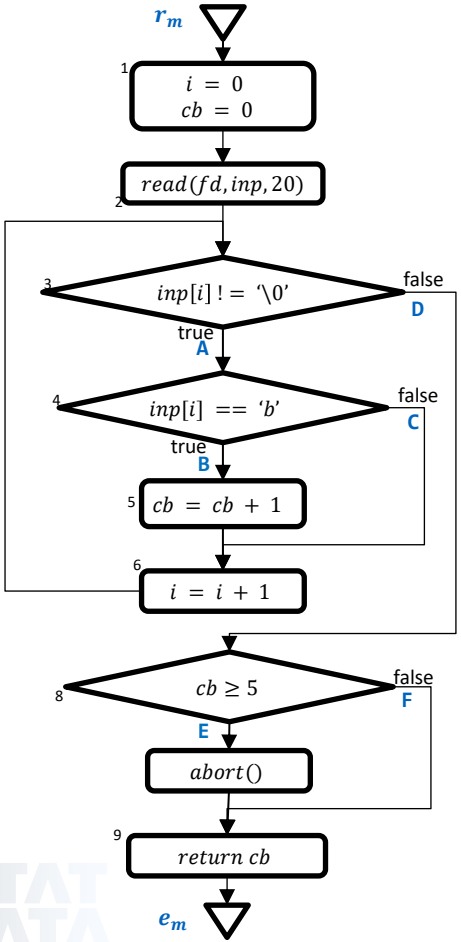
An Illustration of AFL

M. Raveendra Kumar, TCS Research

Coverage Guided Fuzzing – Working example

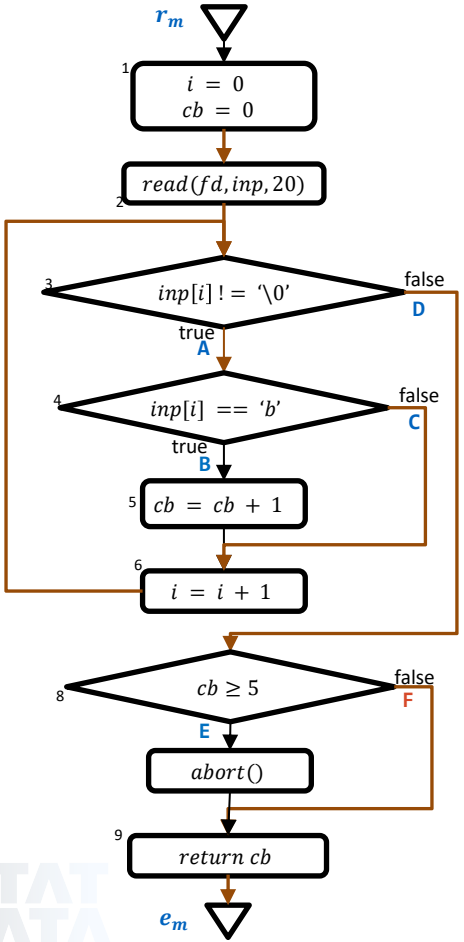


AFL aims for *branch-pair* coverage.

A *branch-pair* is a sequence of two consecutive branches in the program.

