A THEORY OF JUSTIFIED REFORMULATIONS[1]

Devika Subramanian
Department of Computer Science
Cornell University
Ithaca, NY 14853

ABSTRACT

Present day systems, intelligent or otherwise, are limited by the conceptualizations of the world given to them by their designers. In this paper, we propose a novel, first-principles approach to performing incremental abstraction reformulations for computational efficiency based on *justifying* shifts in conceptualization. A new class of meta-theoretical justifications for a reformulation called *irrelevance* explanations, is presented. A logical irrelevance explanation demonstrates that certain distinctions made in the formulation are not *necessary* for the computation of a given class of problems. A computational irrelevance explanation shows that some distinctions are not *useful* with respect to a given problem solver for a given class of problems. We then present a meta-theoretical principle of ontological economy called the *irrelevance principle*. The irrelevance principle logically *minimizes* a formulation by removing all facts and distinctions that are either logically or computationally irrelevant to the specified goals.

REFORMULATION

When we formulate a problem, we determine the distinctions needed to state and solve it. These distinctions constitute a conceptualization of the problem. Reformulation is *re*conceptualization, a change in the basic objects, functions and relations assumed by a formulation. A reformulation is *correct* with respect to a set of goals if the answers to the goals are preserved across the conceptual shift. Standard examples of correct reformulations include the mutilated array puzzle [Korf, 1980], the rectangular to polar coordinate transformations, and the Laplace and Fourier transforms. In all these cases, the basic primitives used to describe the problem are changed as a result of the reformulation. This shift in conceptualization causes a reconfiguration of the search space of solutions to a problem. For example, in the naive formulation of the mutilated array problem, there are 62 distinct objects; the reformulation that colors the array as a checkerboard grouped them into 30 objects of one colour and 32 of another. This regrouping causes the search space to shrink from size 2^{62} to 1. Reformulations are thus for a purpose: all of the conceptual transformations above lead to formulations that are computationally effective for certain classes of queries.

Since reformulation is a shift in conceptualization, we begin with some examples of conceptualizations. We can conceptualize kinship among a set of individuals as in Figure 1. A conceptualization makes our ontological commitments explicit. Another conceptualization of the kinship problem is in Figure 2. Encodings of the two conceptualizations are in Figure 3.

Definition 1 *A conceptualization $\mathcal{C}_2$ is a reconceptualization of $\mathcal{C}_1$ with respect to some background conceptualization Δ, if the elements of $\mathcal{C}_2$ are definable from $\mathcal{C}_1$ and Δ. The definitions of the elements of $\mathcal{C}_2$ in terms of Δ and $\mathcal{C}_1$ constitute the* articulation theory *between the two conceptualizations.*

In our kinship example, the conceptualization $\mathcal{C}_2$ can be constructed out of $\mathcal{C}_1$ by dropping the **Father** and **Ancestor** relations and introducing the new relation **FoundingFather** which is a subset of the **Ancestor** relation. Since $\Delta = \emptyset$ here, we have not introduced any new distinctions in this shift; we simply coarsened the **Ancestor** distinction. This reconceptualization is an *abstraction.*

Definition 2 *$\mathcal{C}_2$ is a* correct *reconceptualization of $\mathcal{C}_1$ with respect to Δ and the set of goal relations G if G is definable in $\mathcal{C}_2$ only if it is definable in $\mathcal{C}_1$.*

Objects: the set **P** of people {**A,B,C,D,E,F,G**}

Functions: the function **Father** from **P** to **P**, which is the set
{(**A,B**),(**A,C**),(**B,D**),(**B,E**),(**C,F**),(**C,G**)}

