Planejamento & Escalonamento
O algoritmo FF-métrico

Aldebaran Perseke
aldeba@ime.usp.br

25 de setembro de 2002

Planejamento & Escalonamento

Problemas de planejamento e escalonamento envolvem os seguintes aspectos:

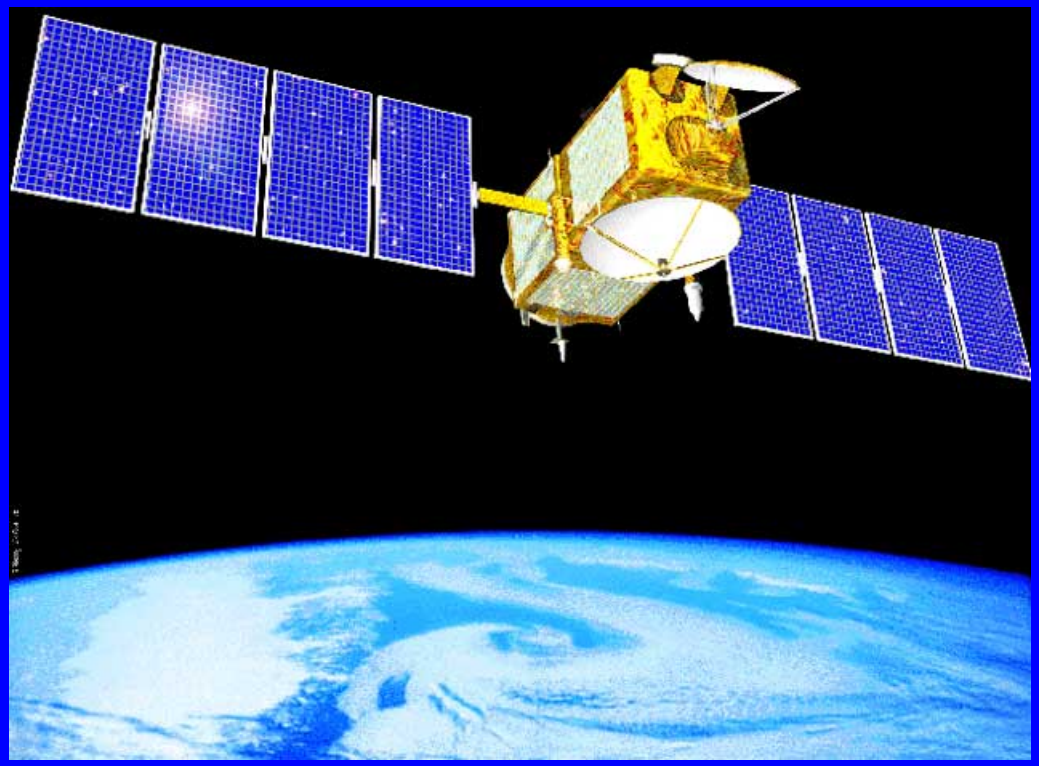
- escolha de ações;
- escolha da ordem em que ações deverão ser realizadas;
- alocação de recursos às ações/tarefas;
- otimização de uma função objetivo;

Planejamento & Escalonamento: estratégias

- Os pesquisadores de planejamento em IA buscam formas de representação e algoritmos independentes dos domínios a serem resolvidos.
- Os pesquisadores de escalonamento buscam formas de classificação dos problemas e algoritmos para a resolução de classes específicas de problemas.
- Muitos problemas reais apresentam aspectos típicos tanto de planejamento quanto de escalonamento.

Novos desafios

O desafio atual dos pesquisadores de planejamento e escalonamento é aliar o melhor de cada uma das áreas de forma a resolver problemas muito mais complexos de forma eficiente.



