Similarity-Based Prioritization in Software Product-Line Testing

Mustafa Al-Hajjaji[(1)], Thomas Thüm[(1)], Jens Meinicke[(1)], Malte Lochau[(2)], Gunter Saake[(1)]

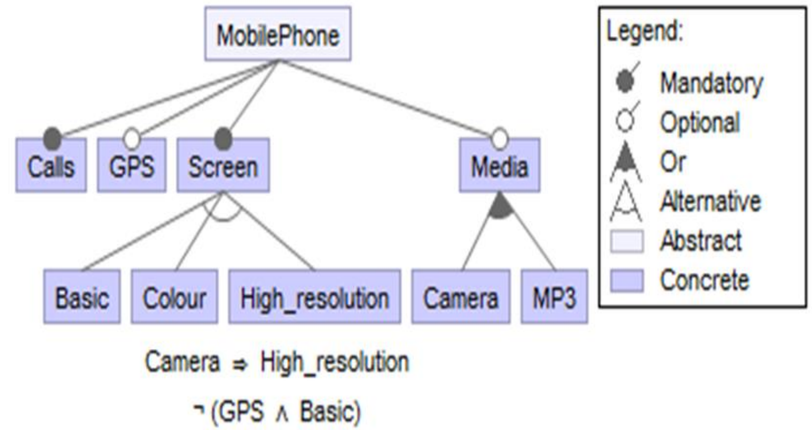
(1) Otto-von-Guericke-Universität Magdeburg
(2) TUD – Technische Universität Darmstadt

FOSD Meeting 2014, Schloss Dagstuhl

Motivation

- Testing a SPL is a difficult task
  - Explosion of possible products 2^n ; where n the feature number.

- Reduce the time to detect a defect?



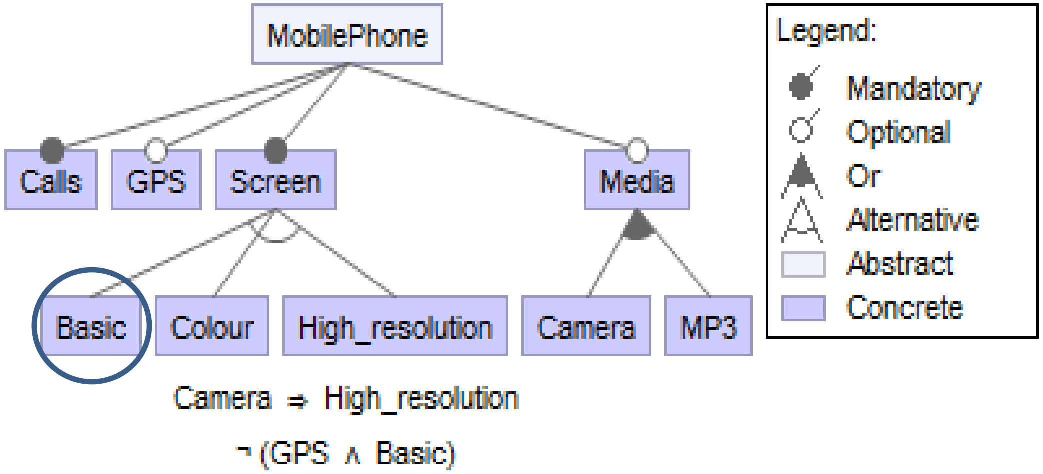
Feature model of *Mobile Phone* SPL

c1={MobilePhone,Calls,Screen,Colour,Media,MP3,GPS}
c2={MobilePhone,Calls,Screen_High resolution, Media, Camera}
c3={MobilePhone,Calls,Screen,High resolution}
c4={MobilePhone,Calls,Screen,Basic}
c5={MobilePhone,Calls,Screen,High resolution,GPS}
c6={MobilePhone,Calls,Screen,Basic,Media,MP3}
c7={MobilePhone,Calls,Screen,Colour}
c8={MobilePhone,Calls,Screen,High resolution, Media,MP3}
c9={MobilePhone,Calls,Screen,Colour,GPS}
c10={MobilePhone,Calls,Screen,Colour,Media,MP3}
c11={MobilePhone,Calls,Screen,High resolution, Media,MP3,GPS}
c12={MobilePhone,Calls,Screen,High resolution, Media,Camera,GPS}
c13={MobilePhone,Calls,Screen,High resolution, Media,MP3,Camera}

