Homogeneous Sets of ATP Problems

Matthias Fuchs
Automated Reasoning Group, RSISE
Australian National University
Canberra, ACT 0200, Australia
Email: fuchs@arp.anu.edu.au

Geoff Sutcliffe
Department of Computer Science
University of Miami
P.O. Box 248154, Coral Gables, FL 33124, USA
Email: geoff@cs.miami.edu

Abstract

This paper describes how the homogeneity of sets of ATP problems can be measured with respect to the performance of ATP systems. Measuring homogeneity is important as a basis for empirical evaluation of ATP systems and problems. A machine learning approach has been used to differentiate between types of problems in situations where heterogeneity is apparent.

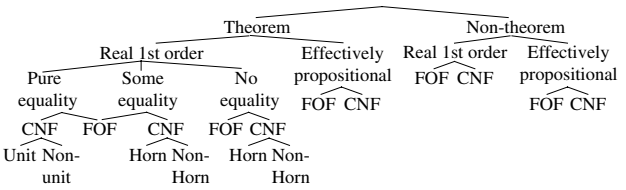
Introduction

In order to build more powerful automated theorem proving (ATP) systems, it is important to know which systems, and hence which techniques (understanding that a system is a collection and combination of techniques), work well for what types of problems. For classical first order ATP, the evaluation of ATP systems is necessarily empirical. Inextricably intertwined with the evaluation of ATP systems is the evaluation of ATP problem difficulty.

Methodologies for the empirical evaluation of ATP systems and problems are presented in (Sutcliffe & Suttner 2001). Due to the specialization of ATP systems and techniques to problems with certain characteristics, e.g., special techniques are deserved for problems with equality, evaluation of ATP systems must be done in the context of sets of ATP problems that are reasonably homogeneous with respect to the systems. These sets of problems are called *Specialist Problem Classes* (SPCs). SPCs are based on logical, language, and syntactic characteristics of the problems. The characteristics that have so far been identified as relevant are: theoremhood - theorems vs non-theorems, order - effectively propositional vs real 1st order, equality - no equality vs some equality vs pure equality, form - clause normal form vs first order form, Hornness - Horn vs non-Horn, and unit equality - unit equality vs non-unit pure equality. (The split between theorems and non-theorems is necessary for system evaluation. In application, where theoremhood is not known in advance, the user can then choose the best prover or disprover, according to the outcome they hope to achieve.) Based on these characteristics 14 SPCs have been defined, as indicated by the leaves of the tree in Figure 1.

The SPCs are named using mnemonic acronyms, abbreviating theorem to THM, non-theorem to SAT, real 1st order to RFO, essentially propositional to EPR, pure equality to PEQ, some equality to SEQ, no equality to NEQ, unit equality to UEQ, non-unit pure equality to NUE, clause normal form to CNF, first order form to FOF, Horn to HRN, and non-Horn to NHN.

Figure 1: Specialist Problem Classes



This paper describes a method for checking the homogeneity of a set of ATP problems, e.g., a SPC, with respect to performance data on those problems. The method can also be used to identify homogeneous subsets of a heterogeneous set of problems. The paper also shows how a machine learning technique can be used to identify problem characteristics that differentiate between different types of problems. These methods and techniques have been used to check the homogeneity of the SPCs shown in Figure 1, and hence to affirm the basis for the system and problem evaluations done using the TPTP problem library (Sutcliffe & Suttner 1998).

Cliques of Problems

Given a SPC, if there are some problems that are solved by system A_1 but not by A_2, and there are some problems that are solved by A_2 but not by A_1, then the performances of A_1 and A_2 are contradictory, and the SPC is not homogeneous with respect to A_1 and A_2. In contrast, if there is no such contradictory performance then the SPC is homogeneous with respect to A_1 and A_2. If a SPC is homogeneous with respect to n ATP systems $A_1,\ldots,A_n$, then the systems can be totally ordered according to the sets of problems they solve, thus providing a meaningful evaluation of the relative abilities of the systems. Note that homogeneity does not prevent there being a substantial variation in the number of

problems solved by the systems. To measure the homogeneity of a SPC, the performance data of ATP systems can be examined for contradictory behavior as follows.

