Towards a Goal-Driven Autonomous Fusion System

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

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

Abstract *This paper addresses the issue of decision fusion when two (or more) sensors and the fusion center have a common language to represent queries and decisions, while each of the sensors has its own interpretation of the formulas of the language. Fusion is achieved through the model-theoretic operation of direct product of models. Since not all (most) formulas are not preserved under the product we need an decision procedure that tell us how to combine decisions from particular sensors into one fused decision. Towards this aim the notion of Galvin system is used. The operation of a decision procedure based on this approach is explained on simple examples. The validity of the solution is formally defined and proved in an appropriate theorem. The main advantages of the approach proposed in this paper are that the decision mechanism is generic, i.e., it can check the validity of any goal formula, and that it is provably correct.*

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

1 Introduction

In this paper we consider a case of decision fusion in which all sensors (two or more) derive decisions that are expressed in a language common to all the sensors. Even though it may seem like a very simple case, it is not quite so, because each of the sensors has its own interpretation of the terms of the language. In other words, for each sensor, there is a (different) model associated with the language. Consequently, the process of fusion (cf. [1]) requires that these different interpretations be taken into account when decisions from different sensors are fused.

We address this problem by fusing the interpretation structures (models) rather than just the decisions. In this paper we use the operation of *product* to combine structures [2]. Unfortunately, in such a case, even if both sensors derive the same decision, it is not necessarily preserved in the product of two models. For instance, the formula

$$\alpha(x, y) \equiv x \cdot y = 0 \Rightarrow (x = 0 \vee y = 0)$$

most typically does not hold in the product. To be more specific, consider two structures A and B such that $A = B = R$, i.e., both are real numbers with two operations - addition and multiplication under usual interpretation. The formula $\alpha(x, y)$ holds in both A and B, since either x or y must be 0 in order for $x \cdot y$ to be zero. We can say then that $A \models \alpha$ and $B \models \alpha$. In the product $A \times B$, however this is not the case. Note first that in the direct product $A \times B$, the zero element, 0, is represented by the pair $0(0, 0)$ and if $x = (x_1, x_2)$ and $y = (y_1, y_2)$ are any elements of $A \times B$ we have $x \cdot y = ((x_1, x_2) \cdot (y_1, Y_2)) = (x_1 \cdot y_1, x_2 \cdot y_2)$. It is easy to see that for $x = (0, 3)$ and $y = (5, 0)$ we have $x \cdot y = (0, 0)4$, but neither x or y are equal 0.

Horn formulas [3], on the other hand, are preserved under products. Even more, Keisler [4] proved that any formula that is preserved under reduced products is equivalent to a Horn formula. Some applications of Horn formulas can be found in [5, 6, 7].

Since, as we stated above, we are interested in fusion by products, our goal in this paper is to deal with more general types of formulas, not necessarily Horn formulas, and thus we need a decision procedure which will allow us to decide when a given formula is preserved in the product of two models. Moreover, our objective is to show how such a decision procedure can be derived automatically, i.e., how to construct *autonomous fusion systems* of this sort, given that the system knows the language in which decisions are expressed.

2 Problem Formulation

We are addressing here the problem of decision fusion. We assume that the goal of the fusion system is to derive a decision φ based upon decisions $\varphi_1, \ldots, \varphi_n$ obtained from n sensors (n decisions based on inputs from sensors). It is assumed that all sensors have the same language and that they interpret information in structures of the same kind of structure. In our example we assume even more - that carriers of models are the same, although in general it is not important. However, the semantical interpretations of the information can be very different. Our goal is to construct a decision procedure which will assert a formula whenever all the sensors report that some witness formulas holds.

We envision a hierarchical scenario in which there is a central fusion unit that collects inputs from all subordinate units (we call them sensors) and then the cenral unit makes a decision. The central unit can send various questions (queries) to the sensors. It is possible since both the central unit and the sensors speak the same language.

To better explain the problem we are addressing, we consider the follwing example scenario (see Figure 1). In this scenario, the goal is to recognize whether a detected object (house) has a gable roof oriented in the East-West direction. Two sensors, N and W (north and west) provide reports to the fusion unit. Suppose one of the terms in the language is $GableEW(x)$, which is one of the *goal formulas* of the system. The fusion center, F, can then send the query to the two sensors, W and N. Both sensors will interpret this formula in their own manner. Sensor W will reply "yes" (or "true") when it sees a triangle. Sensor N, on the other hand, will reply "yes" when it sees a rectangle. The fusion center, F, will conclude GableEW(roof) holds if both sensors say so.

