AUTOMATED REASONING

SLIDES 17:

KNUTH BENDIX EXTRAS (if time permits)
Failure in Knuth-Bendix Procedure
Knuth Bendix and Theorem Proving
Narrowing

KB - AR - 2013

When Knuth Bendix Completion Fails

The Knuth Bendix procedure **fails** if an equation cannot be orientated

- eg x+y = y+x leads to circular rewriting as in 2+3 => 3+2 => 2+3 ...,
  f(x, g(z)) = f(g(z), x) leads to f(g(a),g(b))=>f(g(b),g(a))=>f(g(a),g(b)).....

Avoid failure by allowing superposition to/from either side of an non-orientable equation as long as certain conditions are met to avoid non-termination.

- Can superpose l = r and s => t as long as ¬(lθ ≤ rθ);
  (θ is either mgu of l and a subterm of s, or of s and a subterm of l.)
  (¬(lθ ≤ rθ) means there are some instances for which lθ > rθ, else lθ ≤ rθ)
- \> must be total on ground terms; (i.e. any 2 ground terms can be ordered)
- when *rewriting* using l = r, must have lθ >rθ. (Ideas due to Bachmair)

- eg f(x,g(z)) = f(g(z),x), f(a,y) => y, f(y,b)=>y (use kbo based on counting terms)

<u>Cannot</u> superpose f(a,y) and f(x,g(z)) (f(a,g(z))) because f(a,g(z)) $<_{kbo}$ f(g(z),a)

<u>Can</u> superpose f(y,b) and f(g(z),x) (f(g(z),b)) because f(g(z),b) $>_{kbo}$ f(b,g(z))
gives f(b,g(z))=>g(z)

Examples of conditional Orientating (ppt)

- Can superpose l = r and s => t as long as ¬(lθ ≤ rθ);
  (θ is either mgu of l and a subterm of s, or of s and a subterm of l.)
  i.e. there are some instances for which lθ > rθ (else lθ ≤ rθ).
- \> must be total on ground terms;
- when *rewriting* using l = r, must have lθ >rθ. (Ideas due to Bachmair)

(4) x+ -x => 0 (5) x + 0 => x (6) u+v = v+u

Use kbo: s >≈ t iff # functors in s ≥ # functors in t, and 0 $<_1$ all other terms.

(5)+(6) give critical term x+0

check OK: ¬(x+0 $\le_{kbo}$ 0+x) since for some x (i.e. x≠0) x+0 $>_{kbo}$ 0+x

(In other words, if x≠0 can orient x+0 => 0+x since x $>_1$ 0

(4)+(6) give critical term x+-x

<u>not OK</u> since it's <u>*not*</u> the case that ¬(x+-x $\le_{kbo}$ -x+x);

in fact, x + -x $<_{kbo}$ -x +x as -x>x

However <u>can</u> use (6) to rewrite -a+a

-a+a $>_{kbo}$ a+-a; hence -a+a ==> a+-a ==> 0

Example using the orientation restriction

(1) n(y,y) = n(M,y) (2) n(g(u,v),x) => n(u,n(v,x))

Use kbo: s $\ge_{kbo}$ t if #symbols in s ≥ #symbols in t.

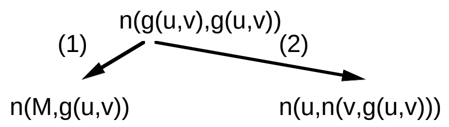
Cannot order (1): y>M==>n(y,y)>n(M,y) and y<M ==>n(M,y)>n(y,y)

Form critical pair – unify LHS of 1 with LHS 2

Check: ¬(n(g(u,v),g(u,v)) ≤n(M,g(u,v)))

In fact, n(g(u,v),g(u,v)) > n(M,g(u,v)) (LHS has more symbols than RHS, so OK)

gives new rule
n(u,n(v,g(u,v))) => n(M,g(u,v))



Note about the constraint ¬(lθ ≤ rθ)
Since ≤ is stable, ¬(lθ ≤ rθ) means that "it is not the case that lθσ ≤ rθσ for every substitution σ". Hence ¬(lθ ≤ rθ) → lθσ > rθσ for *some ground substitution* σ.

Hence it is possible to have both ¬(lθ ≤ rθ) and ¬(rθ ≤ lθ) (for different substitutions σ1 and σ2 - that is lθσ1 > rθσ1 and rθσ2 > lθσ2). In such a case the equation l=r could be used in both directions but at different times.

Informally, the method described on Slides 17ai-17aiii works because the transformation steps applied to any ground proof (using equations) to turn it into a rewrite proof by critical pair formation can be lifted to the general level. The lifted proof will not have been excluded by the restrictions:

- if lθ ≤ rθ (i.e. an excluded step) then all instances of it would lead to excluded steps too; these excluded steps could not have been part of the transformation process of the original ground derivation, leading to a contradiction.

