Project

Formal Methods

Course Instructor
Ms. Nigar Azhar Butt

Submitted By

Aiyza Junaid
21i-1145

Haniya Tariq
21i-1169

Anum Sajid
21i-1233

Section
SE-P

Date
Wednesday, September 4, 2024

Fall 2024

Department of Software Engineering

FAST – National University of Computer & Emerging Sciences
Islamabad Campus

Table of Contents

1. Introduction

The **Feature Model Analysis and Visualization Tool** is designed to help analyze feature models. It provides functionality to:

- Parse feature model XML files.
- Visualize the hierarchical structure of features.
- Validate feature configurations.
- Generate propositional logic rules and evaluate constraints.

---

2. System Requirements

Backend:

- **Python 3.7+**
- Flask
- pysat library

Frontend:

- **Streamlit**

Additional Libraries:

- xml.etree.ElementTree
- requests

---

3. Application Workflow

The tool has two primary components:

1. **Backend Service**: Performs feature model analysis, hosted on Flask.
2. **Frontend Interface**: Provides an intuitive visualization and user interaction through Streamlit.

---

4. Key Functionalities

1. **Parse XML Feature Models**: Automatically extract features and constraints from uploaded XML files.
2. **Visualize Feature Hierarchies**: Render the feature tree interactively with options for feature selection.
3. **Validate Configurations**: Check whether selected features satisfy mandatory rules and constraints.
4. **Generate Propositional Logic**: Derive logical rules for understanding feature relationships.
5. **Find Minimal Working Products**: Identify the smallest valid feature set.

---

5. Step-by-Step Walkthrough

Step 1: Upload XML File

1. Launch the application via Streamlit.
2. Use the **Upload XML File** section to upload a valid feature model XML file.
   - Example file structure:

```xml
<featureModel>
    <feature name="Application"> <!-- Root is mandatory even if it is not specified -->
        <feature name="Catalog" mandatory="true">
            <feature name="Filtered" mandatory="true">
                <group type="xor">
                    <feature name="ByDiscount"/>
                    <feature name="ByWeather"/>
                    <feature name="ByLocation"/>
                </group>
            </feature>
        </feature>
        <feature name="Notification"> <!-- absence of mandatory means mandatory is false -->
            <group type="xor">
                <feature name="SMS"/>
                <feature name="Call"/>
            </group>
        </feature>
        <feature name="Location" mandatory="false"> <!-- mandatory = false means optional feature -->
            <group type="or">
                <feature name="WiFi"/>
                <feature name="GPS"/>
            </group>
        </feature>
        <feature name="Payment" mandatory="true">
            <group type="or">
                <feature name="CreditCard"/>
                <feature name="Discount"/>
            </group>
        </feature>
    </feature>
    <constraints>
        <constraint>
            <englishStatement>The Location feature is required to filter the catalog by location.</englishStatement>
        </constraint>
        <!-- example of boolean
        <constraint>
        <booleanExpression>Payment implies Location</booleanExpression>
        </constraint>
        -->
    </constraints>
</featureModel>
```

3. Once uploaded, the file is parsed, and the feature tree is displayed.

Step 2: Feature Selection

1. **Mandatory Features**:
   - These are pre-selected and cannot be unchecked.
   - Indicated with a pin icon.
2. **Optional Features**:
   - Select/deselect optional features by checking the corresponding box.
   - Indicated with an attach icon.
3. Groups like XOR and OR are marked, indicating their selection rules.

Step 3: Analyze Features

1. After selecting the features, click the **Analyze** button.
2. The application sends the selected features and XML data to the backend.
3. Wait for the results to be processed and displayed.

