A Formal Approach to Information Fusion

Mieczyslaw M. Kokar
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA 02115, USA

Jerzy A.Tomasik
Universite Blaise Pascal
LLAIC1, BP86 63172 AUBIERE
France

Jerzy Weyman
Department of Mathematics
Northeastern University
Boston, MA 02115, USA

Abstract *This paper proposes a formalization of the notion of "information fusion" within the framework of formal logic and category theory. Within this framework information fusion systems can be specified in precise mathematical terms allowing in this way to formally reason about such specifications, designs and implementations. The notion of fusion proposed in this paper differs from other approaches, where either data or decisions are fused. Here, the structures that represent the meaning of information (theories and models) are fused, while data are then simply processed using these structures (filtered through these structures). Within this framework the requirement of consistency of representations is formally and explicitly specified and then can be manipulated by the computer using automatic reasoning techniques.*

Keywords: information fusion, formal methods, category theory, model theory

1 Introduction

An *information fusion system (IFS)* (see Figure 1) may receive inputs from various sources: sensors, data bases, knowledge bases, and other systems (over communication lines). In our discussion we will focus on inputs from sensors, since other sources of information can be considered as special kinds of sensors. Sensors provide measurements of a number of interrelated variables (*n-tuples*). In mathematical sense, sensors output either functions or relations. In general, the *goal* of an IFS is to interpret data received through sensors. It is expressed in a prespecified *goal language* understandable to either the user or another system.

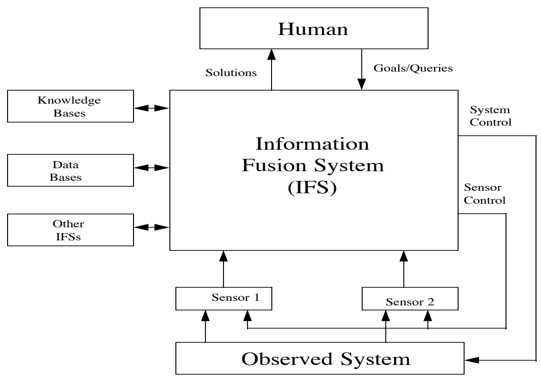

Figure 1: Information Fusion System (IFS)

A natural requirement for an information fusion system is that the interpretation of the data be "correct". Intuitively, this means that the objects identified by the IFS really exist in the world, that these objects have the features as identified by the IFS, that the relations recognized by the IFS really exist in the world, and that the interpretation does not violate the constraints that the world is known to obey, e.g., the laws of physics. In order to maintain the truthfulness of the interpretation, the system must maintain *consistency* of its representation.

To deal with the issue of correctness of interpretations we use the framework of model theory [1]. In particular, we make use of *formal languages* to describe the world and the sensing process and *models* to represent sensor data, operations on data, and relations among the data. Models consist of *carriers* of different sorts (usually sets) and many-sorted *operations*, and *relations* among the elements of different carriers. We use *theories* to represent symbolic knowledge about the world and about the sensors.

Fusion is then treated as a goal-driven operation of combining a fixed number of languages, theories and classes of models related to the goal, the sensors and the background knowledge, into one combined language, one combined theory and one combined class of models of the world. Therefore, fusion is a *formal system operator* that has multiple languages, theories and classes of models for inputs and a single language, a theory, and a class of models as the output.

This understanding of fusion differs from more traditional approaches [2, 3], where issues like consistency are not dealt with explicitly. Rather, there is an underlying presumption that the operations of fusion are implemented in a consistent way by the human. In our approach, on the other hand, a framework is provided in which the requirement of consistency of representations can be formally and explicitly specified and then can be manipulated by the computer using automatic reasoning techniques.

Although there are several definitions of "fusion" in the subject literature, there does not seem to be an agreement on what is and what is not fusion. In Section 2 we argue that the issue of fusion must be addressed in the specification phase. Then in Section 3, we provide our formal definition of fusion. In Section 4 we identify two parts of the fusion problem: syntactic fusion and semantic fusion. Section 5 puts the problem of fusion in the category theory framework and discusses fusion operators. We present and example of a specification developed according to our approach in Section 6. Finally, in Section 7 we provide conclusions.

