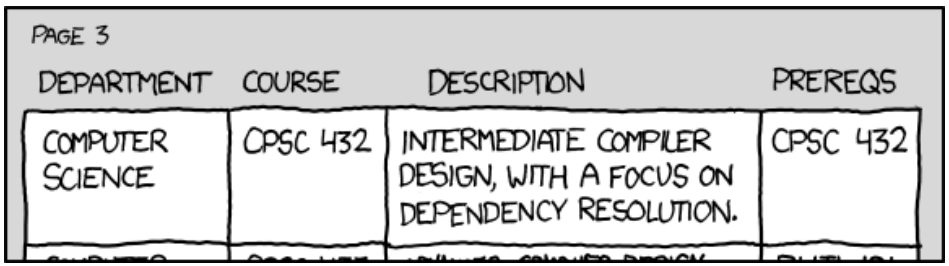
compiler optimization Global Value Numbering in Factor language

Alex Vondrak

ajvondrak@csupomona.edu

September 1, 2011

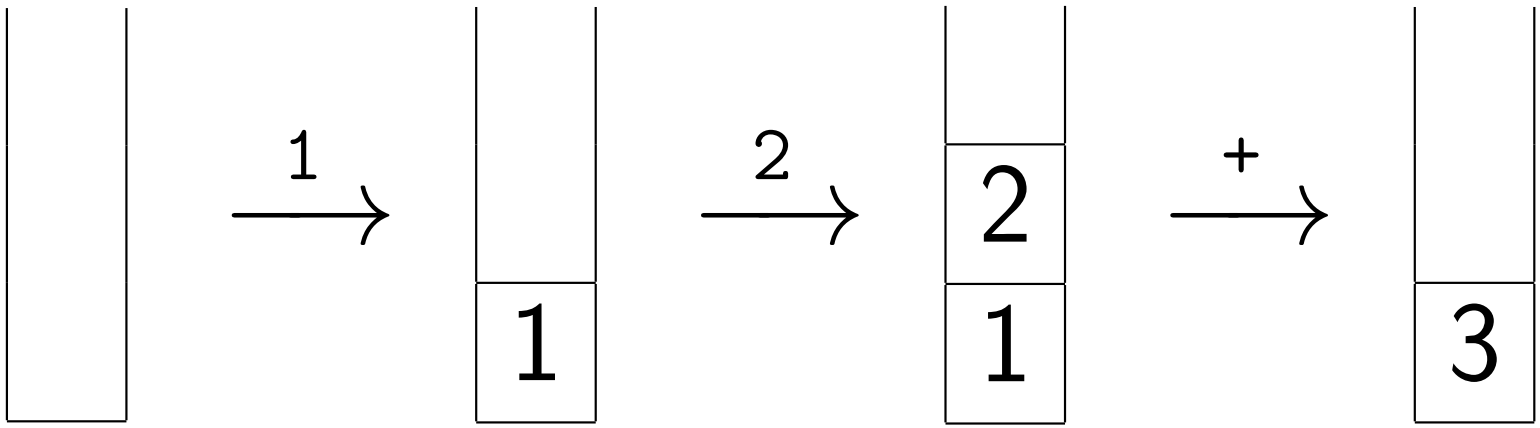


Factor

Factor (`http://factorcode.org/`)