6. Understanding the Results

Propositional Logic

- Displays logical rules derived from the feature hierarchy and constraints.
- Examples:
  - RootFeature → ChildFeature1 (Mandatory rule)
  - Option1 ∨ Option2 (OR group)

Constraints

- Lists constraints specified in the XML file.
- Example: RootFeature → ChildFeature1

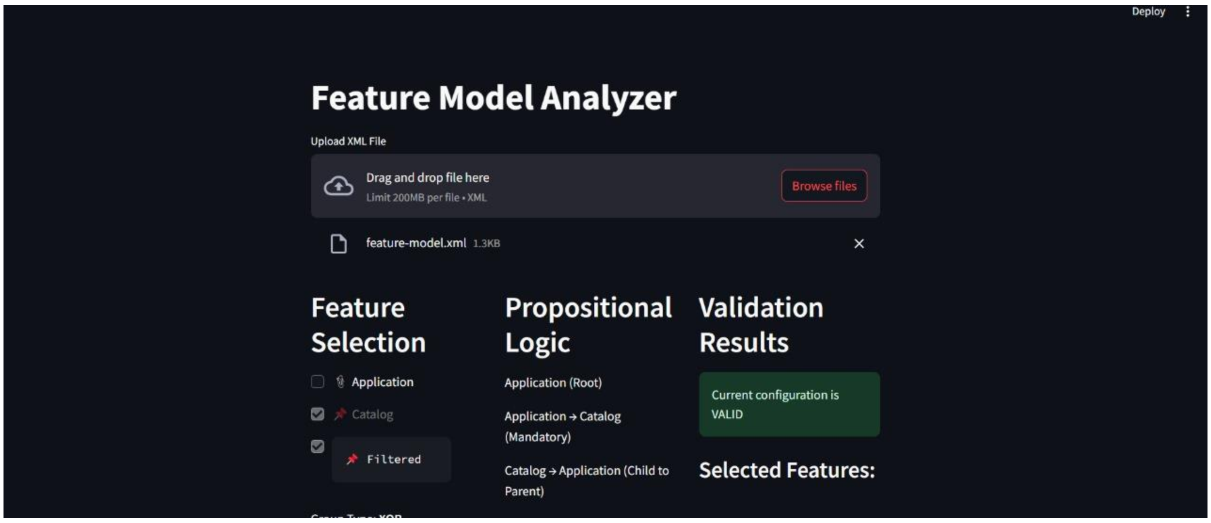
Validation Results

- **Valid Configuration**: Indicates that the selected features meet all rules and constraints.
- **Invalid Configuration**: Lists mandatory or conflicting rules that were violated.

Minimum Working Products

- Displays the smallest feature set that satisfies all constraints and rules.

7. Demo



Feature Model Analyzer

Upload XML File

Drag and drop file here
Limit 200MB per file • XML

feature-model.xml 1.3KB

Feature Selection

- [ ] Application
- [✓] Catalog
- [✓] Filtered

Propositional Logic

Application (Root)

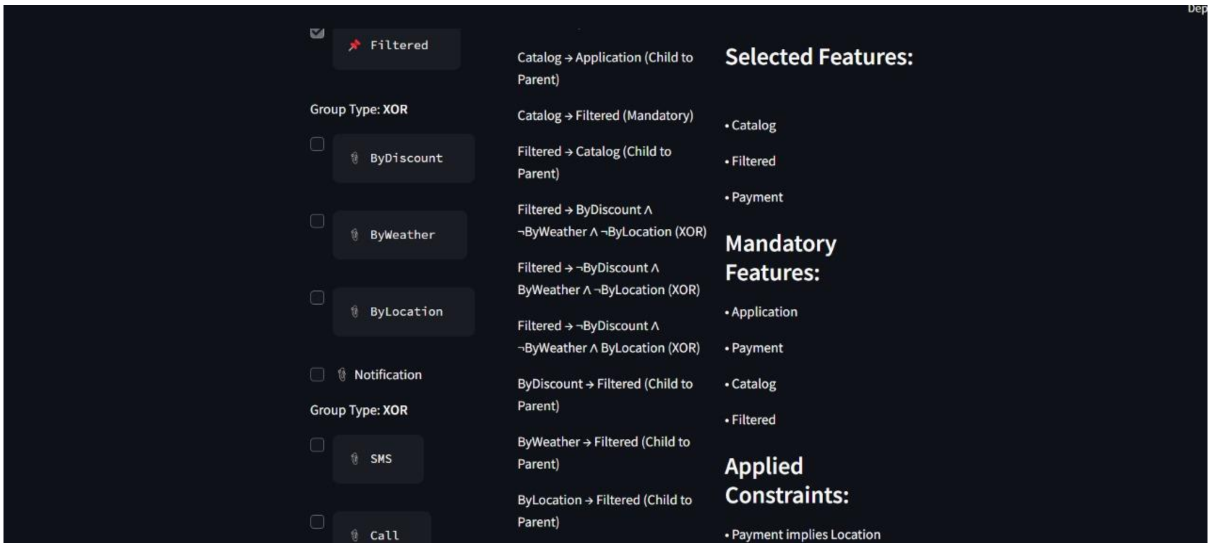
Application → Catalog (Mandatory)

