An Algorithm Based on Equivalence Classes for Tableau with Equality*

LIU Quan[1,2], SUN Ji-gui[2], Wang Song-Jiang[3]

[1](Institute of Daqing Oil Production Technology, Daqing 163453, China)

[2](Institute of Computer Science and Technology, Jilin University, Changchun 130012,China)

[3](University of Toronto, Cannada)

E-Mail: **Liuquan@cys.dq.cnpc.com.cn**

Abstract: This paper introduces an algorithm based on equivalence classes for tableau with equality ——CECA, which using the method based on the transformation into disjunctions of inequalities and the calculation of equivalence classes. It restrict equality applications and to avoid the generation of useless new formulae. It is possible to restrict search space. In the same time, to study the effectiveness of the algorithm, an example is made to analyze and compare its performance to Fitting's approach and Jeffrey's approach. In addition we have implemented the algorithm on a PC workstation using PROLOG. The results show tableau based on equivalence classes is superior to that of other algorithm.

Key words: disjunctions of inequality; equivalence classes; fair Heuristic

1 Introduction

One of the main goals of automated deduction is to efficiently handle first-order logic with equality. Just adding the equality to the data base leads to a huge search space. Very simple theorems cannot be proven. The only solution is to make the handling of equality part of the inference rules. Then, still, equality typically allows a lot of different derivations. Methods have to be used for further restricting the search space. For resolution-based provers such methods — the most important being paramodulation[1] and RUE-resolution[2]—have been known since the 1960s and have often been implemented although the problem of preventing the derivation of redundant information remains to be solved.

At the same time methods for adding equality to Gentzen-type calculi, such as semantic tableaux and the connection method, have been developed[3]. These, have not been used as often. But recently much more efficient methods have been developed, and over the last years there has been a growing interest in handling equality in semantic tableaux and the connection method[4].

In this paper, we adopted the idea to restrict search space by using the method based on the transformation into disjunctions of inequalities and the calculation of equivalence classes. In the same time, to study the effectiveness of the algorithm, we have implemented the algorithm on a PC workstation using PROLOG. The results show tableau based on equivalence classes is superior to that of other algorithm.

2 Analyze to Tableau with Equality

In first-order logic, the substitution can be obtained in a similar way as those needed to close a branch: If an equality $t \approx s$ is to be applied to a formula $\phi[t']$, the application of MGU μ of t and t' to the tableau is sufficient to derive $(\phi[s])^{\mu}$. However, the unifier μ has to be applied not only the

* Supported by the National Natural Science Foundation of China under Grant No.60073039

formulae involved but to the whole tableau, as shown in Fig 1.

$$\frac{t \approx s \qquad \phi[t']}{(\phi[s])^{\mu}} \qquad\qquad \frac{s \approx t \qquad \phi[t']}{(\phi[s])^{\mu}}$$

μ is a MGU of t and t´ that is applied to the whole tableau.

Fig. 1 Additional expansion rules for first-order logic tableau

Theorem 1[4]**.** A first-order sentence ϕ is an E-tautology if and only if there is a sequence $T_0, \ldots, T_n$ of tableaux ($n \geq 0$) such that

. T_0 consists of the single node $\neg\phi$.

. For $1 \leq i \leq n$ the tableau T_i is constructed from T_{i-1} by

(a) applying one of the tableau expansion rules from tableau expansion rules or one of the equality expansion rules. or

(b) applying a substitution σ that closes a branch B of T_{i-1}, i.e., there is an inequality $\neg(t \approx s) \in B$ such that σ is an MGU of t and s or there are formulae $\varphi, \neg\psi \in B$ such that σ is an MGU of ϕ and ψ.

. All branch of T_n are marked as being closed.

Proof. Proofs of the theorem can be found in [4].

