Unity in Diversity:
Phylogenetic-inspired techniques for reverse engineering and detection of malware families

Wei Ming Khoo, Pietro Lio
University of Cambridge

1st SysSec workshop
6th July 2011

Phylogenetics is the study of **evolutionary relationships** among **groups** of organisms (e.g. HIV), discovered through **molecular sequencing data** (DNA/proteins) and **morphological data matrices**.

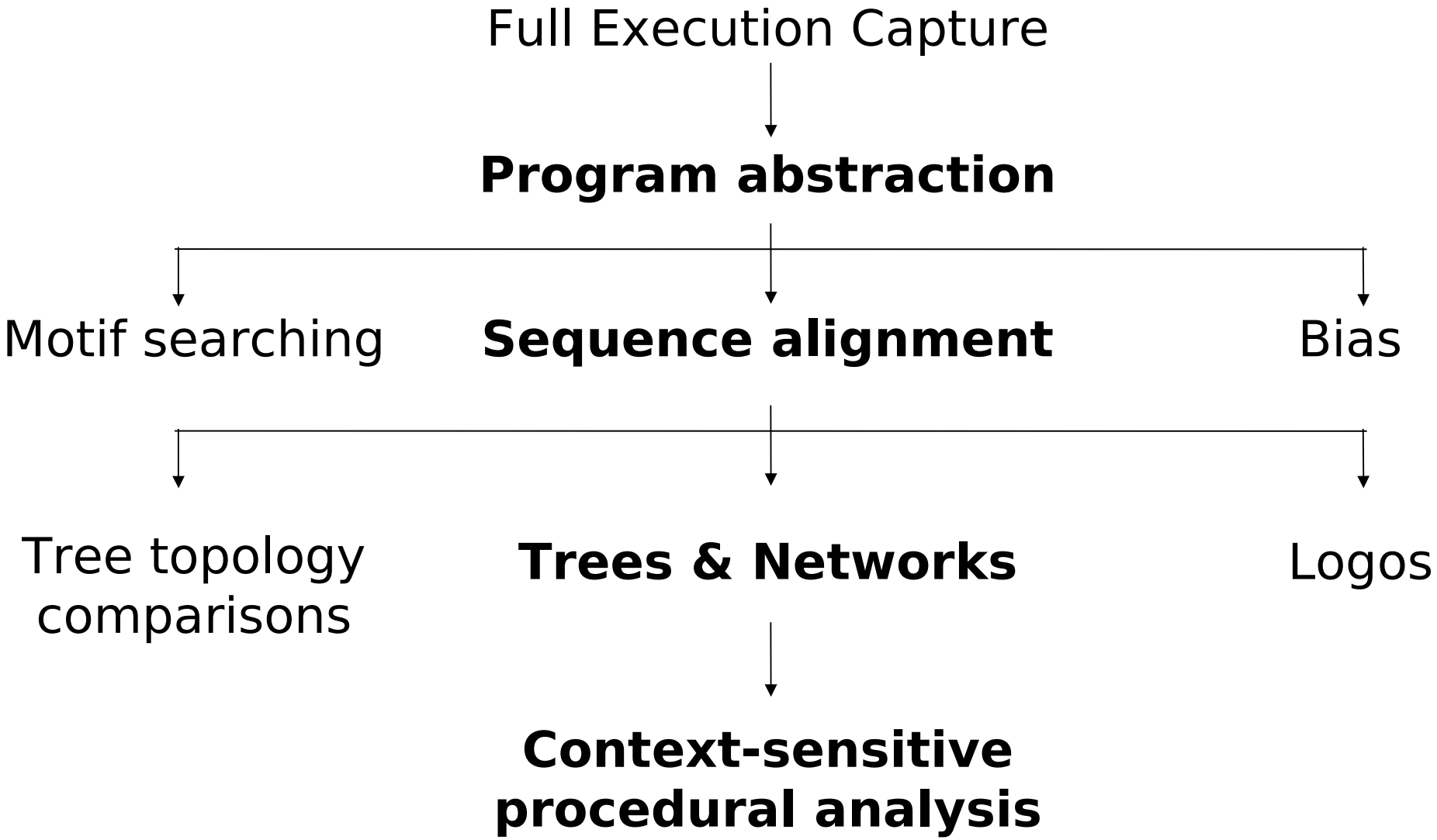
Phylon (tribe, race) + genetikos (origin)

Evolutionary biologists deal with large datasets (e.g. billions of HIV variants)

Can we apply the same techniques to malware?

Wikipedia

Scope


Full Execution Capture
Program abstraction
Motif searching
Sequence alignment
Bias
Tree topology comparisons
Trees & Networks
Logos
Context-sensitive procedural analysis

Part 1 – Execution capture & Abstraction

What is a molecule of program?

