The Reverse Engineering Language REIL and its applications

Sebastian Porst (sebastian.porst@zynamics.com)
zynamics GmbH

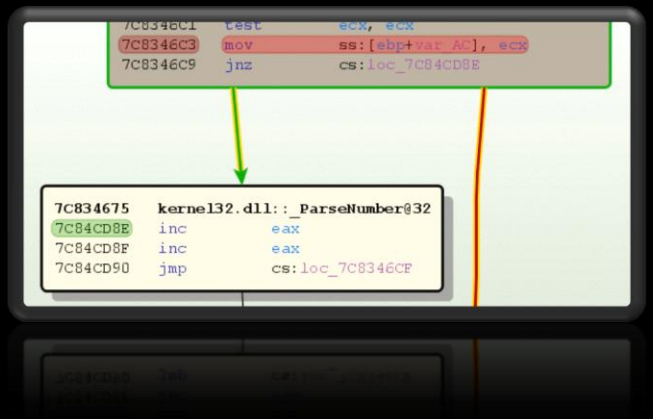
HITB SECCONF 2009

DUBAI دبي

20TH - 23RD APRIL 2009

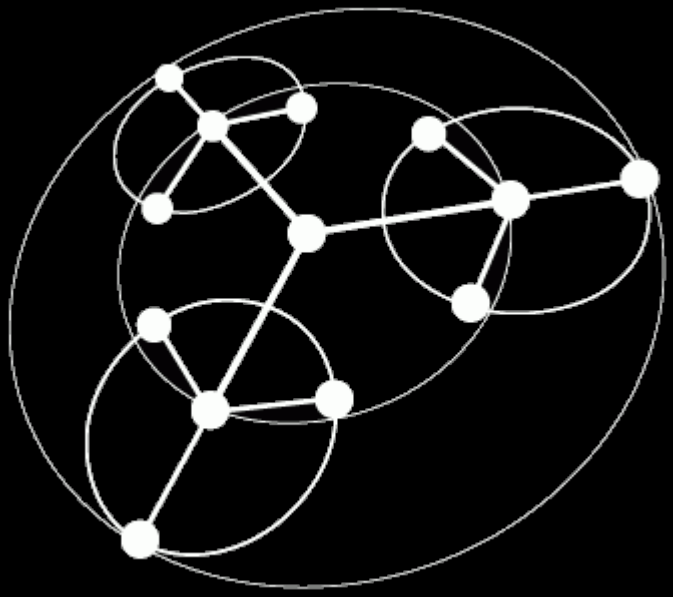
I write code for

... sometimes I debug other people‘s code



We love graphs

We love static analysis



We want to improve binary code RE

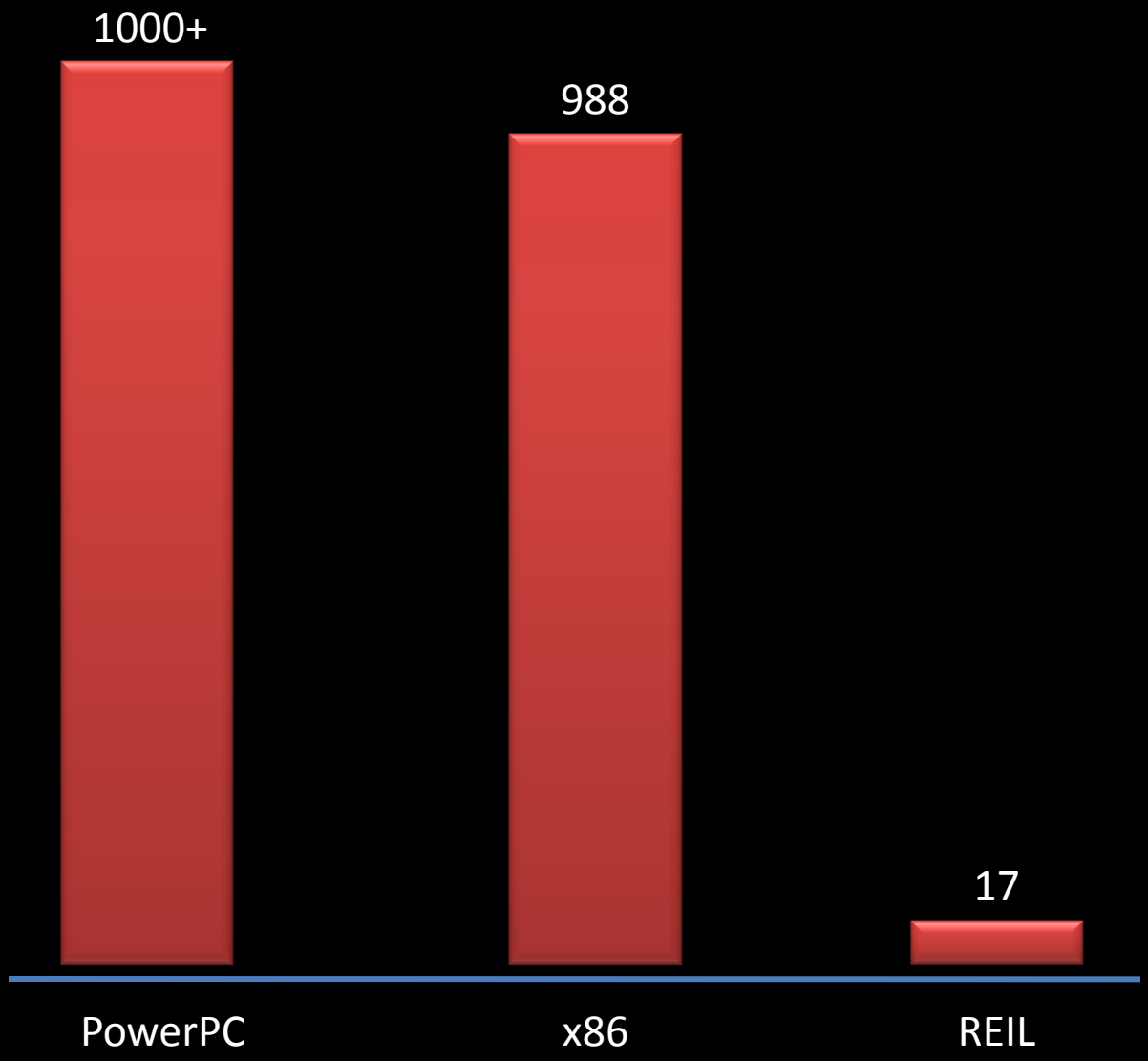
REIL

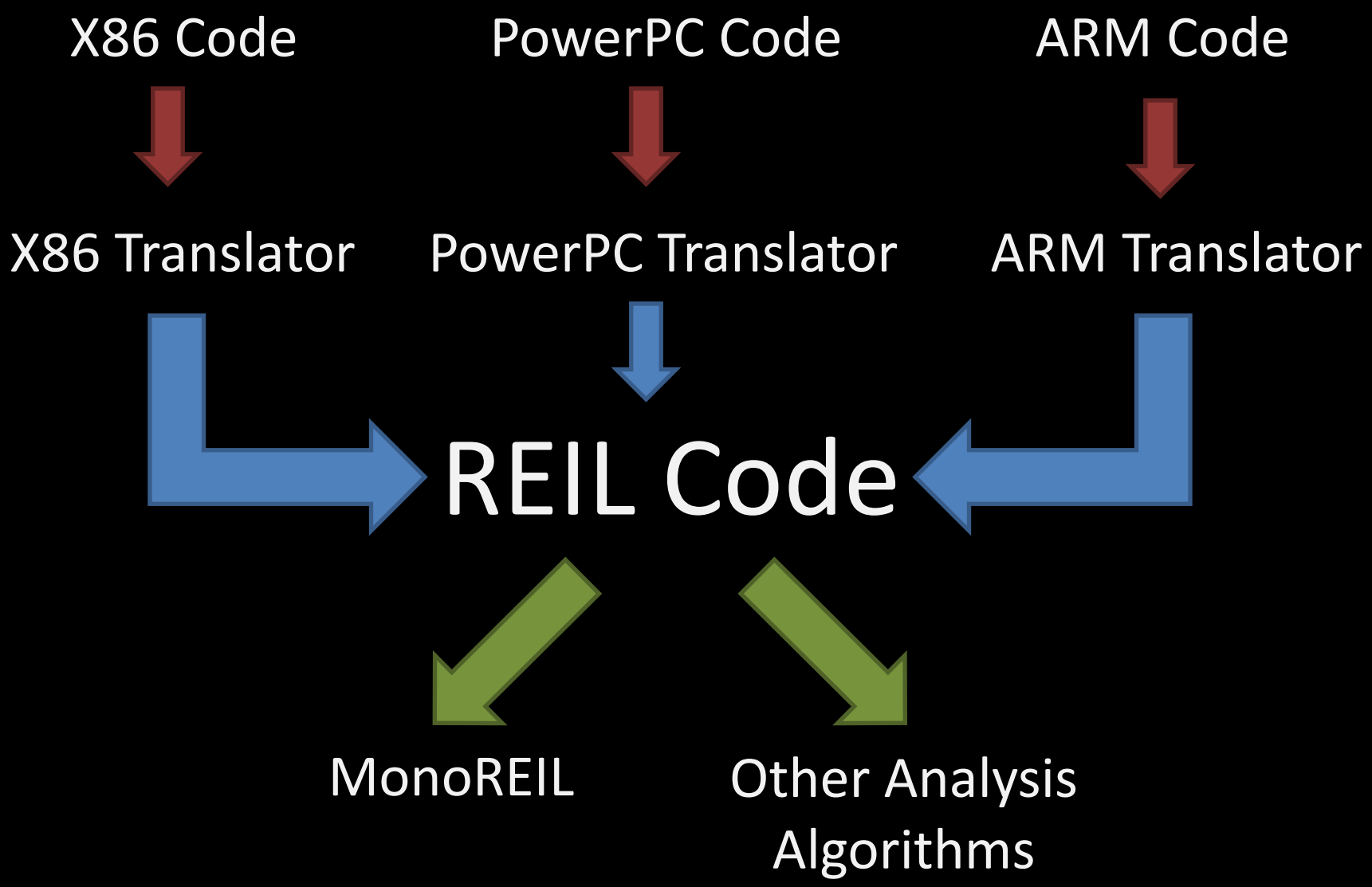
Reverse Engineering Intermediate Language