Example 1. Fig. 2 shows a tableau that proves the following set of formulae to be inconsistent:

$\{(\forall x)((g(x) \approx f(x)) \quad \neg(x \approx a)) \quad (\forall x)(g(f(x)) \approx x) \quad b \approx c \quad P(g(g(a)),b) \quad \neg P(a,c)\}$

```
(∀x)((g(x) ≈f(x))   ¬(x≈a))
(∀x)(g(f(x)) ≈x)
b≈c
P(g(g(a)),b)
¬P(a,c)
g(f(x1)) ≈x1
(g(x2) ≈f(x2))   ¬(x2≈a)
        /                    \
¬(x2≈a)                (g(x2) ≈f(x2)
 *                  P(g(f(a)),b)
                    P(a,b)
                    P(a,c)
                       *
```

Fig.2 Expanded tableau using expansion rules

By applying the standard tableau rules, formula are derived from , from , and from , is derived by applying equality to (the substitution $\{a/x_2\}$ has to be applied), and formula is derived by applying to (the substitution $\{a/x_1\}$ has to be applied),. formula is derived by applying to . Formulae and close the right branch. The left branch is closed by the inequality (the substitution {a/x2} has already been applied).The new expansion rules are symmetrical and their application is completely unrestricted. The leads to an enormous number of irrelevant formulae that can be added.

3 CECA—an Calculate Equivalence Classes Algorithm

3.1 Data Structures for Algorithm

Definition 1. Let B is a branch of tableau. Then B is converted into the equality pair <E(B),P(B)>.

The set E(B) defined as $E(B):=\{t\approx s: t\approx s\in B\}$;

For every pair $<p(t_1,\ldots,t_n), \neg p(s_1,\ldots,s_n)>$ of atoms that potentially close B, in P(B) there is the n-place disjunction $t_1\neq s_1 \quad \ldots \quad t_n\neq s_n$; and for every inequality $\neg(t\approx s)$ on B, In P(B) there is the One-place disjunction $t\neq s$.

In example 1, Let B_1 denotes the left and B_2 denotes the right branch of the tableau in Fig.2. The equality pair contain the problem

$\{b\approx c, g(f(x_1)) \approx x_1\}\subset E(B_1)$

$\{b\approx c, g(f(x_1)) \approx x_1, g(x_2) \approx f(x_2)\}\subset E(B_2)$

$\{(g(g(a)\neq a \quad b\neq c) \quad (x_2\neq a))\subset P(B_1)$.

$\{g(g(a)\neq a \quad b\neq c)\subset P(B_2)$

In general, CECA can be divided into three steps:

Step 1. Calculate the equivalent classes: Breadth-first-search can be implemented using set $<t>_B$ that contain all the equivalence classes of a term t on a branch B associated with different substitutions.

Step 2. Search fair term: The Heuristic H for choosing an element from $<t>_B$ to which equalities are applied has to be fair.

Step 3. Closed tableau: One can check whether a tableau T is closed according to definition of closed tableau by gradually computing the sets $<t>^{0}_{B_i}$,$<t>^{1}_{B_i}$,… for every term t occurring in $P(B_1),\ldots,P(B_k)$, where $B_1,\ldots,B_k$ are the branches of T.

3.2 Calculate the equivalent classes

For ground version of tableau, we handle tableau with equalities based on transformation into disjunctions of inequalities and calculation of equivalence classes, it can be solved by searching for equivalence classes. But for free variable version of tableau, one has to take into account that different substitutions may lead to different equivalence classes. It is important to find a suitable substitution that allow to refute all inequalities in a disjunction simultaneously.

Definition 2. $<t>_B$ is a set that contain all the equivalence classes of a term t on a branch B associated with different substitutions. The elements s_δ of $<t>_B$ are terms labeled with the substitution which is needed to derive them from t using equalities in E(B).

For example, the application of {a/x1} leads to an equivalence class of term a that contains g(f(a)), the element $g(f(a))_{\{a/x1\}}$ is in $<a>_B$.

$<t>_B$ is in general an infinite set, but there is a sequence of approximations $(<t>_B{}^n)_{n\geq 0}$ to $<t>_B$ that can be computed with a deterministic algorithm.

Definition 3. Suppose s,s' are terms and δ,δ' are substitutions. s_δ **subsumes** $s'_{\delta'}$ if s and s' are unifiable with an MGU τ such that $s^{\tau}= s'$ and both δ and τ are more general than δ'.

For example, $f(x)_{\{a/y\}}$ subsumes $f(a)_{\{a/x,a/y\}}$; $a_{\{y/x\}}$ subsumes $a_{\{b/x,b/y\}}$; On the other hand, $a_{\{f(y)/x\}}$ does not subsume $a_{\{f(b)/x\}}$.

