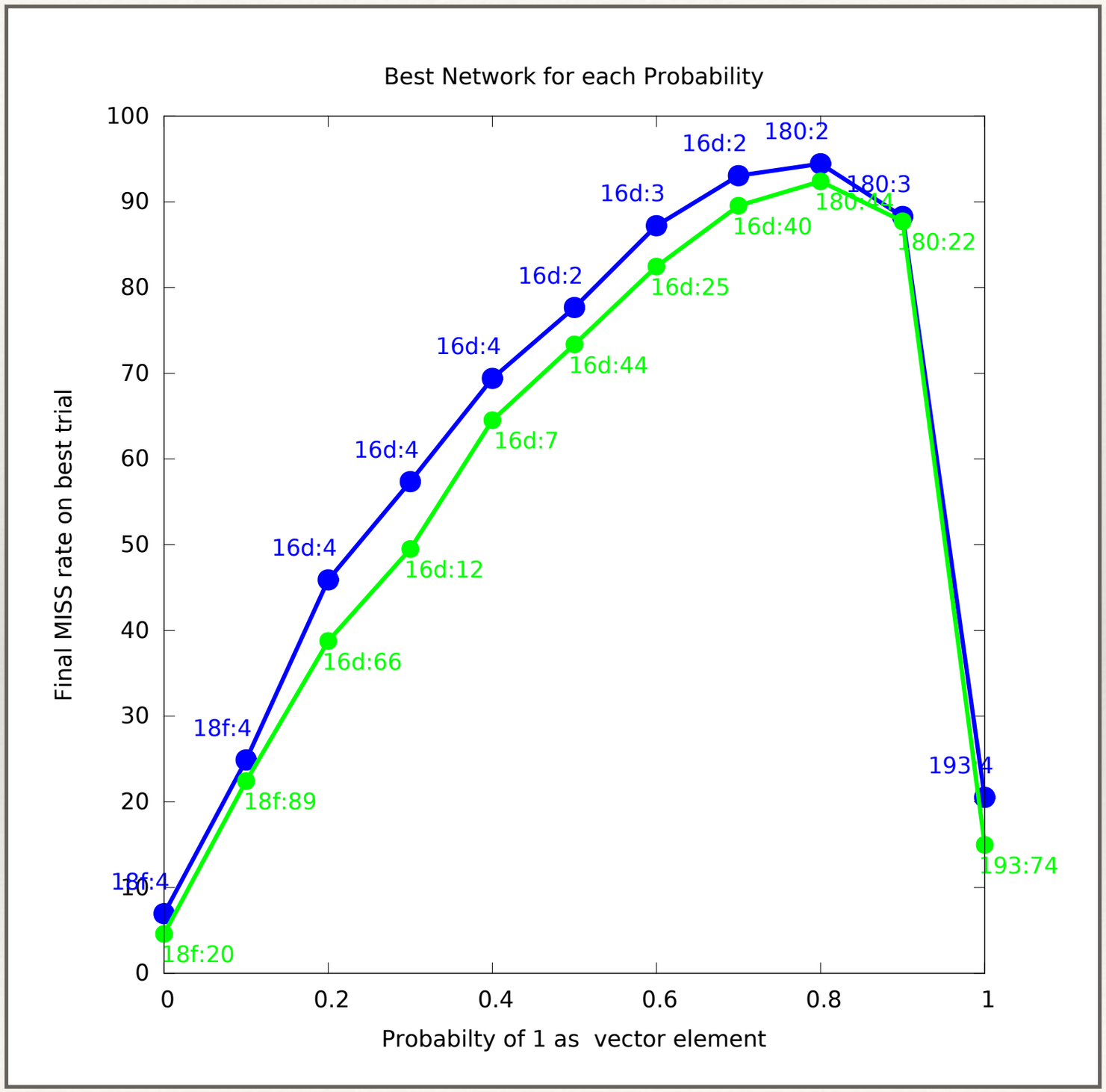
MASK = XOR(LAST,CUR) | MASK
BITS = CONDENSE(CUR & MASK)
FANN_TRAIN(NET,OLDBITS,BITS,…)
OLDBITS = BITS



