

Geração de Planos (Planning)

April 3, 2017

Algoritmo Gerador de Planos: POP – Partial-Order Planner

- Algoritmo *não-determinístico*.
- Utilização de **choose** e **fail**.
- Começa com um plano parcial mínimo.
- A cada passo expande o plano ao tentar alcançar uma pré-condição c de um passo S_{need} .
- Estende o plano: através dos passos do plano parcial ou através de um conjunto de operadores que alcançam as pré-condições.
- Guarda o link causal para a pré-condição alcançada, para poder resolver as 'ameaças'.
- Importante: SELECT_SUBGOAL não tem várias alternativas.
- Razões: todas as precond precisam ser consideradas, comutativo.

Algoritmo Gerador de Planos: POP – Partial-Order Planner

```

function POP(initial, goal, operators) returns plan

  plan ← MAKE-MINIMAL-PLAN(initial, goal)
  loop do
    if SOLUTION?(plan) then return plan
    Snext, c ← SELECT-SUBGOAL(plan)
    CHOOSE-OPERATOR(plan, operators, Snext, c)
    RESOLVE-THREATS(plan)
  end

```

```

function SELECT-SUBGOAL(plan) returns Snext, c

  pick a plan step Snext from STEPS(plan)
  with a precondition c that has not been achieved
  return Snext, c

```

```

procedure CHOOSE-OPERATOR(plan, operators, Snext, c)

  choose a step Sadd from operators or STEPS(plan) that has c as an effect
  if there is no such step then fail
  add the causal link Sadd → Snext to LINKS(plan)
  add the ordering constraint Sadd < Snext to ORDERINGS(plan)
  if Sadd is a newly added step from operators then
    add Sadd to STEPS(plan)
    add Start < Sadd < Finish to ORDERINGS(plan)

```

```

procedure RESOLVE-THREATS(plan)

  for each Sthreat that threatens a link Si → Sj in LINKS(plan) do
    choose either
      Promotion: Add Sthreat < Si to ORDERINGS(plan)
      Demotion: Add Sj < Sthreat to ORDERINGS(plan)
    if not CONSISTENT(plan) then fail
  end

```

Gerador de Planos com Operadores Parcialmente Instanciados

- POP não trata de valores possíveis para uma variável.
- Ex: efeito $\neg At(x)$ de algum operador deve ser considerado uma ameaça para uma condição $At(Home)$?
- O efeito em questão é uma *possível* ameaça. Soluções possíveis:
 - ▶ 1) Resolver agora com uma restrição de igualdade: se o gerador de planos escolhe um operador cujo efeito é $\neg At(x)$, adicionar $x = HWS$ de forma a não ameaçar a condição $At(Home)$ já alcançada.
 - ▶ 2) Resolver agora com uma restrição de desigualdade: $x \neq Home$, complicado de implementar a unificação.
 - ▶ 3) Resolver mais tarde: deixar que $\neg At(x)$ seja uma possível ameaça. Se, mais tarde, $x = Home$, resolver. Desvantagem: difícil de saber se o plano é solução.

Gerador de Planos com Operadores Parcialmente Instanciados

- Usando operadores não totalmente instanciados, temos que garantir que todas as instanciações vão chegar ao objetivo.
- Definição: um passo S_i **alcança** uma pré-condição c do passo S_j se:
 1. $S_i \prec S_j$ e S_i tem um efeito que necessariamente unifica com c , e
 2. não há nenhum passo S_k tal que $S_i \prec S_k \prec S_j$ em alguma linearização do plano, e S_k tem um efeito que possivelmente unifique com $\neg c$.
- Algoritmo POP pode ser visto como um procedimento de prova de que cada pré-condição é alcançada.
- Novo procedimento CHOOSE_OPERATOR devolve o S_i que atende condição (1). Novo RESOLVE_THREAT atende condição (2).
- Usam abordagem de “resolver possíveis ameaças mais tarde”.