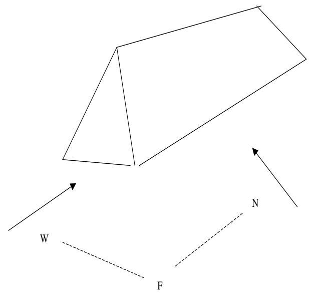

Figure 1: Decision Fusion Scenario

Notice that, as we mentioned above, in a genearal case such a decision procedure is not correct. Our goal in this paper is to propose a solution to this problem. More specifically, we will show how to decide about the truthfulness of such formulas. Since the procedure will allow for automatic answering of such queries, we call this procedure an *autonomous fusion system.*

3 Outline of the Solution

To construct an autonomous fusion system, we use the notion of *Galvin system* (cf. [8, 6]).

In the framework of Galvin systems, we show an algorithm for deriving the goal decision φ. In the first step, for any given goal formula φ, our system constructs a set of formulas S and an operation $\Pi : S^n \to S$. The operation Π asserts which of the goal formulas hold in the fused structure (in the product), given that formulas $\varphi_1, \ldots \varphi_n$ hold for each of the sensors. The algorithm then computes a set $T \subset S$, such that the goal formula φ is equivalent to the disjunction of all the formulas from T. In the next step, the algorithm computes $\bar{\varphi} = \Pi(\varphi_1, \ldots, \varphi_n)$ and checks whether this formula is in T. If it is, then the decision φ holds, otherwise it does not. It is clear that since $\bar{\varphi}$ is a disjunct in the set T, $\bar{\varphi}$ implies the goal formula φ.

Popular wisdom has it that when two sensors derive the same decision, then the decision must hold in reality. In this paper we show how to rationalize such a rule, i.e., when to accept such a rule in decision fusion and when not to. We have already showed simple examples of decision fusion for both when this popular wisdom rule should be used and when it should not be used. In the following we show how the Galvin system approach works on the scenario of Figure 1.

First of all, the fusion center F must select a query, or a *goal formula*. It is obviously related to its higher-level goals. In the next step it analyzes the syntax of the formula. Among others, it identifies *atomic formulas* within the goal formula. (Note, for now we are dealing only with open formulas, i.e., formulas without quantifiers.) In the case of $\varphi \equiv GableEW$, the goal formula consists of one atom. Based upon this analysis, F constructs the set S. In this case $S = \{GableEW, \neg GableEW\}$. In the case of open formulas, S contains all atomic formulas, all conjunctions, and their negations. In the next step, F constructs the mapping Π. In this example it is defined as:

$$\Pi(\varphi, \varphi) = \varphi$$

$$\Pi(\neg\varphi, \varphi) = \Pi(\varphi, \neg\varphi) = \Pi(\neg\varphi, \neg\varphi) = \neg\varphi$$

The set T in this case consists of only one element, $T = \{GableEW\}$. To make a specific decision, the system takes the answers to the query, computes the value of Π and checks whether the result is in T. In this example, the only case that a result is in T is when both sensors say "yes".

4 Proof of the Solution

In this section we present a formal definition of the Galvin information fusion system [8, 6] that was informally described above. We also provide a proof that a decision can be reached for any goal formula.

Definition 4.1 *A* Galvin Information Fusion System IFS *(with variables $v_1, \ldots v_n$) is a pair* (S, Π)*, where:*

(i) S is a finite set of formulas with variables in $\{v_1, \ldots v_n\}$.

(ii) Π is a commutative and associative operation on S (i.e. (S, Π) is a commutative semigroup).

(iii) For any structure $\mathfrak{A}$ and $a_1, \ldots a_n \in \mathfrak{A}$, there is exactly one formula $\alpha \in S$ such that $\mathfrak{A} \models \alpha[a_1, \ldots a_n]$.

