Input Responsiveness: Using Canary Inputs to Dynamically Steer Approximation

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

ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)
June 15, 2016

CSE COMPUTER SCIENCE AND ENGINEERING UNIVERSITY OF MICHIGAN

ClarityLab

Approximate Computing

Approximate Computing

Approximate Computing

- Trade small losses in accuracy for big gains in performance/energy

Approximate Computing

- Trade small losses in accuracy for big gains in performance/energy
- Why approximation?
  - Machine learning, data mining, image, video and sound processing, statistics, graphics
  - Loose constraints on output quality, growing computational demands

Approximation Runtime Systems

Approximation Runtime Systems

- Central question — *how to approximate?*

Approximation Runtime Systems

- Central question — *how to approximate?*
  - How to parameterize approximation

Approximation Runtime Systems

- Central question — *how to approximate?*
  - How to parameterize approximation
  - Where in the application to approximate

The Importance of Input

The Importance of Input

- **Insight** – input is a key component in answering this question

The Importance of Input

- **Insight** – input is a key component in answering this question

- Example – gamma correction

The Importance of Input

- **Insight** – input is a key component in answering this question
- Example – gamma correction

input

The Importance of Input

- **Insight** – input is a key component in answering this question
- Example – gamma correction

input

exact output

The Importance of Input

- **Insight** – input is a key component in answering this question
- Example – gamma correction

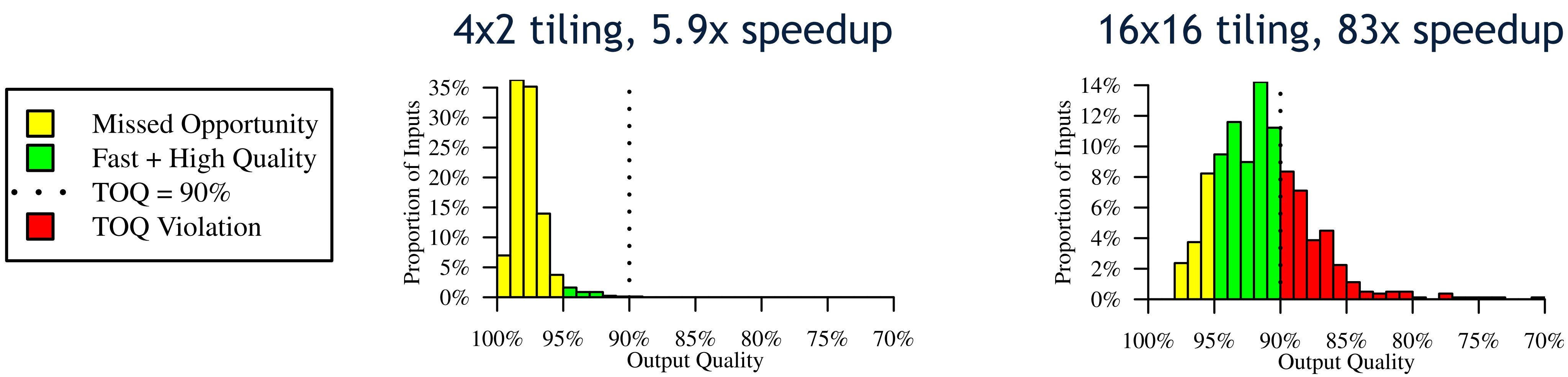
Why Input Responsiveness?

Why Input Responsiveness?

- Example: gamma correction, 2 tiling approximations, 800 inputs

Why Input Responsiveness?

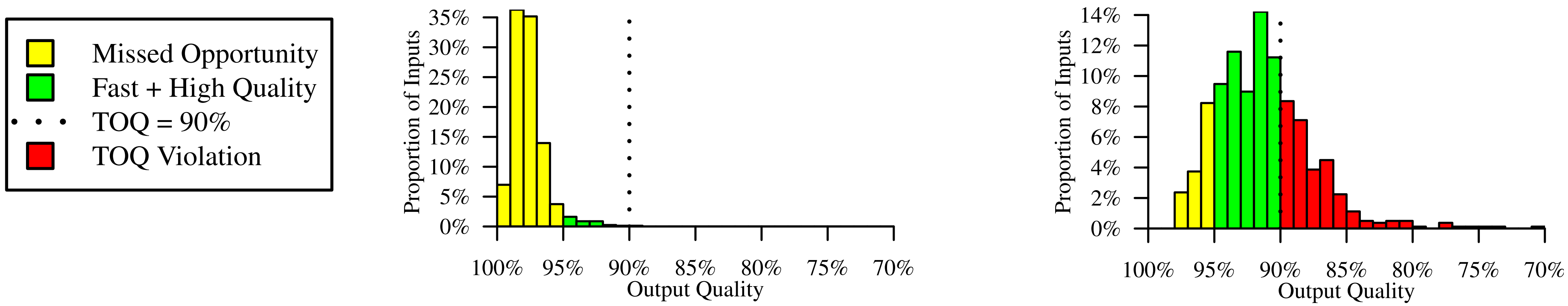
- Example: gamma correction, 2 tiling approximations, 800 inputs



Why Input Responsiveness?

- Example: gamma correction, 2 tiling approximations, 800 inputs

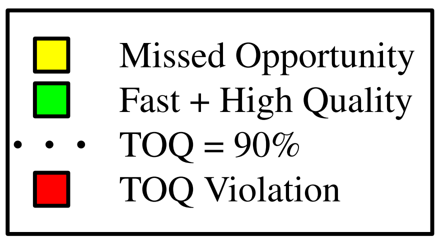
4x2 tiling, 5.9x speedup　　　16x16 tiling, 83x speedup



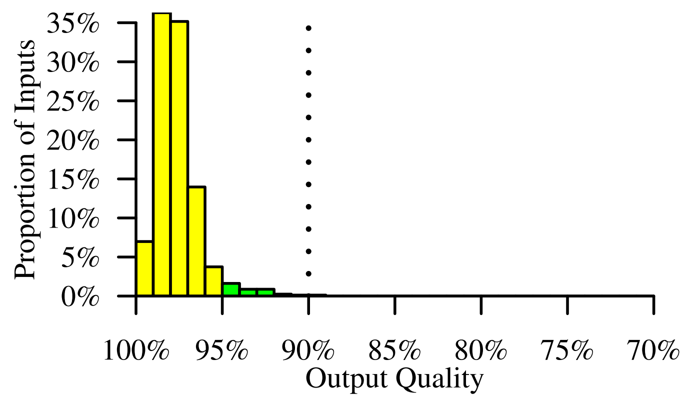
- What if we take full advantage of the differences among inputs?

Why Input Responsiveness?

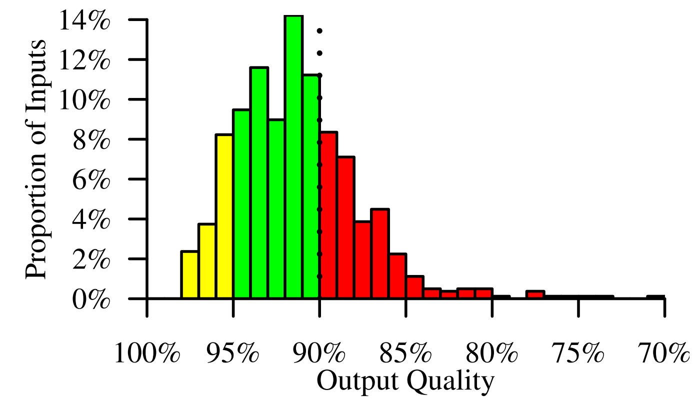
- Example: gamma correction, 2 tiling approximations, 800 inputs



4x2 tiling, 5.9x speedup

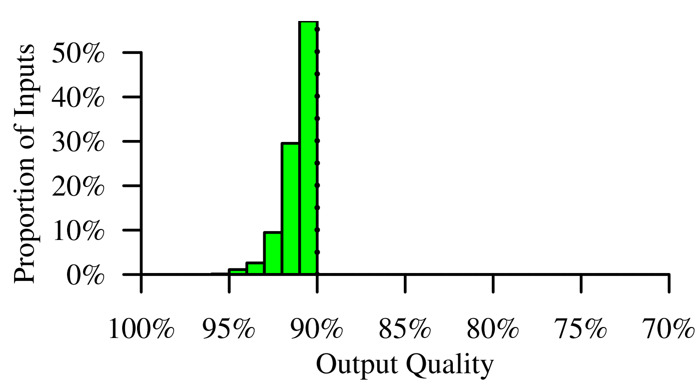


16x16 tiling, 83x speedup

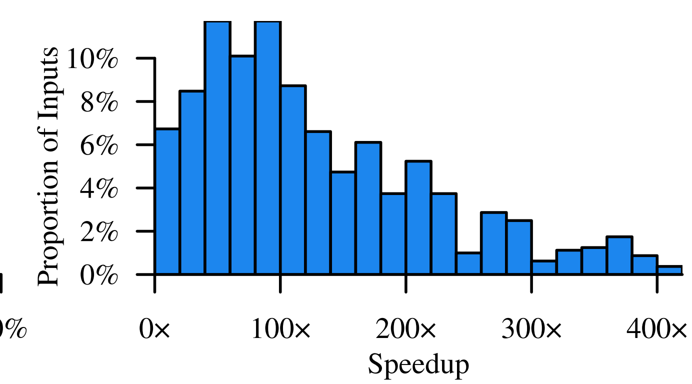


- What if we take full advantage of the differences among inputs?

High output quality



61x speedup



Achieving Input Responsiveness

Achieving Input Responsiveness

- **Goal** — end-to-end runtime system to choose *how to approximate*, customized for each input
  - Maximize performance given an accuracy target

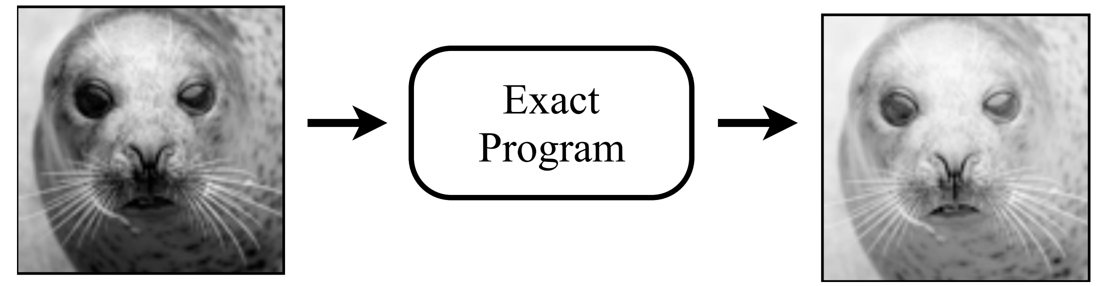
Achieving Input Responsiveness

- **Goal** — end-to-end runtime system to choose *how to approximate*, customized for each input
  - Maximize performance given an accuracy target
- **Challenges**
  - Choose highly effective approximation per input
  - Choose it quickly!

Input Responsive Approximation (IRA)

Input Responsive Approximation (IRA)

exact execution



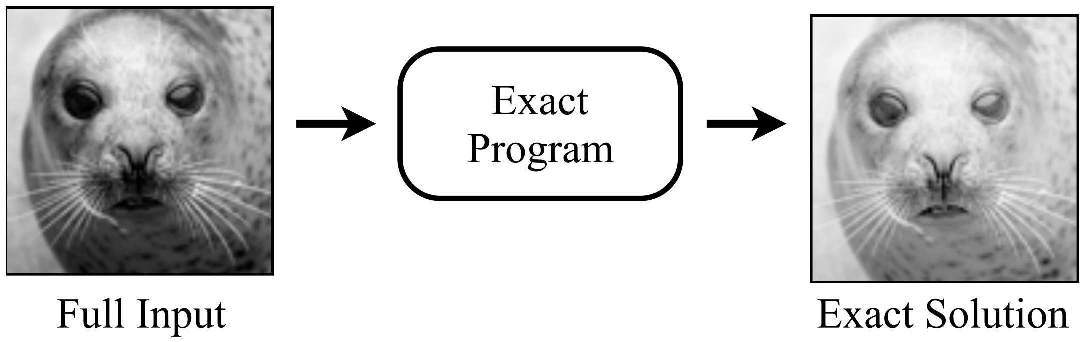
Full Input

Exact Solution

Input Responsive Approximation (IRA)

- Create a small *canary input* from full input

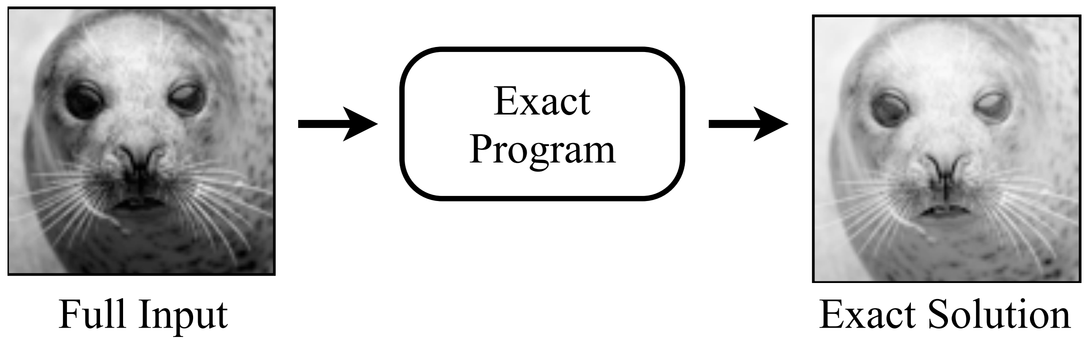
exact execution



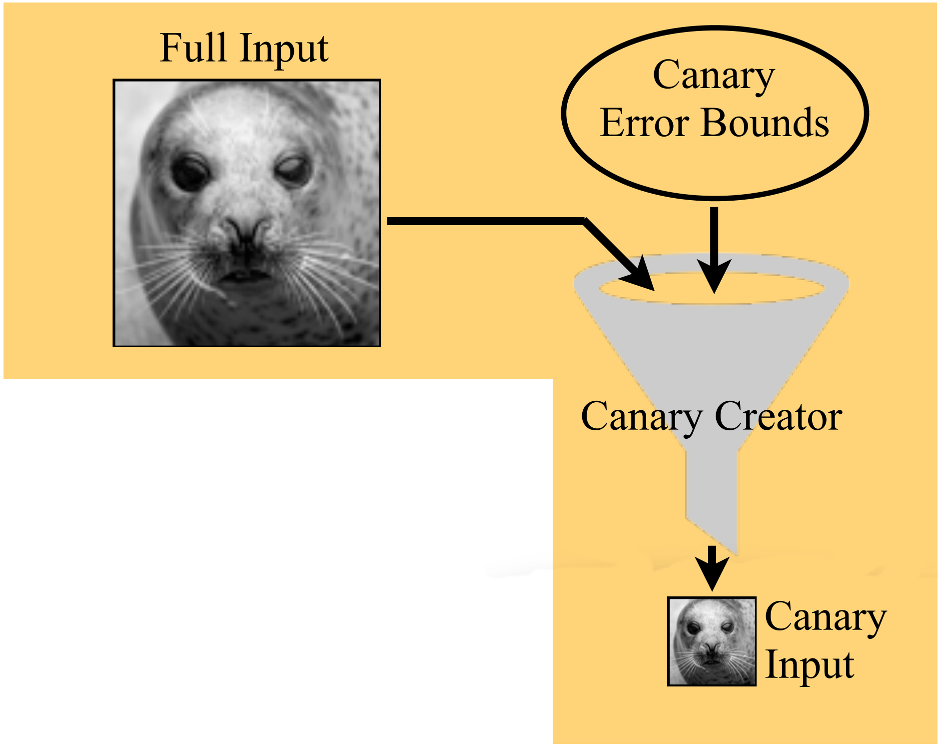
Input Responsive Approximation (IRA)

- Create a small *canary input* from full input

exact execution



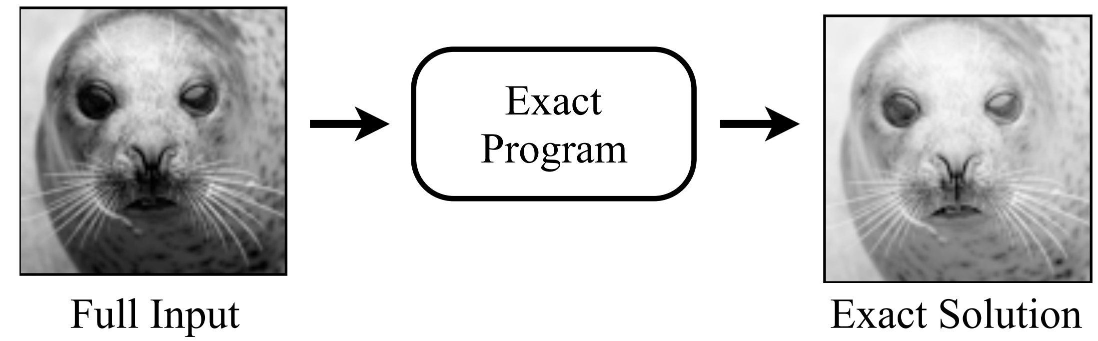
approximation with IRA



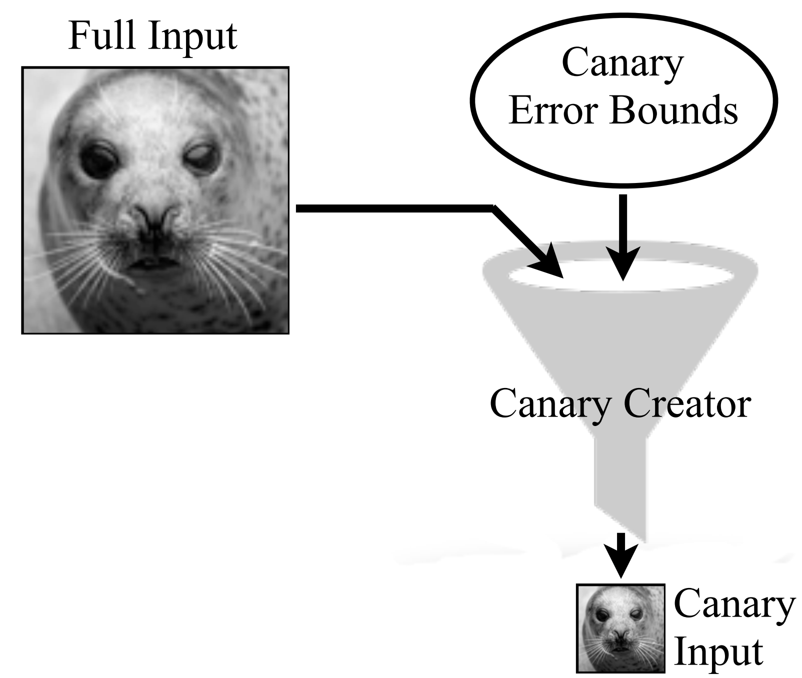
Input Responsive Approximation (IRA)

- Create a small *canary input* from full input
- Choose an approximation using canary

exact execution



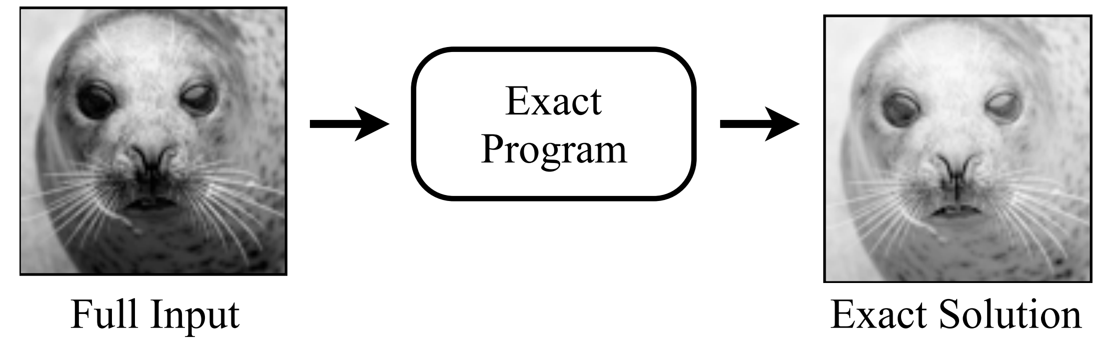
approximation with IRA



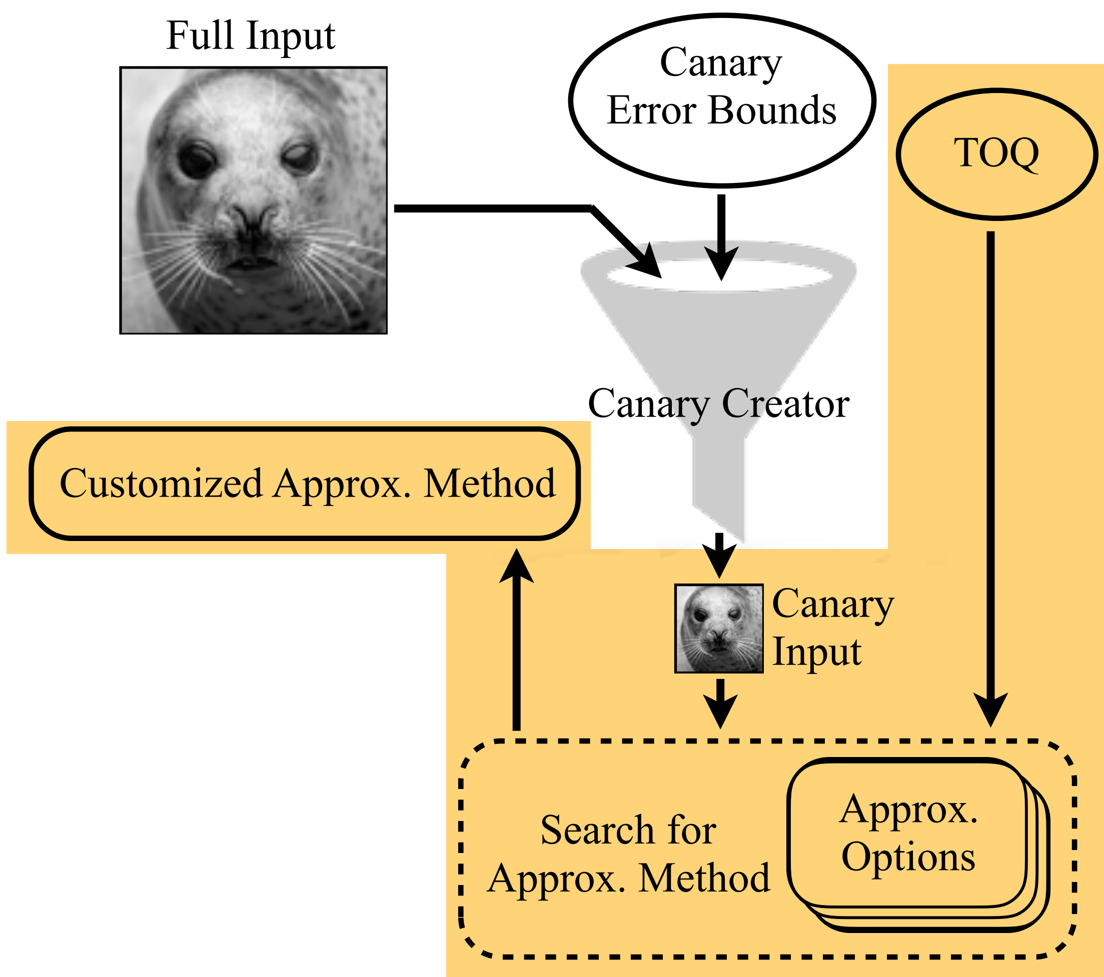
Input Responsive Approximation (IRA)

