MAC5788
Planejamento em IA

O planejador Graphplan

Leliane Nunes de Barros

Grafo de Planejamento [Blum&Furst 1997]

- Grafo de níveis: seqüência de níveis que correspondem a passos de tempo no plano. Nível 0 é o estado inicial.
  - nível *i*: contém um conjunto de proposições
  - nível *i+1*: contém um conjunto de ações

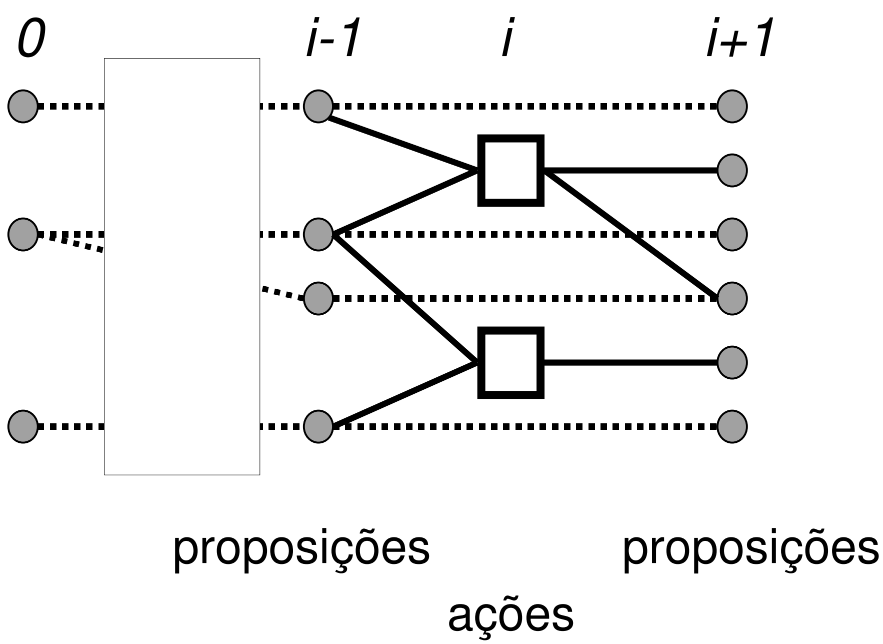
> Nível *i* contém todos os literais que *poderiam* ser verdadeiros no passo *i* , dependendo das ações executadas nos passos *j* > *i*

GRAPHPLAN [Blum&Furst 1997]

- O planejador *Graphplan* alterna entre duas fases:
  - expansão do grafo de planejamento: extende o grafo progressivamente até atingir uma condição necessária (mas não suficiente) para um plano existir.
  - extração da solução: faz uma busca regressiva no grafo a procura de um plano solução; se não o encontrar, *Graphplan* volta a expandir o plano.