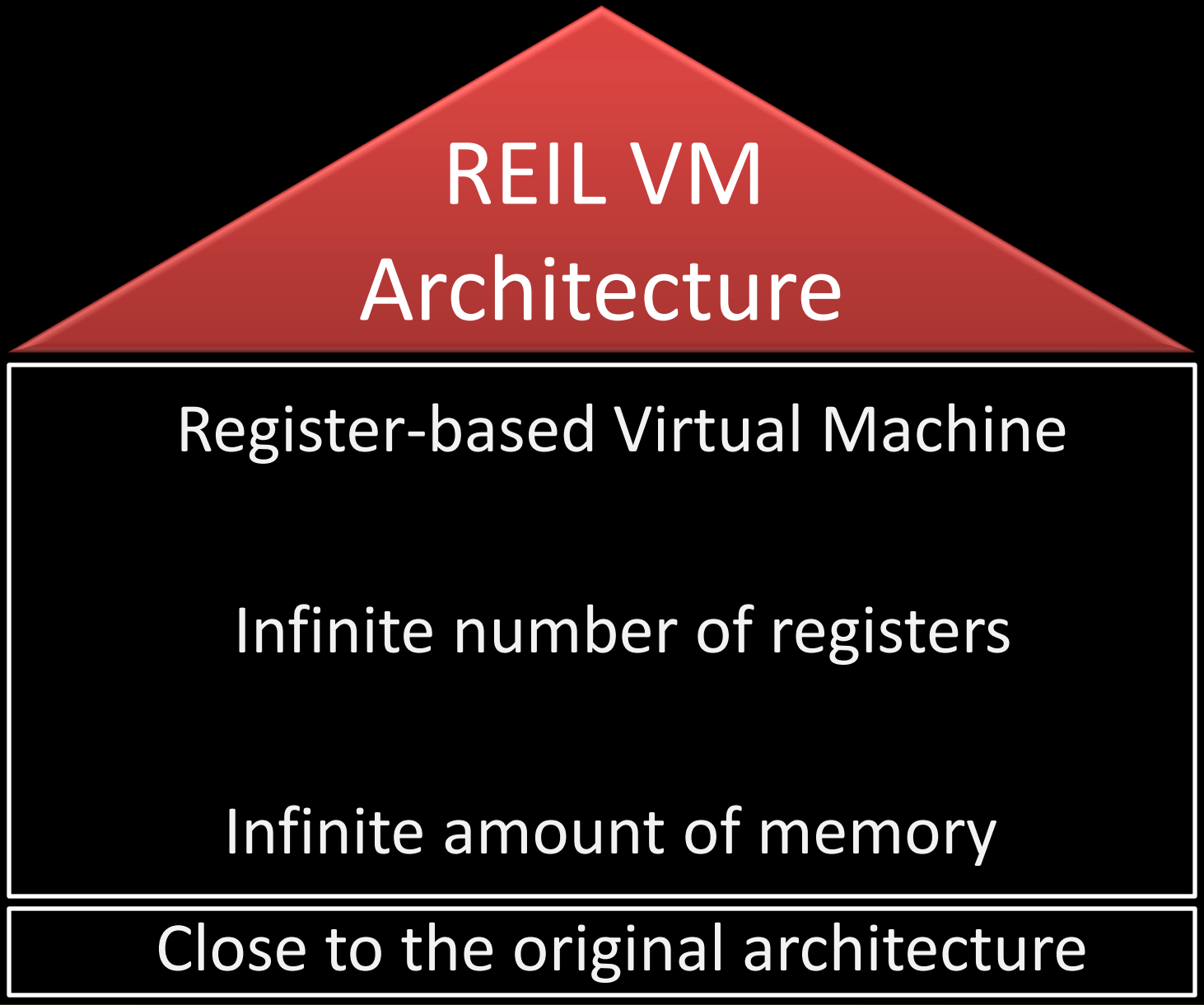
Three Good Reasons for REIL

- Simple
- Free of side-effects
- Platform-Independent

Fewer Instructions

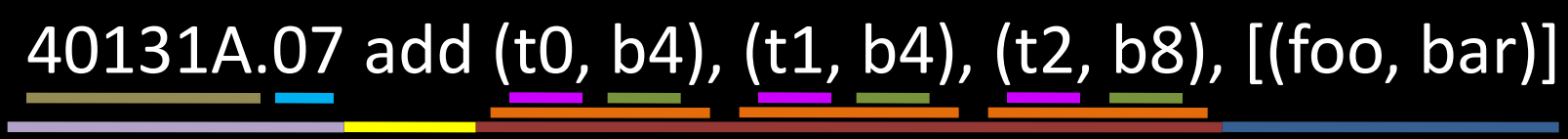



X86 Code
PowerPC Code
ARM Code
X86 Translator
PowerPC Translator
ARM Translator
REIL Code
MonoREIL
Other Analysis Algorithms


REIL VM
Architecture
Register-based Virtual Machine
Infinite number of registers
Infinite amount of memory
Close to the original architecture

17 Instructions

add
and
bisz
bsh
div
jcc
ldm
mod
mul
nop
or
stm
str
sub
undef
unkn
xor



1 REIL Address
- Native Part
- REIL Part

1 REIL Mnemonic
3 Operands
∞ Metadata

1 REIL Operand
2 (really 3) Pieces of Information
3 Operand Types
7 Operand Sizes
∞ Operand Values

Six Arithmetic Instructions

- Addition
- Logical Shift
- Unsigned Division
- Unsigned Modulo
- Unsigned Multiplication
- Subtraction

add
and
bisz
bsh
div
jcc
ldm
mod
mul
nop
or
stm
str
sub
undef
unkn
xor

Three Bitwise Instructions

- Bitwise AND
- Bitwise OR
- Bitwise XOR

add
and
bisz
bsh
div
jcc
ldm
mod
mul
nop
or
stm
str
sub
undef
unkn
xor

Three Data Transfer Instructions

- Load Memory
- Store Memory
- Transfer Register Content

add
and
bisz
bsh
div
jcc
ldm
mod
mul
nop
or
stm
str
sub
undef
unkn
xor

Two Logical Instructions

- Compare to Zero
- Jump Conditional

add
and
bisz
bsh
div
jcc
ldm
mod
mul
nop
or
stm
str
sub
undef
unkn
xor

Three Other Instructions

- No Operation
- Undefine Register
- Unknown Mnemonic

add
and
bisz
bsh
div
jcc
ldm
mod
mul
nop
or
stm
str
sub
undef
unkn
xor

All Instructions are equal ...

```
add t0, t1, t2
and t0, t1, t2
bsh t0, t1, t2
div t0, t1, t2
mod t0, t1, t2
mul t0, t1, t2
or  t0, t1, t2
sub t0, t1, t2
xor t0, t1, t2
```

... but some instructions are more equal than others

```
bisz  t0, , t2
jcc   t0, , t2
ldm   t0, , t2
nop     , ,
stm   t0, , t2
str   t0, , t2
undef   , , t2
unkn    , ,
```

What REIL can‘t do

FPU Instructions

fadd
fabbs
fdivp
frsp.
fnmsubs
…

System Instructions

int
sidt
bkpt
rfi
sc
…

Weird Instructions

setend
pld
mcrr
…

THE FUTURE!

More Architectures (x64, MIPS, ...)

More Operand Information

More Instructions

So …

… what is it all good for?

