# Approximating with Input Level Granularity

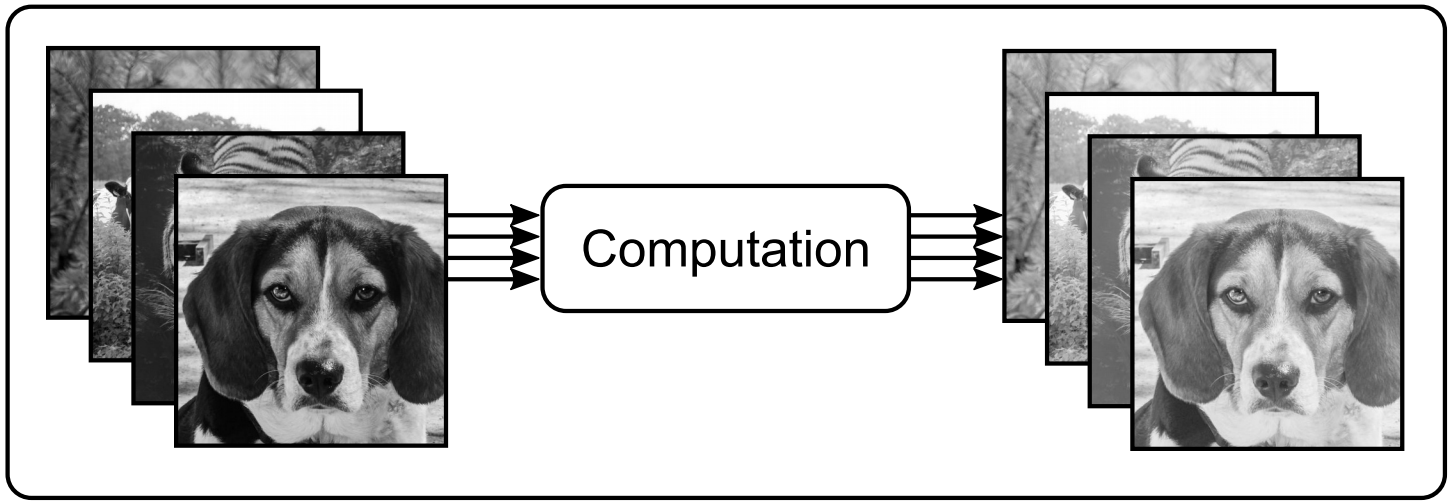
Parker Hill, Michael Laurenzano, Mehrzad Samadi
Scott Mahlke, Jason Mars, Lingjia Tang

ELECTRICAL ENGINEERING
AND COMPUTER SCIENCE
UNIVERSITY OF MICHIGAN

ClarityLab

# Computational Model

- Each operation executed with several inputs



# Sensitivity to Input

# Sensitivity to Input

Input

Gamma Filter

# Sensitivity to Input

Input

Gamma Filter

(16x8 Tiling*) Approximation

*Samadi et al. ASPLOS 2014

# Sensitivity to Input

Input

Gamma Filter

(16x8 Tiling*) Approximation

✓

Is this an acceptable approximation method?

*Samadi et al. ASPLOS 2014

# Sensitivity to Input

Input

Gamma Filter

(16x8 Tiling*) Approximation

✓

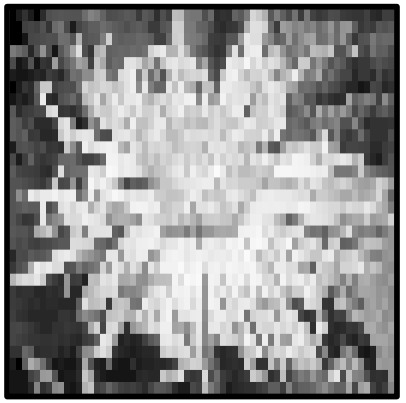
*Samadi et al. ASPLOS 2014

# Sensitivity to Input

Input

Gamma Filter

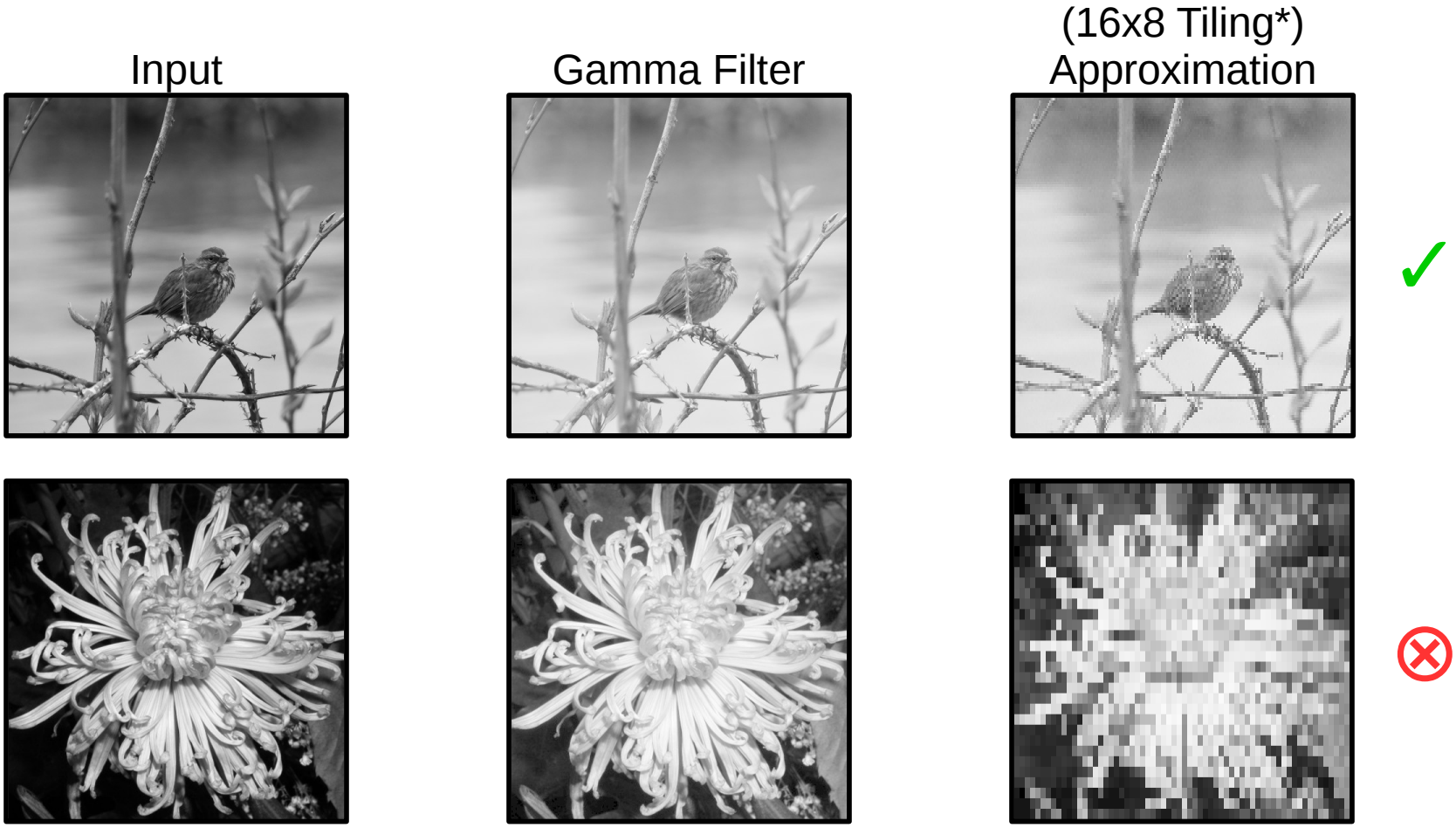
(16x8 Tiling*) Approximation



✓

*Samadi et al. ASPLOS 2014

# Sensitivity to Input

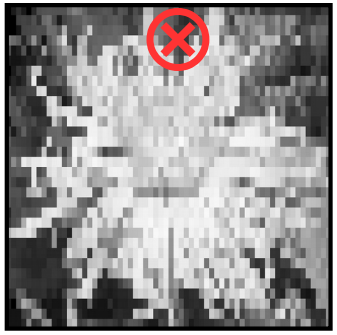



*Samadi et al. ASPLOS 2014

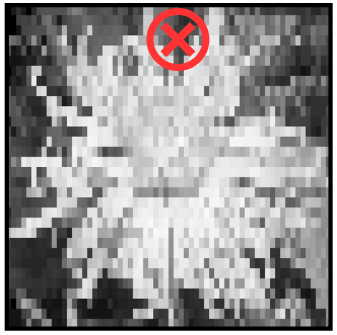
# Previous Work

- Use some set of inputs to:
  - Determine if approximation is accurate enough
  - Pick fastest acceptable approximation
- Reuse the approximation for several inputs

