OSACA

Open Source Architecture Code Analyzer

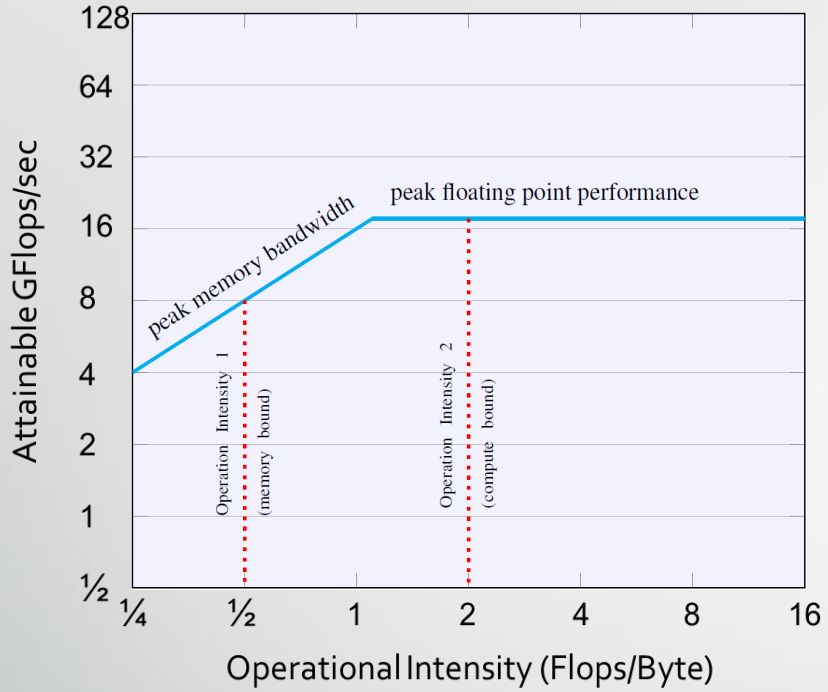
Jan Laukemann

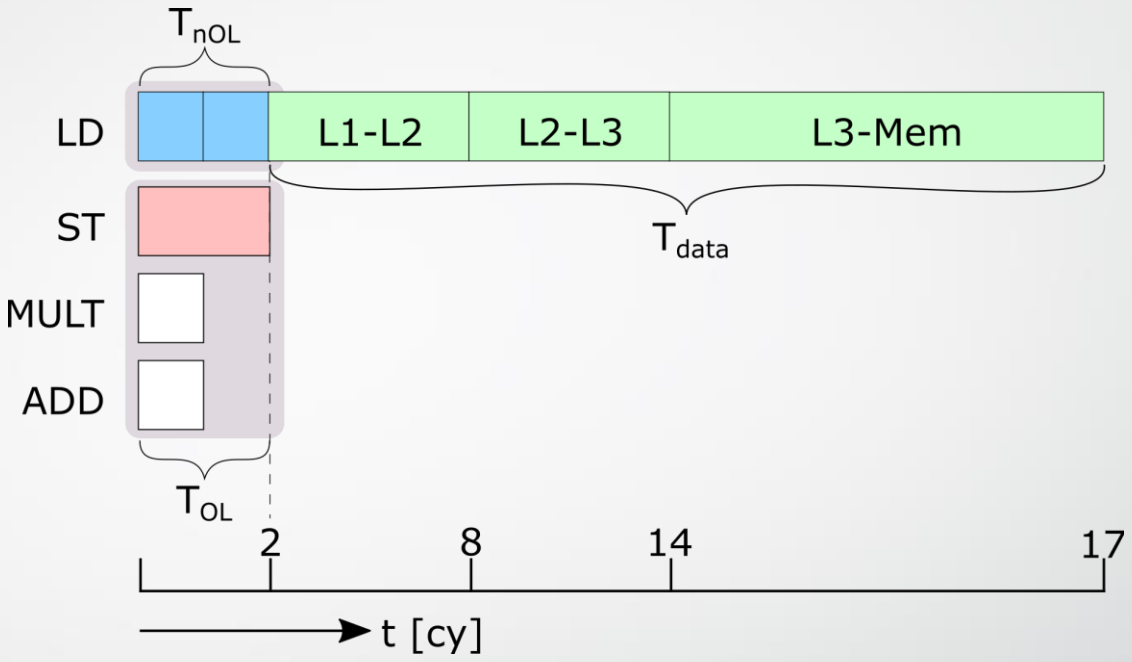
Design and Implementation of a
Framework for Predicting Instruction Throughput

