

Finite Satisfiability and Equivalence of Finitary Sketches

Frank Piessens

Report CW260, February 1998



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Finite Satisfiability and Equivalence of Finitary Sketches

*Frank Piessens**

Report CW 260, February 1998

Department of Computer Science, K.U.Leuven

Abstract

A translation from classical first-order logic to sketches is given. This translation does not preserve model categories, as is usually required when one studies connections between sketches and classical string-based logic. However, the translation preserves the models in some strong sense (although it does not preserve the homomorphisms between the models). In particular, it preserves satisfiability. Based on this translation, some undecidability results from classical logic can be transported to sketches. These undecidability results have practical relevance, because sketches have been proposed by a number of authors as a very convenient formalism for specifying databases.

*Postdoctoral Fellow of the Belgian National Fund for Scientific Research (N.F.W.O)

1 Introduction

The correspondence between sketches and classical logic has been investigated thoroughly. Usually, the expressive power of sketches and fragments of logic is compared by comparing their classes of model categories in **Set**. The locally presentable categories are exactly the model categories of both limit sketches, and limit theories. Moreover, this correspondence also holds when restricted to the finitary case: the locally finitely presentable categories are exactly the model categories of finitary limit sketches, and finitary limit theories. The locally accessible categories are exactly the model categories of both sketches and so-called *basic* theories. However this second correspondence does *not* hold when restricted to the finitary case. A finitely accessible category need not be axiomizable in a finitary basic theory, and a finitary basic theory may have a model category that is not finitely accessible. All these facts are treated in detail by Adámek and Rosický ([AR94]). For finitary sketches, it has been proved recently ([AR95a, AR95c, APJR96]) that they are equivalent (i.e. have the same model categories) to σ -coherent theories.

When using sketches to specify databases ([PS94, PS95, CD95, DCB95, CD94]), one is usually more interested in models in **FinSet**, instead of in models in **Set**. Adámek and Rosický ([AR95b]) have proved that the equivalence between finitary sketches and σ -coherent theories also holds when one considers models in **FinSet**.

It is clear that the correspondence between sketches and string-based logic is not very straightforward when one studies this correspondence based on the categories of models. In particular, there is no clear correspondence with classical finitary first-order logic. To transfer some of the results obtained on classical first-order logic, it would be interesting to have some kind of correspondence between classical (finitary) first-order logic and finitary sketches. In this paper, we describe such a correspondence. In particular, we describe a translation from classical first-order logic to finitary sketches, which is *satisfiability-preserving*. As a consequence, we obtain a number of undecidability results for sketches. From the point of view of databases, the most interesting results are the undecidability of the finite satisfiability problem, and the undecidability of the equivalence problem. These undecidability results are not surprising, but because of the practical relevance of the problems, it is interesting to see them proved.

The rest of this paper is structured as follows. In section 2, we recall some basic definitions concerning classical logic and sketches. Then, in section 3, we describe a translation from classical first-order logic to finite finitary sketches, and we prove that this translation preserves satisfiability. In section 4, we prove a number of undecidability results, based on this translation. Finally, in section 5, we discuss related work, and we summarize the most important results from the paper.

2 Logic and sketches

We first give precise definitions of classical finitary first-order logic and of sketches, and then we discuss the known connections between them. The definitions given in this section are all elementary, but since both classical logic and sketches can be presented in many different ways, we prefer to state the definitions clearly before we embark on constructing a translation between them.

2.1 Classical finitary first-order logic

We describe a minimalistic version of classical finitary first-order logic. A *logical language* is determined by giving a set of relation symbols, denoted $\sigma_1, \sigma_2, \dots$. Every relation symbol has an associated *arity*, which is a natural number. We assume that a countable set of *variables* is given. Variables are denoted x_1, x_2, \dots .

The *atomic formulae* of the language are syntactic expressions of the form $x_i = x_j$ (with x_i and x_j variables) or $\sigma(x_{k_1}, x_{k_2}, \dots, x_{k_n})$ where n is the arity of the relation symbol σ .

