A Short Survey on the State of the Art in Matching and Unification Problems

P.Raulefs, J. Siekmann, P.Szabó, E.Unvericht

P. Raulefs
Institut für Informatik III
Universität Bonn
Kurfürstenstr. 74
D-5300 Bonn 1

J. Siekmann, P.Szabó, E.Unvericht
Institut für Informatik I
Universität Karlsruhe
Postfach 6380
D-7500 Karlsruhe 1

1. Motivations

There is a wide variety of areas where matching and unification problems arise:

(1.1) *Databases*

The user of a *(relational) database* [22] may logically AND the properties she wants to retrieve or else she may be interested in the NATURAL JOIN [17] of two stored relations. In neither case, she would appreciate if she constantly had to take into account that AND is an associative and commutative operation, or that NATURAL JOIN obeys an associative axiom, which may distribute over some other operation [68].

(1.2) *Algebra*

In *algebra*, a famous decidability problem, which inspite of many attacks remained open for over twenty-five years, has only recently been solved: the monoid problem (also called Löb's Problem in Western Countries, Markov's Problem in Eastern Countries and the Stringunification Problem in Automatic Theorem Proving [33],[34],[35],[69],[60], [48],[73]) is the problem to decide whether or not an equation system over a free semigroup possesses a solution. Last year this problem has been shown to be decidable [51]. The monoid problem has important practical applications inter alia for Automatic Theorem Proving (stringunification [69] and second order monadic unification [90],[38]) for Formal Language Theory (the crossreference problem for van Wijngaarden Grammars [88]), and for pattern directed invocation languages in artificial intelligence (see below).

(1.3) *Information retrieval*

A patent office may store all recorded electric circuits [11] or all recorded chemical compounds [79] as some graph structure, and the problem of checking whether a given circuit or compound already exists is an instance of a test for graph isomorphism [82],[83],[20]. More generally, if the nodes of such graphs are labelled with universally quantified variables ranging over subgraphs, these problems are practical instances of a *graph matching problem.*

(1.4) *Computer vision*

In the field of *computer vision* it has become customary to store the internal representation of certain external scenes as some net structure [14],[89]. The problem to find a particular object - also represented as some net - in a given scene is also an instance of the *graph matching problem* [61]. Here one of the main problems is to specify as to what constitutes a successfull match (since a strict test for endomorphism is too rigid for most applications) although serious investigation of this problem is still pending.

(1.5) *Computer algebra*

In *computer algebra (*or *symbol manipulation)* [72] matching algorithms also play an important rôle: for example the integrand in a symbolic integration problem [55] may be matched against certain patterns in or-

der to detect the class of integration problems it belongs to and to trigger the appropriate action for a solution (which in turn may involve several quite complicated matching attempts [56],[52],[7],[13],[23], [29].

(1.6) *Theorem proving*

All present day *theorem provers* have a procedure to unify first order terms as their essential component: i.e. a procedure that substitutes terms for the universally quantified variables until the two given terms are symbolwise equal or the failure to unify is detected. Unification algorithms for such first order terms have independently been discovered by [63],[4] and [44]. Recently this work has been extended to unification problems involving additional axioms. As we shall argue that the theoretical framework for these extended problems may provide a basis of investigation to all the matching problems mentioned in this introduction, a more detailed account is given below.

(1.7) *Programming languages*

An important contribution of artificial intelligence to programming language design is the mechanism of *pattern-directed* invocation of procedures [31],[67],[9],[32], [86]. Procedures are identified by patterns instead of procedure identifiers as in traditional programming languages. Invocation patterns are usually designed to express goals achieved by executing the procedure. Incoming messages are tried to be matched against the invocation patterns of procedures in a procedural data base, and a procedure is activated after having completed a successful match between message and pattern. So,matching is done (1) for looking up an appropriate procedure that helps to accomplish an intended goal, and (2) transmitting information to the involved procedure.

For these applications it is particularly desirable to have methods for matching objects belonging to high level data structures such as strings, sets, multisets etc. Algorithms have been designed for some data structures, but often completeness, minimality or sometimes even correctness has not been shown.

