AUTOMATED REASONING

SLIDES 13:

EQUALITY IN TABLEAUX
Basic use of Equality in Tableaux
Use of Equality in ME
Theory Reasoning and Tableaux

KB - AR - 09

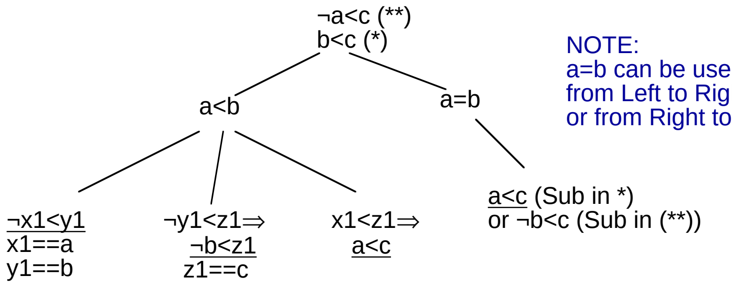
EQUALITY IN TABLEAUX

- In developing a tableau two equality rules are used:
- Reflex (EQAX1) and Substitution (also called <u>paramodulation</u>) which uses EQAX2 and EQAX3 implicitly.:

$$\begin{array}{c} a=b \\ P(\ldots,a,\ldots) \\ \hline P(\ldots,b,\ldots) \end{array} \qquad \begin{array}{c} r=s \\ P(\ldots,t,\ldots) \\ \hline P(\ldots,s\theta,\ldots)\theta \end{array}$$

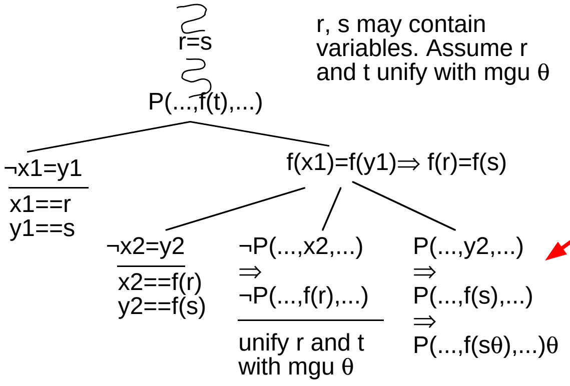
where rθ=tθ
and θ is mgu
of r and t.

Example: (1) a<b ∨ a=b (2) ¬ a<c (3) b<c (4) ¬x<y ∨ ¬y<z ∨ x<z


¬a<c (**)
b<c (*)
a<b
a=b
¬x1<y1
x1==a
y1==b
¬y1<z1⇒
¬b<z1
z1==c
x1<z1⇒
a<c
a<c (Sub in *)
or ¬b<c (Sub in (**))

NOTE:
a=b can be used
from Left to Right
or from Right to Left

Simulation of (Free variable) Tableau Equality Rules using Equality Axioms


r=s
P(...,f(t),...)
¬x1=y1
x1==r
y1==s
f(x1)=f(y1)⇒ f(r)=f(s)
¬x2=y2
x2==f(r)
y2==f(s)
¬P(...,x2,...)
⇒
¬P(...,f(r),...)
unify r and t
with mgu θ
P(...,y2,...)
⇒
P(...,f(s),...)
⇒
P(...,f(sθ),...)θ

r, s may contain variables. Assume r and t unify with mgu θ

This remaining open branch contains exactly the result of using the substitution rule

1. generate required equality by EQAX2

2. Apply appropriate EQAX3

<u>Models including the Equality Literal:</u>

Recall from Slide 12dii that in a *normal* model the equality predicate is interpreted as identity and hence if p=q is true, then p and q must be interpreted as the same domain element. Alternatively, Herbrand models that satisfy the basic requirement of substitutivity can be used and as far as satisfiability is concerned the two approaches are equivalent.

The completeness proof for tableau involved constructing a saturated tableau from some consistent set and then constructing a model from the saturated tableau. A saturated tableau is one in which every rule is applied in every branch in every possible way. For the equality substitution rule, notice that it can be restricted to apply only to ground literals in the tableau. Substitution into sentences with quantifiers can be delayed until after the quantifier has been eliminated and the resulting sentence has been reduced to literals.

