Towards Constructive Abduction
Solving Abductive Problems with Constraint Programming

Antoni Ligęza
AGH - University of Science and Technology, al. Mickiewicza 30, 30-059 Krakow, Poland

Keywords: Abduction, Constraint Programming, Model-based Diagnosis, Consistency-based Reasoning.

Abstract: Abduction can be considered as a principal way of reasoning for problem solving. Abductive inference consists in generation of hypotheses which explain — or logically imply — the phenomenon under investigation in view of accessible background knowledge and are consistent with all other observations. Looking for such hypotheses is typically performed with a spectrum of trial-and-error or search methods and tools. In case of purely logical statements the hypotheses take the form of a set of facts, both positive and negative ones. For example, in case of model based diagnostic reasoning, such diagnostic hypotheses can be generated by consistency based reasoning with minimal search effort. In more complex cases, where values of certain variables are to be found, pure backtracking search becomes inefficient. In this paper we attempt to put forward such abductive inference into a formal framework of Constraint Programming in order to enable the use of constraint propagation techniques. The main idea behind this approach is to make abduction more constructive. The discussion is illustrated with a diagnostic example of a multiplier-adder system.

1 INTRODUCTION

Abduction can be considered as one of principal ways of reasoning for problem solving. Abductive inference consists in *search for hypotheses* which satisfy some required conditions. For example, in case of diagnostic reasoning they should explain — or logically imply — the phenomenon under investigation, i.e. the observed failure. Note that accessible background knowledge must be taken into account and our hypotheses must be *consistent with all the knowledge and observations*.

Looking for such abductive hypotheses is typically performed with a spectrum of trial-and-error or search methods and tools. In case of purely logical statements the hypotheses take the form of a set of facts, both positive and negative ones. In case of Model-Based Diagnostic Reasoning (MBDR) (Hamscher et al., 1992) such diagnostic hypotheses can be generated by Consistency-Based Reasoning (CBR) (Reiter, 1987) with reasonable search effort. In more complex cases, where values of certain variables are to be found as well, a pure search (.e.g. Backtracking depth-first search or search on AND-OR graphs) becomes inefficient.

In this paper we attempt to put such abductive inference into a formal framework of *Constraint Programming* in order to enable the use of constraint propagation techniques. The main aim is to make abduction more *constructive*; this means that (i) the hypotheses found should be more precise, and (ii) inconsistent hypotheses should be eliminated in a more efficient way. The ideas are illustrated with a diagnostic example of multiplier-adder system.

Existing methods of formal description of diagnostic inference are diversified. There are algebraic, graph-based, and logical expert-like or model-based diagnostic approaches. Some of the popular models include Consistency-Based Reasoning (Reiter, 1987; Hamscher et al., 1992; Ligęza, 2004), logical causal graphs (Ligęza and Fuster-Parra, 1997; Ligęza, 2004), and many other (Davis and Hamscher, 1992; Korbicz et al., 2004). A recent survey of various approaches is given by (Travé-Massuyès, 2014). Following classical CBR, this paper explores mainly the abductive approach and it is focused on employing Constraint Programming for developing *constructive abduction*, where purely logical abductive hypotheses are enriched with exact numerical solutions.

The main point of interest in this work is the role of Constraint Programming in abductive problem solving, and mainly in diagnosis of technical systems. An in-depth analysis of applying Constraint Programming in modeling diagnostic reasoning is carried out.

Moreover, a kind of *Constraint Problem Solving* approach is investigated with the ultimate goal of pruning inconsistent behaviors.

The paper is organized as follows. In Section 2 we explain the nature of abduction from logical point of view. In Section 3 a simple motivation example is presented in detail. Section 4 covers some minimal information on Constraint Programming. Finally, Section 5 presents the main proposal of this paper, i.e. *constructive solution to abduction*.

2 ABDUCTION

Abduction can be considered as a principal way of reasoning for problem solving. Roughly speaking, abduction consists in stating hypotheses that possibly imply the facts observed to be true. Hence, the so-called *abductive inference* consists in generation of hypotheses which explain — or logically imply — the phenomenon under investigation taking into account all accessible background knowledge. The generated hypotheses must also be consistent with all other observations.