Agenda

- Motivation
- IACA as role model
- OSACA: design and functionality
- Examples
- Conclusion / Future Challenges

Motivation





- Performance model for optimization
  → concentrate on loop body
- Analysis of source code depending on microarchitecture
- **Incore prediction** [cycles] for more complex performance models necessary

IACA

Intel Architecture Code Analyzer

- Analyzes maximum throughput assuming optimal execution conditions (All memory accesses hit L1 cache, no page faults)
- Supports Intel 64 code for architectures from Sandybridge to Skylake X

```
// C or C++ usage of IACA

#include "iacaMarks.h"


while(cond){
    IACA_START
    // Loop body
    // ...
}
IACA_END
```

```
; ASM usage of IACA
    movl    $111, %ebx
    .byte   100, 103, 144 ; Start marker

.innermostlooplabel:
    ; Loop body
    ; ...
    jb .innermostlooplabel ; conditional branch

    movl    $222, %ebx
    .byte   100, 103, 144 ; End marker
```

Need to compile into object code for IACA

IACA

```
Intel(R) Architecture Code Analyzer Version - 2.3 build:246dfea (Thu, 6 Jul 2017 13:38:05 +0300)
Analyzed File - helloIaca
Binary Format - 64Bit
Architecture  - IVB
Analysis Type - Throughput

Throughput Analysis Report
--------------------------
Block Throughput: 1.90 Cycles       Throughput Bottleneck: FrontEnd

Port Binding In Cycles Per Iteration:
---------------------------------------------------------------------------
|  Port  |  0   -  DV  |  1   |  2   -  D   |  3   -  D   |  4   |  5   |
---------------------------------------------------------------------------
| Cycles | 1.0    0.0  | 1.0  | 1.0    0.5  | 1.0    0.5  | 1.0  | 1.9  |
---------------------------------------------------------------------------

N - port number or number of cycles resource conflict caused delay, DV - Divider pipe (on port 0)
D - Data fetch pipe (on ports 2 and 3), CP - on a critical path
F - Macro Fusion with the previous instruction occurred
* - instruction micro-ops not bound to a port
^ - Micro Fusion happened
# - ESP Tracking sync uop was issued
@ - SSE instruction followed an AVX256/AVX512 instruction, dozens of cycles penalty is expected
X - instruction not supported, was not accounted in Analysis
| Num Of |              Ports pressure in cycles               |    |
|  Uops  |  0  - DV  |  1  |  2  -  D  |  3  -  D  |  4  |  5  |    |
---------------------------------------------------------------------
|   2    |           | 1.0 | 0.5   0.5 | 0.5   0.5 |     |     |    | vaddss xmm3, xmm2, dword ptr [rsp+rax*4-0x4]
|   2    |           |     | 0.5       | 0.5       | 1.0 |     |    | vmovss dword ptr [rsp+rax*4], xmm3
|   1    | 0.9       |     |           |           |     | 0.1 | CP | inc rax
|   1    |           |     |           |           |     | 1.0 | CP | cmp rax, 0x3e8
|   0F   |           |     |           |           |     |     |    | jl 0xffffffffffffffd8
|   1    | 0.1       | 0.1 |           |           |     | 0.9 | CP | mov edi, 0x401d58
Total Num Of Uops: 7
```

Why OSACA?

- Open Source
- Future development path of IACA unclear
- Based on benchmarks of individual instructions
- Easy setting of markers in high level source code
- Provides whole toolchain of Instruction fetching, benchmarking and throughput analysis
- Perspective of
  - Analyses on non Intel architectures
  - Latency analyses (Critical Path, inter loop dependencies)

OSACA

- Analyzes average throughput assuming optimal execution conditions
  - all memory accesses hit L1 cache, no page faults, steady-state, all instructions in cache
- Currently supports Intel architectures from Sandybridge to Skylake X

No need for compiling assembly!

```
//usage of OSACA marker



//STARTLOOP
while(cond){
    // Loop body
    // ...
}
```

```
; ASM usage of IACA byte markers for OSACA
    movl    $111, %ebx
    .byte   100, 103, 144 ; Start marker

.innermostlooplabel:
    ; Loop body
    ; ...
    jb .innermostlooplabel ; conditional branch

    movl    $222, %ebx
    .byte   100, 103, 144 ; End marker
```

