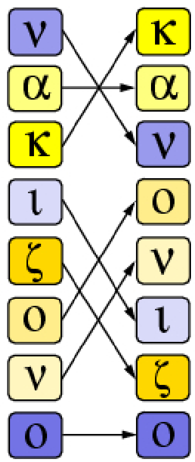
Allegheny College
Department of Computer Science

Towards the Prioritization of Regression Test Suites with Data Flow Information



Kanonizo

Matthew J. Rummel
Gregory M. Kapfhammer
Andrew Thall

Symposium on Applied Computing
Santa Fe, New Mexico March 13-17 2005

Allegheny College
Department of Computer Science

Definitions

- **Test Case** – An individual unit test
- **Test Suite** – A tuple of test cases
- **Regression Testing** – Testing that occurs after the completion of development or maintenance activities when a test suite comprised of all accumulated unit tests is executed
- **Test Prioritization** – The process of arranging test cases in a given test suite to facilitate the detection of defects earlier in the execution of the test suite

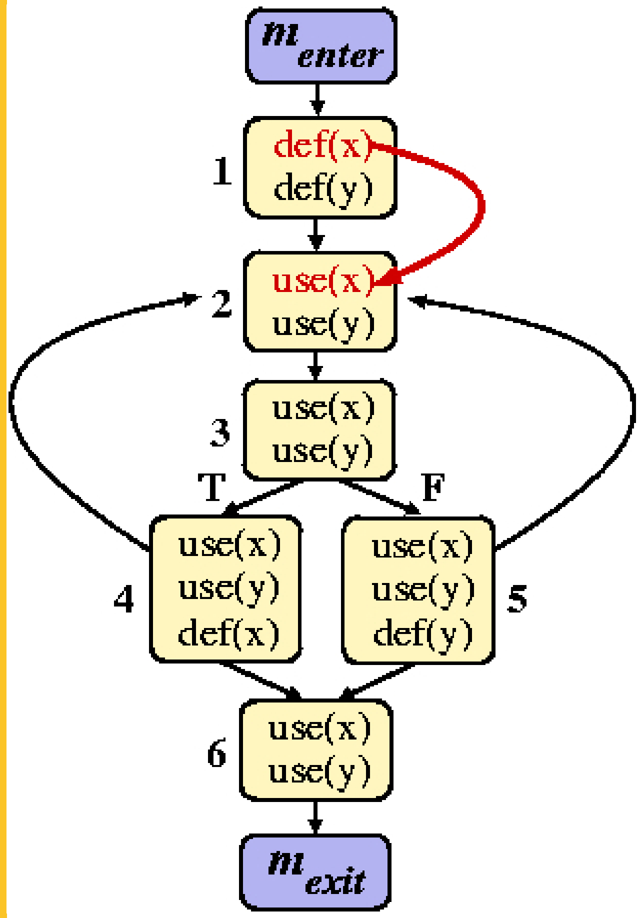
ALLEGHENY COLLEGE
Department of Computer Science

Motivation

- Regression testing may account for as much as one-half the cost of software maintenance
- Prioritization is often more feasible than test selection
- Tests that fulfill the *all-DUs* test adequacy criteria are more likely to reveal defects than those that satisfy control flow based criteria

ALLEGHENY COLLEGE
Department of Computer Science

Dataflow



- Model each method in a program as a control flow graph
- Control flow flow family of test criteria (ex: *all-nodes*, *all edges*, *all-paths*)
- Data flow criteria evolved from control flow (ex: *all-DUs*, *all-P-Uses*, *all-C-Uses*)
- Focus on intraprocedural def-use associations

ALLEGHENY COLLEGE
Department of Computer Science

Metrics

- *APFD* – The rate of fault detection per percentage of test suite execution

$$APFD(T,P) = 1 - \frac{\sum_{i=1}^{g} reveal(i,T)}{rg} + \frac{1}{2r}$$

- *PTR* – Percentage of a given test suite that must be executed for all faults to be detected

