Combinatorial Methods in Software Testing

Rick Kuhn, kuhn@nist.gov Raghu Kacker, raghu.kacker@nist.gov

STATUS QUO

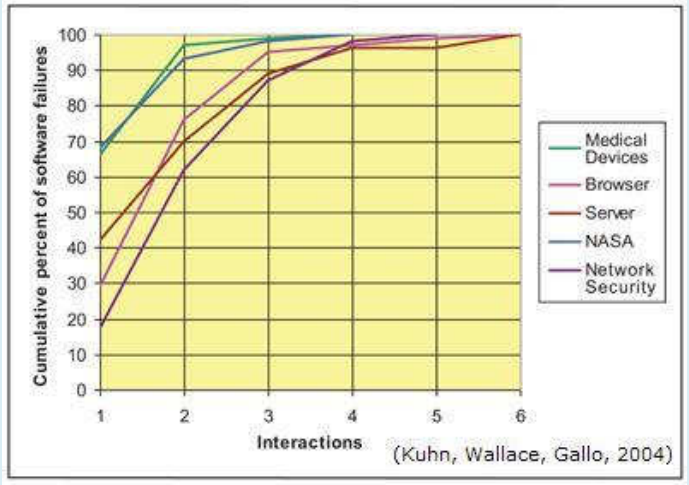
- High assurance software testing is much too expensive
- Testing cost is half or more of total development cost
- Proliferation of software in all devices – mobile apps, internet of things, cyber-physical systems – requires new approach to testing and assurance.

NEW INSIGHTS

- **Interaction Rule:** most software failures are triggered by one or two parameters, with progressively fewer by three, four, or more parameters, and maximum interaction degree is small.
- Validated empirically across a broad range of application domains; most complex interaction seen is 6-way.
- Suggests new approach to testing.

STRATEGY

Interaction Rule:



- Use combinatorial methods including covering arrays of all *t*-way combinations for *t* = 2 .. 6, as appropriate for problem domain.
- Use formal model to provide test oracle.
- Produce fully automated high-strength test suite.

QUANTITATIVE IMPACT

t-way covering array

Formal model of software under test

- Test suite with high fault detection capability, including complete tests (inputs and expected results).
- Reduced test time
- Higher fault detection

GOALS

- Easy-to-use tools for:
  - t-way test data generator
  - combinatorial coverage measurement
  - sequence covering
  - integration with model checking for automated test generation
  - domain specific tools
- Textbook on combinatorial testing
- Industrial usage reports

Better software testing at lower cost through combinatorial methods.