(iv) For any structures $\mathfrak{A}$ and $\mathfrak{B}$ and any elements $a_1, \ldots a_n \in \mathfrak{A}$ and $b_1, \ldots b_n \in \mathfrak{B}$ if for some $\alpha, \beta \in S$ we have $\mathfrak{A} \models \alpha[a_1, \ldots a_n]$ and $\mathfrak{B} \models \beta[b_1, \ldots b_n]$ then $\mathfrak{A} \times \mathfrak{B} \models \Pi(\alpha, \beta)[\langle a_1, b_1\rangle, \ldots \langle a_n, b_n\rangle]$.

Theorem 4.2 — *For any goal formula $\varphi = \varphi(x_1 \ldots x_n)$ we can effectively construct a Galvin IFS (S, Π) with variables $x_1, \ldots x_n$ and a set $T \subseteq S$ such that $\bigvee T \leftrightarrow \varphi$ is a tautology.*

Proof: We will proceed by induction. Let φ be an open formula and $\varphi_1, \ldots, \varphi_k$ be all atomic formulas occurring in φ. Let S consists of $\mathbf{2}^k$ formulas of the form $\psi_1 \wedge \ldots \wedge \psi_k$, where each ψ_i is either φ_i or $\neg\varphi_i$. Suppose $\alpha = \psi'_1 \wedge \ldots \wedge \psi'_k$, $\beta = \psi'' \wedge \ldots \wedge \psi''_k$. Take $\Pi(\alpha, \beta) = \psi_1 \wedge \ldots \wedge \psi_k$ where $\psi_i = \varphi_i$ if $\psi'_i = \psi''_i = \varphi_i$ or $\psi_i = \neg\varphi_i$ otherwise. Now Π satisfies all of our requirements and it is easy to see that (S, Π)

is then a Galvin IFS and we can find the set $T \subseteq S$ such that $\bigvee T \leftrightarrow \varphi$ is a tautology.

Let us remark now that, if (S, Π) and T satisfy our theorem for φ, then (S, Π) and $S - T$ satisfy our theorem for $\neg\varphi$.

Let $\langle S_1, \Pi_1 \rangle$ and T_1 satisfy the theorem for φ_1 and let $\langle S_2, \Pi_2 \rangle$ and T_2 do the same for φ_2. Let us define $S = \{\alpha \wedge \beta : \alpha \in S_1 \text{ and } \beta \in S_2\}$, $T = \{\alpha \wedge \beta : \alpha \in T_1 \text{ or } \beta \in T_2\}$, $\Pi(\alpha_1 \wedge \beta_1, \alpha_2 \wedge \beta_2) = \Pi_1(\alpha_1, \alpha_2) \wedge \Pi_2(\beta_1, \beta_2)$. Then it is easy to check that (S, Π) and T satisfy the theorem for $\varphi_1 \vee \varphi_2$.

Finally let $\varphi = \exists x_0 \varphi_1$, and let (S_1, Π_1) and T_1 satisfy the theorem for φ_1. We will find (S, Π) and T for φ. Let $S = \{\alpha_X : X \subseteq S_1\}$ where $\alpha_X = \bigwedge\{\exists x_0 \gamma : \gamma \in X\} \wedge \bigwedge\{\neg\exists x_0 \gamma : \gamma \in S_1 - X\}$. Then it is easy to see that S satisfies the conditions (i) and (iii).

Let $\Pi(\alpha_X, \alpha_Y) = \alpha_Z$, where $Z = \{\Pi_1(\gamma, \delta) : \gamma \in X \text{ and } \delta \in Y\}$. Obviously Π satisfies (ii). We will show that Π satisfies (iv).

Indeed, suppose $\mathfrak{A} \models \alpha_X[a_1, \ldots, a_n]$ and $\mathfrak{B} \models \alpha_Y[b_1, \ldots, b_n]$. To prove (iv), we will show that for any $\beta \in S_1$, $\mathfrak{A} \times \mathfrak{B} \models (\exists x_0 \beta)[\langle a_1, b_1 \rangle, \ldots \langle a_n, b_n \rangle]$ iff $\beta \in Z$.