The *formulae* of the given logical language are built up from the atomic formulae using the logical combinators \neg, \wedge and the quantifier \exists . More precisely: all atomic formulae are formulae, and if ϕ is a formula, then $\neg\phi$ and $(\exists x)\phi$ are formulae, and if ϕ and ψ are formulae, then $\phi \wedge \psi$ is a formula. The other logical connectors and the universal quantifier can be defined in terms of the connectors and quantifier given above.

The *free* variables of a formula are defined in the standard way. A *sentence* is a formula without any free variables.

A *theory* in a given logical language is a set of formulae of that logical language.

An *interpretation* of a logical language consists of a set U , and for each relation symbol σ of arity n an n -ary relation on U . Given such an interpretation, any formula ϕ with free variables x_1, \dots, x_n defines an n -ary relation on U , called the *extension* of ϕ in the interpretation. We say the formula is *satisfied* if this extension is the total relation (i.e. is the n -th cartesian power of U). A *model* of a theory is an interpretation, such that each formula of the theory is satisfied. A theory is *satisfiable* if there exists a model.

2.2 Finitary sketches

We stick to the definition of sketch, given by Barr and Wells in [BW90]. Hence, a sketch is a quadruple $(\mathcal{G}, \mathcal{D}, \mathcal{L}, \mathcal{K})$, with \mathcal{G} a graph, \mathcal{D} a set of diagrams in \mathcal{G} , \mathcal{L} a set of cones in \mathcal{G} and \mathcal{K} a set of cocones in \mathcal{G} . The sketch is *finitary* if each of the cones and cocones in \mathcal{L} and \mathcal{K} is finite. The sketch is *finite* if \mathcal{G} and each of the sets $\mathcal{D}, \mathcal{L}, \mathcal{K}$ is finite.

A *model* of a sketch $(\mathcal{G}, \mathcal{D}, \mathcal{L}, \mathcal{K})$ in a category \mathcal{C} (e.g. **Set** or **FinSet**), is a graph homomorphism from \mathcal{G} to \mathcal{C} , taking each diagram in \mathcal{D} to a commutative diagram, each cone in \mathcal{L} to a limit cone, and each cocone in \mathcal{K} to a colimit cocone.

We will freely use the notational conventions for sketches described in the textbook by Barr and Wells ([BW90]). In particular, this means that if we introduce an object in the graph \mathcal{G} , with name $a_1 \times a_2 \times \dots \times a_n$, then this implies that there are already objects with names a_1 up to a_n , and moreover this implies that there is a cone $p_i : a_1 \times a_2 \times \dots \times a_n \rightarrow a_i$ in \mathcal{L} . The name a^n is an abbreviation for $a \times a \times \dots \times a$ (n times). Moreover, if there is an arrow in \mathcal{G} , decorated with a tip in the front (like $f : A \twoheadrightarrow B$), this implies that the following cone is in \mathcal{L} (i.e. this implies f is mono in every model):

$$\begin{array}{ccccc}
 & & A & & \\
 & \swarrow \text{Id} & \downarrow f & \searrow \text{Id} & \\
 A & & B & & A \\
 & \xrightarrow{f} & & \xleftarrow{f} &
 \end{array}$$

2.3 Connections between string-based logic and sketches

As already discussed in the introduction, the connections between sketches and classical, string-based logic, are usually studied by looking at the model categories of specifications in both formalisms. A class of sketches and a fragment of logic are said to be equivalent if they give rise to the same class of model categories. For example, in this sense, sketches are equivalent to (infinitary) basic theories.

In the rest of this paper, we will describe a correspondence between sketches and classical first-order logic from a different point of view. We will describe a translation from a first-order theory to a sketch with the property that every model of the theory defines a model of the sketch and vice-versa. Hence, in a sense, the model categories have ‘the same objects’, even though they don’t have ‘the same morphisms’. In particular, the model category of the logical theory is empty iff the model category of the corresponding sketch is empty. Or, in other words, the translation preserves satisfiability.