Predict

```
400163:  mov    -0x4(%rbp),%eax
400166:  movslq %eax,%rdx
400169:  mov    -0x18(%rbp),%rax
40016d:  add    %rdx,%rax
400170:  movzbl (%rax),%eax
400173:  movzbl %al,%edx
400176:  mov    -0x4(%rbp),%eax
400179:  movslq %eax,%rcx
40017c:  mov    -0x20(%rbp),%rax
400180:  add    %rcx,%rax
400183:  movzbl (%rax),%eax
400186:  movzbl %al,%eax
400189:  imul   %edx,%eax
40018c:  add    %eax,-0x8(%rbp)
40018f:  addl   $0x1,-0x4(%rbp)
400193:  mov    -0x4(%rbp),%eax
400196:  cmp    -0x24(%rbp),%eax
400199:  jl     400163 <inner+0x1f>
```

40018f	0.0,0.1
40016d	0.2-0.7
400180	0.8,0.9
400193	1.0

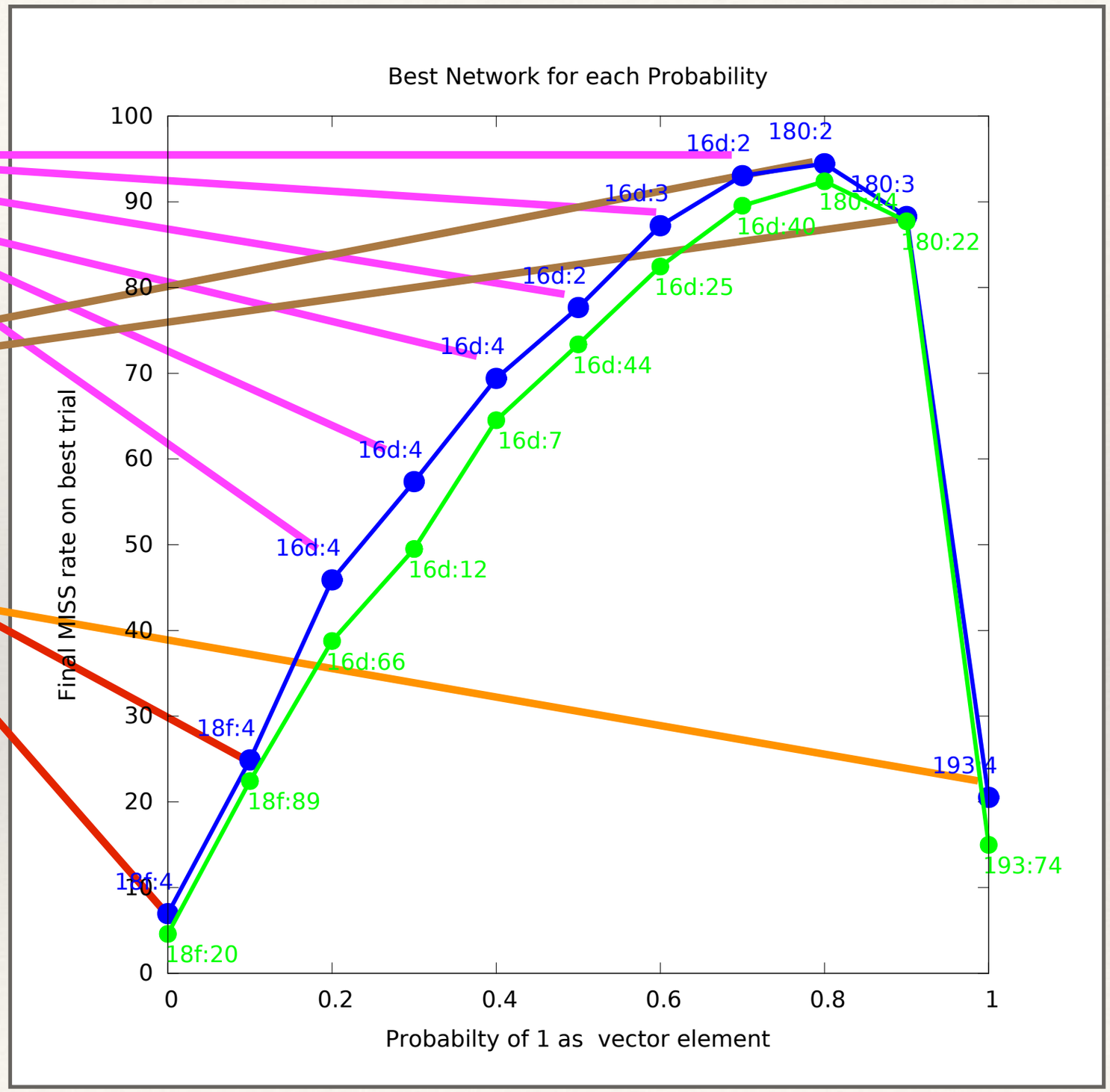


PSML
Programmable Smart Machines Lab

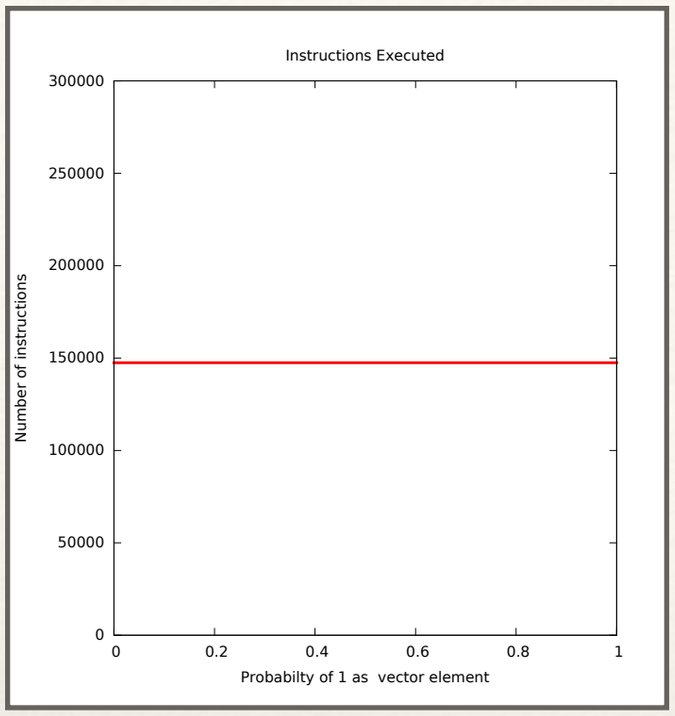
Predict

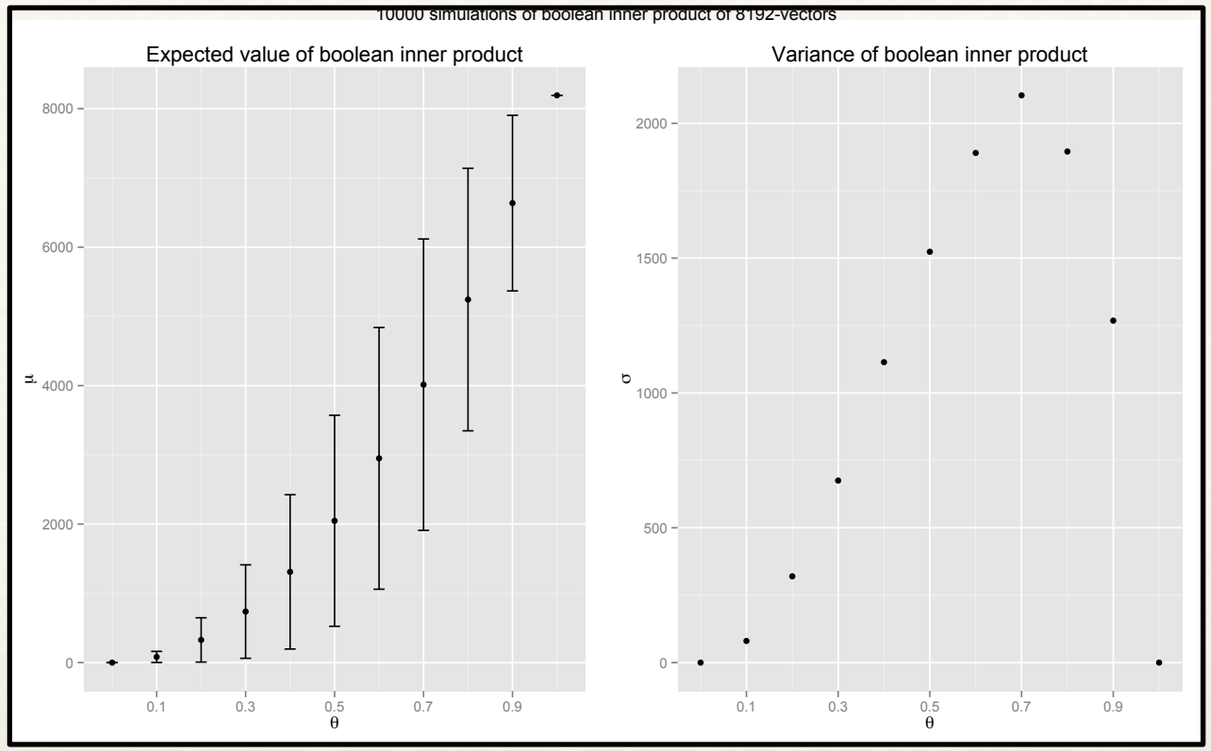
```
. 400163:   mov    -0x4(%rbp),%eax
  400166:   movslq %eax,%rdx
  400169:   mov    -0x18(%rbp),%rax
  40016d:   add    %rdx,%rax
  400170:   movzbl (%rax),%eax
  400173:   movzbl %al,%edx
