Towards Reasoning about Product Lines with Design Choices

Navpreet Kaur, Michalis Famelis

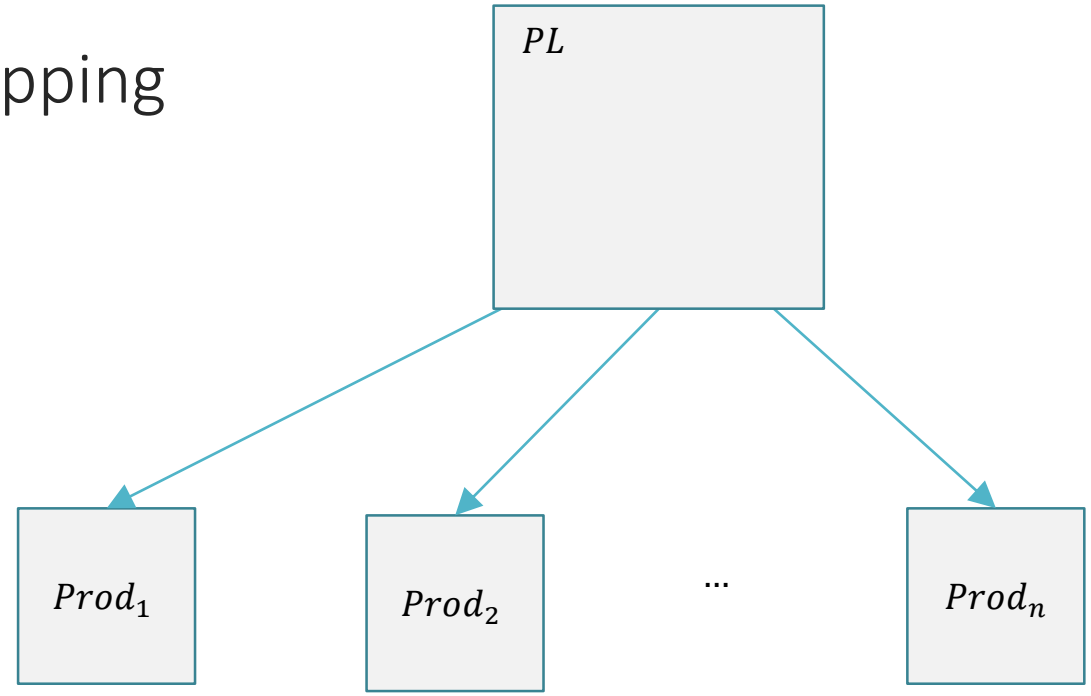
MoDeVVa @ MODELS 2019
Munich, DE

Software Product Line Basics

Feature model, Domain model, Feature mapping

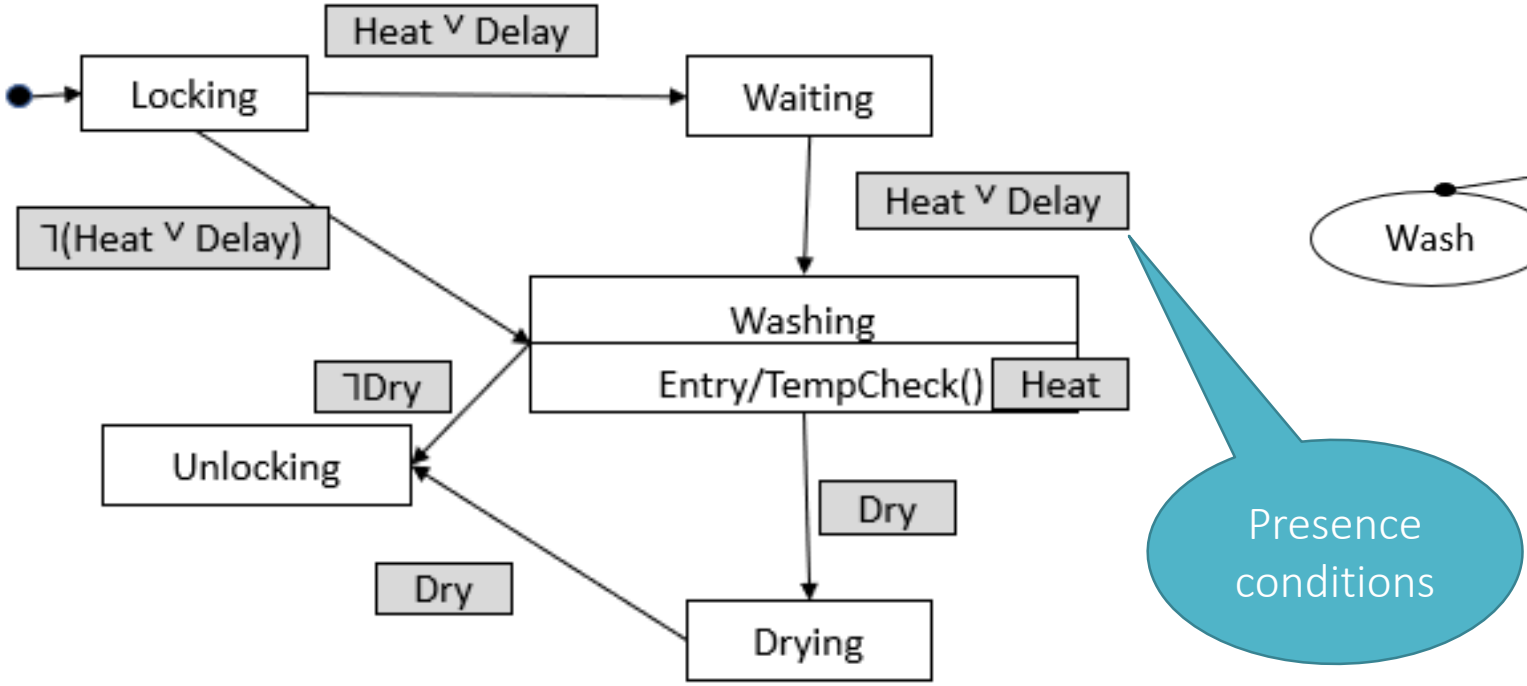
Different feature **configurations** result in different variants