In order to show that the constructed model, which was derived from the literals in an open saturated branch, was indeed a model, a complexity ordering based on the length of formulas was used. When using equality substitution this ordering must be extended to literals. However, how might this be done? eg if the equation a=b and literal P(b) are in a branch, then P(a) can be derived, but in what way is P(a) *smaller* than P(b)? To answer this requires to consider orderings on terms as well as sentences. We will return to the topic in Slides 16 and 17. But notice that the model that is found will have as domain the terms occurring in the branch and will definitely not be a normal model. Instead, the model will satisfy the EQAX and be an E-model.

Controlling Equality Substitution in Tableaux

The most difficult aspect of dealing with equality is in *controlling* substitution as there are usually many ways in which it can be applied in a tableau branch.

Possible Systematic Method1:
- Form a tableau to some limit

(eg allow each universal rule to be expanded once and then allow a maximum number of "extra" applications.)

- Ignore substitution using equality literals, but allow those branches that can close in the usual way to do so.

- For each unclosed branch apply equality rules to equations in it in order to force a closure (see example on right).

P(a)
¬P(b)
Q(u1,f(b))
¬Q(c,g(b))
b=c
f(c)=g(c)

Branch will close if can show (a=b) or (u1=c & f(b)=g(b))

---

b=c (equalities from
f(c)=g(c) branch)

a≠b (from negated goal)
u1≠c ∨ f(b)≠g(b)
(from negated goal)

Close? Yes, if u1==b

Controlling Equality Substitution in Tableaux:

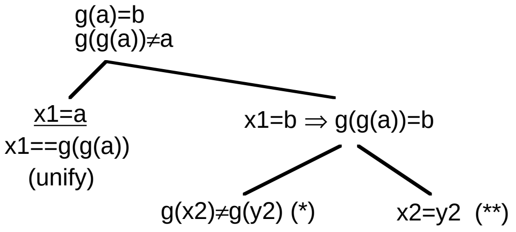
The approach on 13bi, called Systematic Method 1, develops a tableau to a maximum depth and allows, if possible, branches to be closed in the usual way. If there are any open branches remaining, which also contain equations, an attempt is made to find a contradiction using the equations. Potential closure between 2 literals is made, subject to the constraint that the arguments can be made equal. e.g. P(a,f(X)) and ¬P(b, g(b)) would be complementary, if a=b and g(b)=f(X) (for some X) could be derived. (This is quite similar to the RUE refinement.) These can be derived in many ways; they could be refuted by ordinary resolution and substitution (paramodulation) using the equations E in the branch: derive closure from E+{¬a=b}+ {¬g(b)=f(X)}; or by adapting the rewriting methods for reasoning with equality (see Slides 14 onwards). A third way is to systematically enumerate equivalence classes of terms from the equations in the branch in the hope that both sides of an equation will be contained in one class.

The equivalence class approach applied to the open branch in 13bi would enumerate the classes {b,c, ...} and {f(c), g(c), f(b), g(b), ...}, from which f(b)≠g(b) is refuted.

As shown on Slide 13aii using equations in tableaux can be simulated using the equality axioms. An approach to their control would be to incorporate this simulation within the strategy used to develop the tableau. For example, if ME is in use then a ME simulation is used, or if KE, were in use, then a KE simulation would be used. (Actually, no one has done the latter yet, to my knowledge.)

EXAMPLE

(Use systematic method 1)

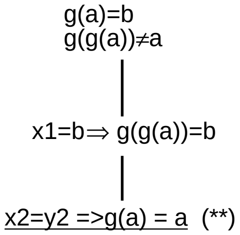


Given:
x=a ∨ x=b
g(x)≠g(y) ∨ x=y
g(a)=b
g(g(a))≠a

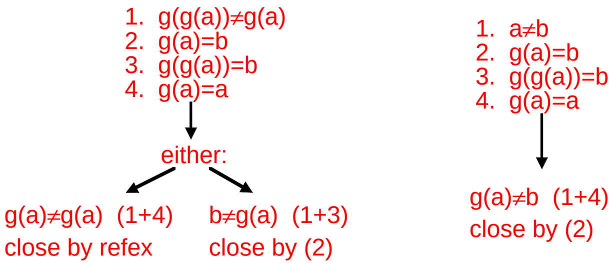
a) Use each non-unit clause a maximum of once in each branch.
b) Can close first branch normally by unification (match with g(g(a))≠a).
c) Can close at (*) if g(y2)=b is shown **and**
either x2==g(a) (match with g(g(a))=b),
or x2==a (match with g(a)=b)