17aiv

Paramodulation and Narrowing

17bi

Recall the definition of rewriting:

- An expression e[s] is *rewritten* by l => r if s = lθ and (e[s])=> (e[r])θ.
  (i.e. no bindings are made to vars in the term s being rewritten)

If we relax the restriction if s = lθ to sθ = lθ we obtain Narrowing

- An expression e[s] is *narrowed* by l => r if sθ = lθ and (e[s])θ=> (e[r])θ.
  (i.e. bindings *may be made to vars in the term s* that is being rewritten)

Example: (1) x+0 => x (2) x+s(y) => s(x+y) (3) y=y
s(0)+v narrows to s(0) by (1), if v==0
s(0)+v *narrows* to s(s(0)+y1) by (2) if v==s(y1), which *narrows* to s(s(0)) by (1) if y1==0

- Narrowing corresponds to using paramodulation with **oriented** equations
- Rewriting corresponds to using restricted paramodulation with **oriented** equations

Using Knuth Bendix Completion as a Theorem Prover

- Consider goals of the form ∃x[t1[x] = t2[x]] and data restricted to equations.
- The negated goal is ∀x[¬(t1[x]= t2[x])]
- This leads to ==> ¬(t1[x1]=t2[x1]) (using free variable rule)
- The two sides of the equality can be *narrowed* until a substitution is found that makes both sides equal
- The resulting inequation can then be resolved with x=x.
- The Knuth Bendix procedure can also be applied incrementally to the rewrite rules and the constrained form (of Slides 17a) used for equations that cannot be oriented. This copes both with failure and divergence.

Example 1: (1) x+0 => x (2) x+s(y) => s(x+y) (3) y=y
Use **oriented** paramodulation - ie use equations in direction of =>

Show ∃x[s(0)+x = s(s(0))] (or find x s.t. s(0)+x = s(s(0)))
¬(s(0)+x1 = s(s(0))) ==> (P 2.) ¬(s(s(0) + y1) = s(s(0))) (if x1==s(y1))
¬(s(s(0) + y1) = s(s(0))) ===> (P 1.) ¬(s(s(0))) = s(s(0))) (if y1 ==0)
¬(s(s(0))) = s(s(0))) ==> (R 3.) [ ] (x1==s(y1)==s(0))

17bii

Using Knuth Bendix Completion as a Theorem Prover (2)

Example 2: (1) g(a,b) => a (2) g(g(x,y),y) => h(y,x)
Superposition of (1) onto (2) gives g(g(a,b),b)
g(g(a,b),b) =>* a (use (1) twice) and => h(b,a) (by (2)) giving (3) h(b,a) => a

Suppose the goal is ∃z[g(a,z)= h(z,a)]. (ie find a z s.t. g(a,z) = h(z,a))
Negated, this is ∀z [¬(g(a,z)=h(z,a))] (leading to ¬(g(a,z1)=h(z1,a)))

Using the rules (1) and (3) we get ¬(a = h(b,a)) (by (1) and binding z1==b)
and then ¬(a = a) (by (3)), which resolves with x=x.

The derivation yields also the witness z1 (here z1==b)

17biii

Example 3 17biv

1 n(x,x) = n(M,x) 2 n(g(u,v),x)=>n(u,n(v,x)) 4 x= x
3 n(z,z) ≠ z (negation of goal “find z s.t. n(z,z)=z”
i.e. ¬(∃z. n(z,z)=z), becomes ∀z. n(z,z)≠z)

Use kbo: s$\geq_{kbo}$t if #symbols in s ≥ #symbols in t (similar to slide 17aiii)

(5) (1+3) n(M,z) ≠ z
(Check: ¬ (n(x,x) ≤ n(M,x)), True - if x>M then n(x,x)>n(M,x)

(6) (1+2) n(u,n(v,g(u,v))) => n(M,g(u,v)) (see 17aiii for details of this step)

(7) (5+6) n(M,g(M,v1)) ≠ n(v1,g(M,v1)) (u==M and z==n(v1,g(M,v1)))

(8) (7+4) [] (v1 ==M)

Hence {z== n(v1,g(M,v1)) == n(M,g(M,M)) }

Question: Are there any other solutions?

Summary of Slides 17

17ci

1. The Knuth Bendix procedure normally has three outcomes: success (a confluent and terminating set of rules is produced), failure (some rule cannot be oriented) and divergence (there are an infinite number of rules). Leads to consider how to deal with failure.

