PCS 3438 / PCS 3838
Inteligência Artificial

Prof. Dr. Jaime Simão Sichman
Prof. Dra. Anna Helena Reali Costa
Prof. Dra. Anarosa Alves Franco Brandão

Planejamento

Planejamento

1. Introdução
2. Planejamento e Resolução de Problemas
3. Abordagens
   - 3.1. Cálculo de situações + provador de teoremas
   - 3.2. STRIPS + POP
4. Conclusões

Conceitos Básicos

- Planejador: mecanismo que permite encontrar/gerar um plano que permita a um agente atingir um objetivo

- Plano: seqüência ordenada de ações
  - problema: obter banana, leite e uma furadeira
  - plano: ir ao supermercado, ir à seção de frutas, pegar as bananas, ir à seção de leite, pegar uma caixa de leite, ir ao caixa, pagar tudo, ir a uma loja de ferramentas, ..., voltar para casa.

Planejamento e Resolução de Problemas

- Representação em RP
  - **Ações** : programas que geram o estado sucessor
  - **Estados** : descrição completa
    - problemático em ambientes inacessíveis
  - **Objetivos**: função de teste e heurística
  - **Planos**: totalmente ordenados e criados incrementalmente a partir do estado inicial
    - Ex. posições das peças de um jogo

- Exemplo do supermercado
  - **estado inicial:** em casa, sem objetos desejados
  - **estado final**: em casa com objetos desejados
  - **operadores**: tudo o que o agente pode fazer
  - **heurística**: número de objetos ainda não possuídos

Exemplo de Resolução de Problemas

Limitações de Resolução de Problemas

- Fator de ramificação grande
- A função heurística apenas escolhe o estado mais próximo do objetivo. Não permite descartar ações a priori
- Não permite abstração dos estados parciais
- Considera ações a partir do estado inicial, uma após a outra
- Objetivo é testado para cada estado, e para cada novo estado gerado um teste idêntico deve ser realizado
- Idéia: combinar busca com uma descrição mais rica dos estados, baseada em conhecimento, e criar um algoritmo eficiente para processá-la!

Planejamento: 3 idéias principais

- Representação dos estados, objetivos e ações usando lógica de predicados (descrições parciais dos estados)
  - pode conectar diretamente estados e ações
  - ex. estado: Have (Milk), ação: Buy(milk) $\rightarrow$ Have(Milk)
- Liberdade de adicionar ações ao plano quando forem necessárias
  - ordem de planejamento $\neq$ ordem de execução
  - primeiro, coloca-se o que é importante (Buy(Milk)) mesmo sem saber quando esta ação será executada!
  - diminui fator de ramificação
- Utilizar a estratégia de dividir para conquistar, resolvendo sub-objetivos
  - sub-plano supermercado, sub-plano loja de ferramentas

Agente Planejador Simples

- Este agente teria um ciclo composto por percepção, planejamento (ilimitado, off-line) e ação (uma a cada passo)
- Algoritmo

Function Simple-planning-agent (*percept*) **returns** *action*
t := 0
Tell (KB, Make-percept-sentence (*percept, t*))
current := State-description (*KB,t*)
If *p* = NoPlan **then**
    G := Ask(KB, Make-a-goal-query(*t*))
    p := Ideal-planner(*current, G, KB*)
If *p* = NoPlan or p is empty **then** *action* := NoOp
else action := First (*p*)
        p := Rest (*p*)
Tell (KB, Make-action-sentence (*action,t*))
t := t + 1
return *action*

Cálculo de Situações

- O mundo consiste em uma sequência de situações
  - situação N ===**ação**===> situação N+1
- Predicados que **mudam** com o tempo têm um argumento de **situação** adicional

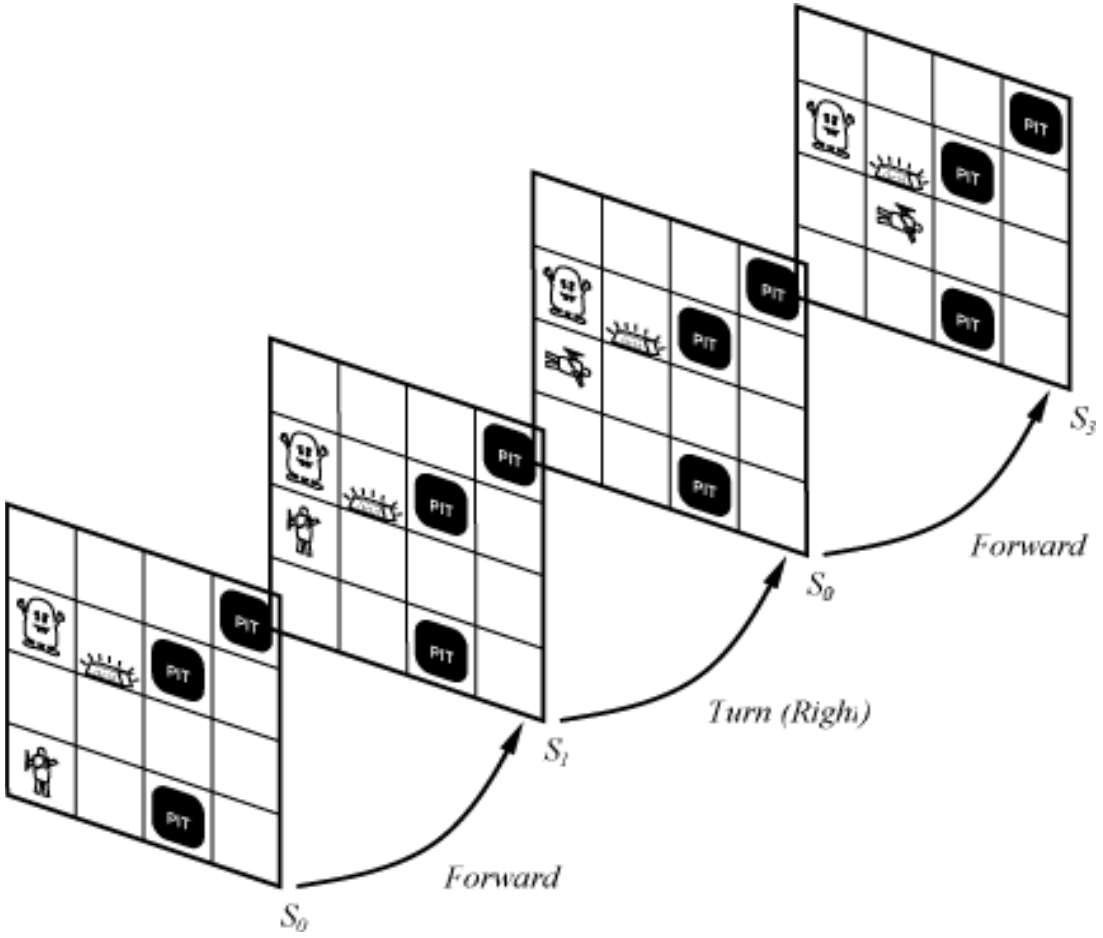
  Em(Agente,[1,1],S0) $\wedge$ Em(Agente,[1,2],S1)
- Predicados que denotam propriedades que **não mudam** com o tempo não usam argumentos de situação

  Parede(0,1) $\wedge$ Parede(1,0)
- Para representar as mudanças no mundo, usa-se uma função Resultado

  Resultado (ação,situação N) = situação N+1

Cálculo de Situações

- Wumpus world



Resultado(Forward,S_0) = S_1
Resultado(Turn(Right),S_1) = S_2
Resultdo(Forward,S_2) = S_3

Axiomas Estado-Sucessor

- **Axioma estado-sucessor:**
  - combinação entre os axiomas de efeito e de quadro
    - **Axiomas de efeito**: descrevem as propriedades do mundo que mudam após uma ação
    - **Axiomas de quadro**: descrevem as propriedades do mundo que **não** mudam após uma ação

uma coisa é verdade depois ↔
[uma ação acabou de torná-la verdade
∨
ela já era verdade e nenhuma ação a tornou falsa]

Axiomas Estado-Sucessor

uma coisa é verdade depois $\leftrightarrow$
[uma ação acabou de torná-la verdade
$\vee$
ela já era verdade e nenhuma ação a tornou falsa]

- Exemplo:

$\forall$ a, x, s Segurando (x, Resultado(a,s)) $\leftrightarrow$
[(a = Pegar $\wedge$ Presente (x, s) $\wedge$ Portável (x))
$\vee$ (Segurando (x, s) $\wedge$ (a $\neq$ Soltar)]

- É necessário escrever um axioma estado-sucessor para cada predicado que pode mudar seu valor no tempo

Planejando com Calculo de Situações

- Estado inicial: sentença lógica

  $At(Home, S0) \land \neg Have(Milk , S0) \land \neg Have(Bananas , S0) \land \neg Have(Drill , S0)$

- Estado Objetivo: pergunta lógica **(p/ unificação)**

  $At(Home, S) \land Have(Milk , S) \land Have(Bananas , S) \land Have(Drill , S)$

- Operadores: conjunto de axiomas de estado sucessor

  $\forall a,s\ Have(Milk, Result(a, s)) \Leftrightarrow$

  $[(a = Buy(Milk) \land At(supermarket, s)$

  $\lor (Have(Milk, s) \land a \neq Drop(Milk))]$

- Notação

Resultado(a,s) - uma ação executada em s => S = Resultado(a, S0)

Resultado'(p,s) - uma seqüência de ações p => S = Resultado' (p, S0)

Planejando com Calculo de Situações

- Reescrevendo o Estado Objetivo: pergunta lógica

  $At(Home, Resultado'(p, S0)) \wedge Have(Milk, Resultado'(p, S0)) \wedge$
  $Have(Bananas, Resultado'(p, S0)) \wedge Have(Drill, Resultado'(p, S0))$
- Solução:

  p = [Go(SuperMarket), Buy(Milk), Buy(Bananas),
  Go(HardwareStore), Buy(Drill), Go(home)]
- Limitações
  - Eficiência da inferência em lógica de primeira ordem
  - Nenhuma garantia sobre a qualidade da solução
    - ex. pode haver passos redundantes
- Solução
  - Criar uma linguagem especializada (STRIPS)
  - Criar um algoritmo para planejar (POP)

STRIPS: Estados e Objetivos

- STRIPS: Stanford Research Institute Problem Solver (Fikes e Nilsson, 1971)
- Estados: **conjunção de literais sem variáveis**
  - Inicial: At(Home)
    - Por default, literal não representado é falso (hipótese do mundo fechado); assim, não precisa escrever:

      $\neg$ Have(Milk , S0) $\wedge \neg$ Have(Bananas , S0) $\wedge \neg$ Have(Drill , S0)
  - Final: At(Home) $\wedge$ Have(Milk) $\wedge$ Have(Bananas) $\wedge$ Have(Drill)
- Objetivos: **conjunção de literais e/ou variáveis ($\exists$)**
  - At(Home) $\wedge$ Have(Milk) $\wedge$ Have(Bananas) $\wedge$ Have(Drill)
  - At(x) $\wedge$ Sells(x, Milk)

STRIPS: Ações

- Ações:
  - Descritor da **ação**: predicado lógico
  - **Pré-condição:** conjunção de literais positivos
  - **Efeito**: conjunção de literais, podendo ser:
    - positivos (adicionados a uma lista)
    - negativos (retirados de uma lista)
- Exemplo: Operador para ir de um lugar a outro
  - Op(ACTION: Go(there),
    PRECOND: At(here) $\wedge$ Path(here, there),
    EFFECT: ADD: At(there), DEL: $\neg$ At(here))
- Notação alternativa

At(here), Path(here, there)

Go(there)

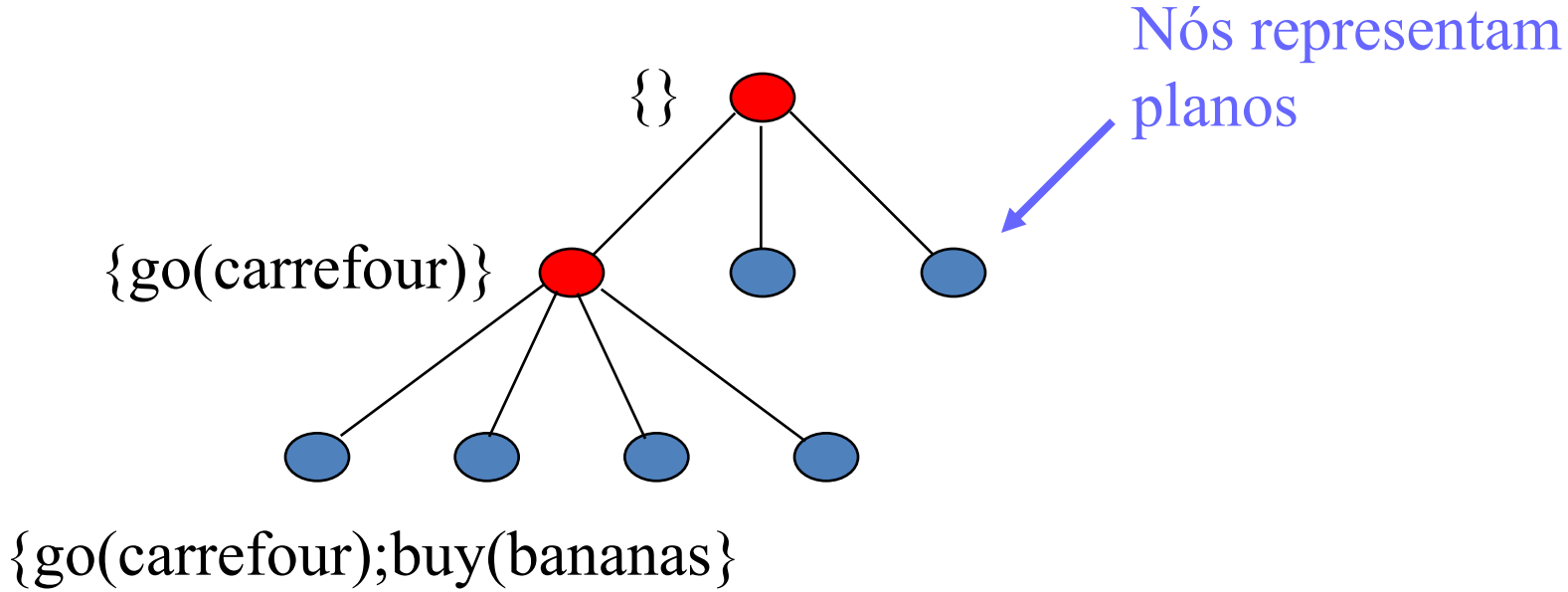
At(there), $\neg$ *At(here)*

Tipos de Planejadores

- Controle
  - Progressivo: estado inicial ➔ objetivo
  - Regressivo: objetivo ➔ estado inicial
    - mais eficiente (há menos caminhos partindo do objetivo do que do estado inicial)
    - problemático se existem múltiplos objetivos
- Espaços de busca
  - Espaço de situações (como em resolução de problemas)
    - Estados na árvore de busca representam estados do mundo
  - Espaço de planos (planos parciais)
    - Estados na árvore de busca representam planos parciais
    - mais flexível
    - evita engajamento prematuro

Busca no Espaço de Planos

- Idéia
  - Buscar um plano desejado em vez de uma situação desejada (espécie de meta-busca)
  - parte-se de um plano inicial (parcial), e aplica-se os operadores até chegar a um plano final (completo)



Busca no Espaço de Planos: Nós

Cada nó representa um plano parcial e contém:

- Ações = {A1, A2, A3, ..., An}
- Restrições de Ordem = { A1 < A2, ..., A3 < An },
- Ligações Causais = $\{Ai \xrightarrow{c} Aj\}$
  - efeito de Ai adiciona c que é uma pré-condição de Aj
- Pré-Condições Abertas: conjunto de pré-condições que ainda não tem ligações causais
- Instanciacão de Variáveis = { x = cte1, v = z}

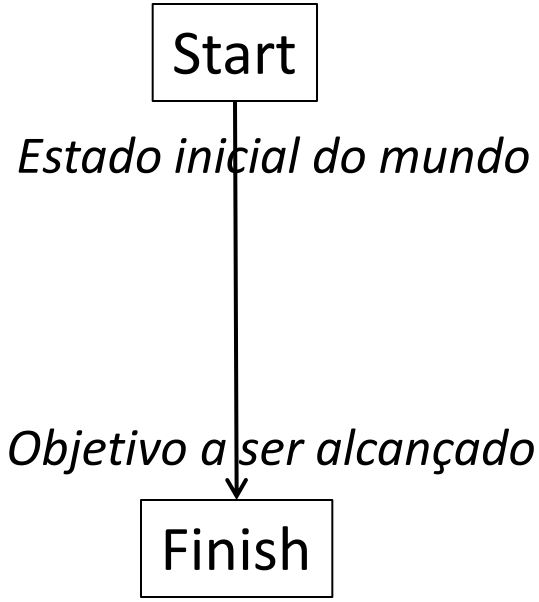
Busca no Espaço de Planos: Operadores

Os operadores possíveis no espaço de planos são:

- Adicionar uma ação para eliminar uma pré-condição aberta
- Adicionar um link causal de uma ação já existente para uma pré-condição aberta
- Adicionar uma restrição de ordem de uma ação em relação a outra
- Instanciar uma variável

Busca no Espaço de Planos: Plano Inicial

- Plano inicial
  - passos Start e Finish
  - Start tem como efeitos o estado inicial do mundo
  - Finish tem como pré-condições o objetivo a ser alcançado



Busca no Espaço de Planos: Plano Final

- Plano final
  - **Completo** – toda a pré-condição de toda ação tem uma ligação causal para alguma outra ação
  - **Consistente** – não há contradições
    - nos ordenamentos das ações
    - nas atribuição de variáveis

Exemplo de Busca no Espaço de Planos

- Plano para calçar meias e sapatos
- Plano inicial
  - Start (efeito: pés descalços)
  - Finish (pré-condição: estar com meias e sapatos)
- Operadores
  - calçar meia direita (pré-condição: pé direito descalço; efeito: pé direito com meia)
  - calçar sapato direito (pré-condição: pé direito com meia; efeito: pé direito com meia e sapato)
  - calçar meia esquerda...
  - calçar sapato esquerdo...
- Plano final?
  - Existem vários possíveis....
  - Como representar isto?

Exemplo de Busca no Espaço de Planos

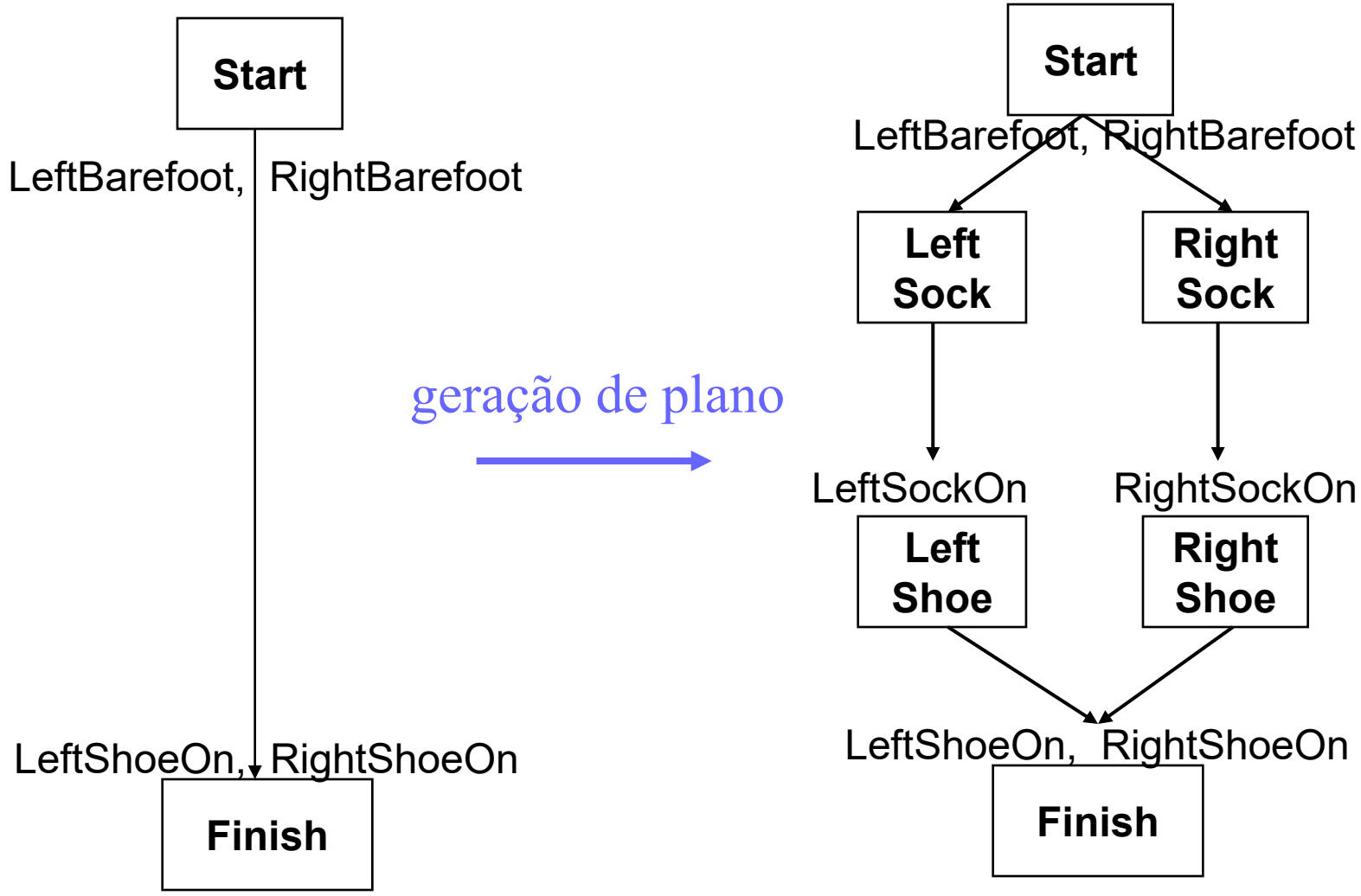
- Objetivo: RightShoeOn $\wedge$ LeftShoeOn
- Operadores
  - Op(ACTION:RightShoe
    PRECOND: RightSockOn ,
    EFFECT: ADD: RightShoeOn)
  - Op(ACTION: RightSock
    PRECOND: RightBarefoot ,
    EFFECT: ADD: RightSockOn)
  - Op(ACTION:LeftShoe
    PRECOND: LeftSockOn ,
    EFFECT: ADD: LeftShoeOn)
  - Op(ACTION: LeftSock
    PRECOND: LeftBarefoot ,
    EFFECT: LeftSockOn)

Exemplo de Busca no Espaço de Planos

- Plano inicial

Plan(ACTIONS = {S1: Op(ACTION: Start,
EFFECT: RightBarefoot $\wedge$
LeftBarefoot),
S2: Op(ACTION: Finish,
PRECOND: RightShoeOn $\wedge$
LeftShoeOn)},
ORDERINGS: { S1 < S2 },
BINDINGS: {},
LINKS: {})

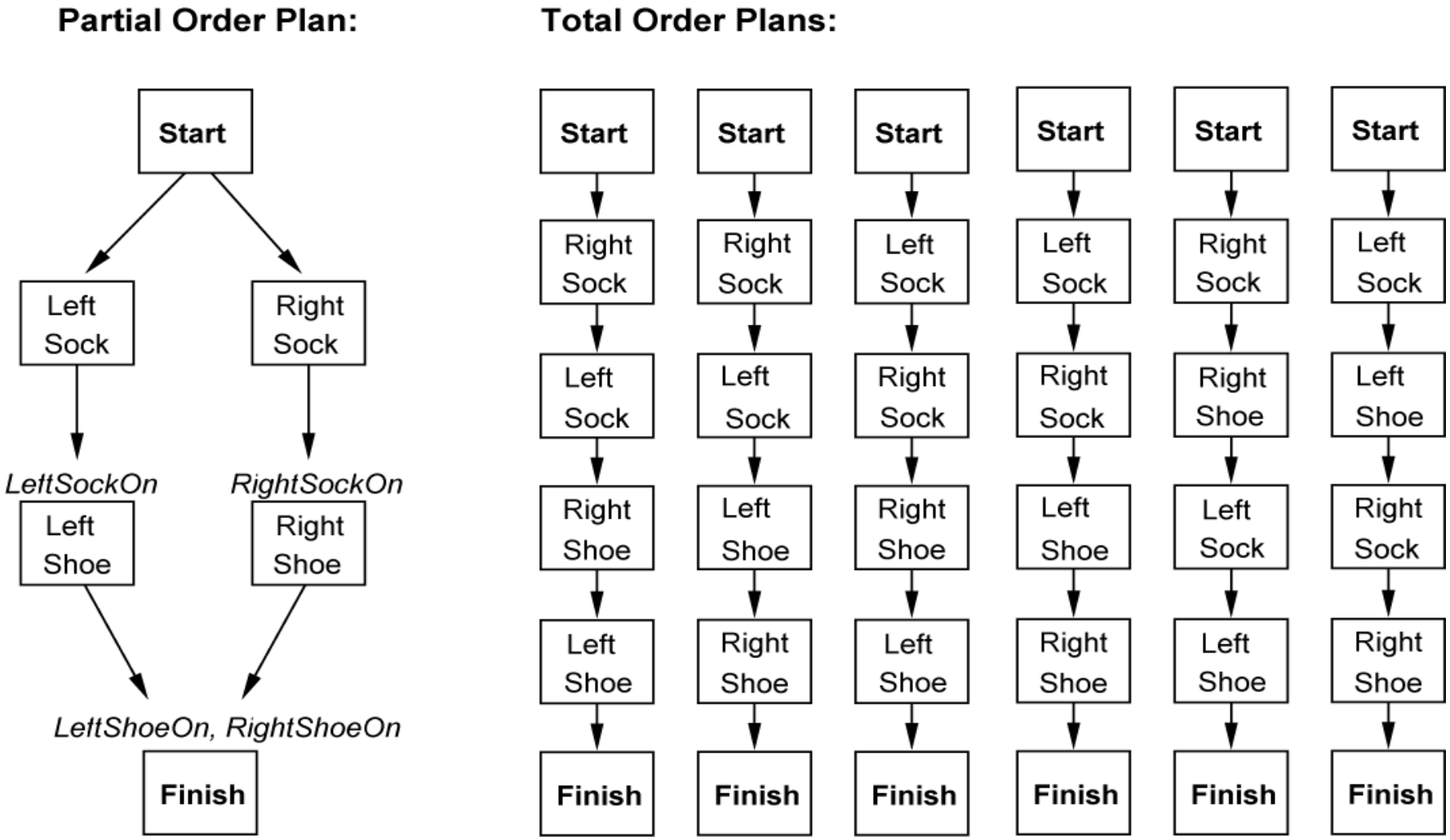
Plano de Ordem Parcial



Plano de Ordem Parcial e Total

- Plano final
  - **Completo** – toda a pré-condição de toda ação tem uma ligação causal para alguma outra ação
  - **Consistente** – não há contradições
    - nos ordenamentos das ações
    - nas atribuição de variáveis
  - mas não necessariamente totalmente ordenado e instanciado!
- Ordem total x Ordem parcial
  - lista simples com todas as ações, uma após outra
  - **Linearizar** um plano é colocá-lo na forma "ordem total"
- Instanciação completa **de um plano**
  - todas variáveis são instanciadas