OSACA functionality

- Benchmarking of application codes and extraction of relevant instructions
- Producing benchmarks for not yet known instructions, adding data to data file

- Throughput analysis of ASM code snippet
- Optional: Insert IACA markers for better loop identification

Throughput Analysis

User input

```
// Multiply i with immediate
// and add to array
int t = 0.19;
int main(void){
    int i = 0;
    //STARTLOOP
    while(i < 1000){
        arr[i] = arr[i-1] + i * t;
        i += 1;
    }
}
```

OSACA marker

IACA byte markers

Extract instructions

Memory GPR XMM

```
xor         %eax,%eax
lea         0x1(%rax,%rax,1),%edx
vcvtsi2ss   %edx,%xmm2,%xmm2
vmulss      %xmm2,%xmm0,%xmm3
lea         0x2(%rax,%rax,1),%ecx
vaddss      %xmm3,%xmm1,%xmm4
vxorps      %xmm1,%xmm1,%xmm1
vcvtsi2ss   %ecx,%xmm1,%xmm1
vmulss      %xmm1,%xmm0,%xmm5
vmovss      %xmm4,0x4(%rsp,%rax,8)
```

OSACA

CSV data files

Throughput analysis

```
Port Binding in Cycles Per Iteration:
---------------------------------------
|  Port  |  0  |  1  |  2  |  3  |  4  |  5  |
---------------------------------------
| Cycles | 4.0 | 5.0 | 3.0 | 3.0 | 2.0 | 2.0 |
---------------------------------------

          Ports Pressure in cycles
|  0   |  1   |  2   |  3   |  4   |  5   |
---------------------------------------------
|      |      | 1.00 | 1.00 |      |      | lea     0x1(%rax,%rax,1),%edx
| 0.50 | 1.00 |      |      |      | 0.50 | vcvtsi2ss %edx,%xmm2,%xmm2
| 1.00 |      |      |      |      |      | vmulss %xmm2,%xmm0,%xmm3
|      |      | 1.00 | 1.00 |      |      | lea     0x2(%rax,%rax,1),%ecx
|      | 1.00 |      |      |      |      | vaddss %xmm3,%xmm1,%xmm4
| 0.33 | 0.33 |      |      |      | 0.33 | vxorps %xmm1,%xmm1,%xmm1
| 0.50 | 1.00 |      |      |      | 0.50 | vcvtsi2ss %ecx,%xmm1,%xmm1
| 1.00 |      |      |      |      |      | vmulss %xmm1,%xmm0,%xmm5
|      |      | 0.50 | 0.50 | 1.00 |      | vmovss %xmm4,0x4(%rsp,%rax,8)
|      | 1.00 |      |      |      |      | vaddss %xmm5,%xmm4,%xmm1
|      |      | 0.50 | 0.50 | 1.00 |      | vmovss %xmm1,0x8(%rsp,%rax,8)
| 0.33 | 0.33 |      |      |      | 0.33 | inc     %rax
| 0.33 | 0.33 |      |      |      | 0.33 | cmp     $0x1f3,%rax
|      |      |      |      |      |      | jb      400bc2 <main+0x62>
Total number of estimated throughput: 5.0
```

Benchmark file

```
#define INSTR vcvtsi2ss
#define NINST 32
#define N edi
#define i r8d
.intel_syntaxnoprefix
.globl ninst
.data
ninst:
.long NINST
.align32
PI:
.long0xf01b866e, 0x400921f9
.text
.globllatency
.typelatency, @function
.align32
loop:
        inc     i
        INSTR   xmm3, xmm0, eax
        INSTR   xmm4, xmm1, ebx
        INSTR   xmm5, xmm2, ecx
        INSTR   xmm6, xmm0, eax
        INSTR   xmm7, xmm1, ebx
        INSTR   xmm8, xmm2, ecx
```

ibench

Fork me on GitHub

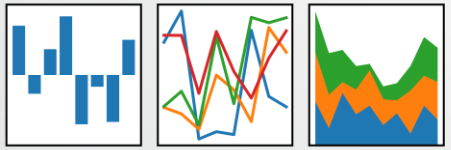
ibench output

vcvtsi2ss-xmm_xmm_r32	1.0
vcvtsi2ss-xmm_xmm_r32-TP	3.0
vmulss-xmm_xmm_xmm	1.0

OSACA dependencies

- NumPy
- pandas

  $y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



- Kerncraft (https://github.com/RRZE-HPC/kerncraft)
- ibench (https://github.com/hofm/ibench)