To show g(y2)=b, set y2==g(a) or y2==a; unify x2 - Leads to 4 possibilities:
di) y2==g(a), x2==g(a): cannot refute g(a)=g(a).
dii) y2==g(a), x2==a: ==> refute a=g(a) (see e on next slide)
diii) y2==a, x2==a: cannot refute a=a.
div) y2==a, x2==g(a): ==> refute g(a)=a (see e)

EXAMPLE continued



e) From 13biii one remaining branch (**) to close: Can close if
(ei) g(g(a))=g(a) can be shown
(match g(a)=a with g(g(a))≠a), or if
(eii) a=b can be shown
(match g(g(a))=b with g(g(a))≠a)
See steps below – add to equations
either a≠b or g(g(a))≠g(a)



Using Systematic Method 1 in Tableaux:

The example on 13biii and 13biv illustrates the method of forming a tableau to some limit (here using each clause a maximum of once in each branch) and then trying to close branches using equations. There are two open branches and two possible closures for the first of these: between g(a)=b and g(x2)≠g(y2) or between g(g(a))=b and g(x2)≠g(y2). One can obtain closure either if x2==a and g(y2)=b can be derived, or if x2==g(a) and g(y2)=b can be derived. i.e. refute ¬g(y2)=b using the set of equations {g(a)=b, g(g(a))=b}, which is easy: y2==a or y2==g(a). There is another possibility, to close g(x2)=g(y2) by reflex, but this yields x2=x2 in the second open branch which cannot be refuted.

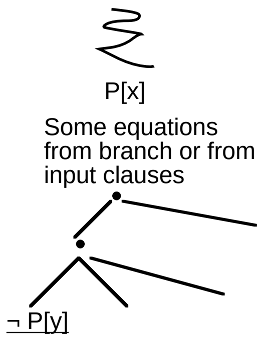
For the second open branch, two of the substitution pairs result in a=a or g(a)=g(a), which clearly cannot be refuted as they are instances of (Reflex). The other two substitution pairs both result in g(a)=a and the branch can be closed if g(g(a))≠a matches either g(g(a))=b or g(a)=a or g(a)=b. The first of these requires b=a to be shown using {g(a)=a, g(a)=b}, which is clearly possible; the second requires g(g(a))=g(a) to be shown, again using {g(a)=a, g(a)=b}. Again this is easy. The third requires to show g(g(a))=g(a) and a=b, which is done as before. (See 13biv.)

Exercise: Show these things).

The equivalence class approach applied to the first open branch of the tableau on 13biii would enumerate the class {g(a), b, g(g(a)), g(b), g(g(b)), g(g(g(a))), ...}, from which g(x2)≠g(y2) is refuted. E.g. put x2=g(a), y2=b.

It is clear that there are many and various possibilities when using equations and that the search space can become very large. Next we see if using the ME strategy is any better.

ME TABLEAUX AND EQUALITIES



P[x] and P[y] are complementary modulo equations in the branch.

Possible Systematic Method 2
The equations are used in substitution (paramodulation) steps to enable P[x] and P[y] to match. There may be bindings made which are propagated as usual.

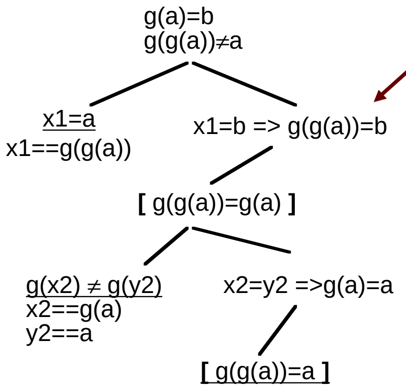
(i) If an equality atom is the selected literal P[x] and it does not resolve in the usual ME-way, then closure is sought between two literals L and M, which may either be ancestors of P[x] or from input clauses, by applying P[x], and possibly other equality atoms in the branch or from input clauses, to L and M so they become complementary.

(ii) If the selected literal P[x] is not an equality atom and it does not resolve in the usual ME-way, then attempt to close by applying equality atoms to P[x] and to another literal M, where M and the equalities are either ancestors or from an input clause.

