IRST

Istituto per la Ricerca Scientifica e Tecnologica
I 38100 Trento – Loc. Panté di Povo – tel. 0461–314444
Telex 400874 ITCRST – Telefax 0461–302040

Tree Subsumption: Reasoning with Outlines

F. Giunchiglia, T. Walsh

May 1992
Technical Report # 9205-01

In *Proceedings ECAI-92, 10th European Conference on Artificial Intelligence*, Vienna, Austria, 1992.

Istituto Trentino di Cultura

Tree Subsumption: Reasoning with Outlines*

Fausto Giunchiglia
Mechanized Reasoning Group
IRST, Istituto Ricerca Scientifica e Tecnologica, 38100 Trento, Italy
DIST, Universita' di Genova, Via Opera Pia 11A, 16145 Genova, Italy
fausto%irst@uunet.uu.net

Toby Walsh
Mathematical Reasoning Group
Department of Artificial Intelligence
University of Edinburgh
80 South Bridge, Edinburgh, Scotland
T.Walsh@ed.ac.uk

Abstract

This paper presents the beginnings of a theory for reasoning with proof outlines; proof outlines are abstract descriptions of proofs, providing the important subgoals on the way to the final conclusion. The use of proof outlines can both aid explanation and reduce search.

*The first author's research is part of the MAIA project which is under development at IRST (described in: "An integrated approach to Artificial Intelligence" by L. Stringa, IRST technical report No. 9012-11). The second author is supported by a SERC PostDoctoral Fellowship. All the members of the Mathematical Reasoning group in Edinburgh and the Mechanized Reasoning group in Trento are thanked for their contributions to this work. Both authors would like to thank Alan Bundy especially.

1 Introduction

Mathematicians often describe a proof by its outline. As well as aiding explanation, outlines are a powerful means of controlling search. Polya, for instance, proposes a strategy for solving mathematical problems based on the use of outlines: *"... the main achievement in the solution of a problem is to conceive the idea of a plan ... The plan gives a general outline; we have to convince ourselves that the details fit into the outline."* [9][pages 8 and 12]

The purpose of this paper is to propose a notion of proof outline for guiding search. We will illustrate the generality of this approach by means of some examples including outlines for proofs of the Diamond Lemma and Gödel's First Incompleteness theorem.

2 Trees

To describe proofs and their outlines, we first give some notions for describing the structure of proofs. Proofs, whether they be written on a piece of paper or represented internally in a computer, usually have some sort of hierarchical or tree-like structure. Proof outlines also usually have a hierarchical structure. We will therefore give some notions for representing and describing this structure.

Proofs are special types of formulae trees. A tree, Π is a directed non-cyclic graph with a distinguished node, the root, and with an unique path from the root to any node. A formulae tree is a tree with each node labelled by a wff. The label of the root node is the root formula, whilst the labels of the leaf nodes are the leaf formulae. Sometimes we will draw formulae trees graphically using the notation:

$$\frac{\Pi_1 ... \Pi_n}{\varphi}$$

where $\Pi_1, ..., \Pi_n$ are formulae trees and φ is the root formula. Note that this notation distinguishes the left to right ordering of formulae trees. A deduction is a finite tree in which the wff labelling every node is derived from the wffs labelling the nodes connected to it by the valid application of an inference rule. A proof is a deduction tree in which the leaf formulae are either discharged assumptions or axioms. A path of a formulae tree is a sequence of wffs, with each wff a label of a node, and the nodes connected

in order by arcs. A branch is a path starting at the root formula and ending at a leaf formula. The depth of a tree Π, written $|\Pi|$, is the length of the longest branch in the tree. A path contains another path if it mentions the same wffs (and possibly more) in the same order. $\mathcal{N}(\varphi, \Pi)$ is the number of occurrences of the wff φ in the tree Π. The weight of a tree Π, written $\|\Pi\|$, is the number of wffs in the tree. That is, $\|\Pi\| = \sum_{\varphi} \mathcal{N}(\varphi, \Pi) = |nodes(\Pi)|$. One very useful notion is the subtree relation. If Π is the tree $\frac{\Pi_1, ... \Pi_n}{\varphi}$ then Π, and $\Pi_1, ...\Pi_n$ plus their subtrees are the (only) subtrees of Π. That is,

Definition 1 (Subtree) *:* Π_1 *is a* **subtree** *of* Π_2 *iff* Π_1 *is a tree with* $nodes(\Pi_1) \subseteq nodes(\Pi_2)$, $arcs(\Pi_1) \subseteq arcs(\Pi_2)$, *and* $leaves(\Pi_1) \subseteq leaves(\Pi_2)$. *Dually* Π_2 *is a* **supertree** *of* Π_1.

