PUC

Series: Monografias em Ciência da Computação
Nº 14/82

OUTLINES OF A MATHEMATICAL THEORY OF PROBLEMS

by

Paulo A. S. Veloso

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
MARQUÊS DE SÃO VICENTE, 225 – CEP-22453
RIO DE JANEIRO – BRASIL

Series: Monografias em Ciencia da Computação

Nº 14 /82

Series Editor: M. A. Casanova

December 1982

OUTLINES OF A MATHEMATICAL THEORY OF PROBLEMS*

by

Paulo A. S. Veloso

* Research partly sponsored by FINEP and CNPq.

Abstract

This paper outlines a program of research aiming at a mathematical theory of problems with a wide applicability. Motivation, present status with typical results, ongoing work and future directions are outlined.

Problems, solutions and problem-solving methods are notions widely dealt with in Heuristics, Artificial Intelligence, etc. However, these notions are generally treated either too restrictively or in vague ways.

In order to undertake a rigorous study of problems and associated notions, one must have them precisely defined. Here we propose a formulation of problem as a mathematical structure, solutions being certain functions. Thereby, we have at our disposal the available logico-algebraic machinery. Moreover, within this framework, notions such as solvability, reduction, decomposition, etc. can be precisely formulated and rigorously examined.

We outline a framework for the study and critical analysis of problems. This framework is matematical in order to have rigor. But, it is not oriented only towards mathematical problems, rather an effort is made so as to have it applicable to a wide diversity of problems.

Key words: problems, solutions, theory of problems, reduction, decomposition

RESUMO

Este trabalho delineia um programa de pesquisa que tem como objetivo uma teoria matemática de problemas com largo espectro de aplicações. São apresentadas motivação, estado atual com resultados típicos, trabalho com progresso e futuras direções.

Problemas, soluções, métodos de resolução de problemas são noções encontradiças em heurística, inteligência artificial, etc., sendo, porém, geralmente tratadas de modo muito restrito ou vago.

Um estudo rigoroso de problemas e noções associadas exige definições precisas: propõe-se formular a noção de problema como sendo uma estrutura matemática, suas soluções sendo certas funções. Assim, não só se pode lançar mão de todo um ferramental lógico - algébrico como também se explicita um cenário onde noções como solubilidade, redução, decomposição, etc. podem ser formuladas e investigadasde maneira rigorosa. O cenário delineado para o estudo e análise crítica de problemas é matemático tendo em vista os objetivos de precisão e vigor. Porém, faz-se um esforço no sentido de evitar orientação excessiva para problemas de caráter matemático de modo a se ter uma teoria abarcando uma gama bastante variada de problemas.

Palavras Chaves: problemas, soluções,
teoria de problemas,
redução, decomposição.

1. INTRODUCTION

The aim of this paper is to outline some aspects of a general theory of problems. More specifically we hope to explain the motivation, outline a program of research and describe its present status.

Problems have been faced and overcome for quite a long time. However, reflection about problems and problem solving has not been such a widespread activity, at least in a systematic way.

Of course, Descartes has touched on the question with his analytical method and the creation of Cartesian geometry was an important step. Other thinkers, such as Leibnitz and Bolzano have devoted some time to this question. More recently, G. Polya has written a series of excellent books on the art of approaching and solving problems, especially of mathematical nature. Also, in the field of artificial intelligence, some progress has been achieved in the study of processes such as inductive generalization, hypothesis formation, etc. [5,9].

However, most of the above approaches address to the questions "what is the problem ?" and " how to solve it ?". Here, we want to address the less usual questions "what is a problem?" and "what is a solution?". Some initial effort in this direction has been done by J.E.F. Barbosa in the context of the teaching of mathematics, our approach being independent.

Our aim here is the development of a framework where notions such as problem, solution, etc. are given precise mathematical formulations. In this context, questions about these concepts can be precisely formulated and studied. In order to clarify this goal an analogy with mathematical logic might be appropriate. In metamathematics, one formalizes notions such as proof, theorem, etc. But, the aim of metamathematics is not to teach how to prove theorems, but a critical analysis of the notions of proof, theorem, etc. Analogously, our theory does not purport to teach how to solve problems. Rather, it is expected to form a framework for the critical analysis of the concepts related to problems.

In order to undertake a rigorous study of properties of problems we must have a clear notion of what a problem is. In the following sections we shall first propose a precise definition of problem, obtained by means of a critical analysis of the familiar, but somewhat vague, everyday ideas. Within this framework we shall then outline how to study rigorously a method of problem transformation - reduction - and a method of problem solving - decomposition. We shall then reexamine critically our formulations, which will suggest some generalizations more apt to capture certain ingredients that are important in some contexts.

2. THE CONCEPT OF PROBLEM

In his excellent book "How to solve it" G. Polya [10] suggests asking the following questions in approaching a problem
"What are the data ?"
"What are the possible results ?"
"What constitutes a satisfactory solution ?