2 Decomposition of the IFS

In this presentation we follow a top-down approach by progressively decomposing the problem of development of an IFS into simpler subproblems. In the first cut we decompose the IFS into three subsystems, as shown in Figure 2. This decomposition follows the formal approach to software development, where code is developed in the process of progressive refinement of a formal software specification. Information Processing represents the actual running system that takes inputs from all the sources and produces outputs in real time. The main fusion problem, as presented in this paper, is solved in Specification Synthesis. This is essentially the only block where expertise of sensors and scenarios is needed. Code Generation can be performed independently of such expertise.

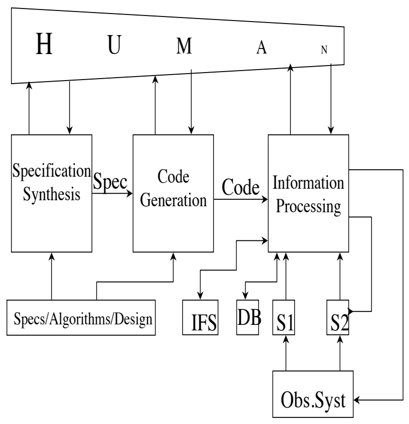

Figure 2: Information Fusion System: First-level Decomposition

3 The Fusion Problem

In this section we consider the main fusion block, i.e., the Specification Synthesis block. By specifications we mean signatures (languages), theories over the signatures, and classes of models of the theories. We show the first decomposition of the Specification Synthesis block in Figure 3. As we said in Section 1, the goal of information fusion is to develop a fused theory T_f and a fused class of models $\{M_f\}$. The inputs to this fusion process, as shown in Figure 3, are some or all of the following:

(1) Σ_G, Σ_i, Σ_j, $\Sigma_b, \ldots$ – signatures associated with the goal of the fusion system, corresponding sensors and background knowledge theories. These signatures include variables and constant symbols of different sorts as well as many-sorted operation and relation symbols.

(2) T_G, T_i, T_j, $T_b, \ldots$ – formal theories describing the *goal theory*, knowledge about the sensors, and theories of the world (background knowledge) expressed in terms of the above described signatures. Background knowledge contains constraints on possible interpretation of the received data and/or special theories like Theory of Reals (Real Closed Field Theory), Random Sets Theory, Elementary Theory of Boolean Algebras that can be utilized in the process of constructing the fused theory.

(3) $\mathcal{G}$ – goal. These are queries about the world that cannot be answered in general by using only one of the sensors (information sources) but can be answered by using many (all) sensors. They are formulas expressed in terms of the signature Σ_G of the goal theory T_G.

(4) $\{M_G\}$, $\{M_i\}$, $\{M_j\}$, $\{M_b\}, \ldots$ – classes of models associated with the theories T_G, T_i, T_j, $T_b, \ldots$, respectively.

The Fusion Problem

Given the knowledge described above, construct a *fused theory* T and an appropriate class of *fused models* $\{M\}$, such that for any model M in $\{M\}$:

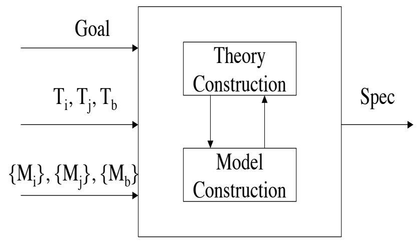

Figure 3: Specification Synthesis

1. $M \models G$
2. $M \models T$
3. $M \models T_b$

In some cases we might be given specific models M_i, M_j, instead of classes of models. Depending on which of the above are available, and depending on some other preferences, the process of developing the fused theory and class of models may be arranged on many different ways. For instance, we might first develop a fused theory T_f, and then find a class of models associated with this theory.

4 Syntactic/Semantic Fusion

One way to achieve the fusion goal is to split the inputs to the Specification Synthesis block (Figure 3) between the two tasks, so that purely semantic information (theories) are input into the Theory Construction task and the semantic inputs (models) are input into the Model Construction task. We denote the syntactic task by ∇_T, and the semantic task by ∇_M.