Arcs describe the local structure of a tree. They also induce a global structure on trees; we can define when one node is below another, when it is above another, or (when neither of these two cases is true) when it is adjacent to another.

Definition 2 ($\preceq$) *: For any two nodes* n_1, n_2 *in a formulae tree,* n_1 *is* **below** n_2, *written* $n_1 \preceq n_2$ *iff*

- $n_1 = n_2$;
- $\langle n_1, n_3 \rangle \in arcs(\Pi)$ *and* $n_3 \preceq n_2$.

The below relation is a weak partial order. If $n_1 \preceq n_2$ but $\neg(n_2 \preceq n_1)$ then we say that n_1 is strictly below n_2, written $n_1 \prec n_2$. When n_1 is below n_2 the wff labelling n_1, $label(n_1)$ is beneath that labelling n_2, $label(n_2)$. We will therefore also say that the wff $label(n_1)$ is below the wff $label(n_2)$. Additionally, we will say that n_1 is above n_2 iff $n_2 \preceq n_1$, and that n_1 is adjacent to n_2, written $n_1 \bowtie n_2$ iff n_1 is not below or above n_2. If n_1 is adjacent to n_2 then the wff labelling n_1, $label(n_1)$ is in a distinct subtree to that labelling n_2, $label(n_2)$. Adjacency is an incomparability relation; that is, exactly one of $n_1 \prec n_2$, $n_2 \prec n_1$ and $n_1 \bowtie n_2$ holds. As an example, consider the tree:

$$\dfrac{\dfrac{a \qquad b}{c} \qquad d}{e}$$

In this tree, $c \preceq c$, $c \preceq a$, $e \prec b$, and $a \bowtie d$.

3 Outlines

We can now define a notion of proof outline. The relationship between a proof and an outline is one of *monotonicity* – the outline can be built simply by deleting formulae from the proof. The relationship therefore satisfies the following two conditions:

- **the preservation of the nodes**; all the formulae in the outline (plus possibly more) appear in the proof;
- **the preservation of the global structure**; the below, above and adjacency relations should be maintained.

These considerations motivate the following very important definitions.

Definition 3 (Tree subsumption) *:* Π_1 **subsumes** Π_2*, written* $\Pi_1 \subseteq \Pi_2$*, iff there is an injective map,* $\tau : nodes(\Pi_1) \rightarrow nodes(\Pi_2)$ *such that,*

- $label(n) = label(\tau(n))$
- $n_1 \preceq n_2$ *iff* $\tau(n_1) \preceq \tau(n_2)$

Definition 4 (Proof outline) *: If* $\Pi_1 \subseteq \Pi_2$ *and* Π_2 *is a proof, then* Π_1 *an* **outline** *of* Π_2*.*

The intuitive interpretation of these definitions is that the same wffs occur in Π_1 as in Π_2 (first condition of Definition 3) with the same global ordering (second condition of Definition 3). We need a *map* between nodes so that we can skip nodes in the tree Π_2; not every wff in Π_2 corresponds to a wff in Π_1. As examples, consider the following four trees.

$$\Pi_1 = \dfrac{\dfrac{a \quad a}{b} \quad c}{d} \qquad \Pi_2 = \dfrac{a \quad a}{b} \qquad \Pi_3 = \dfrac{a \quad c}{d}$$

$$\Pi_4 = \dfrac{\dfrac{a}{a}}{b}$$

Π_2 and Π_3 both subsume Π_1. Note that Π_2 is a subtree of Π_1 but that Π_3 is not; tree subsumption is therefore a more general property than the subtree relation. Finally, Π_2 does not subsume Π_4 even though all the occurrences of wffs, and all the branches in Π_2 appear in Π_4.

Tree subsumption is a preorder being transitive, and reflexive. It is also a *monotonicity* property on the depth, the weight, the number of formulae occurrences, the ordering of wffs and the branches.