PL models a **set** of related, but different products

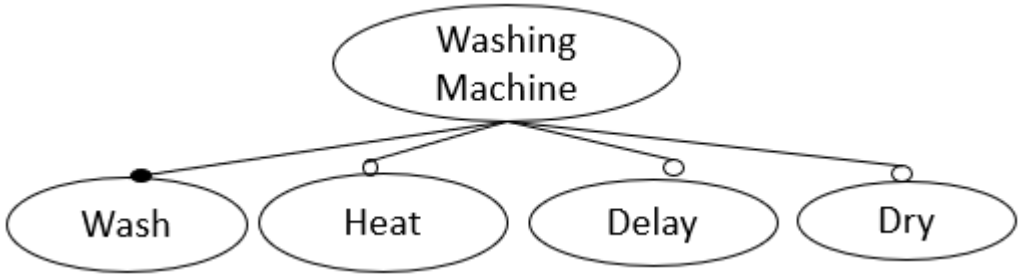


Washing Machine Product Line

Domain model

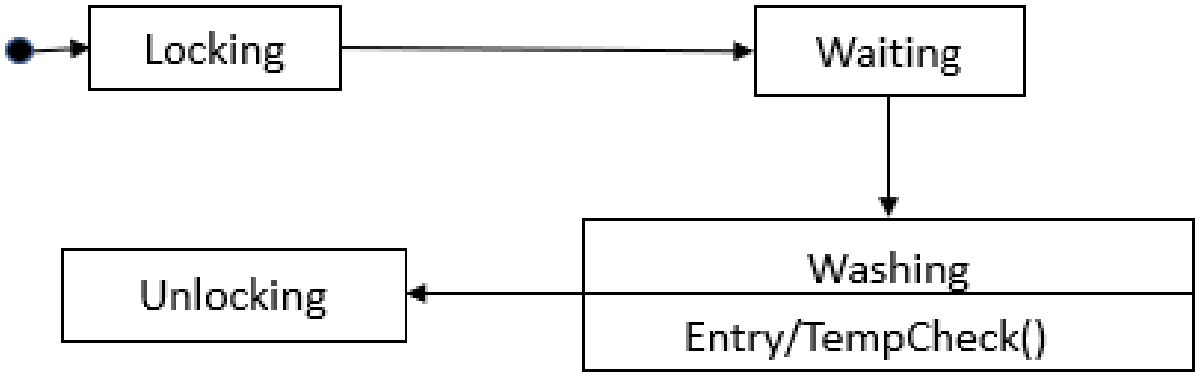


Feature model

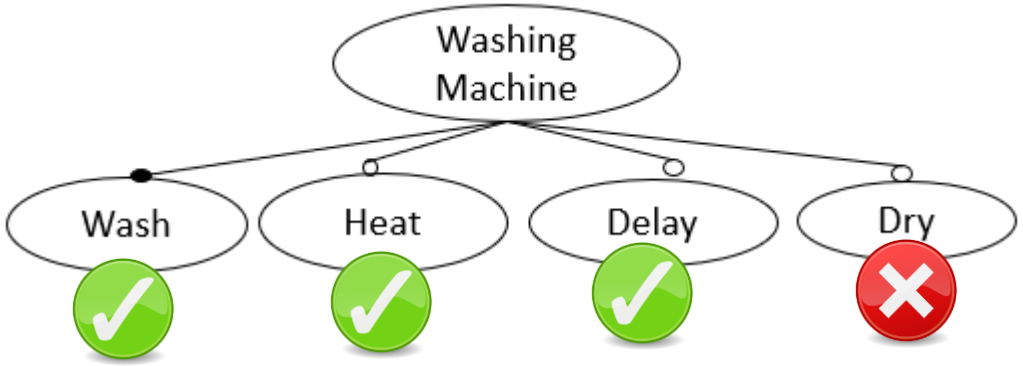


Washing Machine Product Line

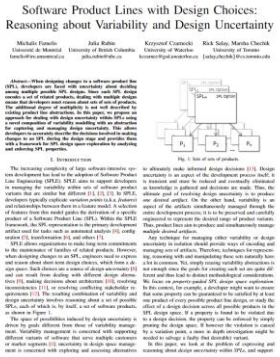
Variant



Feature model



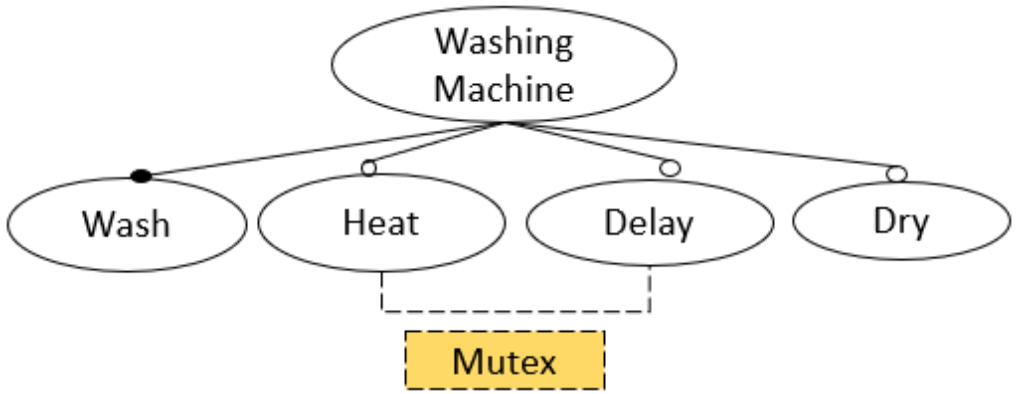
Two Different Kinds of Choices

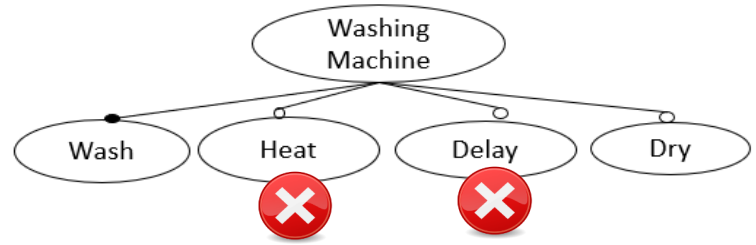

Software Product Lines with Design Choices: Reasoning about Variability and Design Uncertainty

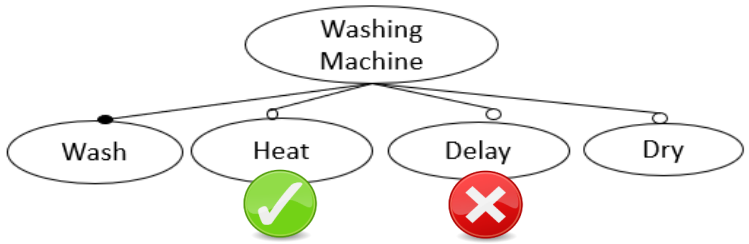
Vision paper
MODELS'17

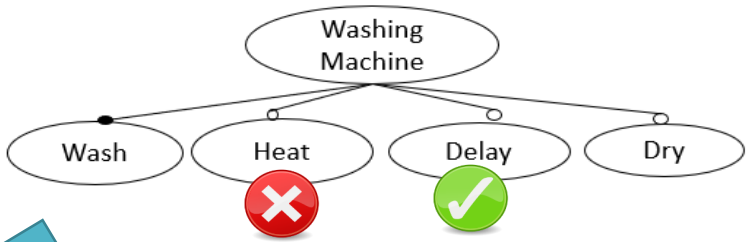
	Variability	Design Uncertainty
Reason	Market demands for product variants	Incomplete information, design alternatives, stakeholder conflicts, etc.
Granularity	Features	Decisions
Expression	Product line (PL) models	Partial models
Semantics	Set of artifacts produced by combinations of features	Set of artifacts produced by combinations of decisions
Horizon	Long term	Short term

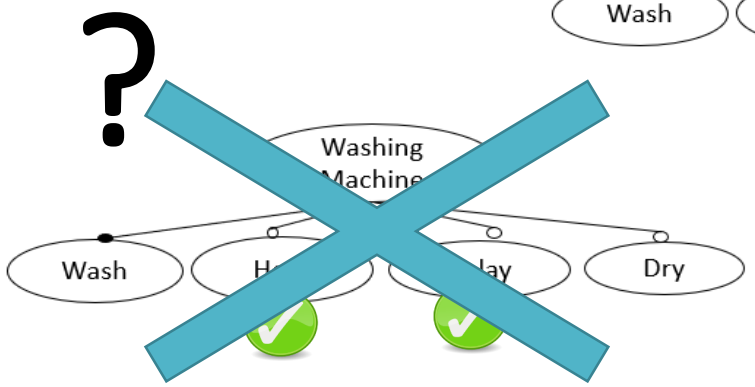
Design 1: Mutex Features?





