# Web-Mining Agents
## Planning

Prof. Dr. Ralf Möller
Universität zu Lübeck
Institut für Informationssysteme

Tanya Braun (Übungen)

# Handlungsplanung

1. Gegeben eine initiale Situation,
2. eine Beschreibung der Zielbedingungen und
3. eine Menge von möglichen Aktionen,

-> Finde eine Sequenz von Aktionen (einen Handlungsplan), der die initiale Situation in eine Situation überführt, in der die Ziel-bedingungen gelten.

# Handlungsplanung vs. Problemlösen durch Suche

- Wesentlicher Unterschied:
  - Bei Handlungsplanung explizite, logikbasierte Repräsentation
- Zustände/Situationen:
  - Durch logische Formeln beschriebene Weltzustände vs. Datenstrukturen
- Operatoren:
  - Axiome oder Transformation von Formeln vs. Modikation von Datenstrukturen durch Programme

# Repräsentation der Operatoren durch Axiome

- Im Prinzip kann man Planung auf logische Inferenz (= Situationskalkül, nicht Beschreibungslogiken) reduzieren
- Bestehende Systeme, die für praktische Anwendungen effizient genug sind, befinden sich allerdings (immer noch) in der Entwicklung
  (-> Schlußsysteme für nichtmonotone Logiken)
- Wir behandeln einen anderen klassischen Ansatz...

# Transformation von Formeln: STRIPS

- STRIPS:
  STanford Research Institute Problem Solver (Planer der frühen 70-er Jahre)
- System ist zwar obsolet, der Formalismus wird aber immer noch benutzt
- Kernidee:
  - "Weltzustand" durch log. Formeln repräsentiert,
  - "Operatoren" manipulieren Weltzustand

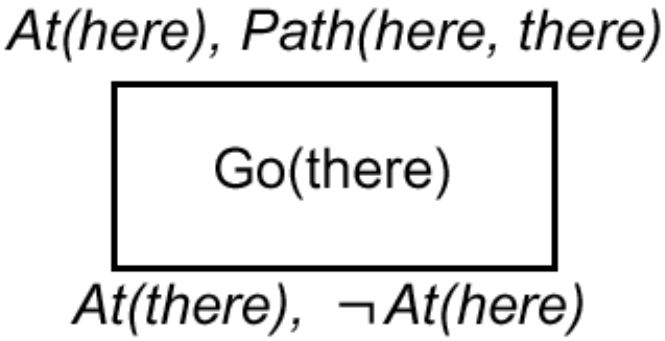
# Der STRIPS-Formalismus

- Weltzustand (inkl. initialer Zustand)
  - Menge von Grundatomen, keine Funktionssymbole außer Konstanten, interpretiert unter CWA (manchmal auch Standardinterpretation, d.h. negative Fakten müssen angegeben werden)
  - Beispiel Blockswelt:
    On_Table(A), On_Table(B), On_Table(C)
- Zielbedingungen:
  - Menge von Grundatomen
  - Beispiel: On_Block(C, B), On_Block(B, A)

# STRIPS-Operatoren

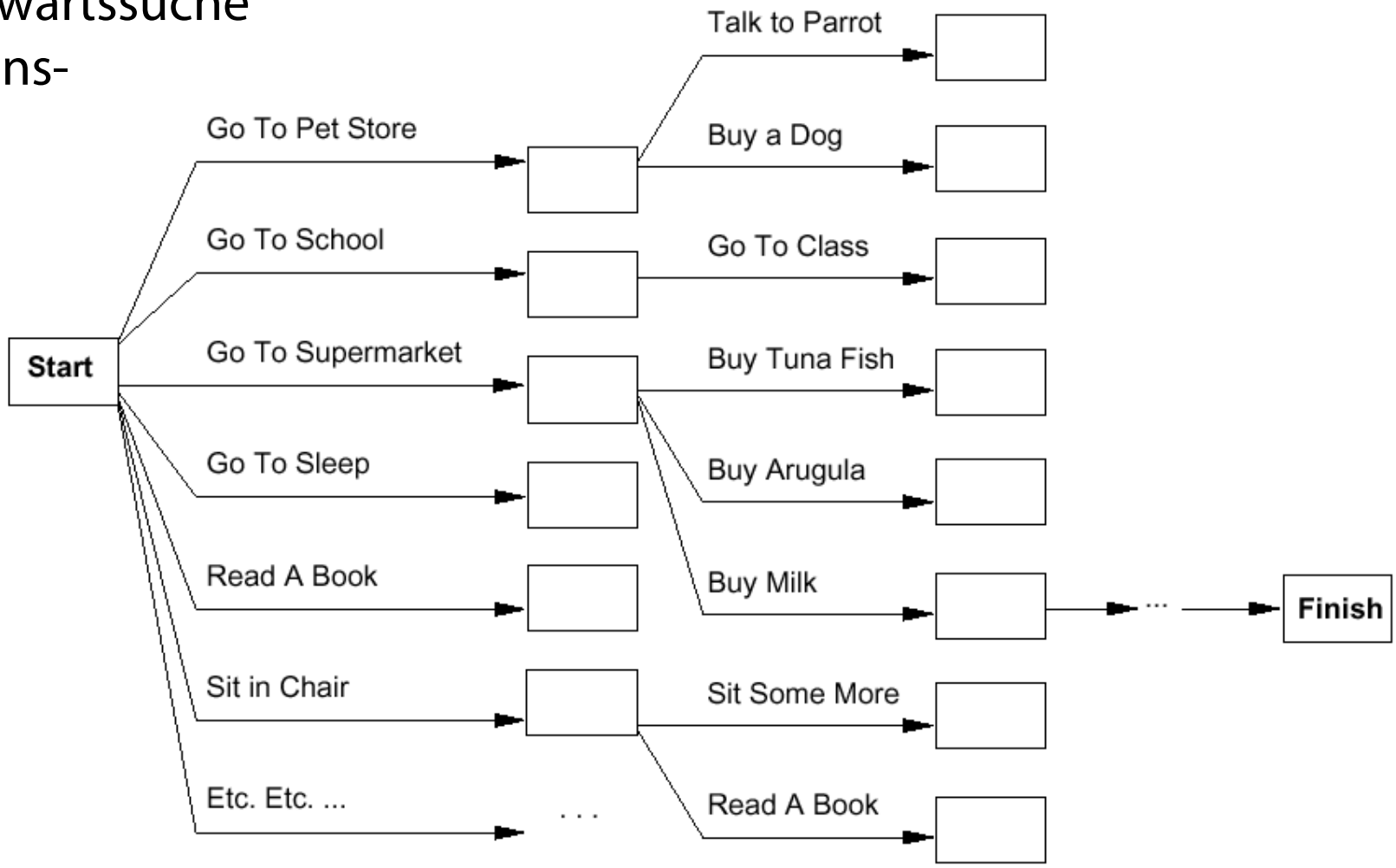
- Aktionen sind Tripel, bestehend aus
  - Aktionsnamen: Funktionsname mit Parametern
  - Vorbedingungen: Konjunktion positiver Literale; müssen gelten, damit Aktion ausführbar ist
  - Effekte: Konjunktion positiver und negativer Literale; positive Literale werden hinzugenommen (ADD Liste), negative gelöscht (DEL Liste)
  - Variablen möglich (Renaming)