Planejamento em IA

Planejamento em **IA** é um problema que pode ser descrito por um modelo de estados, caracterizado por:

- Um espaço de estados S, finito e discreto;
- Um estado inicial $s_0 \in S$;
- Um conjunto de estados objetivo $S_G \subset S$;
- Um conjunto de ações $A(s) \subset A$, aplicáveis em um estado $s \in S$;
- Uma função f de transição de estados, que mapeia um estado s em um estado $s' = f(s, a)$ para $s, s' \in S$ e $a \in A(s)$;

Planejamento Clássico

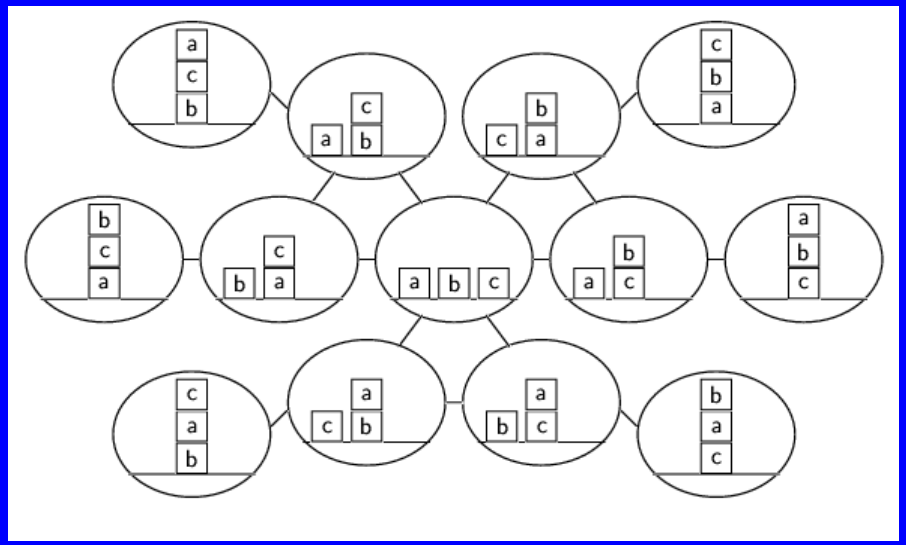
Prova automática de teoremas: utilização de um sistema de prova automática de teoremas empregando uma linguagem lógica para descrição de domínio, como por exemplo, o cálculo de situações ou o cálculo de eventos;

Busca pelo espaço de estados: busca em um grafo onde estados são representados por vértices e ações são representadas por arestas;

Busca pelo espaço de planos: busca em um grafo onde planos, não necessariamente completos e corretos, são representados por vértices e operações de refinamento do plano são representadas por arestas;

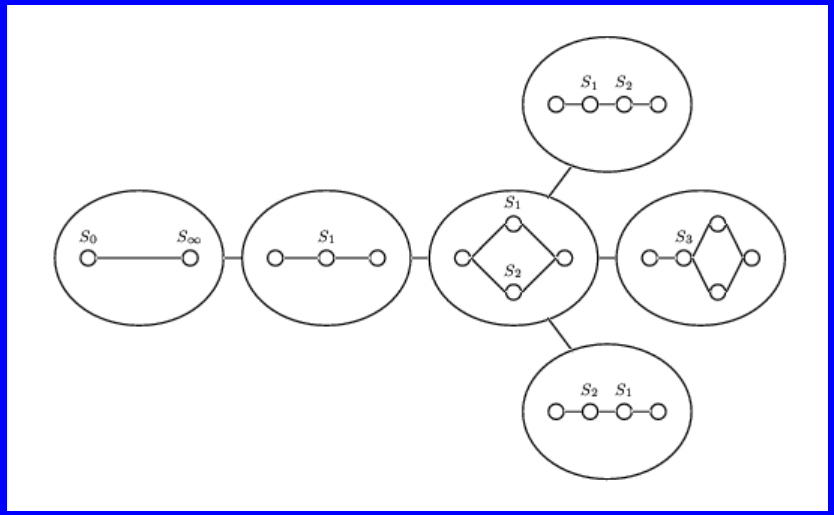
Planejamento Clássico

Espaço de estados



Raciocínio sobre estados do mundo

Espaço de planos



Raciocínio sobre geração de planos

Evolução do planejamento clássico

Planejamento como satisfatibilidade: traduz o problema de planejamento em um conjunto de **CNF**s a ser resolvido por um algoritmo **SAT**;

Planejamento Graphplan: utiliza um *grafo de planejamento* para resolver o problema, representando planos parciais e as relações de exclusão mútua entre possíveis ações;

