# CORAL: Solving Complex Constraints for Symbolic PathFinder

Matheus Souza,
Mateus Borges,
Marcelo d'Amorim*

Federal University of Pernambuco
Recife-PE, Brazil

Corina Pasareanu

CMU SV/NASA Ames Research Center
Moffet Field-CA, USA

* presenter

# Context and Problem

- Context
  - Symbolic execution (SE)
- Problem

Program analysis technique to generate test input data for achieving high path coverage.

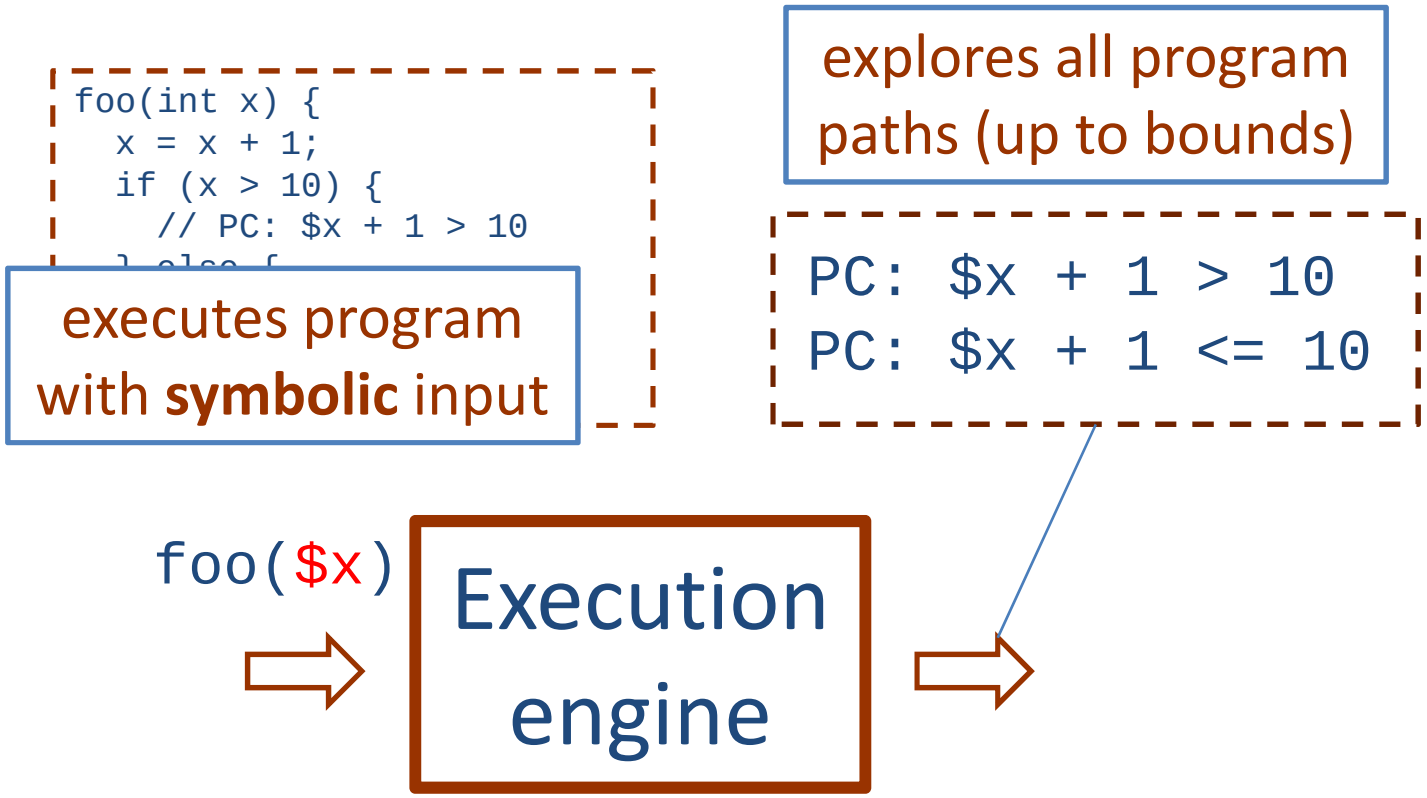
```
foo(int x) {
  x = x + 1;
  if (x > 10) {
    // …
  } else {
    // …
  }
}
```

```
foo(?)
foo(?)
```