- Started development September 2003—a baby among languages
- Stack-based

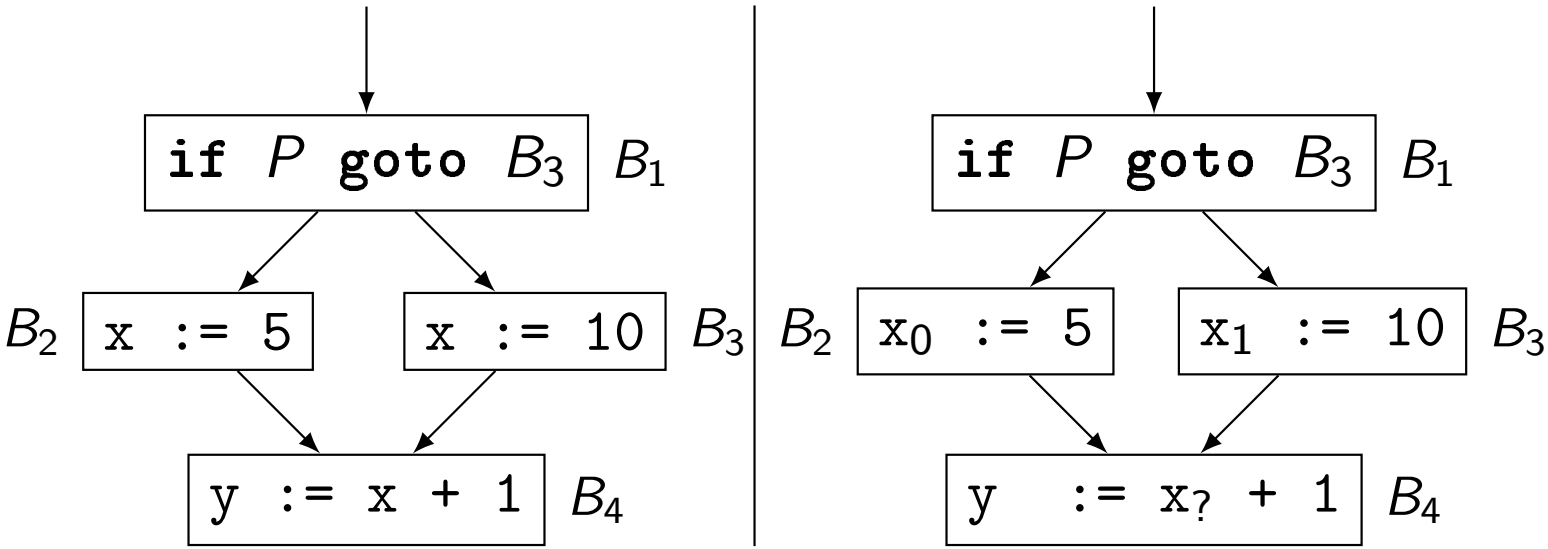
Example (1 2 +)



- High-level, object-oriented, dynamically typed, extensive libraries...
- ...yet fully compiled

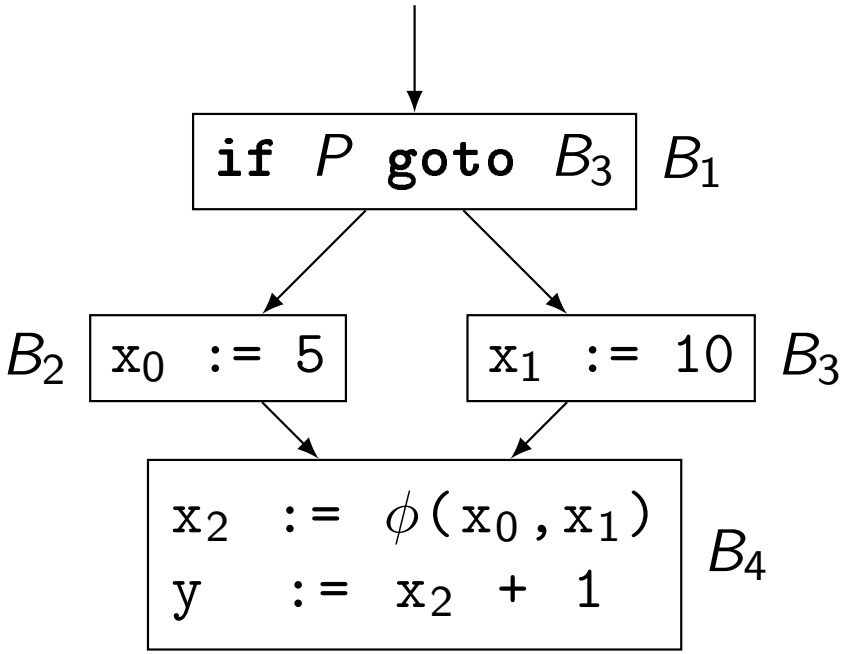
Low-level Representation

- Control flow graph (CFG)
  - Basic blocks = maximal sequence of "straight-line" code
  - Directed edges = transfer of control flow
- Static single assignment (SSA) form



Low-level Representation

- Control flow graph (CFG)
  - Basic blocks = maximal sequence of "straight-line" code
  - Directed edges = transfer of control flow
- Static single assignment (SSA) form



Low-level Representation

Instructions

- Code is translated into `insn` objects
- Modeled closely after typical assembly-like instructions
- Instructions are called on virtual registers

