BugSleuth

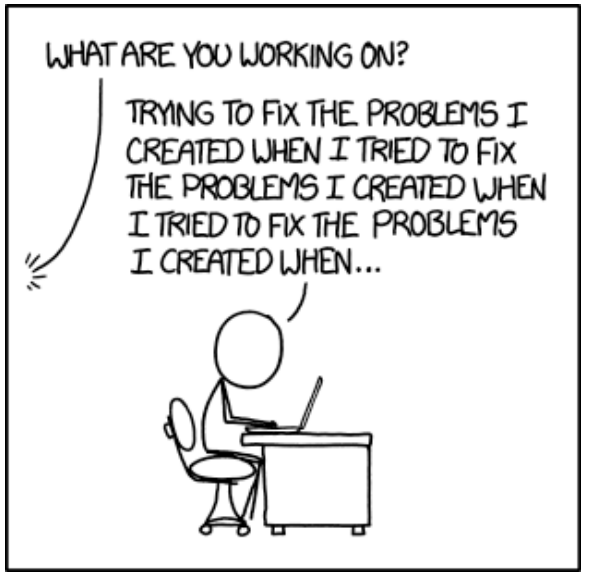
Ashish Ramrakhiani

Nat Bourassa

Sunayana Sanam

Problem

Source: https://xkcd.com/1739/



- Software practitioners spend a lot of time debugging
- Existing Automated Fault localization (FL) techniques
- Reluctance for acceptance in the software industry – suboptimal performance
- Research shows combining FL techniques using machine learning improves FL performance [1]
- Learning to rank algorithm – a huge labelled dataset required
- Is this learning necessary ?

[1] D.Zou, J.Liang,Y.Xiong,M.D.Ernst, andL.Zhang.Anempirical studyof fault localizationfamiliesandtheircombinations. TSE,2019.

Problem

- We propose, BugSleuth.
- An unsupervised fault localization technique that combines multiple ranked lists of suspicious statements
- Uses Rank Aggregation algorithms -
  a) Genetic Algorithm
  b) Cross Entropy Monte Carlo Algorithm
- Provides an optimal ranked list of suspicious statements thereby improving a practitioner's productivity by detecting bugs in their code