# Context and Problem

- Context
  - Symbolic execution (SE)
- Problem

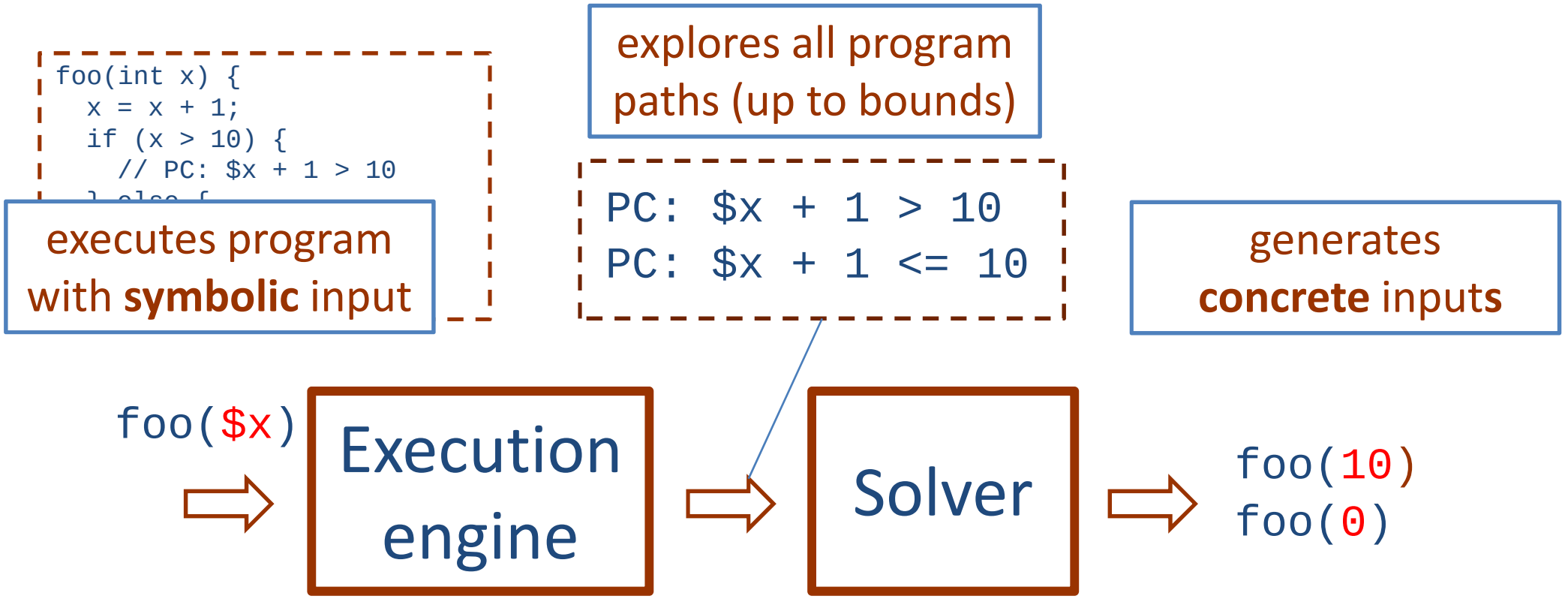
Program analysis technique to generate test input data for achieving high path coverage.



