The application of goal-oriented heuristics for proving equational theorems via the unfailing Knuth-Bendix completion procedure
A case study: lattice ordered groups[1]

Matthias Fuchs
FB Informatik
Universität Kaiserslautern
67663 Kaiserslautern
Germany
E-mail: fuchs@informatik.uni-kl.de

February 1994

Abstract

In this report we present a case study of employing goal-oriented heuristics when proving equational theorems with the (unfailing) Knuth-Bendix completion procedure. The theorems are taken from the domain of lattice ordered groups. It will be demonstrated that goal-oriented (heuristic) criteria for selecting the next critical pair can in many cases significantly reduce the search effort and hence increase performance of the proving system considerably. The heuristic, goal-oriented criteria are on the one hand based on so-called "measures" measuring occurrences and nesting of function symbols, and on the other hand based on matching subterms. We also deal with the property of goal-oriented heuristics to be particularly helpful in certain stages of a proof. This fact can be addressed by using them in a framework for distributed (equational) theorem proving, namely the "teamwork-method".

0. Introduction

The completion-procedure initially proposed by D.E. Knuth and P.B. Bendix (the *KB-procedure* [KB70]) together with further extensions and improvements (the *unfailing KB-procedure* (*UKB-procedure*) [BDP89]) has also proved to be an important tool for proving theorems in equational theories. The major drawback of its usefulness for proving resides in what it was originally designed for, namely for deriving a complete (i.e. convergent) set of rules from a given set of (equational) axioms to yield a decision procedure for the respective equational theory. Since there are in this case hardly any hints to what kind of rules resp. equations might be needed, the strategies (heuristics) employed in the completion process are entirely forward oriented. This way of proceeding does not make sense in case the UKB-procedure is used for proving if no convergent set of rules can be generated. Under these conditions, the theorem to be proved (the goal) can give valuable clues how the rules and equations the UKB-procedure should generate may look like. Forward oriented strategies completely ignore this kind of information and always exhibit the same behaviour regardless of the given goal. This is neither satisfactory nor acceptable because goal-oriented strategies can considerably reduce the search-effort by pruning the most of the time enormous search space and thus substantially increase efficiency. Therefore

1. This work was supported by the Deutsche Forschungsgemeinschaft (DFG).

attempts have been made to devise strategies that allow the UKB-procedure to make use of the goal when choosing the next rule or equation (see, for instance, [AA90]).

In this report we shall demonstrate the usefulness of goal-oriented heuristics by applying them to the domain of lattice ordered groups. (Other examples can be found in appendix A.) The heuristics that will be used are on the one hand based on so-called measures ([AA90]) and on the other hand on matching subterms, where the latter will mainly be employed in a framework for distributed theorem proving (the “teamwork-method”, [De93], [AD93]) which is particularly apt for handling rather specialized heuristics without loss of completeness. Goal-oriented heuristics belong to that category, their degree of specialization certainly depending on their individual realization. (The heuristic based on measures is an example for a goal-oriented heuristic which is at the lower end of the range of specialization, whereas the heuristics based on matching subterms are considerably specialized.). We shall address this topic in more detail in section 3.

The coming sections are organized as follows:
The first section is to make the reader familiar with the foundations of the UKB-procedure and its application to proving in equational theories. Section 2 will introduce the fundamentals of lattice ordered groups which is the domain of our concern. After that, section 3 will deal with the basics of the teamwork-method which will provide the context for some of the goal-oriented strategies, as it was already mentioned above. The goal-oriented heuristics themselves will be described in section 4. In the subsequent section 5, proofs of theorems in the domain of lattice ordered groups using the goal-oriented heuristics presented in section 4 will be discussed, thus displaying their advantages compared to commonly used (forward-oriented) heuristics, but also pointing out some limitations. Section 6 will summarize this report.

All proofs dealt with in this report were conducted by the DISCOUNT-system ([Pi92], [DP92]). Their respective analyses were supported by related software-tools for proof-analysis and -processing ([Sch93]).

1. Equational theorem proving with the UKB-procedure

The KB-procedure ([KB70]) was initially designed for deriving a complete (convergent) set of rules given a set E of equations. Rules are equations whose sides can be compared (oriented) with a reduction ordering > and hence only the application into one direction (from the bigger to the smaller side) needs be considered. The application of a rule, i.e. rewriting resp. reducing, consists in replacing an instance $\sigma(l)$ of a left side of a rule by the respective instance $\sigma(r)$ of the right side (where σ is the appropriate match). Once a complete set of rules is derived, the termination and the convergence of the related reduction relation allow to *decide* the word problem $s =_E t$ defined by E by checking whether the respective normalforms of s and t are identical. Since the word problem is (in general) undecidable, it is obvious that a complete set of rules cannot always exist.

The central inference of the KB-procedure is the generation of new equations (critical pairs) by overlapping the left sides of (not necessarily distinct) rules. The selection strategy for picking the next critical pair to become a new rule is crucial for the efficiency of the procedure, and may be even vital for its success. The notion “fairness” (of a selection strategy) must be seen in this context: A selection strategy is *fair* if every critical pair is taken into consideration for becoming a rule after a finite number of inference steps.

Extensions to the initial version of the KB-procedure, which fails if some critical pairs are not orientable with the given reduction ordering (even though a complete set of rules might exist) made it also interesting for proving equational theorems in general. The idea is to treat a non-orientable equation u=v as two “rules” $u \to v$ and $v \to u$ (what of course necessitates checking $\sigma(u)>\sigma(v)$ resp. $\sigma(v)>\sigma(u)$ when it is used for reduction, i.e. rewriting, σ being the current match). The resulting UKB-procedure (*unfailing* KB-procedure, [BDP89]) can be used as a semi-decision procedure for the word problem defined by E: The UKB-procedure is applied as in the case a convergent system is sought. The equation $s=_E t$ which is to be proved (also referred to as the goal) is negated (thus turning the variables into skolem-constants[1]) and is always kept in normalform w.r.t. the current set of rules and equations. A proof is found if the (negated) goal can be reduced to $s' \neq s'$.

The selection strategy for choosing the next critical pair is here even more important, since we are potentially dealing with cases where no complete systems exist. Experiments have shown that rather naive selection strategies (such as FIFO, for instance) have no chance to cope with the quite often enormous search spaces. Furthermore, the fairness of the strategy is essential to guarantee the completeness of the UKB-procedure. Here, completeness denotes the ability to find a proof in finite time, provided that the goal is actually a consequence of E.

The size and the growth rate of the search space, both of course depending on the current problem, paired with the general undecidability of the word problem, call for the use of powerful heuristics. As such, goal-oriented heuristics play an important role as this report will demonstrate. Moreover, the sheer intractability of a huge search space by only one heuristic, however powerful it may be, points out the necessity to combine heuristics, not by mixing them into one, thus producing just another heuristic, but rather by letting them all search their way through the search space, profiting from each other’s “discoveries”. The teamwork-method forms a basis for such an approach, and especially in this environment goal-oriented heuristics prove to be advantageous (see also [De93] and sections 3 and 5).

2. Lattice ordered groups

In this section the fundamentals of lattice ordered groups will be outlined (see [KK74] for details). The main purpose of the presentation of the definition of lattice ordered groups consists in establishing a set of equational axioms axiomatizing this domain. These axioms will be used as basic axioms throughout this report and were utilized by DISCOUNT for proving the theorems to come. We shall now give the formal definitions of the notions group, partial order, lattice and finally lattice ordered group. After that, a set of equational axioms axiomatizing the theory of lattice ordered groups will be proposed ([DGW93]).

Definition 2.1: *Group*
(G,f) is said to be a **group**, if G is a (non-empty) set, f is a function $f:G \times G \to G$, where f is associative, i.e. $f(f(x,y),z)=f(x,f(y,z))$ for all $x,y,z \in G$, and there is $1 \in G$ with $f(1,x)=x$ for all $x \in G$ (1 is referred to as the neutral element), and for all $x \in G$ there is a $y \in G$ with $f(y,x)=1$ (y is called the inverse of x).

Definition 2.2: *Partial Order*
A set M is said to be **partially ordered**, if there is a binary relation $\leq \subseteq M \times M$ with
(a) $x \leq x$ for all $x \in M$ (*reflexivity*)
(b) $x \leq y \wedge y \leq x$ implies $x=y$ for all $x,y \in M$ (*antisymmetry*)
(c) $x \leq y \wedge y \leq z$ implies $x \leq z$ for all $x,y,z \in M$ (*transitivity*)

1. By convention, all variables occurring in an equation are implicitly all-quantified. For extensions see [De93].

$\leq$ is referred to as the **partial order**.

Definition 2.3: *Lattice*
A partially ordered set M under the partial order $\leq$ is said to be a **lattice**, if for any pair of elements $x,y \in M$ there exist $l,u \in M$ with the following properties

- $l \leq x$, $l \leq y$ and $z \leq x \wedge z \leq y$ implies $z \leq l$ for all $z \in M$ (l is the *greatest lower bound*)
- $x \leq u$, $y \leq u$ and $x \leq z \wedge y \leq z$ implies $u \leq z$ for all $z \in M$ (u is the *least upper bound*)

Definition 2.4: *Lattice Ordered Group*
A group (G,f) is said to be a **lattice ordered group**, if G is a lattice under the partial order $\leq$ and for all $x,y,z \in G$ $x \leq y$ implies $f(z,x) \leq f(z,y)$ and $f(x,z) \leq f(y,z)$ (*laws of monotonicity*).
$(G,f,\leq)$ denotes the lattice ordered group.