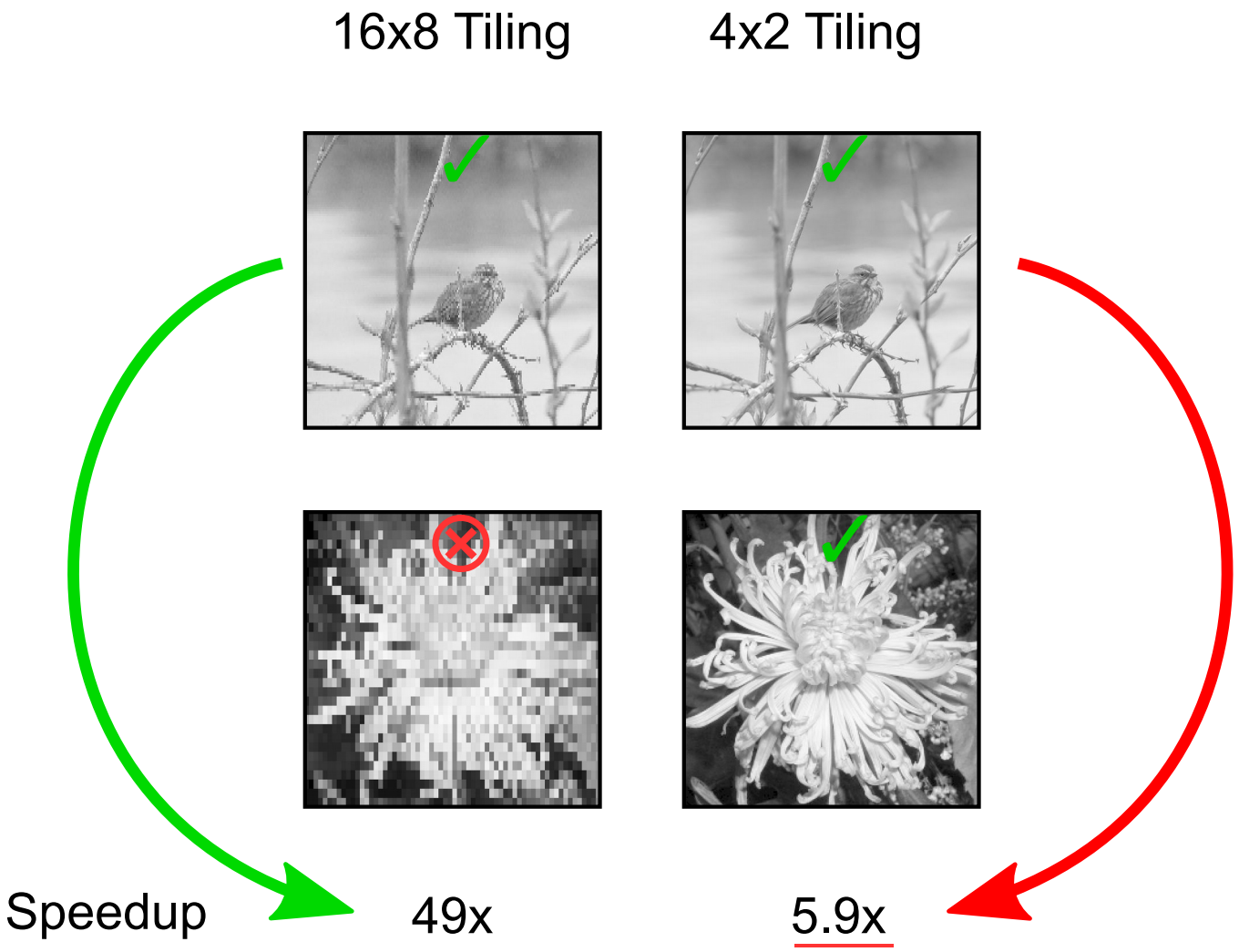
# Performance vs Accuracy

| | 16x8 Tiling | 4x2 Tiling |
|---|---|---|
| |  |  |
| |  |  |
| Speedup | 49x | 5.9x |

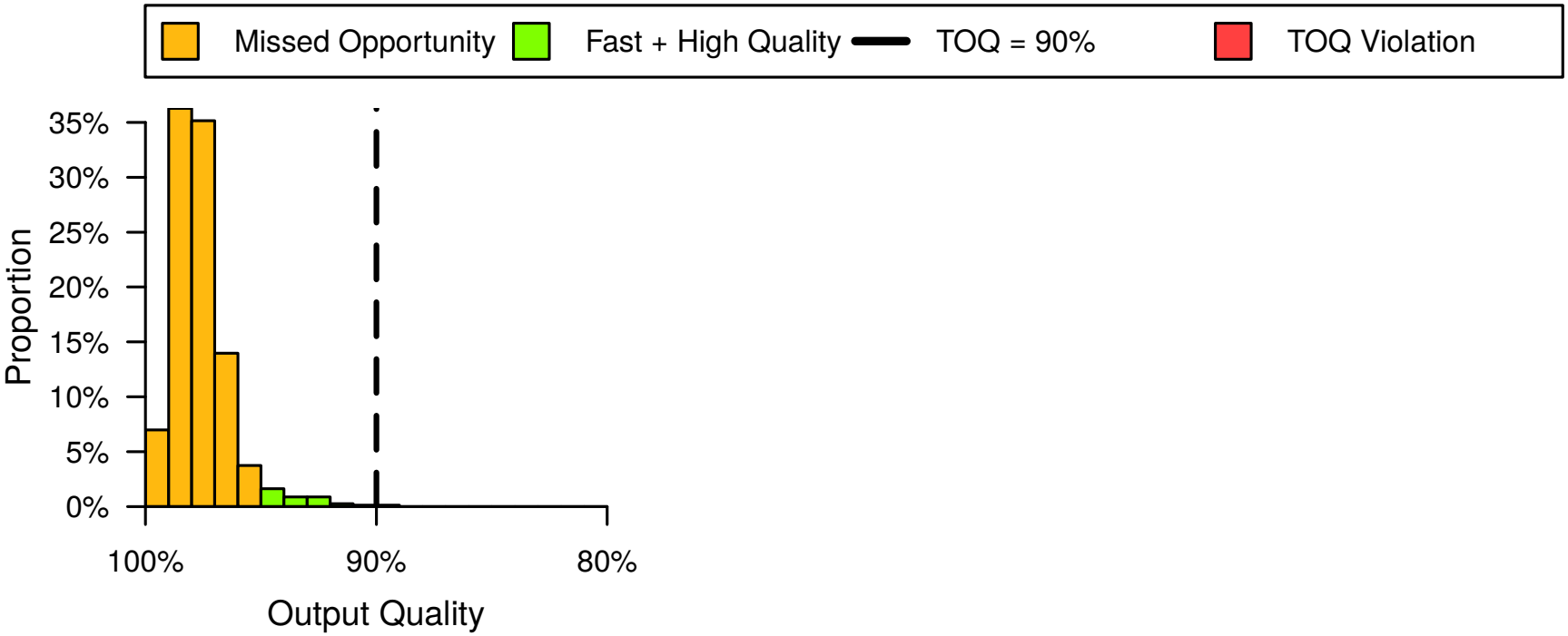
# Performance vs Accuracy

| | 16x8 Tiling | 4x2 Tiling |
|---|---|---|
| |  |  |
| |  |  |
| Speedup | 49x | 5.9x |

# Performance vs Accuracy

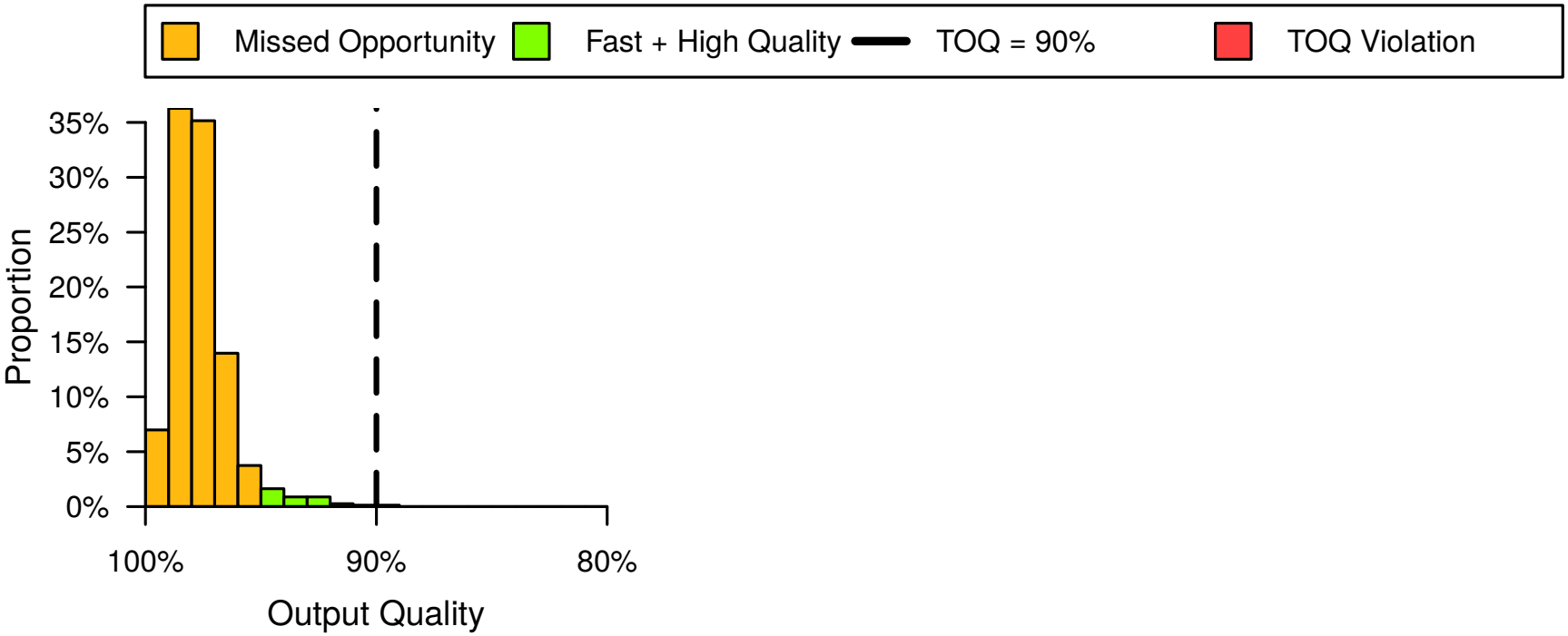


# Trade-off with Many Inputs




4x2 tiling approximation (5.9x speedup)