c11=fMobilePhone,Calls,Screen,High resolution,Media,MP3,GPS}
C1: {MobilePhone,Calls,Screen,Colour,Media,MP3,GPS}
C2: {MobilePhone,Calls,Screen,High resolution,Media,Camera}
c3={MobilePhone,Calls,Screen,High resolution}
c4={MobilePhone,Calls,Screen,Basic}
c5={MobilePhone,Calls,Screen,High resolution,GPS]
c6={MobilePhone,Calls,Screen,Basic,Media,MP3}
c7={MobilePhone,Calls,Screen,Colour}
c8={MobilePhone,Calls,Screen,High resolution, Media,MP3}
c9={MobilePhone,Calls,Screen,Colour,GPS}
c10={MobilePhone,Calls,Screen,Colour,Media,MP3}
c12={MobilePhone,Calls,Screen,High resolution, Media,Camera,GPS}
c13={MobilePhone,Calls,Screen,High resolution, Media,MP3,Camera}

Defect Features

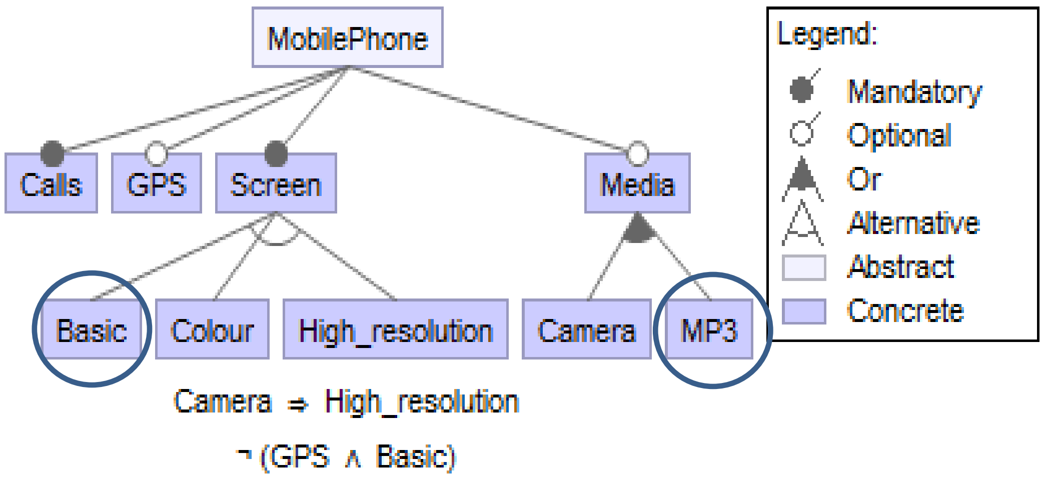
- Unit tests may find defect inside a single feature
  - n test suites required for a product line with n features.