EXAMPLE

(Use Systematic method 2)

Given: x=a ∨ x=b   g(g(a))≠a   g(a)=b   g(x)≠g(y) ∨ x=y



Use equalities and try to close either with g(g(a))≠a or with g(x1)≠g(y1)

Second case can be done:
Use g(a)=b ==> g(g(a))=g(a)
(replace b in g(g(a))=b by g(a))

Use equalities and try to close with g(g(a))≠a.

g(a)=a in g(g(a))=g(a) ==> g(g(a))=a

Problem is that there are *many* different possibilities for closure.
Exercise: find other closures for the right side of this tableau using systematic method 2. Hint: can derive **[**g(b)=b**]** and then **[**g(a)=g(b)**]**

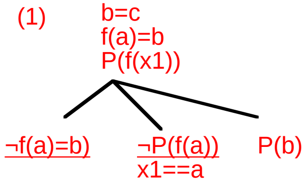
Using Systematic Method 2 in Tableaux:

On Slide 13cii in the second branch of example the aim is either to derive a literal complementary to g(x2)≠g(y2) using g(g(a))=b, or complementary to g(g(a))≠a. The slide shows the first: substitute g(a) for b in g(g(a))=b==> g(g(a))=g(a). For the remaining branch you can easily derive g(g(a))=a from g(g(a))=g(a) and g(a)=a. This is not the only closure using systematic method 2. It's easy also to derive first g(b)=b and then g(a)=g(b), which also closes with g(x2)≠g(y2), but binding x2==a, y2==b , leading to b=a in the final branch. That branch easily closes also, by deriving g(g(a))=a from g(g(a))=b. Note that the derived equality g(g(a))=g(a), derived for branch 2 is used in the closure of branch 3.

If the equality axioms are to be used explicitly in a ME tableau, then one might consider restricting their use. E.g. perhaps an equation can be constrained to be used only if it already occurs in a branch. The clauses P(a), ¬P(b) and a=b , with P(a) as top clause demonstrate that this restriction would <u>not</u> be complete. Other restrictions have been suggested, such as restricting the use of equations unless they are in a clause with all positive literals. Again this won't be complete. (**Exercise**: Find a counterexample.)

So perhaps the axioms themselves can be used in the ME way. If an equation is a leaf literal, then one can match a suitable EQAX3 instance with the equation and with either a literal in the same branch or from another clause. This will give the effect of paramodulation. Sometimes, instances of EQAX2 are required and these can be incorporated too. If an equation is not a leaf literal, still an instance of EQAX3 can be used, this time matching the leaf literal which is to be paramodulated into, and the equation must come either from the same branch or from another input clause. Sometimes it is possible to apply several paramodulation steps to make a leaf literal match the desired complement.

Simulation in ME using Equality Axioms



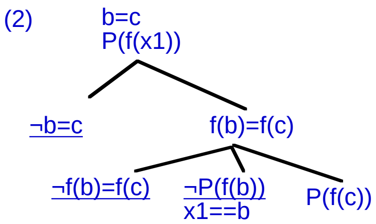
(1)
b=c
f(a)=b
P(f(x1))

¬f(a)=b
¬P(f(a))
x1==a
P(b)

In (1) use EQAX3 explicitly in the ME way to simulate the substitution of f(a)=b into P(f(x1)).

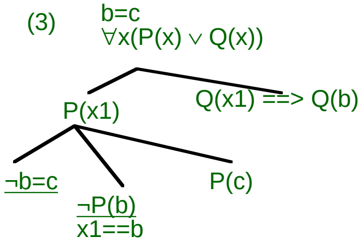
In (2) use EQAX2 and EQAX3 in the ME way to simulate the substitution of b=c into P(f(x1)).

If either the required equation or literal being substituted into is not yet in the branch, can force it to be so by introducing the appropriate input clause, as in (3). Simulate substitution of b=c into P(x1).



(2)
b=c
P(f(x1))

¬b=c
f(b)=f(c)
¬f(b)=f(c)
¬P(f(b))
x1==b
P(f(c))

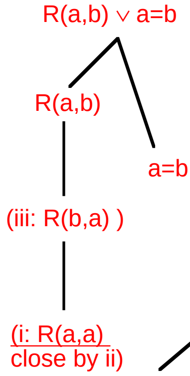


(3)
b=c
∀x(P(x) ∨ Q(x))

P(x1)
Q(x1) ==> Q(b)
¬b=c
¬P(b)
x1==b
P(c)