Example

In Factor syntax:

```
{
  T{ ##load-integer { dst 867 } { val 1 } }
  T{ ##load-integer { dst 5309 } { val 2 } }
  T{ ##add { dst 31337 } { src1 867 } { src2 5309 } }
}
```

Low-level Representation

Instructions

- Code is translated into `insn` objects
- Modeled closely after typical assembly-like instructions
- Instructions are called on virtual registers

Example

In diagrams:

```
##load-integer 867 1
##load-integer 5309 2
##add 31337 867 5309
```

Optimizations

```
: optimize-cfg ( cfg -- cfg' )
    optimize-tail-calls
    delete-useless-conditionals
    split-branches
    join-blocks
    normalize-height
    construct-ssa
    alias-analysis
    value-numbering
    copy-propagation
    eliminate-dead-code ;
```

Value Numbering

Idea: assign each virtual register a value number

- Equal value numbers $\implies$ equal at runtime
- Turn recomputations into ##copy instructions

Problem: equivalence is generally undecidable

- Seek conservative solution over Herbrand equivalences
- Consider two values congruent if
  - They're computed by the same operator
  - Their operands are congruent

Value Numbering

Implementation

- Expressions are constructed from instructions and value numbers

Example

Suppose...

- virtual register 2 has the value number 200
- virtual register 3 has the value number 300

Then

```
T{ ##add { dst 1 } { src1 2 } { src2 3 } } >expr
```

returns

```
{ ##add 200 300 }
```

Value Numbering

Implementation

- Expressions are constructed from instructions and value numbers
- Expression graph = 3 global hash tables
  - `vregs>vns`
  - `exprs>vns`
  - `vns>insns`
- If possible, instructions are simplified using data from expression graph

value-numbering

- Factor currently uses local value numbering
- Thought to be invented by Balke in the 1960s
- Largely credited to Cocke & Schwartz in the 1970s
- Pros:
  - Easy to understand
  - Easy to implement
  - Easy to extend
- Cons:
  - Locality and pessimism discover fewer congruences

Local Value Numbering

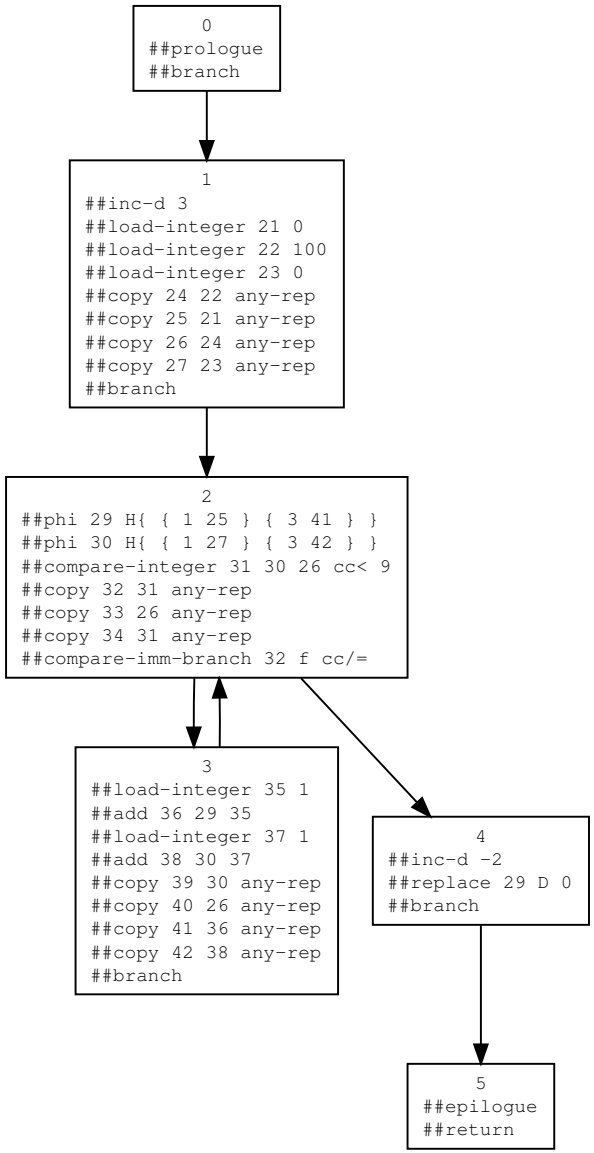
Example (In Factor)

```
0 100 [ 1 fixnum+fast ] times
```