Feature model of *Mobile Phone* SPL

Interaction defects

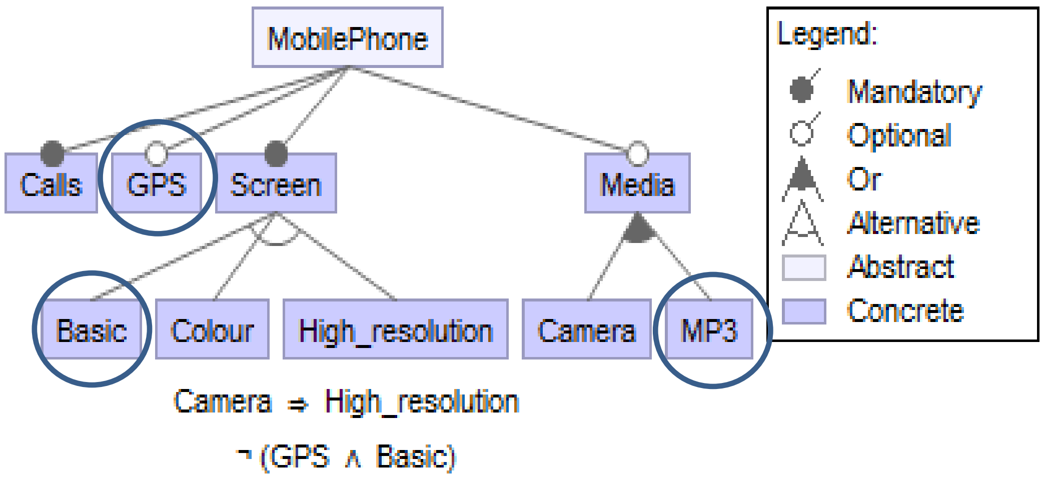
- 2-wise interaction defect
  - Reproducible by including 2 specific features



Feature model of *Mobile Phone* SPL

Interaction defect

- 3-wise interaction defect
  - Reproducible by including 3 specific features



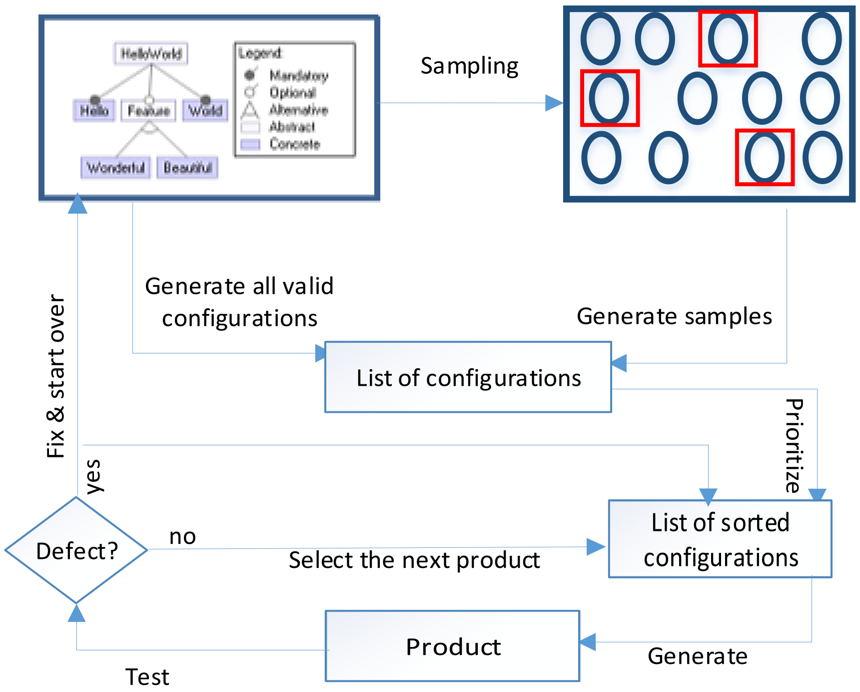
Feature model of *Mobile Phone* SPL

Motivation

- Kuhn et al. (2004)
  - Pairwise interaction → 70% of defects,
  - 3-wise interaction → 95% of defects,
  - 6-wise interaction → almost all the defects.

