Oracle-free Testing with Two-layer Covering Arrays

Rick Kuhn

National Institute of
Standards and Technology
Gaithersburg, MD

East Carolina University

NSF Research Experiences for Undergraduates

June 29, 2015

Some current approaches

- Fuzz testing
  - crash system w/ random values, then analyze memory dump
  - Good for major faults that cause crashes
- Metamorphic testing –
  - e.g. $cos(x) = cos(x+360)$, so compare outputs for both, with a difference indicating an error
  - Good for numerical software
- Partial test oracle –
  - e.g., insert element x in data structure *S* check x in S after
  - Usually not fully automatable

New method

- Consider equivalence classes
- Example: shipping cost based on distance *d* and weight *w*, with packages < 1 pound are in one class, 1..10 pounds in another, 10 in a third class.
- Then for cost function $f(d,w)$, $f(d, 0.2) = f(d, 0.9)$, for equal values of *d*.
- *But* $f(d, 0.2) \neq f(d\ 5.0)$, because two different weight classes are involved.

Basic property of equivalence classes

when a_1 and a_2 are in the same equivalence class,

$$f(a_1,\text{b,c,d,}\ldots) \approx f(a_2,\text{b,c,d,}\ldots),$$

where $\approx$ is equivalence with respect to some predicate.

If not,

- then either the code is wrong,
- or equivalence classes are not defined correctly.

Can we use this property for testing?

- Let's do an example: access control. access is allowed if (1) subject is employee and time is in working hours and it's a weekday; or (2) subject is an employee with administrative privileges; or (3) subject is an auditor and it is a weekday.

- Equivalence classes for time of day and day of the week
- time = minutes past midnight (0..0539), (0540..1020), (1021..1439).
- Days of the week weekend and weekdays, designated as (1,7) and (2..6) respectively.

Code we want to test

```
int access_chk() {
    if (emp && t >= START && t <= END &&
          d >= MON && d <= FRI) return 1;
    else
    if (emp && p) return 2;
    else
    if (aud && d >= MON && d <= FRI)
         return 3;
    else
    return 0;
}
```

Establish equivalence classes

emp: boolean
day: (1,7), (2,6) A1 A2
time: (0,100,539),(540,1020),(1021,1439) B1 B2 B3
priv: boolean
aud: boolean