Example 1: STREAM Scale

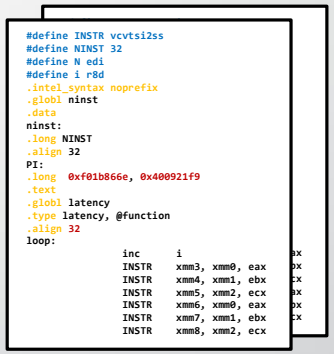
Initial situation

```
double a[n], b[N];
double s;

    //STARTLOOP
    for(int i = 0; i < N; ++i)
        a[i] = s * b[i];
```

```
$ osaca --arch IVB PATH/TO/FILE
```

```
Throughput Analysis Report
--------------------------
X - No information for this instruction in database
* - Instruction micro-ops not bound to a port

Port Binding in Cycles Per Iteration:
-------------------------------------------------
| Port   |   0  |   1  |  2  |  3  |  4  |   5  |
-------------------------------------------------
| Cycles | 2.00 | 1.00 | 5.0 | 5.0 | 2.0 | 1.00 |
-------------------------------------------------
            Ports Pressure in cycles
|   0  |   1  |   2  |   3  |   4  |   5  |
-------------------------------------------
|      |      | 0.50 | 0.50 | 1.00 |      | movl $0x0,-0x24(%rbp)
|      |      |      |      |      |      | jmp 10b <scale+0x10b>
|      |      | 0.50 | 0.50 |      |      | mov -0x48(%rbp),%rax
|      |      | 0.50 | 0.50 |      |      | mov -0x24(%rbp),%edx
| 0.33 | 0.33 |      |      |      | 0.33 | movslq %edx,%rdx
|      |      | 0.50 | 0.50 |      |      | vmovsd (%rax,%rdx,8),...
| 1.00 |      | 0.50 | 0.50 |      |      | vmulsd -0x50(%rbp),...
|      |      | 0.50 | 0.50 |      |      | mov -0x38(%rbp),%rax
|      |      | 0.50 | 0.50 |      |      | mov -0x24(%rbp),%edx
| 0.33 | 0.33 |      |      |      | 0.33 | movslq %edx,%rdx
|      |      | 0.50 | 0.50 | 1.00 |      | vmovsd %xmm0,...
|      |      |      |      |      |      | X addl $0x1,-0x24(%rbp)
|      |      | 0.50 | 0.50 |      |      | mov -0x24(%rbp),%eax
| 0.33 | 0.33 | 0.50 | 0.50 |      | 0.33 | cmp -0x54(%rbp),%eax
|      |      |      |      |      |      | jl e4 <scale+0xe4>
Total number of estimated throughput: 5.0
```

