VMC: A Tool for Product Variability Analysis

Maurice H. ter Beek
ISTI–CNR, Pisa, Italy

joint work with fellow FMT lab members:
P. Asirelli, A. Fantechi, S. Gnesi, F. Mazzanti & A. Sulova

FM 2012

Paris, France
29 August 2012

Outline

(Software) Product Line Engineering

Paradigm

To develop a family of products (product line) using a common platform and mass customization

Aim

To lower production costs of the individual products by

- letting them share an overall reference model of the product family
- allowing them to differ w.r.t. particular characteristics to serve, e.g., different markets

Product variants can be derived from a product family, thus allowing for reuse and differentiation

Production process

Maximize commonalities of product whilst minimizing cost of variations

(Software) Product Line Engineering

Paradigm

To develop a family of products (product line) using a common platform and mass customization

Aim

To lower production costs of the individual products by

- letting them share an overall reference model of the product family
- allowing them to differ w.r.t. particular characteristics to serve, e.g., different markets

Product variants can be derived from a product family, thus allowing for reuse and differentiation

Production process

Maximize commonalities of product whilst minimizing cost of variations

(Software) Product Line Engineering

Paradigm

To develop a family of products (product line) using a common platform and mass customization

Aim

To lower production costs of the individual products by

- letting them share an overall reference model of the product family
- allowing them to differ w.r.t. particular characteristics to serve, e.g., different markets

Product variants can be derived from a product family, thus allowing for reuse and differentiation

Production process

Maximize commonalities of product whilst minimizing cost of variations

Variability management/modeling

Feature modeling

Provide compact representations of all the products of a product family in terms of their *features* (pieces of functionality)

Variability modelling

How to explicitly define **optional**, **alternative**, **mandatory**, **required**, or **excluded** features of a product family as variation points

Managing variability with formal methods

Show that a certain product belongs to a product family or—instead—derive a product from a family by properly selecting features

Formally prove characteristics of products and families alike

Variability management/modeling

Feature modeling

Provide compact representations of all the products of a product family in terms of their *features* (pieces of functionality)

Variability modelling

How to explicitly define **optional**, **alternative**, **mandatory**, **required**, or **excluded** features of a product family as variation points

Managing variability with formal methods

Show that a certain product belongs to a product family or—instead—derive a product from a family by properly selecting features

Formally prove characteristics of products and families alike

Variability management/modeling

Feature modeling

Provide compact representations of all the products of a product family in terms of their *features* (pieces of functionality)

Variability modelling

How to explicitly define **optional**, **alternative**, **mandatory**, **required**, or **excluded** features of a product family as variation points

Managing variability with formal methods

Show that a certain product belongs to a product family or—instead—derive a product from a family by properly selecting features

Formally prove characteristics of products and families alike

Aim of our research activity at large

Aim

- One formal framework to express both feature-based constraints over the products of a family and constraints over their behavior
- Tool support for product derivation and for the formal verification (by model checking) of properties over products and families alike

Outcome : iFM'10, ACOTA @ ASE'10, PLEASE @ ICSE'11, FMOODS'11, SEW-34 @ FM'11, SPLC'11, iFM'12, SPLC'12

- MTS: Modal Transition Systems (Larsen et al.)
- MHML: CTL-like action- and state-based branching-time temporal logic (a.k.a. vACTL)
- VMC: Variability Model Checker

Aim of our research activity at large

Aim

- One formal framework to express both feature-based constraints over the products of a family and constraints over their behavior
- Tool support for product derivation and for the formal verification (by model checking) of properties over products and families alike

Outcome : iFM'10, ACOTA @ ASE'10, PLEASE @ ICSE'11, FMOODS'11, SEW-34 @ FM'11, SPLC'11, iFM'12, SPLC'12

- MTS: Modal Transition Systems (Larsen et al.)
- MHML: CTL-like action- and state-based branching-time temporal logic (a.k.a. vACTL)
- VMC: Variability Model Checker

Running example: family of coffee machines

Feature-based constraints

- The only accepted coins are 1€, exclusively for European products, and 1$, exclusively for Canadian products (**alternative** features)
- All products must offer coffee (**mandatory** feature); only European products may offer cappuccino (**excludes** relation among features)
- A ringtone must be rung in products offering cappuccino (**requires** relation among features), while it may be rung in other products (**optional** feature)

Behavioral constraints

- After coin insertion, user must press a button to choose whether (s)he wants sugar, after which (s)he may select a beverage
- The optional ringtone is rung after delivering a beverage

Running example: family of coffee machines

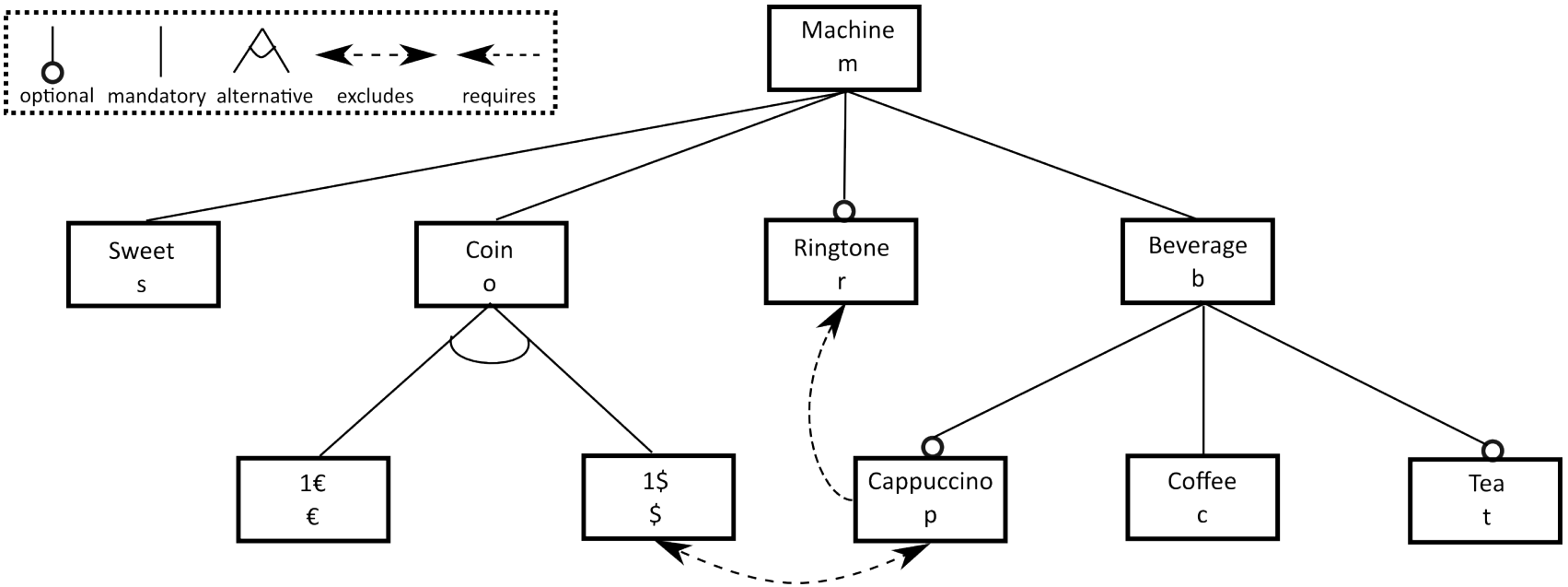
Feature-based constraints

- The only accepted coins are 1€, exclusively for European products, and 1$, exclusively for Canadian products (**alternative** features)
- All products must offer coffee (**mandatory** feature); only European products may offer cappuccino (**excludes** relation among features)
- A ringtone must be rung in products offering cappuccino (**requires** relation among features), while it may be rung in other products (**optional** feature)

Behavioral constraints

- After coin insertion, user must press a button to choose whether (s)he wants sugar, after which (s)he may select a beverage
- The optional ringtone is rung after delivering a beverage