3 A satisfiability-preserving translation

Given a logical theory \mathcal{T} in a logical language \mathcal{L} , we will construct a corresponding finitary sketch $\mathcal{S}_{\mathcal{T}}$. The construction happens in three stages. First, we construct the sketch corresponding to a logical language. Then, we show how logical formulae can be represented in the sketch. And finally, we show how a logical formula can be required to be satisfied in every model.

3.1 The sketch corresponding to a logical language

We start with proving the (obvious and easy) fact that interpretations of a logical language can be captured by means of a sketch.

Lemma 1 *For every logical language \mathcal{L} , there exists a finitary sketch $\mathcal{S}_{\mathcal{L}}$ such that every interpretation of \mathcal{L} defines a model of $\mathcal{S}_{\mathcal{L}}$, and vice-versa.*

In fact, we could say something stronger: the category of interpretations of \mathcal{L} is equivalent to the category of models of $\mathcal{S}_{\mathcal{L}}$.

Proof: Let \mathcal{L} consist of the relation symbols σ_i , and let a_i be the arity of σ_i . Let A be the set of all the a_i . The sketch $\mathcal{S}_{\mathcal{L}}$ will have the following objects:

- An object U .
- An object U^k for all $k \in A$.
- An object S_{σ_i} for each relation symbol σ_i .

(Recall our notational conventions for objects: the fact that an object U^k is introduced implies that the necessary arrows and cones making this the k -th power of U are also introduced.)

The sketch has the following arrows:

- An arrow $\sigma_i : S_{\sigma_i} \twoheadrightarrow U^{a_i}$ for each relation symbol σ_i .

(Recall our notational conventions for arrows: for each arrow introduced above, a cone will be added stating that this arrow must be mono)

A model of this sketch in **Set** will assign a set to U , and a subset of the a_i -th power of U to each S_{σ_i} . In other words, models of the sketch $\mathcal{S}_{\mathcal{L}}$ and interpretations of the logical language \mathcal{L} are essentially the same. \square

Note that, if \mathcal{L} is finite, then $\mathcal{S}_{\mathcal{L}}$ will also be finite. Note also that finite models of \mathcal{L} define finite models of $\mathcal{S}_{\mathcal{L}}$ and vice-versa.

3.2 Representing a logical formula in a sketch

Now, given a formula ϕ in a logical language \mathcal{L} , we will show how we can extend the sketch $\mathcal{S}_{\mathcal{L}}$ (in a conservative way) so that it contains an arrow $\widehat{\phi}$ which is taken to the extension of ϕ in every model. More precisely, we prove:

Lemma 2 *For every formula ϕ in a logical language \mathcal{L} , there exists a finitary sketch $\mathcal{S}_{\mathcal{L}}^{\phi}$ such that every interpretation I of \mathcal{L} defines a model M of $\mathcal{S}_{\mathcal{L}}^{\phi}$ and vice-versa. Moreover, $\mathcal{S}_{\mathcal{L}}^{\phi}$ contains an arrow $\widehat{\phi} : [\phi] \rightarrow U^k$ (where k is the number of free variables of ϕ) such that this arrow is taken by M to the extension of ϕ in I .*

Proof: Given lemma 1, it suffices to prove that the extension of a formula ϕ can be computed by (finitary) limits and colimits, starting from the arrows $\sigma_i : S_{\sigma_i} \rightarrow U^{a_i}$ representing the interpretation of the relation symbols of the logical language. We show by induction on the size of the formula ϕ that this can be done. We proceed as follows. We assume that a model of the sketch $\mathcal{S}_{\mathcal{L}}$ (or equivalently, an interpretation of the underlying logical language of \mathcal{T}) is given. For any formula ϕ with n free variables, we will compute an arrow $\widehat{\phi} : [\phi] \rightarrow U^n$ together with an isomorphism l_{ϕ} from the set $\mathbf{n} = \{0, 1, \dots, n-1\}$ to the set of free variables of ϕ . This arrow $[\phi]$ represents the extension of the formula ϕ in the following sense: Let μ be a mapping of the free variables of ϕ to U (the universe of the given interpretation); ϕ is satisfied by this mapping iff the arrow $\langle \mu(l_{\phi}(0)), \mu(l_{\phi}(1)), \dots, \mu(l_{\phi}(n-1)) \rangle : 1 \rightarrow U^n$ factors through $\widehat{\phi}$. The computation of the arrow $\widehat{\phi}$ will start from the interpretation of the arrows of the sketch $\mathcal{S}_{\mathcal{L}}$ in the given model of $\mathcal{S}_{\mathcal{L}}$, and moreover, the computation will be done using only finitary limits and colimits.