THEORIES AND TABLEAUX - Some Examples

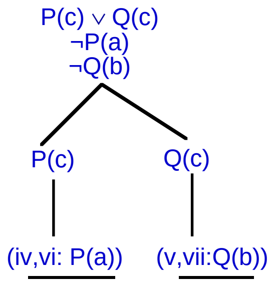
- A *theory* is any consistent set of sentences.
- e.g. equality axioms, relational theory, sorts and types, taxonomic theory



R(a,b) ∨ a=b
R(a,b)
a=b
(iii: R(b,a))
(i: R(a,a)
close by ii)

Theory T1

i) R is transitive
ii) R is irreflexive
iii) R is symmetric



P(c) ∨ Q(c)
¬P(a)
¬Q(b)
P(c)
Q(c)
(iv,vi: P(a))
(v,vii:Q(b))

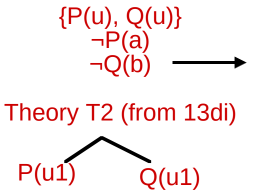
Theory (T2)
iv) c≤a
v) c≤b
vi) x≤y ∧P(x) → P(y)
vii) x≤y ∧ Q(x) → Q(y)

In both cases, the theory is used to derive new atoms (indicated in brackets) which close a branch.
c.f. with earlier method of incorporating EQAX.

Theory Tableaux - How does it work?

- Let B be a tableau branch, Th be a theory and K1,…,Km be *ground* literals in B
- Suppose Th + K1,…,Km |= R1 ∨ … ∨ Rn then add clause (R1 ∨ … ∨ Rn) to B
- Equivalent to Th + K1,...,Km, ¬R1,...,¬Rn |= ⊥
  i.e. find literals to close B and add the disjunction of their complements
- Often, n is restricted to be =0 or 1
- If n=0, can close B if Th+K1,…,Km are inconsistent
- If n=1 just add a single literal to B

eg On 13di consider branch with P(c), ¬P(a) and c≤a and x≤y ∧P(x) → P(y) in Th
Either: let K1=P(c) and derive R1=P(a),
Or: let K1=P(c), K2=¬P(a) and close branch



{P(u), Q(u)}
¬P(a)
¬Q(b)
→

Theory T2 (from 13di)

P(u1)
Q(u1)

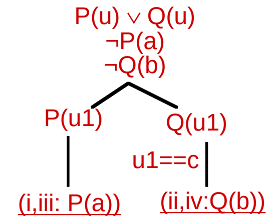
More generally:

¬P(a)
P(u1)
c≤a
x≤y ∧P(x) → P(y)
→

Either:
K1=P(u1) and Th + K1θ |= R1=P(a)
where θ = {u1==c}

Or:
K1=P(u1), K2= ¬P(a) and
Th +P(u1)θ + ¬P(a) closes branch

General Case of Theory Reasoning



P(u) ∨ Q(u)
¬P(a)
¬Q(b)
P(u1)
Q(u1)
u1==c
(i,iii: P(a))
(ii,iv:Q(b))

Theory T3:
(i) z≤a
(ii) c≤b
(iii) x≤y ∧ P(x) → P(y)
(iv) x≤y ∧ Q(x) → Q(y)

T3 |= ∀u1(P(u1) → P(a)),
or T3 +∀u1.P(u1) |= P(a)

And for θ = {u1==c}
T3 |=(P(u1) → P(a))θ,
or T3 + P(u1)θ + ¬P(a)
is inconsistent

For general clauses (here for case n≤1 only):

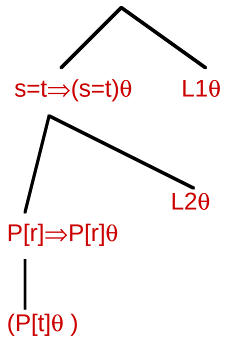
- If literals K1,…,Km occur in a branch
and Th |= ∀[(K1∧…∧Km → R1)θ],
where ∀ closure is over free variables in {R1θ,Kiθ},
and θ is *minimal substitution to achieve this*,
then (i) add (R1)θ to branch,
and (ii) propagate θ to remaining open branches

- The main property is equivalent to
Th + ∀[(K1 ∧… ∧ Km)θ + ¬R1θ] is inconsistent

- As a special case:
If Th + ∀[(K1 ∧… ∧ Km)θ] is inconsistent,
then close branch and propagate θ to remaining open branches.
θ is minimal substitution to achieve this.