$$PTR(T,P) = \frac{r_g}{r}$$

Allegheny College
Department of Computer Science

Metrics Example

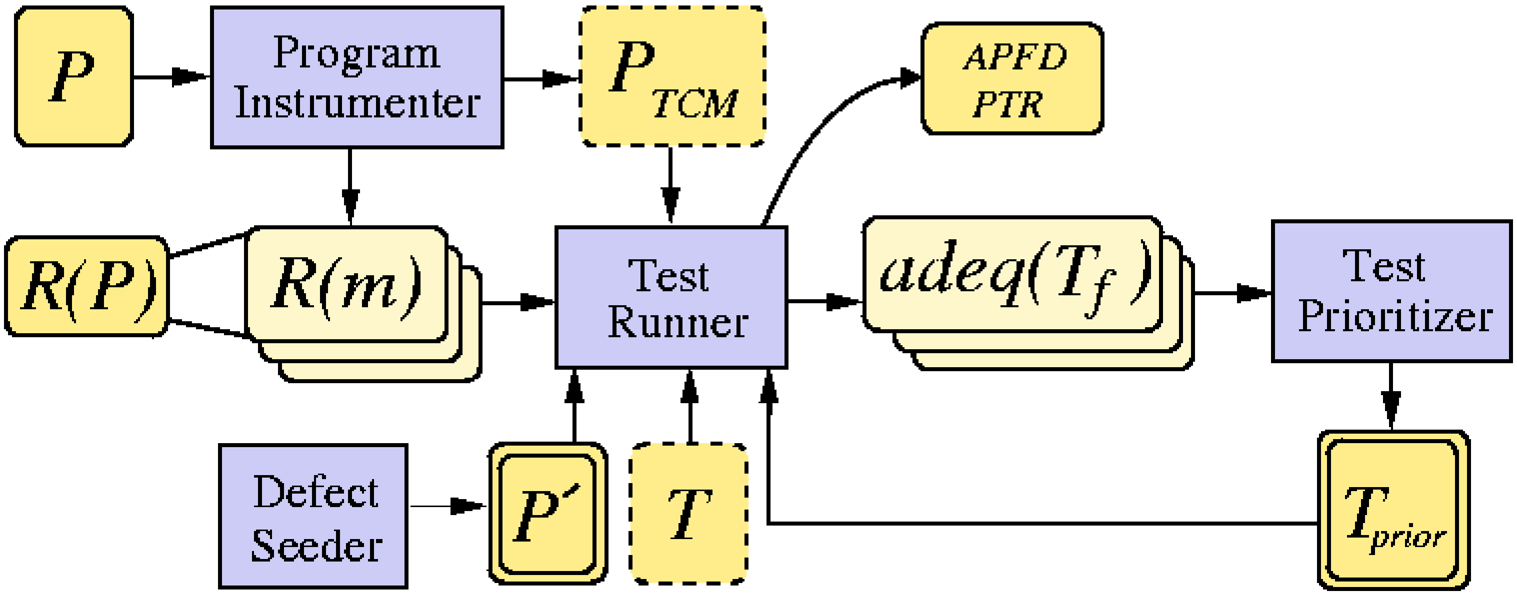
Test Case	Faults				
	f_1	f_2	f_3	f_4	f_5
T_1			×	×	
T_2	×	×			
T_3	×	×	×		
T_4			×	×	×
T_5		×	×		

$\sigma_1 = \langle T_1, T_2, T_3, T_4, T_5 \rangle$ $\sigma_2 = \langle T_3, T_4, T_1, T_2, T_5 \rangle$

- $APFD(T_1, P) = 1 - .4 + .1 = .7$
- $PTR(T_1, P) = \frac{4}{5}$
- $APFD(T_2, P) = 1 - .2 + .1 = .9$
- $PTR(T_2, P) = \frac{2}{5}$

ALLEGHENY COLLEGE
Department of Computer Science

Experiment Design



InstrumentandEnumerate

- Calculate the set of test requirements for program *P*
- Introduce test coverage monitoring instrumentation
- Execute test suites and report *APFD* and *PTR* calculations

