towards system analysis with variability model metrics

thorsten berger, jianmei guo
how big is my system?

thorsten berger, jianmei guo
simple systems

simple model / simple feature modeling language

- Mobile phone
  - Camera
    - Fixed optics
    - Zoom
  - Flash
    - Basic flash
    - Adaptable flash
  - Redeye reduction

model adapted from keynote of J. Savolainen
systems software

Variability Model
(Problem Space)

Mapping

Code
(Solution Space)

Kernel

Header files

Source file

select and compile

optional source artifacts
target artifacts
a systems software model

feature modeling concepts

- hierarchy
- Boolean, integer, string features
- feature groups
- cross-tree constraints

scalability concepts

- derived features
- ranges
- expressive constraint languages
- visibility conditions
- defaults (also computed)
- capabilities
- binding modes
- hierarchy manipulation...

more systems software models

129 models
  108 – 8355 features
  languages: CDL and Kconfig
  system: 26K – 10.2M LOC

analysis tools
  CDLTools
  LVAT (credits: S. She)

abstractions
  configuration space in propositional logic (DIMACS)
  hierarchy plots
WHY TO MEASURE?
quantifying model properties

but we need to build models explaining relationships between measures
assure quality attributes

but our understanding of the relationship between measures and quality attributes is poor
HOW TO MEASURE?
measurement using metrics

understanding of low-level attributes of variability models is low!
metrics definition

goal: define metrics for low-level characteristics

9 structural metrics
   reflect size, shape, hierarchy, grouping

7 feature representation metrics
   reflect data types (switch, none, number, string), value domain restrictions
   (e.g., ratio of open value domain features), capabilities

10 feature constraint metrics
   constrained features, ratio of constraint types (e.g., derived, visibility, default)

3 dependency metrics
   CTCR, density, connectivity

prospective metrics
   hierarchy specifics, feature descriptions, feature-to-code mapping
examples

RConstr ... ratio of features declaring any constraint
RPurelyBoolConstr ... ratio of purely Boolean constraints
RCon ... connectivity of an abstracted dependency graph
RDen ... density of an abstracted dependency graph
preliminary experiment

ANALYSES USING METRICS
possible analysis techniques

interest in co-variance:
  association (correlation) analysis

interest in prediction:
  classification and regression

interest in outliers:
  clustering and anomaly detection
preliminary experiment

CORRELATION ANALYSIS
methodology

eight real-world systems with models and proper (C-based) codebases

correlation test criteria (limitations)
  model metrics have no normal distribution
  low sample size compared to the number of variables (34 model metrics, 23 code metrics)

Spearman correlation test
  significant level: p-value < 0.05
  Spearman is non-parametric and can detect non-linear relationships, to account for limitations of our dataset

qualitative inspection of correlations
selection of preliminary

RESULTS
model metric correlation test

goal: identify inherent model characteristics

<table>
<thead>
<tr>
<th>Model</th>
<th>Metric</th>
<th>Correlation</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Metric1</td>
<td>0.751</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>B</td>
<td>Metric2</td>
<td>0.842</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>C</td>
<td>Metric3</td>
<td>0.723</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>D</td>
<td>Metric4</td>
<td>0.811</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>E</td>
<td>Metric5</td>
<td>0.913</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>F</td>
<td>Metric6</td>
<td>0.692</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>G</td>
<td>Metric7</td>
<td>0.902</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>H</td>
<td>Metric8</td>
<td>0.734</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>I</td>
<td>Metric9</td>
<td>0.734</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>J</td>
<td>Metric10</td>
<td>0.734</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>K</td>
<td>Metric11</td>
<td>0.734</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>L</td>
<td>Metric12</td>
<td>0.734</td>
<td><strong>&lt;0.001</strong></td>
</tr>
<tr>
<td>M</td>
<td>Metric13</td>
<td>0.734</td>
<td><strong>&lt;0.001</strong></td>
</tr>
</tbody>
</table>