We have taken these three questions as a guide in formulating our concept of problem. We shall introduce it by means of a simple example, that of "finding a root of a polynomial". How are we to specify precisely this problem ?

. What are the data ?
We have to specify the domain of problem instances. Are we dealing with polynomials with integral coefficients or real coefficients? Are these quadratic polynomials or do they have arbitrary degree ? Of course, the answers to these questions affect the very nature of the problem and how to approach it: finding roots of quadratic polynomials with integral coefficients is much easier than finding roots of polynomials of arbitrary degree with real coefficients.

. What are the possible results ?
We have to specify the domain of problem results. Do we want integral roots, real roots, complex roots? Again this affects the very nature of the problem by influencing its solvability: a polynomial may have no integral root, one real root and two complex roots. Also, knowing the domain of results a priori may influence the choice of the method used to approach the problem.

. What constitutes a satisfactory solution ?
We have to specify the notion of solution: which results solve which problem instances. For instance, a polynomial in general has more than one root; do we want the smallest one, the largest one, or any one ? Again, this affects the very nature of the problem: finding any root is apparently easier than finding, say, the smallest one.

Thus, in order to specify precisely the problem of finding roots of polynomials we have to supply the following information.

1. The domain D of data, or problem instances, consists of, say ,

all polynomials with real coefficients in the indeterminate x;

2. The domain R of results consists of, say, all complex numbers;

3. The relation q of being a satisfactory solution is the relation from D to R defined as follows
(p(x), c) $\in$ q (i.e. c is a solution for p(x)) iff p(c) = 0 (the value of p at the point c is 0).

So, we have specified our example problem precisely as that of "finding any complex root of a polynomial with real coefficients and arbitrary degree". If we wanted the problem of finding the smallest real root of an arbitrary degree polynomial with real coefficients we would specify it by taking the same domain D of data but changing the domain of results to R' consisting of all real numbers and the relation of being a satisfactory solution to the relation q' from D to R' defined by (p(x),r) $\in$ q' iff p(r) = 0 and r $\leq$ r' for any r'$\in$R' such that p(r') = 0.

With this example in mind we can formulate our definition of problem, which we call unconstrained problem for reasons that will become apparent later on.

An unconstrained problem Q consists of [15]
- a nonempty set D, called the domain of data ;
- a nonempty set R, called the domain of results;
- a binary relation q from D to R, called the problem condition.

Now, by a solution for problem Q we mean a function a from D to R, assigning to each data d$\in$D a possible result a(d)$\in$R satisfying the problem condition, i.e. a : D $\rightarrow$ T is such the for every d$\in$D we have (d, a(d)) $\in$ q.

Some remarks concerning this formulation may be in order. First, we want to consider general problems and not only specific instances. Returning to our example we would call finding a root of $3x^4 + 5x^3 + 1$ an instance of the (general) problem of finding a root of a polynomial. This viewpoint has been influenced by the concept of "decision problem" in recursive function theory.

A second remark concerns the notion of solution. It may appear unnecessarily rigid: why do we not take as a solution a

relation from D to R ? Well, then q itself would be a solution. This has the unpleasant consequence that in specifying a problem one already gives a solution: there is no problem to be solved !

A third remark is that these notions of problem and solution might be too wide. We shall come back to this point later on. For the moment, we just mention that we do want these notions to have wide applicability.

However, we shall give two examples in order to illustrate the flexibility of these notions. First, consider our problem instance of finding a complex root of the specific polynomial $3x^4 + 5x^3 + 1$. This can be formulated as an (unconstrained) problem according to our definition. For this purpose, take

- the domain D of data to consist of the single polynomial $3x^4 + 5x^3 + 1$;
- the domain R of results to consist of all complex numbers ;
- the problem condition q to be defined as $(3x^4 + 5x^3 + 1, c) \in q$ iff $3c^4 + 5c^3 + 1 = 0$.

Now, let us look at another example, that of finding all complex roots of real polynomials. Here we take

- the domain D of data to consist of all polynomials with real coefficients;
- the domain R of results to consist of all (finite) sets of complex numbers;
- the problem condition to be defined by $(p(x), S) \in q$ iff $S = \{c \in R \ / \ p(c) = 0\}$

Incidentally this latter example illustrates a general strategy in translating a problem formulated in current language into our formalism. It frequently appears that a solution should be a relation assigning to a data several results. This can be handled by taking as domain of results the powerset of what we were considering as domain of results.

The above examples are of a mathematical nature. This is so for commodity only: our notion of problem being mathematical, mathematical examples are easier to describe. However it is not difficult to see that our formulation is sufficiently general so

as to include examples such as

- proving a mathematical theorem,
- assembying an automobile,
- the monkey-and-bananas problems (a well-known problem in Artificial Intellingence),
- selecting flight itineraries,
- classifying plant specimens,
- ordering lists of numbers,
- finding centers of geometric figures,
- puzzles, such as the tower of Hanoi.