Algorithm description 1. (Sequence of Sets $<t>^{n}_{B}$)

The sets $<t>^{n}_{B}$ are inductively calculate as follows:

1. $<t>^{0}_{B} =\{t_{id}\}$

2. $\Theta_n=\{$ s_δ: all the elements from $<t>^{n}_{B}$ and in addition the terms r_τ that can be derived in one step

from s_δ= H($<t>^n_B$), where r_τ can be derived in one step from s_δ as follows:}

τ is an MGU of a subterm of s and side of equality G E(B).

r can be derived from s^τ by application of G^τ.

σis more general thanτ.

3. $<t>^{n+1}_B$ =($<t>^n_B$ Θ_n)\{ s_τ: elements are subsumed by another element inΘ_n}.

Example 2. Table 1 shows the computation of $<a>^n_B$ (n=0,1,2) using the set of equalities E(B)={g(f(x_1)≈x_1 g(x_2)=f(x_2)).

Table Computation of $<a>^n_B$ (n=0,1,2)

n	$<a>^n_B$	H($<a>^n_B$)	Θ_n
0	a_{id}	a_{id}	a_{id} $g(f(a))_{\{a/x1\}}$
1	a_{id} $g(f(a))_{\{a/x1\}}$	$g(f(a))_{\{a/x1\}}$	a_{id} $g(f(a))_{\{a/x1\}}$ $a_{\{a/x1\}}$ $g(f(g(f(a))))_{\{a/x1\}}$ $g(g(a))_{\{a/x1,a/x2\}}$ $f(f(a))_{\{a/x1,f(a)/x2\}}$
2	a_{id} $g(f(a))_{\{a/x1\}}$ $a_{\{a/x1\}}$ $g(f(g(f(a))))_{\{a/x1\}}$ $g(g(a))_{\{a/x1,a/x2\}}$ $f(f(a))_{\{a/x1,f(a)/x2\}}$		

3.3 Search fair term

The Heuristic H for choosing an element from $<t>_B$ to which equalities are applied has to be fair in the following sense:

Definition 4(Fair Heuristic) A heuristic H is fair if for each term t, each n≥0, and each element $s_\sigma \in <t>^n_B$ there is an m≥0 such that H($<t>^m_B$) subsumes s_σ.

For our implementation we used the following heuristic for choosing the next element from $<t>^n_B$ to which equalities are applied:

Algorithm description 2. (Implemented Heuristic) The criteria for selection, ordered by their importance, are as follows:

1. Elements that have been chosen before are not considered again.

2. The term weight W(s) and the distance D(s)=W(s')-W(s) to the weight of the term s' from which s has been derived. Terms s are preferred that

have a positive weight distance D(s),

have a lower weight W(s),

have a higher weight distance D(s).

3. The number of steps necessary to derive a term. Terms that can be derived in fewer steps are preferred.

Theorem 2. The heuristic H that is chosen an element from $<t>^n_B$ by Algorithm 1 is fair.

Proof. Let any element s_σ $<t>^n_B$. For all m≥n, there must exist $s'_{\sigma'}$ $<t>^m_B$,and $s'_{\sigma'}$»s_σ. By algorithm 1, the elements of $\langle t\rangle_B^{n+k}$ is derived from $\langle t\rangle_B^{n+k+1}$. So $\langle t\rangle_B^{n+k+1}$ contains all elements of $\langle t\rangle_B^{n+k}$ Then for all $s'_{\sigma'}$ $\langle t\rangle_B^{m}$ there must exist W(s)≤W(s´) So D(s)= W(s')-W(s) is a positive distance.

Suppose s_σ has been chosen when m<n. Then it must be chosen in $\langle t\rangle_B^{n+m}$ when m≥0. It must have a lower weight W(s). Because every term can be chosen only one, this case is impossible.

3.4 Closed tableau

Algorithm description 3. (Closed Tableau): One can check whether a tableau T is closed according to definition of closed tableau by gradually computing the sets $\langle t\rangle_{B_i}^0, \langle t\rangle_{B_i}^1, \ldots$ for every term t occurring in $P(B_1),\ldots,P(B_k)$, where $B_1,\ldots,B_k$ are the branches of T.

1. For each branch $B_i(1\le i\le k)$: $D_i = (t_{i1} \ne s_{i1} \vee \cdots \vee t_{in_i} \ne s_{in_i}) \in P(B_i)$
2. For $1\le j\le n_i$: $r_{\rho_{ij}}^{ij} \in \langle t_{ij}\rangle_{B_i}^{l_{ij}}$ *and* $\underline{r}_{\underline{\rho}_{ij}}^{ij} \in \langle s_{ij}\rangle_{B_i}^{\underline{l}_{ij}}$ where r^{ij} and $\underline{r}^{ij}$ are unifiable with an MGU $\underline{\rho}_{ij}$
3. There is a grounding substitution σ such that all of the substitutions ρ_{ij} $\underline{\rho}_{ij}$ $\underline{\rho}_{ij}(1\le i\le k, 1\le j\le n_i)$ are more general than σ.