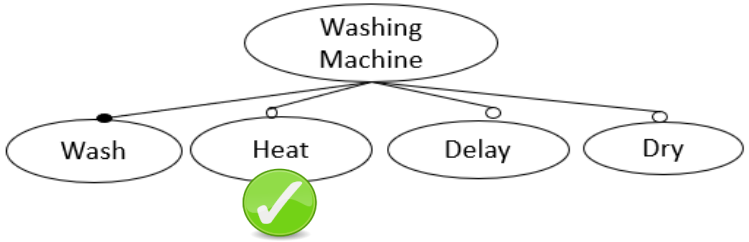


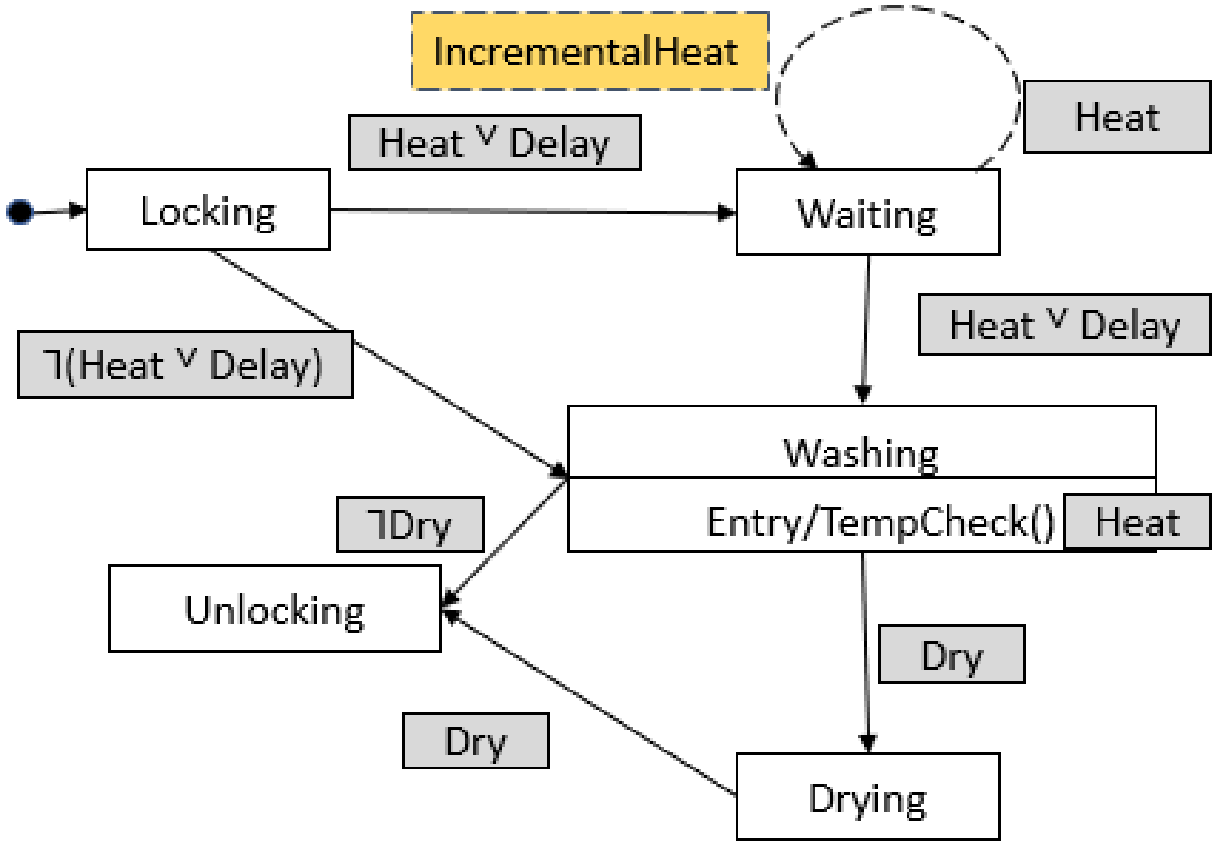


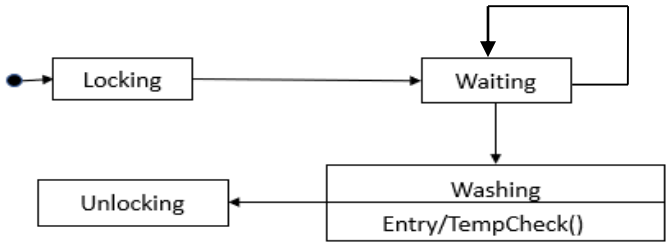


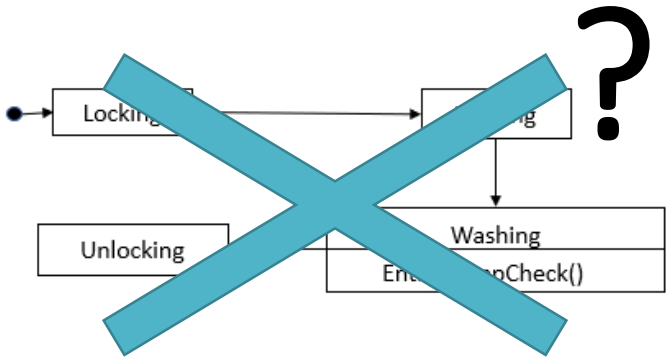
Uncertainty in the design of the feature model

Design 2: Incremental Heat?









Uncertainty in the design of the domain model

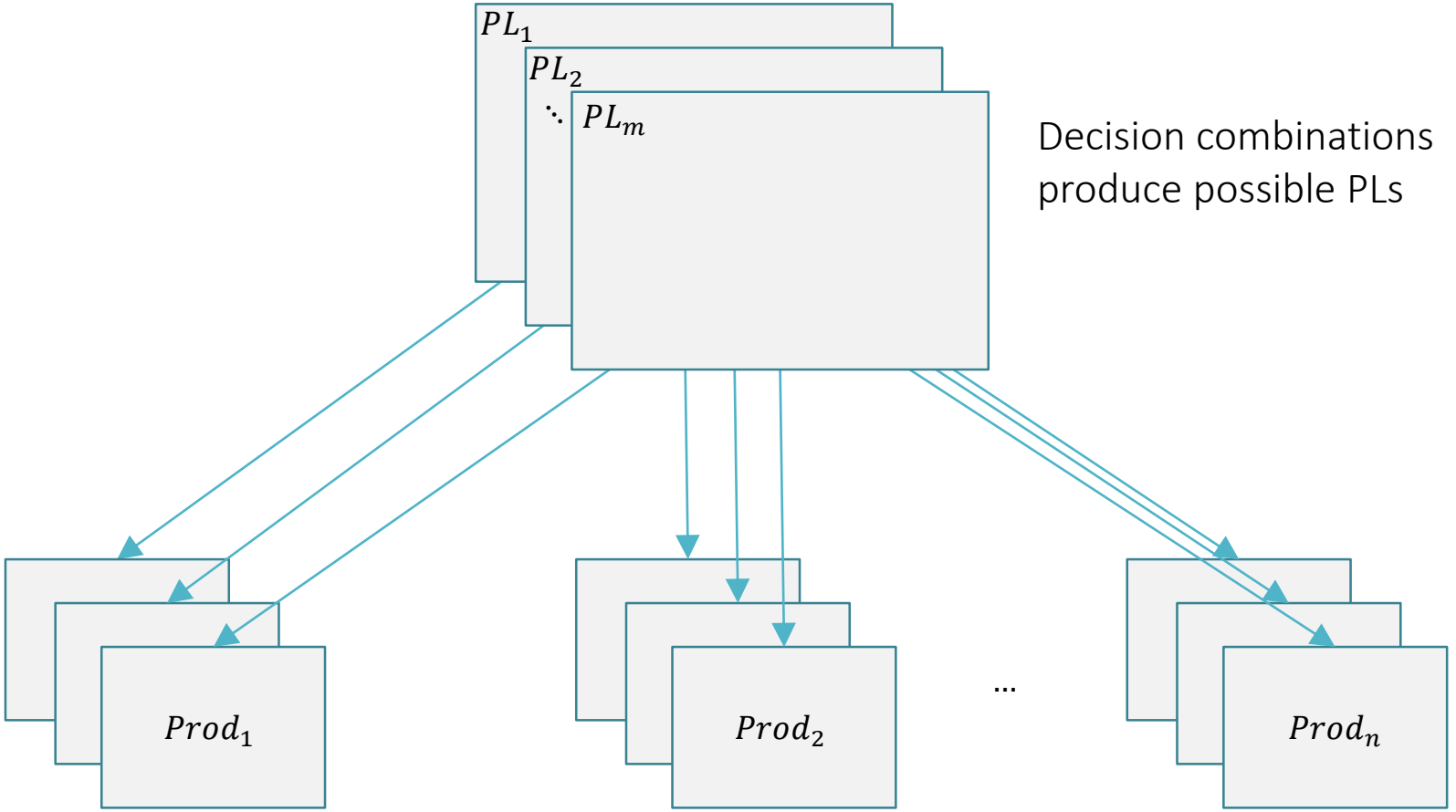
Software Product Lines with Design Choices

(SPLDCs)

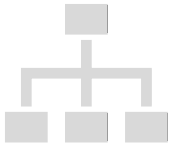
Design choices can affect the Feature model, Domain model, Feature mapping

They define a **design space** of product lines

Given a space of product lines, which one should be selected (and why)?



Software Product Lines with Design Choices

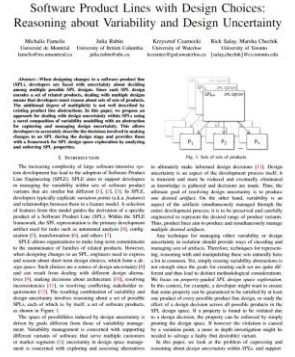


Motivation

How to model this design space?