Having in mind our definitions of problem and solution, it is quite natural to call a problem solvable iff it has a solution. Also, let us call a problem $Q = \langle D, R, q \rangle$ feasible iff for every possible data $d \in D$ there exists a matching result $r \in R$ in that $(d,r) \in q$, or in short, the domain of the relation q is all of D.

These notions are interesting in that they give solvability conditions as stated in the following result [15].

Proposition. Given an unconstrained problem $Q = \langle D, R, q \rangle$ the following conditions are equivalent

1. Q is solvable
2. Q is feasible
3. Q is a model of the sentence

$$(\forall d: D)(\exists r: R)\ q(d,r) \qquad (\sigma)$$

Moreover the solutions of Q are exactly the Skolem functions of σ.

Now that we have defined a problem as a mathematical structure we can carry over in a natural way the available mathematical machinery. For instance, a subproblem is defined as a substructure, the notion of homomorphism for problems is exactly what one would expect. Also, the various homomorfism theorems carry over [7].

However, in addition to these mathematical relationships among problems, there exist others that arise from their being problems. These include methods for transforming or solving

problems. In the next two sections we shall see how they can be precisely formulated.

3. THE METHOD OF REDUCTION

The idea of reduction is widely known albeit in vague terms. Probably the best known example of reduction is that of geometric problems to algebraic ones achieved by Descartes's method of coordinates. As Polya puts it "Here is a problem related to yours and solved before". Other names attached to this idea or variations thereof are generalization, subsumption, analogy, etc. Also, in recursive function theory one encounters some precise - and specialized - formulations of the notion of reduction.

The basic idea in reducing one problem to another is to use a solution of the latter to obtain a solution for the former. This can be done by transforming data of the former into data of the latter and then retrieving results of the former from results of the latter [12].

The following definitions attempt to formalize these intuitive ideas, stressing the basic feature of two links: one between the data domains and another one between the result domains.

So, consider two problems $Q = \langle D, R, q\rangle$ and $Q' = \langle D', R', q'\rangle$. A reduction Γ of Q to Q' consists of [15].

. a function $t : D \to D'$, called the translation map;

. a function $v : D' \times R' \to R$, called the retrieval map.

Notice that the retrieval map assigns to each pair (data, result) of Q' a result of Q. Thus, in retrieving a result of Q we are allowed to use not only a result of Q' but also a corresponding problem instance.

We are not interested in arbitrary reductions but rather in those that embody the basic idea of a solution for Q' yielding a solution Q. Thus we say that a reduction Γ from Q to Q' is good iff for any solution $a' : D' \to R'$ of Q', the assignment

$$a(d) = v(t(d), a(t(d)))$$

defines a solution $a : D \to R$ for Q.

In order to see a simple example, let us consider the geometric problem of finding the center of a rectangle. This can be formulated as an (unconstrained) problem Q with

. D consisting of all rectangles in the plane;

. R consisting of all points of the plane ;

. q defined by $(d,r) \in q$ iff the point r is the center of the rectangle d, i.e. r is the intersection of the two diagonals of d.
Now, the problem of finding the center of a rectangle can be reduced to that of finding the halfway point of a straight line segment. Let us formulate the latter as Q' with

. D' consisting of all straight line segments in the plane ;

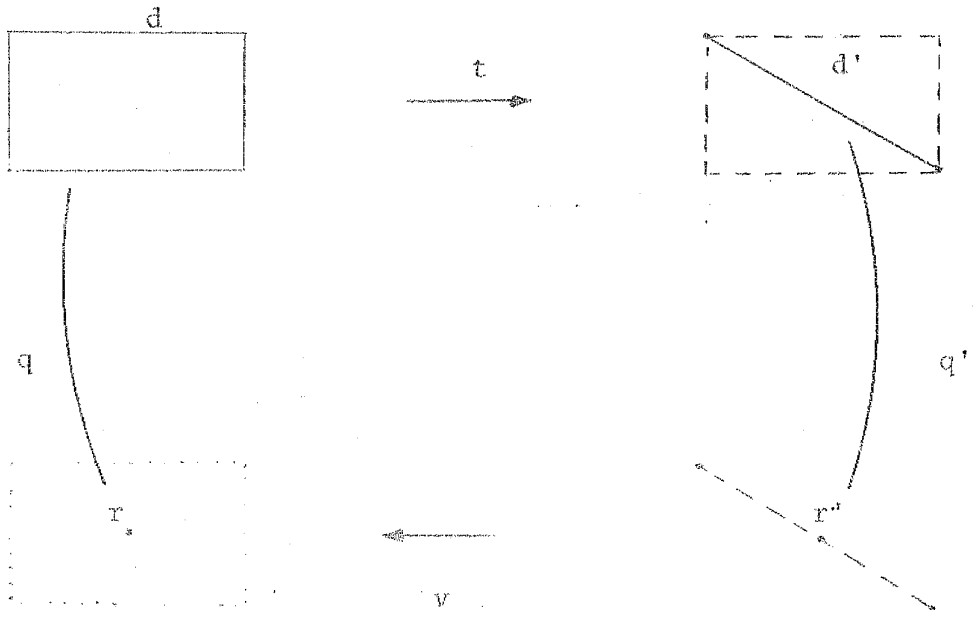
. R' = R ;