$$Op(\text{Action: } Go(there),\ \text{Precond: } At(here) \wedge Path(here, there),\ \text{Effect: } At(there) \wedge \neg At(here))$$



# Reduktion von Planen auf Suche in einem Zustandsraum

- Vorwärtssuche (Progressionsplanung)
- oder Rückwärtssuche (Regressions-planung)



# Suche im Planraum

Der Ausgangszustand ist dann ein **partieller Plan**, der nur einen Start- und einen Zielzustand enthält:

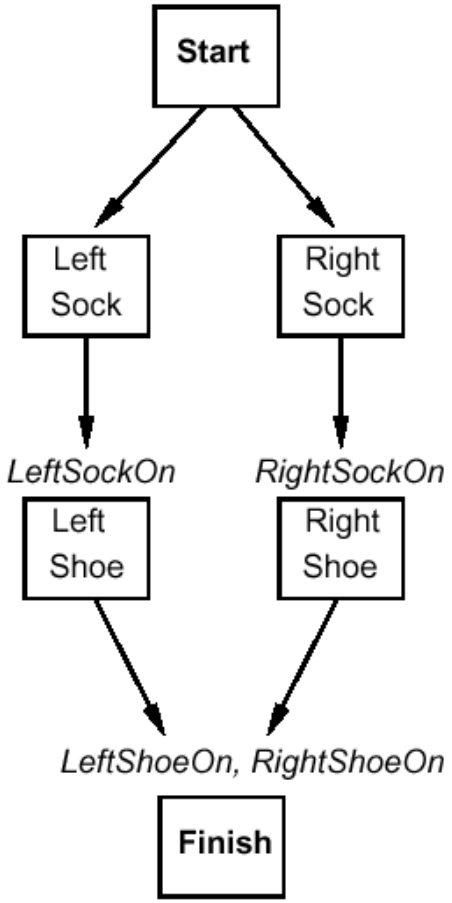

Der Zielzustand ist ein **vollständiger Plan**, der das gegebene Problem löst:


**Verfeinerungsoperatoren** machen den Plan konkreter (mehr Schritte usw.)

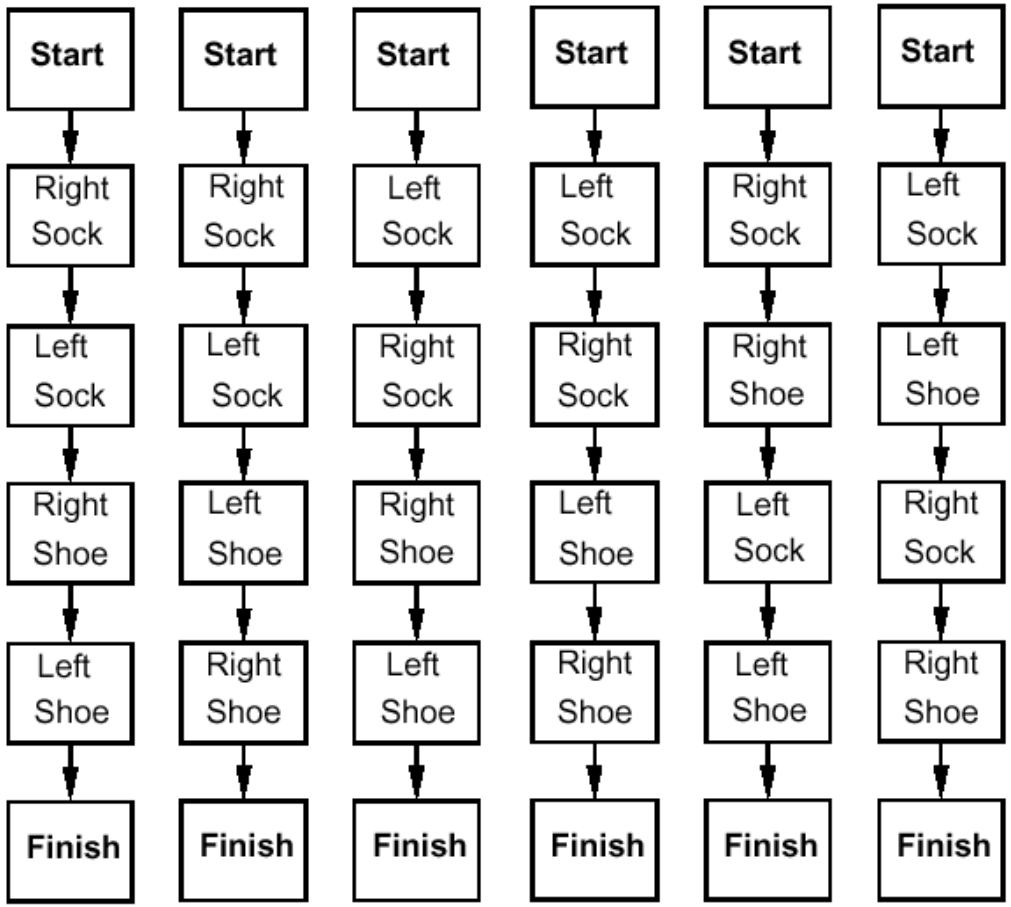
**Modifikationsoperatoren** modifizieren den Plan (im folgenden nur V.-Op.)