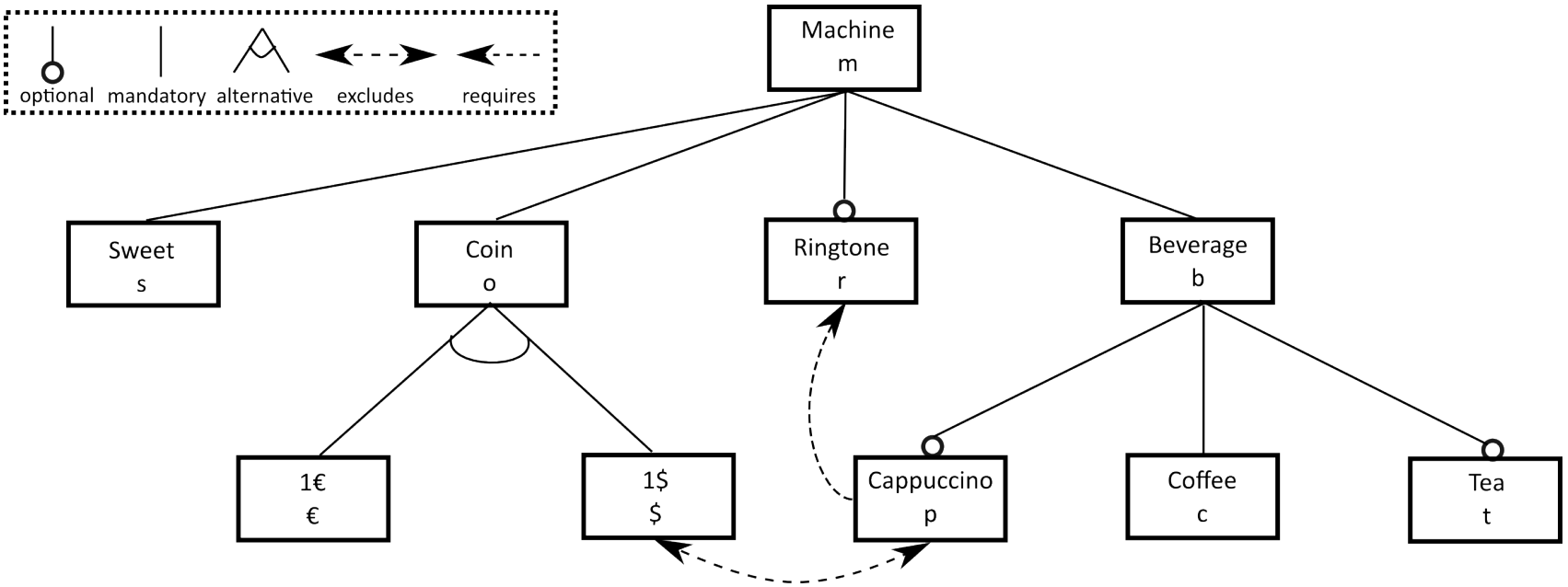
Coffee machine family: Feature model



10 different valid products (coffee machines defined by features)

$$\{\{m, s, o, b, c, €\}, \{m, s, o, b, c, €, r\}, \{m, s, o, b, c, €, t\},$$
$$\{m, s, o, b, c, \$\}, \{m, s, o, b, c, \$, r\}, \{m, s, o, b, c, \$, t\},$$
$$\{m, s, o, b, c, €, t, r\}, \{m, s, o, b, c, €, p, r\},$$
$$\{m, s, o, b, c, \$, t, r\}, \{m, s, o, b, c, €, p, r, t\}\}$$

Coffee machine family: Feature model



10 different valid products (coffee machines defined by features)

$$\{\{m, s, o, b, c, €\}, \{m, s, o, b, c, €, r\}, \{m, s, o, b, c, €, t\},$$
$$\{m, s, o, b, c, \$\}, \{m, s, o, b, c, \$, r\}, \{m, s, o, b, c, \$, t\},$$
$$\{m, s, o, b, c, €, t, r\}, \{m, s, o, b, c, €, p, r\},$$
$$\{m, s, o, b, c, \$, t, r\}, \{m, s, o, b, c, €, p, r, t\}\}$$

Modal Transition Systems (MTSs)

Use for behavioral modeling of SPLs recognized by Uchitel et al.

An MTS is an LTS distinguishing **optional** (may) and **mandatory** (must) transitions to formalize a product family's

- *underlying behavior*, shared among all products, and
- *variation points*, differentiating between products

A product (LTS) is derived by including all (reachable) must transitions and a subset of the (reachable) may transitions

MTS however cannot model variability constraints regarding **alternative** features nor regarding the **requires** and **excludes** inter-feature relations

Our solution: add a set of variability constraints (hiding our logic) to the MTS to define which derivable products must be considered valid ones (we defined an algorithm to derive only—and possibly all—valid ones)

Modal Transition Systems (MTSs)

Use for behavioral modeling of SPLs recognized by Uchitel et al.

An MTS is an LTS distinguishing **optional** (may) and **mandatory** (must) transitions to formalize a product family's

- *underlying behavior*, shared among all products, and
- *variation points*, differentiating between products

A product (LTS) is derived by including all (reachable) must transitions and a subset of the (reachable) may transitions

MTS however cannot model variability constraints regarding **alternative** features nor regarding the **requires** and **excludes** inter-feature relations

Our solution: add a set of variability constraints (hiding our logic) to the MTS to define which derivable products must be considered valid ones (we defined an algorithm to derive only—and possibly all—valid ones)

Modal Transition Systems (MTSs)

Use for behavioral modeling of SPLs recognized by Uchitel et al.

An MTS is an LTS distinguishing **optional** (may) and **mandatory** (must) transitions to formalize a product family's

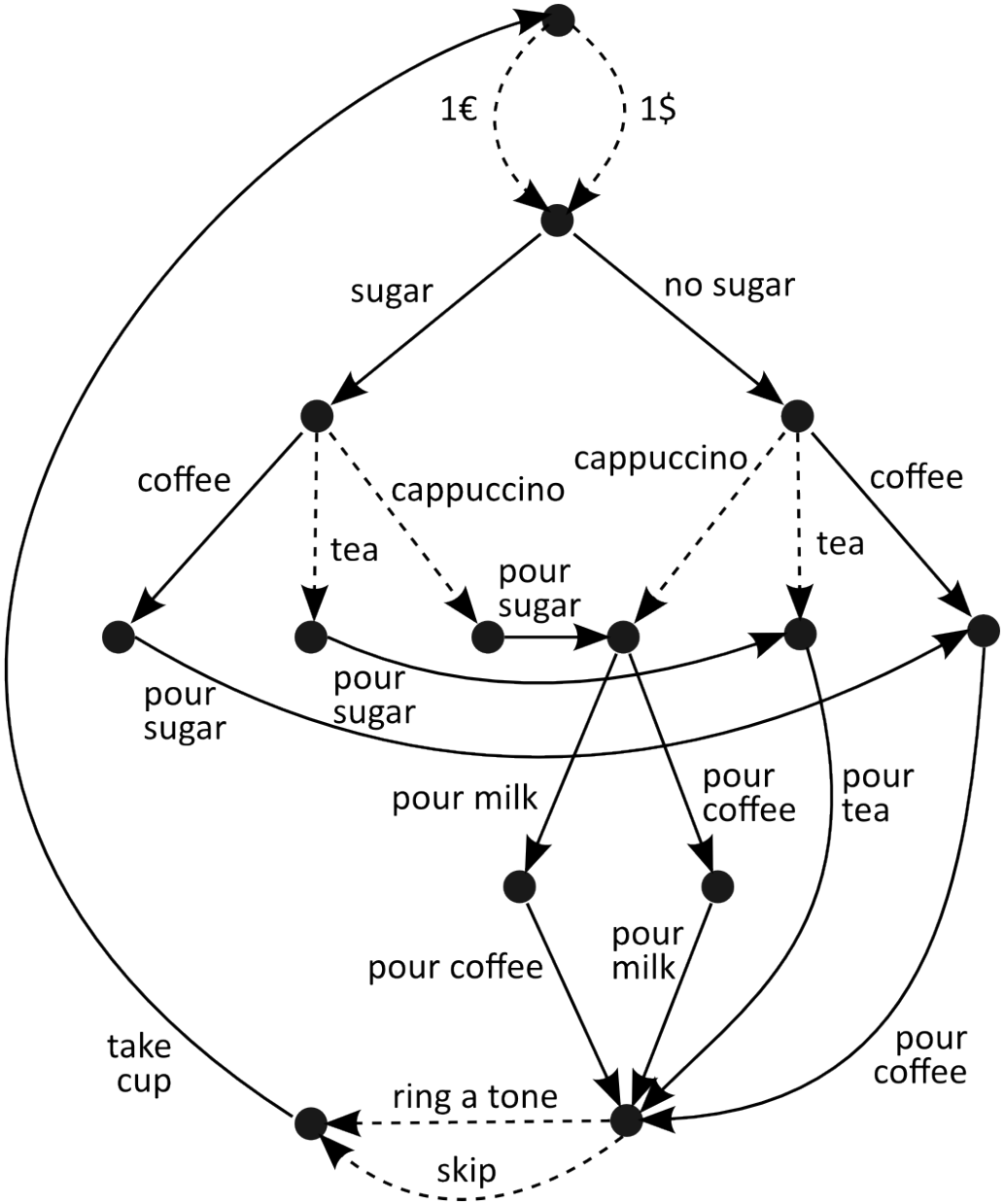
- *underlying behavior*, shared among all products, and
- *variation points*, differentiating between products