A little reflection will show that for very rich matching structures, as it has e.g. been proposed in MATCHLESS in PLANNER [31], the matching problem is undecidable. This presents a problem for the designer of such languages: on the one hand, very rich and expressive matching structures are desirable, since they form the basis for the invocation and deduction mechanism. On the other hand, drastic restrictions will be necessary if matching algorithms are to be found. The question is just how severe do these restrictions have to be.

2. Matching Problems

All the above applications share a problem, which in its most abstract form is as follows:

> *Suppose two expressions s and t are given, which by some convention denote a particular structure and let s and t contain some free variables. We say s and t match iff there are expressions to be substituted into the free variables of both terms such that s and t become equal.*

We write <s;t> for a matching problem. In addition we may want to denote the set of structures explicitely as S and also we may want to provide additional information on what we mean by two expressions in S being equal or not. Say this information is contained in a set of equations in Π, then a general matching problem is denoted as the triple:

$$\mathfrak{M} := \langle\langle s;t\rangle, S, \Pi\rangle$$

[71] contains the beginnings of a theoretical framework within which such matching problems could be analyzed.

Since this framework is heavily based on the theory of unification problems in automatic theorem proving as developed by G. Plotkin [60] and G. Huet [40], we shall present a brief account of it below.

3. The Current State of the Art

For almost as long as attempts at proving theorems by machines have been made, a critical problem has been well known [6],[19], [57]: Certain equational axioms, if left without precautions in the data base of an automatic theorem prover (ATP), will force the ATP to go astray. In 1967, Robinson [64] proposed that substantial progress ("a new plateau") would be achieved by removing these troublesome axioms from the data base and building them into the deductive machinery.

Four approaches to cope with equational axioms have been proposed:

(1) To write the axioms into the data base, and use an additional rule of inference, such as paramodulation [66].
(2) To use special "rewrite rules" [44], [1],[41],[59].
(3) To design special inference rules incorporating these axioms [74].
(4) To develop special unification algorithms incorporating these axioms [60].

At least for equational axioms, the last approach (4) appears to be most promising, however it has the drawback that for every new set of axioms a new unification algorithm has to be found. Also recently there has been interesting work on combinations of approach (2) and (4) [59].

The theoretical basis for utilizing unification algorithms incorporating equational axioms has been developed by G. Plotkin [60]. Plotkin has shown that whenever an ATP is to be refutation complete, its unification procedure must satisfy conditions given as follows: Assume Π is the theory (set of axioms) considered, and t_1, t_2 are terms to be unified; then, the following properties should hold for the set Σ of unifiers of t_1 and t_2:

1. Σ is correct, i.e. for every $\sigma \in \Sigma$, $\sigma t_1 \overset{\Pi}{=} \sigma t_2$. ($\overset{\Pi}{=}$ denotes equality with respect to the theory Π).
2. Σ is complete, i.e. for any substitution δ with $\delta t_1 \overset{\Pi}{=} \delta t_2$ there is some $\sigma \in \Sigma$ s.t. there exists a substitution λ so that $\delta \overset{\Pi}{=} \lambda \circ \sigma$.
3. Σ is minimal, i.e. no unifier σ_1 in Σ is an instance of some other unifier σ_2 in Σ.

Minimality is a property that has often been overlooked in the literature:

If minimality is completely ignored we arrive at simply enumerating all substitutions and removing all that do no unify as an algorithm satisfying our requirements. Generating such a set of all unifiers, instead of a set of *most general* unifiers, essentially amounts to proving a theorem by the 'British Museum Algorithm' (i.e. by enumerating all Herbrand instances). Such procedures are called conservative in [66] and are distinctively different from proofs by the resolution principle at the 'lifted' most general level. However, minimality is more difficult to achieve than correctness and completeness.

Looking at unification of terms in first-order predicate calculus with an equational theory Π, unification problems may be classified with respect to the cardinality of minimal and complete sets Σ of unifiers.

(i) Σ may always be a singleton: e.g. for $T = \emptyset$, that is for ordinary first order unification as in [63],[44],[6]. Another (trivial) problem in this class is the string matching problem for constant strings only, as encountered in string manipulation languages such as SNOBOL [24]. The nontrivial aspect of this problem is to find *efficient* algorithms [45],[25]. Another example is unification under homomorphism, isomorphism and automorphism [85].