AB	AC	BA	CA	BD	CD	DE	DF

Coverage Guided Fuzzing – Working example

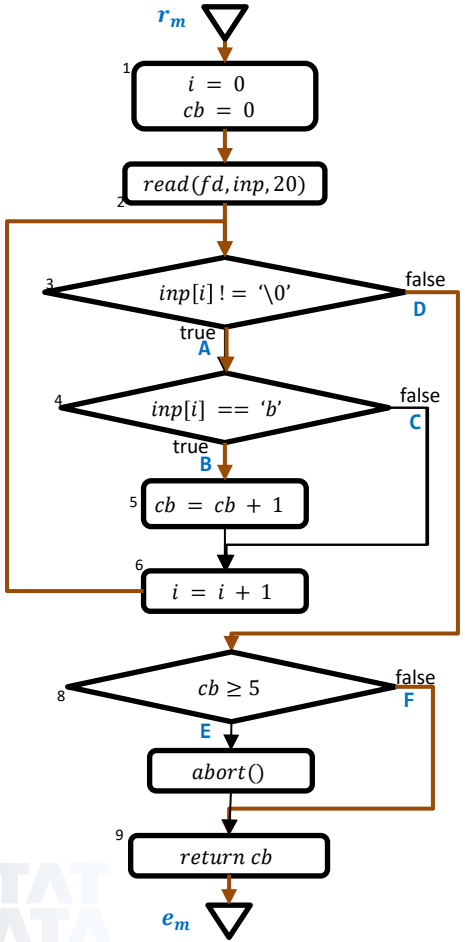


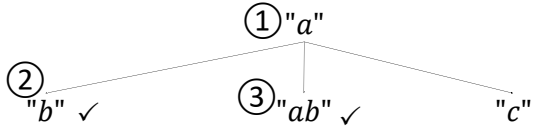
Initial input ①"a"

Evaluation

Id	input	AB	AC	BA	CA	BD	CD	DE	DF
1	"a"		1				1		1

Coverage Guided Fuzzing – Working example



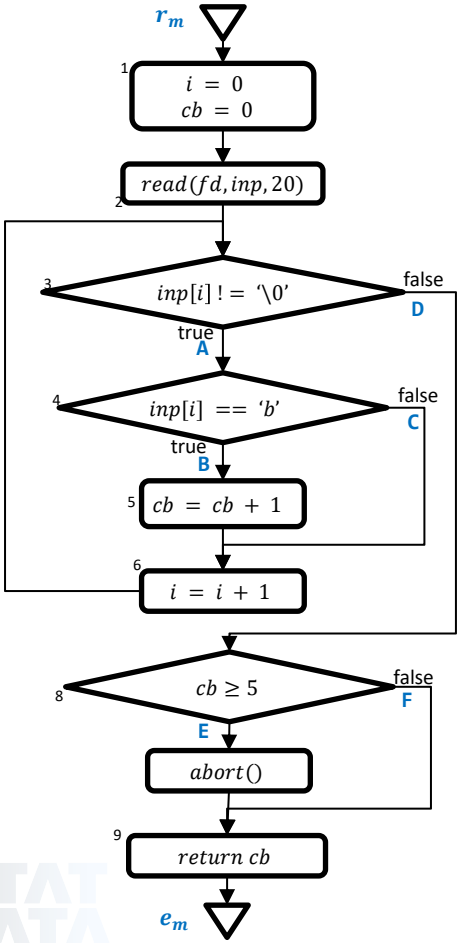


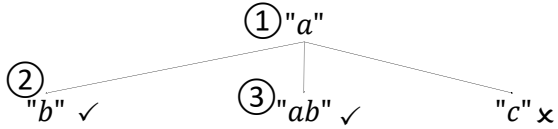
Fitness check

Id	input	AB	AC	BA	CA	BD	CD	DE	DF
1	"a"		1				1		1
2	"b"	1				1			1
3	"ab"	1	1		1	1			1
	"c"		1				1		1

Fitness criteria: Each input file *f* that is retained in the queue covers some branch-pair *significantly different* number of times than *all* input files that were generated and retained *before f*.
(Any generated file that does not meet this requirement is thrown away.)

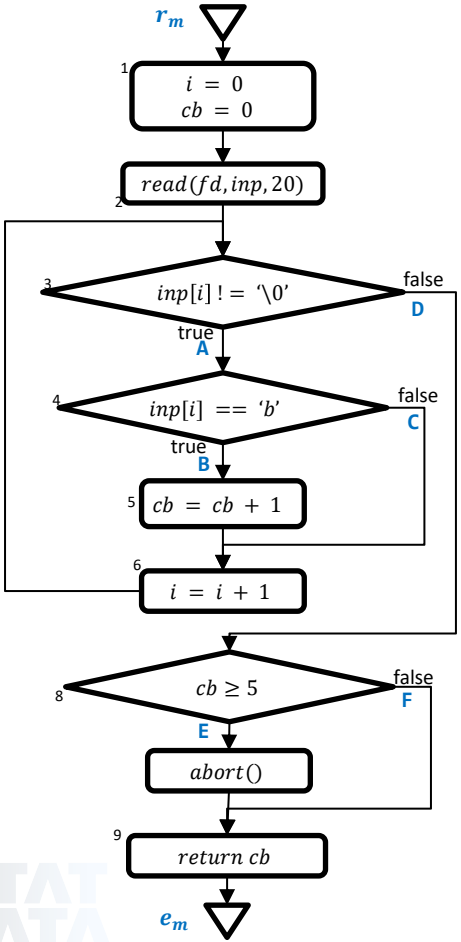
Coverage Guided Fuzzing – Working example

