System Description: PTTP+GLiDeS
Semantically Guided PTTP

Marianne Brown and Geoff Sutcliffe

School of Information Technology
James Cook University
{marianne,geoff}@cs.jcu.edu.au

Introduction

PTTP+GLiDeS is a semantically guided linear deduction theorem prover, built from PTTP [9] and MACE [7]. It takes problems in clause normal form (CNF), generates semantic information about the clauses, and then uses the semantic information to guide its search for a proof.

In the last decade there has been some work done in the area of semantic guidance, in a variety of first order theorem proving paradigms: SCOTT [8] is based on OTTER and is a forward chaining resolution system; CLIN-S [3] uses hyperlinking; RAMCS [2] uses constrained clauses to allow it to search for proofs and models simultaneously; and SGLD [11] is a chain format linear deduction system based on Graph Construction. Of these, CLIN-S and SGLD need to be supplied with semantics by the user. SCOTT uses FINDER [8] to generate models, and RAMCS generates its own models.

The Semantic Guidance

PTTP+GLiDeS uses a semantic pruning strategy that is based upon the strategy that can be applied to linear-input deductions. In a completed linear-input refutation, all centre clauses are FALSE in all models of the side clauses. This leads to a semantic pruning strategy that, at every stage of a linear-input deduction, requires all centre clauses in the deduction so far to be FALSE in a model of the side clauses. To implement this strategy it is necessary to know which are the potential side clauses, so that a model can be built. A simple possibility is to choose a negative top clause from a set of Horn clauses, in which case the mixed clauses are the potential side clauses. More sensitive analysis is also possible [4, 10]. Linear-input deduction and this pruning strategy are complete only for Horn clauses. Unfortunately, the extension of this pruning strategy to linear deduction, which is also complete for non-Horn clauses, is not direct. The possibility of ancestor resolutions means that centre clauses may be TRUE in a model of the side clauses.

In PTTP+GLiDeS, rather than placing a constraint on entire centre clauses, a semantic constraint is placed on certain literals of the centre clauses. The input clauses other than the chosen top clause of a linear deduction are named
the model clauses. In a completed linear refutation, all centre clause literals that have resolved against input clause literals are required to be FALSE in a model of the model clauses. TRUE centre clause literals must be resolved against ancestor clause literals.

PTTP+GLiDeS implements linear deduction using the Model Elimination [6] (ME) paradigm. ME uses a chain format, where a chain is an ordered list of A- and B-literals. The disjunction of the B-literals is the clause represented by the chain. Input chains are generated from the input clauses and are composed entirely of B-literals. The chains that form the linear sequence are called the centre chains. A-literals are used in centre chains to record information about ancestor clause in the deduction. The input chains that are resolved against centre chains are called side chains.

PTTP+GLiDeS maintains a list of all the A-literals created throughout the entire deduction. This list is called the A-list. The pruning strategy requires that at every stage of the deduction, there must exist at least one ground instance of the A-list that is FALSE in a model of the model clauses. The result is that only FALSE B-literals are extended upon, and TRUE B-literals must reduce. Figure 1 shows an example of a PTTP+GLiDeS refutation. The problem clauses are \{\sim \text{money} \lor \text{tickets(buy)}, \sim \text{tickets(sell)} \lor \text{money}, \text{money} \lor \text{tickets(X)}, \sim \text{money} \lor \sim \text{tickets(X)}\}. The clause \sim \text{money} \lor \sim \text{tickets(X)} is chosen to form the top chain, so that the other three clauses are the model clauses. The model M is \{\text{money}, \text{tickets(buy)}, \sim \text{tickets(sell)}\}. The A-list is shown in braces under the centre chains.

Since the work described in [1], PTTP+GLiDeS has been enhanced to order the use of side chains, using the model of the model clauses. The model is used to give a score to each clause as follows: If there are \(N\) ground domain instances of a clause \(C\) with \(k\) literals, then for each literal \(L\), \(n_L\) is the number of TRUE instances of \(L\) within the \(N\) ground instances. \(L\) is given the score \(\frac{n_L}{N}\). The score for the clause \(C\) is \(\frac{1}{Nk} \sum_{L=1}^{k} n_L\). The clause set is then ordered in descending order of scores. This gives preference to clauses that have more TRUE literal instances in the model. The use of these clauses leads to early pruning and forces the deduction into areas more likely to lead to a proof.

Implementation

PTTP+GLiDeS consists of a modified version of PTTP version 2e and MACE v1.3.3, combined with a csh script. It requires the problem to be presented in both PTTP and OTTER formats. The OTTER format file is processed so that it contains only the model clauses, and is used by MACE.

Initially the domain size for the model to be generated by MACE is set to equal the number of constants in the problem. If a model of this size cannot be found, the domain size is reset to 2 and MACE is allowed to determine the domain size. If no model is found PTTP+GLiDeS exits. If a model is found, the modified PTTP is started. The modified PTTP uses the model to reorder the clause set, then transforms the reordered clauses into Prolog procedures.
Fig. 1. A PTTP+GLiDeS refutation
that implement the ME deduction and maintain the A-list. A semantic check is performed on the A-list after each extension and reduction operation. If the A-list does not have an instance in which every literal evaluates to FALSE in the model provided by MACE, then the extension or reduction is rejected.

