On Resolution with Short Clauses

Michael Buro and Hans Kleine Bning
Universitt Paderborn
FB 17 Mathematik–Informatik
Warburger Strae 100
33098 Paderborn
Germany

Abstract

We investigate properties of k–resolution, a restricted version of resolution in which one parent clause must have length at most k. Starting from a unit–preference strategy, we compare minimal proof lengths of unit–resolution and unrestricted resolution. In particular, we show that the speed-up by using resolution is bound by $\sqrt{t}$ if the shortest unit–resolution refutation needs t steps. Next we present an algorithm which decides whether the empty clause can be deduced by 2–resolution from a formula Φ and has time complexity $O(\text{length}(\Phi)^4)$. Finally we describe effects on k–resolution if a formula is transformed into t–CNF and show that extended 3–resolution is complete and sound.

1 Introduction

We are interested in the efficiency of one of the resolution strategies, namely the unit-preference strategy [7]. In this strategy it is demanded that whenever a new unit-clause has been generated the unit-clause has to be resolved first. A more sophisticated version of this strategy is based on the idea not only to resolve the unit-clauses as soon as possible, but to resolve short clauses first. This establishes a priority which says that one of the parent clauses has to be one of the shortest clauses. In other terms we can describe the strategy by a resolution restriction called k–resolution. In a k–resolution step one of the parent clauses must be a clause of at most k literals. Then the modified unit-preference strategy can be considered as a sequence of applications of unit–resolution, 2–resolution, 3–resolution and so on.

We will consider the pros and cons of this strategy for propositional formulas in conjunctive normal form (CNF). For this reason we investigate what happens with the length of shortest refutations comparing unit–resolution and unrestricted resolution. Another problem arises when applying the well known transformation algorithm to transform a formula in CNF into a formula in t-CNF where each clause consists of at most t literals. We discuss under what circumstances such transformations should be applied to a formula. Furthermore we state that extended 3–resolution is complete and sound.

So far the power of unit–resolution is well understood. Roughly speaking, only the satisfiability of Horn formulas up to renaming can be solved efficiently. An algorithm running in linear time in the length of the formula exists which decides whether for a formula Φ in CNF a unit–resolution refutation exists. For 2–resolution we present an algorithm which decides the question above in time $O(\text{length}(\Phi)^4)$.

2 Preliminaries

A literal L is either an atom A or its negation $\neg A$. A clause is a disjunction of literals and a t-clause consists of at most t literals. A unit clause is a 1-clause. A formula is in conjunctive normal form (CNF) if it is a conjunction of clauses and is in t-CNF if it consists of t-clauses. Clauses and formulas can be written as sets of literals resp. sets of clauses, e.g. $\sqcup$ and $\emptyset$ denote the empty clause. For a formula Φ atomset(Φ) consists of all atoms in Φ and literalset(Φ) is the set of all literals over the atoms of Φ. The length of Φ is the sum of the cardinalities of its clauses. The necessary terms regarding k–resolution are introduced in the following

Definition: (Resolution, k–Resolution)
Let Φ be a formula in CNF and c a clause. We say c can be deduced from Φ in one resolution step ($\Phi \mid\!\frac{1}{Res}\; c$) if and only if there are clauses c' and c'' in Φ and a literal L with $L \in c'$, $\neg L \in c''$ and $c = (c' \setminus \{L\}) \cup (c'' \setminus \{\neg L\})$. A k–resolution step is a resolution step where one parent clause consists of at most k literals. A clause c is deducible from Φ in $r+1$ steps if and only if there are two parent clauses of c which are deducible from Φ exactly in r steps resp. in at most r steps.

We call c deducible from Φ iff there is an $r \geq 0$ with $\Phi \mid\!\frac{r}{Res}\; c$ and denote this property by $\Phi \mid\!\frac{}{Res}\; c$ ($\Phi \mid\!\frac{0}{Res}\; c$ stands for $c \in \Phi$). k-Res$^r(\Phi)$ is the set of clauses which can be deduced in at most r steps from Φ by means of k–resolution and k-Res(Φ) is defined as the set of all deducible clauses from Φ via k–resolution. Finally, the length of a refutation is defined as the number of resolution steps performed.

That k–resolution is not complete for formulas in CNF can be seen by choosing an unsatisfiable formula which consists only of clauses of length greater than k. But k–resolution suffices to decide the satisfiability of formulas in the class H_k which have been introduced in [3],[8]. Furthermore the H_k build a hierarchy of formulas in CNF.
A simple version of a modified unit-preference strategy can be described as follows:

```
Input: $\Phi$ in CNF    Output: 'yes' if $\Phi$ is satisfiable and 'no' otherwise

$k := 0$
while $\sqcup \notin \Phi$ or $k < |atomset(\Phi)|$ do
    $k := k + 1$
    $\Phi := k$-Res($\Phi$)        {k-th level}
endwhile
if $\sqcup \in \Phi$ then write('no') else write('yes') endif
```

Obviously, the algorithm can be refined, i.e. write 'no' if the empty clause $\sqcup$ occurs in the computation of k-Res(Φ) and organize the computation of k-Res(Φ) again as an unit-preference algorithm.

Thus whether a refutation within the k-th level can be reached can be considered as the question whether there is a refutation when in each step one of the parent clauses has length less or equal then k.

3 Short Proofs

In this section we will compare the length of shortest unit–resolution refutations and the shortest resolution refutations. We show that the length of a shortest unit–resolution proof for minimal unsatisfiable formulas can be $\Theta(n^2)$ whereas $\Theta(n)$ is the length of the shortest resolution refutation. Additionally we prove that this is the best speed-up we can achieve.

Theorem 3.1 *For each $n \geq 1$ there is a formula Φ_n with the following properties:*

(1) Φ_n is minimal unsatisfiable

(2) length(Φ_n) $= n^2 + 4n + 1$

(3) the shortest unit–resolution refutation needs $n^2 + 2n$ steps

(4) the shortest resolution refutation needs $2n + 1$ steps

Proof: For each n we define the formula

$$\begin{aligned}\Phi_n \;:=\; & \Big\{ (B_1 \vee \ldots \vee B_n) \,,\, (D) \Big\} \cup \Big\{ (\neg B_i \vee A_1 \vee \ldots \vee A_n) \mid 1 \leq i \leq n \Big\} \\ \cup \; & \Big\{ (\neg A_i \vee \neg D) \mid 1 \leq i \leq n \Big\}\end{aligned}$$

Φ_n is unsatisfiable and removing one clause yields a satisfiable formula, i.e. Φ_n is minimal. Its length is $n^2 + 4n + 1$ and the shortest unit–resolution refutation needs $n^2 + 2n$ steps. Applying $(\neg B_i \vee A_1 \vee \ldots \vee A_n)$ for $1 \leq i \leq n$ to $(B_1 \vee \ldots \vee B_n)$ we obtain in n steps the clause $(A_1 \vee \ldots \vee A_n)$ and then in $n+1$ steps the empty clause. Hence there is a resolution refutation of Φ_n in $2n + 1$ steps.