Theorem 1 (Monotonicity) *: If* $\Pi_1 \subseteq \Pi_2$ *then*

- $|\Pi_1| \leq |\Pi_2|$
- $\|\Pi_2\| \leq \|\Pi_2\|$
- $\mathcal{N}(\varphi, \Pi_1) \leq \mathcal{N}(\varphi, \Pi_2)$
- *if* φ *is below* ψ *in* Π_1 *then* φ *is below* ψ *in* Π_2
- *if* b *is a branch of* Π_1 *then some branch of* Π_2 *contains* b.

Note that monotonicity on depth and on the ordering of wffs follows from the monotonicity property on branches, and that monotonicity on the weight follows from monotonicity on the number of formulae occurrences. Note also that this theorem does not reverse; the trees, Π_2 and Π_4 in the example above are a counter-example. These five properties only guarantee monotonicity on the nodes and partial monotonicity on the structure; that is, from these properties it follows that $n_1 \preceq n_2$ implies $\tau(n_1) \preceq \tau(n_2)$. To get tree subsumption and monotonicity on the whole structure, we also need $\tau(n_1) \preceq \tau(n_2)$ implying $n_1 \preceq n_2$ (or, equivalently, a monotonicity property on adjacency: $n_1 \bowtie n_2$ implying $\tau(n_1) \bowtie \tau(n_2)$). Indeed, *tree subsumption is as weak a relation on trees as is possible whilst being monotonic with respect to the nodes and the global structure.* No part of the definition of tree subsumption can be weakened without losing monotonicity on the structure or the nodes. For example, if we drop the injectivity requirement on τ then tree subsumption is not a monotonicity property on the nodes (and thus, for example, on the weight).

Other monotonicity properties on trees have been defined in the past; all of them are too strong for describing outlines. For instance, the subtree relation is also a monotonicity property but is too strong:

Theorem 2 *: If* Π_1, Π_2 *are two trees and* Π_1 *is a subtree of* Π_2 *then* $\Pi_1 \subseteq \Pi_2$.

As in the examples given earlier, the reverse does not hold. There are two main differences between tree subsumption and the subtree relation. With tree subsumption, wffs anywhere in the subsuming tree can be skipped and the ordering of trees is ignored, whilst with the subtree relation, only wffs in the supertree beneath the subtree are skipped and the ordering of the trees is fixed.

Further proof of the naturalness of tree subsumption, and of its intrinsic interest, is that it is closely related to the conventional notion of tree isomorphism. Indeed, if we simply make τ, the mapping between nodes, a bijection instead of an injection, we get tree isomorphism.

Definition 5 (Tree isomorphism) *:* $\Pi_1 \simeq \Pi_2$ *iff there is a bijective map* $\tau : nodes(\Pi_1) \rightarrow nodes(\Pi_2)$

- $label(n) = label(\tau(n))$
- $n_1 \preceq n_2$ *iff* $\tau(n_1) \preceq \tau(n_2)$

Intuitively, two trees are isomorphic iff they are equal up to reordering of their subtrees. Trees that are isomorphic have the same depth, the same weight, the same formulae occurrences, the same ordering of formulae and the same branches. Trivially, tree isomorphism is the equivalence relation generated by tree subsumption.

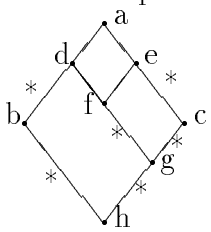
Theorem 3 *:* $\Pi_1 \simeq \Pi_2$ *iff* $\Pi_1 \subseteq \Pi_2$ *and* $\Pi_2 \subseteq \Pi_1$.

This result adds weight to our claim that tree subsumption is a natural relationship between the structure of trees.

4 Using Outlines

We now turn to the problem of how we use proof outlines to guide theorem proving. The intuitive idea is that of *jumping between islands*: the outline suggests the major subgoals which need to be bridged in building a proof. Outlines thereby "divide-and-conquer" the search.