. q' defined by $(d',r') \in q'$ iff the point r' is equidistant from both extremes of the segment d'.
Now, let us formulate explicitly the reduction:

. the translation map $t : D \to D'$ assigns to each rectangle $d \in D$ one of its diagonals ;

. the retrieval map $v : D' \times R' \to R$ assigns to each pair straight line segment and point (d',r') the point r'.

Pictorially



which shows the reduction to be good.

Now when is a reduction good? An answer is provided by the following result, which indicates that our formalization captures the basic intuitive idea [12, 15].

Lemma. A sufficient condition for a reduction Γ from Q to Q' to be good is the following

for every $d \in D$ and every $r' \in R'$
if $(t(d), r') \in q'$ then $(d, v(t(d), r')) \in q$.

More interesting than asking whether a given reduction is good the question of whether a good reduction exists at all. In order to answer it let us examine more closely the basic idea of reduction.

In reducing Q to Q' we first have to assign to each problem instance of Q one in Q'. Then we sort of forget about Q for a while and concentrate in finding a solution for Q', which will later be retrieved back into a solution for Q. One way to formulate this more precisely is a follows [15]

Let us consider two problems Q and Q' as before. Let us say that a pair $(d',r') \in D' \times R'$ is analogous to a pair $(d,r) \in D \times R$ iff either $(d',r') \notin q'$ or $(d,r) \in q$.

Now, let us say that a problem instance $d \in D$ subsumes a problem instance $d' \in D'$ iff given any $r' \in R'$ there exists $r \in R$ such that the pair (d',r') is analogous to (d,r).

Notice that we have defined above a non-symmetrical concept of local analogy, not an analogy between problems.

Theorem. Given problems $Q = \langle D,R,q \rangle$ and $Q' = \langle D',R',q' \rangle$ the following conditions are equivalent.

1. For every problem instance $d \in D$ there exists a problem instance $d' \in D'$ subsumed by d.
2. The compound structure $\langle D,D', R,R', q,q' \rangle$ is a model of the sentence

   $(\forall\, d{:}D)\ (\exists d'{:}D')\ (\forall r'{:}R')\ (\exists r{:}R)\ (q'(d',r') \rightarrow q(d,r))$

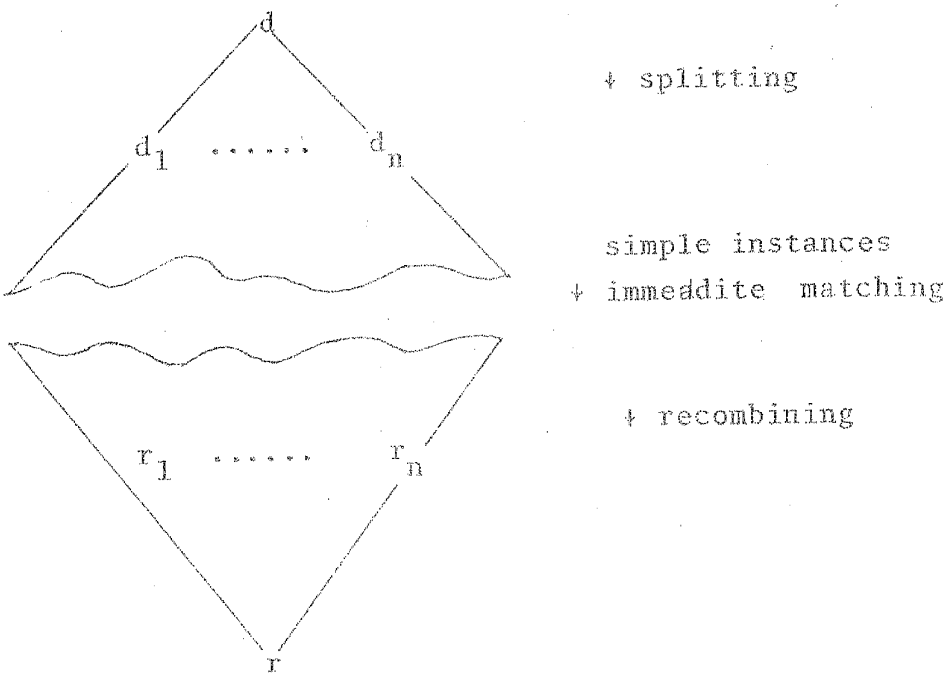
Furthermore, any one of these conditions is sufficient for the existence of a good reduction form Q to Q'.

