Abductive Reasoning and Automated Analysis of Feature Models: How are they connected?*

Pablo Trinidad, Antonio Ruiz–Cortés
Dpto. Lenguajes y Sistemas Informáticos
University of Seville
{ptrinidad,aruiz} at us.es

Abstract

In the automated analysis feature models (AAFM), many operations have been defined to extract relevant information to be used on decision making. Most of the proposals rely on logics to give solution to different operations. This extraction of knowledge using logics is known as deductive reasoning. One of the most useful operations are explanations that provide the reasons why some other operations find no solution. However, explanations does not use deductive but abductive reasoning, a kind of reasoning that allows to obtain conjectures why things happen. As a first contribution we differentiate between deductive and abductive reasoning and show how this difference affect to AAFM. Secondly, we broaden the concept of explanations relying on abductive reasoning, applying them even when we obtain a positive response from other operations. Lastly, we propose a catalog of operations that use abduction to provide useful information.

1. Introduction

The automated analysis feature models (AAFM) intends to extract relevant information to assist on decision making and even to produce design models or code. The general process that most of the works propose to deal with automated analysis is transforming a Feature Model(FM) into a logic paradigm and solving declaratively the problem. We have noticed that most of the proposed operations use deductive reasoning techniques to extract such an information. The way deductive reasoning works is obtaining objective conclusions from its knowledge base (KB) making explicit an implicit information.

But in some situations, it may be interesting not only obtaining conclusions but knowing the reasons why that conclusion is inferred. For example, if we find an error in a FM such as a dead feature we must be interested in the relationships that make this error appearing. So we can use this information to assist on error repairing. In case we are searching for the cheapest product to be produced in a family and we obtain a specific product, we may be searching for the relationships and criteria that have been taken into account. This transverse operation is commonly known in FM analysis community as *explanation* and may be used in conjunction with any deductive operation.

These two examples, remarks the automated analysis as a two-step process, where an information is extracted from a FM firstly by means of deductive reasoning, and just in case we are interested in obtaining further information, we may ask for the reasons why we have obtained such an information using abductive reasoning. As a first contribution of this paper we remark this difference, distinguishing between two kinds of operations: *deductive operations*, that use deductive reasoning to reach for a result; and explanatory or *abductive operations*, which use abductive reasoning to explain a result obtained from a deductive operation (see Figure 1). As a consequence, we have observed that most of the proposed operations in automated analysis are deductive operations, and abductive operations have only been proposed to solve particular problems such as obtaining explanations for void FMs and dead features. Therefore, and as a second contribution, we propose a catalog of abductive operations that broadens their field of action to be applied to the results of any deductive operation.

One of the main contributions in [2] is proposing a general transformation from a FM into many logic paradigm or solver such as CSPs, SAT problems or BDDs, by means of a formal description of the problem in the so called FAMA Theoretical Framework. However, his proposal was centered in deductive reasoning and explanations were proposed as an operation that did not fit into his deductive framework so solving them was considered to be an open

*This work has been partially supported by the European Commission (FEDER) and Spanish Government under CICYT project Web-Factories (TIN2006-00472) and by the Andalusian Government under project ISABEL (TIC-2533).

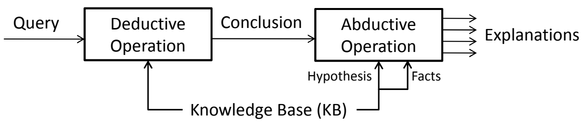


Figure 1. The Link between Deductive and Abductive Reasoning

issue. Now we know that explanations may never fit into his deductive framework as it is an abductive operation. However, we envision that we may follow the same structure than Benavides' FAMA Theoretical Framework for abductive operations so the problem may be represented in a theoretical level so several solvers and logic paradigms may be used to solve them. Therefore, as a last contribution, we envision how some of the current proposals in abductive reasoning may fit into such a framework and which are the solvers, techniques or algorithms that can be used to deal with abductive operations.

This paper is structured as follows: In Section 2 we briefly present a study of the works in the automated analysis of FM from the point of view of deductive and abductive reasoning. In Section 3, we introduce the concept of abductive reasoning more in depth, pointing out its relationship with diagnosis problems. The catalog of abductive operations is presented in Section 4. We envision the future works and research lines, exposing some conclusions in Section 5.