■

A formula Φ is called minimal for unit–resolution if $\Phi \vdash_{\overline{1-Res}} \sqcup$ and for each proper subformula $\Phi^* \subset \Phi$ there exists no unit–resolution refutation. For a formula Φ a unit–resolution refutation exists if and only if there is a subformula $\Phi^* \subset \Phi$, such that $\Phi^* \vdash_{\overline{1-Res}} \sqcup$ and Φ^* is a Horn formula up to a renaming, which replaces some atoms by their complements and vice versa. Therefore minimal for unit–resolution means $\Phi \vdash_{\overline{1-Res}} \sqcup$ and each proper subformula $\Phi^* \subset \Phi$ is satisfiable. This can be seen as follows: If $\Phi \vdash_{\overline{1-Res}} \sqcup$ then we choose some minimal subformula $\Phi^* \subset \Phi$, for which we achieve $\Phi^* \vdash_{\overline{1-Res}} \sqcup$. By an induction on the cardinality of atomset(Φ^*) one can prove that a renaming into a Horn formula exists.

Theorem 3.2 *Let Φ be an unsatisfiable formula which is minimal for unit–resolution, $\Phi \vdash_{\overline{1-Res}} \sqcup$ and the shortest unit–resolution refutation needs t steps. Then the shortest resolution refutation requires at least $\sqrt{t}$ resolution steps.*

Proof: (By induction on the length of Φ)
For $\Phi = \{(A), (\neg A)\}$ the shortest unit–resolution and resolution refutation needs one step. Now we suppose length(Φ) $= t+1$. Since Φ is minimal for unit–resolution, the formula Φ is a Horn formula up to renaming and Φ contains at least one unit clause (L). We apply (L) to each clause $(\neg L \vee \ldots)$ in Φ and remove the clause (L). The resulting formula is denoted by $\Phi[L = 1]$. The formula $\Phi[L = 1]$ is minimal for unit–resolution and $\Phi[L = 1] \mid\!\underline{\ \ }_{1-Res}\ \sqcup$ follows. By induction hypothesis we know if the shortest unit–resolution refutation has length r then the shortest resolution refutation needs at least $\sqrt{r}$ steps for $\Phi[L = 1]$. Let q_1 be the length of the shortest resolution refutation for $\Phi[L = 1]$.
Assume that the literal $\neg L$ occurs in s clauses of Φ. Then the shortest unit–resolution refutation needs exactly $r + s$ steps. Furthermore the shortest resolution refutation needs at least $q_1 + 1$ steps, because of the minimality of Φ. Note that Φ is a Horn formula up to renaming. Let q be the length of the shortest resolution refutation for Φ, then we obtain

$$q^2 \geq (q_1 + 1)^2 \geq q_1^2 + 2q_1 + 1 \geq r + 2q_1 + 1 \geq r + s.$$

The last inequality holds, because the number of resolution steps q_1 must be greater or equal to s. This follows directly, because $\Phi[L = 1]$ is a minimal Horn-formula up to renaming.
Hence we have proved our desired result, that the speed-up we might obtain by applying resolution instead of unit–resolution to the formulas is bound by the square root of the length of the shortest unit–resolution refutation.

■

For $k \geq 2$ the same speed-up can be shown for k–resolution using the formulas Φ_n of theorem 3.1, but no non-trivial upper bound is known.

4 A 2-Resolution Algorithm

In this section we present an algorithm which decides whether the empty clause is deducible from a given formula in CNF by 2–resolution. We show the correctness of the algorithm and then we analyze an efficient implementation of it which needs $O(\text{length}(\Phi)^4)$ running time for a formula Φ in CNF.

The key idea of the algorithm is to avoid the generation of any clauses with length greater than two. In a simple approach one could produce all deducible clauses and could check whether the empty clause has been deduced. Unfortunately there are simple formulas from which one can not deduce the empty clause by 2–resolution but a set of clauses with a cardinality superpolynomial in the length of the formula. Therefore we want to generate only 2-clauses and look whether we can deduce a pair of complementary units. Two cases of a 2–resolution step can be distinguished: If we use an unit $\{L\}$ then we can cancel the literal $\neg L$ in another clause. In the other case we can replace one literal by another one. The algorithm generates 2-clauses in this way one by one and looks if there are new 2–resolvents in applying the new 2-clauses to the previously generated 2-clauses and to the clauses in Φ with length greater than two.

Algorithm 2-Res(I)

Input: Φ in CNF

Output: 'yes' if $\Phi \vdash_{2-Res} \emptyset$ and 'no' otherwise

```
(1)  $S := \{c \mid c \in \Phi \text{ and } 1 \le |c| \le 2\} \cup \Big\{ \{A, \neg A\} \mid A \in \text{atomset}(\Phi) \Big\}$

(2)  $T := S;\ \Phi' := \Phi \setminus S;\ R := \emptyset$        { $T$ is the set of derived 2-clauses }
                                           { $R$ collects clauses from $T$ one by one }
(3)  while $S \neq \emptyset$ do
(4)      $S' := \emptyset$
(5)      forall $c \in S$ do $new(c)$ endfor   { update $R$, put new 2-clauses into $S'$ }
(6)      $S := S'$
(7)  endwhile                              { until there are no more new 2-clauses}

(8)  if $\exists A \in \text{atomset}(\Phi) : \{A\}, \{\neg A\} \in R$ then write('yes') else write('no') endif

                         procedure $new(c)$
(9)  $R := R \cup \{c\};\ D := \emptyset$    { $D$ collects 2–clauses which arise from applying $c$}

(10) if $c = \{L\}$ then $D := \Big\{ \{L, L'\} \mid L' \in \text{literalset}(\Phi) \setminus \{L\} \Big\}$ endif

(11) $D := D \cup \Big\{ c' \mid 1 \le |c'| \le 2 \text{ and } \exists d = \{L_1, \ldots, L_n\} \in \Phi' : \Big( c' = \bigcup_{i=1}^{n} c_i \text{ and}$
       $\forall i : \Big( c_i = \{L_i\} \text{ or } \Big[ c_i = \emptyset \text{ and } \{\neg L_i\} \in R \Big] \text{ or } \Big[ c_i = \{L'_i\} \text{ and } \{L'_i, \neg L_i\} \in R \Big] \Big) \Big) \Big\}$

(12)          $\cup \Big\{ c' \mid 1 \le |c'| \le 2 \text{ and } \exists d \in R : \{c, d\} \vdash^{1}_{2-Res} c' \Big\}$

(13) $S' := S' \cup (D \setminus T);\ T := T \cup S'$   { generate only clauses which are not in $T$ }
```

Theorem 4.1 *The algorithm 2-Res(I) terminates for all inputs Φ in CNF and writes 'yes' if $\Phi \vdash_{2-Res} \emptyset$ and 'no' otherwise.*

To prove the correctness of the algorithms we need a

Lemma 4.2

(i) $\Phi \vdash_{2-Res} \emptyset$ *if and only if* $\Phi \cup \Big\{ \{A, \neg A\} \mid A \in \textit{atomset}(\Phi) \Big\} \vdash_{2-Res} \emptyset$

(ii) If $\Phi \vdash_{2-Res} \{L\}$ *then*

$$\Big[\Phi \vdash_{2-Res} \emptyset \iff \Phi \cup \Big\{ \{L, L'\} \mid L' \in \textit{literalset}(\Phi) \setminus \{L\} \Big\} \vdash_{2-Res} \emptyset \Big]$$

Proof:

(i) "$\Rightarrow$" This follows immediately because the formula on the right side contains all clauses of the left side.
"$\Leftarrow$" If there is a resolution step like $\big\{c, \{A, \neg A\}\big\} \mid\!\frac{1}{2-Res}\ d$ in the refutation of the formula on the right then $c = d$ follows from the definition of $\mid\!\frac{1}{2-Res}$. Thus all such steps can be canceled to get the desired refutation of Φ.