Planejamento Blackbox: codifica o *grafo de planejamento* em um conjunto de **CNF**s a ser resolvido por um algoritmo **SAT**;

Planejamento como busca heurística: utiliza alguma função heurística específica para o problema de planejamento de forma a orientar um algoritmo de busca em grafo;

Arquiteturas para planejamento e escalonamento

Planejamento e escalonamento estratificados: a escolha das ações é feita antes, por um planejador, e o problema de alocação de recursos e otimizao é resolvido depois;

Planejamento e escalonamento intercalados: após a escolha de uma ação os conflitos são resolvidos através da adição de restrições ao plano em construção;

Planejamento e escalonamento homogêneos: o problema como um todo é resolvido por um único algoritmo de propósito geral, por exemplo, programação inteira ou programação linear.

Exemplo de descrição de ações - A linguagem PDDL

```
( :action turn_to
  :parameters ( ?s - satellite ?d_new - direction ?d_prev - direction )
  :precondition ( and ( pointing ?s ?d_prev )
      ( not ( = ?d_new ?d_prev ) ) )
  :effect ( and ( pointing ?s ?d_new )
      ( not ( pointing ?s ?d_prev ) ) )
)

( :action take_image
  :parameters ( ?s - satellite ?d - direction ?i - instrument ?m - mode )
  :precondition ( and ( calibrated ?i )
      ( on_board ?i ?s ) ( supports ?i ?m )
      ( power_on ?i ) ( pointing ?s ?d ) )
  :effect ( have_image ?d ?m )
)
```

O sistema FF

- O sistema **FF** é baseado na idéia da heurística do sistema **HSP**, que estima as distâncias para o estado objetivo pelo comprimento de uma solução aproximada para uma tarefa de planejamento relaxada;
- A heurística do sistema **FF** emprega o sistema **Graphplan** para resolver a tarefa relaxada;
- O algoritmo de busca empregado pelo **FF** utiliza a técnica de busca *Hill-Climbing* reforçado, combinando busca local e sistemática.

Notação utilizada pelo FF

- Um estado S é um conjunto finito de átomos;
- Representação **STRIPS** de ações;
- Uma ação **STRIPS** é uma tripla $o = (pre(o), add(o), del(o))$, onde $pre(o)$ são as pré-condições de o, $add(o)$ é a lista de efeitos adicionados por o, e $del(o)$ é a lista de efeitos removidos por o;
- O resultado da aplicação de uma ação o em um estado S é definido como:

$$Result(S, \langle o \rangle) = S \cup add(o) \backslash del(o) \quad se \quad pre(o) \subseteq S$$

Notação utilizada pelo FF

- O resultado da aplicação de uma seqüência de mais de uma ação em um estado S é recursivamente definido como:

$$Result(S, \langle o_1, \ldots, o_n \rangle) = Result(Result(S, \langle o_1, \ldots, o_{n-1} \rangle), \langle o_n \rangle)$$

- Uma tarefa de planejamento $P = (O, I, G)$ é uma tripla onde O é um conjunto de ações, e I (estado inicial) e G (estado objetivo) são conjuntos de átomos;

- Dada uma tarefa de planejamento $P = (O, I, G)$, a tarefa relaxada P' obtida a partir de P é definida como $P' = (O', I, G)$, onde O' é dado por:

$$O' = \{(pre(o), add(o), \emptyset) | (pre(o), add(o), del(o)) \in O\}$$

Notação utilizada pelo FF

- Dada uma tarefa de planejamento $P = (O, I, G)$, um plano é uma seqüência $A = \langle o_1, \ldots, o_n \rangle$ de ações de O que resolvem a tarefa de planejamento, ou seja, para o qual $G \subseteq Result(I, A)$;

- Uma seqüência de ações é dita um plano relaxado de uma tarefa de planejamento P se e somente se ela resolve a tarefa relaxada P' obtida a partir de P;

O algoritmo de busca *Hill-Climbing* reforçado

Função Hill-Climbing-reforçado (O, I, G)
Entradas: uma descrição de um domínio O, uma conjunção de proposições do estado inicial I, e uma conjunção de proposições do estado objetivo G
Saída: falha ou um plano solução para o problema de planejamento

