Peruse and Profit

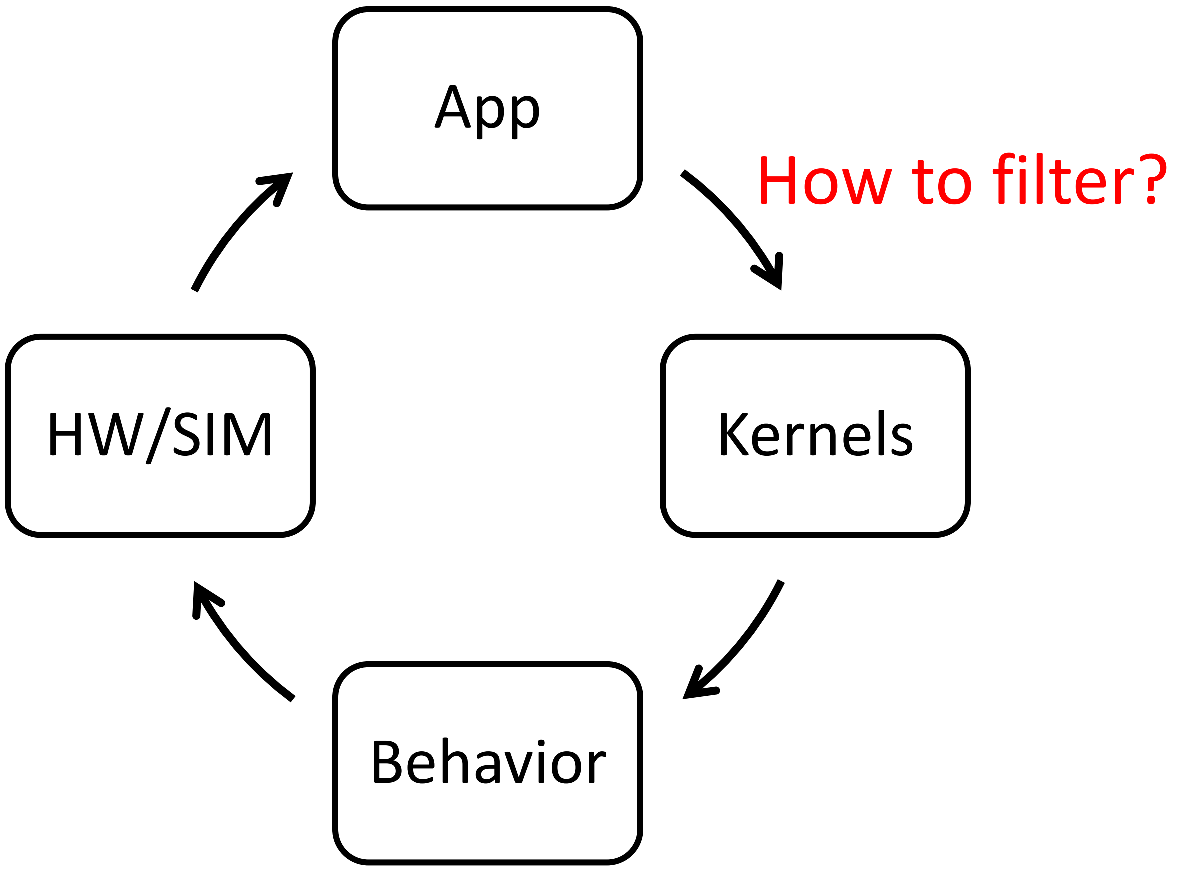
Estimating the Accelerability of loops

Snehasish Kumar, Vijayalakshmi Srinivasan,
Amirali Sharifian, Nick Sumner, Arrvindh Shriraman