  400176:   mov    -0x4(%rbp),%eax
  400179:   movslq %eax,%rcx
  40017c:   mov    -0x20(%rbp),%rax
  400180:   add    %rcx,%rax
  400183:   movzbl (%rax),%eax
  400186:   movzbl %al,%eax
  400189:   imul   %edx,%eax
  40018c:   add    %eax,-0x8(%rbp)
  40018f:   addl   $0x1,-0x4(%rbp)
  400193:   mov    -0x4(%rbp),%eax
  400196:   cmp    -0x24(%rbp),%eax
  400199:   jl     400163 <inner+0x1f>
```

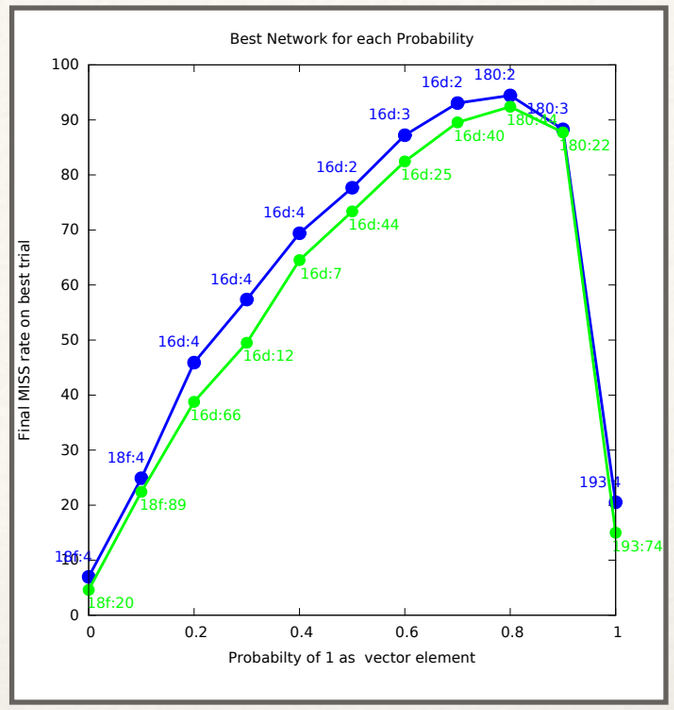
40018f	0.0,0.1
40016d	0.2-0.7
400180	0.8,0.9
400193	1.0