To be consistent with the formulation of the fusion problem in Section 3, the diagram represented in Figure 4 must commute. This can be described by the following relations.

$$M \models \nabla_T(T_G, T_i, T_j, T_b)$$

$$M = \nabla_M(M_G, M_i, M_j, M_b)$$

$$M_G \models T_G, M_i \models T_i, M_j \models T_j, M_b \models T_b$$

$$\begin{tikzcd}
T_G,T_i,T_j,T_b \arrow[r, "\nabla_T"] \arrow[d, "\bullet"'] & T \arrow[d, "\bullet"] \\
\{M_G\},\{M_i\},\{M_j\},\{M_b\} \arrow[r, "\nabla_M"'] & \{M\}
\end{tikzcd}$$

Figure 4: Syntactic and Semantic Levels of Specification Synthesis

4.1 Syntactic Theory Construction

The Theory Construction task can be considered as a goal-driven process that starts with the goal theory T_G. This theory contains a goal sentence G. The intent is to prove that the goal is true. This theory has to be combined with (extended by) other theories in order to make such a proof possible. It is natural to use the theories of the sensors in the first place. If the goal still cannot be proved, other background theories T_b need to be added. Various standard mathematical theories are added also in this step. The signatures of the goal theory and of the sensor theories, as well as some non-logical symbols appearing in these theories, can be used in the search for theories to add. As a result, we obtain a sufficiently rich theory T (specification) in which all sorts and operations from the goal theory T_G should be definable. In other words, the transition from the goal and sensor theories to the fused theory T can be achieved by appropriate definitional extensions of these theories using the background theories T_b.

4.2 Semantic Model Construction

In the Model Construction task we need to combine structures (classes of models of the particular theories fused in the syntactic task) into one class of structures. Since as a net result, this operation should produce such a class of structures $\{M\}$ that each one of them is a model of the fused theory T, the semantic model construction operation ∇_M must be so chosen that this property holds.

5 The Fusion Operator

In Section 4 we presented fusion as consisting of two operators, ∇_T and ∇_M. What can these operators be? In this section we propose a category theory based approach to this problem, similar to the one taken in the Specware approach [4]. In this approach theories are represented as specifications. They are objects in the Small Categories (Cat). Relationships among them are morphisms. Composition of theories is done using the *colimit* operation. Models of the theories are objects of another category (Mod).

According to this paradigm, Figure 4 can be represented as in Figure 5. In this diagram, corners of the diagram represent objects (or collection of objects). The arrows represent morphisms. The operators became:

$$\nabla_T(T_G, T_i, T_j, T_b) = Col(T_G, T_i, T_j, T_b)$$

$$\nabla_T(\{M_G\}, \{M_i\}, \{M_j\}, \{M_b\}) =$$

$$Lim(\{M_G\}, \{M_i\}, \{M_j\}, \{M_b\})$$

where Col represents the colimit operator and Lim represents the limit operator. Note that, since Lim and Col are two contravariant operators, the morphism arrows point in opposite directions.

According to this diagram, fusion is accomplished by two operators: colimit and limit. The colimit operation combines (glues) two theories (specifications) along the common part. It is a *shared union* of two theories. In other words, first, common parts are identified in the languages associated with particular theories, then these common parts are renamed so that they have the same symbols in both theories, then the renaming is reflected in the axioms of the theories, and finally, the theories are put together into one structure (one theory).

$$\begin{tikzcd}
T_G,T_i,T_j,T_b \arrow[r, "Col"] \arrow[d, "Mod"'] & T \arrow[d, "Mod"] \\
\{M_G\},\{M_i\},\{M_j\},\{M_b\} & \{M\} \arrow[l, "Lim"]
\end{tikzcd}$$

Figure 5: Category Theory Fusion Operators

Note also that the arrows from theories to models are of a different kind - they are *functors* that map objects of one category into the objects of another category.