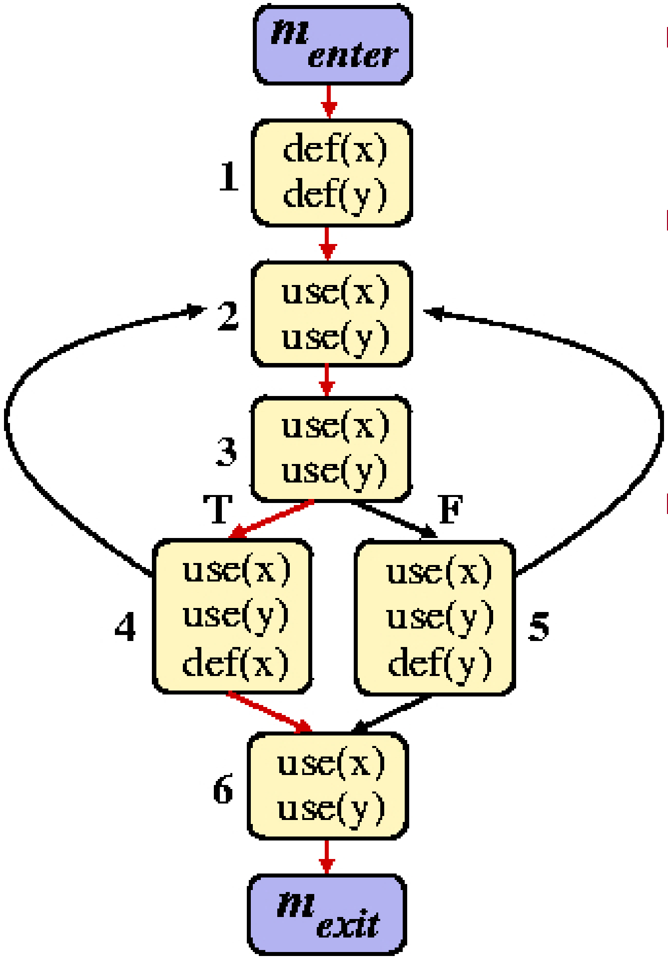
Allegheny College
Department of Computer Science

Cumulative Adequacy of a Test Case

- When a test case has covered both a *def* and corresponding *use* statement, the coverage of that association is stored
- **Test case adequacy** – The ratio between the number of covered test requirements and the total number of test requirements for all of the methods under test

$$adeq(T_f) = \frac{\sum_{k=1}^{h} |R_c(m_k)|}{\sum_{k=1}^{h} |R(m_k)|}$$

Cumulative Adequacy Example



- Model each method in a program as a control flow graph
- T_f enters method *m* and executes the **`true`** branch of node 3
- $$adeq(T_f) = \frac{7}{16} = 43.75\%$$

Allegheny College
Department of Computer Science

Experimentation Statistics

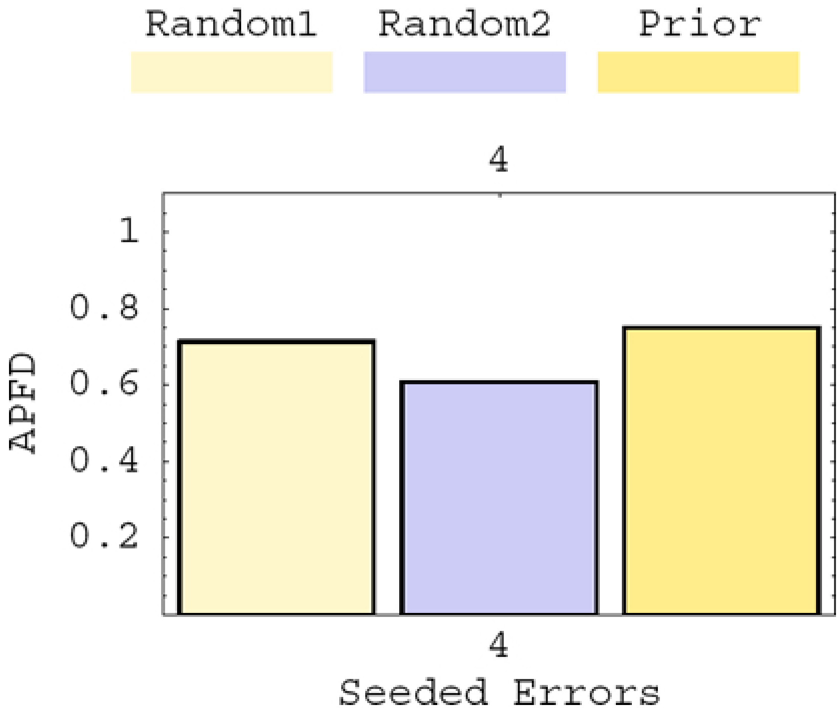
- Experiments conducted on a GNU/Linux workstation with dual 1GHz Pentium III Xeon processors, 512 MB of main memory