It is not hard to convert the definition for group into equational axioms. The well-known axiomatization with three axioms is chosen (all variables x,y,z are assumed to be all-quantified; we can designate the inverse of x as i(x), since for each x its inverse exists (and is even unique)):

(1) $f(1,x)=x$
(2) $f(i(x),x)=1$
(3) $f(f(x,y),z)=f(x,f(y,z))$

We shall now present the missing axioms completing the axiomatization. The crucial problem consists in expressing the partial order $\leq$, which is a relation resp. a predicate, purely with equations. The following corollary is the key to the solution.

Corollary 2.1:
Let $(G,f,\leq)$ be a lattice ordered group. Let furthermore l(x,y) denote the greatest lower bound for any pair $x,y \in G$, u(x,y) denote the least upper bound of any pair $x,y \in G$. Then, we have:
(I) $\forall x,y \in G$: $x \leq y$ if and only if $l(x,y)=x$
(II) $\forall x,y \in G$: $x \leq y$ if and only if $u(x,y)=y$
(Note: The greatest lower bound and the least upper bound do exist for any pair $x,y \in G$ according to definition 2.3, and they are unique since, if c_1,c_2 are greatest lower (least upper) bounds of x and y, then $c_1 \leq c_2$ and $c_2 \leq c_1$ and therefore $c_1=c_2$ because of definition 2.2.b.)
Proof:
(I) [a] $x \leq y$ implies $l(x,y)=x$ for all $x,y \in G$.
We have $x \leq y$ (hypothesis) and $x \leq x$ (from definition 2.2.a). Using definition 2.3, we obtain $x \leq l(x,y)$ and $l(x,y) \leq x$. Therefore $x=l(x,y)$ (cf. definition 2.2b).
[b] $l(x,y)=x$ implies $x \leq y$ for all $x,y \in G$
According to definition 2.3 we have $l(x,y) \leq x$ and $l(x,y) \leq y$, the latter statement immediately yielding $x \leq y$ with hypothesis $l(x,y)=x$.
(II) Analogous to (I).

❒

Thus, in the following, the binary function-symbols l and u represent the least upper and the greatest lower bound, respectively.

(4) $l(x,y)=l(y,x)$
(5) $u(x,y)=u(y,x)$
(6) $l(l(x,y),z)=l(x,l(y,z))$

(7) u(u(x,y),z)=u(x,u(y,z))
(8) l(x,x)=x
(9) u(x,x)=x
(10) u(x,l(x,y))=x
(11) l(x,u(x,y))=x
(12) f(x,l(y,z))=l(f(x,y),f(x,z))
(13) f(l(x,y),z)=l(f(x,z),f(y,z))
(14) f(x,u(y,z))=u(f(x,y),f(x,z))
(15) f(u(x,y),z)=u(f(x,z),f(y,z))

The proofs necessary to show that any lattice ordered group is a model of this axiomatization are listed in appendix B (correctness of the axiomatization). The remaining proofs corroborating the completeness of the axiomatization (i.e. any model of axioms (1) through (15) is a lattice ordered group) were conducted automatically by the DISCOUNT-system and are a part of section 5.

After this concise presentation of the formulation of lattice ordered groups as an equational theory the teamwork-method will be introduced in the following section.

3. The teamwork-method

The teamwork-method ([De93],[AD93]) is a framework for distributing deduction (or, more generally speaking, for problem solving procedures that rely on the generation of facts). Its design is based on the behaviour of a team of human experts. Its major components are a supervisor and a batch of experts and referees. These components will now be briefly discussed in the context of the application of the teamwork-method to (equational) theorem proving by the UKB-procedure.

As in human teams the supervisor is responsible for giving each expert (i.e. the members of the team) the problem at hand, and calling team meetings from time to time. During a team meeting so-called referees assess the work accomplished so far by each and every expert. The problem state of the expert which is considered to be the best at the moment of the team meeting (w.r.t. the assessment of the referees) is adopted by the supervisor and is supplemented with results from all other experts. These results are also selected by the referees. The problem specification obtained this way is again assigned to each and every expert. Furthermore, experts can be exchanged if they turn out to be performing significantly worse than other members of the team. Between two meetings the experts work independently of each other on the problem. There is therefore no exchange of information between two meetings. Consequently, the working phase can be efficiently realized by parallel processes.

Thus the problem solving process consists of several cycles (as many as it takes to find a solution) each of which has two phases:

phase 1: composition (modification) of a team, assignment of the current problem specification to each expert by the supervisor;
phase 2: The experts work on the problem independently (in a distributed environment). At the end of this phase referees assess the achievements of each expert and select the results considered best in order to supply the supervisor with information for phase 1 of the next cycle.

In our case of employing the UKB-procedure, the problem consists in proving a given equational theorem. Experts correspond to the completion procedure using distinct heuristics for selecting critical pairs. In the current implementation DISCOUNT ([Pi92],[DP92]), referees assess the work done by each expert on account of statistical data such as, for instance, the number of rules and equations generated. Results are also picked on that basis.

Thus, the teamwork-method allows the exploration of different paths of the search space, realized through the distinct strategies applied by each expert (using experts with the same strategy is not forbidden, but does not make any sense). Favourable discoveries (e.g. rules that account for many simplifying reductions) of the experts during their search can be detected by referees during a team meeting. In including good results into the problem state of the best expert, synergetic effects can take place, possibly boosting experts into positions in the search space they would never have reached in a reasonable time if they had worked alone. This is particularly true when distributing (equational) theorem proving via the UKB-procedure with the teamwork-method, and when employing goal-oriented heuristics. Some heuristics in this category are quite specialized in the sense that, in some situations, they get straight down to a solution, while in many other situations they wander around without getting anywhere (in acceptable time)[1]. Although the employment of such heuristics alone usually results in failures, they proved to be extremely beneficial when being used as members of a team, consequently profiting from the mentioned synergetic effects, but also being the only heuristic that could find the remaining path to a solution quickly.

After this concise presentation of the teamwork-method, the subsequent section will introduce two kinds of goal-oriented heuristics, both of which will be applied to problems in the domain of lattice ordered groups (see section 5 for the latter).

4. Goal-oriented heuristics

Automatic theorem proving systems based on the (unfailing) KB-procedure are forward reasoning systems by design. The lack of being goal-oriented has always been a major disadvantage of this method for proving in equational theories. One way to overcome this drawback consists in creating heuristics for the control of the crucial inference rule, namely the selection of the next critical pair, which -in some way- incorporate aspects of the goal into the selection criteria. Two principles how this can be achieved will now be presented.

4.1. Goal-orientation through measures

S. Anantharaman and N. Andrianarievelo ([AA90]) proposed a method for selecting the next critical pair which is based on so-called measures. These measures are basically integer values expressing, for instance, the number of occurrences of function-symbols in a given critical pair. A relation to the goal is established by comparing the corresponding values obtained from a critical pair with the respective values of the goal. The idea is to prefer those critical pairs whose measures best coincide with the measures of the goal. This rather simple technique proved to be quite effective.

We shall now outline how measures were implemented to be used by DISCOUNT as a heuristic for picking the next critical pair. (Note: We do not at all claim that our implementation is optimal in the sense that there is no other variation that would increase performance when employed by DISCOUNT. It is merely one way to utilize the ideas connected with measures. Furthermore, other measures than those considered here may also be useful

1. It should be noted here that the teamwork-method allows to use unfair strategies without necessarily losing completeness: Only the team as a whole needs be fair (*team-fair*, [De93]).

(cf. [AA90]).) At the end of this section we shall concisely point out the main differences between the way we used measures and the way they were employed in [AA90].

First of all the weighting of critical pairs (i.e. associating an integer value with a critical pair) is based on a general weighting function ϕ for terms which is also used for some "standard" heuristics used as standard of comparison to substantiate the superior performance of goal-oriented heuristics in the domain of lattice ordered groups (see section 5 and appendix A for examples from a different domain). The convention to prefer critical pairs with small weights was adopted. The general weighting function ϕ is defined as follows:

Definition 4.1.1: *weight of a term*
Let t be a term. The weight $\phi(t)$ of t is
(a) $\phi(t)=1$, if t is a variable
(b) $\phi(t)=2$, if t is a constant
(c) $\phi(t)=2+f(t_1)+..+f(t_n)$, if $t\equiv f(t_1,..,t_n)$

One of the standard-heuristics weights a critical pair $u \leftrightarrow v$ with $\phi(u)+\phi(v)$. This heuristic, which we shall call add, was also chosen to form the basis of the goal-oriented heuristic based on measures. The measures themselves are used to compute multipliers m_f for each and every function-symbol f depending on the occurrences and nesting of f in the critical pair to be weighted compared to the occurrences and nesting of f in the goal. The weight finally associated with a critical pair $u \leftrightarrow v$ will be $(\phi(u)+\phi(v))\cdot m_{f1}\cdot\ldots\cdot m_{fn}$, where $F=\{f1,..,fn\}$ is the set of all function-symbols defined by the current signature.

As we have just alluded, the measures we are considering here are occurrences and nesting of function-symbols in a critical pair resp. goal, i.e. in pairs of terms. We first define both notions for terms and then extend these definitions to pairs of terms.

