

Eficiência num Planejador Abdutivo



Pesquisa financiada pela CAPES e CNPq

Silvio do Lago Pereira &
Leliane Nunes de Barros,
DCC / IME-USP

Introdução



- O GPS (*Newell & Simon, 1961*)
- O QA3 (*Green, 1969*)
- O STRIPS (*Fikes & Nilsson, 1971*)
- O AECIP (*Shanahan, 2000*)



Objetivo

Sistemas de
planejamento
lógicos

≡

Sistemas de
planejamento
algorítmicos

Organização

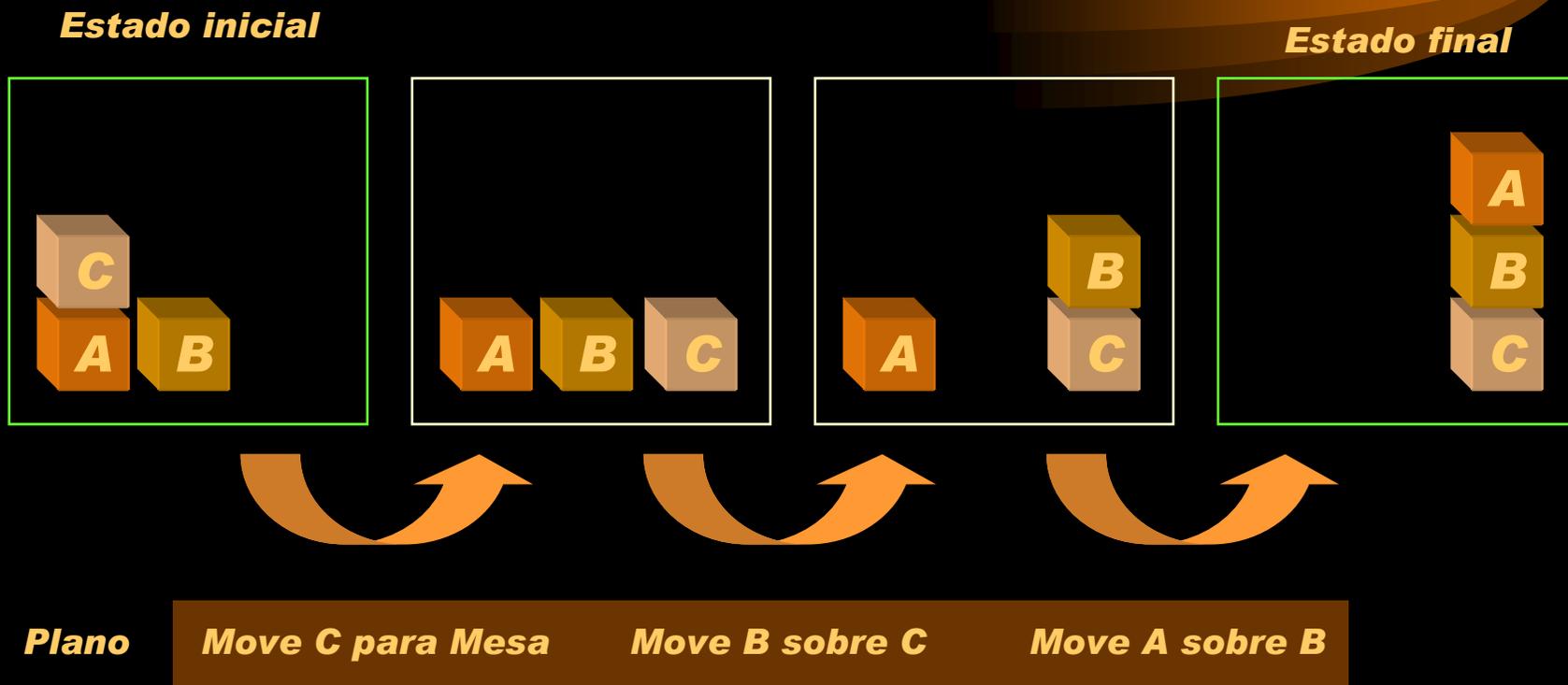


- A tarefa de planejamento
- Raciocínio lógico abduutivo
- Planejamento abduutivo com cálculo de eventos
- Resultados empíricos
- Conclusão