A remark might be in order here. The quotation from Polya given at the beginning of this section suggests a special case of reduction, that to a problem with a given solution. We have been viewing reduction as problem transformation prior to and independently of any particular solution. In fact, we might even have a good reduction of a problem to another one with no solutions. Our notion of good reduction involves quantification over all solutions. On the other hand, the conditions given in the results are formalizable within (many-sorted) first-order logic. This explains why they are only sufficient but not always necessary.

4. THE METHOD OF DECOMPOSITION

A quite common, and fruitful, approach to solve a problem consists of breaking it into, hopefully, simpler problems. Often this is done repeatedly. This is the basic idea of problem decomposition: repeated splitting of problem instances into smaller subproblem instances until they are simple enough to have direct results, which are then combined into a result for the original problem instance [1].

A pictorial description of the process of decomposition should clarify the basic ideas involved. One starts with a problem instance $d \in D$; if it is not easy enough, one decomposes it into subproblem instances $d_1,\ldots,d_n$ all in D, which replace the original problem instance. Each one of $d_1,\ldots,d_n$ is in turn examined and possibly decomposed. We can visualize this process as generating a tree with d at the root. This process of splitting - or tree generation - continues until we have at the frontier of the tree data that are sufficiently simple so that matching results can be easily found. This gives the leaves of a tree in the result domain. One now starts building up this second tree from the frontier towards the root. Finally, one will have results $r_1,\ldots,r_n$ corresponding, respectively, to $d_1,\ldots,d_n$. Then $r_1,\ldots,r_n$ will be merged into a result r for the original problem instance d.



In order to give a precise formulation for these ideas, let us consider an (unconstrained) problem $Q = \langle D, R, q \rangle$.

An n-ary decomposition Δ_n for Q consists of [12,15]

- n unary functions $s_i : D \to D$, $i = 1,\ldots, n$, called the splitting maps;
- an $(n+1)$ - ary function $m : D \times R^n \to R$, called the recombining map;
- a unary function $e : D \to R$, called the immediate matching ;
- a unary relation $p \subseteq D$, called the simplicity predicate.

Notice that this formulation merely lays explicit the tools for the process of tree generation described above as underlying the basic idea. The only point deserving attention is the recombining map, which is allowed to use as argument a problem instance in addition to n results. This is similar to the case of the retrieval map in problem reduction.

As it was the case with reductions we are not interested in arbitrary decompositions. So, let us an call an n-ary decomposition Δ_n for Q good iff the assigment

$$a(d) = \begin{cases} e(d) & \text{if } d \in p \\ m(d, a(s_1(d)),\ldots,a(s_n(d))) & \text{otherwise} \end{cases}$$

defines a solution $a: D \to R$ for Q.

One should notice that the above formulation requires that at each step of the process of decomposition a problem instance is decomposed into exactly n problem instances, namely $s_1(d),\ldots, s_n(d)$. And this number n, which we call the decomposition index is fixed throughout the process. Thus we are dealing with uniform decompositions. This is not general enough. Sometimes one does not know the index n a priori. Often, this decomposition index is not uniform: at some stage, d is decomposed into, say, d' and d"; then we split d' into d'_1, d'_2 and d'_3, and d" into d''_1 and d''_2. Even worse, it may be happen that

no uniform bound for n exists.

Thus, we ought to deal with non-uniform decompositions, as well. In order to do this we shall first define what appears to be a special case of the n-ary decomposition.

Consider a problem $Q = \langle D, R, q\rangle$. A decomposition Δ for Q consists of

- a unary function $s : D \to D$, called the spliting map;
- a binary function $b : D \times R \to R$, called the recombination map;
- a unary function $c : D \to R$, called the immediate simple solution;
- a unary relation $g \subseteq D$, called the simplicity test.

As before, we say that the decomposition Δ is good iff the assigment

$$a(d) = \begin{cases} c(d) & \text{if } d \in g \\ b(d, a(s(d))) & \text{otherwise} \end{cases} \qquad (\delta)$$

defines a solution $a : D \to R$ for Q.

Now, it might appear that the above definition of decomposition is merely that of 1-ary decomposition. Indeed this is so. However, (unary) decompositions are sufficiently general to encompass arbitrary n-ary decompositions, as we shall now argue. In fact, it will become apparent that they also encompass arbitrary non-uniform decompositions [15].

The reason for the generality of (unary) decomposition is as follows. Given a problem $Q = \langle D, R, q\rangle$, consider the problem $Q^+ = \langle D^+, R^+, q^+\rangle$ where D^+ and R^+ consist of the finite non-null sequences of elements of, respectively, D and R; and q^+ is the natural extension of q to sequences. We then have a natural good reduction of Q to Q^+. Furthermore, an n-ary decomposition Δ_n of Q induces naturally a decomposition Δ of Q^+.

Proposition. With the above notation, Δ_n is a good n-ary decomposition for Q iff Δ is a good decomposition for Q^+.

Having established the generality of decompositions vis-a-vis n-ary decompositions, we may restric ourselves to study decompositions.