- Create a small *canary input* from full input
- Choose an approximation using canary

exact execution



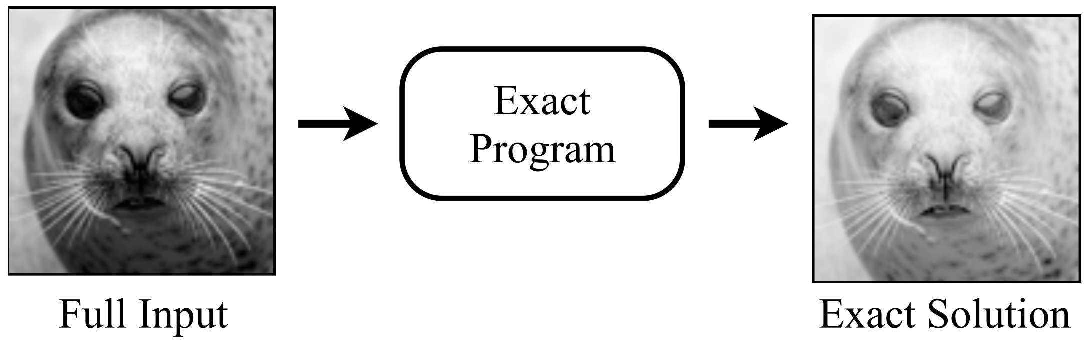
approximation with IRA



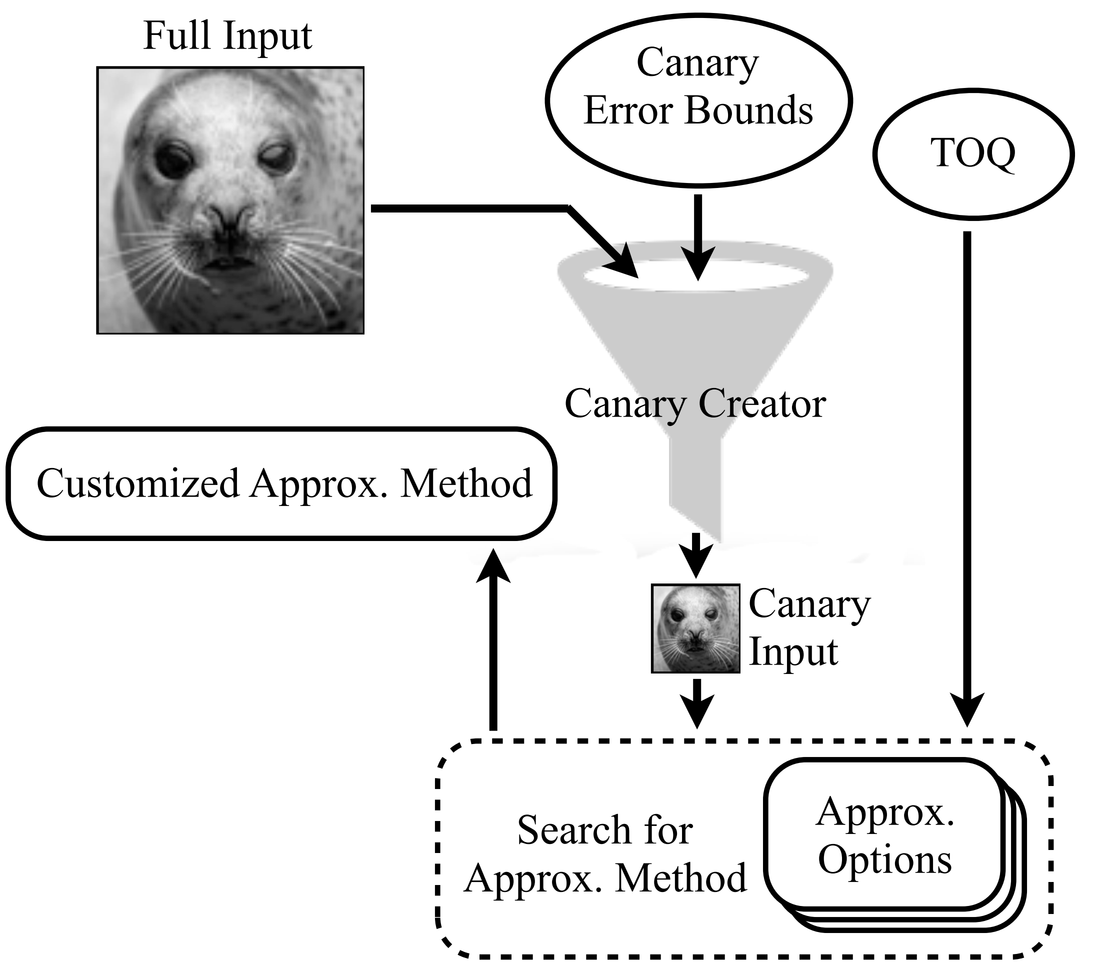
Input Responsive Approximation (IRA)

- Create a small *canary input* from full input
- Choose an approximation using canary
- Compute approximate solution on full input

exact execution



approximation with IRA



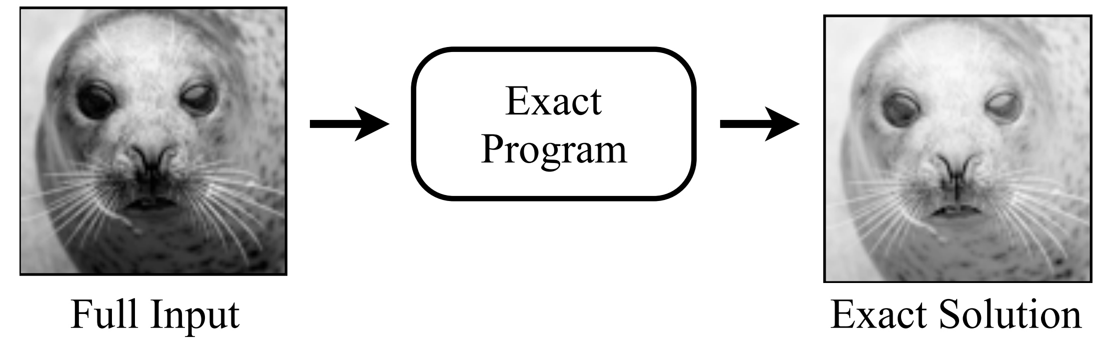
COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

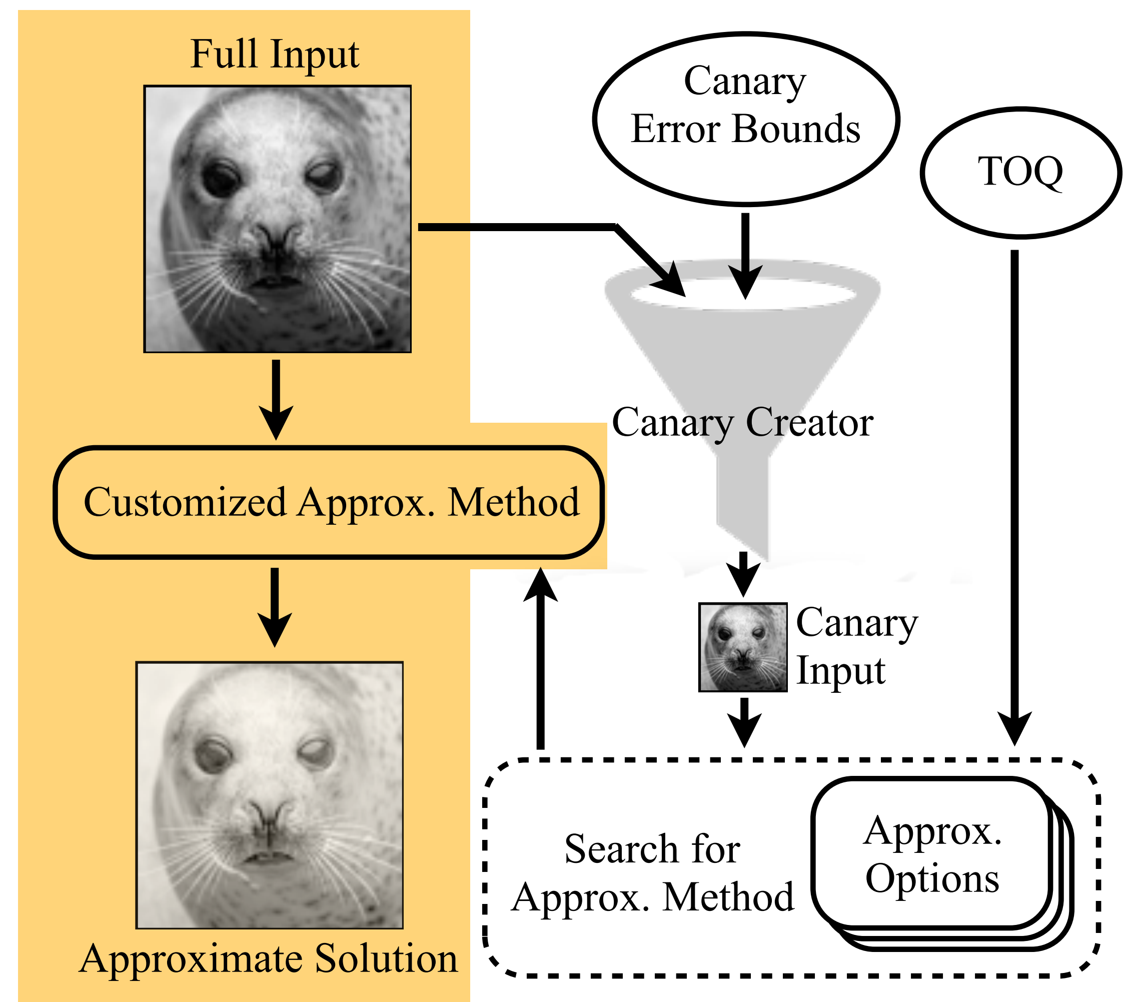
Input Responsive Approximation (IRA)

- Create a small *canary input* from full input
- Choose an approximation using canary
- Compute approximate solution on full input

exact execution



approximation with IRA



COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

Goals of Canary Creation in IRA

Goals of Canary Creation in IRA

- **Similarity** – sufficient similarity to full input

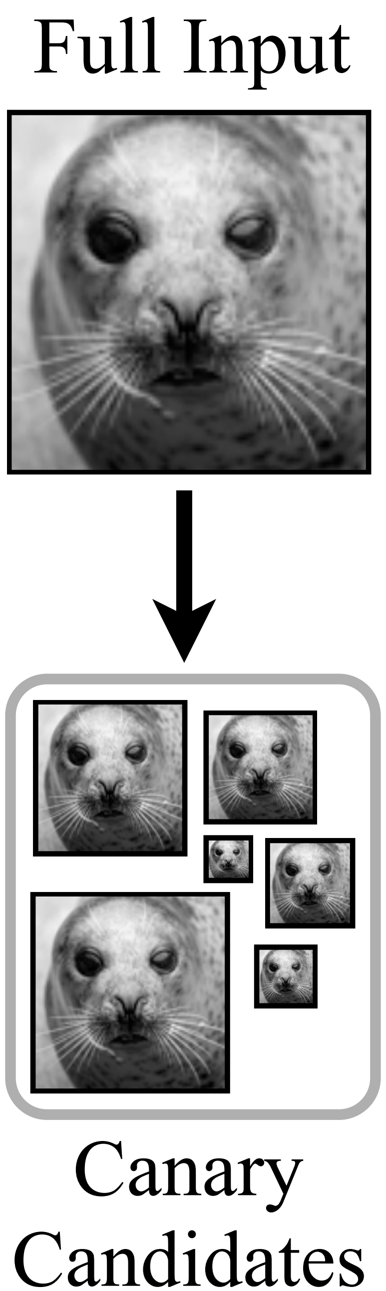
Goals of Canary Creation in IRA

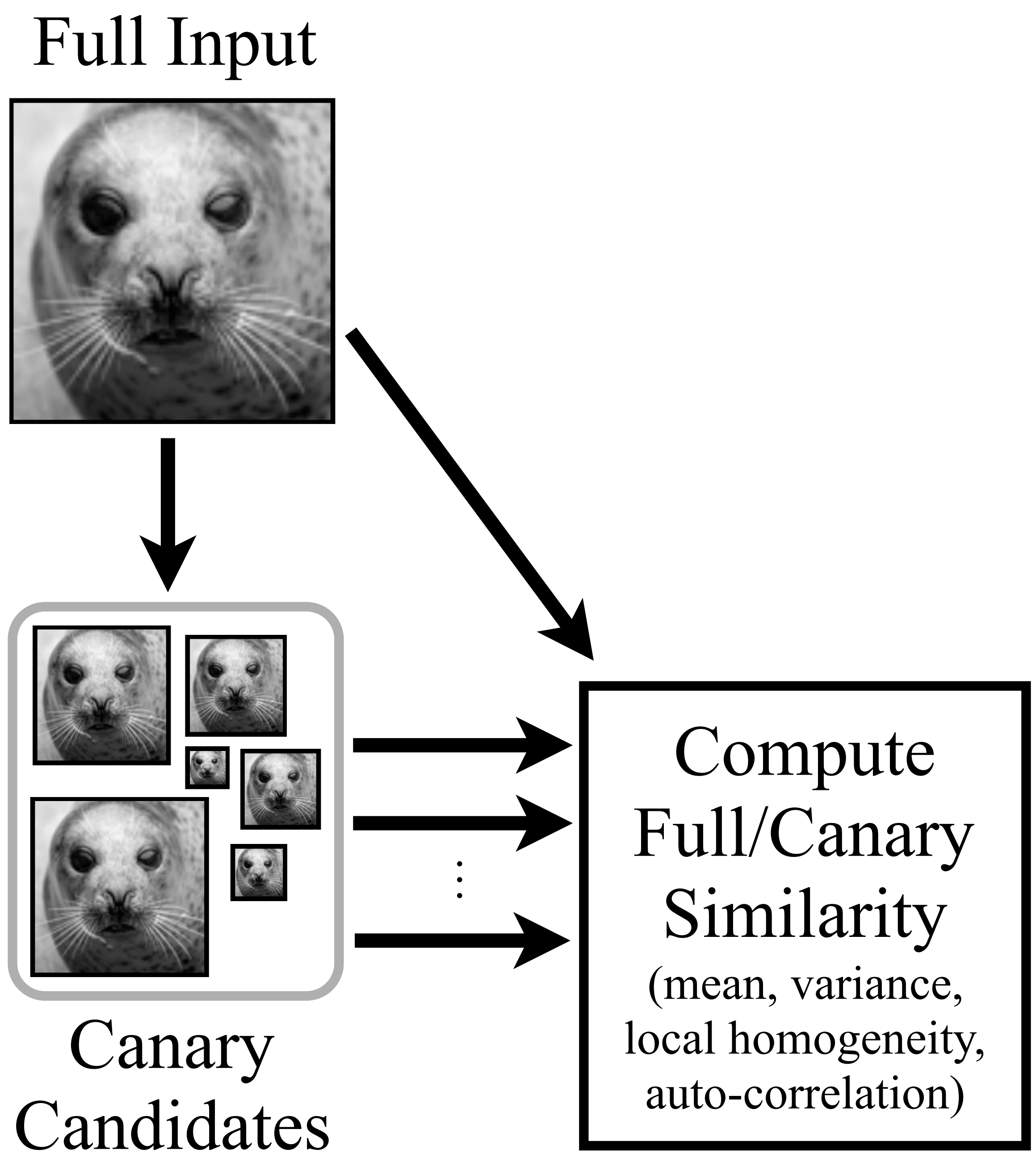
- **Similarity** – sufficient similarity to full input
- **Inexpensive Computation** – create it quickly

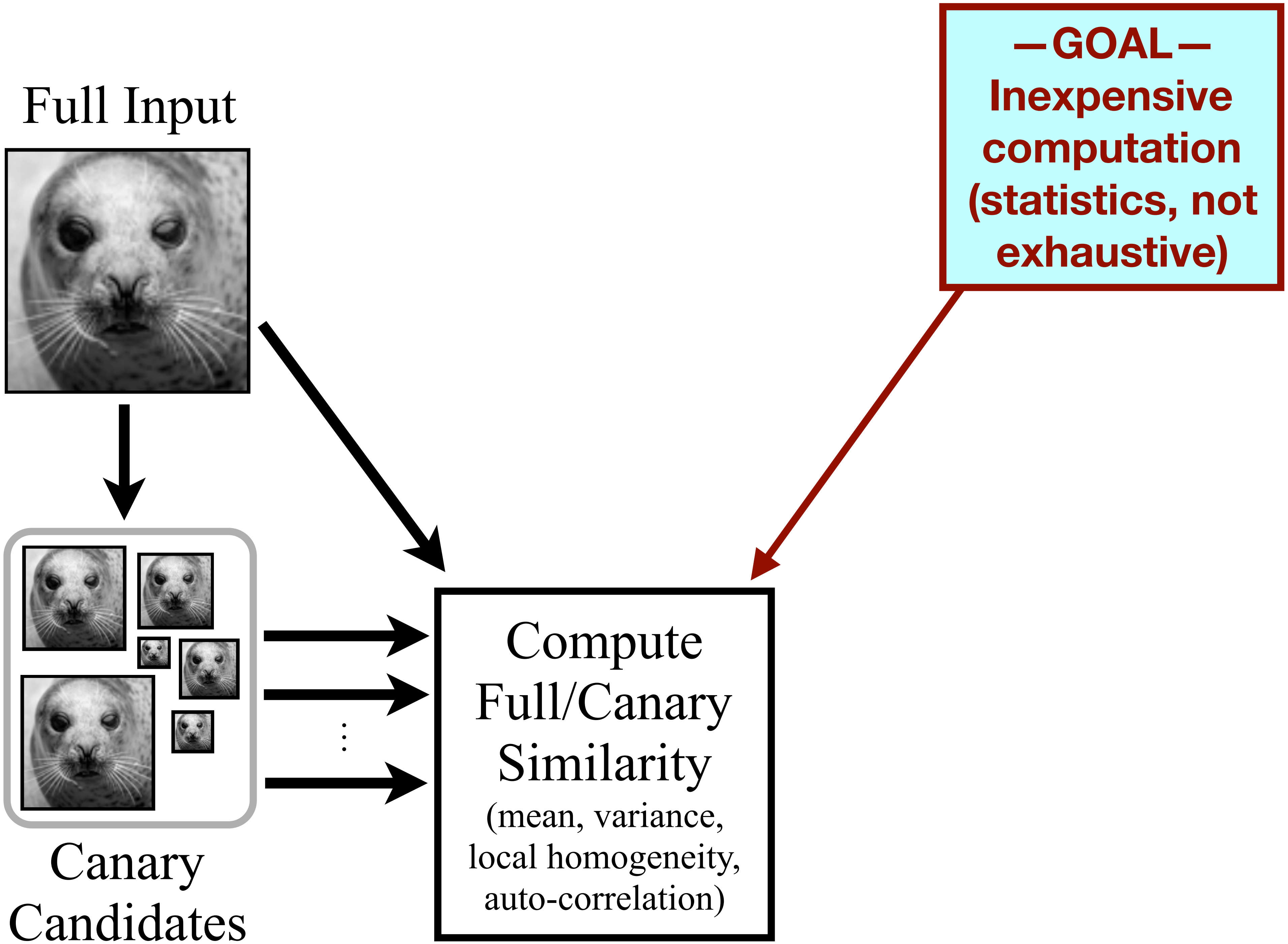
Goals of Canary Creation in IRA

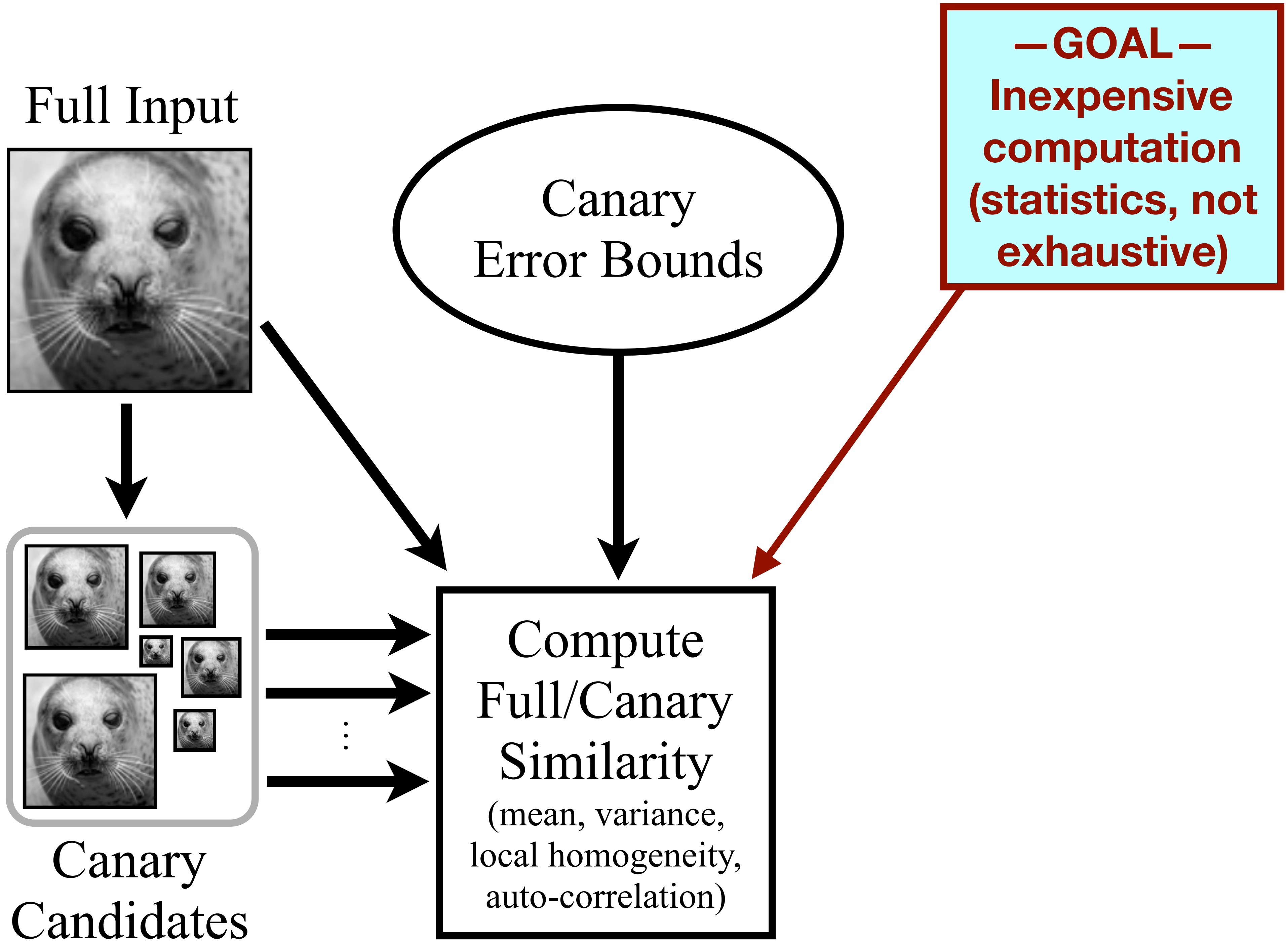
- **Similarity** – sufficient similarity to full input
- **Inexpensive Computation** – create it quickly
- **Size Reduction**
    - Computational complexity often depends on input size
    - Canary should be as small as possible