A TAREFA DE PLANEJAMENTO



A tarefa de planejamento



A representação STRIPS

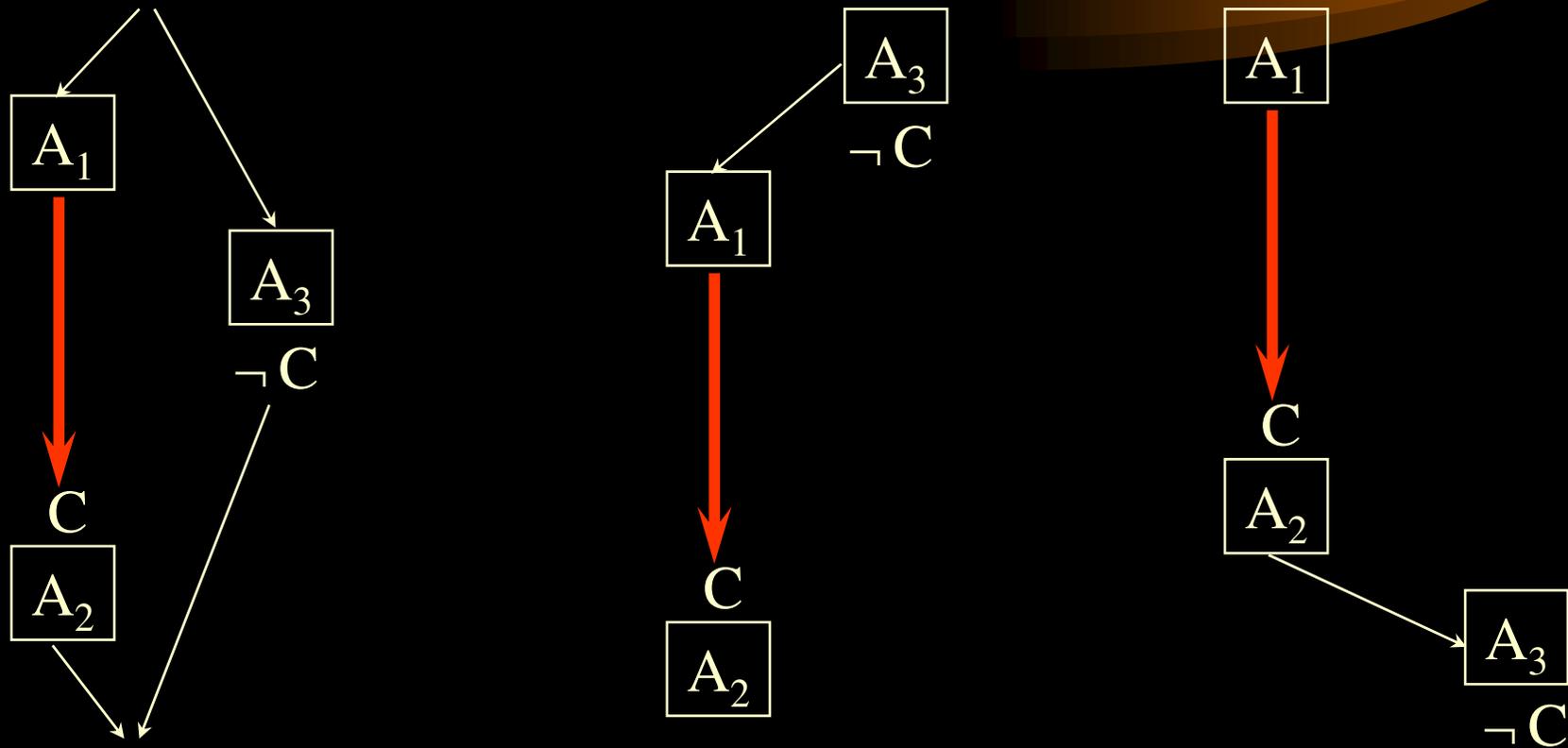
- Um *estado* é um conjunto de literais positivos.
- Uma *ação* é um operador que transforma um estado em outro.