Definition 4.1.2: *occurrences*
The number of **occurrences** (or the occurrences for short) of a function-symbol f in a term t is given by
$occ(f,t)=0$, if t is a variable
$occ(f,t)=occ(f,t_1)+..+occ(f,t_n)$, if $t\equiv g(t_1,..,t_n)$, where $f\neq g$
$occ(f,t)=1+occ(f,t_1)+..+occ(f,t_n)$, if $t\equiv f(t_1,..,t_n)$

Definition 4.1.3: *nesting*
The **nesting** of a function-symbol f in a term t is given by
$nest(f,t)=0$, if f is a constant
$nest(f,t)=hnest(f,t,0,0)$, if f is not a constant, where
$hnest(f,t,current,absolute)=MAX(\{current,absolute\})$, if t is a variable or a constant
$hnest(f,t,current,absolute)=MAX(\{hnest(f,t_i,0,MAX(\{current,absolute\}))|1\leq i\leq n\})$, if $t\equiv g(t_1,..,t_n)$ and $f\neq g$
$hnest(f,t,current,absolute)=MAX(\{hnest(f,t_i,current+1,absolute)|1\leq i\leq n\})$, if $t\equiv f(t_1,..,t_n)$

Example 4.1.1
Let $t\equiv f(g(a,x),f(a,g(a,b)))$. Then

$occ(f,t) =2$	$nest(f,t) =2$
$occ(g,t) =2$	$nest(g,t) =1$
$occ(a,t) =3$	$nest(a,t) =0$
$occ(b,t) =1$	$nest(b,t) =0$

The following extensions to the definitions 4.1.2 and 4.1.3 make occ and nest applicable to pairs of terms.

Let <u,v> be a pair of terms (e.g. a critical pair).
$occ(f,\langle u,v\rangle):=MAX(\{occ(f,u),occ(f,v)\})$
$nest(f,\langle u,v\rangle):=MAX(\{nest(f,u),nest(f,v)\})$

We are now able to compute the multipliers m_f depending on how the two measures represented by occ and nest correlate w.r.t. the critical pair to be weighted and the current goal. Hence the formal definition of our selection heuristic based on measures (occnest) can be presented.

Definition 4.1.4: *weighting function occnest*
Let $u \leftrightarrow v$ be a critical pair, $s \neq t$ be the current goal. Let furthermore $D \subseteq F$, where F is the set of all function-symbols (including constants) of the current signature.
$occnest(\langle u,v\rangle)=(\phi(u)+\phi(v))\cdot m_{f1}\cdot\ldots\cdot m_{fn}$, where $fi \in F$ for all $1 \leq i \leq n$;
$m_f=1$ if $f \notin D$, otherwise
$m_f=\theta(\Psi_1(occ(f,\langle u,v\rangle)-occ(f,\langle s,t\rangle)),\Psi_2(nest(f,\langle u,v\rangle)-nest(f,\langle s,t\rangle)))$
and
$\Psi_1,\Psi_2:\mathbf{Z}\to\mathbf{Z}$, Ψ_1,Ψ_2 both monotonous for positive arguments,
$\theta:\mathbf{Z}\times\mathbf{Z}\to\mathbf{Z}$, θ monotonous in both arguments.
($\mathbf{Z}$ denotes the set of all integers.)

Notes:
- The monotony-requirements in the above definition ensure that m_f will increase (or at least will not decrease) the more the occurrences or the nesting of a function-symbol f exceed the respective values found in the goal. Such a proceeding makes sense, because the amount by which critical pairs necessary for the proof overstep the limits set by occurrences or nesting of function-symbols in the goal will usually[1] be bounded by a small (natural) number.
- The instances of Ψ_1, Ψ_2 and θ used when implementing the occnest weighting heuristic in the DISCOUNT-system are:
  $\Psi_1(x)=1$ if $x \leq 0$, otherwise $\Psi_1(x)=x+1$
  $\Psi_2(x)=\Psi_1(x)$ for all $x \in Z$
  $\theta(x,y)=x\cdot y$ for all $x,y \in Z$
- In our current implementation D can be either F or the set of all function-symbols occurring in the goal. (If not stated otherwise D=F is assumed.)

Before we go on, we now review the ideas presented in [AA90] in order to compare those methods to the heuristic just outlined.

Firstly, in [AA90] the set of function symbols is split up into function symbols with fixed arity and varyadic function symbols which are specially treated when computing measures. Varyadic function symbols are basically function symbols which are associative and commutative (AC). Since DISCOUNT (in its current implementation) does not give a special treatment to the AC-theory, all function symbols are considered non-varyadic. In this case occ and m_0 (see [AA90]) coincide, while nest is not among the measures proposed in [AA90].

Secondly, in [AA90] the set of function symbols F_i relevant for the measures (in our case

1. Theoretical considerations of this issue reveal that this is not the case in general (cf. [MOS93]).

this set is designated by D) is computed according to criteria motivated by theoretical considerations. They suggest to determine F_i after each inference step i, not only depending on the goal(s), but also on the current set of rules and equations. So, possible gains through a more sophisticated choice of the relevant function symbols are paid for by increasing the time spent for computations. Since D is only changed if D is the set of all function symbols occurring in the goal *and* the goal could be rewritten, the costs for updating D are at a rather low level.

Furthermore, [AA90] consider some special cases where measures can be used to actually *prove* that certain rules or equations definitively are useless for a proof. Since we intend to make use of occnest purely as a heuristic criterion, we do not contemplate this issue. (Besides, those special cases are not satisfied by the equational axiomatization of lattice ordered groups.)

A further difference consists in the way measures defined on terms are extended so as to be applicable to pairs of terms, i.e. critical pairs, rules, equations and goals. While we always use the maximum (of the measures of both sides of a pair of terms), [AA90] choose the maximum if a goal is to be measured, whereas the minimum is chosen otherwise. Their choice can be explained by the fact that it is essential for some of the properties they associate with measures (e.g. HC_i, p.189). We, being independent of such restrains, picked a "stronger" (i.e. giving higher measures and hence increasing weight) variation which punishes exceeding measures in any side of a critical pair as opposed to exceeding measures in both sides. (Naturally, our choice is not to be considered as being inherently better.)

The last major difference between [AA90] and our approach (occnest) is the way the results of various measures and other selection strategies (such as add) are combined. While [AA90] utilize lexicographic combinations, we decided to produce a single weight, what makes us less prone to the unfavourable situation where a profitless strategy or measure dominates more advantageous ones (if it occurs earlier in the list of strategies and neasures), but, admittedly, entails a loss of transparency of the effects each strategy resp. measure has.

Once again we want to emphasize that we do not claim nor argue that neither the methods described in [AA90] nor our approach are superior or inferior to each other. We merely point out that (heuristic) selection strategies related to those advocated in [AA90] can indeed be very profitable.

We have now completed the presentation of the heuristic for selecting critical pairs which is based on measures. Despite its simplicity and the fact that much of the possible features of heuristics based on measures remain unexplored (cp. the concluding remarks of section 5) it proved to be considerably superior to so-called standard heuristics (see section 5 and appendix A). In the sequel, two related goal-oriented heuristics which follow a different approach will be examined.

4.2. Goal-orientation through matching

Proving with the UKB-procedure is basically a search for rules and equations that will eventually reduce the goal to a trivial goal $s \neq s$, thus concluding the proof (by refutation). So, on the one hand, one obvious goal-oriented selection criterion for critical pairs is to check whether one or both sides of a critical pair match a subterm of the goal. On the other hand it is quite often the case that one or both sides of a rule or equation have subterms which are generalizations of one or both sides of the goal. This means that there are unwelcome function-symbols clustered around generalizations of one or both sides of the goal. Therefore, a further goal-oriented heuristic consists in looking for critical pairs $u \leftrightarrow v$, instances of which host one or both sides of the goal as a subterm resp. subterms of u or v, the hope being that disturbing function-symbols can be removed by some other rules or equations as the search goes on.

Based on the two principles just outlined two heuristics, called “CP_in_Goal” and “Goal_in_CP”, will be introduced (cp. [De93]).

We begin with CP_in_Goal. The name already expresses that we are looking for (instances of) sides of a critical pair $u \leftrightarrow v$ occurring in the goal $s \neq t$. We consider three cases. (In the following, let $\bar{u} \in \{u,v\}$, $\bar{v} \in \{u,v\}-\{\bar{u}\}$, $\bar{s} \in \{s,t\}$, $\bar{t} \in \{s,t\}-\{\bar{s}\}$; O(t) designates the set of all places of a term t.)

(1) There is a match σ so that $\sigma(u) \equiv \bar{s}|p_1$ and $\sigma(v) \equiv \bar{t}|p_2$, where $p_1 \in O(\bar{s})$, $p_2 \in O(\bar{t})$, i.e. both sides of $u \leftrightarrow v$ match a subterm of a distinct side of the goal $s \neq t$.
(2) There is a match σ so that $\sigma(\bar{u}) \equiv \bar{s}|p$, $p \in O(\bar{s})$, and there is no $p' \in O(\bar{t})$ so that $\sigma(\bar{v}) \equiv \bar{t}|p'$, i.e. exactly one side of $u \leftrightarrow v$ matches a subterm of the goal $s \neq t$.
(3) Neither side of $u \leftrightarrow v$ matches a subterm of the goal.

Note:

These three cases are not exhaustive. Case 1, for instance, could be split by allowing distinct matches. Furthermore, in addition to matches, we could also consider unifiers. This, however, makes sense only if variables occur in the goal. Since the theorems we want to prove are without exception all-quantified, we content ourselves with the three cases listed above.