(ii) Σ may have more than one element but at most finitely many: examples are the theory of idempotence as well as idempotence plus commutativity [62]. Other examples are unification under commutativity [70]; unification under associativity and commutativity [49],[78]; unification under associativity, commutativity and idempotence [49] and the one way unification problem for strings.

(iii) Σ may sometimes be an infinite set: examples are unification under associativity [69],[48],[60].

This problem is equivalent to the problem of solving a set of equations over a free semigroup (the monoid problem) [53]. Other problems in this class are unification under distributivity [81] and the unification problem for second order monadic logic [90],[38].

(iv) Σ may sometimes not exist at all,e.g. for unification in ω-order predicate calculus. In such cases there exist infinite chains of unifiers (ordered by increasing generality)

$$\delta_1 \subseteq \delta_2 \subseteq \delta_3 \subseteq \cdots$$

with no upper bound [27],[38],[42].

For unification problems where complete sets of unifiers are always finite, it is not necessarily important that the unification procedure returns a minimal set of unifiers, since dependent unifiers can always be checked off. In this case, minimality of the unification procedure comes down to be a matter of computational efficiency.

The significance of this work for other fields derives from the fact that certain axioms in the theory Π define structures which closely resemble familiar datastructures: if S is the set of first order predicative terms and Π is empty we have the usual tree structure. If S is as above and Π contains the associativity axiom (A) we have strings. If S is as above and Π contains A and the commutativity axiom (C) or A+C+I, where I is an idempotence axiom we have multisets (bags) or sets respectively.

The following chart provides a quick survey of the state of the art at the time of writing.

The axioms are:

A	(associativity)	$f(f(x,y),z)=f(x,f(y,z))$
C	(commutativity)	$f(x,y)=f(y,x)$
D	(distributivity)	$f(x,g(y,z))=g(f(x,y),f(x,z))$
		$f(g(x,y),z)=g(f(x,z),f(y,z))$
H	(homomorphism)	$\varphi(x\bullet y)=\varphi(x)*\varphi(y)$
I	(idempotence)	$f(x,x)=x$

Axioms	Problem decidable?	card(Σ)	rep. Alg.is minimal	Data-structure	Investigators
A	yes	∞	yes	strings	[33],[69], [48],[51]
C	yes	fin.	no	unordered trees (lists)	[70]
I	yes	fin.	no		[62]
A+C	yes	fin.	yes	finite multisets	[78],[49]
A+I	?	?	?		in progress
C+I	yes	fin.	no		[62]
A+C+I	yes	fin.	yes	finite sets	[49]
D	?	∞	?		[81]
D+A	no	∞	?		[80]
D+I	?	?	?		in progress
D+C	?	∞	?		in progress
D+C+I	?	?	?		in progress
D+A+C	no	∞	?		[80]
H,E	yes	1	yes		[85]
H+A	yes	∞	yes		[85]
H+A+C	yes	fin.	yes		[85]
ω-order terms	$\omega \geq 3$ no	doesn't exist	no		[39]
first order terms	yes	1	yes	ordered trees (lists)	[6],[63]
E+A+C	?	∞	?		[85]

where E is an endomorphism.

It has also been shown that for Is (isomorphism) the unification problem of Is+Γ, where Γ is any of the above theories, is equivalent to the unification problem of Γ alone.

Selected Bibliography

[1] A. Ballantyne, D. Lankford; 'Decision Procedures for simple equational theories', University of Texas at Austin, ATP-35, ATP-37, ATP-39, 1977

[2] Barrow, Ambler, Burstall; 'Some techniques for recognizing Structures in Pictures', Frontiers of Pattern Recognition, Academic Press Inc., 1972

[3] L.D. Baxter; 'An efficient Unification Algorithm', Rep. CS-73-23, University of Waterloo, Dept. of Analysis and Computer Science, 1973

[4] Bennett, Easton, Guard, Settle; 'CRT-aided semiautomated mathematics', AFCRL-63, Applied Logic Corp., 1963

[5] Bennett, Easton; 'CRT-aided semi-automated mathematics', AFCRL-65, Applied Logic Corporation, 1965

[6] Bennett, Easton, Guard, Settle; 'CRT-aided semi-automated mathematics', AFCRL-67-0167. Applied Corp., Princeton