emp (bool) : 0,1

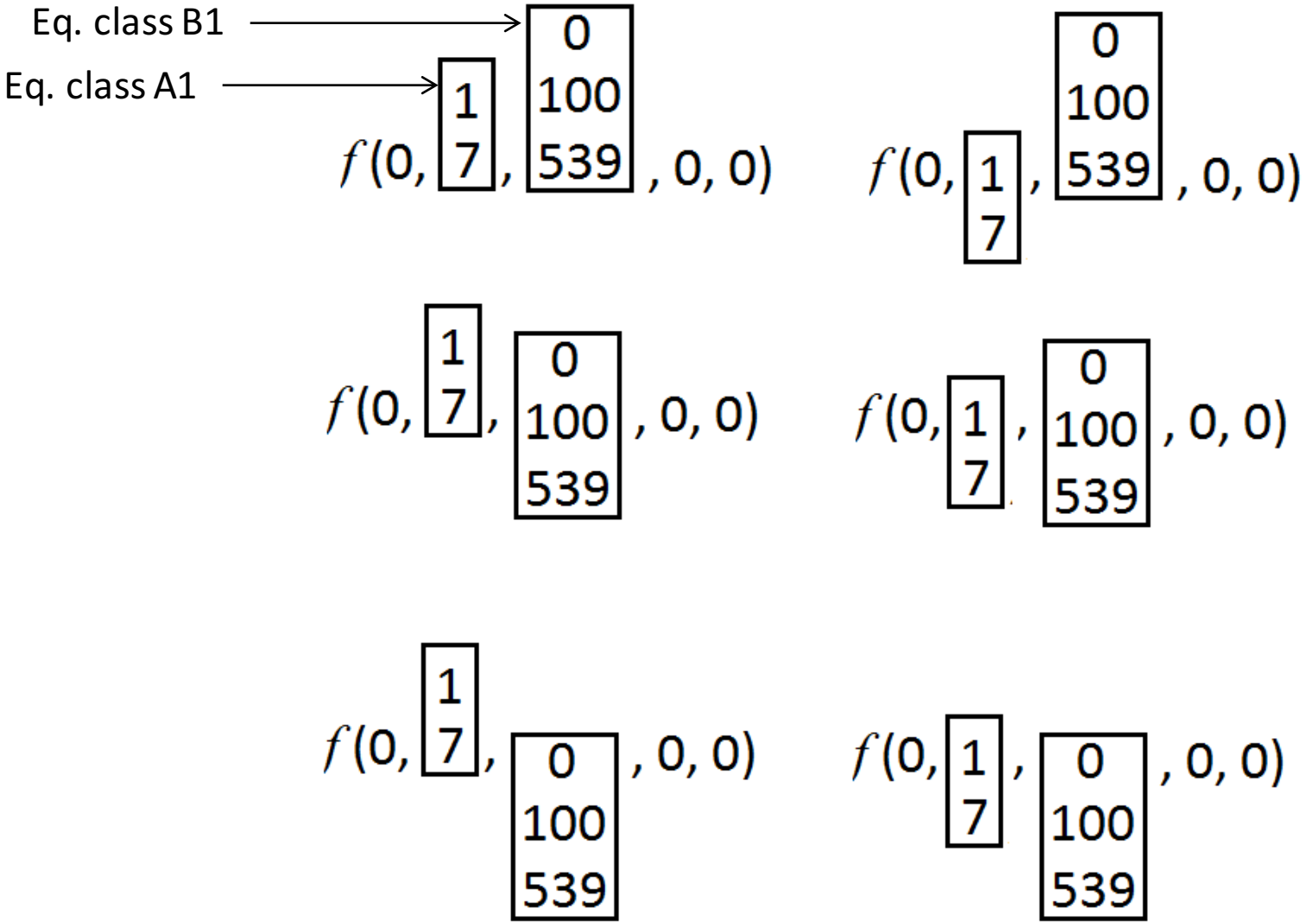
day (enum) : A1,A2

time (enum): B1,B2,B3

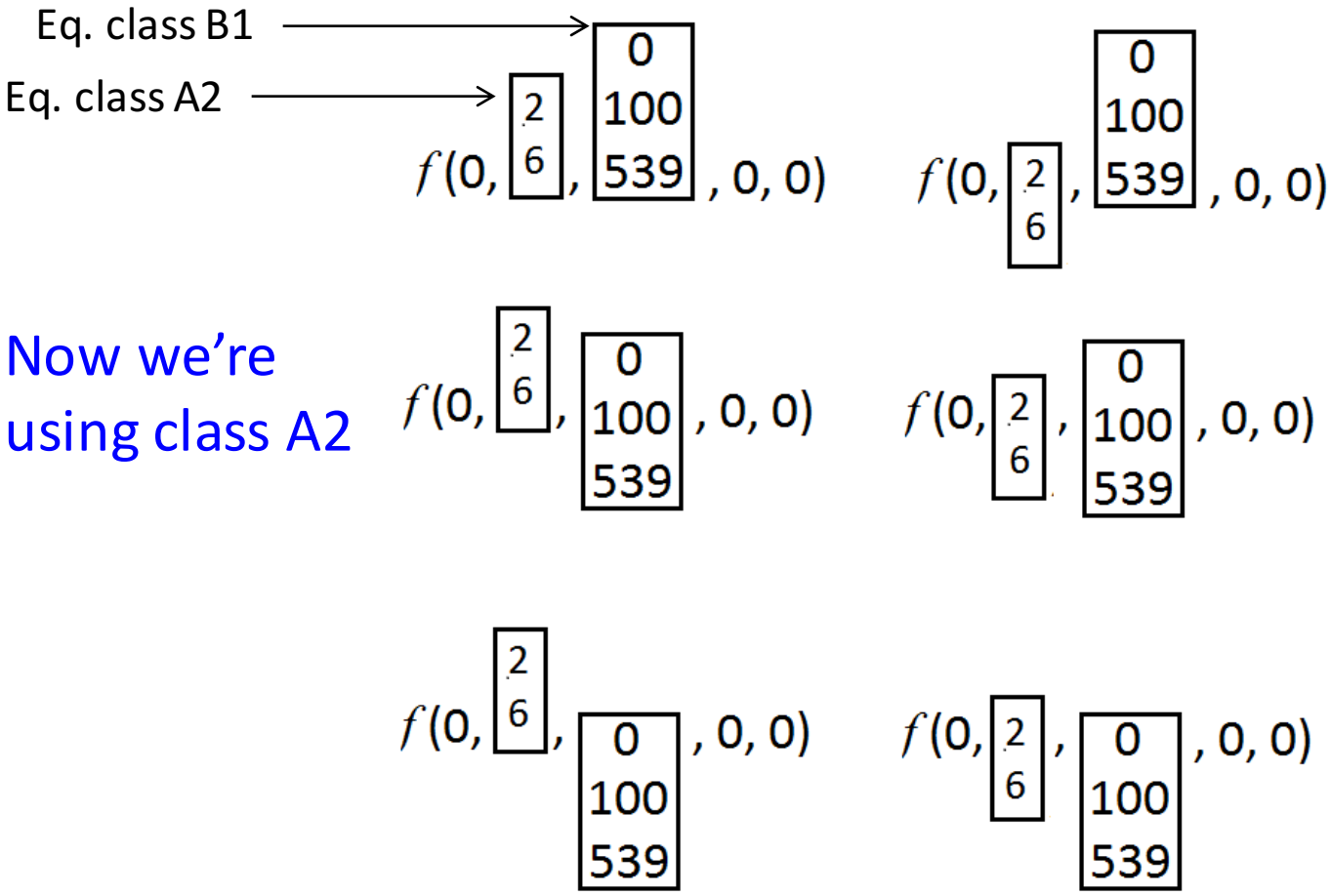
priv (bool): 0,1

aud (bool) : 0,1

All of these should be equal



These should also be equal



Covering array

Primary array:

0,A2,B1,1,1

1,A1,B1,0,0

0,A1,B2,1,0

1,A2,B2,0,1

0,A1,B3,0,1

1,A2,B3,1,0

One secondary array for each row

emp: boolean

day: (1,7), (2,6)

A1 A2

time: (0,539),(540,1020),(1021, 1439)

B1 B2 B3

priv: boolean

aud: boolean

Class A2 = (2,6)

Class B1 = (0,539)

0 2 0 1 1

0 6 0 1 1

0 2 539 1 1

0 6 539 1 1

Run the tests

- Correct code output:

  3333

  0000

  0000

  1111

  0000

  2222

Faulty code:

```
if (emp && t>=START &    t==END
&& d>=MON && d<=FRI) return 1;
```

Faulty code output:

3333

0000

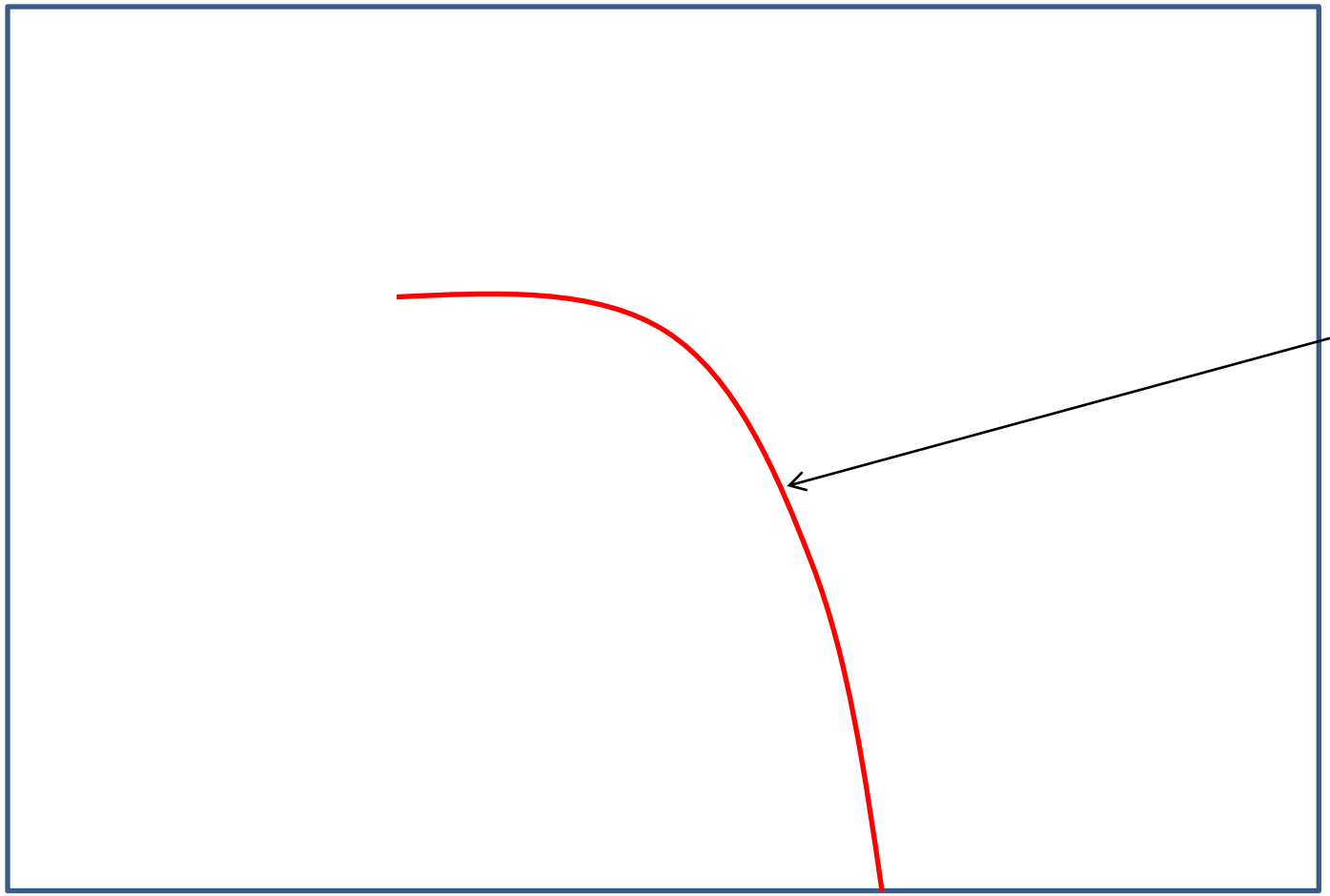
0000

3311

0000

2222

What's happening here?



Can this really work on practical code?

Experiment: TCAS code (same used in earlier model checking tests)