A product (LTS) is derived by including all (reachable) must transitions and a subset of the (reachable) may transitions

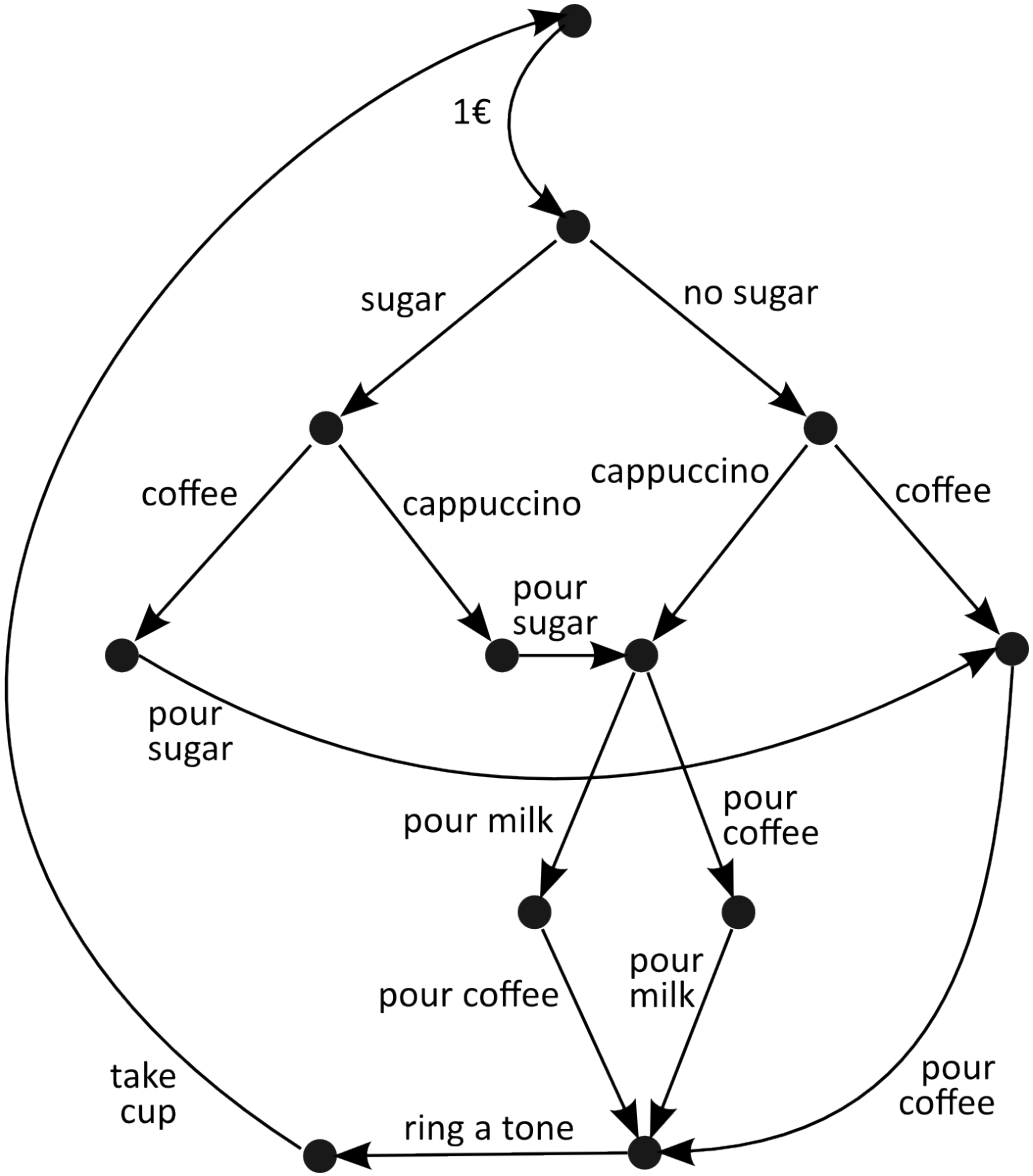
MTS however cannot model variability constraints regarding **alternative** features nor regarding the **requires** and **excludes** inter-feature relations

Our solution: add a set of variability constraints (hiding our logic) to the MTS to define which derivable products must be considered valid ones (we defined an algorithm to derive only—and possibly all—valid ones)