What are relevant properties for exploring it?
How to check them?

What to do when properties are violated?



Vision paper
MODELS'17

In what follows,  signifies new contribution.

Tyson: an SPLDC language

Textual Syntax

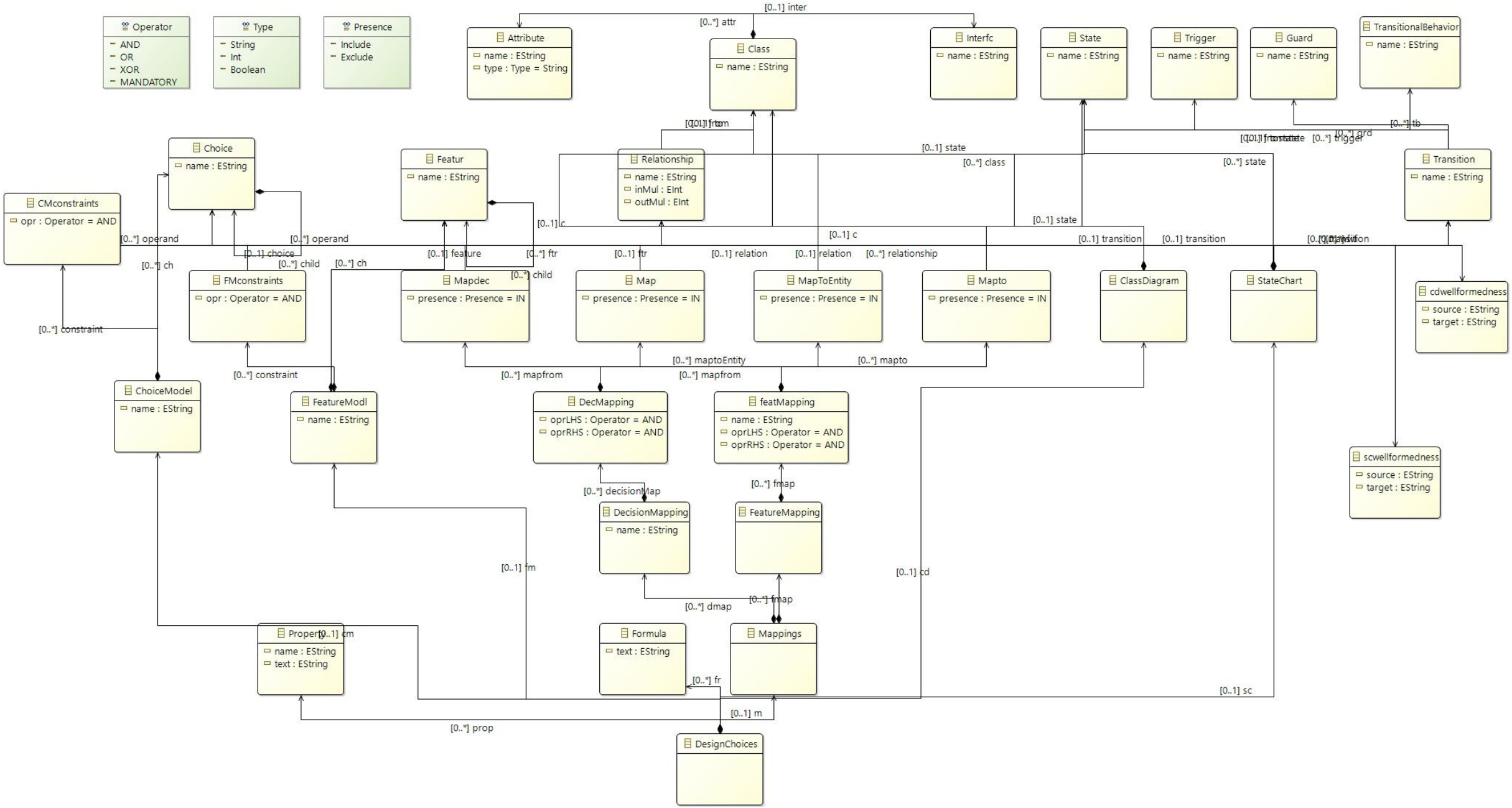
Supports Class Diagrams, Statecharts

Feature and Choice submodels (FM, CM)