Relations: the relation **Ancestor**, which is the following subset of $\mathbf{P}^2$,
{(**A,B**),(**A,C**),(**A,D**),(**A,E**),(**A,F**),(**A,G**),(**B,D**),(**B,E**),(**C,F**),(**C,G**)}

the relation **SameFamily**, which is the set $\mathbf{P}^2$

Figure 1: The conceptualization $\mathcal{C}_1$

Objects: the set **P** of people {**A,B,C,D,E,F,G**}

Relations: the relation **FoundingFather**, which is the following subset of $\mathbf{P}^2$,
{(**A,B**),(**A,C**),(**A,D**),(**A,E**),(**A,F**),(**A,G**)}

the relation **SameFamily**, which is the set $\mathbf{P}^2$

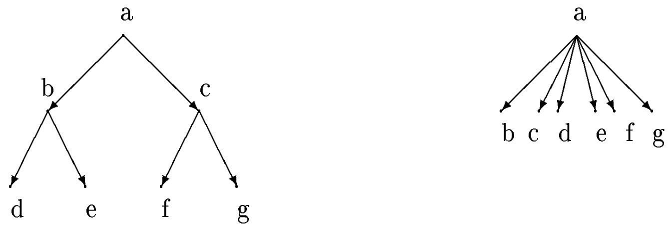
Figure 2: Another conceptualization $\mathcal{C}_2$

Definition 3 *A reformulation $\mathcal{C}_2$ of the conceptualization $\mathcal{C}_1$ is* good *with respect to a problem solver PS and time and space bounds S on the computation of the goal wff g in the canonical language $\mathcal{L}_{\mathcal{C}_2}$, if there is an encoding $\mathcal{E}$ of $\mathcal{C}_2$ that allows computation of g within S. The interpretation of g in $\mathcal{C}_2$ is the goal relation G.*

The problem of automating reformulations for computational efficiency is to find a good conceptualization with respect to the given correctness and goodness constraints specified with respect to a problem solver. Previous approaches to the reformulation include [Amarel, 1968], [Newell, 1965], [Korf, 1980], [Lowry, 1987] and [Riddle, 1986].
We take the kinship problem conceptualized in Figure 1 and its encoding in Figure 3. The goal relation is the **SameFamily** relation. The problem solver that works on this formulation of the problem is a depth first backward chainer (e.g., Prolog). The correctness constraint is the preservation of the **SameFamily** relation. The goodness constraints are: the new formulation should be able to solve any SameFamily query faster(in particular, we want these queries to be answered in O(1) time), also, the new formulation should only consume as much space as the old formulation. The answers that we expect are the conceptualization in Figure 2 and the encoding in Figure 3 as well as the intensional description of FoundingFather as $\forall xy.\ \mathsf{FoundingFather}(x,y) \equiv \mathsf{Ancestor}(x,y) \wedge \neg\exists z.\mathsf{Ancestor}(z,x)$

JUSTIFYING REFORMULATIONS

Our approach is to provide a unifying framework and a set of concepts that allow declarative specification of the knowledge that is required for automatic reformulation. The main stumbling block to automated reformulation is the fact that the intended interpretations of terms and symbols is never represented in the system itself. So a system has no logical basis for changing conceptualizations. A justification for a conceptualization examines the role of each conceptual element in computation of the goals. We use meta-theoretical justifications that tie the change in formulation directly to a change in computational properties. One such class of justifications is the class of *irrelevance explanations*. An irrelevance explanation proves that certain distinctions in the formulation are not logically necessary for the solution of a given class of questions. We say that an irrelevance claim holds in the meta-theory of a formulation, if it is the case that the distinction specified by that claim is irrelevant. The crux of our proposal is the transformation of these justifications into generative procedures for choosing new terms in order to improve system performance.



Father(a,b)	FoundingFather(a,b)
Father(a,c)	FoundingFather(a,c)
Father(b,d)	FoundingFather(a,d)
Father(b,e)	FoundingFather(a,e)
Father(c,f)	FoundingFather(a,f)
Father(c,g)	FoundingFather(a,g)
Father(x,y) $\Rightarrow$ Ancestor(x,y)	FoundingFather(z,x) $\wedge$ FoundingFather(z,y)
Ancestor(x,z) $\wedge$ Ancestor(z,y) $\Rightarrow$ Ancestor(x,y)	$\Longrightarrow$ SameFamily(x,y)
Ancestor(z,x) $\wedge$ Ancestor(z,y) $\Rightarrow$ SameFamily(x,y)	

Figure 3: The encodings E_1 of $\mathcal{C}_1$(left) and E_2 of $\mathcal{C}_2$ (right)

This is done by meta-theoretic reduction inferences that attempt to modify the formulation so that the irrelevance claims are no longer true of the new formulation. This is guided by a local optimisation principle called the *irrelevance principle* whose informal statement is: the minimal revision of a formulation in whose meta-theory the irrelevance claims are no longer true is a computationally better one.