SFU SIMON FRASER UNIVERSITY
ENGAGING THE WORLD

IBM®

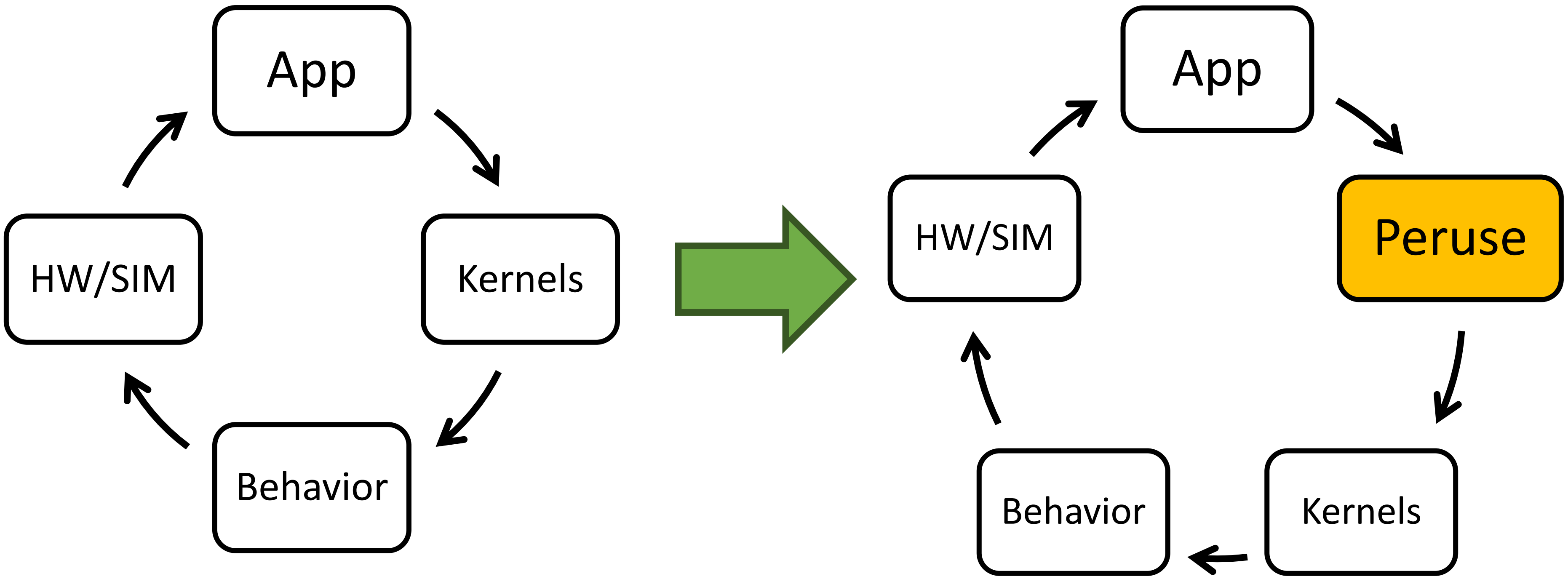
Accelerator design cycle



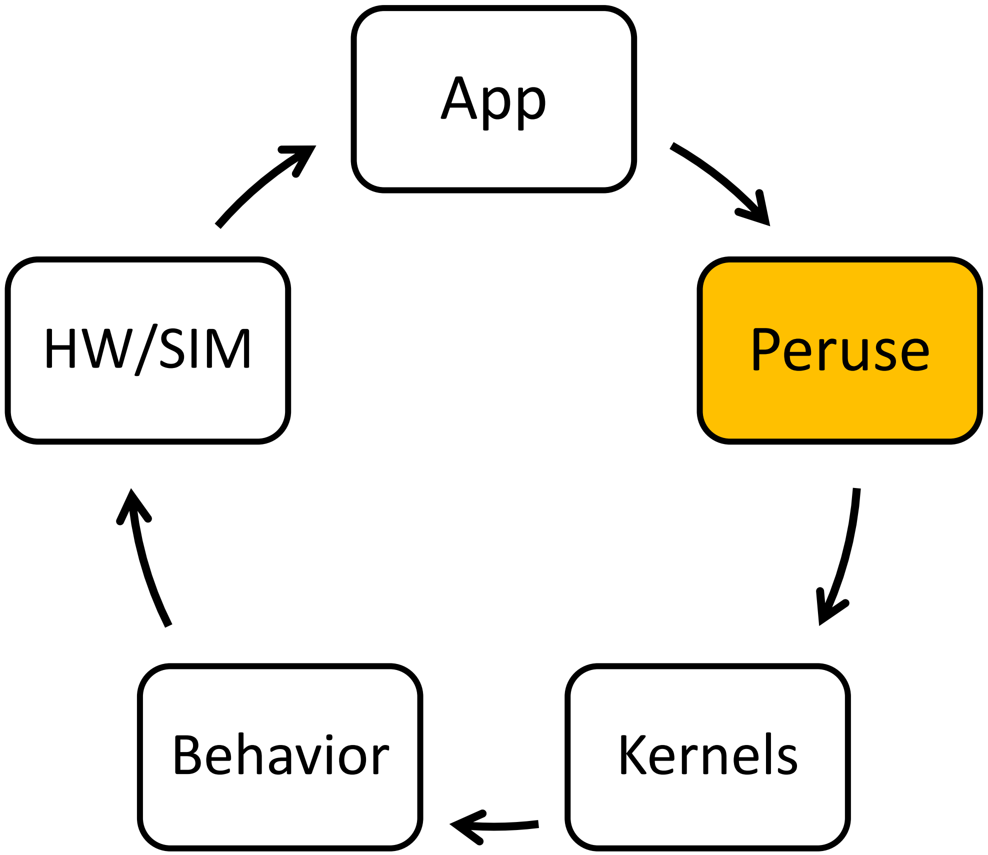
What do we need to know from our program?