Because there may be several matches satisfying the conditions of case 1 or 2, we integrate them into the following two sets M_1 and M_2 which reflect case 1 and 2, respectively.

$$M_1(s \neq t, u \leftrightarrow v) := \{(\bar{s},p_1,\bar{t},p_2) \mid \exists\sigma:[\sigma(u) \equiv \bar{s}|p_1 \wedge \sigma(v) \equiv \bar{t}|p_2]\}$$
$$M_2(s \neq t, u \leftrightarrow v) := \{(\bar{s},p) \mid \exists\sigma:[\sigma(\bar{u}) \equiv \bar{s}|p \wedge \forall p' \in O(\bar{t}):\sigma(\bar{v}) \not\equiv \bar{t}|p']\}$$

The general idea behind CP_in_Goal is to favour critical pairs whose sides match subterms of the goal, and the bigger these subterms are, the more suitable the respective critical pair should be considered. (An “optimal” match is there if case 1 applies, twice at top level.) The size of (sub-) terms can be measured with ϕ (cf. definition 4.1.1). Since we want to associate small weights with critical pairs regarded as suitable, it is recommendable to subtract the weights (w.r.t. ϕ) of the subterms of $s \neq t$ matched by $u \leftrightarrow v$ from the total weight (w.r.t. ϕ) of $s \neq t$. In order to distinguish the cases, the results of the subtraction are multiplied with a natural number $\omega_i > 0$ which is associated with case i. Hence:

$$\Psi_1(s \neq t, u \leftrightarrow v) := \{(\phi(s)+\phi(t)-(\phi(\bar{s}|p_1)+\phi(\bar{t}|p_2)))\cdot\omega_1 \mid (\bar{s},p_1,\bar{t},p_2) \in M_1(s \neq t, u \leftrightarrow v)\}$$
$$\Psi_2(s \neq t, u \leftrightarrow v) := \{(\phi(s)+\phi(t)-\phi(\bar{s}|p))\cdot\omega_2 \mid (\bar{s},p) \in M_2(s \neq t, u \leftrightarrow v)\}$$
$$\Psi_3(s \neq t, u \leftrightarrow v) := \{(\phi(s)+\phi(t))\cdot\omega_3\}$$

Finally, we chose to incorporate the weight of $u \leftrightarrow v$ and to pick the minimum of the above weights, yielding:

Definition 4.2.1: *CP_in_Goal*

$$\text{CP\_in\_Goal}(s \neq t, u \leftrightarrow v) = \phi(u)+\phi(v)+\text{MIN}(\Psi_1(s \neq t, u \leftrightarrow v) \cup \Psi_2(s \neq t, u \leftrightarrow v) \cup \Psi_3(s \neq t, u \leftrightarrow v))$$

Note:

In order to reflect the general idea of CP_in_Goal and the importance of the cases (case 1 being the most important, case 2 the second most important), the parameters ω_1, ω_2 and ω_3 should be appropriately selected, at least satisfying $0<\omega_1\leq\omega_2\leq\omega_3$.

Goal_in_CP is defined similarly. Again we contemplate three cases:

(1) There is a match σ so that $\sigma(\bar{u}|p_1)\equiv s$ and $\sigma(\bar{v}|p_2)\equiv t$, where $p_1\in O(\bar{u})$, $p_2\in O(\bar{v})$.
(2) There is a match σ so that $\sigma(\bar{u}|p)\equiv\bar{s}$, $p\in O(\bar{u})$, and there is no $p'\in O(\bar{v})$ with $\sigma(\bar{v}|p')\equiv\bar{t}$.
(3) Neither case 1 nor case 2 apply.

Once again we are potentially dealing with several matches satisfying the conditions of case 1 or 2. Furthermore, any variable x in u or v will cause $u\leftrightarrow v$ to satisfy at least case 2 because $\sigma(x)\equiv s$ resp. $\sigma(x)\equiv t$ is always possible. To obviate these possibly confusing trivial matches, we require the subterms of u and v to have a minimal structure, again measuring the size of the structure with ϕ, representing its lower bound with a natural number ε.

$$M_1^{\varepsilon}(s\neq t,u\leftrightarrow v):=\{(\bar{u},p_1,\bar{v},p_2)\mid\exists\sigma:[\sigma(\bar{u}|p_1)\equiv s\wedge\phi(\bar{u}|p_1)\geq\varepsilon\wedge\sigma(\bar{v}|p_2)\equiv t\wedge\phi(\bar{v}|p_2)\geq\varepsilon]\}$$
$$M_2^{\varepsilon}(s\neq t,u\leftrightarrow v):=\{(\bar{u},p)\mid\exists\sigma:[\sigma(\bar{u}|p)\equiv\bar{s}\wedge\phi(\bar{u}|p)\geq\varepsilon\wedge\forall p'\in O(\bar{t}):[\sigma(\bar{v}|p')\neq\bar{t}\vee\phi(\bar{v}|p')<\varepsilon]]\}$$

Thus

$$\Psi_1^{\varepsilon}(s\neq t,u\leftrightarrow v):=\{(\phi(u)+\phi(v)-(\phi(\bar{u}|p_1)+\phi(\bar{v}|p_2)))\cdot\omega_1\mid(\bar{u},p_1,\bar{v},p_2)\in M_1^{\varepsilon}(s\neq t,u\leftrightarrow v)\}$$
$$\Psi_2^{\varepsilon}(s\neq t,u\leftrightarrow v):=\{(\phi(u)+\phi(v)-\phi(\bar{u}|p))\cdot\omega_2\mid(\bar{u},p)\in M_2^{\varepsilon}(s\neq t,u\leftrightarrow v)\}$$
$$\Psi_3^{\varepsilon}(s\neq t,u\leftrightarrow v):=\{(\phi(u)+\phi(v))\cdot\omega_3\}$$

and we have

Definition 4.2.2: *Goal_in_CP*

$$\text{Goal\_in\_CP}(s\neq t,u\leftrightarrow v,\varepsilon)=\text{MIN}(\Psi_1^{\varepsilon}(s\neq t,u\leftrightarrow v)\cup\Psi_2^{\varepsilon}(s\neq t,u\leftrightarrow v)\cup\Psi_3^{\varepsilon}(s\neq t,u\leftrightarrow v))$$

Considering the design of CP_in_Goal and Goal_in_CP it becomes obvious that both will be the most beneficial if there are critical pairs satisfying either one of the matching criteria (case 1 or 2). Since at least at the beginning of the UKB-procedure there usually are no such critical pairs, these heuristics can prove their usefulness mainly within the team-work-method (cf. 5.2), while occnest is already impressively successful when working individually (cf. 5.1).

The following section will cover a range of proofs taken from the domain of lattice ordered groups.

5. Proving theorems with goal-oriented heuristics

5.1. occnest vs. standard-heuristics

In this section the superior performance of goal-oriented criteria for selecting critical pairs (henceforth *goal-oriented heuristics*) compared to so-called standard criteria (henceforth *standard-heuristics*) in the domain of lattice ordered groups will be demonstrated. Three standard-heuristics[1] were chosen to be competitors of the goal-oriented heuristics. All of

1. We have already come to know one of these, namely add, in section 4.

them rely on the weighting function ϕ for terms introduced by definition 4.1.1, and they weight a critical pair $u \leftrightarrow v$ in the following way:

(a) standard-heuristic add
$\mathsf{add}(u \leftrightarrow v) = \phi(u)+\phi(v)$

(b) standard-heuristic max
$\mathsf{max}(u \leftrightarrow v) = \mathrm{MAX}(\{\phi(u),\phi(v)\})$

(c) standard-heuristic gt ("greater term")
Let > be the reduction-ordering used;

$$\mathsf{gt}(u \leftrightarrow v) = \begin{cases} \phi(u), & \text{if } u>v \\ \phi(v), & \text{if } v>u \\ (\phi(u)+\phi(v)) \text{ div } 2, & \text{if u and v cannot be compared by } >. \end{cases}$$

These three standard-heuristics are the most commonly used heuristic guides for the (unfailing) KB-procedure. Despite their simplicity they have proved to be considerably successful and therefore must be regarded as serious competitors for the goal-oriented heuristics presented in section 4. Moreover, these heuristics do not contain knowledge about the problem to be solved, what also makes them fair competitors, since neither one of the goal-oriented heuristics can profit from such expertise.

The equational axiomatization of lattice ordered groups was introduced in section 2. Since it can be crucial in which order the axioms are given to the UKB-procedure, the following listing represents the order of the axioms as they were supplied to the UKB-procedure for each and every proof, regardless of which heuristic was used.

Set of axioms Λ:
(1) l(x,y)=l(y,x)
(2) u(x,y)=u(y,x)
(3) l(l(x,y),z)=l(x,l(y,z))
(4) u(u(x,y),z)=u(x,u(y,z))
(5) u(x,x)=x
(6) l(x,x)=x
(7) u(x,l(x,y))=x
(8) l(x,u(x,y))=x
(9) f(x,f(y,z))=f(f(x,y),z)
(10) f(1,x)=x
(11) f(i(x),x)=1
(12) f(x,u(y,z))=u(f(x,y),f(x,z))
(13) f(x,l(y,z))=l(f(x,y),f(x,z))
(14) f(u(x,y),z)=u(f(x,z),f(y,z))
(15) f(l(x,y),z)=l(f(x,z),f(y,z))

If it was necessary to include further hypotheses in case the theorem to be proved was a conditional equation, these hypotheses were appended in the order in which they occurred in the antecedent (see also example 5.1 below).