Register Tracking

Follow the effects
of a register
on the program state

Demo
Register Tracking

Negative Array Access

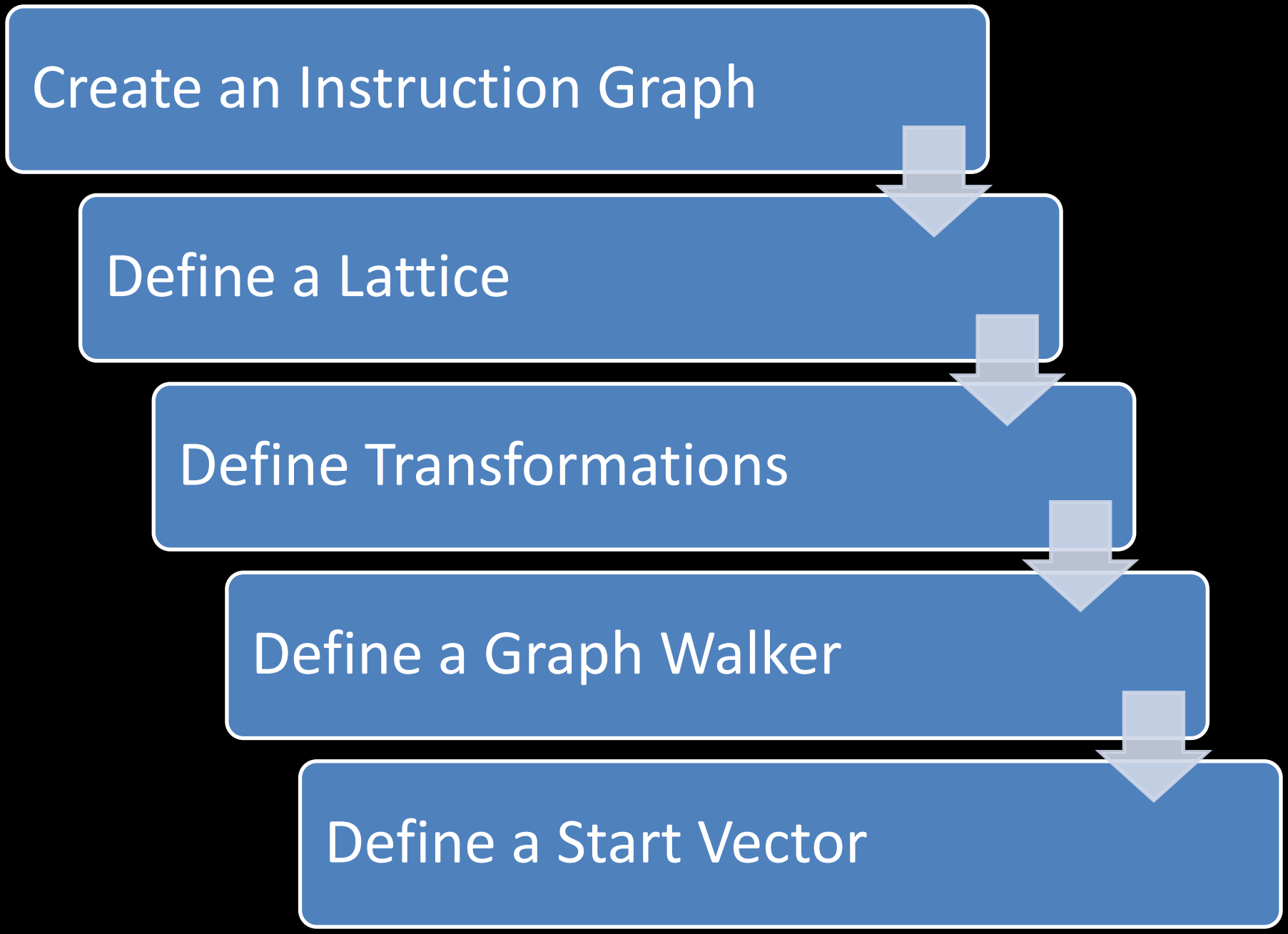
MS08-67

Demo
MS08-67

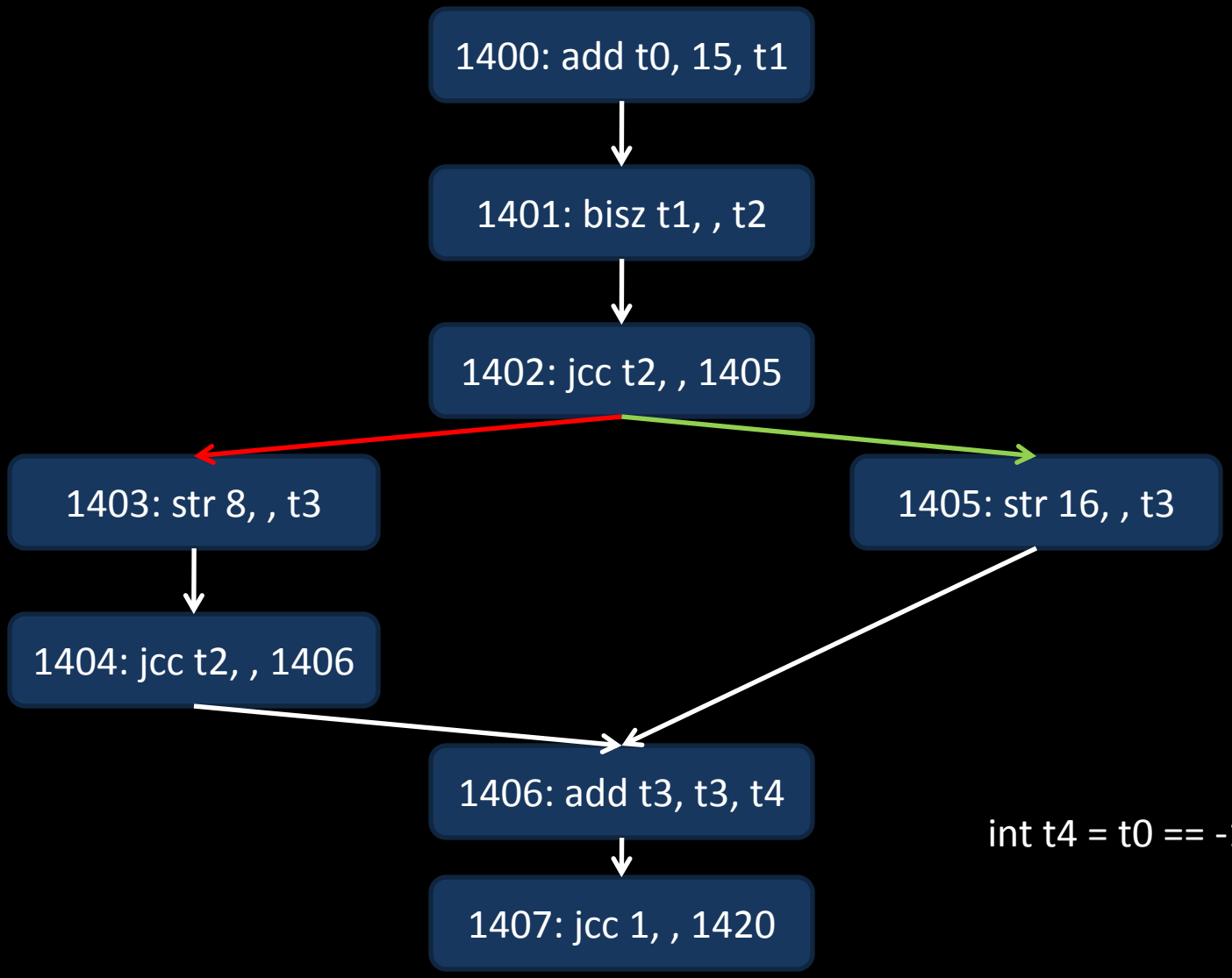