Full Input

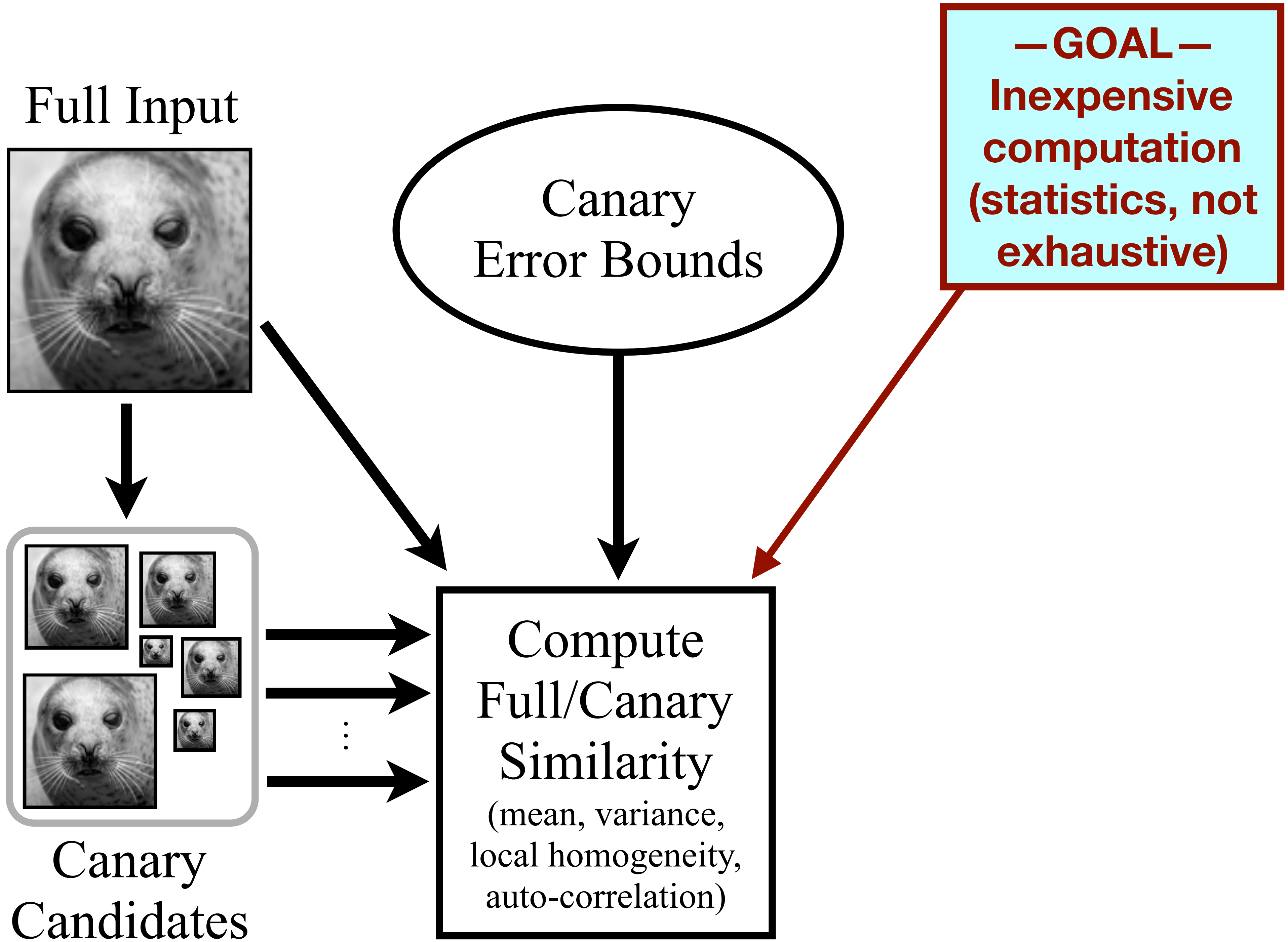

Full Input
Canary
Candidates


Full Input
Canary
Candidates
Compute
Full/Canary
Similarity
(mean, variance,
local homogeneity,
auto-correlation)

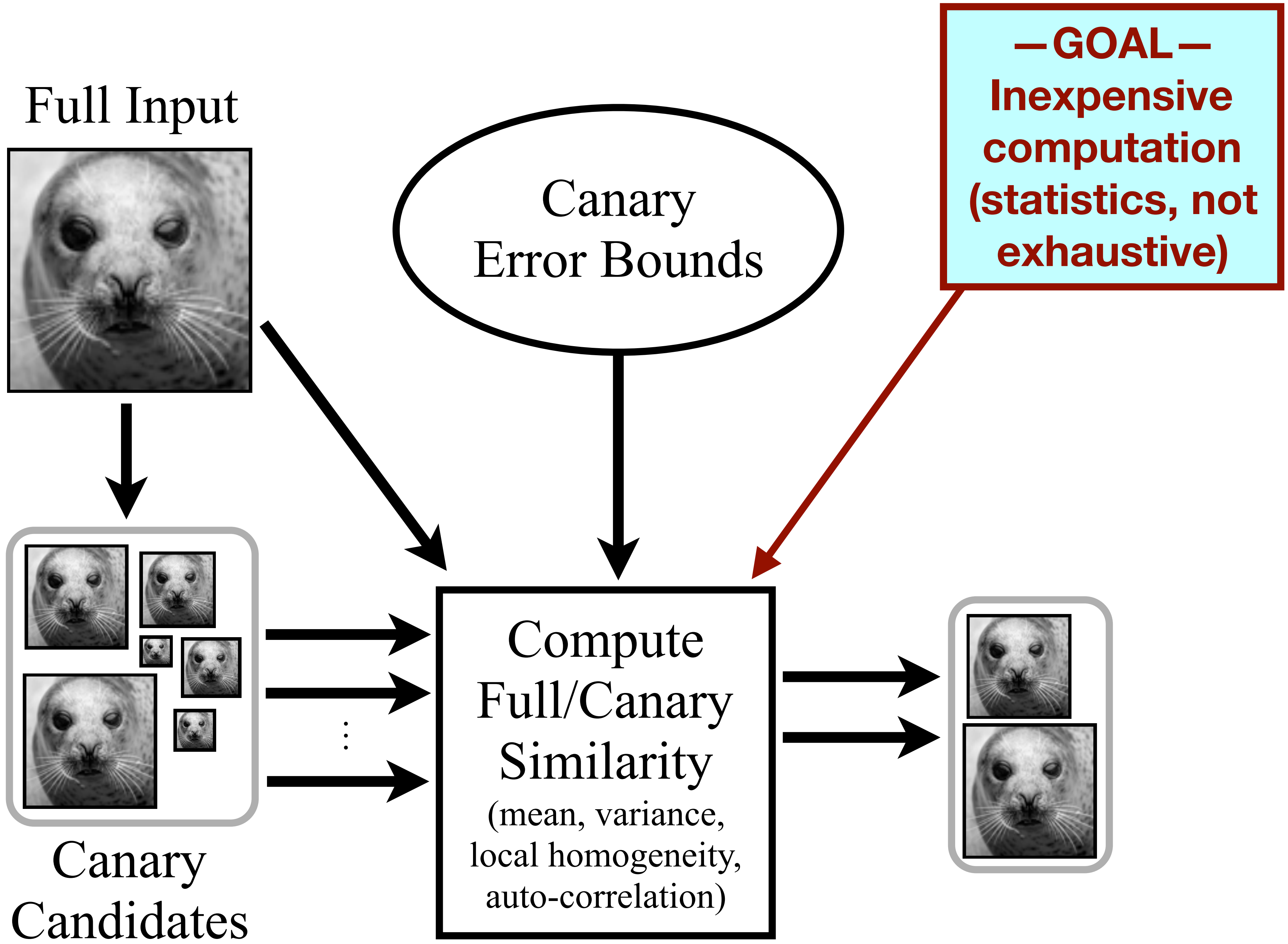

—GOAL—
Inexpensive computation (statistics, not exhaustive)
Full Input
Canary Candidates
Compute Full/Canary Similarity
(mean, variance, local homogeneity, auto-correlation)


Full Input
Canary Error Bounds
—GOAL—
Inexpensive computation (statistics, not exhaustive)
Canary Candidates
Compute Full/Canary Similarity
(mean, variance, local homogeneity, auto-correlation)

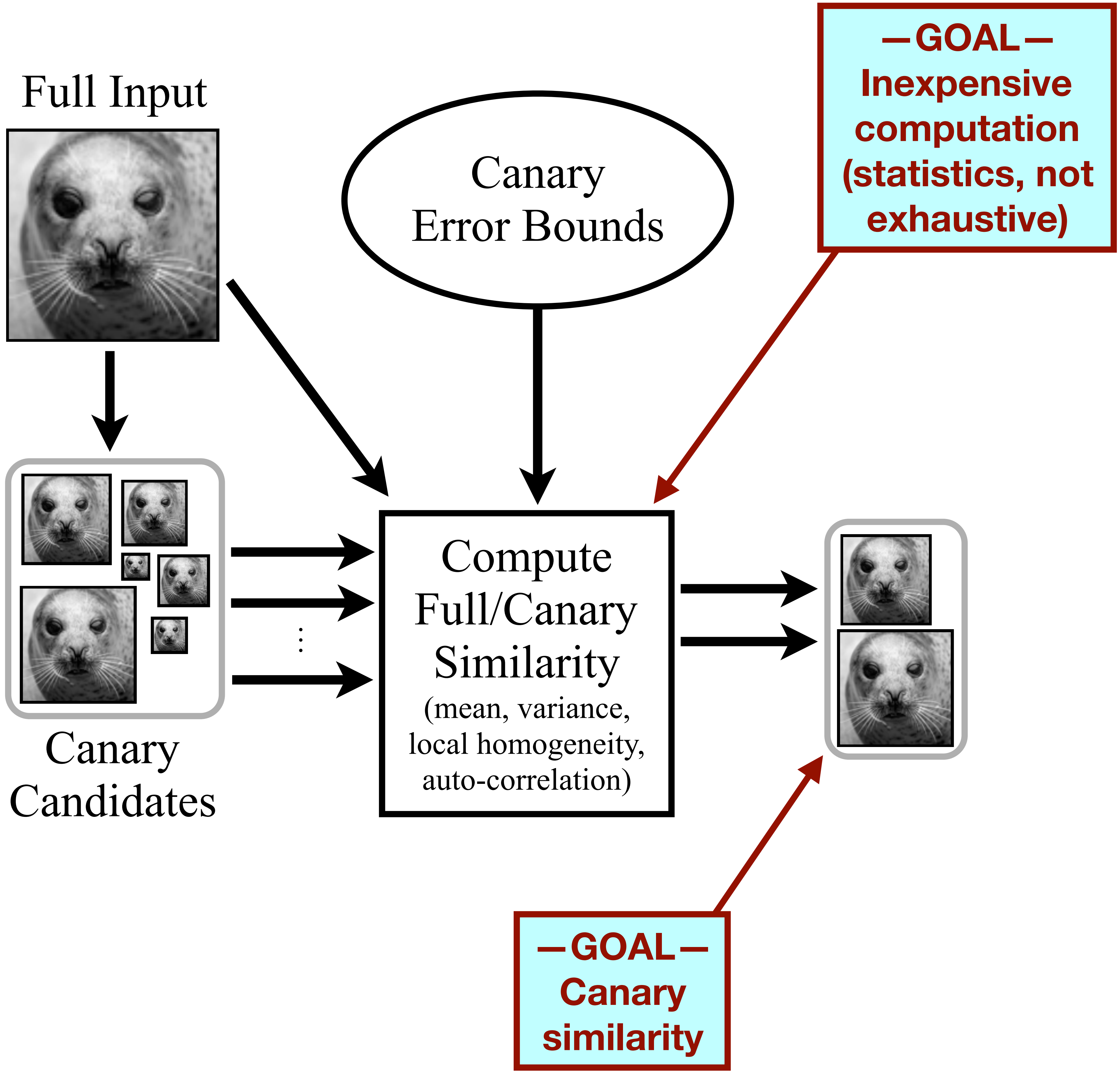
	Mean	Variance	Local Homogeneity	Autocorrelation
Description	Mean μ of input elements	Variance σ^2 of input elements	Proportion Λ of elements represented by λ in canary $\notin [\lambda - \sigma z_{1-\alpha/2}, \lambda + \sigma z_{1-\alpha/2}]$	Correlation ρ among pairs of input elements (y_j, y_{j+1})
Null Hypothesis (H_0) **Alt. Hypothesis** (H_A)	$H_0 : \overline{\mu_i} = \mu_0$ $H_A : \overline{\mu_i} \neq \mu_0$	$H_0 : \overline{\sigma_i}^2 = \sigma_0^2$ $H_A : \overline{\sigma_i}^2 \neq \sigma_0^2$	$H_0 : \overline{\Lambda_i} \leq 0.1$ $H_A : \overline{\Lambda_i} > 0.1$	$H_0 : \overline{\rho_i} = \rho_0$ $H_A : \overline{\rho_i} \neq \rho_0$
Test Statistic (t_i)	$t_i = \frac{\mu_0 - \overline{\mu_i}}{\overline{\sigma_i}\sqrt{n}}$	$t_i = \frac{\overline{\sigma_i}^2}{\sigma_0^2}$	$t_i = \frac{\sqrt{n}\,\lvert\overline{\Lambda_i} - 0.1\rvert}{\sqrt{0.1(1-0.1)}}$	$t_i = \frac{ln\left(\frac{(1+\rho_0)(1-\overline{\rho_i})}{(1-\rho_0)(1+\overline{\rho_i})}\right)}{2\sqrt{n-3}}$
p-value (p_i)	$p_i = 2P(Z > t_i)$	$p_i = 2P(F_{n-1,n-1} > t_i)$	$p_i = 2P(Z > t_i)$	$p_i = 2P(Z > t_i)$
Sample Size (n)	$n = 2(z_{1-\alpha/2k}$ $+ z_{1-\beta/k})^2$	Formula yields no simple form; see Cohen [14] for details.	$g(x) = \sqrt{x(1-x)}$ $n = 0.1^{-2}(g(0.1)z_{1-\alpha/2k} + g(\overline{\Lambda_i})z_{1-\beta/k})^2$	$n = \frac{4(z_{1-\alpha/2k} + z_{1-\beta/k})^2}{ln((1+\rho_0)/(1-\rho_0))^2}$
Acceptability Test	*Holm-Bonferroni method:* sort p-values $p_1, p_2, \ldots p_k$ to obtain sorted p-values $p_{(1)}, p_{(2)}, \ldots p_{(k)}$. Find the minimum index m such that $p_{(m)} > \frac{\alpha}{k+1-m}$, then reject all canaries $C_{(i)}$ where $i \geq m$.			
Definitions	α: the desired bound on the probability of committing any Type I errors (false negative), β: the desired bound on Type II errors (false positive) k: the number of canary candidates, C_i: the ith canary candidate, $\overline{x_i}$: the sample statistic x for canary C_i, x_0: the sample statistic x for the full input Z: the standard normal distribution, z_y: the quantile function at y of Z, $F_{b,c}$: the F-distribution with degrees of freedom b and c			

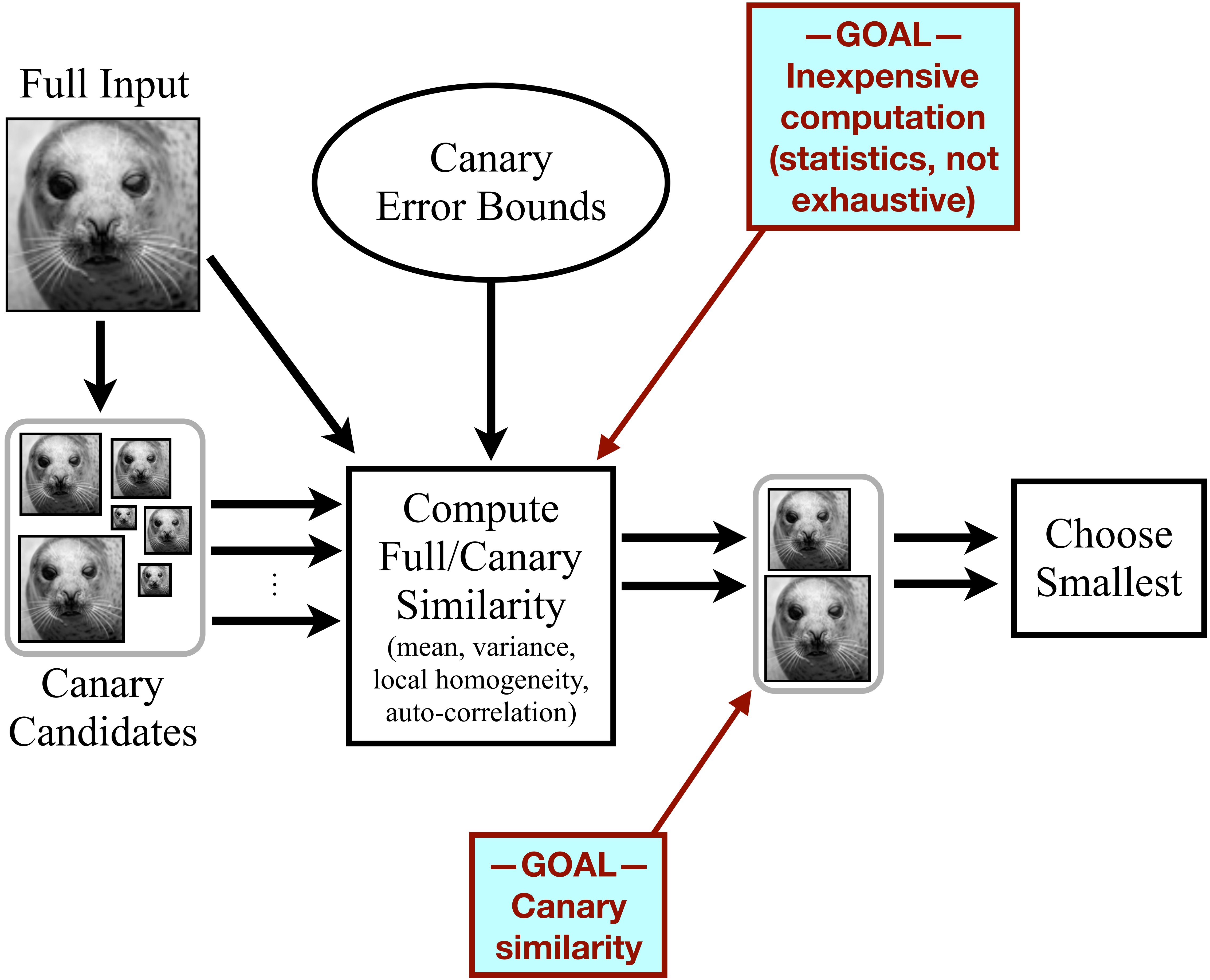


—GOAL—
Inexpensive computation (statistics, not exhaustive)
Full Input
Canary Error Bounds
Canary Candidates
Compute Full/Canary Similarity
(mean, variance, local homogeneity, auto-correlation)