(ii) "$\Rightarrow$" Again the formula on the right side contains all clauses of the left side.
"$\Leftarrow$" (Sketch) The statement follows from a more general one:

If $\Phi \vdash_{2-Res} \{L\}$ then

$$\Big[\ \forall k \geq 0\ \forall c \in \text{2-Res}^k\Big(\ \Phi\ \cup\ \big\{\{L, L'\} \mid L' \in \text{literalset}(\Phi) \setminus \{L\}\big\}\ \Big)$$
$$\exists d \in \text{2-Res}(\Phi) : d \subset c\ \Big]$$

which can be proved by induction on k.

$\blacksquare$

Proof of theorem: T is a set of 2-clauses. Hence we have $|T| \leq |\text{literalset}(\Phi)|^2 < \infty$. If S' is not empty after execution of line (5) then T has been enlarged by this step. Thus the algorithm terminates.
In what follows, R_i denotes the set R after the i-th call of *new* and R_n the set R after termination with input Φ. We have to show the following equivalence:

$$\Big(\exists A \in \text{atomset}(\Phi) : \{A\}, \{\neg A\} \in R_n\Big) \quad \Longleftrightarrow \quad \Phi \vdash_{2-Res} \emptyset$$

"$\Rightarrow$" We prove $\forall i \leq n : \Big(\ \Phi \cup R_i \vdash_{2-Res} \emptyset\ \Rightarrow\ \Phi \vdash_{2-Res} \emptyset\ \Big)$ using induction on i and conclude the stated implication from this. Since if there is an A with $\{A\}, \{\neg A\} \in R_n$ then $\Phi \cup R_n \vdash_{2-Res} \emptyset$ holds and hereby $\Phi \vdash_{2-Res} \emptyset$.
According to the setting of S in line (1) we see that for $c \in R_1$ ($c \in \Phi$ or $c = \{A, \neg A\}$) holds. Hence the induction base can be proved by lemma 4.2 (i). For the induction step ($\Phi \cup R_{i+1} \vdash_{2-Res} \emptyset \ \Rightarrow\ \Phi \vdash_{2-Res} \emptyset$) have to be shown. Let *new*(e) be the $(i+1)$-st call of procedure *new*. e could have been generated in different ways. In the case ($e \in \Phi$ or $e = \{A, \neg A\}$) again lemma 4.2 (i) can be used. Otherwise e has been generated by a previous call of *new*. If we look at the lines (10)-(12) we see that only clauses can be produced which are valid according to lemma 4.2 (ii) resp. which are 2–resolution deductions. Thus the use of the induction hypothesis completes this part of the proof.

"$\Leftarrow$" Now we want to show that the following statement holds:

$$\forall k \geq 0\ \forall c \in \text{2-Res}^k(\Phi) \setminus \{\emptyset\}\ \exists i :$$

$$\Big[\ \Big(\ 1 \leq |c| \leq 2 \text{ and } c \in R_i\ \Big) \ \text{ or } \ \Big(\ |c| > 2 \text{ and } \exists d = \{L_1, \ldots, L_n\} \in \Phi' : c = \bigcup_{j=1}^{n} c_j \ \text{ and}$$

$\forall j : \Big(c_j = \{L_j\}$ or $\Big[c_j = \emptyset$ and $\{\neg L_j\} \in R_i \Big]$ or $\Big[c_j = \{L'_j\}$ and $\{L'_j, \neg L_j\} \in R_i \Big] \Big) \Big) \Big]$

This formula says that we can find any resolvent $c \neq \emptyset$ of Φ either in a set R_i or we can deduce it by means of 2–resolution from a clause d in Φ', where there is at most one resolution step for each literal in d.
The proof is by induction on k. If the statement is true then we may conclude

$$\Phi \vdash_{\overline{2-Res}} \emptyset \;\Rightarrow\; \exists A : \Big[\Phi \vdash_{\overline{2-Res}} \{A\} \text{ and } \Phi \vdash_{\overline{2-Res}} \{\neg A\} \Big] \;\Rightarrow$$

$$\exists A\ \exists i : \{A\}, \{\neg A\} \in R_i \;\Rightarrow\; \exists A : \{A\}, \{\neg A\} \in R_n$$

and the necessity is shown.
For the proof let k be 0 and $c \in$ 2-Res$^0(\Phi) \setminus \{\emptyset\}$. Then $c \in \Phi$ follows and according to line (1) there is an i with $c \in R_i$ in the case $|c| \leq 2$. If $|c| > 2$ we set $d = c \in \Phi'$ and $c_j = \{L_j\}$ for all j.
We now assume that the statement is true for all values less or equal to k. Let c be an element of 2-Res$^{k+1}(\Phi)$. There exist clauses $c_1 \in$ 2-Res$^k(\Phi)$, $c_2 \in$ 2-Res$^{\leq k}(\Phi)$ with the property $\{c_1, c_2\} \vdash^{1}_{\overline{2-Res}} c$. Hence the induction hypothesis can be applied. In the case $|c_1|, |c_2| \leq 2$ there are minimal numbers i_1 and i_2 with $c_1 \in R_{i_1}$, $c_2 \in R_{i_2}$ and without loss of generality $i_1 < i_2$. It follows that c will be generated during the i_2-th call of *new* according to line (12). Therefore we have an i with $c \in R_i$.
In the remaining cases analogous arguments can be used. We have to distinguish whether a literal $L_j \in d$ is not touched by 2–resolution ($c_j = \{L_j\}$), or has been canceled by using the unit $\{\neg L_j\}$ ($c_j = \emptyset$), or has been replaced by another literal L'_j ($c_j = \{L'_j\}$).
■

Next we present an efficient implementation of algorithm 2-Res(I). Technical difficulties have arised from the addition of tautologies (line (1)) and clauses $\{L, L'\}$ for deduced units $\{L\}$ (line (10)). Even without them the algorithm is correct but the following implementation profits from these extensions.

Algorithm 2-Res(II)

Input: Φ in CNF

Output: 'yes' if $\Phi \vdash_{\overline{2-Res}} \emptyset$ and 'no' otherwise

```
(1)  $S := \{c \mid c \in \Phi$ and $1 \leq |c| \leq 2\} \cup \big\{\{A, \neg A\} \mid A \in \text{atomset}(\Phi)\big\}$
(2)  $T := S$;  $\Phi' := \Phi \setminus S$;
(3)  forall $L \in \text{literalset}(\Phi)$ do $V(L) := \emptyset$ endfor      $\{ R := \emptyset \}$

(4)  while $S \neq \emptyset$ do
(5)     $S' := \emptyset$
(6)     forall $c \in S$ do
```

```
(7)             if $c = \{L\}$ then $new(L, L)$
(8)             elseif $c = \{L_1, L_2\}$ then $new(L_1, L_2)$
(9)             endif
(10)    endfor
(11)    $S := S'$
(12) endwhile

(13) if $\exists A : \Big[\; A \in V(\neg A)$ and $\neg A \in V(A) \;\Big]$ then write('yes') else write('no') endif
```

<u>procedure *add*(c)</u>

```
(14) if $c \notin T$ then $S' := S' \cup \{c\}$; $T := T \cup \{c\}$ endif
```