Execution capture

- Chronicler – captures all register, memory, disassembly information per instruction
- Uses the Pin binary instrumentation framework
- ~5 million instructions per execution trace (~500Mb)
- Post-hoc analysis using scripts
- VMWare Server with fully patched Windows XP SP3 VM
- No anti-anti-VM techniques were deployed

http://www.cl.cam.ac.uk/~wmk26/chronicler

Dataset

- 6 malware families ~4Gb (courtesy of Sophos)
  - Mainly analysed FakeAV-DO family
- 'Skyhoo' botnet (courtesy of Richard Clayton)
  - Propagation through Skype and Yahoo IM (Photo :))
  - ~600k infected hosts
  - Monitored passively by being operator on bot-herder IRC server
  - Collecting malware since Nov 2010
  - ~a new binary every 3-5 days

Abstraction

- We used mnemonics and API calls
- Easily mapped to an alphabet, and fed to existing phylogenetic tools
- 2 alphabets per symbol
- Mnemonics good for certain types of code metamorphism
- Weak as operand info is discarded
- Future work: Reversible abstractions

```
push ebp
mov ebp, esp
push dword ptr
[ebp+8]
call <foo>
```

push, mov, push, call

PHMOPHCA

What is a motif?

- A motif is a sequence pattern that occurs repeatedly in a group of related sequences
- Seen another way, a motif encodes information about the group of sequences (entropy)
- E.g. a motif 'AGC' from columns *a* to *c* is able to encode 3 x 2 = 6 bits of information
- A motif 'T' in column *e* encodes 0 bits
- Motif search = maximise information encoded for a given width *w*
- Sequence alignment not required
- Motifs can be pre-defined or emergent

```
   abcde
1  AGCTA
2  AGCTC
3  AGCGG
4  AGCGC
```

Unpacking via API motif search

FakeAV-do\00ec

QDILILILILHYHYQRTVHYTVNSILILILILNSILNSILNSILNSILNSILNSILNSIL
NSILNS

FakeAV-do\0b19

VEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVE
VEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVE
VEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVE
VEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVE
VEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVEVE
VEVEVEVEVEVEVEVEVEVEVEVEVEVEVEILILILILILHYHYQRTVHYTVNSIL
ILILILNSILNSILNSILNSILNSILNSILNSILNSILNS

FakeAV-do\0c59

IRLTILILILILILHYHYQR

FakeAV-do\0d77

TAAQILILILILILHYHYQR

```
HY  kernel32.dll:LocalAlloc
IK  kernel32.dll:GlobalFree
IL  kernel32.dll:GetProcAddress
NS  kernel32.dll:LoadLibraryA
QR  kernel32.dll:LocalFree
TV  kernel32.dll:VirtualProtect
```

Part 2 –
Sequence alignment

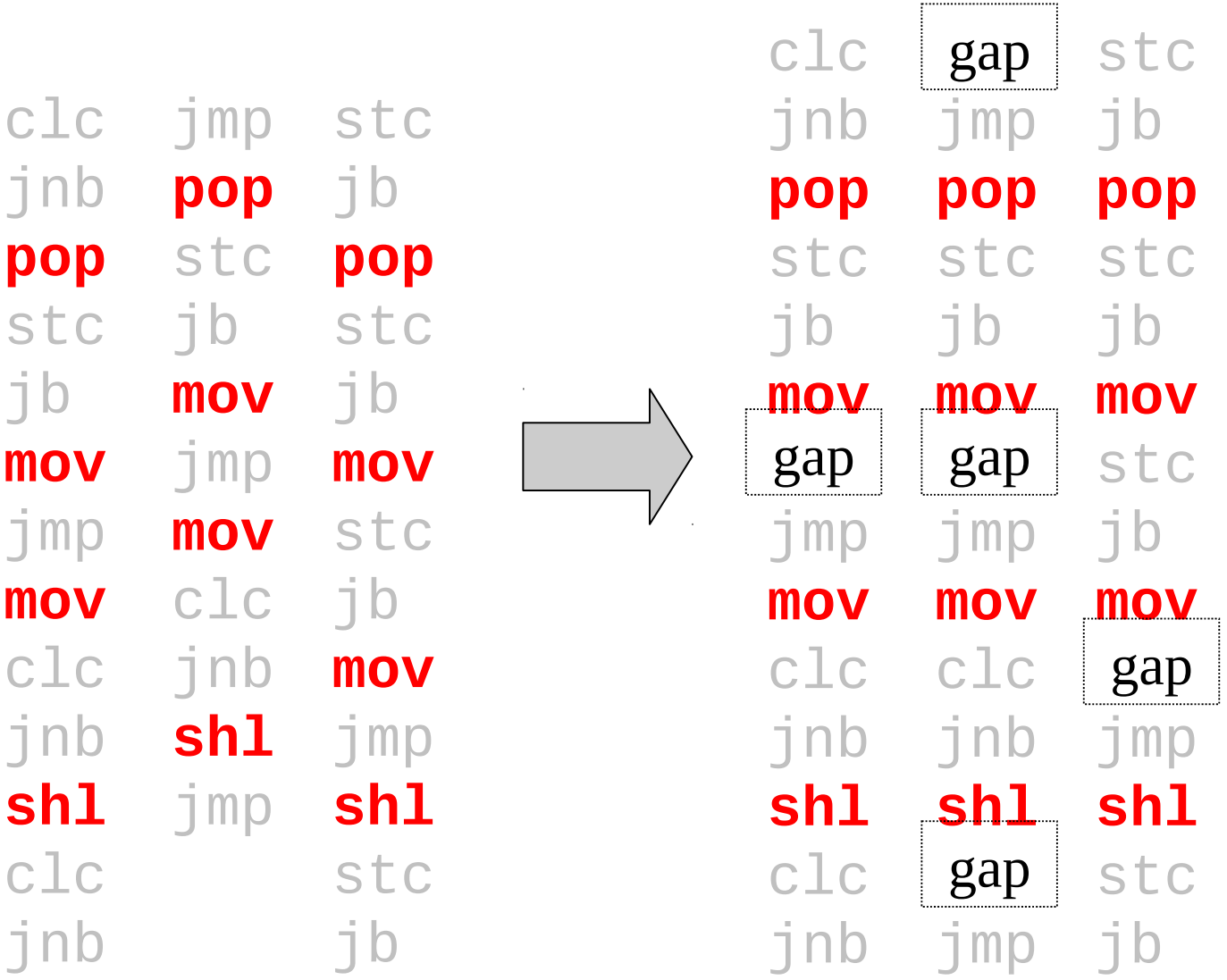
Unity: How similar are they?