Exemplo de Linearização



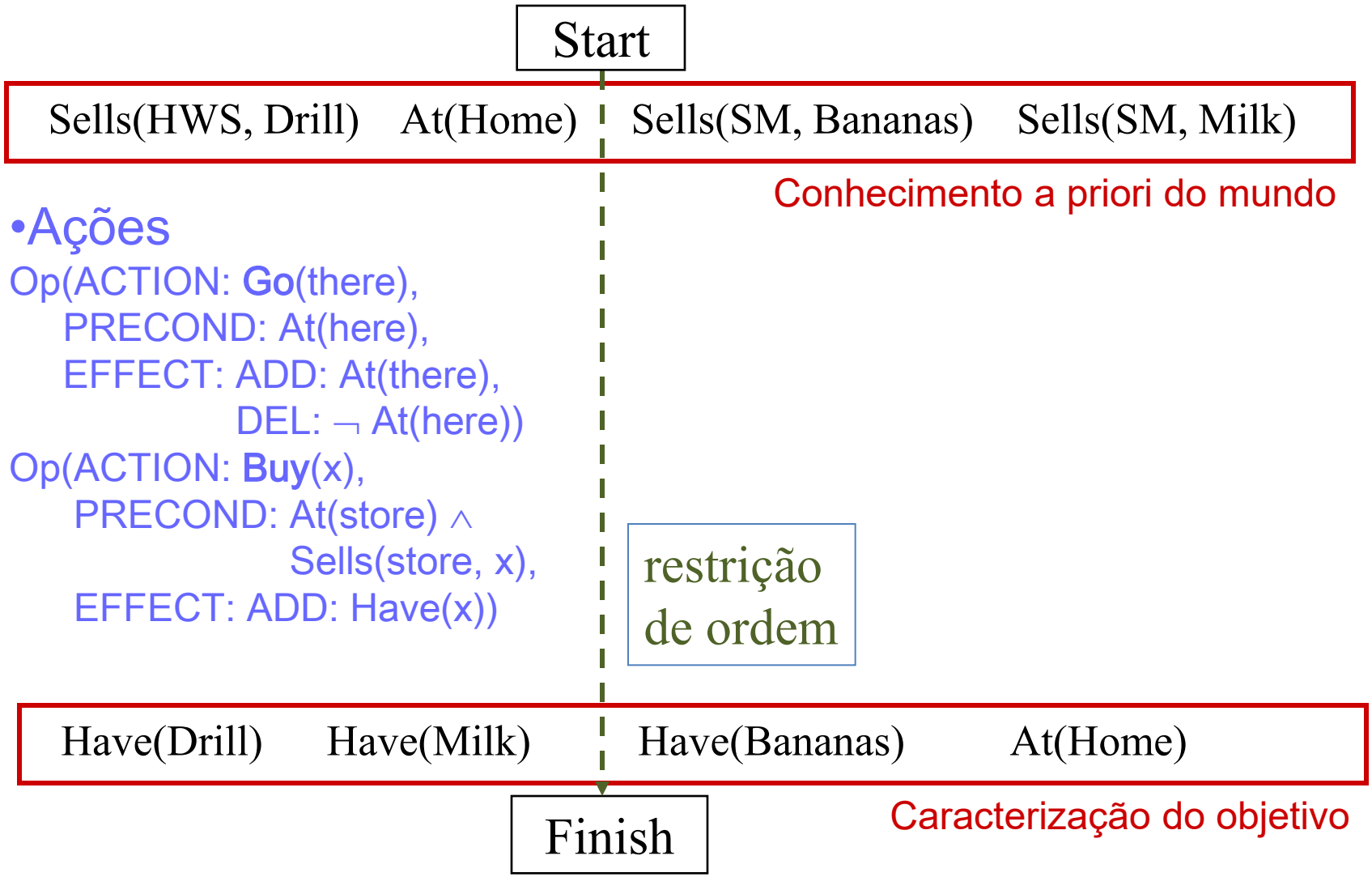
Princípio do Menor Engajamento

- Por quê não tornar o plano totalmente ordenado e instanciado?
  - Princípio do menor engajamento **(least commitment planning)**
    - não faça hoje o que você pode fazer amanhã
    - ordem e instanciação parcial são decididas quando necessário
    - evita-se backtracking!
  - Exemplo
    - para objetivo Have(Milk), a ação Buy(item, store), instancia-se somente o item ➔ Buy (Milk,store)
    - para as meias/sapatos: vestir cada meia antes do sapato, sem dizer por onde começa (esq ou dir)

POP (Partial Order Planning)

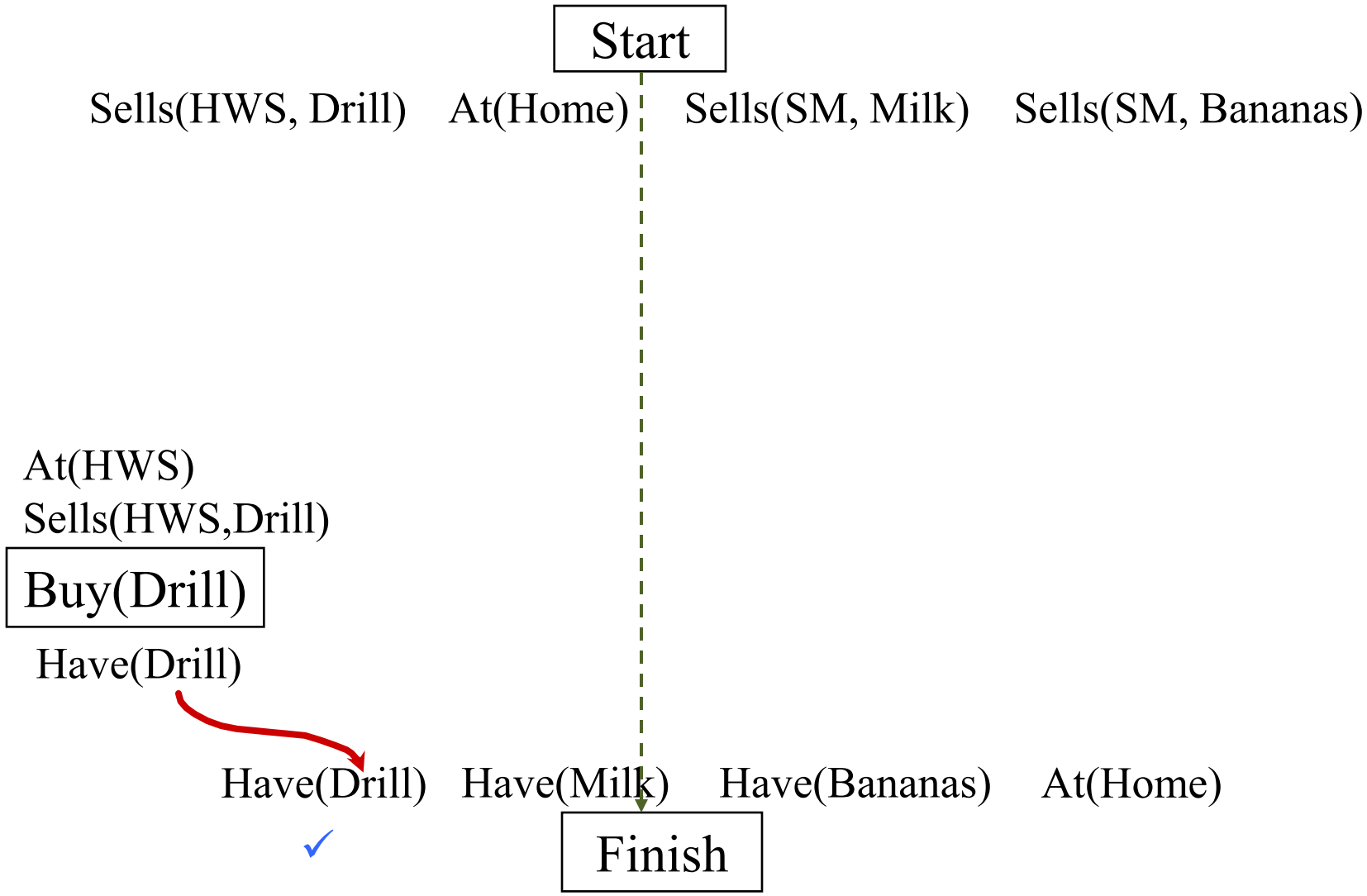
- Existindo a linguagem (STRIPS), falta o algoritmo...
- Características do POP
  - algoritmo não determinista
  - a inserção de uma ação só é considerada se atender uma pré-condição aberta
  - planejador regressivo (do objetivo para o início)
  - é correto e completo, assumindo busca em largura ou em profundidade iterativa
- Ideia do algoritmo
  - identifica ação com pré-condição aberta
  - introduz ação cujo efeito é satisfazer esta pré-condição
  - instancia variáveis e atualiza os links causais
  - verifica se há ameaças e corrige o plano se for o caso