An important parameter of the (unfailing) KB-procedure is the reduction-ordering. For our experiments the lexicographic path ordering (LPO, [De87]) was used exclusively. If not indicated otherwise the precedence was i>f>l>u>1. Possible skolem-constants (cp. section

1) a,b,c,... were appended in alphabetical order to this precedence.

As shown by corollary 2.1, whenever the partial order ≤ appears in a theorem, we have two ways to transform it so that it fits our equational axiomatization. Let us therefore adopt the following convention: Each theorem we want to prove is given a name. If it is prepared for being proved by the DISCOUNT-system using solely the transformation A $x \leq y \Rightarrow u(x,y)=y$, then ".a" is appended to its name. If only the transformation B $x \leq y \Rightarrow l(x,y)=x$ is applied, then ".b" is appended. In case different parts of a theorem are transformed using different transformations, ".c", ".d" etc. are appended, and it will be explicitly specified which transformation was applied to what part of the theorem.

We can now start the comparison of goal-oriented heuristics and standard-heuristics in the light of proving theorems in the domain of lattice ordered groups. The first proofs we turn our attention to are those proofs announced in section 2 which are needed to confirm that our equational axiomatization is complete in the sense given in section 2. All variables are assumed to be all-quantified; $\rightarrow$ denotes the implication:

reflex	:	$x \leq x$
antisym	:	$(x \leq y \wedge y \leq x) \rightarrow x=y$
trans	:	$(x \leq y \wedge y \leq z) \rightarrow x \leq z$
glb1	:	$(z \leq x \wedge z \leq y) \rightarrow z \leq l(x,y)$
glb2	:	$l(x,y) \leq x$
glb3	:	$l(x,y) \leq y$
lub1	:	$(x \leq z \wedge y \leq z) \rightarrow u(x,y) \leq z$
lub2	:	$x \leq u(x,y)$
lub3	:	$y \leq u(x,y)$
mono1	:	$x \leq y \rightarrow f(x,z) \leq f(y,z)$
mono2	:	$x \leq y \rightarrow f(z,x) \leq f(z,y)$

Since the DISCOUNT-system as an instance of an equational prover based on the UKB-completion procedure requires that the theorem to be proved is negated, we shall demonstrate with the following example the process of negation and skolemization and in particular the way the results of this process (the goal and possibly a set of hypotheses) are integrated into the set of axioms Λ.

<u>Example 5.1</u>

Given antisym as the theorem we want to prove, negation and skolemization yields

$$a \leq b \wedge b \leq a \wedge a \neq b$$

(We adopt the convention to skolemize by replacing x,y,z,... with a,b,c,..., respectively.) In order to get rid of ≤ we apply one of the transformations introduced above, let's say $x \leq y \Rightarrow l(x,y)=x$ (transformation B). Thus we obtain $l(a,b)=a \wedge l(b,a)=b \wedge a \neq b$. The hypotheses $l(a,b)=a$ and $l(b,a)=b$ are added to the set of axioms (to be exact: $l(a,b)=a$ and $l(b,a)=b$ are appended to Λ as "axioms" (16) and (17), respectively), while $a \neq b$ becomes the current goal. The complete specification of the problem consisting of the axioms and hypotheses (1) through (17) and the goal $a \neq b$ carries the name antisym.b according to our agreement on naming.

The transformation of the (remaining) theorems given above and of those yet to come will be carried through in a corresponding way and will henceforth not be explicitly outlined.

Table 1 contrasts the performance of occnest with the performance of the best standard-heuristic when proving the above theorems in their various formulations. Columns two and three display the run times (obtained on a SPARCstation 1, averaging at least five runs) of occnest and the best standard-heuristic for the problem whose name is designated in the first column. Note that DISCOUNT does not give any special treatment to underlying theories (such as AC in this case).

Table 1:

name of problem	occnest	best std.-heuristic
refl.a	0.004 sec.	0.009 sec.
refl.b	0.004 sec.	0.010 sec.
antisym.a	0.014 sec.	0.015 sec.
antisym.b	0.014 sec.	0.015 sec.
trans.a	1.329 sec.	0.248 sec.
trans.b	1.267 sec.	0.255 sec.
lub1.a	0.088 sec.	1.589 sec.
lub1.b	1.971 sec.	41.963 sec.
lub1.c[a]	1.978 sec.	41.993 sec.
lub1.d[b]	0.869 sec.	2.250 sec.
lub2.a	0.051 sec.	0.231 sec.
lub2.b	0.029 sec.	0.033 sec.
lub3.a	0.037 sec.	0.042 sec.
lub3.b	0.040 sec.	0.018 sec.
glb1.a	0.600 sec.	27.537 sec.
glb1.b	0.090 sec.	0.544 sec.
glb1.c[c]	0.597 sec.	28.503 sec.
glb1.d[d]	0.896 sec.	0.531 sec.
glb2.a	0.025 sec.	0.027 sec.
glb2.b	0.044 sec.	0.059 sec.
glb3.a	0.016 sec.	0.017 sec.
glb3.b	0.024 sec.	0.057 sec.
mono1.a	0.045 sec.	45.639 sec.
mono1.b	0.045 sec.	46.668 sec.

Table 1:

name of problem	occnest	best std.-heuristic
mono1.c[e]	0.098 sec.	48.135 sec.
mono2.a	0.030 sec.	43.995 sec.
mono2.b	0.030 sec.	44.917 sec.
mono2.c[f]	0.062 sec.	43.322 sec.

a. hypotheses: u(a,c)=c, u(b,c)=c;
goal: l(u(a,b),c)≠u(a,b)
b. hypotheses: l(a,c)=a, l(b,c)=b;
goal: u(u(a,b),c)≠c
c. hypotheses: l(a,c)=c, l(b,c)=c;
goal: u(l(a,b),c)≠l(a,b)
d. hypotheses: u(a,c)=a, u(b,c)=b;
goal: l(l(a,b),c)≠c
e. hypothesis: u(a,b)=b;
goal: l(f(a,c),f(b,c))≠f(a,c)
f. hypothesis: l(a,b)=a;
goal: u(f(c,a),f(c,b))≠f(c,b)

Notes:

- Standard-heuristics work better than occnest for some of the examples listed in table 1. But this does not contradict our claim that occnest is more than a match for standard-heuristics. In taking a closer look at those problems a standard-heuristic was more apt for we recognize that their proofs can be found rather simply, since occnest, which performs in these cases less well, can nonetheless find a proof in less than 1.5 seconds[1]. Hence these examples cannot be considered to indicate a major weakness of occnest.
- The monotony problems (mono1.a, mono1.b, mono2.a, mono2.b) are the first examples for occnest outperforming the *best* standard-heuristic.
- lub1.a and lub1.b as well as glb1.a and glb1.b are the first paradigms for the profitable properties of occnest as an instance of a goal-oriented heuristic, since it cannot be fooled as easily as the standard-heuristics by a slightly different formulation of the goal (due to using a different transformation of ≤).

After these relatively simple and hence not very expressive problems which we contemplated mainly because they provided the missing proofs corroborating the completeness of our axiomatization of lattice ordered groups, we shall now tackle more challenging theorems.

p1 : $x \le y \rightarrow f(i(z),f(x,z)) \le f(i(z),f(y,z))$
p3 : $(x \le y \wedge z \le u) \rightarrow f(x,z) \le f(y,u)$

---

1. At this point we must address the fact that the computations involved in occnest cause it to be more time consuming than any of the standard-heuristics (esp. add and max). On the one hand, this "time-penalty" should be taken into account when comparing the run times. On the other hand, it clarifies the fact that a (significantly) lesser run time of occnest can only stem from a (substantially) smaller amount of rules and equations generated during the proof process.

p4 : $(1 \le x \wedge 1 \le y) \rightarrow 1 \le f(x,y)$
p6 : $1 \le y \rightarrow 1 \le f(i(x),f(y,x))$
p9 : $(1 \le x \wedge 1 \le y \wedge 1 \le z \wedge 1 = l(x,y)) \rightarrow l(x,f(y,z)) = l(x,z)$
p39 : $x \le y \rightarrow i(y) \le i(x)$
lat1 : $1 \le x \rightarrow x \le f(x,x)$
lat3 : $(1 \le x \wedge 1 \le y) \rightarrow x \le f(y,x)$

The subsequent table 2 compares again occnest and the best standard-heuristic. We should mention at the outset that occnest beats all standard-heuristics impressively, not only by finding proofs significantly faster, but also by finding proofs that were beyond the scope of standard-heuristics (marked “>1h”).

Table 2:

name of problem	occnest	best std.-heuristic
p1.a	0.272 sec.	>1h
p1.b	0.281 sec.	>1h
p3.a	4.135 sec.	>1h
p3.b	2.547 sec.	>1h
p3.c[a]	2.522 sec.	>1h
p3.d[b]	4.095 sec.	>1h
p4.a	1.840 sec.	32.437 sec.
p4.b	1.712 sec.	9.263 sec.
p4.c[c]	1.691 sec.	9.119 sec.
p4.d[d]	1.805 sec.	32.091 sec.
p6.a	0.388 sec.	> 1h
p6.b	0.157 sec.	160.049 sec.
p6.c[e]	0.163 sec.	156.359 sec.
p6.d[f]	0.399 sec.	>1h
p9.a	19.568 sec.	209.102 sec.
p9.b	50.953 sec.	207.176 sec.
p39.a	5.202 sec.	44.779 sec.
p39.b	3.787 sec.	44.933 sec.
p39.c[g]	3.762 sec.	46.510 sec.
p39.d[h]	5.139 sec.	45.412 sec.