Sequence alignment

- Arranging two or more sequences placed one below each other, so that regions of similarity coincide
- Scoring system
  - Reward sequence agreement, penalise disagreement (“mutation”) and insertion of a blank (“gap”)
  - Penalty function (a matrix) for mutations and gaps dependent on application
  - E.g. stc (set carry flag) ; jb (jump if below or CF=1)
    clc (clear carry flag) ; jnb (jump if not below or CF=0)
    jmp (unconditional jump)

    All 3 code snippets are semantically equivalent, thus the penalty for such mutations can be set to 0

Sequence alignment

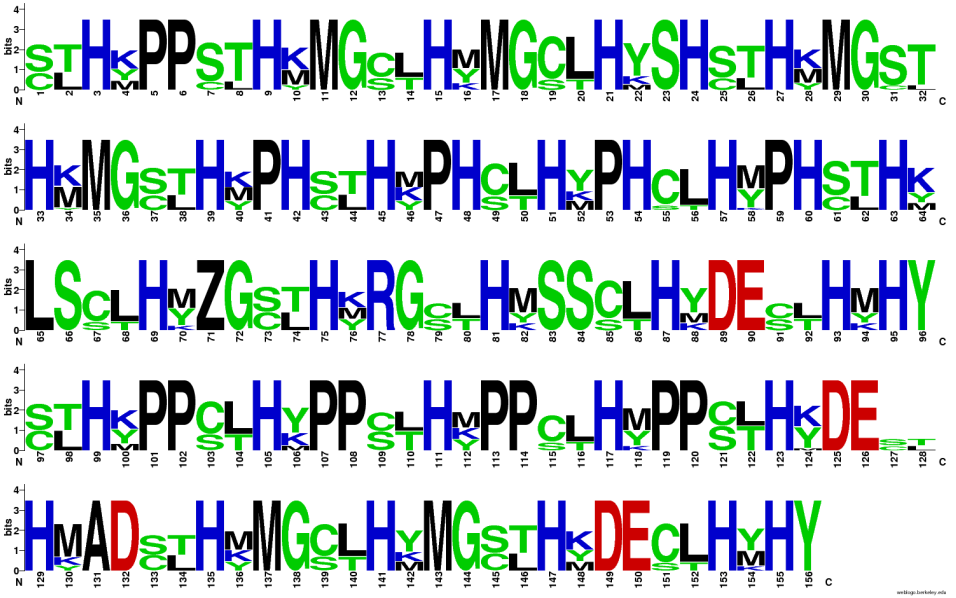


FakeAV procedures extracted after LocalFree() API call

Code normalisation via logos

A logo is a graphical method for identifying patterns in a set of aligned sequences

Characters are stacked for each position in the sequence, height of each letter is proportional to its frequency



```
clc         stc
jnb  jmp    jb
pop  pop    pop
stc  stc    stc
jb   jb     jb
mov  mov    mov
            stc
jmp  jmp    jb
mov  mov    mov
clc  clc
jnb  jnb    jmp
shl  shl    shl
clc         stc
jnb  jmp    jb
```