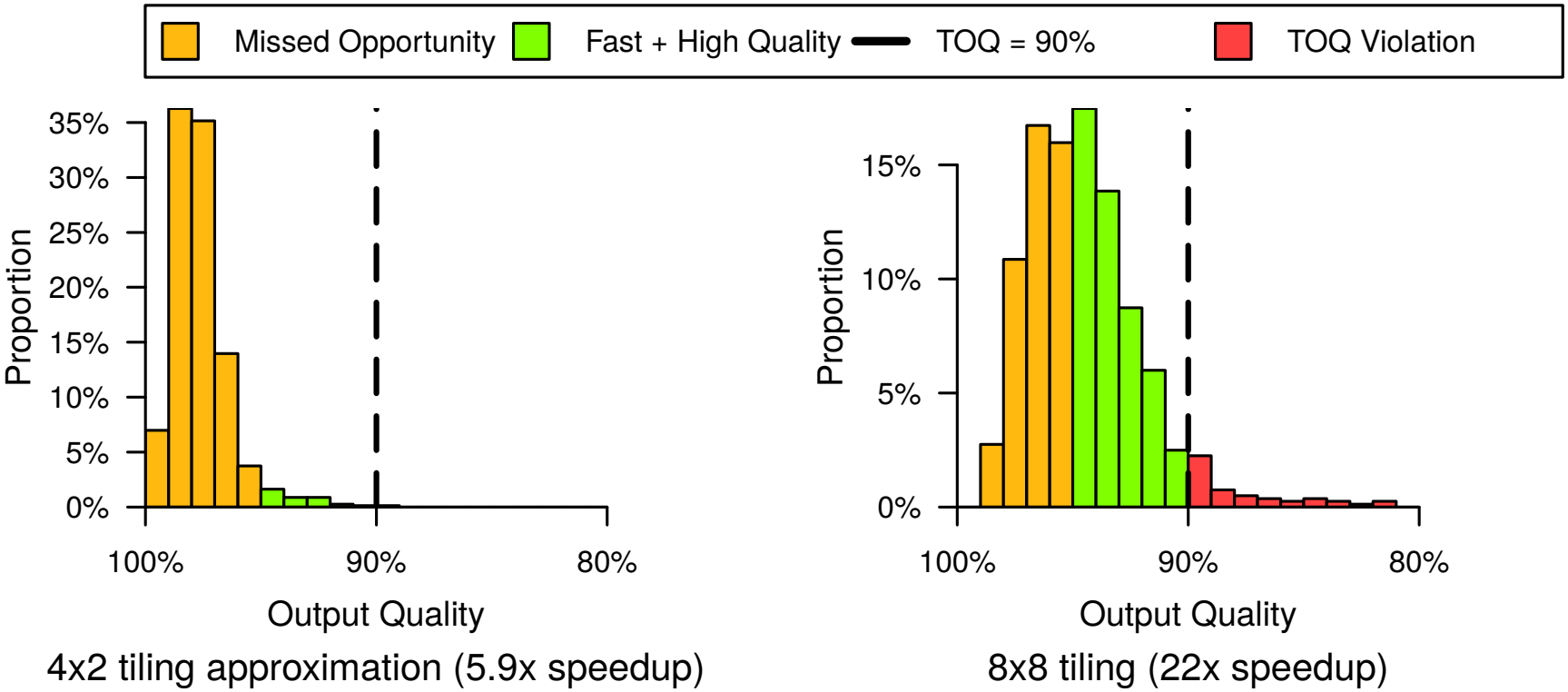
# Trade-off with Many Inputs




4x2 tiling approximation (5.9x speedup)

- Conservative approximation → small speedup

# Trade-off with Many Inputs

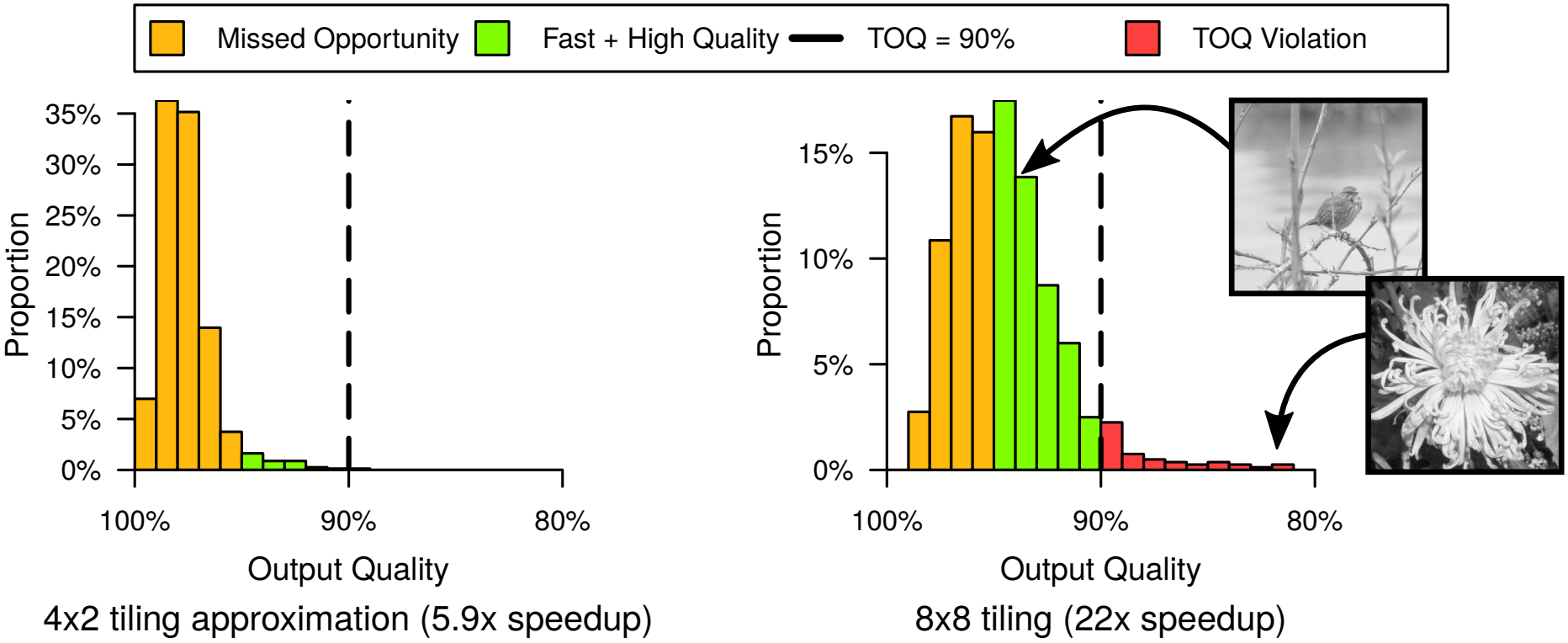



4x2 tiling approximation (5.9x speedup)

8x8 tiling (22x speedup)

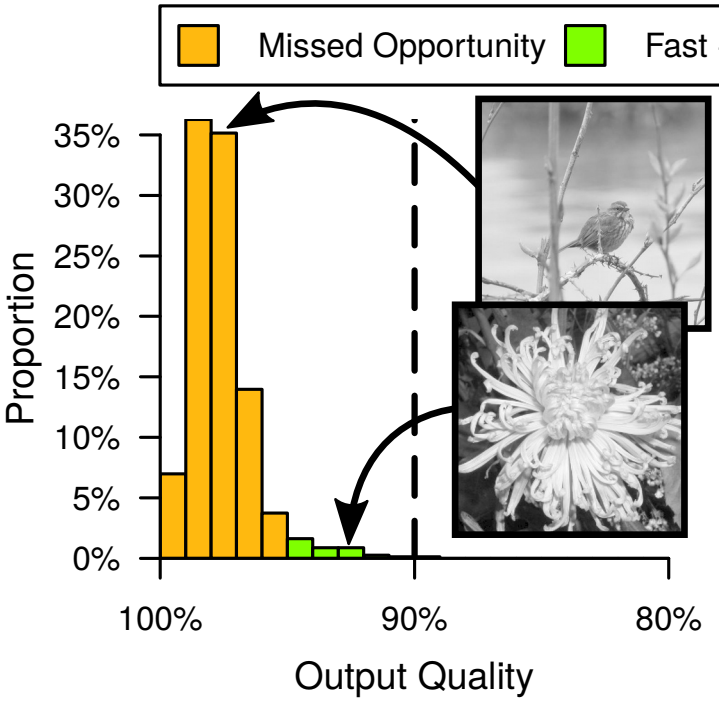
- Conservative approximation → small speedup
- Cannot approximate more aggressively