Table 2:

name of problem	occnest	best std.-heuristic
lat1.a	0.062 sec.	1.966 sec.
lat1.b	0.060 sec.	2.019 sec.
lat3.a	0.078 sec.	3.258 sec.
lat3.b	0.074 sec.	3.379 sec.

a. hypotheses: u(a,b)=b, u(c,d)=d;
goal: l(f(a,c),f(b,d))≠f(a,c)
b. hypotheses: u(a,b)=b, l(c,d)=c;
goal: u(f(a,c),f(b,d))≠f(b,d)
c. hypotheses: u(1,a)=a, u(1,b)=b;
goal: l(1,f(a,b))≠1
d. hypotheses: l(1,a)=1, l(1,b)=1;
goal: u(1,f(a,b))≠f(a,b)
e. hypothesis: u(1,b)=b
goal: l(1,f(i(a),f(b,a)))≠1
f. hypothesis: l(1,b)=1;
goal: u(1,f(i(a),f(b,a)))≠f(i(a),f(b,a))
g. hypothesis: u(a,b)=a; goal: l(i(a),i(b))≠i(a)
h. hypothesis: l(a,b)=b; goal: u(i(a),i(b))≠i(b)

Before we go on we would like to analyze[1] two proofs in order to reveal the cause for the superiority of occnest (in these cases). We chose to investigate the proofs belonging to p6.a[2] and p6.b[3]. There are several reasons for this choice: First of all, the run times for both problems do not substantially differ if occnest is used as the selecting heuristic, whereas they do differ considerably when the best standard-heuristic (add) is employed. Furthermore, occnest performs significantly better than the (best) standard-heuristic. Besides, the proofs found by occnest are rather short (only eight rules and equations are needed, five of them stemming from the set of axioms Λ, so that merely three critical pairs have to be considered).

We could make the following observations: For the proof of p6.a the rule

(❋) u(f(x,y),f(x,f(b,y)))→f(x,f(b,y))

is crucial. According to add its weight[4] is 22. For the proof of p6.b the corresponding rule

(☆) l(f(x,y),f(x,f(b,y)))→f(x,y)

is required, its weight being 18 according to add. In both cases (especially for p6.a) there is a large number of critical pairs with a weight lesser than or equal[5] to the weight of (❋) and (☆), so that add selects a lot of critical pairs which yield redundant[6] rules and equa-

1. The analyses were supported by tools for proof-analysis and -processing which are available for the DISCOUNT-system ([Sch93]).
2. p6.a: additional “axiom” (antecedent of p6): u(1,b)=b; goal: u(1,f(i(a),f(b,a)))=f(i(a),f(b,a)
3. p6.b: additional “axiom” (antecedent of p6): l(1,b)=1; goal: l(1,f(i(a),f(b,a)))=1
4. Weights are usually associated with critical pairs. When talking about the weight of a rule we refer to the weight of the corresponding critical pair (i.e. we view the rule as a critical pair).
5. If the weights of critical pairs are equal then the order in which they were generated determines which one is selected before the other (usually FIFO).
6. Redundancy is to be seen w.r.t. the current proof.

tions before either (❋) or (☆) is selected. Among these rules are, for instance,

(a) u(l(x,l(y,z)),l(z,y))→l(z,y) (weight=17)
(b) l(u(x,u(y,z)),u(z,y))→u(z,y) (weight=17)
(c) u(l(x,1),l(b,x))→l(b,x) (weight=17)

occnest, which is based on add, increases selectivity[1] through the multiplication of the basic weight with multipliers expressing the amount by which the measures "occurrences" and "nesting" (cf. definitions 4.1.2 and 4.1.3) exceed the corresponding values given by the goal (see definition 4.1.4). These values are:

goal of p6.a	i	f	l	u	1	a	b
occurrences	1	2	0	1	1	2	1
nesting	1	2	0	1	0	0	0

goal of p6.b	i	f	l	u	1	a	b
occurrences	1	2	1	0	1	2	1
nesting	1	2	1	0	0	0	0

Both rule (❋) and rule (☆) exceed the occurrences of f in the respective goal by 1. So, occnest computes the weights of (❋) and (☆) to be 44 and 36, respectively. The measures of the rules (a), (b) and (c) cause (the product of) the multipliers (cp. definition 4.1.4) to be at least 3. Hence occnest associates a weight to each of these rules which is greater than or equal to 51. Consequently, occnest prefers (❋) and (☆) to (a), (b) and (c).

A further interesting observation is an obvious redundancy in the proof found for p6.b by add (also caused by add's inferior selectivity). Here, as well as in the case occnest is used, the rule l(x,f(b,x))→x is needed. Since add cannot "see" any difference between the rules

(❋) f(u(x,y),z)→u(f(x,z),f(y,z))
(✣) f(l(x,y),z)→l(f(x,z),f(y,z)),

and (❋) happens to occur in front of (✣) in the list of axioms Λ, add selects (❋) before (✣).With (❋) and rule u(b,1)→b (obtained by overlapping l(b,1)→1 into u(x,l(x,y))→x, where l(b,1)→1 stems from l(1,b)→1, reduced with l(x,y)=l(y,x)), we get u(x,f(b,x))→f(b,x), which yields l(x,f(b,x))→x after being overlapped into l(x,u(x,y))→x. occnest, however, makes a difference between (❋) and (✣) (no multipliers > 1 for (✣), m_u=4 for (❋)) and prefers (✣) to (❋). An overlap of l(b,1)→1 into (✣) immediately produces l(x,f(b,x))→x, thus avoiding the detour of add caused by conversions of the form u(x,y)=x ⇔ l(x,y)=y which cancel each other out.

Remark:

Lattice ordered groups are "good-natured" in the sense that (in most cases) no rules or

1. A fair heuristic with a bad selectivity (i.e. weighting almost every critical pair with the same weight) will in an extreme case degenerate into the FIFO-strategy for which empirical results have shown that it is not well suited for selecting critical pairs.

equations with occurrences- or nesting-values exceeding substantially the limits given by the goal are needed for a proof. This is the main prerequisite for the success of occnest and its (in general) superior performance w.r.t. standard-heuristics. In general, not every domain is so "good-natured" (cp. [MOS93]), but, on the other hand, lattice ordered groups are not the only domain with such a property (see appendix A).

5.2. Goal-oriented heuristics and teamwork

We now take a look at some proofs found when using the teamwork-method. In addition to p9 the following theorems were examined:

p2 : $i(y) \leq i(x) \rightarrow x \leq y$
p8 : $(1 \leq x \wedge 1 \leq y \wedge 1 \leq z) \rightarrow l(x,f(y,z)) \leq f(l(x,y),l(x,z))$
p10 : $i(u(x,z))=l(i(x),i(y))$

This time the goal-oriented heuristics occnest and Goal_in_CP (see definitions 4.1.4 and 4.2.2) are employed in a team (see section 3) together with add or variations of standard-heuristics. These variations are addR and maxR which correspond to add and max, respectively. In both cases critical pairs yielding equations (i.e. their sides cannot be compared w.r.t. the reduction ordering) are eluded as long as there are critical pairs resulting in rules[1]. Similar to the preceding tables, the subsequent table lists in columns one through three the (complete) name of the problem, the run time needed to find a proof and the members of the respective team. Apart from that, column four displays the best results obtained when a heuristic was working individually. Whenever a heuristic could find a proof alone, then its name and the respective run time are given, whereas no entry (i.e. '-') indicates that no heuristic could accomplish this.

problem	*run time*	*members of team*	*best heuristic alone*
p2.a	13.820 sec.	Goal_in_CP, addR	Goal_in_CP (79.516 sec.)[a]
p2.b	12.728 sec.	Goal_in_CP, addR	-
p2.a	5.413 sec.	occnest, addR	-
p2.b	5.381 sec.	occnest, addR	-
p8.b[b]	56.837 sec.	Goal_in_CP, maxR	-
p9.a	8.659 sec.	occnest, add	occnest (19.568 sec.)
p9.b	8.440 sec.	occnest, add	occnest (50.953 sec.)
p10	23.203 sec.	Goal_in_CP, maxR	-

a. For this proof the precedence l>u>i>f>1>a>b was used.
b. p8.a could -so far - not be proved by DISCOUNT in any way.

Before we discuss some limitations of goal-oriented heuristics, we shall concisely illustrate the reasons why the teamwork method could significantly decrease the time spent for find-

1. Of course, both addR and maxR are not fair, i.e. completeness of the UKB-procedure cannot be guaranteed if addR or maxR are used individually.

ing a proof (cf. p9.a and p9.b), and, what is even more interesting, why it succeeded where all heuristics failed when they were working individually.

Problems p9.a and p9.b could both be proved by occnest (see table 2). Those proofs could do without the rule (✲) f(x,1)→x. occnest does not generate this rule because intermediate rules necessary for its generation are given a relatively high weight. Contrarily, add derives (✲) quite fast and can supply occnest with (✲) during a team meeting. (Note that selected results are at any rate accepted without being assessed by the recipient.) Now, (✲) simplifies the proof considerably and thus causes occnest to succeed faster than it did without it.