Research questions

RQ1: How effective are rank aggregation algorithms for localizing defects

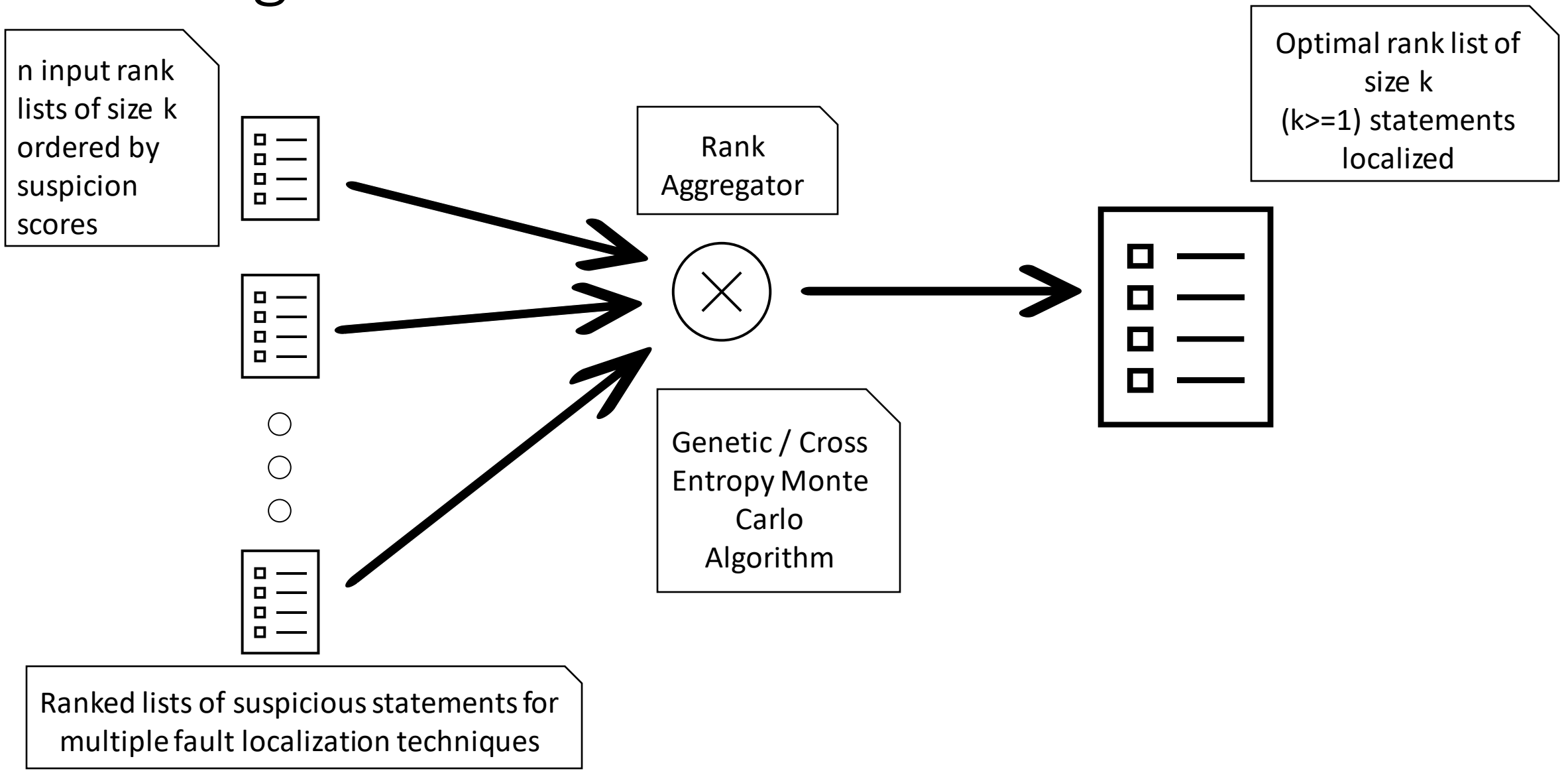
(A) Genetic Algorithm

(B) Cross Entropy Monte Carlo Algorithm

RQ2: How efficient are rank aggregation algorithms for fault localization

RQ3: How do these rank aggregation algorithms compare against state-of-the-art fault localization techniques

Design


n input rank lists of size k ordered by suspicion scores
Rank Aggregator
Optimal rank list of size k (k>=1) statements localized
Genetic / Cross Entropy Monte Carlo Algorithm
Ranked lists of suspicious statements for multiple fault localization techniques

Evaluation

Dataset

Defects4J (v2.0) benchmark to evaluate our FL technique

Defects4J consists of 835 reproducible defects from 17 large open-source Java projects.

Each defect comes with

(1) one buggy and one developer-repaired version of the project code

(2) a set of developer written tests, all of which pass on the developer-repaired version and at least one of which evidences the defect by failing on the buggy version and

(3) defect information

Evaluation

Metrics

We will use two metrics, common to FL evaluations

Top-N is the number of defects localized in the top-n ranked statements

- It tells us how efficient our FL technique is in localizing bugs
- Top-1, Top-3 and Top-5

EXAM is the fraction of ranked statements one must inspect before finding a buggy statement.

- EXAM tells us how high the buggy statements are ranked
- saves practitioner's time and resources otherwise spent on manual bug detection

Plan

Task No.	Task Name	Duration	Assigned to
1	Implement Genetic Algorithm	2 weeks (01/29 - 02/10)	Ashish
2	Implement Cross Entropy Monte Carlo Algorithm	2 weeks (01/29 - 02/10)	Nat
3	Finetune Genetic Algorithm	1 week (02/11 to 02/17)	Ashish
4	Finetune Cross Entropy Monte Carlo Algorithm	1 week (02/11 to 02/17)	Nat
5	Create Ground Truth data from Defects4J dataset	1 week (01/29 - 02/03)	Sunayana
6	Develop scripts to calculate Top-N and EXAM	1 week (02/03 - 02/10)	Sunayana
7	Gather input rank lists from multiple FL techniques and document total run time	1 week (02/11 - 02/17)	Sunayana
8	Finalize state-of-the-art techniques for evaluation and gather results for comparison	2 weeks (02/18 - 03/02)	Ashish, Nat and Sunayana

Thank You