addl-mem_imd
(TP & LT)



```
#define INSTR vcvtsi2ss
#define NINST 32
#define N edi
#define i r8d
.intel_syntax noprefix
.globl ninst
.data
ninst:
.long NINST
.align 32
PI:
.long  0xf01b866e, 0x400921f9
.text
.globl latency
.type latency, @function
.align 32
loop:
        inc     i
        INSTR   xmm3, xmm0, eax
        INSTR   xmm4, xmm1, ebx
        INSTR   xmm5, xmm2, ecx
        INSTR   xmm6, xmm0, eax
        INSTR   xmm7, xmm1, ebx
        INSTR   xmm8, xmm2, ecx
```

```
$ ./ibench ./AVX 2.2
```

```
Using frequency 2.20GHz.
add-mem_imd-TP: 1.023 (clock cycles) [DEBUG - result: 1.000000]
add-mem_imd:    6.050 (clock cycles) [DEBUG - result: 1.000000]
```

```
$ osaca --include-ibench -arch IVB PATH/TO/IBENCH_OUTPUT
```

```
Ibench output FILE successfully in database included.
2 values were added.
```

```
$ osaca --arch IVB PATH/TO/FILE
```

```
Throughput Analysis Report
--------------------------
X - No information for this instruction in database
* - Instruction micro-ops not bound to a port

Port Binding in Cycles Per Iteration:
-------------------------------------------------
| Port   |   0  |   1  |  2  |  3  |  4  |   5  |
-------------------------------------------------
| Cycles | 2.33 | 1.33 | 6.0 | 6.0 | 3.0 | 1.33 |
-------------------------------------------------
            Ports Pressure in cycles
|   0  |   1  |   2  |   3  |   4  |   5  |
-------------------------------------------
|      |      | 0.50 | 0.50 | 1.00 |      | movl $0x0,-0x24(%rbp)
|      |      |      |      |      |      | jmp 10b <scale+0x10b>
|      |      | 0.50 | 0.50 |      |      | mov -0x48(%rbp),%rax
|      |      | 0.50 | 0.50 |      |      | mov -0x24(%rbp),%edx
| 0.33 | 0.33 |      |      |      | 0.33 | movslq %edx,%rdx
|      |      | 0.50 | 0.50 |      |      | vmovsd (%rax,%rdx,8),...
| 1.00 |      | 0.50 | 0.50 |      |      | vmulsd -0x50(%rbp),...
|      |      | 0.50 | 0.50 |      |      | mov -0x38(%rbp),%rax
|      |      | 0.50 | 0.50 |      |      | mov -0x24(%rbp),%edx
| 0.33 | 0.33 |      |      |      | 0.33 | movslq %edx,%rdx
|      |      | 0.50 | 0.50 | 1.00 |      | vmovsd %xmm0,...
| 0.33 | 0.33 | 1.00 | 1.00 | 1.00 | 0.33 | addl $0x1,-0x24(%rbp)
|      |      | 0.50 | 0.50 |      |      | mov -0x24(%rbp),%eax
| 0.33 | 0.33 | 0.50 | 0.50 |      | 0.33 | cmp -0x54(%rbp),%eax
|      |      |      |      |      |      | jl e4 <scale+0xe4>
Total number of estimated throughput: 6.0
```