Example (In Java)

```
int i = 0;
for (int j = 0; j < 100; j++) {
  i += 1;
}
```

Local Value Numbering



Local Value Numbering

Basic Block 1

```
vregs>vns = H{ }
exprs>vns = H{ }
```

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

(no-op)

Local Value Numbering

Basic Block 1

```
vregs>vns = H{ { 21 21 } }
exprs>vns = H{ {  0 21 } }
```

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

```
(no-op)
>expr = 0
```

Local Value Numbering

Basic Block 1

```
vregs>vns = H{ { 21 21 } {  22 22 } }
exprs>vns = H{ {  0 21 } { 100 22 } }
```

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

```
(no-op)
>expr = 0
>expr = 100
```

Local Value Numbering

Basic Block 1

```
vregs>vns = H{ { 21 21 } {  22 22 } { 23 21 } }
exprs>vns = H{ {  0 21 } { 100 22 } }
```

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

```
(no-op)
>expr = 0
>expr = 100
>expr = 0
```

Local Value Numbering

Basic Block 1

```
vregs>vns = H{ { 21 21 } {  22 22 } { 23 21 } { 24 22 } ... }
exprs>vns = H{ {  0 21 } { 100 22 } }
```

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##copy 23 21 any-rep
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

```
(no-op)
>expr = 0
>expr = 100
>expr = 0
...
...
...
...
```

Local Value Numbering

Basic Block 2

```
vregs>vns = H{ }
exprs>vns = H{ }
```

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

```
(no-op)
(no-op)
```

Local Value Numbering

Basic Block 2

```
vregs>vns = H{ { 30 30 } { 26 26 } { 31 31 } }
exprs>vns = H{ { { ##compare-integer 30 26 cc< } 31 } }
```

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

```
(no-op)
(no-op)
>expr = ...
```

Local Value Numbering

Basic Block 2

```
vregs>vns = H{ { 30 30 } { 26 26 } { 31 31 } { 32 31 } ... }
exprs>vns = H{ { { ##compare-integer 30 26 cc< } 31 } }
```

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

```
(no-op)
(no-op)
>expr = ...
...
...
...
```

Local Value Numbering

Basic Block 2

```
vregs>vns = H{ { 30 30 } { 26 26 } { 31 31 } { 32 31 } ... }
exprs>vns = H{ { { ##compare-integer 30 26 cc< } 31 } }
```

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-integer-branch 30 26 cc<
```

```
(no-op)
(no-op)
>expr = ...
...
...
...
```

Local Value Numbering

Basic Block 3

```
vregs>vns = H{ }
exprs>vns = H{ }
```

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

Local Value Numbering

Basic Block 3

```
vregs>vns = H{ { 35 35 } }
exprs>vns = H{ {  1 35 } }
```

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

```
>expr = 1
```

Local Value Numbering

Basic Block 3

```
vregs>vns = H{ { 35 35 } { 29 29 } { 36 36 } }
exprs>vns = H{ {  1 35 } { { ##add-imm 29 1 } 36 } }
```

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

```
>expr = 1
>expr = { ##add-imm 29 1 }
```

Local Value Numbering

Basic Block 3

```
vregs>vns = H{ { 35 35 } { 29 29 } { 36 36 }  { 37 35 } }
exprs>vns = H{ {  1 35 } { { ##add-imm 29 1 } 36 } }
```

```
##load-integer 35 1
##add-imm 36 29 1
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

```
>expr = 1
>expr = { ##add-imm 29 1 }
>expr = 1
```

Local Value Numbering

Basic Block 3

```
vregs>vns = H{ { 35 35 } { 29 29 } { 36 36 }  { 37 35 } ... }
exprs>vns = H{ {  1 35 } { { ##add-imm 29 1 } 36 } ... }
```

```
##load-integer 35 1
##add-imm 36 29 1
##copy 37 35 any-rep
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