2 Background

2.1 Analysis of Feature Models

The automated analysis of FMs intends to extract relevant information from FMs to assist decision making during SPL development. To obtain such an information, many authors have proposed different operations for products counting, filtering, searching and error detecting that are summarized in a survey in [4]. Most of the proposals rely on declarative techniques and logics to extract information such as Constraint Satisfaction Problems (CSP) [3], SAT solvers [7] and Binary Decision Diagrams (BDD)[5].

In the works where logics are used to give a response to those operations, they use a common way of reasoning called deduction. Informally speaking, Deduction makes explicit an implicit information in a theory. It means that the only information that may be extracted from a model is the one that is modeled, and what we are doing when reasoning deductively about a FM is making explicit a hard-to-see information. For example, if we select feature A in the FM in Figure 2, deductive reasoning may reach the conclusion that feature C may not be selected. If we select features A and C deduction is only able to determine that there is no possible configuration containing both features at the same time. If we want to explain the reason why A and C are mutually exclusive, deductive reasoning is not the right choice.

2.2 Explanations in Feature Models

The need of explanations were firstly detected by Kang *et al.*[6] to determine the reasons why a FM is void. In this work, Prolog was proposed to model and explaining void FMs if it were the case. Batory proposed in [1] using Logic Truth Maintenance Systems (LTMS) to explain why a configuration is not valid. Sun *et al.* [10] use Alloy Analyzer, a model checking tool based on first-order logic, to detect the sources of void FMs. Wang *et al.*[14] propose using description logic and RACER tool to deal with dead features and void FMs. Trinidad *et al.* describe in [13, 11] the errors explanation problem in terms of theory of diagnosis[9], dealing with different kinds of error. They propose a framework where different implementations were accepted and giving details about using constraint satisfaction optimization problems (CSOP) to deal with them. White *et al.*[15] proposed using CSOP to deal with invalid configurations.

Notice that the techniques proposed to search for explanations are different from those proposed to deal with deductive reasoning.

Moreover, most of the proposals that deal with explanations focus on error analysis. We already presented in [11] a framework to deal with errors relying in diagnostic reasoning which is a particular application of abductive reasoning as we will remark in next Section.

2.3 Catalog of Deductive Operations

There are two main works [4, 2] that have summarized the state of the art in the automated analysis of FMs. Both of them present an exhaustive survey of the operations that have been proposed in the most relevant works.

- Determining if a product, feature model or configuration is valid.
- Counting and obtaining all the products.
- Calculating a feature commonality and variability and determining the core and variant features.
- Filtering and searching for optimal products.
- Dead and false-optional features and wrong-cardinalities detection.
- Explanations and error correction.
- Model transformations such as simplification and merging.

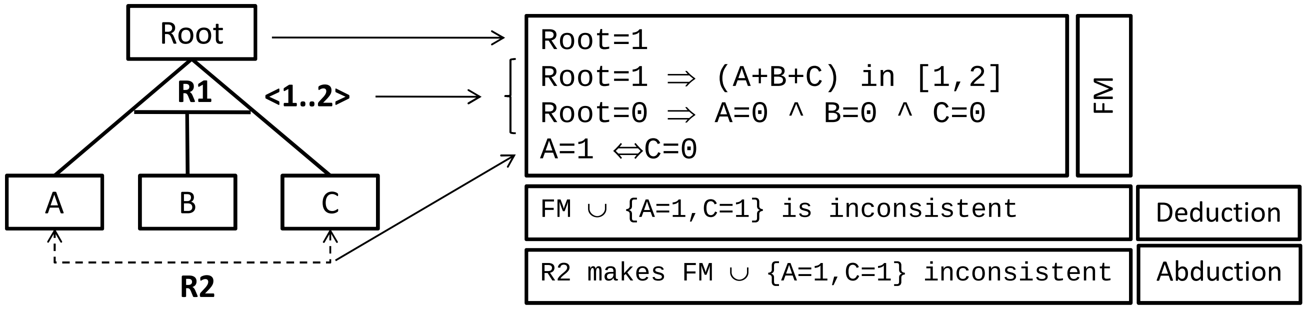

Root
R1
<1..2>
A
B
C
R2
Root=1
Root=1 ⇒ (A+B+C) in [1,2]
Root=0 ⇒ A=0 ^ B=0 ^ C=0
A=1 ⇔C=0
FM
FM ∪ {A=1,C=1} is inconsistent
Deduction
R2 makes FM ∪ {A=1,C=1} inconsistent
Abduction

Figure 2. Difference between deductive and abductive reasoning in FM analysis

A more detailed list of deductive operations may be seen in Table 1. All the above operations are deductive ones but explanations and error correction which are abductive operations. Properly speaking, model transformations are not analysis operations as they change the FM so they will be out of our scope. Next Sections we analyse the structure of abductive reasoning and refine the explanation operation to provide into a wider set of abductive operations.