[7] F. Blair et al; 'SCRATCHPAD/1: An interactive facility for symbolic mathematics',Proc. of the 2nd Symposium on Symbolic Manipulation, Los Angeles,1971

[8] D.G. Bobrow (ed.); 'Symbol Manipulation Languages', Proc. of IFIP, North Holland Publishing Comp., 1968

[9] H.P. Böhm, H.L. Fischer, P. Raulefs; 'CSSA: Language Concepts and Programming Methodology', Proc. of ACM, SIGPLAN/ART Conference, Rochester, 1977

[10] H. Boley; 'Directed Recursive Labelnode Hypergraphs: A New Representation Language', Journal of Artificial Intelligence, vol 9, no. 1, 1977

[11] H. Bryan, J. Carnog; 'Search methods used with transistor patent applications', IEEE Spectrum 3, 2, 1966

[12] Caviness; 'On Canonical Form and Simplification', JACM, vol 17, no. 2, 1970

[13] C. Christensen, M. Karr; 'IAM, a System for interactive algebraic Manipulation', Proc. of the 2nd Symposium on Symbolic Manipulation, Los Angeles, 1971

[14] M. Clowes; 'On Seeing Things',Journal of Artificial Intelligence, 1971

[15] CODASYL Systems Committee; 'A survey of Generalized Data Base Management Systems', Techn. Rep. 1969, ACM and IAG

[16] CODASYL Systems Committee; 'Feature Analysis of Generalized Data Base Management Systems', TR 1971, ACM, BC and IAG

[17] E.F.Codd; 'A Relational Model of Data for Large shared Databanks', CACM, 13, 6, 1970

[18] E.F. Codd; 'Relational Completeness of Data Base Sublanguages', in Data Base Systems, Prentice Hall, Courant Comp. Science Symposia Series, vol 6, 1972

[19] Cook; 'Algebraic techniques and the mechanization of number theory', RM-4319-PR, Rand Corp., Santa Monica, Cal.,1965

[20] D.G. Corneil; 'Graph Isomorphism', Ph. D. Dept. of Computer Science, University of Toronto, 1968

[21] J.L. Darlington; 'A partial Mechanization of Second Order Logic', Mach.Int. 6, 1971

[22] C.J. Date; 'An Introduction to Database Systems', Addison-Wesley Publ.Comp. Inc. 1976

[23] R. Fateman; 'The User-Level Semantic Matching Capability in MACSYMA', Proc. of the 2nd Symposium on Symbolic Manipulation, Los Angeles, 1971

[24] D.J. Farber, R.E. Griswald, I.P. Polonsky; 'SNOBOL as String Manipulation Language', JACM, vol 11, no. 2, 1964

[25] J. Fischer, S. Patterson; 'String Matching and other Products', MIT, Project MAC, Report 41, 1974

[26] J.F. Gimpel; 'A Theory of Discrete Patterns and their Implementation in SNOBOL4, CACM 16, 2, 1973

[27] W.E. Gould; 'A matching procedure for ω-order logic', Scientific report No.4, Air Force Cambridge Research Labs.,1966

[28] J.R. Guard; 'Automated logic for semi-automated mathematics', Scientific report No. 1, Air Force Cambridge Research Labs., AD 602 710, 1964

[29] A. Hearn; 'REDUCE2, A System and Language for Algebraic Manipulation', Proc. of the 2nd Symposium on Symbolic Manipulation, Los Angeles, 1971

[30] S. Heilbrunner; 'Gleichungssysteme für Zeichenreihen', TU München, Abtl.Mathematik, Ber. Nr. 7311, 1973

[31] C. Hewitt; 'Description and Theoretical analysis of PLANNER a language for proving theorems and manipulating models in a robot', Dept. of Mathematics,Ph.D. Thesis, MIT, 1972

[32] C. Hewitt; 'Viewing Control Structures as Patterns of Passing Massages', MIT, AI-Lab., Working paper 92, 1976

[33] J.I. Hmelevskij; 'The solution of certain systems of word equations', Dokl. Akad. Nauk SSSR, 1964, 749 Soviet Math. Dokl. 5, 1964, 724