# Trade-off with Many Inputs

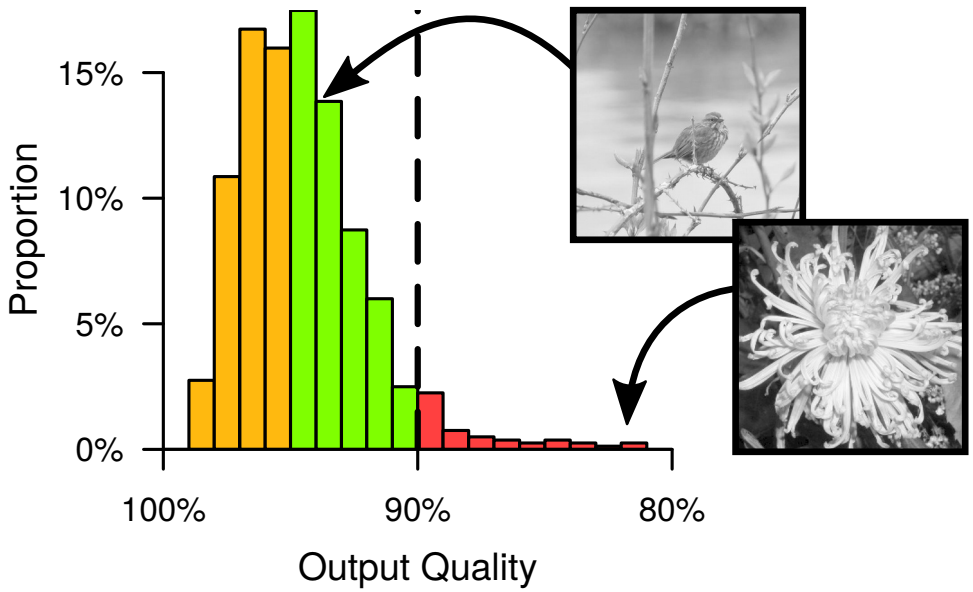



- Conservative approximation → small speedup
- Cannot approximate more aggressively

# Trade-off with Many Inputs




4x2 tiling approximation (5.9x speedup)



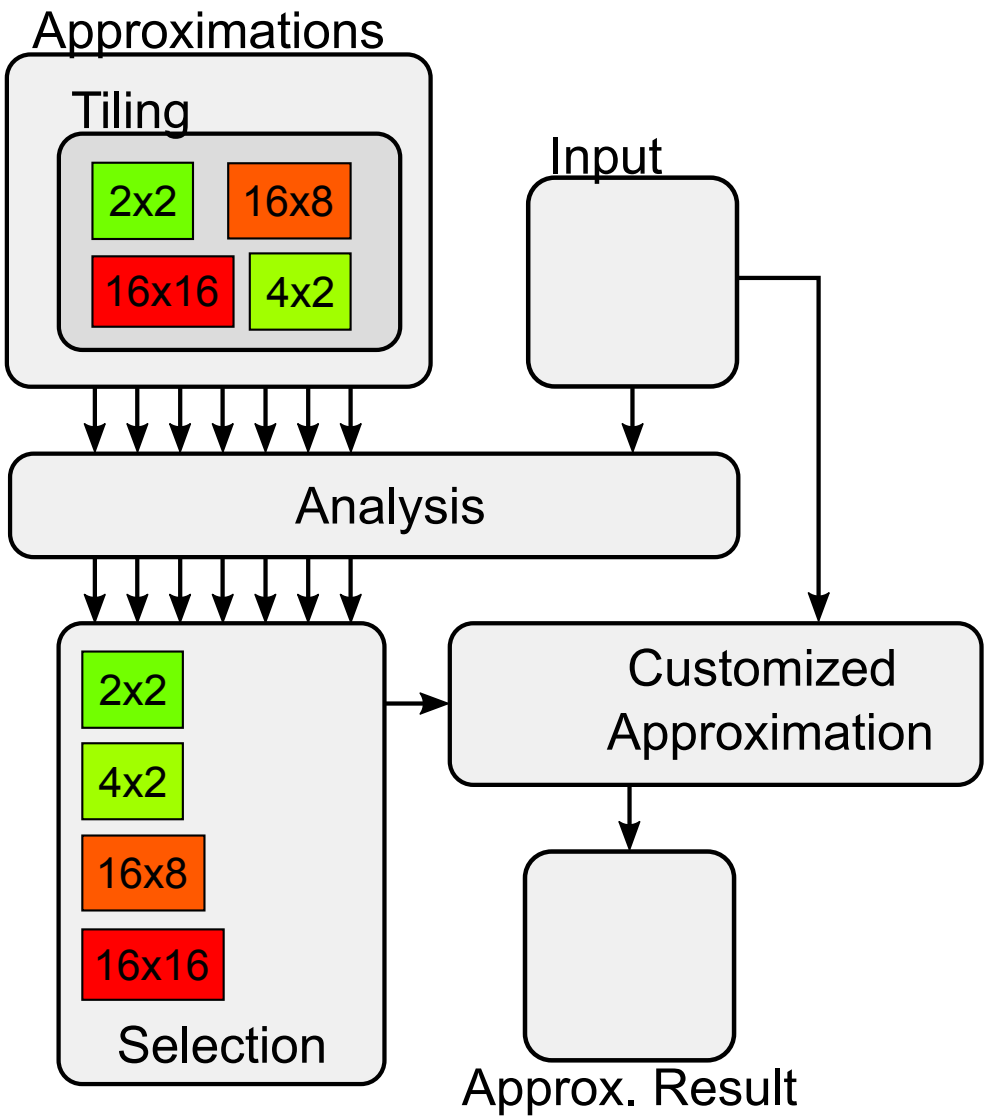

8x8 tiling (22x speedup)

- Conservative approximation → small speedup
- Cannot approximate more aggressively
- We would like to approximate inputs differently

# Dynamic Approximation Challenges

- Must analyze accurately
  - Cannot violate TOQ
  - Need to pick a fast approximation
- Must analyze quickly
  - Limits potential speedup