POP: Exemplo das Compras

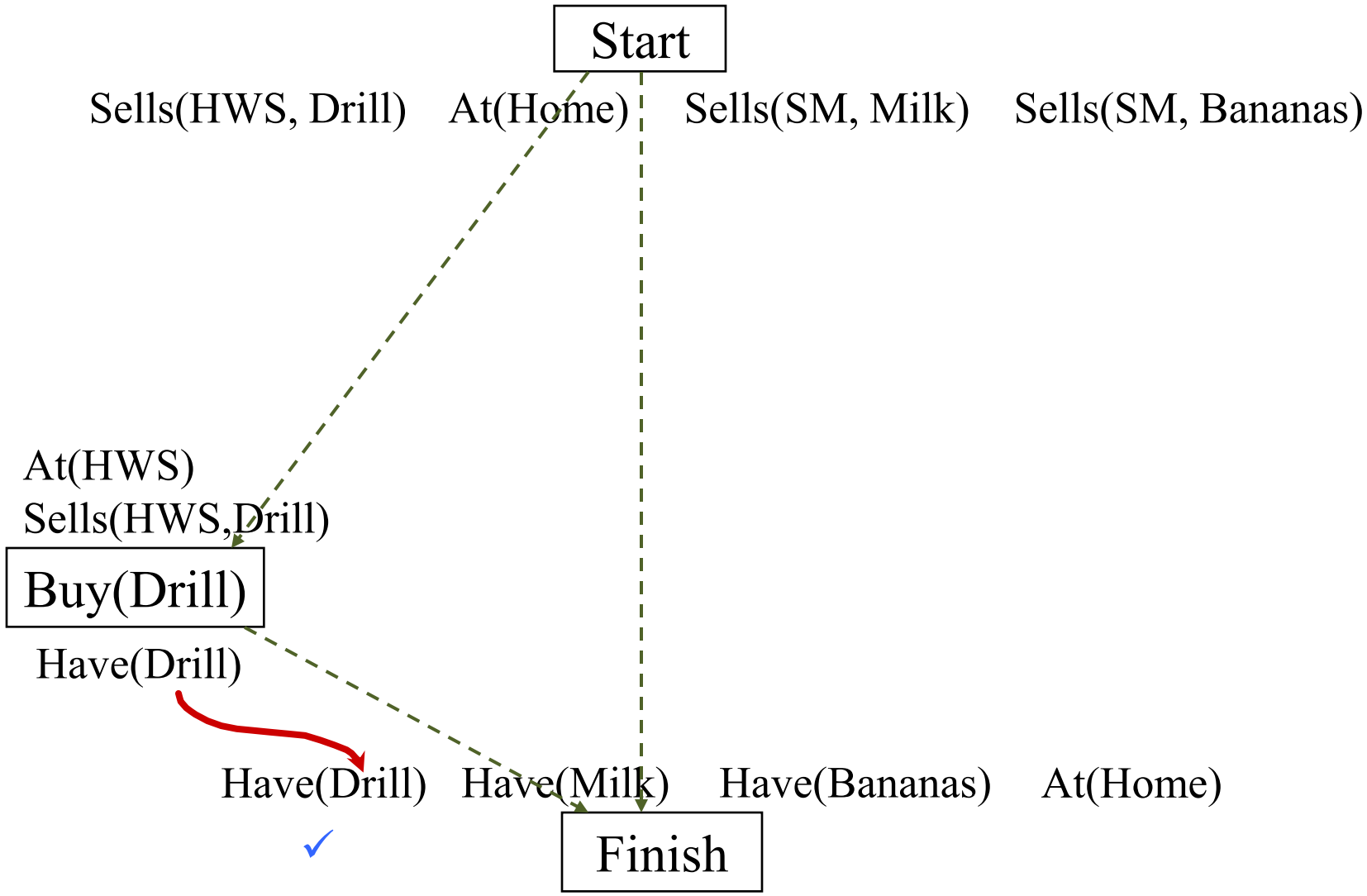


POP: Exemplo das Compras

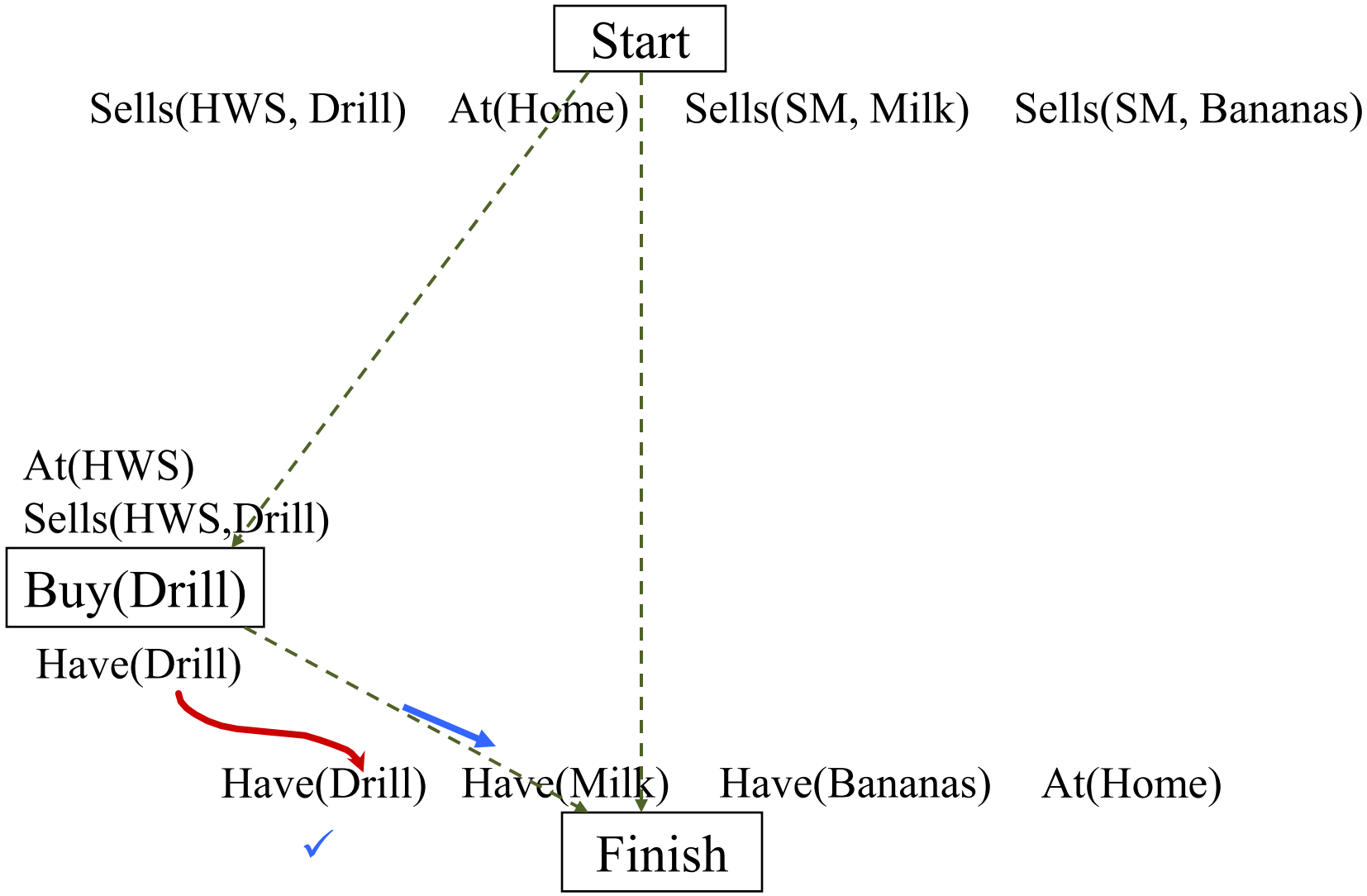
POP: Exemplo das Compras



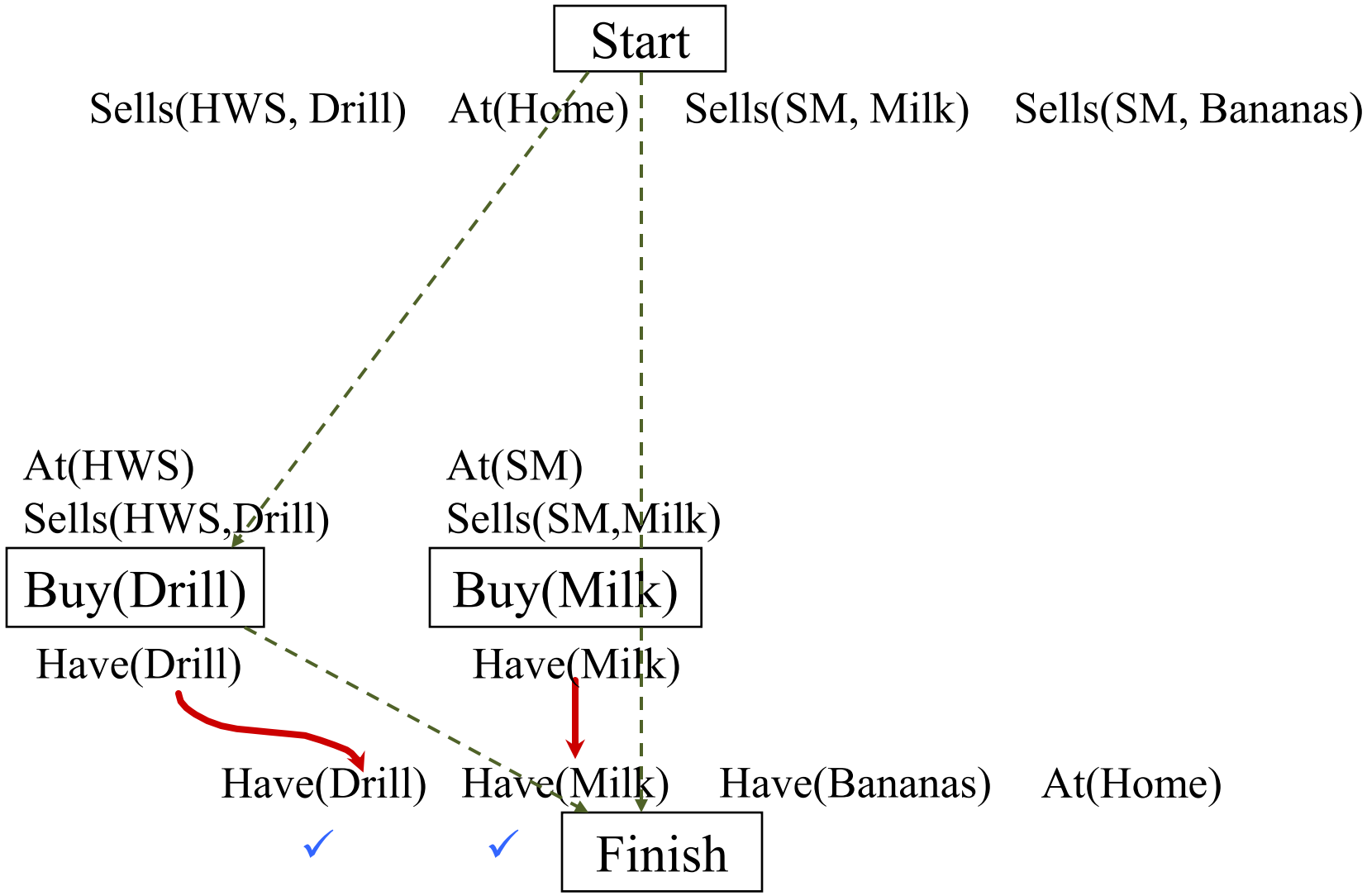
POP: Exemplo das Compras



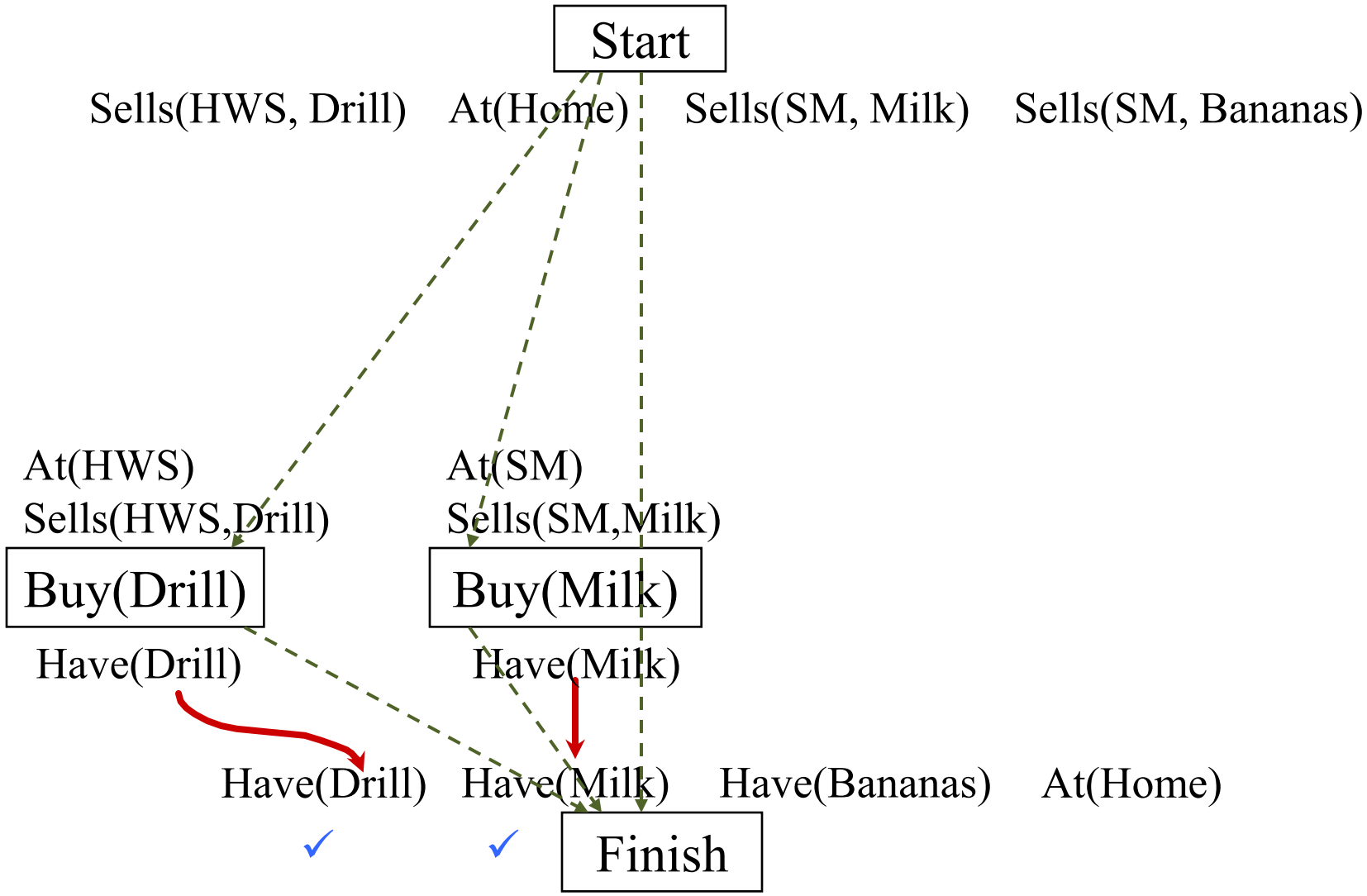
POP: Exemplo das Compras



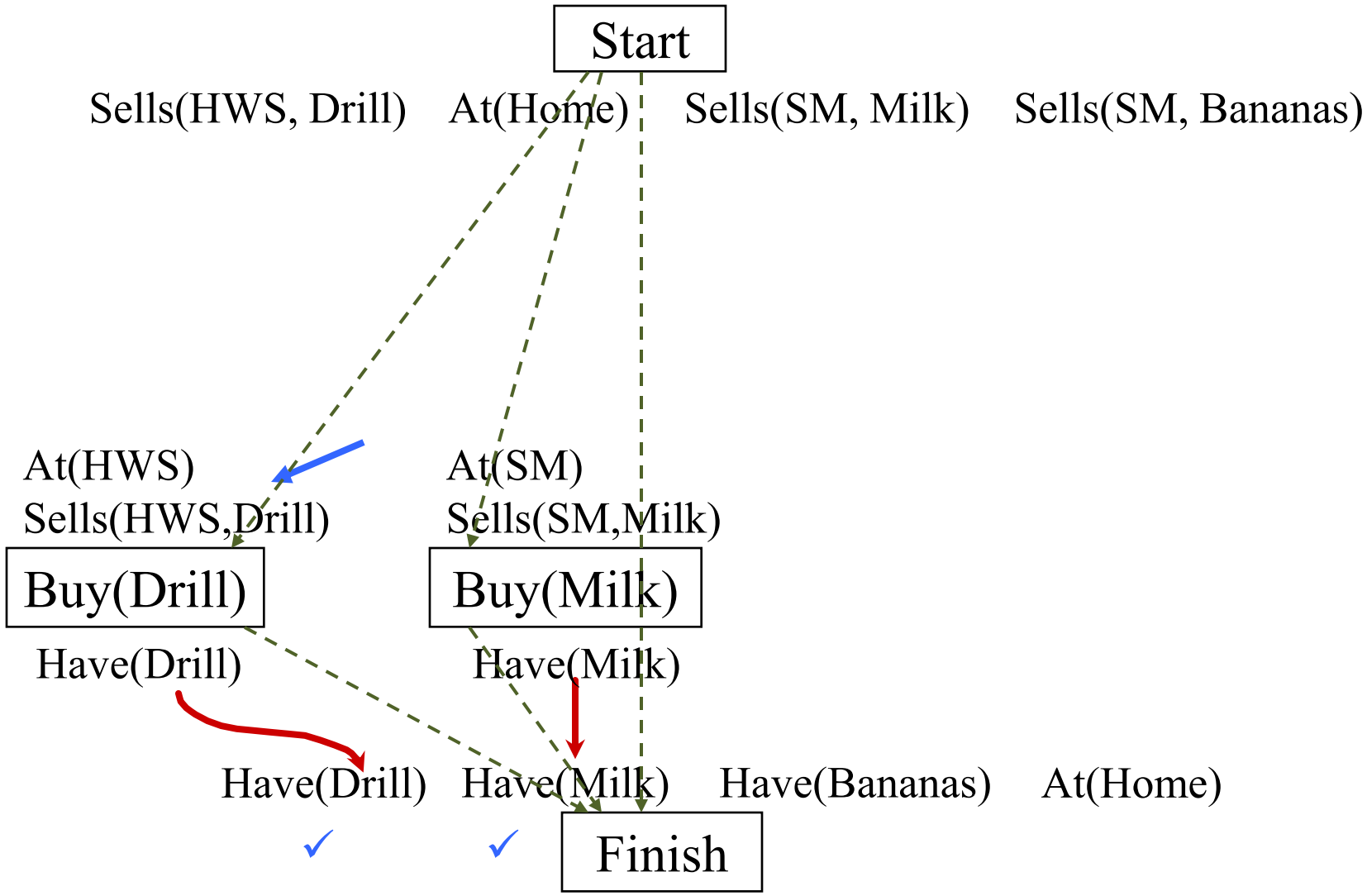
POP: Exemplo das Compras



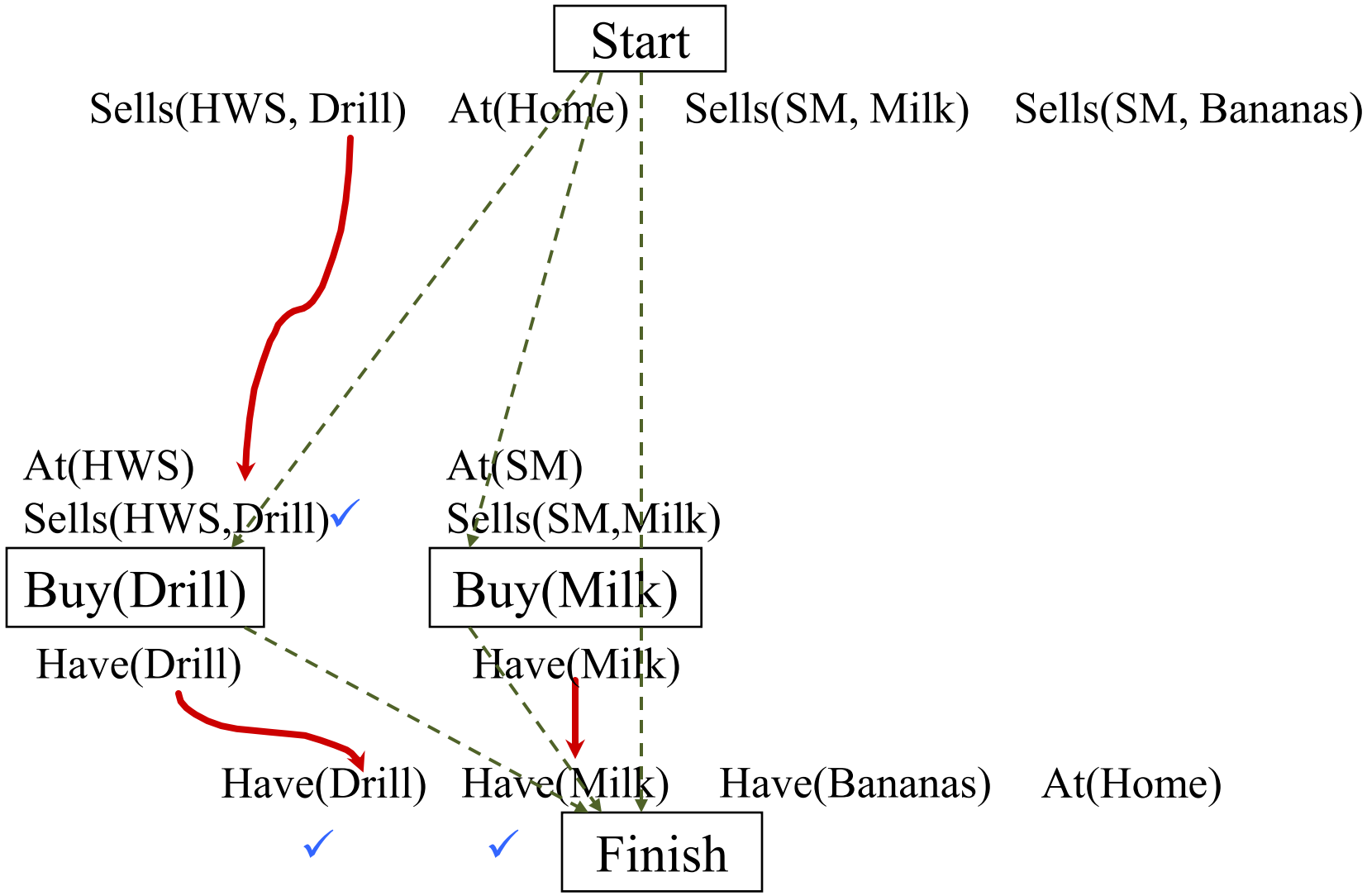
POP: Exemplo das Compras



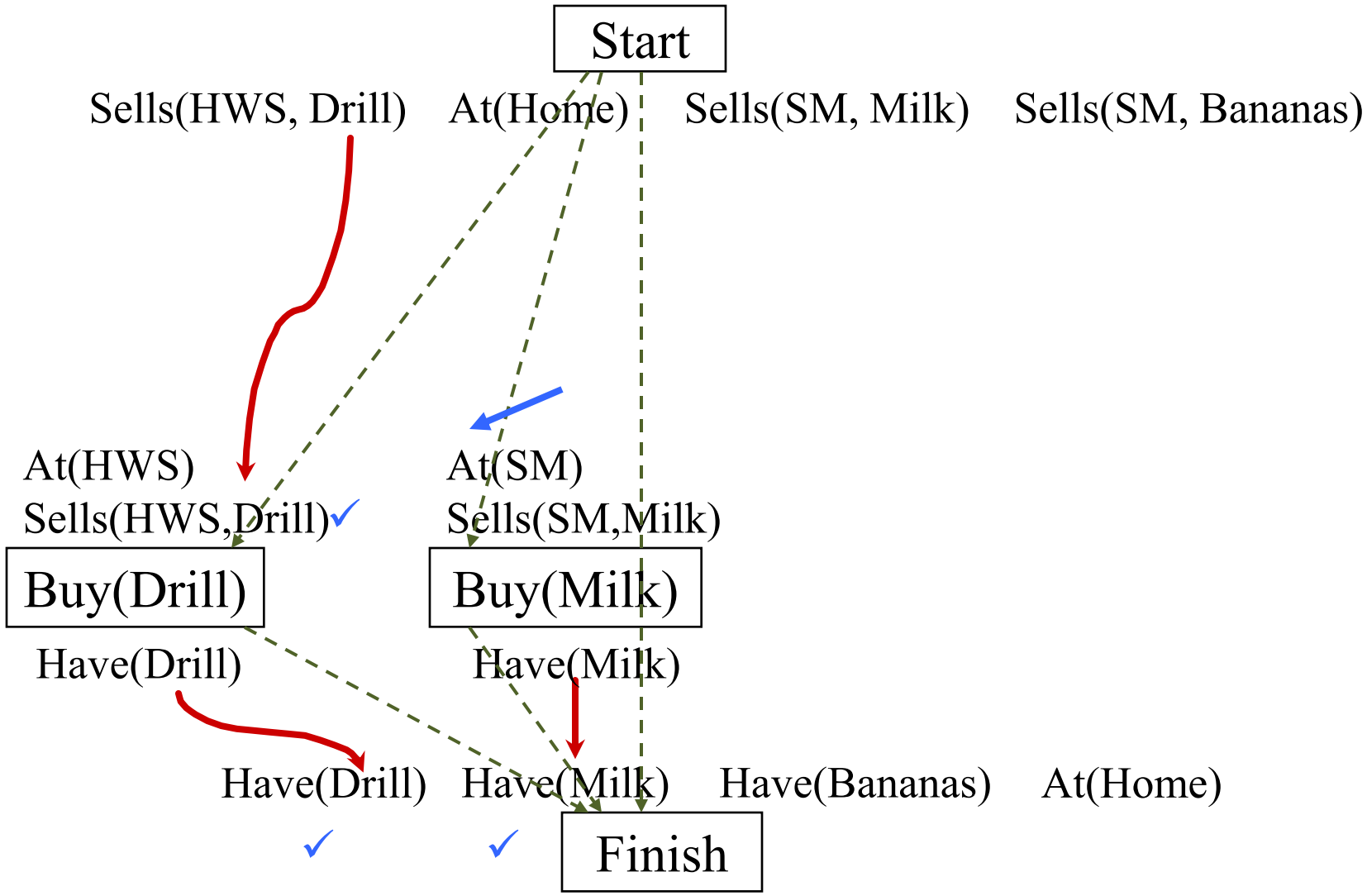
POP: Exemplo das Compras



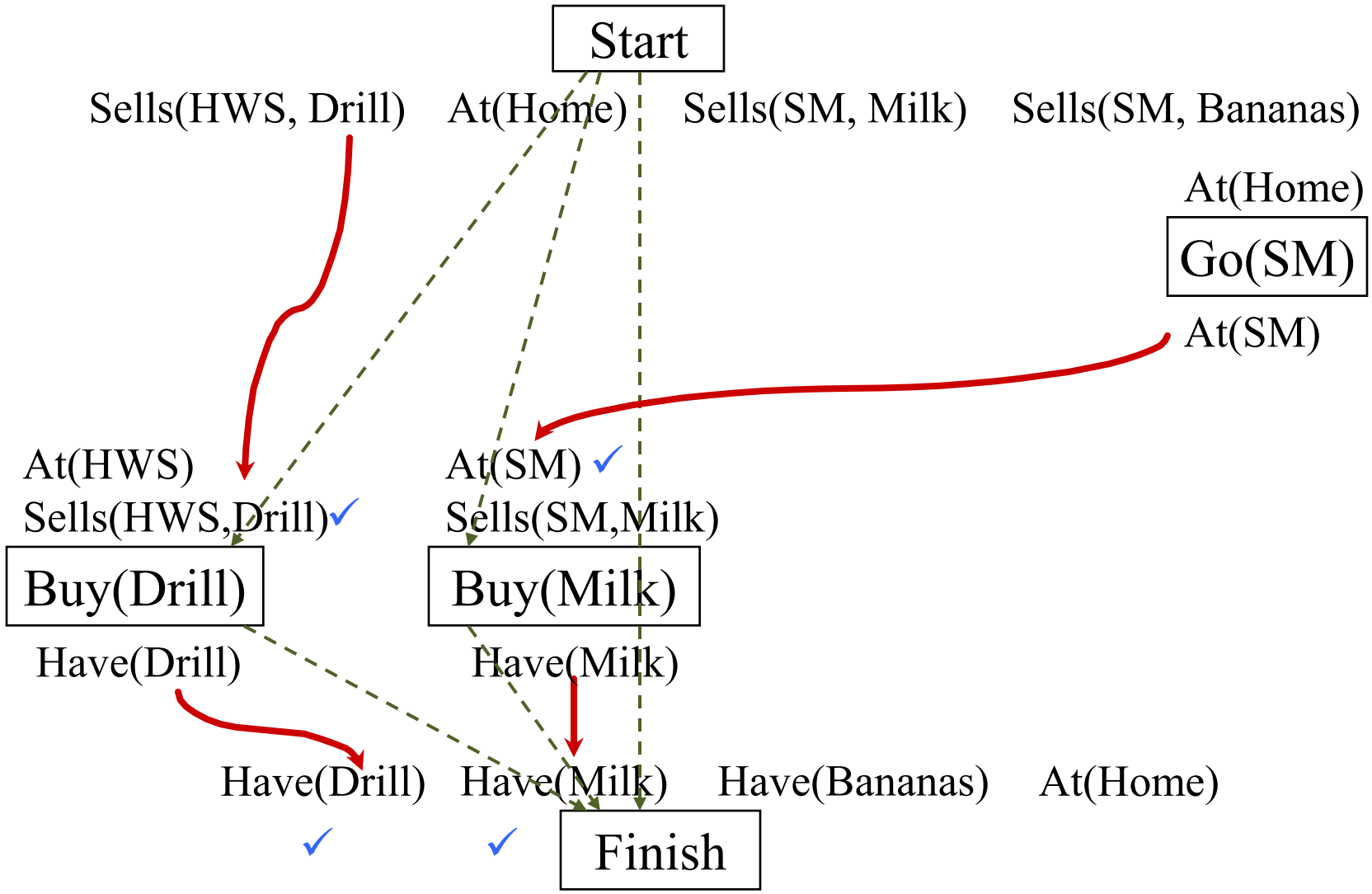
POP: Exemplo das Compras



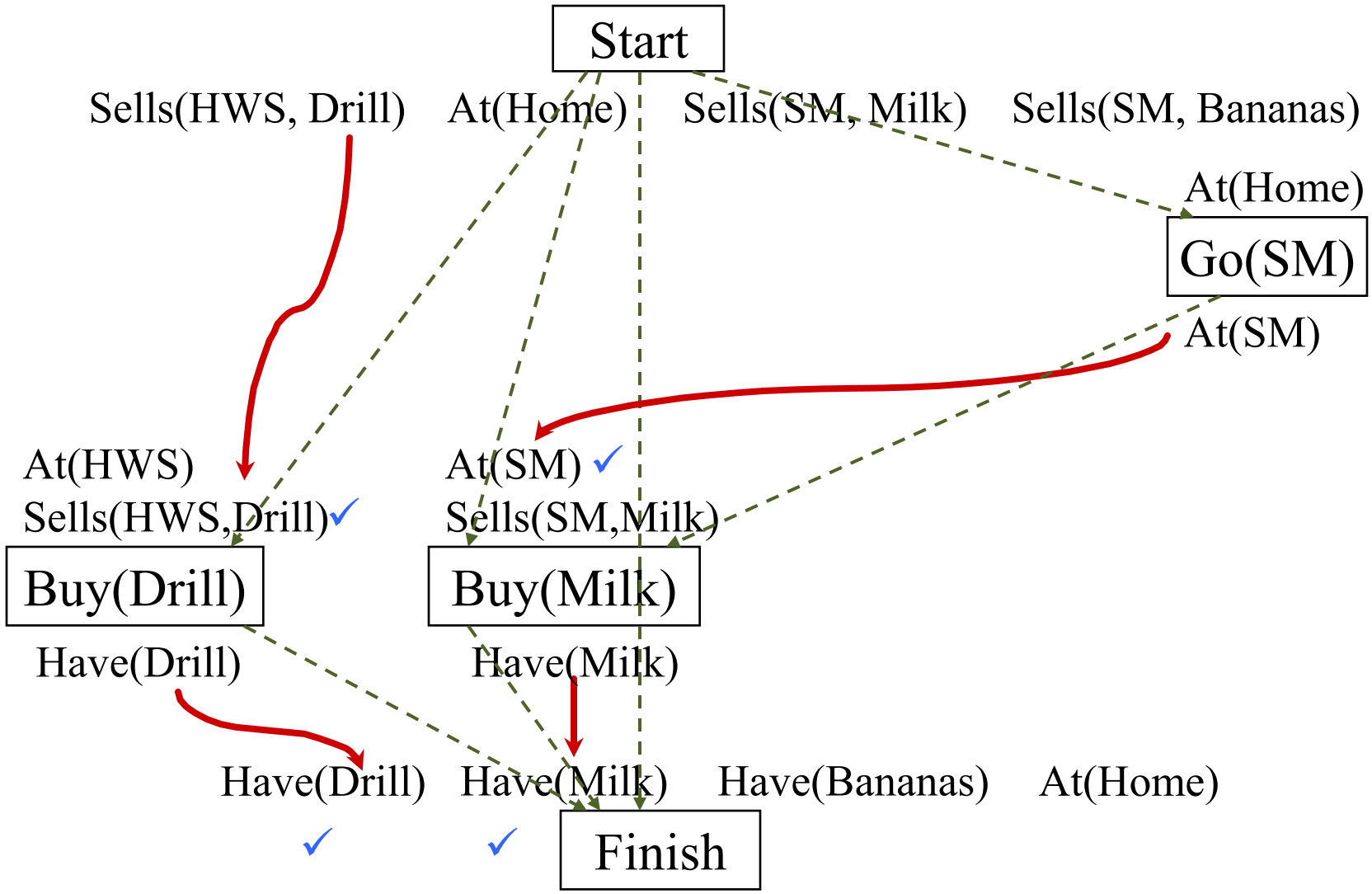
POP: Exemplo das Compras



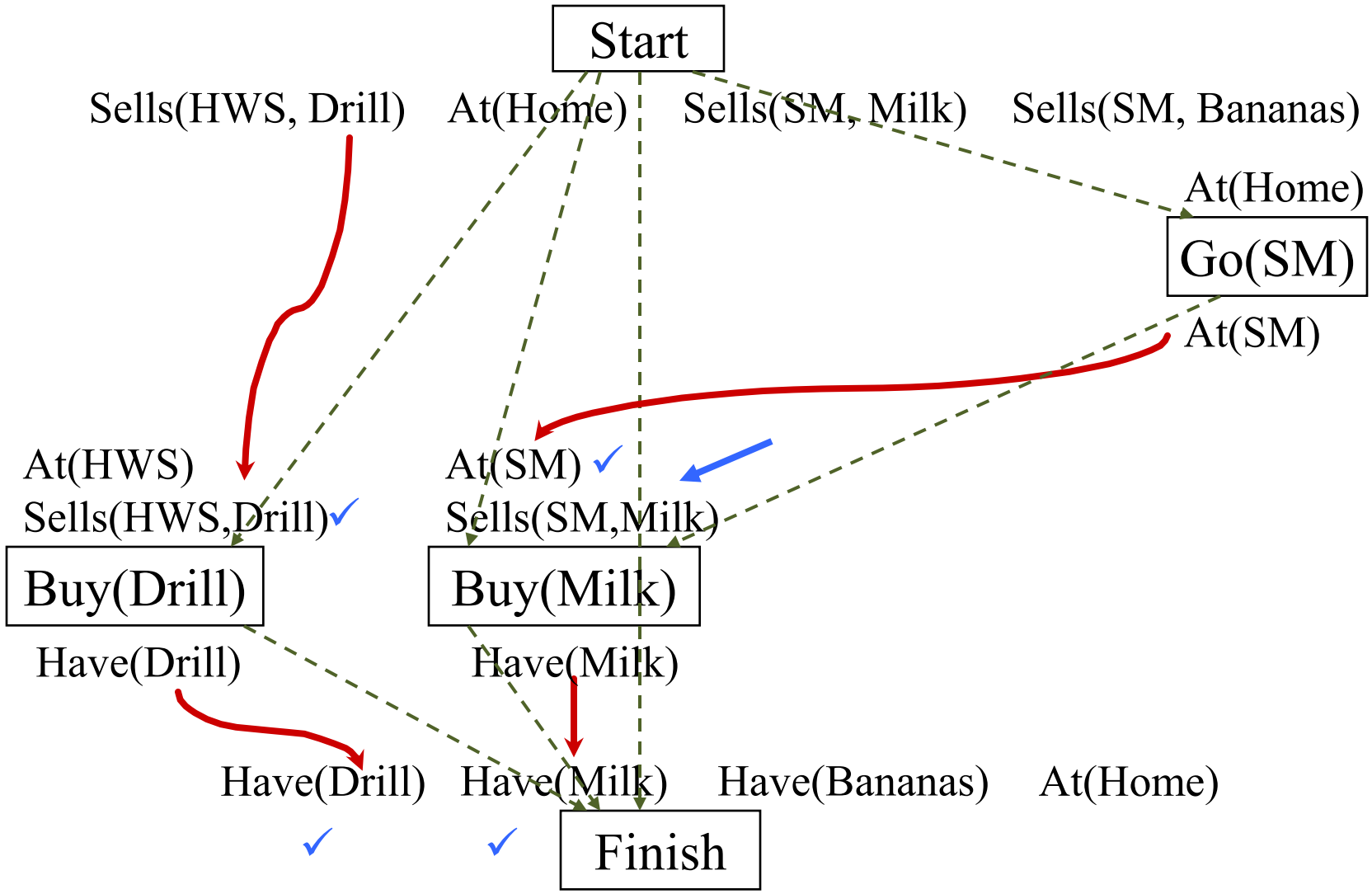
