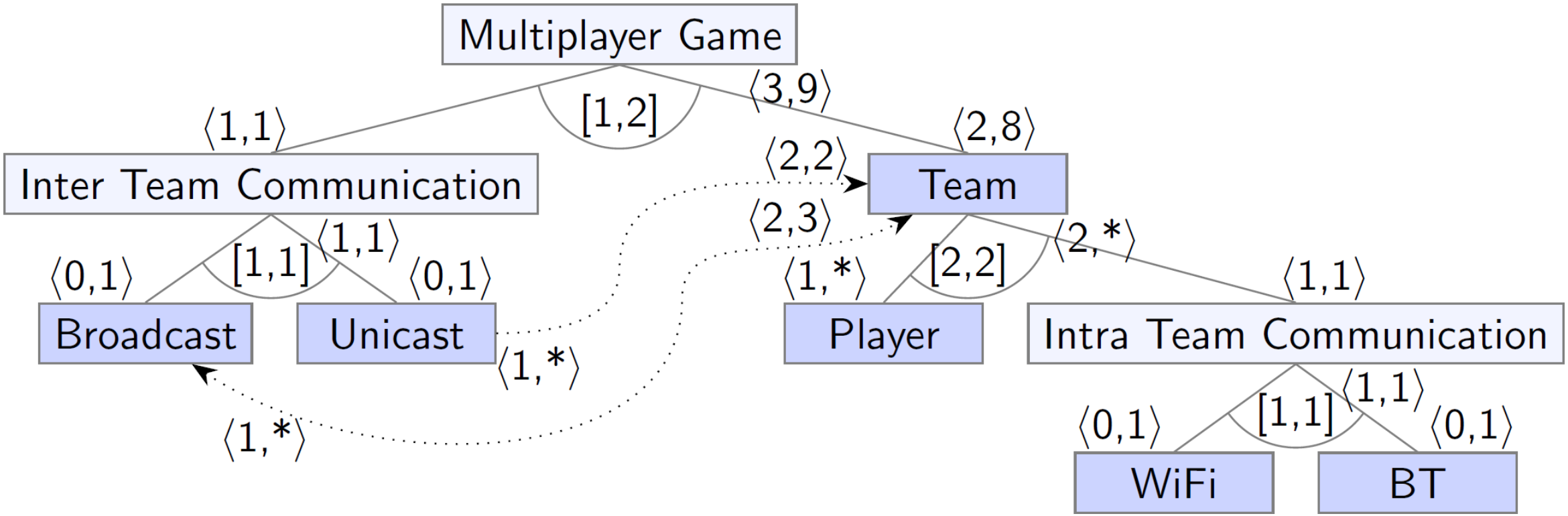
Sampling Cardinality-Based Feature Models

