The Practice of Clausification in Automatic Theorem Proving

Geoff Sutcliffe
Department of Computer Science
James Cook University, Australia
Email: geoff@cs.jcu.edu.au
Phone: +61 77 814622, FAX: +61 77 814029

Stuart Melville
Department of Computer Studies
M.L. Sultan Technikon, South Africa
Email: stuart@wpogate.mlsultan.ac.za
Phone: +27 31 3085339, FAX: +27 31 3085355

Abstract. In the process of resolution based Automatic Theorem Proving, problems expressed in First Order Form (FOF) are transformed by a clausifier to Clause Normal Form (CNF). This research examines and compares clausifiers. The boundaries between clausification, simplification, and solution search are delineated, and common clausification and simplification operations are documented. Four known clausifiers are evaluated, thus providing insight into their relative performance, and also providing baseline data for future evaluation of clausifiers.

Keywords. Automated theorem proving, Resolution, Clausifiers.

C.R. Categories. F.4.1, I.2.3

1. Introduction

Automatic Theorem Proving (ATP) is the study and development of computer programs that build proofs of theorems. These programs are called *ATP systems*. The inputs to an ATP system are axioms, hypotheses, and a conjecture. The output is a proof object that shows that the conjecture is a logical consequence of the axioms and hypotheses, i.e., it is a theorem. See any of [2, 15, 41] for an introduction to the topic.

ATP problems are expressed in some formal language. Most commonly the problems are expressed in a logic, ranging from classical propositional logic to more exotic logics. Current research in ATP is dominated by the use of classical logic, at the propositional and 1st order levels. These logics, and proof within these logics, are well understood and documented (see almost any mathematical logic text, e.g., [3]). In particular, proof within propositional logic is decidable (and NP-complete [4]) while at the 1st order level it is semi-decidable (but beyond any complexity bound; see, e.g., [5]). Henceforth attention is limited to classical logics.

Although the details of the syntax for writing 1st order logic vary from author to author (see [28] for a survey), the same constituents are used consistently. In short, *terms* are either *variables* or *functions*. Functions are built from a *functor* with terms as arguments; the number of arguments is the functor's *arity* (a function of 0 arguments is often called a *constant*). *Atoms* are structurally the same as functions, but use *predicate symbols* rather than functors. Propositional logic is obtained by excluding variables and functions from the language and allowing predicate symbols only with arity zero, i.e., propositional logic is a strict subset of 1st order logic. Atoms are the simplest type of *formula* in 1st order logic. More complex formulae are built up by applying *connectives* and *quantifiers* to existing formulae. Commonly used connectives are negation, disjunction, conjunction, implication, and equivalence. The two standard quantifiers are existential quantification, and universal quantification. In this paper the Prolog syntax for terms and atoms is used, with variables starting with uppercase, and functors and predicate symbols starting with lowercase. The connectives are ~ for negation, ∨ for disjunction, ∧ for conjunction, → for implication, ↔ for equivalence. The quantifiers are ∃ for existential quantification, and ∀ for universal quantification. For example:

```
∀X(word(X) → ∃Y(statement(Y) ∧ defines(Y,concept_of(X))))
```

In this paper upper case *Zapf Chancery* letters are used to denote arbitrary formulae.

ATP problems expressed using the full expressive power of 1st order logic are said to be in *First Order Form* (FOF). ATP systems for FOF problems are necessarily complex, in order to deal with all aspects of the FOF language. In 1964 the *resolution* inference rule was introduced to ATP [26], along with a restricted form of 1st order logic: *Clause Normal Form* (CNF). Problems in CNF are presented as a set of *clauses*. A clause is the disjunction of zero or more *literals*, where a literal is an atom (a positive literal) or the negation of an atom (a negative literal). For example, the following set has two clauses, each of which has two literals:

```
{   ~word(X) ∨ statement(definition_of(X)),
    ~word(X) ∨ defines(definition_of(X),concept_of(X)) }
```

The resolution inference rule takes two clauses as input, and produces a *resolvant* clause as output. *Unification* is used to make some of the atoms in the input clauses the same, by applying a *substitution* of terms for variables in the atoms. Resolution unifies the atoms of some of the positive literals in one input clause with the atoms of some of the negative literals in the other input clause. The resolvant is the disjunction of the literals not involved in the unification. Variables in the resolvant literals, which also appeared in the unified atoms, may have been given values in the unification. For example, the resolvant obtained using the positive literals of:

```
~word(X) ∨ statement(definition_of(X))
```

and both the negative literals of:

```
~statement(definition_of(atp)) ∨ ~statement(Y) ∨ defines(Y,Z)
```

is:

```
~word(atp) ∨ defines(definition_of(atp),Z)
```

There are two other possible resolvants of the above two input clauses, using each of the negative literals separately. The resolution inference rule may be decomposed into two simpler inference rules, *binary resolution* and *factoring*. Binary resolution restricts resolution to use only one literal from each of the input clauses. Factoring unifies two or more literals in a clause, and produces a *factor* consisting of the remaining literals and one copy of the unifed literals. Every resolution inference can be done by factoring and binary resolution. For example, from the resolution example above:

```
~statement(definition_of(atp)) v ~statement(Y) v defines(Y,Z)
```

factors, to produce:

```
~statement(definition_of(atp)) v defines(definition_of(atp),Z)
```

which binary resolves with:

```
~word(X) v statement(definition_of(X))
```

to produce:

```
~word(atp) v defines(definition_of(atp),Z)
```

As well as being the basis for resolution based ATP systems, resolution and factoring are also used in the simplification operations described in Section 4.1. There are many other resolution based inference rules, e.g., hyper-resolution [27], clause linking [12], paramodulation [25].

Although CNF uses only a subset of the FOF language, it is still expressive enough for all ATP problems that can be written in FOF. A *clausifier* negates the FOF conjecture, and then converts the FOF axioms, hypotheses, and negated conjecture to CNF. The ATP system then uses resolution based inference rules to derive new clauses, which are added to the clause set, aiming to eventually produce an empty clause (a clause with zero literals). The empty clause can be inferred only from contradictory parent clauses. The derivation of the empty clause is called a *refutation*. As the operations of the clausifier and ATP system are satisfiability preserving, i.e., they cannot create a contradiction from non-contradictory input, finding a refutation shows that the axioms, hypotheses, and negated conjecture are together contradictory. Providing that the axioms and hypotheses are not contradictory within themselves, this establishes that the negated conjecture is the source of the contradiction, and hence that the unnegated conjecture is a logical consequence of the axioms and hypotheses. Resolution based theorem proving is *refutation complete*, i.e., if a refutation exists, then it can be found using resolution based inference rules.

The conversion from FOF to CNF is not unique. With respect to the subsequent ATP process, some CNF versions of a FOF problem are better than others, in the sense that a refutation can be found using less resources. It is therefore of high interest to find the 'best' CNF version of a FOF problem. The focus of this paper is to examine and compare clausifiers.

The notion of what makes a good CNF version of a FOF problem must be measured with respect to the ATP systems that will take the CNF as input. Thus in order to compare CNF versions of a FOF problem, it is necessary to consider the workings of a (generic) ATP system. An ATP system finds a solution by search. The major choice in the search is which clause(s) to use in the next inference step. Much research in ATP has been focussed on designing refinements of resolution based inference rules, e.g., ordering [11], semantic resolution [30], model elimination [14], that restrict the choice of clauses to use in each inference step without destroying completeness. However, even with such refinements, it is still necessary to decide which of the eligible clauses to use. The choice of clause(s) to use is determined by a heuristic, which measures the quality of the clauses with respect to the ATP system's inference rules. The study of heuristics for ATP is underdeveloped, and at the 1st order level it impossible for the heuristic to ensure that every inference performed is part of a proof of the conjecture (or P = NP). A common, simple, and very effective heuristic is to count the number of variable, functor, and predicate symbol occurrences in the clause. For example, the clause:

```
~word(X) v defines(definition_of(X),concept_of(X))
```

has a symbol count of 7. A lower symbol count indicates a better clause. Symbol count is used, at least in part, in the heuristic functions of several contemporary ATP systems, e.g., Gandalf [37], METEOR [1], Otter [18], SETHEO [13], SPASS [40], Vampire [38], Violet [8]. Otter, SETHEO, and SPASS were the category winners of the CADE-13 ATP system competition [36], thus indicating that symbol count is a reasonable heuristic for measuring the quality of clauses, for contemporary state-of-the-art ATP systems[1].

As heuristics measure the quality of clauses, they can also be used for comparing sets of clauses, by combining all the clause's heuristic values in an appropriate fashion. For the symbol count heuristic, summing the individual clause's symbol counts is appropriate. For example, the doubleton clause set in the first example above has a symbol count of 12. This measure provides a basis for comparing CNF versions of a FOF problem, and hence for comparing clausifiers by their output. The remaining sections of this paper discuss and compare clausifiers, using the symbol count heuristic as a basis for comparison.

Following this introduction, Section 2 examines the boundary between clausifiers and ATP systems. Section 3 discusses clausification. Common operations used in clausification are described in Section 3.1, and four well known clausifiers are described in Section 3.2. Section 4 discusses simplification of clause sets. Common operations used in simplification are described in Section 4.1, and a publicly available simplifier is described in Section 4.2. An experimental comparison of clausifiers, using the simplifier, is reported in Section 5. Section 6 presents conclusions.