POP: Exemplo das Compras



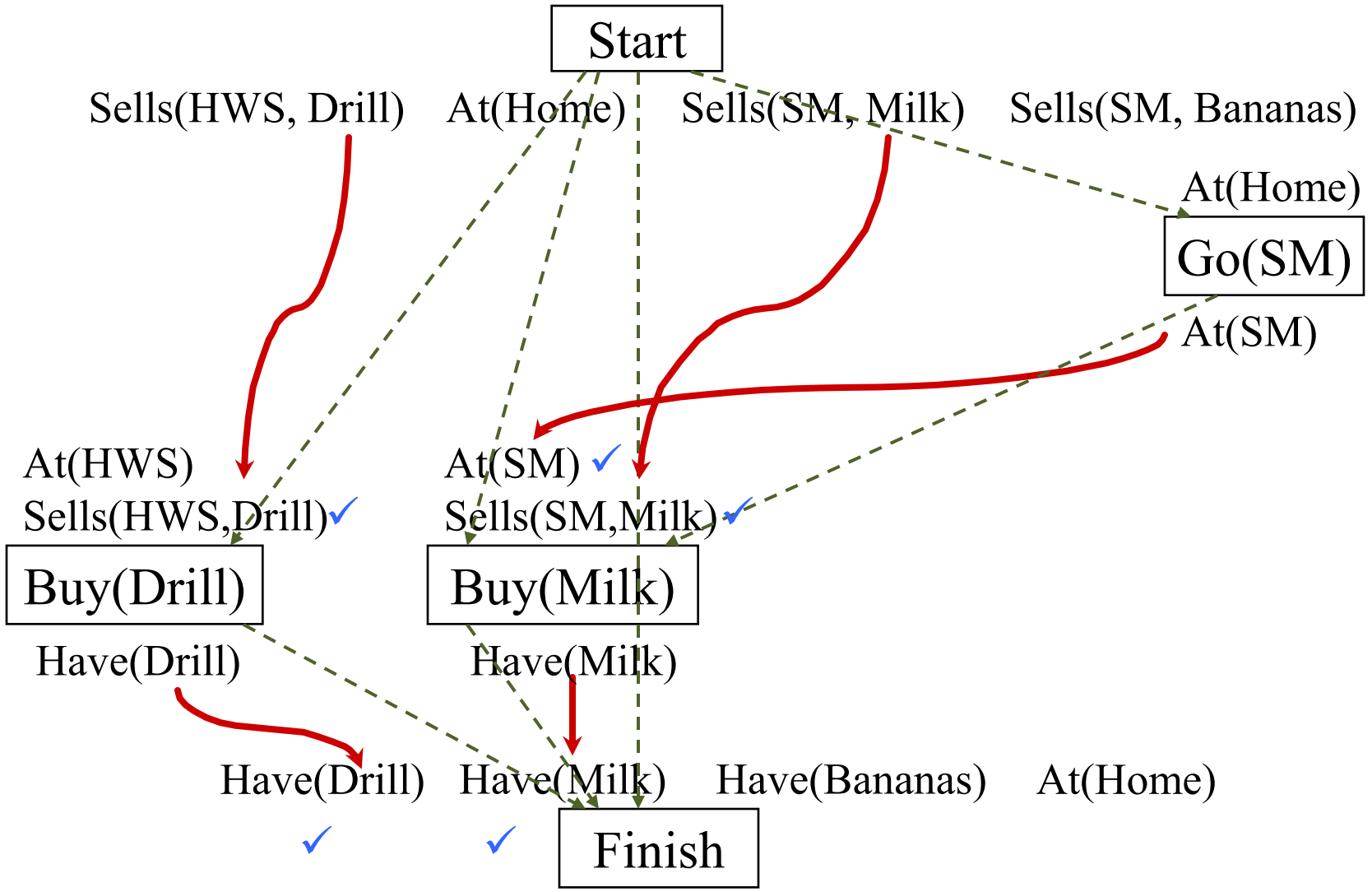
POP: Exemplo das Compras



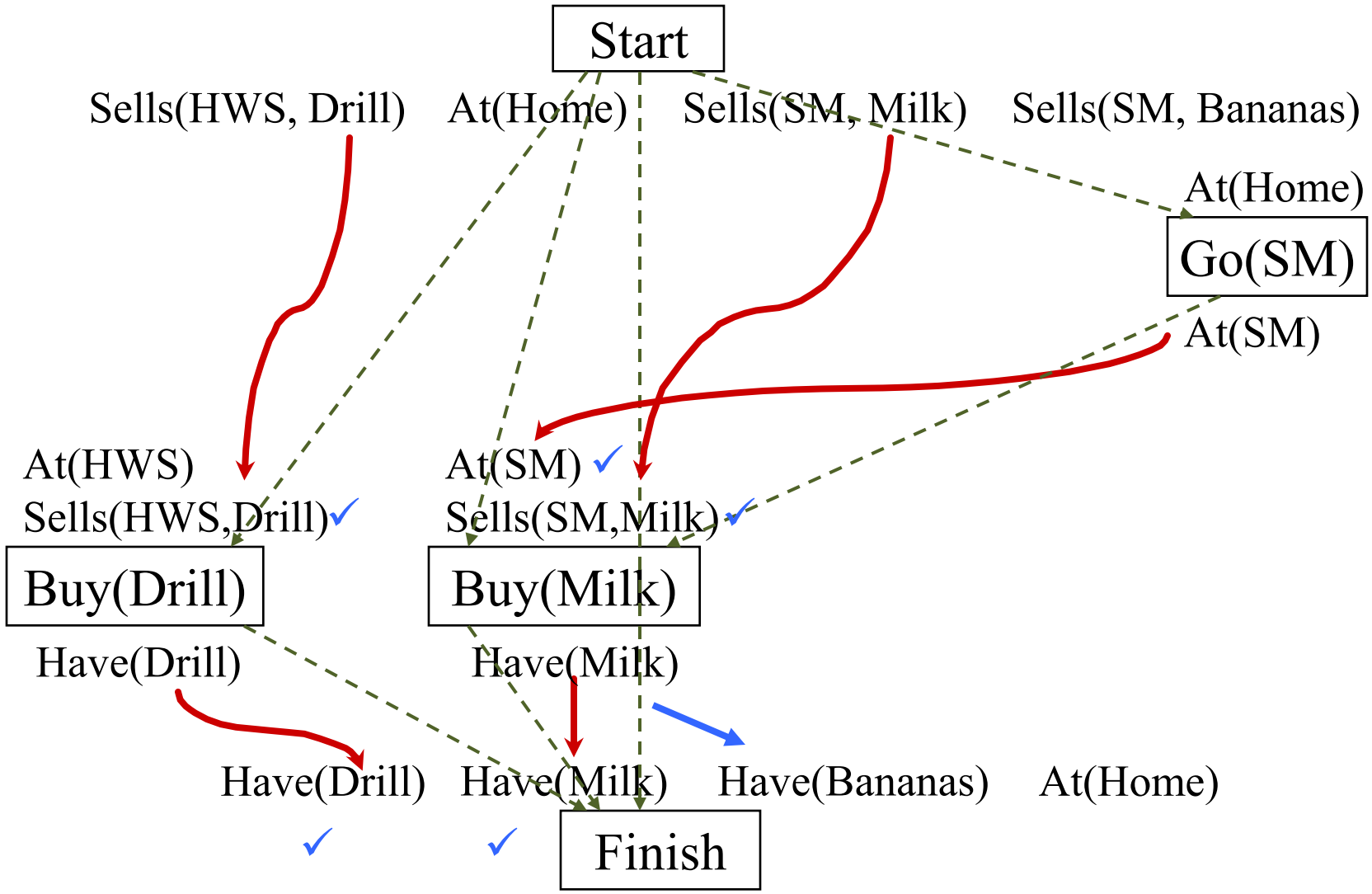
POP: Exemplo das Compras



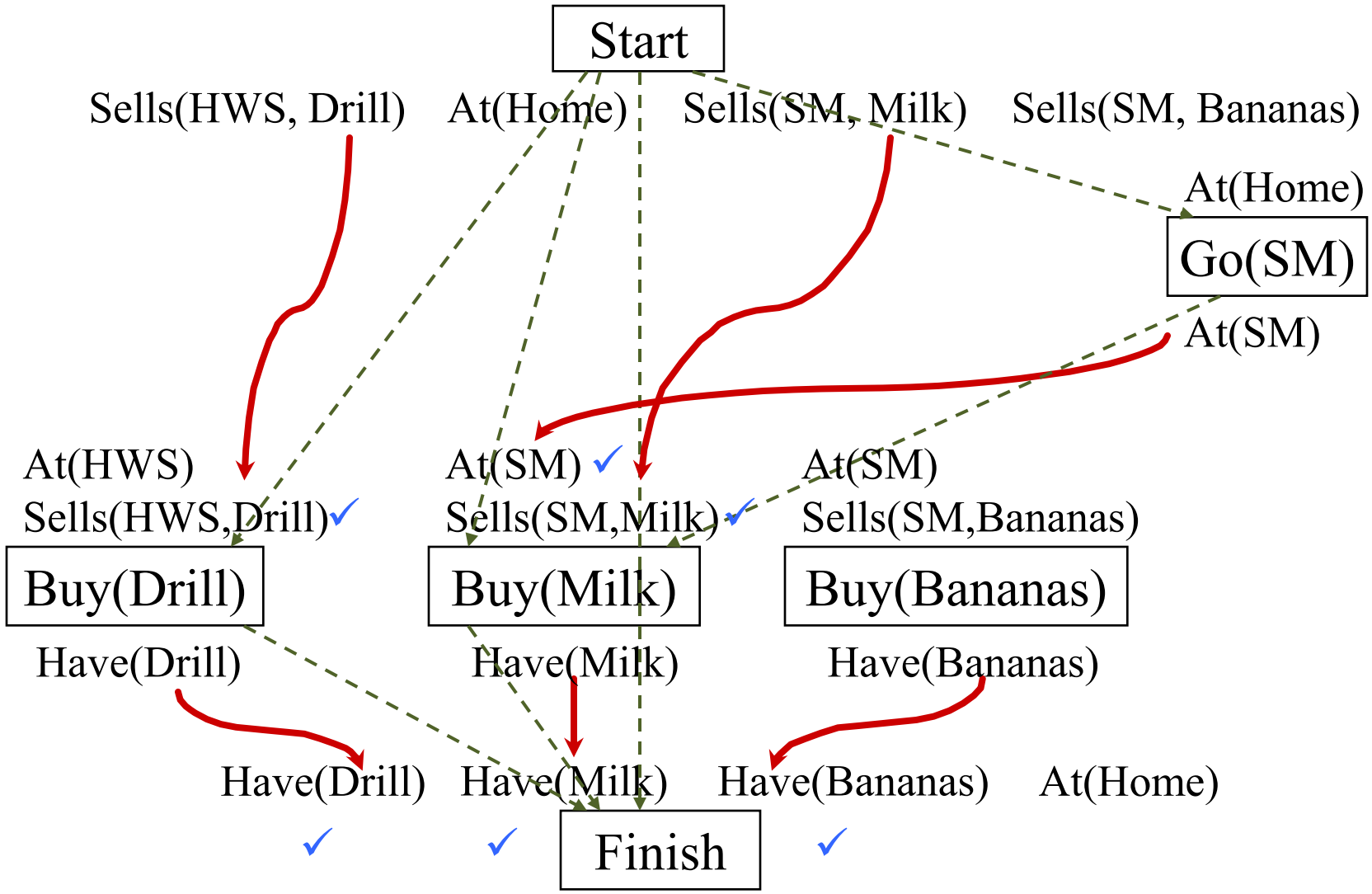
POP: Exemplo das Compras



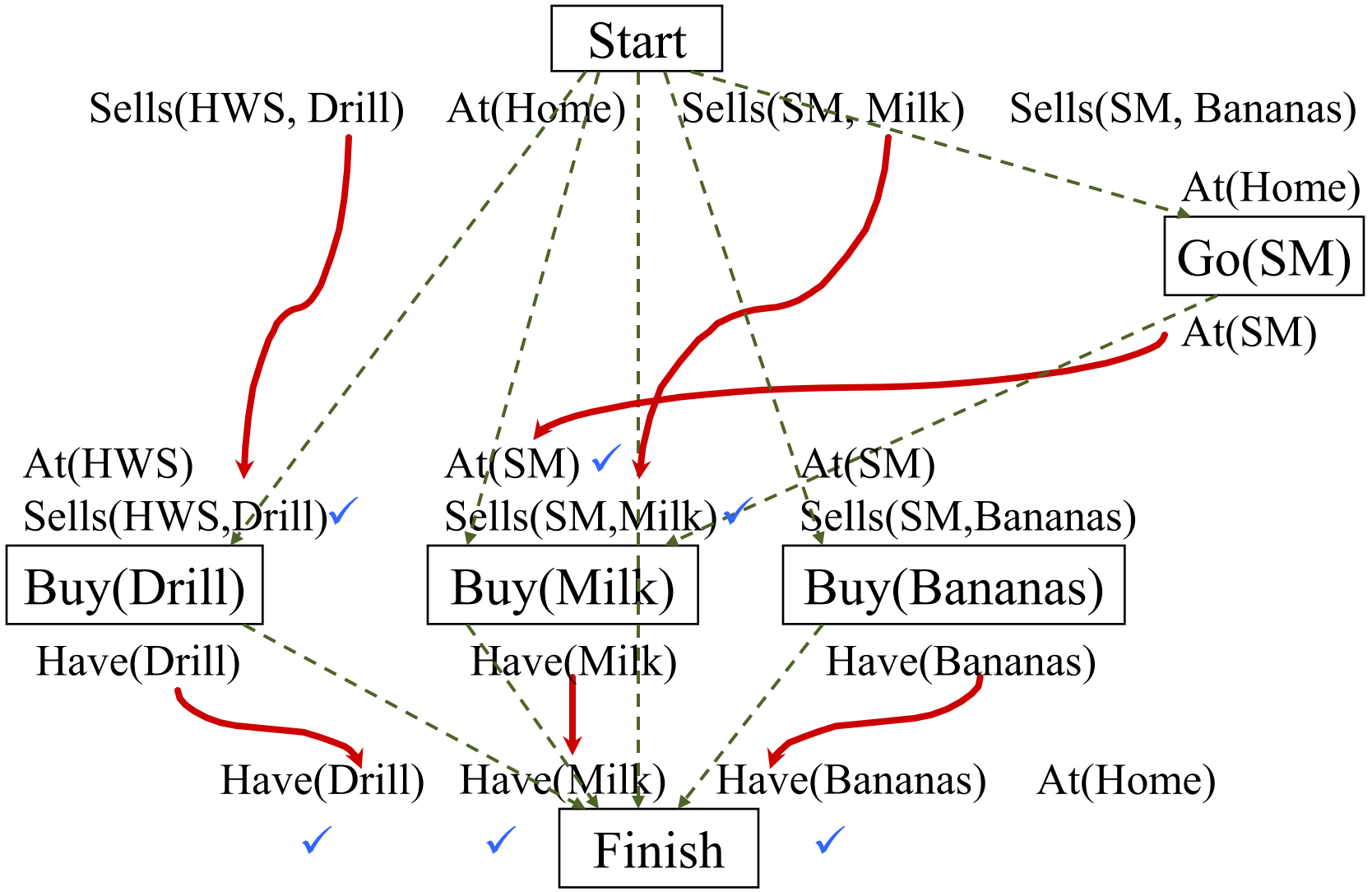
POP: Exemplo das Compras



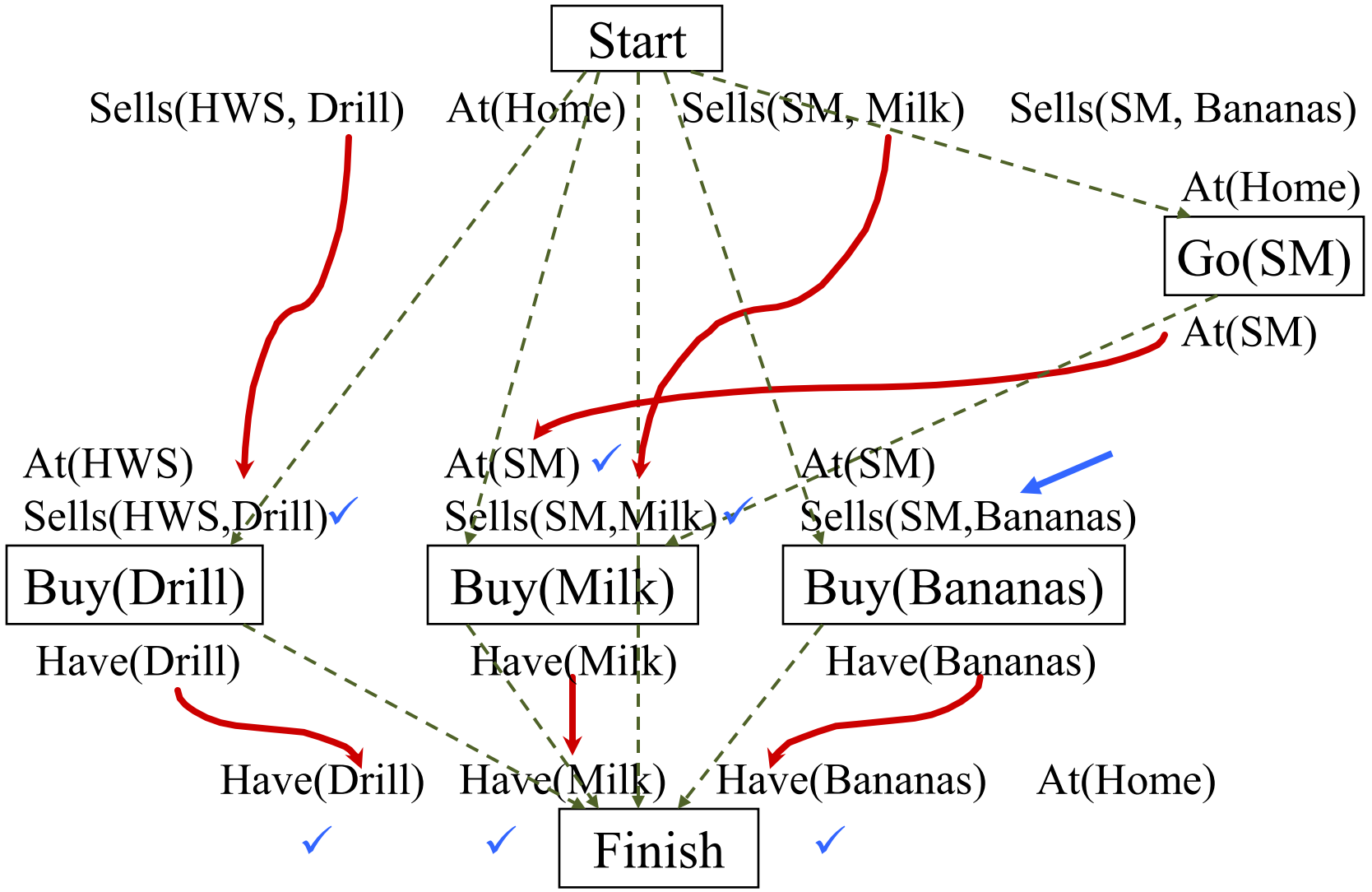
POP: Exemplo das Compras



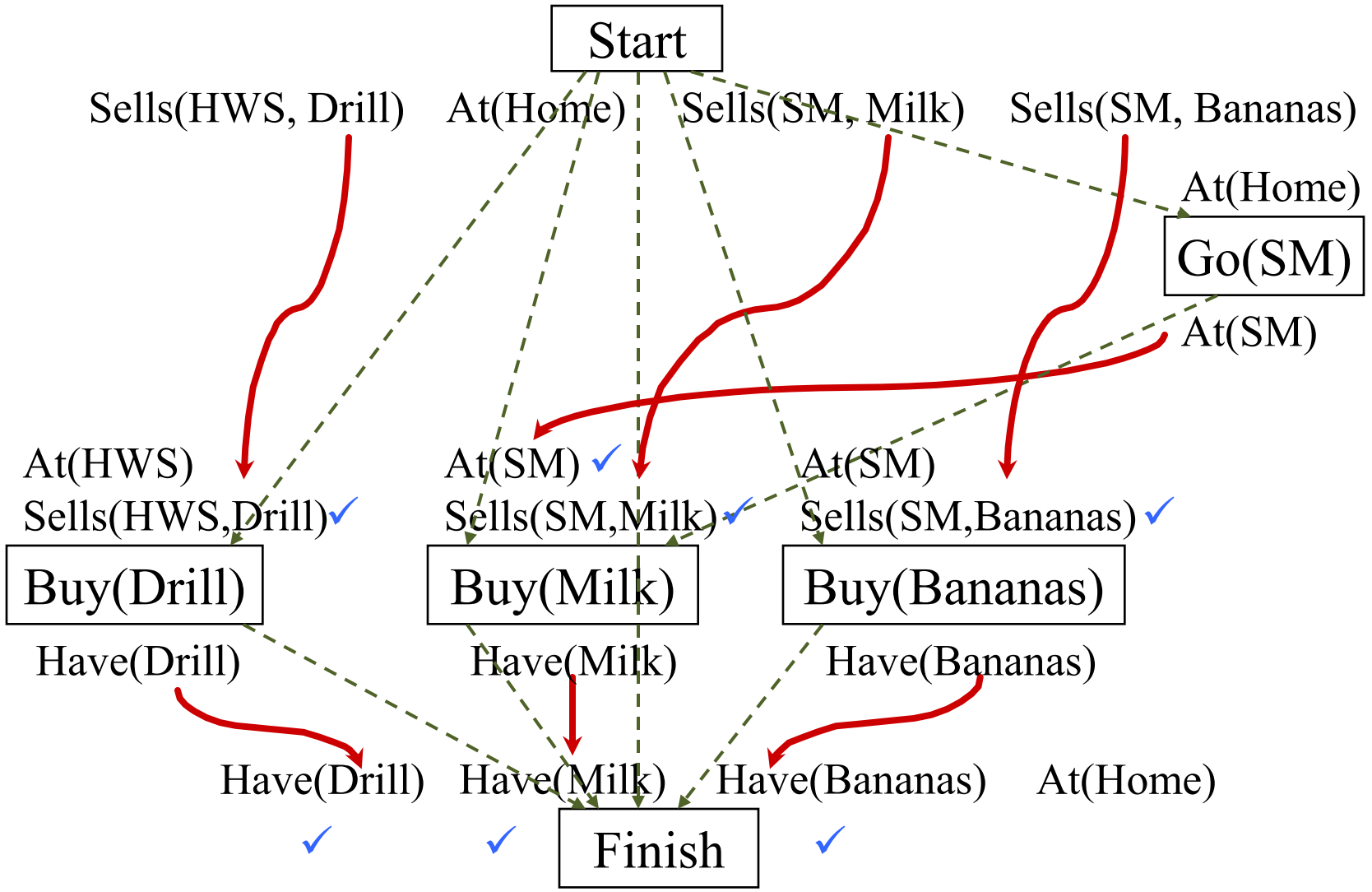
POP: Exemplo das Compras



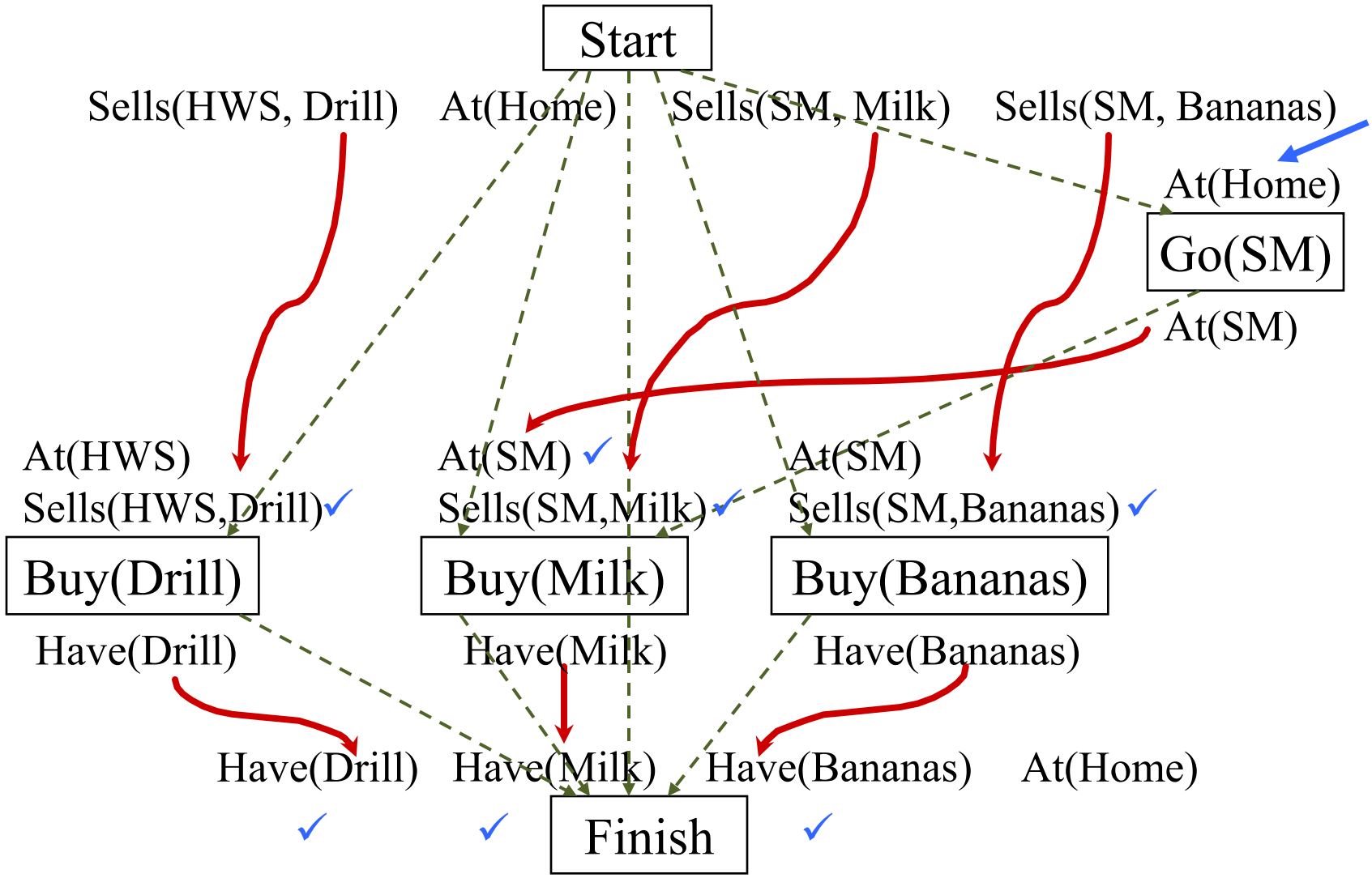
POP: Exemplo das Compras



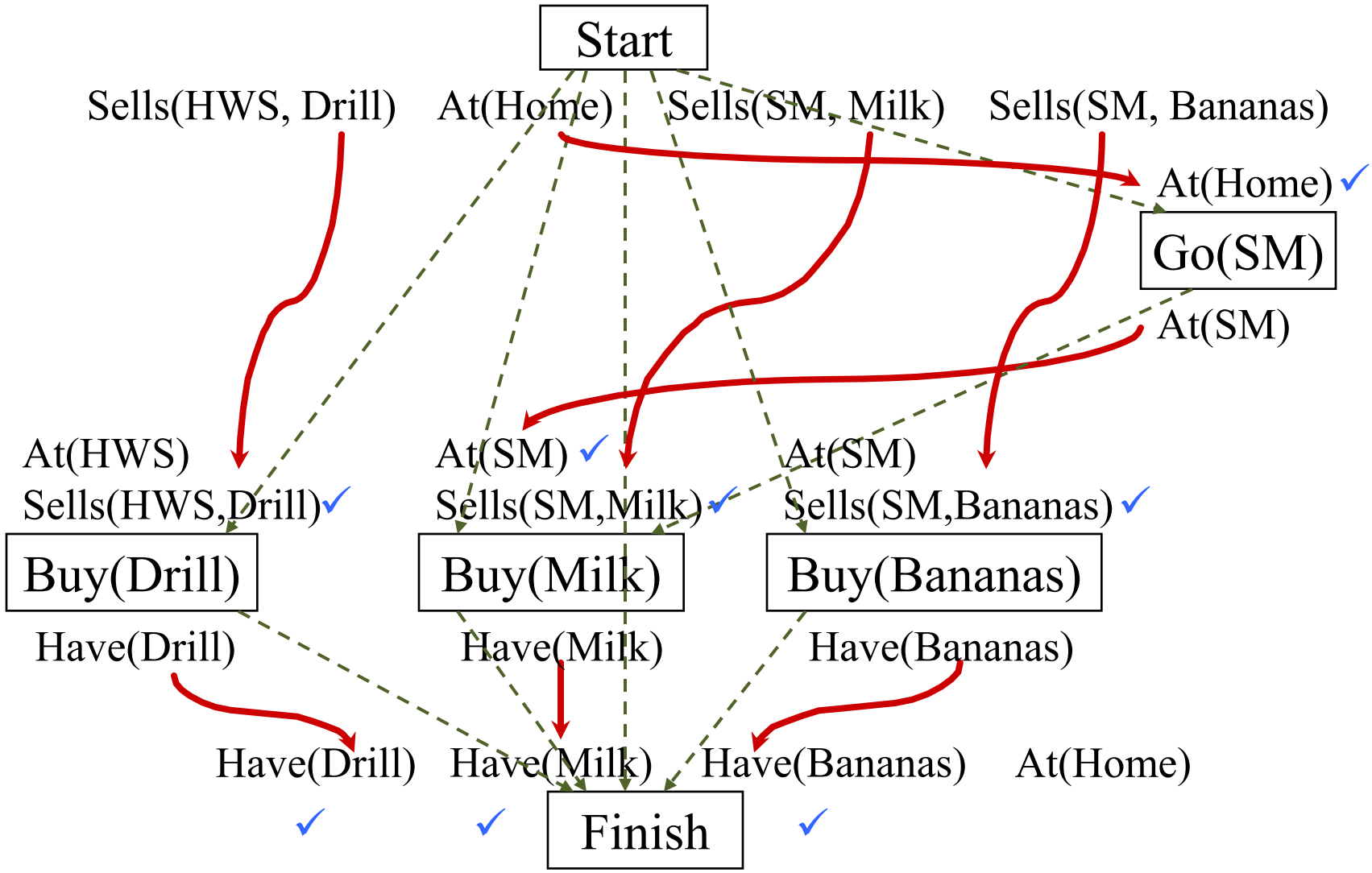
POP: Exemplo das Compras



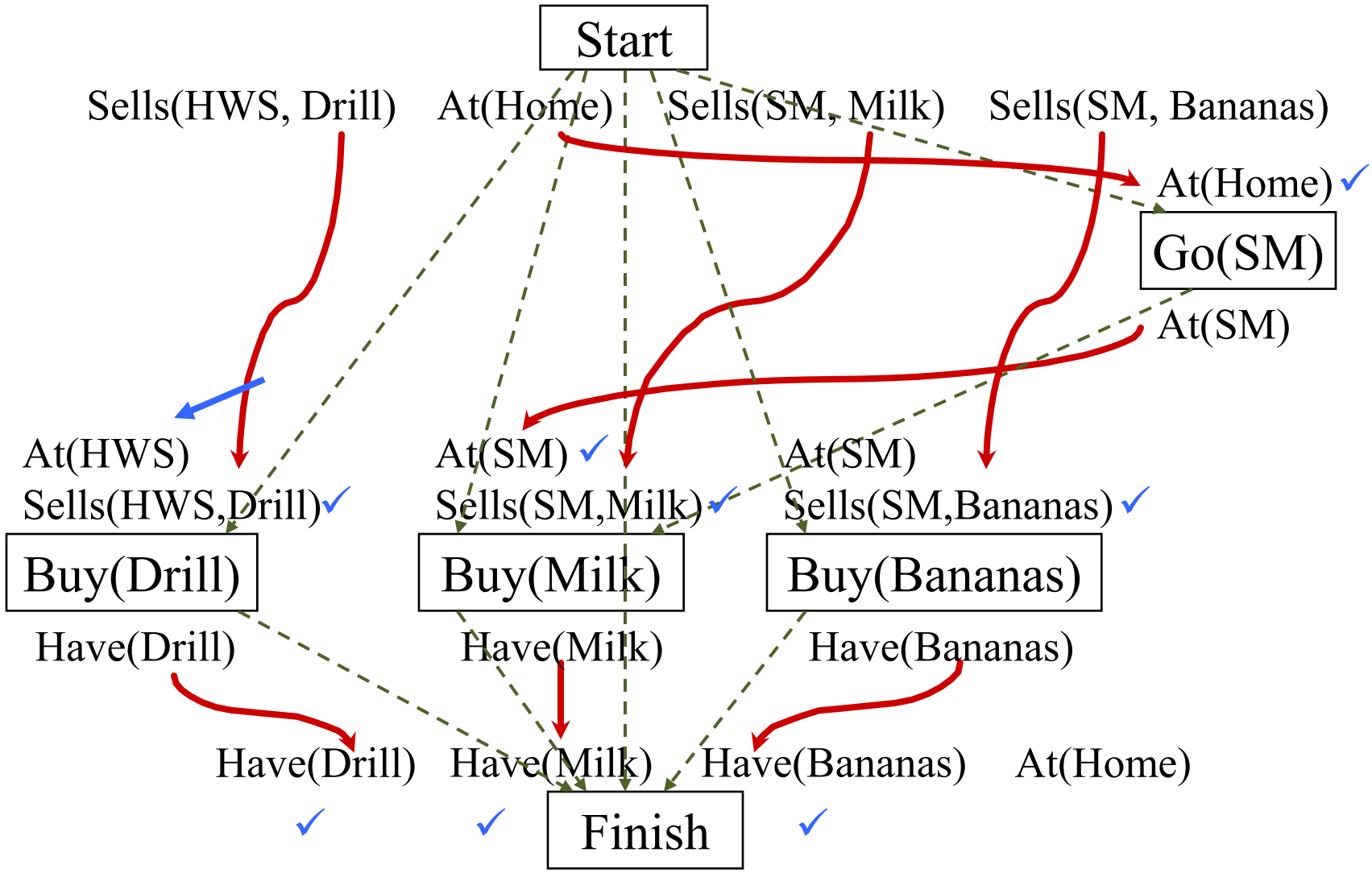
POP: Exemplo das Compras



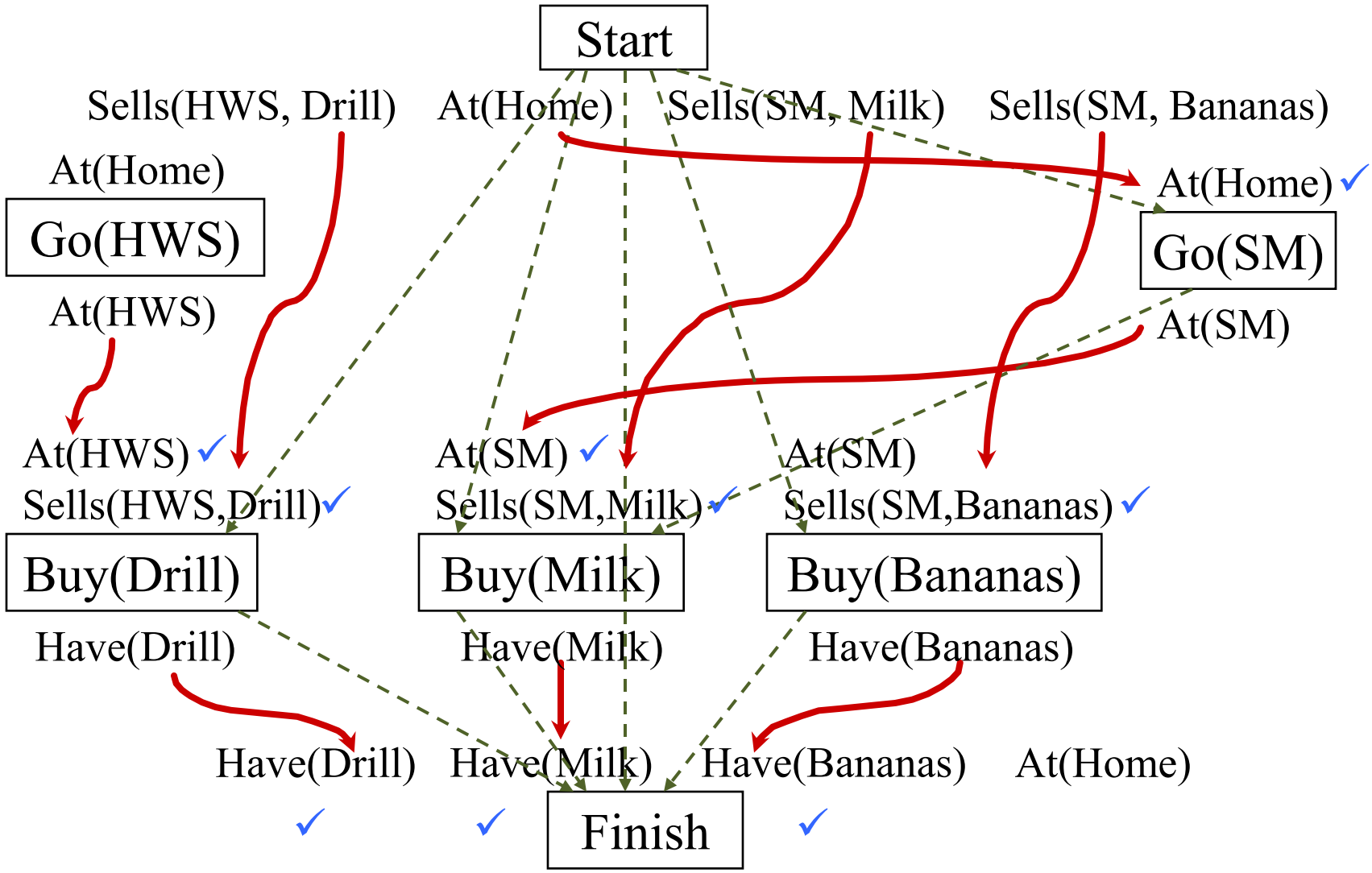