Grafo de Planejamento

- Expansão do Grafo de Planejamento

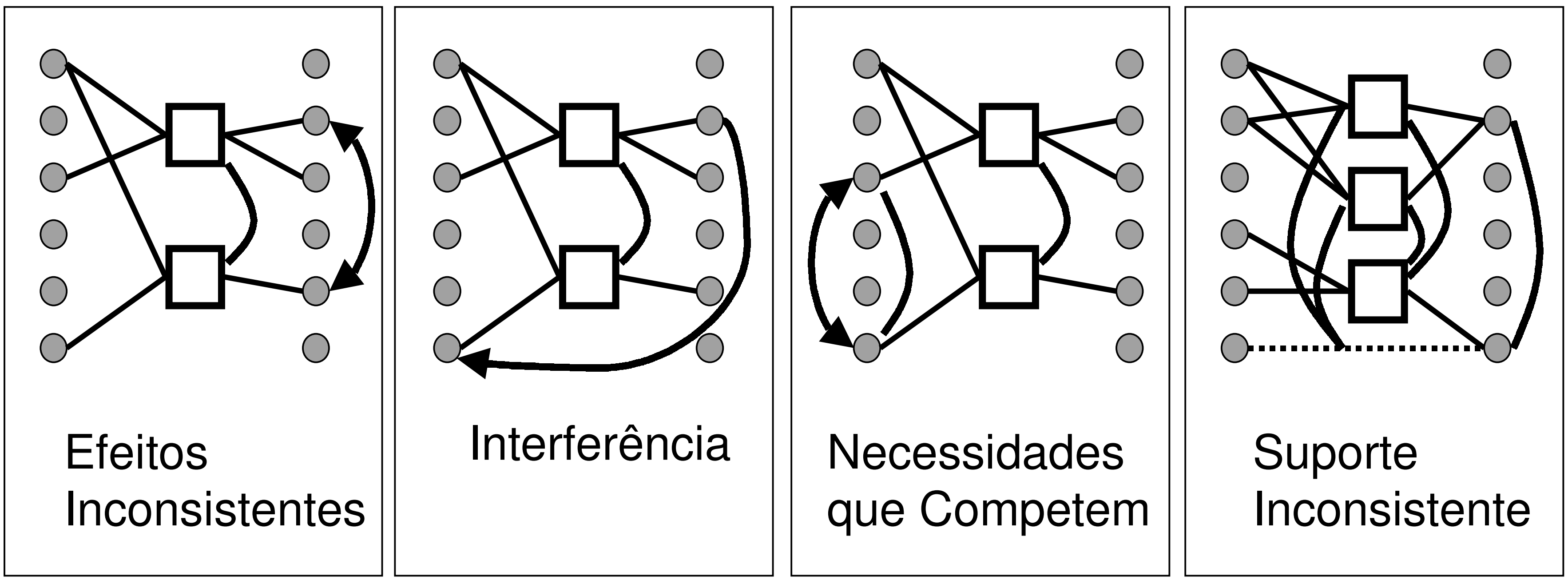


Grafo de Planejamento

- Funciona apenas com ações proposicionais e estados completos
- O processo de construção do grafo não envolve escolha de ações (busca combinatorial) mas registra ações que não podem ser escolhidas com ligações *mutex* (*mutual exclusion*)
- A construção do grafo é polinomial no número de ações e proposições
- ações "paralelas" num mesmo nível não significa que elas serão executadas ao mesmo tempo.

Grafo de Planejamento

- Relações de Mutex (Exclusão Mútua - arcos em amarelo)



Ligações de *mutex* entre ações

- Uma relação de *mutex*(*A*,*B*) ocorre entre duas ações se uma das 3 condições ocorrerem:
  - Efeitos inconsistentes: o efeito da ação *A* é a negação do efeito da ação *B*;
  - Interferência: a ação *A* elimina a precondição da ação *B*;
  - Necessidades que competem: as ações *A* e *B* tem precondições com ligação mutex no nível anterior.

Ligações de *mutex* entre proposições

- <u>Suporte inconsistente</u>: Duas proposições *p* e *q* no nível *i* são mutex se (1) *p* é a negação de *q* *ou* (2) se todas as maneiras de se atingir *p* e *q* são mutexs maneiras

Problema: "Have a cake and eat cake too"