Example 2: 2D-5pt stencil

Initial situation

```
for(j=1; j<M-1; ++j){
    #pragma vector aligned
    //STARTLOOP
    for(int i=1; i<N-1; ++i){
        b[j][i] = (a[j][i-1] + a[j][i+1]
                   + a[j-1][i] + a[j+1][i]) * s;
    }
}
```

```
$ osaca --arch IVB PATH/TO/FILE
```

```
Throughput Analysis Report
--------------------------
X - No information for this instruction in database
* - Instruction micro-ops not bound to a port


Port Binding in Cycles Per Iteration:
-----------------------------------------------------
|  Port  |   0   |   1   |  2  |  3  |  4  |   5   |
-----------------------------------------------------
| Cycles | 16.67 | 19.17 | 8.5 | 7.0 | 8.0 | 11.67 |
-----------------------------------------------------


          Ports Pressure in cycles
|  0   |  1   |  2   |  3   |  4   |  5   |
-------------------------------------------
| 0.50 | 0.50 |      |      |      |      | lea    (%r15,%rcx,8),%r11
| 0.50 | 0.50 |      |      |      |      | lea    (%r14,%rcx,8),%rdx
| 0.33 | 0.33 |      |      |      | 0.33 | add    $0xfffffffe,%edi
| 0.33 | 0.33 |      |      |      | 0.33 | mov    %rcx,%r9
| 0.33 | 0.33 |      |      |      | 0.33 | mov    %edi,%eax
| 0.33 | 0.33 |      |      |      | 0.33 | and    $0xfffffff0,%eax
|      | 0.50 |      |      |      | 0.50 | shl    $0x4,%r9
| 0.33 | 0.33 |      |      |      | 0.33 | movslq %eax,%rax
| 0.33 | 0.33 |      |      |      | 0.33 | add    %r14,%r9
| 0.33 | 0.33 |      |      |      | 0.33 | movslq %edi,%rdi
|      |      | 0.50 | 0.50 | 1.00 |      | mov    %rax,-0x50(%rbp)
|      |      | 0.50 | 0.50 | 1.00 |      | mov    %rdi,-0x58(%rbp)
| 0.33 | 0.33 |      |      |      | 0.33 | cmp    $0x2,%rcx
|      |      |      |      |      |      | jle    196 <jacobi2D5pt+0x196>
| 0.33 | 0.33 |      |      |      | 0.33 | cmp    $0x10,%edi
|      |      |      |      |      |      | jl     20b <jacobi2D5pt+0x20b>
|      |      | 0.50 | 0.50 |      |      | mov    -0x48(%rbp),%r14d
| 0.33 | 0.33 |      |      |      | 0.33 | xor    %edx,%edx
|      |      | 0.50 | 0.50 |      |      | mov    -0x50(%rbp),%r12
| 0.50 | 0.50 |      |      |      |      | lea    (%r11,%r8,1),%rax
|      |      | 0.50 | 0.50 | 1.00 |      | vmovupd %ymm0,0x8(%rax,%rdx,8)
|      |      | 0.50 | 0.50 | 1.00 |      | vmovupd %ymm0,0x28(%rax,%rdx,8)
|      |      | 0.50 | 0.50 | 1.00 |      | vmovupd %ymm0,0x48(%rax,%rdx,8)
|      |      | 0.50 | 0.50 | 1.00 |      | vmovupd %ymm0,0x68(%rax,%rdx,8)
| 0.33 | 0.33 |      |      |      | 0.33 | add    $0x10,%rdx
| 0.33 | 0.33 |      |      |      | 0.33 | cmp    %r12,%rdx
|      |      |      |      |      |      | jb     d9 <jacobi2D5pt+0xd9>
| 0.50 | 0.50 |      |      |      |      | lea    0x1(%r14),%eax
| 0.33 | 0.33 |      |      |      | 0.33 | cmp    %edi,%eax
|      |      |      |      |      |      | ja     196 <jacobi2D5pt+0x196>
| 0.33 | 0.33 |      |      |      | 0.33 | movslq %r14d,%r14
|      |      | 0.50 | 0.50 |      |      | mov    -0x58(%rbp),%r13
| 0.33 | 0.33 |      |      |      | 0.33 | sub    %r14,%r13
| 0.33 | 0.33 |      |      |      | 0.33 | cmp    $0x4,%r13
|      |      |      |      |      |      | jl     203 <jacobi2D5pt+0x203>
```

```
| 0.33 | 0.33 |      |      |      | 0.33 | mov    %r13d,%r15d
| 0.50 | 0.50 |      |      |      |      | lea    (%r11,%r8,1),%rax
| 0.33 | 0.33 |      |      |      | 0.33 | and    $0xfffffffc,%r15d
| 0.33 | 0.33 |      |      |      | 0.33 | xor    %edx,%edx
| 0.33 | 0.33 |      |      |      | 0.33 | movslq %r15d,%r15
| 0.50 | 0.50 |      |      |      |      | lea    (%rax,%r14,8),%rax
|      |      | 0.50 | 0.50 | 1.00 |      | vmovupd %ymm0,0x8(%rax,%rdx,8)
| 0.33 | 0.33 |      |      |      | 0.33 | add    $0x4,%rdx
| 0.33 | 0.33 |      |      |      | 0.33 | cmp    %r15,%rdx
|      |      |      |      |      |      | jb     12e <jacobi2D5pt+0x12e>
| 0.33 | 0.33 |      |      |      | 0.33 | cmp    %r13,%r15
|      |      |      |      |      |      | jae    196 <jacobi2D5pt+0x196>
|      |      | 0.50 | 0.50 |      |      | mov    -0x38(%rbp),%rax
| 0.50 | 0.50 |      |      |      |      | lea    (%r11,%r8,1),%r12
|      |      | 0.50 | 0.50 |      |      | mov    -0x40(%rbp),%rsi
| 0.50 | 0.50 |      |      |      |      | lea    (%r9,%r8,1),%rdx
| 0.50 | 0.50 |      |      |      |      | lea    (%r12,%r14,8),%r12
| 0.33 | 0.33 |      |      |      | 0.33 | add    %r8,%rax
| 0.33 | 0.33 |      |      |      | 0.33 | add    %r8,%rsi
| 0.50 | 0.50 |      |      |      |      | lea    (%rdx,%r14,8),%rdx
| 0.50 | 0.50 |      |      |      |      | lea    (%rax,%r14,8),%rax
| 0.50 | 0.50 |      |      |      |      | lea    (%rsi,%r14,8),%r14
|      |      | 0.50 | 0.50 |      |      | vmovsd (%r14,%r15,8),%xmm2
|      | 1.00 | 0.50 |      |      | 0.50 | vaddsd 0x10(%r14,%r15,8),%xmm2,%xmm3
|      | 1.00 | 0.50 |      |      | 0.50 | vaddsd 0x8(%rax,%r15,8),%xmm3,%xmm4
|      | 1.00 | 0.50 |      |      | 0.50 | vaddsd 0x8(%rdx,%r15,8),%xmm4,%xmm5
| 1.00 |      |      |      |      |      | vmulsd %xmm5,%xmm1,%xmm6
|      |      | 0.50 | 0.50 | 1.00 |      | vmovsd %xmm6,0x8(%r12,%r15,8)
| 0.33 | 0.33 |      |      |      | 0.33 | inc    %r15
| 0.33 | 0.33 |      |      |      | 0.33 | cmp    %r13,%r15
|      |      |      |      |      |      | jb     168 <jacobi2D5pt+0x168>
| 0.33 | 0.33 |      |      |      | 0.33 | xor    %r15d,%r15d
|      |      |      |      |      |      | jmpq   13d <jacobi2D5pt+0x13d>
| 0.33 | 0.33 |      |      |      | 0.33 | xor    %r14d,%r14d
|      |      |      |      |      |      | jmpq   fa <jacobi2D5pt+0xfa>
|      |      |      |      |      |      | * nopl   (%rax)
|      |      |      |      |      |      | * nopw   %cs:0x0(%rax,%rax,1)
Total number of estimated throughput: 19.17
```

