Comparing Test Sets and Criteria in the Presence of Hypotheses

Rob Hierons, Brunel University
Structure of this talk

- Motivation and background
- Some previous comparators
- A new comparator
- Comparing test sets
- Incremental test generation
- Comparing criteria
- Future work
- Conclusions
Motivation

• When testing we would like to answer questions such as:
  – What is the best technique/criterion to use?
  – How might I extend my test to make it more effective?
  – Is it worth adding the following test case?
• In each case we would like to make some comparisons regarding test effectiveness.
Notation

- S will denote the set of specifications.
- P will denote the set of programs.
- X will denote the input domain of the programs being considered.
- Y will denote the output domain of the programs being considered.
Test hypotheses and fault models

• A test hypothesis is some property the tester believes the implementation has.

• A fault model is a set $F$ of behaviours where the tester believes that the implementation behaves like some unknown element of $F$. 
Testing in the presence of test hypotheses and fault models

- It may be possible to produce a test set that determines correctness under the assumption being used.
- Note: such tests need not be practical
The Uniformity Hypothesis for partition $\Pi$

- Here it is assumed that:
  - For every $\pi_i \in \Pi$, if any value in $\pi_i$ leads to a failure, all values in $\pi_i$ lead to failures.

- A test containing a value from each $\pi_i \in \Pi$ determines correctness under this assumption.
Testing in context

- When testing a component $p$ within a context $C$ we might assume that $C$ is correct:
  - a failure may only occur through a fault either in $p$ or in the interaction between $p$ and $C$. 
Fault models and testing from a finite state machine

- When testing from a finite state machine (FSM) $M$ it is normal to assume that the implementation behaves like some unknown FSM $M_1$ with the same input and output alphabets as $M$.
- Often we assume that $M_1$ is contained in some fault model.
Examples of fault models

- The following are commonly used:
  - There are only output faults.
  - $M_I$ has no more states than $M$.
  - $M_I$ has at most $m$ states (some predefined $m$).

- In each case, we can test to determine correctness under the assumption made.
Test criteria

- A test criterion $C$ is a function that takes a specification, a program, and a test set and returns a boolean that states whether the test set is ‘sufficient’.

- Given criterion $C$, specification $s$ and program $p$, $C(s,p)$ will denote a function that takes a test set $T$ and returns $C(s,p,T)$. 
Previous comparators
Comparing test criteria with $\leq$ 

- $C_2 \leq C_1$ if and only if, for every $s \in S, p \in P$: 
  - if there exists a test set $T_2$ such that $C_2(s,p,T_2)$ is true and $p$ fails on $T_2$, then for every test set $T_1$ such that $C_1(s,p,T_1)$ is true, $p$ fails $T_1$.

- This means: if we can determine that a program is faulty using $C_2$ we must do so using $C_1$. 
Comparing test sets with $\leq$

- We can extend $\leq$ to test sets:
  - $T_2 \leq T_1$ if and only if for all $s \in S$, $p \in P$, if $p$ fails on $T_2$ then $p$ fails on $T_1$.

- Where this is the case, we know that by using $T_1$ instead of $T_2$ we cannot lose anything in terms of our ability to show that a program is faulty.
Problems with $\leq$

- All effective (feasible) monotonic test criteria are incomparable under $\leq$.
- Assuming $P$ contains a representative of every computable function from $X$ and $Y$, $T_2 \leq T_1$ if and only if $T_2 \subseteq T_1$. 
The subsumes relation

- Criterion $C_1$ subsumes criterion $C_2$ if and only if:
  - For all $s \in S$, $p \in P$, $T \subseteq X$, $C_1(s,p,T) \Rightarrow C_2(s,p,T)$

- Many criteria are comparable under the subsumes relation but this need not tell us anything about fault detecting ability.

- A test generation technique for $C_2$ may be better than some test generation techniques for $C_1$. 
An observation

• When testing a program we only need to consider how good our test set or criterion is for that program.

• Thus: we might make our comparisons specific to the program (and specification) being considered:
  – we may utilise known system properties.
Comparing test sets under $\leq_H$

- Given test hypothesis H, test set $T_1$ is at least as strong as test set $T_2$ under H if and only if:
  - whenever $p$ satisfies H and $p$ fails $T_2$ then $p$ fails $T_1$.
- This is denoted $T_2 \leq_H T_1$. 
Extreme test hypotheses

• The following will be useful when demonstrating properties of $\leq_H$.
  – $H_{\text{min}}$ will denote the minimal hypothesis
  – $H_{\text{corr}}$ will denote the hypothesis that the program is correct.
A lower bound on $\leq_H$

- $T_2 \subseteq T_1 \Rightarrow T_2 \leq_H T_1$.
- These may be equivalent – simply let $H = H_{\text{min}}$.
- In effect, under $H_{\text{min}}$, $\leq_H$ is equivalent to $\leq$. 
An upper bound on $\leq_H$