2. Where Clausifiers and ATP Systems meet

There are two main approaches to building clausifiers. The standard approach (see, e.g., [2, 39]) is based on the application of equivalences to transform the FOF problem to CNF. In the standard approach the structure of the FOF problem is lost, and the use of distributivity laws can in some cases lead to a very large number of clauses for an apparently simple FOF problem. However, this approach is simple and well understood. The definitional approach (see e.g., [6, 22, 7]) introduces definitions of subformulae in an attempt to reduce the number and size of the clauses that are produced. Although this technique shows promise, it is not yet adequately researched and its general utility has not been established. The standard approach is currently dominant, and therefore this paper examines only clausifiers that use the standard approach.

An examination of standard approach clausifiers shows that the transformation process has two distinct parts. One part, *clausification*, transforms the FOF to CNF. The other part, *simplification*, simplifies the CNF obtained from clausification in order to improve it (the notion of improve varies from clausifier to clausifier, but typically conforms to the symbol count heuristic). The two parts are not necessarily done

[1]It is of historical interest to note that symbol counting was one of the early heuristics implemented in ATP systems; the first explicit reference found is in 1976 [21], but both Larry Wos [42] and Bill McCune [19] believe that Ross Overbeek was using symbol count right from about 1969. Despite the efforts of ATP researchers since then, no other general purpose heuristic has proved to be better, as evidenced by the CADE-13 ATP system competition results. The use of semantically based heuristics appears to be one potential avenue for finding improved heuristics [29, 20, 33, 31].

separately in sequence. Some clausifiers interleave operations from the two parts. Necessarily, both clausification and simplification operations are satisfiability preserving.

The processes of clausification, simplification, and solution search, may be put into context by considering the overall ATP process:

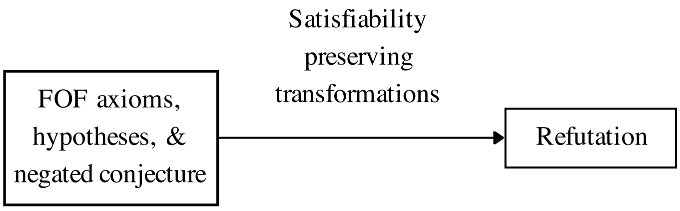

Figure 1: Overall ATP Process

The "Satisfiability preserving transformations" are clausification, simplification, and inferencing. There are two common ways in which the satisfiability preserving transformations are organised:

- A clausifier is used to produce input for an ATP system. The ATP system does no or little simplification of its input, but rather proceeds directly to the solution search. The picture expands to:

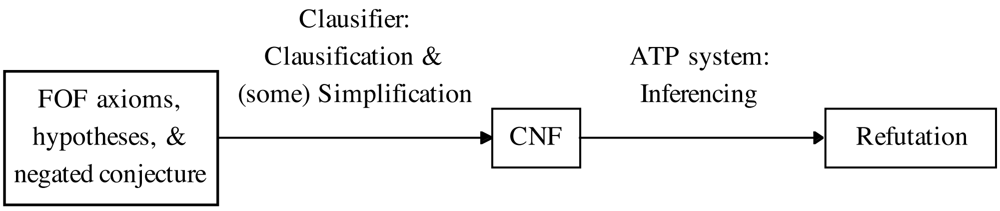

Figure 2: Simplification in Clausifier

  Such clausifiers are often built simply to support the development of an ATP system. The ATP system is the focus of the research, and the clausifier is merely a required tool to support that research. Although such clausifiers produce a CNF version of the FOF problem to be solved, the CNF may not be a simple as possible due to inadequate development of the clausifier. In the context of the research this may significantly affect the evaluation of the ATP system, and hence the perceived quality of the ATP system.

- The CNF problems are part of an existing test suite, such as those provided by the TPTP Problem Library [34, 35]. The problems are used for the evaluation of an ATP system. The manner in which the CNF has been obtained is typically unknown. Recognising that the CNF problems are not simplified to the extent that they could be, the ATP system includes a simplification stage before the solution search is started. In this case the picture expands to:

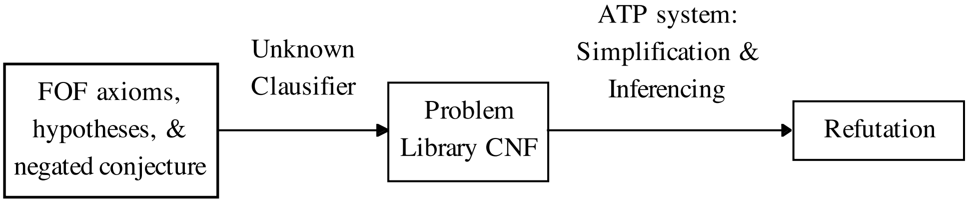

Figure 3: Simplification in ATP System

> The benefit of simplifying the problem before starting the solution search can be large; in the extreme the problem may be solved in the simplification stage.

It is evident that the process of simplification forms an fuzzy boundary between clausification and the search for a solution. The simplification may be associated with either the clausifier or the ATP system, and may be applied to clauses at any stage. The distinction between clausification and simplification, and between simplification and solution search, has not been well defined. As part of a meaningful evaluation of an ATP system, the various components need to be evaluated independently as well as in combination. This precise point motivated the decision to compare only CNF ATP systems in the CADE-13 ATP system competition [36]. It is therefore necessary to (try to) clearly define what constitutes each of the three parts: clausification, simplification, and solution search. In particular, in order to support meaningful evaluation of current ATP systems, it is necessary to distinguish between simplification and solution search. This is necessary because, as mentioned in Section 1, the search process is controlled by a heuristic, and the development of heuristics is a critical but weak area in ATP research. By clearly removing the influence of simplification from the solution search, it is possible to accurately evaluate the effect of a heuristic. It is inadequate for a search controlling heuristic to be evaluated as “good” simply because the preceding simplifier is performing well.

The fundamental distinction between simplification and solution search lies in the word "search". As indicated in Section 1, the search controlling heuristics cannot be perfect in the 1st order case. Thus it is necessarily the case that ATP systems perform inference steps that do not form part of any refutation. Such steps may reduce the quality of the clause set, as measured by the heuristic function. This worsening can be used to clearly differentiate between simplification and solution search. Simplification steps are those that monotonically and non-asymptotically improve the clause set, as measured by the heuristic function, so that the subsequent search for a solution is easier. For example, using the symbol count heuristic, simplification would reduce the total symbol count of the clause set. In contrast, solution search may worsen the clause set without contributing to a solution. For example, resolution steps are part of the solution search, because resolution steps may increase the total symbol count of the clause set increases without making any progress towards a refutation.

The above gives a clear distinction between simplification and solution search, and hence makes it possible to clearly separate clausifiers from ATP systems, as required by this research. The other boundary, between clausification and simplification, is harder to delineate. Depending on what clausification operations are used, and precisely how they are formulated, some simplification happens incidentally within the clausification stage. However, commonly there is an identifiable point within clausifiers at which a clause set has been created, and simplification operations explicitly commence. For the purposes of this research, this point has been identified in each of the clausifiers examined in Section 3.2. The classification operations used by these (and most other) clausifiers are described in Sections 3.1, and simplification operations are described in Section 4.1.

3. Clausification

In standard clausification algorithms the process is broken into distinguishable operations. Each of the operations is satisfiability preserving. Some operations have prerequisites on the form of the input data, and those prerequisites are met by ordering the operations appropriately, so that the output from each operation meets the prerequisites of the next. Following is a survey of commonly used operations. An examination of these shows that there are many acceptable orderings of the operations, thus giving rise to different clausification algorithms. Four algorithms are given in Section 3.2. Although these operations have been used in clausifiers before, this separation and documentation of the operations provides explicit building blocks with which new clausifiers can easily be constructed.

3.1 Clausification Operations

Removing implications and equivalences

This operation removes implications and equivalences, replacing them with conjunctions, disjunctions, and negations. Implications are transformed into a disjunction of the negated antecedent and the consequent, e.g., $\mathcal{A} \rightarrow \mathcal{B}$ becomes $\sim\mathcal{A} \vee \mathcal{B}$. If an implication is in the immediate scope of a negation, i.e. $\sim(\mathcal{A} \rightarrow \mathcal{B})$, then this is converted directly to $\mathcal{A} \wedge \sim\mathcal{B}$. Equivalences are transformed into a conjunction of two implications that are then transformed separately. If an equivalence is in the immediate scope of a negation, i.e., $\sim(\mathcal{A} \leftrightarrow \mathcal{B})$, then this is transformed directly to $(\mathcal{A} \vee \mathcal{B}) \wedge (\sim\mathcal{A} \vee \sim\mathcal{B})$.

Moving quantifiers out

In this operation quantifiers ($\exists$ and $\forall$) are moved outwards (textually, leftwards) so that they are outside all connectives. The scope of the quantifiers increases. A negated quantified formula is transformed by swapping the quantifier and negating the quantified formula, e.g., $\sim\forall X\mathcal{A}$ becomes $\exists X\sim\mathcal{A}$. Quantifiers within conjunctions and disjunctions are moved out to apply to the entire conjunction or disjunction, e.g., $(\mathcal{A} \wedge \forall X\mathcal{B})$ becomes $\forall X(\mathcal{A} \wedge \mathcal{B})$. In this transformation variables are renamed as necessary to avoid the incorrect quantification of variables. It is possible to move quantifiers out through implications and equivalences, but typically implications and equivalences are removed first to avoid this. After moving quantifiers out the formula is in Prenex normal form.