Looking for such hypotheses in abductive reasoning is typically performed with a spectrum of trial-and-error or search methods and tools. In case of purely logical statements the hypotheses take the form of a set of facts, both positive and negative ones. For example, in case of Model-Based Diagnostic Reasoning (MBDR) such diagnostic hypotheses can be generated by Consistency-Based Reasoning (CBR) with reasonable search effort.

Consider the following rule:

$$\frac{\alpha \Longrightarrow \beta, \beta}{\alpha}. \tag{1}$$

In fact, this is a basic scheme for abductive inference. Knowing that $\alpha \Longrightarrow \beta$, and observing that β holds, a possible explanation of it is α.

Note that abduction is not a *legal inference rule*, i.e. one preserving *logical consequence*. In our case, α is not the logical consequence of the antecedents. However, it is still a kind of production rule, often used in practice. The main use of abduction is the search for *hypotheses* explaining the current observations.

A general scheme of abductive reasoning is as follows. Consider a theory *SD* (System Description; this is a set of logical formulas describing in a formal way behavior of the system under discourse i.e. *background knowledge*) and a set *OBS* of some current observations to be explained. Abduction consists in finding a set *EXP* of explanatory hypotheses, such that:

- $SD \cup EXP \models OBS$, i.e. the hypotheses fully explain current observations taking into account knowledge about the system *SD*,
- $SD \cup EXP$ must be consistent.

Typically, it is also required that *EXP* is *minimal*, i.e. only the absolutely necessary hypotheses are specified within *EXP*. Further, if several competitive hypotheses are available, the most likely ones may be selected with some auxiliary tests, heuristics or statistical information.

Note that, abductive reasoning, although might provide invalid conclusions, constitutes a useful, practical way of guessing potential solutions to problem stated by system description, current observations, and question for the reasons for them. It is a natural way of diagnostic inference (both medical and technical), and can be applied in puzzle solving, constraint problem solving, and crime mystery solving[1]. Abduction, in order to be efficient, must be combined with reasonable search procedure and elimination method for incorrect solutions.

In case of diagnostic reasoning, abduction is often performed by detection and elimination of inconsistencies, i.e the so-called Consistency-Based Reasoning (CBR). The main idea of Model-Based Consistency-Based Diagnosis was presented in the seminal paper (Reiter, 1987), and widely explored in (Reiter, 1987; Hamscher et al., 1992; Ligęza, 2004; Davis and Hamscher, 1992; Korbicz et al., 2004). It rests in generation of diagnostic hypotheses stating which components of the system may be faulty (abduction), so that assuming them faulty explains the current observations with the model in mind in a consistent way (deduction).

More precisely, taking into account the background knowledge, i.e. System Description *SD* and the current observations *OBS* one looks for the so-called *conflict sets* — these are sets of components such that assuming all of them working correct is inconsistent with the observations. In fact, we look for such minimal sets and there may be several such sets. Now, by assuming that at least one of the elements of each minimal conflict set is faulty leads to regaining consistency. Simultaneously, a set containing one element from any minimal conflict set (the so-called *hitting set* (Reiter, 1987)) form a hypothetical diagnosis — a possible explanation of the observed misbehavior.

In the next section we analyze such an approach in a more detailed way.

[1]Note that, contrary to what was stated in the books, it was abduction — not deduction — as the principal way of reasoning applied by Sherlock Holmes (Ligęza, 2006), p. 31.

3 MOTIVATION EXAMPLE

In this section we briefly recall a classical diagnostic example of a feed-forward arithmetic circuit. This is the multiplier-adder example presented in the seminal paper by R. Reiter (Reiter, 1987). This example was further re-explored in numerous papers, including selected readings (Hamscher et al., 1992) and diagnostic handbook (Chapter (Ligęza, 2004)). It was further explored in the discussion carried out in domain literature concerning comparative analysis of diagnostic approaches (Cordier and et al., 2000b; Cordier and et al., 2000a; Travé-Massuyès, 2014). Here we shall base on an in-depth analysis presented in (Ligęza and Kościelny, 2008), and also explored in (Ligęza, 2009).

The basic, intuitive schema of the system is presented in Figure 1.