Lukas Güthing, Mathis Weiß, Malte Lochau, Ina Schaefer

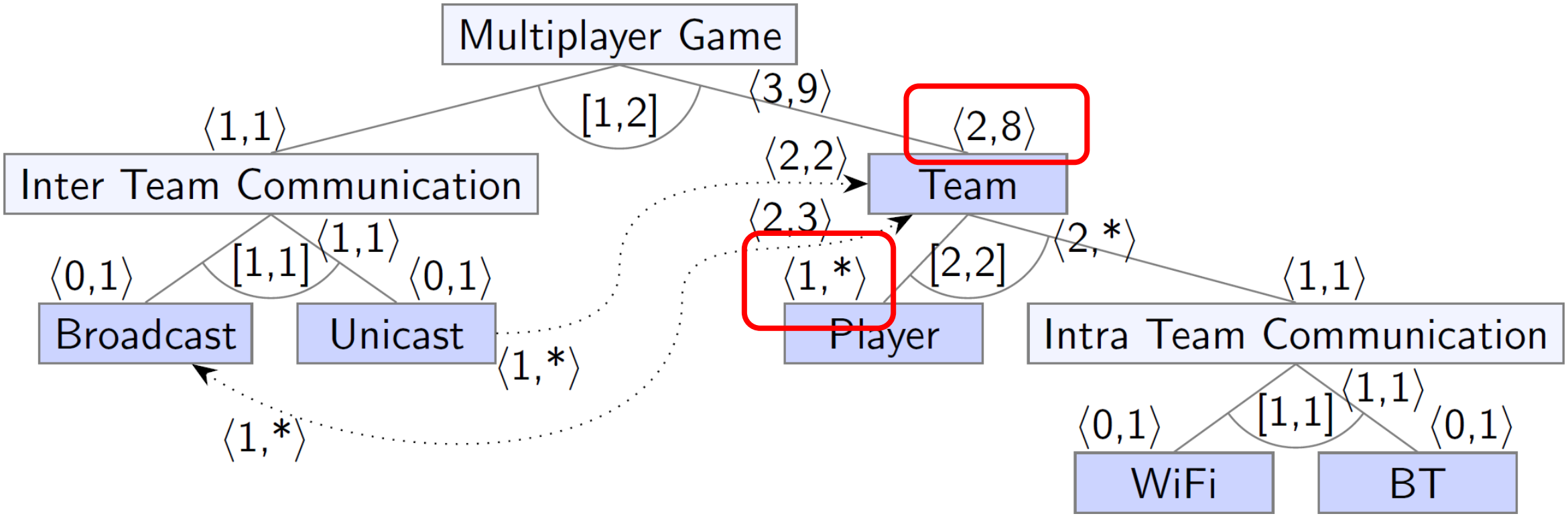
Cardinality-Based Feature Models



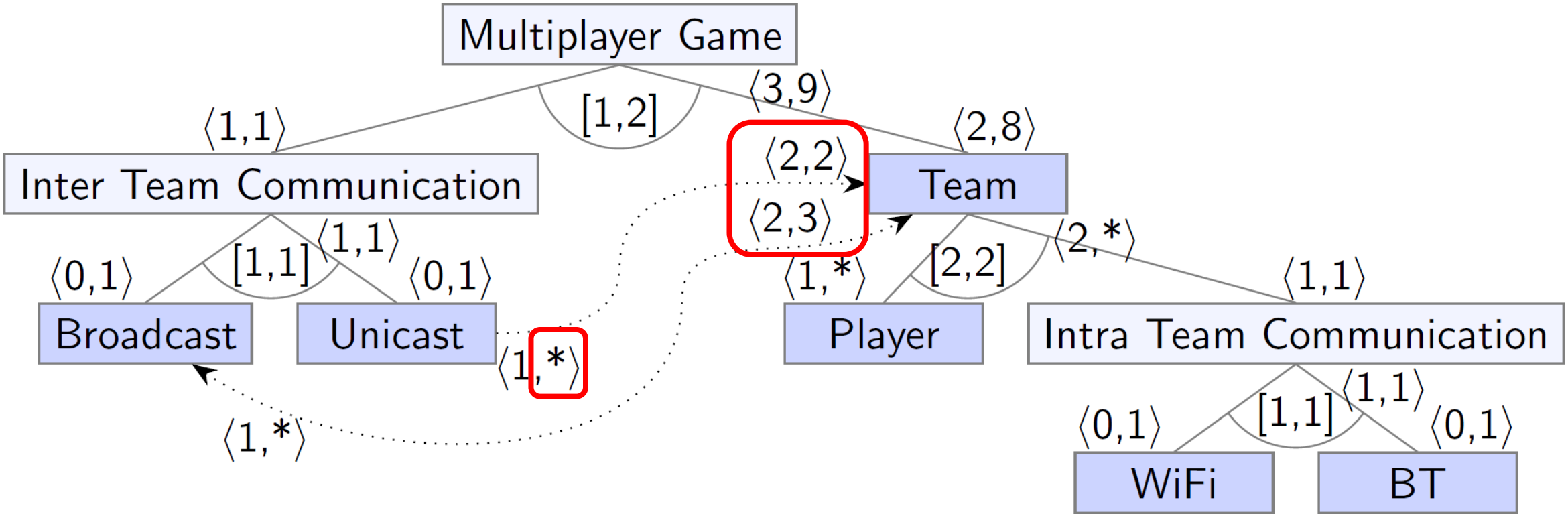
Challenges for Sampling: Sampling

- “Classical” sampling not trivial
- Different heuristics
  - Uniform Random
  - Distance-Based
  - *t*-wise
- Combinatorics increase effort