Basic Compatibility and Homogeneity Measures

For each problem in a SPC, an ATP system either solves it or fails to solve it, within realistic resource limits ((Sutcliffe & Suttner 2001) shows that there exist such resource limits, and a linear increase in resource allocation beyond these limits does not result in the solution of significantly more problems). These two cases are represented by `S` and `F`. Each problem P in a SPC is associated with a performance vector $v^P \in \{\mathtt{S}, \mathtt{F}\}^n$, where v_i^P indicates the performance of ATP A_i on P. Two performance vectors are *compatible* to the extent that the systems' performances do not contradict each other. Two performance vectors u and v are compatible if there are no two vector positions i and j such that $u_i = \mathtt{S}$ and $u_j = \mathtt{F}$ but $v_i = \mathtt{F}$ and $v_j = \mathtt{S}$. For example, for $n = 3$, the performance vectors `SSS`, `SSF`, `SFF`, and `FFF` are pairwise compatible, whereas `SSF` and `FSS` are each compatible with `SSS` but are not compatible with each other. (Finer grained levels of compatibility are considered below.)

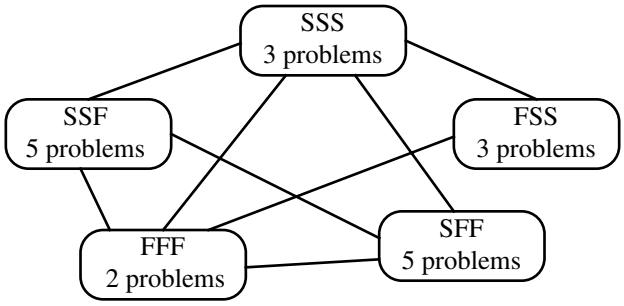
The *compatibility graph* for an SPC has a node for each performance vector that occurs for the SPC. Each node holds its performance vector and the group of problems with that performance vector. An edge connects two nodes in a compatibility graph if the respective performance vectors are compatible. The cliques of the graph then identify homogeneous sets of problems. The homogeneity of a SPC is expressed in terms of the maximal clique size in the graph. Two *homogeneity measures* are defined, using two ways of measuring clique size. The first measure is the ratio of the number of problems in the maximal clique's nodes and the number of problems in the SPC, i.e., measuring the homogeneity of the actual problems in the SPC. The second measure is the ratio of the number of nodes in the maximal clique and the number of nodes in the graph. This measure acknowledges that the number of problems may be biased by the source of the problems, and thus puts the problems in a node into an equivalence class, i.e., measuring homogeneity of the problem types.

If one clique covers most of a SPC, then the SPC can be considered homogeneous. Conversely, there is a strong argument for splitting a SPC if it generates two or more rather large cliques. Figure 2 shows a compatibility graph, in which each node is shown with its performance vector and the number of problems. The maximal clique is {`SSS`, `SSF`, `SFF`, `FFF`}, containing 15 of the 18 problems and 4 of the 5 nodes. The problem set is thus 83% homogeneous by problem count and 80% homogeneous by node count.

Refinements of the Measures

The notion of compatibility introduced above is strictly Boolean: two nodes are either compatible or not, depending on which systems solve and which systems fail to solve the associated problems. Although decent ATP systems have reasonably stable performance characteristics, it is a feature of ATP that small changes in problems can sometimes lead

Figure 2: Simple Compatibility and Clique Example



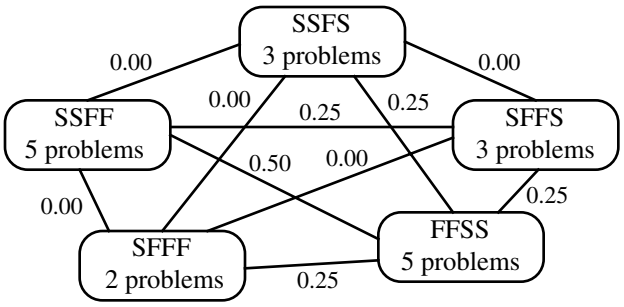
to strange changes in system performance. Such aberrant behavior may cause the performance vectors of two problems to be incompatible, when for the underlying purpose of system evaluation the problems really do fall into the same homogeneous SPC. To make the homogeneity measures robust to such noise, a *degree of compatibility* between two performance vectors is defined. The connectivity of the compatibility graph is then determined with respect to a degree of compatibility, with a subsequent effect on the homogeneity measure.