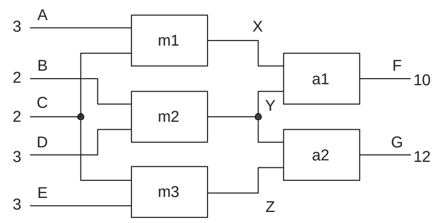

Figure 1.

The system is composed of two layers. The first one contains three multipliers $m1$, $m2$, and $m3$, and receiving the input signals A, B, C, D and E. The second layer is composed of two adders, namely $a1$ and $a2$, producing the output values of F and G. Only inputs (of the first layer) and outputs of the system (of the second layer) are directly observable. The intermediate variables, namely X, Y and Z, are hidden and cannot be observed.

Observe that the current state of the system is defined by the input values; they are: A=3, B=2, C=2, D=3 and E=3. It is easy to check — under the assumption of correct work of all the system elements — that the outputs should be F=12 and G=12. Note also that they *should be equal* to each other, which is due to the symmetry of the system and the symmetry of the input vales; this observation will be important for the analysis and we shall see later on why.

Now, since the current value of F is incorrect, namely F=10, the system is faulty. At least one of its components must be faulty[2]. At this stage, for simplicity, we consider only *correct* components and *faulty* ones; no details about the type of fault are taken into consideration so far.

In order to perform diagnostic reasoning let us start with *abduction* (Ligęza, 2004), (Ligęza and Kościelny, 2008); we shall try to build a conflict set specifying hypothetical faulty elements; in other words we search for a *Disjunctive Conceptual Faults; DCF for short* (Ligęza and Kościelny, 2008).

Note that the value of F is influenced by the inputs (observed) and the work of elements $m1$, $m2$ and $a1$ all of them are located in the signal path having direct influence on the value of F. In other words, one can say that there is direct *causal dependency* of influence of $m1$, $m2$ and $a1$ on F. If all the three elements work correctly, then the output would be correct. Since it is not, we can conclude that a DCF_1 is observed: at least one of the elements $\{m1, m2, a1\}$ must be faulty. Formally, we have

$$DCF_1 = \{m1, m2, a1\} \tag{2}$$

Note that, at this stage of reasoning only $m1$ and $a1$ are satisfactory explanations for the observed fault; $m2$ alone is not a good explanation, because it is not consistent with the observation G=12.

A further analysis leads to detection of DCF_2 (Ligęza and Kościelny, 2008): under the assumed manifestations one of the elements $\{m1, a1, a2, m3\}$ must also be faulty.

In order to explain the origin of DCF_2 let us notice that if all the four elements were correct, then Z=C*E calculated by $m3$ must be equal to 6, and since G is observed to be 12, Y (calculated backwards and under the assumption that $a2$ works correctly) must also equal 6; hence, if $m1$ is correct, then X must be 6 as well, and if $a1$ is correct F would be equal to 12. Since it is not the case, at least one of the mentioned components must be faulty. So we have the following disjunctive conceptual fault:

$$DCF_2 = \{m1, a1, a2, m3\} \tag{3}$$

Note that the type of DCF_1 and DCF_2 are different; this is due to their origin (or character). DCF_1 is of *causal* type — all the elements directly influence the conflicting variable. On the other hand, DCF_2 is of *constraint* type; this is a kind of mathematical constraint which must be satisfied, but there is not necessary causal dependency between the components and the value of the faulty variable (F). This observation (and fault classification) will be used later on, when we pass to qualitative analysis.

Note that if both the outputs were incorrect (e.g. F=10 and G=14), then, in general case one can observe DCF_1, DCF_2 and DCF_3, where:

$$DCF_3 = \{m2, m3, a2\}. \tag{4}$$

[2]In Model-Based Diagnosis it is typically assumed that faulty behavior is caused by a fault of a named component or a simultaneous fault of a set of such components; no faults caused by faulty links, parameter setting or the internal structure are considered.

The DCF_3 is a *causal type* conflict.

Note also, that whether the constraint-type DCF_2 is a valid conflict may depend on the observed outputs. For example, if F=10 and G=10 (both outputs are incorrect but equal), then the structure and equations describing the work of the system does not lead to a conceptual fault (Cordier and et al., 2000a; Cordier and et al., 2000b).