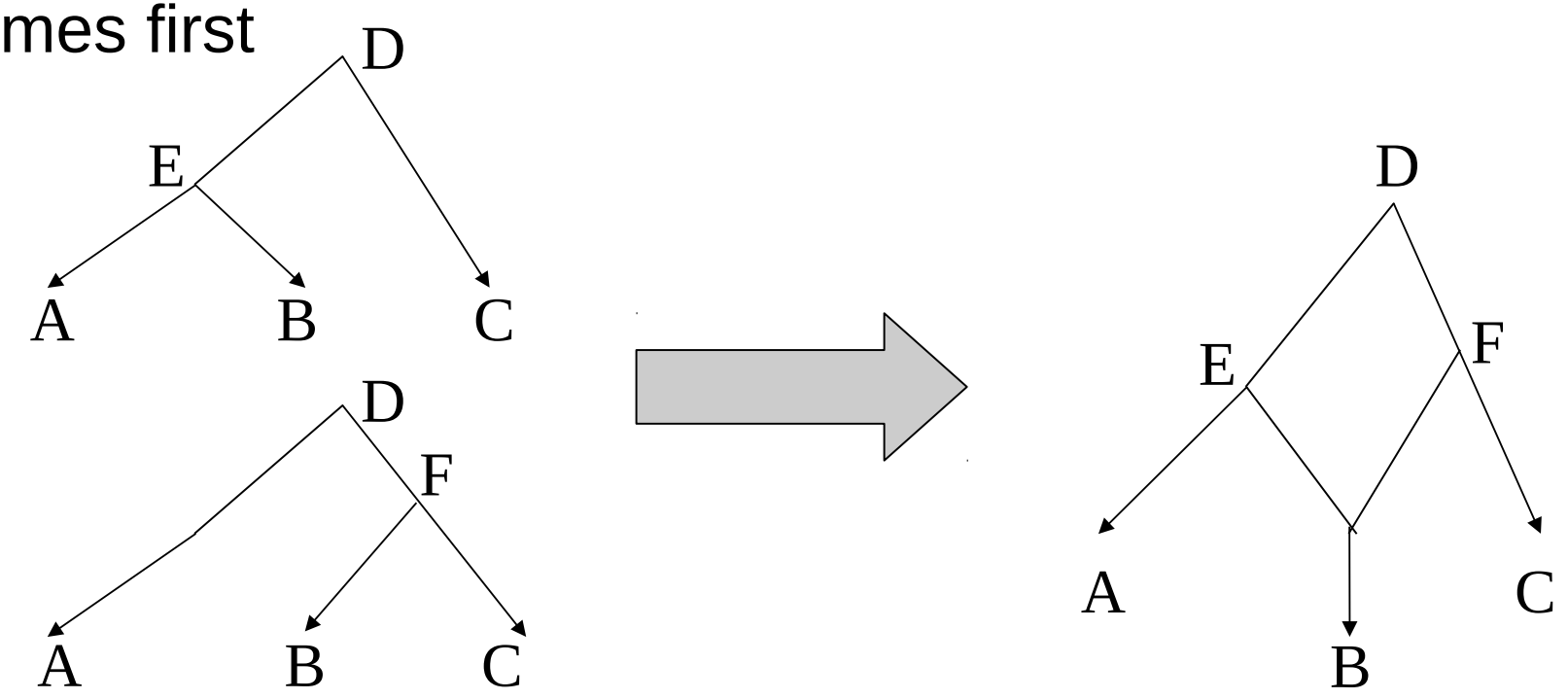
Part 3 –
Trees and networks

Diversity: How different are they?