Atomic formulae

Consider an atomic formula $\sigma(x_{k_1}, \dots, x_{k_r})$ where σ is a relation symbol of arity r . Let the set of free variables of this formula have cardinality n . Choose an arbitrary isomorphism $l_{\sigma(x_{k_1}, \dots, x_{k_r})} : \mathbf{n} \rightarrow \text{FV}(\sigma(x_{k_1}, \dots, x_{k_r}))$ (we will abbreviate the name of this morphism to l). The arrow $\sigma(x_{k_1}, \dots, x_{k_r})$ is computed by the following pullback.

$$\begin{array}{ccc} [\sigma(x_{k_1}, \dots, x_{k_r})] & \xrightarrow{\quad} & S_{\sigma} \\ \sigma(x_{k_1}, \dots, x_{k_r}) \downarrow & & \downarrow \sigma \\ U^n & \xrightarrow{\langle l^{-1}(x_{k_1}), \dots, l^{-1}(x_{k_r}) \rangle} & U^r \end{array}$$

It is straightforward to verify that this arrow indeed represents the extension of $\sigma(x_{k_1}, \dots, x_{k_r})$.

Next consider the atomic formula $x_i = x_j$. We must consider two cases. In the first case, x_i and x_j are distinct variables, and hence we have a formula on two free variables. Let $l_{x_i = x_j}$ be the isomorphism mapping 0 to x_i and 1 to x_j . The following arrow represents the formula:

$$[x_i = x_j] = U \xrightarrow{(\text{Id}, \text{Id})} U^2$$

In the second case, x_i and x_j are the same variable, say x . Let $l_{x=x}$ be the only possible isomorphism from 1 to $\{x\}$. The following arrow represents the formula:

$$[x = x] = U \xrightarrow{\text{Id}} U$$

Other formulae

The formula $\neg\phi$. Suppose l_ϕ is the isomorphism from \mathbf{n} to the set of free variables of ϕ . $\neg\phi$ has the same set of free variables, and we take $l_{\neg\phi}$ to be equal to l_ϕ . By induction, we assume we have an arrow $\widehat{\phi}$ representing ϕ . We take $\widehat{\neg\phi}$ to be the complement of this arrow (and this complement is unique up to isomorphism in **Set** or **FinSet**). This complement is completely determined in **Set** or **FinSet** by requiring the following cocone to be a sum-cocone.

$$\begin{array}{ccc} & U^n & \\ \widehat{\phi} \nearrow & & \nwarrow \widehat{\neg\phi} \\ [\phi] & & [\neg\phi] \end{array}$$

It is again easy to prove that $\widehat{\neg\phi}$ correctly represents the formula $\neg\phi$. Note however that, when we will use this construction in a sketch, this will strongly influence the homomorphisms of models of the sketch. Although the sketch will have in essence the same models as the logical theory, there will be much less homomorphisms between models of the sketch than there are homomorphisms between models of the logical theory.

The formula $(\exists x)\phi$. Again we have to consider two cases. The first case is the case where x is a free variable of ϕ . In that case, if ϕ has n free variables, $(\exists x)\phi$ will have $m = n - 1$ free variables. Now let l_ϕ be the isomorphism from \mathbf{n} to the free variables of ϕ , and let i be the inclusion of $\text{FV}((\exists x)\phi)$ into $\text{FV}(\phi)$. Choose $l_{(\exists x)\phi}$ to be an arbitrary isomorphism from \mathbf{m} to $\text{FV}((\exists x)\phi)$. Let j be the (unique) morphism making the following diagram commute.