First, we shall give can alternative view of our formulation of good decomposition. The assignment (δ) suggests the following. The immediate simple solution c: $D \to R$ is a "partial solution", in that it does solve Q but only for those data $d \in D$ satisfying the simplicity test. Our task is to extend it to a total solution. Decomposition does that by repeated splittings and recombinations.

Now, when is a decomposition good? It might happen that the above splitting process never ends in that one never reaches a simple instance. Also the goal of splitting is not only obtaining simpler problem instances; from the results of the latter one should be able to get a solution for the original problem instance.

We shall now formalize these intuitive ideas in order to answer the above question. It will be seen that a crucial role is played by a notion of comparison of problem instances.

Consider a problem $Q = \langle D, R, q \rangle$. Let us also consider a binary relation $<$ on D, called comparison. We may read $d < d'$ as d is simpler than d', so that $<$ embodies a notion of being easier to solve. We use this notion to classify problem instances. We call a problem instance $d \in D$ extremely easy (with respect to $<$) iff there exists no $d' \in D$ so that $d' < d$. Also, we say that d' covers (with respect to $<$) d iff given any $r' \in R'$ such that $(d', r') \in q$ there exists $r \in R$ with $(d, r) \in q$.

With these definitions we can now state sufficient conditions for a decomposition to be good. Recall that a binary relation $<$ on D is well-founded iff there exists no infinite descending chain $d_o > d_1 > d_2 > \ldots$.

Proposition. Let Q be an (unconstrained) problem and $<$ be a binary relation on D. Given a decomposition Δ for Q assume that the following conditions hold

1. $<$ is well-founded.
2. For all $d \in D$ such that $d \in g$ $(d, c(d)) \in q$.
3. For all $d \in D$ such that $d \notin g$ $(s(d) < d$ and whenever $(s(d), r) \in q$ then also $(d, b(s(d), r)) \in q$.

Then Δ is a good decomposition for Q.

The next result displays sufficient conditions for the existence of a good decomposition. In doing so, it lays bare the fundamental role played by the comparison of problem instances by means of a well-founded relation. This is due to the fact that the maps of a good decomposition can be taken as Skolem functions associated to the conditions of the theorem.

Theorem Let $Q = \langle D, R, q \rangle$ be an (unconstrained) problem and $<$ a binary relation on D. Assume that

1. $<$ is well-founded
2. Whenever $d \in D$ is extremely easy then there exists $r \in R$ such that $(d,r) \in q$.
3. Whenever $d \in D$ is not extremely easy there exists $d' \in D$ such $d' < d$ and d' covers d.

Then there exists a good decomposition for Q.

5. A CRITIQUE

In the preceding sections we have presented precise formulations for the notions of problem and solution, the method of reduction of a problem to another and the method of decomposition of a problem. The advantages of these formulations should now be apparent not only due to their being precise explicitations of current, but vague, notions, but also by the kind of results one can now obtain [12,15].

However, these formulations are not immune to criticisms. These criticisms center in our notion of solution and they stem from two sources: some have a mathematical nature, others have to do with the intended applications.

The criticisms of mathematical nature are related to the Axiom of Choice. Solutions, the maps of a reduction and those of a decomposition are basically Skolem functions. In fact, a feasible problem has a problem condition q with domain D. A solution for it consists of a function contained in q but with the same domain. The Axiom of Choice is what guarantees the existence of a solution of a feasible problem. This is not too bad, for we do not want to equate "finding a solution" with "electing a function of which all that we know is that, by the Axiom of Choice, it exists". However, we do have the following equivalences [13].

Proposition The following assertions are equivalent

1. The Axiom of Choice.
2. "Every feasible (unconstrained) problem is solvable".
3. "Every feasible (unconstrained) problem has a good decomposition".

This result shows that the Axiom of Choice is coextensive with the method of decomposition. This is somewhat puzzling as one generally regards decomposition as a constructive method of solving a problem, whereas the Axiom of Choice is a prototypical assertion of non-constructive existence.

This paradox is only apparent. We must realize that the constructive character of decomposition is relative: we have an

explicit, constructive, definition of a solution only given the maps of the decomposition. And these maps may be non-constructive.

Our formulation is intended to be general; we do not want to be tied down to any particular notion of constructiveness. However, the question that remains is "Does our formulation allow room for a notion of constructivity to be incorporated as needed?". Or are we perhaps condemned, so to speak, by the Axiom of Choice ?

The above remarks lead us to examine more closely criticisms of another nature. This has to do with the notions of "context", and the kinds of solutions permissible in a given context.

In connection with geometric problems one aften encounters the expression "ruler and compass construction". This is intended to rule out certain functions as allowable solutions; we only admit as candidates for solutions those functions that can be constructed by means of ruler and compass.

This ruling out of certain functions occurs quite frequently. In connection with programming problems we would want only computable functions. In a topological context we may be interested only in continuous functions.