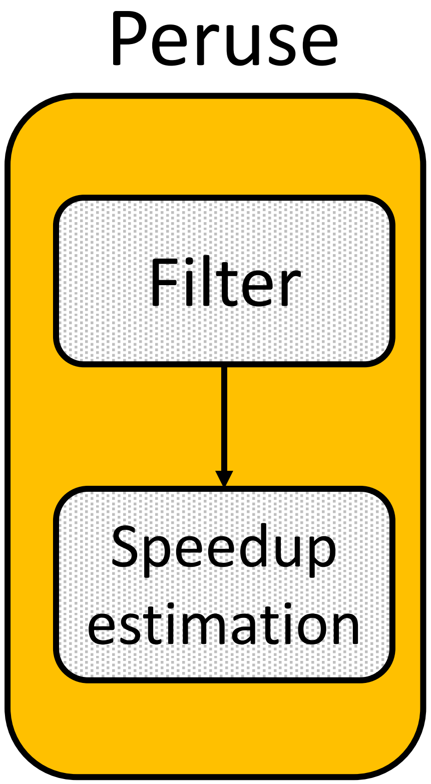
- Where should I ***start***?
  - Finding regions of interest
- What is ***acceleratable***?
  - Correlating characteristics with execution models
- How to ***prioritize*** candidates?
  - Speedup classification

Accelerator design cycle



Peruse


App
Peruse
Kernels
Behavior
HW/SIM


Peruse
Filter
Speedup estimation

Where should I start?

- Peruse focuses on *loops*
- Loop characteristics:
  - **General**
  - **Instruction**
  - **Code**
  - **Data structure**

- Static instructions
- Memory Allocations
- Branches
- Data Structure
- Innermost
- Data footprint
- Atomics
- Containers
- Annotated Parallel
- Compute-to-Communicate Ratio
- Big Iterations
- Accessors
- Trip Count
- Fence
- Mutators
- Loop Exits
- Function Calls
- Side Effect
- Strided Accesses
- Loop Nest Depth
- Vectorizable
- Carried Memory Dependencies
- Idempotent

What is Acceleratable?

```
for (i = 0; i <= N; i++){
    for (j = 0; j <= M; j++){
       data[i][j] -= mean[j];
       data[i][j] /= sqrt(float_n) * stddev[j];
    }
}
```

Characteristics	Fields
General	
Instruction	
Code	
Data Structure	

What is Acceleratable?

```
for (i = 0; i <= N; i++){
    for (j = 0; j <= M; j++){
       data[i][j] -= mean[j];
       data[i][j] /= sqrt(float_n) * stddev[j];
    }
}
```

Characteristics	Fields
General	Memory Dependency: 0
Instruction	
Code	
Data structure	