... but aren‘t those really difficult to implement?

NO!

MonoREIL

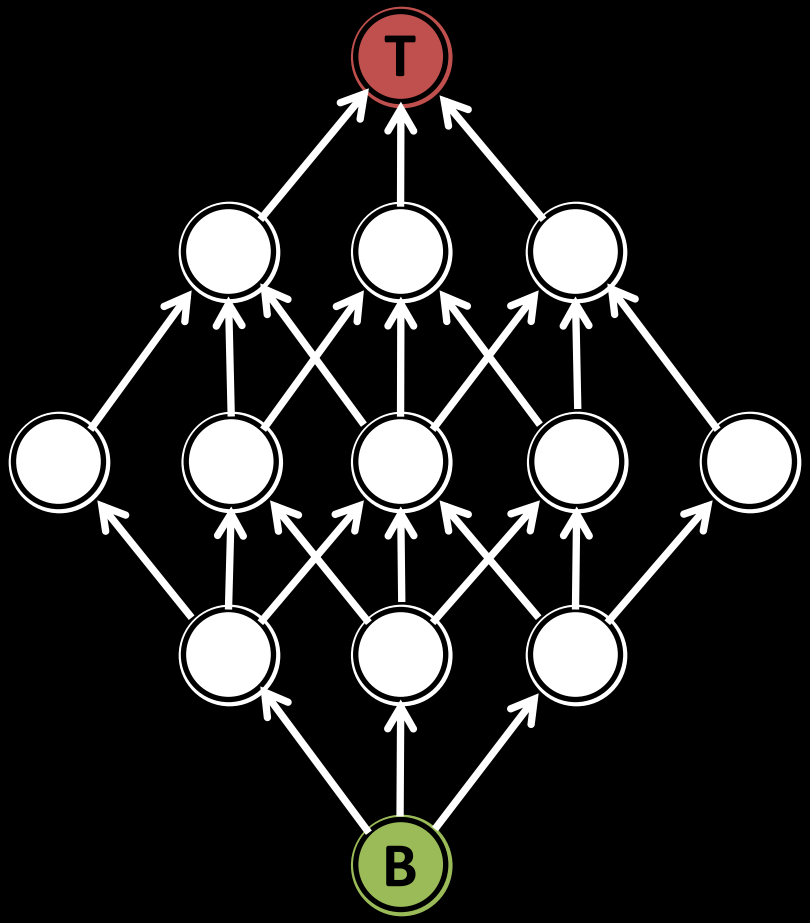

Create an Instruction Graph
Define a Lattice
Define Transformations
Define a Graph Walker
Define a Start Vector

