TEST ORACLES

Formal Definitions and Classifications

Shin Yoo & Mark Harman(UCL)
Muzammil Shahbaz & Phil McMinn(University of Sheffield)

OVERVIEW

- Formal Definitions
- Oracle Literature Timeline & Classification

FORMAL DEFINITIONS

- Few attempts to form an universal framework
- Recent work on formalising testing process (Staats et al. 2011) but with different focus

Definition 1 (Alphabet) The input to the program under test will be considered to be drawn from a set I, while the output will be drawn from a set O.

Definition 2 (Test Case) A test case is a pair (i, o) in $I \times 2^O$ such that o is non-empty and is singleton in the case that the system, is deterministic for input i .

Definition 3 (Test Instance) A test instance is a element of the set $I \times O$

VERSION 0.1

- Define oracle as a set of test cases that establishes acceptable behavioural relationship between input and output

$$I \times O \to \{0, 1\}$$

VERSION 0.2

- We want to cater for probabilistic decision on acceptance

$$I \times O \to [0, 1]$$

INEXACTNESS

Exact 0.54% 7.58%

Algorithmic Methodologies for Ultra-efficient Inexact Architectures for Sustaining Technology Scaling
Lingamneni et al., ACM Computing Frontiers 2012

VERSION 0.3

- We want to cater for metamorphic relations
- Acceptable behaviour is defined as a relation to other test instances

$$I \times O \times 2^{I \times O} \rightarrow [0, 1]$$

METAMORPHIC RELATIONSHIP

- If certain relation holds between two inputs, you expect a specific relation to hold between corresponding outputs
  - Example: $x' = \pi - x \rightarrow \sin x' = \sin x$
- Traditional examples focus on two instances, but it can be generalised to n instances
  - Example: linearity between input/output requires 3 instances

VERSION 0.3

- We want to cater for metamorphic relations
- Acceptable behaviour is defined as a relation to other test instances

$$I \times O \times 2^{I \times O} \to [0, 1]$$

VERSION 0.4

- We want to cater for inferred specification/regression suites
- Acceptable behaviour is defined as a relation to other test instances

$$I \times O \times 2^{I \times O} \to [0, 1]$$

Definition 4 (Definite Oracle) A *Definite Oracle* is a function from test instances to $\{0, 1\}$. That is, a definite oracle is an element of the set $I \times O \times 2^O \to \{0, 1\}$.

Definition 5 (Probabilistic Oracle) A *Probabilistic Oracle* is a function from test instances to $[0, 1]$. That is, a probabilistic oracle is an element of the set $I \times O \times 2^{I \times O} \to [0, 1]$.

Definition 6 (Completeness) An *Oracle* is compete if it is a total function.

Definition 7 (Ground Truth) The Ground Truth, $\mathcal{G}$ is a definite oracle.

We can now define soundness of an oracle with respect to the Ground Truth, $\mathcal{G}$.

Definition 8 (Soundness) A *Probabilistic Oracle*, PO is sound iff

$$PO(i,o) \in \begin{cases} [0, 0.5) & \text{when } \mathcal{G}(i,o) = 0 \\ (0.5, 1] & \text{when } \mathcal{G}(i,o) = 1 \end{cases}$$

Definition 9 (Correctness) An oracle is partially correct iff it is sound. An oracle is totally correct iff it is sound and complete.