1. Inicializa o plano solução como um plano vazio;
2. $S \longleftarrow I$
3. Enquanto $h(S) \neq 0$, fazer
4. efetuar uma busca em largura por um estado S‘ com $h(S\text{‘}) < h(S)$
5. Se não encontrou um estado S‘, encerrar com falha
6. adicionar as ações no caminho até S‘ no plano solução
7. $S \longleftarrow S\text{‘}$
8. Fim fazer
9. Encerrar com o plano solução

A função heurística do sistema HSP

- A heurística do sistema **HSP** é uma aproximação grosseira baseada na computação dos seguintes pesos:

$$peso_S(f) = \begin{cases} 0 & se \quad f \in S \\ i & se \quad [min_{o \in O, f \in add(o)} \sum_{p \in pre(o)} peso(p)] = i - 1 \\ \infty & senão \end{cases} \tag{1}$$

- O sistema **HSP** assume que cada átomo é satisfeito independentemente dos demais, de forma que o peso total para a satisfação de um conjunto de átomos, como as pré-condições de uma ação, é computado como a soma dos pesos individuais para satisfazer cada um dos átomos do conjunto;

$$h(S) := peso_S(G) = \sum_{g \in G} peso_S(g) \tag{2}$$

O sistema Graphplan para o cálculo da heurística

- Para corrigir os desvios na heurística do sistema **HSP**, que ignora interações positivas entre duas ações, a heurística do sistema **FF** emprega o sistema **Graphplan** para a resolver a tarefa relaxada;

- A vantagem no uso do **Graphplan** sobre o **HSP** é que a busca pela solução para a tarefa relaxada é regressiva, ou seja, inicia no estado objetivo e escolhe ações que adicionam os átomos do estado objetivo;

O sistema Graphplan para o cálculo da heurística

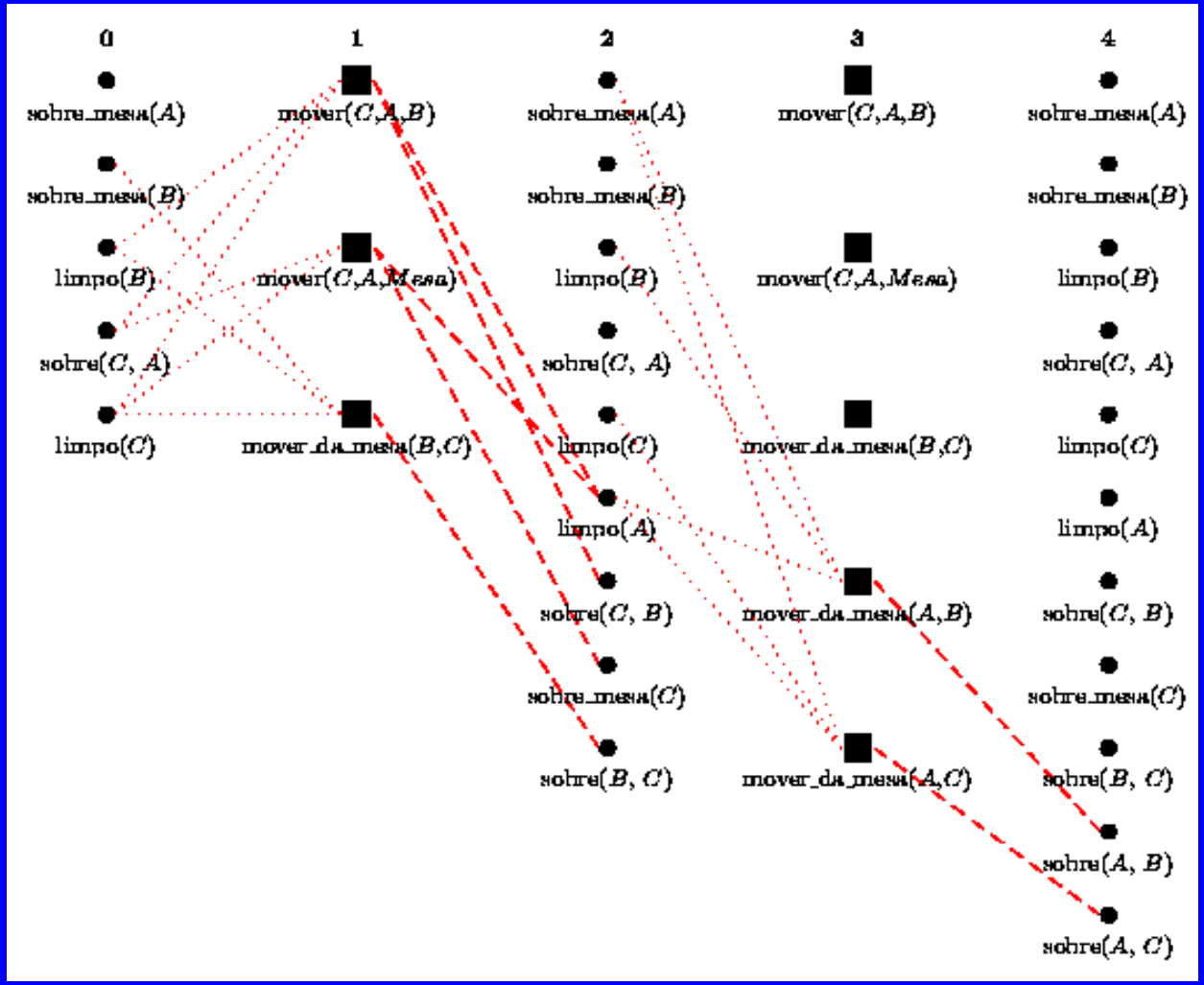
1. O grafo de planejamento para a tarefa de planejamento relaxada é construido em tempo polinomial;