# Plan = Sequenz von Aktionen?

**Partial Order Plan:**




**Total Order Plans:**



# Prinzip der geringsten Festlegung

Oft ist es aber nicht sinnvoll oder möglich, sich bei der Reihenfolge früh festzulegen (Socken und Schuhe anziehen).

$\rightsquigarrow$ **nicht-lineare** oder **partiell geordnete Pläne** (*least-commitment planning*)

# Repräsentation nicht-linearer Pläne

- Planschritt = STRIPS-Operator
- Plan besteht aus

  - Menge von **Planschritten** mit partieller Ordnung ($\prec$), wobei $S_i \prec S_j$ gdw. $S_i$ muß vor $S_j$ ausgeführt werden.
  - Menge von **Variablenbelegungen** $x = t$, wobei $x$ eine Variable und $t$ eine Konstante oder Variable ist.
  - Menge **kausaler Beziehungen**: $S_i \xrightarrow{c} S_j$ bedeutet „$S_i$ erzeugt die Vorbedingung $c$ für $S_j$“ (impliziert $S_i \prec S_j$)

- L … n vollständig und k …

# Vollständigkeit und Konsistenz

**Vollständiger Plan:**

Jede Vorbedingung eines Schritts wird erfüllt:

$\forall S_j$ mit $c \in$ Precond$(S_j)$ und

$\exists S_i$ mit $S_i \prec S_j$ und $c \in$ Effects$(S_i)$ und

für jede Linearisierung des Plans gilt:

$\forall S_k$ mit $S_i \prec S_k \prec S_j$, $\neg c \notin$ Effect$(S_k)$.

**Konsistenter Plan:**

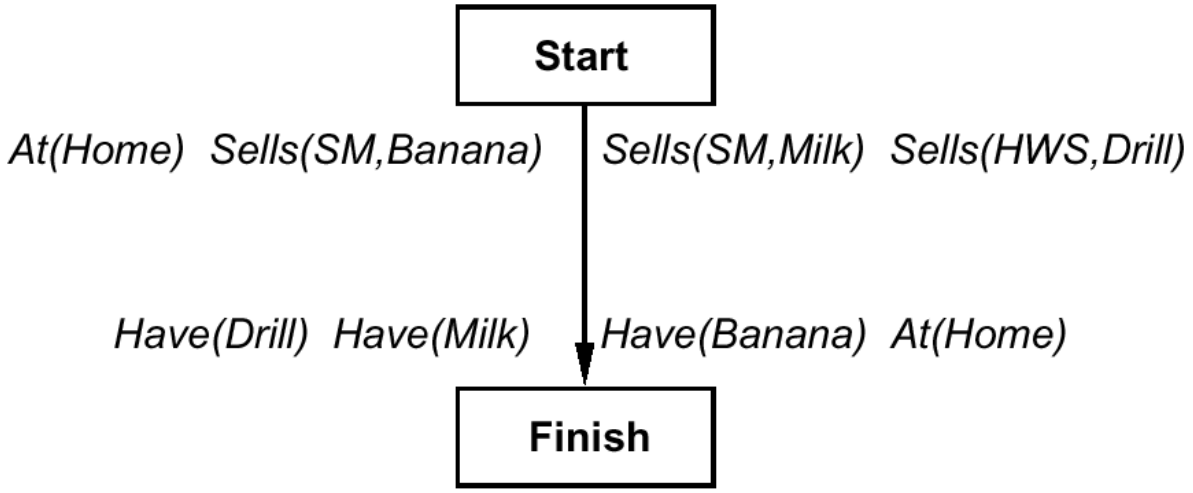
wenn $S_i \prec S_j$, dann $S_j \not\prec S_i$ und

wenn $x = A$, dann $x \neq B$ für verschiedene $A$ und $B$ für eine Variable $x$

(Unique Name Assumption!)

Ein *vollständiger, konsistenter Plan* heißt **Lösung** eines Planungsproblems.