Gerador de Planos com Operadores Parcialmente Instanciados

```

procedure CHOOSE-OPERATOR(plan, operators, Snext, c)

  choose a step Sadd from operators or STEPS(plan) that has cadd as an effect
  such that  $u = \text{UNIFY}(c, c_{add}, \text{BINDINGS}(plan))$ 
  if there is no such step
    then fail
  add u to BINDINGS(plan)
  add  $S_{add} \xrightarrow{c} S_{next}$  to LINKS(plan)
  add  $S_{add} \prec S_{next}$  to ORDERINGS(plan)
  if Sadd is a newly added step from operators then
    add Sadd to STEPS(plan)
    add  $Start \prec S_{add} \prec Finish$  to ORDERINGS(plan)



---


procedure RESOLVE-THREATS(plan)

  for each  $S_i \xrightarrow{c} S_j$  in LINKS(plan) do
    for each Sthreat in STEPS(plan) do
      for each c' in EFFECT(Sthreat) do
        if SUBST(BINDINGS(plan), c) = SUBST(BINDINGS(plan),  $\neg c'$ ) then
          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
        end
      end
    end
  end

```

Gerador de Planos com Operadores Parcialmente Instanciados

- POP é “sound” e completo qdo trata de operadores parcialmente instanciados?
- Se todas as ameaças forem resolvidas e POP conseguir gerar um plano completo totalmente instanciado (por exemplo, instanciando variáveis através do conjunto de valores possíveis, “binding list”): sound.
- É completo porque o algoritmo gera todos os possíveis planos que atendem condição (1), e remove todos os planos que não atendem condição (2).

Engenharia do Conhecimento para Geração de Planos

- Metodologia para resolver problemas com a abordagem de geração de planos:
 - ▶ Decidir sobre o que falar.
 - ▶ Decidir o vocabulário de condições (literais), operadores, e objetos.
 - ▶ Codificar operadores.
 - ▶ Codificar uma descrição de uma instância do problema (estado inicial, por exemplo).
 - ▶ Apresentar problemas ao gerador de planos e obter planos (problema = objetivo).

Ex 1: O Mundo dos Blocos

- Sobre o que falar? blocos, mesa, pilhas de blocos, regras para movimentação de blocos.
- Vocabulário?
 - ▶ objetos (blocos e mesa) representados por constantes.
 - ▶ $Sobre(b, x)$ usado para representar que bloco b está sobre x , onde x pode ser um outro bloco ou a mesa.
 - ▶ Operador para mover blocos: $Move(b, x, y)$, move bloco b da posição no topo de x para a posição no topo de y .
 - ▶ Não podemos representar $\neg\exists x Sobre(x, b)$ como pré-condição, então usamos $TopoLimpo(b)$.

Ex 1: O Mundo dos Blocos

- Operadores?

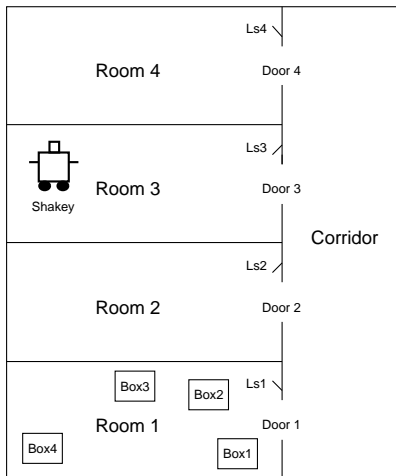
Op(ACAO: Move(b,x,y),

PRECOND: Sobre(b,x) \wedge TopoLimpo(b) \wedge TopoLimpo(y),

EFEITO: Sobre(b,y) \wedge TopoLimpo(x) \wedge
 \neg Sobre(b,x) \wedge \neg TopoLimpo(y))

- Problema: não opera bem quando x ou y é uma mesa.
- Solução: introduzir nova ação p/ mover um bloco b de x para a mesa (MoveParaMesa(b,x)) e nova interpretação para TopoLimpo(x) (há espaço vazio em x para colocar um bloco).
- Nada para prevenir planner de sempre usar Move($b,x,Mesa$) invés de MoveParaMesa(b,x). Espaço de busca maior, mas encontra solução.
- Sobre(B,C,C) pode ocorrer no plano.

Ex 2: O Mundo de Shakey



Ex 2: O Mundo de Shakey

- O que falar? caixas, interruptores, portas, posições, regras para movimentação.
- Vocabulário e Operadores?
 - ▶ $Va(y)$: vai da posição corrente para a posição y , pré-condição $Em(Shakey, x)$ representa a posição corrente. Para este problema, $EstaEm(x, r) \wedge EstaEm(y, r)$ para poder representar q uma porta está em duas salas ao mesmo tempo e traçar plano para Shakey passar de uma sala p / outra.
 - ▶ $Empurrar(b, x, y)$: empurrar objeto b de posição x para y .
 - ▶ $Subir(b)$: subir numa caixa.
 - ▶ $Descer(b)$: descer de uma caixa.
 - ▶ $AcenderLuz(int)$.
 - ▶ $ApagarLuz(int)$.