The degree of compatibility between two performance vectors u and v is defined as the minimal fraction of ATP systems that have to exchange `S` for `F`, or vice versa, to make u and v fully compatible. For example, if $u = \mathtt{SSFS}$ and $v = \mathtt{FFSS}$ (i.e., there are four systems), by changing the third position of u (v) to `S` (`F`), the two vectors become fully compatible. The degree of compatibility therefore is $1/4 = 0.25$. The degree of compatibility can range from 0 to 0.5 (less than 0.5 if there is an odd number of ATP systems). The value 0 indicates full compatibility (no change required), as is the case for the performance vectors `SSFF` and `SFFF`. The value 0.5 indicates complete incompatibility, as is the case for `SSFF` and `FFSS`.

When constructing a compatibility graph G, a *compatibility threshold* $d \in [0; 0.5]$ is specified, which allows two nodes to be connected if their degree of compatibility is less than or equal to d. In this way absolute compatibility need not be expected, but excessive contradictory behavior can be avoided in order to support a reasonable evaluation of (relative) system performances. Note that for $d = 0$ we have the original case that requires full compatibility. Figure 3 shows a compatibility graph, in which the edges are annotated with the degree of compatibility between the nodes. With a compatibility threshold of 0.00 the maximal clique is {`SSFS`, `SSFF`, `SFFF`} containing 10 of the 18 problems and 3 of the 5 nodes. With an increased compatibility threshold of 0.25 the maximal clique is {`SSFS`, `SFFF`, `SFFS`, `FFSS`} containing 13 of the 18 problems and 4 of the 5 nodes.

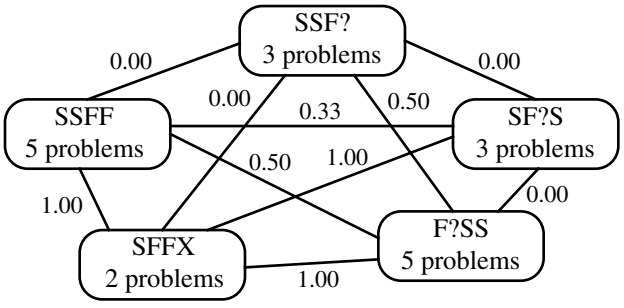
The development so far assumes that every system has attempted every problem in the SPC being considered. In reality this may not be the case, for two reasons. Firstly, if a problem is added to the TPTP after a certain ATP system was tested, there is no performance data available for that system

Figure 3: Degree of Compatibility Example



on that problem. In the performance vectors such cases are denoted by `?`. When computing the compatibility of two performance vectors, positions where either vector has a `?` entry are ignored. For example, the degree of compatibility of `SSF?` and `F?SS` is $1/2 = 0.5$. Secondly, certain ATP systems are incapable of attempting certain types of problems, e.g., systems based on unfailing completion can deal with only unit equality problems, and are therefore not tested on some problems. In the performance vectors such cases are denoted by `X`. If two problems P and Q have performance vectors v^P and v^Q respectively, and there is an ATP system A_i (position i) so that $v_i^P = \texttt{X}$ and $v_i^Q \in \{\texttt{S}, \texttt{F}\}$, this indicates that P and Q should be considered to be inherently different in nature. Consequently these two problems should not be in the same SPC and hence should not be connected in the compatibility graph G. This is achieved by assigning a degree of compatibility 1.0 to v^P and v^Q if the above situation occurs. 1.0 can be viewed as "infinity", given that the degree of compatibility is bounded by 0.5 under normal circumstances. Figure 4 shows the degrees of compatibility between the five nodes, some of which have `?` and `X` entries.

Figure 4: Missing Data Example



Finding Maximal Cliques

Maximal clique detection is known to be NP complete (Mehlhorn 1984). Hence, except for small graphs, it is necessary to resort to efficient approximation algorithms. A fair amount of research has gone into approximation algorithms of this kind, most recently in the field of evolutionary computation (Haynes 1998). The algorithm used to find the maximal cliques in the compatibility graphs is best characterized as a greedy or steepest ascent hill-climbing algorithm. Starting with the graph $G_0 = G$, the node of minimal degree is removed from G_i, resulting in G_{i+1}. If there is more than one such node, the one containing the smallest number of problems is removed, thus creating a bias towards cliques containing more problems. The process is stopped as soon as G_i is a clique. After a clique is identified, this procedure is applied to the graph consisting of the remaining nodes, until the original graph G has been partitioned into $k \geq 1$ cliques $C_1, \ldots, C_k$. C_{i+1} may contain more problems than C_i since the number of problems is only a secondary criterion. C_{i+1} may also have more nodes than C_i because the algorithm is an approximation algorithm and is therefore not guaranteed to find the clique with the maximal number of nodes. The test results demonstrate that this simple algorithm performs satisfactorily well for the compatibility graphs.