A finer analysis will reveal still another aspect of the notion of solution. Imagine that a geometrical problem has been assigned to two different students and that both present correct solutions, but one is more elegant than the other. It may happen that as functions both solutions are equal, in that they assign to the same data the same results. Even so, one may be more elegant than the other.

What the above example points out is that we are sometimes interested in non-extensional properties of solutions. Typically, these properties evaluate with respect to some criteria solutions which might be indistinguishable as set-theoretical functions. In a programming context this is quite frequent: programs for the same task with diverse degrees of efficiency.

To sum up, we are sometimes interested in solutions not as assignments of results to data. Rather, we want to view as solution the very description of this assignment. But then we must have at our disposal a language for describing such functions.

In the next section we shall suggest how to meet these objections. Another point will be dealing with is the following. In the process of solving a problem about equilateral triangles we may find convenient to work with some non-equilateral triangles. This occurs quite frequently in connection with decomposition when a generalization of the original problem is more amenable to treatment by the method [1,10,12].

6. SOME GENERALIZATIONS

We shall now indicate some generalizations of our concepts related to (unconstrained) problem so as to meet the objections raised in the preceding section.

Let us first consider the case of ruling out certain functions as a priori not acceptable as solutions. For this purpose we assume given a class A of functions, called admissible functions.

An A-problem (short for problem in the context A) Q consists of [14]

- domains D and R, as before;
- a relation q from D to R, as before;
- a subset I of D, called the set of instances of interest.

A solution for the A-problem Q is a function a : D → R such that

- $a \in A$ (i.e. a is admissible);
- for every $d \in I$ $(d, a(d)) \in q$.

Notice that the definition of solution is now relativized to the class A of admissible functions. Of course, for a given A-problem Q it is not necessary to consider the whole class A of admissible functions; it suffices to consider the set A(D,R) of the functions from D to R that are admissible. Also, notice that unconstrained problems are special cases of A-problems with A consisting of all functions and I=D (i.e. every problem instance is of interest).

Now, let us call an A-problem Q feasible iff for every $d \in I$ there exists $r \in R$ such that $(d,r) \in q$ (i.e. the domain of q includes I). Being feasible is a necessary condition for the solvability of an A-problem, but it is no longer sufficient.

The method of decomposition can be generalized to A-problems. The main point now is guaranteeing that the function obtained is still admissible [14].

Similarly, the method of reduction can be generalized to A-problems, the main point being that the function obtained is still admissible. For this purpose, it may be desirable to consider two cases:

- reduction within the same context (of an A-problem Q to another A-problem Q').

. reduction from one context to another (of an A-problem Q to an A'-problem Q').

It is frequently interesting to have some idea of the behavior of all possible solutions for a problem. This may provide valuable information concerning solvability, decomposability, etc. This is of interest for unconstrained problems, the more so for A-problems. A helpful concept in this connection is that of solution space. The solution space of an A-problem Q is the set S(Q) of all its solutions.

A natural question here is how the solution space of Q is affected by variations of the components of Q. For instance, it is clear that S(Q) decreases if I increases or if q or A decrease.

Another interesting aspect is the behavior of the solution space under operations on problems. For instance, the operation of direct product is of special interest, for we may be able to reduce Q to $Q_1 \times Q_2$ and then solve Q_1 and Q_2 separately. This is so because, with the natural definitions, we have $S(Q_1 \times Q_2) = S(Q_1) \times S(Q_2)$. Similarly, one can study the behavior of the solution spaces under various other operations deemed important in this context, such as variations of union (sum and join), sequence (the generalization of the operation $Q \to Q^+$ of section 3 to A-problems), composition of problems, etc. [7,8]

Unconstrained problems and A-problems appear to be formulations of the notion of problem appropriate in certain circumstances. However both formulations regard a solution as an extensional object. We now propose a formulation where a solution is the description of a function, a non-extensional object. We call these intentional problems, for lack of a better name.

An intentional problem Q consists of

- domains D and R, as before;
- a relation q from D to R, as before;
- a subset I of D, as before;
- a nonempty set P, called the domain of programs or function descriptions;
- a function $w : P \times D \to R$, called the evaluation map.

We think of the evaluation map as responsible for applying a

given function description $p \in P$ to a data argument $d \in D$ yielding a result $w(p,d) \in R$. Thus, each function description $p \in P$ induces a function $w_p : D \to R$, called the function induced by p, defined by the assignment $w_p(d) = w(p,d)$. It is now quite possible that distinct programs $p \neq p'$ induce the same function $w_p = w_{p'}$. So, we may be able to distinguish p from p' by means of non-extensional attributes (e.g. efficiency, elegance, etc.) which would not make much sense about the induced functions.

Accordingly, a solution for the intentional problem $Q = \langle D, R, q, I, P, w \rangle$ is a program $p \in P$ such that the induced function w_p solves the underlying extensional problem, in the sense that for every $d \in I$ $(d, w(p,d)) \in q$.