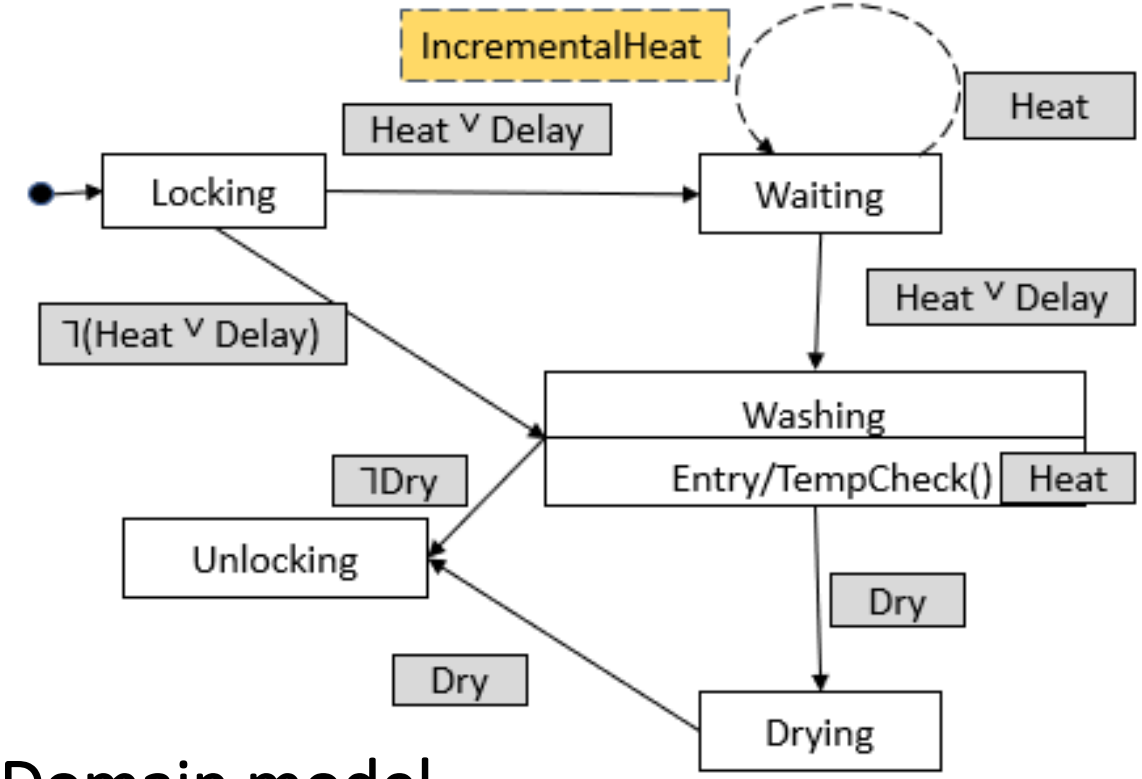
Emphasis on expressing mappings

Alloy semantics

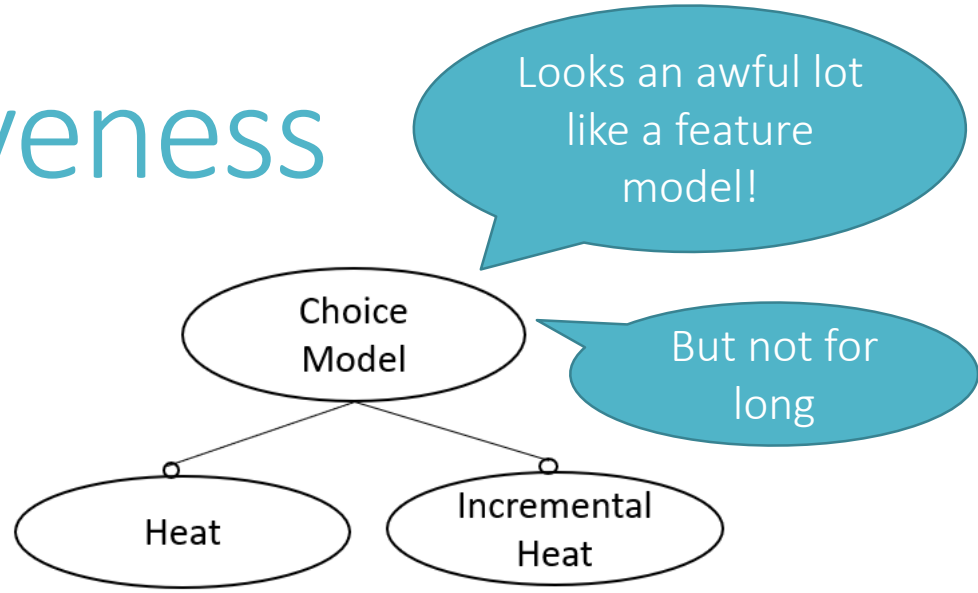
```
1   CM { Mutex; IncrementalHeat; }
2
3   FM { Wash; Delay; Dry; Heat;
4      FMConst: [ Mandatory (Wash) ] }
5
6   StateChart {
7     State Locking; State Waiting;
8     State Washing; State Drying;
9     State UnLocking;
10    Transition T1: Locking to Waiting
11    Transition T2: Waiting to Washing
12    Transition T3: Locking to Washing
13    Transition T4: Washing to UnLocking
14    Transition T5: Washing to Drying
15    Transition T6: Drying to UnLocking
16    Transition T7: Waiting to Waiting
17  }
18
19  Mappings {
20    FMap {
21      F1: {(Wash IN ) =>
22        AND (Transition T3 IN, Transition T4 IN)}
23      F2: {OR (Heat IN, Delay IN) =>
24        AND (Transition T1 IN, Transition T2 IN)}
25      F3: {(Dry IN) =>
26        AND (Transition T5 IN, Transition T6 IN)}
27    }
28    DMap {
29      D1: {Mutex IN =>
30        XOR (Feature Heat IN, Feature Delay IN)}
31      D2: {IncrementalHeat IN => Transition T7 IN}
32    }
33  }
```


Operator
AND
OR
XOR
MANDATORY
Type
String
Int
Boolean
Presence
Include
Exclude
Attribute
name : EString
type : Type = String
Class
name : EString
Interfc
name : EString
State
name : EString
Trigger
name : EString
Guard
name : EString
TransitionalBehavior
name : EString
Choice
name : EString
Featur
name : EString
Relationship
name : EString
inMul : EInt
outMul : EInt
Transition
name : EString
CMconstraints
opr : Operator = AND
FMconstraints
opr : Operator = AND
Mapdec
presence : Presence = IN
Map
presence : Presence = IN
MapToEntity
presence : Presence = IN
Mapto
presence : Presence = IN
ClassDiagram
StateChart
cdwellformedness
source : EString
target : EString
scwellformedness
source : EString
target : EString
ChoiceModel
name : EString
FeatureModl
name : EString
DecMapping
oprLHS : Operator = AND
oprRHS : Operator = AND
featMapping
name : EString
oprLHS : Operator = AND
oprRHS : Operator = AND
DecisionMapping
name : EString
FeatureMapping
Property
name : EString
text : EString
Formula
text : EString
Mappings
DesignChoices
[0..1] inter
[0..*] attr
[0..1] state
[0..*] class
[0..*] state
[0..*] operand
[0..*] operand
[0..1] choice
[0..*] child
[0..*] ch
[0..1] feature
[0..*] ftr
[0..*] child
[0..1] c
[0..1] ftr
[0..1] relation
[0..1] relation
[0..*] relationship
[0..1] c
[0..1] transition
[0..1] transition
[0..*] ch
[0..*] constraint
[0..*] constraint
[0..*] mapfrom
[0..*] maptoEntity
[0..*] mapfrom
[0..*] mapto
[0..*] decisionMap
[0..*] fmap
[0..1] fm
[0..1] cd
[0..*] dmap
[0..*] fmap
[0..1] cm
[0..*] fr
[0..1] sc
[0..1] m
[0..*] prop

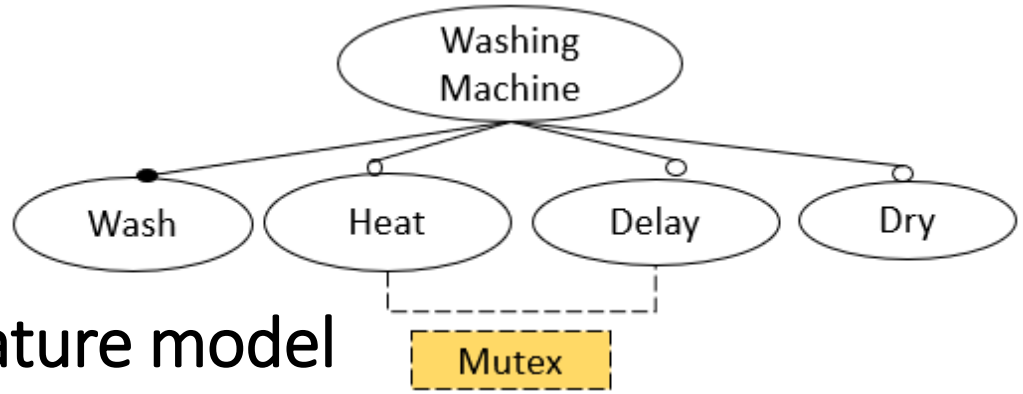
Currently: Limited Expressiveness



Domain model



Choice model



Feature model

Tyson: an SPLDC language

Textual Syntax

Supports Class Diagrams, Statecharts

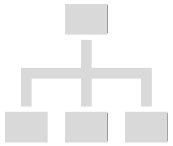
Feature and Choice submodels (FM, CM)

Emphasis on expressing mappings

Alloy semantics

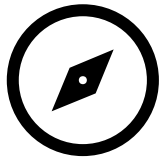
```
1  CM { Mutex; IncrementalHeat; }
2
3  FM { Wash; Delay; Dry; Heat;
4     FMConst: [ Mandatory (Wash) ] }
5
6  StateChart {
7    State Locking; State Waiting;
8    State Washing; State Drying;
9    State UnLocking;
10   Transition T1: Locking to Waiting
11   Transition T2: Waiting to Washing
12   Transition T3: Locking to Washing
13   Transition T4: Washing to UnLocking
14   Transition T5: Washing to Drying
15   Transition T6: Drying to UnLocking
16   Transition T7: Waiting to Waiting
17 }
18
19 Mappings {
20   FMap {
21     F1: {(Wash IN ) =>
22       AND (Transition T3 IN, Transition T4 IN)}
23     F2: {OR (Heat IN, Delay IN) =>
24       AND (Transition T1 IN, Transition T2 IN)}
25     F3: {(Dry IN) =>
26       AND (Transition T5 IN, Transition T6 IN)}
27   }
28   DMap {
29     D1: {Mutex IN =>
30       XOR (Feature Heat IN, Feature Delay IN)}
31     D2: {IncrementalHeat IN => Transition T7 IN}
32   }
33 }
```

Software Product Lines with Design Choices



Motivation

How to model this design space?



What are relevant properties for exploring it?
How to check them?

What to do when properties are violated?