What is Acceleratable?

```
for (i = 0; i <= N; i++){
    for (j = 0; j <= M; j++){
       data[i][j] -= mean[j];
       data[i][j] /= sqrt(float_n) * stddev[j];
    }
}
```

Characteristics	Fields
General	Memory Dependency: 0
Instruction	Static IR Ins: 21
Code	
Data structure	

What is Acceleratable?

```
for (i = 0; i <= N; i++){
    for (j = 0; j <= M; j++){
        data[i][j] -= mean[j];
        data[i][j] /= sqrt(float_n) * stddev[j];
    }
}
```

Characteristics	Fields
General	Memory Dependency: 0
Instruction	Static IR Ins: 21
Code	C-to-C Ratio: 4.31 bytes/IR
Data structure	

What is Acceleratable?

```
for (i = 0; i <= N; i++){
    for (j = 0; j <= M; j++){
       data[i][j] -= mean[j];
       data[i][j] /= sqrt(float_n) * stddev[j];
    }
}
```

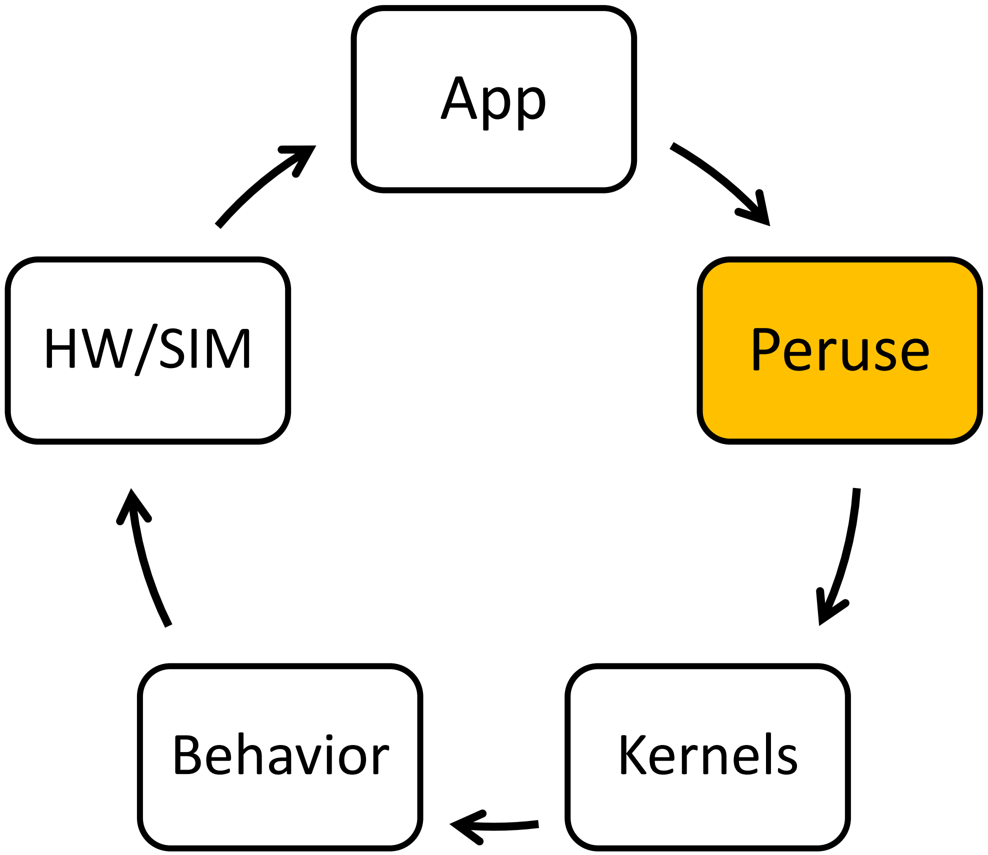
Characteristics	Fields
General	Memory Dependency: 0
Instruction	Static IR Ins: 21
Code	C-to-C Ratio: 4.31 bytes/IR
Data structure	Strided Array Access: 0