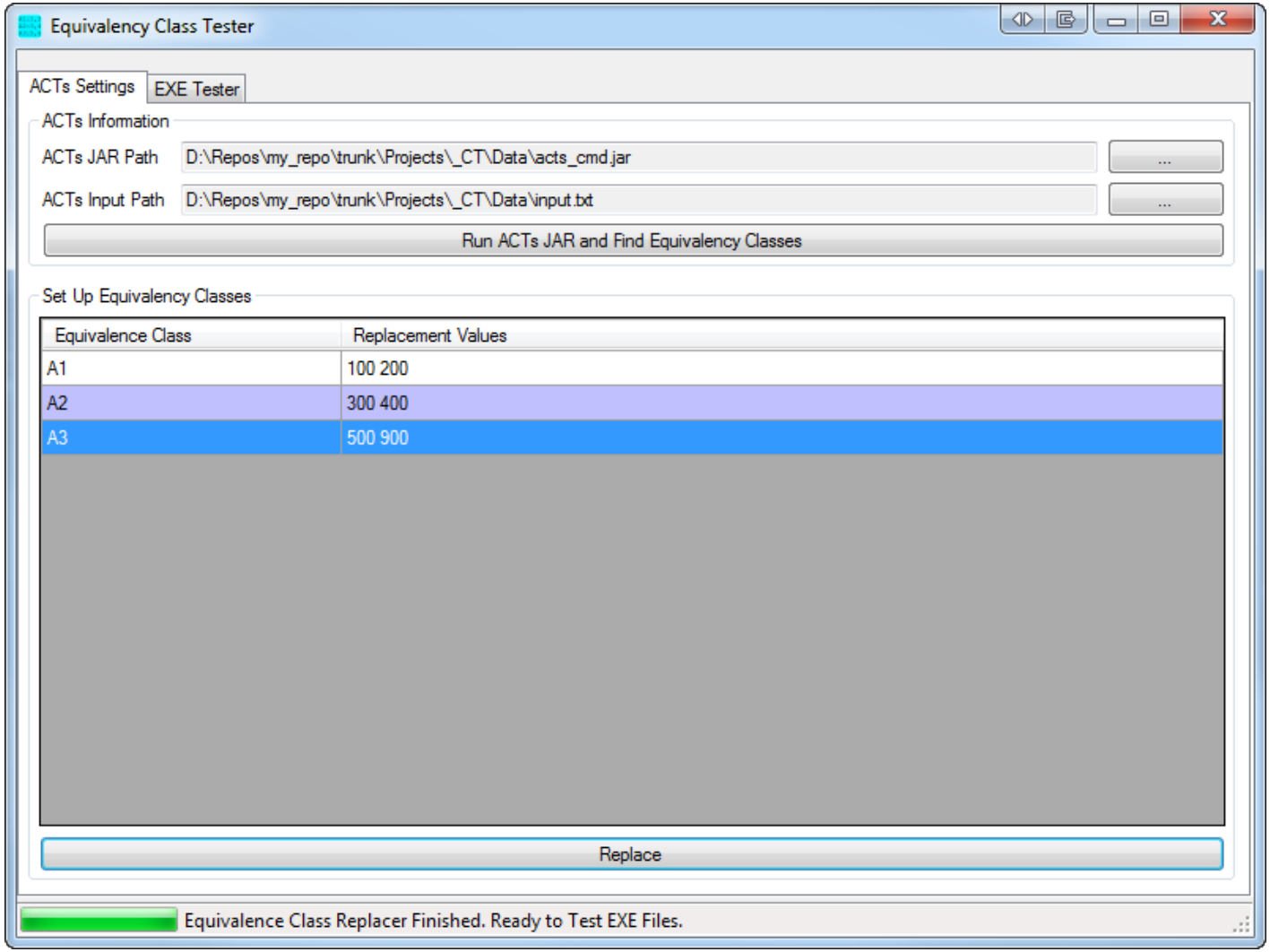
- Small C module, 12 variables
- Seeded faults in 41 variants

- Results:

Primary x secondary	#tests	total	faults detected
3-way x 3-way	285x8	2280	6
4-way x 3-way	970x8	7760	22

- More than half of faults detected
- Large number of tests -> but fully automated, no human intervention
- We envision this type of checking as part of the build process; can be used in parallel with static analysis, type checking

Prototype tool has been developed



Next Steps

- Realistic trial use
- Different constructions for secondary array, e.g., random values
- Formal analysis of applicability – range of applicability/effectiveness, limitations, special cases
- Determine how many faults can be detected this way
- Develop tools to incorporate into build process