Case study applications:

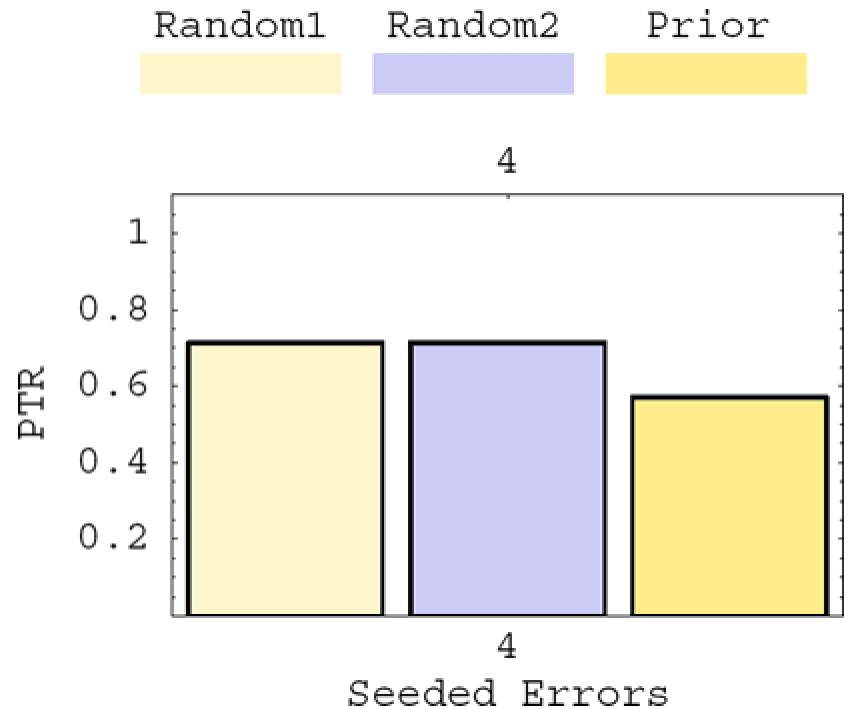
- **`Bank`** – 1 class, 53 def-use associations, 5 methods, 7 test cases, 4 seeded errors
- **`Identifier`** – 3 classes, 81 def-use associations, 13 methods, 11 test cases, 2 sets of 3 seeded errors
- **`Money`** – 3 classes, 302 def-use associations, 33 methods, 21 test cases, 3 sets of 3 seeded errors