Constraining the Design Space using Properties

For a product-level property P, we define four SPLDC-level properties using the modalities:

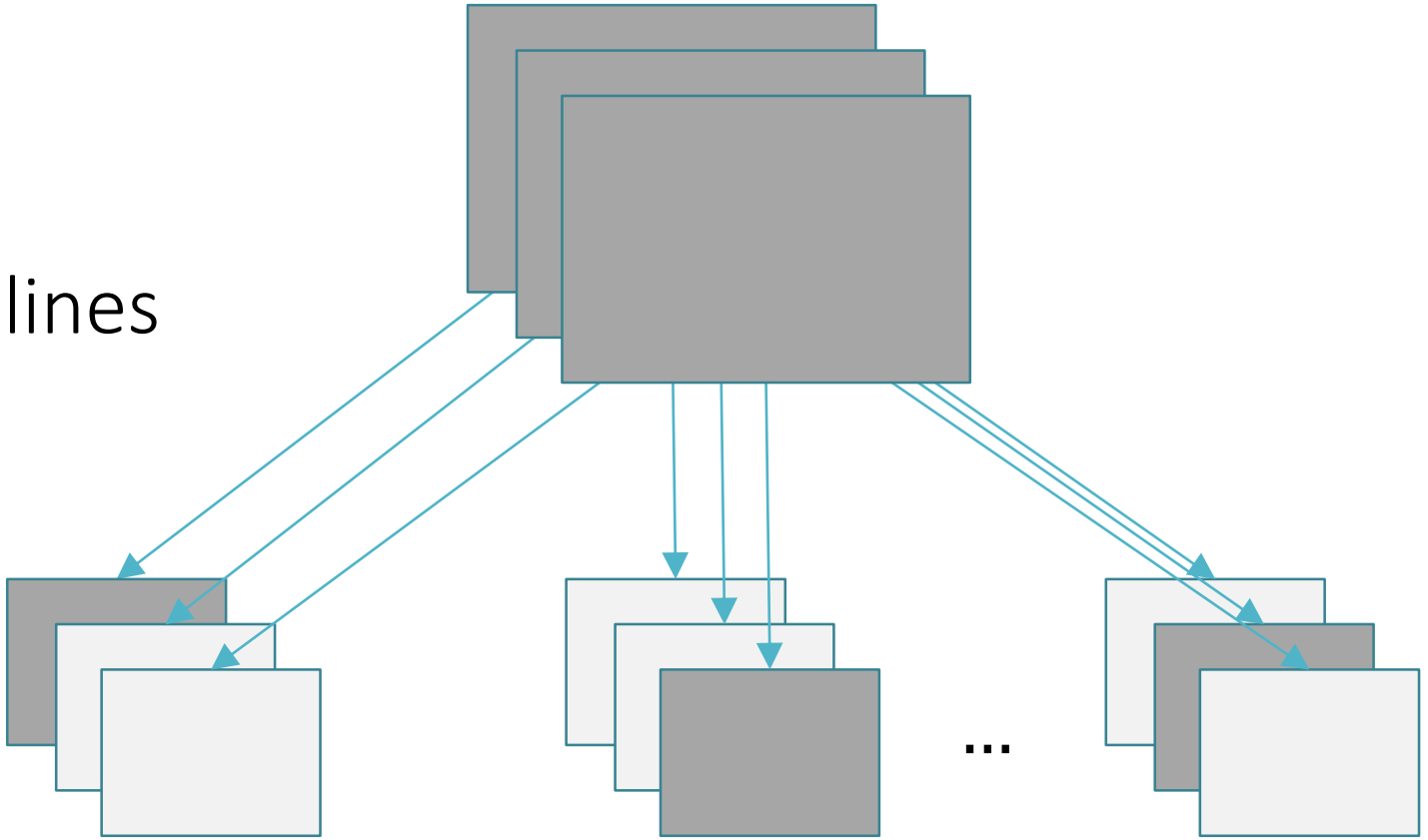
- Use All for **critical** properties and Some for **desirable** properties
- Use Necessary when you are **sure** it is needed and Possible when **unsure** but don't want to exclude the possibility

	Necessary for the product line	Possible for the product line
All products have P	All products in All product lines	All products in Some product line
Some products have P	Some product in All product lines	Some product in Some product line

Necessary-Some (*NS*)

All
product lines

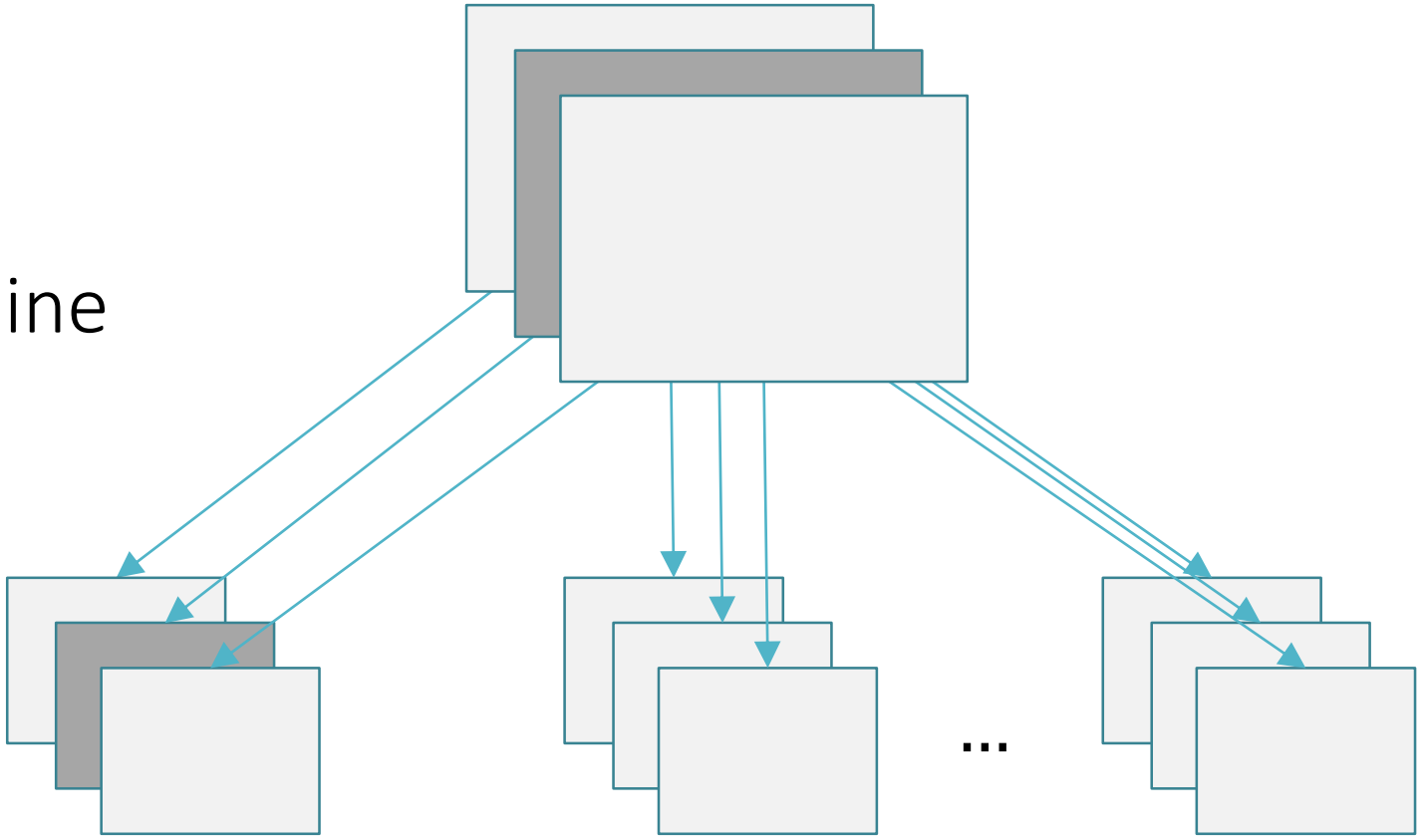
Some
product



Possible-Some (*PS*)