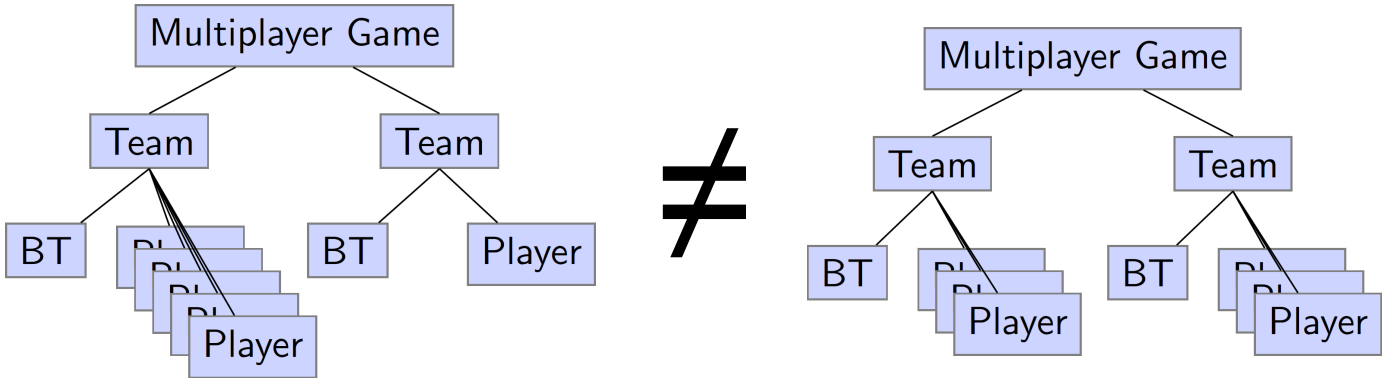
Challenges for Sampling: Complexity



Challenges for Sampling: New Anomalies



Challenges for Sampling: Semantics



#configurations:

734,638

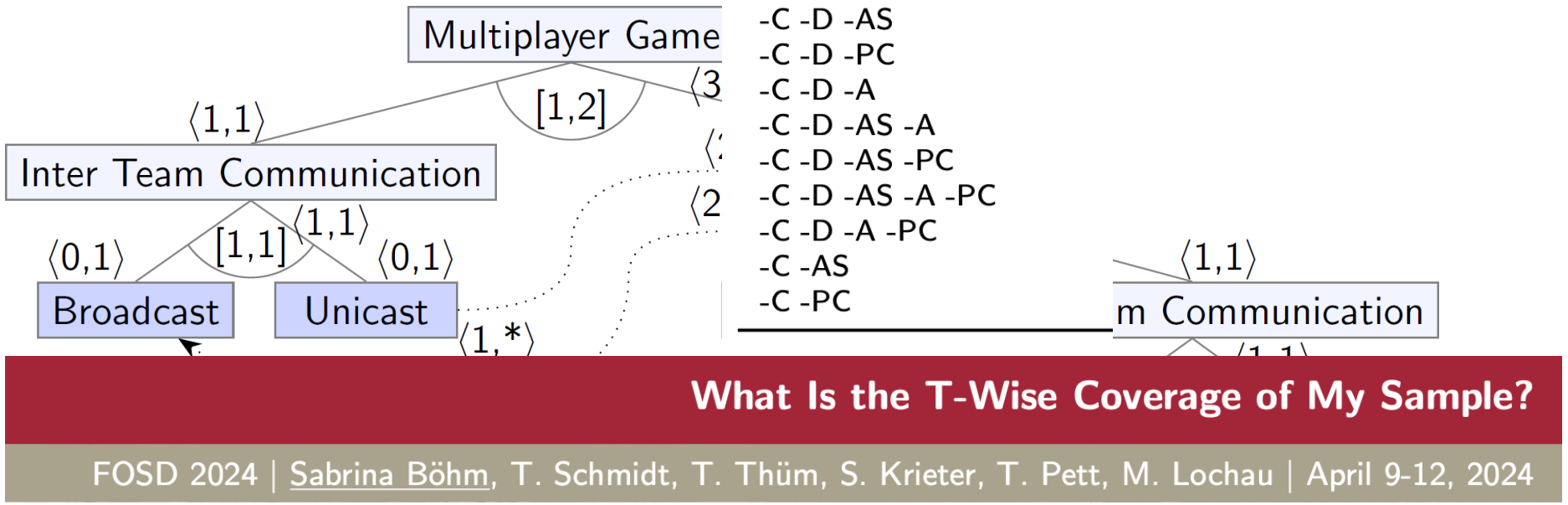
$$\{Team^2, BT^2, Player^6\} = \{Team^2, BT^2, Player^6\}$$

#configurations:

1,200

Challenges for Sampling: Criteria

- Criteria not trivially generalizable:
  - What is cardinality *t*-wise?