*Op(act: Mover(A,B,C),
pre: { Sobre(A,B), Livre(A), Livre(C) },
add: { Sobre(A,C), Livre(B) },
del: { Sobre(A,B), Livre(C) })*

A representação STRIPS

- Um *problema* é uma tripla da forma $\langle B, I, M \rangle$, onde:
 - B é uma biblioteca de operadores,
 - I é um estado inicial e
 - M é uma conjunção de submetas.
- Um *plano* é uma tripla da forma $\langle P, O, V \rangle$, onde:
 - P é um conjunto de passos,
 - O é um conjunto de restrições de ordem temporal e
 - V é o conjunto de vínculos causais.

Vínculos causais e tratamento de ameaças



RACIOCÍNIO LÓGICO ABDUTIVO



Princípio da Abdução

<i>(conhecemos)</i>	$\alpha \rightarrow \beta$
<i>(observamos)</i>	β
	<hr/>
<i>(concluimos)</i>	α

Abdução

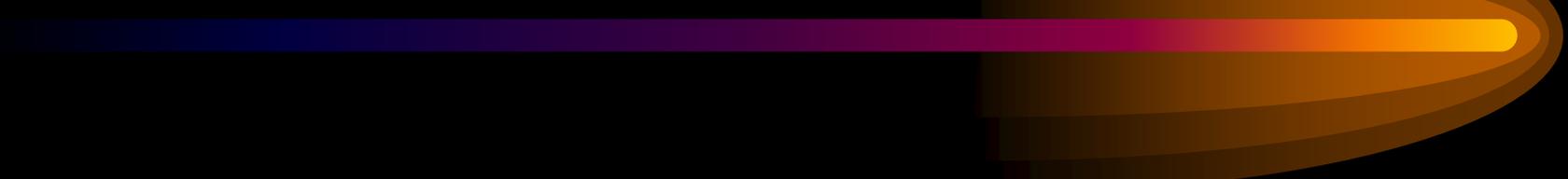


Seja Δ a descrição de um domínio e seja Γ_0 a descrição de uma observação nesse domínio.

Abdução consiste em encontrar um conjunto de sentenças Σ , *explicação abdutiva* de Γ_0 , tal que

- ♦ $\Delta \cup \Sigma$ é consistente
- ♦ $\Delta \cup \Sigma \models \Gamma_0$

O meta-interpretador abduativo



O *meta-interpretador* é um programa lógico que estende a capacidade do PROLOG de modo que, além de dedução, ele também seja capaz de realizar abdução.

PLANEJAMENTO ABDUTIVO COM CÁLCULO DE EVENTOS



Axiomas do cálculo de eventos

$$\text{HoldsAt}(f,t) \leftarrow \text{Initially}(f) \wedge \neg \text{Clipped}(0,f,t)$$

$$\text{HoldsAt}(f,t) \leftarrow \text{Happens}(a,t') \wedge \text{Initiates}(a,f,t') \wedge \\ t' < t \wedge \neg \text{Clipped}(t', f, t)$$

$$\text{Clipped}(t_1,f,t_2) \leftrightarrow \exists a,t' [\text{Happens}(a,t') \wedge t_1 < t' \wedge \\ t' < t_2 \wedge \text{Terminates}(a,f,t')]$$

Axiomas do domínio: exemplo

Initially(Em(A))

Initiates(Andar(A,B), Em(B), t) ←

HoldsAt(Em(A), t)

Terminates(Andar(A,B), Em(A), t) ←

HoldsAt(Em(A), t)

Planejamento abduutivo

- *Domínio Δ : Initiates e Terminates*
- *Situação inicial Σ : Initially*
- *Meta Γ : HoldsAt*
- *Plano Π : Happens e Before*
- *Planejamento: $\text{CIRC}[\Delta; \text{Initiates}, \text{Terminates}] \wedge \text{CIRC}[\Sigma \wedge \Pi; \text{Happens}] \wedge \text{EC} \wedge \Omega \models \Gamma$*

O meta-interpretador abduativo especializado para o CE

$$\text{HoldsAt}(f,t) \leftarrow \text{Happens}(a,t') \wedge \text{Initiates}(a,f,t') \wedge t' < t \wedge \neg \text{Clipped}(t', f, t)$$

```
abp([holds_at(F1, T3) | Gs1], R1, R5, N1, N4) :-
  abresolve(initiates(A, F1, T1), R1, Gs2, R1),
  abresolve(happens(A, T1, T2), R1, [], R2),
  abresolve(before(T2, T3), R2, [], R3),
  add_neg([clipped(T1, F1, T3)], N1, N2),
  nafs(N2, R3, R4, N2, N3),
  append(Gs2, Gs1, Gs3),
  abp(Gs3, R4, R5, N3, N4).
```