POP: Exemplo das Compras



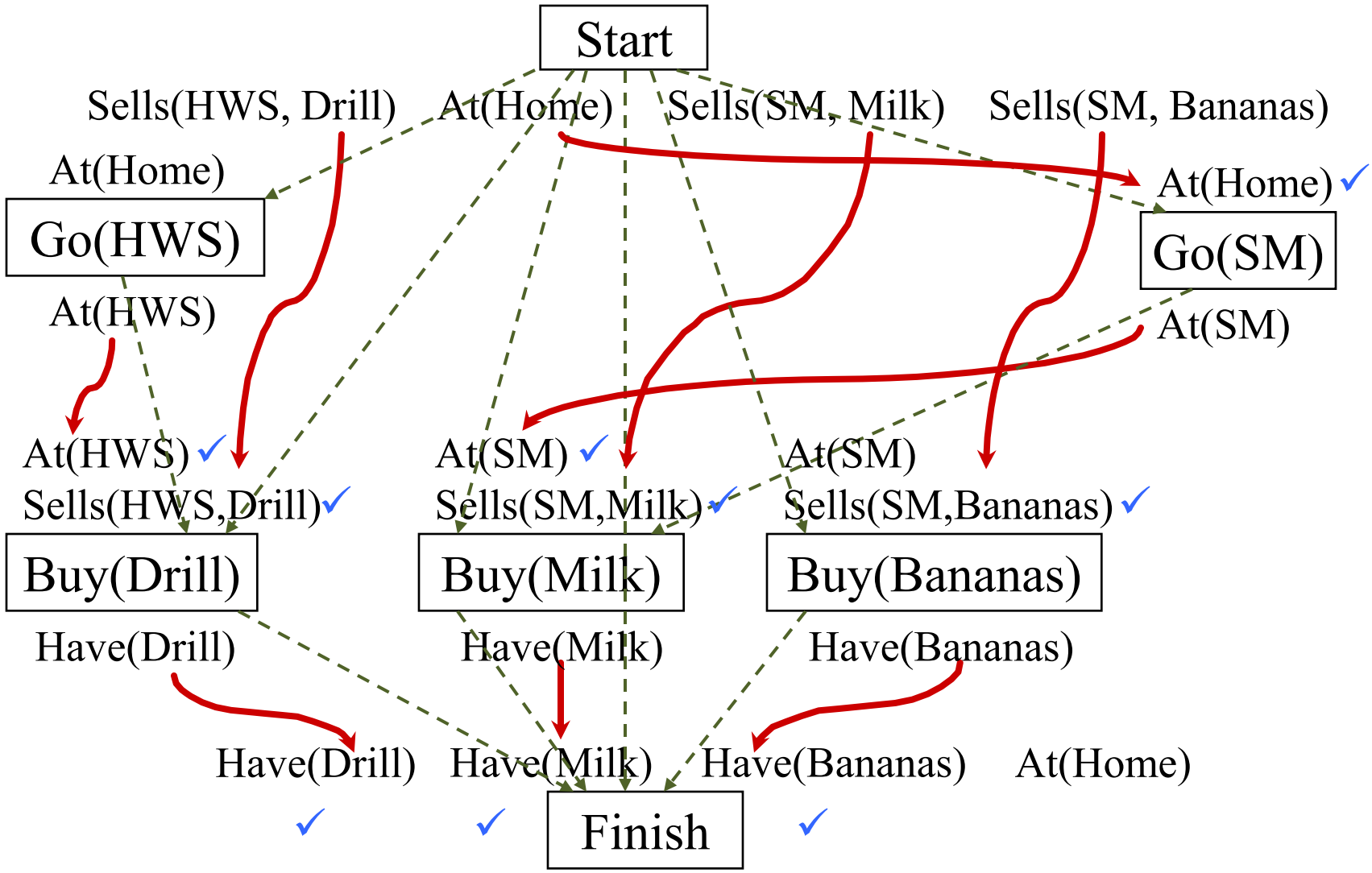
POP: Exemplo das Compras



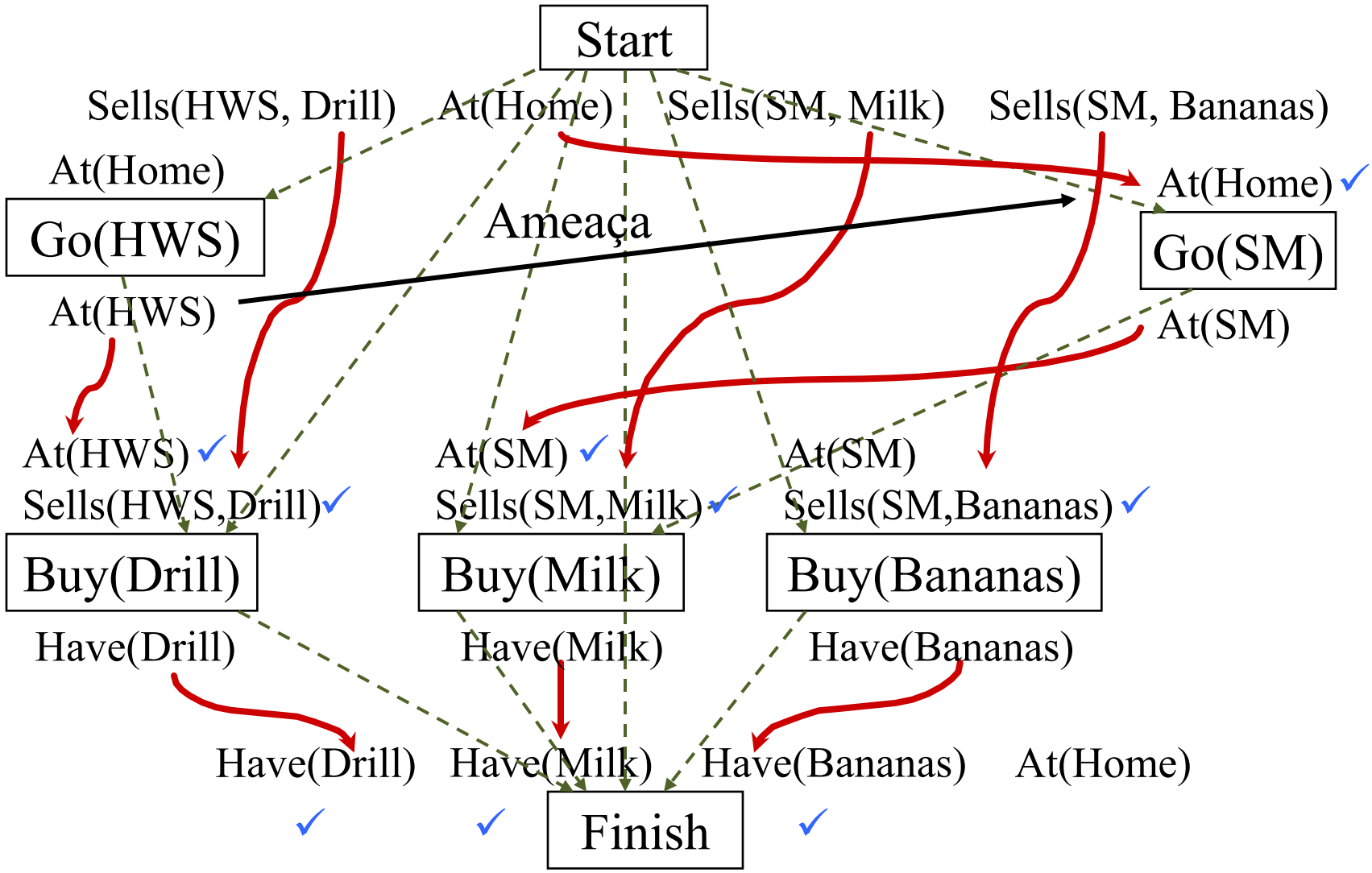
POP: Exemplo das Compras



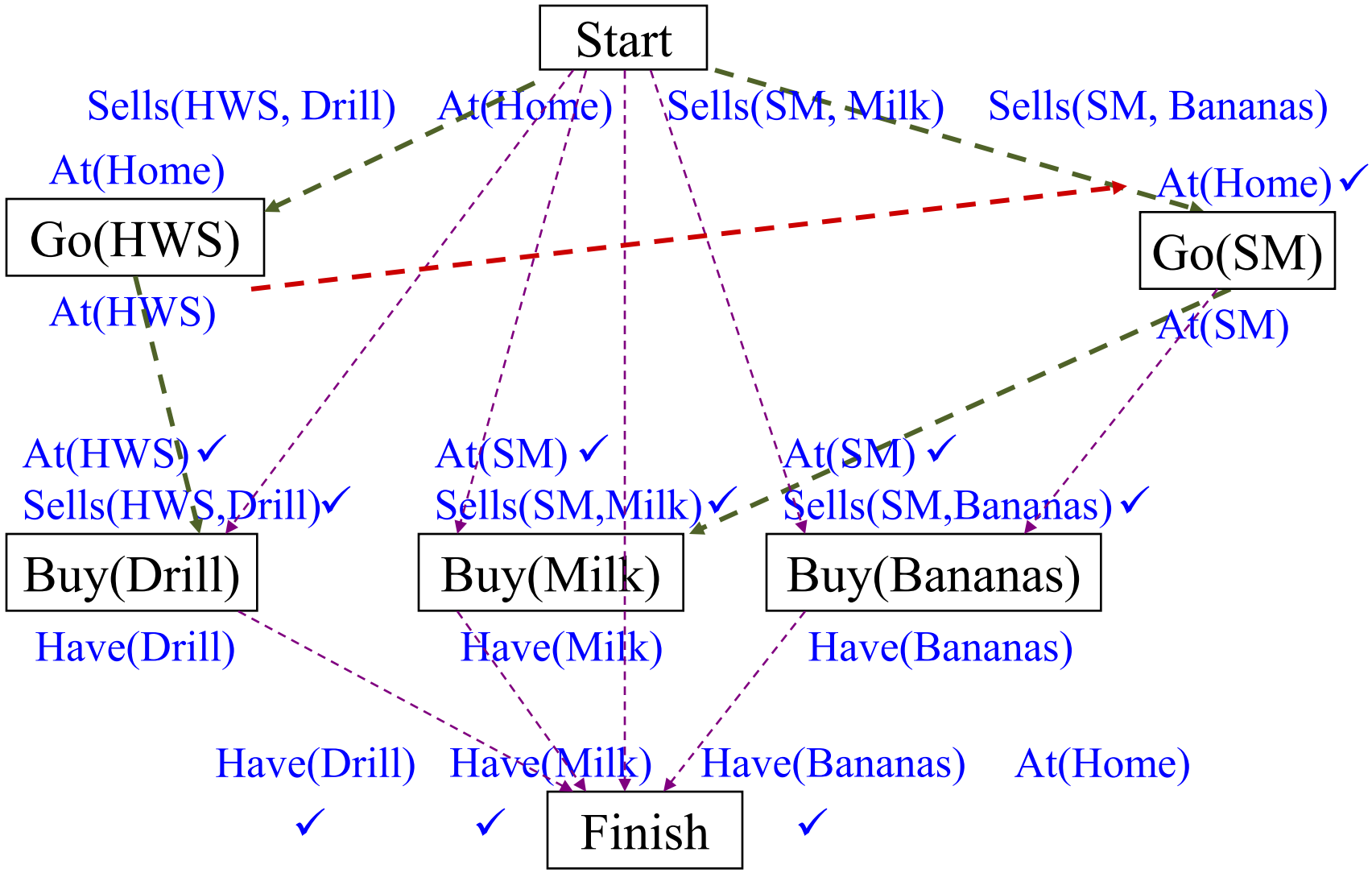
POP: Exemplo das Compras



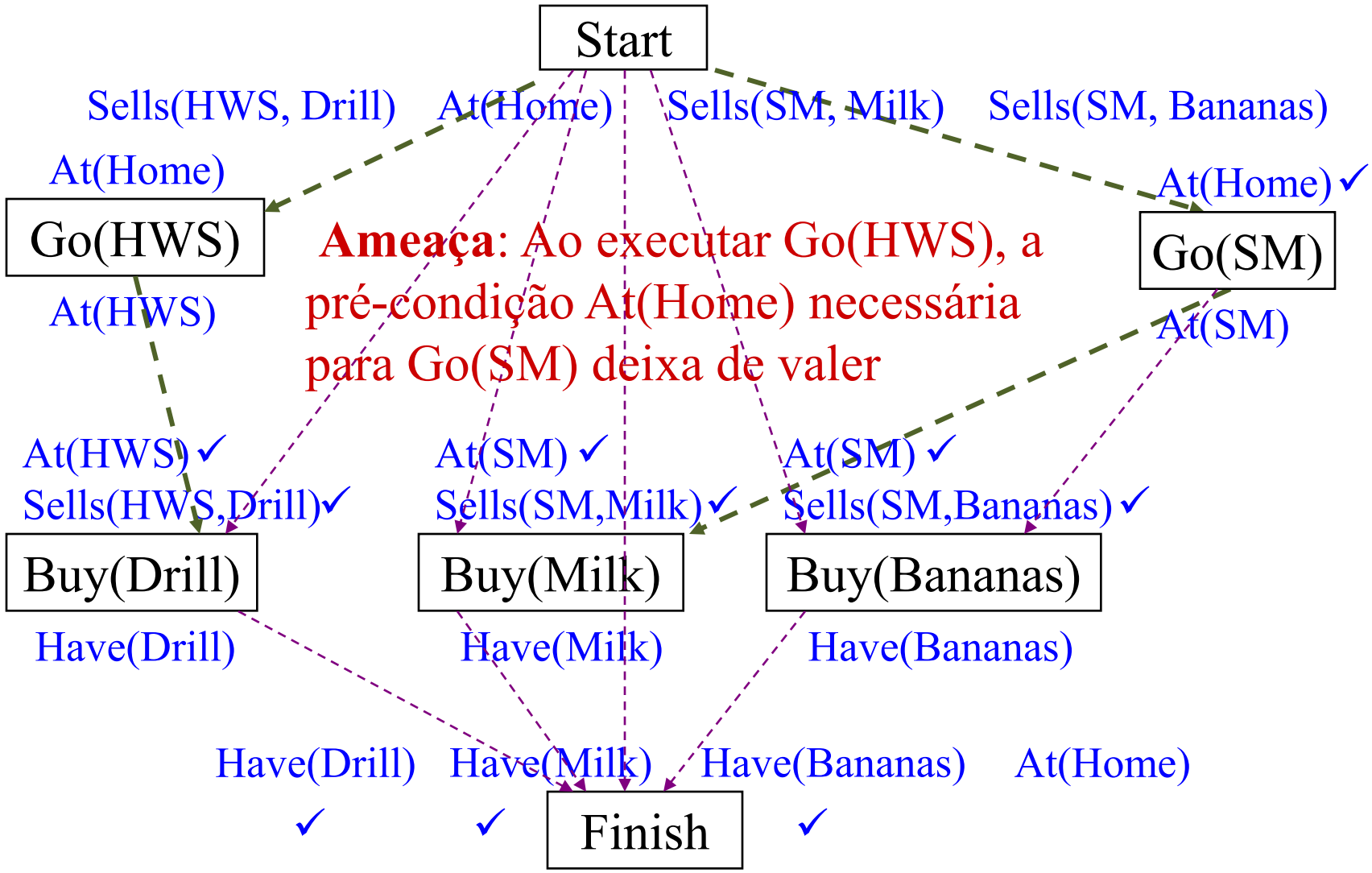
POP: Exemplo das Compras



POP: Exemplo das Compras

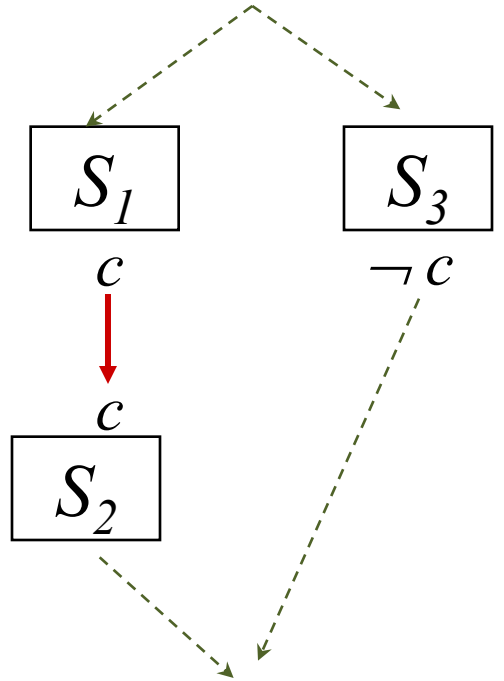


POP: Exemplo das Compras



POP: Problema da Ameaça

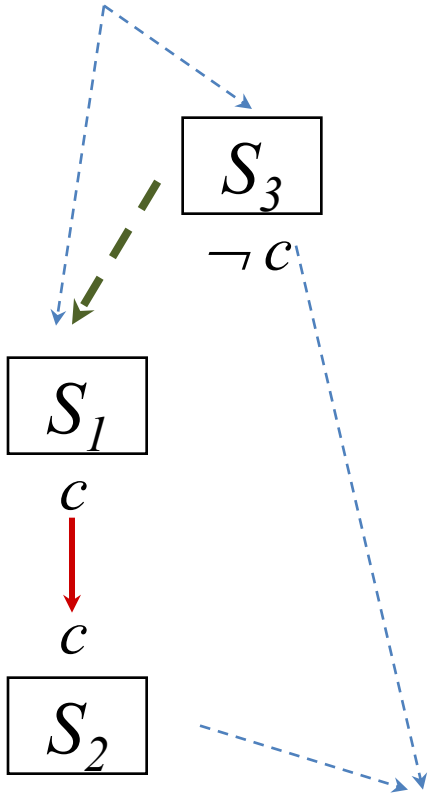
- Ameaça
  - ocorre quando os efeitos de um passo põem em risco as pré-condições de outro
- Como testar?
  - O efeito do novo passo é inconsistente com condição protegida
  - O passo antigo é inconsistente com nova condição protegida