2. Por não possuir proposições negadas, o grafo não contém relações de exclusão mútua entre pares de proposições ou pares de ações;

3. A busca por uma solução para esse grafo não terá pontos de escolha nem $backtracking$;

4. A busca por uma solução para a tarefa de planejamento relaxada tem complexidade polinomial.

Grafo de planejamento de uma tarefa relaxada



O algoritmo de construção do grafo de planejamento

Função constrói-grafo (O, S, G)
Entradas: uma descrição de um domínio O, uma conjunção de proposições de um estado S, e uma conjunção de proposições do estado objetivo G
Saída: falha ou um grafo de planejamento para o problema relaxado

1. $P_0 \longleftarrow S$
2. $t \longleftarrow 0$
3. Enquanto $G \not\subseteq P_t$, fazer
4. $\quad A_t \longleftarrow \{a \in O | pre(a) \subseteq P_t\}$
5. $\quad P_{t+1} \longleftarrow P_t \cup \bigcup_{a \in A_t} eff(a)^+$
6. $\quad$ Se $P_{t+1} = P_t$, encerrar com falha
7. $\quad t \longleftarrow t + 1$
8. Fim fazer
9. $UltimoNivel \longleftarrow t$
10. Encerrar com o grafo de planejamento P

O algoritmo de busca por um plano relaxado

Função busca-plano-relaxado (P, G)
Entradas: um grafo de planejamento P, e uma conjunção de proposições do estado objetivo G
Saída: falha ou um plano para o problema relaxado

1. Para $t = 1$ até $UltimoNivel$, fazer
2. $\quad G_t \longleftarrow \{g \in G | nivel(g) = t\}$
3. Fim fazer
4. Para $t = UltimoNivel$ até 1, fazer
5. $\quad$ Para todo $g \in G_t$, fazer
6. $\quad\quad$ Selecionar a, $nivel(a) = t - 1, g \in eff(a)^+$
7. $\quad\quad$ Para todo $p \in Pre(a)$, fazer
8. $\quad\quad\quad G_{nivel(p)} \longleftarrow G_{nivel(p)} \cup \{p\}$
9. $\quad\quad$ Fim fazer
10. $\quad$ Fim fazer
11. Fim fazer

Cálculo da heurística do FF

- Seja uma solução $\langle o_0, \ldots, o_{m-1} \rangle$ de um sub-problema relaxado (O', S, G), a partir de um estado S da árvore de busca, onde o_i é o conjunto de ações escolhidas em paralelo no nível i do grafo de planejamento, e m é o primeiro nível de proposições contendo todos os átomos do estado objetivo;