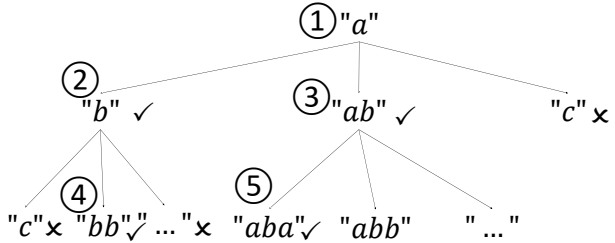




Id	input	AB	AC	BA	CA	BD	CD	DE	DF
1	"a"		1				1		1
2	“b”	1				1			1
3	“ab”	1	1		1	1			1
✕	“c”		1				1		1

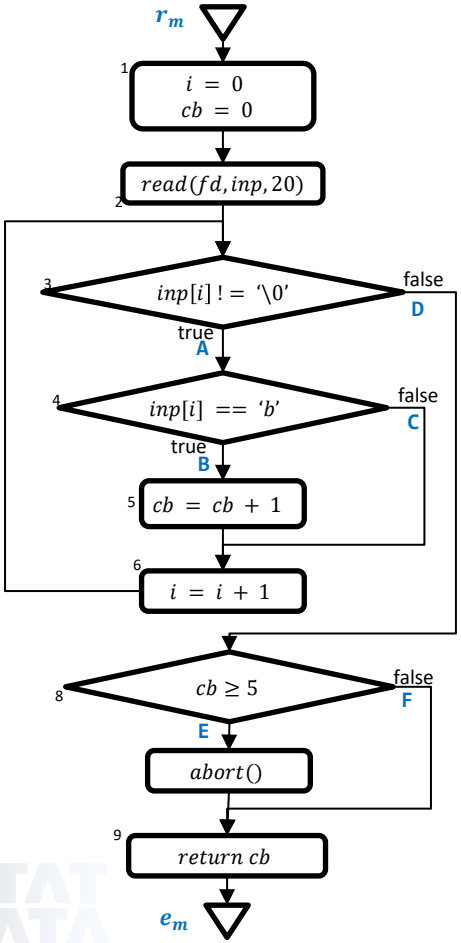
Coverage Guided Fuzzing – Working example

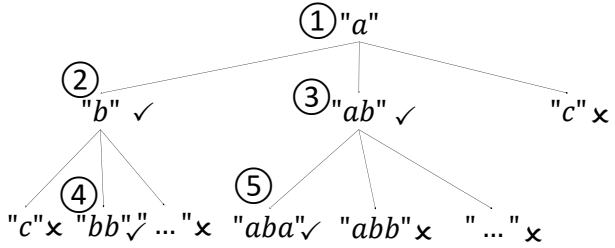




Id	input	AB	AC	BA	CA	BD	CD	DE	DF
1	"a"		1				1		1
2	“b”	1				1			1
3	“ab”	1	1		1	1			1
4	“bb”	2		1		1			1
5	“aba”	1	2	1	1		1		1
	“abb”	2	1	1	1	1			1