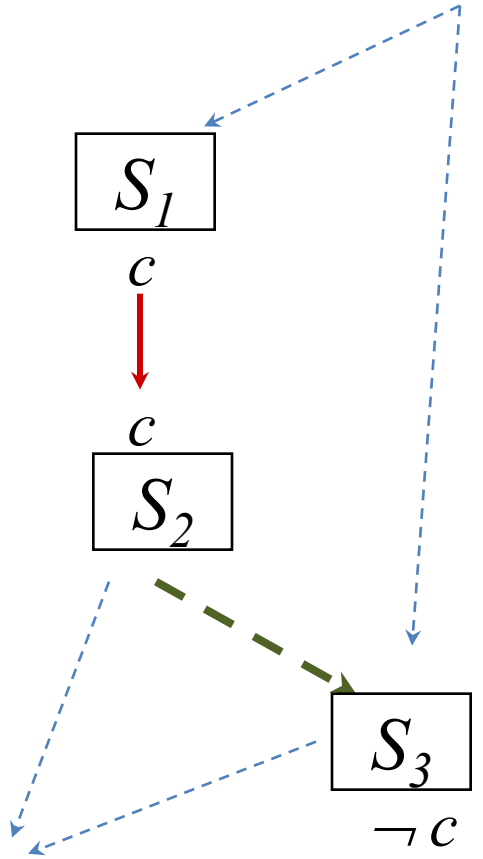


S3 ameaça a condição c estabelecida por S1 e protegida pelo link causal S1 para S2

POP: Solução do Problema da Ameaça

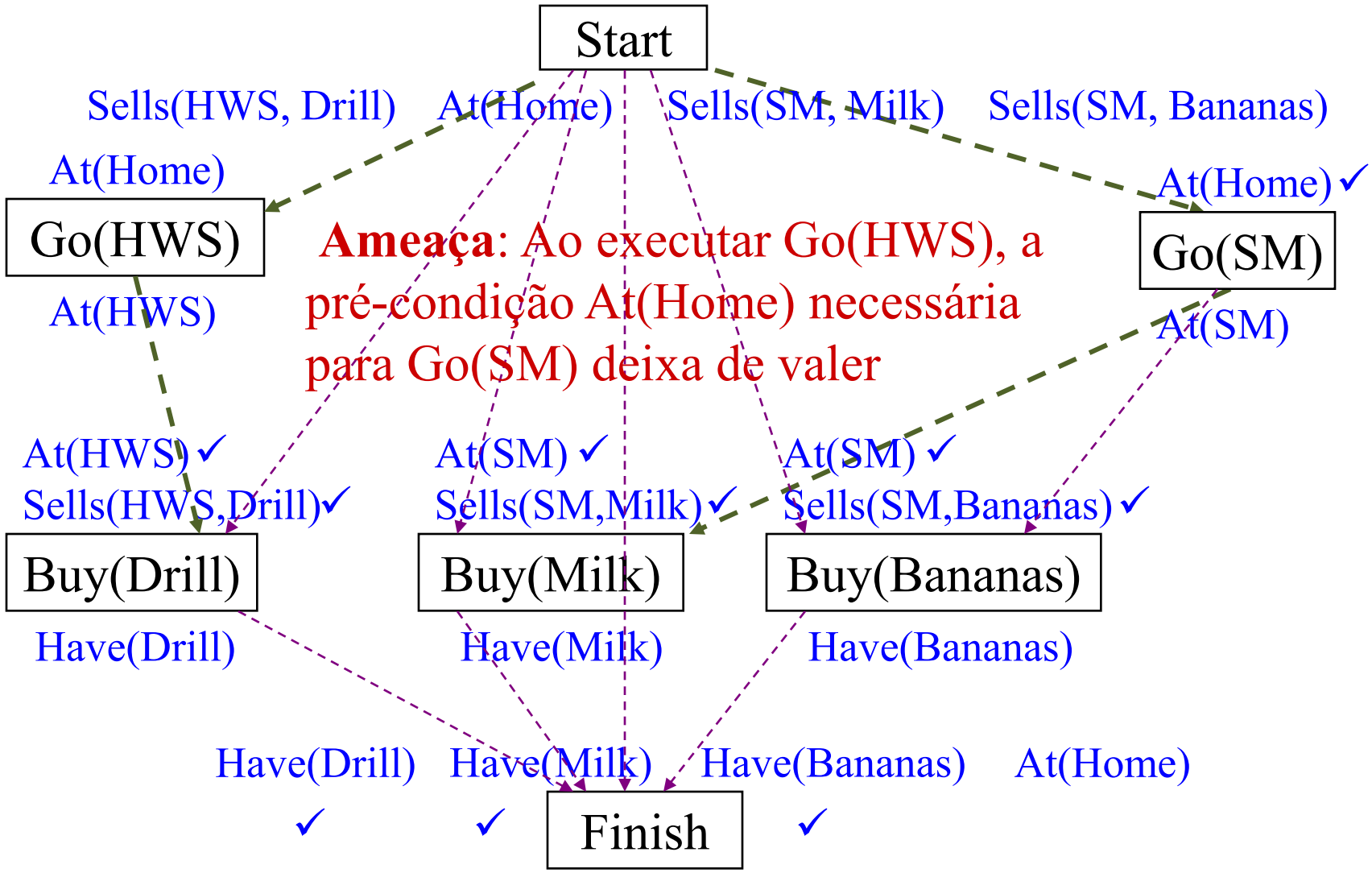
Demoção: adiciona uma restrição de ordem, forçando S3 a ocorrer antes de S1