```
>expr = 1
>expr = { ##add-imm 29 1 }
>expr = 1
>expr = { ##add-imm 30 1 }
```

Local Value Numbering

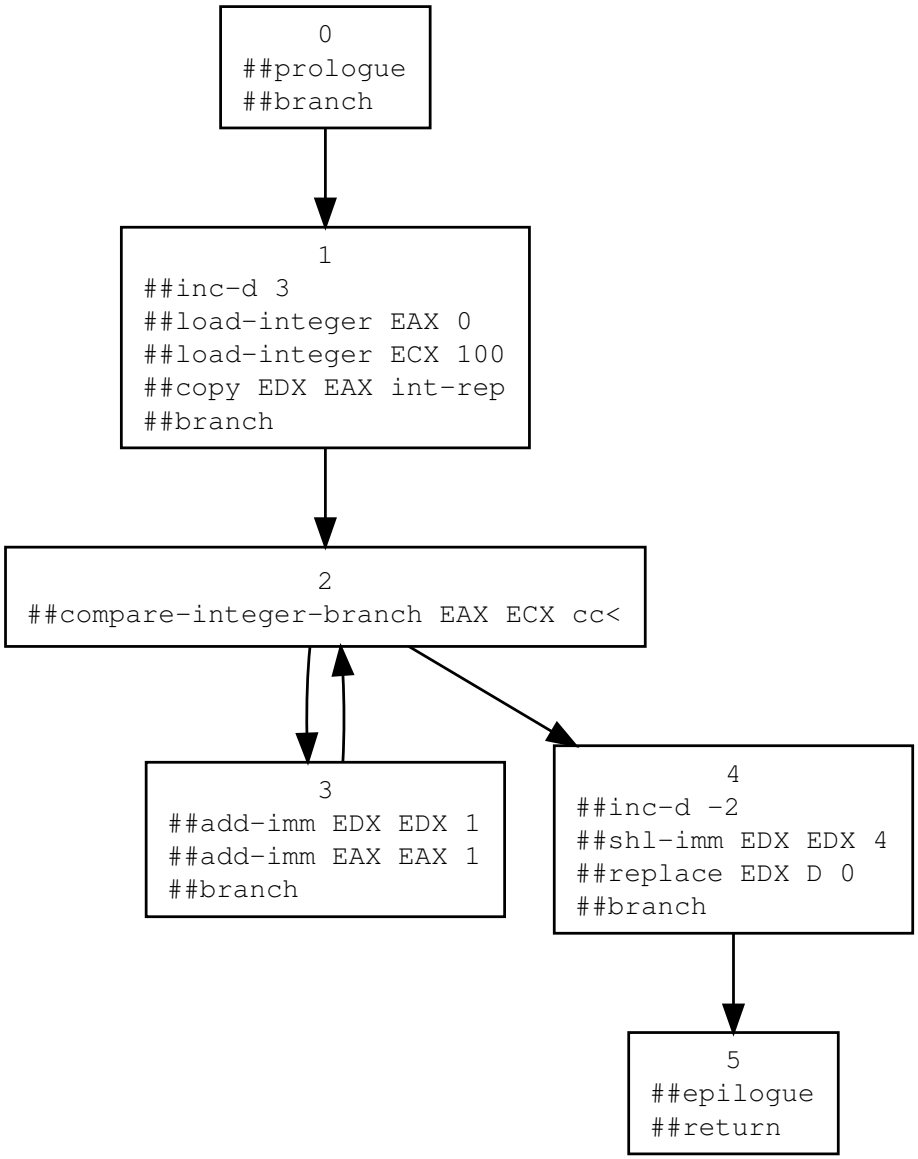
Basic Block 3

```
vregs>vns = H{ { 35 35 } { 29 29 } { 36 36 }  { 37 35 } ... }
exprs>vns = H{ {  1 35 } { { ##add-imm 29 1 } 36 } ... }
```

```
##load-integer 35 1
##add-imm 36 29 1
##copy 37 35 any-rep
##add-imm 38 30 1
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

```
>expr = 1
>expr = { ##add-imm 29 1 }
>expr = 1
>expr = { ##add-imm 30 1 }
...
...
...
...
```

Local Value Numbering Results



Same Example, Global Algorithm

Changes:

- Global—don't wipe out hash tables after each block
- Optimistic—previously unseen values considered redundant
- Fixed point iteration—optimism may be wrong first time around
- Offline replacements—only on final pass