The composition of the three disjunctive conceptual faults can be presented with use of an OR-matrix for the diagnosed system (see Table 1): It defines the component elements for particular DCF-s for DCF_1, DCF_2, and DCF_3.

Table 1: An OR binary diagnostic matrix for the adder system (the lower level).

DCF	$m1$	$m2$	$m3$	$a1$	$a2$
DCF_1	1	1		1	
DCF_2	1		1	1	1
DCF_3		1	1		1

In the analyzed case, i.e. F being faulty and G correct, the final diagnoses for the considered case are calculated as reduced elements of the Cartesian product of $DCF_1 = \{m1, m2, a1\}$ and $DCF_2 = \{m1, m3, a1, a2\}$ (Ligęza and Kościelny, 2008). There are the following potential diagnoses: $D_1 = \{m1\}$, $D_2 = \{a1\}$, $D_3 = \{a2, m2\}$ and $D_4 = \{m2, m3\}$. They all are shown in Figure 2.

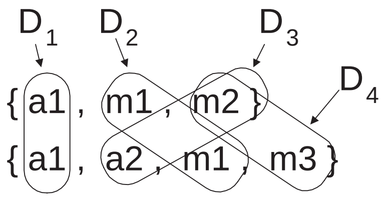

Figure 2: Generation of potential diagnoses.

Note that so far only binary faults were considered (i.e. a component may be faulty or not). In (Ligęza and Kościelny, 2008) and further in (Ligęza, 2009) an attempt at introducing qualitative diagnoses was undertaken. After calculation of possible binary diagnoses their qualitative forms were considered, and with use of inference rules representing simple constraints inconsistent qualitative diagnoses were eliminated. In the next section a still more precise, in-depth, numerical analysis will be carried out. The model of the system will be used as constraints. The binary logical diagnoses (i.e. *faulty* or *correct*) will be replaced with exact numerical characteristics.

4 CONSTRAINT PROGRAMMING

In this section a brief note on Constraint Programming is presented. We aim at explaining the basic ideas of this promising technology for solving complex, combinatorial problems. Our presentation is based on (Ligęza, 2009).

A Constraint Satisfaction Problem (CSP) is one where the goal consists in finding a legal assignment of values to a set of predefined variables so that a set of given constraints is satisfied.

More formally, after (Dechter, 2003) let $X = \{X_1, X_2, \ldots, X_n\}$ denote a set of variables, $V = \{V_1, V_2, \ldots, V_n\}$ is a set of domains for the variables in X and C is a set of constraints. Each constraint is given by a pair (S_i, R_i), where S_i is referred to as the *scope* (or scheme) and consists of a selection of variables from X, while R_i is a relation defined over a Cartesian Product of domains appropriate for the variables in the scope. The Constraint Satisfaction Problem is given by the triple (X, V, C).

A solution to CSP given by (X, V, C) is any assignment of values to variables of X of the form $\{X_1 = v_1, X_2 = v_2, \ldots, X_n = v_n\}$, such that $v_i \in V_i$, and for any constraint in $(S_i, R_i) \in C$, R_i is satisfied by the appropriate projection of the solution vector $(v_1, v_2, \ldots, v_k)$ over variables of S_i. Obviously, all the constraints must be satisfied, and there can be more one or many solutions; no solution may exist for an over-constrained problem.

If the domains of V are *finite* (e.g. binary or decimals digits are allowed) we speak about Finite Domains (FD) problems. Obviously, such problems suffer from *combinatorial explosion* while attempting at solving them. Constraint Programming (CP) is a set of techniques (or a technology) for efficient dealing which constraints.

The basic technique for solving a CSP given by (X, V, C) consists in subsequent assignment of admissible values to variables of X (e.g. by *backtracking search*); the order is chosen in an arbitrary way, and it can influence how fast a solution is found.

In order to improve search efficiency both heuristics and strategies are used. The most typical strategies are based on (i) variable ordering, (ii) values ordering, and (iii) *look-ahead* techniques. Especially the look-ahead strategies can improve search efficiency. The principal idea is that the algorithm looks how current decisions will affect the future search.