ALLEGHENY COLLEGE
Department of Computer Science

Bank *APFD* and *PTR* Measurements



Bank *APFD*

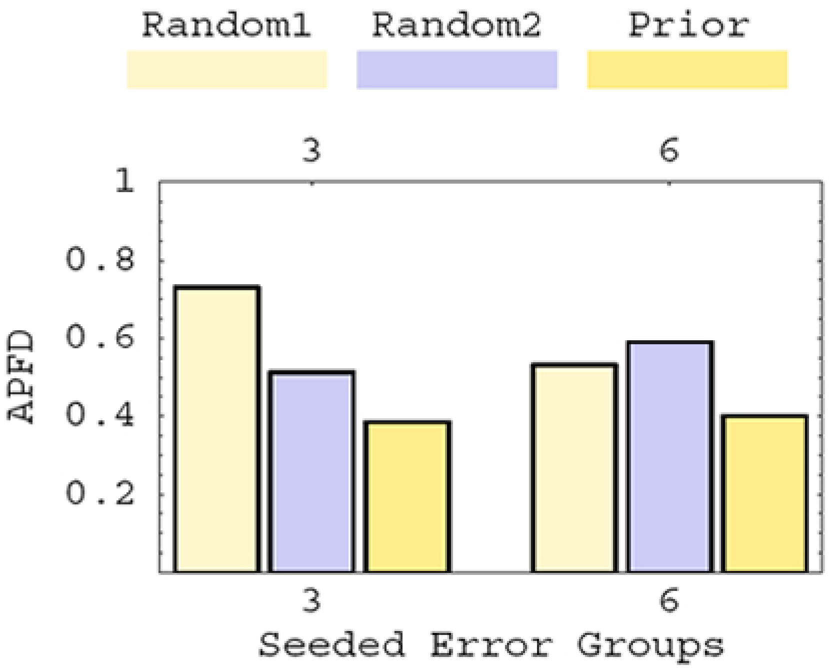


Bank *PTR*

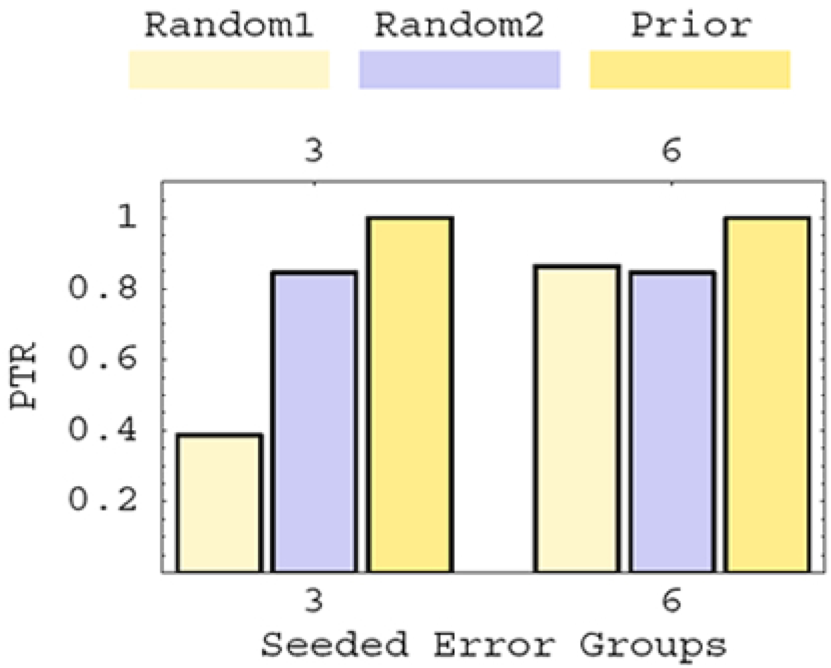
- Prioritized suite has best *PTR* value
- Prioritized suite has the best *APFD* value, slightly better than Random1

