Cap. 13 – Geração de Planos e Ações

- No mundo real, info pode ser incorreta ou incompleta.
- Incompleta: o agente pode não saber onde está o leite.
- Incorreta: preço do leite pode ter subido, agente pode ter sido roubado.
- Duas abordagens:
  - Geração de *planos condicionais* (planejamento de contigências). Inclui *ações sensoriais* no plano para testar condições apropriadas.
  - Monitoração da Execução: monitora a execução do plano e, se necessário, gera outro plano – *replanning*.
- Combinação possível.

Cap. 13 – Geração de Planos Condicionais

Op(ACAO: Remover(x),
PRECOND: Colocado(x),
EFEITO: Removido(x) $\wedge$ RodaVazia(x) $\wedge$ $\neg$ Colocado(x))
Op(ACAO: Colocar(x),
PRECOND: Removido(x) $\wedge$ RodaVazia(x),
EFEITO: Colocado(x) $\wedge$ $\neg$ RodaVazia(x) $\wedge$ $\neg$ Removido(x))
Op(ACAO: Encher(x),
PRECOND: Novo(x) $\wedge$ Vazio(x),
Cheio(x) $\wedge$ $\neg$ Vazio(x))

- Objetivo: Colocado(x) $\wedge$ Cheio(x)
- Cond iniciais: Cheio(Estepe) $\wedge$ Novo(Estepe) $\wedge$ Removido(Estepe) $\wedge$ Colocado(Pneu1) $\wedge$ Vazio(Pneu1)
- Solução assumindo que nada acontece de imprevisto: [Remover(Pneu1),Colocar(Estepe)]

Cap. 13 – Geração de Planos Condicionais

- Plano gerado vai sempre trocar Pneu1 mesmo q ele não esteja furado.
- Cond: *if Pneu1 não furado então encher, senão remover e colocar o estepe.*
- Plano gerado: if(NaoFurado(Pneu1),[Encher(Pneu1)], [Remover(Pneu1),Colocar(Estepe)])
- Plano condicional pode ser melhor do que plano sem condições, mais geral.
- Pode ser necessário: se o agente não souber se o estepe está cheio ou vazio.
- Se o pneu a ser trocado e o estepe estiverem furados, nenhum gerador de planos vai gerar um plano que seja solução.

Cap. 13 – Geração de Planos Condicionais

- Gerador de Planos condicionais gera todos os planos possíveis de sucesso e *Falhas* nos caminhos onde não há solução possível.
- Execução de planos condicionais:
  - Se for um passo condicional, testar condição contra o banco de dados.
  - Se condição for falsa, executar plano do ramo 'then'.
  - senão, executar plano do ramo 'else'.

```
function CONDITIONAL-PLANNING-AGENT(percept) returns an action
   static: KB, a knowledge base (includes action descriptions)
           p, a plan, initially NoPlan
           t, a counter, initially 0, indicating time
           G, a goal

   TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
   current ← STATE-DESCRIPTION(KB, t)
   if p = NoPlan then p ← CPOP(current, G, KB)
   if p = NoPlan or p is empty then action ← NoOp
   else
       action ← FIRST(p)
       while CONDITIONAL?(action) do
           if ASK(KB, CONDITION-PART[action]) then p ← APPEND(THEN-PART[action], REST(p))
           else p ← APPEND(ELSE-PART[action], REST(p))
           action ← FIRST(p)
       end
       p ← REST(p)
   TELL(KB, MAKE-ACTION-SENTENCE(action, t))
   t ← t + 1
   return action
```

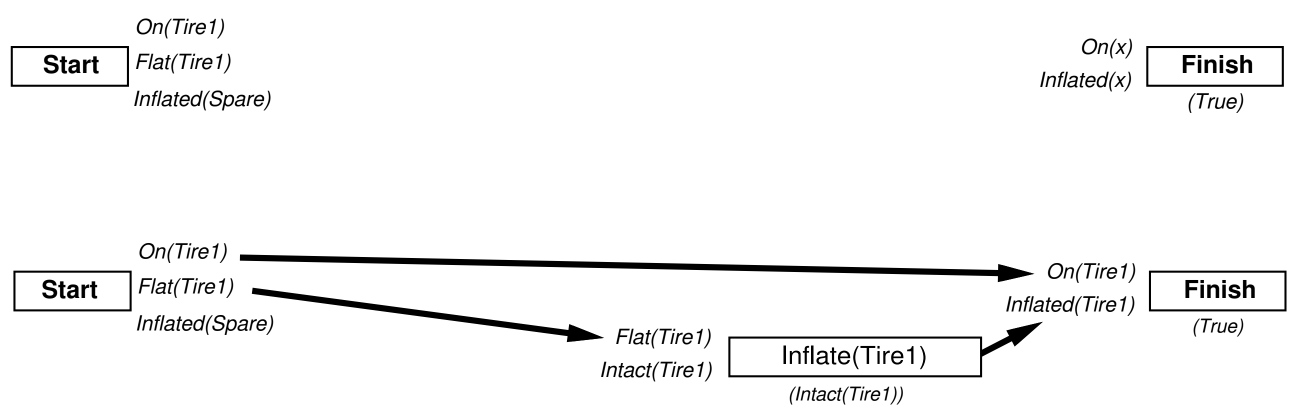
Cap. 13 – Geração de Planos Condicionais