RESULTADOS EMPÍRICOS



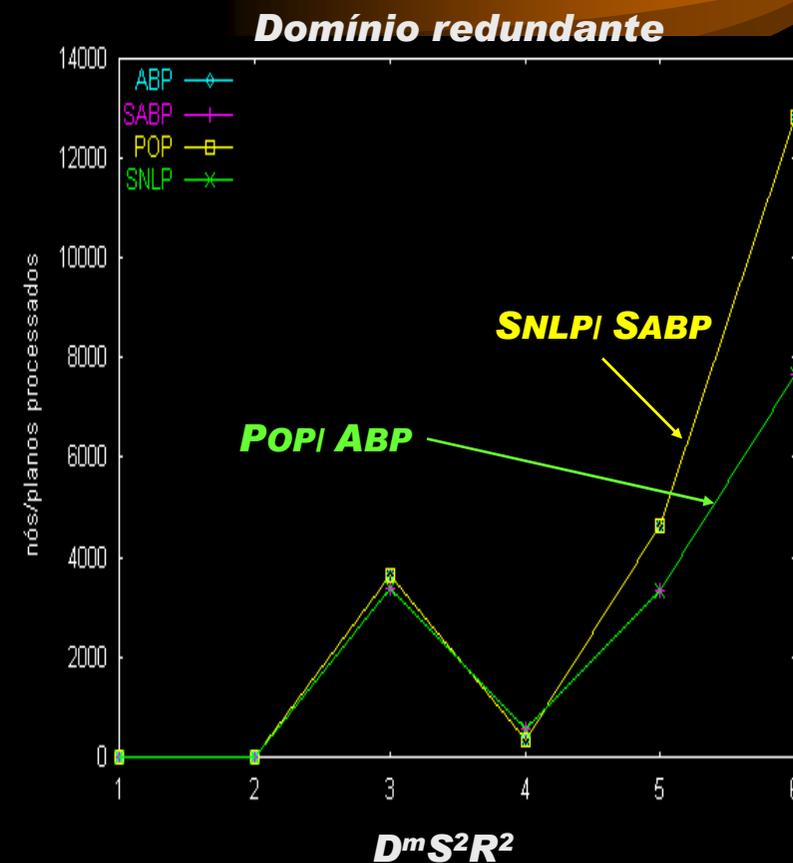
Domínios de teste

- *Submetas Independentes: D^0S^1*
- *Submetas trivialmente-serializáveis: D^1S^1*
- *Submetas laboriosamente-serializáveis: D^mS^1*
- *Submetas não-serializáveis: D^mS^2*
- *Versões redundantes desses domínios: D^mS^nR*