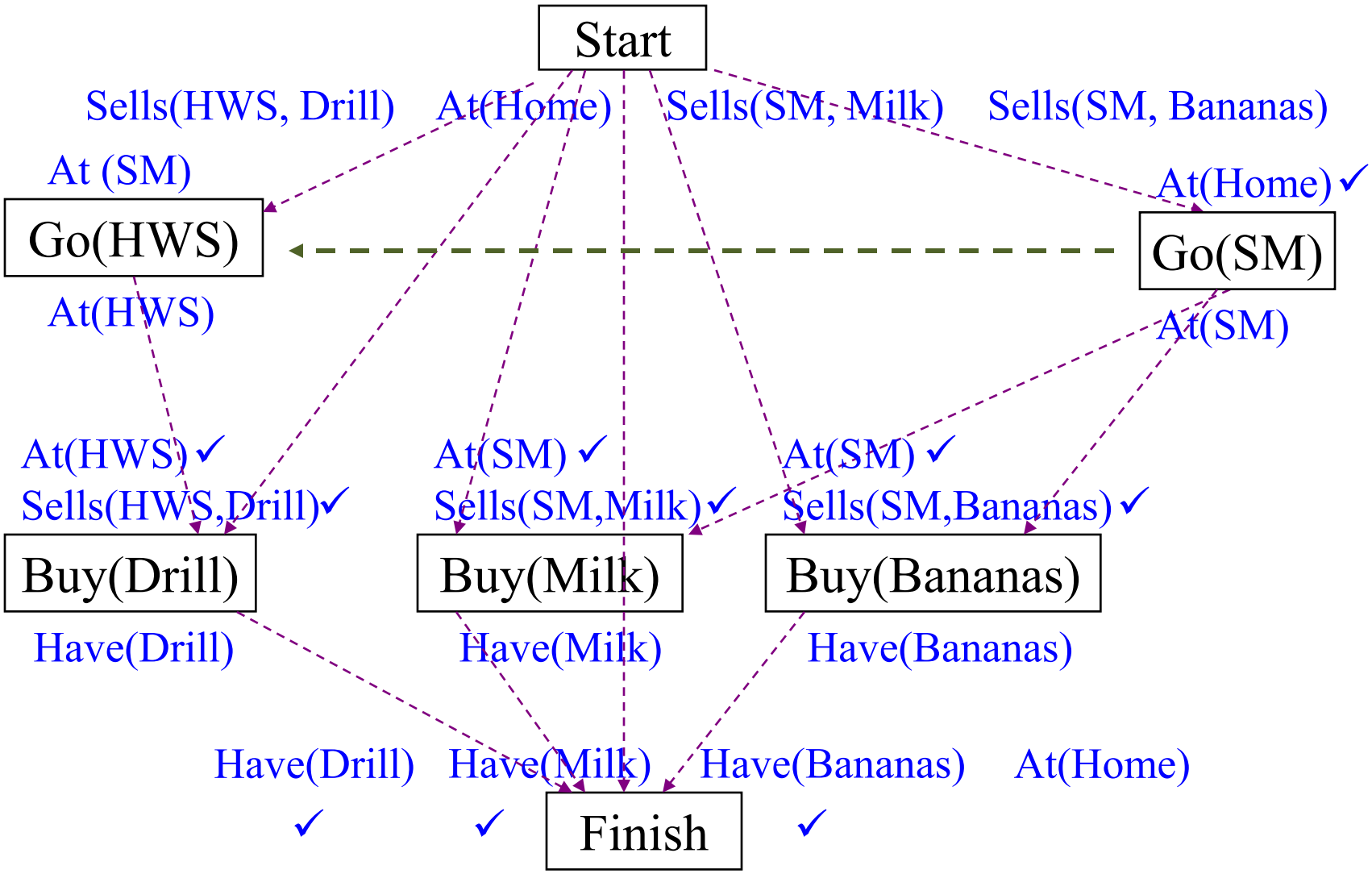


Promoção: adiciona uma restrição de ordem, forçando S3 a ocorrer depois de S2

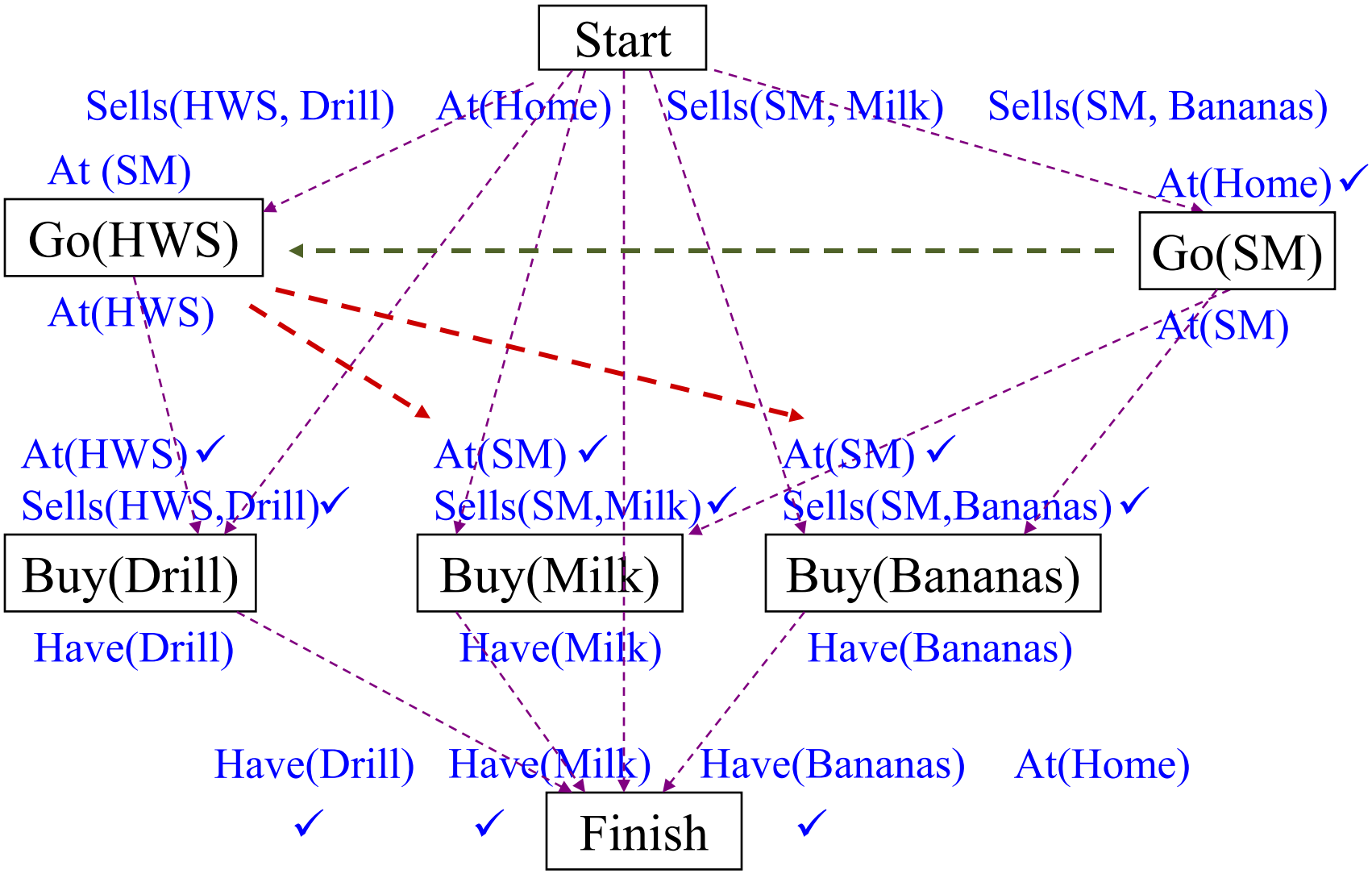
POP: Exemplo das Compras



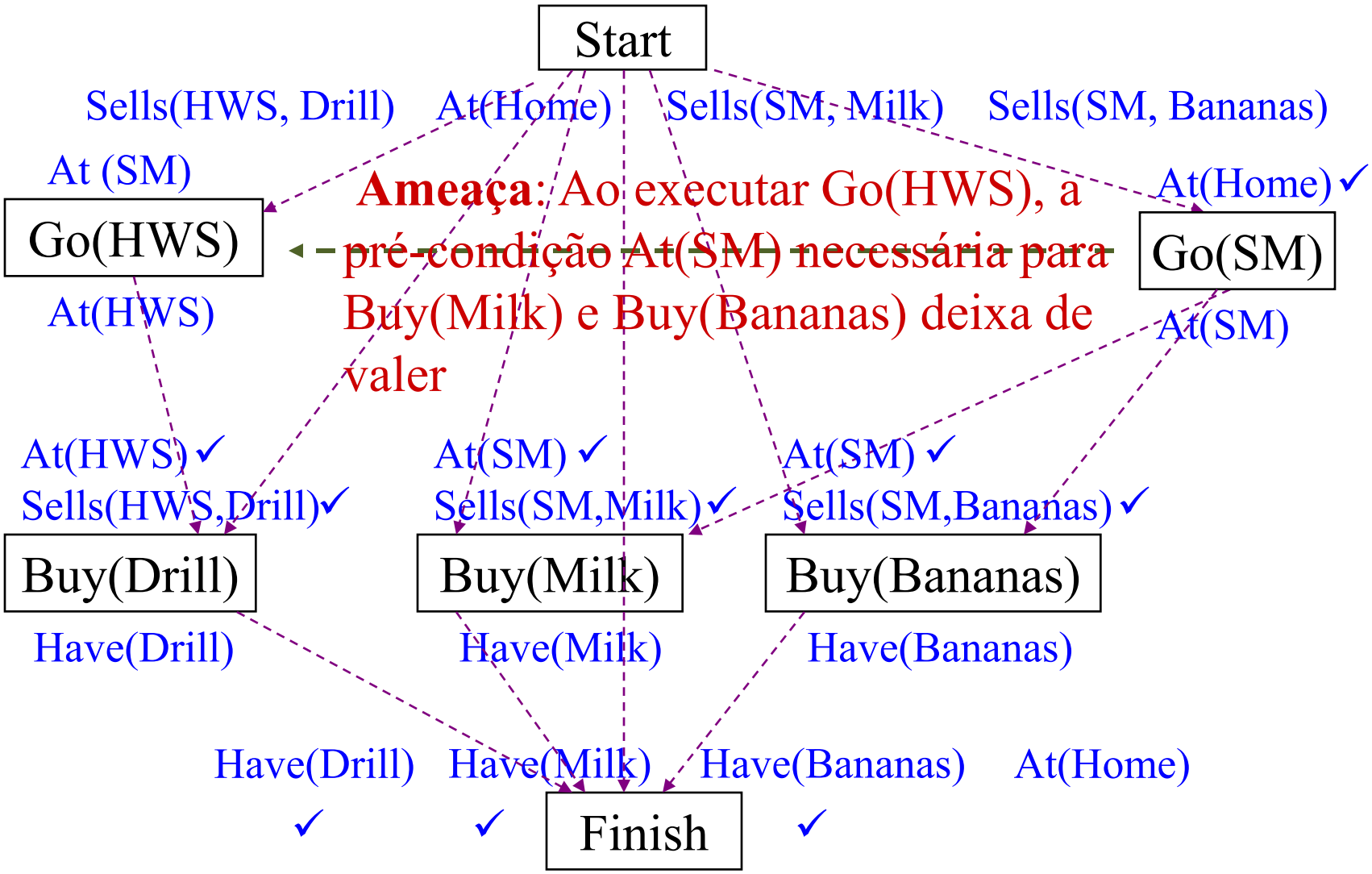
POP: Exemplo das Compras



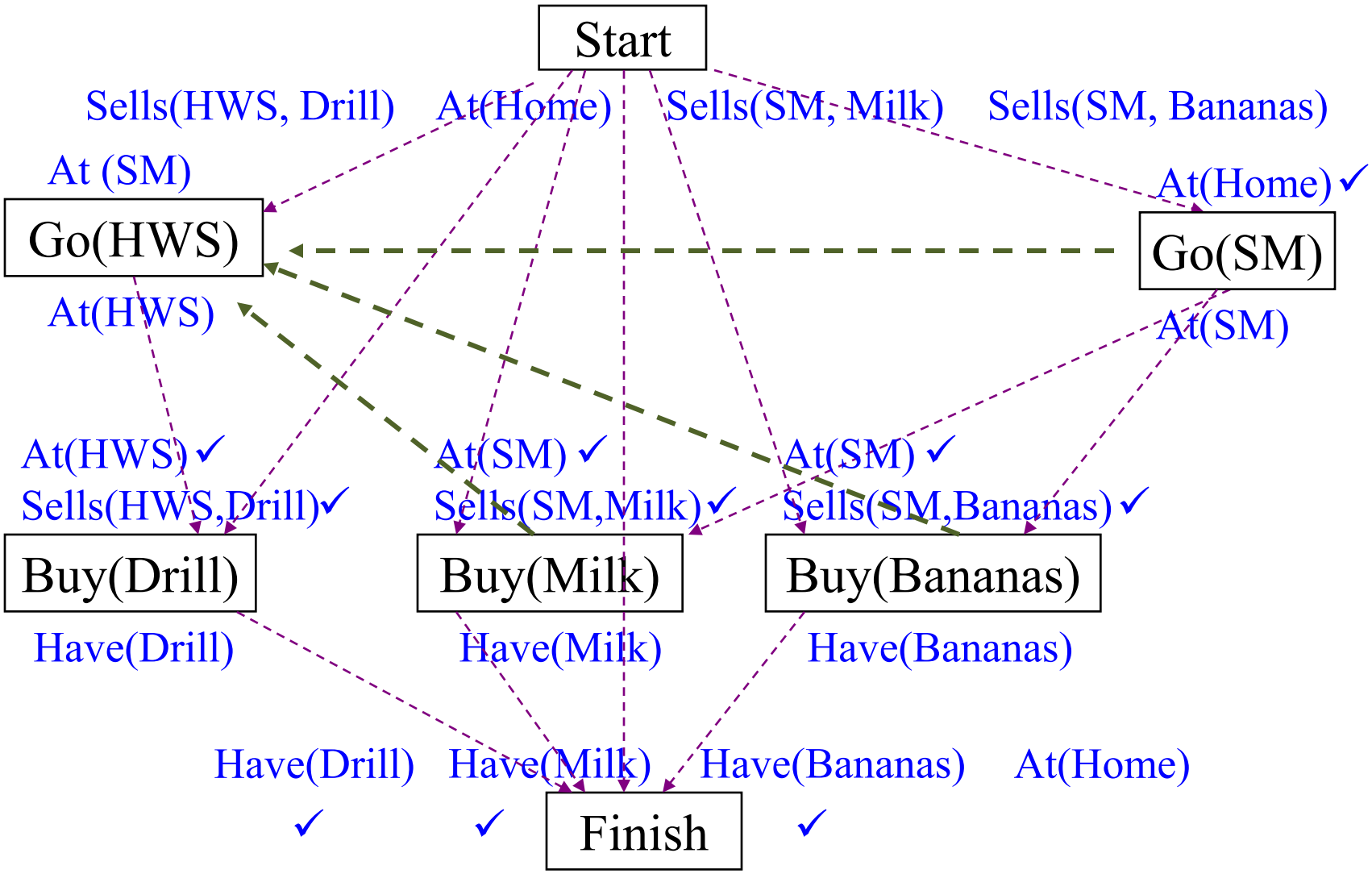
POP: Exemplo das Compras



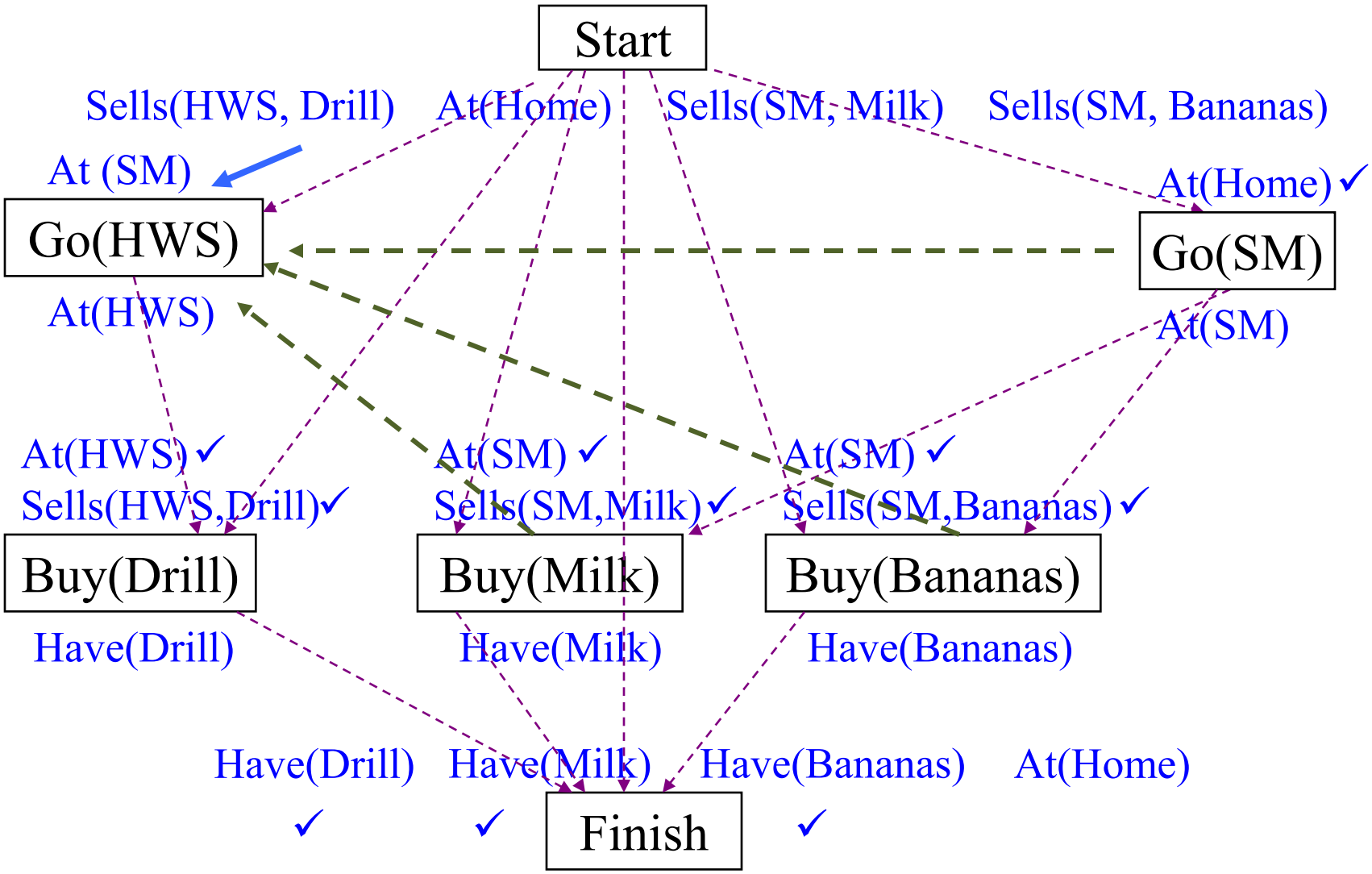
POP: Exemplo das Compras



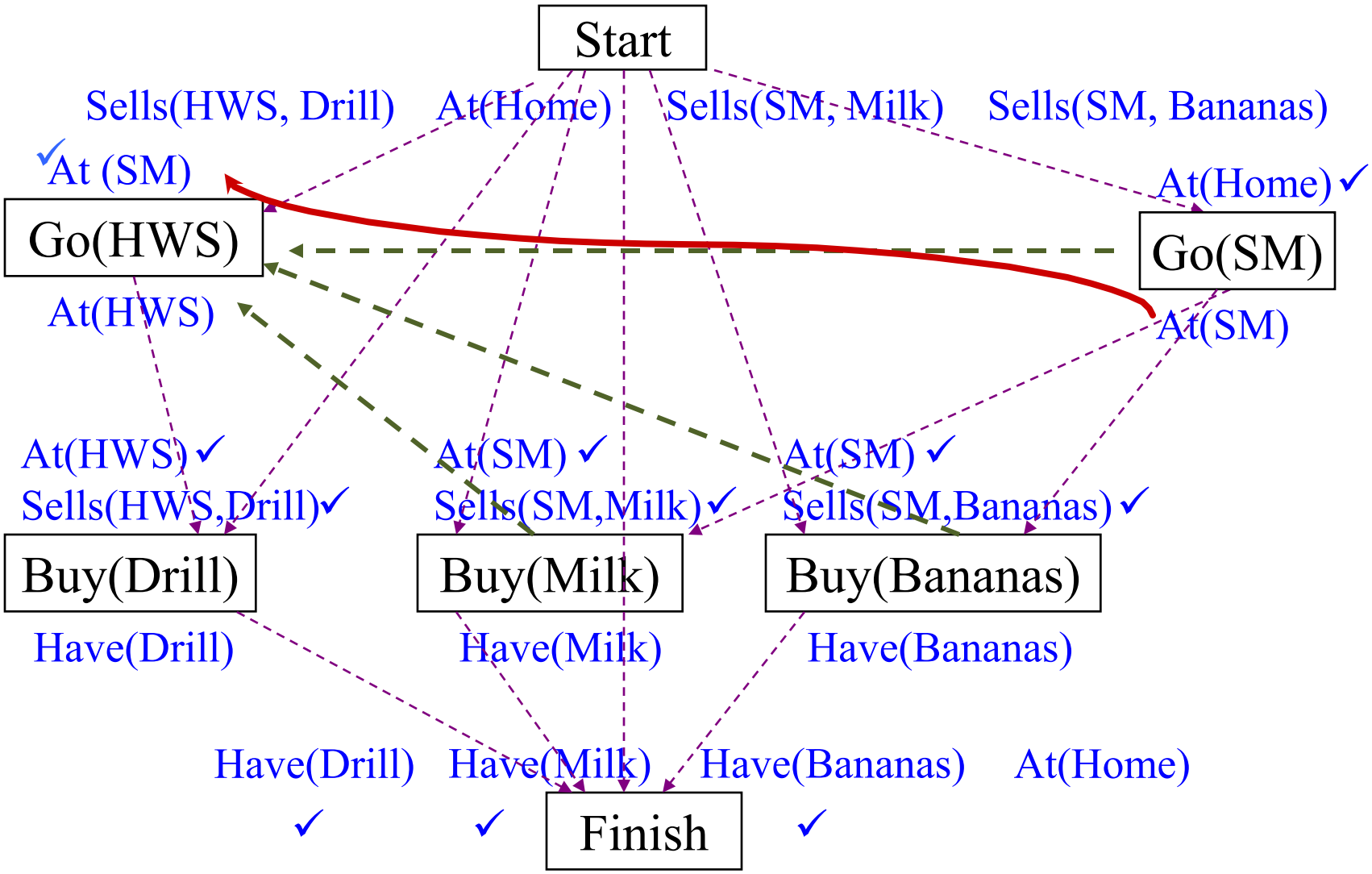
POP: Exemplo das Compras



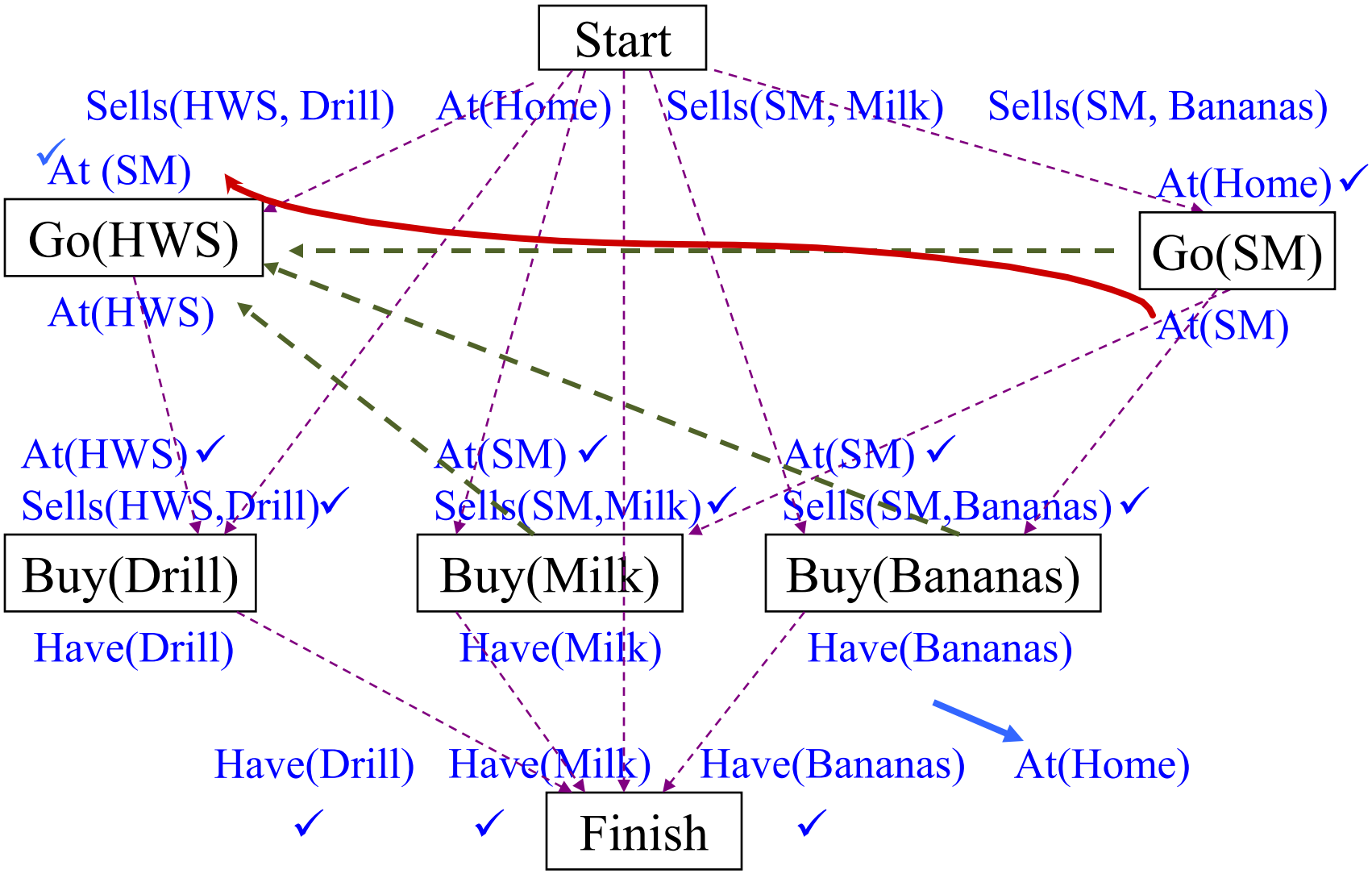
POP: Exemplo das Compras



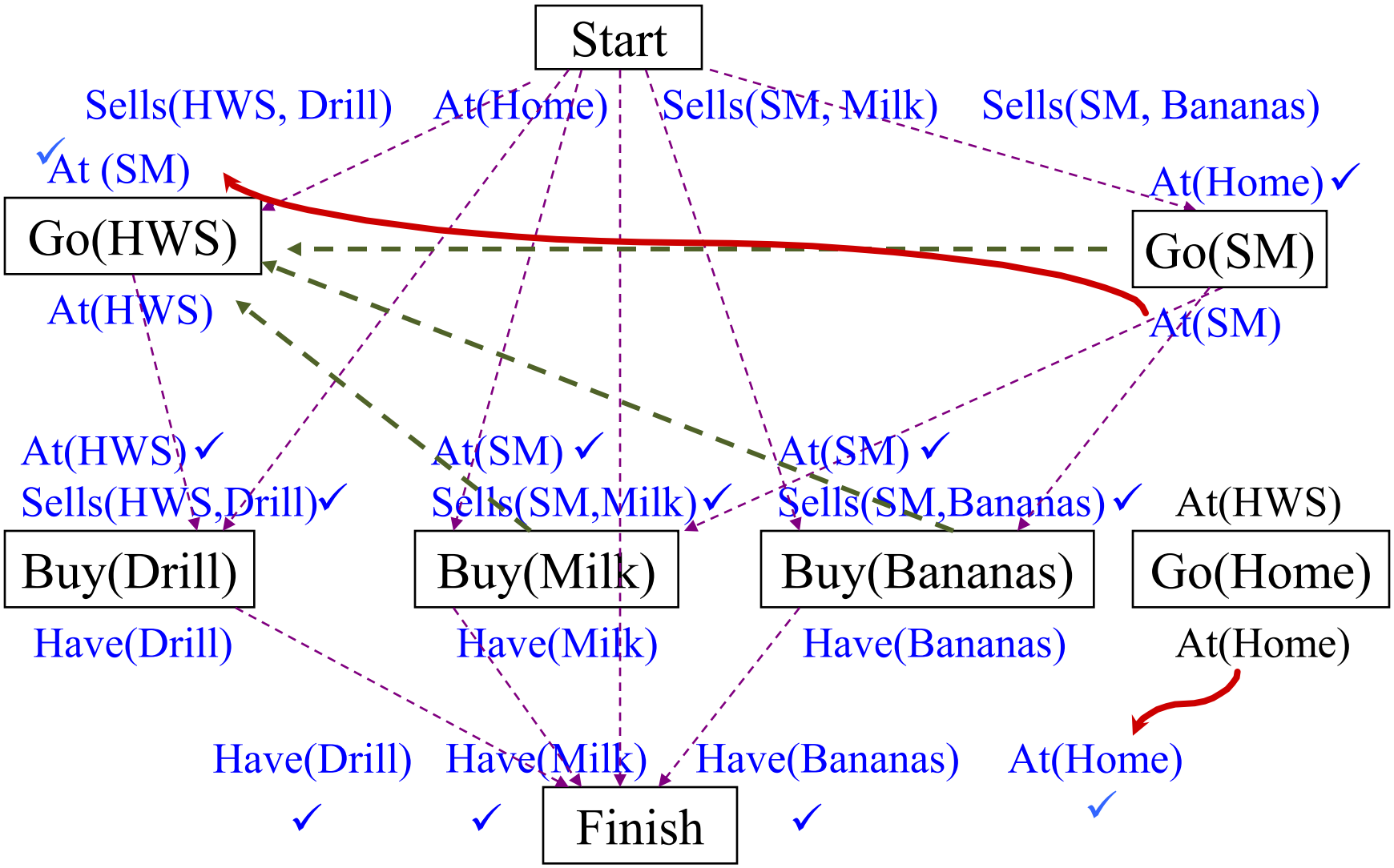
POP: Exemplo das Compras



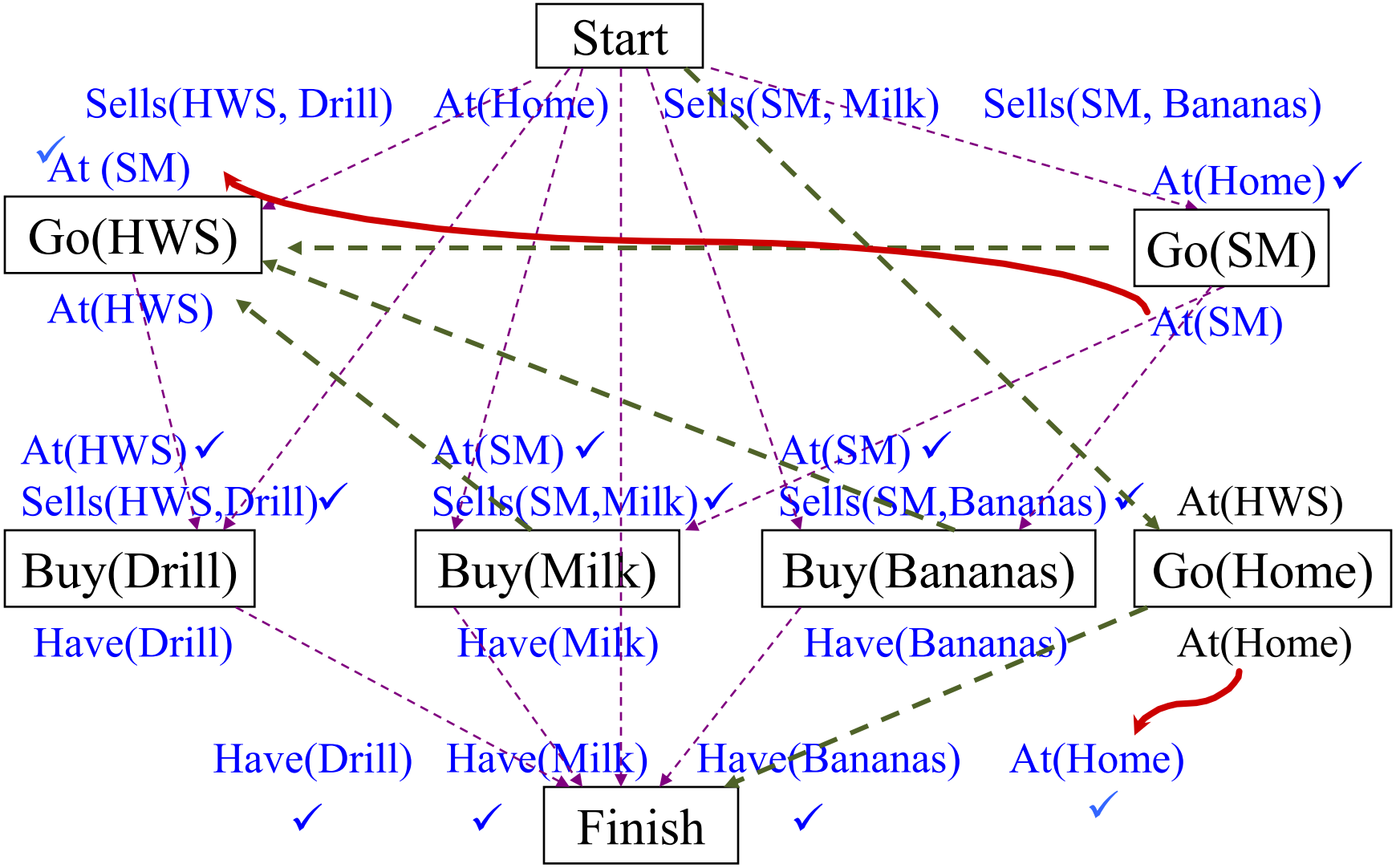
POP: Exemplo das Compras



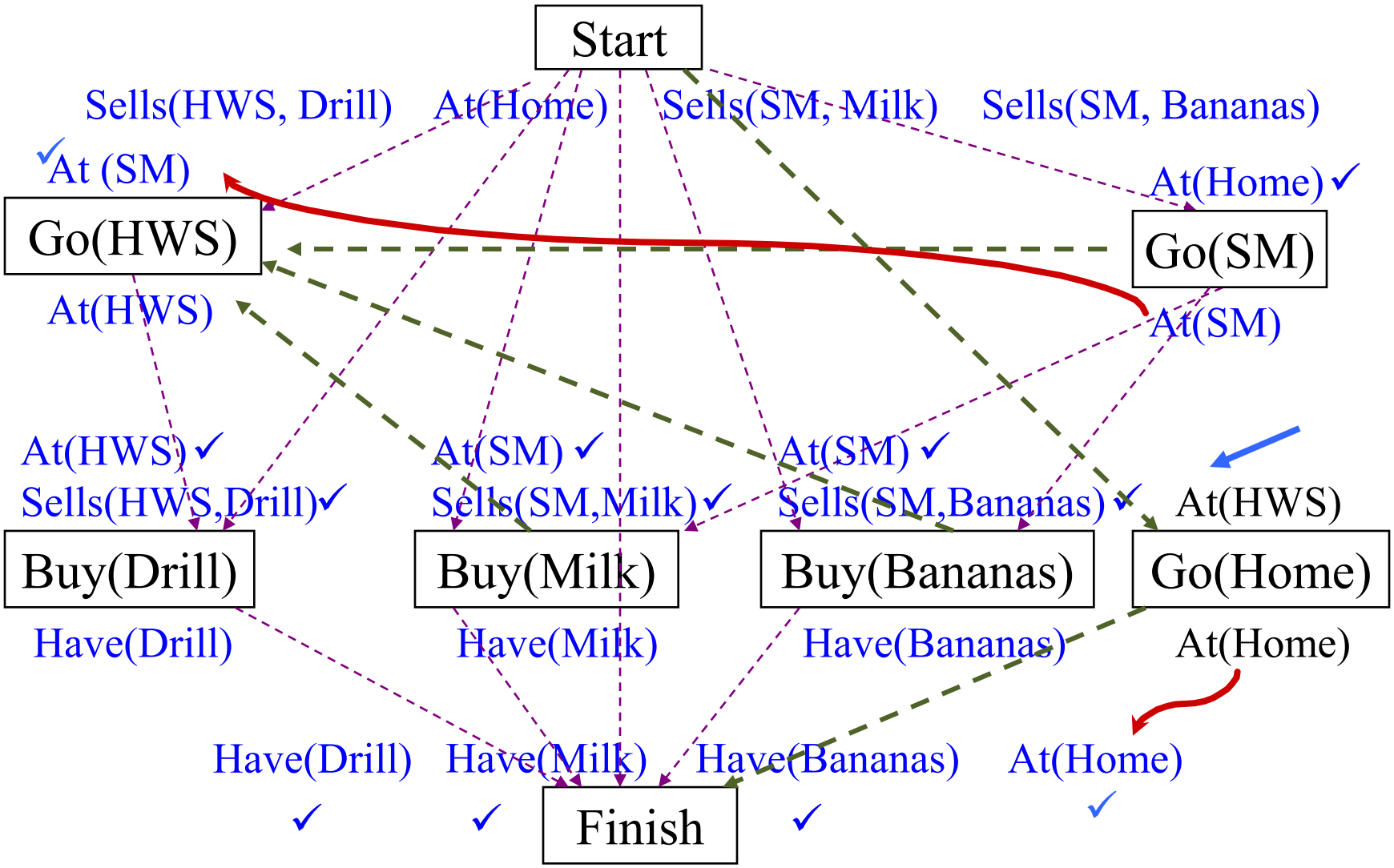
POP: Exemplo das Compras



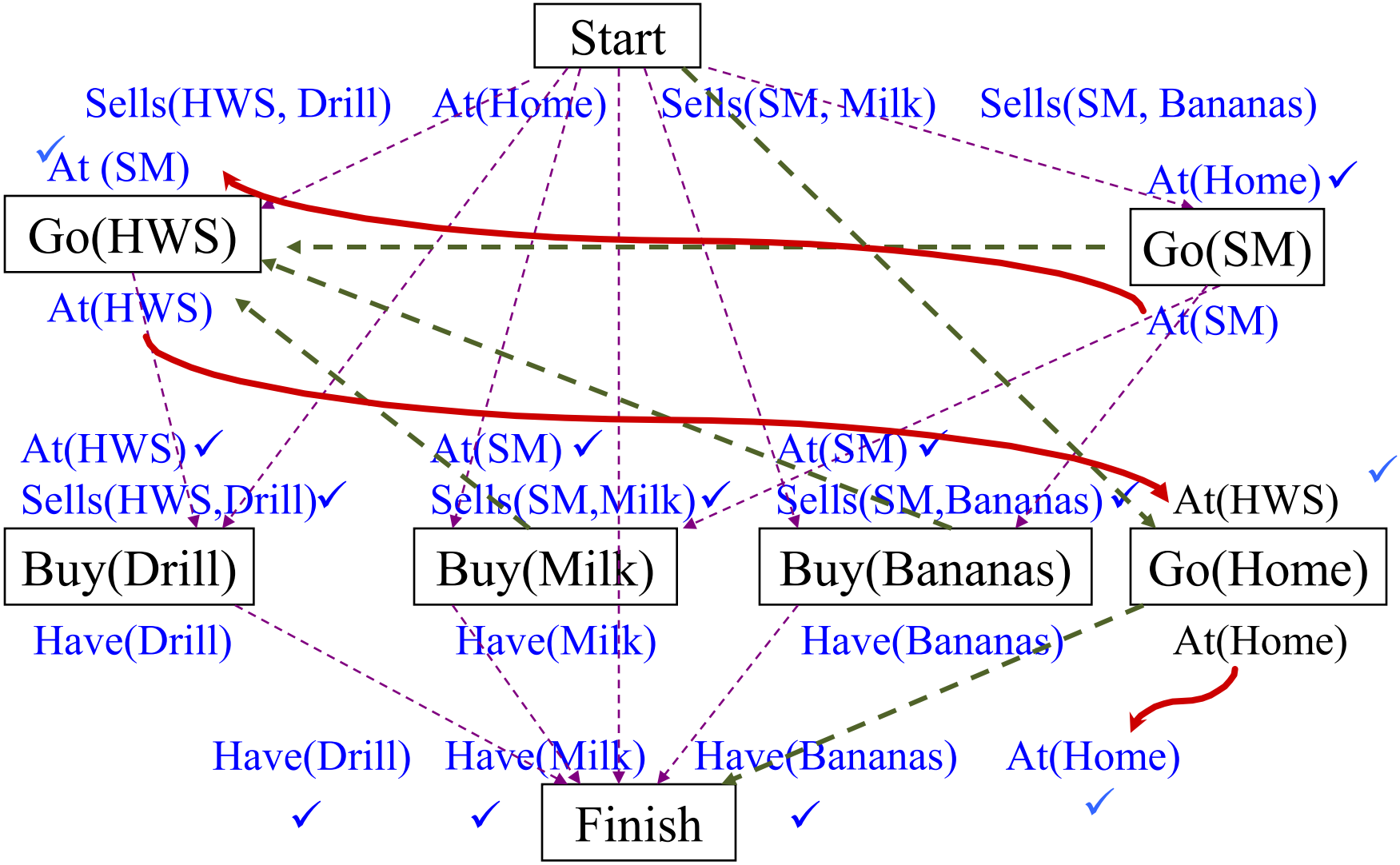
POP: Exemplo das Compras



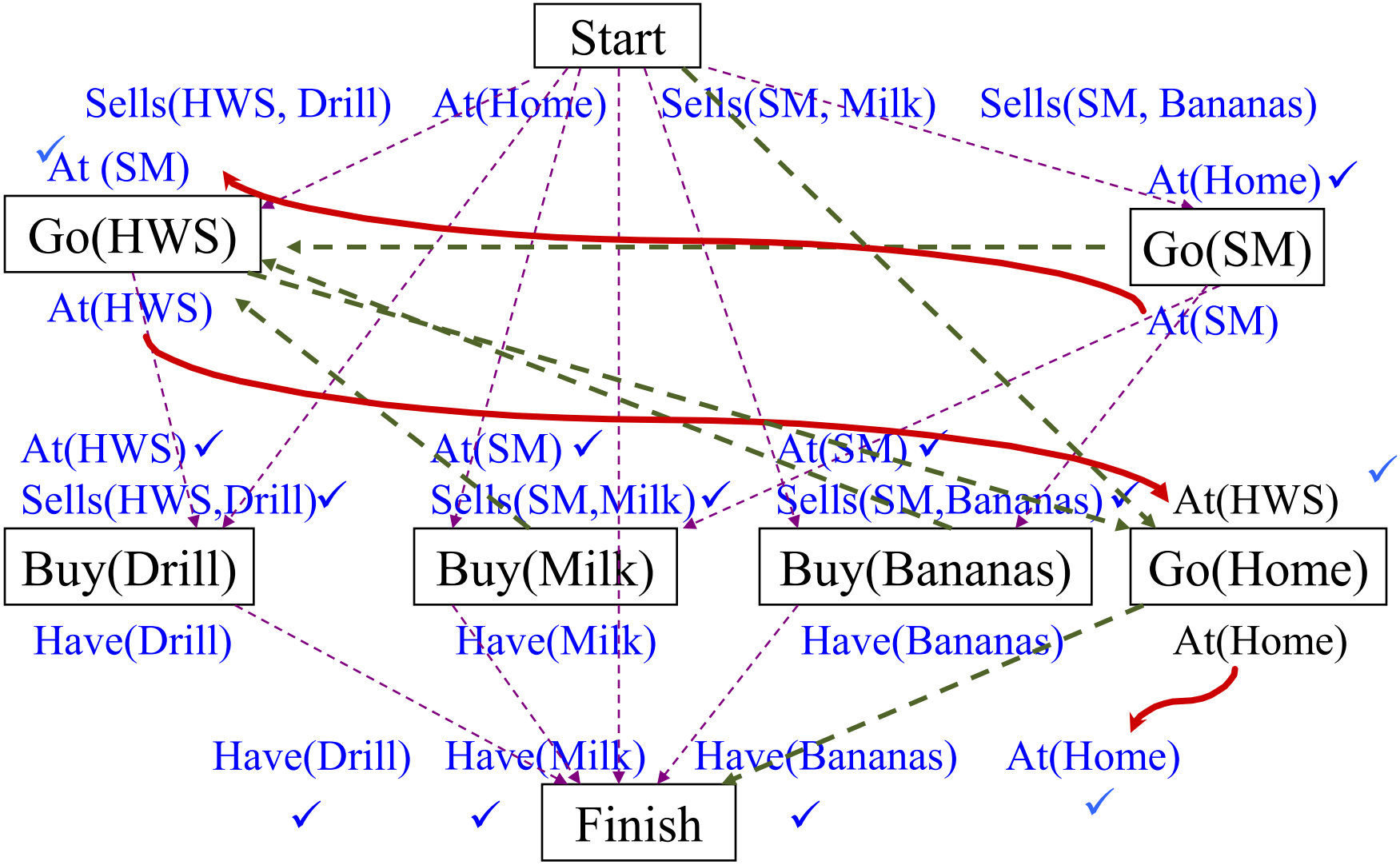
POP: Exemplo das Compras



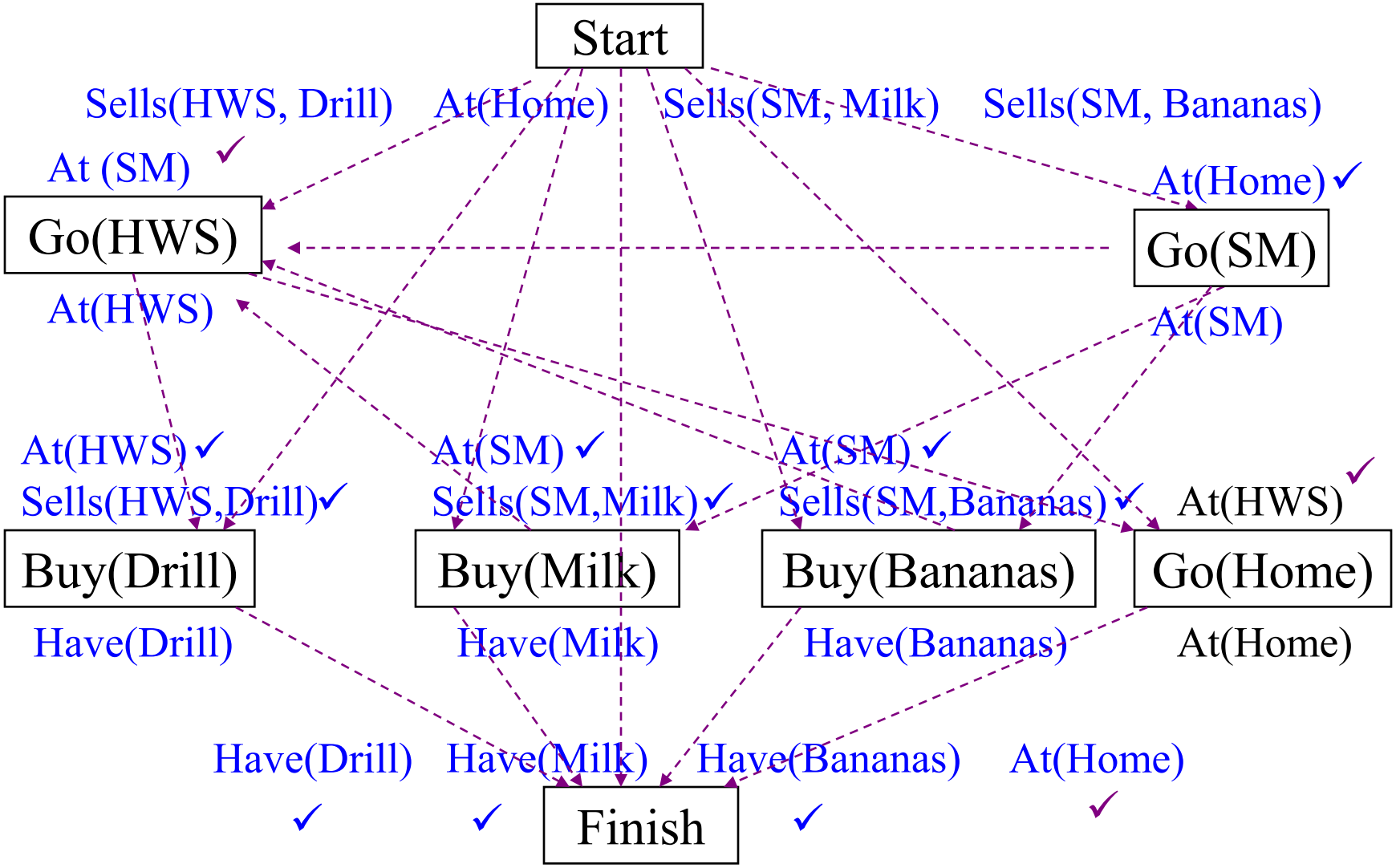
POP: Exemplo das Compras



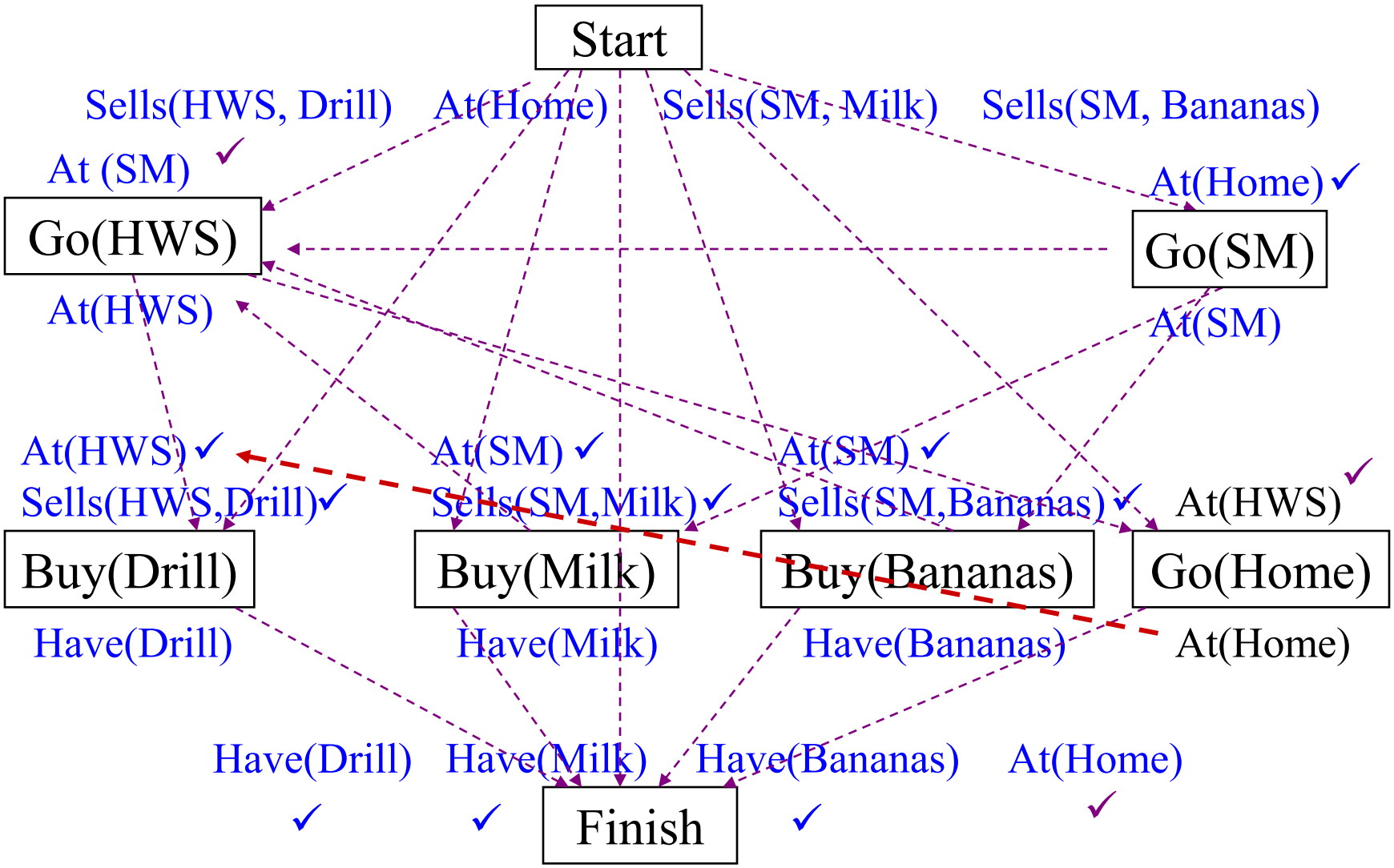
POP: Exemplo das Compras



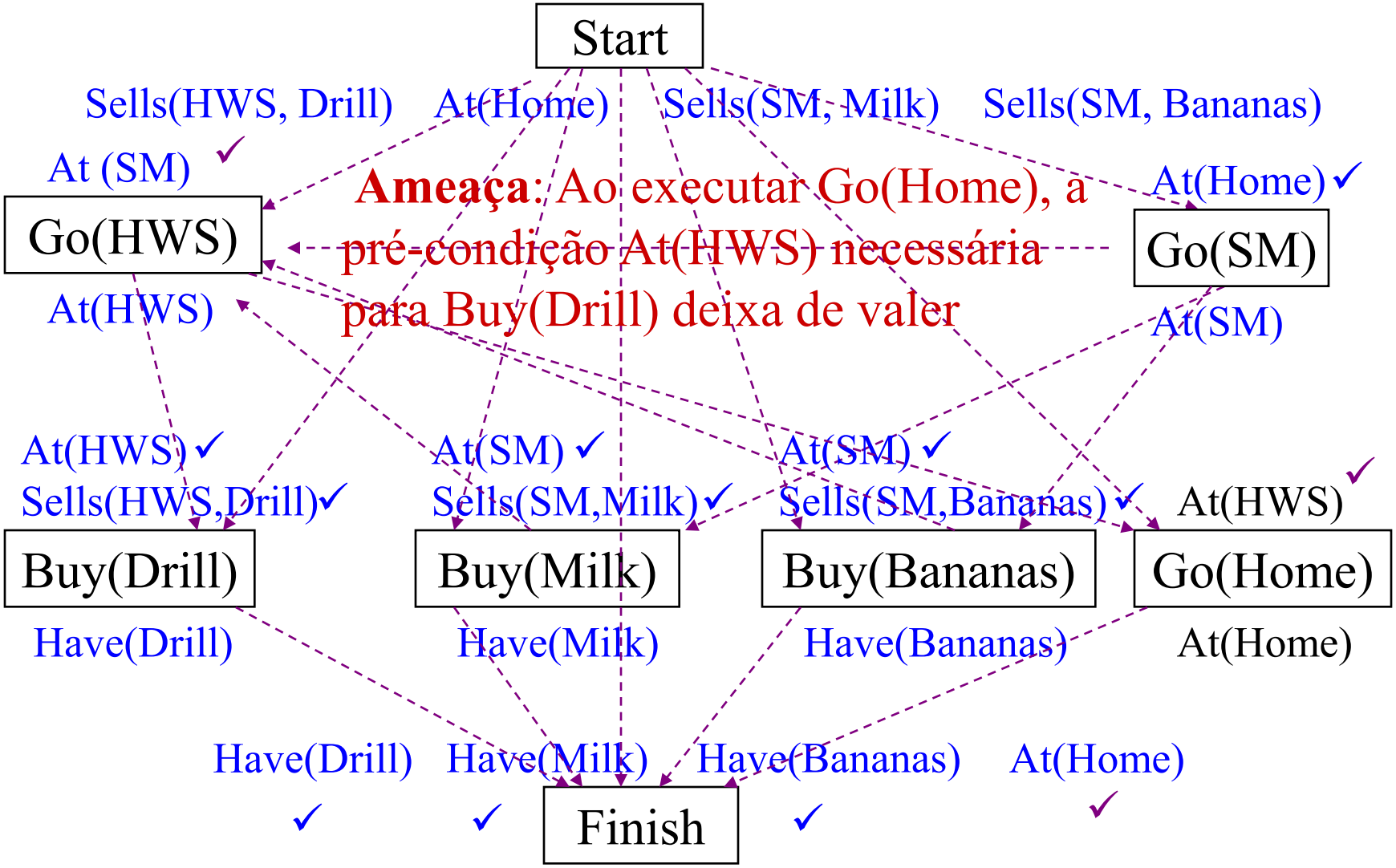
POP: Exemplo das Compras



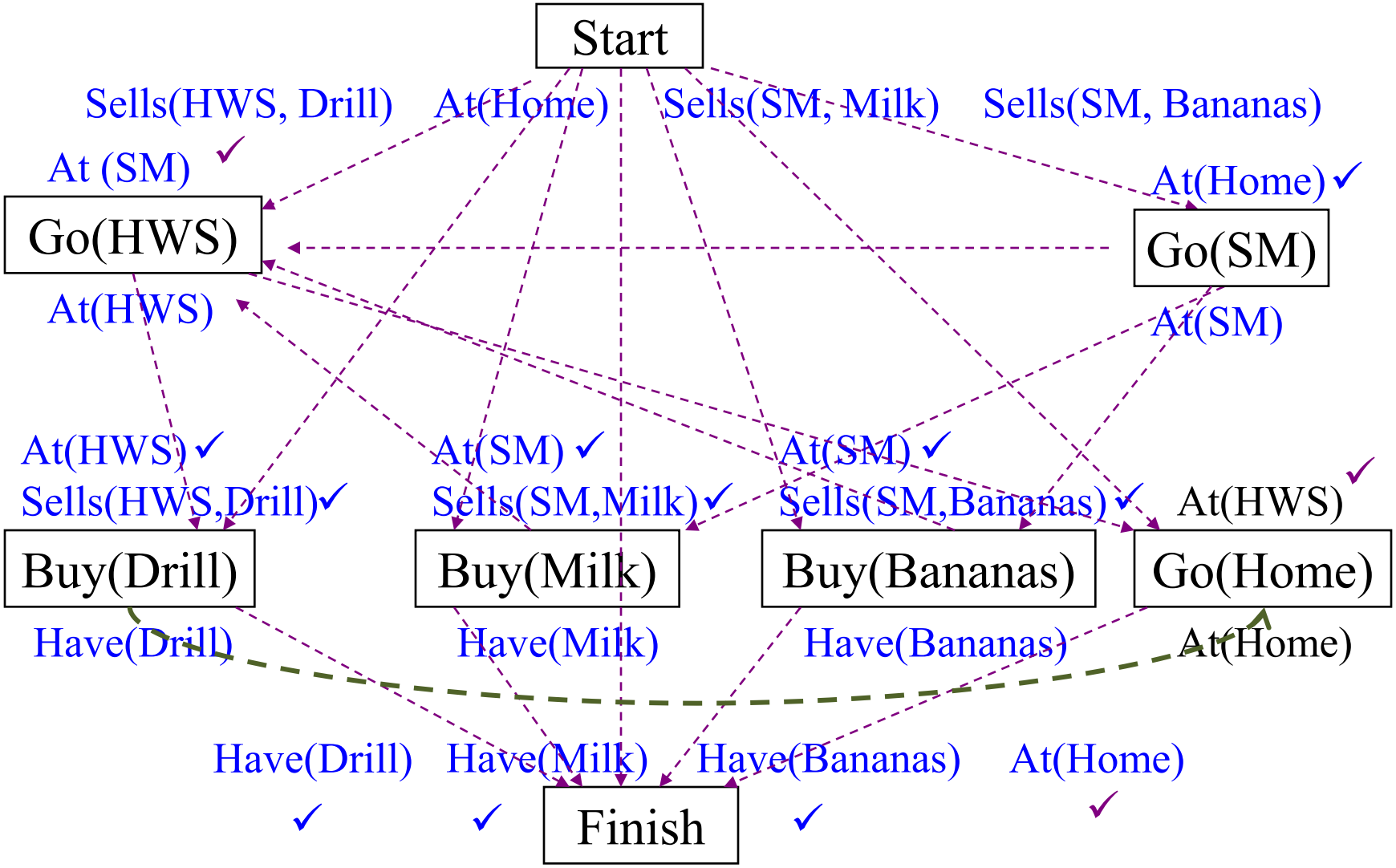
POP: Exemplo das Compras



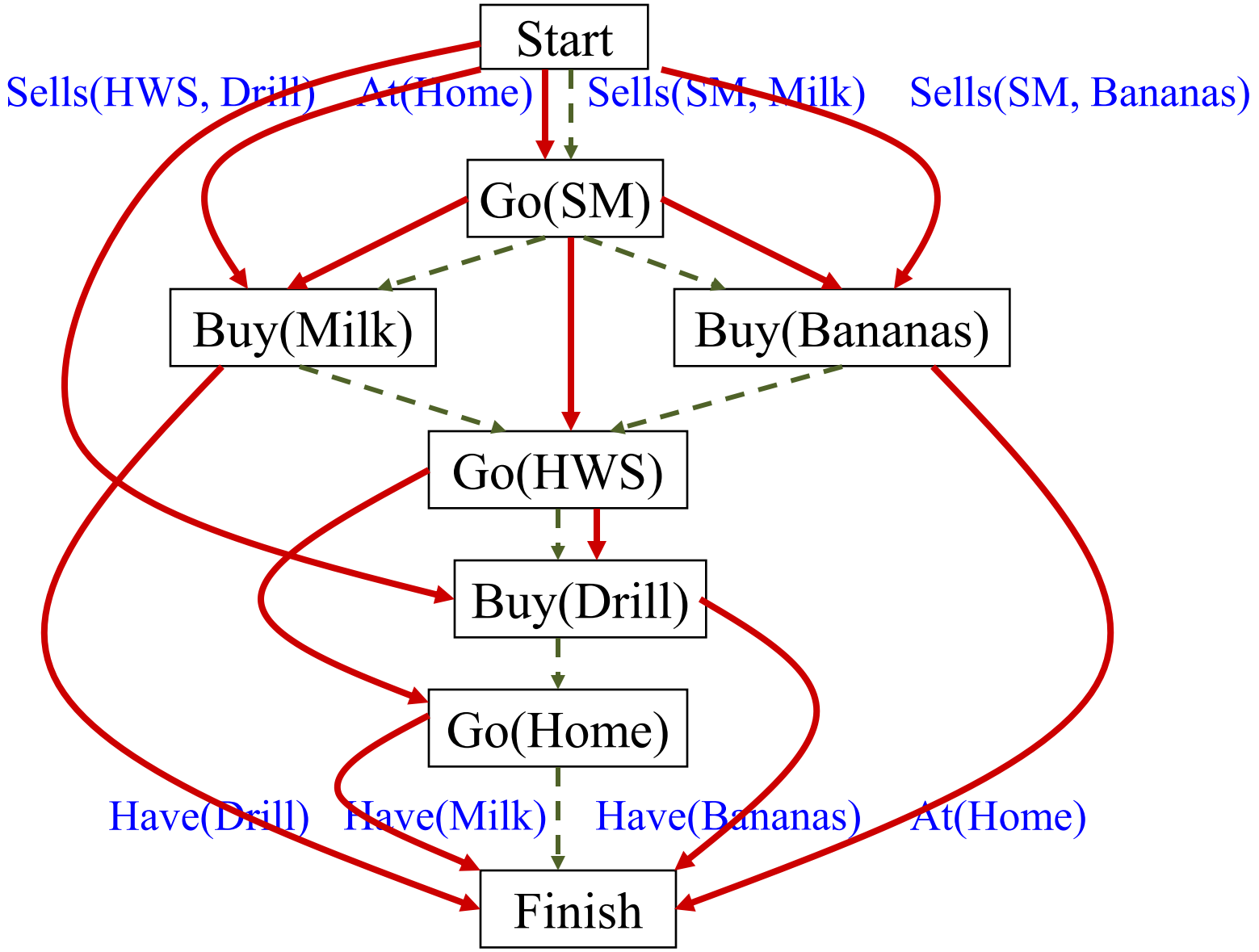
POP: Exemplo das Compras



POP: Exemplo das Compras



POP: Solução do Exemplo das Compras

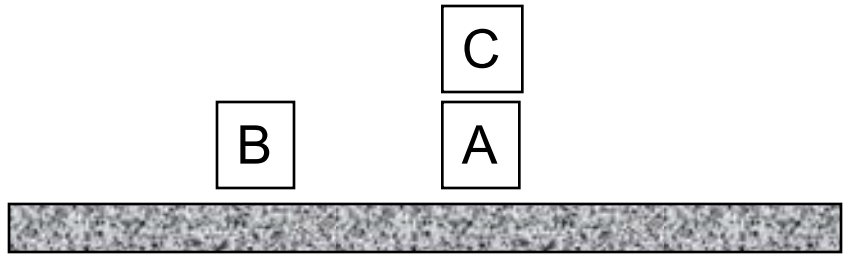


Planejamento: Engenharia de Conhecimento

- Decidir sobre o que falar
- Decidir sobre um vocabulário de condições, operadores e objetos
- Codificar os operadores para o domínio
- Codificar uma descrição da instância do problema
- Colocar o problema para o planejador existente e obter os planos

Mundo dos Blocos: Estados e Ações

- O que falar
  - um conjunto de blocos sobre uma mesa a serem empilhados numa certa ordem
  - só se pode mover um bloco se não houver nada em cima dele



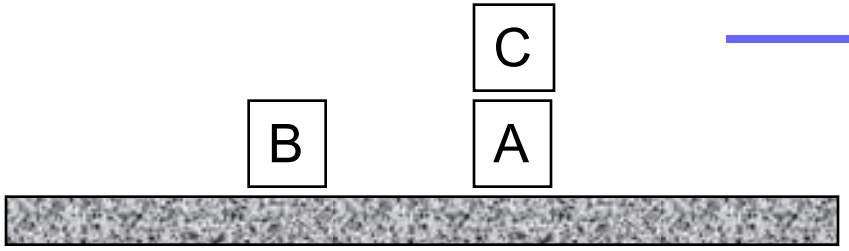