Classification of Oracles in the literature review

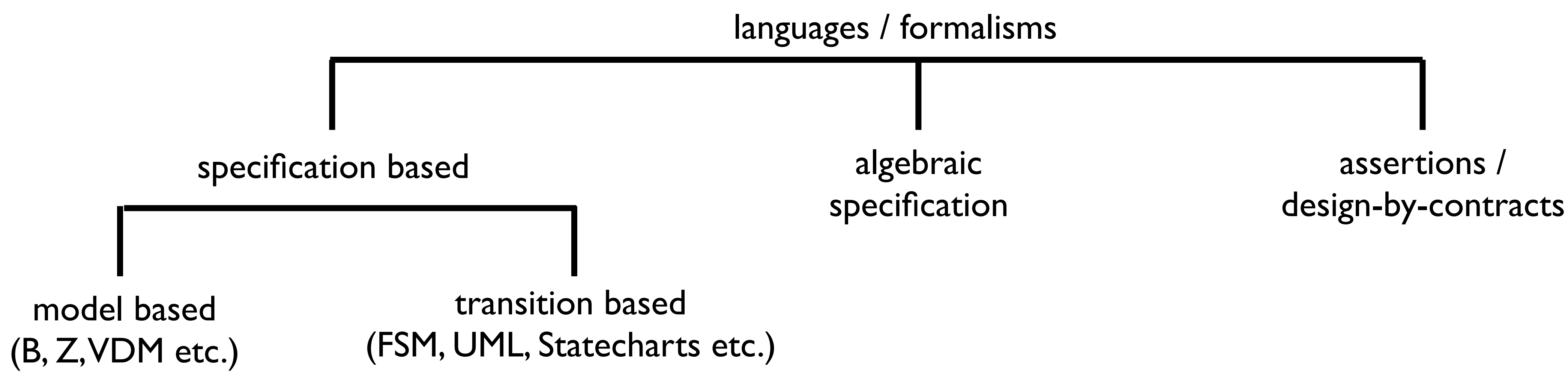
- **Origin:**
  - "test oracle" coined by W. E. Howden (1978)
  - *"a program specification, table of examples, or the programmer's knowledge on how the program should operate"*

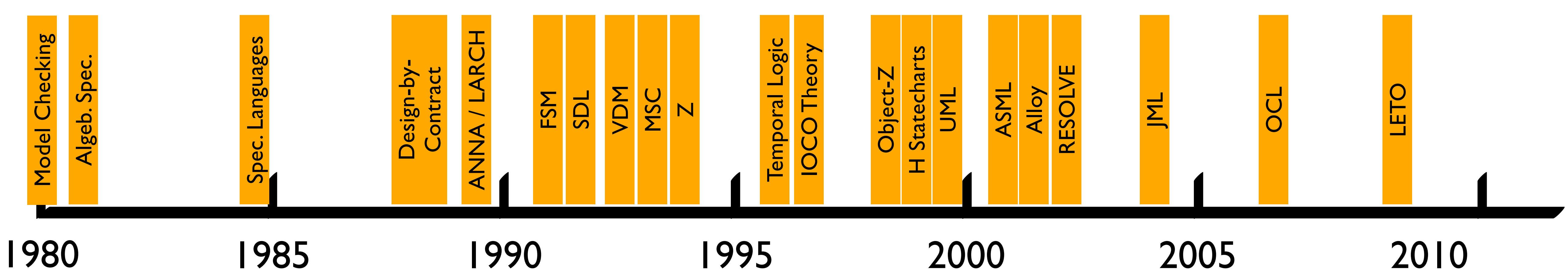
    (Howden and Eichhorst, Tutorial: Software Testing and Validation Techniques, 1978)

- **Classification:**
  - Specified Oracles
  - Derived Oracles
  - Implicit Oracles
  - No Oracles

Specified Oracles

oracles that are formally specified





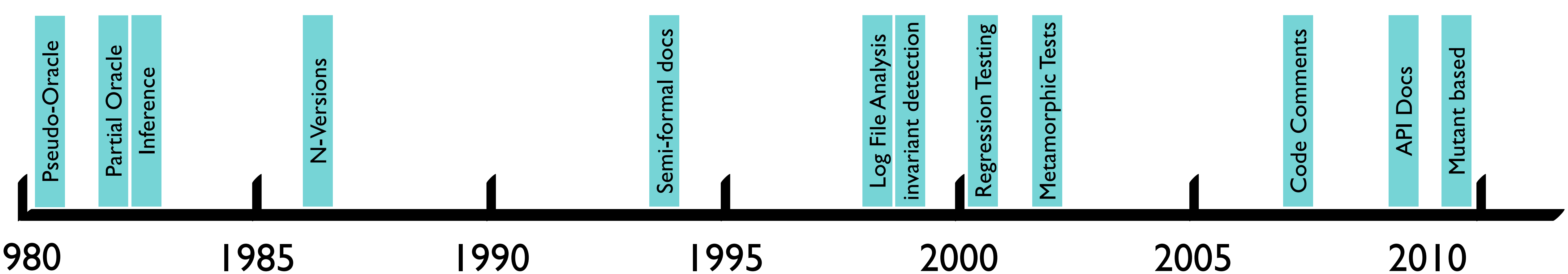
Derived Oracles

oracles that can be derived from the given artefacts

system executions
(traces, log files, invariants, ...)

documentations
(source code, comments, APIs, specs, ...)

existing knowledge
(partial / pseudo oracles, regression, ...)