IRRELEVANCE REFORMULATIONS

Returning to our example, notice that the reformulation can be explained by two irrelevance claims. One, as far as the SameFamily query is concerned, the distinction between immediate ancestry (Father) and non-immediate ancestry is irrelevant. Further, the actual identity of the common ancestor is irrelevant. The first irrelevance claim provides the justification for relabeling the Father tree as the Ancestor tree. The second irrelevance claim allows us to shortcircuit chains through the Ancestor tree. In what follows, we will state these irrelevance claims and provide reasoning mechanisms that enable us to carry out the reformulation steps.

A fact f is irrelevant to the goal schema g in the context of a set of sentences T, written as $Irrelevant(f, g, T)$, if perturbing the value of f in T does not affect that of g. To verify this, we construct a weakening of T that allows us to establish the truth of g without committing ourselves on f. An important class of weakenings of a set T of sentences is obtained by taking subsets of T that weaken its deductive closure. We call this category of irrelevance claims *weak irrelevance* claims.

Definition 4 *$f(x,z)$ is weakly irrelevant to $g(x,y)$ modulo the set of sentences T, written as WI($f(x,z), g(x,y),\ T$), if we can construct the set $T^{'}$ such that $T^{'} \subseteq T$ and $[\forall xy.\ T \models g(x,y) \equiv T^{'} \models g(x,y)$ where $[\forall z.\ T^{'} \not\models f(x,z)]]$ and $T^{'}$ is a maximal subset of T with this property.*

We will use the definition above to state the two irrelevance claims in our example.

IC1:$\forall xymn.\ Ancestor(x,y) \in T \Rightarrow WI(Father(x,y), SameFamily(m,n), T)$

IC2: $\forall xyzmn.\ Ancestor(x,y) \in T \wedge Ancestor(y,z) \in T \wedge Ancestor(x,z) \in T \quad \Rightarrow WI(Ancestor(y,z), SameFamily(m,n), T)$

We generate lemmas that satisfy Definition 4. These lemmas constitute the proof theory for WI [Subramanian, 1987]. Removal of irrelevant or redundant paths leads to savings both in space and time especially if the proofs of g that fail as a result of the irrelevance removal are the ones that are more complex. This notion is made precise in the following definition.

Definition 5 *f is computationally irrelevant to g given T, written as CI(f,g,T), if it is the case that WI(f,g,T), and the T' that results from the WI construction is such that Cost($T \vdash g$) $\geq$ Cost($T' \vdash g$).*

The two weak irrelevance claims IC1 and IC2 are computational irrelevance claims too. This is because, with respect to the proof-height metric on which our time cost model is based, both lead to shorter proofs of SameFamily.

THE RELEVANCE OF IRRELEVANCE

What is the role of the irrelevance principle in generating reformulations? We propose reduction inferences that operate on encodings of conceptualizations and that use irrelevance claims. The task of the reduction inference is to implement the minimization of distinctions suggested by the irrelevance principle. We introduce the relation *Reduce* that takes a formulation and an irrelevance claim and produces a new formulation in whose meta-theory the irrelevance claim does not hold.

Definition 6 *An irrelevance claim IC of the form $c \Rightarrow$ WI(f,g,T) holds in the meta-theory of T, if we can apply the T' construction of Definition 4 on T augmented to make c hold.*

In order to reduce a formulation with respect to an irrelevance claim, we apply the construction of Definition 4 until the irrelevance claim no longer holds in the meta-theory of the new formulation. When this happens, the T' returned by the construction is identical to T.

Theorem 1 *Applying Reduce iteratively with respect to a given WI claim until a fix point of the theory is reached, produces a new conceptualization that is correct and that is computationally better than the old one.*

Depending on whether the reduction process operates directly on the present formulation or on a description of it, we have two modes of reduction: extensional and intensional. Extensional reduction performs direct surgery on the formulation as dictated by the irrelevance claims and generates a new formulation that obeys both the goodness and the correctness constraints. Intensional reduction takes a description of a set of sentences T and some irrelevance claims and produces a new description of a theory T' in which those claims do not hold.