This isn’t the code you are looking for…

```
$ osaca --insert-marker PATH/TO/ASM_FILE
```

```
Blocks found in assembly file:
   block    | OPs | pck. | AVX || Registers |    YMM    |    XMM    |    GP    ||ptr.inc|
------------+-----+------+-----++-----------+-----------+-----------+----------++-------|
0 ..B1.8    |   8 |   0  |   0 ||  15 (  4) |   4 ( 1) |   0 ( 0) | 11 ( 3) ||   128 |
1 ..B1.13   |   5 |   0  |   0 ||   6 (  4) |   1 ( 1) |   0 ( 0) |  5 ( 3) ||    32 |
2 ..B1.17   |  12 |   0  |   0 ||  28 ( 13) |   1 ( 1) |  11 ( 6) | 16 ( 6) ||  None |
Choose block to be marked [2]:
```

```
$ osaca --arch IVB --iaca PATH/TO/(ASM)FILE
```

```
Throughput Analysis Report
--------------------------
X - No information for this instruction in database
* - Instruction micro-ops not bound to a port


Port Binding in Cycles Per Iteration:
-------------------------------------------------
|  Port  |   0  |   1  |  2  |  3  |  4  |   5  |
-------------------------------------------------
| Cycles | 1.67 | 3.67 | 2.5 | 2.5 | 1.0 | 0.67 |
-------------------------------------------------


          Ports Pressure in cycles
|  0   |  1   |  2   |  3   |  4   |  5   |
-------------------------------------------
|      |      | 0.50 | 0.50 |      |      | vmovsd    (%r14,%r15,8), %xmm2
|      | 1.00 | 0.50 | 0.50 |      |      | vaddsd    16(%r14,%r15,8), %xmm2, %xmm3
|      | 1.00 | 0.50 | 0.50 |      |      | vaddsd    8(%rax,%r15,8), %xmm3, %xmm4
|      | 1.00 | 0.50 | 0.50 |      |      | vaddsd    8(%rdx,%r15,8), %xmm4, %xmm5
| 1.00 |      |      |      |      |      | vmulsd    %xmm5, %xmm1, %xmm6
|      |      | 0.50 | 0.50 | 1.00 |      | vmovsd    %xmm6, 8(%r12,%r15,8)
| 0.33 | 0.33 |      |      |      | 0.33 | incq      %r15
| 0.33 | 0.33 |      |      |      | 0.33 | cmpq      %r13, %r15
|      |      |      |      |      |      | jb        ..B1.17
Total number of estimated throughput: 3.67
```