Challenges for Sampling

- Problem Space “Solutions”:
  - Resolve Infinities
  - Multiset semantics
  - Boundary Interior Concepts
- Next Steps:
  - Generalizing Criteria
  - Solver Representations

Sampling Cardinality-Based Feature Models

Lukas Güthing
lukas.guething@kit.edu
Karlsruhe Institute of Technology
Karlsruhe, Germany

Mathis Weiß
mathis.weiss@uni-siegen.de
University of Siegen
Siegen, Germany

Ina Schaefer
ina.schaefer@kit.edu
Karlsruhe Institute of Technology
Karlsruhe, Germany

Malte Lochau
malte.lochau@uni-siegen.de
University of Siegen
Siegen, Germany

ABSTRACT

The goal of sample-based testing of variant-rich software systems is to reduce usually very large configuration spaces to significantly smaller, yet still representative subsets of configurations to be tested for quality assurance. Recent sampling techniques and tools are restricted to finite-dimensional, Boolean configuration spaces specified by a feature model. However, in many modern application domains like cloud computing and cyber-physical systems, customers not only decide about the presence or absence of features in a configuration but also about the multiplicity (number of instances) of configurable resources. Cardinality-based feature models extend Boolean feature models by cardinality annotations and respective constraints to enable multiple, and even potentially a-priori unbounded, copies of features and their respective sub-trees. The resulting infinite and inherently non-convex configuration spaces are no longer tractable by established sampling criteria and corresponding sampling algorithms for Boolean feature models like pairwise feature interaction coverage. In this paper, we first revisit the subtleties of the configuration semantics of cardinality-based feature models. We propose novel sampling criteria explicitly taking multiplicity of feature selections into account. Finally, we present evaluation results gained from applying our tool implementation to a collection of example models, showing applicability of the proposed approach.

CCS CONCEPTS

• **Software and its engineering → Software product lines.**

KEYWORDS

software product lines, software variability, sampling, cardinality-based feature models

VaMoS 2024, February 07–09, 2024, Bern, Switzerland

ACM ISBN 979-8-4007-0877-0/24/02…$15.00
https://doi.org/10.1145/3634713.3634719

ACM Reference Format:
Lukas Güthing, Mathis Weiß, Ina Schaefer, and Malte Lochau. 2024. Sampling Cardinality-Based Feature Models. In *18th International Working Conference on Variability Modelling of Software-Intensive Systems (VaMoS 2024), February 07–09, 2024, Bern, Switzerland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3634713.3634719