# Context and Problem

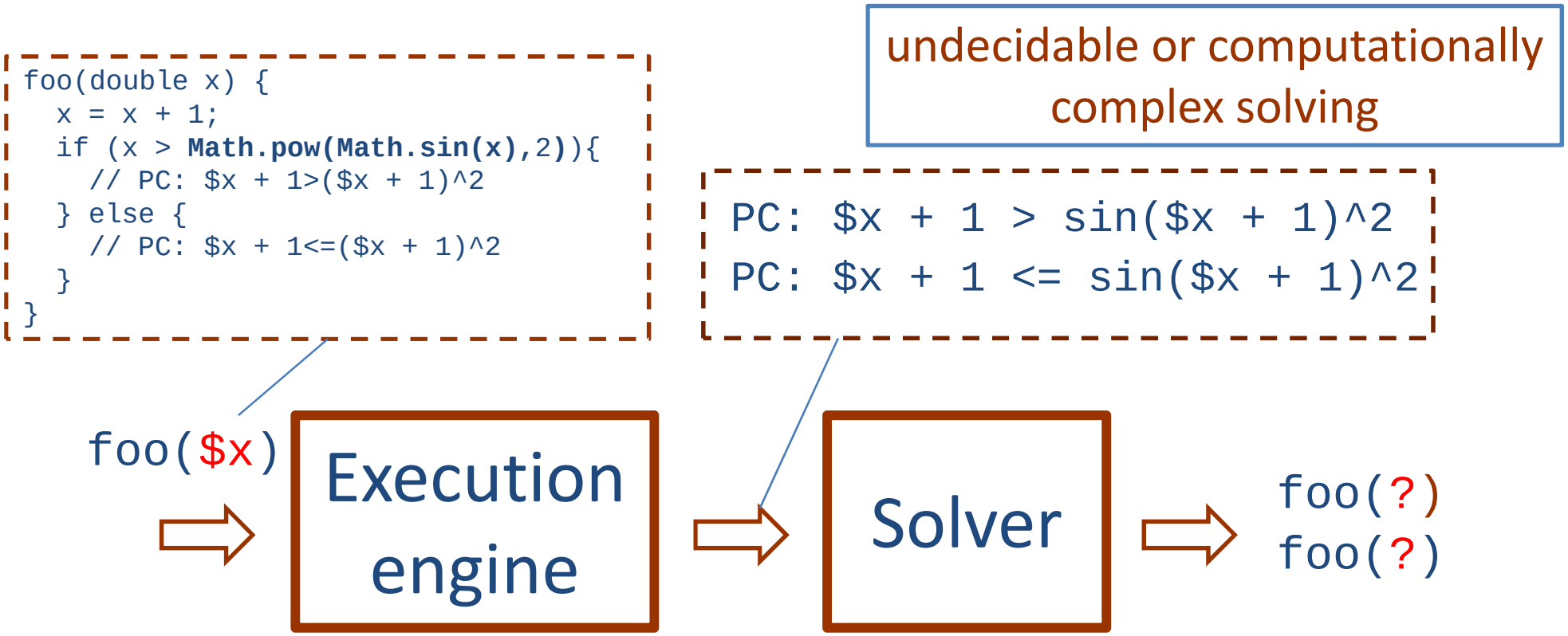
- Context
  - Symbolic execution (SE)
- Problem

Program analysis technique to generate test input data for achieving high path coverage.



# Context and Problem

- Context
  - Symbolic execution (SE)
- Problem
  - Inability of solvers to deal with complex constraints



# Context and Problem

- Context
  - Symbolic execution (SE)
- Problem
  - Inability of solvers to deal with

Program analysis technique to generate test input data for achieving high path coverage.

Implemented in Symbolic PathFinder (SPF) and used at NASA and Fujitsu.

# Note about SPF

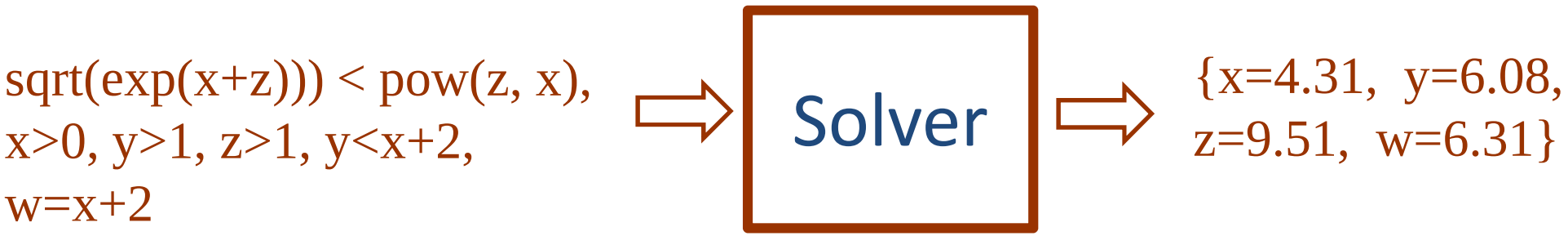
- Model-level interpretation of calls to math functions


Symbolic expression denoting the result value of the call

# CORAL solvers

- **Target applications**: SE of programs that manipulate floating-point variables
  - Use floating-point arithmetic
  - Call specific math functions (from java.lang.Math)

Common in software from NASA.