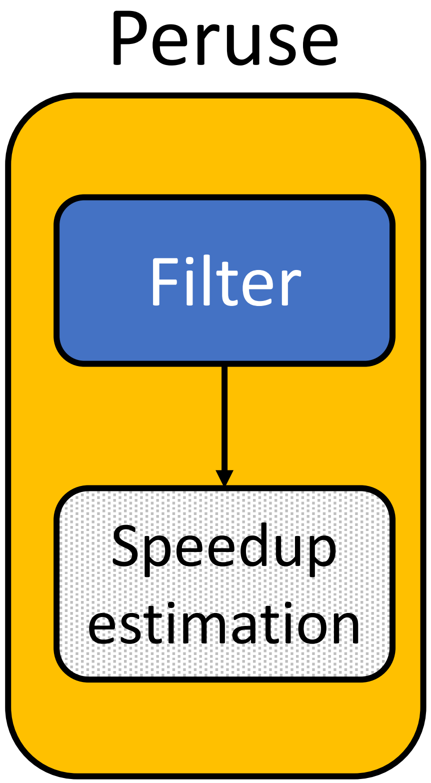
Query based interface

- Peruse output for ***astar:***

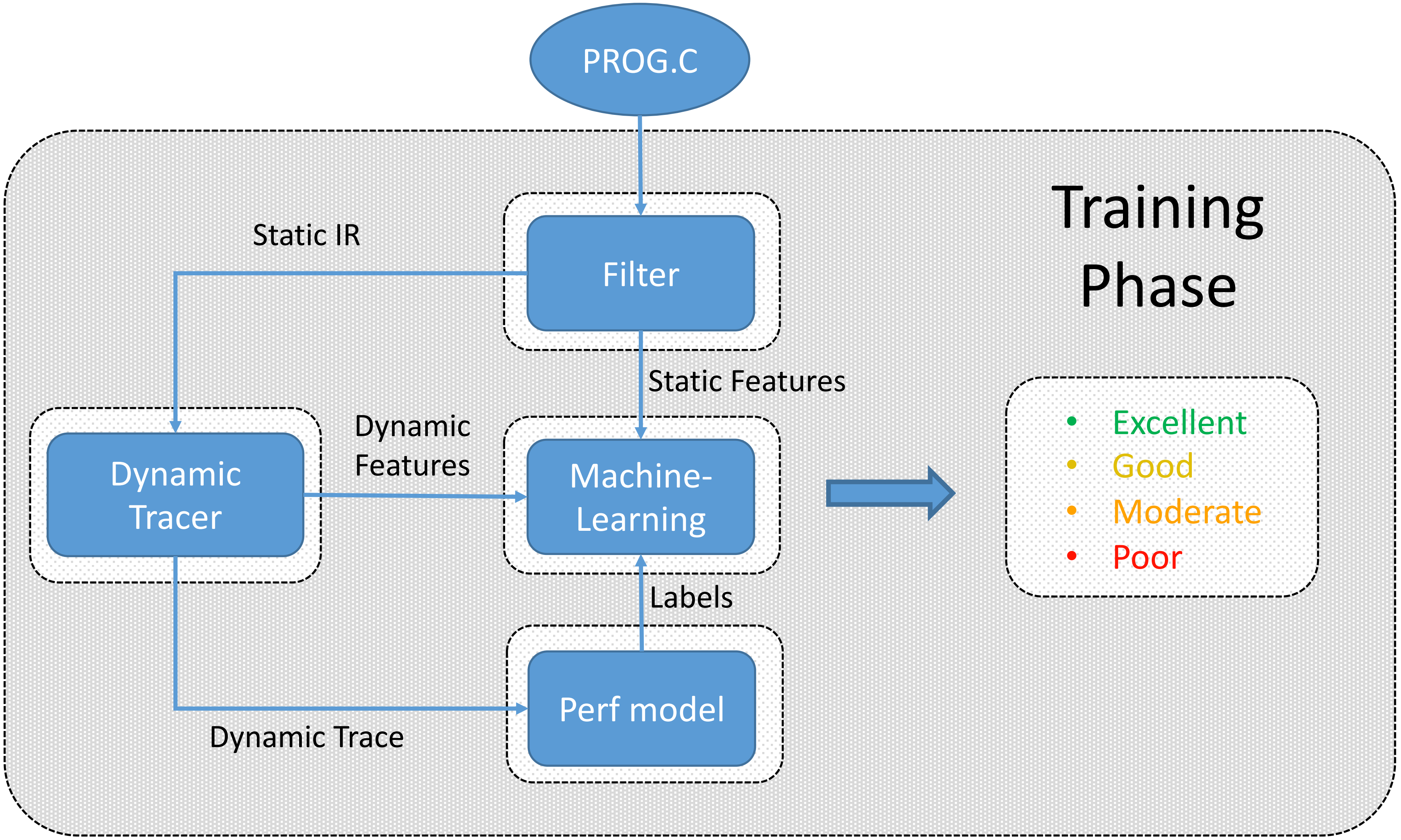
Workload	Total	Inner most
Astar	119	44
Bzip2	244	100
namd	623	222