Catalog → Application (Child to Parent)

Validation Results

Current configuration is VALID

Selected Features:



Dep

☑ Filtered

Group Type: **XOR**

☐ ByDiscount

☐ ByWeather

☐ ByLocation

☐ Notification

Group Type: **XOR**

☐ SMS

☐ Call

Catalog → Application (Child to Parent)

Catalog → Filtered (Mandatory)

Filtered → Catalog (Child to Parent)

Filtered → ByDiscount ∧ ¬ByWeather ∧ ¬ByLocation (XOR)

Filtered → ¬ByDiscount ∧ ByWeather ∧ ¬ByLocation (XOR)

Filtered → ¬ByDiscount ∧ ¬ByWeather ∧ ByLocation (XOR)

ByDiscount → Filtered (Child to Parent)

ByWeather → Filtered (Child to Parent)

ByLocation → Filtered (Child to Parent)

Selected Features:

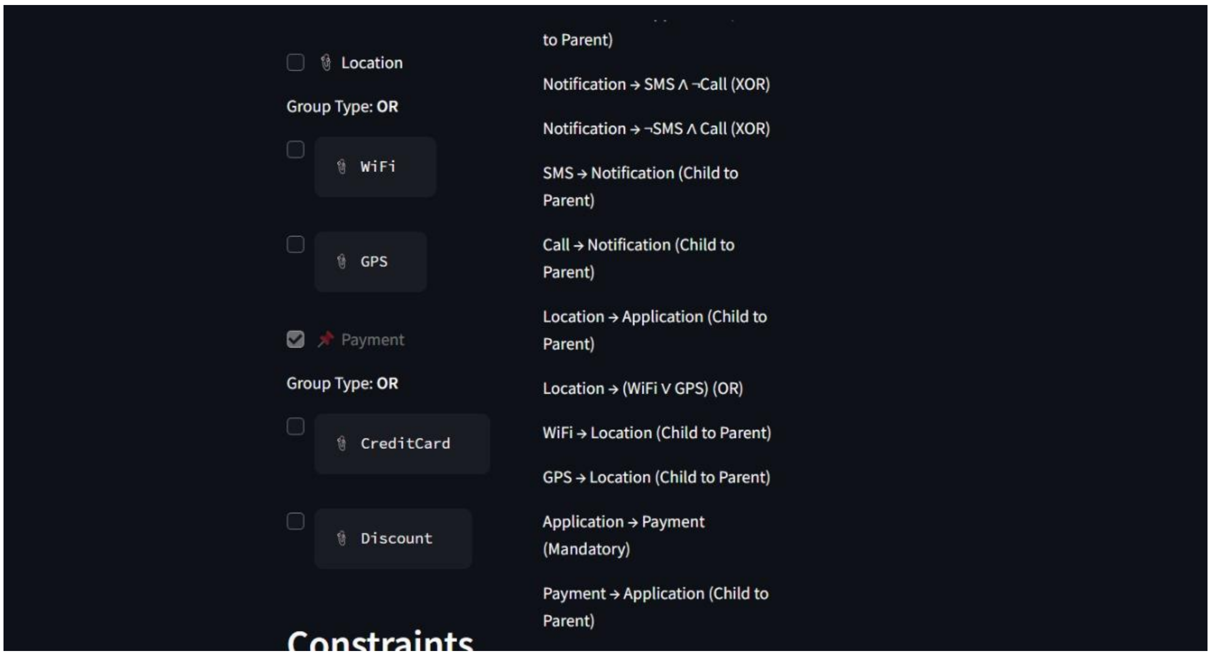
- Catalog
- Filtered
- Payment

Mandatory Features:

- Application
- Payment
- Catalog
- Filtered

Applied Constraints:

- Payment implies Location



to Parent)

Notification → SMS ∧ ¬Call (XOR)

Notification → ¬SMS ∧ Call (XOR)

SMS → Notification (Child to Parent)

Call → Notification (Child to Parent)

Location → Application (Child to Parent)

Location → (WiFi ∨ GPS) (OR)

WiFi → Location (Child to Parent)

GPS → Location (Child to Parent)

Application → Payment (Mandatory)

Payment → Application (Child to Parent)

☐ Location

Group Type: **OR**

☐ WiFi

☐ GPS

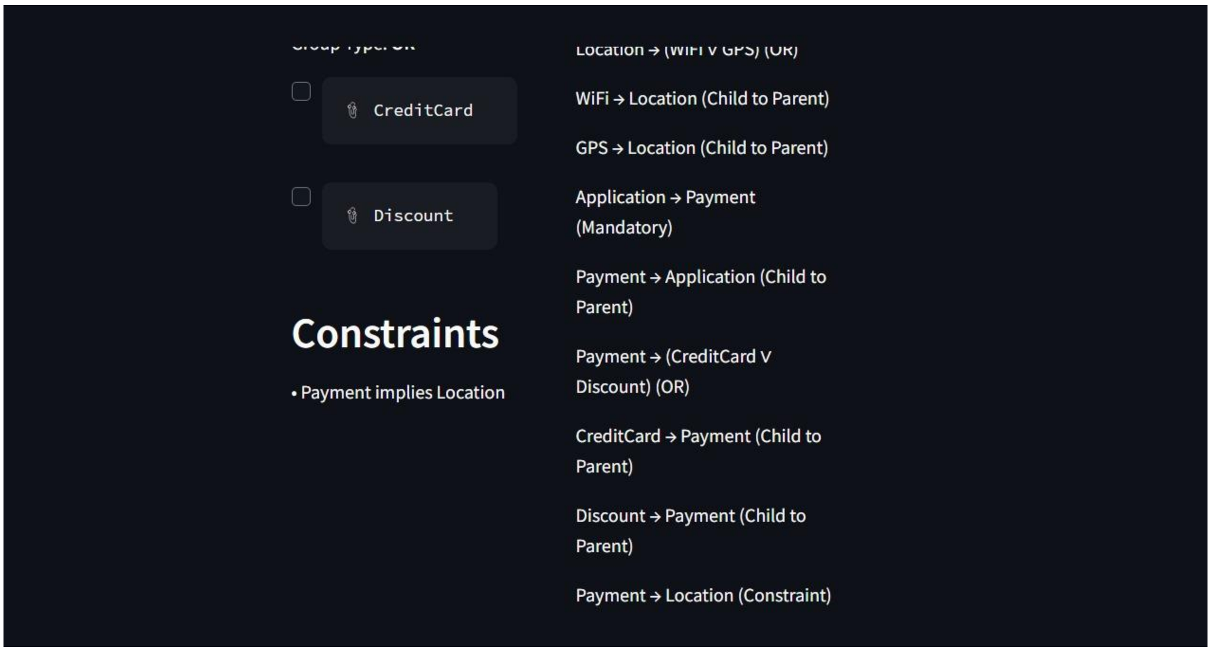
☑ Payment

Group Type: **OR**

☐ CreditCard

☐ Discount

Constraints



Group Type: OR

☐ CreditCard

☐ Discount

Constraints

• Payment implies Location

Location → (WiFi V GPS) (OR)

WiFi → Location (Child to Parent)

GPS → Location (Child to Parent)

Application → Payment (Mandatory)

Payment → Application (Child to Parent)

Payment → (CreditCard V Discount) (OR)

CreditCard → Payment (Child to Parent)

Discount → Payment (Child to Parent)

Payment → Location (Constraint)

8. Troubleshooting

Common Issues:

1. **Invalid XML File**:
   - Ensure your file follows the required structure.
2. **Backend Connection Errors**:
   - Confirm the Flask server is running on http://localhost:5000.