- Durante a execução do plano, agentes têm que ser capazes de decidir se uma condição é verdadeira ou falsa.
- Para garantir que um plano condicional é executável, o agente deve inserir novas ações ao plano.
- Gerador de Planos condicionais deve gerar ações que o agente possa executar para obter mais informações sobre o mundo. Ex:

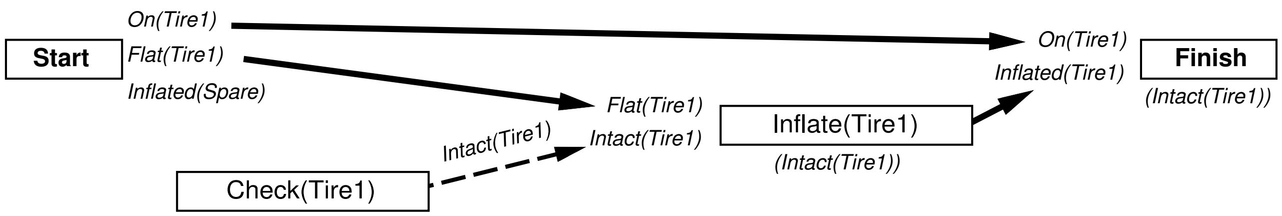
  Op(ACAO: VerificarPneu(x),
  PRECOND: Pneu(x),
  EFEITO: SabeSe("NaoFurado(x)"))

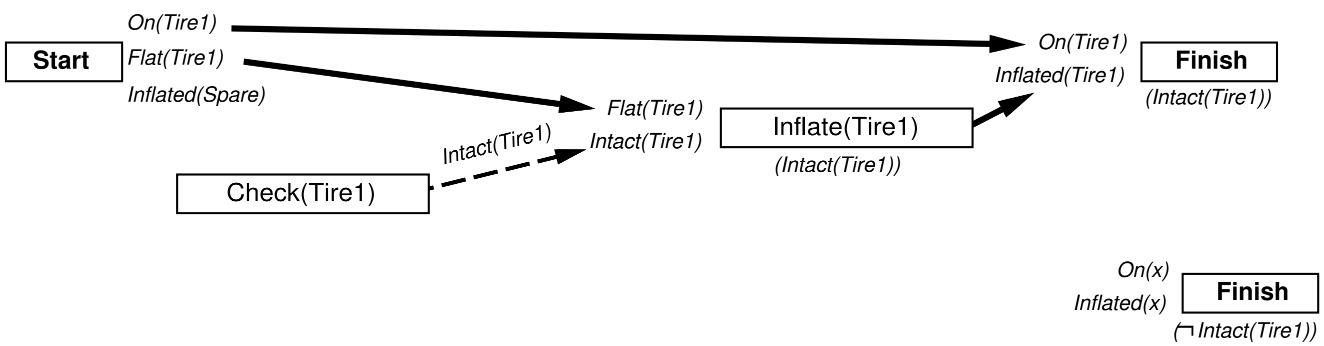
- Algoritmo: CPOP, como POP, mas utiliza estrutura *contexto* associada a cada passo (lista de condições que devem ser atendidas para que o passo seja executado).
- Passos subsequentes no plano herdam o contexto anterior.