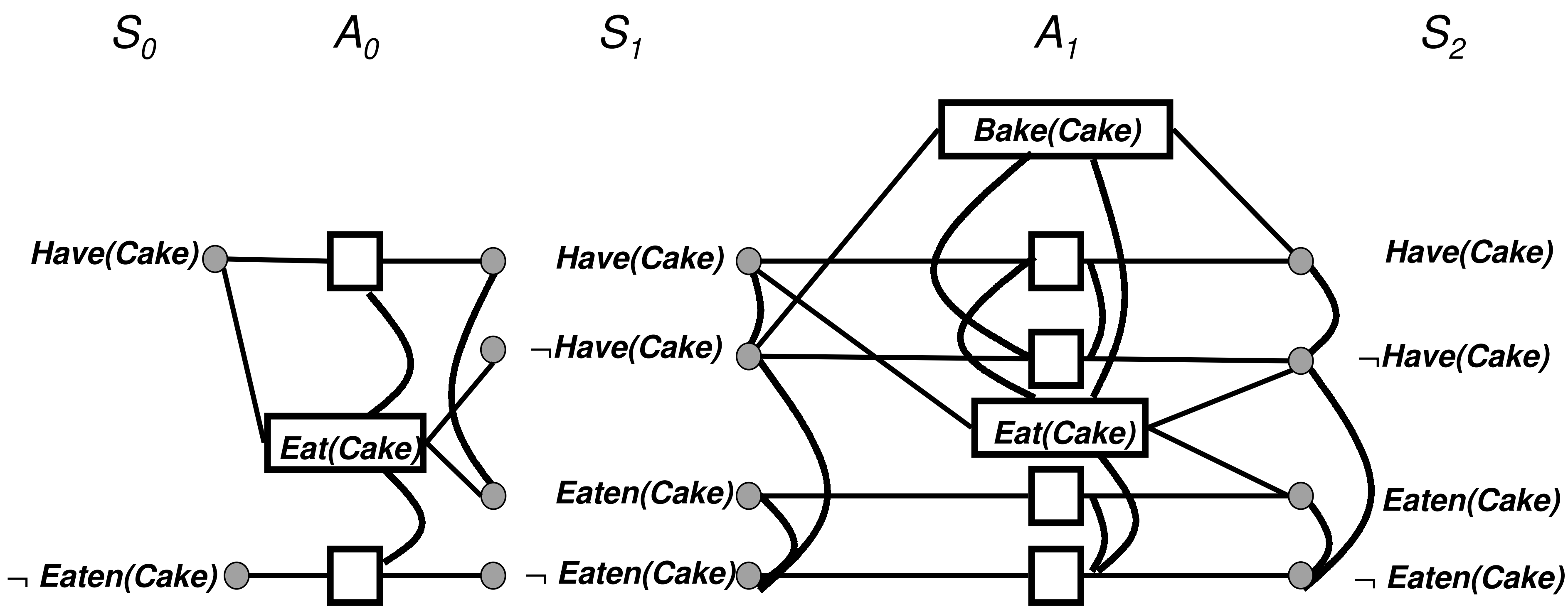
Init (*Have*(*Cake*))

Goal (*Have*(*Cake*) $\wedge$ *Eaten*(*Cake*))

Action (*Eat*(*Cake*)
Precond: *Have*(*Cake*)
Effect: $\neg$ *Have*(*Cake*) $\wedge$ *Eaten*(*Cake*))

Action (*Bake*(*Cake*)
Precond: $\neg$ *Have*(*Cake*)
Effect: *Have*(Cake))