Create an Instruction Graph

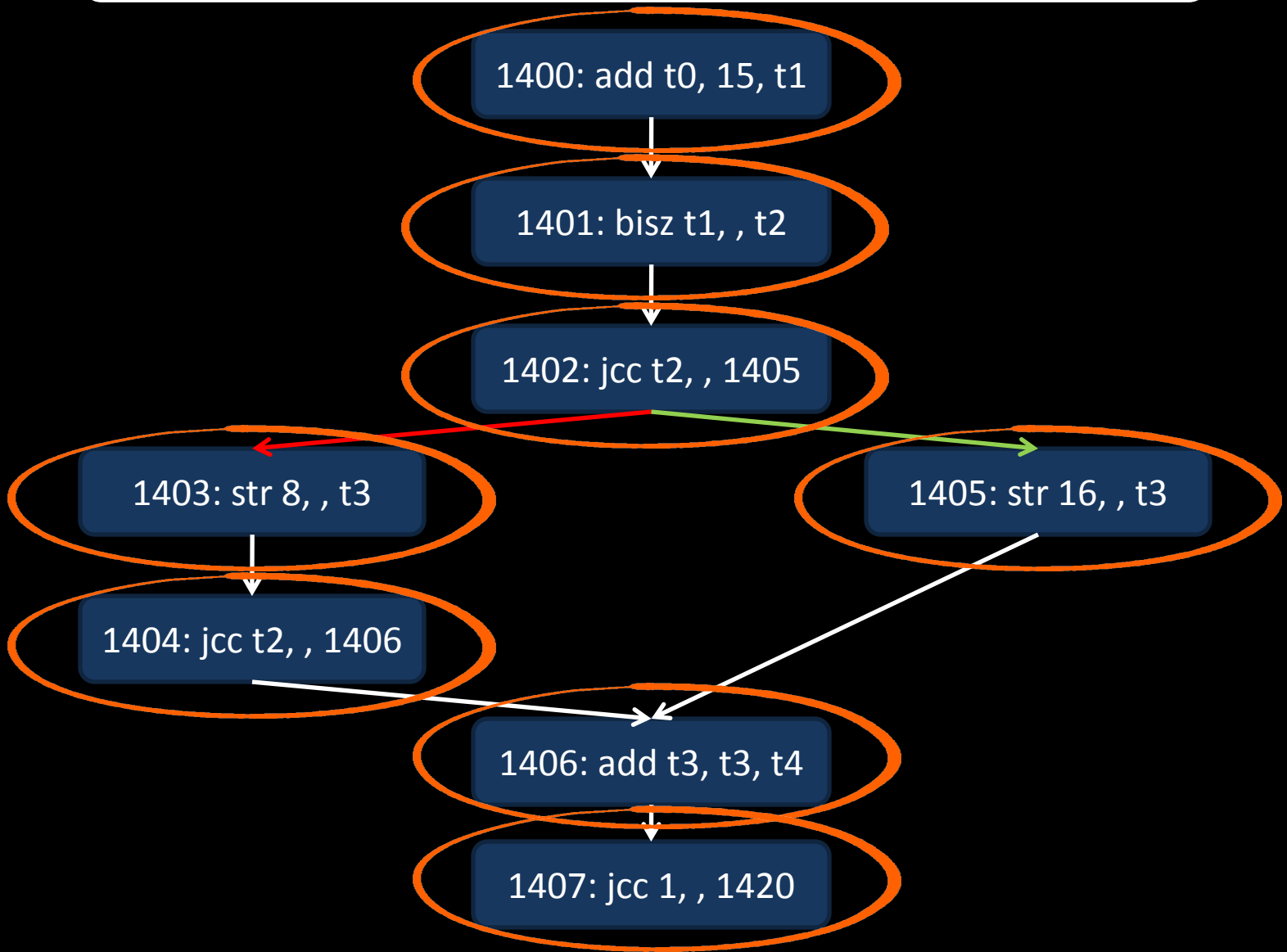


int t4 = t0 == -15 ? 32 : 16

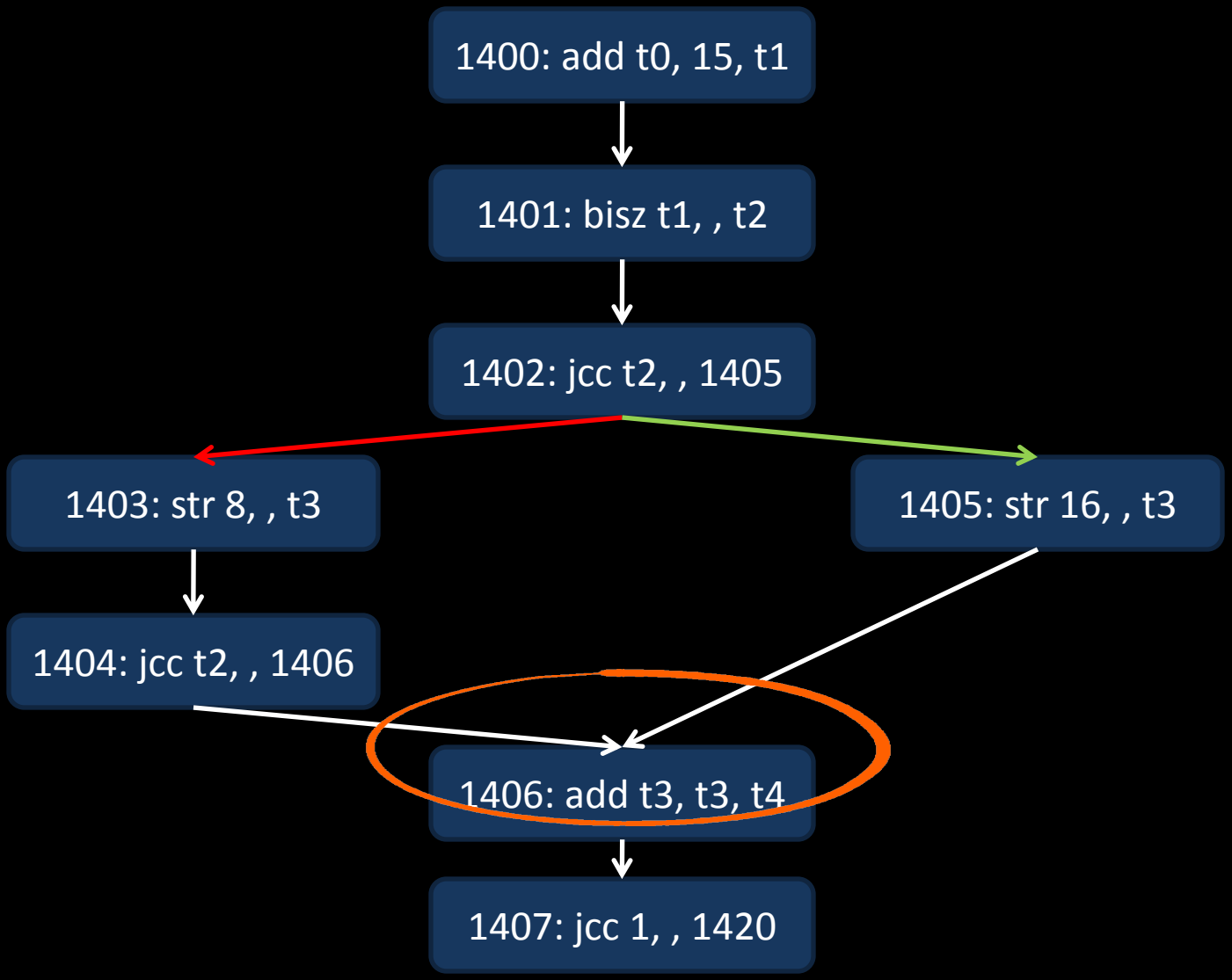
Define Lattice Elements



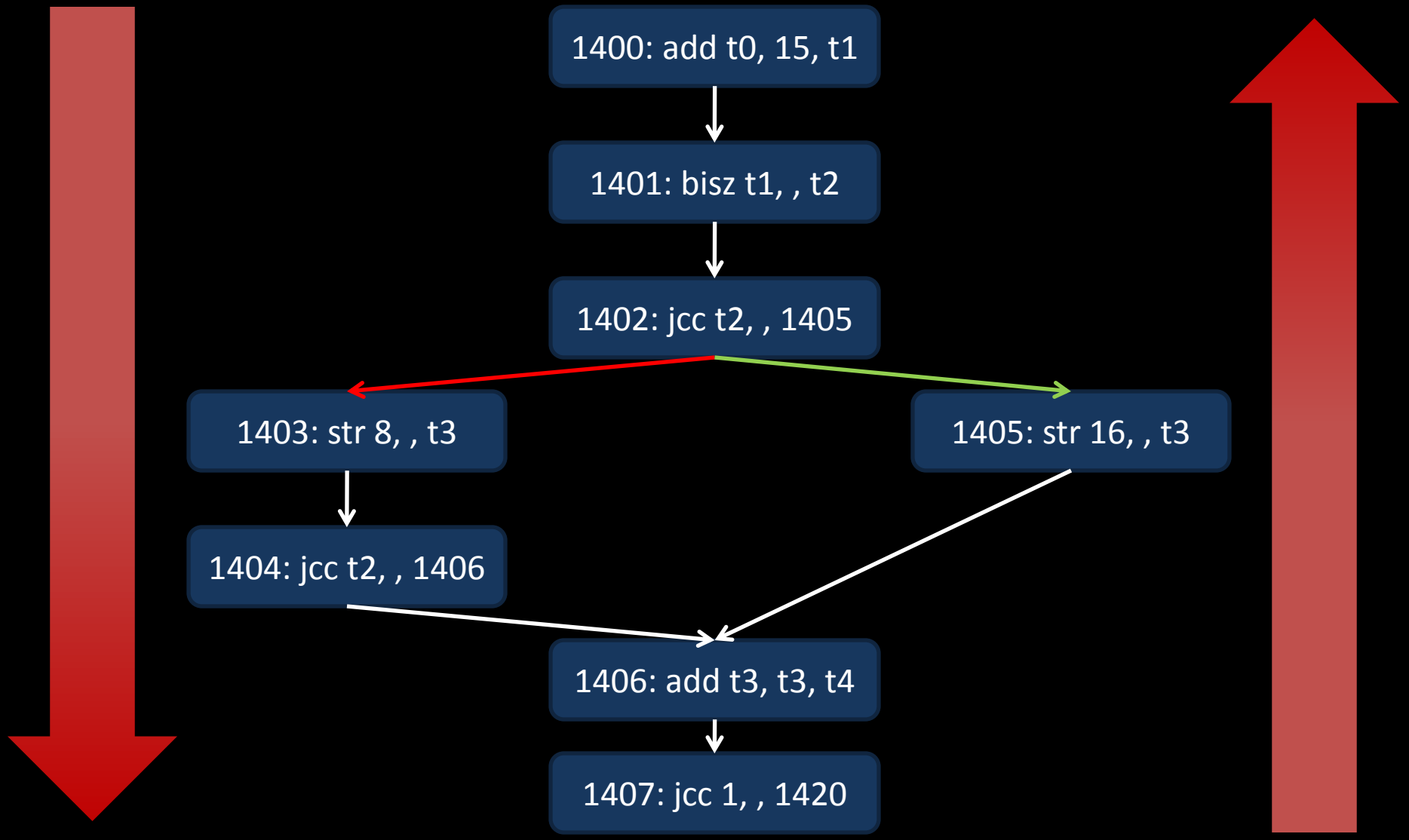
Define Transformations



Define a Join function



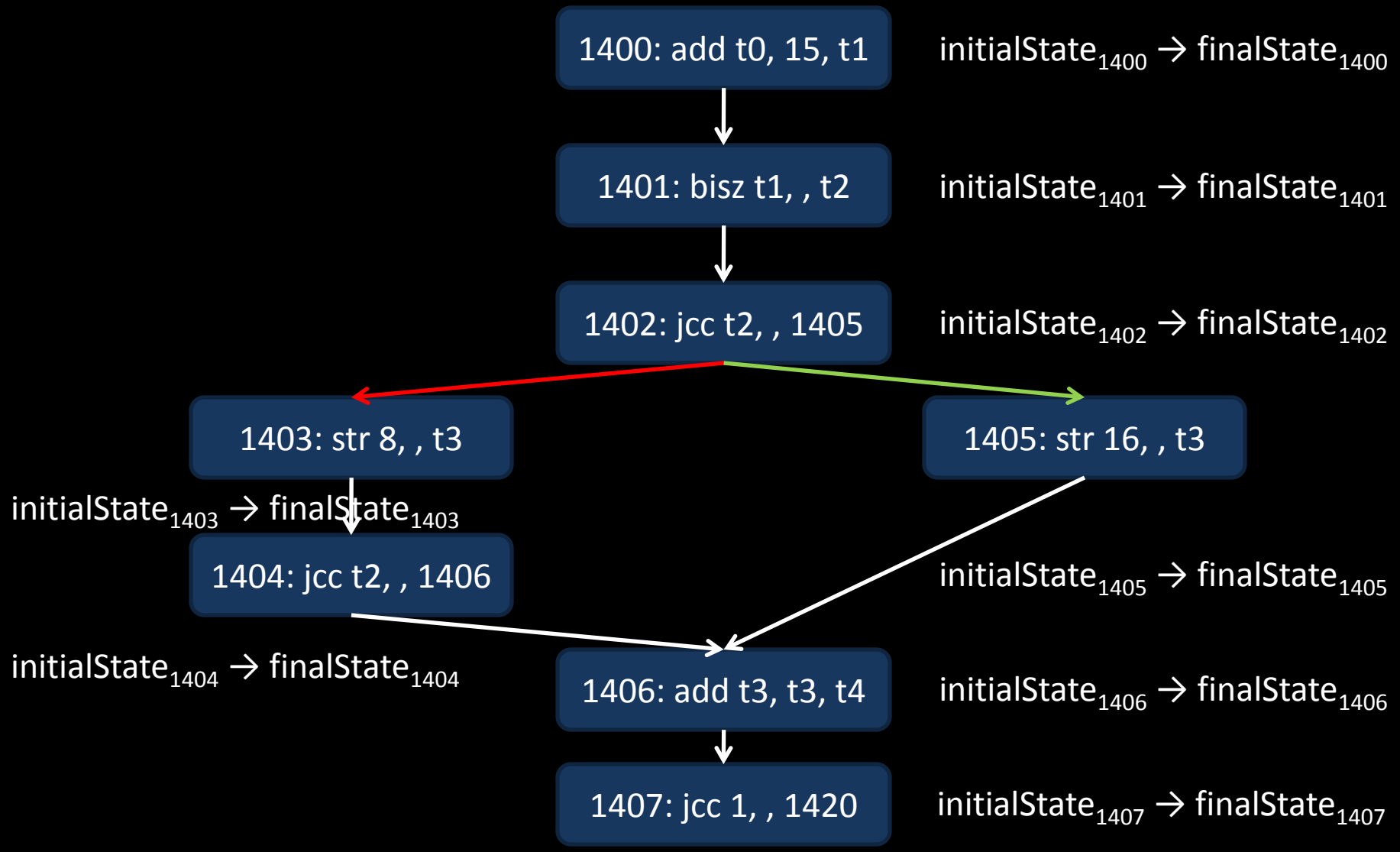
Define a Graph Walker



Define a Start Vector

$$
\begin{Bmatrix}
node_1 \rightarrow initialState_1 \\
node_2 \rightarrow initialState_2 \\
node_3 \rightarrow initialState_3 \\
node_4 \rightarrow initialState_4 \\
node_5 \rightarrow initialState_5 \\
\cdots \\
node_n \rightarrow initialState_n
\end{Bmatrix}
$$