Testing the SPCs for Homogeneity

The homogeneities of the 14 SPCs shown in Figure 1 have been measured, using data from systems tested on the TPTP since the release of v2.0.0 on 5 June 1997, up to 14 September 2000. Data from 16 systems was used. For each SPC the homogeneity measures by problem count and node count were computed for compatibility thresholds 0.000, 0.0625, 0.1250, 0.2500, and 0.500. Table 1 shows sample results for two SPCs. The first column gives the compatibility threshold, the second column gives the number of nodes in the maximal clique, the third column expresses that as a percentage of the number of nodes in the graph, the fourth column gives the number of problems in the maximal clique, and the last column expresses that as a percentage of the number of problems in the SPC.

THM_RFO_SEQ_CNF_HRN				
C.T.	67 nodes		381 problems	
0.0000	20	(29%)	246	(64%)
0.0625	38	(56%)	325	(85%)
0.1250	55	(82%)	349	(91%)
0.2500	66	(98%)	380	(99%)
0.5000	67	(100%)	381	(100%)
SAT_RFO_CNF				
C.T.	38 nodes		88 problems	
0.0000	14	(36%)	41	(46%)
0.0625	14	(36%)	41	(46%)
0.1250	25	(65%)	56	(63%)
0.2500	30	(78%)	68	(77%)
0.5000	30	(78%)	68	(77%)

Table 1: Homogeneity statistics for the SPCs

For all but one of the SPCs, a homogeneity measure in excess of 80% by problem count is reached with a compatibility threshold of 0.125 (and in many cases with a compatibility threshold of 0.0625). The exception is `SAT_RFO_CNF`,

as shown in Table 1. Here the homogeneity measure remains below 100% even with a compatibility threshold of 0.5000, indicating that some nodes are incompatible due to X entries in performance vectors. Closer examination shows that some systems had (quite reasonably) been tested on only the unit equality problems in that SPC. Excluding those systems from consideration produces a homogeneity measure of 86% by both node count and problem count with a compatibility threshold of 0.1250. This suggests a possible split for the SPC, based on equality characteristics.

For all but four of the SPCs (excluding SAT_RFO_CNF discussed above), a homogeneity measure in excess of 80% by node count is reached with a compatibility threshold of 0.125. The exceptions are THM_RFO_NEQ_CNF_HRN, THM_RFO_NEQ_CNF_NHN, THM_RFO_SEQ_CNF_NHN, and THM_RFO_PEQ_CNF_UEQ. For THM_RFO_NEQ_CNF_NHN and THM_RFO_SEQ_CNF_NHN, homogeneity measures of 78% and 81% by node count are reached respectively with a compatibility threshold of 0.14. For THM_RFO_NEQ_CNF_HRN, 12 of the 15 nodes in the second clique have only one problem. Interestingly, there is one node with 24 problems, all but one of which are blocks world problems from the TPTP's PLA domain. The node is excluded from the first clique due to different performance from two tableau based (i.e., strongly goal oriented) systems, which are particularly well suited to the formulation of the blocks world problems. At present there seems to be no simple way to capture such subtle problem characteristics for use in defining SPCs. For THM_RFO_PEQ_CNF_UEQ, the second and subsequent cliques found have reasonably large numbers of nodes, but all nodes have only very few (typically one or two) problems. This low homogeneity by node count is somewhat surprising, given the specialized nature of unit equality problem solving. One possible cause is systems' different selections of term orderings.