Grafo de Planejamento



Domínio Jantar Surpresa

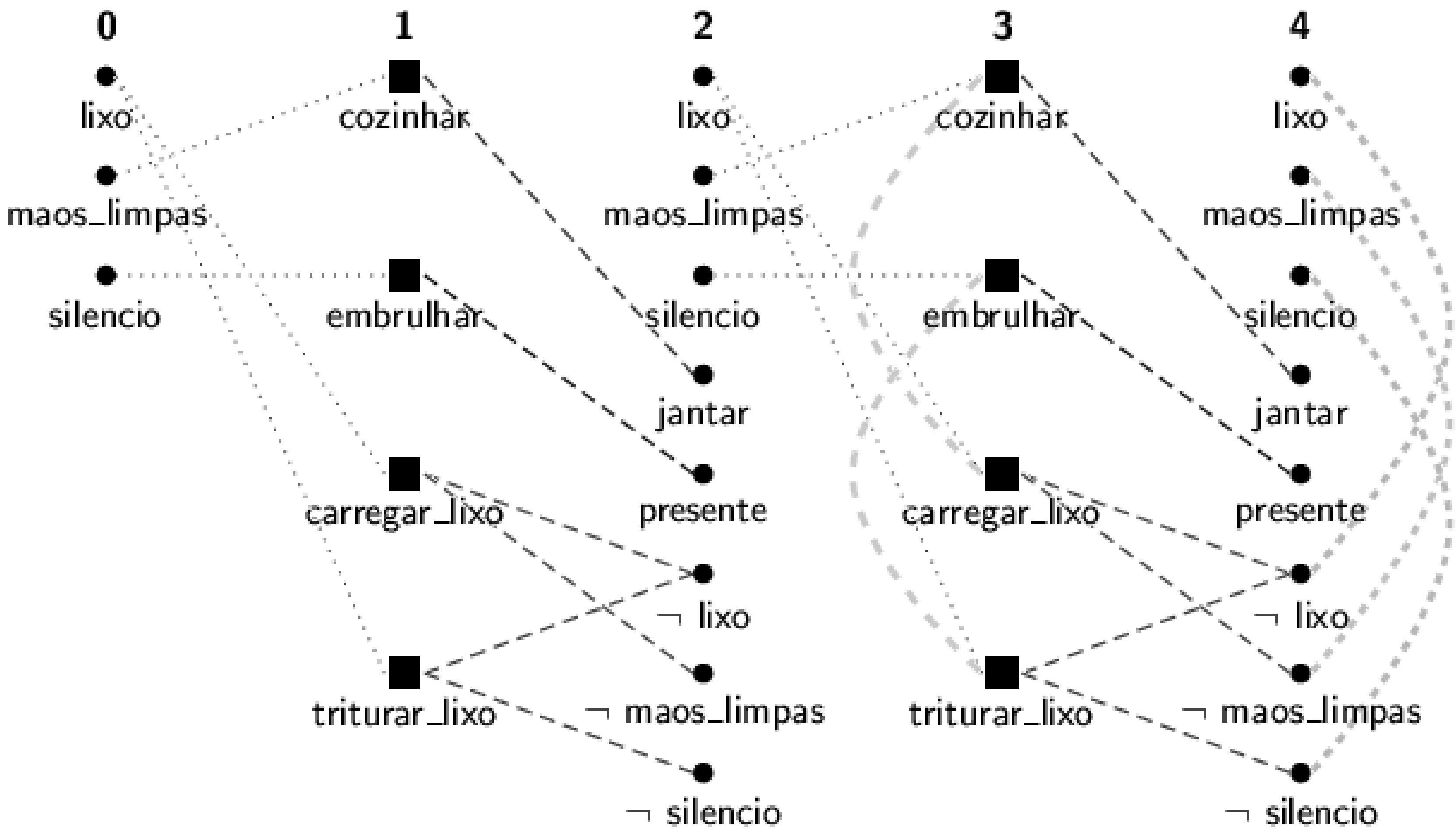
cozinhar :prec (mãos_limpas)
:efeito (jantar)

embrulhar :prec (silêncio)
:efeito (presente)

carregar_lixo :prec (lixo)
:efeito (¬lixo, ¬mãos_limpas)

triturar_lixo :prec (lixo)
:efeito (¬lixo, ¬silêncio)

Grafo de Planejamento



Grafos de Planejamento

- O grafo de planejamento para um dado problema, por sere rico em informação, pode ser usado para o cálculo de heurística :
  - uma proposição que não pertence ao estado final não pode ser satisfeita por nenhum plano: $h(n) = \infty$
  - o nível em que uma sub-meta *g* aparece pela primeira vez no grafo pode ser usado como uma estimativa do custo de se atingir a meta (*level cost*)

Grafos de Planejamento

- Pode ser visto como uma solução eficiente para um problema relaxado (para o qual alguns conflitos não são resolvidos)
- Seja *g* uma sub-meta de um problema de planejamento