Moving negations in

This operation moves negations inwards (to the right), so that they apply to only atoms. Negated quantified formulae are dealt with as when moving quantifiers out. Negated conjunctions and disjunctions are transformed using De Morgan's equivalences, e.g., $\sim(\mathcal{A} \wedge \mathcal{B})$ becomes $\sim\mathcal{A} \vee \sim\mathcal{B}$. Double negations are cancelled. It is possible to move negations in through implications and equivalences, but typically implications and equivalences are removed first to avoid this. After moving negations in the formula is in Literal normal form.

Mini-scoping

Mini-scoping reduces the scope of quantifiers. The motivation is to simplify and reduce the number of Skolem functions created in Skolemization, which is described below. There are four transformations that are used:

- If the quantified variable does not occur in the quantified formula, the quantification is removed.
- If the quantified variable does not occur in an operand of a quantified conjunction or disjunction, the quantification is applied to only the operand that contains the quantified variable.
- A universally quantified conjunction is transformed into a conjunction of two universally quantified operands, e.g., $\forall X(\mathcal{A} \wedge \mathcal{B})$ becomes $\forall X \mathcal{A} \wedge \forall Y \mathcal{B}$. Note that the quantified variable is renamed in one of the conjuncts.
- An existentially quantified disjunction is transformed into a disjunction of two existentially quantified operands, i.e., $\exists X(\mathcal{A} \vee \mathcal{B})$ becomes $\exists X \mathcal{A} \vee \exists Y \mathcal{B}$. Note that the quantified variable is renamed in one of the disjuncts.

It is possible to mini-scope formulae containing implications and equivalences, but typically implications and equivalences are removed first to avoid this.

Skolemization

Skolemization (in the context of clausification) removes existential quantifiers. Skolemization requires, in effect, that all quantifiers are outside (to the left of) all negations[2]. Thus Skolemization is performed after either moving quantifiers out or moving negations in. Each existentially quantified variable in an atom is replaced by a Skolem function; the functor is a newly generated symbol, and the arguments are all the variables that are universally quantified at the position of the existential quantification. The existential quantifiers and their variables are removed. For example, $\forall X(\mathcal{A} \wedge \exists Y \forall Z p(X,Y,Z))$ becomes $\forall X(\mathcal{A} \wedge \forall Z p(X,sk(X),Z))$. After Skolemization the formula is in Skolem normal form.

As Skolemization leaves only universal quantification, the universal quantifiers are typically removed as part of this operation, leaving all variables implicitly universally quantified, e.g., $\forall X(\mathcal{A} \wedge \exists Y \forall Z p(X,Y,Z))$ becomes $\mathcal{A} \wedge p(X,sk(X),Z)$. To avoid variables with the same name being mistakenly considered to be the same variable, variables are uniquely renamed before the universal quantifiers are removed.

Distributing disjunctions

Conjunctions are moved to outside disjunctions, i.e., no disjuncts are conjuncts. Distributing disjunctions requires that negations have been moved in and Skolemization has been done. Conjunctions are moved to outside disjunctions by distributing disjunctions over conjunctions, e.g., $\mathcal{A} \vee (\mathcal{B} \wedge \mathcal{C})$ becomes $(\mathcal{A} \vee \mathcal{B}) \wedge (\mathcal{A} \vee \mathcal{C})$. After distributing disjunctions the formula is in Conjunctive normal form.

---

[2] If there are negations outside quantifiers, then Skolemization has to take the polarity of negation into account at the point of each quantifier. The polarity is negative if an odd number of negations apply and positive otherwise. If the polarity is negative, then the roles of the two types of quantifier are reversed. See [9, 10] for examples of clausifiers that take this approach.

Converting to Clause Normal Form (CNF)

Formulae in conjunctive normal form are split into clauses by removing the conjunctions, to form a set of clauses. For example, $(\mathcal{A} \vee \mathcal{B}) \wedge (\mathcal{A} \vee \mathcal{C})$ becomes $\{ \mathcal{A} \vee \mathcal{B}, \mathcal{A} \vee \mathcal{C} \}$.

3.2 Clausifiers

Many clausifiers have been built in ATP research. The importance of good clausification has been recognised as an important facet of successful ATP, and research into building good clausifiers for specific ATP systems is well documented, e.g., [18, 9]. General purpose clausifiers, which can be expected to be at least adequate for most ATP systems, also exist [2, 35, 40]. Described below are four clausifiers that use the standard approach. Each uses a different combination or ordering of the operations described in Section 3.1. Attention should be paid to both the similarities and differences between the sequences of operations. The effects of the differences are shown in the experimental data, given in Section 5.

Bundy

Alan Bundy's clausifying algorithm, described in [2], has been a starting point for many ATP researchers. The clausifier does the following sequence of operations:

- Remove implications and equivalences
- Move quantifiers out
- Skolemize
- Move negations in
- Distribute disjunctions
- Convert to CNF

The notable feature of this algorithm is the early moving of quantifiers to the outermost level. This gives greater scope to the quantifiers, and may therefore increase the arity of the Skolem functions created.

Quaife

Art Quaife developed an effective clausifier as part of his investigation into the use of ATP in set theory and formal mathematical theories [23, 24]. The clausifier does the following steps:

- Remove implications and equivalences, and Move negations in. Note that these two operations are done together.
- Mini-scope
- Skolemize
- Distribute disjunctions
- Convert to CNF

This algorithm tries to limit the number of symbols introduced in Skolemization through mini-scoping. By moving the negations in it is possible to then reduce the scope of quantifiers as much as possible, hopefully reducing the arity and number of Skolem functions.

Otter

Bill McCune's Otter ATP system [18] is undoubtedly the most widely known ATP system. Otter has been used for many years by many ATP researchers, and there are many documented successes, e.g., [16, 17, 9]. A clausifier is built into Otter, which does the following steps:

- Remove implications and equivalences
- Move negations in
- Skolemize
- Distribute disjunctions
- Convert to CNF

This clausifier differs from Quaife's only in that it omits mini-scoping.

TPTP

The TPTP Problem Library includes a utility, `tptp2X`, for manipulating and formatting FOF and CNF formulae [35]. The utility contains a clausifier, which does the following steps:

- Remove implications and equivalences
- Mini-scope
- Move negations in
- Skolemize
- Distribute disjunctions
- Convert to CNF

The TPTP clausifier reduces the scope of quantifiers as early as possible, in a (as it turns out) unsuccessful attempt to reduce the size and number of Skolem functions.

4. Simplification

The operations of simplification improve the quality of the clause set produced by clausification. The operations that are typically inter-dependant, in the sense that the application of one may enable the application of another (or itself again). Some simplification operations are described below. Some of the operations are well known, but others are less common. As in the case of the clausification operations, this documentation of the operations provides explicit building blocks with which new simpifiers can easily be constructed.

4.1 Simplification Operations

Merge identical literals

Repeated literals in a clause are removed, e.g., $\mathcal{A} \vee \mathcal{B} \vee \mathcal{A} \vee \mathcal{C} \vee \mathcal{A}$ becomes $\mathcal{A} \vee \mathcal{B} \vee \mathcal{C}$. Note that the literals must be identical. This operation is generalised in the repeatable Factor simplify operation, described below.

Remove tautologies

Clauses which are tautologous (i.e., necessarily true, regardless of the truth of the atoms in the formula, e.g., $\mathcal{A} \vee \sim\mathcal{A}$) are removed. This is possible because ATP systems for CNF problems search for a contradiction in the clause set, and it is impossible for a tautology to contribute to a contradiction.

Eliminate pure literals

Pure literals are literals in clauses that cannot be resolved against any literal in any clause. Clauses that contain a pure literal are removed, because such a clause cannot contribute to a contradiction.

Remove subsumed clauses

A clause $\mathcal{C}$ subsumes a clause $\mathcal{D}$ if there is a substitution of terms for some of the variables of $\mathcal{C}$ such that the literals of $\mathcal{C}$ (after the substitution) are a subset of the literals in $\mathcal{D}$. For example:

```
~statement(definition_of(atp)) v ~statement(Y) v defines(Y,Z)
```

subsumes:

```
~statement(definition_of(atp)) v defines(definition_of(atp),Z)
```

with `Y` substituted by `definition_of(atp)`.

Any clause that is subsumed by another in the clause set, is removed.

Factor simplify

If a factor of any clause subsumes the clause, then the factor replaces the clause.

Isolated resolution

An *isolated resolution* is a binary resolution in which the literals resolved against cannot resolve against any other literals in the clause set. If an isolated resolution is possible then the resolvant replaces the two parent clauses.

Subsuming unit resolution

A *subsuming unit resolution* is a resolution between a unit clause and second clause, where the atom of the unit clause subsumes the atom of the literal resolved upon in the second clause (analogous to clausal subsumption, an atom $\mathcal{P}$ subsumes an atom Q if there is a substitution of terms for some of the variables of $\mathcal{P}$ such $\mathcal{P}$ (after the substitution) is the same as Q). The resolvant necessarily subsumes the second clause, which is then removed. Such resolution steps are desirable in ATP systems [32, 1] because they always improve the clause set (according to the symbol count heuristic). In simplification, if a subsuming unit resolution is possible, the resolvant replaces the second parent clause.