# Beispiel




**Aktionen:**

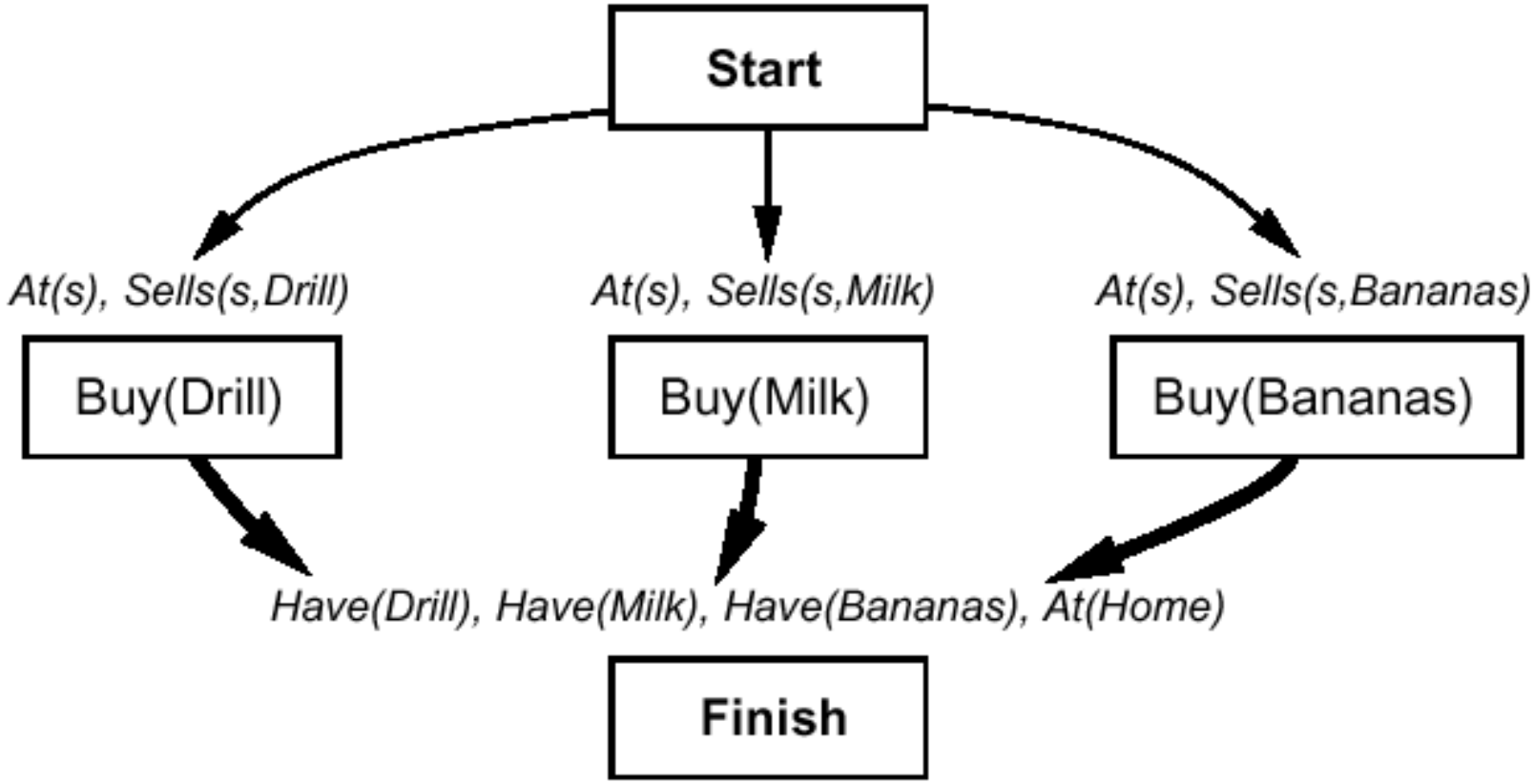
*Op* (Action: *go(there)*,
Precond: *At(here)*,
Effect: *At(there)* $\wedge$ $\neg$*At(here)*)

*Op* (Action: *buy(x)*,
Precond: *At(store)* $\wedge$ *Sells(store,x)*,
Effect: *Have(x)*)

there, here, x, store sind Variablen.

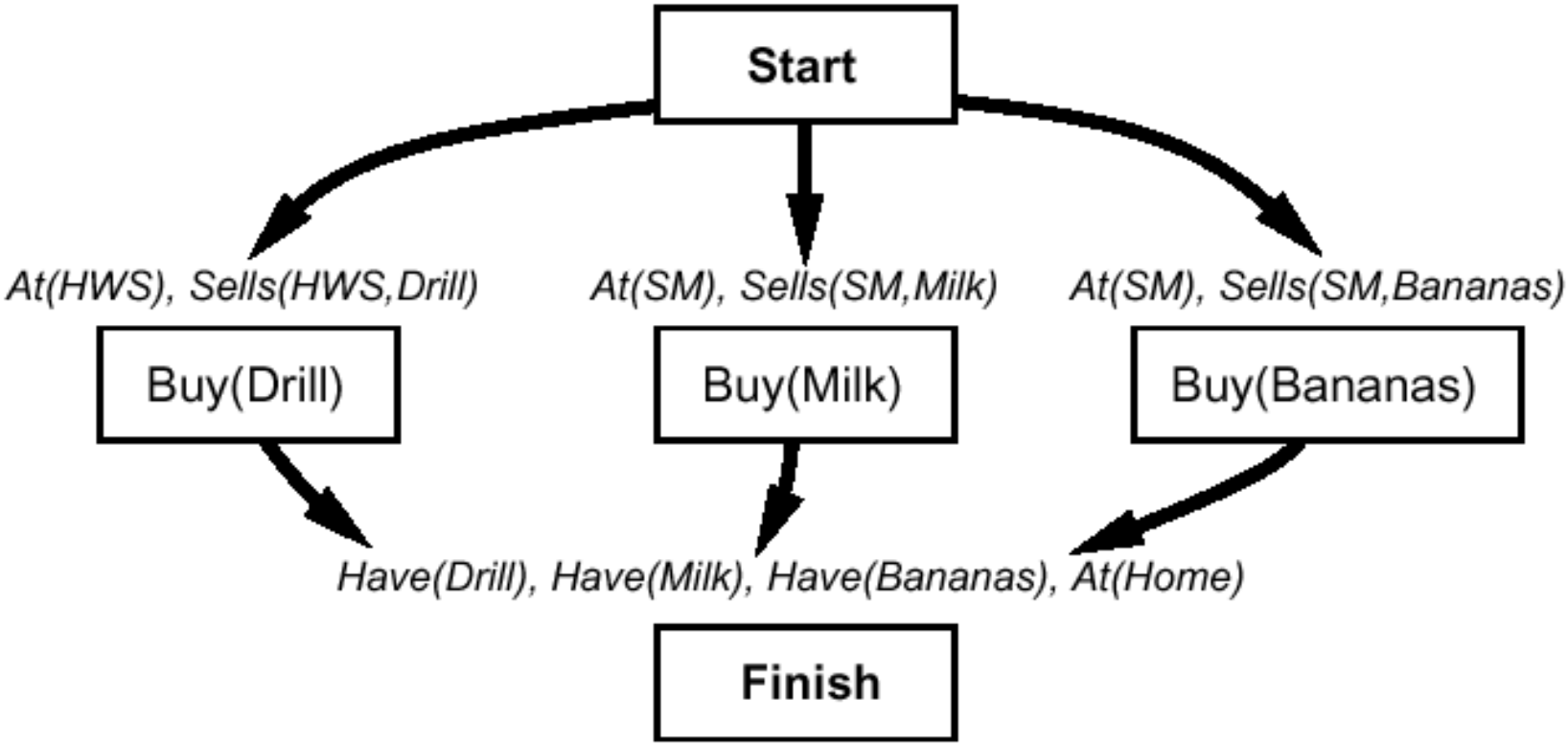
# Planverfeinerung (1)