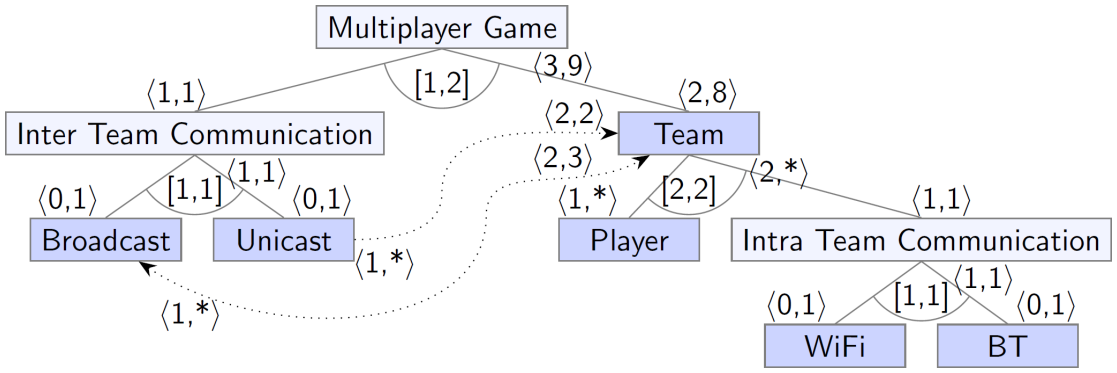
1 INTRODUCTION

Background and Motivation. Many modern software systems are highly configurable to diverse customer needs and application platforms. Feature models provide a graphical modeling language to describe valid configuration spaces from a problem-oriented point of view [7]. Feature models organize the set of features (user-configurable increments of functionality) in a tree-like hierarchy with further notational elements to define configuration constraints. Today, feature models enjoy a rich background theory and mature tool support. For instance, feature models serve as input for many recent tools for *sample-based testing* of variant-rich software [21]. The goal of sampling is to systematically reduce the inherently very large configuration space to significantly smaller, yet still representative subsets (samples) of configurations to be tested for quality assurance. As an effectiveness measure, recent sampling techniques and tools apply combinatorial coverage criteria inspired by functional black-box testing. In this way, fully automated sample selection can be performed. The widely considered k-wise feature interaction coverage criteria requires every possible on/off combination for any subset of k features to be covered in the sample [21].

In its original form, feature models and sampling tools are limited to *finite Boolean* configuration spaces: the number of features is fixed during domain analysis and every feature constitutes an on/off configuration decision of some optional functionality. However, in modern domains like cloud computing and cyber-physical systems, customers may also decide about the *multiplicity* (number of instances) of configurable resources. Cardinality-based feature models (CFM) extend Boolean feature models by cardinality constraints for features [3, 12, 23]. These constraints specify cardinality intervals on the number of feature instances: interval $\langle l, u \rangle$ denotes that feature f must be selected at least l and at most u times. Wildcard $*$ may be used instead of a fixed integer value, allowing a-priori unbounded number of feature instances. Moreover, for every instance of a feature f, the configuration contains an individually configurable copy (clone) of the whole sub-tree of f. This gives rise to *infinite* and *non-convex* configuration spaces of CFMs which are

46

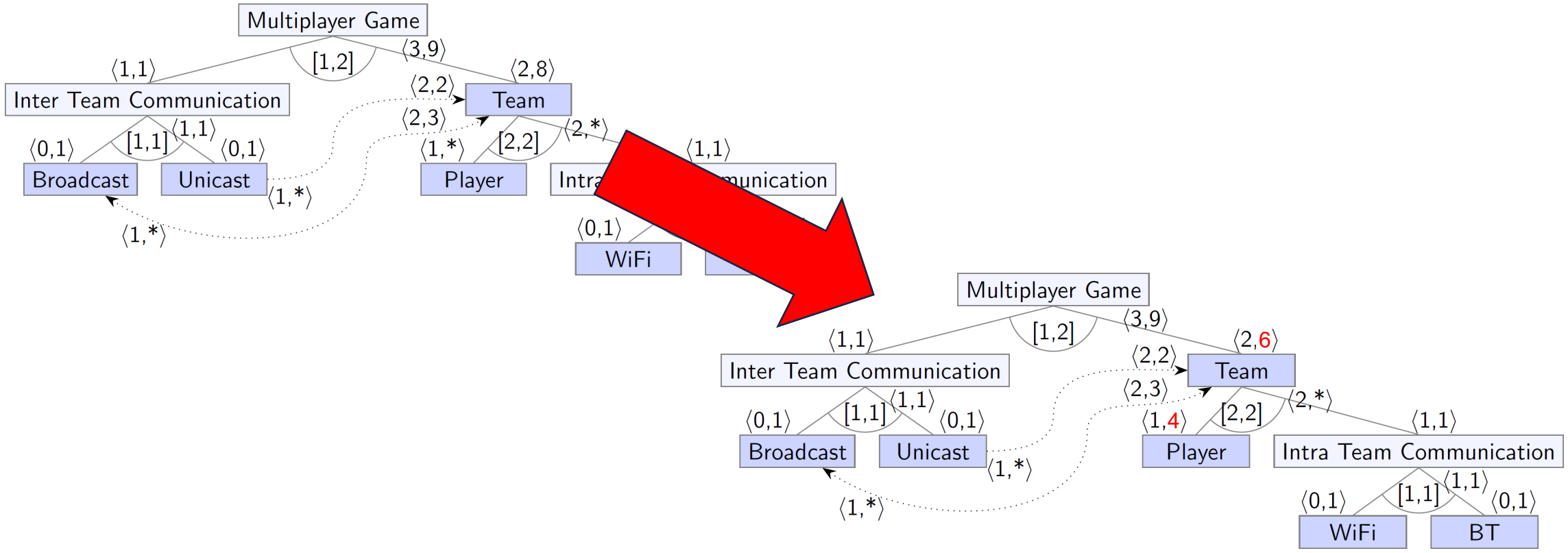
Solution Space as the “Solution”?