3. Abductive Reasoning in a Nutshell

Most of the applications that use logics commonly use deductive reasoning or deduction. In Deductive reasoning we have a conception of our relevant world that is synthesized within a *Knowledge Base*(KB). A KB is composed by a set of facts that are accepted to be true. For example, a FM will be the KB in automated analysis. The objective of deduction is concluding a set of consequences from a KB.

In many contexts, the available information is incomplete or imprecise, normally due to the inability or difficulty of explicitly capturing all the knowledge in a KB. In classical logic, a proposition may be true or false. Anything that is not known or may be inferred is considered to be false in what is called the Closed World Assumption (CWA)[8]. However, when incomplete knowledge appears, we also consider a third state where a proposition is not known to be true or false. Here is where default rules or hypotheses appear. A hypothesis may be considered to be true whenever we have no clue that it is false. However, it makes that a conclusion that we infer from our KB based on hypotheses must be invalidated when new knowledge contradicting the hypothesis appears.

So we need a framework to represent an incomplete knowledge, distinguishing between:

- *Facts* (F): the knowledge that we certainly know to be true. It is a set $\{f_1, \cdots, f_n\}$ of formulas that must be consistent.
- *Default Rules or Hypotheses* (H): A set $\{h_1, \cdots, h_m\}$ of formulas which subsets can be assumed to be true if they are consistent together with the set of facts.

With this structure, for a set of facts and a set of hypothesis, we may have different possible *scenarios* S each of them taking into account a different and valid subset of hypothesis ($S \subseteq H$) consistent with the facts F.

A way to exploit this framework is called *abductive reasoning* or simply *abduction*. The objective of abduction is searching for the scenarios that may explain an observed situation or behaviour in the world. An observed behaviour or *observation* (*obs*) may be for example a measurement in a physical system or a conclusion obtained using deductive reasoning, and is described as a set of formulas. Rigorously speaking, an scenario S is an *explanation* to an observation *obs* iff

$$\begin{aligned} F \cup S &\models \mathit{obs} \\ F &\not\models \neg S \end{aligned}$$

3.1 Minimalistic Reasoning

From the above definition, we may obtain more than one explanation to an observation so abduction is a non-deterministic problem. In most of the cases, we need a criterion to choose the most suitable explanation and minimalistic reasoning may help on this issue.

Minimalistic reasoning relies on the principle that we normally we are not interested in all the explanations but in the best explanation. To determine the best explanation, we may apply different criteria, but the most typical one is taking the succinctest explanation in what is commonly known as the Occam's razor principle or parsimony law.

Here is where the concept of *minimal explanation* implements the parsimony law. An explanation E is minimal iff for no subset $E' \subset E$, $E\prime$ is an explanation. Therefore, in a problem we will obtain two explanations for an observation $\{h_1, h_2\}$ and $\{h_3\}$ if neither $\{h_1\}$ nor $\{h_2\}$ are able to explain the observation. It means that $\{h_1, h_2, h_3\}$ may be an explanation but it is removed for the sake of simplicity. A similar but not equivalent criterion to be considered will be choosing the explanations is taking the smallest explanations in terms of the number of hypotheses that are considered. Following this criterion, $\{h_1, h_2\}$ will be removed as an observation since its size is bigger than $\{h_3\}$.