Peruse

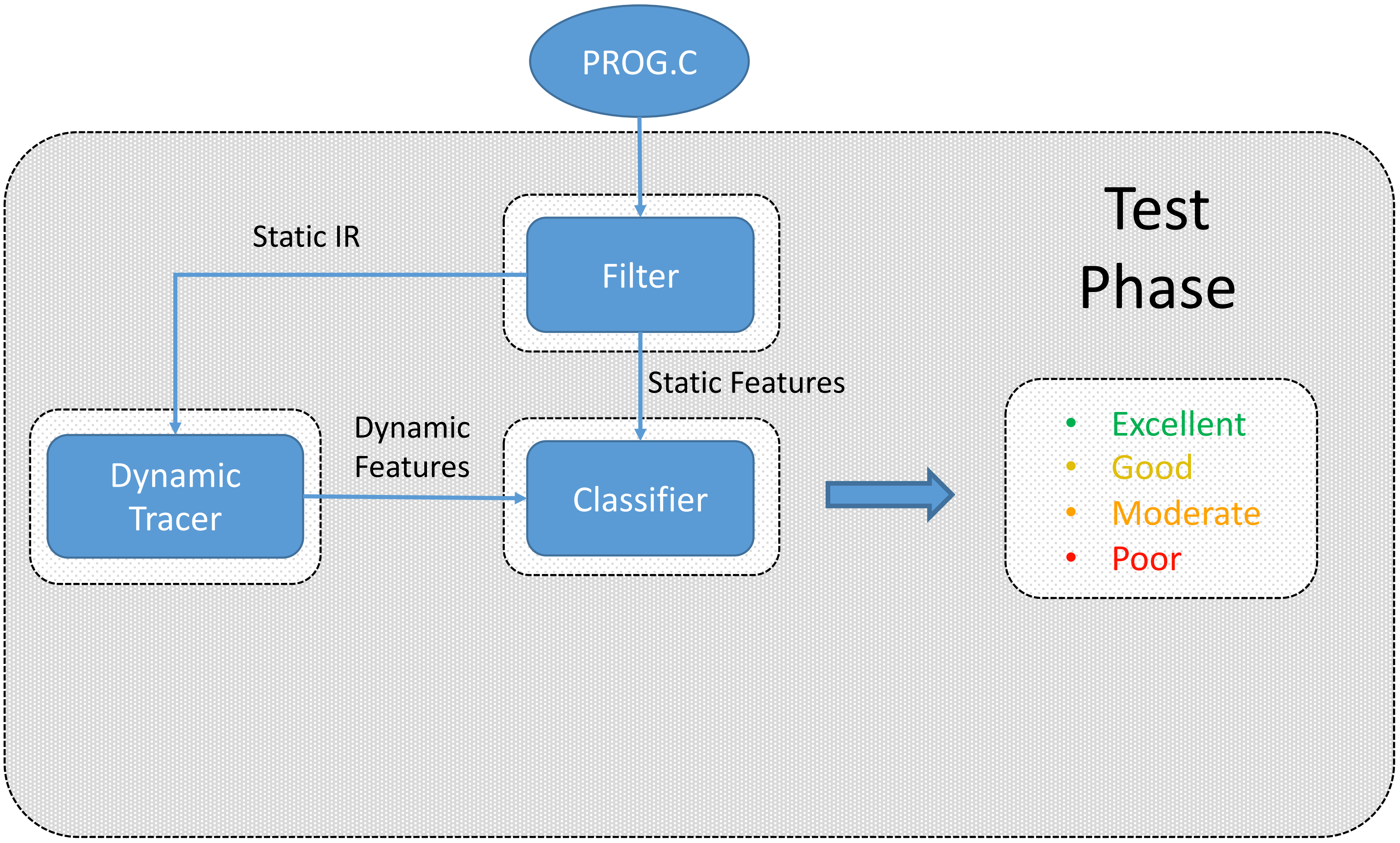

App
Peruse
Kernels
Behavior
HW/SIM


Peruse
Filter
Speedup estimation

Speedup Estimation



Speedup Estimation



Evaluation

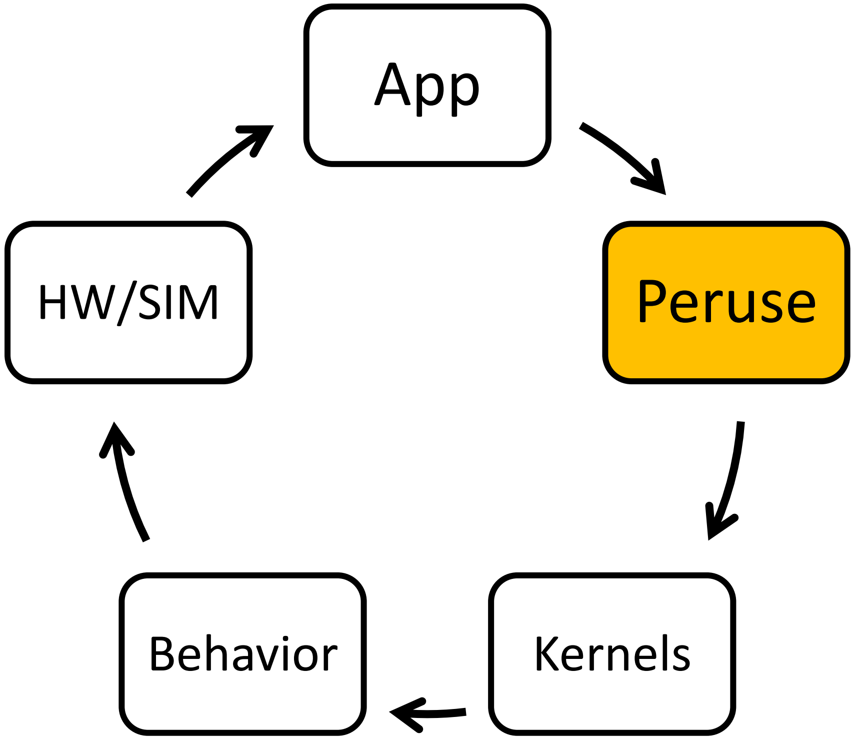
- Training set: ***Polybench*** and ***SHOC*** (~3200)
- Test set: ***470.lbm, 433.milc*** (48)

Avg: 79%

Class	True	Miss
Poor	**0**	**0**
Moderate	0.667	0.095
Good	0.455	0.081
Excellent	**0.935**	**0.176**

Easily predict the best candidates

Summary



- Where should I ***start***?
- What is ***acceleratable***?
- What is speedup ***estimation***?