# One Possible Dynamic System



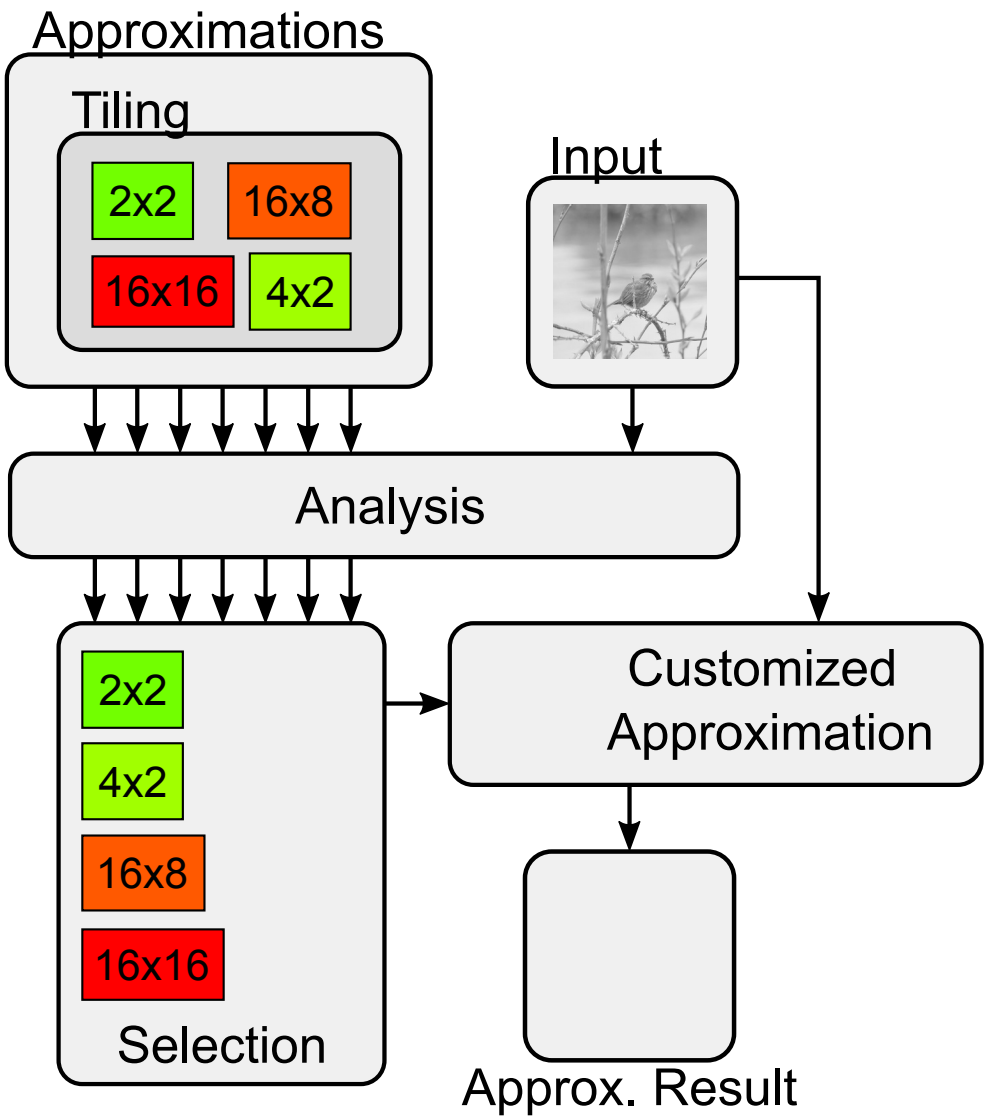

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

4) Apply approximation

# One Possible Dynamic System



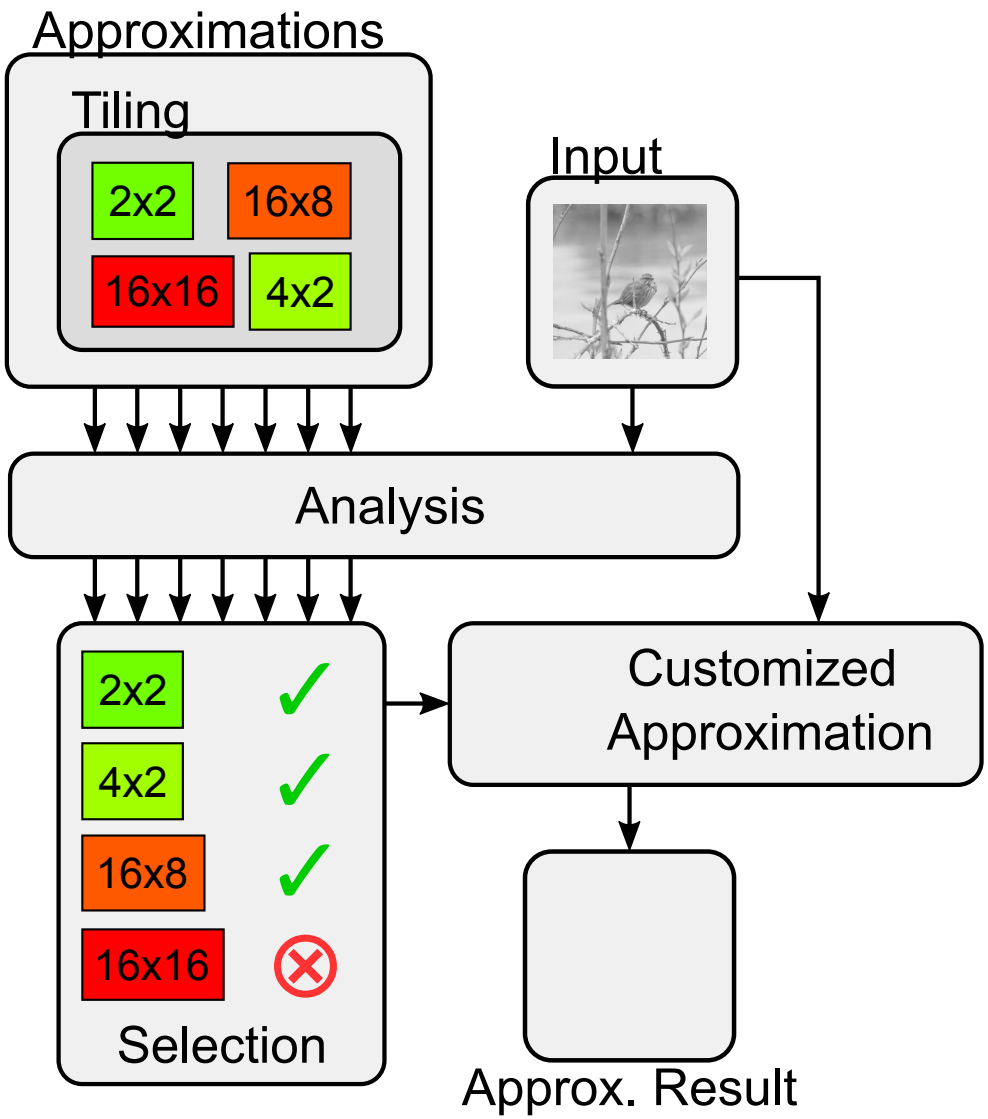

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

4) Apply approximation

# One Possible Dynamic System



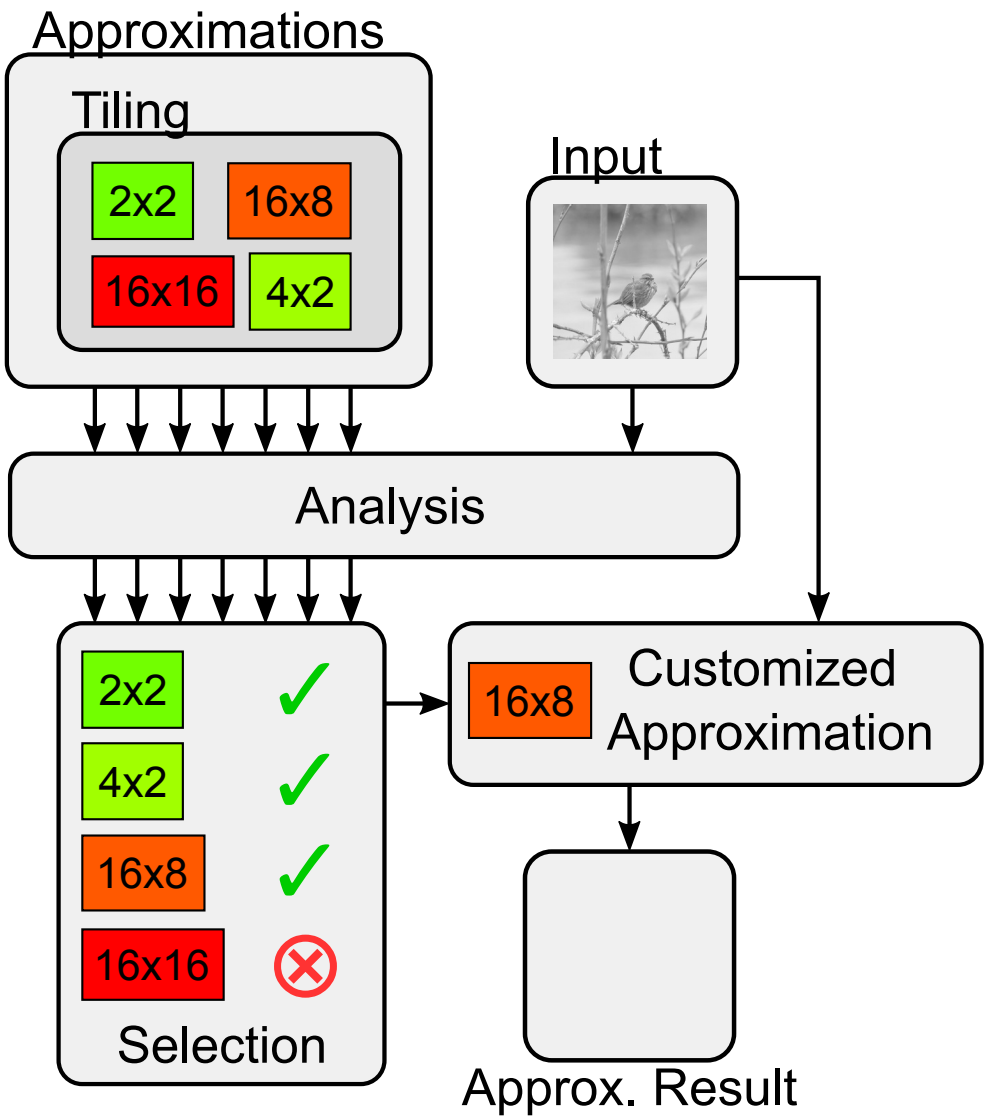

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

4) Apply approximation

# One Possible Dynamic System



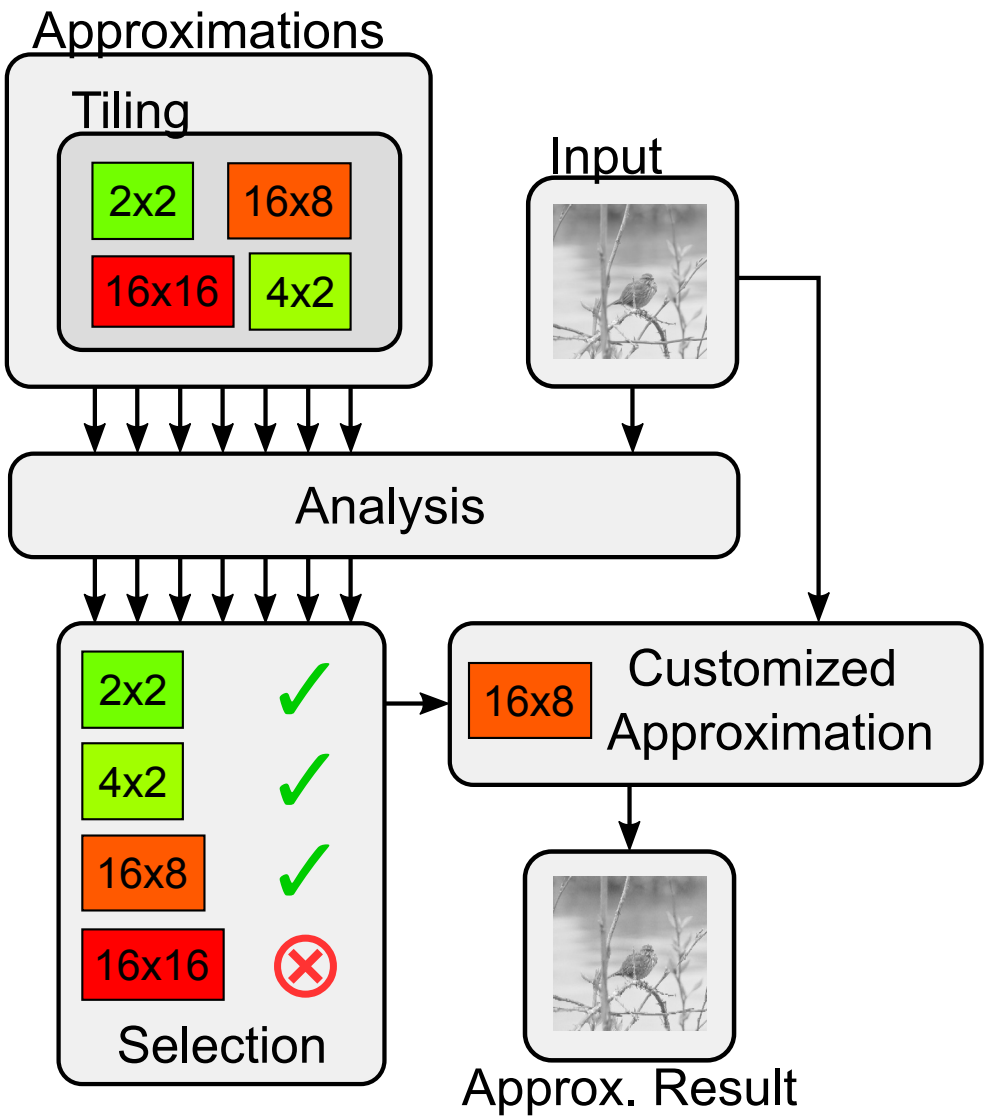