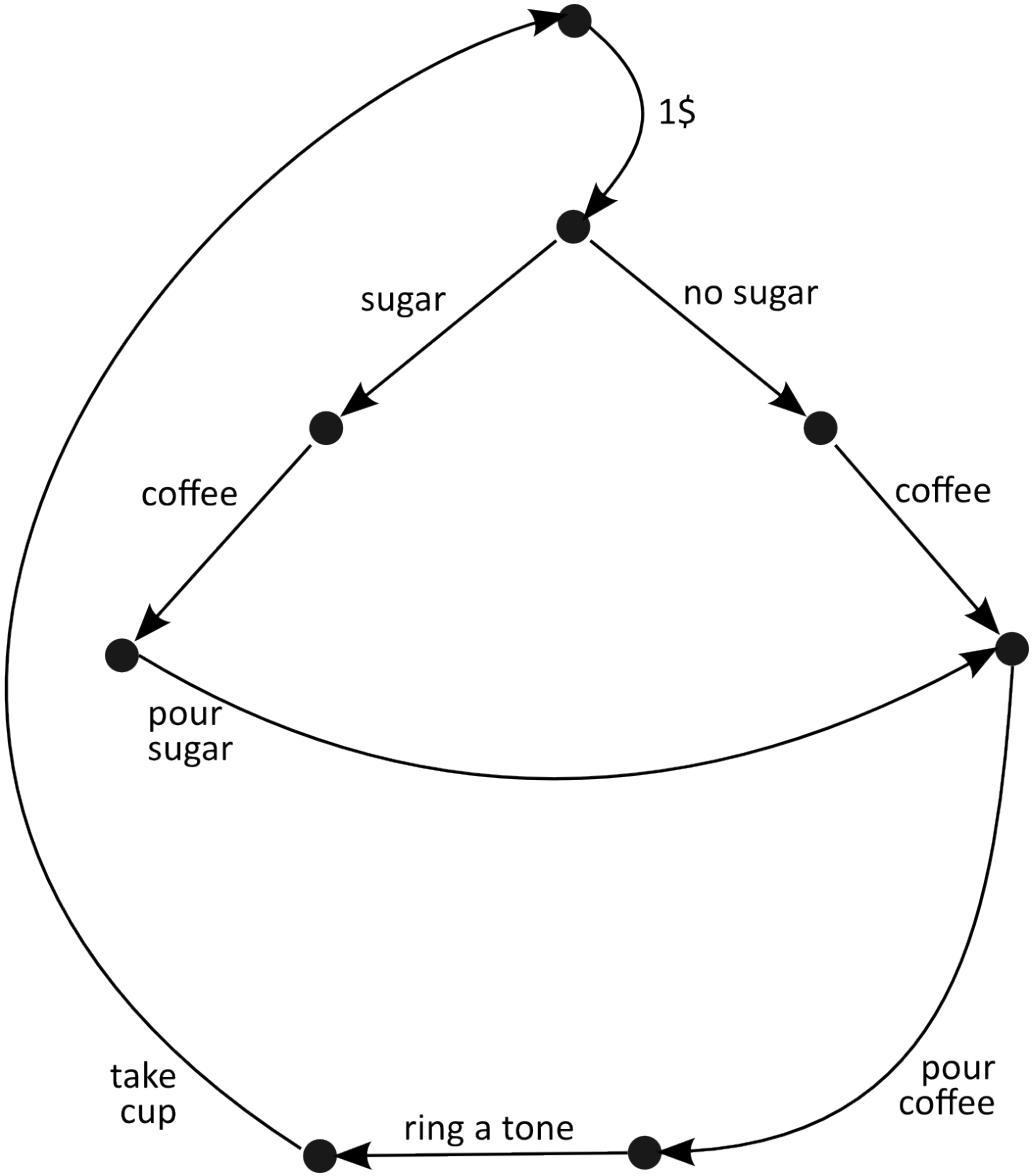
Coffee machine family: MTS



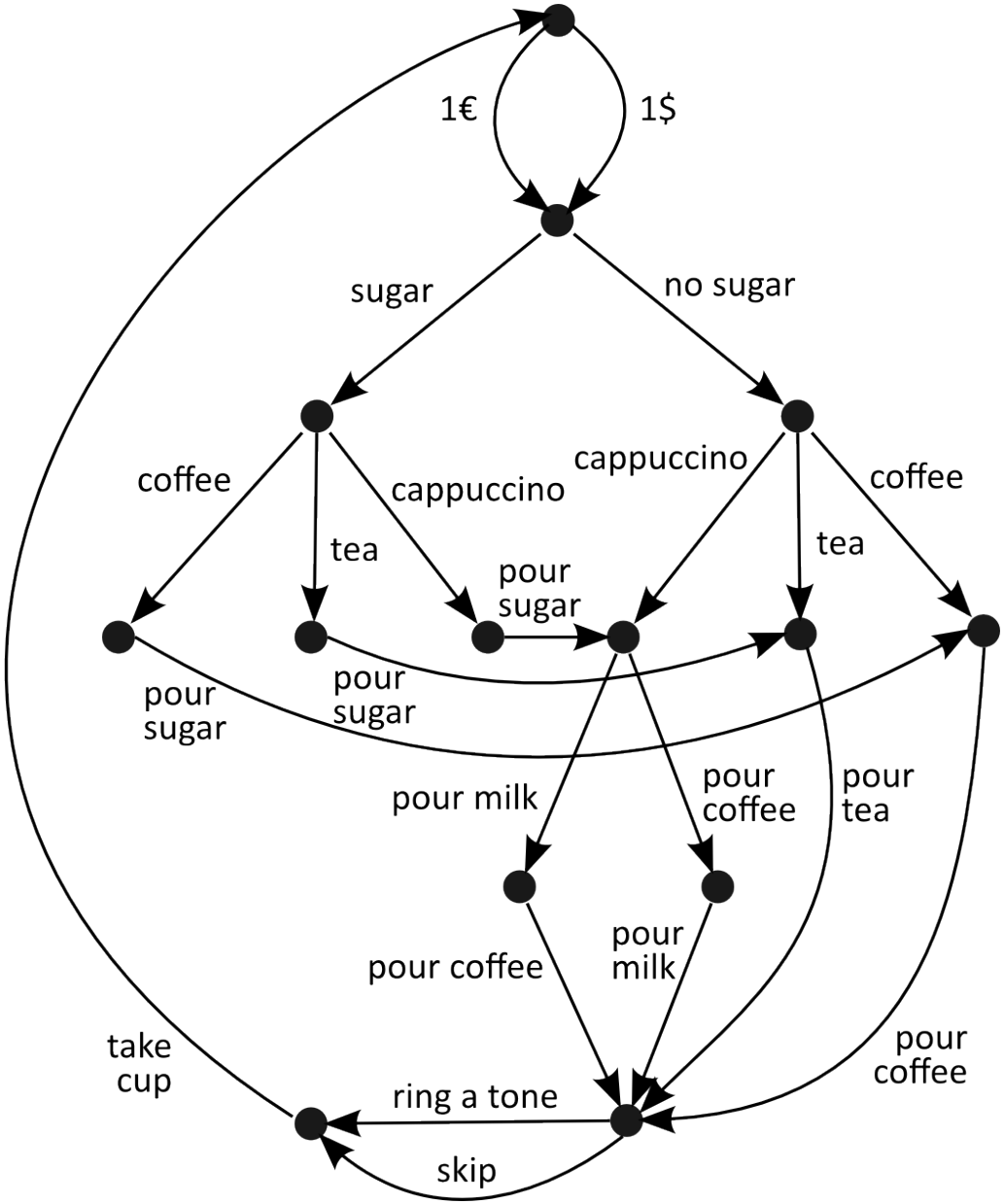
A valid European coffee machine ($\{m, s, o, b, c, €, p, r\}$)



A valid Canadian coffee machine ($\{m, s, o, b, c, \$, r\}$)



A correct but not a valid product LTS of MTS



VMC: Variability Model Checker

A tool for modeling and analysis of behavioral variability in SPLs

Given a textual encoding of an MTS and a set of variability constraints:

- interactively explore the MTS
- derive and explore (all) the family's valid products (LTSs)
- visualize the family/products graphically as MTS/LTSs
- verify branching-time temporal logic properties over family/products
- interactively explain why a product does (not) satisfy a property

Freely usable online: `http://fmtlab.isti.cnr.it/vmc/`

Verification engine and advanced explanation techniques are features of the highly optimized family of on-the-fly model checkers developed at our FMT lab over the last decades for verifying formulae in CTL-like action- and state-based branching-time temporal logics (e.g. FMC, UMC, CMC)

VMC: Variability Model Checker

A tool for modeling and analysis of behavioral variability in SPLs

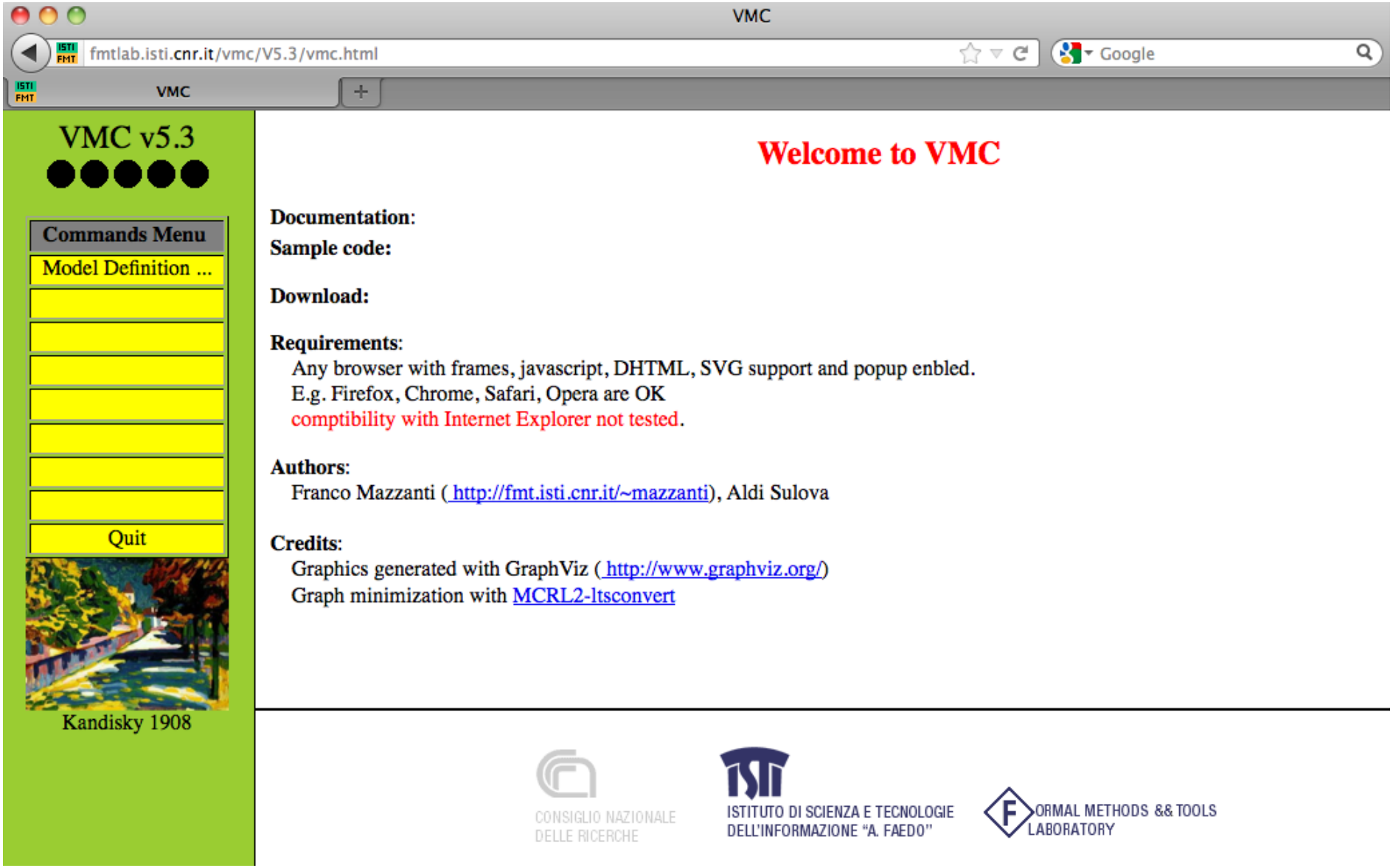
Given a textual encoding of an MTS and a set of variability constraints:

- interactively explore the MTS
- derive and explore (all) the family's valid products (LTSs)
- visualize the family/products graphically as MTS/LTSs
- verify branching-time temporal logic properties over family/products
- interactively explain why a product does (not) satisfy a property

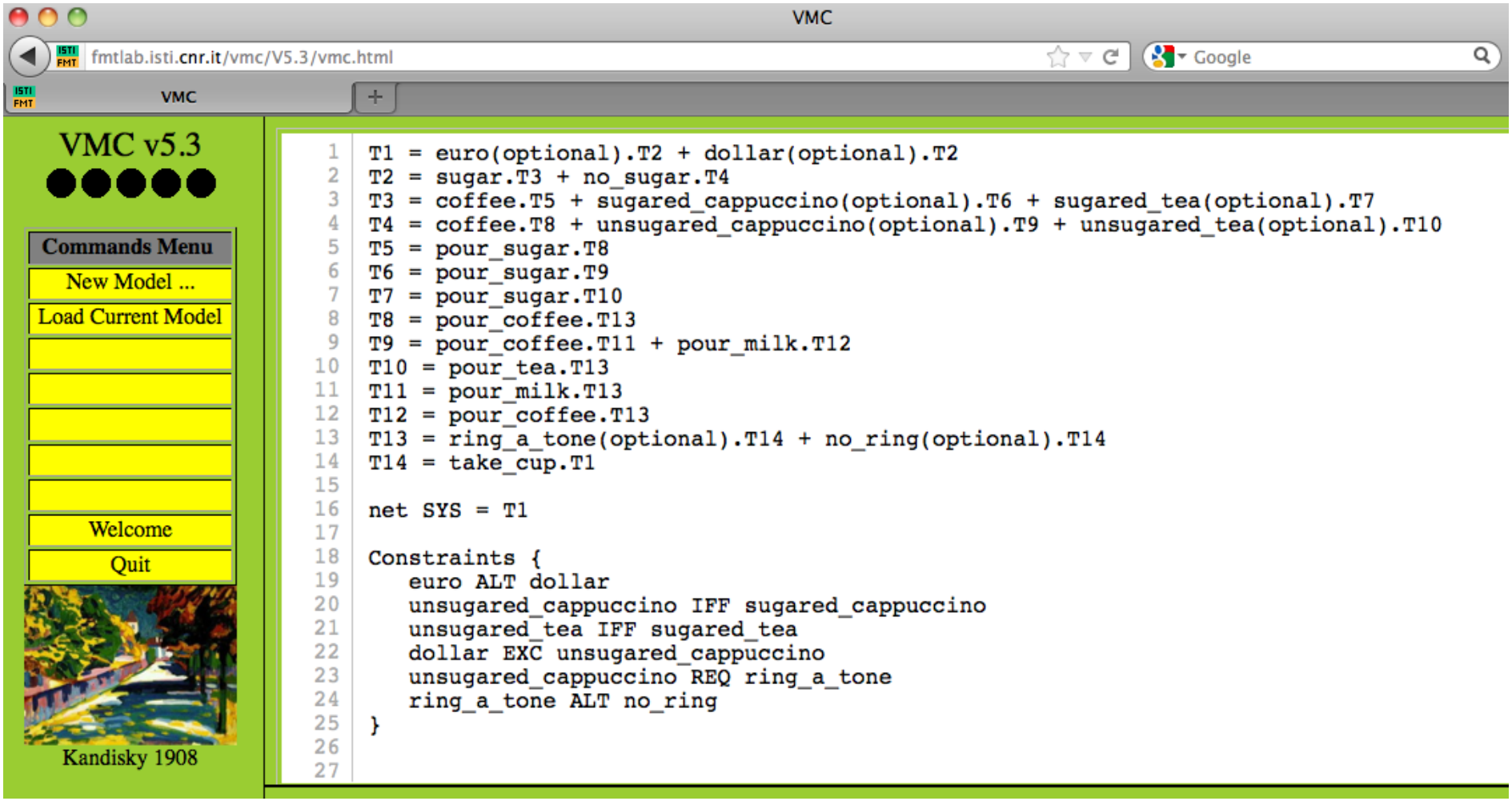
Freely usable online: `http://fmtlab.isti.cnr.it/vmc/`

Verification engine and advanced explanation techniques are features of the highly optimized family of on-the-fly model checkers developed at our FMT lab over the last decades for verifying formulae in CTL-like action- and state-based branching-time temporal logics (e.g. FMC, UMC, CMC)