Generating Homogeneous Problem Sets

The technique of finding maximal cliques in a compatibility graph has also been used to divide up the TPTP into subsets that are reasonably homogeneous with respect to the performance data. The subsets are formed from the problems in the nodes of the cliques found in the compatibility graph for the performance data over the whole TPTP. Using a compatibility threshold of 0.125, the 4229 problems in the TPTP are divided into 35 homogeneous subsets, with 3972 problems falling into the first seven subsets. All of the remaining 28 subsets contain 25 or less problems, and are ignored as insignificant.

The generated homogeneous subsets have been compared to the existing SPCs. If any generated subset is a strict superset of a SPC, then the SPC may be considered to be 100% homogeneous. If multiple SPCs fall within a single homogeneous subset, then the union of those SPCs is homogeneous with respect to the systems' performances, and merging them may be appropriate. In contrast, if a SPC is split across multiple subsets, then the SPC is apparently heterogeneous and may need to be split. In order to make such judgements, for each homogeneous subset and each SPC, the ratio of the size of their intersection and the size of the SPC has been computed. The results are shown in Table 2 (the last eight rows are for the THM_RFO SPCs).

SPC	Homogeneous Subset							Probs
	1	2	3	4	5	6	7	
SAT_EPR_CNF	0.00	0.00	0.00	0.00	0.00	0.86	0.00	139
SAT_EPR_FOF	0.00	0.00	0.00	0.00	0.99	0.00	0.00	83
SAT_RFO_CNF	0.03	0.00	0.00	0.00	0.00	0.57	0.00	88
SAT_RFO_FOF	0.00	0.00	0.00	0.00	1.00	0.00	0.00	9
THM_EPR_CNF	0.81	0.00	0.00	0.00	0.00	0.00	0.15	401
THM_EPR_FOF	0.00	0.00	0.00	0.00	1.00	0.00	0.00	235
_EQU_FOF	0.00	0.00	0.00	0.00	0.96	0.00	0.00	323
_NEQ_FOF	0.00	0.00	0.00	0.00	1.00	0.00	0.00	21
_NEQ_CNF_HRN	0.81	0.08	0.00	0.03	0.00	0.01	0.01	379
_NEQ_CNF_NHN	0.77	0.08	0.00	0.06	0.00	0.00	0.01	430
_SEQ_CNF_HRN	0.78	0.00	0.00	0.13	0.00	0.00	0.03	381
_SEQ_CNF_NHN	0.88	0.05	0.00	0.02	0.00	0.00	0.02	1194
_PEQ_CNF_NUE	0.80	0.03	0.00	0.04	0.00	0.00	0.05	122
_PEQ_CNF_UEQ	0.00	0.00	0.85	0.00	0.00	0.00	0.00	424
Problems	2418	128	359	119	656	174	118	4229

Table 2: Correlation between Problem Cliques and SPCs

The first subset encompasses 2418 of the 3558 CNF problems in the TPTP, including most of those in the THM_EPR_CNF, THM_RFO_NEQ_CNF_HRN, THM_RFO_NEQ_CNF_NHN, THM_RFO_SEQ_CNF_HRN, THM_RFO_SEQ_CNF_NHN, and THM_RFO_PEQ_CNF_NUE SPCs. This shows that there are techniques (and hence systems) that are effective for all these problem types. The relative homogeneity of these problems types has also been noted in the context of the CADE ATP System Competition (Sutcliffe 2001), in which all these problem types are used together in the MIX division of the competition. The third subset identifies the unit equality SPC. The fifth subset covers all the FOF SPCs. Evidently the techniques suitable for one type of FOF problem are adequate for most types. The sixth subset identifies the CNF non-theorems, both the effectively propositional problems and the real first order ones. The only SPC that does not have a large fraction contained within only one problem clique is SAT_RFO_CNF. This heterogeneity is the same as discussed earlier, in the context of SPC homogeneity.

Using Machine Learning to Differentiate SPCs

The preceding sections show that the SPCs in Figure 1, formed using the problem characteristics listed in the introduction, are mostly homogeneous. In situations where some heterogeneity is apparent, it is useful to have a method of identifying problem characteristics that differentiate between the types of problems. The clique approach to measuring homogeneity assigns graph nodes, and hence performance vectors and problems, to cliques. This process can be considered to be a problem classification process, with cliques representing classes. Taking this viewpoint, machine learning (ML) techniques can be used to obtain a classifier based on problem characteristics. The classifier then provides the information needed to differentiate between types

of problems. This information may also be further used for the selection of a suitable ATP system or search guiding heuristic, as done in (Fuchs 1997). There are many ML techniques dealing with supervised classification, but not all of them are suitable for this purpose. When determining the SPCs for ATP system and problem evaluation, it is important that the classifier be in a comprehensible form, so that it is intuitive to ATP researchers and users. Conventional *decision-tree* algorithms appear to be the best choice, since the way they perform classification is easily presentable to and understandable by a human reader. `C4.5` (Quinlan 1993) is one of the most popular classification systems based on decision trees, and has been used here.