5 ABDUCTION AS CONSTRAINT PROGRAMMING

The key message of this section is to show that abduction can be solved in a constructive way with efficient approach of Constraint Programming. The multiplier-adder example (presented in Section 3; see Figure 1) will be used as an illustrative guideline. The obtained diagnoses are more precise than in the case of purely logical abduction of Consistency-Based Diagnosis. In case of more than one potential diagnoses, the exact values of variables can be used for further analysis, and — if applicable — elimination of some spurious diagnoses due to impossible values predicted by CP or thanks to introduction of additional measurements for confirming or rejecting the generated values.

Let us refer to the multiplier-adder system as presented in Figure 1 once again. As the starting point consider the four *logical* potential diagnoses: $D_1 = \{m1\}$, $D_2 = \{a1\}$, $D_3 = \{a2, m2\}$ and $D_4 = \{m2, m3\}$. They are logical in the sense that the only information provided is of logical value: *True* if a diagnosis is valid or *False* if a diagnosis is invalid.

We shall build a constraint model for the diagnostic case presented in Figure 1 so as more detailed, numerical characteristics can be abducted.

First, the set of common observations is modeled with simple constraints as follows[3]:

```
[...]
A #= 3,
B #= 2,
C #= 2,
D #= 3,
E #= 3,
F #= 10,
G #=12,
[...]
```

In this code excerpt comma is used to separate constraints, while `#<op>` (like `#=` or `#>`) is used to express constraints; several typical relations can be used in place of `<op>`.

Now, consider the first case of $m1$ being faulty. We assume that a faulty multiplier produces the incorrect output and its value can be expressed as multiplication of the correct value by a factor `K1/M1`, where both the numbers are integers. The model takes the following form:

```
A * C * K1 #= X *M1,
B * D #= Y,
C * E #= Z,
X + Y #= F,
Y + Z #= G,
K1 #> 0, M1 #> 0.
```

[3]The constraints are direct codes of SWI-Prolog; for constraint modeling we use the clp(fd) package

The first constraint models the fault of $m1$. For the sake of operation in the domain of integers, `M1` is placed as a multiplication factor on the right-hand side. The other constraints correspond directly to the operation and connections of the components. The produced output is:

```
X=4, Y=6, Z=6, K1=2, M1=3
```

and it can be easily checked by hand.

As the second case consider the diagnosis $a1$ being faulty. We assume that a this time it is the adder that produces the incorrect output and its value can be expressed as subtraction from the correct value a factor `A1` (an integer). The model takes the following form:

```
A * C #= X,
B * D #= Y,
C * E #= Z,
X + Y - A1 #= F,
Y + Z #= G.
```

The fourth constraint models the fault of $a1$. The other constraints correspond directly to the operation and connections of the components. The produced output is:

```
X=6, Y=6, Z=6, A1=2
```

and again, it can be easily checked by hand.

The third, a bit more complex case is the one of active diagnosis $\{a2, m2\}$. The model for this case is as follows:

```
A * C #= X,
B * D * K2 #= Y * M2,
C * E  #= Z,
X + Y  #= F,
Y + Z +A2 #= G,
K2 #> 0, M2 #> 0.
```

Variables `K2/M2` model the multiplicative fault of $m2$ and variable `A2` models the fault of $a2$. The produced output is:

```
X=6, Y=4, Z=6, K2=2, M2=3, A2=2
```

and again, it can be easily checked by hand.

The fourth and perhaps the most complex case is the one of active diagnosis $\{m2, m3\}$. Here we have to introduce four variables, namely `K2/M2` and `K3/M3` for modeling two multiplicative faults, namely the one of $m2$ and $m3$, respectively. The model is as follows:

```
A * C #= X,
B * D * K2 #= Y * M2,
C * E * K3 #= Z * M3,
X + Y #= F,
Y + Z #= G,
K2 #> 0, M2 #> 0,
K3 #> 0, M3 #> 0.
```

The produced output is:

```
X=6, Y=4, Z=8, K2=2, M2=3, K3=4, M3=3
```

and again, it can be checked by hand.