It is almost entirely due to addR that proofs for p2.a and p2.b can be obtained, both when using occnest and when using Goal_in_CP as the complementary member of the team. This observation underlines the profitableness of unfair heuristics, which the teamwork method can employ without necessarily losing (overall) completeness (cf. [De93]). So, occnest or Goal_in_CP merely act as suppliers, providing addR with an equation and a rule, the latter generated with the help of an equation, both of which are not considered by addR (since there are lots of orientable critical pairs to choose from), but they are also essential. By avoiding equations, addR generates rules faster, even rules with a relatively high weight. When addR is used in a team, this restriction (which seriously jeopardizes the completeness of a UKB-procedure solely relying on addR) is (partly) compensated for by the fact that other heuristics (here occnest or Goal_in_CP) can supply addR with equations or rules created with the help of equations. Hence, the restriction is alleviated while still profiting from the benefit not having to consider equations (each equation basically corresponding to two rules).

For the proof of p10, circumstances are similar, though the other way round. This time, the equation eluding heuristic maxR plays the role of the supplier (of rules). Goal_in_CP is first driven by its non-goal-oriented component add (cp. definition 4.2.2). When it selects f(x,u(y,z))→u(f(x,y),f(x,z)) critical pairs are generated which host generalizations of a side of the goal as subterms (mainly i(u(x,y)), but also l(i(x),y), l(x,i(y)) etc.). Goal_in_CP then sort of neglects to select (✥) f(u(x,y),z)→u(f(x,z),f(y,z)), which is also needed, because it associates with it a higher weight than with those critical pairs containing subterms of the kind mentioned above. maxR, however, literally “injects” (✥) into the system of rules and equations held by Goal_in_CP during a team meeting, thus clearing the way for success. The referee responsible for selecting good results of maxR has little trouble identifying (✥) as a good result, especially not because there are no equations that can complicate its choice. Once again, a heuristic which is definitely unfair and has, as addR before, no chance to prove the goal by itself, plays the key role for the success of a team.

5.3. Limitations of our goal-oriented heuristics

It is in general the nature of heuristics to be extremely successful in one case while completely failing in another case. Even if the range of application is narrowed down, it is rarely possible to design a heuristic that will *always* outperform any other heuristic. This is particularly true for heuristics guiding the inference process of automatic deduction systems. The goal-oriented heuristics presented in this report are no exception. Apart from the fact that Goal_in_CP as well as CP_in_Goal become really useful only if there are critical pairs that bear enough resemblance to the goal in terms of matching subterms[1], we encountered also a few examples where the performance of the best standard-heuristic was significantly better than that of occnest:

1. We already discussed this issue in section 4.2.

lat2 : $(1 \le x \wedge 1 \le y) \rightarrow x \le f(x,y)$
p5 : $(x \le 1 \wedge i(x) \le 1) \rightarrow 1=x$

	lat2.a	lat2.b	p5.a	p5.b
occnest	22.847 sec.	20.971 sec.	171.049 sec.	172.785 sec.
best std.-heuristic[a]	2.505 sec.	2.649 sec.	2.047 sec.	2.060 sec.

a. The best standard-heuristic was in all four cases max.

The analysis of the proof for lat2.a (and lat2.b) reveals that occnest selects one vital rule, namely (❃) $f(i(x),f(x,y)) \rightarrow y$, which is needed to derive $f(x,1) \rightarrow 1$, very late due to its large weight 160 ($=10 \cdot 4 \cdot 4$). The inference of rule $l(x,f(x,b)) \rightarrow x$ concluding the proof merely consists in overlapping $l(b,1) \rightarrow 1$ into $f(x,l(y,z)) \rightarrow l(f(x,y),f(x,z))$ with subsequent reductions of the resulting critical pair with (mainly) $f(x,1) \rightarrow 1$. These reductions are delayed because of the late generation of (❃). This example points out a striking problem facing goal-oriented heuristics: If rules and equations are needed that do significantly differ from the goal (at least in the "eyes" of a goal-oriented heuristic), then the retarded generation of these rules and equations will slow down finding a proof considerably. This is why goal-oriented heuristics are even more valuable and powerful if they are used within a team: Other heuristics (preferably non-goal-oriented ones) may have much lesser trouble to generate those rules and equations, and goal-oriented heuristics can be supplied with them during a team meeting (cp. problems p9.a and p9.b, section 5.2).

The reason why occnest comes off badly when proving p5.a or p5.b is basically the lack of structure of the goal $1 \neq a$. A sufficient structure of the goal is crucial for the capability of a goal-oriented heuristic to make meaningful assessments of critical pairs. Therefore, small-sized goals are usually particularly inappropriate for being handled by goal-oriented heuristics. This observation is confirmed by a further example. Recall the theorems p2 and p39 which are the two implications resulting from the equivalence $i(x) \le i(y) \leftrightarrow y \le x$. While p39 ($y \le x \rightarrow i(x) \le i(y)$) was no challenge for occnest, p2 ($i(x) \le i(y) \rightarrow y \le x$) was out of reach for it. Once again, the low degree of structure of the goal of p2 (in particular the fact that the function symbol i does not occur in that goal) accounts for the failure of occnest[1].

But not only small-sized goals can cause trouble. For a similar reason, large goals can destroy the benefits of goal-oriented heuristics, too. While small goals make nearly every critical pair look like miles away from the goal, large goals make a lot of critical pairs look as if they were appropriate. In both cases, goal-oriented heuristics almost completely lose their eminent ability to narrow down the search. (p8 might be an example where the goal is or becomes (through rewriting) too large to enable a beneficial application of goal-oriented heuristics.)

Further problems arise if several goals have to be taken into account. (This becomes necessary if a reduction ordering is used which is not total on ground terms, or if proofs for existentially quantified propositions are sought, yielding goals containing variables after negation and skolemization. See [De93] for details.) Goal-oriented heuristics have to make their assessments considering all goals, since the concentration on just one goal cannot be justified. (It is undecidable which goal will help finding the proof and which will not.) But

1. Proofs for p2.a (p2.b) and p39.a (p39.b) can nonetheless be conducted in full correspondence, the only difference consisting in the use of the respective hypothesis.

this way, possibly different information has to be intertwined, usually entailing more negative than positive effects. The teamwork-method can help here, too. Since we can not only use several heuristics simultaneously, but - in this context - can also focus on distinct goals at the same time, the problems just sketched become less aggravating.

Remark:

The implementation of occnest we used for our experiments is only one possibility. We have not attempted to modify the realization of occnest, for instance by changing Ψ_1 or Ψ_2, or by extending ϕ so as to distinct function symbols, i.e. associating an individual value with each function symbol instead of the "global" value 2 (cf. definitions 4.1.1 and 4.1.4). It might be possible that *some* (but which?) configuration of these parameters can improve the *overall* performance of occnest. But it is highly probable that such modifications will only cause occnest to become better for some examples while deteriorating for others. So, the attempt to improve *overall* performance on the basis of adapting parameters is bound to be a "wild goose chase". Nevertheless, adapting occnest through the modification of its parameters for an appropriate subset of problems is an interesting issue, especially when considering to re-use it for "similar" problems (i.e. conducting "analogical reasoning" in the wider sense).

6. Conclusion

We have demonstrated the usefulness of goal-oriented heuristics in the context of equational theorem proving in the domain of lattice ordered groups using the unfailing Knuth-Bendix completion procedure. Goal-oriented heuristics proved to be especially beneficial when employed by the teamwork-method ([De93],[AD93]) which is a framework for distributing deduction. Goal-orientation was achieved through the comparison of "measures", namely occurrences and nesting of function symbols, and through the test for matching (sub-) terms. Although we have focused on lattice ordered groups as the environment for our experiments, we have also found other examples where these goal-oriented heuristics are profitable (see appendix A). The domain of lattice ordered groups was chosen because it provides a wide variety of problems ranging from (nearly) trivial to rather difficult. Furthermore, this was of practical interest to the members of the ILF-project at the Humboldt University, Berlin, as users of the DISCOUNT-system which is using the goal-oriented heuristics described in this report ([DGW93]).

The comparison of the run times of various proofs in the domain of lattice ordered groups documents the (in general) superior performance of goal-oriented heuristics compared to non-goal-oriented ones. The selection of non-goal-oriented heuristics comprised three very common heuristics which proved their usefulness in many cases, and hence have to be considered as serious contestants. Besides the advantages of (our) goal-oriented heuristics we also discussed their conceptual limitations and illuminated occasional weakness, thus putting goal-oriented heuristics in their true light, since they are nothing more, but also nothing less than heuristics being impressively advantageous in suitable domains.

Conceptual limitations of goal-oriented heuristics and resulting constrains for their (successful) application can be substantially defused if goal-oriented heuristics are employed within the teamwork approach. In this case all major disadvantages can be overcome through the cooperation with other heuristics, which compensate for weaknesses of goal-oriented heuristics, while still fully profiting from the prominent properties of goal-orientation.