Consider, for example, a proof of the Diamond Lemma that local Church-Rosser implies global Church-Rosser for any well-founded relation r. A relation r is local Church-Rosser iff $r(a,b)$ and $r(a,c)$ implies there exists some d such that $r(b,d)$ and $r(c,d)$. A relation r is global Church-Rosser iff $r*(a,b)$ and $r*(a,c)$ implies there exists some d such that $r*(b,d)$ and $r*(c,d)$ where $r*$ is the transitive closure of r. David Barker-Plummer and Sidney Bailin have described [1] a theorem proving system called `Grover` which is guided in its search for a proof of the Diamond Lemma by an outline generated automatically from a diagram similar to the following (unlabelled arcs represent the relation r whilst those labelled with * represent $r*$).



The proof of the Diamond Lemma given by Barker-Plummer and Bailin uses transfinite induction. After unfolding the definitions of global and local Church-Rosser, their goal is to show that there exists some h with $r*(b,h)$ and $r*(c,h)$ given $r*(a,b)$ and $r*(a,c)$. They first show that there exists d and e immediately below a; that is, such that $r(a,d)$ and $r(a,e)$. By local Church-Rosser, there exists f such that $r(d,f)$ and $r(e,f)$. By the induction hypothesis, there exists g such that $r*(f,g)$ and $r*(c,g)$. Hence, by the induction hypothesis again, there exists h such that $r*(b,h)$ and $r*(g,h)$. Finally, by transitivity of $r*$, $r*(b,h)$ and $r*(c,h)$.

Collecting together all these subgoals suggests the following proof outline:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{r(a,d) \quad r(a,e)}{r(d,f) \wedge r(e,f)}}{r*(f,g) \wedge r*(c,g)}}{r*(b,h) \wedge r*(g,h)}}{r*(b,h) \wedge r*(c,h)}$$

This is an outline of the Diamond Lemma as it tree subsumes a complete proof. It is not a proof in its own right as it only contains the key subgoals in the proof. Each step in the outline represents several proof steps. For example, consider the last step of the outline:

$$\dfrac{r*(b,h) \wedge r*(g,h)}{r*(b,h) \wedge r*(c,h)}$$

This corresponds to a much larger part of the proof where the transitivity of $r*$ is used to deduce that $r*(c,h)$ given $r*(c,g)$ and $r*(g,h)$. To build a complete proof, we *refine* the outline by bridging the gaps between formulae. The outline provides the key subgoals we must prove on the way to the final goal. Proof outlines therefore allow us to do a form of *middle-out reasoning* [4]: instead of starting at the axioms and applying inference rules until we reach the goal (forward reasoning), or starting at the goal and generating subgoals until we reach axioms (backwards reasoning), we identify some key steps in the proof and fill in the gaps between them. We thereby construct the proof from the middle-out.

5 Outlines from Abstraction

In the last section, an outline of a proof was constructed by analysing a diagram. Another very general method for building outlines is with abstraction. The proof to an abstracted theorem can serve as an outline for a proof of the original unabstracted (or "ground") theorem. We refine this outline by adding in extra steps removed by abstraction.

In [5] we argued that abstraction can be seen as the mapping of one representation of a problem, the ground representation onto a new and simpler representation, the abstract representation. Problems can be represented by axiomatic formal systems. An abstraction $f : \Sigma_1 \Rightarrow \Sigma_2$ is then simply a mapping from one formal system, Σ_1 (often called the "ground space") to another formal system, Σ_2 (often called the "abstract space") [6]. It is given by a triple consisting of Σ_1, Σ_2 and a function, f which maps formulae in Σ_1 onto formulae in Σ_2. An interesting restriction is to those abstractions whose abstract proofs are *guaranteed* to be outlines of ground proofs. For example, one very important class of abstractions is given by the following definition:

Definition 6 (PI-abstraction) *: An abstraction, $f : \Sigma_1 \Rightarrow \Sigma_2$ is said to be a PI-abstraction iff, for any proof Π_1 of a theorem φ in Σ_1, there exists a proof Π_2 of $f(\varphi)$ in Σ_2 with $\Pi_2 \subseteq f(\Pi_1)$.*

By $f(\Pi_1)$ we mean the formulae tree built by applying f to every formula in Π_1.

The essential idea in using such abstractions is:

(i) we *abstract* the formula to be proved;

(ii) we *find* an abstract proof;

(iii) we *unabstract* the abstract proof; this gives us an outline for the ground proof;

(iv) we *refine* this outline by filling in the gaps between the formulae.