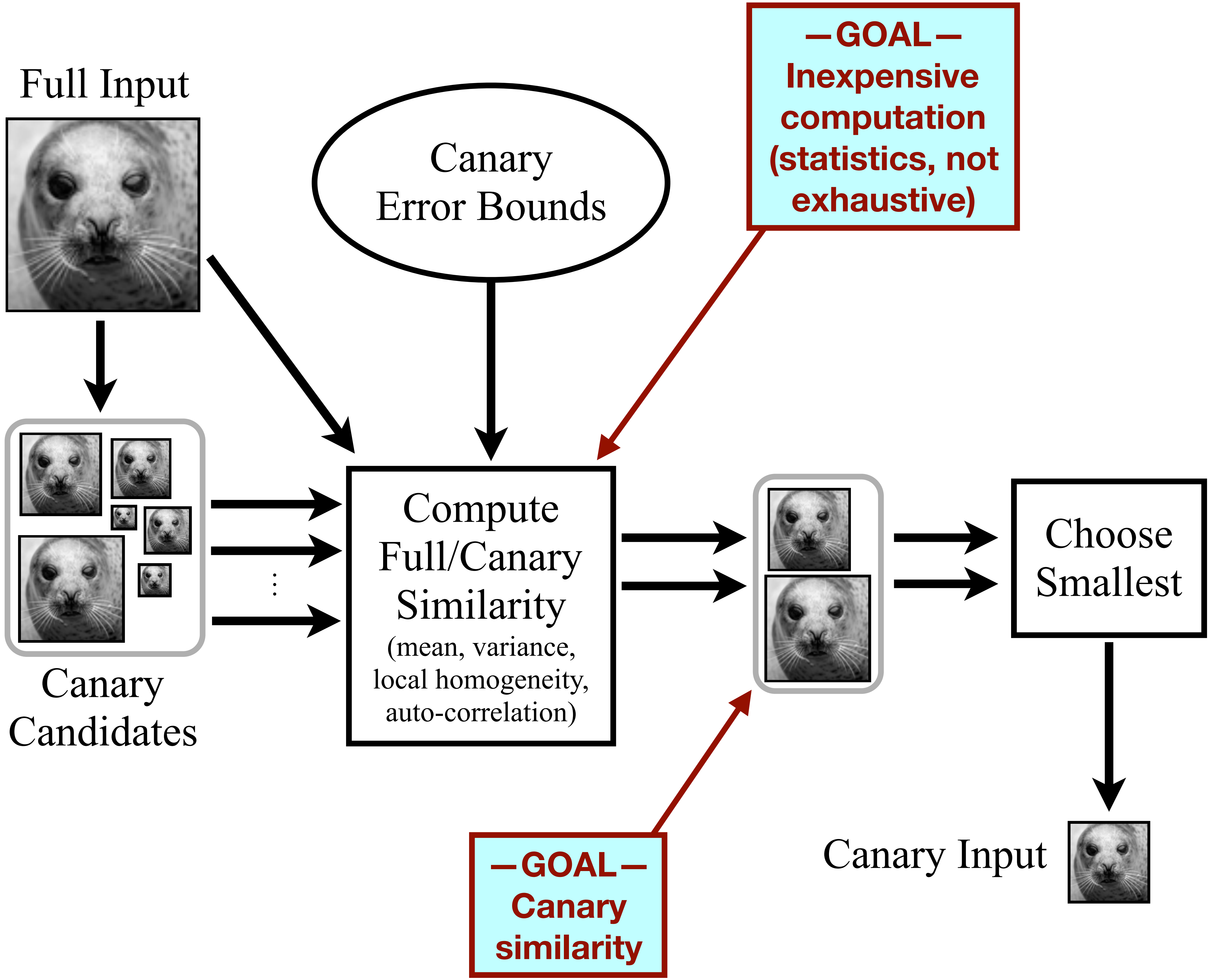


—GOAL—
Inexpensive computation (statistics, not exhaustive)
Full Input
Canary Error Bounds
Compute Full/Canary Similarity
(mean, variance, local homogeneity, auto-correlation)
Canary Candidates

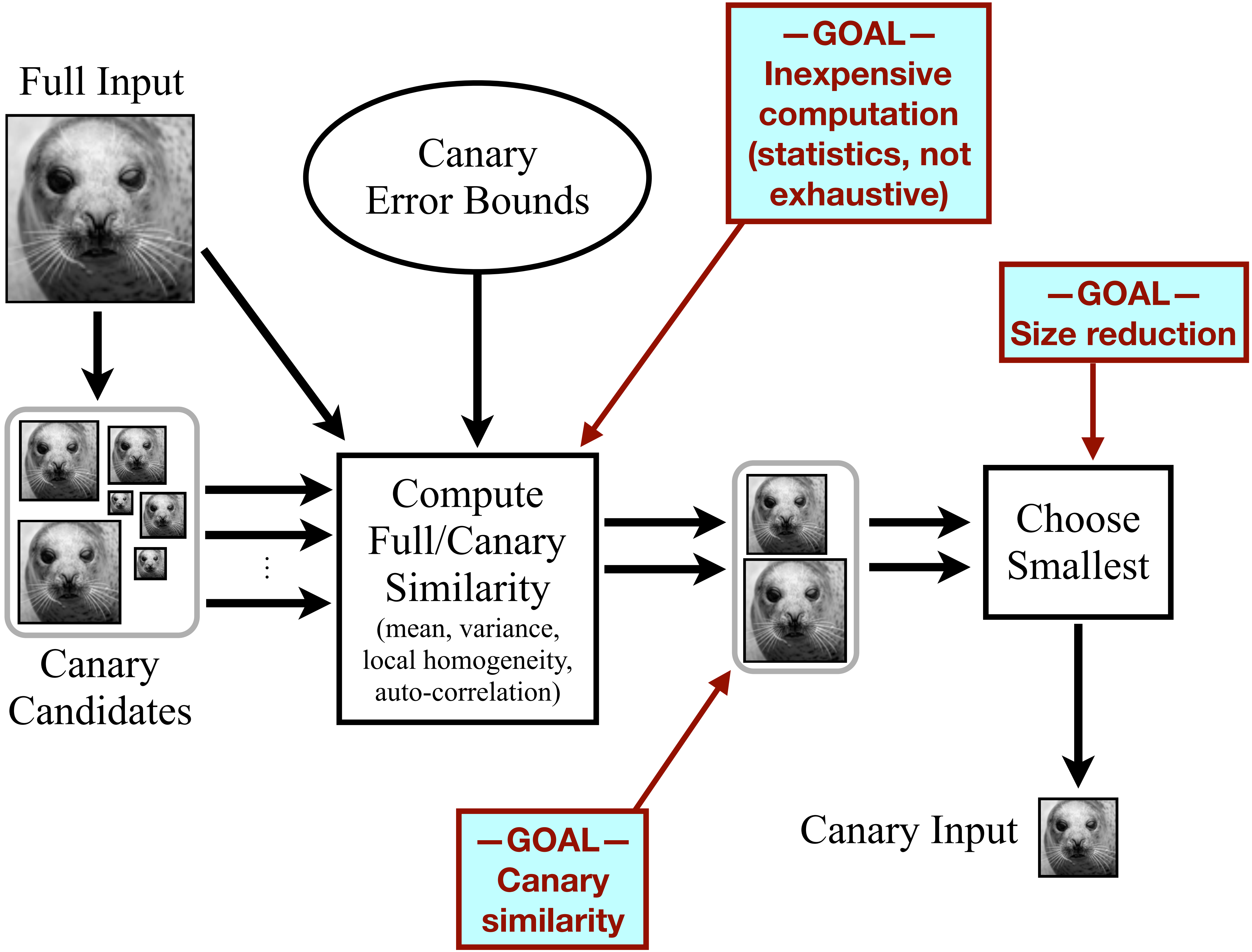

Full Input
Canary Error Bounds
—GOAL—
Inexpensive computation (statistics, not exhaustive)
Canary Candidates
Compute Full/Canary Similarity
(mean, variance, local homogeneity, auto-correlation)
—GOAL—
Canary similarity



Full Input
Canary Error Bounds
—GOAL—
Inexpensive computation (statistics, not exhaustive)
Canary Candidates
Compute Full/Canary Similarity
(mean, variance, local homogeneity, auto-correlation)
Choose Smallest
—GOAL—
Canary similarity



Full Input
Canary
Error Bounds
—GOAL—
Inexpensive
computation
(statistics, not
exhaustive)
Canary
Candidates
Compute
Full/Canary
Similarity
(mean, variance,
local homogeneity,
auto-correlation)
Choose
Smallest
—GOAL—
Canary
similarity
Canary Input


Full Input
Canary
Error Bounds
—GOAL—
Inexpensive
computation
(statistics, not
exhaustive)
—GOAL—
Size reduction
Canary
Candidates
Compute
Full/Canary
Similarity
(mean, variance,
local homogeneity,
auto-correlation)
Choose
Smallest
—GOAL—
Canary
similarity
Canary Input

Choosing How to Approximate

Choosing How to Approximate

- Tune approx. parameters within all code regions

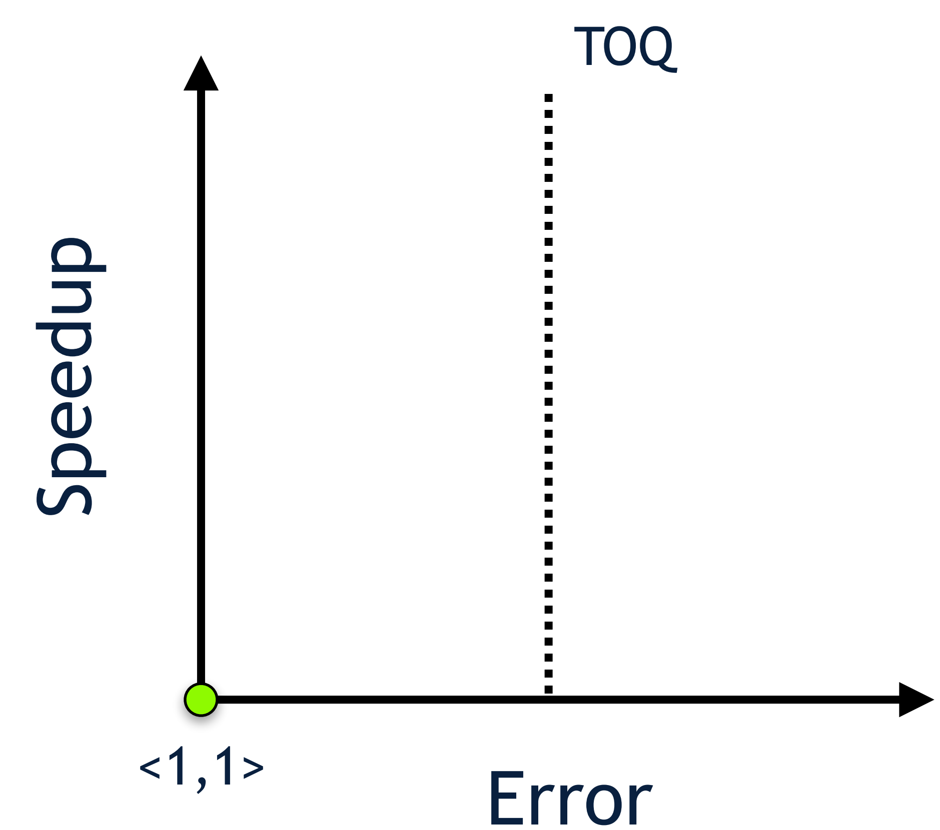
Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

Canary-driven Search



Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

Canary-driven Search

Encoding of approximation parameters (2 code regions)

Choosing How to Approximate

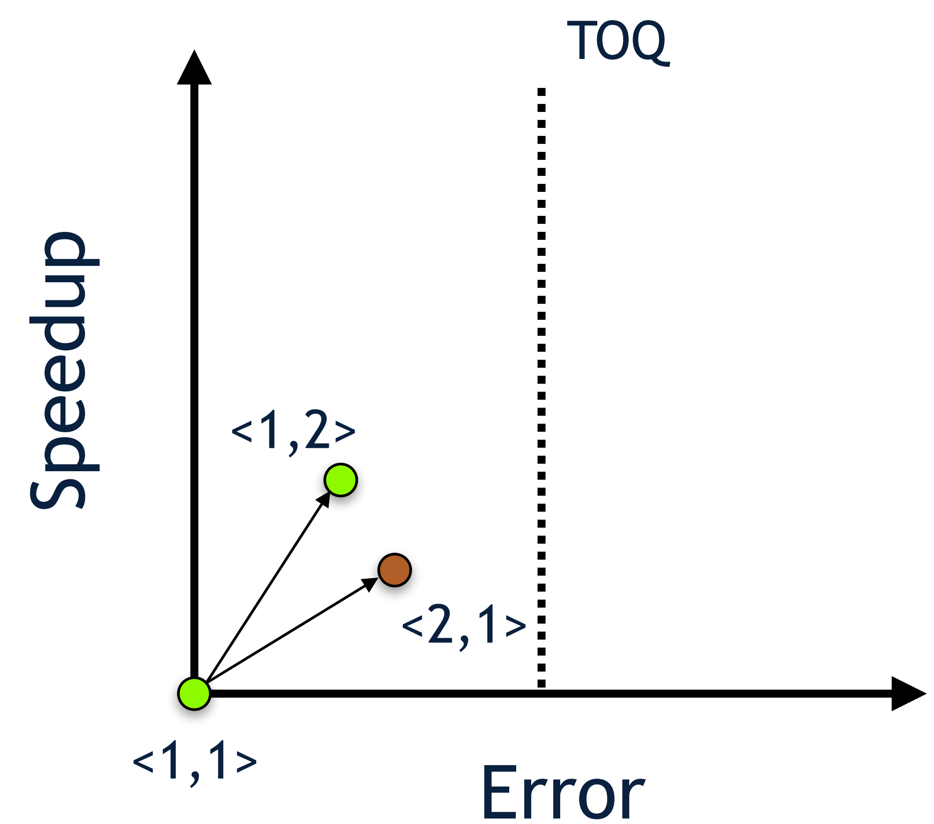
- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

Canary-driven Search

Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

Canary-driven Search



Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

Canary-driven Search

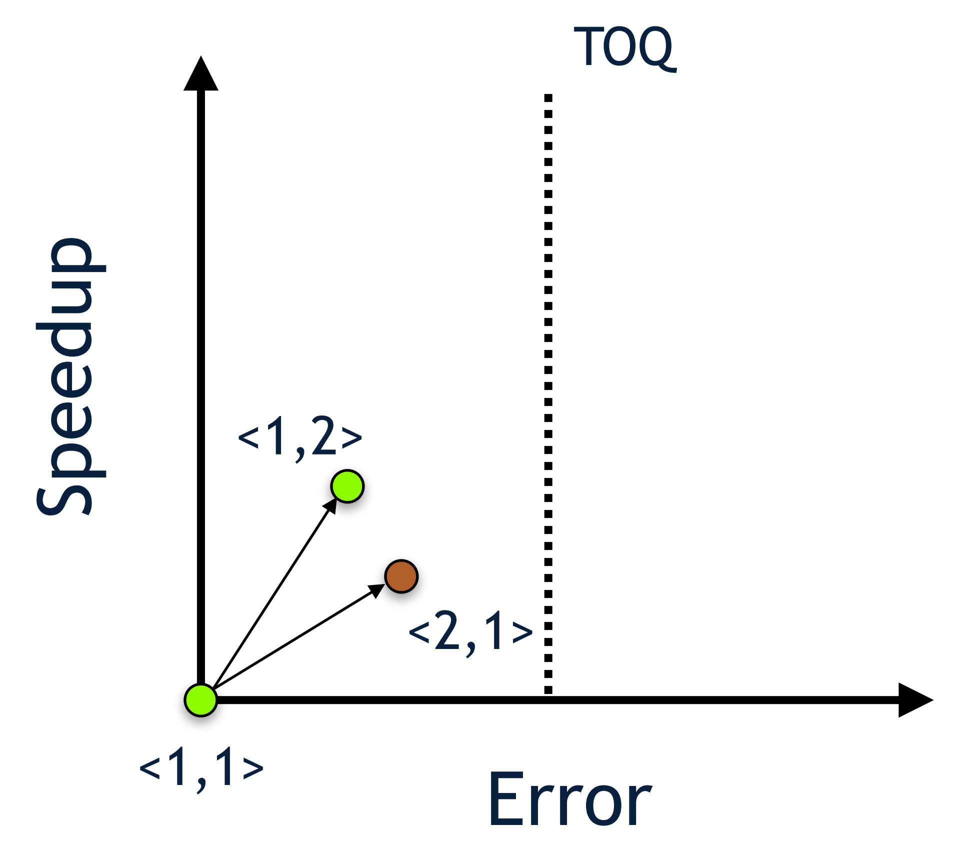
COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