- Sampling algorithms
  - CASA (Garvin et al. 2011),
  - Chvatal (Chvatal 1979),
  - And ICPL (Johansen et al. 2012)

Similarity-based Prioritization



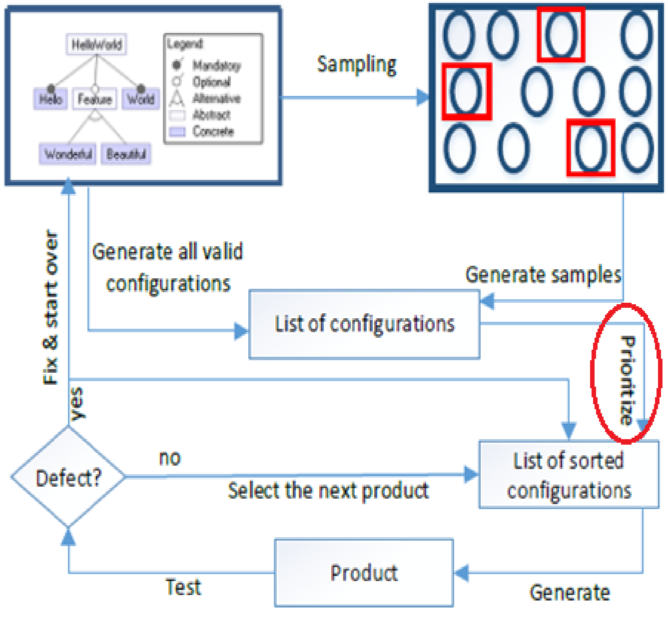
Similarity-based Prioritization

- Dissimilar test cases are likely to detect more defects than the similar ones!! (Hemmati et al. 2010)

- Hamming distance

$$d(c_i, c_j, F) = 1 - \frac{|c_i \cap c_j| + |(F \setminus c_i) \cap (F \setminus c_j)|}{|F|} \qquad (1)$$

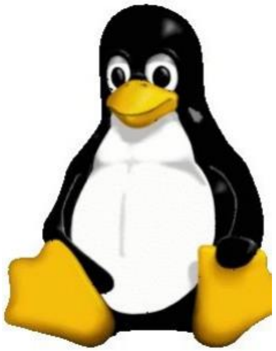
where c_i and c_j are configurations and F is the set of all features in a SPL.



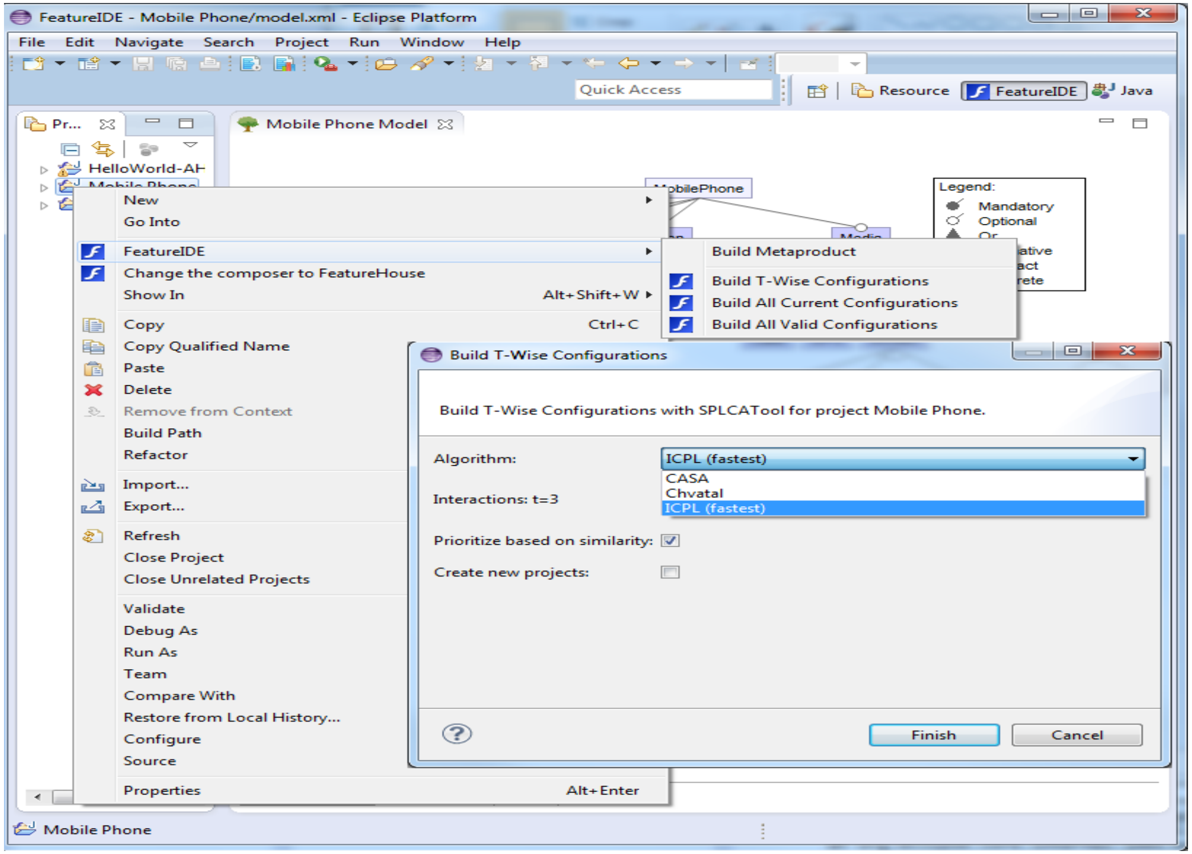
Similarity-Based Prioritization Approach