<u>procedure *new*(L_1, L_2)</u>

```
(15) $V(L_2) := V(L_2) \cup \{\neg L_1\}$; $V(L_1) := V(L_1) \cup \{\neg L_2\}$
(16) if $L_1 = L_2$ then
(17)    forall $L \in$ literalset($\Phi$) do $add(\{L\} \cup \{L_1\})$ endfor
(18) endif

(19) forall $L \in$ literalset($\Phi$) do
(20)        forall $d \in \Phi'$ do
(21)                if $d \subset V(L_1) \cup V(L)$ then $add(\{L_1\} \cup \{L\})$
(22)                elseif $d \subset V(L_2) \cup V(L)$ then $add(\{L_2\} \cup \{L\})$
(23)                endif
(24)        endfor
(25) endfor

(26) forall $L \in$ literalset($\Phi$) do
(27)        if $L_1 \in V(L)$ then
(28)          if $L = \neg L_1$ then $add(\{L_2\})$ else $add(\{L\} \cup \{L_2\})$ endif
(29)        endif
(30)        if $L_2 \in V(L)$ then
(31)          if $L = \neg L_2$ then $add(\{L_1\})$ else $add(\{L\} \cup \{L_1\})$ endif
(32)        endif
(33) endfor
```

To get a fast implementation of line (11) of the first algorithm the set R is represented by sets $V(L)$ according to $\neg L \in V(L') \;:\Longleftrightarrow\; \{L\} \cup \{L'\} \in R$. By this, the new clauses can be determined using unions of V sets because the clause $\{L\} \cup \{L'\}$ will be generated from d in Φ' if and only if $d \subset V(L) \cup V(L')$. In this situation we take advantage of the extensions. These guarantee $L \in V(L)$ for all literals L and $L' \in V(\neg L)$ for all deduced units $\{L\}$ and all L'. Therefore, all resolvents can be determined easily. Suppose now that *new* is called with L_1 and L_2 as parameters. Of course, not all clauses $\{L\} \cup \{L'\}$

have to be examined to check whether there is a new covering of a clause in Φ' since only the sets $V(L_1)$ and $V(L_2)$ have been altered. It suffices to test the clauses $\{L_1\} \cup \{L\}$ and $\{L_2\} \cup \{L\}$ for all L in literalset(Φ) which is done at lines (19)-(25). Finally, all possible resolvents of $\{L_1\} \cup \{L_2\}$ with previously generated 2-clauses are determined at lines (26)-(33).

Procedure *new* will be called at most $|\text{literalset}(\Phi)|^2$ times and has time complexity $O(|\text{literalset}(\Phi)|\cdot\text{length}(\Phi))$ provided the sets are implemented as bit arrays. Note that the time complexity of the for loop at lines (20)...(24) is only linear in the length of Φ' because for every literal of clauses in Φ' we have to check whether it appears in the unions and this can be done in constant time. Thus the time requirement for the whole algorithm is $O(|\text{literalset}(\Phi)|^3 \cdot \text{length}(\Phi))$ which is $O(\text{length}(\Phi)|^4)$. These observations lead to

Theorem 4.3 *Algorithm 2-Res(II) decides in time $O(\mathit{length}(\Phi)^4)$ whether the empty clause is deducible from a given formula Φ in CNF by 2–resolution.*

For unit–resolution an upper bound linear in the length of the formula is known. This can be seen applying unit–resolution step by step adepting the algorithm introduced in [1]. The gap between the time complexities gives room for future research.

5 Transformation

It is well-known that for each $t \geq 3$ a formula in CNF can be transferred to a formula in t–CNF in polynomial time, such that both formulas are equivalent with respect to satisfiability. In this section we shall discuss the effects of these transformations to the existence of k–resolution refutations.

For $t \geq 3$ the transformation algorithm *Trans-t* applied to a formula Φ in CNF replaces each clause $\pi = (L_1 \vee \ldots \vee L_m)$ for $m > t$ by the clauses $\pi_0, \ldots, \pi_{r+1}$, where

$$\begin{array}{lcl}
m & = & (t-1) + r(t-2) + s, \\
s & < & t, \\
\pi_0 & = & (L_1 \vee \ldots \vee L_{t-1} \vee P_1), \\
\pi_i & = & (\neg P_i \vee L_{t-1+(i-1)(t-2)+1} \vee \ldots \vee L_{t-1+i(t-2)} \vee P_{i+1}) \ \text{ for } \ 1 \leq i \leq r \ \text{ and} \\
\pi_{r+1} & = & (\neg P_{r+1} \vee L_{t-1+r(t-2)+1} \vee \ldots \vee L_m)
\end{array}$$

($P_1, \ldots, P_{r+1}$ are new atoms)
The resulting formula is written Φ^t and equivalent to Φ with respect to satisfiabilty.

Theorem 5.1

(i) For each $t \geq 3$ and Φ in CNF: $\Phi \vdash_{1-Res} \sqcup$ if and only if $\Phi^t \vdash_{1-Res} \sqcup$.

(ii) For each $k \geq 2$ there is some Φ such that $\Phi \vdash_{k-Res} \sqcup$ but not $\Phi^{k+1} \vdash_{k-Res} \sqcup$.

Proof: *Ad (i):* Since in each unit–resolution step a parent clause with more than one literal can be replaced by the resolvent and the unit–clauses of both Φ and Φ^t are the same, we immediately obtain (i).
Ad (ii): The construction of the formulas Φ is motivated by the following observation. Let be given the formula

$$\varphi = \{(x_0 \vee x_1 \vee y_0 \vee y_1), (\neg x_0 \vee A_0), (\neg x_1 \vee A_1), (\neg y_0 \vee A_0), (\neg y_1 \vee A_1)\}.$$

Then there is a 2–resolution deduction

$$(x_0 \vee x_1 \vee y_0 \vee y_1), (\neg x_0 \vee A_0) \vdash_{\overline{2-Res}} (A_0 \vee x_1 \vee y_0 \vee y_1), (\neg x_1 \vee A_1) \vdash_{\overline{2-Res}}$$

$$(A_0 \vee A_1 \vee y_0 \vee y_1), (\neg y_0 \vee A_0) \vdash_{\overline{2-Res}} (A_0 \vee A_1 \vee y_1)$$

(Note that there is only one A_0 in the resolvent)
and finally

$$(A_0 \vee A_1 \vee y_1), (\neg y_1 \vee A_1) \vdash_{\overline{2-Res}} (A_0 \vee A_1).$$