Lemma 1. If for some n≥0 there is an element s_σ $\langle t\rangle_B^n$, and σ is more general than the grounding substitutionτ, then the equivalence class of t that is associated with τ contains the term s^τ.

Proof. By induction on n.

For n=0 there is s_σ $\langle t\rangle_B=\{t_{id}\}$ with $s^\tau=t^\tau$ $[t^\tau]_\tau$ By definition 2, there must exist $(s^\tau)_\tau$ $\langle t\rangle_B$. The Lemma 1 holds.

Suppose for n=k Lemma 1 holds. For s_σ $\langle t\rangle_B$ and s_σ Θ, we divide into two cases.

For s_σ $\langle t\rangle_B$ $(s^\tau)_\tau$ $\langle t\rangle_B$ can be derived directly.

For s_σ Θ_n there is an element $s'_{\sigma'}$ $\langle t\rangle_B$ that can be derived by some s_σ step by step. Term s can be derived in one step from s'^{σ} and equality G E(B) by a certain heuristic H. And there must be exist $\tau' \succ \sigma$ Since $s'_{\sigma'}$ $\langle t\rangle_B$ and τ is a substitution, it must be exist $(s'^{\tau})_\tau$ $\langle t\rangle_B$ and s'^{τ} $[t^\tau]_\tau$.

Since there is a substitution σ that is more general than σ, and s^τ is be derived in one step from $s'^{\tau'}$ and equality G E(B) by a certain heuristic H, s^τ is in $[t^\tau]_\tau$. By definition 2, there must exist $(s^\tau)_\tau$ $\langle t\rangle_B$. The Lemma 1 holds.

Lemma 2. If r_σ $\langle t\rangle_B$, then for some n≥0 there is an element $r'_{\sigma'}$ $\langle t\rangle_B^n$ and $r'_{\sigma'}$» r_σ.

Proof. If r_σ $\langle t\rangle_B$, by definition 7, there must be exist r $[t^\sigma]_\sigma$.

For m≥0 from tσ r can be derived from u^0 … u^m by equalities sequence of E_1 … E_m in E(B).

$t^\sigma = u^0 \xrightarrow{E_1} u^1 \xrightarrow{E_2} \ldots \xrightarrow{Em} u^m = r = t_{id}$

By induction on 0≤k≤m.

For k=0 there is only an element $r_{\sigma_0}^0 = t_{id}$ $\langle t\rangle_B^n$ that subsumes $u_\sigma^0 = (t\sigma)_\sigma$, $r_{\sigma_0}^0$ »u_σ^0

Suppose k=n for l_k≥0 there is an element $r_{\sigma_k}^k$ $\langle t\rangle_B^{l_k}$ and $r_{\sigma_k}^k$ »u_σ^k

For k=n+1 there must be exist $r'^{k}_{\sigma'_k}$ »$r_{\sigma_k}^k$ by a heuristic H. Also there must be exist $H(\langle t\rangle_B^{l_{k+1}-1}) = r'^{k}_{\sigma'_k}$ »$r_{\sigma_k}^k$ »u_σ^k There is a substitution σ such that r'^{k}_{σ} $r^{k\sigma}$=

$u^{k\sigma}=u^k \xrightarrow{E_{k+1}} u^{k+1}$ Where $\sigma'_k \succ \sigma$ holds. By Algorithm 1, there is $\sigma'_{k+1} \succ \sigma$ such that $u^{k+1}_{\sigma'_{k+1}}$ Θ_{k+1} holds.

At the same theory, there is an element $r^{k+1}_{\sigma'_{k+1}}$ $<t>^{k+1}_{B}$,and $u^{k+1}_{\sigma'_{k+1}}$ »u^{k+1}_{σ} For m=k, there is $r'_{\sigma'}=r^{m}_{\sigma_m}$ $<t>^{n}_{B}=<t>^{l_m}_{B}$,and $r'_{\sigma'}$»$u^{m}_{\sigma}=r_\sigma$

Theorem 3. Suppose T is a tableau with root formula ϕ.