Definition (Notation)

$$\langle \mathtt{n} \rangle = \{\mathtt{x}, \mathtt{y}, \mathtt{z}\} \qquad (\mathtt{expr})$$

versus

```
vregs>vns = H{ { x n } { y n } { z n } }
exprs>vns = H{ { expr n } }
```

Global Value Numbering

Iteration 1, Basic Block 1

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

$\langle \mathtt{f} \rangle = U$ (everything)

Global Value Numbering

Iteration 1, Basic Block 1

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

$$\langle 21 \rangle = \{21\} \tag{0}$$

Global Value Numbering

Iteration 1, Basic Block 1

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

$$\langle 21 \rangle = \{21\} \qquad (0)$$
$$\langle 22 \rangle = \{22\} \qquad (100)$$

Global Value Numbering

Iteration 1, Basic Block 1

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23\} \quad (0)$$

$$\langle 22 \rangle = \{22\} \quad (100)$$

Global Value Numbering

Iteration 1, Basic Block 1

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26\} \quad (100)$$

Global Value Numbering

Iteration 1, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26\} \quad (100)$$

Global Value Numbering

Iteration 1, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26\} \quad (100)$$

Global Value Numbering

Iteration 1, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26\} \quad (100)$$

Global Value Numbering

Iteration 1, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26\} \quad (100)$$

$$\langle 31 \rangle = \{31\} \quad (\mathtt{t})$$

Global Value Numbering

Iteration 1, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33\} \quad (100)$$

$$\langle 31 \rangle = \{31, 32, 34\} \quad (\texttt{t})$$

Global Value Numbering

Iteration 1, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33\} \quad (100)$$

$$\langle 31 \rangle = \{31, 32, 34\} \quad (\mathtt{t})$$

Global Value Numbering

Iteration 1, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30\} \quad (0)$$
$$\langle 22 \rangle = \{22, 24, 26, 33\} \quad (100)$$
$$\langle 31 \rangle = \{31, 32, 34\} \quad (\mathtt{t})$$
$$\langle 35 \rangle = \{35\} \quad (1)$$

Global Value Numbering

Iteration 1, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30\} \quad (0)$$
$$\langle 22 \rangle = \{22, 24, 26, 33\} \quad (100)$$
$$\langle 31 \rangle = \{31, 32, 34\} \quad (\mathtt{t})$$
$$\langle 35 \rangle = \{35, 36\} \quad (1)$$

Global Value Numbering

Iteration 1, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33\} \quad (100)$$

$$\langle 31 \rangle = \{31, 32, 34\} \quad (\mathtt{t})$$

$$\langle 35 \rangle = \{35, 36, 37\} \quad (1)$$

Global Value Numbering

Iteration 1, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33\} \quad (100)$$

$$\langle 31 \rangle = \{31, 32, 34\} \quad (\mathtt{t})$$

$$\langle 35 \rangle = \{35, 36, 37, 38\} \quad (1)$$

Global Value Numbering

Iteration 1, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30, 39\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$

$$\langle 31 \rangle = \{31, 32, 34\} \quad (\mathtt{t})$$

$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (1)$$

Global Value Numbering

Iteration 2, Basic Block 1

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30, 39\} \quad (—)$$
$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (—)$$
$$\langle 31 \rangle = \{31, 32, 34\} \quad (—)$$
$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (—)$$

Global Value Numbering

Iteration 2, Basic Block 1

```
##inc-d 3
##load-integer 21 0
##load-integer 22 100
##load-integer 23 0
##copy 24 22 any-rep
##copy 25 21 any-rep
##copy 26 24 any-rep
##copy 27 23 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30, 39\} \quad (0)$$
$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$
$$\langle 31 \rangle = \{31, 32, 34\} \quad (—)$$
$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (—)$$

Global Value Numbering

Iteration 2, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 29, 30, 39\} \quad (0)$$
$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$
$$\langle 31 \rangle = \{31, 32, 34\} \quad (—)$$
$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (—)$$

Global Value Numbering

Iteration 2, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 30, 39\} \quad (0)$$
$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$
$$\langle 29 \rangle = \{29\} \quad (\texttt{\#\#phi 2 21 35})$$
$$\langle 31 \rangle = \{31, 32, 34\} \quad (—)$$
$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (—)$$

Global Value Numbering

Iteration 2, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 39\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$

$$\langle 29 \rangle = \{29, 30\} \quad (\texttt{\#\#phi 2 21 35})$$

$$\langle 31 \rangle = \{31, 32, 34\} \quad (—)$$

$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (—)$$

Global Value Numbering

Iteration 2, Basic Block 2