?

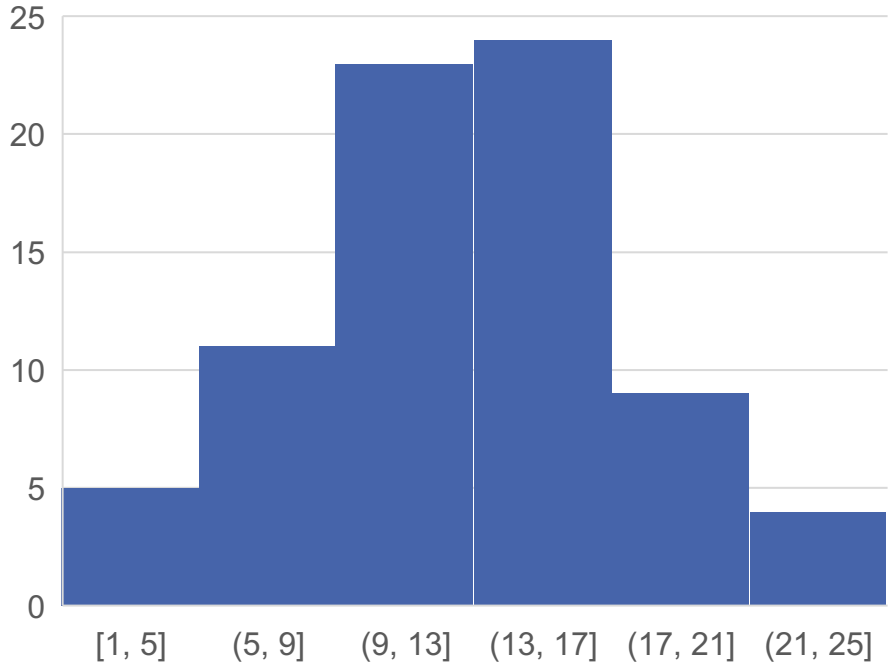
Prioritization: Solution-Space Knowledge

- Evaluate built variants



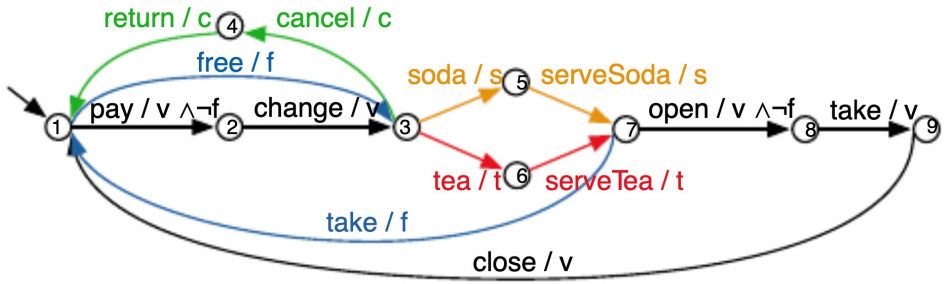
Prioritization: Solution-Space Knowledge