Soundness: If T is closed according to Algorithm 3 then ϕ is not satisfiable in a normal model.

Completeness: If ϕ is not satisfiable in a normal model and if the limit q for γ-rule applications is sufficiently high, T is closed according to Algorithm 3.

Proof. Soundness.

Suppose a grounding substitution τ is more general than σ. By definition 11, all substitution of τ are included in ρ_{ij} $\underline{\rho}_{ij}$ $\underline{\rho}_{ij}$:$1\le i\le k, 1\le j\le n_i$. Then for any $1\le i\le k, 1\le j\le n_i$, $r^{ij}_{\rho_{ij}} \in \langle t_{ij}\rangle^{l_{ij}}_{B_i}$ and $\underline{r}^{ij}_{\underline{\rho}_{ij}} \in \langle s_{ij}\rangle^{l_{ij}}_{B_i}$ By lemma 1, there is $(r^{ij}\tau)_\tau$ $<t_{ij}>_{B_i}$ and $(\underline{r}^{ij}\tau)_\tau$ $<s_{ij}>_{B_i}$. Where r^{ij} and $\underline{r}^{ij}$ are unifiable with an MGU $\underline{\rho}_{ij}$. So $r^{ij}\tau$ $\underline{r}^{ij}\tau$ holds, i.e. there are common elements between $<t_{ij}>_{B_i}$ and $<s_{ij}>_{B_i}$. By theorem 1, formula ϕ is not satisfiable in a normal model.

Completeness.

If formula ϕ is not satisfiable in a normal model. And if the limit q for γ-rule applications is sufficiently high. Then there must be exist a grounding substitution σ, for each branch B there is a disjunction $t_1\not\approx s_1$ … $t_n\not\approx s_n$ P(B) such that for$1\le i\le n$ $[t_i^\sigma]_B=[s_i^\sigma]_B$ and $1\le i\le k, 1\le j\le n_i$ there is an element r^{ij}_{σ} $<t_{ij}>_{B_i} \cap <s_{ij}>_{B_i}$ .By lemma 2, $r_{\rho_{ij}}$ $<t_{ij}>^{l_{ij}}_{B_i}$ and $\underline{r}_{\underline{\rho}_{ij}}$ $<s_{ij}>^{l_{ij}}_{B_i}$. $r_{\rho_{ij}}$ »r^{ij}_{σ} and $\underline{r}_{\underline{\rho}_{ij}}$ » r^{ij}_{σ} holds.

At same time, there are substitution $\mu$and ν such that $r^{ij}\mu=\underline{r}^{ij}\nu$,where μ ν ρ_{ij} $\underline{\rho}_{ij}$ $\underline{\rho}_{ij}(1\le i\le k, 1\le j\le n_i)$ are more general than σ. Since $r^{ij\sigma}$ $\underline{r}^{ij\sigma}$ holds and r^{ij} and $\underline{r}^{ij}$ are unifiable with an MGU $\underline{\rho}_{ij}$ T is closed i.e.

4 Example and Assay to Algorithm

In this section we are going through an example in detail which shows how the method works and which also demonstrates most of its advantages.

Example 3. Fig. 3 shows a tableau that proves the following formulae to be tautology.

$(\forall x)[g(f(x))\approx c \quad (f(x)\approx g(x) \quad \neg(x\approx a))]\rightarrow g(g(a))\approx g(f(b))$.

$\neg((\forall x)[g(f(x))\approx c \quad (f(x)\approx g(x) \quad \neg(x\approx a))]\rightarrow g(g(a))\approx g(f(b)))$ √

$(\forall x)[g(f(x))\approx c \quad (f(x)\approx g(x) \quad \neg(x\approx a))]$ √

$\neg(g(g(a))\approx g(f(b)))$

$g(f(x))\approx c \quad (f(x)\approx g(x) \quad \neg(x\approx a))$ √

$g(f(x))\approx c$

$f(x)\approx g(x) \quad \neg(x\approx a)$ √

$f(x)\approx g(x)$ $\qquad\qquad$ $\neg(x\approx a)$

*

Fig.3 Expanded tableau after first stage

The first stage of tableau expansion yields the tableau shown in Fig.2. By applying the standard tableau rules, formula and are derived from , from , and and from , and from . The right branch is close The expansion using logical rules is finished, but the tableau cannot be closed by the inequality . (the substitution {a/x} has already been applied).