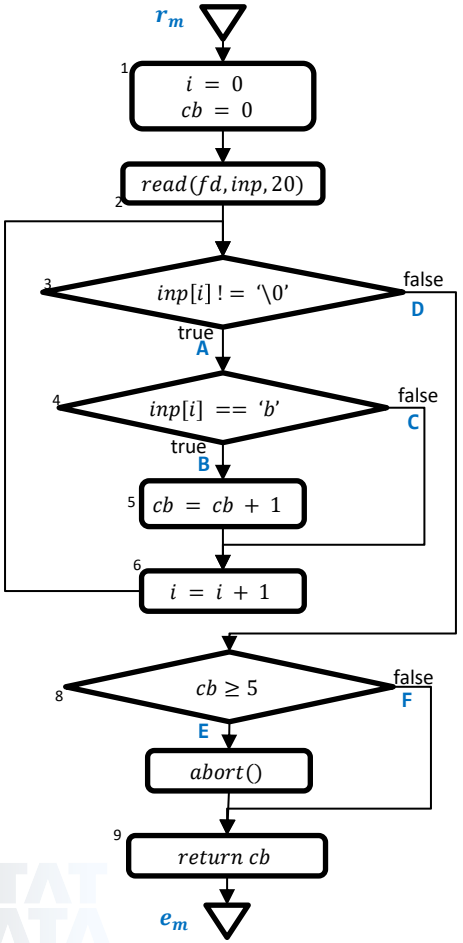
Coverage Guided Fuzzing – Working example

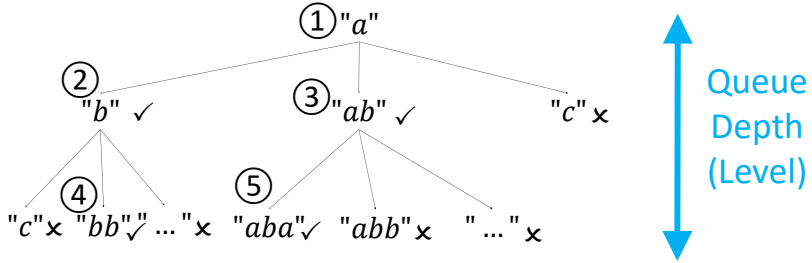




Id	input	AB	AC	BA	CA	BD	CD	DE	DF
1	"a"		1				1		1
2	"b"	1				1			1
3	"ab"	1	1		1	1			1
4	"bb"	2		1		1			1
5	"aba"	1	2	1	1		1		1
	"abb"	2	1	1	1	1			1

Coverage Guided Fuzzing – Working example



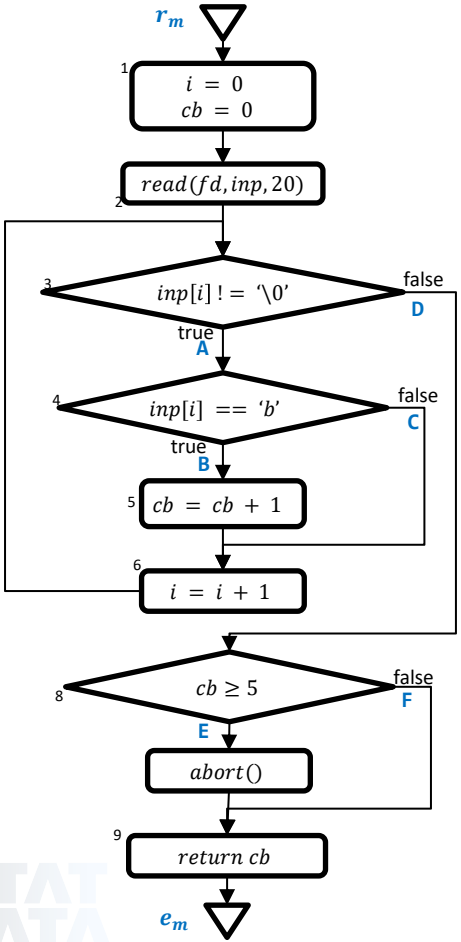


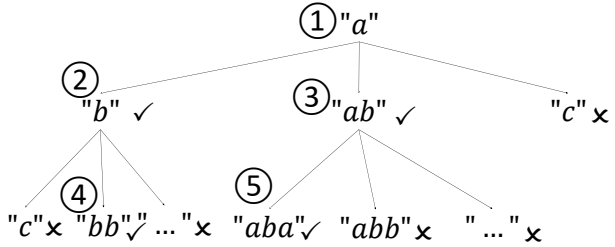
Id	input	AB	AC	BA	CA	BD	CD	DE	DF
1	"a"		1				1		1
2	"b"	1				1			1
3	"ab"	1	1		1	1			1
4	"bb"	2		1		1			1
5	"aba"	1	2	1	1		1		1
	"abb"	2	1	1	1	1			1