- Regressionsplanung: Erfülle die Have-Prädikate




Dünne Pfeile $= \prec$
Fette Pfeile $=$ Kausale Beziehungen $+ \prec$

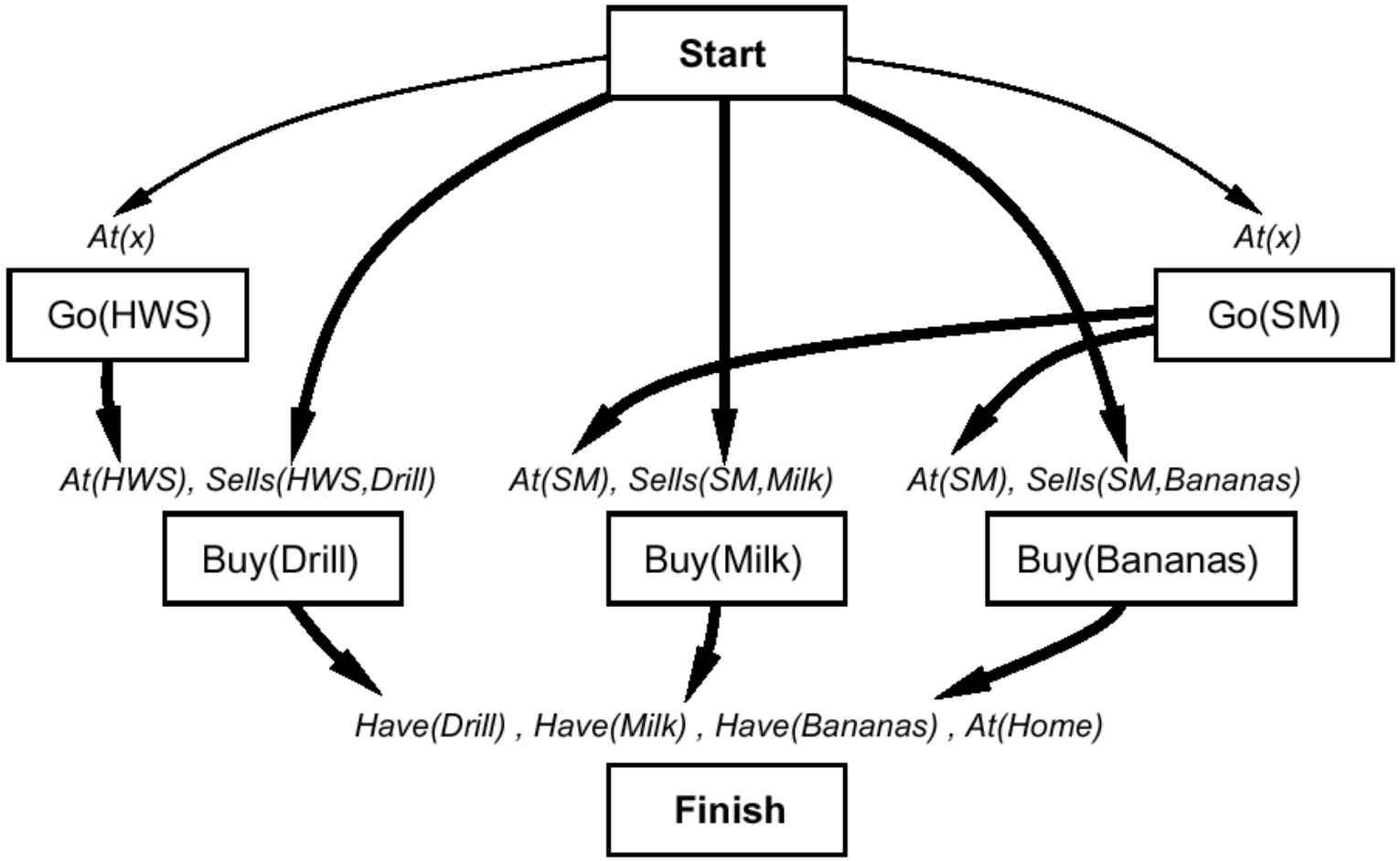
# Planverfeinerung (1)




. . . nach Instantiierung der Variablen

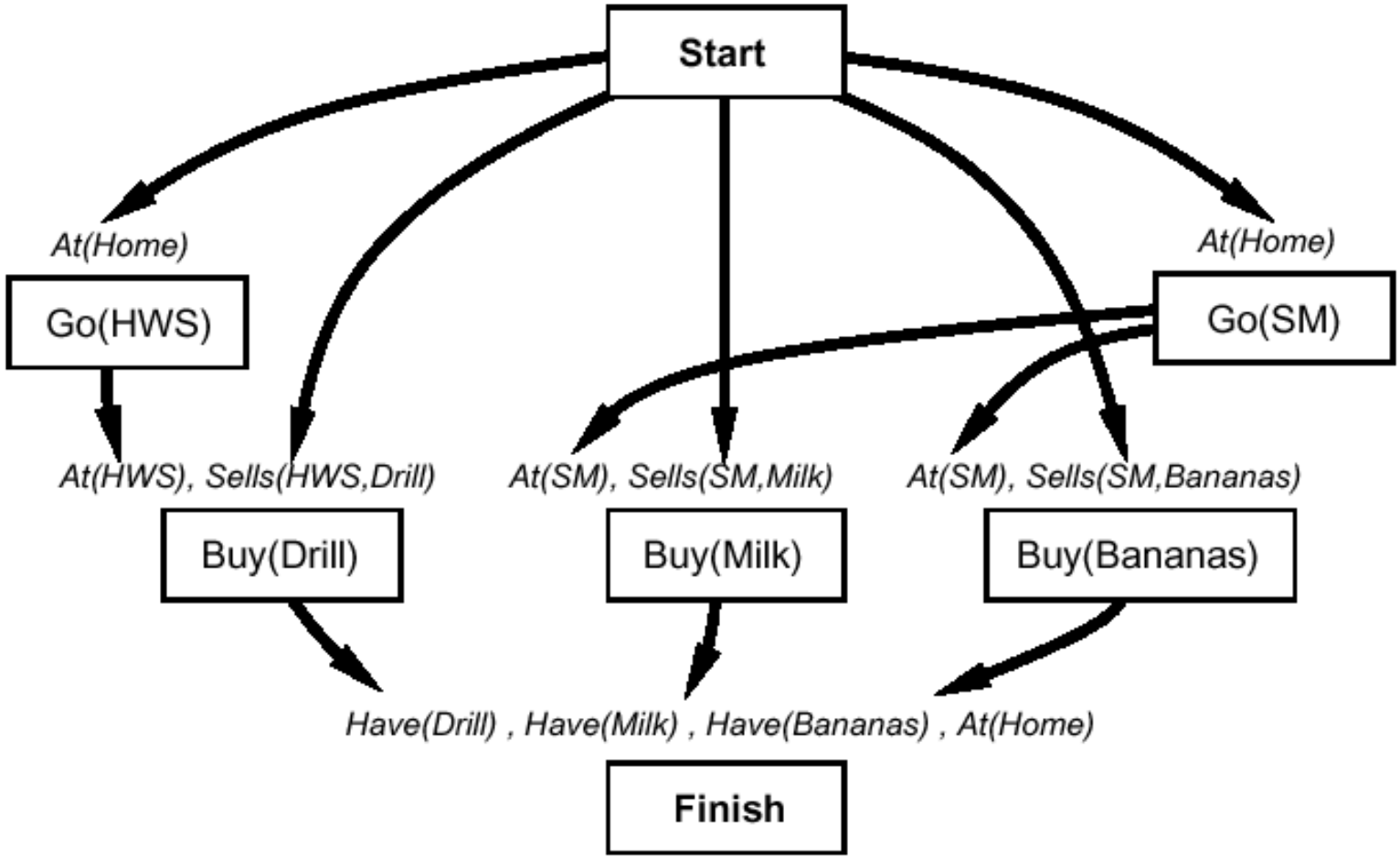
# Planverfeinerung (2)