4.2 The `tptp2X` Simplifier

As well as a classifier, the `tptp2X` utility contains a simplifier that can be used in conjunction with any other of its transformations. The simplifier first applies two simplifications once only:

- Merge identical literals
- Remove tautologies

and then applies the following sequence of repeatable simplifications repeatedly, until no element of the sequence makes any simplification:

- Factor simplify
- Subsuming unit resolution
- Remove subsumed clauses
- Eliminate pure literals

The first two simplifications are applied once only because they do not interact with any of those that are applied repeatedly. If isolated resolution were added to the second list, then the first two simplifications would also have to be applied repeatedly. Isolated resolution has not been used because it is not guaranteed to improve the clause set, as measured by the symbol count heuristic.

5. Performance Data

To test the clausifiers described in Section 3.2, their algorithms have been encoded in the framework of the `tptp2X` utility. Each clausifier was given 145 FOF problems from the SYN domain of the TPTP Problem Library (the FOF part of the TPTP is currently in α-release) to convert to CNF. Due to memory constraints, two of the 145 problems could not be converted by one of the clausifiers, so those two problems have been omitted from the results. The symbol counts of each of the remaining 143 CNF versions have been recorded in Table 1 for each clausifier (B=Bundy, Q=Quaife, O=Otter, T=TPTP). Each CNF version was then passed through the `tptp2X` simplifier to measure the extent to which the clauses produced by the clausifiers are amenable to simplification. The symbol counts after simplification are also recorded in Table 1 (indicated by +S). The righthand half of Table 1 gives some comparative analysis of the data. The columns marked "Best" and "Best+S" give the lowest symbol count before and after simplification, respectively. The columns marked "Best/?" and "Best+S/?+S" give the ratio between the lowest symbol count and that achieved by each of the clausifiers, before and after simplification. Thus in these eight columns a value of 1.00 is the best achievable, and lower values are worse. In the cases where the algorithm achieves a symbol count of 0, the lowest symbol count is necessarily 0, and a value of 1.00 is given as the ratio, indicating that the algorithm also achieves the best possible symbol count. The average is given at the bottom of these eight columns.

	Symbol counts								Analysis before simplification					Analysis after simplification				
Problem	B	B+S	Q	Q+S	O	O+S	T	T+S	Best	Best /B	Best /Q	Best /O	Best /T	Best +S	Best +S / B+S	Best +S / Q+S	Best +S / O+S	Best +S / T+S
SYN040-1	24	0	24	0	24	0	24	0	24	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN041-1	8	0	8	0	8	0	8	0	8	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN001-1	8	0	8	0	8	0	8	0	8	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN045-1	88	0	88	0	88	0	88	0	88	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN044-1	26	26	26	26	26	26	26	26	26	1.00	1.00	1.00	1.00	26	1.00	1.00	1.00	1.00
SYN046-1	24	0	24	0	24	0	24	0	24	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN047-1	228	0	228	0	228	0	228	0	228	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN048-1	7	0	6	0	7	0	6	0	6	0.86	1.00	0.86	1.00	0	1.00	1.00	1.00	1.00
SYN049-1	14	0	12	0	14	0	12	0	12	0.86	1.00	0.86	1.00	0	1.00	1.00	1.00	1.00
SYN050-1	29	0	27	0	29	0	27	0	27	0.93	1.00	0.93	1.00	0	1.00	1.00	1.00	1.00
SYN051-1	30	20	30	20	30	20	30	20	30	1.00	1.00	1.00	1.00	20	1.00	1.00	1.00	1.00
SYN052-1	30	20	30	26	30	26	30	26	30	1.00	1.00	1.00	1.00	20	1.00	0.77	0.77	0.77
SYN053-1 †	42	20	30	0	30	0	30	0	30	0.71	1.00	1.00	1.00	0	0.00	1.00	1.00	1.00
SYN054-1	39	39	39	39	39	39	39	39	39	1.00	1.00	1.00	1.00	39	1.00	1.00	1.00	1.00
SYN055-1	53	37	51	36	51	36	51	36	51	0.96	1.00	1.00	1.00	36	0.97	1.00	1.00	1.00
SYN056-1	98	98	72	72	72	72	72	72	72	0.73	1.00	1.00	1.00	72	0.73	1.00	1.00	1.00
SYN057-1	39	39	39	39	39	39	39	39	39	1.00	1.00	1.00	1.00	39	1.00	1.00	1.00	1.00
SYN058-1	48	30	48	30	48	30	48	30	48	1.00	1.00	1.00	1.00	30	1.00	1.00	1.00	1.00
SYN059-1	426	420	186	144	186	144	186	144	186	0.44	1.00	1.00	1.00	144	0.34	1.00	1.00	1.00
SYN060-1	39	39	39	39	39	39	39	39	39	1.00	1.00	1.00	1.00	39	1.00	1.00	1.00	1.00
SYN061-1	30	24	30	24	30	24	30	24	30	1.00	1.00	1.00	1.00	24	1.00	1.00	1.00	1.00
SYN062-1	42	33	42	33	42	33	42	33	42	1.00	1.00	1.00	1.00	33	1.00	1.00	1.00	1.00
SYN063-1 †	386	20	342	0	342	0	342	0	342	0.89	1.00	1.00	1.00	0	0.00	1.00	1.00	1.00
SYN064-1	12	0	8	0	12	0	8	0	8	0.67	1.00	0.67	1.00	0	1.00	1.00	1.00	1.00
SYN065-1	62	62	62	62	62	62	62	62	62	1.00	1.00	1.00	1.00	62	1.00	1.00	1.00	1.00
SYN066-1	61	61	57	57	61	61	57	57	57	0.93	1.00	0.93	1.00	57	0.93	1.00	0.93	1.00
SYN068-1	46	38	46	38	46	38	46	38	46	1.00	1.00	1.00	1.00	38	1.00	1.00	1.00	1.00
SYN069-1	150	150	141	91	141	91	141	91	141	0.94	1.00	1.00	1.00	91	0.61	1.00	1.00	1.00
SYN070-1	82	58	82	58	82	58	82	58	82	1.00	1.00	1.00	1.00	58	1.00	1.00	1.00	1.00
SYN071-1	48	48	48	48	48	48	48	48	48	1.00	1.00	1.00	1.00	48	1.00	1.00	1.00	1.00
SYN072-1	55	55	55	55	55	55	55	55	55	1.00	1.00	1.00	1.00	55	1.00	1.00	1.00	1.00
SYN073-1	15	15	13	13	13	13	13	13	13	0.87	1.00	1.00	1.00	13	0.87	1.00	1.00	1.00
SYN074-1	446	156	372	372	372	234	372	372	372	0.83	1.00	1.00	1.00	156	1.00	0.42	0.67	0.42