Notice that intentional problems generalize A-problems, for we can take the domain P of programs to consist of all admissible functions from D to R and the evaluation map to be simply functional application.

Naturally, notions related to unconstrained problems and to A-problems can be generalized to intentional problems. The main point here is that one cannot say much without having some more information about the domain P of programs.

Another, variation that is closer to A-problems is obtained by considering a fixed language L for describing functions, (e.g., recursive definitions). Given domains D and R, each element p of L will then be interpreted as a function from D to R.

7. CONCLUSION

We have outlined some aspects of a program of research aiming at a rigorous study of problems and related notions. We have given precise formulations for these notions in a mathematical context. But we have not overemphasized the mathematical aspects, for the idea is to have a framework with wide applicability, to problems form several walks of life.

We have proposed some formulations which appear to embody ingredients important from diverse viewpoints . It might be helpful to review these formulations under the form of a taxonomy of problems.

First, we have a division of our formulations into extensional and intentional.

A. Extensional problems are those having as solutions set-theoretical functions. They further subdivide into unconstrained and constrained problems.

A.1 Unconstrained problems have no a priori limitation on which functions are solutions;
they only have to satisfy the problem condition.

A.2 Constrained problems have the a priori restriction that only certain functions are admissible as possible solutions. We may consider two ways of incorporating this restriction:

- by means of a context, a class of admissible functions for several problems;
- by attaching to each problem its own set of admissible functions.

B. Intentional problems are those having as solutions, not mere functions, but descriptions of functions in some appropriate language. As with contrained problems we have two ways of giving this language:

- by means of a general language for the description of programs, which are then interpreted as functions;
- by attaching to each problem its own language for the

description of programs, which induce functions.

The above classification is presented in the order of increasing generality. In each case, we can, for technical convenience, consider the addition of a set of problem instances of interest.

The above formulations serve distinct purposes . Unconstrained problems are the simplest kind but they appear to be adequate for several cases. Constrained problems come into play when we have some restriction on the kind of functions acceptable as possible solutions (e.g., ruler-and-compass constructions, computable functions, etc.). Finally, intentional problems appear appropriate for cases where we want to evaluate solutions according to non-extensional criteria (e.g. efficiency, clarity elegance, etc.).

One should mention the possibility of capturing constraints and non-extensional solutions by means of suitable redefinitions of the problem components. However, in some cases this may force the original problem problem to appear in a somewhat unnatural disguise.

We have not mentioned some aspects of this program that are already under development. These include notions of similarity and analogy between problems, problem specification and heuristic advice on how to go about solving a problem. The most important aspect now under development is the study of more cases of real problems from various disciplines in order to have a better feeling of the adequacies and limitations of our present formulations. Of course, this study will heavily influence future directions.

REFERENCES

1 . A. V. Aho, J. E. Hopcroft, J.D. Ullman- The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, MA., 1975

2 . R. B. Banerji - Theory of Problem Solving: an approach to Artificial Intelligence. American Elsevier, New York, 1969

3 . H. B. Enderton - A Mathematical Introduction to Logic. Academic Press New York, 1972

4 . G. Grätzer - Universal Algebra, D. van Nostrand, New York, 1968

5 . P. Hájek, T. Havránek - Mechanizing Hypothesis Formation. Spring-Verlag, Berlin, 1978

6 . E. Horowitz, S. Sahni - Fundamentals of Computer Algorithms. Computer Science Press, Potomac, MD, 1978

7 . M. A. Lopes - Introduction to a General Theory of Problems (in Portuguese). Doctoral Dissertation. Dept. Informática, PUC/RJ, forthcoming

8 . M. A. Lopes, P. A. S. Veloso - Operations on problems and their solution spaces. Proc. 6th. European Meeting on Cybernetics and Systems Research, Vienna, April 1982

9 . N. J. Nilsson - Problem Solving Methods in Artificial Intelligence. McGraw-Hill, New York, 1971

10. G. Polya - How to Solve it : a new aspect of the mathematical method. Princeton Univ. Press, Princeton, NJ, 1957

11. G. Polya - Mathematical Discovery: on understanding, learning and teaching problem solving. Wiley, New York, 1962

12. P. A. S. Veloso - Divide-and-conquer via data types. Proc. VII Latin-American Conf. on Informatics. Caracas, Venezuela, Jan. 1980

13. P. A. S. Veloso - A "problem-theoretic" equivalent of AC. Annual Meeting of the Assoc. for Symbolic Logic. San Francisco, CA, Jan. 1981.

14. P. A. S. Veloso, M. A. Lopes - A framework for problem solving : theory and methodology. Proc. Internat. Conf. on Applied Systems Research and Cybernetics. Acapulco, Mexico, Dec. 1980

15. P. A. S. Veloso, S. R. M. Veloso - Problem decomposition and reduction : applicability, soundness, completeness. Proc. 5th European Meeting on Cybernetics and Systems Research. Vienna, Apr. 1980