6 Example

In this section we discuss a simple and idealized fusion scenario. In this example (see Figure 6) we consider a world which is a two dimensional plane with two kinds of objects possible: rectangles and triangles (with one right angle). The objects are illuminated with parallel light; the light direction is denoted by the angle α, as indicated in the figure. The world is measured through two sensors: a one-dimensional vision sensor, and a one dimensional range sensor.

The goal is object recognition. In some cases the range sensor is sufficient for the classification of an object into one of the three classes. E.g., when an acute angle is at the sensor side, the range sensor gives enough information to classify the object as either a right triangle or as an illegal object. Nevertheless, in some other cases, the information provided by the range sensor is not sufficient to make such a distinction. The advantage of the vision sensor stems from its ability to see shadows. In some configurations (sizes of an object and its rotational location), the size of the shadow and its location can provide the extra information that can be used to decide if the object is a triangle or a rectangle (as shown in Figure 6).

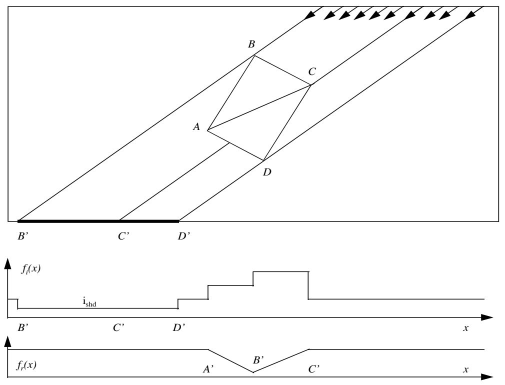

Figure 6: A scenario for sensor data fusion

To understand this example the reader has to possess some knowledge of geometry and physics. We cannot expect that a computer has this kind of capabilities. Our goal is to understand the mechanisms involved in the above example, formalize these mechanisms, and then implement them in the computer so that this knowledge can be incorporated in the processing automatically.

6.1 Formalization of Knowledge

In the following we list the theories involved in the recognition process, show examples of the theories, and describe how they are fused. A complete presentation would include the description of appropriate classes of models. We implemented these theories in the specification language Slang, used by Specware [4], a formal method tool. For readability, however, the theories are presented here using common mathematical notation.

Theory $\mathcal{T}_r$: Range Sensor. The theory of the range sensor, $\mathcal{T}_r$, consists of the following two axioms:

1. $f_r(x) = y \wedge y < 1 \Rightarrow O_r(x, y)$

2. $f_r(x) = y \wedge y = 1 \Rightarrow \neg O_r(x, y)$

where 1 is a constant symbol, f_r is a symbol denoting a one-placed function (sensor measurement function), O_r is a symbol denoting a two-placed relation (detection).

Theory $\mathcal{T}_i$: Intensity Sensor. The theory $\mathcal{T}_i$ contains the knowledge about the intensity sensor. It consists of the following single axiom:

$$f_i(x) = i_{shd} \Rightarrow \ S(x)$$

where i_{shd} - is a constant symbol, f_i is a symbol denoting a one-placed (measurement) function, S is a symbol denoting a one-placed relation (detection of shadow). We have selected a very simple theory of the intensity sensor, since in this example, we use this sensor solely to identify shadows. We extract other relevant information from the range sensor.

Theory $\mathcal{T}_{rt}$: Rectangles and Right Triangles. This theory contains knowledge to distinguish rectangles from right triangles. It includes the following predicates: *segment, constant, length, projection, angle, right-angle, acute-angle, triangle, rectangle.* This knowledge is just a subset of geometry, and thus is not specific to any sensor or a specific scenario. As an example, the *segment* predicate is defined as follows:

$$SEG(x_1, y_1, x_2, y_2) \leftrightarrow \forall_{x_1 \le x \le x_2} O(x, y) \land$$

$$y = \frac{y_2 - y_1}{x_2 - x_1} \cdot x + y_1 \land \forall_{x_2 \le x \le x_1} \neg O(x, y)$$