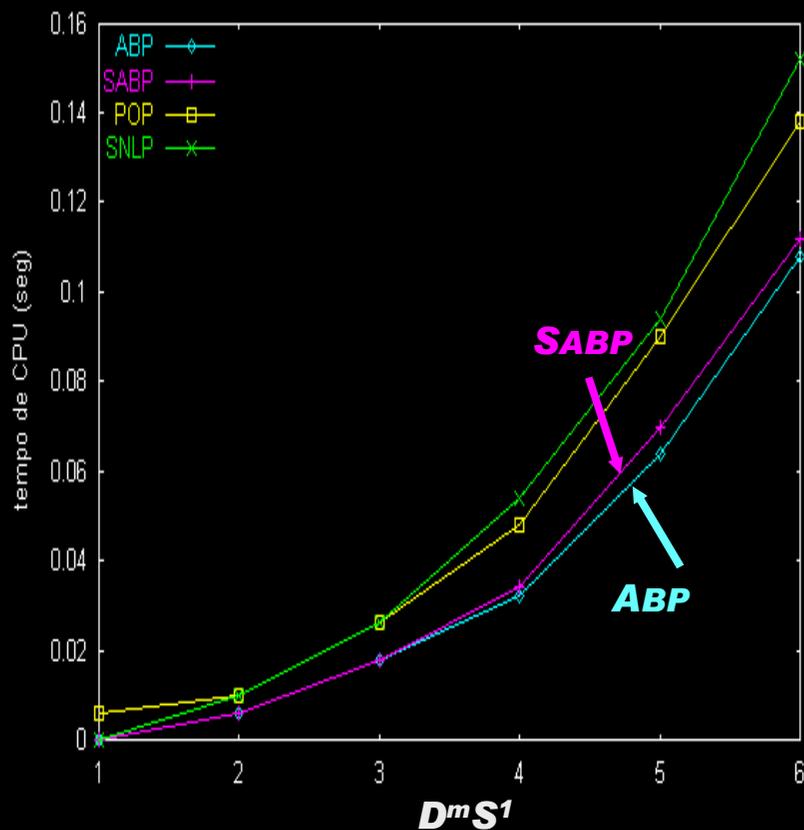
Sistemas implementados e comparados

POP	Planejador de ordem parcial com tratamento não-exaustivo de conflitos
SNLP	Planejador de ordem parcial com tratamento exaustivo de conflitos (sistemático)
ABP	Planejador abduativo baseado em cálculo de eventos
SABP	Planejador abduativo baseado em cálculo de eventos e sistemático