ALLEGHENY COLLEGE
Department of Computer Science

Identifier *APFD* and *PTR* Measurements



Identifier *APFD*

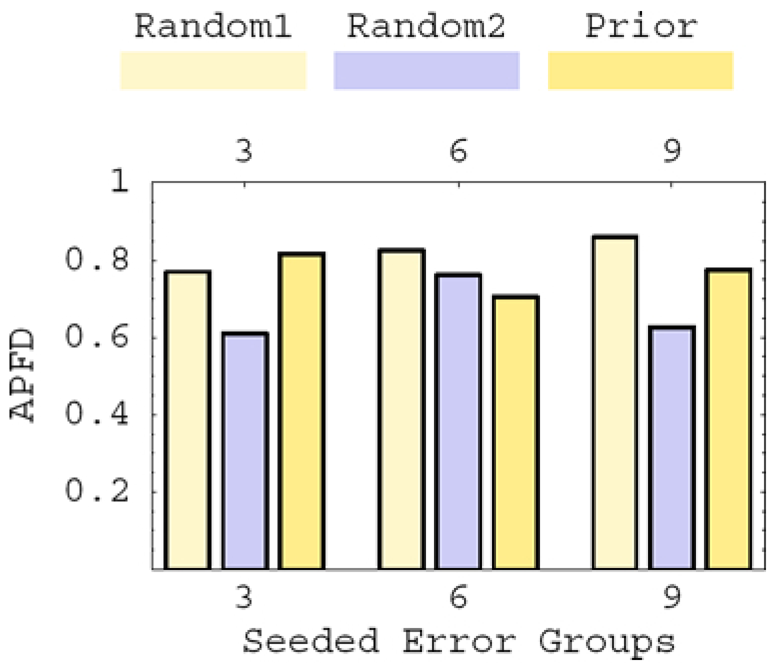


Identifier *PTR*

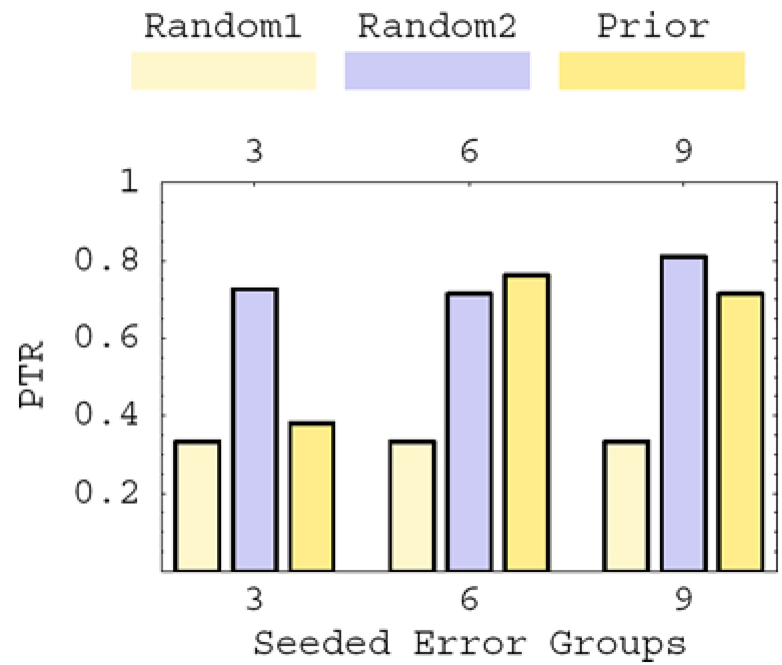
- Prioritized suite has the worst *PTR* value
- Prioritized suite has the worst *APFD* value

ALLEGHENY COLLEGE
Department of Computer Science

Money *APFD* and *PTR* Measurements



Money *APFD*



Money *PTR*

- Prioritized suite has best *APFD* for 3 errors, worst for 6 errors, medium for 9 errors
- Prioritized suite has medium *APFD* for 3 errors, slightly worse than Random1, worst for 6 errors, medium for 9 errors

Allegheny College
Department of Computer Science

Time and Memory Requirements

Program	Time (ms)	Space (bytes)
`Bank`	3,210	1,084,648
`Identifier`	3,351	2,170,871
`Money`	9,176	4,984,648

Time and Memory for *InsturmentandEnumerate* Algorithm

- Test case monitoring did not cause significant increases in the time required to execute test cases

ALLEGHENY COLLEGE
Department of Computer Science

Conclusions

- Test suites can be prioritized according to *all-DUs* with minimal time and space overhead
- Preliminary results indicate that data flow-based prioritizations are not always more effective than random prioritizations
- Successfully created a low-overhead framework for performing test prioritization which can be used in future studies

Future Work

- Incorporation of control flow-based and mutation-based adequacy into **`Kanonizo`**
- The comparison of our prioritization approach to other prioritization schemes beyond random
- The calculation of *APFD* and *PTR* for all permutations of an application's test suite
- Experimentation with additional case studies that have larger program segments and test suites
- The investigation of prioritization techniques for test suites that must be executed within a specified time constraint

ALLEGHENY COLLEGE
Department of Computer Science

Related Work

- Sebastian Elbaum et al. Prioritizing test cases for regression testing. *Proceedings of the International Symposium on Software Testing and Analysis*. ACM Press, August 2000.
- Phyllis G. Frankl et al. An applicable family of data flow testing criteria. *IEEE Transactions on Software Engineering*, October 1988.
- G. Rothermel et al. A framework for evaluating regression test selection techniques. Proceedings of the 16th International Conference on Software Engineering, *IEEE Computer Society Press*, May 1994.

ALLEGHENY COLLEGE
Department of Computer Science

Resources

- Kanonizo Research Group: http://cs.allegheny.edu/~gkapfham/research/kanonizo.
- Gregory M. Kapfhammer *The Computer Science Handbook* Chapter “Software Testing”. CRC Press, June 2004.
- Matthew Rummel, Greg Kapfhammer, and Andrew Thall Towards the Prioritzation of Regression Test Suites with Data Flow Information. *Proceedings of the ACM SIGAPP Symposium on Applied Computing*, Santa Fe, New Mexico, March 2005.