Note that it is the *unabstraction* of the abstract proof, and not the abstract proof, which provides the proof outline for the ground proof. We'll illustrate this procedure by means of an example in which we reason about the properties of various containers. We will use a predicate abstraction which collapses together objects with similar properties. For example, "$box(x)$", "$bottle(x)$" and "$glass(x)$" all map onto the generic "$container(x)$". Consider the ground goal "$movable(a) \wedge graspable(a)$" which gets abstracted onto the abstract goal "$shiftable(a) \wedge shiftable(a)$". Let us suppose that we have performed the steps (i) and (ii), giving the abstract proof:

$$\dfrac{\dfrac{container(a) \quad container(x) \rightarrow shiftable(x)}{shiftable(a)}}{shiftable(a) \wedge shiftable(a)}$$

The third step is to unabstract the abstract proof to give an outline. This can be made as simple as possible by producing the "minimal" outline, whose abstraction is tree isomorphic to the abstract proof. As an abstraction is usually a many-to-one mapping on the language and the abstract proof corresponds to many different (minimal) outlines. For example, up to tree isomorphism, the abstract proof of the example has 36 minimal outlines.

We tackle this problem by *constructing a schema which represents all the (minimal) outlines up to tree isomorphism.* With a predicate abstraction, this construction uses second order sorted meta-variables (other types of abstractions require other types of sorted and unsorted, first and second order meta-variables). These meta-variables allow us to represent any wff in the ground language that abstracts onto a given abstract wff. For example, "$X : container(a)$", where X is a second order sorted meta-variable, represents any atomic wff which abstracts onto "$container(a)$". The elements of the sort are the predicates collapsed into $container$, namely box, $bottle$, and $glass$. This technique allows us *to delay* the unabstraction of parts of a wff until, during the refinement, we have a better knowledge of the other instantiations. The outline resulting from this step is thus:

$$\dfrac{\dfrac{X : container(a) \quad Y : container(x) \rightarrow Z : shiftable(x)}{U : shiftable(a)}}{V : shiftable(a) \ \wedge \ W : shiftable(a)}$$

In the fourth and final step, this outline is instantiated and refined to give the following ground proof of "$movable(a) \wedge graspable(a)$".

$$\dfrac{\dfrac{\dfrac{bottle(a) \quad bottle(x) \rightarrow graspable(x)}{graspable(a)} \quad graspable(x) \rightarrow movable(x)}{movable(a)}}{movable(a) \ \wedge \ graspable(a)}$$

Note that the abstract proof is not a subtree of the abstraction of the ground proof as, in refining the outline, we had *to add nodes to the middle of the tree*. This is a type of *middle-out reasoning*.

We are currently building an abstract proof checker for performing these four steps. This abstract proof checker is built on top of `GETFOL` which is an extension and re-implementation of the `FOL` proof checking system[7, 10]. Actually, `GETFOL` is far more than a conventional proof checker. For instance, it includes derived inference rules and complex deciders. A single proof step in `GETFOL` can thus represent very complex reasoning. We could perhaps call it an "interactive theorem prover". However, we shall stick to "proof checker" as we wish to emphasize our interest in the interaction with the system rather than in the automation of the construction of proofs.

We have implemented facilities within `GETFOL` for abstracting a problem representation. The user can call upon libraries of abstractions and abstraction schemata. Our next goal is to provide facilities for unabstracting proofs and refining outlines; these steps are currently performed by hand. This approach looks very promising. We have used this approach to develop a proof of Gödel's First Incompleteness Theorem using as guidance an outline that was built by unabstracting an abstract proof which is half the size of the ground proof. Under the abstraction used, more than half the axioms become redundant and the proof halves in size; more importantly, every step in the abstract proof corresponds to (the abstraction of) an important step in the ground proof. That is, the abstract proof tree subsumes (the abstraction of) the ground proof. Or equivalently, an unabstraction of the abstract proof is an outline of the ground proof.

