Tactic Based Theorem Proving

A state of a prover can be viewed as having some assumptions and a goal.

$$A_1, \ldots, A_n \vdash G$$

Initial state $\quad . \vdash G$

Final states $\quad \ldots, G, \ldots \vdash G$

There are 2 basic ways in which a prover can proceed

a) growing the set of assumptions and keeping the goal intact

- This is called <u>forward chaining</u>
- It is a form of exhaustive search
- This is how Nelson-Oppen prover works

Consider for equalities

$$E_1, \ldots, E_n \vdash E$$

generate <u>all</u> consequences of $E_1, \ldots, E_n$ for subterms of $E_1, \ldots, E_n, E$ and see if you get to generate E

b) reducing the goal G to simpler subgoals $G_1, \ldots, G_m$ such that

$$G_1 \wedge \ldots \wedge G_m \Rightarrow G$$

Then split the current state

$A_1, \ldots, A_n \vdash G$ into

m states $A_1, \ldots, A_n \vdash G_i \quad i = 1, \ldots, m$

- This works well when m is 1 or very small
- This is called <u>backward chaining</u>
- Prolog works like this

<u>Obs</u>

- Consider a theory with an inference rule

$$\frac{H_1 \ldots H_m}{C}$$

Using this rule for forward chaining means to find an instantiation of variables in $H_1, \ldots, H_m, C$ (say ϕ) such that

for all $i = 1, \ldots, m \; \exists j \; \phi(H_i) = A_j$

then move to the state

$$H_1, \ldots, H_n, \phi(C) \vdash G$$

Example $\dfrac{P \wedge Q}{P}$

State $\ldots, A \wedge B, \ldots \vdash G \longrightarrow \ldots, A \wedge B, A, \ldots \vdash G$

- forward chaining is appropriate for rules where

$$Var(C) \subseteq \bigcup_{i=1,m} Var(H_i)$$

(in that case by instantiating all H_i, C is fully instantiated)

- Using a rule $\dfrac{H_1 \ldots H_m}{C}$ for backward chaining means to find Φ such that $\Phi(C) = G$. Then generate the m subgoals

$$A_1, \ldots, A_n \vdash \Phi(H_1)$$
$$\vdots$$
$$A_1, \ldots, A_n \vdash \Phi(H_m)$$

Example $\dfrac{P \quad Q}{P \wedge Q}$

State $\ldots \vdash A \wedge B \longrightarrow \ldots \vdash A$
$\ldots \vdash B$

- backward chaining is appropriate for rules where

$$\bigcup_{i=1,m} Var(H_i) \subseteq Var(C)$$

For most theories a combination of forward and backward chaining is most appropriate.

There is no single strategy (tactic) that works best for all problems
- when to use forward chaining or backw.
- on which assumption
- which goal to try first
- which rule to use.

Prolog uses a fixed strategy and is quite limited.

Edinburgh LCF (Milner 1970) was an extensible theorem prover.

The user can program <u>tactics</u> in a Meta Language (this is how ML was born)

tactic → a backwards chaining step
conversion → a forward chaining step
rewriting

- To support the programming of tactics, there are <u>tacticals</u> that can combine basic tactics into larger tactics
- A tactic can fail (it is not applicable)

Examples of tacticals

tac_1 THEN tac_2 – try tac_1 and ~~if it fails~~ followed by tac_2

tac_1 ORELSE tac_2 – try tac_1 and if it fails then try tac_2

REPEAT tac – repeat tac until it fails

```
fun ORELSE (tac1, tac2) g =
    tac1 g handle _ => tac2 g

fun THEN (tac1, tac2) g =
    ~~fold List.append [] (ta~~
    fold (fn acc sg => (tac2 sg) @ acc) [] (tac1 g)

fun REPEAT tac g =
  ((tac THEN (REPEAT tac)) ORELSE (fn g => [g])) g
```

- Tacticals can be used to specify the control mechanism
  - If $tac_1, \ldots, tac_n$ are basic tactics corresp. to Prolog clauses $c_1, \ldots, c_n$ then
    REPEAT (tac_1 ORELSE tac_2 ... ORELSE tac_n)
    is the Prolog control mechanism

In tactic based theorem prover a powerful language is available to program the control mechanism.

- for example some measure of cost can be used to select the next goal to be proved. (best first)

Edinburgh LF, Nuprl (Cornell), Isabelle, HOL are examples of tactic based theorem provers.

Introduction to Isabelle (Paulson, 1990)

- Isabelle a proof state is a set of subgoals.
- If proven, they should entail the original goal G
- A state can be written as

$$[G_1, \ldots, G_n] \Rightarrow G$$

(at any given moment Isabelle has proved the theorem $G_1 \wedge \ldots \wedge G_n \Rightarrow G$, so all it is left to do is to prove all of $G_1, \ldots, G_n$

- Initial state is $[G] \Rightarrow G$

- An Isabelle tactic maps a proof state to another one
  - it looks at the entire proof state
- A tactic can return a list of states → choices.