Some
product line

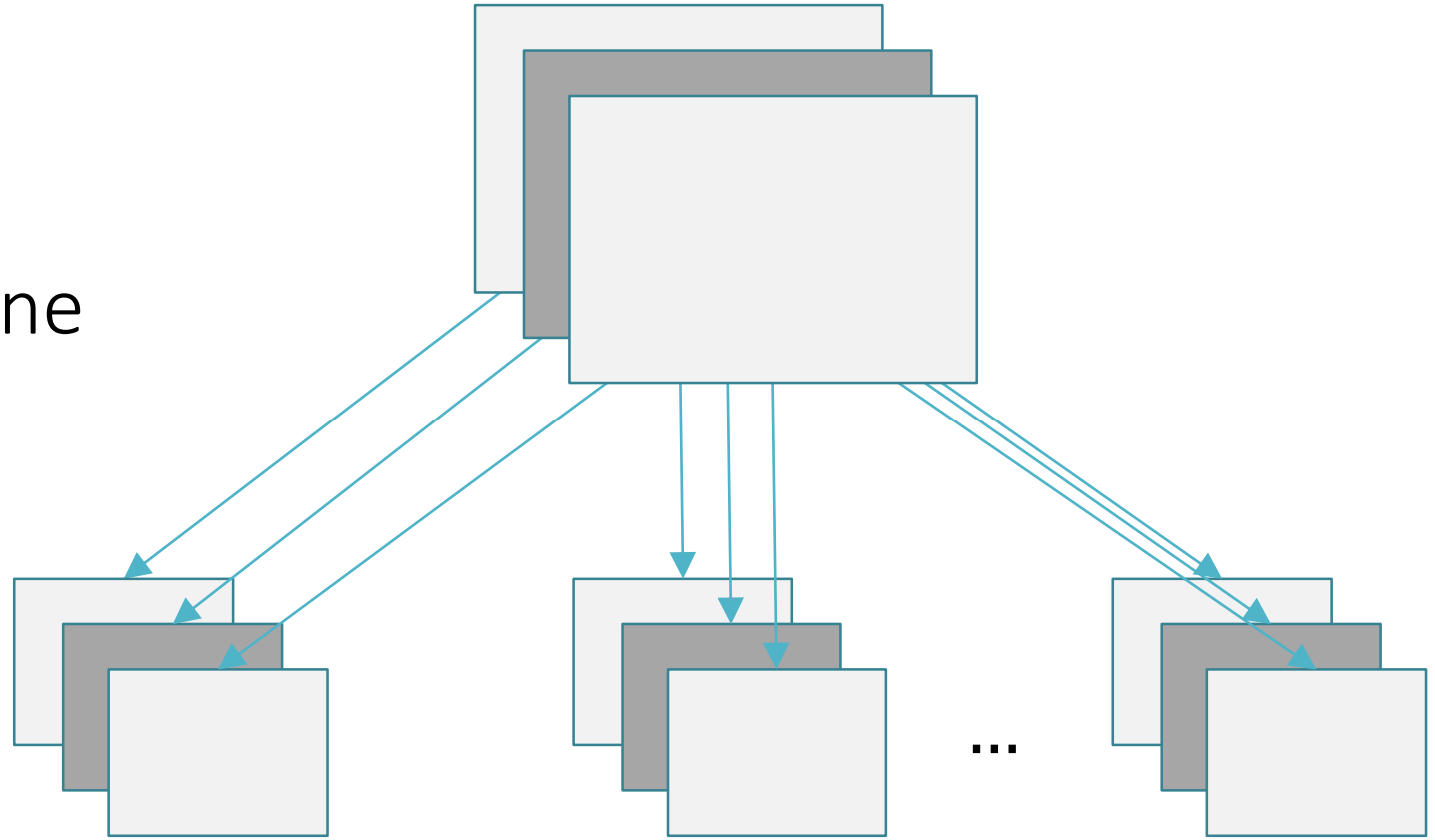
Some
product



Possible-All (*PA*)

Some
product line

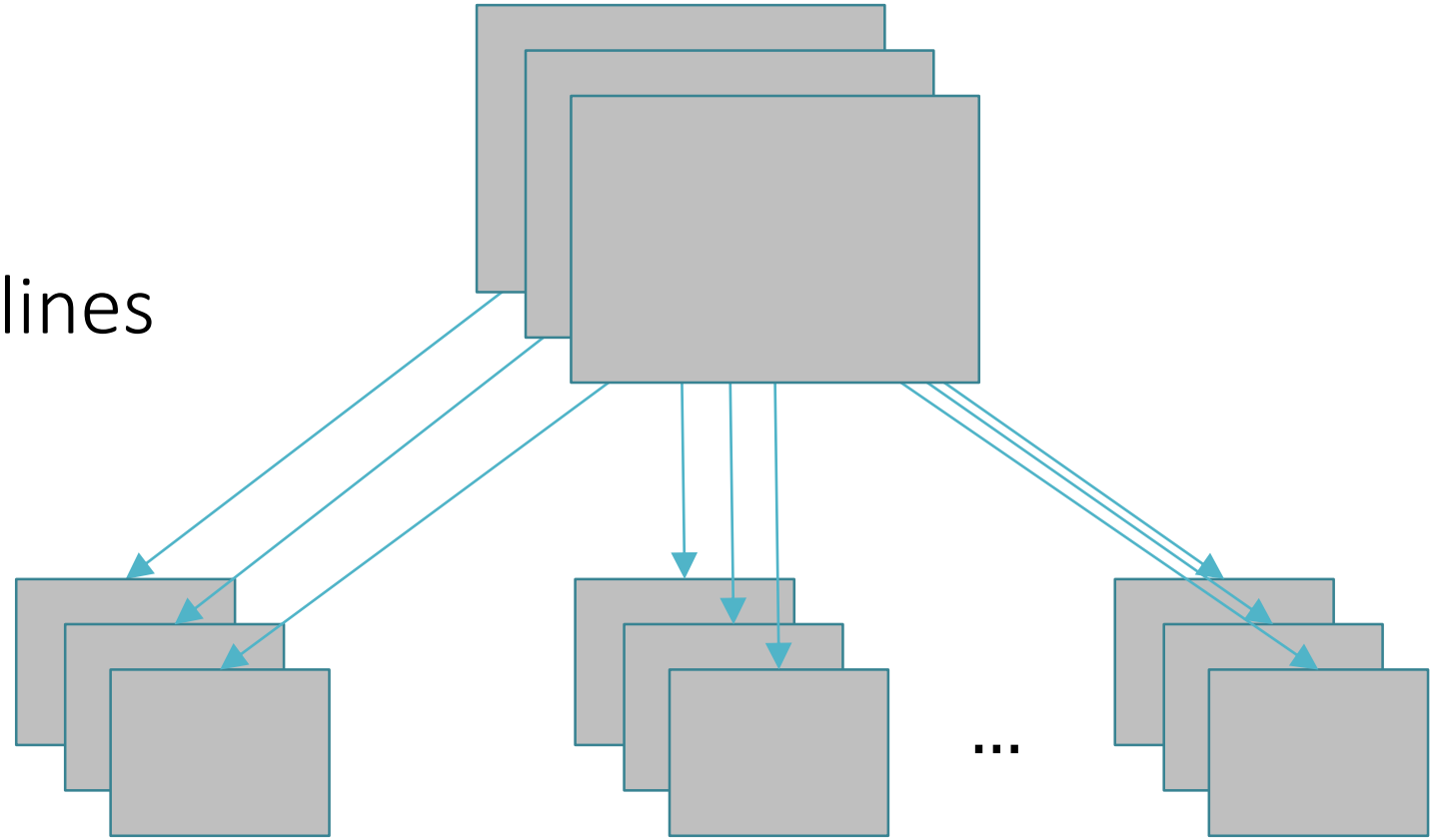
All
products



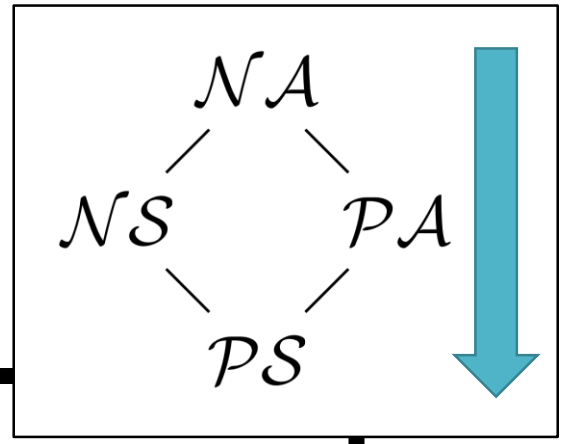
Necessary-All (*NA*)