Similarity-based Prioritization

- The configuration with the maximum number of selected features
  - Covers most defects in individual features
  - Selection of the next configuration with large distance
  - Common in the Linux community (a.k.a. allyesconfig) (Dietrich et al., 2012)



FeatureIDE



Configuration Creation in FeatureIDE

Evaluation

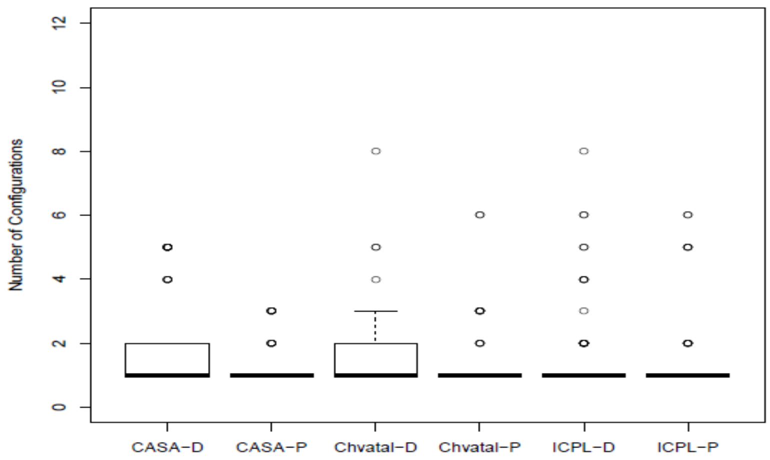
- Two case studies:
  - Mobile Phone SPL (10 features)
  - Smart Home SPL (60 features)

- We simulate defects
  - Caused by single features.
  - Occurring because of pairwise interactions