VMC's web interface

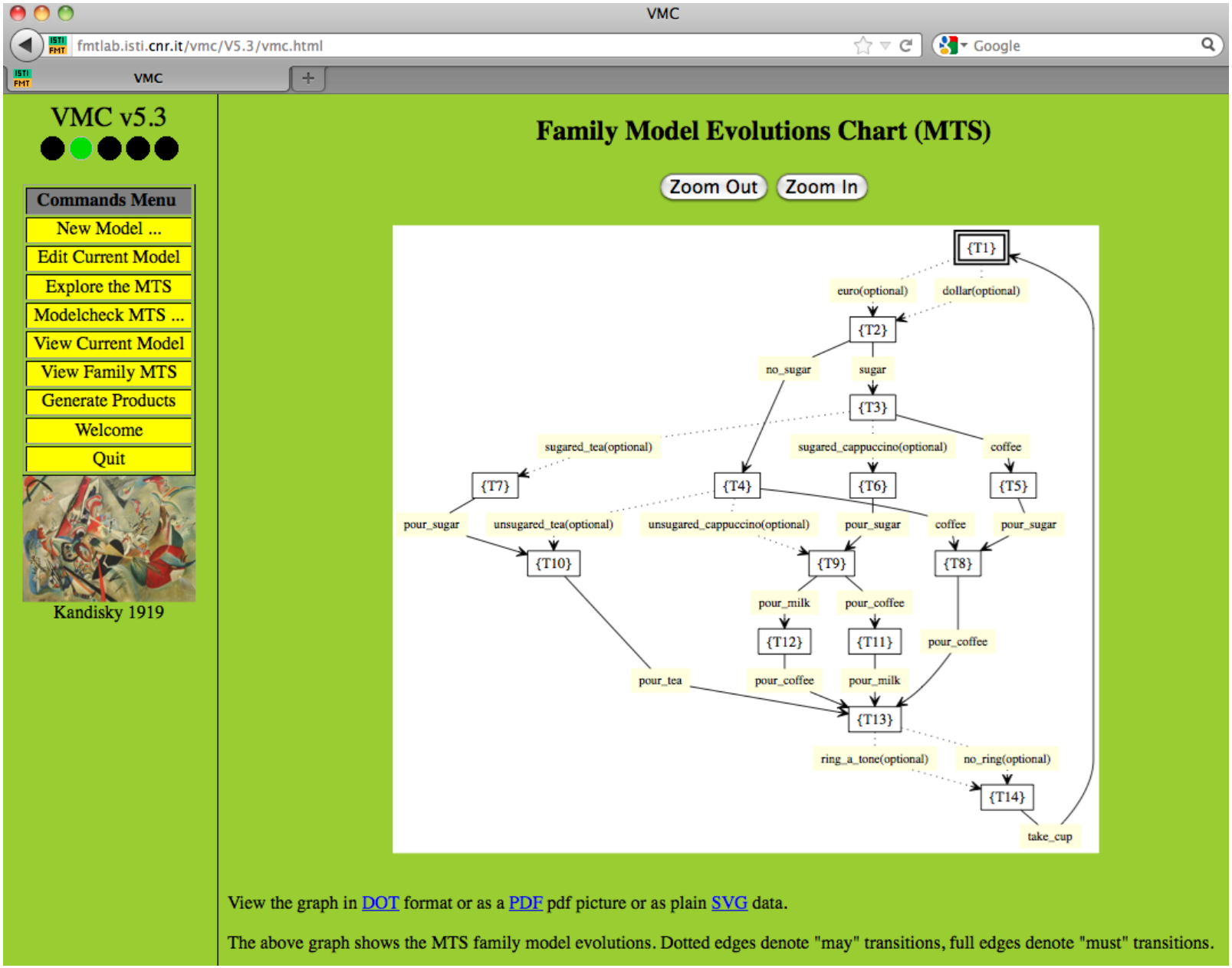


Family of coffee machines specified in VMC

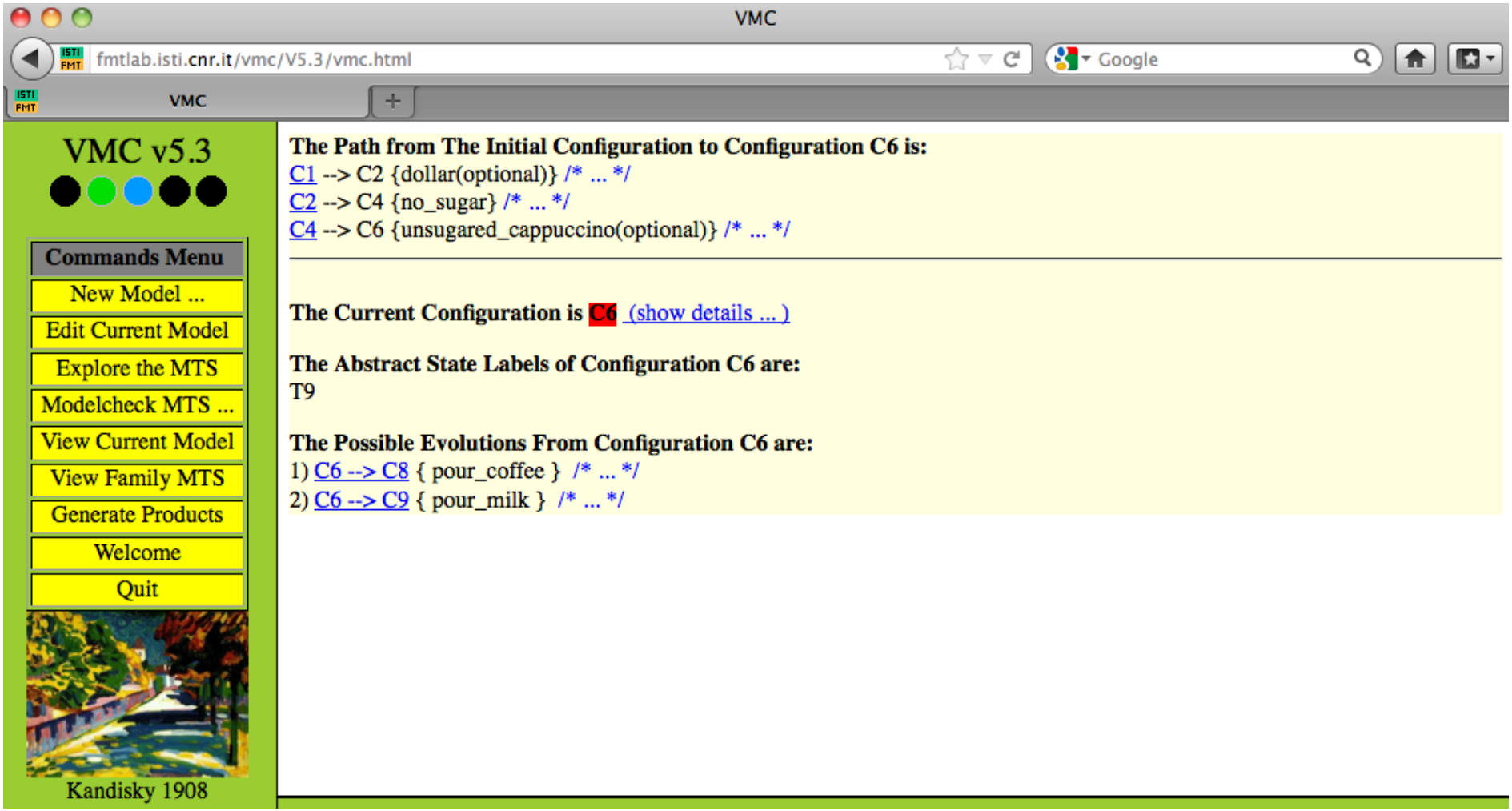


Permitted variability constraints ALTernative, EXCludes, REQuires, and IFF (shorthand for bilateral REQs) hide the logic formalization from user