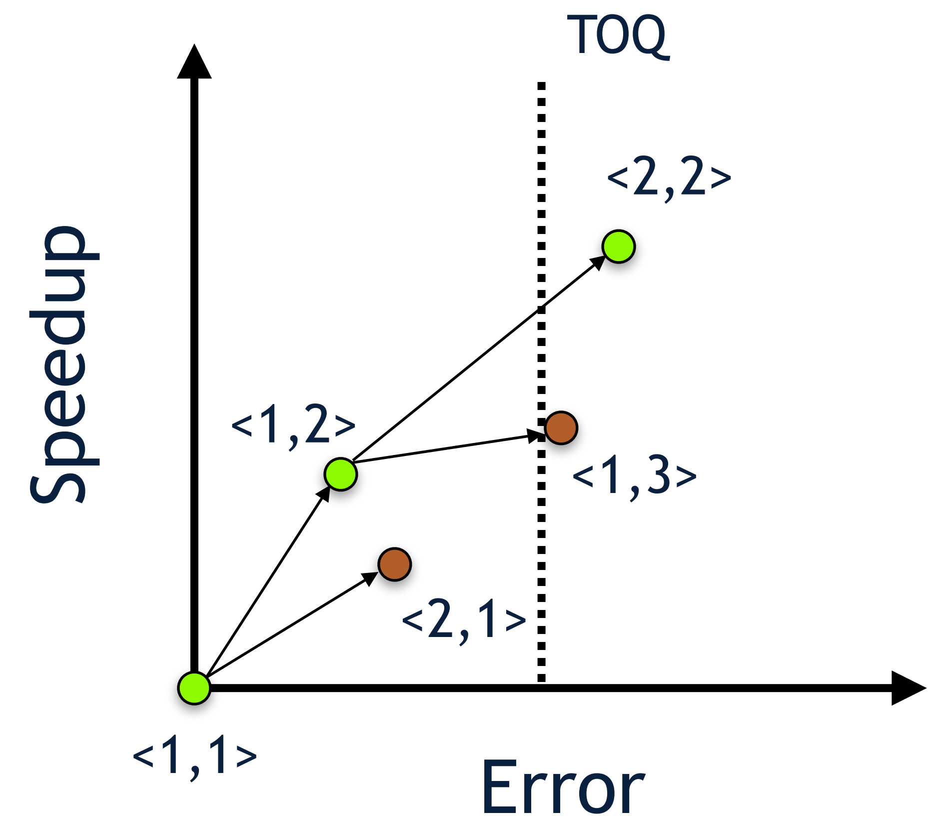
Canary-driven Search



Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

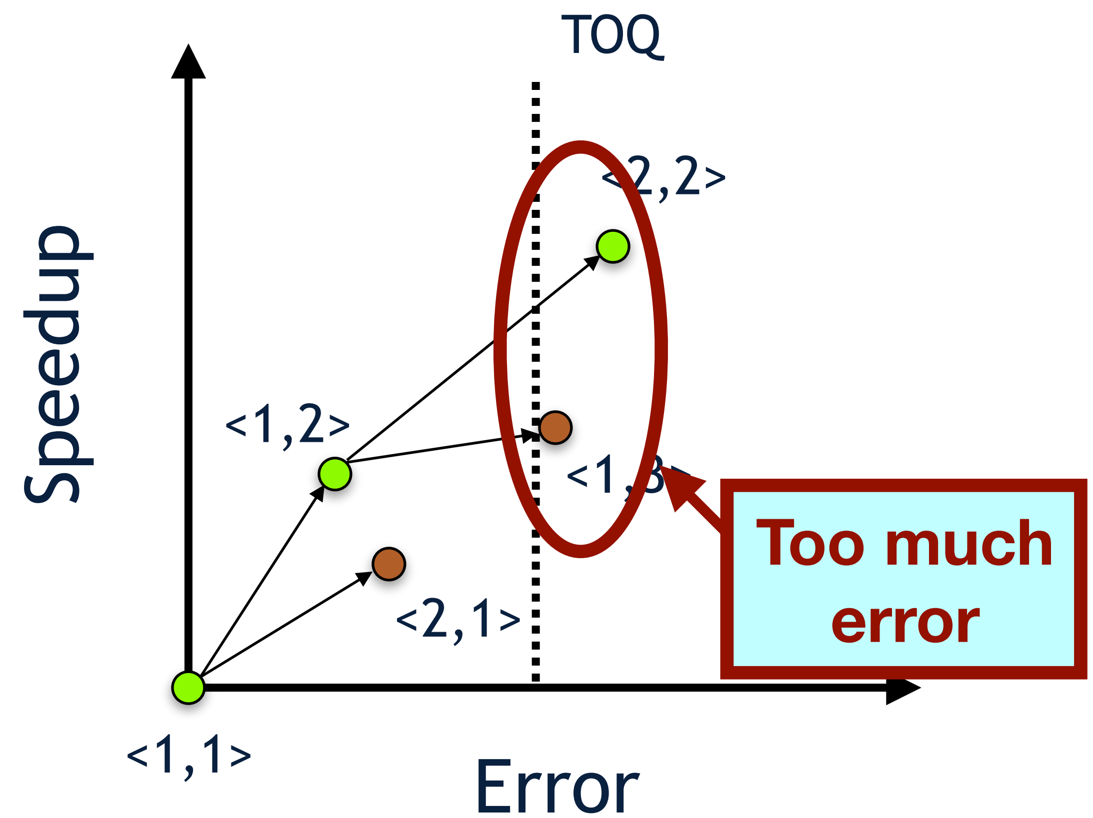
Canary-driven Search



Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

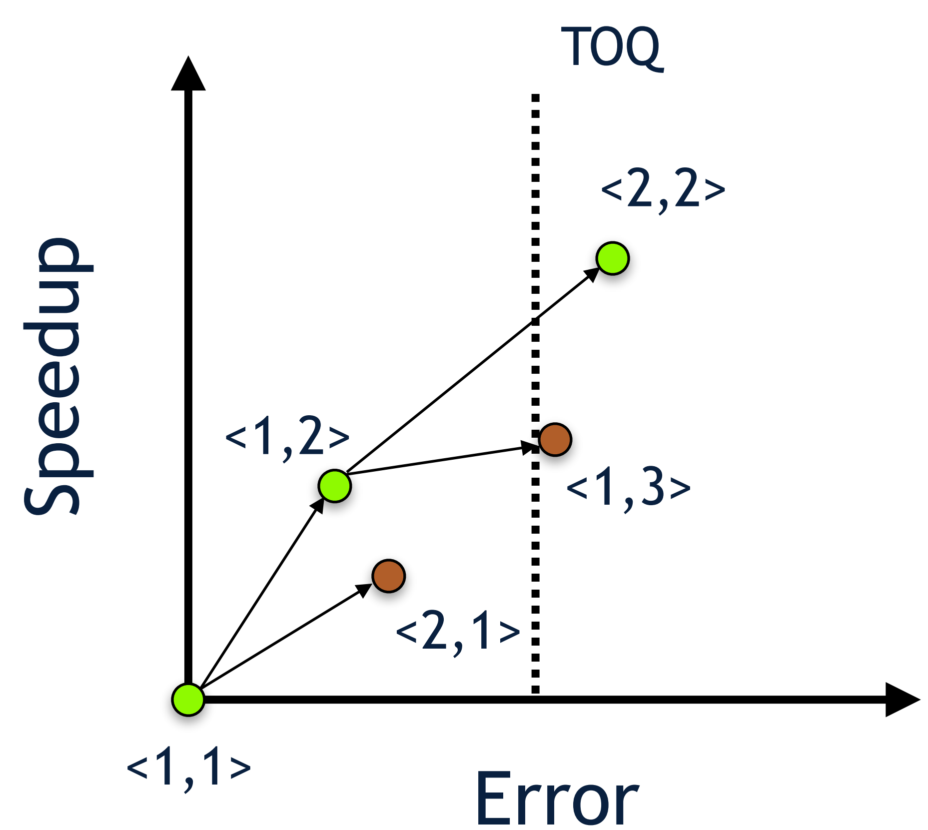
Canary-driven Search



Choosing How to Approximate

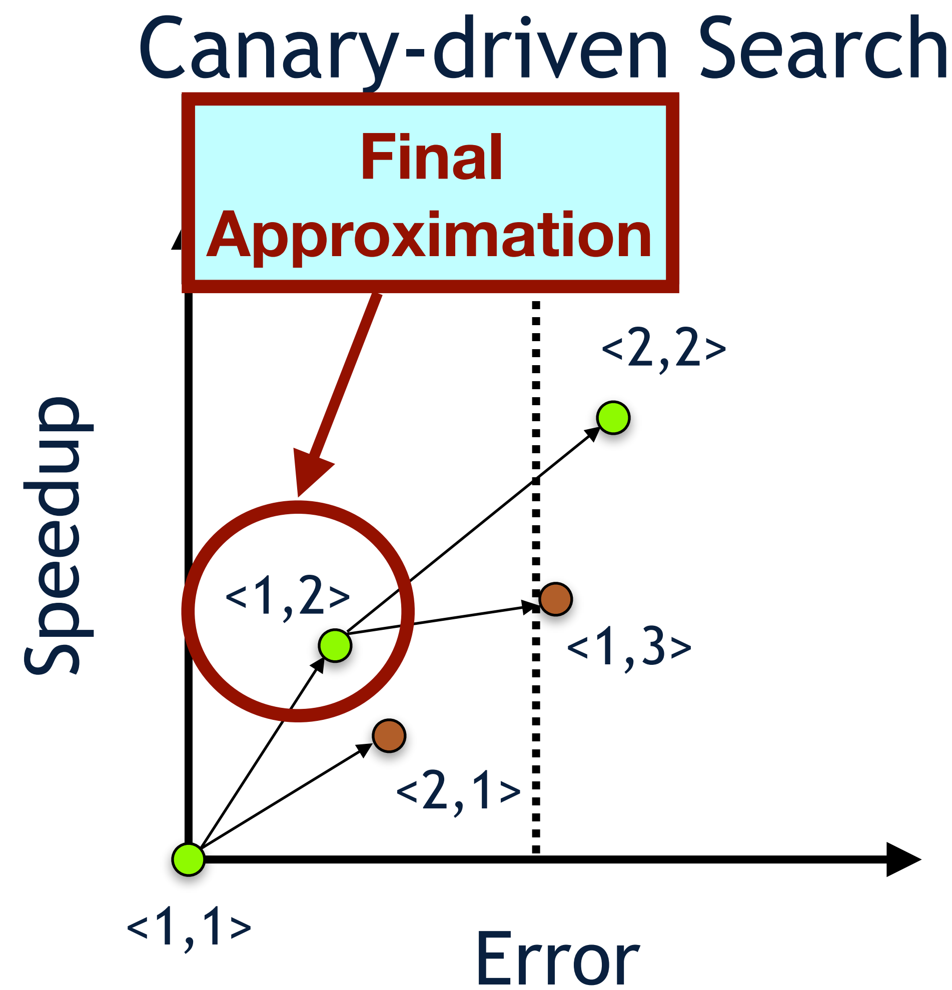
- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

Canary-driven Search



Choosing How to Approximate

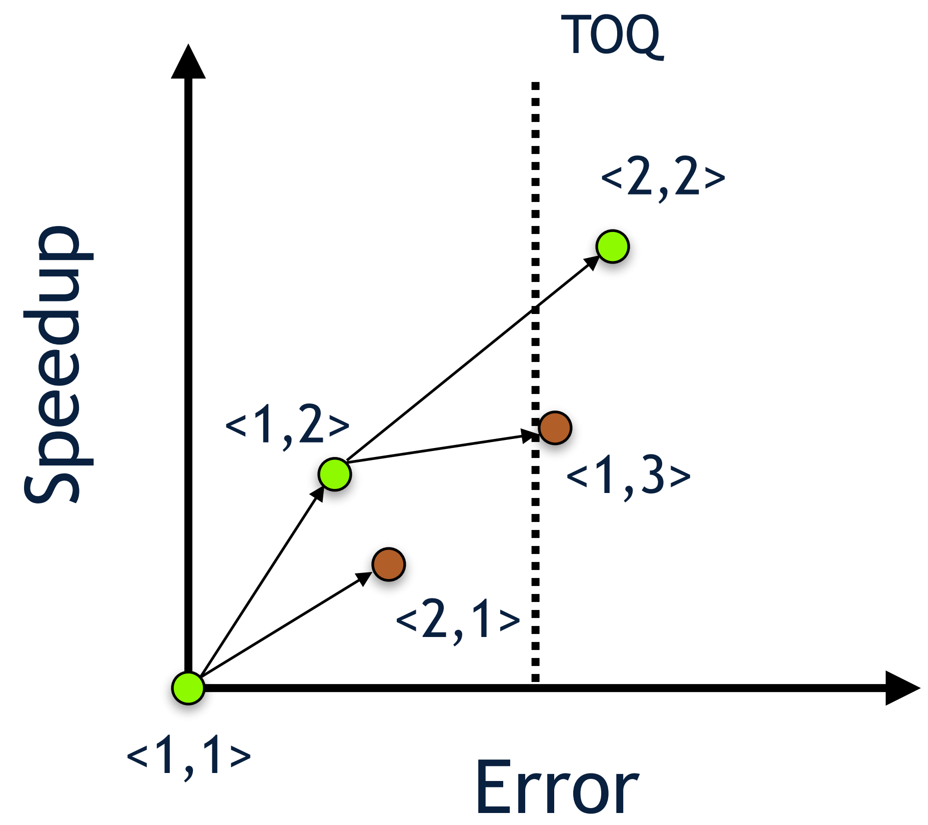
- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing



Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing

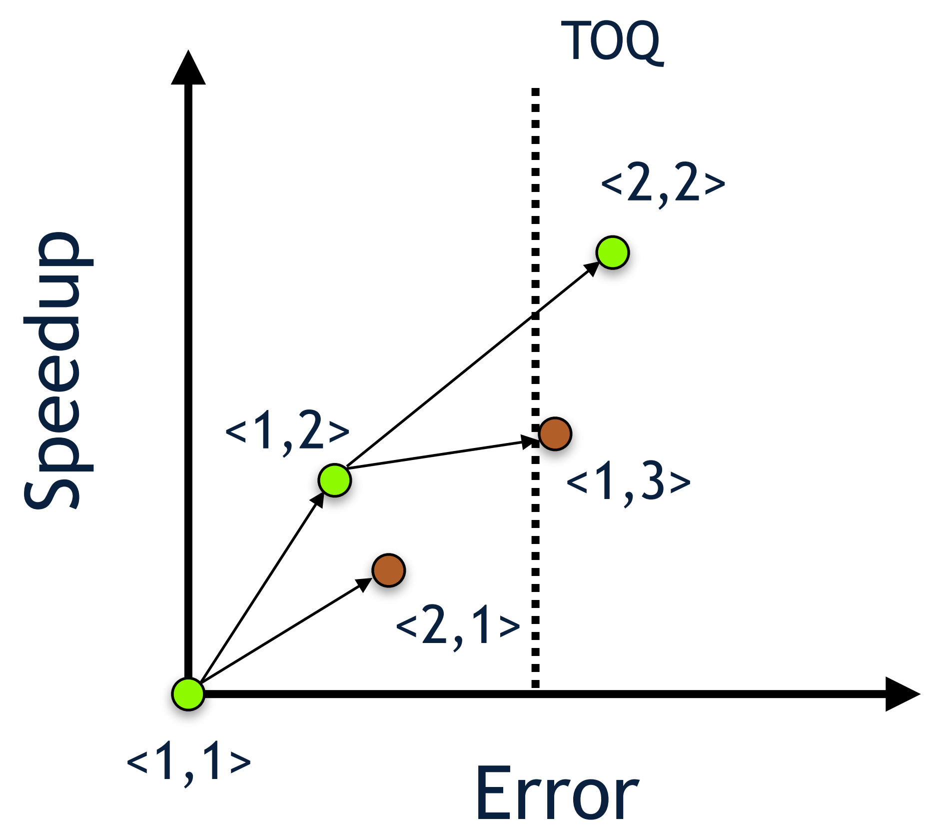
Canary-driven Search



Choosing How to Approximate

- Tune approx. parameters within all code regions
- Search (greedy) with steepest ascent hill climbing
- Use the resulting approximation on full input

Canary-driven Search



Evaluation

Experimental Setup

Experimental Setup

- 13 applications
  - Approximation opportunities per application — 1-5
  - Inputs per application — 2-800

Experimental Setup

- 13 applications
  - Approximation opportunities per application — 1-5
  - Inputs per application — 2-800
- Approximation classes
  - Loop perforation, tiling, algorithm choice, numerical approximation

Experimental Setup

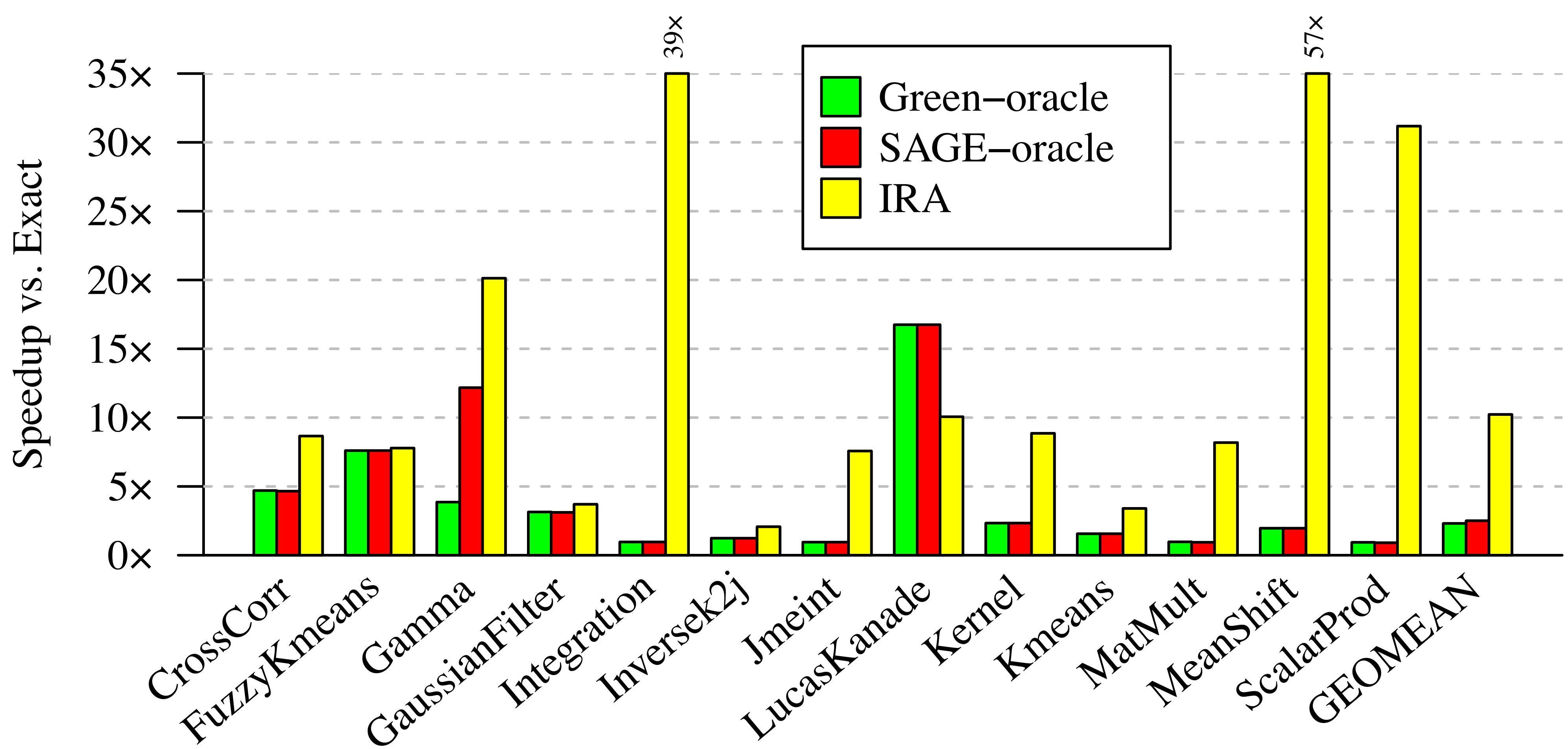
- 13 applications
  - Approximation opportunities per application — 1-5
  - Inputs per application — 2-800
- Approximation classes
  - Loop perforation, tiling, algorithm choice, numerical approximation
- Canary similarity metric — variance

Experimental Setup

- 13 applications
  - Approximation opportunities per application — 1-5
  - Inputs per application — 2-800
- Approximation classes
  - Loop perforation, tiling, algorithm choice, numerical approximation
- Canary similarity metric — variance
- Target output quality (TOQ) — 90%

Speedup — IRA vs. Prior Work

- Calibration — compare exact result to approximate result every so often
  - Oracle versions of SAGE [MICRO'13] and Green [PLDI'10]



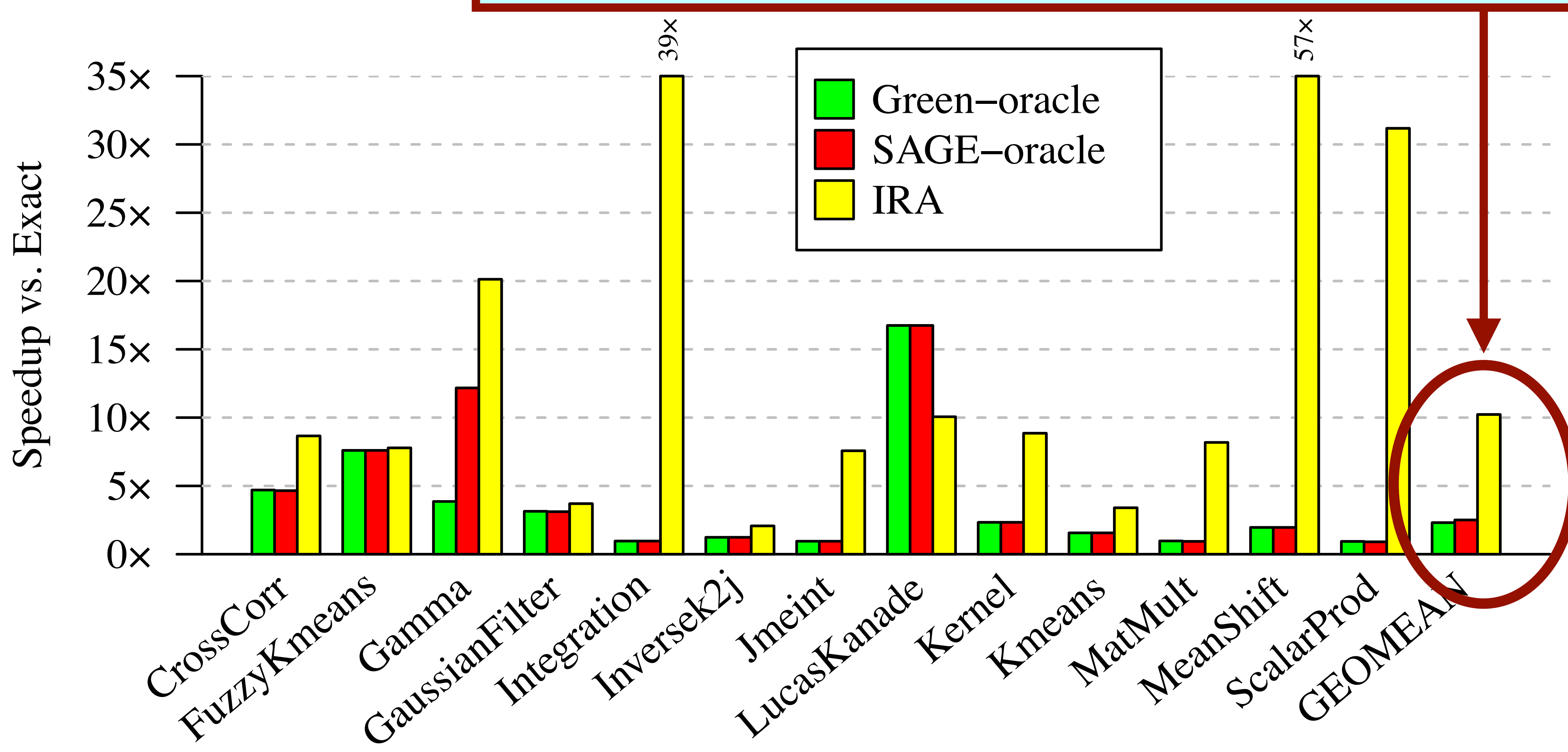
COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

Speedup — IRA vs. Prior Work

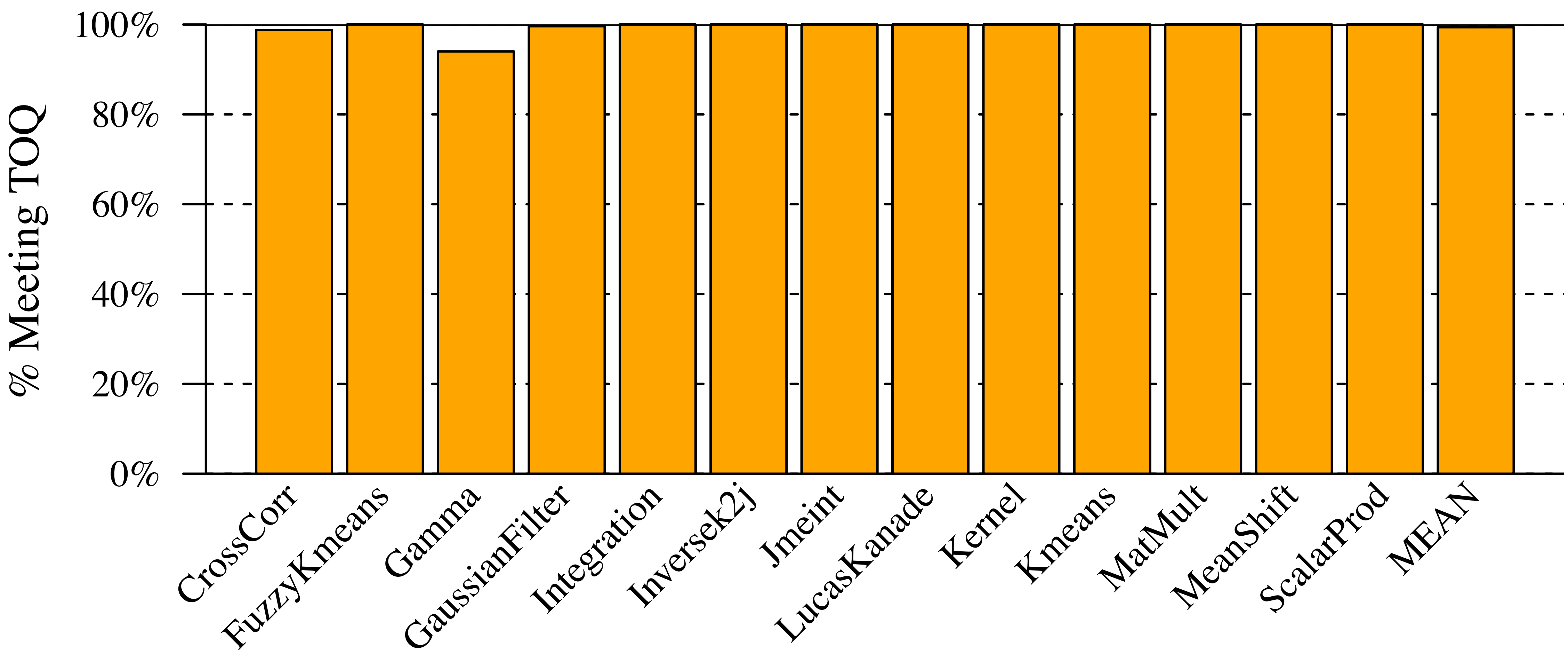
- Calibration — compare exact result to approximate result every so often
    - Oracle versions of SAGE [MICRO'13] and Green [PLDI'10]