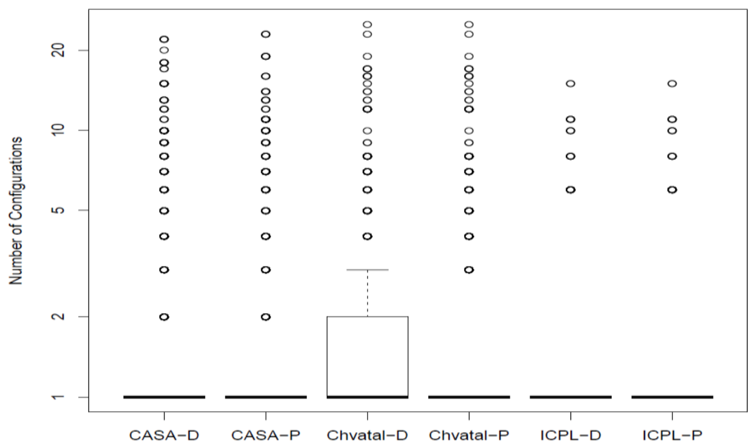
Potential defects

- We simulate five kinds of potential defects
  - $C_i=\{c|c \in SPL \wedge f1 \in C\}$
    - E.g., division by zero
  - $C_i=\{c|c \in SPL \wedge f1 \notin C\}$
    - E.g., f1 initializes a variable, F1 is removed.
  - $C_i=\{c|c \in SPL \wedge f1, f2 \in C\}$
    - E.g., one feature calls a method in another feature and the retrieved value is wrong
  - $C_i=\{c|c \in SPL \wedge f1 \in C \wedge f2 \notin C\}$
    - E.g., one feature calls a method from a feature that is not selected
  - $C_i=\{c|c \in SPL \wedge f1, f2 \notin C\}$
    - E.g., f1 and f2 initializes a variable, f1 and f2 are removed

Evaluation



Mobile Phone SPL



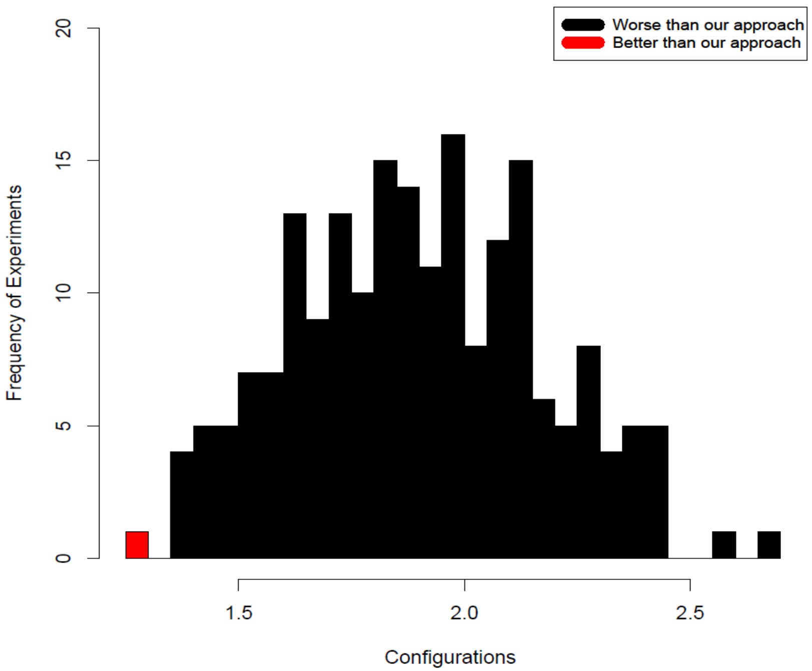
Smart Home SPL

The number of configurations to detect all defects; D- default order of each algorithm, P- similarity-based prioritization approach.