- Vocabulário
- Estado:
  - On(x, y): bloco x está em cima de y
  - Clear(x): bloco x está livre
- Ações:
  - PutOn(x, y): mover x para cima de y
  - PutOnTable(x): mover x para a mesa

Mundo dos Blocos: Estados

- Com isto é possível resolver problemas do mundo dos blocos...

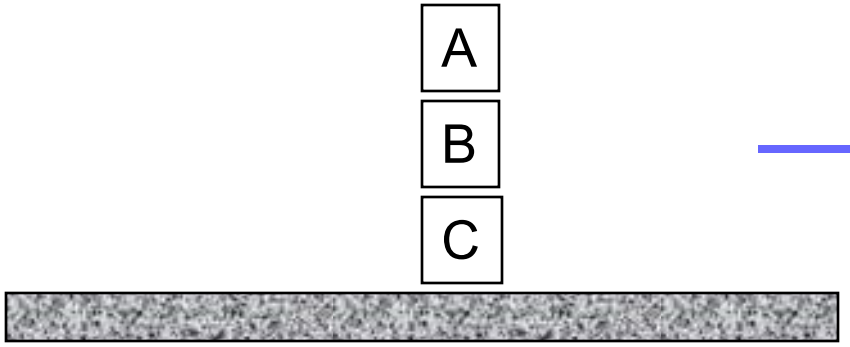
Mundo dos Blocos: Estados

Estado inicial



On(C, A)
On(B, Table)
On(A, Table)
Clear(B)
Clear(C)

Estado final



On(A, B)
On(B, C)
On(C, Table)
Clear(A)

Mundo dos Blocos: Ações

Op(ACTION: **PutOn**(x,y),

PRECOND: On(x,z), Clear(x), Clear(y)

EFFECT: ADD: On(x,y), Clear(z)

DEL: ¬On(x,z), ¬Clear(y))

On(x,z), Clear(x), Clear(y)

PutOn(x,y)

On(x,y), ¬Clear(y), Clear(z), ¬ On(x,z)

Op(ACTION: **PutOnTable**(x),

PRECOND: On(x,z), Clear(x)

EFFECT: ADD: On(x,Table), Clear(z)

DEL: ¬On(x,z))

On(x,z), Clear(x)

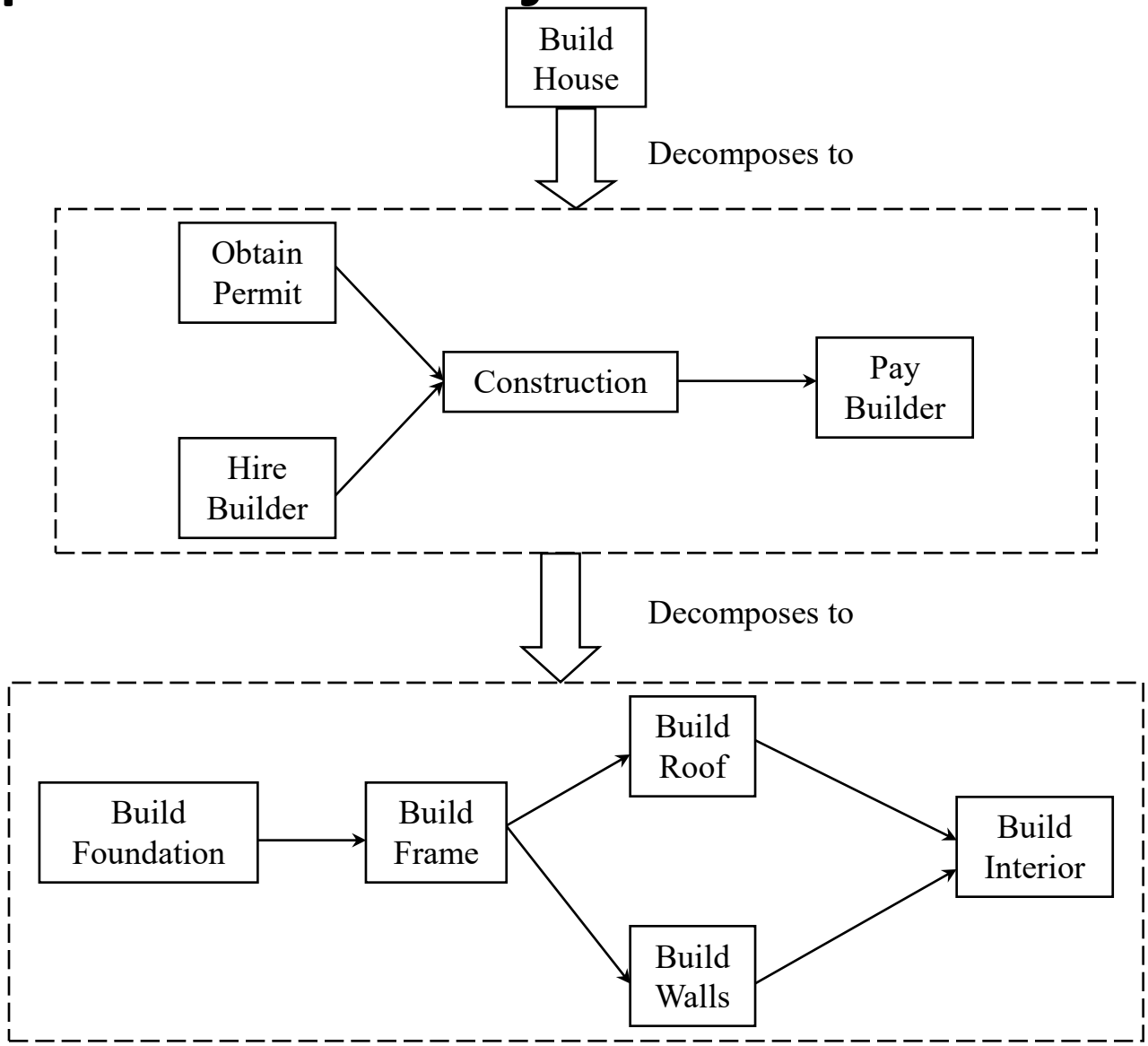
PutOnTable(x)

On(x,Table), Clear(z), ¬ On(x,z)

Limitações de POP-STRIPS

- Planejamento a um único nível de granularidade
  - planejadores hierárquicos
- Precondições e efeitos não contextuais
  - Quantificadores, condicionais
- Representação do tempo
  - ações têm durações
- Representação de recursos limitados
  - ações consomem recursos

Exemplo de Planejamento Hierárquico



Aplicações de Planejamento

- Construção de prédios:
  - SIPE
- Escalonamento de tarefas industriais
  - TOSCA (Hitachi)
  - ISIS (Whestinghouse)
- Construção, integração e verificação de espaçonaves:
  - Optimum-AIV (Agência Espacial Européia)
- Planejamento para Missões Espaciais
  - Voyager, Telescópio espacial Hubble (NASA)
  - ERS-1 (Agência Espacial Europea)
- Robótica, logística, manufatura, etc...

Referências Bibliográficas

- S. Russel and P. Norvig. Artificial Intelligence: A Modern Approach. Prentice Hall, Upper Saddle River, USA. 2nd. Edition, 2003. Chapter 11 and 12.