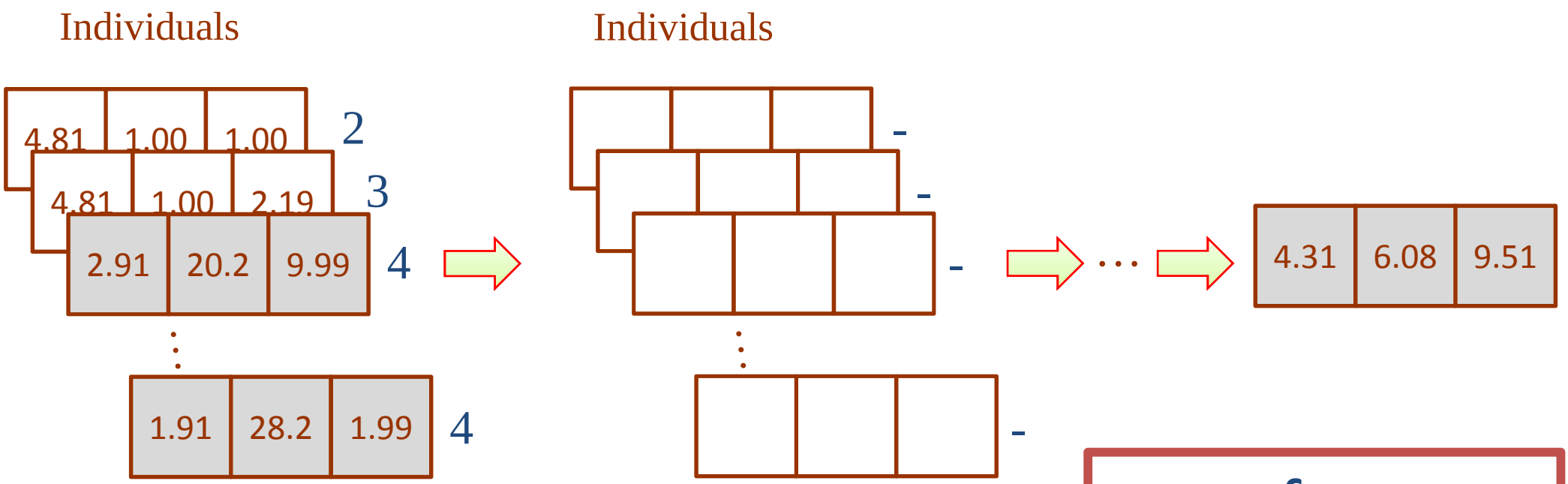


# Summary about CORAL

- Meta-heuristic solver
  - Distance-based fitness function
- Optimizations
  - Identification of dependent variables
  - Inference of variable domains
  - Efficient evaluation of constraints