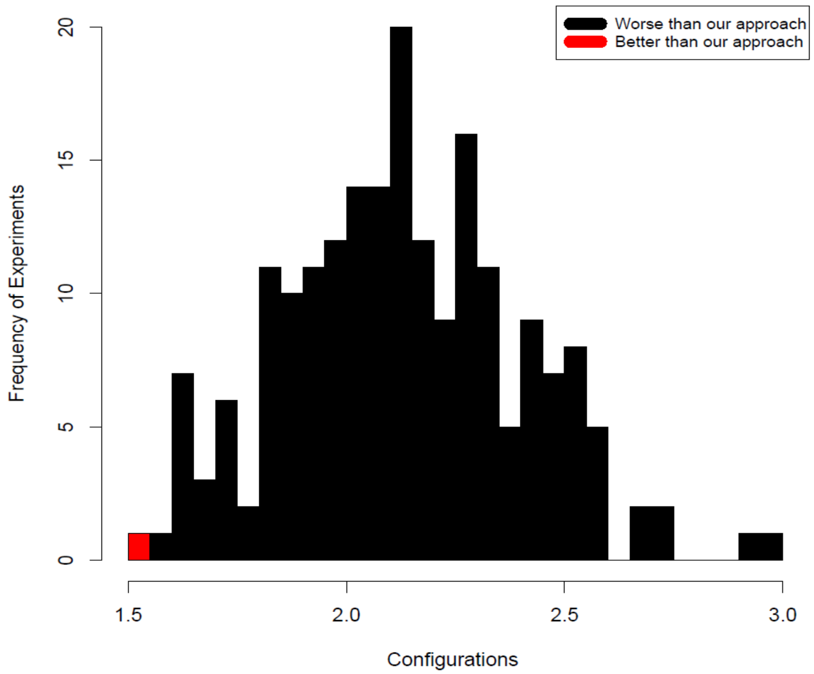
SPL	Sampling algorithm	Default order	Similarity-based prioritization	Improvement
Mobile Phone SPL	ICPL	1.5	1.3	13%
	Chvatal	1.7	1.3	24%
	CASA	1.7	1.2	29%
Smart Home SPL	ICPL	1.08	1.08	0%
	Chvatal	1.80	1.50	17%
	CASA	1.90	1.60	16%

Average number of configurations to detect a defects

Evaluation



Mobile Phone SPL



Smart Home SPL

Average number of configurations to detect a defect for 200 random orders

Evaluation

SPL	algorithm	Sampling	Similarity-based prior.	Percentage of prior. compared to sampling
Mobile Phone SPL	ICPL	$175ms$	$1ms$	0.6%
	Chvatal	$245.1ms$	$1ms$	0.4%
	CASA	$528.6ms$	$1ms$	0.2%
Smart Home SPL	ICPL	$1929.5ms$	$21.3ms$	1.1%
	Chvatal	$31900.7ms$	$20ms$	0.1%
	CASA	$641702.5ms$	$23.1ms$	0.004%

Average execution time of the sampling algorithms and similarity-based prioritization

Conclusion

- The rate of early defect detection of similarity-based prioritization is better than
  - Random,
  - CASA order,
  - And Chvatal order
- Better or at least equal to default order of ICPL algorithm.
- ICPL is better than
  - The default order of CASA,
  - And Chvatal algorithms.

Future Work

- Other criteria to be included in our prioritization approach (Multi-objectives).
- Other sampling algorithms such as,
  - AETG,
  - IPOG,
  - and MoSo-PoLiTe
- Use real test cases

Thank you for your attention.

References:

- B. J. Garvin, M. B. Cohen, and M. B. Dwyer. Evaluating Improvements to a Meta-Heuristic Search
- for Constrained Interaction Testing. Empirical Software Engineering, 16(1):61-102, 2011.
- M. F. Johansen, . Haugen, and F. Fleurey. An Algorithm for Generating T-Wise Covering Arrays from Large Feature Models. In Proc. Int'l Software Product Line Conf. (SPLC), pages 46-55. ACM, 2012.
- V. Chvatal. A Greedy Heuristic for the Set-Covering Problem. Mathematics of Operations Research,
- 4(3):233-235, 1979.
- D. R. Kuhn, D. R. Wallace, and A. M. Gallo, Jr. Software Fault Interactions and Implications for Software Testing. IEEE Trans. Software Engineering (TSE), 30(6):418-421, 2004.
- H. Hemmati and L. Briand. An Industrial Investigation of Similarity Measures for Model-Based Test Case Selection. In Proc. Int'l Symposium Software Reliability Engineering (ISSRE), pages 141-150. IEEE, 2010.
- Refstrup, J.G.: Adapting to change: Architecture, processes and tools: A closer look at HP's experience in evolving the Owen software product line. In: Proc. Int'l Software Product Line Conference, SPLC (2009), Keynote presentation
- C. Dietrich, R. Tartler, W. Schroder-Preikschat, and D. Lohmann. ¨ Understanding Linux Feature Distribution. In Proc. of MISS, pages 15–19. ACM, 2012.