In the following we assume that the reduction with respect to **IC1** has already been performed, so that all Father facts are relabelled as Ancestor facts. Now we want to minimize the formulation using **IC2**. While the extensional reduction process laboriously rewires the family tree fact by fact, intensional reduction attempts to find a description of the facts left at the *end* of the extensional reduction. A description of the Ancestor facts at the start is: $\forall u.\ T \models \mathsf{Ancestor}(root, u)$. We wish to find the description of T' such that Reduce(T',**IC2**) $= T'$. $\mathsf{Ancestor}(root, u)$ is shorthand for the formula $\mathsf{Ancestor}(x, u) \wedge \neg\exists m.\ \mathsf{Ancestor}(m, x)$. The reasoning proceeds in two steps.

1. **IC2** preserves connectedness to the root.
   $\forall u.\ T \models \mathsf{Ancestor}(root, u) \Rightarrow Reduce(T, \mathbf{IC2}) \models \mathsf{Ancestor}(root, u)$
2. When the reduction terminates **IC2** can no longer be applied.
   By examining the preconditions of **IC2**, we see that the required condition is that all Ancestor links be one step long; i.e., $\neg\exists xyz.\ \mathsf{Ancestor}(x, y) \wedge \mathsf{Ancestor}(y, z)$.

Putting 1 and 2 together, we obtain the fact that all Ancestor links are connected to the root and that they are all one step long. That is, the remaining Ancestor facts satisfy: $\mathsf{Ancestor}(root, y)$ which is an abbreviation for $\mathsf{Ancestor}(x, y) \wedge \neg\exists m.\mathsf{Ancestor}(m, x)$ which is the required definition of the Founding-Father relation! This is the first general-purpose method of generating new relations guided by a meta-theoretic minimization principle.

CONCLUSIONS

The chief result from the analysis is the development of a 2-step meta-theoretic method for the automation of abstraction reformulations for computational efficiency: generate irrelevance claims that are true of the given formulation with respect to the given class of goals, and reduce the formulation by the irrelevance principle.

Reduction methods have been developed [Subramanian, 1988 and 1989] that generate a large class of reformulations: the generation of Thevenin equivalents, the transformation of the Fibonacci computation to tail recursion, several compiler optimizations, encoding variations of the FoundingFather example, macro-operators and macro-objects. Effective *mechanization* of this method has been accomplished on the class of abstraction reformulations called *elimination of intermediates.* The irrelevance claims for this class have a very special form: a gross step in the computation renders an intermediate step irrelevant. The meta-theoretic irrelevance lemmas of the irrelevance calculus allow us to generate these claims in a goal-directed fashion. The reductions also have a specific form: they either involve wholesale removal of relations, or the introduction of new relations and new objects that are definable in terms of the existing ones. The theory has been empirically verified by constructing a prototype of a first-principles irrelevance reformulator. The framework explored in this paper has the potential to unify disparate abstraction phenomena as instances of a powerful invariant in granularity shifts: the most economical description is the one that uses concepts of the largest granularity consistent with the correctness constraints.

References

S. Amarel. On the Representation of Problems of Reasoning about Action, In *Machine Intelligence 3*, edited by D. Michie, University of Edinburgh Press (1968)

R. Korf. Towards a Model of Representation Changes, *Artificial Intelligence*, Vol 14 (1980)

M. Lowry. The Abstraction/Implementation Model of Problem Reformulation, in the Proceedings of IJCAI-87, Milan (1987)

A. Newell. Limitations of the current stock of ideas about problem solving, in the Conference on Electronic Information Handling edited by Kent and Taulbee (1965)

P.J. Riddle. An Overview of Problem Reduction: a Shift of Problem Representation, in the Proceedings of the Workshop on Knowledge Compilation, Oregon (1986)

S.J. Russell & D. Subramanian. Mutual Constraints on Representation and Inference, Workshop on Learning, Meta-level reasoning and logics, Sesimbra, Portugal (1988)

D. Subramanian & M. Genesereth. The Relevance of Irrelevance, in the Proceedings of IJCAI-87, Milan (1987)

D. Subramanian. Toward a Theory of Justified Reformulations, Logic Group Tech Report 88-15, Stanford University (1988)

D. Subramanian. A Theory of Justified Reformulations, PhD thesis, Stanford University, March 1989.