3.2 Diagnosis

A diagnosis problem is one of the main applications of abductive reasoning. Its objective is determining the components that are failing in a system. Diagnosis is widely applied to determine the components that are failing in a circuit and diagnosing diseases in patients from their symptom. To deal with diagnosis problems, one of the most common frameworks is Reiter's Theory of diagnosis[9]. Reiter describes a system in terms of the expected behaviour of its components and how they are linked. Optionally, a description of how a component may fail may be introduced in what is called a *fault model*. Errors are detected by means of observations to the system behaviour and comparing them to its expected behaviour. If expected and real behaviours are different, an error is detected. In other terms, let us represent a system as a set of formulas F and let an observation *obs* be another set of formulas. An error is detected iff:

$$F \cup \mathit{obs} \models \bot, \text{ or } F \not\models \mathit{obs}$$

Therefore, error may be detected using deductive reasoning, as we are searching for consequences of adding *obs* to our knowledge. If we intend to go further and explain the reasons why errors happen we face up an abduction problem. As we may observe below, diagnosis problems perfectly fit into the abductive reasoning structure, since:

- The set of facts is the description of the system behaviour, describing both normal and abnormal behaviour of components.
- The set of hypotheses is composed by formulas that represent the normal and abnormal behaviour of each component.
- Observations are provided to obtain explanations to the errors that have been previously detected using deduction.

Therefore, using abductive reasoning we obtain a set of minimal explanations, where an explanation for an error is a list of components that are failing and a list of those that must behave correctly.

Summarizing, a diagnosis problem is an abduction problem where the only available hypothesis are those indicating the normal or abnormal behaviour of components.

3.3 Abduction, Deduction and Automated Analysis

Many operations have been proposed for the AAFM. Most of them are deductive operations since their objective is obtaining conclusions from a logic representation of a FM. However, there is a set of explanatory operations that have been solved using abductive reasoning techniques. As far as we are concerned, there has been no effort to remark this difference. So it is our intention to shed light on the difference between abductive and deductive reasoning so it could be applied in automated analysis.

Figure 3 summarizes our conception of the automated analysis when deductive and abductive operations are distinguished. In deductive operations, we are able to obtain conclusions (or the absence of them) from a FM logical representation that allows deductive reasoning. For abductive operations, we are interested in obtaining explanations from the results or conclusions obtained from a deductive operation. In this case, FMs are represented using logics that distinguish between facts and hypotheses. Deductive and abductive operations use different solvers or reasoners, choosing the most suitable for each kind of operation to be performed.

Next Section, we propose a catalog of abductive operations, and as we will expose later in Section 5, it will be a task of our future work to explain in details the translation of FMs to abductive logics and the solution using different techniques or solvers.

4. Operations Catalog

We present a catalog of operations for the abductive reasoning on FMs. These operations are executed just after a deductive operation. The catalog we present here is inspired by Benavides' [2] catalog of operations. We have selected its deductive operations and some others that have been proposed lately. For each deductive operation, we propose "'why?"' and "'why not?"' abductive questions. "'Why?"' questions are asked when a deductive operation has a solution. "'Why not?"' questions intend to find an answer for a deductive operation that has no solution. Small examples are provided to illustrate their usage.

4.1 Why? questions

A "'Why?"' question intends to explain the result obtained from a deductive operation. It is important to remark that in this case, deductive reasoning is able to obtain a result, but we would also like to know the reason why that result is inferred. We have found four relevant questions of this kind:

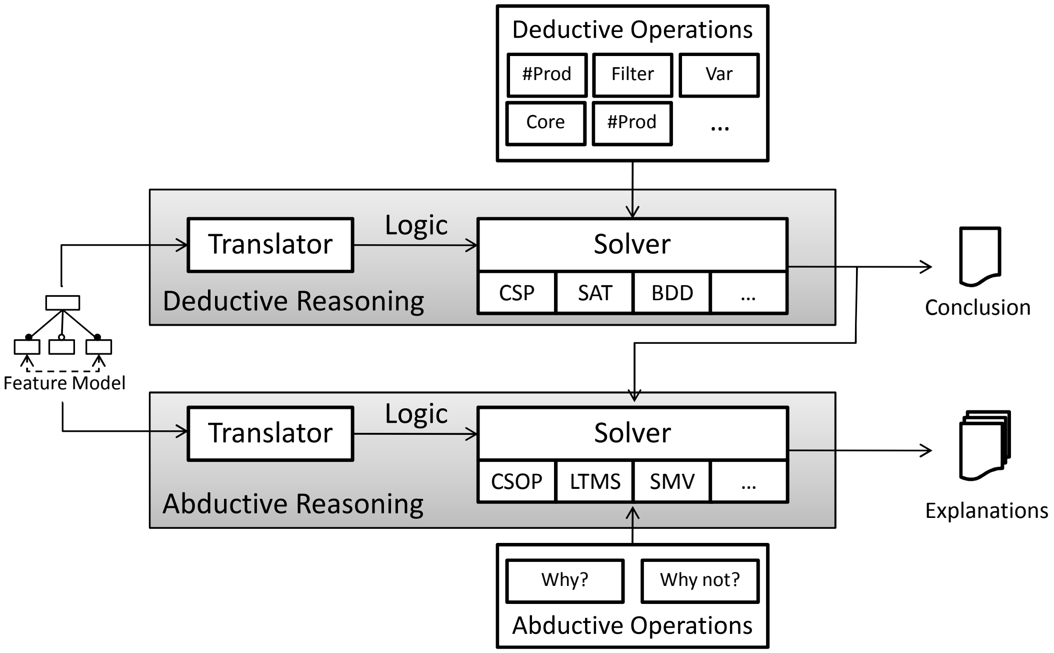
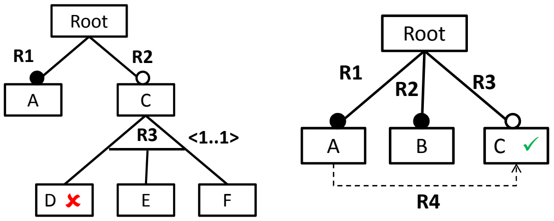


Figure 3. Relating abductive and deductive reasoning to automated analysis of FMs



(a) Operation 1 Example (b) Operation 2 Example

Figure 4. Example Feature Models

Operation 1. Why is it a variant feature? This operation is executed to extend the information obtained from the *"'retrieving the variant features"'* deductive operation. In this scenario, we want to obtain the relationship/s that are becoming a feature variant. Considering the example in Figure 4(a), if we want to determine the relationships that make feature D being variant we have to obtain a justification that concludes that we are able to remove that feature in a configuration. For the example, we will obtain $\{R2\}$ and $\{R3\}$ as two explanations to our question.

Operation 2. Why is it a core feature? The deductive operation *"'Retrieving the core features"'* lists the features that appear in every product or core features. This operation provides the relationships that makes one of those features belong to the core. Considering the example in Figure 4(b), all the features in the FM are core features. We expect C to be a variant feature since it is linked to the root by an optional relationship. *"'Why is it a core feature?"'* operation will highlight $R4$ and $R1$ relationships as a justification for C being a core feature.

We have seen how this operation and the previous one are applied to obtain more information from core and variant features. We must notice that we may also use both of them when we calculate the commonality or variability of a feature. A feature which commonality is 1 is a core feature; if its commonality is not 1 it is a variant feature. Therefore, we may use operation 1 and 2 for these cases.

Operation 3. Why is a partial configuration valid? A partial configuration in a FM is a list of selected and removed features. A complete configuration is a particular case of partial configuration where each feature in the FM is selected or removed. The deductive operation *"'Determining if a configuration is valid"'* infers whether it is possible to select and remove the features in a partial configuration. If a positive response is obtained, we may want to know the relationships that make the partial configuration possible. Let us take the FM in Figure 5 as an example, where the list of selected features is $\{Root, A, C, E\}$

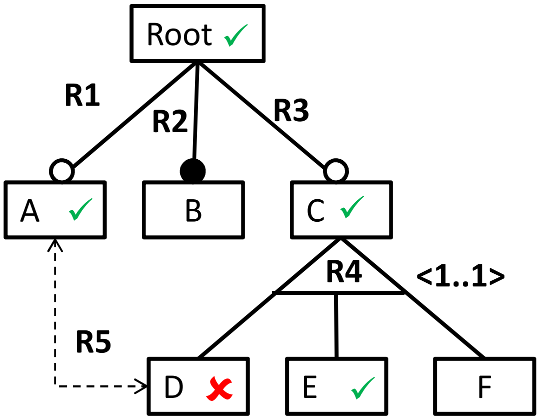


Figure 5. Operation 3 Example

and $\{D\}$ the list of removed features. The result of the abductive operation *”‘Why is a partial configuration valid?”’* will return $\{R1, R3, R4, R5\}$ as the set of relationships that affect those features.