Equivalência das implementações: espaço de busca explorado



Equivalência das implementações: tempo de CPU consumido

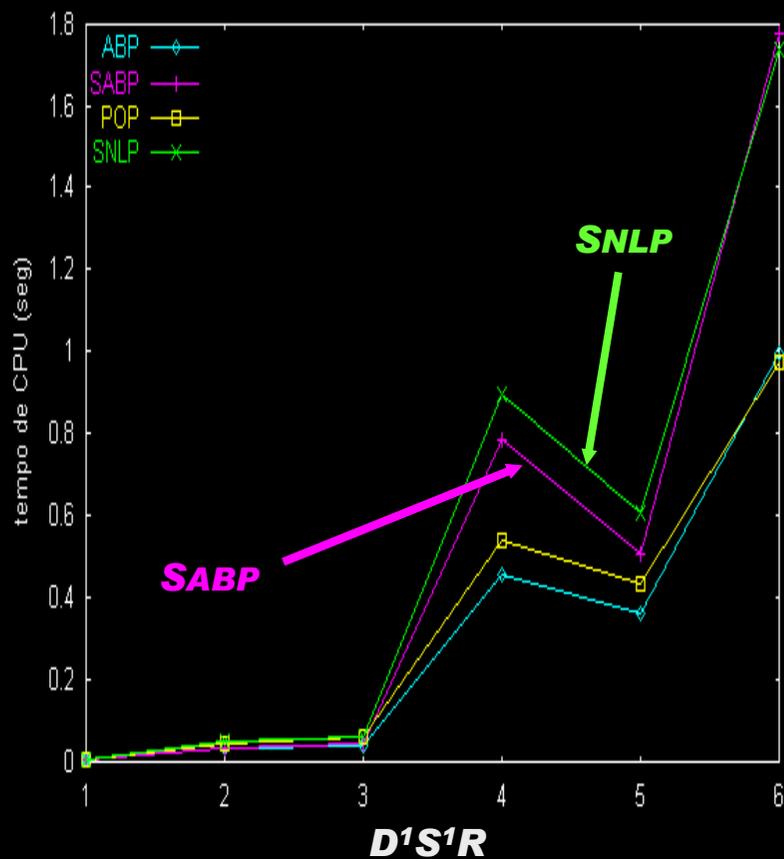


$Op(act: A_i,$
 $pre: \{I_j\},$
 $add: \{G_i\},$
 $del: \{I_j \mid j < i\})$

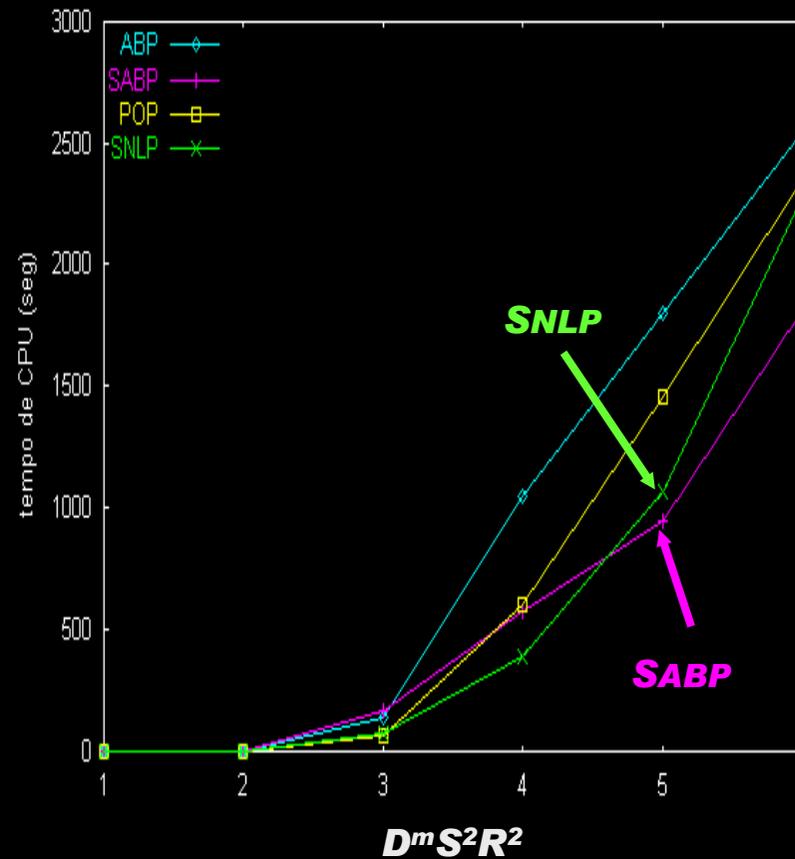
$Initiates(A_i, G_i, T) \leftarrow$
 $HoldsAt(I_i, T)$
 $Terminates(A_i, I_1, T) \leftarrow$
 $HoldsAt(I_i, T)$
...

Influência da sistematicidade

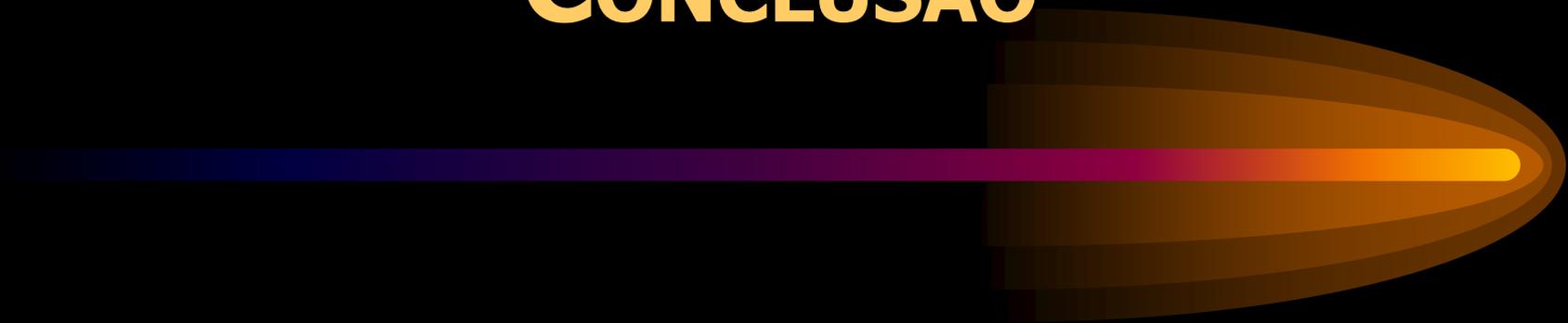
Pouco redundante



Muito redundante



CONCLUSÃO



Resultados

- Sistemas de planejamento baseados em lógica podem ser tão eficientes quanto os sistemas de planejamento clássico.
- O planejador abduutivo baseado em cálculo de eventos pode ser ajustado para apresentar um melhor desempenho, de acordo com as características do domínio de aplicação.

Vantagens

- Por se basear num formalismo lógico, o planejador ABP pode ser mais facilmente entendido, mantido e modificado.
- Sua linguagem de representação de ações pode ser mais facilmente estendida para tratar *efeitos condicionais, ações compostas, consumo de recursos, etc.*

FIM