6 Related work

Bledsoe defines a notion of proof plan for use in his analogy guided theorem prover which can be seen as a restricted type of proof outline [2]. A proof plan is essentially a (finite) sequence of proof steps. Each proof step contains a formula and an optional plan; every formula in the proof plan logically follows from the conjunction of previous formulae. That is, a proof plan, $\mathcal{P}$ is a sequence: $S_1, S_2 ... S_n$ where S_i is a proof step $\langle \varphi_i, \mathcal{P}_i \rangle$, φ_i is an intermediate subgoal, $\mathcal{P}_i$ is itself a plan (possibly empty), and $\bigwedge_{i<k} \varphi_i \rightarrow \varphi_k$ for $k = 1$ to n. A proof plan can thus be seen as the proof outline:

$$\begin{array}{c} \Pi_1 \\ \hline \varphi_1 \\ \hline \vdots \\ \hline \Pi_n \\ \hline \varphi_n \end{array}$$

where Π_i is the proof outline associated with the proof plan $\mathcal{P}_i$.

Bledsoe claims that the use of such proof plans resembles the way humans discover proofs by breaking down a hard proof into easier subgoals, which themselves might also be broken down into easier subgoals. Such a problem solving strategy seems very promising. Bledsoe's analogy guided theorem prover can prove some impressive theorems (*eg.* the completeness of lock resolution using as guidance a proof plan derived from the analogical proof of the completeness of general ground resolution).

A notion of proof plan has also been proposed by Alan Bundy [3] for describing high-level proof strategies. Proof plans are built in terms of methods, meta-level descriptions of compound proof steps. Methods encode discrete proof "chunks". A particular proof plan can capture many different proofs as methods include parameters that need to be instantiated to give an object-level proof. Proof plans are not, however, proof outlines – a proof plan is a meta-level description of a proof whilst a proof outline is an object-level description. Nevertheless, many of the motivations behind this work are similar, and proof plans, like proof outlines, can aid explanation and reduce search greatly.

Plaisted has also defined a shape correspondence between ground and abstract proofs, written "$\Pi_1 \rightarrow_f \Pi_2$" (page 63 of [8]), which is closely related to tree subsumption. Indeed, if we generalize tree subsumption to the relation, "$\subseteq^*$" in which every wff in one tree *logically subsumes* a wff in the other

tree, shape correspondence implies tree subsumption. Shape correspondence is, however, weaker than tree subsumption as the abstract proof must be of the *same* depth as the ground proof.

Theorem 4 *: If* $\Pi_1 \rightarrow_f \Pi_2$ *then* $\Pi_1 \subseteq^* f(\Pi_2)$ *and* $|\Pi_1| = |\Pi_2|$.

7 Conclusions

This paper presents the beginnings of a theory for reasoning with proof outlines. We have defined *tree subsumption*, a very general monotonicity property between a proof and its outline. We have explored several ways in which proof outlines can be built (from diagrams, by abstraction, and so on). We are currently implementing these ideas in `GETFOL`, a powerful proof checking system. As well as aiding explanation, outlines seem a powerful means for controlling search.

References

[1] D. Barker-Plumer and S.C. Bailin. Graphical Theorem Proving: An Approach to Reasoning with the Help of Diagrams Proceedings of ECAI-92, 1992.

[2] W.W. Bledsoe. A precondition prover for analogy. Technical report, Computer Science Department, University of Texas at Austin, 1990.

[3] A. Bundy. The Use of Explicit Plans to Guide Inductive Proofs. In Proceedings of *CADE9*. Springer-Verlag, 1988.

[4] A. Bundy, A. Smaill, and J. Hesketh. Turning eureka steps into calculations in automatic program synthesis. In *Proceedings of UK IT 90*, 1990.

[5] F. Giunchiglia and T. Walsh. Abstract Theorem Proving. In *Proceedings of the 11th IJCAI*, 1989.

[6] F. Giunchiglia and T. Walsh. A Theory of Abstraction. Research paper 516, Dept of AI, University of Edinburgh, 1990. Accepted to *Artificial Intelligence*.

[7] F. Giunchiglia and R.W. Weyhrauch `FOL` User Manual - `FOL` version 2 Technical Report 9107-05, DIST, University of Genova, Genova, Italy, 1991.

[8] D.A. Plaisted. Theorem proving with abstraction. *Artificial Intelligence*, 16:47–108, 1981.

[9] G. Polya. *How to Solve It.* Princeton University Press, 1945.

[10] R.W. Weyhrauch. Prolegomena to a theory of Mechanized Formal Reasoning. *Artificial Intelligence*, 13(1), 1980.