SYN075-1	446	156	372	372	372	234	372	372	372	0.83	1.00	1.00	1.00	156	1.00	0.42	0.67	0.42
SYN077-1	166	166	138	138	138	138	138	138	138	0.83	1.00	1.00	1.00	138	0.83	1.00	1.00	1.00
SYN078-1	165	165	105	105	105	105	105	105	105	0.64	1.00	1.00	1.00	105	0.64	1.00	1.00	1.00
SYN079-1	36	36	36	36	36	36	36	36	36	1.00	1.00	1.00	1.00	36	1.00	1.00	1.00	1.00
SYN080-1	58	58	58	58	58	58	58	58	58	1.00	1.00	1.00	1.00	58	1.00	1.00	1.00	1.00
SYN081-1	21	21	21	21	21	21	21	21	21	1.00	1.00	1.00	1.00	21	1.00	1.00	1.00	1.00
SYN082-1	135	35	126	126	126	116	126	126	126	0.93	1.00	1.00	1.00	35	1.00	0.28	0.30	0.28
SYN083-1	76	76	76	76	76	76	76	76	76	1.00	1.00	1.00	1.00	76	1.00	1.00	1.00	1.00
SYN084-1	592	172	434	133	434	65	434	133	434	0.73	1.00	1.00	1.00	65	0.38	0.49	1.00	0.49
SYN315-1	34	22	30	26	34	22	30	26	30	0.88	1.00	0.88	1.00	22	1.00	0.85	1.00	0.85
SYN316-1	28	14	28	22	28	14	28	22	28	1.00	1.00	1.00	1.00	14	1.00	0.64	1.00	0.64
SYN317-1	36	0	36	0	36	0	36	0	36	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN318-1	23	0	17	0	18	0	17	0	17	0.74	1.00	0.94	1.00	0	1.00	1.00	1.00	1.00
SYN319-1	77	49	57	48	77	49	57	48	57	0.74	1.00	0.74	1.00	48	0.98	1.00	0.98	1.00
SYN320-1	22	0	19	0	22	0	19	0	19	0.86	1.00	0.86	1.00	0	1.00	1.00	1.00	1.00
SYN321-1	48	32	52	44	52	44	52	44	48	1.00	0.92	0.92	0.92	32	1.00	0.73	0.73	0.73
SYN322-1	16	0	16	0	16	0	16	0	16	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN323-1	32	32	32	32	32	32	32	32	32	1.00	1.00	1.00	1.00	32	1.00	1.00	1.00	1.00
SYN324-1	62	40	62	40	62	40	62	40	62	1.00	1.00	1.00	1.00	40	1.00	1.00	1.00	1.00
SYN325-1	40	20	40	20	40	20	40	20	40	1.00	1.00	1.00	1.00	20	1.00	1.00	1.00	1.00
SYN326-1	52	28	52	28	52	28	52	28	52	1.00	1.00	1.00	1.00	28	1.00	1.00	1.00	1.00
SYN327-1	67	41	67	51	67	41	67	51	67	1.00	1.00	1.00	1.00	41	1.00	0.80	1.00	0.80
SYN328-1	102	102	99	99	102	102	99	99	99	0.97	1.00	0.97	1.00	99	0.97	1.00	0.97	1.00
SYN329-1	20	0	20	0	20	0	20	0	20	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN330-1	108	80	108	80	108	80	108	80	108	1.00	1.00	1.00	1.00	80	1.00	1.00	1.00	1.00
SYN331-1	60	60	60	60	60	60	60	60	60	1.00	1.00	1.00	1.00	60	1.00	1.00	1.00	1.00
SYN332-1	382	178	382	178	382	178	382	178	382	1.00	1.00	1.00	1.00	178	1.00	1.00	1.00	1.00
SYN333-1	68	0	56	0	68	0	56	0	56	0.82	1.00	0.82	1.00	0	1.00	1.00	1.00	1.00
SYN334-1	124	100	124	100	124	100	124	100	124	1.00	1.00	1.00	1.00	100	1.00	1.00	1.00	1.00
SYN335-1	168	148	168	148	168	148	168	148	168	1.00	1.00	1.00	1.00	148	1.00	1.00	1.00	1.00
SYN336-1	28	10	28	10	28	10	28	10	28	1.00	1.00	1.00	1.00	10	1.00	1.00	1.00	1.00
SYN337-1	23	0	23	0	23	0	23	0	23	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN338-1	15	0	15	0	15	0	15	0	15	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN339-1	12	12	12	12	12	12	12	12	12	1.00	1.00	1.00	1.00	12	1.00	1.00	1.00	1.00
SYN340-1	16	16	16	16	16	16	16	16	16	1.00	1.00	1.00	1.00	16	1.00	1.00	1.00	1.00
SYN341-1	14	14	14	14	14	14	14	14	14	1.00	1.00	1.00	1.00	14	1.00	1.00	1.00	1.00
SYN342-1	11	0	11	0	11	0	11	0	11	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN343-1	27	0	25	0	27	0	25	0	25	0.93	1.00	0.93	1.00	0	1.00	1.00	1.00	1.00
SYN344-1	48	28	48	28	48	28	48	28	48	1.00	1.00	1.00	1.00	28	1.00	1.00	1.00	1.00
SYN345-1	133	81	133	81	133	81	133	81	133	1.00	1.00	1.00	1.00	81	1.00	1.00	1.00	1.00
SYN346-1	36	36	36	36	36	36	36	36	36	1.00	1.00	1.00	1.00	36	1.00	1.00	1.00	1.00
SYN347-1	344	0	344	0	344	0	344	0	344	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN349-1	4099	226	4099	226	4099	226	4099	226	4099	1.00	1.00	1.00	1.00	226	1.00	1.00	1.00	1.00
SYN350-1	174	86	174	86	174	86	174	86	174	1.00	1.00	1.00	1.00	86	1.00	1.00	1.00	1.00
SYN351-1	234	44	234	44	234	44	234	44	234	1.00	1.00	1.00	1.00	44	1.00	1.00	1.00	1.00
SYN352-1	176	104	176	104	176	104	176	104	176	1.00	1.00	1.00	1.00	104	1.00	1.00	1.00	1.00
SYN353-1	398	310	398	310	398	310	398	310	398	1.00	1.00	1.00	1.00	310	1.00	1.00	1.00	1.00
SYN354-1	124	80	124	80	124	80	124	80	124	1.00	1.00	1.00	1.00	80	1.00	1.00	1.00	1.00
SYN355-1 *	30	0	18	18	18	18	18	18	18	0.60	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN356-1	36	36	36	36	36	36	36	36	36	1.00	1.00	1.00	1.00	36	1.00	1.00	1.00	1.00
SYN357-1	6	0	6	0	6	0	6	0	6	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN358-1	30	0	30	0	30	0	30	0	30	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN359-1 *	28	0	26	26	26	26	26	26	26	0.93	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN360-1 *	44	0	17	17	17	17	17	17	17	0.39	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN361-1 *	36	0	23	17	23	17	23	17	23	0.64	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN362-1	16	10	15	9	15	9	15	9	15	0.94	1.00	1.00	1.00	9	0.90	1.00	1.00	1.00
SYN363-1	20	20	17	17	17	17	17	17	17	0.85	1.00	1.00	1.00	17	0.85	1.00	1.00	1.00
SYN364-1 *	75	0	43	43	43	43	43	43	43	0.57	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN365-1 *	49	0	41	41	41	41	41	41	41	0.84	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN366-1 *	56	56	40	40	40	40	40	40	40	0.71	1.00	1.00	1.00	40	0.71	1.00	1.00	1.00
SYN367-1 *	32	0	26	22	26	22	26	22	26	0.81	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN368-1	7	0	6	0	7	0	6	0	6	0.86	1.00	0.86	1.00	0	1.00	1.00	1.00	1.00
SYN369-1	16	16	13	13	13	13	13	13	13	0.81	1.00	1.00	1.00	13	0.81	1.00	1.00	1.00
SYN370-1	22	14	22	14	22	14	22	14	22	1.00	1.00	1.00	1.00	14	1.00	1.00	1.00	1.00
SYN371-1	22	0	16	0	18	0	16	0	16	0.73	1.00	0.89	1.00	0	1.00	1.00	1.00	1.00
SYN372-1	12	0	12	0	12	0	12	0	12	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN373-1	36	0	36	0	36	0	36	0	36	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN374-1 *	240	0	256	74	256	52	256	74	240	1.00	0.94	0.94	0.94	0	1.00	0.00	0.00	0.00
SYN375-1	800	80	240	21	242	21	240	21	240	0.30	1.00	0.99	1.00	21	0.26	1.00	1.00	1.00
SYN376-1 *	22	0	18	18	18	18	18	18	18	0.82	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN377-1 *	240	0	240	21	242	21	240	21	240	1.00	1.00	0.99	1.00	0	1.00	0.00	0.00	0.00
SYN378-1	14	0	12	0	12	0	12	0	12	0.86	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN379-1	16	0	16	0	16	0	16	0	16	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN380-1	26	26	24	24	24	24	24	24	24	0.92	1.00	1.00	1.00	24	0.92	1.00	1.00	1.00
SYN381-1	55	23	27	27	27	27	27	27	27	0.49	1.00	1.00	1.00	23	1.00	0.85	0.85	0.85
SYN382-1 *	22	0	17	17	18	18	17	17	17	0.77	1.00	0.94	1.00	0	1.00	0.00	0.00	0.00
SYN383-1	14	0	12	0	14	0	14	0	12	0.86	1.00	0.86	0.86	0	1.00	1.00	1.00	1.00
SYN384-1	12	0	11	0	12	0	11	0	11	0.92	1.00	0.92	1.00	0	1.00	1.00	1.00	1.00
SYN385-1	10	0	9	0	10	0	9	0	9	0.90	1.00	0.90	1.00	0	1.00	1.00	1.00	1.00
SYN386-1 *	159	0	56	56	56	56	56	56	56	0.35	1.00	1.00	1.00	0	1.00	0.00	0.00	0.00
SYN387-1	4	0	4	0	4	0	4	0	4	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN388-1	4	0	4	0	4	0	4	0	4	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN389-1	10	0	10	0	10	0	10	0	10	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN390-1	8	0	8	0	8	0	8	0	8	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN391-1	16	16	16	16	16	16	16	16	16	1.00	1.00	1.00	1.00	16	1.00	1.00	1.00	1.00
SYN392-1	160	16	160	16	160	16	160	16	160	1.00	1.00	1.00	1.00	16	1.00	1.00	1.00	1.00
SYN393-1	3552	48	3552	48	3552	48	3552	48	3552	1.00	1.00	1.00	1.00	48	1.00	1.00	1.00	1.00
SYN394-1	13	0	12	0	12	0	12	0	12	0.92	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN395-1	14	0	12	0	12	0	12	0	12	0.86	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN396-1	12	0	12	0	12	0	12	0	12	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00

SYN397-1	20	0	12	0	12	0	12	0	12	0.60	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN398-1	42	0	30	0	30	0	30	0	30	0.71	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN399-1	38	30	30	26	30	26	30	26	30	0.79	1.00	1.00	1.00	26	0.87	1.00	1.00	1.00
SYN400-1	8	0	8	0	8	0	8	0	8	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN401-1	6	0	6	0	6	0	6	0	6	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN402-1	6	0	6	0	6	0	6	0	6	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN403-1	18	0	18	0	18	0	18	0	18	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN404-1	6	0	6	0	6	0	6	0	6	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN405-1	13	0	12	0	12	0	12	0	12	0.92	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN406-1	20	20	18	18	18	18	18	18	18	0.90	1.00	1.00	1.00	18	0.90	1.00	1.00	1.00
SYN407-1	23	23	21	21	21	21	21	21	21	0.91	1.00	1.00	1.00	21	0.91	1.00	1.00	1.00
SYN408-1	11	0	9	0	9	0	9	0	9	0.82	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN409-1	30	0	21	0	21	0	21	0	21	0.70	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN410-1	8	0	8	0	8	0	8	0	8	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN411-1	56	0	20	0	20	0	20	0	20	0.36	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN412-1	16	0	16	0	16	0	16	0	16	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
SYN413-1	37	37	35	35	35	35	35	35	35	0.95	1.00	1.00	1.00	35	0.95	1.00	1.00	1.00
SYN414-1	358	358	172	72	172	72	172	72	172	0.48	1.00	1.00	1.00	72	0.20	1.00	1.00	1.00
SYN415-1	219	219	163	145	163	145	163	145	163	0.74	1.00	1.00	1.00	145	0.66	1.00	1.00	1.00
SYN416-1	8	0	8	0	8	0	8	0	8	1.00	1.00	1.00	1.00	0	1.00	1.00	1.00	1.00
										.902	.999	.983	.998		.945	.890	.901	.890

Table 1: Symbol counts and analysis for CNF versions produced by the clausifiers

The range of average ratios indicates that there are differences between the four clausifiers, both before and after simplification. To establish this statistically, t-tests for difference were performed between the before and after simplification results, for all pairs of clausifiers. The results are shown in Table 2.