Family/MTS of coffee machines visualized by VMC

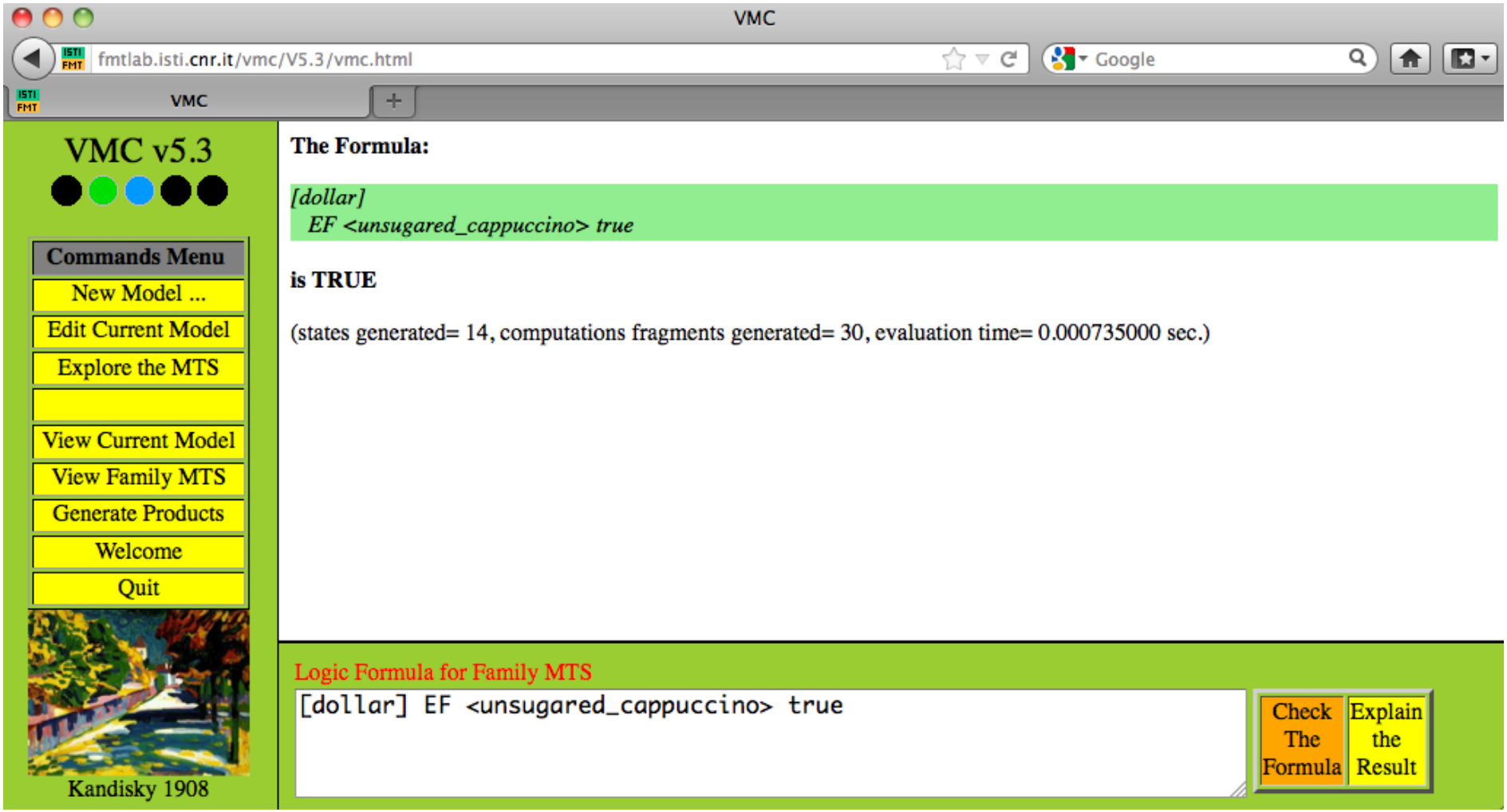


Family/MTS of coffee machines explored in VMC



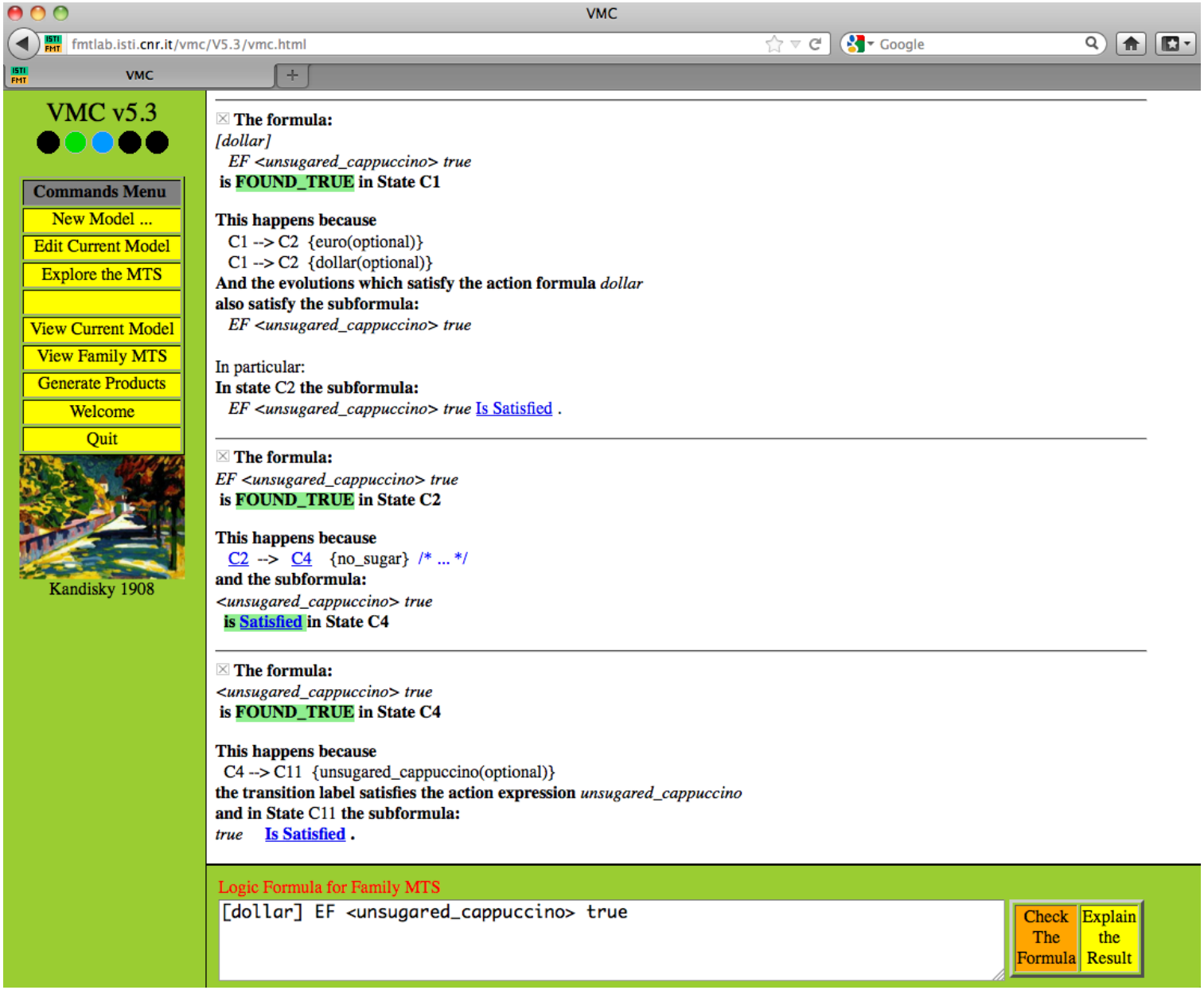
MTS model of coffee machine family actually permits a user to buy a cappuccino with a dollar, something which is forbidden for its products by the variability constraint `dollar EXC unsugared_cappuccino`

Outcome of a property verified over a family with VMC

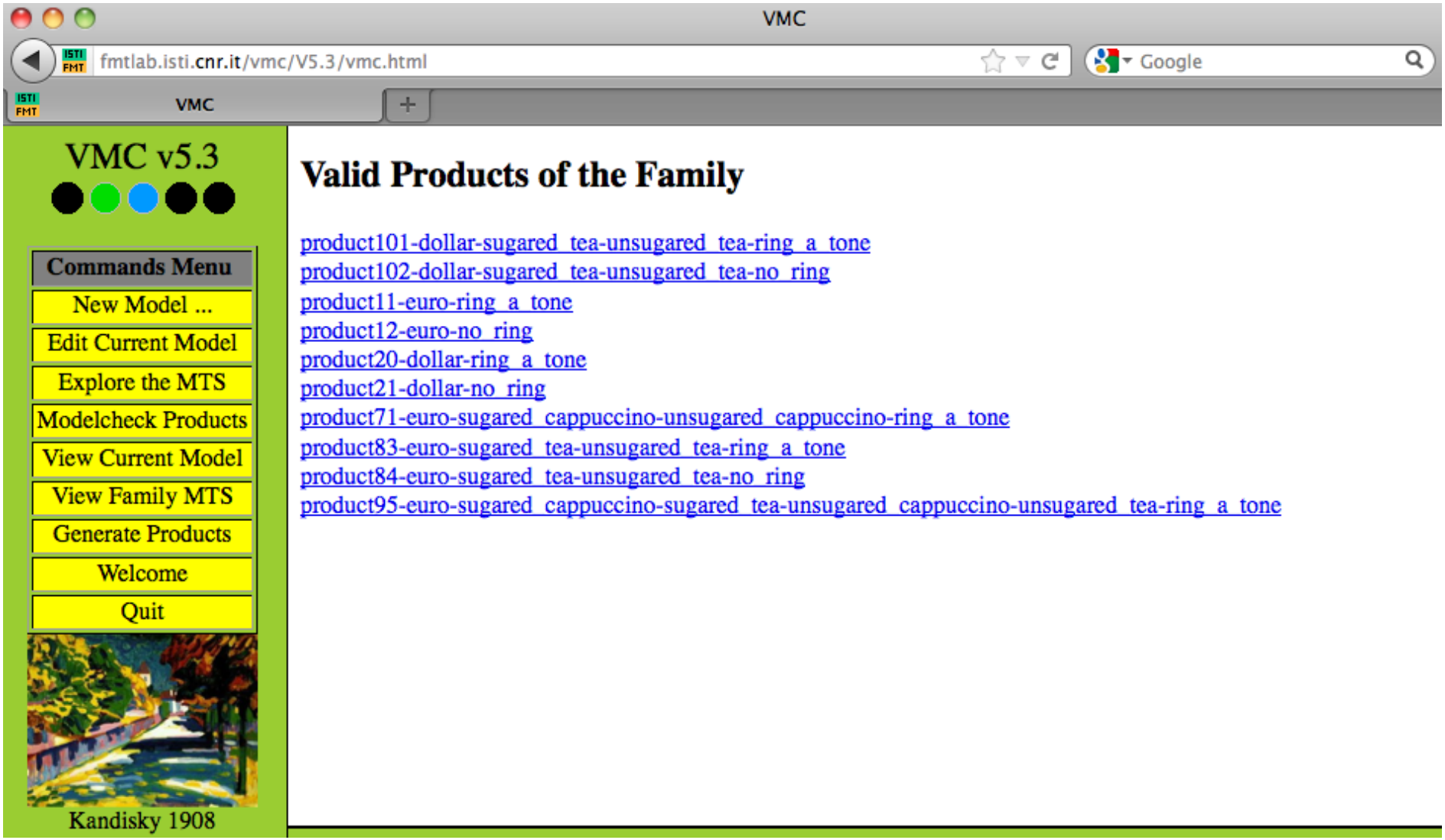


The formula expresses that every path through the MTS that starts with the insertion of a dollar, eventually leads to an unsugared cappuccino

Outcome of a property explained by VMC



Products of family of coffee machines derived by VMC



VMC indeed generates all 10 valid products/LTSs that are derivable from the family/MTS if the set of variability constraints is considered

Outcomes of a property verified over products with VMC

As required, no valid product (i.e. coffee machine) can deliver an (unsugared) cappuccino upon the insertion of a dollar!

