Proof Planning

Erica Melis, University Saarbrücken
http://jswww.ags.uni-sb.de/~melis/

Alternative to classical methodology in theorem proving

adopts AI planning paradigm

⇒ domain-specific operators

⇒ more abstract level

⇒ hierarchic planning for free

Cognitive Motivation

“*We know from experience that certain inferences are usually successful under certain prerequisites. So first we ponder about a reasonable way to proceed to prove the theorem. In other words, we roughly plan: If we get a certain result the next result will follow and then the next etc. Afterwards we have to fill in the details, and to check whether the plan really works.*” G.Faltings

Computational Motivation

- Superexponential search space in TP
- Global rather than local guidance in TP
- Domain-specific methods needed
- Domain-specific control **and** general-purpose control needed

⇒ CL^AM, INKA, OMEGA

Problems of Theorem Proving as Planning

Planning in static and deterministic environment and without goal interaction at object-level

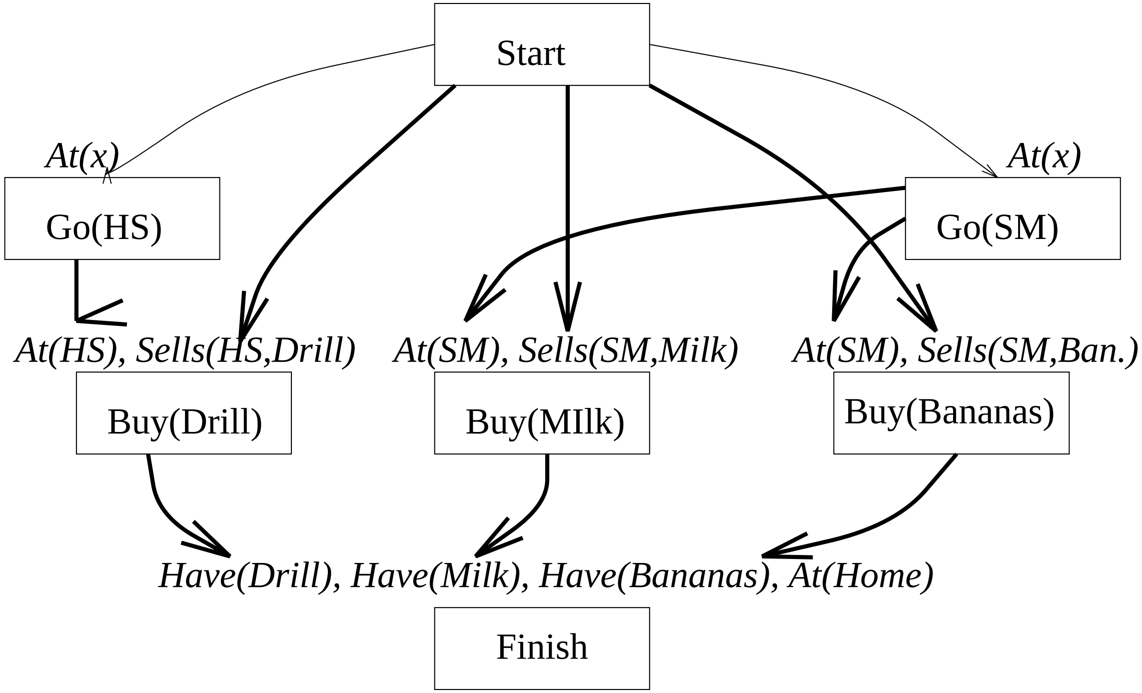
Problems

- ▷ design of operators is a major research problem
  structured plans are clearly preferred by users
- ▷ typically, solutions are deep ⇒ **search space**
- ▷ infinite branching can occur, infinitely many bindings
  ⇒ **search space**

General Solutions for the Problems

1. Operators for classes of problems
2. Search space:
    - ▷ planning operators: **methods** = tactics + spec (large steps)
    - ▷ **meta-level control** (domain-specific)
    - ▷ partial plan representation (S, O, B, Aux) + constraint solving
    - • controlled mixture of (hierarchical) planning strategies

Constraints in AI Planning



$$B = \{x \neq Home\}$$

$$O = \{Start < Go(HS), Go(HS) < Buy(Drill), \ldots\}$$

General Means of Control in PP

- *CLAM* (inductive proofs):
  - **annotations** for goals (and for rules): difference between induction hypothesis and induction conclusion
    $plus(x, y) = plus(y, x)$
    $plus(s(x), y) = plus(y, s(x))$
  - difference reduction to $s(plus(x, y)) = s(plus(y, x))$
- Omega (general proofs):
  - **control-rules** for choices of goals, methods, <u>instantiations</u>, strategies,
    maybe dependent on resources
  - flexibly assigning **focus**

The Class of Limit Theorems*

$$\lim_{x \to a} f(x) = L_1 =_{Def}$$

$$\forall \epsilon_1 \exists \delta_1 \forall x_1 (0 < \epsilon_1 \to 0 < \delta_1 \wedge x_1 \neq a \wedge val(x_1 - a) < \delta_1 \to val(f(x_1) - L_1) < \epsilon_1)$$

- LIM*

  $$\lim_{x \to a} f(x) = L_1 \wedge \lim_{x \to a} g(x) = L_2 \to \lim_{x \to a} f \cdot g(x) = L_1 \cdot L_2$$

- LIM+

  $$\lim_{x \to a} f(x) = L_1 \wedge \lim_{x \to a} g(x) = L_2 \to \lim_{x \to a} f + g(x) = L_1 + L_2$$

- If a function is differentiable, then it is continous.

Methods for Limit Plans

method: `Solve`$_{<}$`b`	
parameter	prln:L1, term: a, b,F, const:<, var: result, list(prln):Δ
premises	
conclusions	$\ominus$L1
constraints	eval(T $\mathit{result} \sim (\mathit{result} = \mathit{nil})$)
proof schema	L1. Δ $\vdash$ $(a < b)$ (solver-CS;)
procedure	solve$_{<}(a < b)$

Control: 1

- if goal: $val(B) < \epsilon$

  then `Solve`$_<$, `Solve*` or towards `LimHeuristic`

```
(cri~def-control-rule val-methods
    (kind METHOD)
    (if (and (goal-matches(g (val(X) < Y)))
        (and (sub-of-latest-assumption(h ass))
                      (matches(h (val(X) < Y))))))
    (then
      (do (select-method(SOLVE-b(g) SOLVE*(g)
                      Focus(h ass))))))
```

Control: 2

instantiation of meta-variables:

- least commitment or eager commitment?
- eagerly updating constraint pool?

Benefits of Proof Planning

- ▷ Reduction of search
- ▷ Support of user interaction and explanation
  Use of proof plans for analogy
  - abstract representation
  - structured representation
  - reasons for decisions stored in plans