	Before simplification		After simplification	
Pair	t-value	Direction of difference	t-value	Direction of difference
Bundy/Quaife	2.920	Positive (Quaife better)	0.103	Positive (Quaife better)
Bundy/Otter	2.810	Positive (Otter better)	0.950	Positive (Otter better)
Bundy/TPTP	2.910	Positive (TPTP better)	0.103	Positive (TPTP better)
Quaife/Otter	2.820	Negative (Quaife better)	1.882	Positive (Otter better)
Quaife/TPTP	1.000	Negative (Quaife better)	0.000	No differences
Otter/TPTP	2.740	Positive (TPTP better)	1.882	Negative (Otter better)

Table 2: t-tests for difference between the clausifiers

In all the comparisons before simplification, except the Quaife/TPTP comparison, one clausifier is shown to be better than the other at the 99% confidence level (the required t-value for these one-tailed tests is 2.35). In the Quaife/TPTP comparison the magnitude of difference is not enough for statistical significance, even at only 95% confidence (requiring a t-value of 1.65). In the comparisons after simplification, the Quaife/Otter and the Otter/TPTP comparisons are significant, and it can be asserted with 95% confidence that Otter is better than Quaife and TPTP. While Otter also yields better results than Bundy, the magnitude of these (t-value 0.95) is not enough for statistical significance.

Both the average ratios and the t-tests show that before simplification the Quaife and TPTP clausifiers are superior to the Bundy and Otter clausifiers. This was expected, and is probably due to the positive effect of mini-scoping in the Quaife and TPTP clausifiers, and the negative effect of the early moving of quantifiers to the outermost level in the Bundy clausifier. After simplification, both the ratio and t-test figures show that the Otter clausifier performs better than the others. The ratio figures suggest that the Bundy clausifier also performs better than the Quaife and TPTP clausifiers after simplification, although this is not supported by the t-test figures (this non-correspondence is discussed below). This change of ranking from before to after simplification shows that amenability of the clauses to simplification is an important factor in the quality of a clausifier. The common difference between the Bundy and Otter clausifiers on one hand, and the Quaife and TPTP clausifiers on the other, is the use of mini-scoping by

Quaife and TPTP. This suggests that the use of mini-scoping makes the resultant clause set less amenable to simplification.

Comparing the Bundy and Otter clausifiers, before simplification the Otter clausifier performs better. After simplification the situation is less clear: the ratio figures indicate that the Bundy clausifier performs better while the t-tests indicate that the Otter clausifier performs better. This apparent paradox is caused by the simplifier reducing twelve of the clause sets (marked with an *) from the Bundy classification, to the empty set. The corresponding clause sets for the other three classifications are not simplified that far. The reverse phenomenon occurs in only two problems (marked with an †), where simplifier reduces all except the Bundy clause sets to the empty set. The implicit normalisation within the ratio data allows these fourteen results to have a very strong effect, while in the t-tests this is not the case. It is thus not possible to make a meaningful quantitative comparison between the Bundy and Otter clausifiers, after simplification. Despite the weaker performance of Bundy's clausification, the clauses produced appear to be more likely to be simplifiable to the empty set than those produced by the other three clausifiers. The resultant benefit of not having to invoke the ATP system for some problems may be desirable for some users. The Otter clausifier, on the other hand, may offer better performance over a large set of problems.

Comparing the Quaife and TPTP clausifiers, before simplification there is a very small difference between their performances. After simplification the two produce identical clause sets. The difference between the two clausifiers is the order in which moving negations in and mini-scoping are done. This swap evidently has no appreciable affect.

6. Conclusion

This research has examined and compared four known clausifiers. Common clausification and simplification operations have been documented, thus providing a starting point for the construction of clausifiers. The boundaries between clausification, simplification, and solution search have been delineated, thus contributing to the meaningful evaluation of clausifiers and ATP systems. This is important, as the evaluation of ATP systems is often inadequate (see [35, 36] for a full discussion). The comparison shows that steps taken to reduce the symbol count during clausification are effective, and that amenability to simplification can compensate for poorer raw clausification performance. This is a noteworthy result, as it suggests that the construction of clausifiers must consider how the clausification and simplification parts interact. The data also forms a baseline for future evaluation of clausifiers.

7. References

1. O.L. Astrachan. *METEOR: Exploring Model Elimination Theorem Proving*. Technical Report CS-1992-22, Department of Computer Science, Duke University, Durham, USA, 1992.
2. A. Bundy. *The Computer Modelling of Mathematical Reasoning*. Academic Press, London, England, 1983.
3. A. Church. *Introduction to Mathematical Logic I*. Princeton University Press, Princeton, USA, 1956.
4. S. Cook. 'The Complexity of Theorem Proving Procedures'. *Conference Record of the Third Annual ACM Symposium on Theory of Computing*, pp. 151-158, Shaker Heights, USA, (1971), ACM Press.

5. H.D. Ebbinghaus, J. and Flum, W. Thomas. *Mathematical Logic*. Springer-Verlag, 1978.
6. E. Eder. 'An Implementation of a Theorem Prover Based on the Connection Method'. In W. Bibel, B. Petkoff (Eds), *AIMSA 84, Artificial Intelligence - Methodology, Systems, Applications* (Varna, Bulgaria), 1984.
7. U. Egly, and T. Rath, 'On the Practical Value of Different Definitional Translations to Normal Form'. M. McRobbie, and J. Slaney (Eds), *Proceedings of the 13th International Conference on Automated Deduction*, pp. 403-417, New Brunswick, USA, (1996), Springer-Verlag (Lecture Notes in Artificial Intelligence 1104), New York, USA.
8. S. Greenbaum. *Email to G. Sutcliffe*, 1996.
9. X. Huang, M. Kerber, M. Kohlhase, E. Melis, D. Nesmith, J. Richts, and J. Siekmann. 'KEIM: A Toolkit for Automated Deduction'. A. Bundy (Ed), *Proceedings of the 12th International Conference on Automated Deduction*, pp.807-810, Nancy, France, (1994), Springer-Verlag (Lecture Notes in Artificial Intelligence 814), New York, USA.
10. M. Kaufmann. *DEFN-SK: An Extension of the Boyer-Moore Theorem Prover to handle First-Order Logic Quantifiers*. Technical Report 43, Computation Logic Inc., Austin, Texas, 1989.
11. R.A. Kowalski and P.J. Hayes. 'Semantic Trees in Automatic Theorem Proving'. *Machine Intelligence* 4:87-101, (1969).
12. S-J. Lee, and D.A. Plaisted. 'Eliminating Duplication with the Hyper-Linking Strategy'. *Journal of Automated Reasoning* 9(1):25-42, (1992).
13. R. Letz, and J. Schumann, and S. Bayerl, and W. Bibel. 'SETHEO: A High-Performance Theorem Prover'. *Journal of Automated Reasoning* 8(2):183-212, (1992).
14. D.W. Loveland. 'Mechanical Theorem Proving by Model Elimination'. *Journal of the ACM* 15(2):236-251, (1968).
15. D.W. Loveland. *Automated Theorem Proving : a logical basis*. Elsevier Science, Amsterdam, The Netherlands, 1978.
16. W.W. McCune. 'Automated Discovery of New Axiomatisations of the Left Group and Right Group Calculi'. *Journal of Automated Reasoning* 9(1):1-24, (1992).
17. W.W. McCune. 'Single Axioms for Groups and Abelian Groups with Various Operations'. *Journal of Automated Reasoning* 10(1):1-13, (1993).
18. W.W. McCune. *Otter 3.0 Reference Manual and Guide*. ANL-94/6, Argonne National Laboratory, Argonne, USA, 1994.
19. W.W. McCune. *Email to G. Sutcliffe*, 1996.
20. X. Nie and D.A. Plaisted. 'A Complete Semantic Back Chaining Proof System'. M. Stickel (Ed) *Proceedings of the 10th International Conference on Automated Deduction*, pp.16-27, Kaiserslautern, Germany, (1990), Springer-Verlag (Lecture Note in Artificial Intelligence 449), New York, USA.
21. R. Overbeek, J. and McCharen, and L. Wos. 'Complexity and Related Enhancements for Automated Theorem-Proving Programs'. *Computers and Mathematics with Applications* 2:1-16, (1976).
22. D.A. Plaisted and S. Greenbaum. 'A Structure-Preserving Clause Form Translation', *Journal of Symbolic Computation* 2:293-304, (1986).