# Quick outlook on heuristic search

**Input**: sqrt(exp(x+z))) < pow(z, x) , x>0, y>1, z>1, y<x+2
**Solution**: ?




Identify best fit individuals

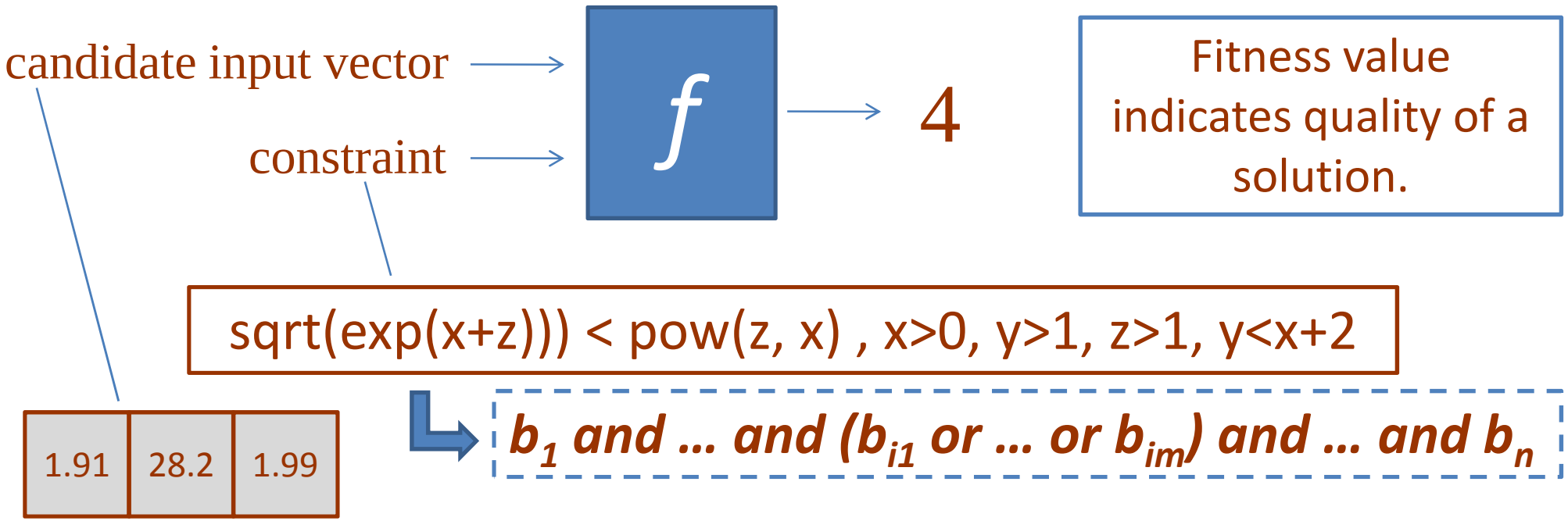
create new generations

stop after num. iterations or solution found

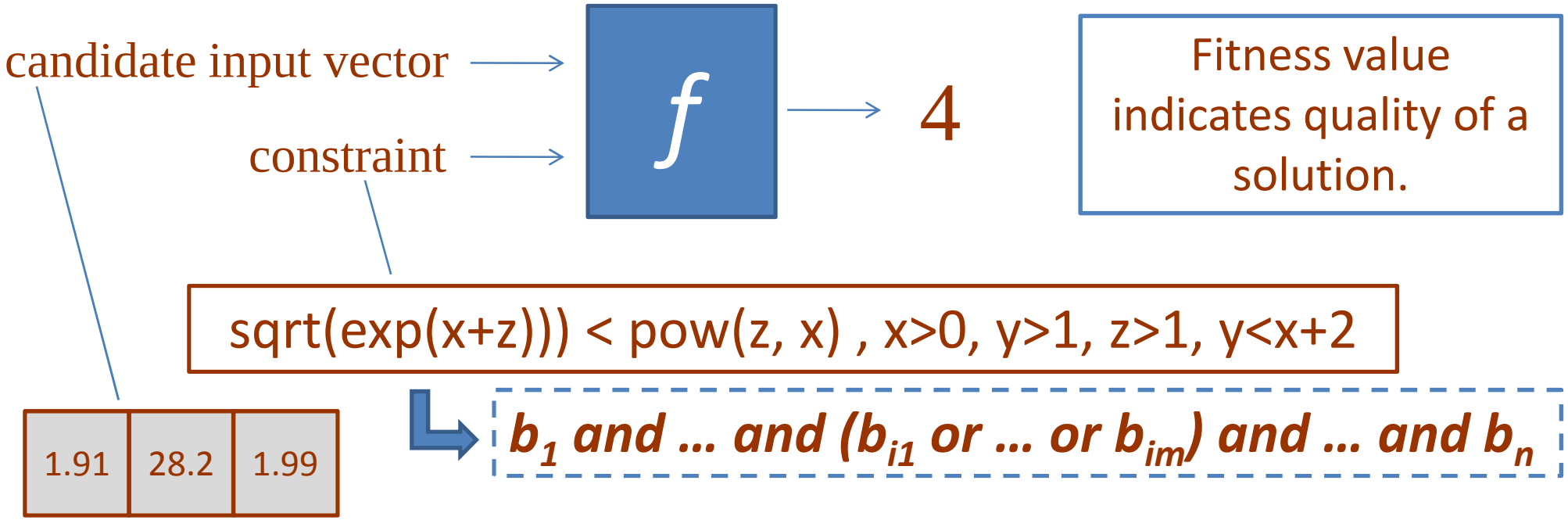
# Particle swarm optimization (PSO)

- Similar to GA, but with fixed-sized population
  - Search simulates movements in a group of animals
    - Implemented very efficiently (matrix operations)
    - Parameters to calibrate local and global influence
- Used opt4j library (see opt4j.sourceforge.net)