- ... im richtigen Geschäft kaufen



# Planverfeinerung (3)

- ... da muß man erst einmal hin

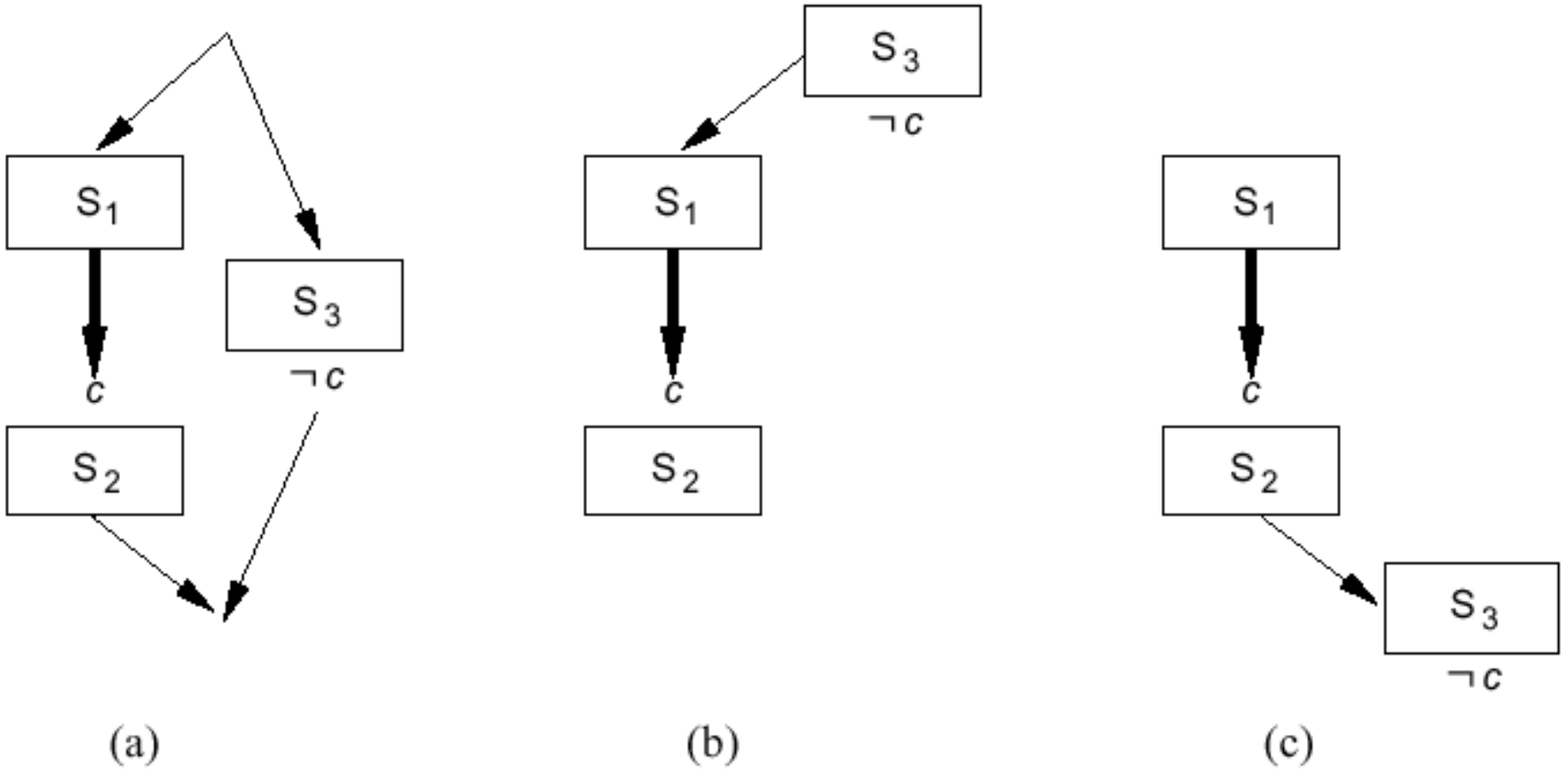


# Planverfeinerung (3)

**Beachte**: Bisher keine Suche, sondern einfache Rückwärtsverkettung.

**Jetzt:** *Konflikt!* Wenn wir *go(hws)* gemacht haben, sind wir nicht mehr *At(home)*. Das gleiche gilt für *go(sm)*.

# Schutz kausaler Beziehungen



# Schutz kausaler Beziehungen

a) Konflikt: $S_3$ „bedroht“ die kausale Beziehung zwischen $S_1$ und $S_2$

Konfliktlösungen:

b) Demotion: Den „Bedroher“ vor die kausale Beziehung legen

c) Promotion: Den „Bedroher“ hinter die kausale Beziehung legen

# Eine andere Planverfeinerung...

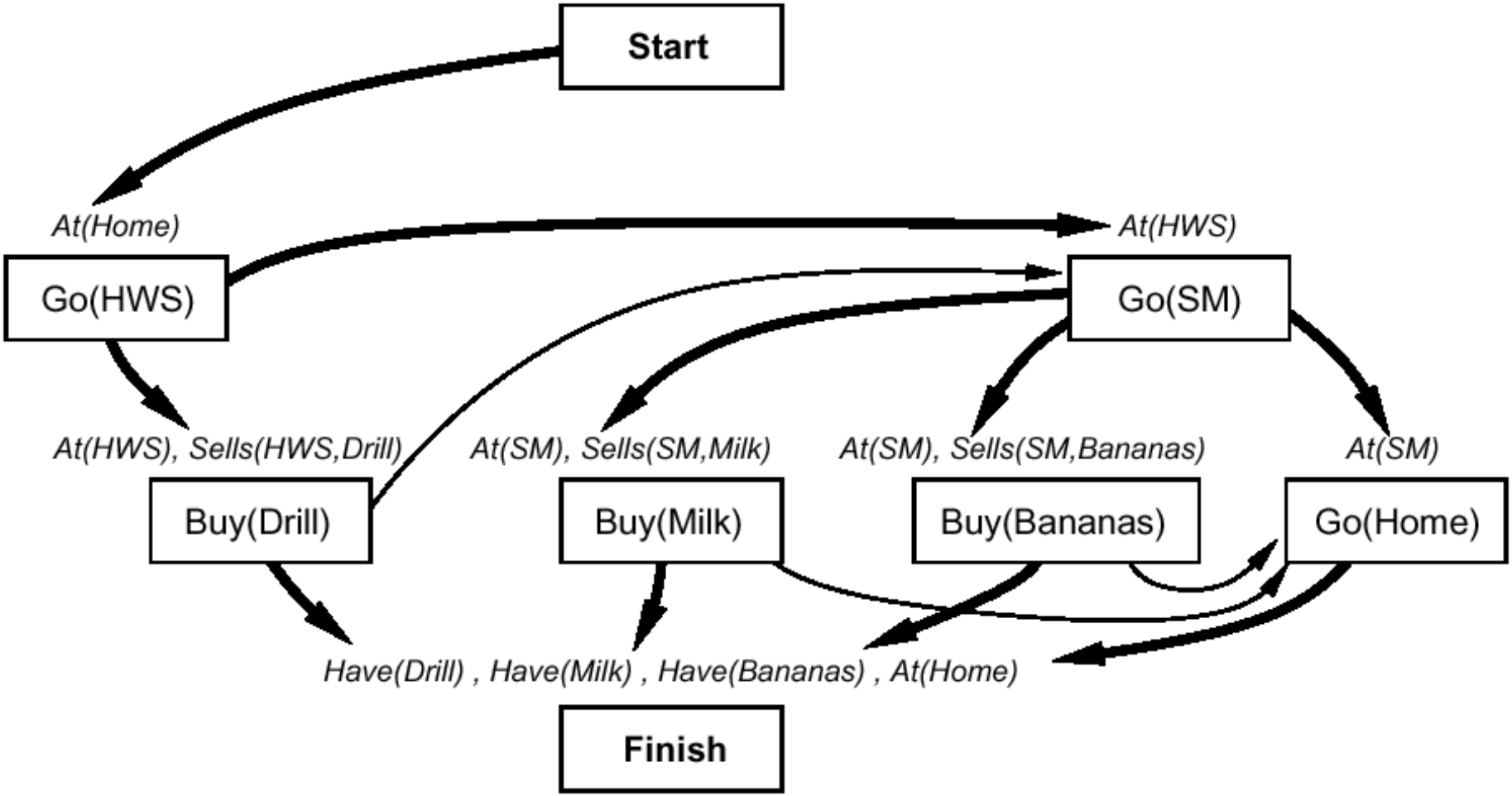
- Wir können den Konflikt **nicht** durch „Schutz“ auflösen.

$\rightsquigarrow$ Wir haben uns bei einer Planverfeinerung **falsch** entschieden.

- **Alternative:** Bei der Instantiierung von *At*($x$) in *go(sm)* wählen wir $x = hws$ (mit *kausaler Beziehung*)