2. In the unfailing KB procedure, superposition is allowed between l=r and s=>t if lθσ>rθσ is for some substitution σ, where θ is the unifying substitution of the superposition step.

3.The Knuth Bendix procedure can be used as a theorem prover. The goal (often of the form ∃x[t1[x] = t2[x]]) is negated to give ∀x[t1[x] ≠ t2[x]]. Knuth Bendix is applied to generate rewrite rules and they are used in narrowing steps to reduce both sides of the inequality to a common term. Resolution with x=x then gives [ ]. Even if the KB procedure diverges, interleaving of rule generation with narrowing can give a solution.

START of OPTIONAL MATERIAL (SLIDES 17)

Non-termination: a special case
Oriented paramodulation and resolution

Example of a special case of non-termination 17di

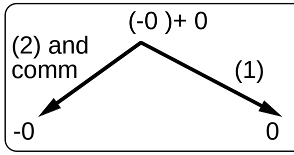
For a *commutative and associative operator* (eg +, or set union), there is a special unification algorithm called *AC-unification*, which takes these properties into account during superposition. The result is always a finite number of unifiers (possibly none). (See Bundy: Computer Modelling of Mathematical Reasoning)

Commutativity and associativity can also be included when rewriting

eg x+-x =>0 together with commutativity of + allows -(b+a)+(a+b) => 0:

-(b+a)+(a+b) can be unified with x+-x with substitution x==(b+a) or x==(a+b)
by using commutativity (either once at outer level, or twice) => 0 (by the rule)

(Note that adding x+y=y+x explicitly will not necessarily allow to derive -x+x=>0 from x+-x =>0 because of the oriented restriction: -x+x ≥ x+-x is likely the case; see17aii)



eg (1) -x+x => 0 (2) 0+y => y give critical pair -0 = 0
(-x+x = x+-x (by commutativity)
then obtain substitutions x==0 and y==-0;)

adding x+y=y+x explicitly allows to derive x +-x => 0 since -x+x $>_{kbo}$ x+-x
but does not allow to derive y +0 => y since ¬(0+y $>_{kbo}$ y+0)
In fact, if y>0 then 0 +y $\leq_{kbo}$ y+0 and can't superpose as orientated restriction not satisfied ¬(0+y ≤ y+0) only holds if y=0 (when it is obviously useless!)

Oriented Paramodulation (OP)
17ei

- We can use the idea of ordering an equation to control paramodulation steps:
- Restrict paramodulation by requiring the replacing term to be **definitely not greater** than the one being replaced.
- In case an equation can be orientated (ie every instance satisfies LHS>RHS) then the restriction allows to order the equation LHS ==> RHS.

Oriented Paramodulation: l= r ∨ C paramodulates into s[u], u not a variable
if lθ =uθ and ¬ (lθ ≤ rθ), where
≤ is a *stable monotonic simplification* ordering (eg rpo, kbo).

(Method due to Hsiang and Rusinowitch CADE 8, 1986)

Example:
n(x,x)=n(M,x) and the kbo: n(x,x)<n(M,x) if x is bound to a term t<M;
n(x,x)>n(M,x) if x is bound to a term s>M;

Can apply oriented paramodulation into P(n(u,v)):
use L to R to give: P(n(M,v)) or R to L to give: P(n(v,v))

Thus θ may be u==v and must check ¬(n(v,v) ≤ n(M,v)) (OK)

Or θ may be u==M and must check ¬(n(M,v) ≤ n(v,v)) (OK)

Oriented Paramodulation (Example)
17eii

- In case an equation can be orientated (ie every instance satisfies LHS>RHS) then the restriction allows to order the equation LHS ==> RHS.
- l= r ∨ C paramodulates into s[u] (u not a variable) **if** lθ =uθ and ¬(lθ ≤ rθ)

Example: 1. x=a ∨ x=b 2. ¬ g(x) = g(y) ∨ x=y 3. ¬ (g(g(a)) = a) 4. g(a) =>b
Choose a < b < g and ≥rpo (so a<b <g(a) < g(b) < g(g(a)) , …)

5	[P (1+3)]	g(g(a))=>a ∨ ¬b=a	(put x=g(g(a)) and replace by b in 3; check ¬(g(g(a)) ≤ b) ; OK)
6	[R (3+5)]	¬ b=a	
7	[R (6+ 2)]	¬ g(b) = g(a)	
8	[P (4+3)]	¬ g(b) = a	(OK g(a) >b)
9	[R (1 + 8)]	g(b) => b	
10	[P (9+7)]	¬ b = g(a)	(OK g(b) >b)
11	[R (10+ 4)]	[]	(use symmetry)