Cap. 13 – Geração de Planos Condicionais

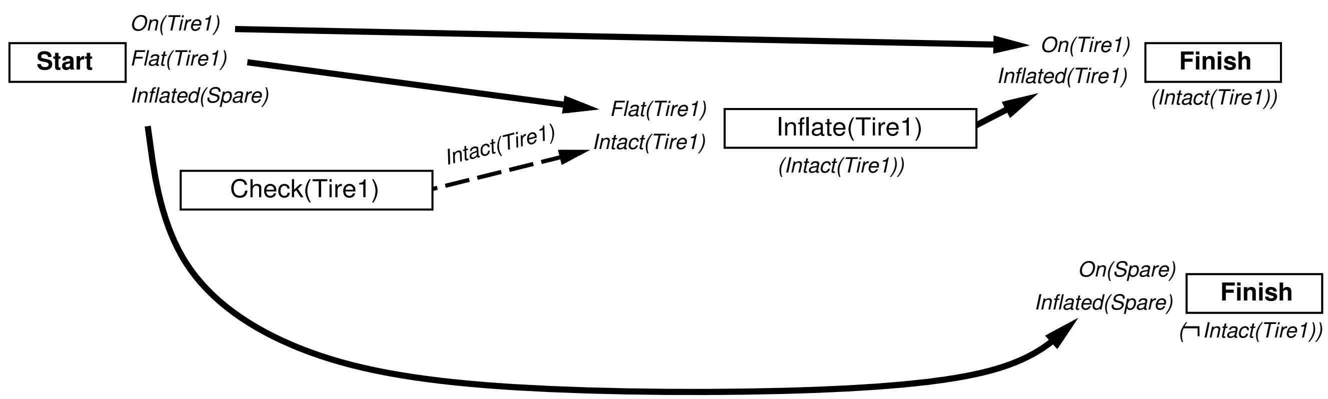


Cap. 13 – Geração de Planos Condicionais

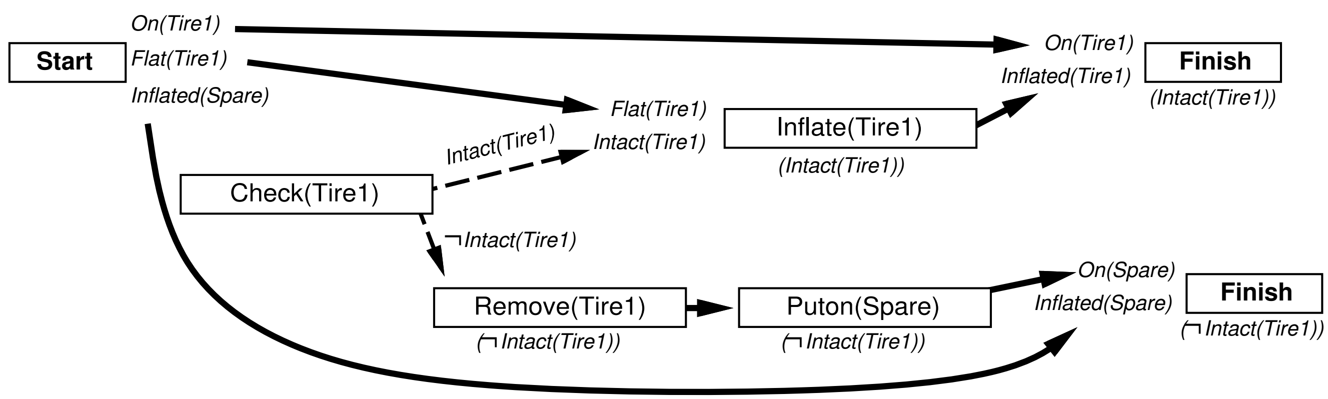




Cap. 13 – Geração de Planos Condicionais



Cap. 13 – Geração de Planos Condicionais



function CPOP(*initial, goals, operators*) **returns** *plan*

plan ← MAKE-PLAN(*initial, goals*)
loop do

Termination:
if there are no unsatisfied preconditions
and the contexts of the finish steps are exhaustive
then return *plan*

Alternative context generation:
if the plans for existing finish steps are complete and have contexts $C_1 \ldots C_n$ **then**
add a new finish step with a context $\neg(C_1 \vee \ldots \vee C_n)$
this becomes the *current context*

Subgoal selection and addition:
find a plan step S_{need} with an open precondition c

Action selection:
choose a step S_{add} from *operators* or STEPS(*plan*) that adds c or
knowledge of c and has a context compatible with the current context
if there is no such step
then fail
add $S_{add} \xrightarrow{c} S_{need}$ to LINKS(*plan*)
add $S_{add} < S_{need}$ to ORDERINGS(*plan*)
if S_{add} is a newly added step **then**
add S_{add} to STEPS(*plan*)
add $Start < S_{add} < Finish$ to ORDERINGS(*plan*)

Threat resolution:
for each step S_{threat} that potentially threatens any causal link $S_i \xrightarrow{c} S_j$
with a compatible context **do**
choose one of
Promotion: Add $S_{threat} < S_i$ to ORDERINGS(*plan*)
Demotion: Add $S_j < S_{threat}$ to ORDERINGS(*plan*)
Conditioning:
find a conditional step S_{cond} possibly before both S_{threat} and S_j, where
1. the context of S_{cond} is compatible with the contexts of S_{threat} and S_j;
2. the step has outcomes consistent with S_{threat} and S_j, respectively
add conditioning links for the outcomes from S_{cond} to S_{threat} and S_j
augment and propagate the contexts of S_{threat} and S_j

if no choice is consistent
then fail
end
end