Appendix A

A further domain where the selection strategy occnest could notch up some major successes is the prepositional logic axiomatized by the following set of axioms (cp. [Ta56]):

c(x,n(n(x))) = t
c(n(n(x)),x) = t
c(c(x,y),c(n(y),n(x))) = t
c(c(x,c(y,z)),c(y,c(x,z))) = t
c(c(x,c(y,z)),c(c(x,y),c(x,z))) = t
c(x,c(y,x)) = t
c(t,x) = x

The subsequent theorems (tautologies of propositional logic) were proved. The table summarizes the run times (averaging again at least five runs) when occnest resp. the best standard-heuristic was used.

pl1 : c(c(x,y),c(c(y,z),c(x,z))) = t
pl2 : c(c(n(x),x),x) = t
pl3 : c(x,c(n(x),y)) = t
pl10 : c(n(x),c(x,y)) = t
pl17a : c(x,c(n(y),c(n(x),z))) = t

	pl1	pl2	pl3	pl10	pl17a
occnest	1.234 s	8.063 s	13.899 s	13.758 s	57.484 s
best std.-heuristic	91.942 s	41.545 s	273.627 s	277.137 s	277.374 s

Notes:
- We used the LPO (lexicographic path ordering) as reduction ordering, with precedence c>n>t.
- Except c(t,x)=x, every critical pair has t as one of its sides. Therefore, add, max and gt behave exactly the same way.

Appendix B

It is shown that the equational axiomatization E for lattice ordered groups given in section 2 is correct, i.e. each equation can be derived from the initial axiomatization of lattice ordered groups represented by the definitions 2.1 through 2.4.

<u>Notational conventions</u>

- $x \le y_1,..,y_n$ and $x_1,..,x_n \le y$ are used as abbreviations for $x \le y_1,..,x \le y_n$ and $x_1 \le y,..,x_n \le y$, respectively.
- In the following we shall write $x \cdot y$ rather than f(x,y).
- $\leftrightarrow$ denotes the (logic) equivalence, $\rightarrow$ the (logic) implication.
- The inverse of x will be denoted x^{-1}.
- Apart from these deviations we shall stick to the notation used throughout section 2.

For the subsequent proofs we shall make use of corollary 2.1 which states:
<u>Corollary 2.1:</u> $\forall x,y:[x \le y \leftrightarrow u(x,y)=y \leftrightarrow l(x,y)=x]$
Once again it is pointed out that u(x,y) and l(x,y) stand for the least <u>u</u>pper bound resp. the greatest <u>l</u>ower bound of any pair $x,y \in G$. The representation of these two bounds by two functions is permissible because definition 2.3 guarantees their existence for any pair $x,y \in G$, and their uniqueness follows from definitions 2.3 and 2.2.b (antisymmetry of $\le$).

For the reader's convenience we repeat here the equational axiomatization E which was first given in section 2 in "old" and "new" notation:

(1)	$f(1,x)=x$	(1)	$1 \cdot x=x$
(2)	$f(i(x),x)=1$	(2)	$x^{-1} \cdot x=1$
(3)	$f(f(x,y),z)=f(x,f(y,z))$	(3)	$(x \cdot y) \cdot z=x \cdot (y \cdot z)$
(4)	$l(x,y)=l(y,x)$	(4)	$l(x,y)=l(y,x)$
(5)	$u(x,y)=u(y,x)$	(5)	$u(x,y)=u(y,x)$
(6)	$l(l(x,y),z)=l(x,l(y,z))$	(6)	$l(l(x,y),z)=l(x,l(y,z))$
(7)	$u(u(x,y),z)=u(x,u(y,z))$	(7)	$u(u(x,y),z)=u(x,u(y,z))$
(8)	$l(x,x)=x$	(8)	$l(x,x)=x$
(9)	$u(x,x)=x$	(9)	$u(x,x)=x$
(10)	$u(x,l(x,y))=x$	(10)	$u(x,l(x,y))=x$
(11)	$l(x,u(x,y))=x$	(11)	$l(x,u(x,y))=x$
(12)	$f(x,l(y,z))=l(f(x,y),f(x,z))$	(12)	$x \cdot l(y,z)=l(x \cdot y,x \cdot z)$
(13)	$f(l(x,y),z)=l(f(x,z),f(y,z))$	(13)	$l(x,y) \cdot z=l(x \cdot z,y \cdot z)$
(14)	$f(x,u(y,z))=u(f(x,y),f(x,z))$	(14)	$x \cdot u(y,z)=u(x \cdot y,x \cdot z)$
(15)	$f(u(x,y),z)=u(f(x,z),f(y,z))$	(15)	$u(x,y) \cdot z=u(x \cdot z,y \cdot z)$

<u>Theorem</u>:
The equations (1) through (15) follow from the definitions 2.1 through 2.4.

<u>Proofs</u>:

(1),(2),(3): These equations are straight-forward consequences of definition 2.1, employing skolemization.

(4),(5): The commutativity of u and l trivially follows from definition 2.3.

(6) Associativity of l; $\forall x,y,z$: $l(l(x,y),z)=l(x,l(y,z))$
We have $l(x,y)\leq x,y$, $l(l(x,y),z)\leq l(x,y),z$, hence $l(l(x,y),z)\leq x,y,z$ (using transitivity of $\leq$). Similarly, we obtain $l(x,l(y,z))\leq x,y,z$. Since $l(l(x,y),z)\leq y,z$, we have $l(l(x,y),z)\leq l(y,z)$ (definition 2.3). With $l(l(x,y),z)\leq x,l(y,z)$ and definition 2.3 we get $l(l(x,y),z)\leq l(x,l(y,z))$. Furthermore, we obtain $l(x,l(y,z))\leq l(x,y)$ because of $l(x,l(y,z))\leq x,y$ and definition 2.3. Using again definition 2.3 and $l(x,l(y,z))\leq l(x,y),z$ yields $l(x,l(y,z))\leq l(l(x,y),z)$. The antisymmetry of $\leq$ completes the proof. ❐

(7) Associativity of u; $\forall x,y,z$: $u(u(x,y),z)=u(x,u(y,z))$
Completely analogous to proof of (6). (Substitute u for l and swap sides of $\leq$.)

(8) $\forall x$: $u(x,x)=x$
Reflexivity $x\leq x$ and corollary 2.1 conclude the proof.

(9) $\forall x$: $l(x,x)=x$ is also proved by reflexivity and corollary 2.1.

(10) $\forall x,y$: $u(x,l(x,y))=x$
Definition 2.3 asserts $l(x,y)\leq x$. Hence $u(x,l(x,y))=x$ with corollary 2.1.

(11) $\forall x,y$: $l(x,u(x,y))=x$
Definition 2.3 asserts $x\leq u(x,y)$. Hence $l(x,u(x,y))=x$ with corollary 2.1.

(12) $\forall x,y,z$: $x\cdot u(y,z)=u(x\cdot y,x\cdot z)$
We have

$y,z\leq u(y,z)$	(definition 2.3)
$\Rightarrow$ $x\cdot y,x\cdot z\leq x\cdot u(y,z)$	(monotonicity)
$\Rightarrow$ $u(x\cdot y,x\cdot z)\leq x\cdot u(y,z)$	(definition 2.3)

Furthermore,

$x\cdot y,x\cdot z\leq u(x\cdot y,x\cdot z)$	(definition 2.3)
$\Rightarrow$ $y,z\leq x^{-1}\cdot u(x\cdot y,x\cdot z)$	(monotonicity)
$\Rightarrow$ $u(y,z)\leq x^{-1}\cdot u(x\cdot y,x\cdot z)$	(definition 2.3)
$\Rightarrow$ $x\cdot u(y,z)\leq u(x\cdot y,x\cdot z)$	(monotonicity)

The antisymmetry of $\leq$ concludes the proof. ❐

(13),(14),(15): These proofs are analogous to the proof of (12).

References

[AA90] Anantharaman, S.; Andrianarievelo, N.:
"*Heuristical criteria in refutational theorem proving*"
Proc. DISCO '90, LNCS 429, 1990, pp. 184-193

[AD93] Avenhaus, J.; Denzinger, J.:
"*Distributing equational theorem proving*"
Proc. RTA '93, LNCS 690, 1993, pp. 62-76

[BDP89] Bachmair, L.; Dershowitz, N.; Plaisted, D.A.:
"*Completion without failure*"
Coll. on the resolution of equations in algebraic structures, Austin 1987
Academic Press 1989

[De87] Dershowitz, N.:
"*Termination*"
Journal of symbolic computation 3 (1987), pp. 69-116

[De93] Denzinger, J.:
"*Teamwork: Eine Methode zum Entwurf verteilter, wissensbasierter Theorembeweiser*"
Dissertation, FB Informatik, Universität Kaiserslautern, 1993.

[DGW93] Dahn, B.I.; Gehne, J.; Wolf, A.:
private communication, 1993

[DP92] Denzinger, J.; Pitz, W.:
"*Das DISCOUNT-System: Benutzerhandbuch*"
SEKI working paper SWP-92-16

[KB70] Knuth, D.E.; Bendix, P.B.:
"*Simple word problems in universal algebra*"
Computational algebra, J. Leach, Pergamon Press 1970, pp. 263-297

[KK74] Kokorin, A.I.; Kopytov, V.M.:
"*Fully ordered groups*"
Halsted Press, 1974

[MOS93] Madlener, K.; Otto, F.; Sattler-Klein, A.:
"*On the problem of generating small convergent systems*"
Journal of symbolic computation 16 (1993), pp. 167-187

[Pi92] Pitz, W.:
"*Realisierung eines Systems zum verteilten, wissensbasierten Gleichheitsbeweisen mit Hilfe der Teamwork-Methode*"
Diplomarbeit, FB Informatik, Universität Kaiserslautern, 1992

[Sch93] Schulz, S.:
"*Analyse und Transformation von Gleichheitsbeweisen*"
Projektarbeit, FB Informatik, Universität Kaiserslautern, 1993.

[Ta56] Tarski, A.:
"*Logic, semantics, metamathematics*"
Oxford University Press, 1956