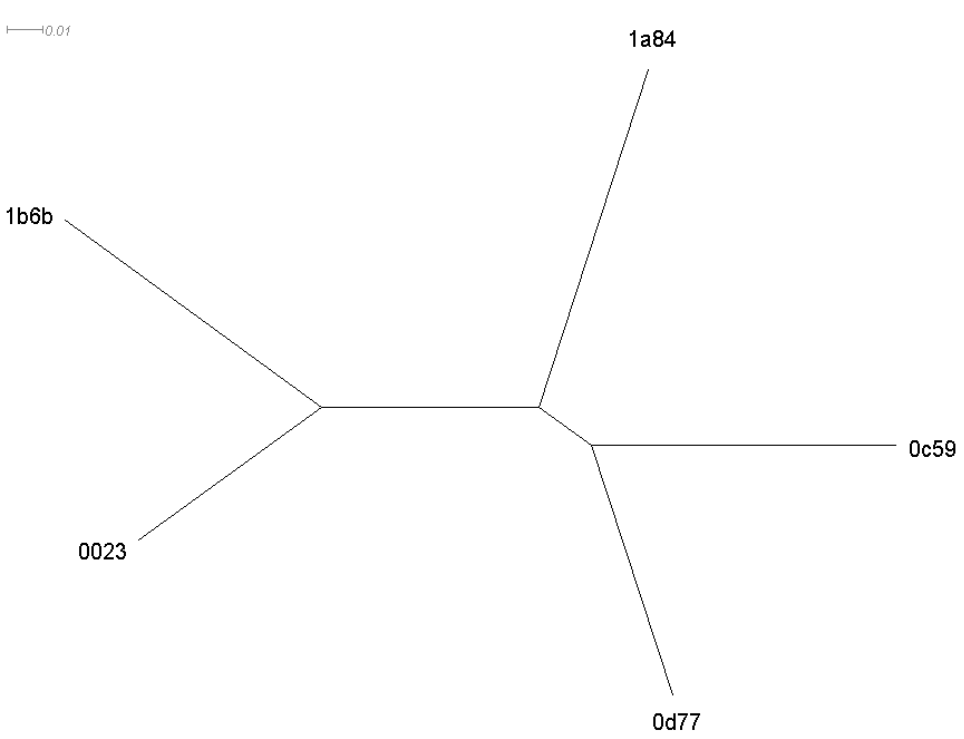
Trees vs Networks

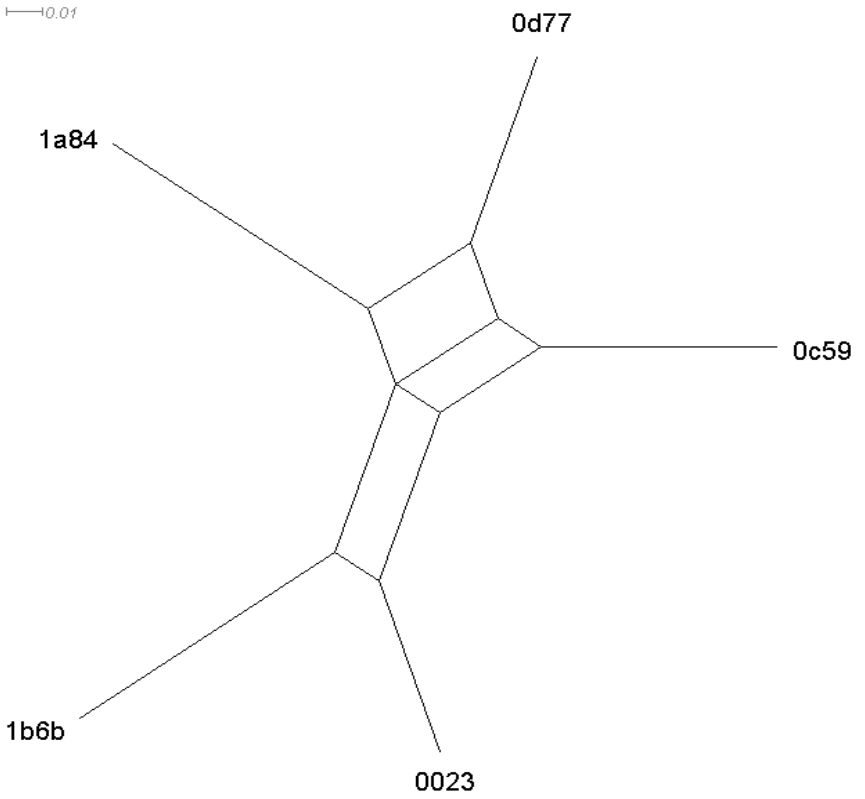
Phylogenetic trees are constructed from multiple sequence alignment, however more than one tree may be possible

Phylogenetic networks are a generalisation of trees for modelling uncertainty about which bifurcation in the tree comes first