Specification of one of the products derived by VMC

Clicking on a product, VMC opens a window with its textual encoding

Product/LTS on previous slide visualized by VMC

VMCP

fmtlab.isti.cnr.it/vmcp/V5.2/vmcp.html?remotemodel

VMC | VMCP

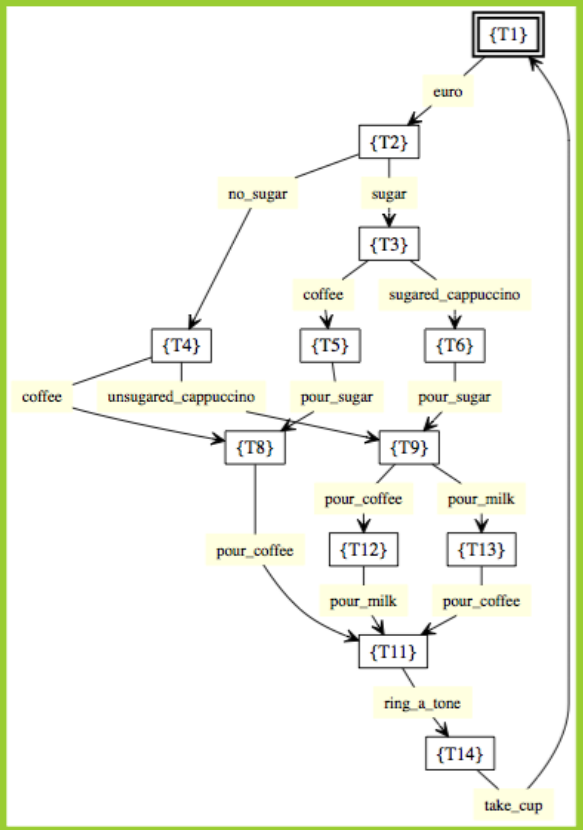
VMC v5.3
(on product)

Commands Menu
- New Model ...
- Edit Current Model
- Explore the LTS
- View Current Model
- View the LTS Graph
- Quit

Kandisky 1919

Product Model Evolutions Chart (LTS)

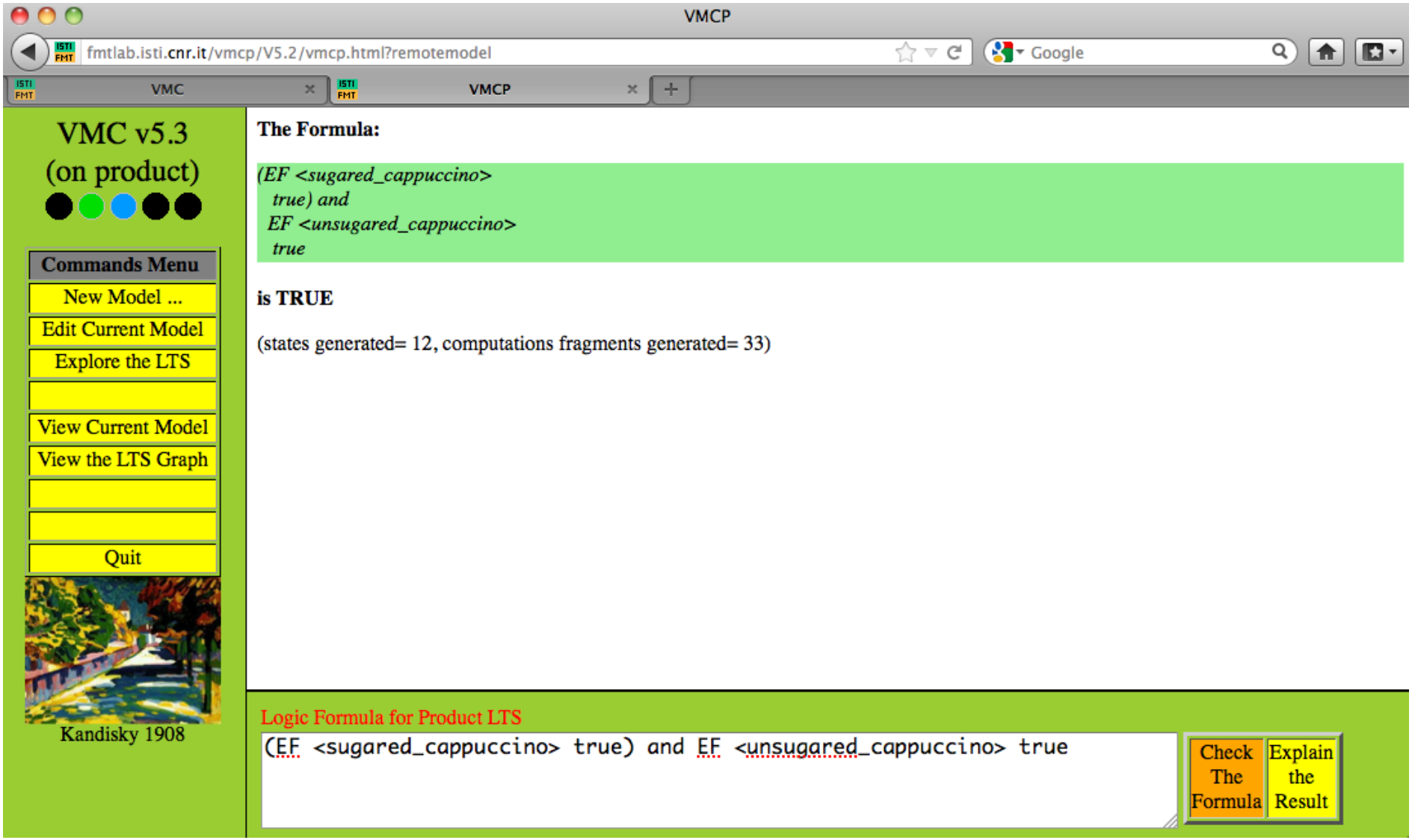
Zoom Out | Zoom In



View the graph in DOT format or as a PDF pdf picture or as plain SVG data.

The above graph shows the LTS product model evolutions, which by defintion contains only full edges.

Outcome of a property verified over a product with VMC



The formula expresses that in this particular LTS, there exists both a path to a sugared cappuccino and a path to an unsugared cappuccino

Outcome property on previous slide explained by VMC

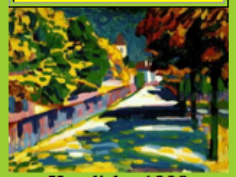
VMCP

fmtlab.isti.cnr.it/vmcp/V5.2/vmcp.html?remotemodel

VMC | VMCP

VMC v5.3
(on product)

Commands Menu
- New Model ...
- Edit Current Model
- Explore the LTS
- View Current Model
- View the LTS Graph
- Quit



Kandisky 1908

The formula:
(EF <sugared_cappuccino> true) and
EF <unsugared_cappuccino> true
is FOUND_TRUE in State C1

This happens because the subformula:
EF <sugared_cappuccino> true
is Satisfied in State C1

And because the subformula:
EF <unsugared_cappuccino> true
is Satisfied in State C1

The formula:
EF <sugared_cappuccino> true
is FOUND_TRUE in State C1

This happens because
C1 --> C2 {euro} /* ... */
C2 --> C3 {sugar} /* ... */
and the subformula:
<sugared_cappuccino> true
is Satisfied in State C3

The formula:
EF <unsugared_cappuccino> true
is FOUND_TRUE in State C1

This happens because
C1 --> C2 {euro} /* ... */
C2 --> C4 {no_sugar} /* ... */
and the subformula:
<unsugared_cappuccino> true
is Satisfied in State C4

Logic Formula for Product LTS

```
(EF <sugared_cappuccino> true) and EF <unsugared_cappuccino> true
```

Check The Formula | Explain the Result

Discussion and future work

VMC can also be used to specify and analyze only specific subsets of a product family's valid products by applying restrictions via constraints

Add constraint `coffee EXC dollar` to the family specification

Only European coffee machines will be derived as valid products, which can then be analyzed both as a subset and individually

Future work required before possible application in industry

- A high-level language hiding all semantic details
  (investigate the relation between features and actions)
- A predefined taxonomy for exemplary logical properties
  (e.g. the specification patterns repository for LTL, (A)CTL, etc.)
- Scale to large, industrial-size product families
  (with many variation points and many features)

Discussion and future work

VMC can also be used to specify and analyze only specific subsets of a product family's valid products by applying restrictions via constraints

Add constraint `coffee EXC dollar` to the family specification

Only European coffee machines will be derived as valid products, which can then be analyzed both as a subset and individually

Future work required before possible application in industry

- A high-level language hiding all semantic details
  (investigate the relation between features and actions)
- A predefined taxonomy for exemplary logical properties
  (e.g. the specification patterns repository for LTL, (A)CTL, etc.)
- Scale to large, industrial-size product families
  (with many variation points and many features)

Discussion and future work

VMC can also be used to specify and analyze only specific subsets of a product family's valid products by applying restrictions via constraints

Add constraint `coffee EXC dollar` to the family specification

Only European coffee machines will be derived as valid products, which can then be analyzed both as a subset and individually

Future work required before possible application in industry

- A high-level language hiding all semantic details
  (investigate the relation between features and actions)
- A predefined taxonomy for exemplary logical properties
  (e.g. the specification patterns repository for LTL, (A)CTL, etc.)
- Scale to large, industrial-size product families
  (with many variation points and many features)

Publicity: consider submitting to VaMoS 2013 in Pisa

Variability Modelling of Software-Intensive Systems (VaMoS'13)
7th International Workshop

Pisa, Italy, January 23–25, 2013

⇒ `http://www.vamos-workshop.net`

Submission deadline: November 4, 2012

PC chairs:

- Philippe Collet (Université Nice Sophia Antipolis, France)
- Klaus Schmid (Stiftung Universität Hildesheim, Germany)

Organized by our FMT lab at the CNR research area in Pisa