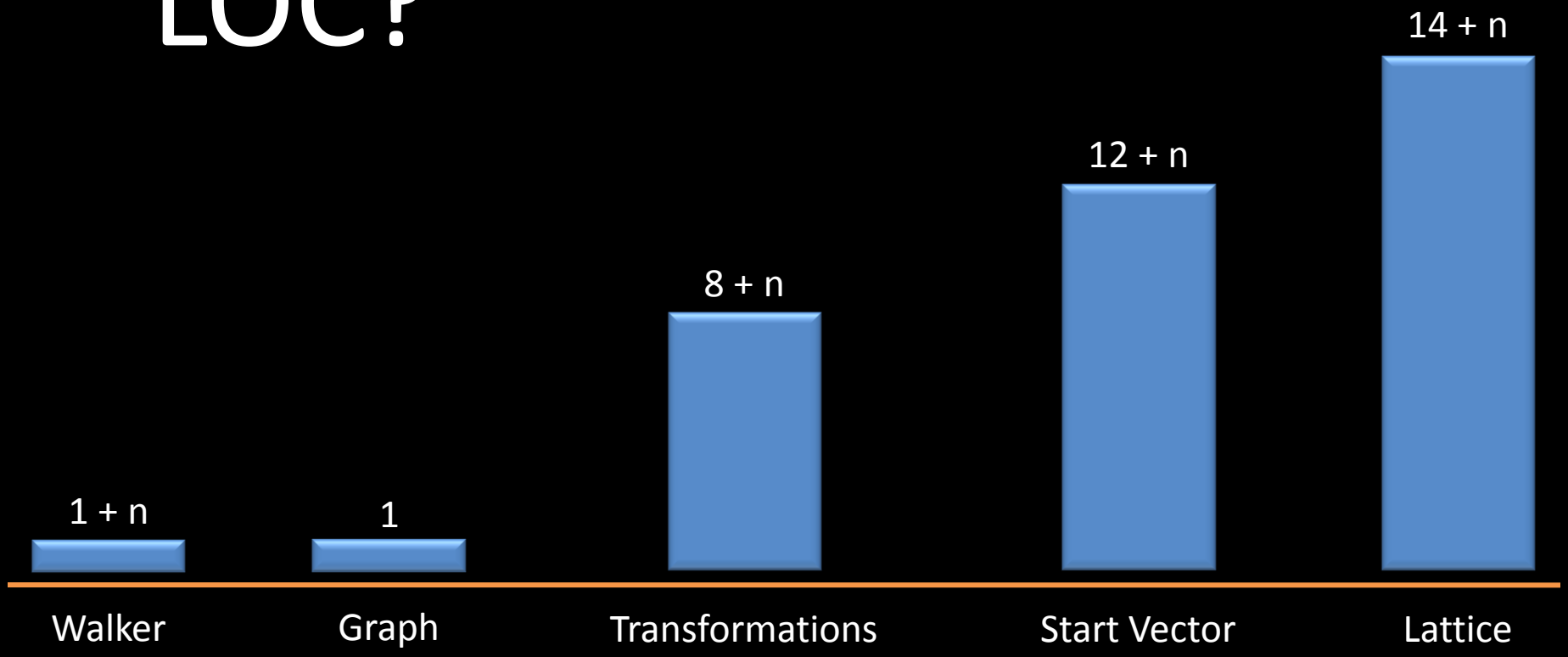
Running MonoREIL



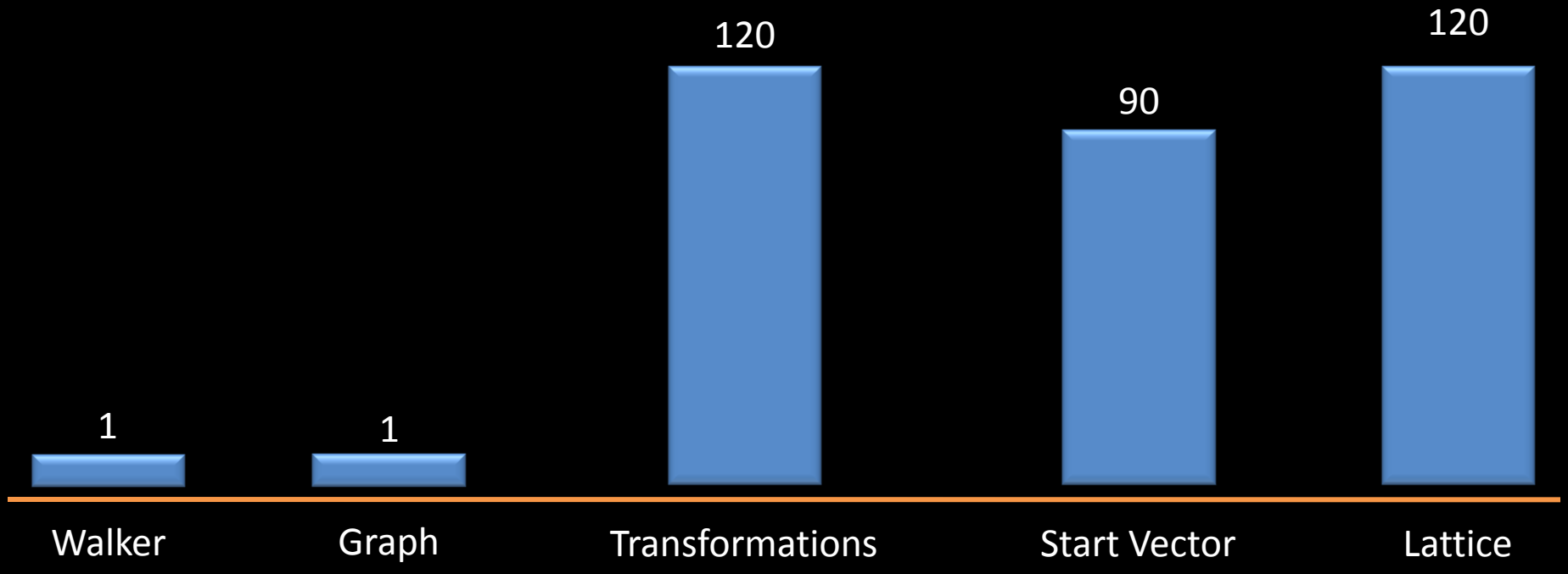
The Result

$$\left\{ \begin{array}{l} node_1 \rightarrow finalState_1 \\ node_2 \rightarrow finalState_2 \\ node_3 \rightarrow finalState_3 \\ node_4 \rightarrow finalState_4 \\ node_5 \rightarrow finalState_5 \\ \quad \ldots \\ node_n \rightarrow finalState_n \end{array} \right\}$$