[34] J.I. Hmelevskij; 'Word equations without coefficients', Dokl. Akad. Nauk. SSSR 171, 1966, 1047 Soviet Math. Dokl.7, 1966, 1611

[35] J.I. Hmelevskij; 'Solution of word equations in three unknowns',Dokl.Akad.Nauk. SSSR 177, 1967, No.5 Soviet Math. Dokl. 8, 1967, No. 6

[36] G.P. Huet; 'Constrained resolution: a complete method for theory', Jenning's Computing Centre rep. 1117, Case Western Reserve Univ., 1972

[37] G.P. Huet; 'The undecidability of unification in third order logic', Information and Control 22 (3), 257-267,1973

[38] G. Huet; 'Unification in typed Lambda Calculus', in: λ-Calculus and Comp.Sci. Theory, Springer Lecture Notes, No. 37, Proc.of the Symp. held in Rome, 1975

[39] G.P. Huet; 'A Unification Algorithm for typed λ-Calculus', J. Theor.Comp. Sci., 1, 1975

[40] G. Huet; 'Résolution d'équations dans des langages d'ordre 1..2,...,w; Thèse d'etat, IRIA-Laboria, 1976

[41] G. Huet; 'Confluent Reductions: Abstract Properties and Applications to Term Rewriting Systems', IRIA-Laboria, Rap.de Recherche no. 250, Rocquencourt, 1977

[42] D. Jensen, T. Pietrzykowski; 'Mechanising ω-order type theory through unification', Rep. CS-73-16, Dept. of Applied Analysis and Comp. Sci. University of Waterloo, 1973

[43] Karp, Miller, Rosenberg; 'Rapid Identification of repeated Patterns in Strings, Trees and Arrays', ACM Symposium on Th. of Comp. 4, 1972

[44] D.E. Knuth, P.B. Bendix; 'Simple Word Problems in Universal Algebras', in Computational Problems in Abstract Algebra, J. Leech (ed.), Pergamon Press, Oxford, 1970

[45] Knuth, Morris, Pratt; 'Fast Pattern Matching in Strings', Stan-CS-74-440, Stanford University, Comp.Sci. Dept., 1974

[46] S. Kühner, Ch. Mathis, P. Raulefs, J. Siekmann; 'Unification of Idempotent Functions', Proceedings of fourth IJCAI-77, MIT, Cambridge

[47] G. Levi, F. Sirovich; 'Pattern Matching and Goal directed Computation', Nota Interna B73-12, Univ. of Pisa,1973

[48] M. Livesey, J. Siekmann; 'Termination and Decidability Results for String Unification', Essex University, Computer Centre Memo CSM-12, 1975

[49] M. Livesey, J. Siekmann; 'Unification of Sets', Int. Ber. 3/76,Inst. f. Informatik I, Univ. Karlsruhe

[50] C.L. Lucchesi; 'The undecidability of the unification problem for third order languages', Rep. CSRR 2059, Dept. of Applied Analysis and Comp. Science, University of Waterloo, 1972

[51] G.S. Makanin; 'The Problem of Solvability of Equations in a Free Semigroup', Soviet. Akad. Nauk SSSR, TOM 233,no.2, 1977

[52] Manove,Bloom,Engelmann;'Rational Functions in MATHLAB',IFIP Conf.on Symb.Manip., Pisa, 1968

[53] A.A. Markov; 'Trudy Mat. Inst. Steklov', No. 42, Izdat. Akad. Nauk SSSR, 1954, MR 17, 1038, 1954

[54] Maurer; 'Graphs as Strings', Universität Karlsruhe, Techn. Report, 1977

[55] J. Moses; 'Symbolic Integration: The Stormy Decade', CACM 14, 8, 1971

[56] J. Moses; 'MACSYMA - the fifth Year', Project MAC, MIT, Cambridge, 1974

[57] A. Nevins; 'A human-oriented logic for automatic theorem proving', George Washington University, Techn.Rep.,1971

[58] Paterson, Wegman; 'Linear Unification', IBM Research Report 5304, 1976

[59] G.E. Peterson, M.E. Stickel; 'Complete Sets of Reductions for Equational Theories with Complete Unification Algorithms', University of Arizona, Techn. Rep., 1978

[60] G. Plotkin; 'Building in equational theories', Mach. Intelligence, vol 7, 1972