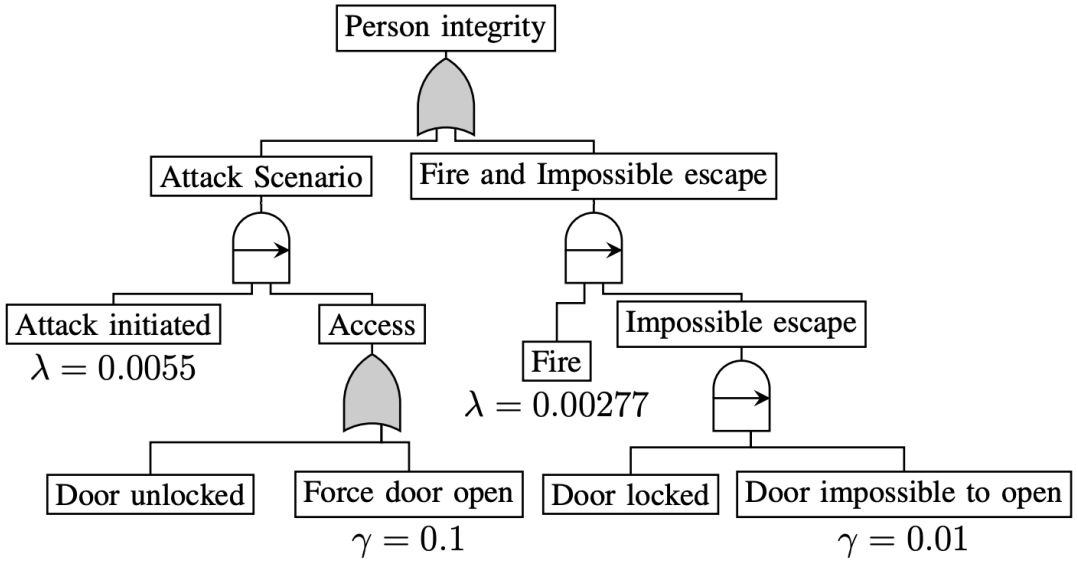
- Evaluate built variants
  - Probabilistic Feature Models?
  - Distributions of Multiplicities



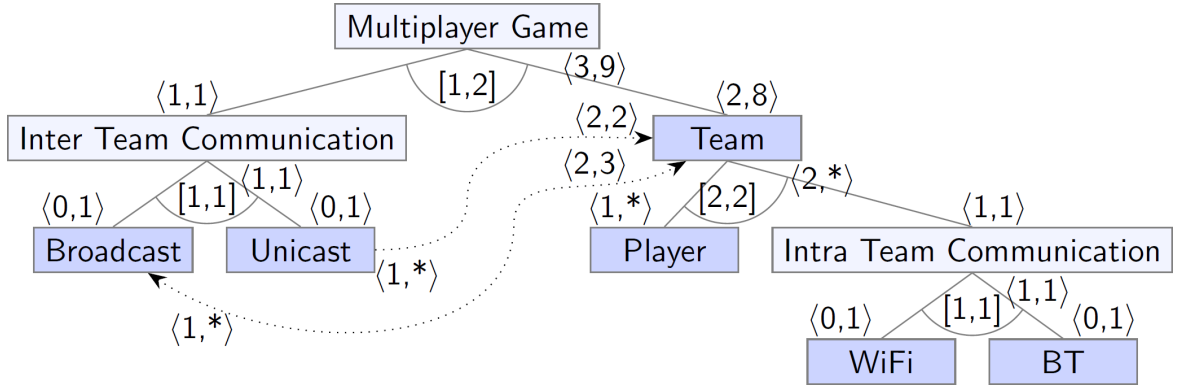
Prioritization: Solution-Space Knowledge

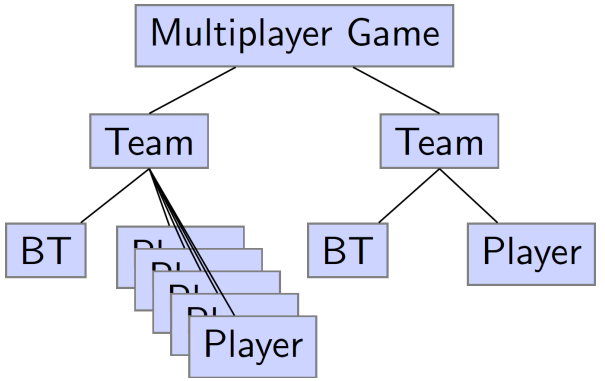
- Evaluate behavior
  - Model checking
  - Other behavioral models





Input is appreciated!





- Ideas
- Existing Concepts
- Available Tools
  - *Are you sure?*
- Implementations & Case Studies