Comparison OSACA vs. IACA

```
Throughput Analysis Report
--------------------------
X - No information for this instruction in database
* - Instruction micro-ops not bound to a port


Port Binding in Cycles Per Iteration:
-------------------------------------------------
|  Port  |   0  |   1  |  2  |  3  |  4  |   5  |
-------------------------------------------------
| Cycles | 1.67 | 3.67 | 2.5 | 2.5 | 1.0 | 0.67 |
-------------------------------------------------


          Ports Pressure in cycles
|  0   |  1   |  2   |  3   |  4   |  5   |
--------------------------------------------
|      |      | 0.50 | 0.50 |      |      | vmovsd    (%r14,%r15,8), %xmm2
|      | 1.00 | 0.50 | 0.50 |      |      | vaddsd    16(%r14,%r15,8),%xmm2,%xmm3
|      | 1.00 | 0.50 | 0.50 |      |      | vaddsd    8(%rax,%r15,8),%xmm3, %xmm4
|      | 1.00 | 0.50 | 0.50 |      |      | vaddsd    8(%rdx,%r15,8),%xmm4, %xmm5
| 1.00 |      |      |      |      |      | vmulsd    %xmm5, %xmm1, %xmm6
|      |      | 0.50 | 0.50 | 1.00 |      | vmovsd    %xmm6, 8(%r12,%r15,8)
| 0.33 | 0.33 |      |      |      | 0.33 | incq      %r15
| 0.33 | 0.33 |      |      |      | 0.33 | cmpq      %r13, %r15
|      |      |      |      |      |      | jb        ..B1.17
Total number of estimated throughput: 3.67
```

```
Throughput Analysis Report
--------------------------
Block Throughput: 3.00 Cycles       Throughput Bottleneck: FrontEnd

Port Binding In Cycles Per Iteration:
---------------------------------------------------------------------------
|  Port  |  0   -  DV  |  1   |  2   -  D   |  3   -  D   |  4   |  5   |
---------------------------------------------------------------------------
| Cycles | 1.0    0.0  | 3.0  | 2.5    2.0  | 2.5    2.0  | 1.0  | 2.0  |
---------------------------------------------------------------------------

N - port number or number of cycles resource conflict caused delay, DV - Divider pipe (on port 0)
D - Data fetch pipe (on ports 2 and 3), CP - on a critical path
F - Macro Fusion with the previous instruction occurred
* - instruction micro-ops not bound to a port
^ - Micro Fusion happened
# - ESP Tracking sync uop was issued
@ - SSE instruction followed an AVX256/AVX512 instruction, dozens of cycles penalty is expected
X - instruction not supported, was not accounted in Analysis

| Num Of |              Ports pressure in cycles               |    |
|  Uops  |  0  - DV  |  1  |  2  -  D  |  3  -  D  |  4  |  5  |    |
---------------------------------------------------------------------
|   1    |           |     | 1.0   1.0 |           |     |     |    | vmovsd xmm2, qword ptr [r14+r15*8]
|   2    |           | 1.0 |           | 1.0   1.0 |     |     | CP | vaddsd xmm3, xmm2, qword ptr ...
|   2    |           | 1.0 | 1.0   1.0 |           |     |     | CP | vaddsd xmm4, xmm3, qword ptr ...
|   2    |           | 1.0 |           | 1.0   1.0 |     |     | CP | vaddsd xmm5, xmm4, qword ptr ...
|   1    | 1.0       |     |           |           |     |     |    | vmulsd xmm6, xmm1, xmm5
|   2    |           |     | 0.5       | 0.5       | 1.0 |     |    | vmovsd qword ptr [r12+r15*8+0x8]...
|   1    |           |     |           |           |     | 1.0 |    | inc r15
|   1    |           |     |           |           |     | 1.0 |    | cmp r15, r13
|   0F   |           |     |           |           |     |     |    | jb 0xffffffffffffffd4
Total Num Of Uops: 12
```

Current level of development

- Fetch instructions out of compiled high-level code with OSACA marker
- Fetch instructions out of assembly code with IACA byte markers (no matter if compiled)
- Automated creation of µ-benchmarks for ibench[1]
- Template for easy manual creation of µ-benchmarks
- Continuous integration of benchmark results in database
- Throughput analysis with average port pressure for code snippets

- Supports Intel architectures from Sandy Bridge to Skylake

[1] https://github.com/hofm/ibench

Future Challenges

- Automatic identification of **processor port binding**
- Identification of **critical path** (for latency analysis)
- Identification of **loop-carried dependencies**
- Different x86 and non-x86 **architectures to support** (ARM, Power, AMD, …)
- **User defined test values** for benchmarking in ibench (rand, NaN, 0, all same regs, …)
- Publishing on Python Package Index (pypi)
- Enhance instruction fetching (with or without objdump)

https://github.com/RRZE-HPC/osaca