Queue

Cycle

Coverage Guided Fuzzing – Key Questions





- ✓ How to produce new test inputs from existing one?
  - E.g. How to produce "b", "ab", "c" from "a“ ?
- ✓ *Among the generated inputs, which ones to retain? (answered)*
- ✓ Which of the test inputs among the retained test inputs should be selected for fuzzing?
  - E.g. which one among "b", "a", "ab", "c" should be selected for fuzzing ?
- ✓ How many new test inputs should be produced from a given test input?
  - How to determine that 3 inputs to be generated from “a”

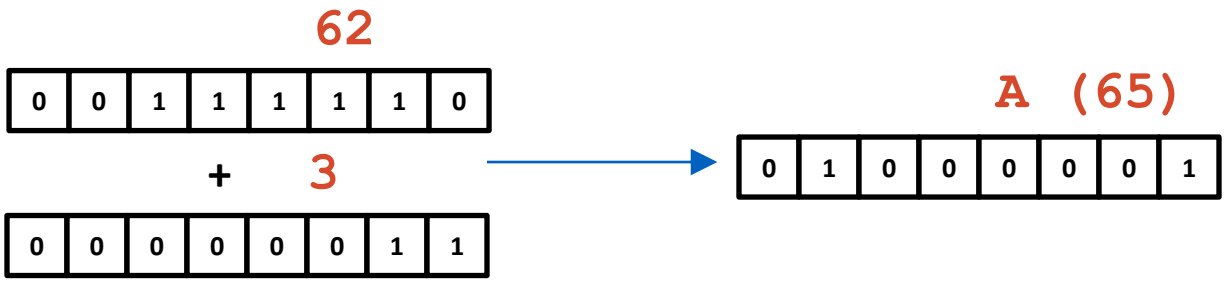
Mutation operations -- examples

Walking bit flip operations

Single bit

Two bits

Arithmetic



Deterministic: flip the bits in a fixed order starting from left most bit position.

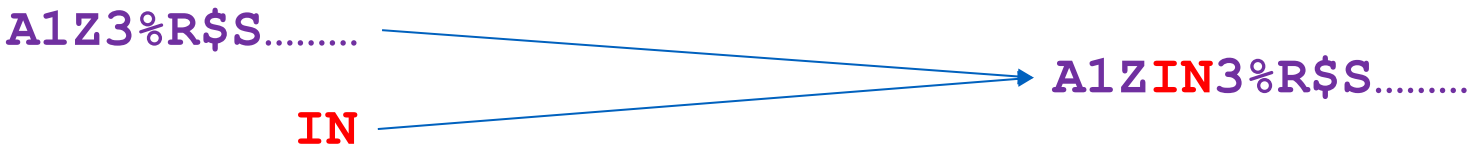
Non-deterministic: flip bits at a random location

Random Mutation operations

S.No	Class of mutation operation	Randomness
1	Walking bits	1. Selecting a byte, 2. Selecting a flip position
2	Setting interesting value	1. Selection of interesting value, 2. Selection location in the input.
3	Subtraction / addition of a value	1. Selection of value to subtract/add. 2. Selection of Location in the input
4	Adding/Deleting the content	1. Location to add/subtract 2. Length of data to add/subtract. 3. Value of byte to add
5	Clone bytes	1. From location 2. To location 3. Clone size
6	Set to random value	1. From location 2. To location 3. Size and Value
7	Override data	1. From location 2. To location 3. Size and value

Example Mutation operations

Adding and deleting the content



Clone the content from a random location to another random location

Set random value at a random location

Override data at a random location

High-level Flow