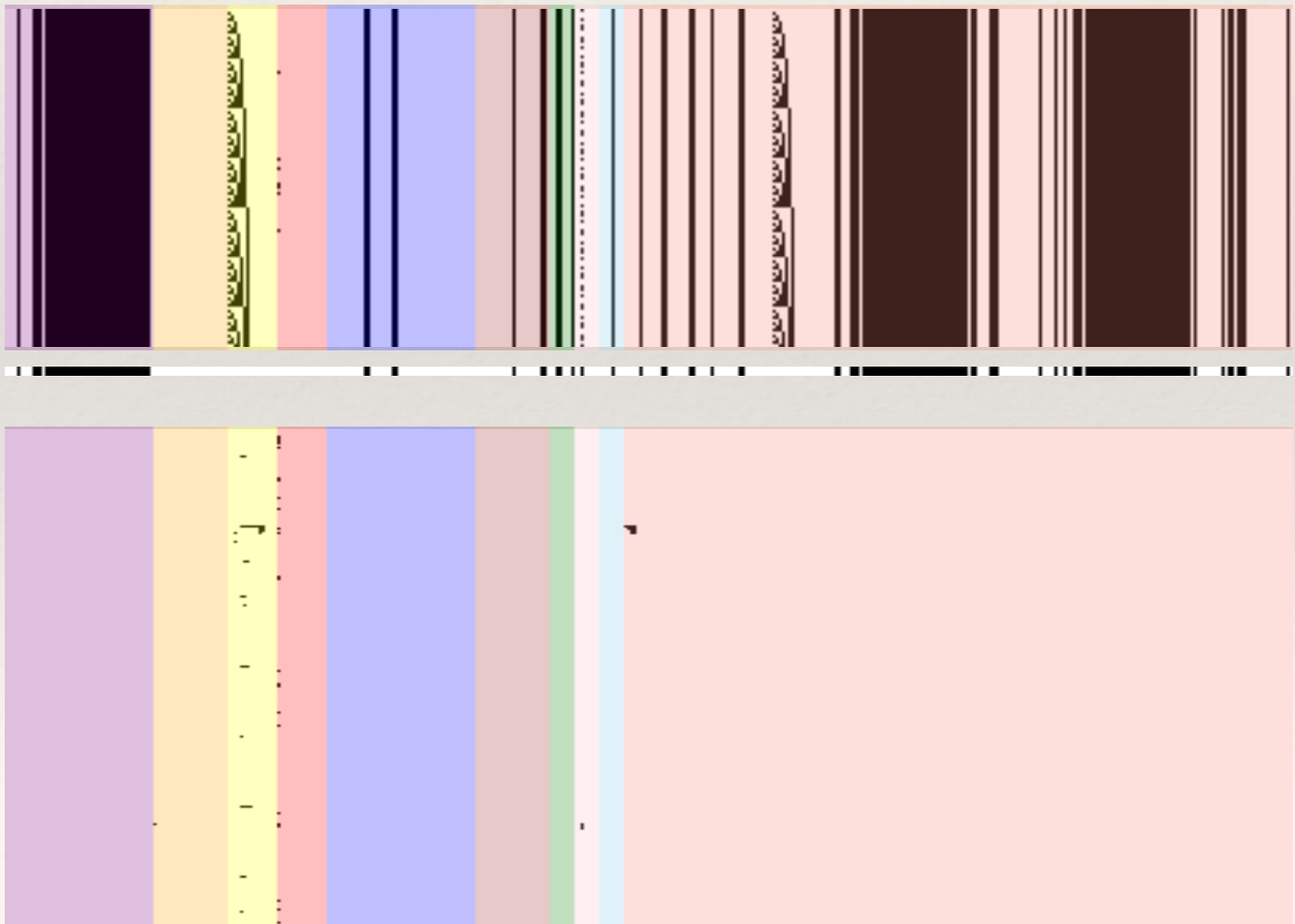
PSML
Programmable Smart Machines Lab







RBP	purple
RAX	orange
RCX	yellow
RDX	red
RSI	blue
RDI	brown
RIP	green
EFLAGS	pink
RSP	skyblue
[stack]	salmon



PSML
Programmable Smart Machines Lab

Bridge to Some (or No) Where?

Will the promise of Neuromorphic technologies be on the other side?

PSML
Programmable Smart Machines Lab

“Spare a little Silicon?”