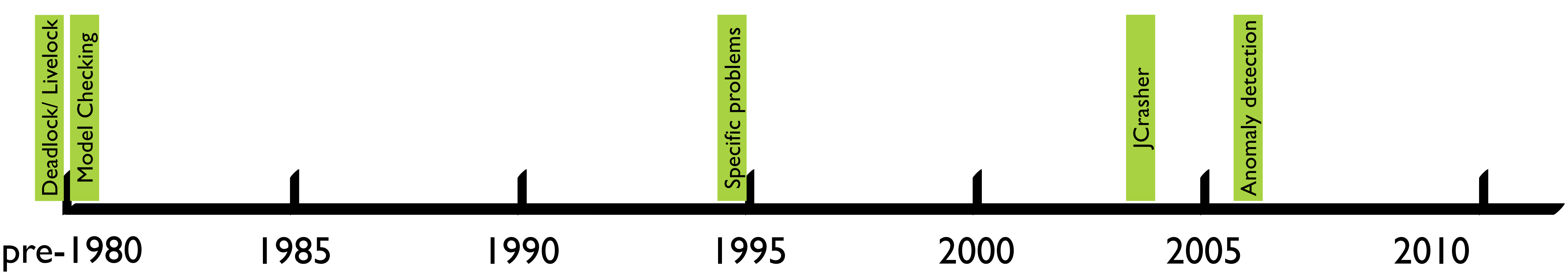
Implicit Oracles

oracles which do not require specification

anomalies
(deadlock, livelock)

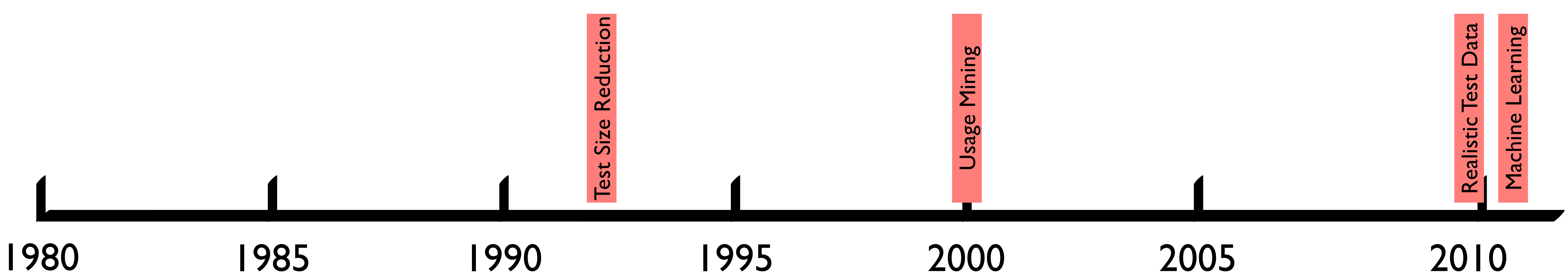
errors
(divide-by-zero, memory leaks, ...)

exceptions
(crash, ...)



No Oracles

no way of automatic validation!!!



http://recost.group.shef.ac.uk

The University Of Sheffield.

The RECOST Project

UCL

- Home
- Publications
- Partners
- People

Testing involves examining the behaviour of a system in order to discover potential faults. The problem of determining the desired correct behaviour for a given input is called the **Oracle Problem**. Since manual testing is expensive and time consuming there has been a great deal of work on automation and part automation of Software Testing. Unfortunately, it is often impossible to fully automate the process of determining whether the system behaves correctly. This must be performed by a human, and the cost of the effort expended is referred to as the Human Oracle Cost.

RE-COST will develop Search-Based Optimisation techniques to attack the Human Oracle Cost problem quantitatively and qualitatively. The quantitative approach will develop methods and algorithms to both reduce the number of test cases and the evaluation effort per test case. The qualitative approach will develop methods and algorithms that will reduce test case cognition time.

The RE-COST project seeks to transform the way that researchers and practitioners think about the problem of Software Test Data Generation. This has the potential to provide a breakthrough in Software Testing, dramatically increasing real world industrial uptake of automated techniques for Software Test Data Generation.

EPSRC
Engineering and Physical Sciences Research Council

berner & mattner
optimizing your development

MOTOROLA

SOGETI