Se *g* <u>não aparece</u> em nenhum nível do grafo de planejamento **então** não existe um plano que atinja *g*

Se *g* <u>aparece</u> num nível S_i do grafo de planejamento **então** é *possível* que exista um plano que satisfaz *g* (condição necessária mas não suficiente)

(podem existir outras relações de conflito além de mutex, envolvendo 3 ou mais ações, porém, são mais difíceis de serem tratadas)

Grafos de planejamento para cálculo de heurística

- Como estimar o custo de metas conjuntivas
  - **max-level:** heurística admissível mas pouco informativa
  - **level sum:** heurística não-admissível mas é mais informativa que a heurística simples que calcula o número de metas não satisfeitas
  - **set-level**: nível em que todas as metas da conjunção aparecem sem existir mutex entre elas. Domina a heurística max-level e funciona melhor em domínios com muita interação

Grafo de Planejamento

- Alguns planejadores baseados em Grafo de Planejamento
  - GraphPlan [Blum&Furst 1997]
  - IPP [Koehler&al 1997]
  - STAN [Long&Fox 1999]
  - SAPA [Do&Kambhampati 2001]
- Codificação SAT
  - SatPlan + Blackbox [Kautz&Selman 1999]

Algoritmo Graphplan

- Dois passo que alternam dentro de um laço
  - checa se todos as sub-metas estão no último nível do grafo (cond-1)
    - se isto for verdade então pode existir uma solução no grafo atual e se tenta extrair uma solução.
  - caso a cond-1 não seja satisfeita ou não seja possível extrair uma solução, o grafo deve ser expandido.

Algoritmo Graphplan

function Graphplan(*problem)* **returns** *solution ou failure*

graph ← Initial-Planning-Graph(*problem*)

goals ← Goals[*problem]*

loop do

if *goals* all non-mutex in last level of *graph* **then do**

solution ← Extract-Solution(*graph, goals,* Length(graph))

if *solution* ≠ *failure* **then return** *solution*

else if No-Solution-Possible(*graph*) **then return** *failure*

graph ← Expand-Graph(*graph*, *problem*)

Extração da solução - CSP

- Extract-Solution resolve um CSP booleano cujas variáveis são as ações a cada nível e os valores para cada variável são 1 (dentro) ou 0 (fora do plano)
- uso de um algoritmo de CSP

Extração da solução - busca

- Extract-Solution resolve um problema de busca regressiva onde cada nó aponta para um nível do grafo
  - estado inicial corresponde ao último nível do grafo, (S_n) junto com o conjunto de sub-metas do problema
  - as ações em A_{n-1} devem ser selecionadas de forma que cubram as sub-metas e que estejam "livres-de-conflito"
  - ações "livres-de-conflito" são ações que não são mutex (2 a 2) e cujas pré-condições também não tenham relação de mutex.
  - o objetivo do problema de busca é alcançar o nível S_0 tal que todas as sub-metas sejam satisfeitas
  - o custo de cada ações é 1 (no-ops é 0)

Extração da solução - heurística

- Planejamento é PSPACE-complete
  - construção do grafo de planejamento gasta tempo polinomial
  - a extração da solução pode ser intratável no pior caso.

Extração da solução

É preciso usar algumas heurísticas para guiar a escolha das ações durante a busca regressiva

Exemplo: um algoritmo guloso baseado no *level cost* dos literais. Para qualquer conjunto de sub-metas:

1. Selecione primeiro o literal de maior custo
2. Para atingir esse literal, escolha a ação com pré-condições mais "fáceis" (pré-condição com menor **max-level** ou **level sum**)

Condição de Término do Graphplan

- Graphplan pode entrar num laço infinito
  - literais crescem monotonicamente (persistência dos literais)
  - ações crescem monotonicamente (literais da precondição estão sempre presentes)
  - o número de mutex decresce monotonicamente (ações mutex no nível A_i serão mutex nos níveis anteriores que elas aparecerem)