The formulae marked with an asterisk are no longer present on the tableau. After the first stage of tableau, the expansion using logical rules is finished, but the tableau cannot be closed. Let us the letter B to the open branch. Then B is converted into the following data structure:

E(B)={ g(f(x))≈c ,f(x)≈g(x)} P(B)={ g(g(a))≠g(f(b))}

In the second stage of tableau expansion, the algorithm tries to find unifiable elements in the sets $<g(g(a))>^n_B$ and $< g(f(b))>^n_B$.

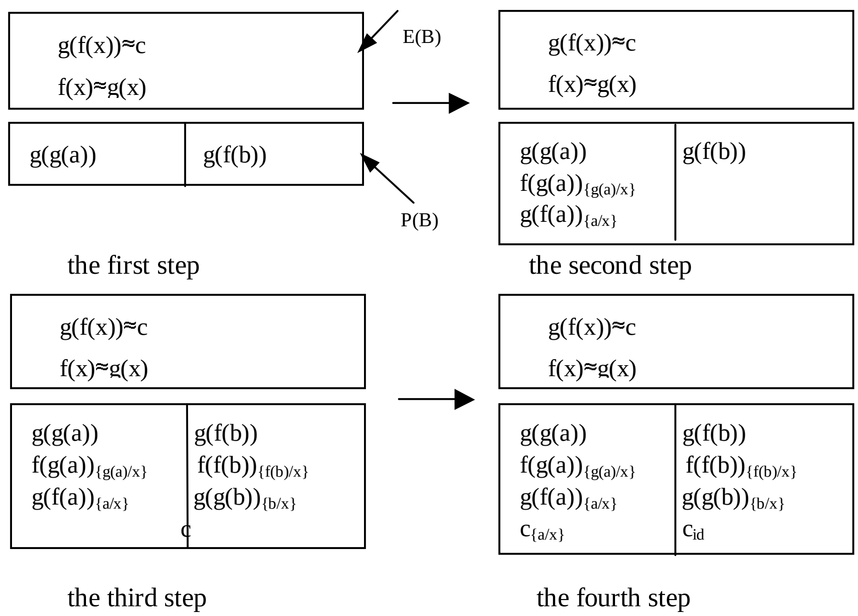

Fig. 5 Second stage of tableau expansion

In figure 3, set P(B) is separated two parts in the first step. In the second step by equation the terms f(g(a)) and g(f(a)) are derived from g(g(a)), the required substitutions is {g(a)/x} and {a/x}. In the third step by equation the terms f(f(a)) and g(g(b)) are derived from $<g(f(b))>^1_B$, the required substitutions is {f(b)/x} and {b/x}, the term c is derived from g(f(b))using equation . In the fourth step by equation the terms c is derived from g(f(a)). Now the disjunction g(g(a))≠g(f(b)) and thus branch B can be closed using the compatible elements $c_{\{a/x\}}$ $<g(g(a))>^2_B$ c_{id} $< g(f(b))>^1_B$, then the tableau is closed.

In the example 3, if the same formula was to be proven to be a tautology using Fitting's method, first of all, the γ-limit q would have to be raised and more branches would have to be closed in the subtree for T ϕ. In addition, more equality applications would be necessary to be prove the inequality g(g(a))≠g(f(b)) to be unsatisfiable. If the same heuristic was used to select the terms and equalities that we used for our implementation, the about 15 equality applications would be needed. Several times free variable substitutions would be applied that prevent the branch from being closed resulting in backtracking.