Operation 4. Why is a product optimal for a criteria? Finding a product that optimizes a criteria is the objective of the deductive operation *”‘Optimizing”’*. This operation is commonly used when extra-functional information is attached to a FM in the so-called extended FMs[3]. In some situations we may be interested in knowing the relationships that have been taken into account to reach a solution. In the example in Figure 4.1, $\{Root, C, E\}$ features form the product that is found to be the cheapest product in the family. The abductive operation *”‘Why is a product optimal for a criteria?”’* will obtain $\{R2, R3, cost_{Root}, cost_C, cost_E\}$ as the relationships that make this product optimal. This operation may be seen as a particular case of Operation 3 where the configuration is obtained from an optimization process.

4.2 Why not? questions

Many deductive operations may obtain no solution or a negative response when inconsistencies are found. In the abductive operations that we analyse next, their objective is obtaining further information about the relationships that are making a deductive operation impossible to obtain a solution. As we intend to find the components (relationships in our case) that explain a failure or inconsistent situation, these operations fit into the diagnosis problem, so their results may be used to repair a FM or a configuration.

Operation 5. Why is a feature model not valid? A void FM is the one where it is not possible to derive any product. A FM is valid if it defines at least one product, i.e. it is not void. Void FMs are produced due to contradicting relationships. The deductive operation ”‘Determining if a FM is void”’ tries to find a valid product to demonstrate that a FM is valid. In case it finds no product, the FM is determined to be void and we need to extract information about the relationships that make the FM be void or not valid. *”‘Why is a feature model not valid?”’* operation obtains one or more explanations for a void FM, i.e. sets of relationships that prevent the selection of a product. In the example in Figure 7(a), three explanations are obtained: $\{R1\}$,$\{R3\}$ and $\{R4\}$. This information may be used by a feature modeler to correct the FM by relaxing or removing one or more of those relationships.

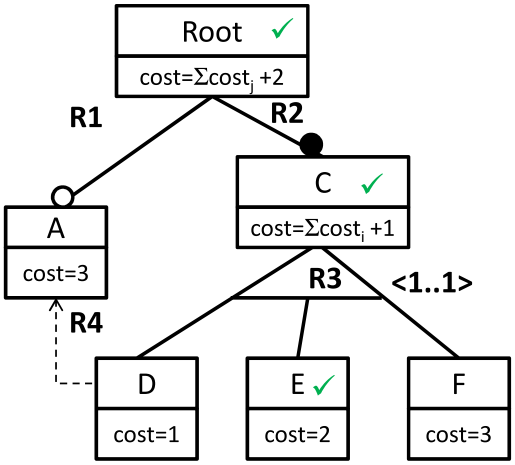
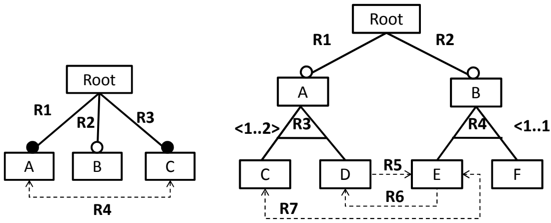


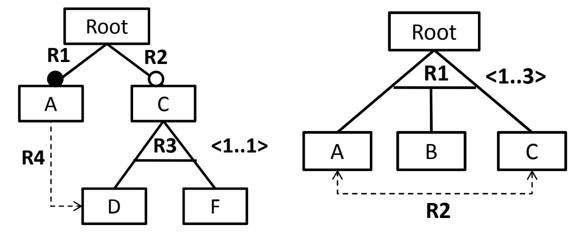
Figure 6. Operation 4 and 11 Example



(a) Operation 5 Example (b) Operation 6-7 Example

Figure 7. Example Feature Models 2

Operation 6. Why is a product not valid? Whenever the deductive operation *”‘Determining if a product is valid”’* detects an invalid product selection, it is mandatory to obtain further information about the relationships that are making the product impossible to



(a) Operation 8 and 9 Example (b) Operation 10 Example

Figure 8. Example Feature Models 3

derive. This operation will be useful when we want to check for a FM to include a set of well-known products. In the example in Figure 7(b), we want the FM to define the product $\{A, B, C, E\}$ (remaining features are supposed to be removed). Deductive operation detects this product as invalid and *”‘Why is a product not valid”’* explains this unexpected result by detecting $\{R4, R6\}$ as the relationships that are causing it.