[61] J. Rastall; 'Graph-family Matching', Univ. of Edinburgh, MIP-R-62, 1969

[62] P. Raulefs, J. Siekmann; 'Unification of Idempotent Functions', (extended and revised version of [46], 1978

[63] J.A. Robinson; 'A machine oriented logic based on the resolution principle', JACM: 12, 1965

[64] J.A. Robinson; 'A review on automatic theorem proving', Symp. Appl. Math., vol 19, 1-18, 1967

[65] J.A. Robinson; 'Computational Logic: The Unification Computation', Mach. Intell. 6, 1970

[66] G. Robinson, L. Wos; 'Maximal models and refutation completeness: Semidecision procedures in automatic theorem proving', in Boone et al. (eds.), "Word problems", North Holland, 1973

[67] Rulifson, Derksen, Waldinger; 'QA4: A procedural calculus for intuitive reasoning', Stanford Univ., Nov. 1972

[68] This interesting area of application was pointed out to us by W. Schönfeld, Universität Stuttgart

[69] J. Siekmann; 'Stringunification' Essex University, Memo CSM-7, 1975

[70] J. Siekmann; 'Unification of commutative terms', Universität Karlsruhe, Int. Bericht 2, 1976

[71] J. Siekmann; 'Unification and Matching Problems', Ph.D., Essex University, MEMO CSM-4-78

[72] SIGSAM Bulletin; 'ACM special interest group on Symbolic and Algebraic Manipulation', vol 11, no. 3, 1977 (issue no. 43) contains an almost complete bibliography

[73] D. Skordew, B. Sendow; 'Z. Math. Logic Grundlagen', Math. 7 (1961), 289, MR 31, 57 (russian) (English translation at University of Essex, Comp.Sci.Dept.)

[74] J.R. Slagle; 'ATP with built-in theories including equality, partial ordering and sets, JACM 19, 120-135, 1972

[75] J.R. Slagle; 'ATP for theories with simplifiers, commutativity and associativity', JACM 21, 4, 1974

[76] B.C. Smith, C. Hewitt; 'A Plasma Primer', MIT, AI-Lab., 1975

[77] G.F. Stewart; 'An Algebraic Model for String Patterns', University of Toronto, CSRG-39, 1974

[78] M.E. Stickel; 'A complete unification algorithm for associative, commutative Functions', Proc. 4th IJCAI, Tbilis, USSR, 1975

[79] E. Sussenguth; 'A graph-theoretical algorithm for matching chemical structures', J. Chem. Doc. 5,1, 1965

[80] P. Szabó; 'The Undecidability of the D_A-Unification Problem; Universität Karlsruhe, Int. Ber. 3, 1978

[81] P. Szabó, E. Unvericht;'D-Unification has infinitely many mgus', Universität Karlsruhe, Inst. f. Informatik I (forthcoming)

[82] J.R. Ullman; 'An Algorithm for Subgraph Isomorphism', JACM, vol 23, no. 1, 1976

[83] S.H. Unger; 'GIT - A Heuristic Program for Testing Pairs of directed Line Graphs for Isomorphism', CACM, vol 7, no. 1, 1964

[84] J. van Vaalen; 'An Extension of Unification to Substitutions with an Application to ATP', Proc. of Fourth IJCAI, Tbilisi, USSR, 1975

[85] E. Vogel; 'Unifikation von Morphismen', Diplomarbeit, forthcoming, 1978

[86] D.H.D. Warren; 'Implementing PROLOG', vol 1 and vol 2, D.A.I. Research Rep. no. 39, University of Edinburgh, 1977

[87] P. Weiner; 'Linear Pattern Matching Algorithms', IEEE Symp. on Sw. and Automata Theory, 14, 1973

[88] van Wijngaarden (et al); 'Revised Rep. on the Algorithmic Language ALGOL68', Springer-Verlag, Berlin, Heidelberg, N.Y., 1976

[89] Winston; 'The Psychology of Computer Vision', 1975, McGraw Hill

[90] G. Winterstein; 'Unification in Second Order Logic', Bericht 3, Univ. Kaiserslautern, 1976

[91] K. Wong, K. Chandra; 'Bounds for the String Editing Problem', JACM, vol 23, No. 1, 1976