$$\begin{array}{ccc} \mathbf{m} & \xrightarrow{j} & \mathbf{n} \\ l_{(\exists x)\phi} \downarrow & & \downarrow l_\phi \\ \text{FV}((\exists x)\phi) & \xrightarrow{i} & \text{FV}(\phi) \end{array}$$

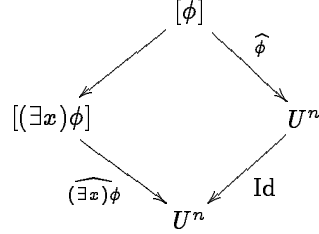
Then the arrow $\widehat{(\exists x)\phi}$ is defined by means of the following diagram:

$$\begin{array}{ccc} & [\phi] & \\ & \swarrow & \searrow \widehat{\phi} \\ [(\exists x)\phi] & & U^n \\ \widehat{(\exists x)\phi} \searrow & & \swarrow (p_{j(0)}, \dots, p_{j(m-1)}) \\ & U^m & \end{array}$$

In this diagram, we define the lefthandside to be the epi-mono factorization of the righthandside. In **Set** or **FinSet**, this factorization is unique. Moreover, we can state using only limits and colimits that an arrow must be epi or that an arrow must be mono. The verification that the arrow $\widehat{(\exists x)\phi}$ correctly represents the formula $(\exists x)\phi$ is left to the reader.

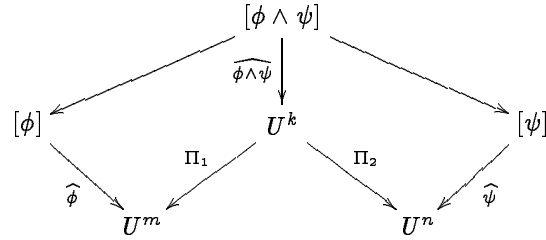
The second case we have to consider for the formula $(\exists x)\phi$, is the case where x is not a free variable of ϕ . In this case, we define the arrow $\widehat{(\exists x)\phi}$ using the

following diagram:



Again, the lefthandside of this diagram is required, by means of limit cones and colimit cocones, to be the epi-mono factorization of the righthandside.

The formula $\phi \wedge \psi$. Let l_ϕ be the isomorphism from \mathbf{m} to $\text{FV}(\phi)$ and l_ψ the isomorphism from \mathbf{n} to $\text{FV}(\psi)$. Let k be the number of free variables of $\phi \wedge \psi$. Choose an arbitrary isomorphism $l_{\phi \wedge \psi}$ from \mathbf{k} to $\text{FV}(\phi \wedge \psi)$. Let i_1 be the inclusion of $\text{FV}(\phi)$ into $\text{FV}(\phi \wedge \psi)$ and i_2 be the inclusion of $\text{FV}(\psi)$ into $\text{FV}(\phi \wedge \psi)$. Let $j_1 : \mathbf{m} \rightarrow \mathbf{k}$ be $l_{\phi \wedge \psi}^{-1} \circ i_1 \circ l_\phi$ and $j_2 : \mathbf{n} \rightarrow \mathbf{k}$ be $l_{\phi \wedge \psi}^{-1} \circ i_2 \circ l_\psi$. The arrow representing $\phi \wedge \psi$ is defined using the following diagram.



where $\Pi_1 = \langle p_{j_1(0)}, \dots, p_{j_1(m-1)} \rangle$ and $\Pi_2 = \langle p_{j_2(0)}, \dots, p_{j_2(n-1)} \rangle$. This diagram is required to be a limit cone. It is again left to the reader to verify that the arrow $\widehat{\phi \wedge \psi}$ correctly represents the formula $\phi \wedge \psi$.

This completes the proof that we can compute the extension of a formula using only limit and colimit constraints. \square

3.3 The sketch corresponding to a logical theory

Theorem 1 *For every finite logical theory \mathcal{T} , there exists a finite finitary sketch $\mathcal{S}_{\mathcal{T}}$, such that every model of \mathcal{T} defines a model of $\mathcal{S}_{\mathcal{T}}$ and vice-versa. Moreover, every finite model of \mathcal{T} defines a finite model of $\mathcal{S}_{\mathcal{T}}$ and vice-versa.*

Contrary to the situation in lemma 1, it is *not* true here, that the model categories of \mathcal{T} and $\mathcal{S}_{\mathcal{T}}$ are equivalent. The theorem does generalize to the infinite case, but since we do not need this generalization, this will not be proved.