Trees vs Networks

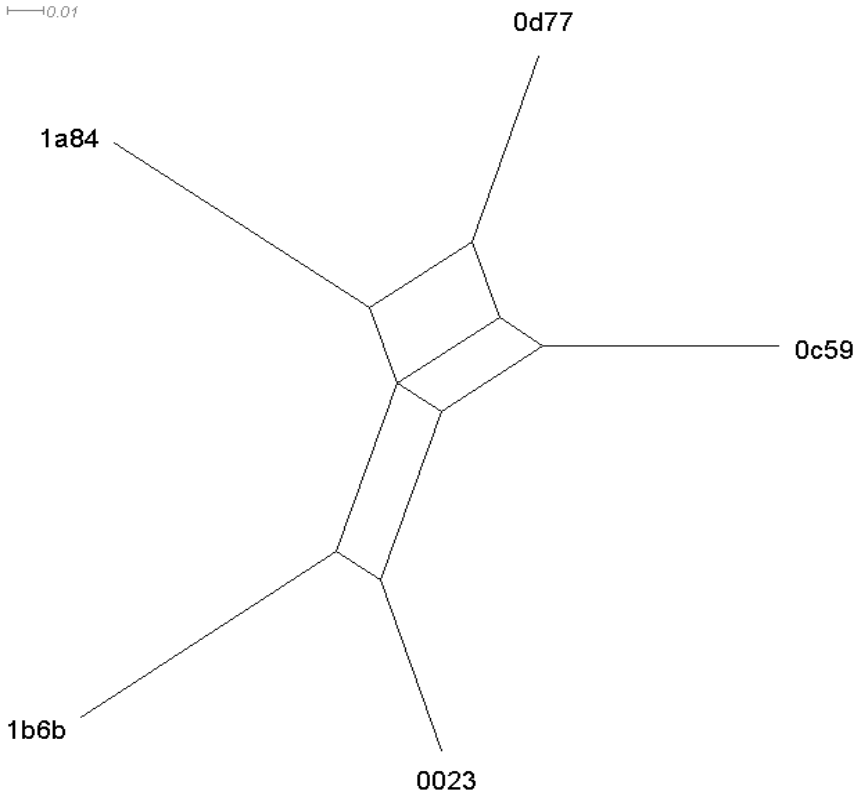


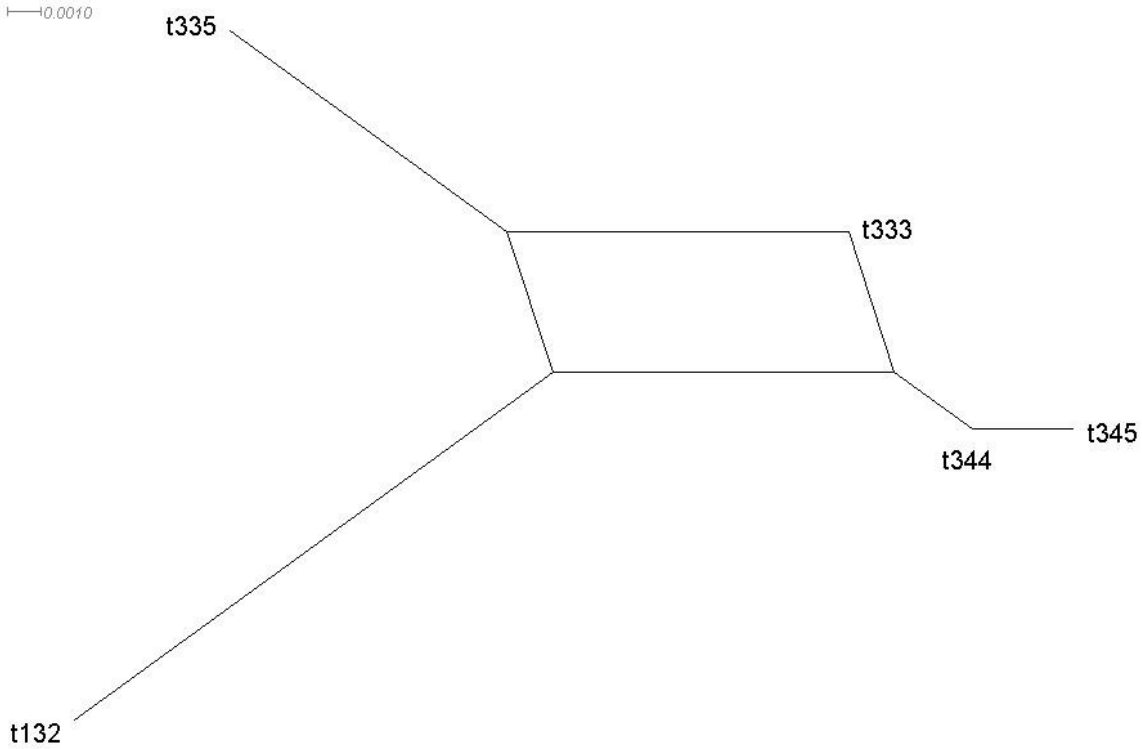


FakeAV-DO family tree and network

Diversity as a result of `nop`-equivalent code metamorphism more evident in network

Diversity patterns via networks





FakeAV-DO

Triangle (benign)