Classification methods in general, and `C4.5` in particular, require that a feature vector be associated with each object to be classified. Each feature captures a certain property of the objects, and expresses that property with a numerical value.[1] Given a set of features $a_1, \ldots, a_m$, each problem P is then associated with its feature vector $(a_1(P), \ldots, a_m(P))$. The features used to differentiate between types of problems are syntactic problem characteristics, which are supplied with each TPTP problem.

The use of a classifier provides useful information when examining apparently heterogeneous SPCs. For example, Figure 5 shows the output of `C4.5` that suggests that the SPC `SAT_RFO_CNF` can be split on equality characteristics (the feature `Lits==EqL` indicates if the number of literals equals the number of equality literals, i.e., determining if problems are pure equality problems). Another use is to apply the classifier to the cliques generated from the TPTP, shown in Table 2, in order to identify the characteristics that differentiate the cliques.

Figure 5: Differentiating in `SAT_RFO_CNF`

```
Simplified Decision Tree:

Lits==EqL = 0: CL01 (43.0)
Lits==EqL = 1:
|   Dp <= 2 : CL01 (9.0)
|   Dp > 2 :
|   |   Sz <= 4 : CL02 (22.0/2.0)
|   |   Sz > 4 : CL01 (2.0)
```

Conclusion

This paper describes how the homogeneity of sets of ATP problems can be measured with respect to the performance of ATP systems. The method developed assigns problems to nodes in a graph, and finding homogeneous sets of problems is reduced to finding maximal cliques in the graph. An approximation algorithm for finding the maximal cliques has proved satisfactory, thus overcoming the NP-completeness of finding maximal cliques in general. Some extensions to the basic idea have made the measurement robust to the realities of empirical data collection in ATP. In addition, a machine learning approach has been used to differentiate between types of problem in situations where heterogeneity is apparent.

The techniques developed are important, as they can be used to check the homogeneity of the Specialist Problem Classes (SPCs) used as a basis for system and problem evaluation using the TPTP. The testing done shows that the SPCs are apparently almost all highly homogeneous. In the exceptional case of the SPC `SAT_RFO_CNF`, where heterogeneity is apparent, equality has been identified as the problem characteristic that differentiates between the types of problems. As a result the SPCs can now be refined to take this into account.

[1] Other types of values can also be used, but numerical (integer) values are sufficient here.

References

Fuchs, M. 1997. Automatic Selection of Search-guiding Heuristics. In D., D., ed., *Proceedings of the 10th Florida Artificial Intelligence Research Symposium*, 1–5. Florida AI Research Society.

Haynes, T. 1998. Perturbing the Representation, Decoding, and Evaluation of Chromosomes. In Koza, J.; Banzhaf, W.; Chellapilla, K.; Deb, K.; Dorigo, M.; Fogel, D.; Garzon, M.; Goldberg, D.; Iba, H.; and Riolo, R., eds., *Proceedings of the 3rd Annual Conference on Genetic Programming*, 122–127. Morgan Kaufmann.

Mehlhorn, K. 1984. *Data Structures and Algorithms 2: Graph Algorithms and NP-Completeness*. Monographs on Theoretical Computer Science. Springer-Verlag.

Quinlan, R. 1993. *C4.5 Programs for Machine Learning*. Morgan Kaufmann.

Sutcliffe, G., and Suttner, C. 1998. The TPTP Problem Library: CNF Release v1.2.1. *Journal of Automated Reasoning* 21(2):177–203.

Sutcliffe, G., and Suttner, C. 2001. Evaluating General Purpose Automated Theorem Proving Systems. *Artificial Intelligence* 131(1-2):39–54.

Sutcliffe, G. 2001. The CADE-17 ATP System Competition. *Journal of Automated Reasoning* 27(3):227–250.