- It is possible that all test sets are comparable under $\leq_H$.
- This happens when $H = H_{corr}$.
- Note: from this we can see that a non-empty test set may be no more effective than the empty test set.
Further relationships

- We will say that $T_1$ and $T_2$ are equivalent under $H$ if and only if $T_1 \leq_H T_2$ and $T_2 \leq_H T_1$.
- This is denoted $T_1 \equiv_H T_2$.
- $T_2 < T_1$ if $T_2 \leq_H T_1$ and not $T_1 \leq_H T_2$. 
Example

- Consider the application of the uniformity hypothesis \( H_\Pi \) with partition \( \Pi = \{ \pi_1, \ldots, \pi_n \} \).
- Then a test set \( T \) determines correctness under \( H_\Pi \) if and only if \( T \) contains one or more elements from each \( \pi_i \).
Comparisons under $H_{\Pi}$

- The following are clear (and as expected):
- $T_2 \leq_{H_{\Pi}} T_1$ if and only if:
  - $\{ \pi_i \in \Pi | \pi_i \cap T_2 \neq \{\} \} \subseteq \{ \pi_i \in \Pi | \pi_i \cap T_1 \neq \{\} \}$
- $T \leq_{H_{\Pi}} T \cup \{t\}$ if and only if:
  - $t \in \pi_k$ for some $\pi_k \in \{ \pi_i \in \Pi | \pi_i \cap T = \{\} \}$
Comparisons: testing in context

• Suppose we are testing a system composed of p and context C with hypothesis H that states: C is correct.

• Let \( X_p \) denote the set of elements of X that lead to p receiving input.

• Then \( T_2 \leq_H T_1 \) if and only if:
  - \( T_2 \cap X_p \subseteq T_1 \cap X_p \)
Further results

- $T_2 <_H T_1$ if and only if
  - $T_2 \cap X_p \subseteq T_1 \cap X_p$
- $T_2 \equiv_H T_1$ if and only if
  - $T_2 \cap X_p = T_1 \cap X_p$
Observation

- Given test set $T$ and test $t \in X \setminus T$, it is possible that:
  - $T \cup \{t\} \equiv_H T$.
- Thus: extending a test set might not make it more effective.
Testing from FSM M: output faults

- Suppose hypothesis H states:
  - only output faults can occur

- \( T_2 \leq_H T_1 \) if and only if:
  - When \( T_1 \) and \( T_2 \) are executed on \( M \), \( T_1 \) covers every transition covered by \( T_2 \).
Incremental Test Development
Observations

• If $T=\{t_1, \ldots, t_n\}$ is a minimal (non-redundant) test set then for all $1 \leq i < n$:
  - $\{t_1, \ldots, t_i\} \leq_H \{t_1, \ldots, t_{i+1}\}$

• If $T$ does not determine correctness under $H$ then there is some test case $t$ such that:
  - $T <_H T \cup \{t\}$
Incremental test development and $\lesssim_H$

- Under $H$ it is only worth extending test set $T$ by test case $t$ if:
  - $T \lesssim_H T \cup \{t\}$
- We might start with the empty set and at each step add tests that strengthen the test set.
- Note: we might still have redundant test cases.
Refining hypotheses

- H' is a refinement of H if H' $\Rightarrow$ H.
- Observe that if H' $\Rightarrow$ H then:
  \[ T_2 \leq H T_1 \Rightarrow T_2 \leq H, T_1 < H. \]
- This suggests we might refine test hypotheses and test sets together (though
  \[ \neg(T_2 < H T_1 \Rightarrow T_2 < H, T_1) \] so this might reduce the test efficiency).
Refinement and the uniformity hypothesis

- One instance $H_{\Pi_1}$ of the uniformity hypothesis is a refinement of another instance $H_{\Pi_2}$ if and only if:
  - Each subdomain of $\Pi_2$ is the union of a set of subdomains from $\Pi_1$. 
Comparing test criteria using $\leq_H$

- We can extend $\leq_H$ to test criteria by, $C_2 \leq_H C_1$ if and only if:
  - For every $p \in P$ that satisfies $H$, if there is some non-redundant test set $T_2$ that satisfies $C_2(s,p)$ such that $p$ fails $T_2$, then for every non-redundant test set $T_1$ that satisfies $C_1(s,p)$, $p$ fails $T_1$.
- Note the use of 'non-redundant' (without it we get the same problems as $\leq$).
Observations

- All test criteria are equivalent under $H_{corr}$. 
Comparing the strength of test criteria

- Given test criteria $C_1$ and $C_2$ it might be interesting to know the answer to:
  - What is the weakest hypothesis $H$ under which $C_2 \leq_H C_1$?
Future work

- Consider probabilistic comparators.
- Investigate alternative hypotheses and criteria.
- Investigate weakest hypotheses that allow criteria to be compared.
Conclusions

• It is useful to be able to compare test criteria and test sets.
• By including a test hypothesis, we can utilise properties of the problem.
• Comparisons between test sets might drive incremental test development.