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

4) Apply approximation

# One Possible Dynamic System



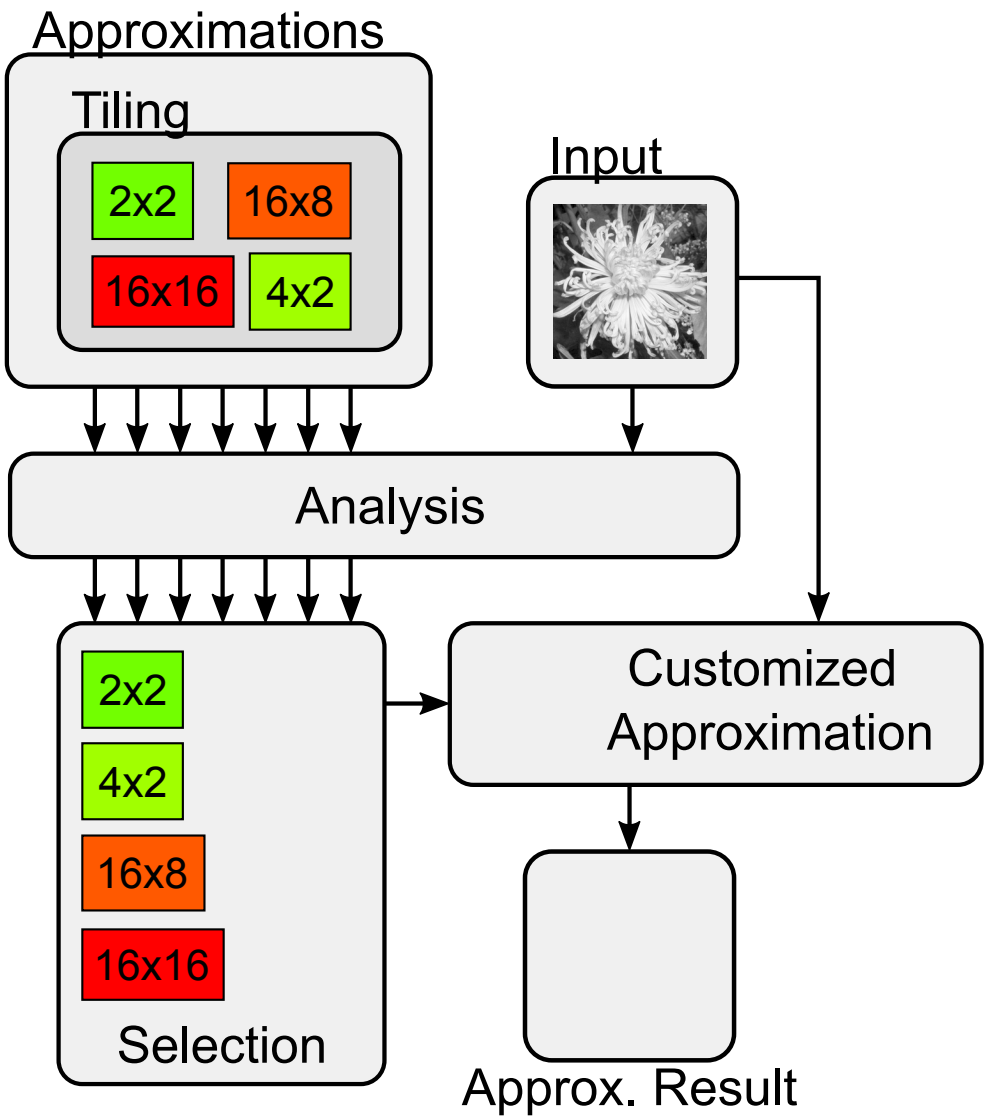

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

4) Apply approximation

# One Possible Dynamic System



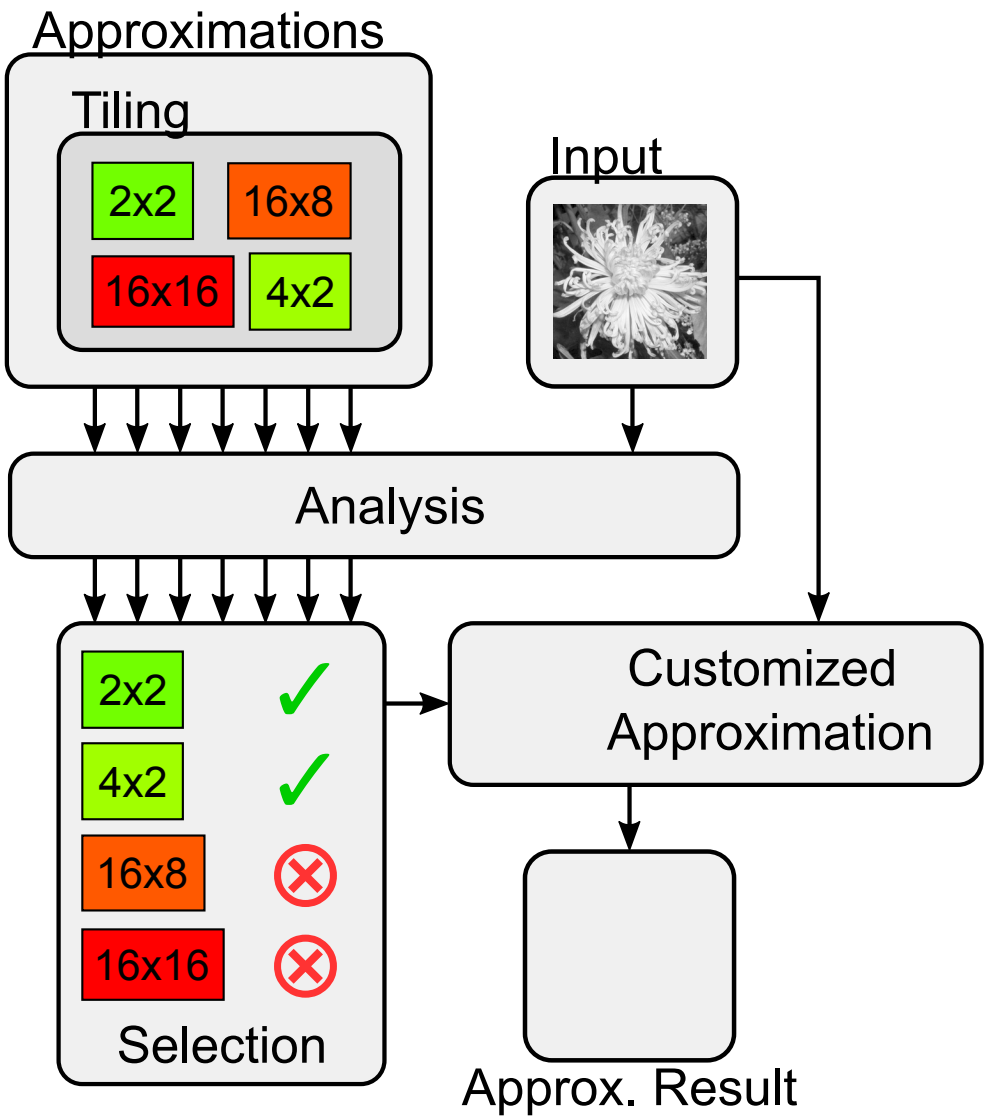

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

4) Apply approximation

# One Possible Dynamic System



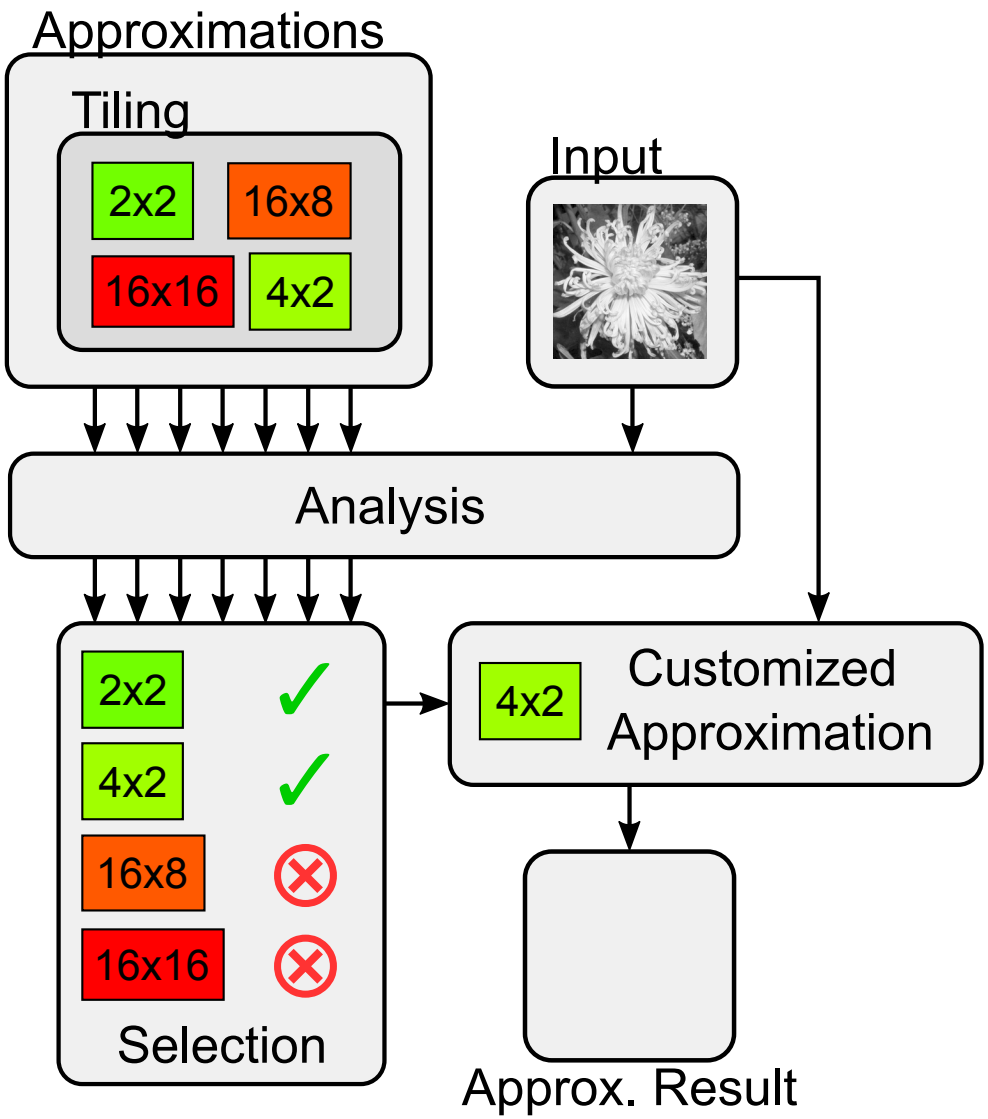