Paramodulation is Theory Reasoning (!):

Given clauses: L1 ∨ s=t   L2 ∨ P[r]   Theory is EQAX
Suppose also, that rθ = sθ, so we expect the result P[t]θ ∨ L2θ ∨ L1θ



K1, K2 are s=t, P[r]
R1 is P[t]

Using criterion on 13diii, find θ s.t.
Theory |= ∀ [(s=t ∧ P[r]) → P[t]) θ]

Or equivalently: Theory + (s=t ∧ P[r])θ |= P[t]θ
where θ is minimal substitution for free variables in s, t and r to satisfy the criterion

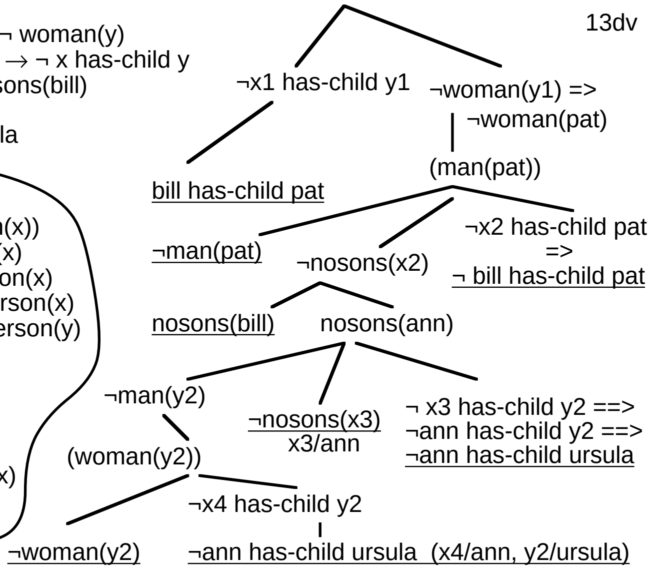
(also equivalently:
Theory+ (s=t ∧ P[r] ∧ ¬P[t])θ closes branch)

This is exactly what paramodulation achieves.

(Remember, θ is applied to whole of tableau and leaf literals in open branches are the resolvent, or in this case paramodulant.)

Example:
¬ x has-child y ∨ ¬ woman(y)
man(y)∧nosons(x) → ¬ x has-child y
nosons(ann) ∨ nosons(bill)
bill has-child pat
ann has-child ursula

<u>(Type) Theory</u>:
¬(man(x) ∧woman(x))
man(x) → person(x)
woman(x) → person(x)
x has-child y →person(x)
x has-child y → person(y)
person(pat)
person(bill)
person(ann)
person(ursula)
person(x) →
man(x) ∨ woman(x)



Summary of Slides 13

1. Equality reasoning can be incorporated into tableau, either standard tableau or free variable tableau or ME tableau.

2. In standard tableau the equality rule allows to derive new ground literals using equality substitution; in free variable tableau it allows to derive new literals, possibly applying a unifying substitution (also called paramodulation).

3. Use of the equality axioms can be simulated within ME tableau, by using equality axioms explicitly.

4. Closure in a branch with equality literals can be (i) between complementary literals, or (ii) by deriving from equality atoms new equality atoms that will enable two literals to become complementary by substitution. This is done by matching the potentially complementary literals and trying to show the required equations using the equality atoms in the branch.

5. Usually, a tableau is developed to some depth, closing branches normally if possible, and then attempting to close remaining branches as in 5.

6. Most methods using equality in tableau are quite difficult for humans and lead to large seach spaces.

---

6. An alternative approach can be employed in ME tableau: equality atoms in a branch are used to make two literals complementary. In case a leaf literal L is not an equality, then equality atoms (either in the branch or from new instances of input clauses) are used to derive ¬L. If L is an equality atom then L must be used in the derivation, and the two complementary literals may be from the branch or from new instances of input clauses. The depth of a tableau would normally be limited, including the extra depth to derive the complementary literal(s).

8. Paramodulation is a special case of Theory Reasoning. Theory reasoning can be useful for other theories, such as some theory of ordering, or theories of subsets, or lists, for example. Reasoning with theories is easier using tableau, which is the only method described, though it was originally proposed as a resolution refinement (Stickel).

Question for next week

When simplifying an equation you are using equality reasoning.

How can the task of simplifying (1+ x) - 4 = 2 to give a binding for x, namely x==5 be recast as a paramodulation problem.

What reasoning steps do you use to solve the equation?
(Hint: there are more than you think, perhaps)