The transformation *Trans-3* generates the clauses $(x_0 \vee x_1 \vee P_1)$ and $(\neg P_1 \vee y_0 \vee y_1)$ from $(x_0 \vee x_1 \vee y_0 \vee y_1)$. Then there is no 2–resolution deduction $\varphi^3 \vdash_{\overline{2-Res}} (A_0 \vee A_1)$, because a 2–resolution of the 2-clauses with the 3-clauses leads to $(A_0 \vee A_1 \vee P_1)$ and $(\neg P_1 \vee A_0 \vee A_1)$.
For $k \geq 2$ we define the formula Φ as follows:

$$\begin{aligned} \text{atomset}(\Phi) &= \{x_{i,j} \mid 0 \leq j < k,\ 0 \leq i < 2^k\} \cup \{A_0, \ldots, A_{k-1}\} \cup \\ & \quad \{y_{i,j} \mid 0 \leq j < k,\ 0 \leq i < 2^k\} \end{aligned}$$

$$\Phi := \Big\{ \ (x_{i,0} \vee \ldots \vee x_{i,k-1} \vee y_{i,0} \vee \ldots \vee y_{i,k-1}) \mid 0 \leq i < 2^k \ \Big\} \cup \bigcup_{0 \leq i < 2^k} \beta_i \ \cup \bigcup_{0 \leq i < 2^k} \beta'_i$$

where

$$\beta_i := \Big\{ \ (\neg x_{i,j} \vee A_j^{\varepsilon_j}) \mid i = \sum_{l=0}^{k-1} \varepsilon_l 2^l,\ \ \varepsilon_l \in \{0,1\},\ 0 \leq j \leq k-1 \ \Big\}$$

$$\beta'_i := \Big\{ \ (\neg y_{i,j} \vee A_j^{\varepsilon_j}) \mid i = \sum_{l=0}^{k-1} \varepsilon_l 2^l,\ \ \varepsilon_l \in \{0,1\},\ 0 \leq j \leq k-1 \ \Big\}$$

We define $\varphi_k = \{(A_0^{\varepsilon_0} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}}) \mid \varepsilon_0, \ldots, \varepsilon_{k-1} \in \{0,1\}\}$, where $A^0 := \neg A$ and $A^1 := A$.
Next we show $\Phi \vdash_{\overline{2-Res}} \varphi_k$. For $i = \sum_{j=0}^{k-1} \varepsilon_j \, 2^j$ we apply $(x_{i,0} \vee \ldots \vee x_{i,k-1} \vee y_{i,0} \vee \ldots \vee y_{i,k-1})$ to $(\neg x_{i,j} \vee A_j^{\varepsilon_j})$ and $(\neg y_{i,j} \vee A_j^{\varepsilon_j})$ for $0 \leq j < k$. Then we obtain the clause $(A_0^{\varepsilon_0} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}})$.
Hence the formula φ_k containing all possible clauses of length k with atoms $A_0, \ldots, A_{k-1}$ can be deduced by means of 2–resolution. Therefore, the empty clause is deducible using k–resolution since all clauses of length k have been deduced.
Now it remains to show that there is no k–resolution refutation for Φ^{k+1}. The transformation into $(k+1)$–CNF transfers each clause $(x_{i,0} \vee \ldots \vee x_{i,k-1} \vee y_{i,0} \vee \ldots \vee y_{i,k-1})$ to the $(k+1)$-clauses $(x_{i,0} \vee \ldots \vee x_{i,k-1} \vee P_i)$ and $(\neg P_i \vee y_{i,0} \vee \ldots \vee y_{i,k-1})$ with new atoms P_i.
Applying the 2-clauses directly to the generated $(k+1)$-clauses leads to $(A_0^{\varepsilon_0} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}} \vee P_i)$ and $(\neg P_i \vee A_0^{\varepsilon_0} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}})$ and no further 2–resolution step can be performed with these clauses. But we could resolve a pair of 2-clauses $(\neg x_{i,j} \vee A_j^{\varepsilon_j}), (\neg x_{i',j} \vee A_j^{\varepsilon_j - 1})$ or

$(\neg x_{i,j} \vee A_j^{\varepsilon_j}), (\neg y_{i',j} \vee A_j^{\varepsilon_j - 1})$. Note that i and i' must be different numbers. The resulting clauses are $(\neg x_{i,j} \vee \neg x_{i',j})$ and $(\neg x_{i,j} \vee \neg y_{i',j})$. Resolving these clauses with the generated $(k+1)$-clauses does not help to deduce a k-clause, because no double occurrence of a literal can be obtained.
Resolving the 2-clauses yields $(\neg x_{i,j} \vee A_j^{\varepsilon_j}), (\neg x_{i',j} \vee A_j^{\varepsilon_j - 1})$ for $i = \sum_{l=0}^{k-1} \varepsilon_l 2^l$ and $(\neg x_{i,j} \vee A_j^{\varepsilon_j}), (\neg y_{i',j} \vee A_j^{\varepsilon_j - 1})$ for $i = \sum_{l=0}^{k-1} \varepsilon_l 2^l$ and $i' = i - \varepsilon_j 2^j + (1-\varepsilon_j)2^j$.
If we want to deduce the empty clause we must resolve the 2-clauses with clauses of length $k+1$. Then the deducible clauses are

$$(\neg z_{i_0,0} \vee \ldots \vee \neg z_{i_{k-1},k-1} \vee P_i) \ , \ (\neg P_i \vee \neg w_{s_0,0} \vee \ldots \vee \neg w_{s_{k-1},k-1}),$$

for $w_{i,j}, z_{i,j} \in \{x_{i,j}, y_{i,j}\}$ or $w_{i,j}, z_{i,j} \in \{A_j^{\varepsilon_j}, A_j^{1-\varepsilon_j}\}$.
Note that $z_{i_0,0}, \ldots, z_{i_{k-1},k-1}$ and $w_{s_0,0}, \ldots, w_{s_{k-1},k-1}$ are pairwise different. Hence the clauses of length $k+1$ can not be reduced and therefore the empty clause is not deducible by k–resolution.

■

Theorem 5.2

(i) *For each $3 \le t \le k$ and Φ in CNF: If $\Phi \vdash_{\overline{k-Res}} \sqcup$ then $\Phi^t \vdash_{\overline{k-Res}} \sqcup$.*

(ii) *For each $3 \le t \le k$ there is some Φ in CNF, such that $\Phi^t \vdash_{\overline{k-Res}} \sqcup$ but not $\Phi \vdash_{\overline{k-Res}} \sqcup$.*

Proof: Proof (i) is obvious.
Ad (ii): Let Φ_{k+1} be the formula consisting of all possible clauses with atoms $A_1, \ldots, A_{k+1}$. Then there is no k–resolution refutation $\Phi_{k+1} \vdash_{\overline{k-Res}} \sqcup$ because Φ_{k+1} does not contain a k-clause. For each $3 \le t \le k$ the transformation algorithm *Trans-t* generates from $\pi = (L_1 \vee \ldots \vee L_{k+1})$ the clauses $(k+1 := t-1+r(t-2)+s, \ \ s < t, \ \ n := t-1+r(t-2)\ )$:

$$\begin{array}{lcl}
\pi_0 & = & (L_1 \vee \ldots \vee L_{t-1} \vee P_1) \\
\pi_i & = & (\neg P_i \vee L_{t-1+(i-1)(t-2)+1} \vee \ldots \vee L_{t-1+i(t-2)} \vee P_{i+1}) \ \text{ for } 1 \le i \le r \\
\pi_{r+1} & = & (\neg P_{r+1} \vee L_{n+1} \vee \ldots \vee L_{k+1}) \text{ for new atoms } P_1, \ldots, P_{r+1}
\end{array}$$