```
##phi 29 H{ { 1 25 } { 3 41 } }
##phi 30 H{ { 1 27 } { 3 42 } }
##compare-integer 31 30 26 cc< 9
##copy 32 31 any-rep
##copy 33 26 any-rep
##copy 34 31 any-rep
##compare-imm-branch 32 f cc/=
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 39\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$

$$\langle 29 \rangle = \{29, 30\} \quad (\texttt{\#\#phi 2 21 35})$$

$$\langle 31 \rangle = \{31, 32, 34\}$$

$$(\texttt{\#\#compare-integer-imm 29 100 cc<})$$

$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (—)$$

Global Value Numbering

Iteration 2, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 39\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$

$$\langle 29 \rangle = \{29, 30\} \quad (\texttt{\#\#phi 2 21 35})$$

$$\langle 31 \rangle = \{31, 32, 34\}$$

$$(\texttt{\#\#compare-integer-imm 29 100 cc<})$$

$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (—)$$

Global Value Numbering

Iteration 2, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 39\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$

$$\langle 29 \rangle = \{29, 30\} \quad (\texttt{\#\#phi 2 21 35})$$

$$\langle 31 \rangle = \{31, 32, 34\}$$

$$(\texttt{\#\#compare-integer-imm 29 100 cc<})$$

$$\langle 35 \rangle = \{35, 36, 37, 38, 41, 42\} \quad (1)$$

Global Value Numbering

Iteration 2, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 39\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$

$$\langle 29 \rangle = \{29, 30\} \quad (\texttt{\#\#phi 2 21 35})$$

$$\langle 31 \rangle = \{31, 32, 34\}$$

$$(\texttt{\#\#compare-integer-imm 29 100 cc<})$$

$$\langle 35 \rangle = \{35, 37, 38, 41, 42\} \quad (1)$$

$$\langle 36 \rangle = \{36\} \quad (\texttt{\#\#add-imm 29 1})$$

Global Value Numbering

Iteration 2, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 39\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$

$$\langle 29 \rangle = \{29, 30\} \quad (\texttt{\#\#phi 2 21 35})$$

$$\langle 31 \rangle = \{31, 32, 34\}$$

$$(\texttt{\#\#compare-integer-imm 29 100 cc<})$$

$$\langle 35 \rangle = \{35, 37, 38, 41, 42\} \quad (1)$$

$$\langle 36 \rangle = \{36\} \quad (\texttt{\#\#add-imm 29 1})$$

Global Value Numbering

Iteration 2, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$$\langle 21 \rangle = \{21, 23, 25, 27, 39\} \quad (0)$$

$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$

$$\langle 29 \rangle = \{29, 30\} \quad (\texttt{\#\#phi 2 21 35})$$

$$\langle 31 \rangle = \{31, 32, 34\}$$

$$(\texttt{\#\#compare-integer-imm 29 100 cc<})$$

$$\langle 35 \rangle = \{35, 37, 41, 42\} \quad (1)$$

$$\langle 36 \rangle = \{36, 38\} \quad (\texttt{\#\#add-imm 29 1})$$

Global Value Numbering

Iteration 2, Basic Block 3

```
##load-integer 35 1
##add 36 29 35
##load-integer 37 1
##add 38 30 37
##copy 39 30 any-rep
##copy 40 26 any-rep
##copy 41 36 any-rep
##copy 42 38 any-rep
##branch
```

$\langle 21 \rangle = \{21, 23, 25, 27\}$ (0)

$\langle 22 \rangle = \{22, 24, 26, 33, 40\}$ (100)

$\langle 29 \rangle = \{29, 30, 39\}$ (##phi 2 21 35)

$\langle 31 \rangle = \{31, 32, 34\}$
(##compare-integer-imm 29 100 cc<)

$\langle 35 \rangle = \{35, 37\}$ (1)

$\langle 36 \rangle = \{36, 38, 41, 42\}$ (##add-imm 29 1)