23. A. Quaife. 'Automated Deduction in von Neumann-Bernays-Godel Set Theory'. *Journal of Automated Reasoning* 8(1):91-147, (1992).
24. A. Quaife. *Automated Development of Fundamental Mathematical Theories*. Kluwer Academic Publishers, 1992.
25. G.A. Robinson, and L. Wos. 'Paramodulation and Theorem Proving in First-Order Theories with Equality'. *Machine Intelligence* 4:135-150, (1969).
26. J.A. Robinson. 'A Machine-Oriented Logic Based on the Resolution Principle'. *Journal of the ACM* 12(1):23-41, (1965).
27. J.A. Robinson. 'Automatic Deduction with Hyper-resolution'. *International Journal of Computer Mathematics* 1: 227-234 , (1965).
28. M.K. Rennie and R.A. Girle. *Logic: Theory and Practice*. University of Queensland Press, Brisbane, Australia, 1973.
29. D.M. Sandford. *Using Sophisticated Models in Resolution Theorem Proving* (Lecture Note in Computer Science 90). Springer-Verlag, New York, USA, 1980.
30. J.R. Slagle. 'Automatic Theorem Proving with Renamable and Semantic Resolution'. *Journal of the ACM* 14(4):687-697, (1967).
31. J. Slaney. 'SCOTT: A Model-Guided Theorem Prover'. R. Bajcsy (Ed), *Proceedings of the 13th International Conference on Artificial Intelligence*, pp.109-114, Chambery, France, (1993), Morgan-Kaufman, San Mateo, USA.
32. M.E. Stickel. *A Prolog Technology Theorem Prover: A New Exposition and Implementation in Prolog*. Technical Note 464, SRI International, Menlo Park, USA, 1989.
33. G. Sutcliffe. *A Linear Deduction System with Integrated Semantic Guidance*. PhD Thesis, Department of Computer Science, The University of Western Australia, Perth, Australia, 1992.
34. G. Sutcliffe, C.B. Suttner and T. Yemenis. 'The TPTP Problem Library'. A. Bundy (Ed), *Proceedings of the 12th International Conference on Automated Deduction*, pp.252-266, Nancy, France, (1994), Springer-Verlag (Lecture Notes in Artificial Intelligence 814), New York, USA.
35. C.B. Suttner and G. Sutcliffe. *The TPTP Problem Library (TPTP v1.2.0)*. Technical Report AR-95-03, Institut für Informatik, Technische Universität München, Munich, Germany; Technical Report 95/6, Department of Computer Science, James Cook University, Townsville, Australia, 1995.
36. C.B. Suttner and G. Sutcliffe. 'The Design of the CADE-13 ATP System Competition'. M. McRobbie, and J. Slaney (Eds), *Proceedings of the 13th International Conference on Automated Deduction*, pp.146-160, New Brunswick, USA, (1996), Springer-Verlag (Lecture Notes in Artificial Intelligence 1104), New York, USA.
37. T. Tammet. 'A Resolution Theorem Prover for Intuitionistic Logic'. M. McRobbie, and J. Slaney (Eds), *Proceedings of the 13th International Conference on Automated Deduction*, pp.2-16, New Brunswick, USA, (1996), Springer-Verlag (Lecture Notes in Artificial Intelligence 1104), New York, USA.
38. A. Voronkov. 'The Anatomy of Vampire'. *Journal of Automated Reasoning* 15(2): 237-265, (1995).
39. P.H. Winston. *Artificial Intelligence*. Addison-Wesley, Reading, USA, 1984.
40. C. Weidenbach, B. Gaede, and G. Rock. 'SPASS & FLOTTER'. M. McRobbie, and J. Slaney, (Eds) *Proceedings of the 13th International Conference on Automated Deduction*, pp.141-145, New

Brunswick, USA, (1996), Springer-Verlag (Lecture Notes in Artificial Intelligence 1104), New York, USA.

41. L. Wos, R. Overbeek, E. Lusk, and J. Boyle. *Automated Reasoning - Introduction and Applications*, McGraw-Hill, 1992.
42. L. Wos. *Email to G. Sutcliffe*, 1996.

8. Old Stuff Not Used

8.1 Old Sutcliffe

REMOVE ME **** Geoff do you want to put this under Jim Adams?? ***

An alternative method was proposed by ??Adams?? \cite{adams}.

- Remove implications.
- Move negations in
- Skolemize
- Remove universal quantifiers.
- Conjunctive Normal Form.

8.1.1 ABOUT THE TPTP CLAUSIFIER

Our aim in investigating these conversion techniques is to deduce an algorithm which will produce clauses with the minimal number of symbols. The occurance that creates the greatest growth in the number of symbols within a formula is skolemization. During this process, an existentially quantified variable is substituted by a function of all universally quantified variables that it is within the scope of. In order to reduce this number of variables it is necessary to ensure that the scope of any quantifier is no larger that needed. Thus in contrast to the first two techniques shown, instead of moving quantifiers to the left (thereby widening their scope), they are moved right, into the formula. When their scope is reduced, skolemization then produces terms with considerably less literals. This technique of reducing the scope of quantifiers is named {\em miniscoping} and has been used by a variety of systems including the Boyer--Moore and Conpro1 theorem provers~\cite{kaufmann92, quaife, eisinger91}.

Problem	B	B+S	Q	Q+S	O	O+S	T	T+S	B+S /B	Q+S /Q	O+S /O	T+S /T
SYN040-1	24	0	24	0	24	0	24	0	0.00	0.00	0.00	0.00
SYN041-1	8	0	8	0	8	0	8	0	0.00	0.00	0.00	0.00
SYN001-1	8	0	8	0	8	0	8	0	0.00	0.00	0.00	0.00
SYN045-1	88	0	88	0	88	0	88	0	0.00	0.00	0.00	0.00
SYN044-1	26	26	26	26	26	26	26	26	1.00	1.00	1.00	1.00
SYN046-1	24	0	24	0	24	0	24	0	0.00	0.00	0.00	0.00
SYN047-1	228	0	228	0	228	0	228	0	0.00	0.00	0.00	0.00
SYN048-1	7	0	6	0	7	0	6	0	0.00	0.00	0.00	0.00
SYN049-1	14	0	12	0	14	0	12	0	0.00	0.00	0.00	0.00
SYN050-1	29	0	27	0	29	0	27	0	0.00	0.00	0.00	0.00
SYN051-1	30	20	30	20	30	20	30	20	0.67	0.67	0.67	0.67
SYN052-1	30	20	30	26	30	26	30	26	0.67	0.87	0.87	0.87
SYN053-1	42	20	30	0	30	0	30	0	0.48	0.00	0.00	0.00
SYN054-1	39	39	39	39	39	39	39	39	1.00	1.00	1.00	1.00
SYN055-1	53	37	51	36	51	36	51	36	0.70	0.71	0.71	0.71
SYN056-1	98	98	72	72	72	72	72	72	1.00	1.00	1.00	1.00
SYN057-1	39	39	39	39	39	39	39	39	1.00	1.00	1.00	1.00
SYN058-1	48	30	48	30	48	30	48	30	0.63	0.63	0.63	0.63
SYN059-1	426	420	186	144	186	144	186	144	0.99	0.77	0.77	0.77
SYN060-1	39	39	39	39	39	39	39	39	1.00	1.00	1.00	1.00
SYN061-1	30	24	30	24	30	24	30	24	0.80	0.80	0.80	0.80