Next we show that there is a k–resolution deduction $\Phi_{k+1}^t \vdash_{\overline{k-Res}} \Phi_k$, where Φ_k is the formula consisting of all clauses of length k with atoms $A_1, \ldots, A_k$. Suppose we have a clause $(A_1^{\sigma_1} \vee \ldots \vee A_{k+1}^{\sigma_{k+1}})$ in Φ_{k+1} with $\sigma_1, \ldots, \sigma_{k+1} \in \{0,1\}$. Applying *Trans–3* to this clause yields the clauses

$$(A_1^{\sigma_1} \vee \ldots \vee A_{t-1}^{\sigma_{t-1}} \vee P_1^{\sigma_1,\ldots,\sigma_{k+1}}) \ , \ (\neg P_1^{\sigma_1,\ldots,\sigma_{k+1}} \vee \ldots \ ) \ , \ \ldots \ ,$$

$$(\ \ldots \vee P_{r+1}^{\sigma_1 \ldots,\sigma_{k+1}}) \ , \ (\neg P_{r+1}^{\sigma_1,\ldots,\sigma_{k+1}} \vee A_{n+1}^{\sigma_{n+1}} \vee \ldots \vee A_{k+1}^{\sigma_{k+1}}) \text{ for new atoms } P_i^{\sigma_1,\ldots,\sigma_{k+1}}$$

Since $t \le k$ there is a k–resolution deduction of $(A_1^{\sigma_1} \vee \ldots \vee A_n^{\sigma_n} \vee P_n^{\sigma_1,\ldots,\sigma_{k+1}})$. Hence we can continue with clauses $(A_1^{\delta_1} \vee \ldots \vee A_n^{\delta_n} \vee P_n^{\delta_1,\ldots,\delta_n,\varepsilon_1,\ldots,\varepsilon_s})$ and $(\neg P_n^{\delta_1,\ldots,\delta_n,\varepsilon_1,\ldots,\varepsilon_s} \vee A_{n+1}^{\varepsilon_1} \vee \ldots \vee A_{k+1}^{\varepsilon_s})$ for $\delta_1, \ldots, \delta_n, \varepsilon_1, \ldots, \varepsilon_s \in \{0,1\}$. Note that the second clause has at most t literals. Now we resolve $(\neg P_n^{\delta_1,\ldots,\delta_n,\varepsilon_1,\ldots,\varepsilon_s} \vee A_{n+1}^{\varepsilon_1} \vee \ldots \vee A_{k+1}^{\varepsilon_s})$ and $(\neg P_n^{\delta_1,\ldots,\delta_n,\varepsilon_1,\ldots,\varepsilon_{s-1},1-\varepsilon_s} \vee$

$A_{n+1}^{\varepsilon_1} \vee \ldots \vee A_{k+1}^{1-\varepsilon_s})$ obtaining the clause $\beta := (\neg P_n^{\delta_1,\ldots,\delta_n,\varepsilon_1,\ldots,\varepsilon_s} \vee \neg P_n^{\delta_1,\ldots,\delta_n,\varepsilon_1,\ldots,\varepsilon_{s-1},1-\varepsilon_s} \vee A_{n+1}^{\varepsilon_1} \vee \ldots \vee A_k^{\varepsilon_{s-1}})$. Next we apply β to $(A_1^{\delta_1} \vee \ldots \vee A_n^{\delta_n} \vee P_n^{\delta_1,\ldots,\delta_n,\varepsilon_1,\ldots,\varepsilon_{s-1},\sigma})$ for $\sigma = 0$ and $\sigma = 1$. Then we have deduced the clause $(A_1^{\delta_1} \vee \ldots \vee A_n^{\delta_n} \vee A_{n+1}^{\varepsilon_1} \vee \ldots \vee A_k^{\varepsilon_{s-1}})$. Since $\delta_1, \ldots, \delta_n, \varepsilon_1, \ldots \varepsilon_{s-1}$ are arbitrarily chosen we have shown that each clause in Φ_k can be deduced from Φ_{k+1}^t by k–resolution. Altogether we obtain a k–resolution refutation $\Phi_{k+1}^t \vdash_{\overline{k-Res}} \Phi_k \vdash_{\overline{k-Res}} \sqcup$.

■

The last theorem says that the transformation *Trans-3* can be emphasized because for the resulting formula a k–resolution refutation exists if there is a k–resolution refutation of the original formula. More important is the fact that additional formulas are decidable by k–resolution

Next we ask how long the clauses in a k–resolution deduction can be if the initial formula is in $(k+1)$–CNF.

Theorem 5.3 *For each natural numbers $k \geq 3$ and t there exists a formula Φ in $(k+1)$–CNF and a clause $(A_1 \vee \ldots \vee A_k)$, such that*

(i) $\Phi \vdash_{\overline{k-Res}} (A_1 \vee \ldots \vee A_k)$

(ii) each k–resolution deduction from Φ to $(A_1 \vee \ldots \vee A_k)$ contains at least a clause of length greater than t.

Proof: Let $k \geq 3$ and t be given and let $(A_1 \vee \ldots \vee A_k)$ be a clause with atoms $A_1, \ldots, A_k$. At first we construct a formula Φ in $(k+1)$–CNF, for which a k–resolution deduction $\varphi \vdash_{\overline{k-Res}} (A_1 \vee \ldots \vee A_k)$ exists.

$$\begin{aligned}
\varphi \;:=\; & \Big\{(B_1 \vee \ldots \vee B_{k+1})\Big\} \cup \Big\{(\neg B_i \vee C_1(i) \vee \ldots \vee C_{k-1}(i)) \mid 1 \leq i \leq k+1\Big\} \\
\cup \;& \Big\{(\neg C_j(i) \vee C_1(j,i) \vee \ldots \vee C_{k-1}(j,i)) \mid 1 \leq i \leq k+1, 1 \leq j < k\Big\} \\
\cup \;& \bigcup_{2 \leq r \leq t} \Big\{(\neg C_j(i_1,\ldots,i_r) \vee C_1(j,i_1,\ldots,i_r) \vee \ldots \vee C_{k-1}(j,i_1,\ldots,i_r)) \mid \\
& \qquad 1 \leq j, i_1, \ldots i_{r-1} < k,\; 1 \leq i_r \leq k+1\Big\} \\
\cup \;& \Big\{(\neg C_1(i_1,\ldots,i_t) \vee A_2 \vee \ldots \vee A_k) \mid 1 \leq i_1, \ldots i_{t-1} < k,\; 1 \leq i_t \leq k+1\Big\} \\
\cup \;& \Big\{(\neg C_j(i_1,\ldots,i_t) \vee A_1 \vee \ldots \vee A_{k-1}) \mid \\
& \qquad 1 \leq i_1, \ldots i_{t-1} < k,\; 2 \leq j < k,\; 1 \leq i_t \leq k+1\Big\}
\end{aligned}$$

where $C_j(i_1, \ldots, i_q)$ are atoms.
Then there exists a k–resolution deduction $\Phi \vdash_{\overline{k-Res}} (A_1 \vee \ldots \vee A_k)$ as follows

$$\begin{array}{ll}
(B_1 \vee \ldots \vee B_{k+1}) & \\
\quad\vdots & (\neg B_i \vee C_1(i) \vee \ldots \vee C_{k-1}(i)) \\
\quad\vdots \swarrow & \\
(C_1(1) \vee C_1(2) \ldots \vee C_{k-1}(k+1) \vee C_{k-1}(k+1)) & \\
\quad\vdots & (\neg C_j(i) \vee C_1(j,i) \vee \ldots \vee C_{k-1}(j,i)) \\
\quad\vdots \swarrow & \\
(C_1(1,1) \vee C_1(1,2) \vee \ldots \vee C_{k-1}(k-1,k) \vee C_{k-1}(k-1,k+1)) & \\
\quad\vdots \quad \swarrow & \\
\quad\vdots \swarrow & \\
(C_1(1,\ldots,1) \vee C_1(1,\ldots,1,2) \vee \ldots \vee C_{k-1}(k-1,\ldots,k-1,k+1)) & \\
\quad\vdots \swarrow & \\
\quad\vdots & (\neg C_1(i_1,\ldots,i_t) \vee A_2 \vee \ldots A_k) \\
\quad\vdots \swarrow & \\
\quad\vdots & (\neg C_j(i_1,\ldots,i_t) \vee A_1 \vee \ldots A_{k-1}) \\
\quad\vdots \swarrow & \\
(A_1 \vee \ldots \vee A_k) &
\end{array}$$