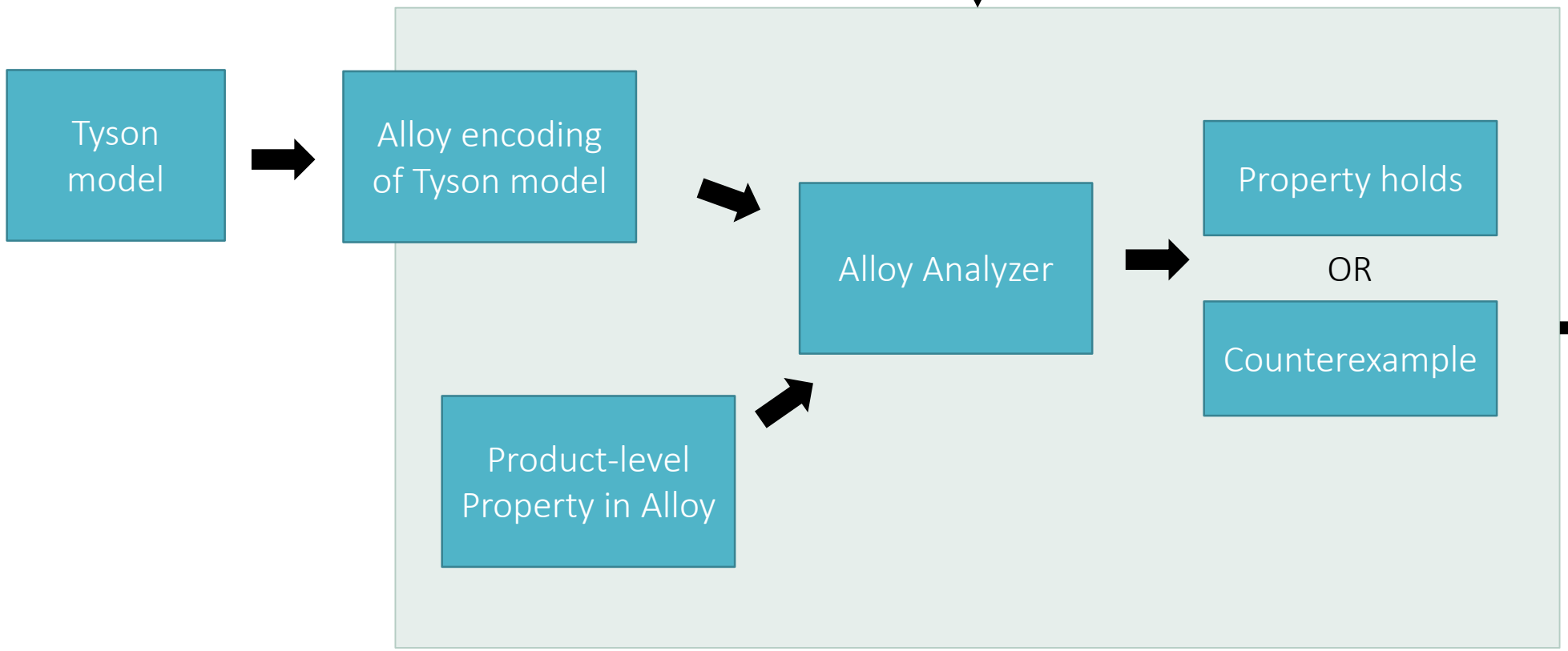
All
product lines

All
product



SPLDC Reasoning in Tyson





Evaluation

Does property checking Tyson SPLDCs scale?

Reasonably

http://splot-research.org/

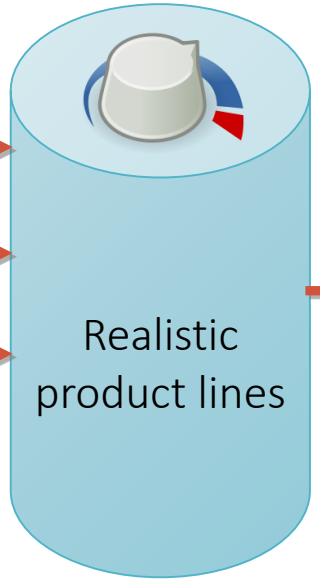
30 Feature models

30 Choice models

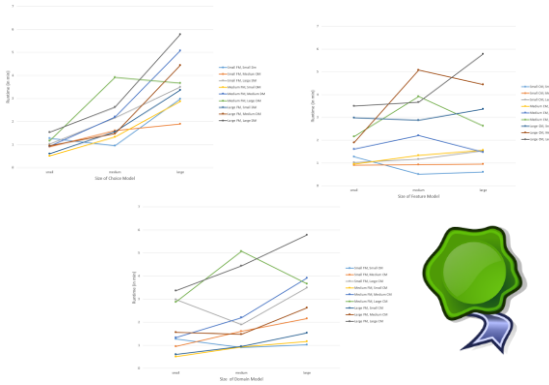
30 Domain models

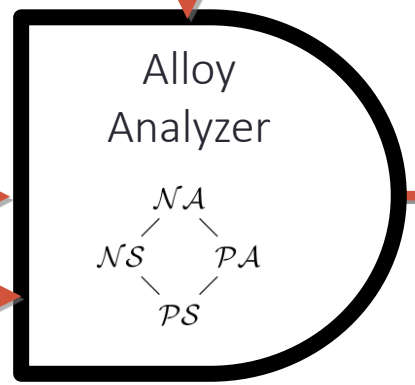
Metamodel zoo

a) Random combination
b) Random mappings



Fixed scope
(after pilots)



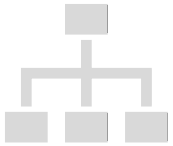


Property check result

Runtime

3 Properties
Inspired from
Van Der Straeten et al. UML'03

Software Product Lines with Design Choices



Motivation

How to model this design space?

What are relevant properties for exploring it?
How to check them?

What to do when properties are violated?

Next Steps

1. Improve expressiveness of choice models
   - Go beyond the closedness of feature modelling
   - Model the evolution/elicitation of uncertainty
2. Modularize Tyson
   - Who needs yet another DSL? UML Papyrus profile? MPS module?
3. Implement the property-based design space exploration vision of [MODELS'17]
   - Some very interesting properties of the lattice structure
4. Go beyond Alloy
   - Second order reasoners, Alloy*, QBF, …
5. Apply to reuse of pull-based software development

Towards Reasoning about Product Lines with Design Choices

Navpreet Kaur

Michalis Famelis
@MFamelis

geodes

Université de Montréal

Tyson: https://bitbucket.org/Navpreet15/dsl/

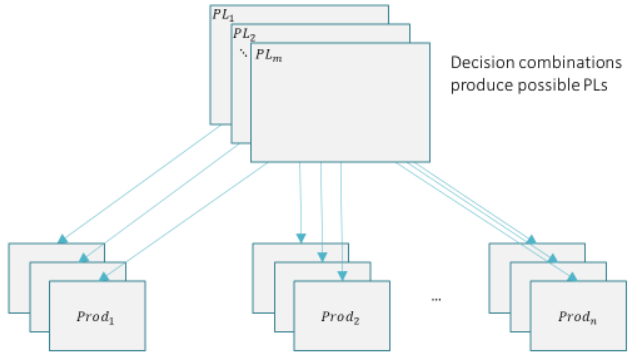
Software Product Lines with Design Choices

(SPLDCs)

Design choices can affect the Feature model, Domain model, Feature mapping

They define a **design space** of product lines

Given a space of product lines, which one should be selected (and why)?



Feature combinations produce possible products

Constraining the Design Space using Properties

For a product-level property P, we define four SPLDC-level properties using the modalities:

Use All for **critical** properties and Some for **desirable** properties

Use Necessary when you are **sure** it is needed and Possible when **unsure** but don't want to exclude the possibility

	Necessary for the product line	Possible for the product line
All products have P	All products in All product lines	All products in Some product line
Some products have P	Some product in All product lines	Some product in Some product line

Tyson: an SPLDC language

Textual Syntax

Supports Class Diagrams, Statecharts

Feature and Choice submodels (FM, CM)

Emphasis on expressing mappings

Alloy semantics

```
CM { Mutex; IncrementalHeat; }

FM { Wash; Delay; Dry; Heat;
   FMConst: [ Mandatory (Wash) ] }

StateChart {
  State Locking; State Waiting;
  State Washing; State Drying;
  State UnLocking;
  Transition T1: Locking to Waiting
  Transition T2: Waiting to Washing
  Transition T3: Locking to Washing
  Transition T4: Washing to UnLocking
  Transition T5: Washing to Drying
  Transition T6: Drying to UnLocking
  Transition T7: Waiting to Waiting
}

Mappings {
  FMap {
    F1: {(Wash IN ) =>
      AND (Transition T3 IN, Transition T4 IN)}
    F2: {OR (Heat IN, Delay IN) =>
      AND (Transition T1 IN, Transition T2 IN)}
    F3: {(Dry IN) =>
      AND (Transition T5 IN, Transition T6 IN)}
  }
  DMap {
    D1: {Mutex IN =>
      XOR (Feature Heat IN, Feature Delay IN)}
    D2: {IncrementalHeat IN => Transition T7 IN}
  }
}
```

SPLDC Reasoning in Tyson