8 vs 3 predicted ancestors

Future work: Parametrise code metamorphic diversity patterns

Sequence bias

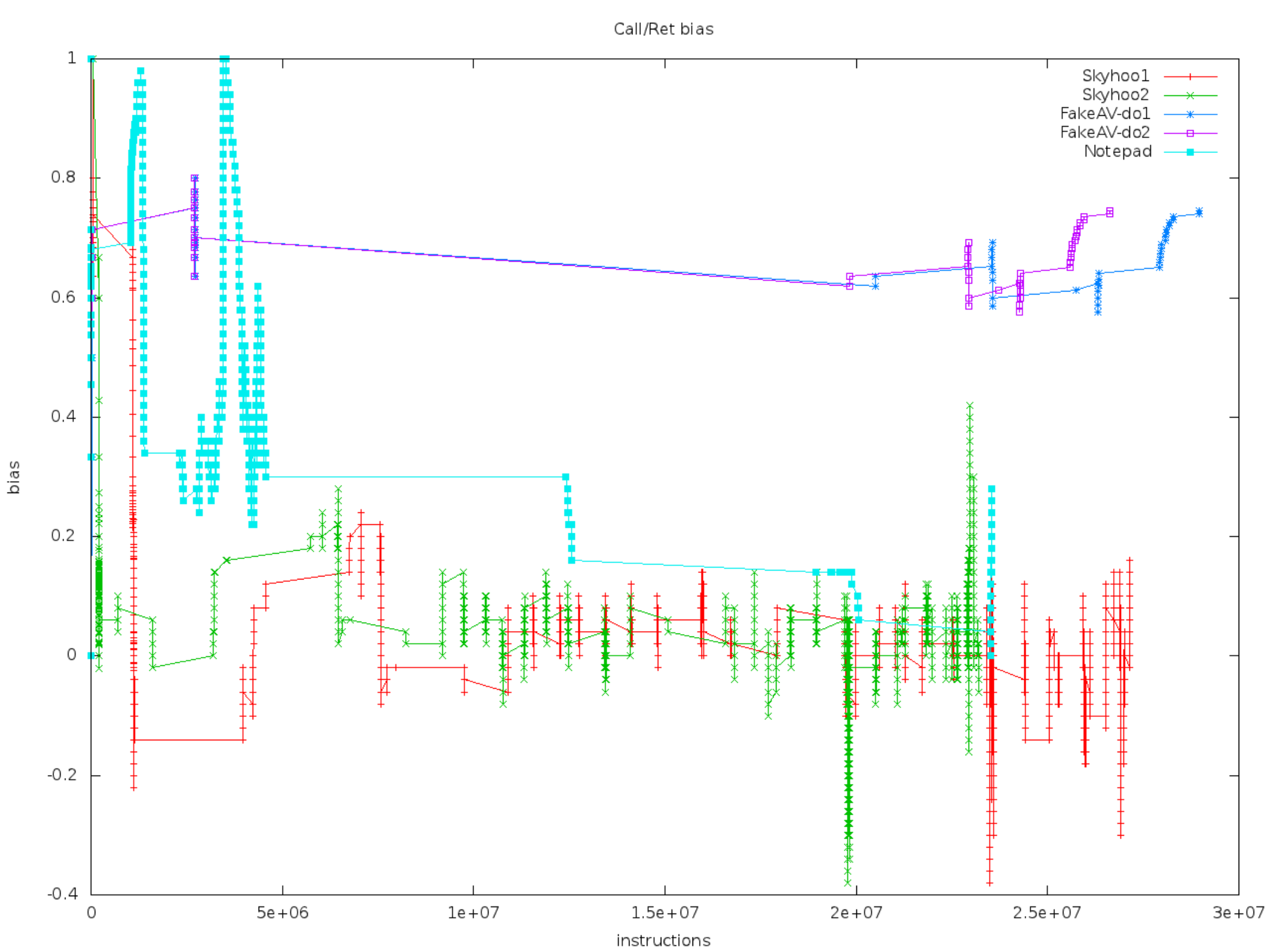
Composition of symmetric or opposite instructions

E.g. call/ret, push/pop

Let *q* be the number of instruction *Q* in a window of *n* instructions, and *w* the number of instruction *W*, a symmetric instruction to *Q*

$$bias = \frac{q - w}{q + w}$$

Both FakeAV-DO and Skyhoo abuse call/ret instructions and thus have abnormal composition biases


Call/Ret bias
Skyhoo1
Skyhoo2
FakeAV-do1
FakeAV-do2
Notepad
bias
instructions
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
0
5e+06
1e+07
1.5e+07
2e+07
2.5e+07
3e+07

Context-sensitive procedure analysis

Investigate similarities between procedures used by FakeAV-DO, Skyhoo, Notepad, 7zip, WinSCP and the triangle program

Context of procedure = APIs preceding or succeeding it

6 Experiments:

1) Sequences preceding the GetProcAddress API

2) Sequences preceding the GetProcAddress API, normalised

3) Sequences succeeding the GetProcAddress API

4) Sequences succeeding the GetProcAddress API, normalised

5) Sequences between two consecutive GetProcAddress APIs

6) Sequences between two consecutive GetProcAddress APIs, normalised

Context-sensitive procedure analysis

Expt	Num of sequen-ces Benign	FakeAV	Skyhoo	TP0	TP1	TP2	Total
1	20	94	42	86.1%	99.3%	90.8%	95.0%
2	20	94	42	87.3%	79.0%	89.8%	83.1%
3	24	94	43	85.0%	100%	84.6%	93.8%
4	24	94	43	82.1%	100%	85.2%	93.8%
5	**10**	**67**	**12**	**83.9%**	**100%**	**100%**	**98.3%**
6	**10**	**67**	**12**	**81.6%**	**100%**	**100%**	**97.9%**

- Matches were highest when both contexts were taken into consideration (experiments 5 and 6)
- Code normalisation did not affect true positive rate

  => Analysis can cope with nop-equivalent code metamorphism
- Future work: more comprehensive study, exploratory analysis