5 Implementation

To study the effectiveness of the algorithm, we have implemented the algorithm on a PC workstation using PROLOG. It is crucial to use most general substitutions to build uo equivalence classes leads to a search tree for side of an inequality. In the implementation, we have used breadth-first-search , i.e. all branches are searched simultaneously. Breadth-first-search is much more powerful, because any heuristic can be used to push ahead the search in some of the branches more quickly than in others. Since breach-first-search cannot be based on PROLOG's backtracking, it is, however, slightly more difficult to implement in PROLOG. In general, It will take less than a second to solve a tableau with equality. For example 1, procedure of implementation is as follows:

```
Beginning search for closure with equality of branch b1
Equalities extracted from branch:
1:  [ ]  b = c
2:  [ ]  c = b
3:  [x1]  g(f(x1)) = x1
4:  [x1]  x1 = g(f(x1))
Disjunctions extracted from branch:
(1,1):  g(g(a)) ≠ a
(1,2):  b ≠ c
(1,2,1,0): (0,1) b [ ]
(1,2,r,0): (0,1) c [ ]
(1,1,1,0): (0,3) g(g(a)) [ ]
(1,1,r,0): (0,1) a [ ]
(2,1): a ≠ x2 closed by [[x2 = a]]
(2,1,r,0): (0,1) a [ ]
(2,1,1,0): (0,1) x2 [ ]
Branch closed by one of its inequalities with [x2 = a]
------------------------------
Beginning search for closure with equality of branch b2
Equalities extracted from branch:
1:  [ ]  b = c
2:  [ ]  c = b
3:  [ ]  g(a) = f(a)
4:  [ ]  f(a) = g(a)
5:  [x1]  g(f(x1)) = x1
6:  [x1]  x1 = g(f(x1))
Disjunctions extracted from branch:
(1,1):  g(g(a)) ≠ a
(1,2):  b ≠ c
(1,2,1,0): (0,1) b [ ]
(1,2,r,0): (0,1) c [ ]
(1,1,1,0): (0,3) g(g(a)) [ ]
(1,1,r,0): (0,1) a [ ]
(1,1,1,0): (0,3) g(g(a)) -3-> (1,1,1,1): (0,3) g(f(a)) [ ]
(1,1,1,0): (0,3) g(g(a)) -6-> (1,1,1,2): (-2,5) g(f(g(g(a)))) [ ]
(1,1,1,0): (0,3) g(g(a)) -6-> (1,1,1,3): (-2,5) g(g(f(g(a)))) [ ]
(1,1,1,0): (0,3) g(g(a)) -6-> (1,1,1,4): (-2,5) g(g(g(f(a)))) [ ]
(1,2,1,0): (0,1) b -1-> (1,2,1,1): (0,1) c [ ]
(1,2,1,0): (0,1) b -6-> (1,2,1,2): (-2,3) g(f(b)) [ ]
Ineq. (1,2) closed by c = c [ ]
(1,1,r,0): (0,1) a -6-> (1,1,r,1): (-2,3) g(f(a)) [ ]
Ineq. (1,1) closed by g(f(a)) = g(f(a)) [ ]
Disj. (1) closed with [ ]
------------------------------
-------------PROOF--------------
Branch     Backtracking     Time     γ-limit
  2             0           0.423s      1
------------------------------
```

6 Conclusion

In this paper, based on adding new tableau expansion rules, a new tableau algorithm with equality, called 'two separate tableau', is proposed. In this algorithm, we adopted the idea of separating the tableau expansion into two stages. In the first stage the standard rules are applied until the tableau is exhausted. Thus, in the second stage, it is possible to restrict equality applications and to avoid the generation of useless new formulae. It is possible to restrict search space by using the method based on the transformation into disjunctions of inequalities and the calculation of equivalence classes. In the implemention, it can be improved for efficiency raised in the follow two: Firstly, we can improved the algorithm of heuristic in order to raise the speech of calculation of equivalence classes; Secondly, we can store the result of calculation of equivalence classes by the structure of tree which has a lot of peculiarities that can be used to improve efficiency.

References:

[1] Liu Xu-hua and Sun Ji-gui, Generalized Resolutioon and NC-Resolution, Journal of Computer Science and Technology, China.

[2] Sun Ji-gui and Liu Xu-hua, NC-RUE-NRF resolution, Journal of Computer Science and Technology, China.

[3] Richard C.Jeffrey. Formal Logic. Its Scope and Limits. McGraw-Hill, New York, 1967.

[4] Melvin Fitting, First-Order Logic and Automated Theorem Proving, New York:Springer Verlag, 1996

[5] Melvin Fitting,Types and Tableau New York Springer Verlag 2000

[6] B.Beckert and C.Pape, "Incremental Theory Reasoning Methods for Semantic Tableaux. In P.Miglioli", U.Moscato, D.Mundici, and Related Methods, Palermo, Italy, LNCS 1071, pages 93-109, Springer, 1996.

[7] B.Beckert, "Semantic Tableaux with Equality". Journal of Logic and Computation, &(1) : 39-58, 1998.

[8] B.Beckert and R. Hähnle. Analytic tableau,In W.Bibel and P.H. Schmitt, editors, Automated Deduction – A Basis for Application, colume 1. Kluwer, 1998.