Relevance and Abstraction

David A. Plaisted[1]

UNC Chapel Hill
Chapel Hill, UNC, North Carolina
USA
plaisted@cs.unc.edu

When proving theorems from thousands or millions of clauses, it is imortant to select those clauses that are relevant to the particular theorem to be proved. Many people have been looking for ways to select clauses that are relevant, that is, likely to be involved in a proof. Especially for large clause sets, selecting such relevant clauses can lead to a dramatic decrease in the proof time. The problem is that it is difficult to find good relevance criteria and some of those that perform well seem to have some arbitrary features and parameters that have to be carefully adjusted.

Here we propose a new method of producing relevance criteria, based on the use of abstractions. In past work[5], the author has made use of abstractions as a guide to the structure of the proof. Now it appears to the author that abstractions are more useful for selecting relevant clauses than for giving information about the structure of the proof.

There has been related work by various people [3] [2] [1], [7], and [4], and also see the references to recent work on relevance in a previous paper [6] by the author. The paper [3] by Lopez *et al* uses an identical definition of an abstraction which they call an over-approximating abstraction and applies it to relevance using various abstraction mappings. The authors find a proof in the abstract space if possible, map it back to the original space, and attempt to obtain a refutation in the original space. Kaliszyk *et al* [4] consider a variety of features of clauses and train their system on past proofs from large axiom sets. For training they use distance-weighted k-NN (k-nearest neighbors) and naive Bayes. Pudlak[7] uses finite models in a manner similar to that proposed here. Fuchs [1], [2] investigates the use of lemmas to shorten model elimination and connection tableaux proofs and chooses which lemmas to save based on various similarity measures between the lemmas and the desired goal clauses.

Basically, given a set S of clauses and looking for a refutation, one desires a subset S' of S that is likely to be unsatisfiable. Then one can search for a refutation from S' instead of S. Abstractions provide a method to obtain such sets S'.

We define an abstraction to be a mapping α from clause sets to clause sets such that if clause set S is unsatisfiable then so is $\alpha(S)$. The idea of abstractions is to find a mapping α such that $\alpha(S)$ is simpler than S so that proof search is easier. Then the proofs from $\alpha(S)$ can be used as a guide to finding proofs from S. There are many such abstractions. One is the *propositional abstraction* which replaces each atom $P(t_1, \cdots, t_n)$ by the predicate symbol P. Another abstraction replaces each term t_i by its top level symbol, and if the term is a variable, then top level symbols of ground instances are considered, roughly speaking. One can also delete certain subterms or rename function and predicate symbols.

Now, instead of using abstractions as a guide to the structure of a proof, we propose to use them to generate relevant clause sets. Suppose $T \subseteq \alpha(S)$. T can be generated from $\alpha(S)$ by any of the relevance methods that have been proposed so far. Then if T is unsatisfiable, one can take a set S' such that $\alpha(S') = T$ as a candidate for a relevant subset of S. If S' were unsatisfiable, then T would also be unsatisfiable, so such an S' is a good candidate for a relevant subset of S. In addition, if $\alpha(S)$ is a set of ground clauses, then theorem proving in $\alpha(S)$ is much easier than S because one can use some version of $DPLL$ there.

Also, one can use many abstractions $\alpha_i, 1 \leq i \leq k$ together. Then if S' is a subset of S and for all i, $1 \leq i \leq k$, $\alpha_i(S')$ is unsatisfiable, one has considerable evidence that S' is also unsatisfiable. A minimal such subset S' can be found as follows, assuming S is unsatisfiable:

```
relset(S) →
  S' ← S
  T ← S
  while T ≠ ∅ do
    choose C in T
    T ← T \ {C}
    if α_i(S' \ {C}) is unsatisfiable for all i then S' ← S' \ {C}
  od
  return(S')
end relset;
```

Such a subset S' can then be tested for unsatisfiability using a theorem prover. This approach may be inefficient for large clause sets S; in that case, one may want to start from a small subset S' of S and gradually add clauses to it until one obtains an S' such that $\alpha_i(S')$ is unsatifiable for all i, $1 \leq i \leq k$.

If all $\alpha_i(S)$ are propositional, then one can apply some version of DPLL in each set $\alpha_i(S)$ to obtain unsatisfiable sets $T_i \subseteq \alpha_i(S)$ of propositional clauses. Let S_i' be such that $\alpha_i(S_i') = T_i$ and let S' be $S_1' \cup \cdots \cup S_k'$. Then $\alpha_i(S')$ is unsatisfiable for all i, $1 \leq i \leq k$ so that S' can be given to a theorem prover, or possibly $relset(S')$.

Semantics can be used as a promising candidate for such abstractions. Let $\mathcal{F}$ be the set of function and constant symbols in S with a new constant symbol added if S has no constant symbols.