- O resultado da função heurística do sistema **FF** é uma estimativa do comprimento da linearização dessa solução para a tarefa de planejamento relaxada, computada conforme a seguinte expressão:

$$h(S) := \sum_{i=0,\cdots,m-1} |o_i| \tag{3}$$

Otimizações

1. Sempre que existir uma ação do tipo $NO - OP$ que adicione uma proposição p, essa ação é preferida sobre a escolha de uma ação real, o que garante a minimalidade da solução encontrada;

2. Na escolha da uma ação que adiciona um sub-objetivo p é sempre preferida aquela que tiver o menor número de pré-condições, o que tende a minimizar o número de proposições a serem satisfeitas no nível imediatamente anterior do grafo de planejamento;

Planejamento com restrições numéricas

- Sejam um conjunto P de proposição e um conjunto V de variáveis numéricas;
- Um estado do mundo é descrito por $s = (p(s), v(s))$, onde $p(s) \in P$ é um subconjunto de P, e $v(s) = (v^1(s), \ldots, v^n(s)) \in Q^n$ é um vetor de valores para cada uma das variáveis numéricas;
- Uma restrição numérica é uma tripla $(exp, comp, exp')$, onde exp e exp' são $expressões$, e $comp \in \{<, \leq, =, \geq, >\}$ é um operador de comparação;
- Um efeito numérico é uma tripla (v_i, atr, exp), onde $v_i \in V$ é uma variável numérica, $atr \in \{:=, + =, - =, * =, / =\}$ é um operador de atribuição, e exp é uma expressão;

Exemplo de descrição de ações - A linguagem PDDL

```
( :action turn_to
  :parameters ( ?s - satellite ?d_new - direction ?d_prev - direction )
  :precondition ( and ( pointing ?s ?d_prev )
    ( not ( = ?d_new ?d_prev ) )
    ( >= ( fuel ?s ) ( slew_time ?d_new ?d_prev ) ) )
  :effect ( and ( pointing ?s ?d_new ) ( not ( pointing ?s ?d_prev ) )
    ( decrease ( fuel ?s ) ( slew_time ?d_new ?d_prev ) )
    ( increase ( fuel-used ) ( slew_time ?d_new ?d_prev ) ) )
)

( :action take_image
  :parameters ( ?s - satellite ?d - direction ?i - instrument ?m - mode )
  :precondition ( and ( calibrated ?i ) ( on_board ?i ?s ) ( supports ?i ?m )
    ( power_on ?i ) ( pointing ?s ?d )
    ( >= ( data_capacity ?s ) ( data ?d ?m ) ) )
  :effect ( and ( have_image ?d ?m )
    ( decrease ( data_capacity ?s ) ( data ?d ?m ) )
    ( increase ( data-stored ) ( data ?d ?m ) ) )
)
```

O sistema FF-Métrico

- Estende a função heurística do **FF**, introduzindo restrições e efeitos numéricos;
- A implementação da função heurística considera a seguinte linguagem restrita:

$$(\{(v_i, comp, c) | v_i \in V, comp \in \{\geq, >\}, c \in Q\},$$
$$\{+=, -=\},$$
$$\{c | c \in Q, c > 0\})$$