Notes:
(i) can replace u=v by u=>v if u>v for <u>every</u> instance of u,v
– so g(a)=b becomes g(a)=>b and g(b)=b becomes g(b)=>b
(ii) ¬(lθ ≤ rθ) means "it is not true that for every ground substitution σ, lθσ ≤ rθσ"
i.e. "there is some ground substiution σ, lθσ > rθσ"

OP and Predicate Ordering
17eiii

- Oriented paramodulation can be combined with an ordering on predicate symbols (note the **largest** predicate symbol has highest priority here):
- ≤ is extended to literals as well as terms such that "=" ≤ all predicates

- C1: s=t ∨ D1 can paramodulate by oriented paramodulation into literal with largest predicate in C2 if D1 consists of predicates equal in the order to "="
- C1: E1∨ D1 and C2: ¬E2 ∨ D2 can be resolved if E1σ and E2σ are unifiable and no predicate in D1 is > E1 and no predicate in D2 is > E2.

i.e. E1/E2 use the largest predicates in C1/C2

Example: the Aunt Agatha problem

1. K(d,a), 2. d=>a ∨ d=>b ∨ d=>c, 3. H(b,d), 4. x=b ∨ <u>H(b,x)</u>, 5. ¬a =b
6. ¬<u>K(x,y)</u> ∨ H(x,y), 7. ¬<u>H(c,x)</u> ∨ x=b, 8. ¬K(a,a), 9. ¬H(x,f(x))

Order functors as f>d>a>b>c and predicates K>H>'=' (K has highest priority).

10. (1+2, P)	<u>K(a,a)</u> ∨ d=>b ∨ d=>c	15. (13+14, R)	d=>c
11. (10+8, R)	d=>b ∨ d=>c	16. (1+6, R)	H(d,a)
12. (4+9, R)	f(b) =>b	17. (16+15, P)	H(c,a)
13. (12+9, P)	¬H(b,b)	18. (7+17, R)	a=>b
14. (11+3, P)	<u>H(b,b)</u> ∨ d=>c	19. (18+5, R)	[]

17eiv

Combining Oriented Paramodulation and Predicate Ordering:

Oriented Paramodulation allows to control the use of paramodulation. It can also be combined with predicate ordering if we treat predicates as functors for the purpose of ordering. It is easiest to make the greatest predicate have the highest priority (in contrast to what we did in Slides 7, but like Otter does), and to give the = predicate lowest priority. In case paramodulation is explicitly simulated by resolution, this behaves similarly to locking the equality axioms as we suggested in Slides 12. We can also extend the use of quasi-orderings to other refinements, even if paramodulation is not involved, such as atom ordering and hyper-resolution. Some examples of using these ideas are given on Slide 17ev.

An example of an ordering of terms that's combined with a predicate ordering was given in the optional material in slides 7 (the lexicographic ordering). However, once orderings are combined also with paramodulation steps, we require that the order be a simplification order too; for instance, kbo or rpo. If < is such an order, then we can compare *two atoms* thus:

s=P(s1,...,sn)>t=Q(t1,...,tm) if
(i) P>Q in the predicate order, or
(ii) P=Q, P is not "=" and [s1,...,sn]>*[t1,...,tm], where >* is the lexicographic order based on <, or
(iii) P=Q, P is "=" and {s1,s2}>>{t1,t2} (multi-set order because = is symmetric).

Further Examples: (Extension to atom ordering)

1) P(0) 2) ¬P(x) ∨ P(s(x))

P(s(x)) > P(x) because s(x) > x (using any simplification ordering)
so P(s(x)) is the literal that must be selected in (2).
There are then no ordered resolvents between these clauses.

Group Theory problem:

1. f(a,b) => c 2. ¬f(b,a) = c 3. f(x,x) => e 4. f(x,e) => x
5. f(e,x) => x 6. f(f(x,y),z) => f(x, f(y,z))

Use kbo based on length of terms.

7. (1+6, P) f(a, f(b,z)) => f(c,z)
8. (3+6+5, P) f(x,f(x,z)) => z
9. (1+8, P) f(a,c) => b
10. (9+6, P) f(a, f(c,z)) => f(b,z)
11. (10+ 3+4, P) f(b,c) => a
12. (8+11, P) f(b,a) => c
13. (2+12, R) [ ]

Completeness of the method is shown in Hsiang and Rusinowitch, CADE- 8.

Summary of Optional material in Slides 17

1. There are special procedures for the particular case of an associative and commutative operator, eg +, in which the properties are built into the unification.

2. Oriented paramodulation restricts paramodulation according to some term ordering. It can be combined with resolution restricted by atom ordering. An equation l=r may be used for paramodulation from l to r as long as there are some instances such that lθ>rθ. Otherwise, r≥l and the rule must be used in that direction.