# Fitness Function



# Fitness Function

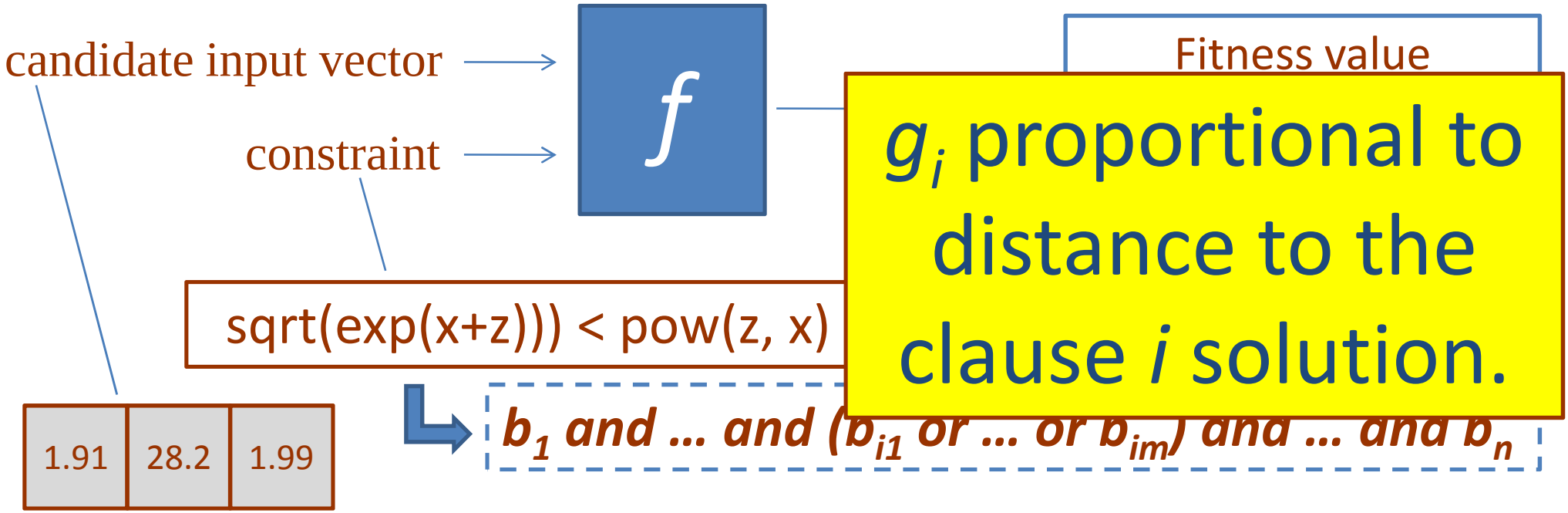



$$f(\vec{x}) = \sum_i w_i * g_i(\vec{x})$$

$$g_i(\vec{x}) = \max_{1<j<m} 1 - d(b_{ij}, \vec{x})$$

Adapted SAW [T. Back et *al.*, 1998]