IRA's 10x speedup outperforms oracle calibration by 4x



Accuracy of IRA

- % of outputs meeting TOQ (TOQ=90%)

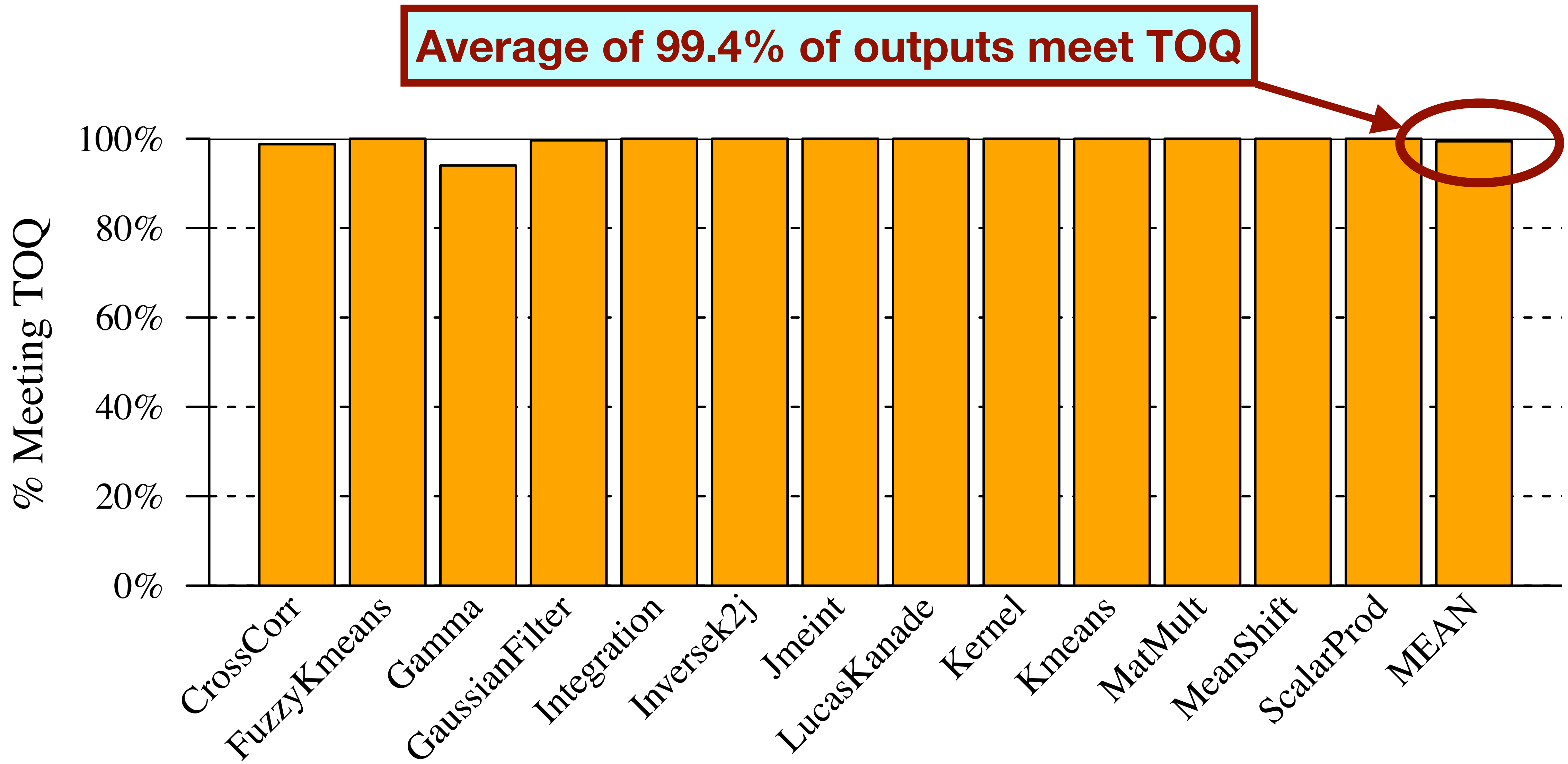


COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

Accuracy of IRA

- % of outputs meeting TOQ (TOQ=90%)

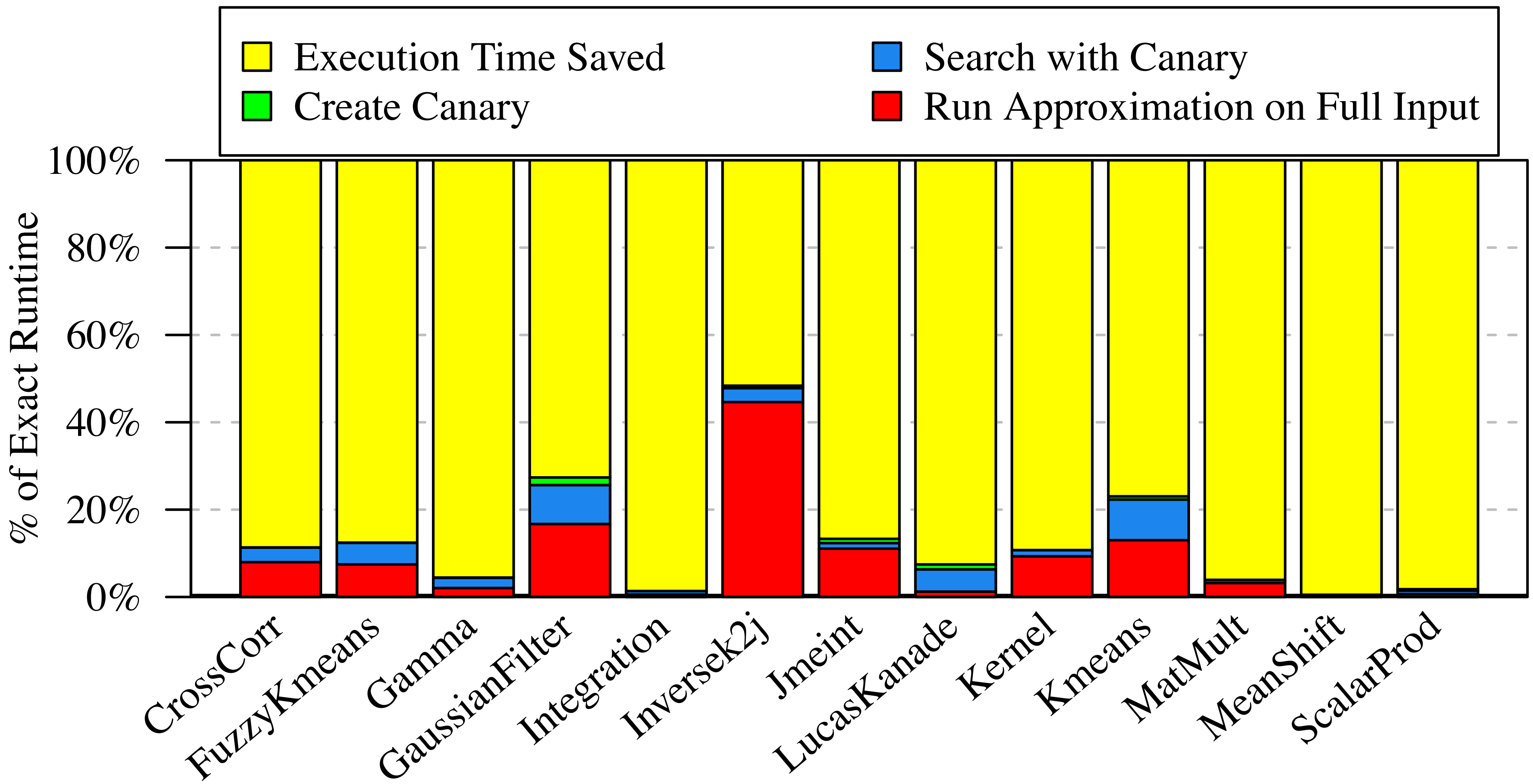


COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

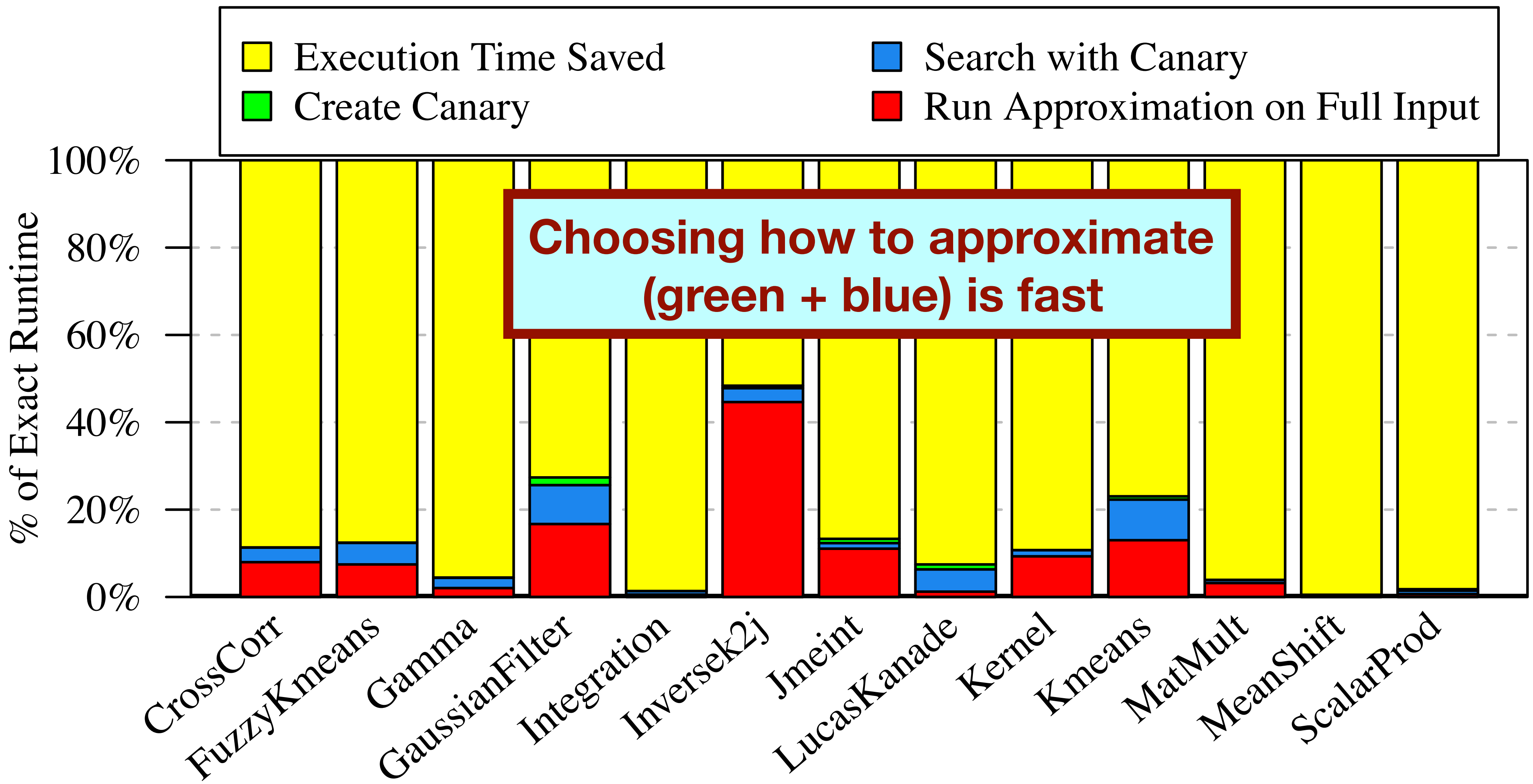
Where is the Time Spent?

- End-to-end execution time broken down



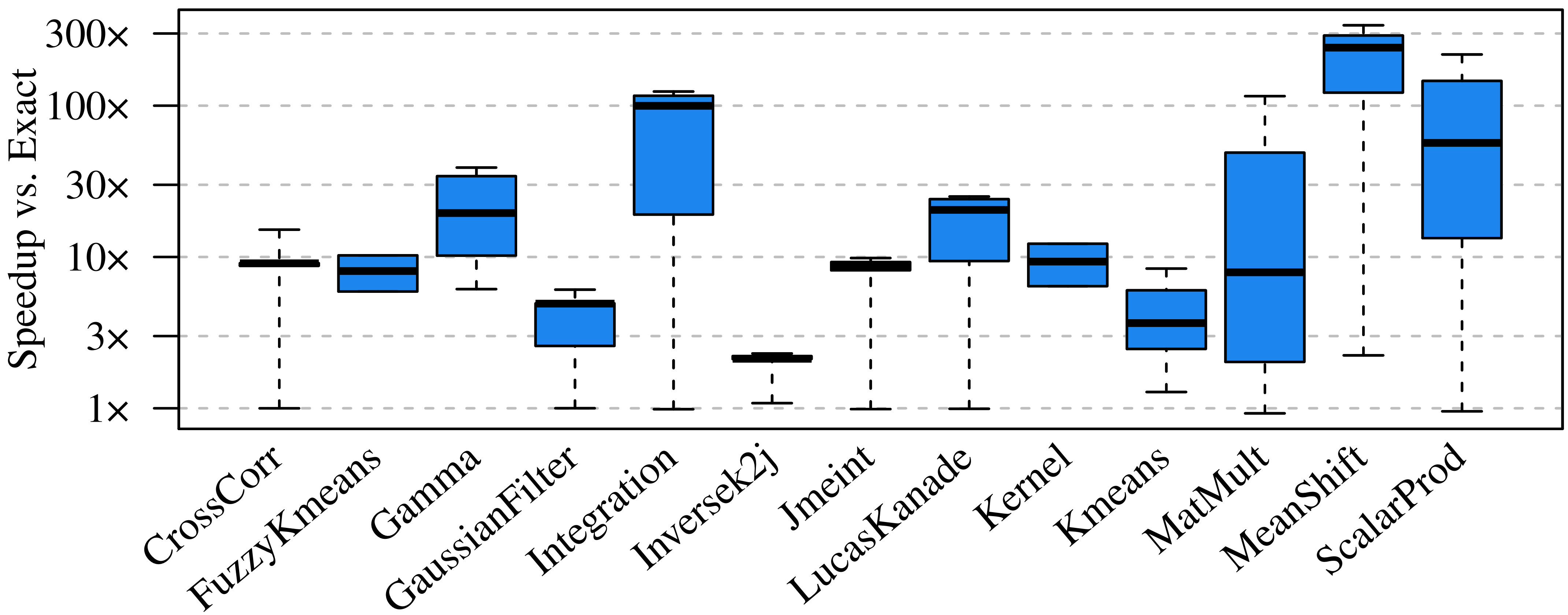
Where is the Time Spent?

- End-to-end execution time broken down



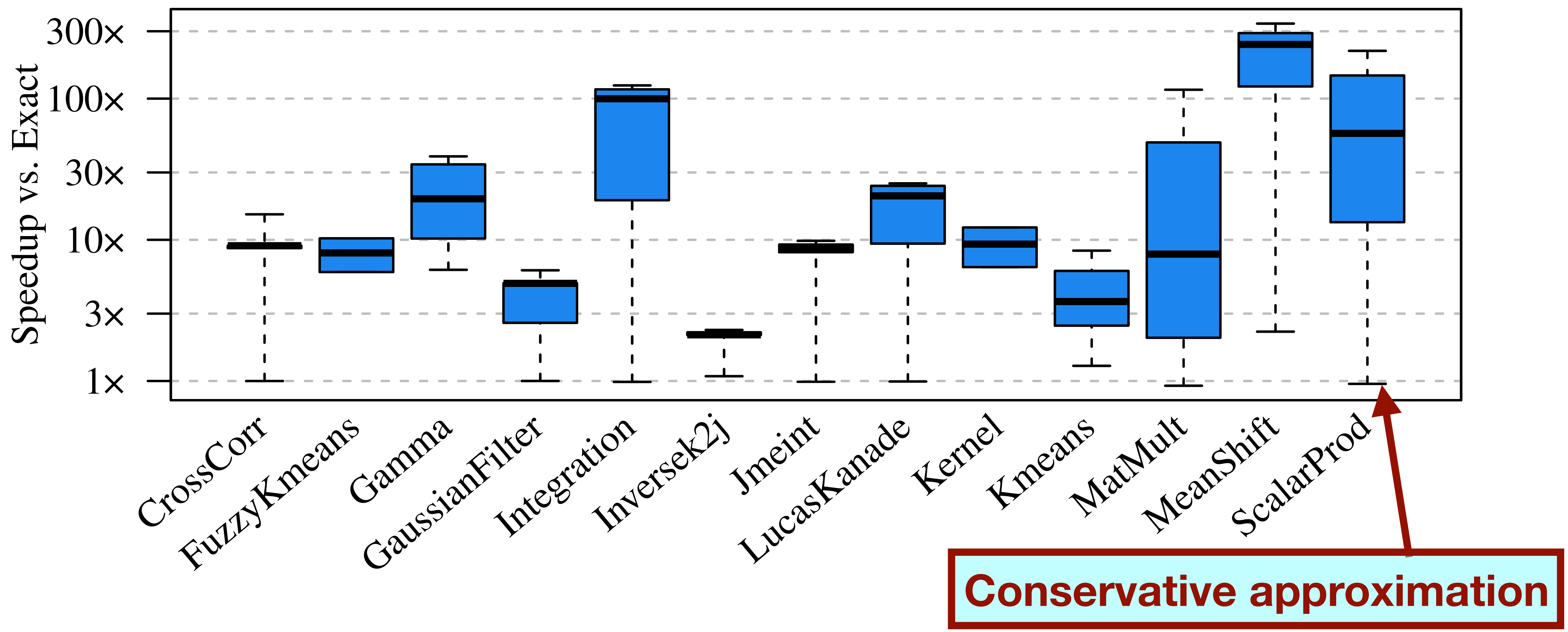
Input Responsiveness

- Wide range of speedups (and approximations) used across different inputs



Input Responsiveness

- Wide range of speedups (and approximations) used across different inputs

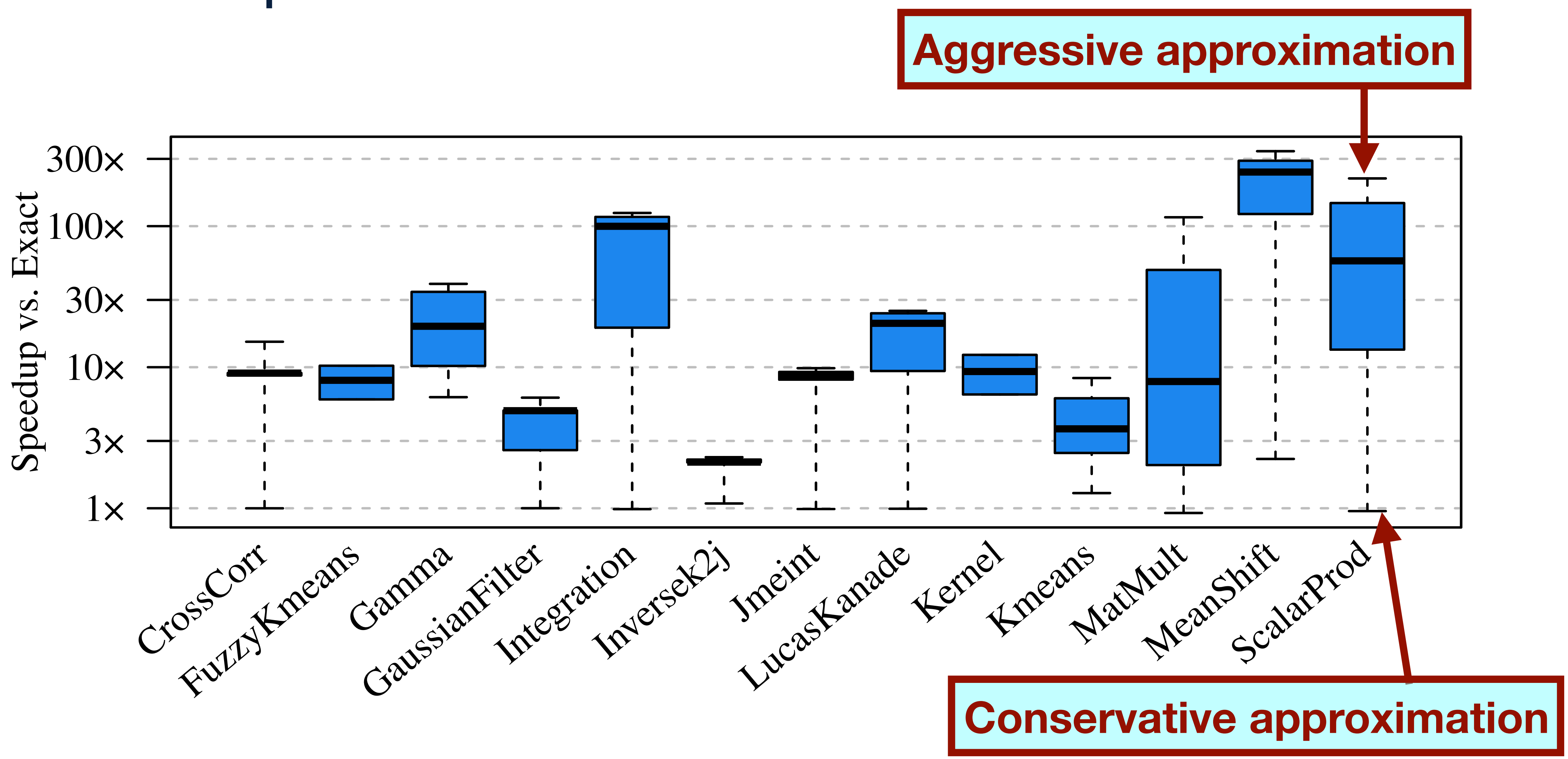


COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

Input Responsiveness

- Wide range of speedups (and approximations) used across different inputs



COMPUTER SCIENCE AND ENGINEERING UNIVERSITY OF MICHIGAN

ClarityLab

Conclusion

Conclusion

- Input is a key component in approximation
- IRA — end-to-end system for input responsive approximation
  - Driven by canaries — smaller representations of full input
  - Chooses code regions to approximate and how to parameterize approximation
  - 10x speedup across applications (90% TOQ)
  - Outperforms oracle calibration by ~4x

?