Future work

Reversible abstractions

More comprehensive study

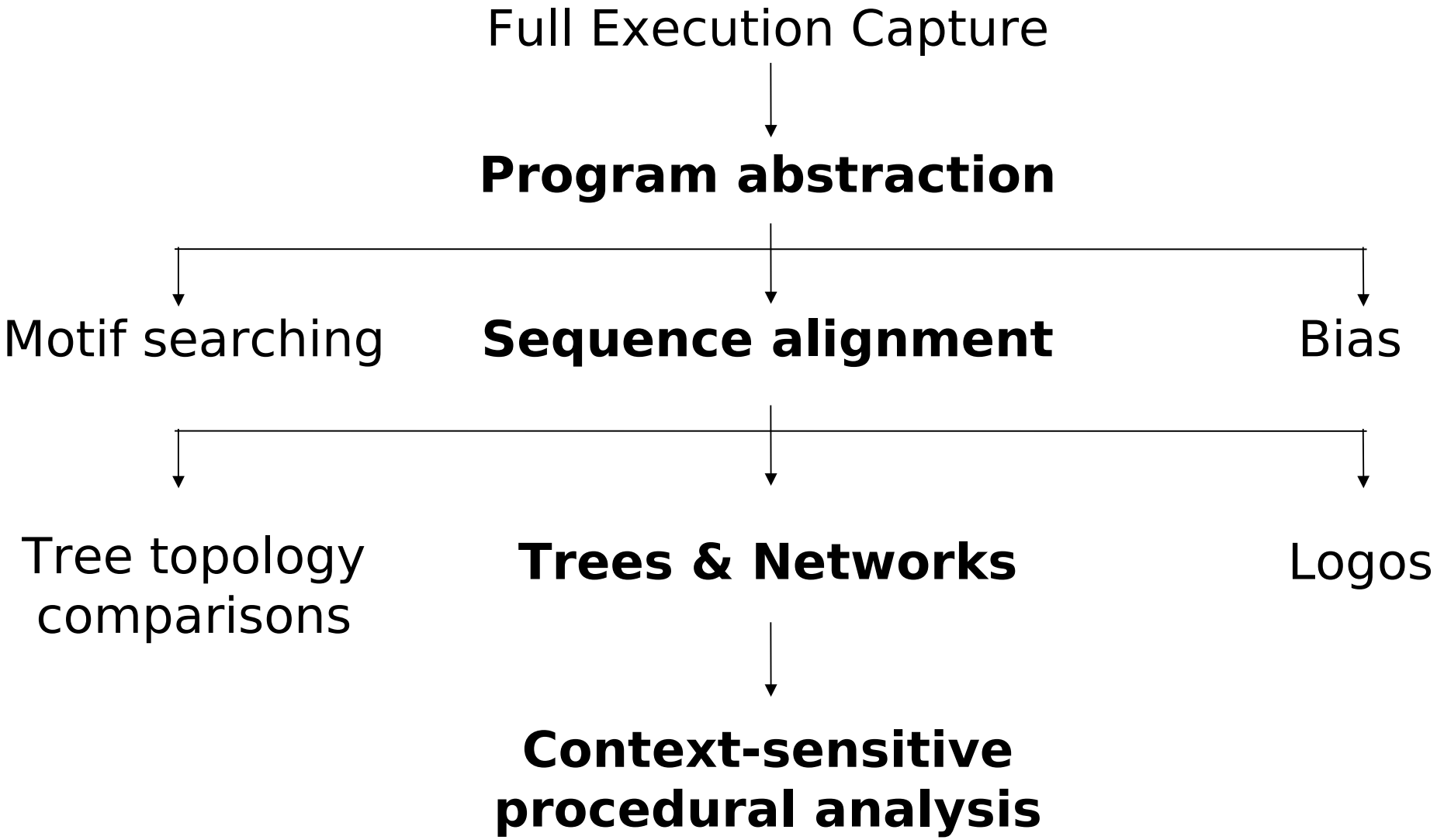
Parametrise networks to understand diversity patterns

Alignment – study penalty functions for different code metamorphic engines, e.g. w32.Evol

Improve execution capture

- using hardware-assisted virtualisation
- dealing with multi-threading

Summary


Full Execution Capture
Program abstraction
Motif searching
Sequence alignment
Bias
Tree topology comparisons
Trees & Networks
Logos
Context-sensitive procedural analysis

Phylogenetic tools

- Lots, many of them are web-based
- Phylip by Joe Felsenstein, U. Washington
  - General purpose
- ClustalW by Gibson et al., European Bioinformatics Inst.
  - Sequence alignment and tree construction, web-based
- Splitstree by Huson and Bryant, Eberhard Karls Universitat Tubingen
  - Network construction
- WebLogo by Crooks et al., UC Berkeley
  - Logo construction, web-based
- MEME by Bailey and Elkan, N'tl Biomedical Computation Resource
  - Motif search, web-based

Prior work in malware phylogenetics

- Goldberg et al. 1998 (n-gram based, DAGs)
- Erdelyi and Carrera 2004 (cfg based, Clustering/Trees)
- Kasim et al. 2005 (opcode based, Clustering/Trees)
- Ma et al. 2006 (n-gram based, Clustering/Trees)
- Wehner 2007 (compression distance based, Clustering/Trees)
- Focus mostly on phylogenetic trees
- Focus on detection and classification

Gibbs sampling

- Gibbs sampling is a Monte-Carlo method, which seeks to maximize a likelihood function over the input sequence data
- Parameters: starting positions of the motif (a_i), width of motif (w), observations/information (x)
- Likelihood function: $L(A,w|x)$
- E.g. A=(c,a,b,c), w=3,
  $x=I_{motif} = \sum I_j = 2w - H_{motif} = 6$
  (motif of width w bases)

```
   abcdefg
1  GCAGCTA
2  AGCTCAG
3  CAGCGGA
4  GAAGCGT
```