# Fitness Function

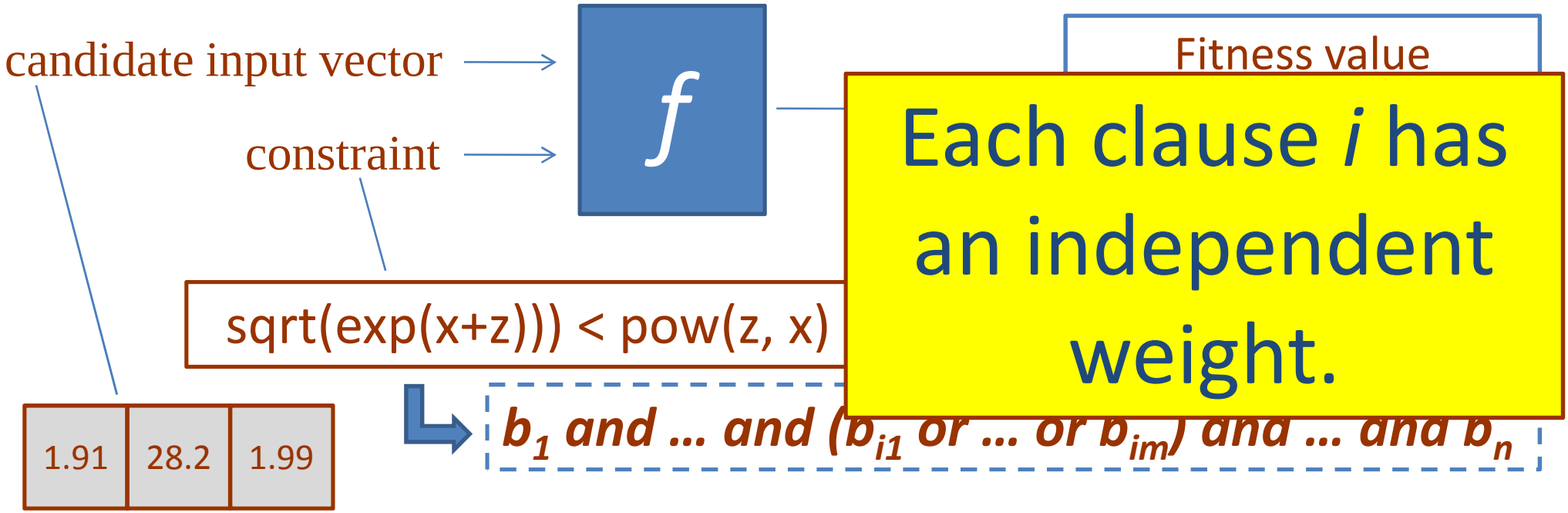



$$f(\vec{x}) = \sum_i w_i * g_i(\vec{x})$$

$$g_i(\vec{x}) = \max_{1<j<m} 1 - d(b_{ij}, \vec{x})$$

Adapted SAW [T. Back et *al.*, 1998]

# Fitness Function




$$f(\vec{x}) = \sum_i w_i * g_i(\vec{x})$$

$$g_i(\vec{x}) = \max_{1<j<m} 1 - d(b_{ij}, \vec{x})$$

Adapted SAW [T. Back et *al.*, 1998]

# RESULTS

# Evaluation

- Impact of search algorithm and optimization
- Effectiveness to solve…
  - simple constraints
  - complex constraints
- NASA case studies
  - PISCES
  - Apollo Lunar Autopilot

# Impact of search and optimizations

- Comparison of 4 variations of CORAL
  - {PSO,Random} x {optimized, non-optimized}

| | pso-opt | pso | random-opt | random |
|---|---|---|---|---|
| pso-opt | - | 74 | 88 | 257 |
| pso | 13 | - | 71 | 196 |
| random-opt | 3 | 47 | - | 180 |
| random | 0 | 0 | 8 | - |

Number of constraints that solver on row 4 solved and solver on column 3 did not solve

# Impact of search and optimizations

- Comparison of 4 variations of CORAL
  - {PSO,Random} x {optimized, non-optimized}

| | pso-opt | pso | random-opt | random |
|---|---|---|---|---|
| pso-opt | - | **74** | 88 | 257 |
| pso | **13** | - | 71 | 196 |
| random-opt | 3 | 47 | - | **180** |
| random | 0 | 0 | **8** | - |

optimization pays off on average

# Impact of search and optimizations

- Comparison of 4 variations of CORAL
  - {PSO,Random} x {optimized, non-optimized}

| | pso-opt | pso | random-opt | random |
|---|---|---|---|---|
| pso-opt | **-** | 74 | 88 | 257 |
| pso | **13** | - | 71 | 196 |
| random-opt | **3** | 47 | - | 180 |
| random | **0** | 0 | 8 | - |

pso-opt performs better on average

# Effectiveness on simple constraints

- Constraints involving decidable theories
  - Bounded-exhaustive testing of BST and Treemap
- Compared CORAL with Choco, CVC3, and Yices

CORAL could solve as many constraints as any other.

# Effectiveness on complex constraints

- 78 manually-written test cases with math functions
- Compared CORAL with Choco
- CORAL solved 92.3% of the constraints (707/766)