In the above example we applied the simplest methodology for constraint modeling applied to solving an abductive problem of finding *detailed numerical characteristics* — in fact *numerical models* — of *minimal diagnoses*. The basic stages can be summarized as follows:

- take a minimal diagnosis for detailed examination; it can be composed of k components,
- for any component define one or more (as in the case of multipliers) variables with use of which one can capture the idea of the misbehavior of the component,
- define the domains of the variables (a set necessary in CP with finite domains),
- define the constraints imposed on these variables for the analyzed case, and finally
- using the intended definition of components in presence of the assumed fault, define the constraints modeling the work of all the components (the correct ones and the faulty ones).

The flow (links) between the component are defined by the appropriate use of defined input-output and internal variables. Note that the *direction of flow*, or *causality* is not of interest here, and one does not need to bother about that. In fact, what we look for is a stable, numerical solution consistent with the diagnoses and the observations. Such a model may be used for further analysis and elimination of unfeasible diagnoses.

In case of many potential diagnoses, the procedure should be repeated for each diagnosis.

6 CONCLUSIONS

The paper presents a note on applying Constraint Programming for enriching abductive reasoning with exact (numerical) knowledge. In this way not only logical or qualitative solutions are obtained (of the form yes/no), but detailed numerical characteristics of the generated solutions are provided. This kind of approach we call *constructive abduction*.

The models presented in Section 5 were relatively simple, and provided for all four diagnoses separately. But it seems straightforward to build one, general model by introducing another five binary variables for representing if a component is faulty or not. In this way appropriate subset of all constraints can be made active, while inappropriate constraints are eliminated.

Finally, the focus of the paper was on a diagnostic example, but it seems that this kind of approach can be applied in a wide spectrum of abductive problems.

ACKNOWLEDGMENTS

The presented research was carried out within AGH University of Science and Technology Internal Project No. 11.11.120.859.

REFERENCES

Cordier, M.-O. and et al. (2000a). Ai and automatic control approaches of model-based diagnosis: Links and underlying hypotheses. In Edelmayer, A. M., editor, *Preprints: SAFEPROCESS 2000, 4th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*, pages 274–279. IFAC.

Cordier, M.-O. and et al. (2000b). A comparative analysis of ai and control theory approaches to model-based diagnosis. In Horn, W., editor, *ECAI'2000. 14th European Conference on Artificial Intelligence*, pages 136–140. IOS Press.

Davis, R. and Hamscher, W. (1992). *Model-Based Reasoning: Troubleshooting*, pages 3–24. Morgan Kaufmann Publishers, San Mateo, CA.

Dechter, R. (2003). *Constraint Processing*. Elsevier Science.

Hamscher, W., Console, L., and de Kleer, J., editors (1992). *Readings in Model-Based Diagnosis*. Morgan Kaufmann, San Mateo, CA.

Korbicz, J., Kościelny, J., Kowalczuk, Z., and Cholewa, W., editors (2004). *Fault Diagnosis. Models, Artificial Intelligence, Applications*. Springer-Verlag, Berlin.

Ligęza, A. (2004). *Selected Methods of Knowledge Engineering in System Diagnosis*, chapter 16, pages 633–668. In: (Korbicz et al., 2004). Springer-Verlag.

Ligęza, A. (2006). *Logical Foundations for Rule-Based Systems*. Springer-Verlag, Berlin, Heidelberg.

Ligęza, A. (2009). *A Constraint Satisfaction Framework for Diagnostic Problems*, pages 255–262. Control and Computer Science. Information Technology, Control Theory, Fault and System Diagnosis. Pomeranian Science and Technology Publisher PWNT, Gdańsk.

Ligęza, A. and Fuster-Parra, P. (1997). And/or/not causal graphs – a model for diagnostic reasoning. *Applied Mathematics and Computer Science*, Vol. 7, No. 1:185–203.

Ligęza, A. and Kościelny, J. M. (2008). A new approach to multiple fault diagnosis. combination of diagnostic matrices, graphs, algebraic and rule-based models. the case of two-layer models. *Int. J. Appl. Math. Comput. Sci.*, 18(4):465–476.

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

Travé-Massuyès, L. (2014). Bridges between diagnosis theories from control and ai perspectives. In Korbicz, J. and Kowal, M., editors, *Intelligent Systems in Technical and medila Diagnosis*, pages 3–28. Springer-Verlag.