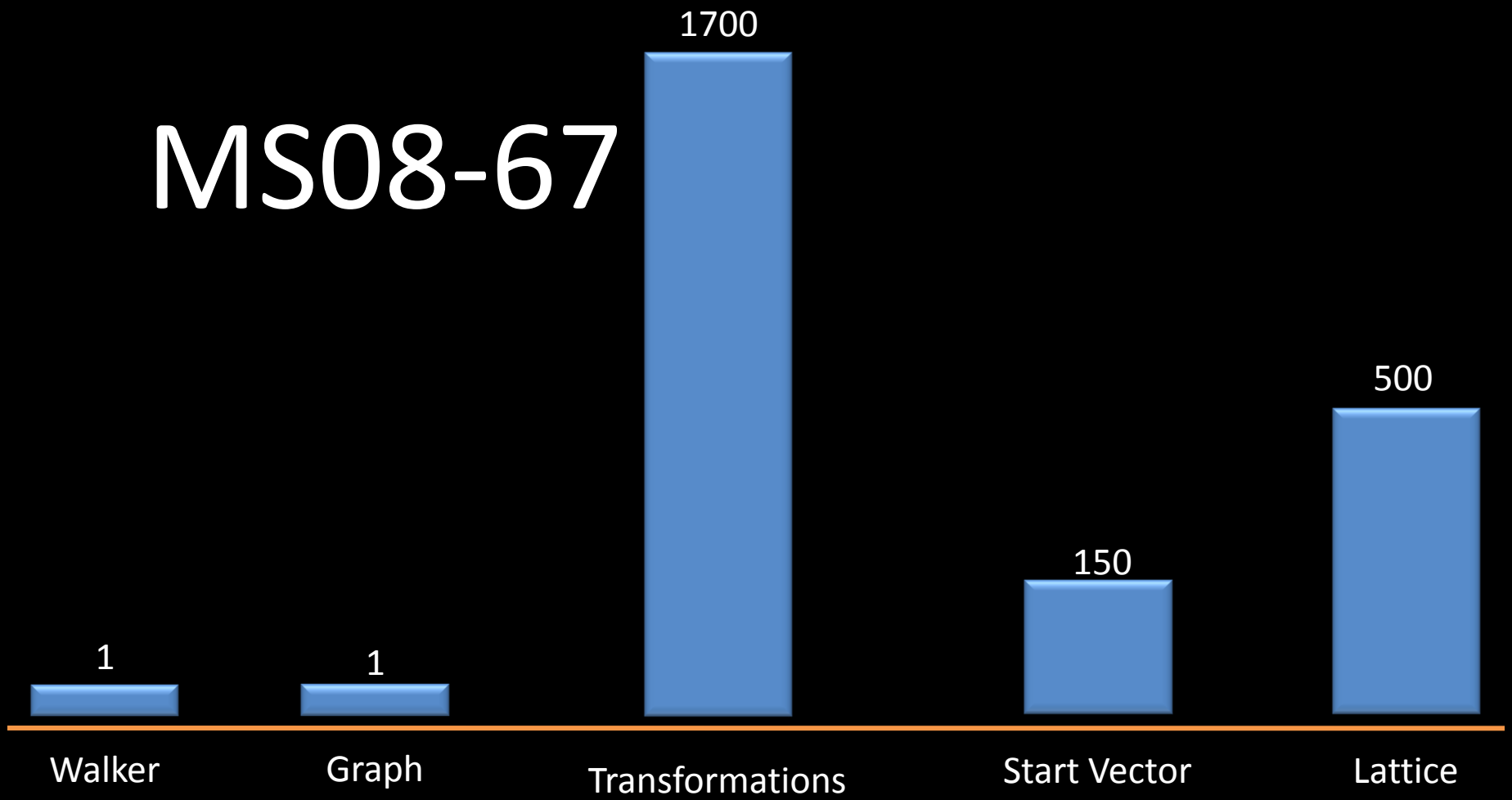
LOC?


1 + n
1
8 + n
12 + n
14 + n
Walker
Graph
Transformations
Start Vector
Lattice

Register Tracking



MS08-67


1700
500
150
1
1
Walker
Graph
Transformations
Start Vector
Lattice

THE FUTURE!

Deobfuscating Optimizer

Type Reconstruction

BinAudit

Related Work

ELIR (ERESI Project)

Vine (BitBlaze)

Silvio Cesare

Hex-Rays

Thank You!

Questions?