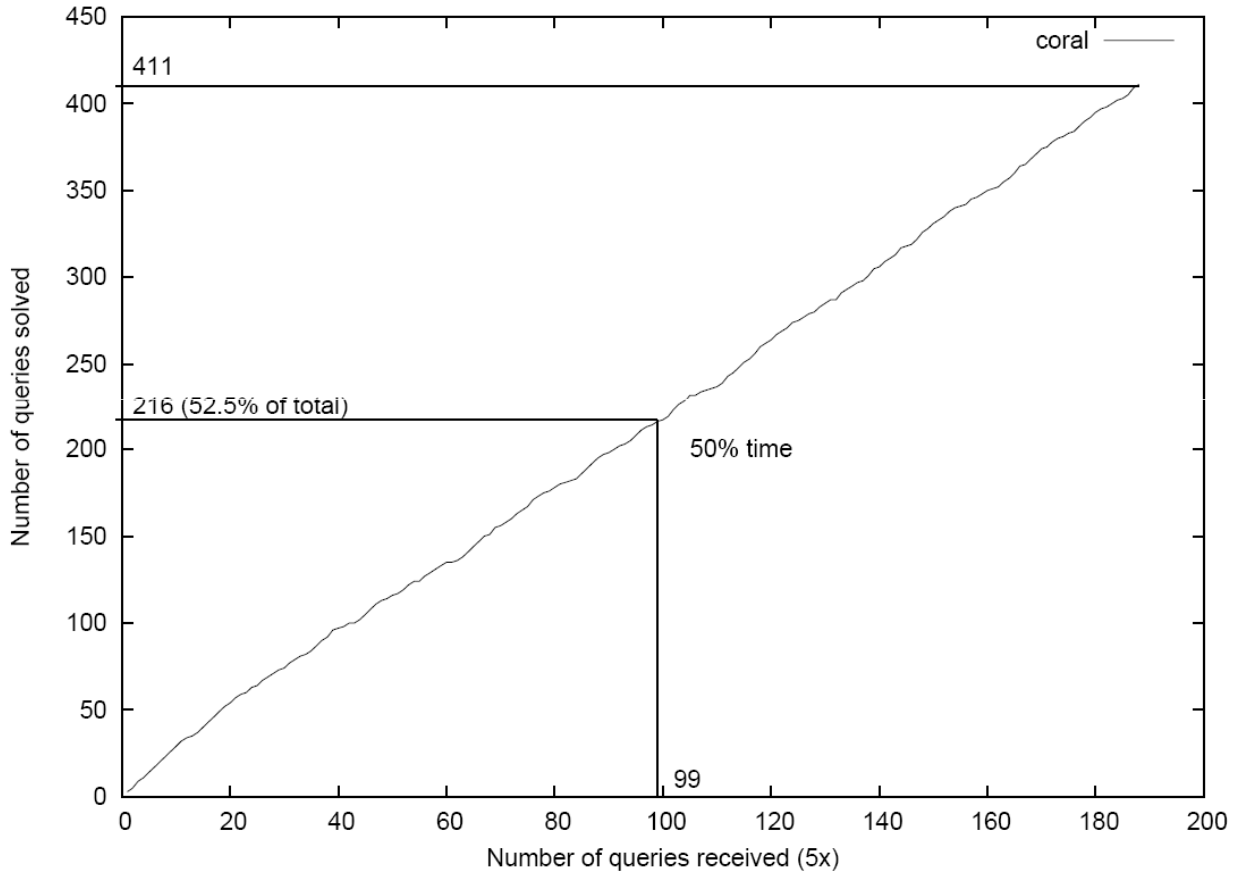
For no query CHOCO could solve and CORAL could not.

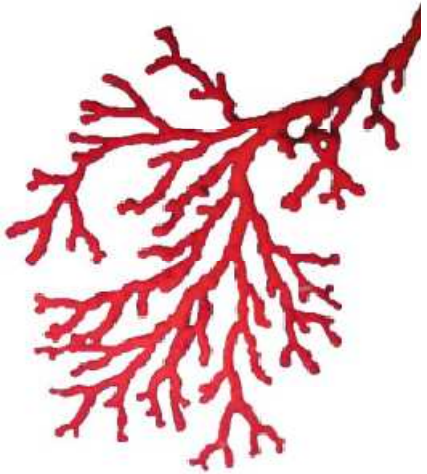
# PISCES

- PISCES library
  - Some of the computations involved: hyperbolic (arc) sine, cosine, tangent, floating point reminder.
- Analyzed 20 methods with SPF+CORAL
- Found undocumented pre-conditions
  - Illegal arguments not properly caught in code

# Apollo Lunar Autopilot

- Simulink model translated to Java
- Bounds
  - max. time = 2h, max. path condition length = 50





# CORAL

**For more information visit**

PAN

pan.cin.ufpe.br/coral