Global Value Numbering

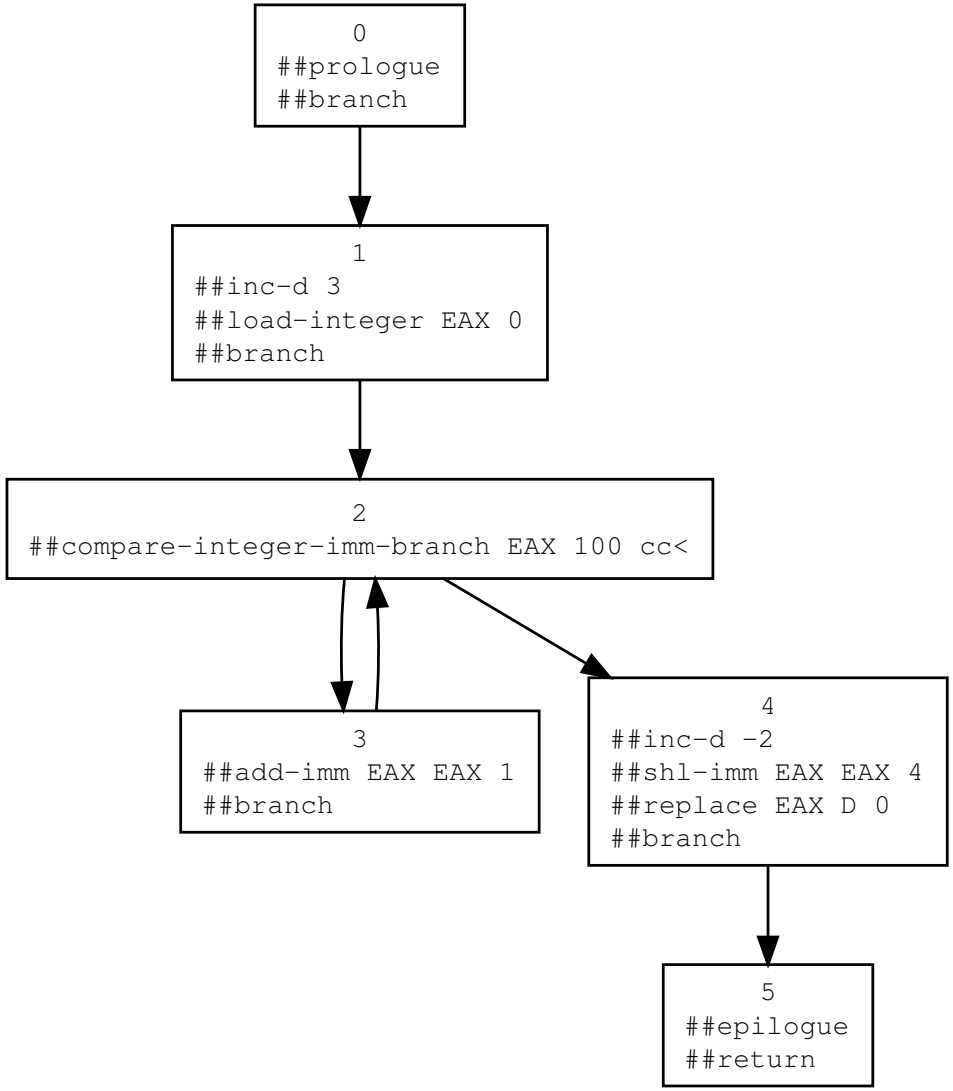
Iteration 3

No value numbers change, only the expressions:

$$\langle 21 \rangle = \{21, 23, 25, 27\} \quad (0)$$
$$\langle 22 \rangle = \{22, 24, 26, 33, 40\} \quad (100)$$
$$\langle 29 \rangle = \{29, 30, 39\} \quad (\texttt{\#\#phi 2 21 36})$$
$$\langle 31 \rangle = \{31, 32, 34\} \quad (\texttt{\#\#compare-integer-imm 29 100 cc<})$$
$$\langle 35 \rangle = \{35, 37\} \quad (1)$$
$$\langle 36 \rangle = \{36, 38, 41, 42\} \quad (\texttt{\#\#add-imm 29 1})$$

Final pass simplifies instructions based on this information

Global Value Numbering Results



Deliverables

In total:

- Completely new Graphviz bindings
  - Officially merged into Factor a few days ago!
- New library to output CFG diagrams
- Altered value numbering pass to make it global
  - Passes same unit tests as before
  - Informal benchmark improvements (mean -16.35%)
  - Many future improvements possible