Let $\beta \in Z$, then there is $\gamma \in X$ and $\delta \in Y$ such that $\beta = \Pi_1(\gamma, \delta)$. Moreover since $\mathfrak{A} \models \alpha_X[a_1, \ldots, a_n]$, we have $\mathfrak{A} \models (\exists x_0 \delta)[a_1, \ldots, a_n]$ and in the same way $\mathfrak{B} \models (\exists x_0 \delta)[b_1, \ldots, b_n]$. Thus there are $a_0 \in A$ and $b_0 \in B$ such that $\mathfrak{A} \models \delta[a_0, \ldots, a_n]$ and $\mathfrak{B} \models \delta[b_0, \ldots, b_n]$. Consequently $\mathfrak{A} \times \mathfrak{B} \models \Pi_1(\gamma, \delta)[\langle a_0, b_0 \rangle, \ldots, \langle a_n, b_n \rangle]$, but $\beta = \Pi_1(\gamma, \delta)$,
whence $\mathfrak{A} \times \mathfrak{B} \models \beta[\langle a_0, b_0 \rangle, \ldots, \langle a_n, b_n \rangle]$ and $\mathfrak{A} \times \mathfrak{B} \models (\exists x_0 \beta)[\langle a_1, b_1 \rangle, \ldots, \langle a_n, b_n \rangle]$.

Conversely, let $\mathfrak{A} \times \mathfrak{B} \models (\exists x_0 \beta)[\langle a_1, b_1 \rangle, \ldots, \langle a_n, b_n \rangle]$. Then there is $\langle a_0, b_0 \rangle \in A \times B$ such that $\mathfrak{A} \times \mathfrak{B} \models \beta[\langle a_0, b_0 \rangle, \ldots, \langle a_n, b_n \rangle]$. Let $\gamma \in S_1$ be such that $\mathfrak{A} \models \gamma[a_0, \ldots a_n]$ and $\delta \in S_1$ be such that $\mathfrak{B} \models \delta[b_1, \ldots, b_n]$. Then $\mathfrak{A} \models (\exists x_0 \gamma)[a_1, \ldots, a_n]$ and by definition of α_X, $\gamma \in X$. In the same way $\delta \in Y$. Moreover $\Pi_1(\gamma, \delta) = \beta$, thus $\beta \in Z$ and Π satisfies (iv).

This completes the proof.

5 Conclusions

In this paper we addressed the issue of decision fusion. We assumed the situation in which various sensors and a central decision fusion system share a common language. More precisely, the syntax of the language is common to all parties. But the interpretation of particular symbols is different for each sensor. The fusion process is based on the model-theoretic operation of direct product of models. Accordingly, the fusion process takes all the decisions from the particular sensors and then derives a positive decision only if all the sensors agree. In this scenario, the fusion center must be able to send queries to the sensors. This query is dependent on its goal represented by a goal formula. In case of a complex query, the fusion center analyzes the structure of the goal formula and derives a decision derivation procedure. This decision process is based upon the notion of Galvin system. In this paper we showed simple examples of how such a system works. We also showed the formal derivation of the correctness of such procedure.

Decision fusion is important in many applications. The scenario in which particular sensors have their own interpretations is quite typical. For instance, a radar "sees" an object in a different way than a vision camera, an infrared camera or an ultrasound. All of these sensors, however, are used in target detection. Decisions from such sensors need to be fused. Typically, fusion algorithms are constructed around a specific set of goal formulas. If an additional formula needs to be added, the system must be redesigned and reimplemented. The approach presented in this paper resolves such a problem by proposing a generic decision mechanism that can check the validity of any goal formula. The main advantage of this mechanism is that, unlike heuristic rule-based approaches, it is provably correct.

References

[1] M. M. Kokar, J. A. Tomasik, and J. Weyman. A formal approach to information fusion. In *Proceedings of the Second International Conference on Information Fusion*, 1999.

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

[3] A. Horn. On sentences which are true of direct unions of algebras. *Journal of Symbolic Logic*, 16:14–21, 1951.

[4] H. J. Keisler. Reduced products and horn classes. *Trans. Amer. Math. Soc. Ser.2*, 81:307–328, 1965.

[5] J. A. Makovsky. Why horn formulas matter in computer science: Initial structure and generic examples. In *CAAP'85 Arbres en Algebre et Programmation 10*, pages 374–385, 1985.

[6] J. Tomasik. On products of neat structures. *Annals of Mathematical Logic*, 36:12–16, 1976.

[7] E. A. Palyutin, J. Saffe, and S. S. Starchenko. Models of superstable horn theories. *Algebra i Logika*, 24:278–326, 1985.

[8] F. Galvin. Horn sentences. *Annals of Mathematical Logic*, 1:389–422, 1970.