Performance

Testing was carried out on 541 “difficult” problems from the TPTP problem library [12] v2.1.0. Both PTTP and PTTP+GLiDeS were tested on the same problems under the same conditions. Experiments were carried out on a SunSPARC 20 server using ECLiPSe v3.7.1 as the Prolog engine. A CPU time limit of 300 seconds was used. The results are summarized in Table 1. MACE failed to generate a model in 272 cases, and so PTTP+GLiDeS couldn’t attempt those problems. Of those 269 problem where models were generated, worst performance is on Horn problems: all of the problems solved by PTTP and not PTTP+GLiDeS are Horn. MACE tends to generate trivial models (positive literals TRUE) for Horn problems. If the top centre clause is negative then, for a Horn clause set, a trivial model does not lead to any pruning. With the additional overhead of the semantic checking this leads to poor performance by PTTP+GLiDeS. Of the 269 models produced by MACE, 155 were effective, i.e., resulted in some pruning. Of the problems with effective models solved by both PTTP and PTTP+GLiDeS, in 13 out of 18 cases PTTP+GLiDeS had a lower inference count; in some cases significantly lower. This is shown by the fact that the average number of inferences for PTTP+GLiDeS is 2.5 times smaller than that of PTTP. This shows that the pruning is having a positive effect. PTTP+GLiDeS performs best on non-Horn problems. Table 2 shows some results for non-Horn problems where PTTP+GLiDeS performed better than PTTP. For these problems even trivial models can be of assistance.

<table>
<thead>
<tr>
<th>Total number of problems:</th>
<th>541 (311/230)</th>
<th>(Horn/Non-Horn)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU time limit:</td>
<td>300 s</td>
<td></td>
</tr>
<tr>
<td>Number of models generated:</td>
<td>269 (227/42)</td>
<td></td>
</tr>
<tr>
<td>Number of problems solved from 269:</td>
<td>PTTP 66 (60/6)</td>
<td>PTTP+GLiDeS 59 (51/8)</td>
</tr>
<tr>
<td>Number of effective models generated:</td>
<td>155 (120/35)</td>
<td></td>
</tr>
<tr>
<td>Number of problems solved from 155:</td>
<td>PTTP 21 (16/5)</td>
<td>PTTP+GLiDeS 20 (13/7)</td>
</tr>
</tbody>
</table>

For the 18 problems (from 155) solved by both systems:

<table>
<thead>
<tr>
<th>Average CPU Time:</th>
<th>PTTP 34.24</th>
<th>PTTP+GLiDeS 60.18</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Number of Inferences:</td>
<td>PTTP 119634.28</td>
<td>PTTP+GLiDeS 47812.22</td>
</tr>
<tr>
<td>Average Number of Rej. Inferences:</td>
<td>389678</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Summary of experimental data.
<table>
<thead>
<tr>
<th>Problem</th>
<th>Model</th>
<th>PTTP+GLiDeS CPU Inferences</th>
<th>PTTP CPU Inferences</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Time</td>
<td>Rejected Inferences</td>
</tr>
<tr>
<td>CD003-3</td>
<td>non-Trivial</td>
<td>68.5</td>
<td>64232</td>
</tr>
<tr>
<td>CD012-3</td>
<td>Trivial</td>
<td>32.0</td>
<td>49220</td>
</tr>
<tr>
<td>GRP009-1</td>
<td>Trivial</td>
<td>248.3</td>
<td>404198</td>
</tr>
<tr>
<td>SYN071-1</td>
<td>non-Trivial</td>
<td>70.1</td>
<td>84908</td>
</tr>
</tbody>
</table>

Table 2. Results for some non-Horn problems where PTTP+GLiDeS out-performs PTTP.

With respect to ordering of the clause set, experiments have been carried out using both ascending and descending with respect to the truth score. Initially it was thought that ordering the clause set in ascending order of truth score (from ‘less TRUE’ to ‘more TRUE’) would lead the search away from pruning and therefore towards the proof. This turns out not to be the case. While the results are not statistically significantly different in terms of rejected inferences and inferences, descending ordering solved 4 more problems overall, of which 3 had effective models. As solving problems is the most significant measure of a theorem prover’s ability this shows that pruning early is more effective.

Conclusion

In those cases where a strongly effective model has been obtained, results are good. This leads to the question, “what makes a model effective?” At present the first model generated by MACE is used. If the characteristics of a strongly effective model can be quantified then it should be possible to generate many models and select the one most likely to give good performance.

PTTP is not a high performance implementation of ME, and thus the performance of PTTP and PTTP+GLiDeS is somewhat worse than that of current state-of-the-art ATP systems. This work has used PTTP to establish the viability of the semantic pruning strategy. It is planned to implement the pruning strategy in the high performance ME implementation SETHEO [5], in the near future.

On the completeness issue, this prover prunes away proofs which contain complementary A-literals on different branches of the tableau. In the few cases examined to date, another proof that conforms to this extended admissibility rule has always been found. Whether there is always another such proof is not known.

References