Obviously, the length of the clause

$$\Big(\ C_1(1,\ldots,1) \vee C_1(1,\ldots,1,2) \vee \ldots \vee C_{k-1}(k-1,\ldots,k-1,k+1)\ \Big)$$

is greater than t.
Now we have to show that there is no k–resolution deduction from Φ to $(A_1 \vee \ldots \vee A_k)$ for which each clause has length less than t. Assume there is a k–resolution deduction with clauses of length less then t. Resolving clauses without atoms A_j increases the length of the deduced clauses step by step. Hence a reduction of the length of generated clauses seems to be possible only with clauses which contain an atom A_i. So we could resolve

$$\big(\neg C_j(i_1,\ldots,i_t) \vee C_1(j,i_1,\ldots,i_t) \vee \ldots \vee C_{k-1}(j,i_1,\ldots,i_t)\big)$$

with $(\neg C_j(i_1,\ldots,i_t) \vee A_1 \vee \ldots \vee A_{k-1})$ or $(\neg C_1(i_1,\ldots,i_t) \vee A_2 \vee \ldots \vee A_k)$. The resolvent is the $(k+1)$-clause $(\neg C_1(i_1,\ldots,i_t) \vee A_1 \vee \ldots \vee A_k)$. When this clause is resolved with a clause

$$\big(\neg C_{i_1}(i_2,\ldots,i_t) \vee C_1(i_1,\ldots,i_t) \vee \ldots \vee C_{k-1}(i_1,\ldots,i_t)\big)$$

in order to match $\neg C_{i_1}(i_1,\ldots,i_t)$, we obtain the clause

$$\big(\neg C_{i_1}(i_2,\ldots,i_t) \vee C_1(i_1,\ldots,i_t) \vee C_{j-1}(i_1,\ldots,i_t) \vee A_1 \vee \ldots \vee A_k \vee C_{j+1}(i_1,\ldots,i_t)\big).$$

Then only a clause with occurrence of the atom $C_{i_1}(i_2,\ldots,i_t)$ can be resolved. If we proceed we see that the length of the resolvents increases.

■

The next theorem states that the length of clauses in a $(k+1)$–resolution refutation can be bound by $k + \max(k,t)$, if Φ is in t–CNF and there is a This shows that if Φ is in $(k+1)$–CNF and $\Phi \vdash_{k-Res} \sqcup$ the length of the clauses can be restricted to $2k+1$. Thus it can happen that in a k–resolution refutation long clauses must be used whereas in a $(k+1)$–resolution refutation the length of the clauses can be bound by $2k+1$.

Theorem 5.4 *For each $k \geq 3$ and Φ in t-CNF there is a $(k+1)$–resolution refutation, where each clause in the refutation contains at most $k + \max(k,t)$ literals, if and only if $\Phi \vdash_{k-Res} \sqcup$.*

Proof: Let Φ be a formula in t–CNF and $\Phi \vdash_{k-Res} \sqcup$. At first we consider subtrees in the refutation of the form

$$\begin{tikzcd}
 & \alpha_0 \arrow[d] & \alpha_1 \arrow[d] & \cdots & \alpha_{n-1} \arrow[d] \\
\phi \arrow[r] & \beta_1 \arrow[r] & \beta_2 & \cdots \arrow[r] & \beta_n
\end{tikzcd}$$

where ϕ is a t-clause in Φ, $\alpha_0, \ldots \alpha_{n-1}$ are k-clauses and $\beta_n = (L_1 \vee \ldots \vee L_k)$ is a k-clause. The subtree can be considered as an input–resolution for the formula $\phi = \{\phi, \alpha_0, \ldots, \alpha_{n-1}\}$ with $\phi \vdash_{Input-Res} \beta_n$.
Next we form the formula $\phi[L_1/0, \ldots, L_k/0]$, that means we remove each clause with literals $\neg L_i$ and delete each occurrence of the literals L_i in the clauses in ϕ.
Then there is an input–resolution refutation $\phi[L_1/0, \ldots, L_k/0] \vdash_{Input-Res} \sqcup$ and therefore a unit–resolution refutation $\phi[L_1/0, \ldots, L_k/0] \vdash_{Unit-Res} \sqcup$. To each clause occurring in the unit-refutation we add the removed literals L_i. Hence we obtain $\phi \vdash_{(k+1)-Res} \beta_n$ and each clause in the deduction has at most length $k + \max(k,t)$, because within the unit-refutation the length of the clause is reduced in each resolution step.
Since a k–resolution refutation tree of $\Phi \vdash_{k-Res} \sqcup$ can be divided into subtrees as considered above, we have shown that there is a $(k+1)$–resolution refutation with clauses of length less or equal to $k + \max(k,t)$.

■

By theorems 5.2 and 5.4 we see that for example if a 3–resolution refutation $\Phi \vdash_{3-Res} \sqcup$ exists we can transform Φ into Φ^3 in 3–CNF and then find a 4–resolution refutation $\Phi^3 \vdash_{4-Res} \sqcup$ with at most $O(\text{length}(\Phi)^6)$ different clauses of length less or equal k. This gives an upper bound for the 4–resolution steps we have to perform.

The procedure can not be used to decide whether a 3–resolution refutation $\Phi \vdash_{3-Res} \sqcup$ exists because the transformation *Trans-3* may lead to formulas refutable with 3-resolution, whereas for the original formula no 3–resolution refutation exists.

6 Extended k-Resolution

A simple form of an extension of resolution is to substitute a disjunction $(L_1 \vee \ldots \vee L_m)$ by a new atom y and to add the equivalence $y \leftrightarrow (L_1 \vee \ldots \vee L_m)$ to the formula. Since we are dealing with formulas in CNF, we have to add $(\neg y \vee L_1 \vee \ldots \vee L_m)$ and $(y \vee \neg L_i)$ for $1 \leq i \leq m$. Then the resulting formula is equivalent with respect to satisfiability to the original formula. Such extensions are investigated for example in [2],[6] and seem to be very powerful because up to now no formulas are known, for which resolution combined with such an extension has minimal proofs of exponential length, whereas for resolution Haken [4] proved an exponential lower bound for the prooflength of the pigeon hole formulas.

Definition: (Extension Rule)
Let Φ be a formula in CNF, $L_1 \dots L_m$ literals in literalset(Φ) and x a new atom, then we define

$$\Phi_x = \Phi \cup \Big\{(\neg x \vee L_1 \vee \dots \vee L_m), (x \vee \neg L_1), \dots, (x \vee \neg L_m)\Big\}$$

The extension rule is denoted by $\Phi \mid\!\frac{1}{Ext}\, \Phi_x$. The formula Φ is satisfiable if and only if Φ_x is satisfiable.

Combining the extension rule and the resolution rule we define the extended resolution.