Proof: Every finite logical theory is equivalent to a theory containing only one sentence (first take the universal closure of all formulae in the theory, and then combine all the resulting sentences by taking their conjunction). Let this sentence be ϕ . Construct the sketch $\mathcal{S}_{\mathcal{L}}^\phi$ whose existence is assured by lemma 2. By lemma 2, the object $[\phi]$ will be taken to the empty set if ϕ is not satisfied in a model, and to the terminal set if ϕ is satisfied. Hence, to construct $\mathcal{S}_{\mathcal{T}}$, it suffices to add one cone to $\mathcal{S}_{\mathcal{L}}^\phi$ stating that the object $[\phi]$ must be taken to the terminal object. The proof of the last sentence of the theorem is trivial. \square

4 Finite satisfiability and equivalence of finite finitary sketches

The satisfiability preserving translation (theorem 1) is used to prove the undecidability of a number of interesting problems.

4.1 Finite satisfiability of a finite finitary sketch is undecidable

First recall that finite satisfiability of a finite first-order theory is undecidable. A proof of this result can be found in [AHV95]. Note that this result is quite different from the classical result that satisfiability (allowing infinite models) of a first-order theory is undecidable. Satisfiability of a first-order theory is co-semi-decidable, whereas finite satisfiability is semi-decidable.

Given theorem 1, we immediately obtain as a corollary:

Corollary 1 *The problem of determining whether a finite finitary sketch has a finite model is undecidable.*

Proof: Suppose this problem was decidable. Given any finite first-order theory, we can construct (theorem 1) a corresponding finite finitary sketch with the property that this sketch is finitely satisfiable iff the original first-order theory was finitely satisfiable. Hence, we could decide the finite satisfiability problem for classical first-order logic. But this is known to be undecidable. Contradiction. \square

4.2 Equivalence of two finite finitary sketches is undecidable

A number of papers ([PS94, PS95, Pie96, CD95, DCB95, CD94]) have proposed sketches or similar formalism as suitable specification formalisms for semantic data modelling. The *equivalence problem* in semantic data modelling is the problem of deciding whether two semantic data specifications have the same semantic content, i.e. whether they are both specifications of the same mini-world. In [Pie96], a number of possible formal definitions for semantical equivalence of semantic data specifications were studied, and one of the proposed definitions stated that semantical equivalence amounts to equivalence of model categories. Hence, the problem of deciding whether two given semantic data specifications are semantically equivalent can be reduced to the question whether two finite finitary sketches have equivalent model categories. Based on corollary 1 proved above, we can easily prove that this is an undecidable problem.

Theorem 2 *The (finite) equivalence problem for finite finitary sketches is undecidable.*

The finite equivalence problem is the problem of determining whether two sketches have equivalent model categories in **FinSet**.

Proof: It is easy to construct a finite finitary sketch that is not satisfiable. Take for example the sketch with one object, a cone stating that the object is terminal, and a cocone stating that the object is initial. Another finite finitary sketch will be equivalent to this sketch iff it has no finite models. But this is undecidable by corollary 1. \square

It remains an interesting open problem to characterize interesting subclasses of sketches that do have a decidable equivalence problem. In [PS94, PS95, Pie96], a number of such subclasses of sketches have been identified, but the question remains whether these subclasses can be enlarged.

It should also be noted that the equivalence problem is in general much harder than the satisfiability problem. For instance, for finite limit sketches satisfiability is a trivial problem (every finite limit sketch is satisfiable), but the equivalence problem is undecidable.