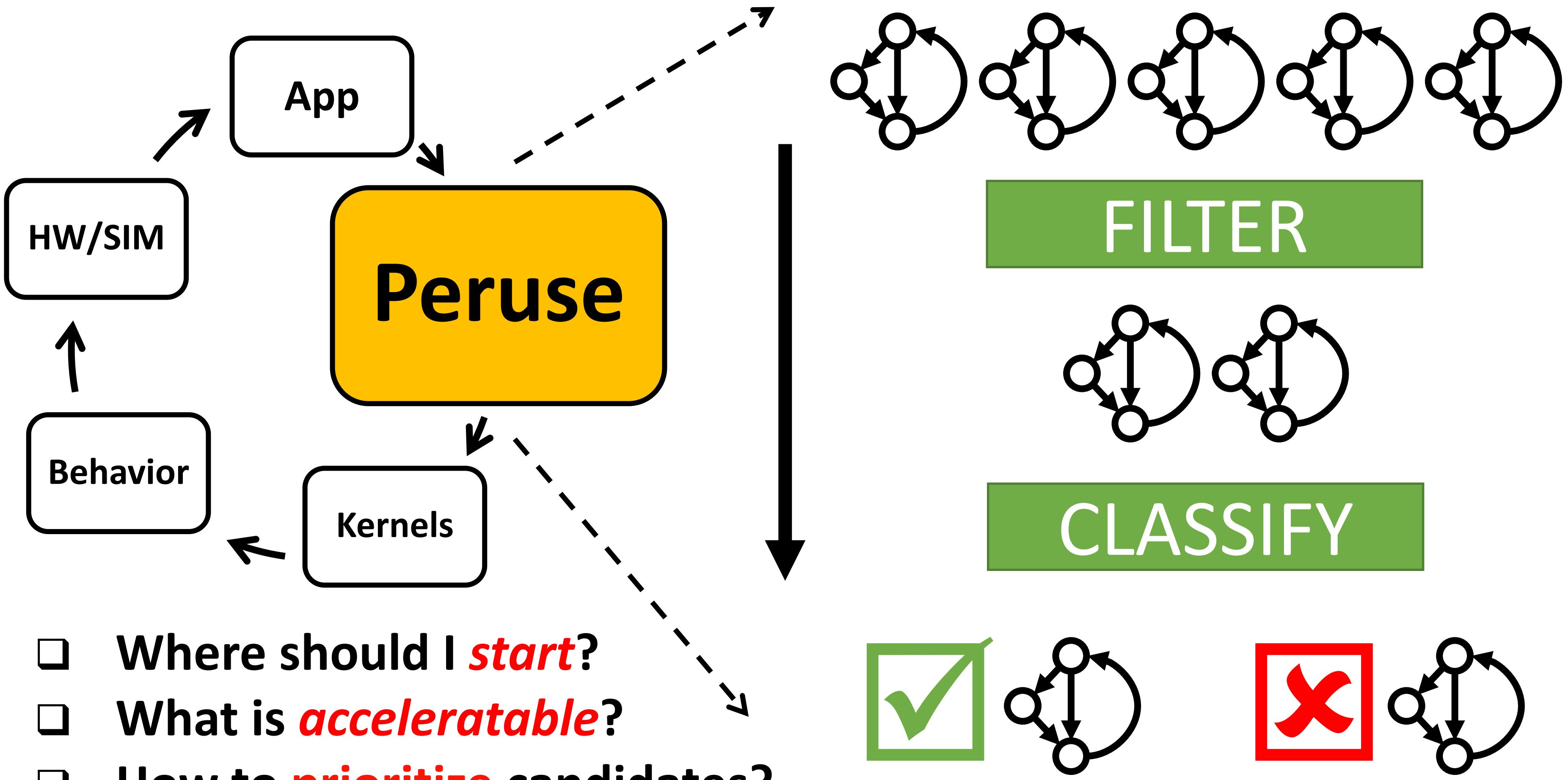
Question?

Query based interface

```
Condition:
        IsInnermost = True AND Mem-Deps-Count = 0
ORDER BY:
        Loop-Data-Tile ASC, Branch-Ins-Count DESC
LIMIT:
        10
```

```
"QUERY" :
    {
        "limit"     : 10,
        "be-true"   : [IsInnermost],
        "be-false"  : [],
        "where"     : [Mem-Deps-count = 0],
        "order-by"  : [
            { "loop-data-tile" : ASC },
            { "Branch-Ins-Count": DESC}
        ]
    }
```

Peruse: Estimating Loop Accelerability



- ❑ Where should I *start*?
- ❑ What is *acceleratable*?
- ❑ How to prioritize candidates?

Thursday, 11:50 Room: Lalezar