Theory $\mathcal{T}_{sh}$: Shadows. This theory contains two axioms:

$$SHD(x_1, x_2) \Leftrightarrow \forall_{x_1 \le x \le x_2} S(x) \land \forall_{x_2 \le x \le x_1} \neg S(x)$$

$$\begin{aligned} TRN(x_1, y_1, x_2, y_2, x_3, y_3) \land \\ RAN(x_1, y_1, x_2, y_2, x_3, y_3) \land \\ PRJ(x_2, y_2) = x_l \land PRJ(x_3, y_3) = x_r \land \\ SHD(x_l, x_r) \Rightarrow TSH \end{aligned}$$

where SHD is the symbol for a two-placed relation (end points of the shadow), TSH – constant representing "shadow of a triangle", and S is part of the language of the theory $\mathcal{T}_r$. The first axiom states that shadows are continuous, and the second axiom defines conditions for when a shadow can be TSH, it is the shadow of a triangle. The idea behind this axiom is the essence of this fusion problem. It can be understood by analyzing the scenario in Figure 6.

Theory $\mathcal{T}_w$: The World. In our example we presume that our world can be in three possible states: either it includes a rectangle, or a right triangle, or is empty.

$$\neg(TRN \land REC)$$

$$(TRN \lor REC) \land \neg TSH \Rightarrow REC$$

Goal: $\mathcal{G}$ The goal of the system is to find our which of the following four situations is the case in the world: (1) there is only a rectangle in the world ($\neg TRN \land REC$), (2) there is only a right triangle in the world ($TRN \land \neg REC$), (3) there is either a single rectangle or a single triangle in the world ($\neg TRN \land \neg REC$) (4) the world contains no objects ($TRN \lor REC$).

6.2 Formalization of Fusion

The specification of the rectangle/triangle recognition system was developed in Slang, the language used by the formal method tool, Specware. Both Specware and the underlying its implementation category theory are described in [5]. The structure of the resulting specification is shown in Figure 7.

The specification was developed in a bottom-up fashion. In the first step we developed the specification XREAL. This is an extension of the theory of real numbers (REAL). REAL is one of the theories that is provided with the library of Specware. We needed some additional functions and thus we needed to extend this theory. The arrow from REAL to XREAL is called *import* in Specware. It is an extension of the category theory concept of *colimit* described in [5].

In the next step, theories of the range sensor, $\mathcal{T}_r$, and of the intensity sensor, $\mathcal{T}_i$, described in Section 6.1, are encoded in Slang. Both theories need to import XREAL. In the Specware implementation they are called RANGE-SENSOR and INTENSITY-SENSOR, respectively. In a similar manner, the RECT-TRIAN specification is created; it imports RANGE-SENSOR and encodes the axioms of the theory $\mathcal{T}_{rt}$. SHADOW imports INTENSITY-SENSOR and encodes the theory $\mathcal{T}_{sh}$.

The next level specification, RECT-TRIAN-SHADOW, is the main fusion block in this whole specification. Here the two theories, RECT-TRIAN and SHADOW, are "glued" together along the common part - the real numbers. In the diagram of Figure 7 this common part is shown as the ONE-SORT specification. The sort defined in this specification serves as the common base that unifies the real numbers from the other two specifications. At the same time all the axioms of the two component specifications are mapped into one set of axioms. Then the sort and the operations of this specification are used to extend the colimit by adding additional axioms specified by the theory $\mathcal{T}_{sh}$.

6.3 Reasoning about the Fusion System

The specification described above can be used for reasoning about the fusion system being specified. For instance, we can reason about the goals of the system. Towards this end, we would have to submit candidate theorems (queries) to a theorem prover and ask whether they could be proven within the theory presented by the specification. In the stage of specification development, such queries could be submitted by either the users (customer side) or by the specification developers (developer side). First, one would need to choose one of the goals from $\mathcal{G}$. The preferable goals are $\{\neg TRN \wedge REC\}$ and $\{TRN \wedge \neg REC\}$, since