Definition 0.1. *For a clause $C \in S$, let $Gr(C)$ be the set of ground instances of C over $\mathcal{F}$. Let I be an interpretation of S and D be its domain. Define the* semantic abstraction $\Psi(I)$ *as follows: For each $B \in Gr(C)$ let B^I be obtained by replacing each top-level ground subterm t of B by its interpretation t^I in I. This gives a clause B^I whose literals are of the form $P(d_1, \cdots, d_n)$ or $\neg P(d_1, \cdots, d_n)$ where the d_i are elements of D. Such clauses can be considered as ground clauses. Then $\Psi(I)(C)$ is $\{B^I : B \in Gr(C)\}$, and $\Psi(I)(S)$ is $\bigcup\{\Psi(I)(C) : C \in S\}$.*

The function $\Psi(I)$ on clause sets is an abstraction mapping. If D is finite then $\Psi(I)(S)$ is a finite set of ground clauses. Then the set $\Psi(I)(S)$ can be computed in bounded time from S without considering every ground instance of C for C in S, by interpreting the variables of C in all possible ways, roughly speaking. Such abstractions $\Psi(I)$ can be used to generate relevant subsets of S, as explained above.

For example, consider the clauses S equal to $\{\{P(a)\}, \{P(x) \rightarrow P(f(x))\}, \{\neg P(f(f(a))\}\}$. If we let the domain be the natural numbers and interpret a as 4 and f as the successor function, then we obtain the abstract clauses $\{P(4)\}, \{\neg P(6)\}, \{P(0) \rightarrow P(1)\}, \{P(1) \rightarrow P(2)\}, \{P(2) \rightarrow P(3)\}, \cdots$. The interpretation of P does not affect the abstract set of clauses; only the interpretations of the function and constant symbols matter. In the abstract space, the four clauses $\{P(4)\}, \{\neg P(6)\}, \{P(4) \rightarrow P(5)\}, \{P(5) \rightarrow P(6)\}$ are unsatisfiable. Mapping these back to the original clause set S gives all three clauses in it. In this case the abstraction function does not help, but if one added a lot of irrelevant clauses to S then the abstraction approach could help a lot. If one chooses the domain to be the integers modulo three and interpret a as 0 and f as adding one modulo three, one obtains the finite set of clauses $\{\{P(0)\}, \{\neg P(2)\}, \{P(0) \rightarrow P(1)\}, \{P(1) \rightarrow P(2)\}, \{P(2) \rightarrow P(0)\}\}$.

Such semantic abstractions are even complete in a sense:

Theorem 0.1. *If S is a satisfiable set of clauses then there is an interpretation I of S such that $\Psi(I)(S)$ is satisfiable.*

Proof. If S is satisfiable then there is an interpretation I such that $I \models S$. Then $\Psi(I)(S)$ is satisfiable. □

But note that if I has an infinite domain then $\Psi(I)(S)$ may have an infinite number of ground clauses.

One may also have a support set for S and in this case even if $\alpha(S')$ is unsatisfiable, if S' does not intersect the support set, one knows that S' is satisfiable independent of the evidence from the abstraction. One also knows that minimal unsatisfiable sets do not have pure literals. In addition, if S' is minimal unsatisfiable, then it should be connected by alternating paths. So there may be extra conditions on S' that have to be fulfilled for it to be a minimal unsatisfiable subset of S. It is an interesting problem to modify this approach to produce an S' satisfying these conditions and also having the property that $\alpha_i(S')$ is unsatisfiable for all i, $1 \leq i \leq k$.

The use of semantic abstractions has the feel of proving a theorem for an example, and may help to explain the use of examples by humans in theorem proving.

We propose that semantic abstractions should be used more than they are, and that more than one should be used at the same time to increase their power. We also propose that somehow semantic abstractions be incorporated into the TPTP problem set. Also, there should be more interaction between the theorem proving communities and the AI communities in the use of abstractions and relevance for very large knowledge bases. There should also be more emphasis on such large knowledge bases which have very short proofs for many queries, because relevance methods can do especially well in this case. An example is the knowledge base used for the Watson system. Certainly humans perform many short inferences quickly from a huge knowledge base, for example, in language understanding. We also propose that alternating path relevance can be used both in the abstract space and in the original set of clauses. Many of the currently used relevance approaches have parameters that have to be carefully set for them to perform well. We propose that relevance methods should be developed that do not require such a fine tuning of so many parameters or even tuning a single parameter to several decimal places.

References

[1] Marc Fuchs. System Description: Similarity-Based Lemma Generation for Model Elimination. In C. Kirchner and H. Kirchner, editors, *Proc. of the 15th CADE, Lindau*, volume 1421 of *LNAI*, pages 33–37. Springer, 1998.

[2] Marc Fuchs. *A Relevancy Based Approach for Lemma Use in Connection Tableau Calculi.* PhD thesis, Institut für Informatik, TU München, 1999.

[3] Julio Cesar Lopez Hernandez and Konstantin Korovin. An abstraction-refinement framework for reasoning with large theories. In *International Joint Conference on Automated Reasoning*, pages 663–679. Springer, 2018.

[4] Cezary Kaliszyk, Josef Urban, and Jirí Vyskocil. Efficient semantic features for automated reasoning over large theories. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 3084–3090. AAAI Press, 2015.

[5] David A. Plaisted. Theorem proving with abstraction. *Artificial Intelligence*, 16(1):47–108, 1981.

[6] David A. Plaisted. Properties and extensions of alternating path relevance - I. *CoRR*, abs/1905.08842, 2019.

[7] P. Pudlák. Semantic selection of premisses for automated theorem proving. In *ESARLT*, 2007.