Operation 7. Why is a partial configuration not valid? Whenever *”‘Detecting if a configuration is valid”’* detects an invalid configuration, and we know that the configuration must be possible, we may be interested in knowing the relationships that are making it impossible. Taking the FM in Figure 7(b) as example and partial configuration $\{C, D\}$, we obtain relationships $\{R5\}$ and $\{R7\}$ as explanations. From this point of view, we may consider previous operation as a particular case of this one, as a product may be considered as a partial configuration. Another approach to this question would be obtaining the features that must be removed from a configuration if we consider that the FM is correct. In this case, this operation would conclude that feature $\{C\}$ or $\{D\}$ must be removed from the configuration to obtain a valid one.

Operation 8. Why is a feature not selectable (dead feature)? A dead feature is the one that despite of appearing in a FM it cannot be selected for any product. The deductive operation ”‘Dead features detection”’ obtains a list of the dead features in a FM. This operation detects the relationships that are making a dead feature, assisting on the correction of the FM. Taking the FM in Figure 8(a) as example, $\{F\}$ is obtained as the only dead feature in the model. The explanations that we obtain are $\{R1\}$, $\{R3\}$ and $\{R4\}$, one of which must be removed or changed at least to correct the dead feature.

Operation 9. Why is a feature a false-optional? A false-optional (a.k.a. full-mandatory) feature is the one that has an implicit mandatory relationship with its parent feature despite of being linked by an optional relationship. The declarative operation *”‘False-optional features detection”’* obtains a list of this kind of features. This abductive operation obtains explanations to repair such an error. In Figure 8(a) example, $\{C, D\}$ features are false-optional, obtaining $\{R1\}$ and $\{R4\}$ as explanations.

Operation 10. Why is a cardinality not selectable (wrong cardinality)? Set-relationships use cardinalities to define the number of child features that may be selected whenever its parent feature is. When a cardinal is never used in any product, we are taking about a wrong cardinality. Although this operation is not theoretically described, it is supported by FAMA Framework [12] which implements a deductive operation *”‘wrong-cardinalities detection”’*. Taking Figure 8(b) example, we may notice that it is impossible to select 3 child features since A and C exclude themselves so $R1$ has a wrong cardinality. This operation will provide two explanations $\{R1\}$ and $\{R2\}$ since to correct the error we may remove the cardinality or the ”‘excludes”’ relationship.

Operation 11. Why is there no product following a criteria? When *”‘filtering”’* or *”‘optimizing”’* deductive operations are unable to find any product, this operation helps on finding the reasons why there is no solution. In the example in Figure 4.1, if we want to find a product which costs less than 4, *”‘filtering”’* will obtain no product at all. This operation will provide explanations such as $\{cost_{root}\}$ and $\{cost_E\}$ since they increase the total cost of a product in 2.

4.3 Summary

We present the relations among abductive and deductive operations in Table 1. The list of deductive operations is mainly inspired in [2] and [4] and extended with error analysis operations[11] and configuration operations[15].

In this table, N/A is used to represent those operations that do not fit into abductive reasoning. In this category we place ”‘determining if two FMs are equivalent”’ as it is an operation that compares two FMs and both deductive and abductive reasoning frameworks are only able to deal with just one FM. Corrective explanations are also out of our scope although they are closely connected to explanations. Corrective explanations may be considered as two-step operations where an error is explained firstly and corrected secondly. We are able to provide explanations via abductive reasoning, but suggesting corrections is not so trivial and will be an aim of our future work.

Deductive Operation	Abductive Operations	
	Why? operation	Why not? operation
Determining if a product is valid	N/S	Op.6
Determining if a FM is void	N/S	Op.5
Obtaining all the products	N/S	Op.7
Determining if two FMs are equivalent	N/A	N/A
Retrieving the core features	Op.2	Op.1
Retrieving the variant features	Op.1	Op.2
Calculating the number of products	N/S	Op.5
Calculating variability	Op.1 or Op.2	Op.8
Calculating commonality	Op.1 or Op.2	Op.8
Filtering a set of products	N/S	Op.6,7,11
Optimizing	Op.4	Op.11
Dead features detection	N/S	Op.8
Proving Explanations[1]	Op.1-4	Op.5-11
Providing Corrective Explanations	N/S	N/A
False-optional features detection	N/S	Op.9
Wrong-cardinalities detection	N/S	Op.10
Determining if a configuration is valid	Op.3	Op.7

[1] All the operations described in the table provide explanations for different contexts

Table 1. Relation between deductive and abductive operations

N/S is used to remark the operations that could be performed but will have no sense from the point of view of the automated analysis. For example, we are not interested in determining why a FM describes 20 products. However we must be interested in knowing why there a FM describes no product.