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

4) Apply approximation

# One Possible Dynamic System



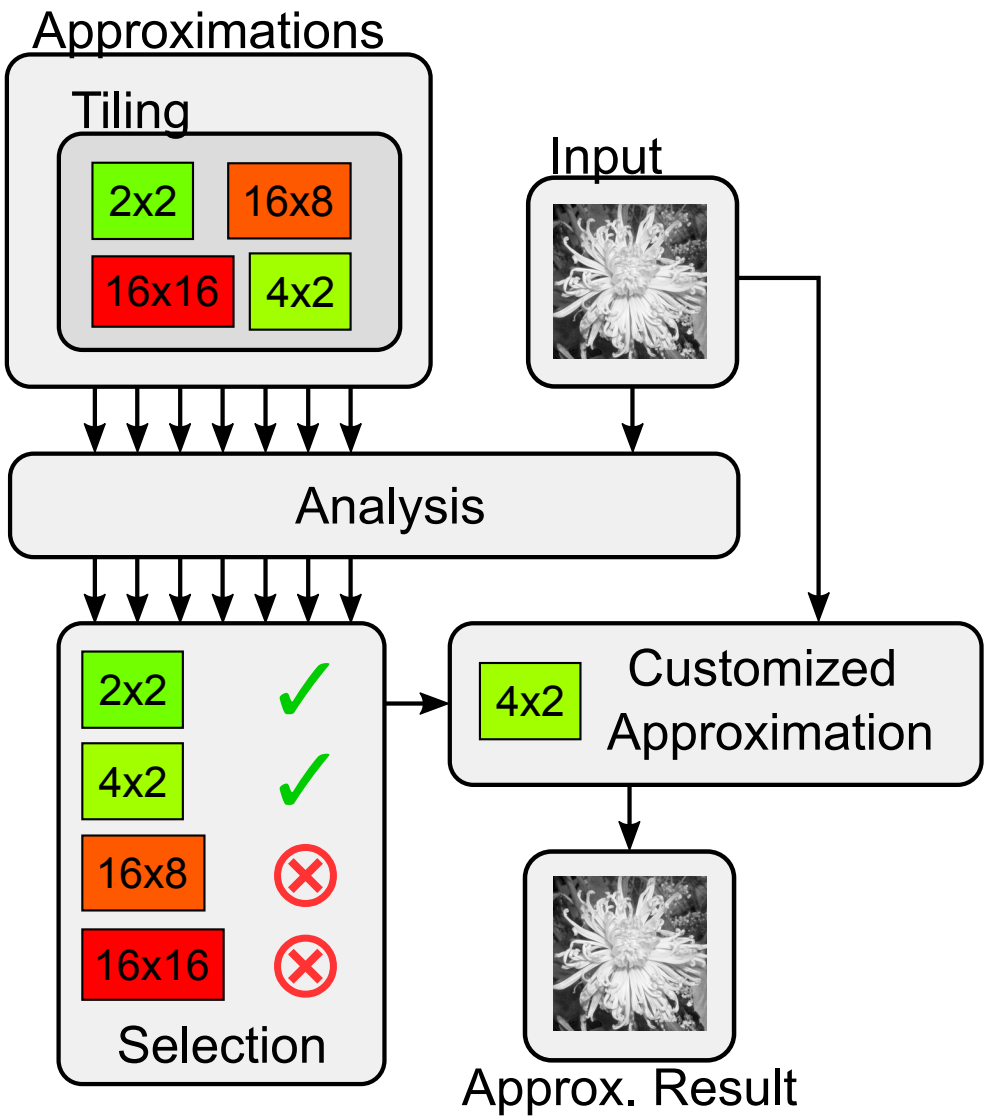

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

4) Apply approximation

# One Possible Dynamic System



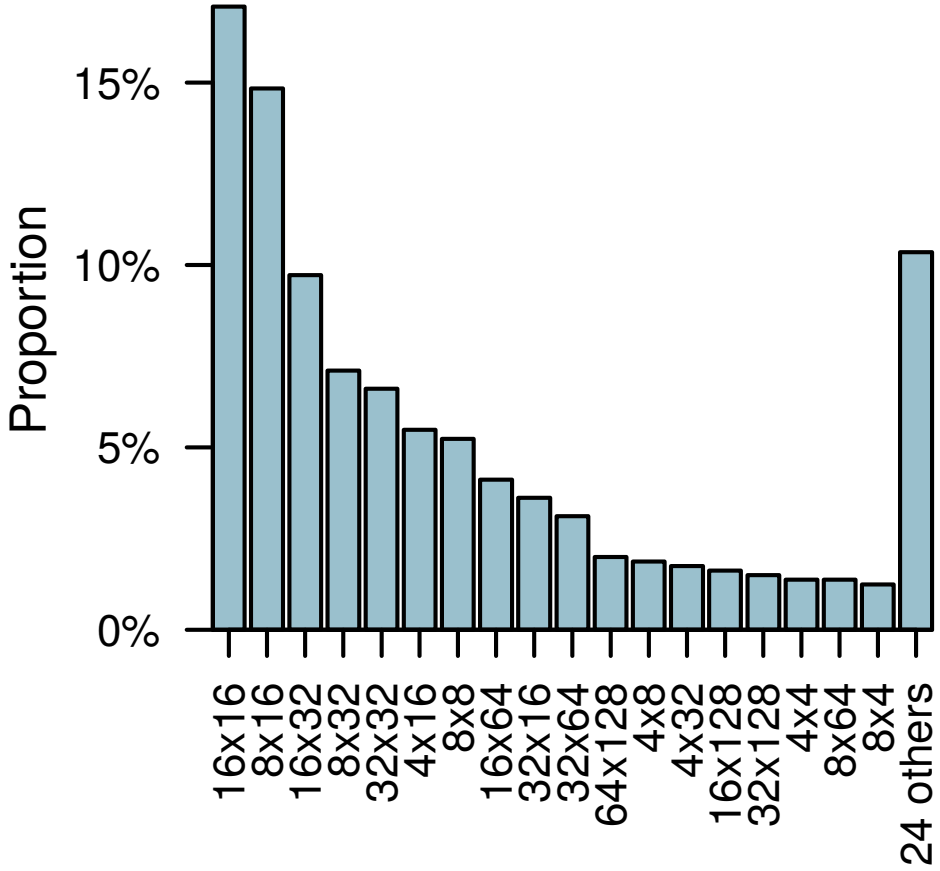

1) Provide:
   - A set of approximations
   - Input

2) Apply analysis to each pair:
   - Performance
   - Output quality

3) Select best approximation:
   - Meets accuracy constraint
   - High performance