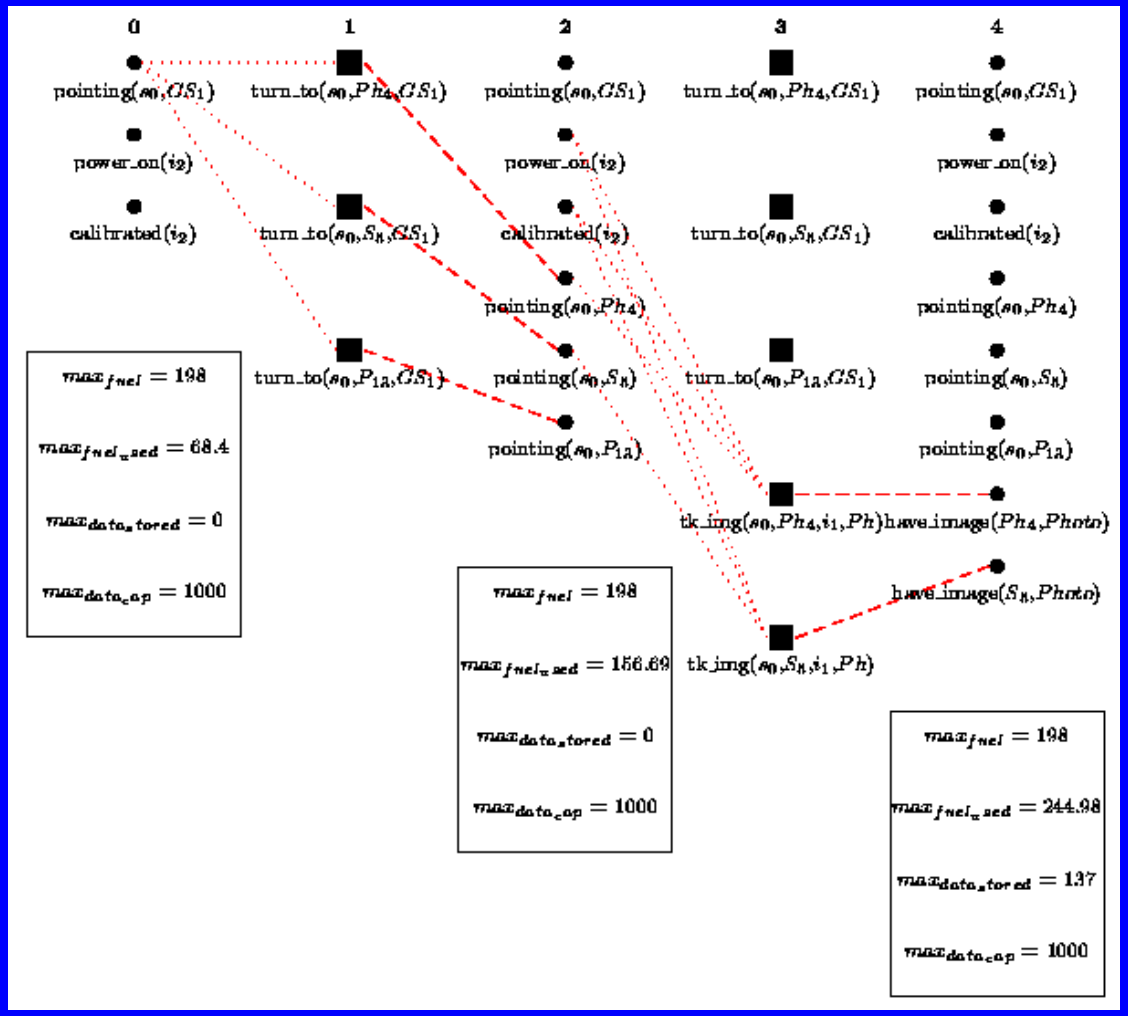
O sistema FF-Métrico

- A função heurística do **FF** é monotônica no número de proposições, pois ignora as listas de remoção;
- A função heurística para o **FF**-métrico estende o conceito de monotonicidade para as variáveis numéricas: os efeitos que decrementam as variáveis numéricas são ignorados;
- Como a linguagem numérica utilizada é restrita, a monotonicidade nas restrições é garantida pela existência apenas dos operadores $\geq$ e $>$;

O sistema FF-Métrico

- Em adição aos níveis de proposições e ações, são propagados valores max_t^i para cada variável i através dos níveis t do grafo, que denotam o valor máximo que a variável v_i pode ter após aplicar t níveis de ações relaxadas;
- max_0 é o valor de uma variável no estado s;
- O estado objetivo é alcançado em um nível t quando os objetivos proposicionais estão contidos em P_t, e quando todas as restrições numéricas são satisfeitas de acordo com os valores de max_t^i;
- Um ponto fixo é alcançado quando um novo nível não adiciona novas proposições ao grafo, ou quando todos os valores de max_t^i são maiores que os valores requeridos por todas as restrições do estado objetivo;