Backup Slides

Canary Inputs

Canary Inputs

- **Canary** – a smaller program input to model full input behavior

Canary Inputs

- **Canary** – a smaller program input to model full input behavior
- Why is a canary useful?

Canary Inputs

- **Canary** – a smaller program input to model full input behavior

- Why is a canary useful?

  - Characterize full input behavior without the computational expense

Canary Inputs

- **Canary** – a smaller program input to model full input behavior
- Why is a canary useful?
  - Characterize full input behavior without the computational expense
  - Customize approximation for each input

Regularly-structured Computation

- **Data-centric** — amount of computation depends on the size of the input
- **Summarizable inputs** — redundancy, pattern, or marginal information in the input data
- E.g., image filters on the following

Why Input Responsiveness?

Why Input Responsiveness?

- Example: gamma correction, 3 tiling approximations, 800 inputs

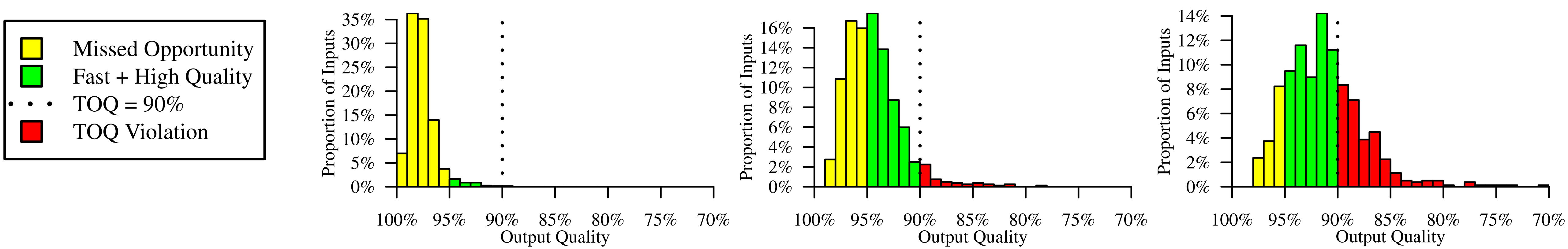
Why Input Responsiveness?

- Example: gamma correction, 3 tiling approximations, 800 inputs

4x2 tiling, 5.9x speedup

8x8 tiling, 22x speedup

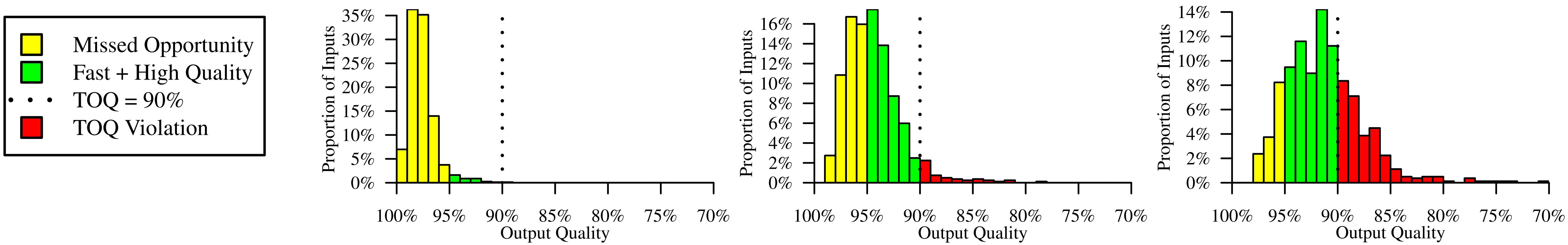
16x16 tiling, 83x speedup



Why Input Responsiveness?

- Example: gamma correction, 3 tiling approximations, 800 inputs

4x2 tiling, 5.9x speedup | 8x8 tiling, 22x speedup | 16x16 tiling, 83x speedup

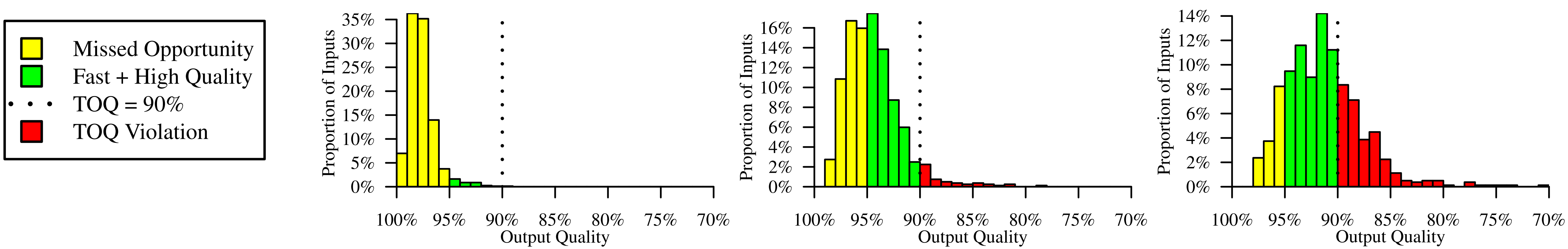


- What if we take full advantage of the differences among inputs?

Why Input Responsiveness?

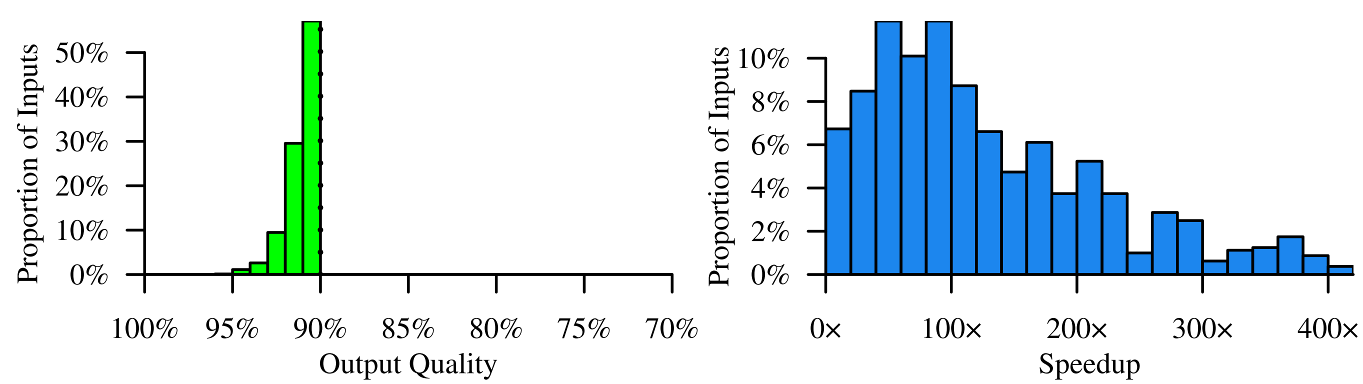
- Example: gamma correction, 3 tiling approximations, 800 inputs

4x2 tiling, 5.9x speedup   8x8 tiling, 22x speedup   16x16 tiling, 83x speedup



- What if we take full advantage of the differences among inputs?

High output quality   61x speedup



Sketch of Canary Creation Approach

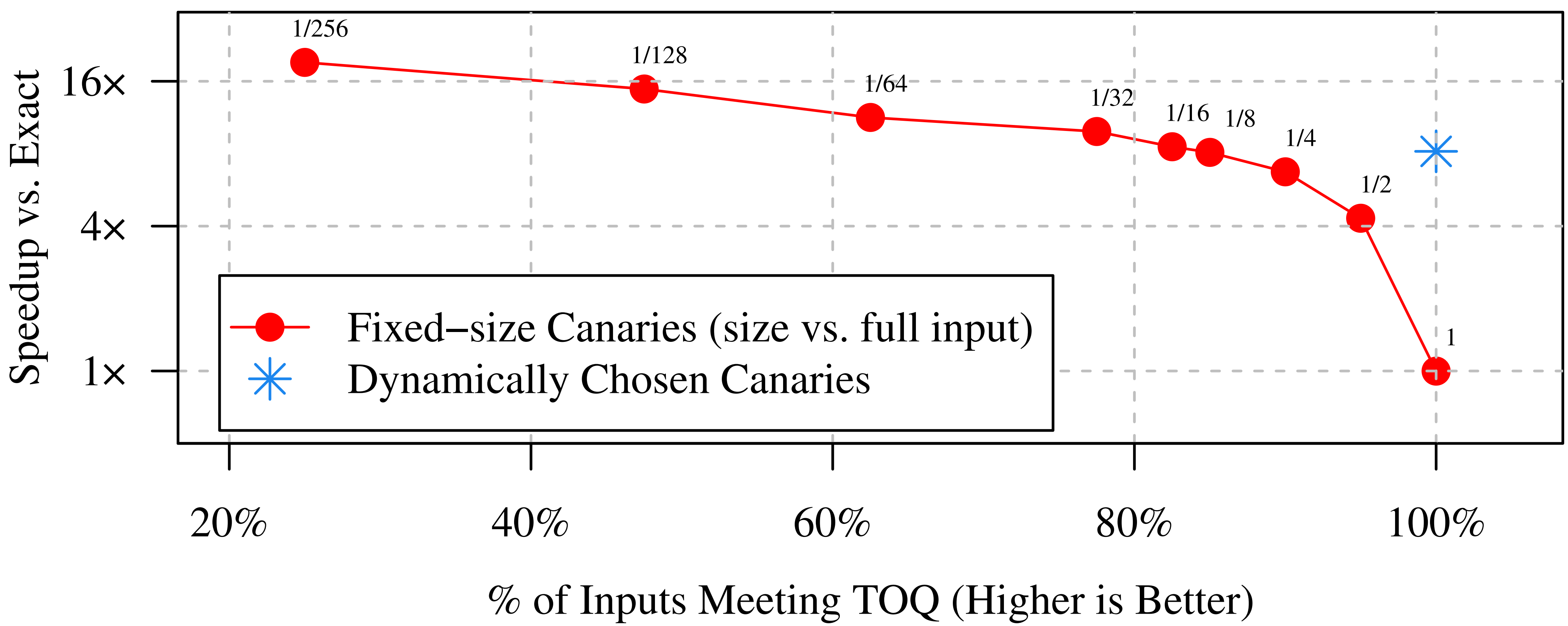
- Create a batch of canary candidates — subsets of full input
- Statistically measure similarity of candidates to full input
  - Hypothesis tests — *is canary candidate X similar to the full input?*
  - Metrics — mean, variance, local homogeneity, auto-correlation
  - Multiple comparisons problem — across all candidates, bound the probability of making a mistake
- Choose the smallest canary that is found to be similar

Addressing Challenges

Challenge	Devised Solution
Similarity to Full Input	Explore 4 definitions of similarity ranging from simple to complex
Computational Expense	Use incomplete measures of similarity (i.e., use statistics)
Significant Size Reduction	Choose smallest from a set of acceptable canaries

Why Dynamically Chosen Canaries?

- Matrix multiplication (40 inputs)



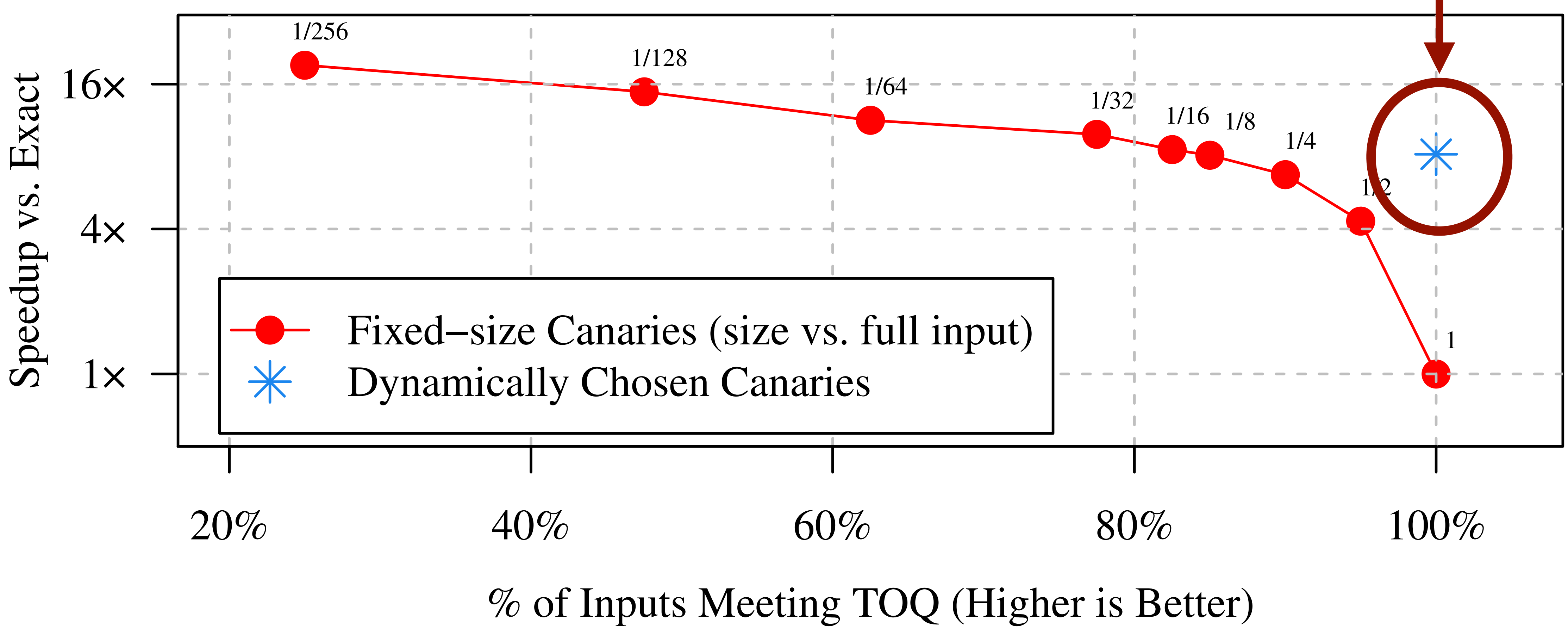
COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

Why Dynamically Chosen Canaries?

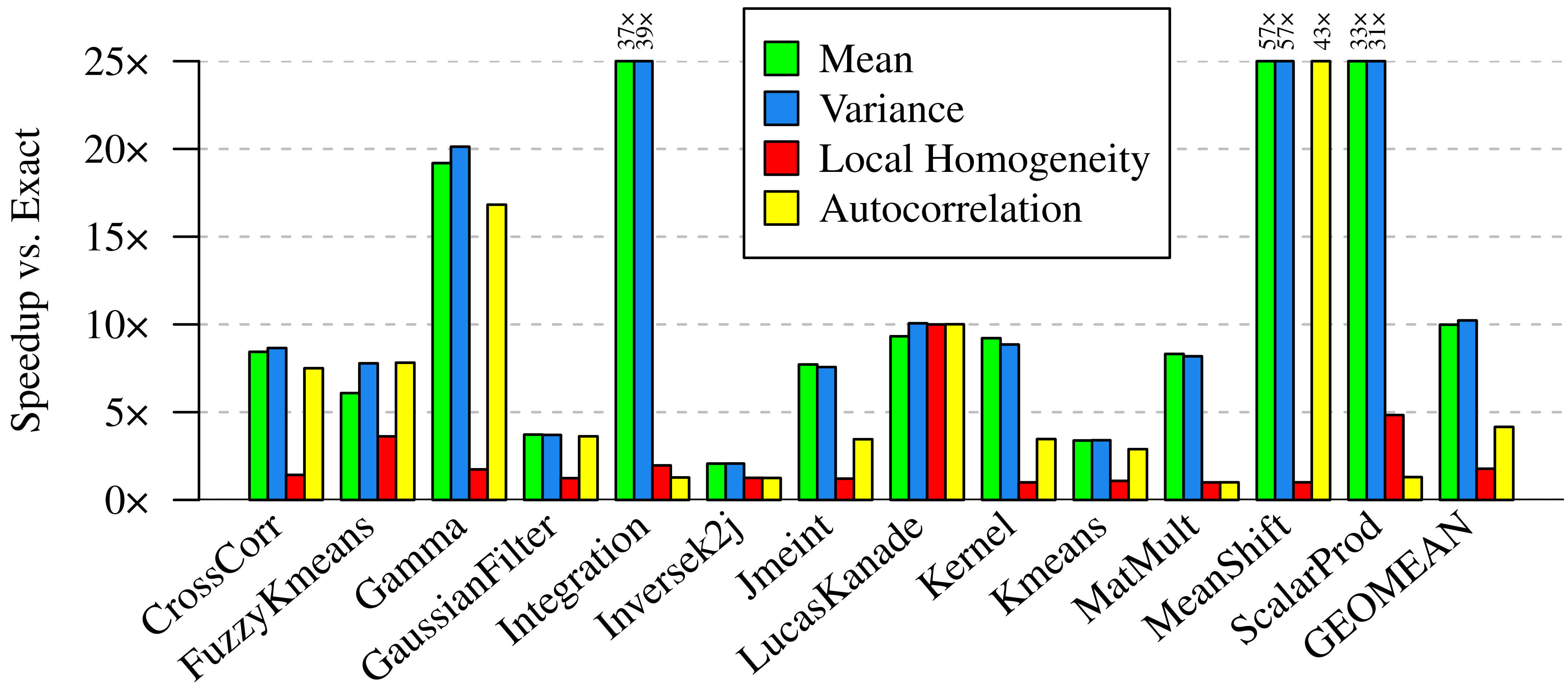
- Matrix multiplication (40 inputs)

Dynamically chosen canaries yield no TOQ violations and 8.2x speedup



Canary Metrics Comparison

- End-to-end IRA speedup across 4 similarity metrics



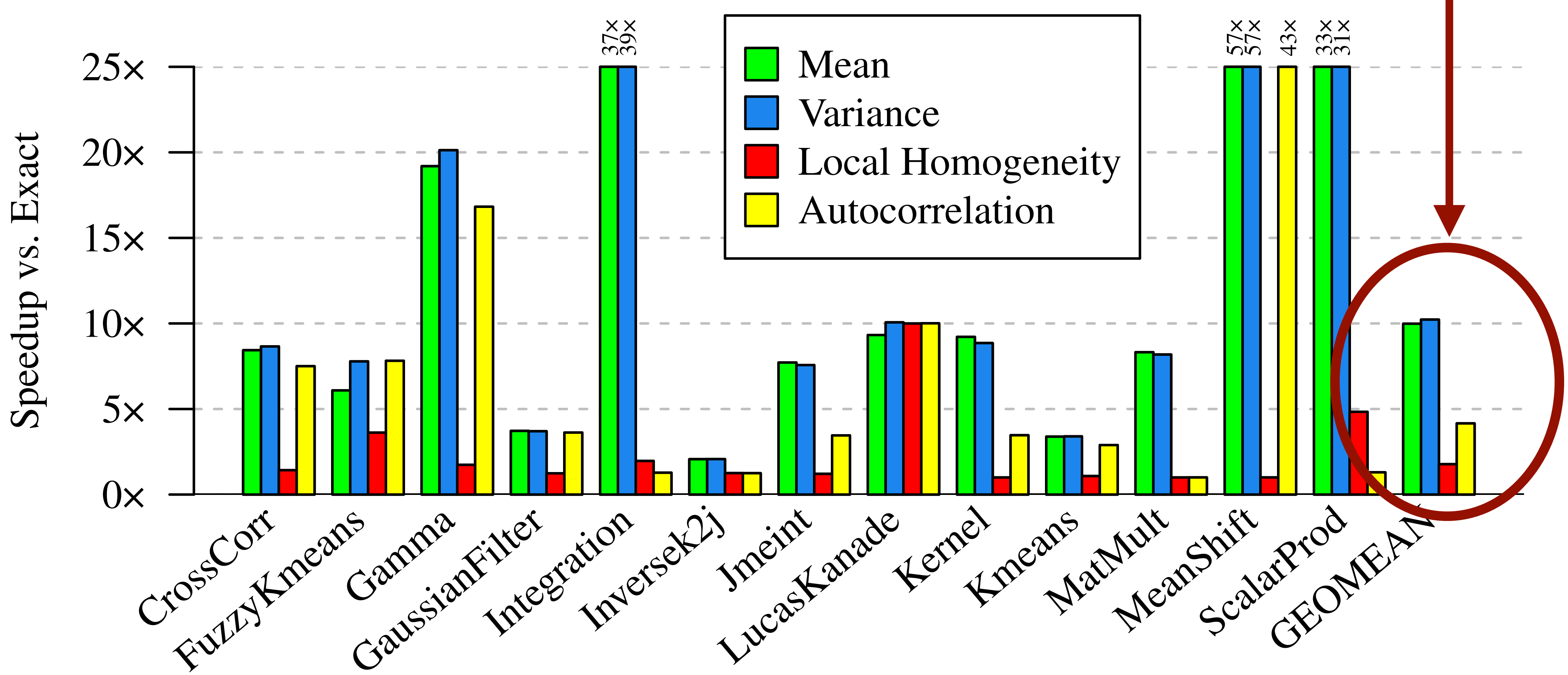
COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