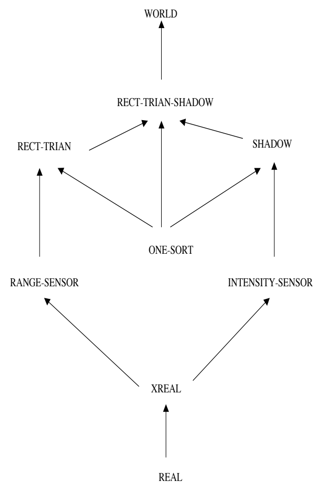

Figure 7: Diagram of the Fusion System

the success of one of these goals means a precise classification of the object in the scene. The goal $\{\neg TRN \wedge \neg REC\}$ is at the same level of detail. The goal $\{TRN \vee REC\}$ is less specific, since its success means that there is an object in the world, but it is not clear whether it is a rectangle or a triangle. In addition to goals, some information about the inputs, or ranges of inputs, would need to be entered into the system, in order for the prover to resolve the validity of a theorem. The goal is posted to the top level system, WORLD. Since this specification (theory) uses terms from the imported specification, the query is propagated down to that specification, and the process continues until all the truth values can be resolved.

Another application of such a specification is to use it for implementing the system. This can be achieved in the process called *refinement*. In this process, the specification goes through a number of refinement steps (called *interpretations*), the final step being translation into a programming language. Specware supports such a software development process.

Once the system is implemented, its operation can be understood as *model checking* (in logical terminology, (cf. [1]). If the system is implemented according to such a rigorous methodology, as can be easily checked, it will always derive correct decisions, i.e., it will be always right whether the world contains a triangle, a rectangle, one of them, or nothing. The system is not perfect, in the sense that in some situations it will not be able to recognize whether it is a rectangle or a triangle (it will simply say that there is an object in the world: rectangle or triangle). Nevertheless, it can be seen that the fused system will be more powerful than a system with only a range sensor, since it will be able to distinguish between a rectangle and a triangle in all the situations similar to the one shown in Figure 6.

7 Conclusions

In this paper we provided a formal definition of *fusion*. Fusion is treated as a formal operator that is applied to two families of objects, *theories* and their *classes of models* and returns a pair – a *fused theory* and a *a class of fused models*. The general fusion procedure consists of two parallel tasks one of the syntactical nature and the second of the semantical nature. Syntactic Theory Construction inputs a goal theory (with a goal formula in it), theories of sensors and background theories and constructs one fused theory for the whole system in which the goal sentence can be proved. Semantic Model Construction inputs models of the theories utilized in the Syntactic Theory Construction task and generates a class of models for the fused theory.

The goal of our research is to find various schemes for performing fusion and to find computationally efficient algorithms to achieve this goal. In this paper we showed an example of developing a fused theory, i.e., of the Syntactic Theory Construction. Since we used category theory as our mathematical basis, and Specware as our implementation tool, the correctness of the resulting specification and of the existence of the properties of the specification are guaranteed by the formal semantics of Specware and of the Specware theorem prover.

Formal specifications of fusion systems, like the one described in this paper, can serve two purposes. For one, we can reason about various properties of such specifications when we are specifying such systems. This is a very valuable feature, since errors discovered in the specification phase of system development are much cheaper to eliminate than in the later stages. For instance, the same error discovered after deployment of a system can cost hundreds of thousands times more. The other purpose is that such specifications can be transformed into code through the process of refinement. This process guarantees that the specification is implemented correctly. This does not imply, however, that the specification is correct, since this decision depends on the specifier and the user to make. However, having a formally defined specification certainly makes such a process much more reliable and robust.

References

[1] C. C. Chang and H. J. Keisler. *Model Theory.* North Holland, Amsterdam, New York, Oxford, Tokyo, 1992.

[2] D. L. Hall. *Mathematical Techniques in Multisensor Data Fusion.* Artech House, Boston - London, 1992.

[3] R. C. Luo and M. G. Kay. Multisensor integration and fusion in intelligent systems. *IEEE Transactions on Systems, Man and Cybernetics*, 19-5:901–931, 1989.

[4] Y. V. Srinivas and R. Jullig. Specware^{TM}: Formal support for composing software. Technical Report KES.U.94.5, Kestrel Institute, 1994.

[5] S. A. DeLoach and M. M. Kokar. Category theory approach to fusion of wavelet-based features. In *Proceedings of the Second International Conference on Information Fusion*, 1999.