5 Conclusions and Related work

More and more authors see the relevance of categorical methods in the field of database theory. Restricting myself to authors who have proposed sketch-like formalisms for specifying databases, there is the work of Diskin ([CD95, DCB95, CD94]), who proposes generalized sketches in the sense of Makkai ([Mak93]) as a framework for schema integration, the work of Tuijn ([Tui94]), who emphasizes the definition of a query language in terms of universal constructions, and the work of the current author ([PS94, PS95, Pie96]), who studied the semantical equivalence problem for semantic data models using sketches as the specification formalism.

For applications of sketches in database theory, it is often interesting to see results on finite models of finitary sketches (because, in fact, an instance of a database can be seen as a finite model of a finitary sketch). In a similar way, the study of finite model theory of classical first-order logic was also inspired by problems from the field of databases ([AHV95]). Finite models of finitary sketches have only recently received some attention of the category theory community. The work of Adámek and Rosický ([AR95a, AR95c, AR95b]) is very relevant for the semantical equivalence problem in database theory.

In this paper, we have established a link between classical first-order logic and finitary sketches, allowing us to transport some of the results of finite model theory to sketches. Because the translation only works in one direction (from classical logic to sketches), it is only possible to transport negative results. In particular, we have proven some practically very interesting problems to be undecidable. It remains an open problem whether some interesting translation from sketches to (some finitary extension) of classical first-order logic can be defined.

References

- [AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley Publishing Company, 1995.
- [APJR96] J. Adámek, P.T.Johnstone, J.M.Makowsky, and J. Rosický. Finitary sketches. preprint, 1996.
- [AR94] Jiří Adámek and Jiří Rosický. *Locally Presentable and Accessible Categories*. Cambridge University Press, 1994.
- [AR95a] Jiří Adámek and Jiří Rosický. Finitary sketches. to appear in Journal of Symbolic Logic, 1995.
- [AR95b] Jiří Adámek and Jiří Rosický. Finite models of sketches. preprint, 1995.
- [AR95c] Jiří Adámek and Jiří Rosický. On geometric and finitary sketches. to appear in Applied Categorical Structures, 1995.
- [BW90] Michael Barr and Charles Wells. *Category Theory for Computing Science*. Prentice-Hall International Series in Computer Science. Prentice-Hall International, New York, 1990.

- [CD94] Boris Cadish and Zinovy Diskin. Algebraic graph-oriented = category-theory-based: Categorical data modelling manifesto. Technical report, Frame Inform Systems, Database Design Laboratory, Latvia, July 1994.
- [CD95] B. Cadish and Z. Diskin. Algebraic graph-based approach to management of multibase systems, I: Schema integration via sketches and equations. In *Next Generation of Information Technologies and Systems, NGITS'95*, pages 69–79, 1995.
- [DCB95] Z. Diskin, B. Cadish, and I. Beylin. Algebraic graph-based approach to management of multibase systems, II: Algebraic aspects of schema integration. To appear in the proceedings of the Moscow ACM Chapter Conference ADBIS'95, 1995.
- [Mak93] Michael Makkai. Generalized sketches as a framework for completeness theorems, 1993. Preprint, McGill University.
- [MP90] Michael Makkai and R. Paré. *Accessible Categories: the Foundations of Categorical Model Theory*, volume 104 of *Contemporary Mathematics*. American Mathematical Society, 1990.
- [MR77] Michael Makkai and Gonzalo E. Reyes. *First Order Categorical Logic*, volume 611 of *Lecture Notes in Mathematics*. Springer-Verlag, 1977.
- [Pie96] Frank Piessens. *Semantic Data Specifications: an Analysis Based on a Categorical Formalization*. PhD thesis, Katholieke Universiteit Leuven, Department of Computer Science, February 1996.
- [PS94] Frank Piessens and Eric Steegmans. Canonical forms for data-specifications. In *Proceedings of Computer Science Logic 94*, number 933 in *Lecture Notes in Computer Science*, pages 397–411. Springer-Verlag, 1994.
- [PS95] Frank Piessens and Eric Steegmans. Categorical data-specifications. *Theory and Applications of Categories*, 1:156–173, 1995.
- [Tui94] Chris Tuijn. *Data Modeling from a Categorical Perspective*. PhD thesis, Universitaire Instelling Antwerpen, 1994.