Canary Metrics Comparison

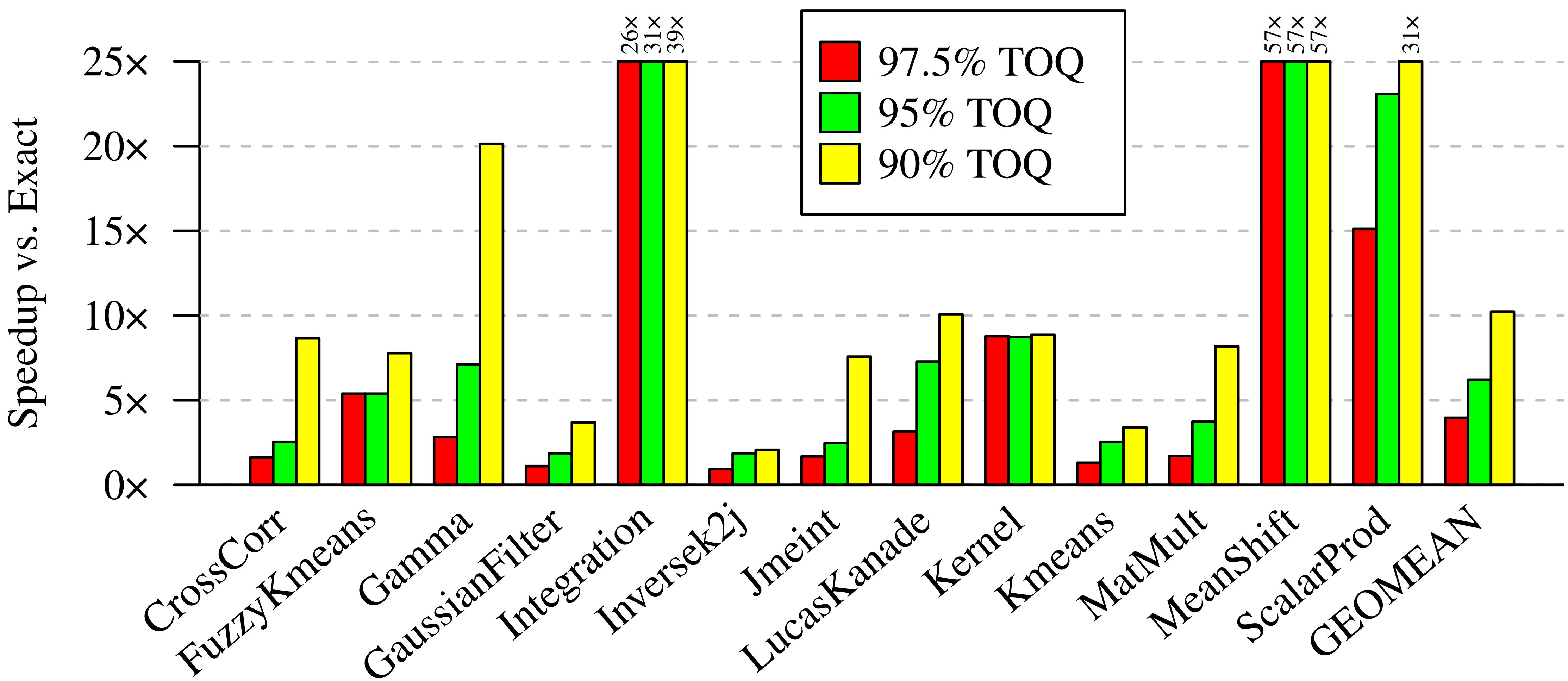
- End-to-end IRA speedup across 4 similarity metrics

Simpler metrics (mean, variance) yield improved end-to-end speedup



Differences in Output Quality

- Higher output quality ~ lower speedup

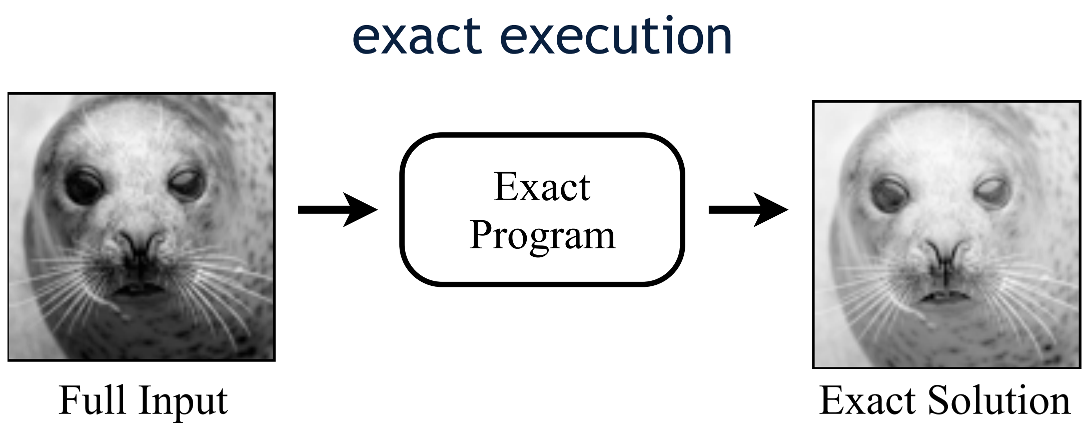


COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF MICHIGAN

ClarityLab

Input Responsive Approximation (IRA)

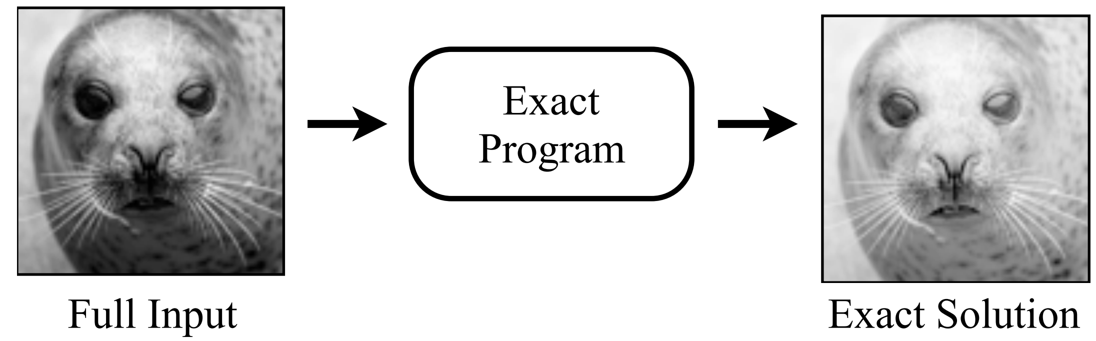
Input Responsive Approximation (IRA)



Input Responsive Approximation (IRA)

- Create a small *canary input* from full input

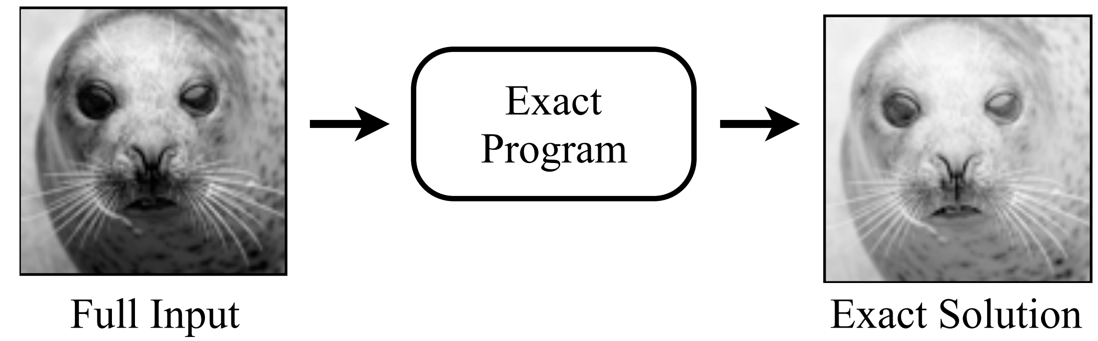
exact execution



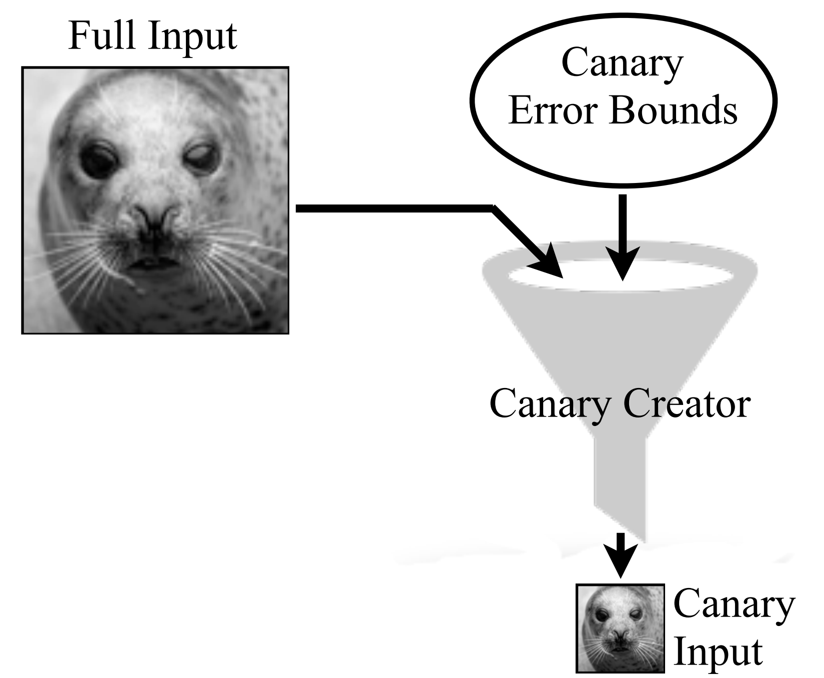
Input Responsive Approximation (IRA)

- Create a small *canary input* from full input

exact execution

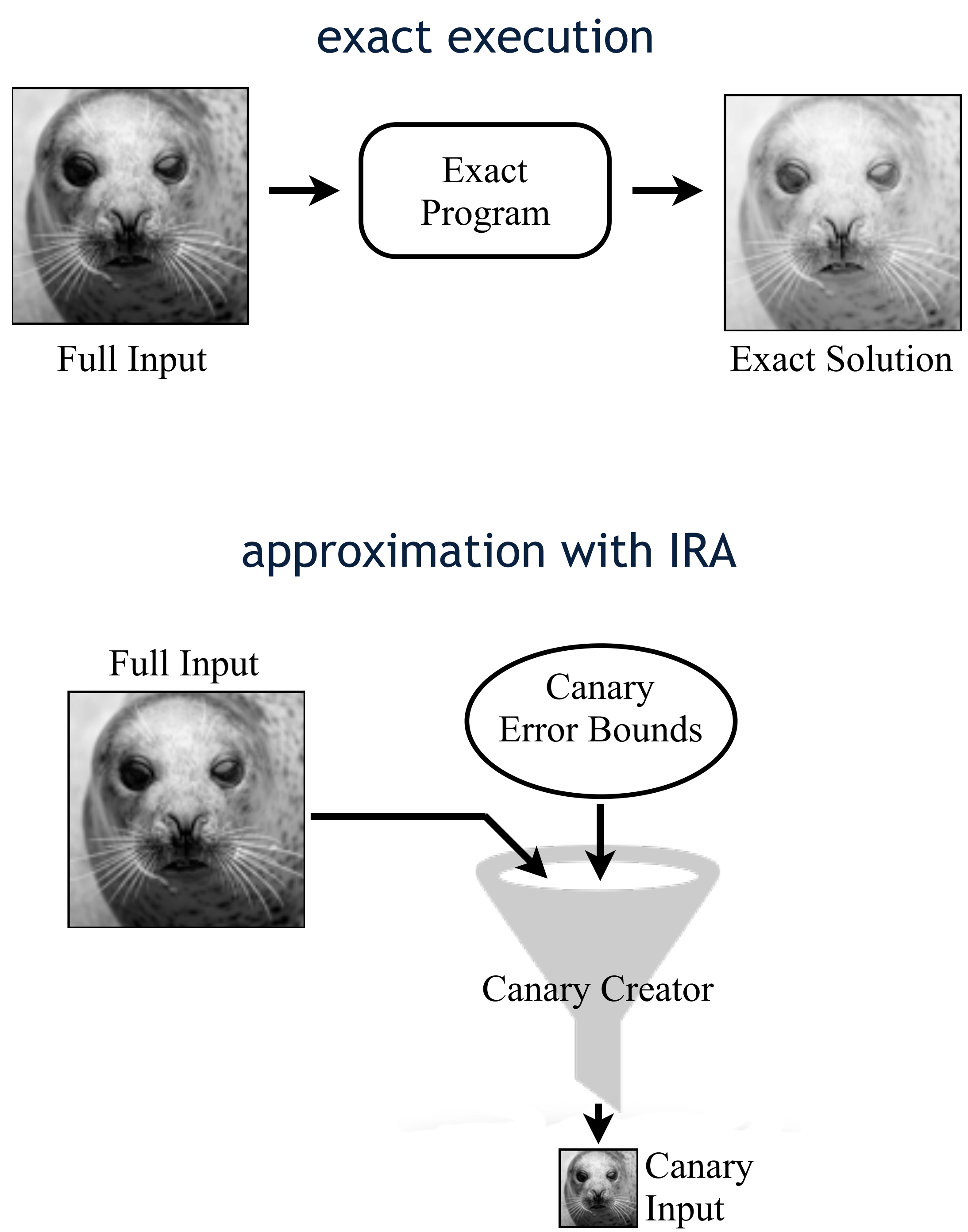


approximation with IRA



Input Responsive Approximation (IRA)

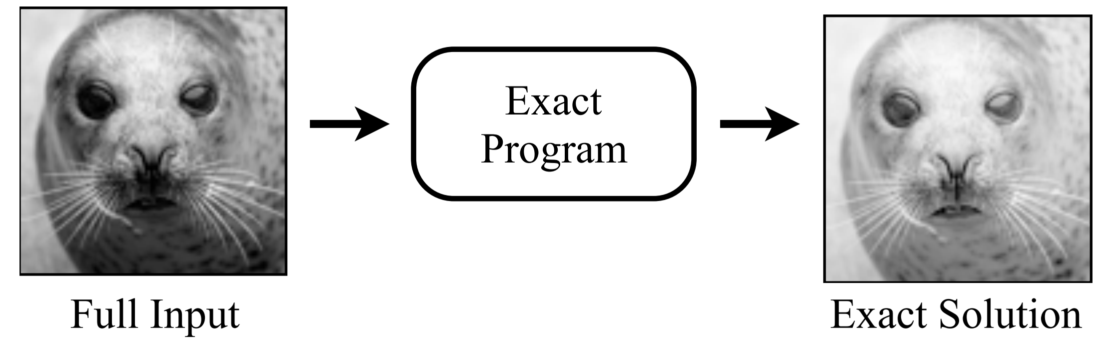
- Create a small *canary input* from full input
- Choose an approximation using canary



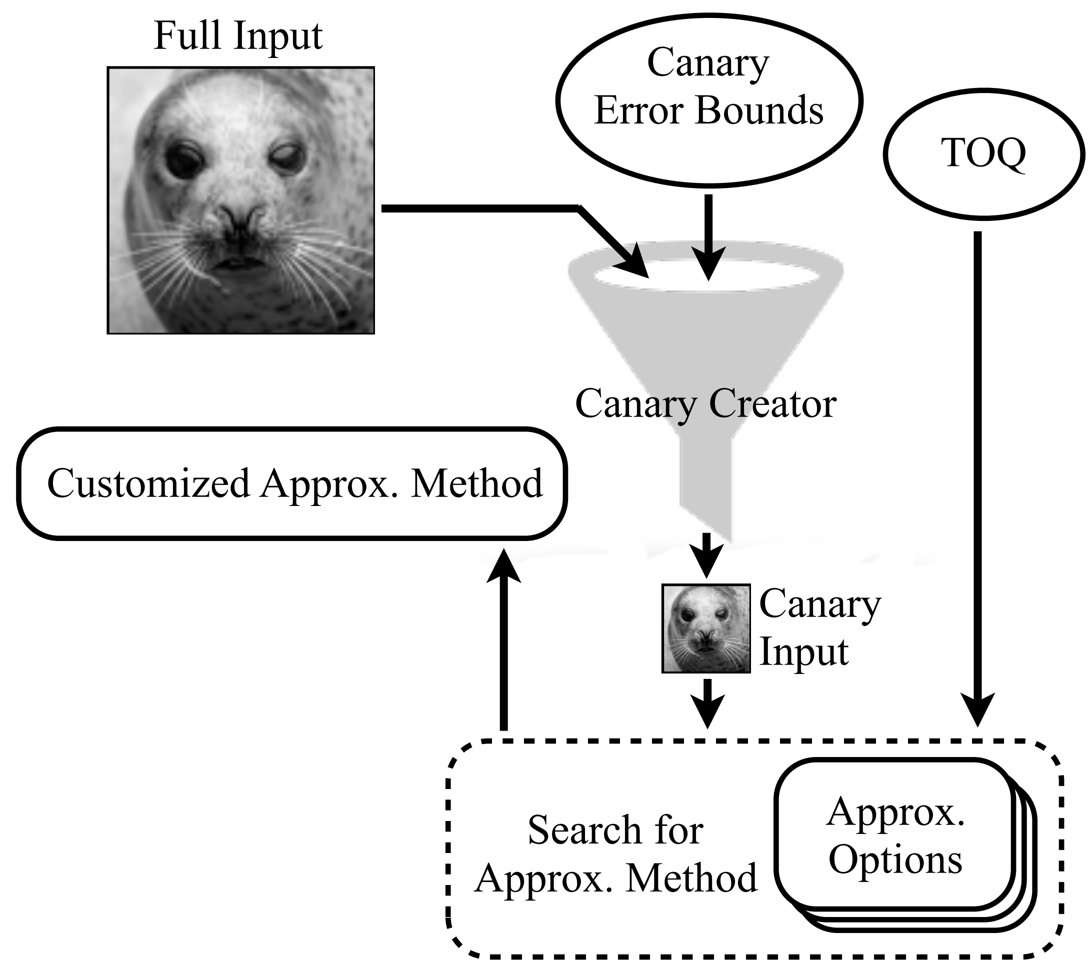
Input Responsive Approximation (IRA)

- Create a small *canary input* from full input
- Choose an approximation using canary

exact execution



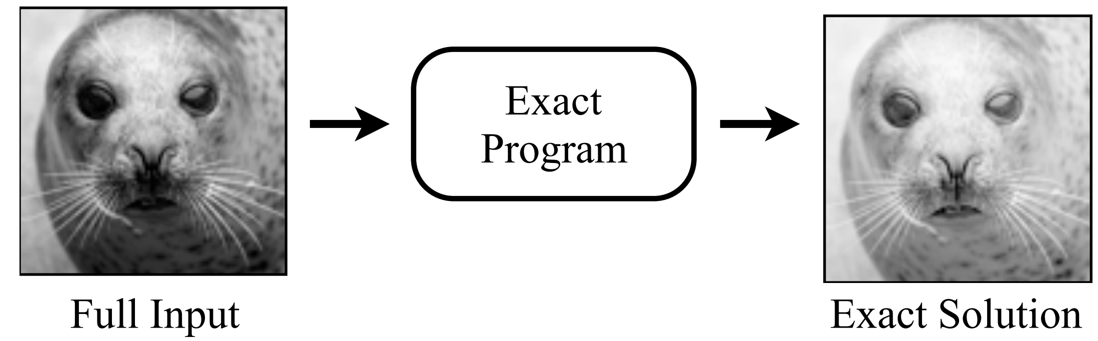
approximation with IRA



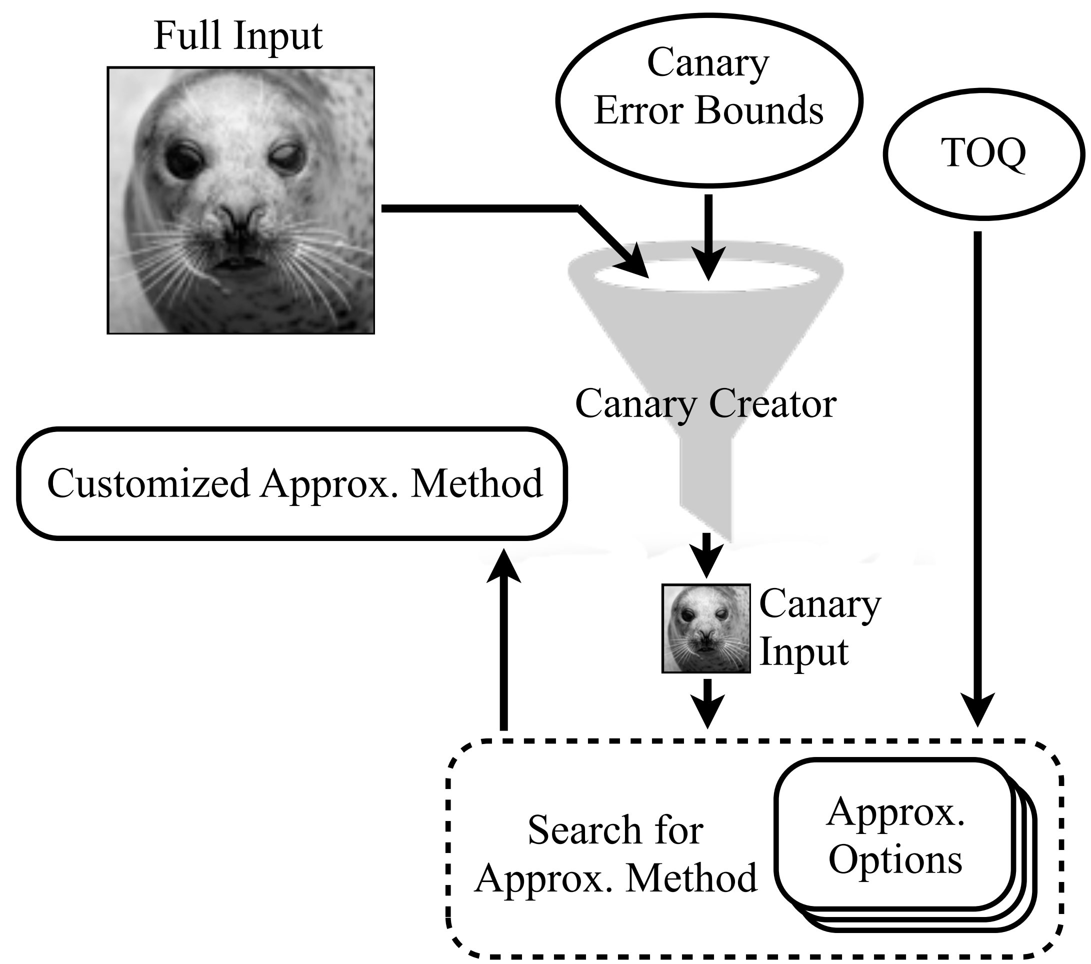
Input Responsive Approximation (IRA)

- Create a small *canary input* from full input
- Choose an approximation using canary
- Compute approximate solution on full input

exact execution



approximation with IRA



Input Responsive Approximation (IRA)

- Create a small *canary input* from full input
- Choose an approximation using canary
- Compute approximate solution on full input