Grafo de planejamento de uma tarefa relaxada



O algoritmo de construção do grafo de planejamento

Função constroi-grafo-metrico (O, S, G)
Entradas: uma descrição de um domínio O, uma conjunção de proposições de um estado S, e uma conjunção de proposições do estado objetivo G
Saída: falha ou um grafo de planejamento para o problema relaxado

1. $P_0 \longleftarrow S$, $\forall v^i | max_0^i \longleftarrow v^i(s)$
2. $t \longleftarrow 0$
3. Enquanto $p(G) \notin P_t$ ou $(v^i, \geq [>], c) \in v(G), max_t^i \not\geq [\not>]c$, fazer
4. $\quad A_t \longleftarrow \{a \in O | pre(a) \subseteq P_t, \forall (v^i, \geq [>], c) \in v(pre(a)) | max_t^i \geq [>]c\}$
5. $\quad P_{t+1} \longleftarrow P_t \cup \bigcup_{a \in A_t} P(eff(a)^+)$
6. $\quad \forall v^i max_{t+1}^i \longleftarrow max_t^i + \sum_{a \in A_t, (v^i, +=, c) \in v(eff(a))} c$
7. $\quad$ Se $P_{t+1} = P_t$ e $\forall v^i | max_{t+1}^i = max_t^i$ ou $max_{t+1}^i > maxneed^i$, encerrar com falha
8. $\quad t \longleftarrow t + 1$
9. Fim fazer
10. $UltimoNivel \longleftarrow t$
11. Encerrar com o grafo de planejamento P

Solução para o grafo da tarefa relaxada

- O nível do grafo em que um objetivo numérico é realizado é o menor t tal que $max_t^i \geq [>]c$ é satisfeito;
- Assim como com as pré-condições representadas por sentenças lógicas, as pré-condições representadas por restrições numéricas das ações escolhidas são inseridas no conjunto de sub-objetivos numéricos;
- Um objetivo numérico $(v_i, \geq [>]\}, c)$ em um nível t é atingido pela escolha de ações no nível imediatamente anterior do grafo que incrementam v_i de c', e subtraindo c' de c, até que o objetivo seja atingido um nível antes;

O algoritmo de busca por um plano relaxado

Função busca-plano-metrico-relaxado (P, G)
Entradas: um grafo de planejamento P, e uma conjunção de proposições do estado objetivo G
Saída: falha ou um plano para o problema relaxado

1. Para $t = 1$ até $UltimoNivel$, fazer
2. $\quad p(G_t) \longleftarrow \{g \in G | nivel(g) = t\}$
3. $\quad v(G_t) \longleftarrow \{(v^i, \geq [>], c) \in v(G) | nivel(v^i, \geq [>], c) = t\}$
4. Fim fazer
5. Para $t = UltimoNivel$ até 1, fazer
6. $\quad$ Para todo $g \in p(G_t)$, fazer
7. $\quad\quad$ Selecionar a, $nivel(a) = t - 1, g \in p(eff(a)^+)$
8. $\quad\quad$ Para todo $p \in p(Pre(a)), (v^i, \geq [>], c) \in v(Pre(a))$, fazer
9. $\quad\quad\quad p(G_{nivel(p)}) \longleftarrow p(G_{nivel(p)}) \cup \{p\}$
10. $\quad\quad\quad v(G_{nivel(v^i, \geq [>], c)}) \longleftarrow v(G_{nivel(v^i, \geq [>], c)}) \cup \{(v^i, \geq [>], c)\}$
11. $\quad\quad$ Fim fazer
12. $\quad$ Fim fazer

O algoritmo de busca por um plano relaxado

13. Para todo $(v^i, \geq [>], c) \in v(G_t)$, fazer
14. Enquanto $max^i_{t-1} \not\geq [\not>]c$, fazer
15. Selecionar a, $nivel(a) = t-1, (v^i, +=, c') \in v(eff(a))$
16. $c \longleftarrow c - c'$
17. Fim fazer
18. $v(G_{t-1}) \longleftarrow v(G_{t-1}) \cup \{(v^i, \geq [>], c)\}$
19. Fim fazer
20. Fim fazer