1. What do affinity lists show that enables you to find a bug with them?

If we can see many of the predicates in an affinity list appear on the same line, or in the same function, or mention the same variable, then this is can direct our attention toward suspicious code. Even if the specific details of any one predictor are not informative, the broad patterns sometimes are.

1. The place of the problem code
2. A branch or other predicate after which a crash occurs

2. Why might multiple predicates appear on the affinity list for the same bug?

Because we have a list of predicates based on the effective score of each predicate. If score for some predicates is high, we can see all of them. Alternatively, maybe we have correlated predictors.

3. How could a tool like CBI be used to find bugs in software that you write? Explain the

CBI process and how it’s useful in the “real world”. You may use an example from your

professional, academic, or hobby experience or you may make up a hypothetical

example.

Most software have errors. Some of these errors are detected as a result of the user's work. Cooperative Error Isolation (CBI) strives to use the vast amount of computing performed by end users of the software. Gathering a bit of information each time a program runs by its user, munity, we can automatically draw conclusions about the causes of errors found in field.

4. Explain the process you used to find and fix bugs based on the CBI report. We encourage

you to be specific about how the report influenced your testing and debugging. A good

response should include a sample test case (a command that you would run outside of

CBI to demonstrate a bug found with CBI).

For example jpegtran

in my opinion the main use of the CBI in fact I can direct the my attention toward suspicious code.

Search bug in jpegtran  
After the first test, I see a problem branch

(int )info->transform == 2 is TRUE

I’m starting to explore what happens if this is true. I note the wrong memory allocation and fix it.