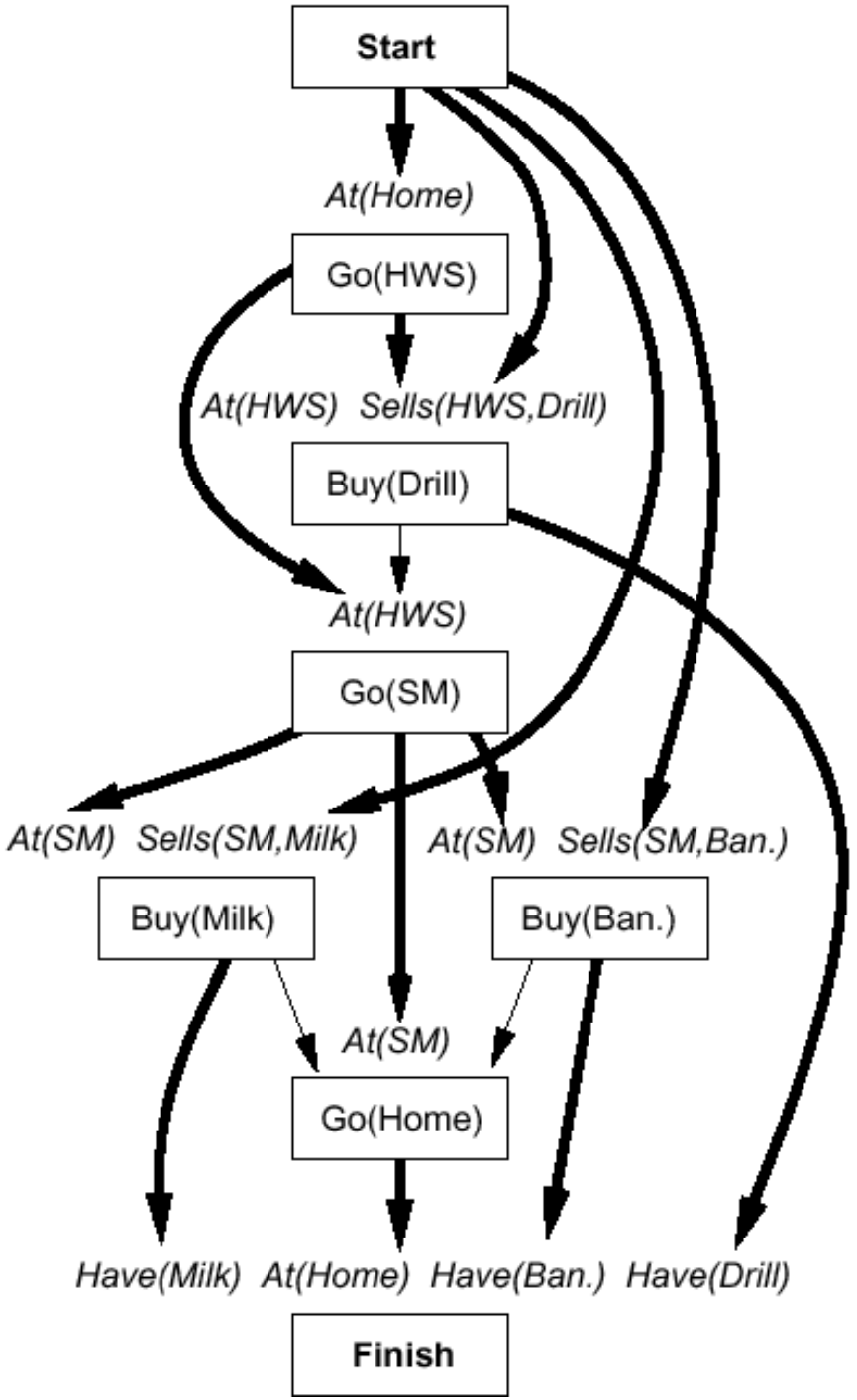
- **Beachte:** Dies bedroht das Kaufen des *drills* $\rightsquigarrow$ Promotion von *go(sm)*.

# Eine andere Planverfeinerung...



# Die vollständige Lösung

- ... mit allen Links
- Bestimmung
  z.B. durch POP-Algorithmus
  - Vollständig
  - ... und korrekt
- Zusätzlich, hier nicht
  näher betrachtet:
  Behandlung von Variablen



**function** POP(*initial, goal, operators*) **returns** *plan*

*plan* ← MAKE-MINIMAL-PLAN(*initial, goal*)
**loop do**
  **if** SOLUTION?(*plan*) **then return** *plan*
  $S_{need}$, $c$ ← SELECT-SUBGOAL(*plan*)
  CHOOSE-OPERATOR(*plan, operators*, $S_{need}$, $c$)
  RESOLVE-THREATS(*plan*)
**end**

---

**function** SELECT-SUBGOAL(*plan*) **returns** $S_{need}$, $c$

pick a plan step $S_{need}$ from STEPS(*plan*)
  with a precondition $c$ that has not been achieved
**return** $S_{need}$, $c$

**procedure** CHOOSE-OPERATOR(*plan, operators,* $S_{need}$, *c*)

**choose** a step $S_{add}$ from *operators* or STEPS(*plan*) that has *c* as an effect
**if** there is no such step **then fail**
add the causal link $S_{add} \xrightarrow{c} S_{need}$ to LINKS(*plan*)
add the ordering constraint $S_{add} \prec S_{need}$ to ORDERINGS(*plan*)
**if** $S_{add}$ is a newly added step from *operators* **then**
    add $S_{add}$ to STEPS(*plan*)
    add $Start \prec S_{add} \prec Finish$ to ORDERINGS(*plan*)

---

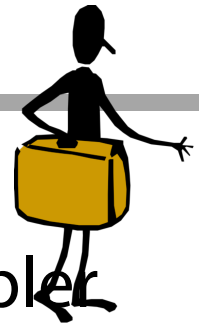
**procedure** RESOLVE-THREATS(*plan*)

**for each** $S_{threat}$ that threatens a link $S_i \xrightarrow{c} S_j$ in LINKS(*plan*) **do**
    **choose** either
        *Promotion:* Add $S_{threat} \prec S_i$ to ORDERINGS(*plan*)
        *Demotion:* Add $S_j \prec S_{threat}$ to ORDERINGS(*plan*)
    **if not** CONSISTENT(*plan*) **then fail**
**end**

# Planungssysteme

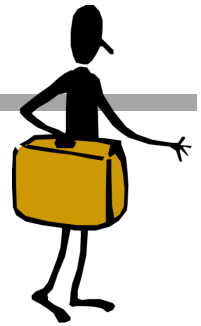
- Prodigy
  - sucht im Zustandsraum und generiert lineare Pläne
  - (http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/prodigy/Web/prodigy-home.html)
- UCPOP (http://www.cs.washington.edu/ai/ucpop.html)
- Neuere, effizientere Systeme:
  - Graphplan (http://www-2.cs.cmu.edu/~avrim/graphplan.html)
  - IPP (http://www.informatik.uni-freiburg.de/~koehler/ipp.html)

# Zusammenfassung, Kernpunkte (1)



- Handlungsplanung unterscheidet sich vom Problemlösen dadurch, daß die Repräsentation flexibler ist.
- Statt im Zustandsraum kann man im Planraum suchen.

# Zusammenfassung, Kernpunkte (2)



- Das Prinzip der geringsten Festlegung (least commitment) besagt, daß man während der Suche Entscheidungen nur dann treffen soll, wenn es unbedingt notwendig ist
- Nichtlineares Planen ist eine Instanz dieses Prinzips
- Der POP-Algorithmus realisiert nichtlineares Planen und ist vollständig und korrekt