5. Conclusions and Future Work

In this work, we have presented our conception of AAFM from the point of view of the kind of reasoning needed to solve the different analysis operations. We have presented a new catalog of operations that rely on abductive reasoning and some of which have already been dealt with in some previous works, but the remaining operations are new. As a first step in our roadmap of integrating abduction in AAFM it is our intention to open a debate where the proposed catalogue of abductive operations is extended or reduced.

Once we have obtained a stable catalogue, we envision that we need two main pieces to complete the puzzle of abductive reasoning:

1. A translation from FMs to non monotonic logics, i.e. logics that are able to represent incomplete knowledge.
2. A solver-independent solution to all the abductive operations so that different solvers can be used to execute these operations.

We will implement the solutions to these operations into FAMA Framework [12]. Currently FAMA Framework supports explanations for operations 7 to 10 by means of CSOP that you may download at fter obtaining these results, we will design benchmarks to analysing the solvers that perform better for each abductive operation.

References

[1] D. Batory. Feature models, grammars, and propositional formulas. In *Software Product Lines Conference, LNCS 3714*, pages 7–20, 2005.

[2] D. Benavides. *On the Automated Analysis of Software Product Lines Using Feature Models. A framework for developing automated tool support.* PhD thesis, University of Seville, http://www.lsi.us.es/˜dbc/dbc_archivos/pubs/benavides07-phd.pdf, 2007.

[3] D. Benavides, A. Ruiz-Cortés, and P. Trinidad. Automated reasoning on feature models. *LNCS, Advanced Information Systems Engineering: 17th International Conference, CAiSE 2005*, 3520:491–503, 2005.

[4] D. Benavides, A. Ruiz-Cortés, P. Trinidad, and S. Segura. A survey on the automated analyses of feture models. In *Jornadas de Ingeniera del Software y Bases de Datos (JISBD)*, 2006.

[5] K. Czarnecki and P. Kim. Cardinality-based feature modeling and constraints: A progress report. In *Proceedings of the International Workshop on Software Factories At OOPSLA 2005*, 2005.

[6] K. Kang, S. Cohen, J. Hess, W. Novak, and S. Peterson. Feature–Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, Nov. 1990.

[7] M. Mannion. Using First-Order Logic for Product Line Model Validation. In *Proceedings of the Second Software Product Line Conference (SPLC2)*, LNCS 2379, pages 176–187, San Diego, CA, 2002. Springer.

[8] R. Reiter. On closed world data bases. pages 300–310, 1987.

[9] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.

[10] J. Sun, H. Zhang, Y. Li, and H. Wang. Formal semantics and verification for feature modeling. In *Proceedings of the ICECSS05*, 2005.

[11] P. Trinidad, D. Benavides, A. Durán, A. Ruiz-Cortés, and M. Toro. Automated error analysis for the agilization of feature modeling. *Journal of Systems and Software*, 81(6):883–896, 2008.

[12] P. Trinidad, D. Benavides, A. Ruiz-Cortés, S. Segura, and A.Jimenez. Fama framework. In *12th Software Product Lines Conference (SPLC)*, 2008.

[13] P. Trinidad, D. Benavides, A. Ruiz-Cortés, S. Segura, and M. Toro. Explanations for agile feature models. In *Procceedings of the 1st International Workshop on Agile Product Line Engineering (APLE'06)*.

[14] H. Wang, Y. Li, J. Sun, H. Zhang, and J. Pan. A semantic web approach to feature modeling and verification. In *Workshop on Semantic Web Enabled Software Engineering (SWESE'05)*, November 2005.

[15] J. White, D. Schmidt, D. B. P. Trinidad, and A. Ruiz-Cortes. Automated diagnosis of product-line configuration errors in feature models. In *Proceedings of the Sofware Product Line Conference*, 2008.