Note: ** indicates significance at the 0.001 level.
correlations and insights

model size and shape

number features, number top-level features and leaf features

-> equal growth at both levels; with other findings: shapes remain when models grow

ratio of abstract features strongly negatively correlated with branching, but strongly correlated with defaults

-> domain modeling does not produce wide trees, and the more manual effort goes into domain modeling, the more defaults are modeled

mean and median branching not correlated

-> many outliers (we knew before), median is the better measure
correlations and insights

feature constraints

CTCR correlated strongly with branching and strongly negatively with maximum depth

-> wider and less-deep trees have less opportunities to encode constraints in hierarchy

CTCR, connectivity and density of dependency graph highly correlate

-> more investigation required, but early indicator of regular, non-skewed structures
model and code relationship

code metrics from [Liebig et al. 2010]

adapted the cppstats tool, and ran it on our codebases

<table>
<thead>
<tr>
<th>metric</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOC</td>
<td>lines of code</td>
</tr>
<tr>
<td>NOFC</td>
<td>number of feature constants (features referenced in source code)</td>
</tr>
<tr>
<td>LOFC</td>
<td>lines of feature code</td>
</tr>
<tr>
<td>ND</td>
<td>average (AND) and maximal (NDMAX) nesting of conditional compilation directives (#IF*)</td>
</tr>
<tr>
<td>SD</td>
<td>scattering degree: average and standard deviation of the number of occurrences of features in different expressions of conditional compilation directives</td>
</tr>
<tr>
<td>TD</td>
<td>tangling degree: average and standard deviation of the number of features in expressions of conditional compilation directives</td>
</tr>
<tr>
<td>GRAN</td>
<td>number of #IFDEFS occurring at a certain kind of language granularity: global level (GRANGL), function or type level (GRANFL), block level (GRANBL), statement level (GRANSL), expression level (GRANEL), method signature level (GRANML)</td>
</tr>
<tr>
<td>TYPE</td>
<td>number of extensions under equivalent #IF* expressions: homogeneous extensions with duplicated code (HOM), heterogeneous extensions with varying code (HET), and mixed (HOOHE)</td>
</tr>
</tbody>
</table>

model and code relationship

goal: explore potential of predicting system characteristics
correlations and insights

sizes

model size metrics and code size metrics (LOC, NOFC, LOFC) very strongly correlated

size metrics very strongly correlated with code extension metrics HOM, HOHE, but not with HET

granularity

sizes strongly correlated with extension granularities (GRANGL, GRANFL, GRANBL, GRANSL, GRANEL, GRANML, and GRANERR)

-> identification of significant system characteristics

-> indications of system characteristics show that forward and reverse inference of model and code characteristics is possible
CONCLUSION
summary and conclusions

contributions: defined and implemented metrics on rich languages, a tool, quantitative datasets, qualitatively inspected correlations

model metrics provide insights
  analysis both confirms earlier findings and provides a complementary picture

model and code metric analysis can potentially provide insights
  for instance, for reverse-engineering techniques
  -> further analysis required, but needs better focus
outlook

evaluation of applicability of metrics to further languages and further real models

investigate prospective model metrics and feature metrics

connect to findings about computational and cognitive complexity

theoretical evaluation of the metrics regarding accepted properties (e.g., additivity, triangle inequality), for instance, using the DISTANCE framework

look at evolution?
and so?

models

https://bitbucket.org/tberger/variability-models
https://code.google.com/p/linux-variability-analysis-tools/source/browse/?repo=extracts

metrics and analysis tools

VMM https://bitbucket.org/tberger/vmm
LVAT (S. She) https://code.google.com/p/linux-variability-analysis-tools/
CDLTools https://bitbucket.org/tberger/cdltools

read on...

She, Berger: Formal Semantics of the Kconfig Language. Technical Note, 2010
Berger, She: Formal Semantics of the CDL Language. Technical Note, 2010
thanks for listening!

towards system analysis
with variability model metrics
thorsten berger, jianmei guo