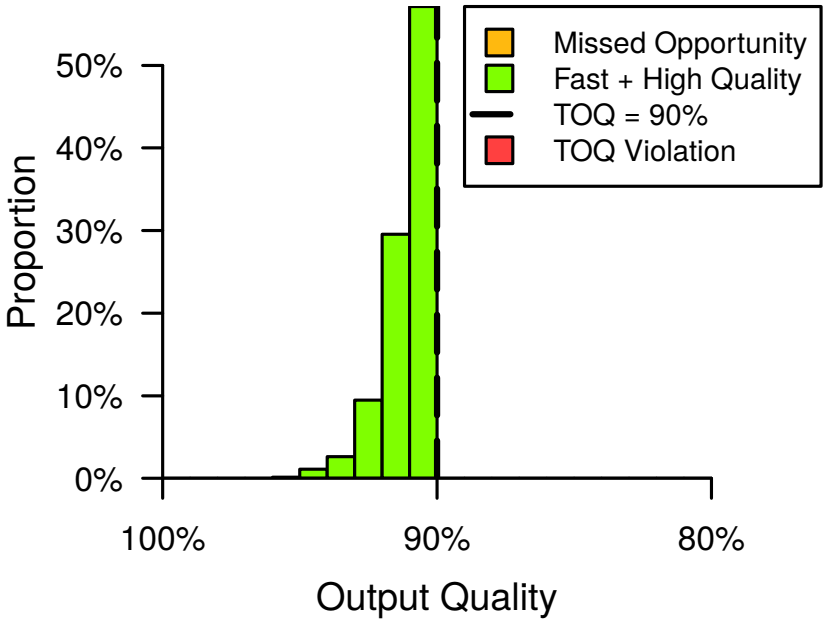
4) Apply approximation

# Dynamic Oracle Selections



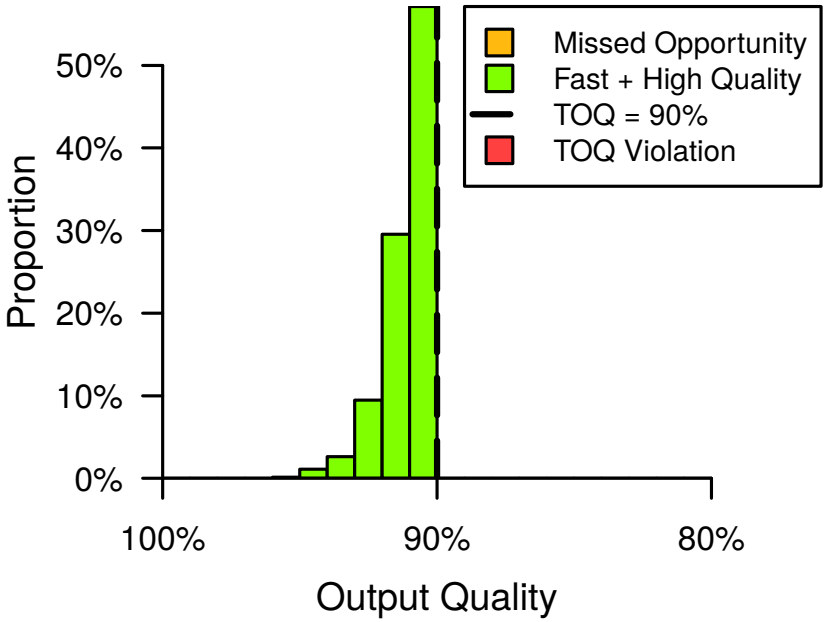

- Optimal choice depends heavily on input

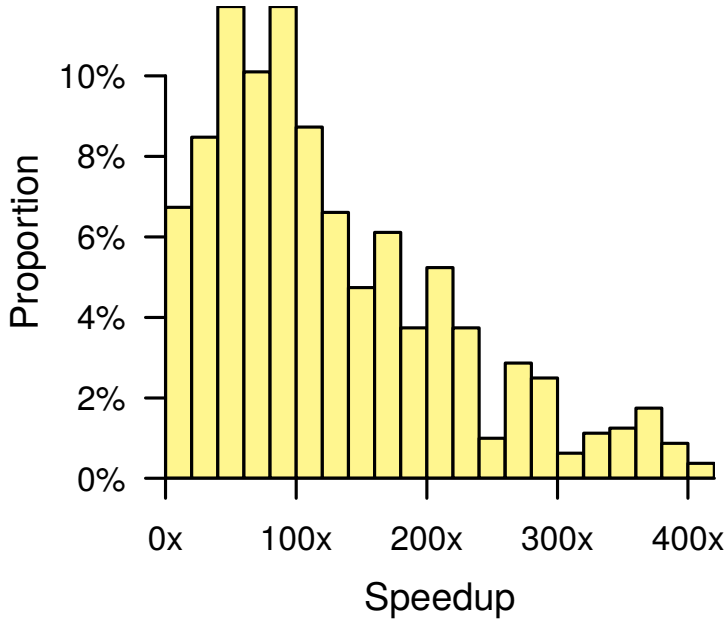
# Dynamic Oracle Performance




- Accuracy near TOQ

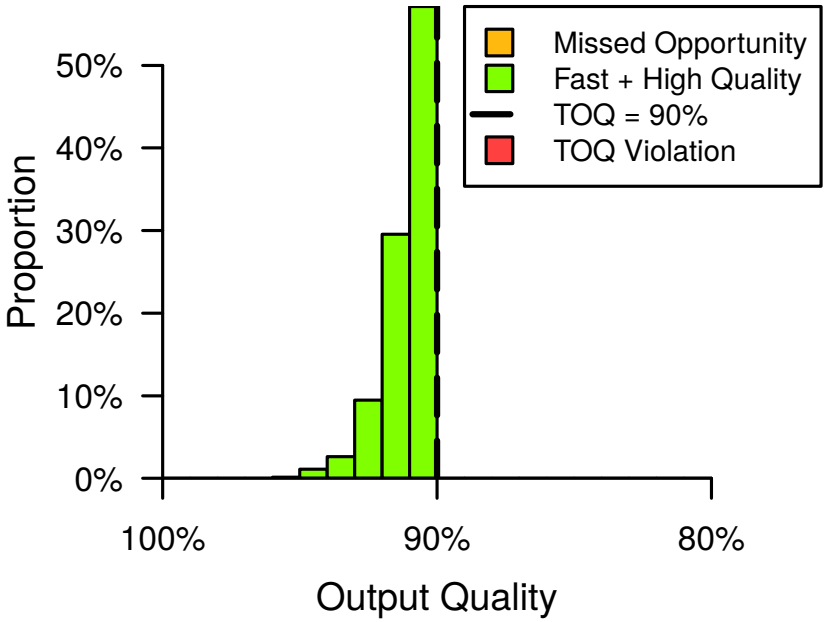
# Dynamic Oracle Performance

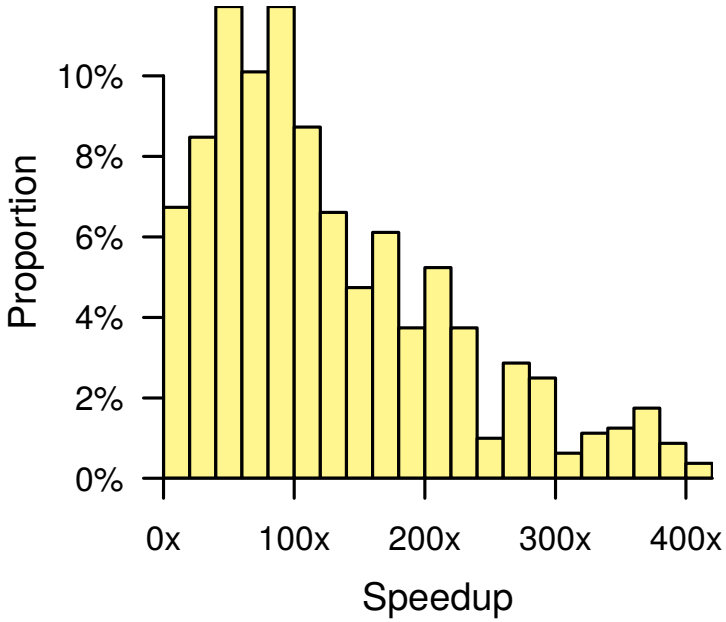


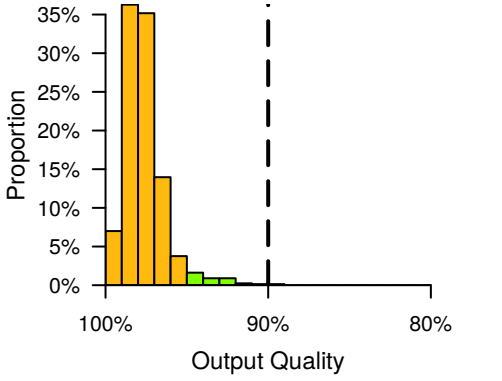




- Accuracy near TOQ
- 61x average speedup

# Dynamic Oracle Performance




4x2 tiling approximation (5.9x speedup)

- Accuracy near TOQ
- 61x average speedup (compared to 5.9x for 4x2 tiling)

# Conclusion

- Adjusting approximation per input is important
  - 61x potential speedup for dynamic system
  - 5.9x potential speedup for static system
- To take advantage of this opportunity:
  - Dynamic system predicts approximation per input
  - High prediction accuracy
  - Quick predictions

# Questions?