Definition: $\mid\!\frac{1}{E-Res}$ is either the resolution rule $\mid\!\frac{1}{Res}$ or the extension rule $\mid\!\frac{1}{Ext}$. $\mid\!\frac{}{E-Res}$ denotes the transitive and reflexive closure. $\mid\!\frac{1}{E-k-Res}$ is either a k-resolution or an extension step.

It is well-known that extended resolution is complete and sound. We shall show that in combination with the extension rule 3-resolution is complete too. The idea of the prove is quite simple because each resolution step can be simulated by a sequence of extensions and applications of 3-resolution.

Lemma 6.1 *(Simulation)*
Let $(L_1 \vee \dots \vee L_n \vee L)$ and $(K_1 \vee \dots \vee K_m \vee \neg L)$ be clauses with $n, m \geq 2$ then
$(L_1 \vee \dots \vee L_n \vee L)$, $(K_1 \vee \dots \vee K_m \vee \neg L) \mid\!\frac{}{E-3-Res}\, (L_1 \vee \dots \vee L_m \vee K_1 \vee \dots \vee K_m)$.

Proof: We first apply the extension rule to $(L_1 \vee \dots \vee L_n \vee L)$. Without loss of generality we assume n is even. Let $y_1^1, \dots, y_{n/2}^1$ be new variables then we introduce the clauses

$$\begin{array}{cccc} (\neg y_1^1 \vee L_1 \vee L_2) & (\neg y_2^1 \vee L_3 \vee L_4) & \dots & (\neg y_{n/2}^1 \vee L_{n-1} \vee L_n) \\ (y_1^1 \vee \neg L_1) & (y_2^1 \vee \neg L_3) & \dots & (y_{n/2}^1 \vee \neg L_{n-1}) \\ (y_1^1 \vee \neg L_2) & (y_2^1 \vee \neg L_4) & \dots & (y_{n/2}^1 \vee \neg L_n). \end{array}$$

Then we obtain the clause $(y_1^1 \vee \dots \vee y_{n/2}^1 \vee L)$ in applying resolution to the 2-clauses and $(L_1 \vee \dots \vee L_n \vee L)$. We again assume $n/2$ is even and introduce new variables $y_1^2, \dots, y_{n/4}^2$ and extend the formula

$$\begin{array}{ccc} (\neg y_1^2 \vee y_1^1 \vee y_2^1) & \dots & (\neg y_{n/4}^2 \vee y_{n/2-1}^1 \vee y_{n/2}^1) \\ (y_1^2 \vee \neg y_1^1) & \dots & (y_{n/4}^2 \vee \neg y_{n/2-1}^1) \\ (y_1^2 \vee \neg y_2^1) & \dots & (y_{n/4}^2 \vee \neg y_{n/2}^1). \end{array}$$

Then we obtain by 2-resolution the clause $(y_1^2 \vee \dots \vee y_{n/4}^2 \vee L)$. For sake of simplicity we suppose $n = 2^t$. Then after $t-1$ steps we obtain $(y_1^{t-1} \vee y_2^{t-1} \vee L)$. We proceed analogously with the clause $(K_1 \vee \dots \vee K_m \vee \neg L)$ and $m = 2^k$ and obtain the clause $(z_1^{k-1} \vee z_2^{k-1} \vee \neg L)$. Now it remains to apply the 3-resolution to the constructed clauses in order to deduce the clause $(L_1 \vee \dots \vee L_n \vee K_1 \vee \dots \vee K_m)$. We first resolve $(y_1^{t-1} \vee y_2^{t-1} \vee L)$ and $(z_1^{k-1} \vee z_2^{k-1} \vee \neg L)$ to get $(y_1^{t-1} \vee y_2^{t-1} \vee z_1^{k-1} \vee z_2^{k-1})$ and continue with the introduced clauses. Finally we obtain our desired clause.

■

As an immediate consequence we achieve

Theorem 6.2 *A formula Φ in CNF is unsatisfiable if and only if* $\Phi \vdash_{\overline{E-3-Res}} \sqcup$.

If we restrict ourselves to extended 2-resolution then we see that this resolution is essential more powerful than 2-resolution because for each formula Φ_k which consists of all possible clauses of length k with the variables $A_1, \ldots, A_k$, there is a refutation $\Phi \vdash_{\overline{E-2-Res}} \sqcup$. The idea is to abbreviate $(A_1^{\varepsilon_1} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}})$ by y. That means

$$(A_1^{\varepsilon_1} \vee \ldots \vee A_k^{\varepsilon_k}) \vdash_{\overline{E-2-Res}} (\neg y \vee A_1^{\varepsilon_1} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}}) \ , \ (y \vee \neg A_i^{\varepsilon_i}) \ \text{ for } 1 \leq i < k,$$

$$(A_1^{\varepsilon_1} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}} \vee \neg A_k^{\varepsilon_k}) \ , \ (y \vee \neg A_i^{\varepsilon_i}) \vdash_{\overline{E-2-Res}} (y \vee \neg A_k^{\varepsilon_k}).$$

Analogously $(y \vee \neg A_k^{1-\varepsilon_k})$ can be deduced. We first resolve $(y \vee \neg A_k^{\varepsilon_k})$ and $(y \vee \neg A_k^{1-\varepsilon_k})$ obtaining the clause (y). Then we apply (y) to $(\neg y \vee A_1^{\varepsilon_1} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}})$ and get $(A_1^{\varepsilon_1} \vee \ldots \vee A_{k-1}^{\varepsilon_{k-1}})$. Hence we can deduce Φ_{k-1} by E-2-resolution. By repeating this procedure we finally obtain Φ_2 for which a 2-resolution refutation exists.

7 Conclusion

We have presented results regarding k–resolution. A possible resolution strategy for deciding whether a formula in CNF is satisfiable is to prefer a unit-resolution step or generally short clauses. It has been proved that the speed-up by using resolution is bound by $\sqrt{t}$ if the shortest unit–resolution refutation needs t steps. A first approach for the 2–resolution subproblem has been shown and more questions have arisen than we have answered: For example, is there an algorithm for 3–resolution which has polynomial time complexity or can we find a better upper bound than $O(\text{length}(\Phi)^4)$ in the 2–resolution case?

References

[1] Dowling, W. F., Gallier, I.H.: *Linear-time Algorithms for Testing the Satisfiability of Propositional Horn Formulas*, I. Logic Programming 3 (1984), pp. 267–284

[2] Eder, E.: *Relative Complexities of First Order Calculi*, Vieweg 1992

[3] Gallo, G., Scutellà, M.G.: *Polynomially Satisfiability Problems*, Information Processing Letters 29 (1988), pp. 267–284

[4] Haken A.: *The Intractability of Resolution*, TCS 39 (1985), pp. 297–308

[5] Kleine Bning, H.: *On Generalized Horn Formulas and k–Resolution*, TCS 116 (1993), pp. 405–413

[6] Tseitin, G.S.: *On the Complexity of Derivations in the Propositional Calculus*, in A.O. Slisenko (ed.): Structures in Constructive Mathematics and Mathematical Logic, Part. II (1968), pp. 115–125

[7] Wos, L., Carson, D., Robinson, G.A.: *The Unit Preference Strategy in Theorem Proving*, AFIPS Conf. Proc. 26, Spantau Books, Washington D.C.

[8] Yamasaki, S., Doshita, S.: *The Satisfiability Problem for a Class Consisting of Horn Sentences and Some None-Horn Sentences in Propositional Logic*, Information and Control 59 (1983), pp. 1–12