SYN062-1	42	33	42	33	42	33	42	33	0.79	0.79	0.79	0.79
SYN063-1	386	20	342	0	342	0	342	0	0.05	0.00	0.00	0.00
SYN064-1	12	0	8	0	12	0	8	0	0.00	0.00	0.00	0.00
SYN065-1	62	62	62	62	62	62	62	62	1.00	1.00	1.00	1.00
SYN066-1	61	61	57	57	61	61	57	57	1.00	1.00	1.00	1.00
SYN068-1	46	38	46	38	46	38	46	38	0.83	0.83	0.83	0.83
SYN069-1	150	150	141	91	141	91	141	91	1.00	0.65	0.65	0.65
SYN070-1	82	58	82	58	82	58	82	58	0.71	0.71	0.71	0.71
SYN071-1	48	48	48	48	48	48	48	48	1.00	1.00	1.00	1.00
SYN072-1	55	55	55	55	55	55	55	55	1.00	1.00	1.00	1.00
SYN073-1	15	15	13	13	13	13	13	13	1.00	1.00	1.00	1.00
SYN074-1	446	156	372	372	372	234	372	372	0.35	1.00	0.63	1.00
SYN075-1	446	156	372	372	372	234	372	372	0.35	1.00	0.63	1.00
SYN077-1	166	166	138	138	138	138	138	138	1.00	1.00	1.00	1.00
SYN078-1	165	165	105	105	105	105	105	105	1.00	1.00	1.00	1.00
SYN079-1	36	36	36	36	36	36	36	36	1.00	1.00	1.00	1.00
SYN080-1	58	58	58	58	58	58	58	58	1.00	1.00	1.00	1.00
SYN081-1	21	21	21	21	21	21	21	21	1.00	1.00	1.00	1.00
SYN082-1	135	35	126	126	126	116	126	126	0.26	1.00	0.92	1.00
SYN083-1	76	76	76	76	76	76	76	76	1.00	1.00	1.00	1.00
SYN084-1	592	172	434	133	434	65	434	133	0.29	0.31	0.15	0.31
SYN315-1	34	22	30	26	34	22	30	26	0.65	0.87	0.65	0.87
SYN316-1	28	14	28	22	28	14	28	22	0.50	0.79	0.50	0.79
SYN317-1	36	0	36	0	36	0	36	0	0.00	0.00	0.00	0.00
SYN318-1	23	0	17	0	18	0	17	0	0.00	0.00	0.00	0.00
SYN319-1	77	49	57	48	77	49	57	48	0.64	0.84	0.64	0.84
SYN320-1	22	0	19	0	22	0	19	0	0.00	0.00	0.00	0.00
SYN321-1	48	32	52	44	52	44	52	44	0.67	0.85	0.85	0.85
SYN322-1	16	0	16	0	16	0	16	0	0.00	0.00	0.00	0.00
SYN323-1	32	32	32	32	32	32	32	32	1.00	1.00	1.00	1.00
SYN324-1	62	40	62	40	62	40	62	40	0.65	0.65	0.65	0.65
SYN325-1	40	20	40	20	40	20	40	20	0.50	0.50	0.50	0.50
SYN326-1	52	28	52	28	52	28	52	28	0.54	0.54	0.54	0.54
SYN327-1	67	41	67	51	67	41	67	51	0.61	0.76	0.61	0.76
SYN328-1	102	102	99	99	102	102	99	99	1.00	1.00	1.00	1.00
SYN329-1	20	0	20	0	20	0	20	0	0.00	0.00	0.00	0.00
SYN330-1	108	80	108	80	108	80	108	80	0.74	0.74	0.74	0.74
SYN331-1	60	60	60	60	60	60	60	60	1.00	1.00	1.00	1.00
SYN332-1	382	178	382	178	382	178	382	178	0.47	0.47	0.47	0.47
SYN333-1	68	0	56	0	68	0	56	0	0.00	0.00	0.00	0.00
SYN334-1	124	100	124	100	124	100	124	100	0.81	0.81	0.81	0.81
SYN335-1	168	148	168	148	168	148	168	148	0.88	0.88	0.88	0.88
SYN336-1	28	10	28	10	28	10	28	10	0.36	0.36	0.36	0.36
SYN337-1	23	0	23	0	23	0	23	0	0.00	0.00	0.00	0.00
SYN338-1	15	0	15	0	15	0	15	0	0.00	0.00	0.00	0.00
SYN339-1	12	12	12	12	12	12	12	12	1.00	1.00	1.00	1.00
SYN340-1	16	16	16	16	16	16	16	16	1.00	1.00	1.00	1.00
SYN341-1	14	14	14	14	14	14	14	14	1.00	1.00	1.00	1.00
SYN342-1	11	0	11	0	11	0	11	0	0.00	0.00	0.00	0.00
SYN343-1	27	0	25	0	27	0	25	0	0.00	0.00	0.00	0.00
SYN344-1	48	28	48	28	48	28	48	28	0.58	0.58	0.58	0.58
SYN345-1	133	81	133	81	133	81	133	81	0.61	0.61	0.61	0.61
SYN346-1	36	36	36	36	36	36	36	36	1.00	1.00	1.00	1.00
SYN347-1	344	0	344	0	344	0	344	0	0.00	0.00	0.00	0.00
SYN349-1	4099	226	4099	226	4099	226	4099	226	0.06	0.06	0.06	0.06
SYN350-1	174	86	174	86	174	86	174	86	0.49	0.49	0.49	0.49
SYN351-1	234	44	234	44	234	44	234	44	0.19	0.19	0.19	0.19
SYN352-1	176	104	176	104	176	104	176	104	0.59	0.59	0.59	0.59
SYN353-1	398	310	398	310	398	310	398	310	0.78	0.78	0.78	0.78
SYN354-1	124	80	124	80	124	80	124	80	0.65	0.65	0.65	0.65
SYN355-1	30	0	18	18	18	18	18	18	0.00	1.00	1.00	1.00
SYN356-1	36	36	36	36	36	36	36	36	1.00	1.00	1.00	1.00
SYN357-1	6	0	6	0	6	0	6	0	0.00	0.00	0.00	0.00
SYN358-1	30	0	30	0	30	0	30	0	0.00	0.00	0.00	0.00
SYN359-1	28	0	26	26	26	26	26	26	0.00	1.00	1.00	1.00
SYN360-1	44	0	17	17	17	17	17	17	0.00	1.00	1.00	1.00
SYN361-1	36	0	23	17	23	17	23	17	0.00	0.74	0.74	0.74

SYN362-1	16	10	15	9	15	9	15	9	0.63	0.60	0.60	0.60
SYN363-1	20	20	17	17	17	17	17	17	1.00	1.00	1.00	1.00
SYN364-1	75	0	43	43	43	43	43	43	0.00	1.00	1.00	1.00
SYN365-1	49	0	41	41	41	41	41	41	0.00	1.00	1.00	1.00
SYN366-1	56	56	40	40	40	40	40	40	1.00	1.00	1.00	1.00
SYN367-1	32	0	26	22	26	22	26	22	0.00	0.85	0.85	0.85
SYN368-1	7	0	6	0	7	0	6	0	0.00	0.00	0.00	0.00
SYN369-1	16	16	13	13	13	13	13	13	1.00	1.00	1.00	1.00
SYN370-1	22	14	22	14	22	14	22	14	0.64	0.64	0.64	0.64
SYN371-1	22	0	16	0	18	0	16	0	0.00	0.00	0.00	0.00
SYN372-1	12	0	12	0	12	0	12	0	0.00	0.00	0.00	0.00
SYN373-1	36	0	36	0	36	0	36	0	0.00	0.00	0.00	0.00
SYN374-1	240	0	256	74	256	52	256	74	0.00	0.29	0.20	0.29
SYN375-1	800	80	240	21	242	21	240	21	0.10	0.09	0.09	0.09
SYN376-1	22	0	18	18	18	18	18	18	0.00	1.00	1.00	1.00
SYN377-1	240	0	240	21	242	21	240	21	0.00	0.09	0.09	0.09
SYN378-1	14	0	12	0	12	0	12	0	0.00	0.00	0.00	0.00
SYN379-1	16	0	16	0	16	0	16	0	0.00	0.00	0.00	0.00
SYN380-1	26	26	24	24	24	24	24	24	1.00	1.00	1.00	1.00
SYN381-1	55	23	27	27	27	27	27	27	0.42	1.00	1.00	1.00
SYN382-1	22	0	17	17	18	18	17	17	0.00	1.00	1.00	1.00
SYN383-1	14	0	12	0	14	0	14	0	0.00	0.00	0.00	0.00
SYN384-1	12	0	11	0	12	0	11	0	0.00	0.00	0.00	0.00
SYN385-1	10	0	9	0	10	0	9	0	0.00	0.00	0.00	0.00
SYN386-1	159	0	56	56	56	56	56	56	0.00	1.00	1.00	1.00
SYN387-1	4	0	4	0	4	0	4	0	0.00	0.00	0.00	0.00
SYN388-1	4	0	4	0	4	0	4	0	0.00	0.00	0.00	0.00
SYN389-1	10	0	10	0	10	0	10	0	0.00	0.00	0.00	0.00
SYN390-1	8	0	8	0	8	0	8	0	0.00	0.00	0.00	0.00
SYN391-1	16	16	16	16	16	16	16	16	1.00	1.00	1.00	1.00
SYN392-1	160	16	160	16	160	16	160	16	0.10	0.10	0.10	0.10
SYN393-1	3552	48	3552	48	3552	48	3552	48	0.01	0.01	0.01	0.01
SYN394-1	13	0	12	0	12	0	12	0	0.00	0.00	0.00	0.00
SYN395-1	14	0	12	0	12	0	12	0	0.00	0.00	0.00	0.00
SYN396-1	12	0	12	0	12	0	12	0	0.00	0.00	0.00	0.00
SYN397-1	20	0	12	0	12	0	12	0	0.00	0.00	0.00	0.00
SYN398-1	42	0	30	0	30	0	30	0	0.00	0.00	0.00	0.00
SYN399-1	38	30	30	26	30	26	30	26	0.79	0.87	0.87	0.87
SYN400-1	8	0	8	0	8	0	8	0	0.00	0.00	0.00	0.00
SYN401-1	6	0	6	0	6	0	6	0	0.00	0.00	0.00	0.00
SYN402-1	6	0	6	0	6	0	6	0	0.00	0.00	0.00	0.00
SYN403-1	18	0	18	0	18	0	18	0	0.00	0.00	0.00	0.00
SYN404-1	6	0	6	0	6	0	6	0	0.00	0.00	0.00	0.00
SYN405-1	13	0	12	0	12	0	12	0	0.00	0.00	0.00	0.00
SYN406-1	20	20	18	18	18	18	18	18	1.00	1.00	1.00	1.00
SYN407-1	23	23	21	21	21	21	21	21	1.00	1.00	1.00	1.00
SYN408-1	11	0	9	0	9	0	9	0	0.00	0.00	0.00	0.00
SYN409-1	30	0	21	0	21	0	21	0	0.00	0.00	0.00	0.00
SYN410-1	8	0	8	0	8	0	8	0	0.00	0.00	0.00	0.00
SYN411-1	56	0	20	0	20	0	20	0	0.00	0.00	0.00	0.00
SYN412-1	16	0	16	0	16	0	16	0	0.00	0.00	0.00	0.00
SYN413-1	37	37	35	35	35	35	35	35	1.00	1.00	1.00	1.00
SYN414-1	358	358	172	72	172	72	172	72	1.00	0.42	0.42	0.42
SYN415-1	219	219	163	145	163	145	163	145	1.00	0.89	0.89	0.89
SYN416-1	8	0	8	0	8	0	8	0	0.00	0.00	0.00	0.00
									0.41	0.491	0.478	0.491