

Planejamento Hierárquico sob Incerteza

Ricardo Guimarães Herrmann
herrmann@ime.usp.br

Orientadora: Profa. Dra. Leliane Nunes de Barros
Instituto de Matemática e Estatística
Universidade de São Paulo

9/5/2007

Motivação

Diferentes formas de Planejamento:

- Planejamento Hierárquico (HTN) permite planejamento para aplicações práticas envolvendo milhares de ações
- Planejamento Não-Determinístico permite lidar com incerteza sobre efeitos de ações

Motivação

Diferentes formas de Planejamento:

- Planejamento Hierárquico (HTN) permite planejamento para aplicações práticas envolvendo milhares de ações
- Planejamento Não-Determinístico permite lidar com incerteza sobre efeitos de ações

Sinergia entre as técnicas:

- O uso de HTNs como controle de busca permite maior eficiência ao Planejamento Não-Determinístico
- Planejamento Não-Determinístico dá maior robustez a planos de ações

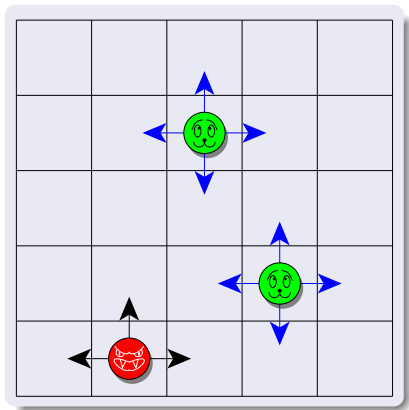
Não existe uma implementação publicamente disponível de um planejador que integre as duas técnicas

Domínio Prototípico: Presas e Predador

Descrição do domínio: Presas e Predador

- Presas e predador movem-se em uma grade $n \times n$
 - Movimentos utilizando as ações **norte**, **sul**, **leste** e **oeste**
 - Predador possui a ação adicional **agarrar**, aplicável quando alcança uma presa
 - Presas possuem a ação adicional **descansar**
-
- Não-determinismo do domínio reside nas ações das presas
 - Possui um grande fator de ramificação

Problema Prototípico: Presas e Predador



- grade 5×5
- 2 presas
- 3 ações possíveis
- $5 \times 5 \times 3 = 75$ possíveis estados sucessores

- 1 **Introdução**
 - Planejamento Clássico e Hierárquico
 - Planejamento Não-Determinístico

- 2 **Planejamento Não-Determinístico Hierárquico**
 - Técnica de “Não-determinização”
 - Planejador ND-SHOP2
 - Implementação
 - Testes Comparativos

1 Introdução

- Planejamento Clássico e Hierárquico
- Planejamento Não-Determinístico

2 Planejamento Não-Determinístico Hierárquico

- Técnica de “Não-determinização”
- Planejador ND-SHOP2
- Implementação
- Testes Comparativos

Planejamento Clássico (Não-Hierárquico)

- Planejamento em IA consiste na busca automatizada de planos de ações que visam alcançar metas pré-estabelecidas em mundos descritos formalmente
- Utiliza como formalismo um Sistema de Transição de Estados

Definição

Um **Sistema de Transição de Estados** é a 3-tupla $\Sigma = \langle \mathcal{S}, \mathcal{A}, \gamma \rangle$:

- \mathcal{S} é um conjunto finito de **estados**
 - \mathcal{A} é um conjunto finito de **ações**
 - $\gamma : \mathcal{S} \times \mathcal{A} \mapsto 2^{\mathcal{S}}$ é uma **função de transição de estados**
-
- Um problema de Planejamento Clássico consiste em encontrar uma sequência de ações que, se executadas a partir de um determinado estado inicial $s_0 \in \mathcal{S}$, leve a um estado meta de $S_g \subseteq \mathcal{S}$

Suposições do Planejamento Clássico

Suposições restritivas feitas pelo Planejamento Clássico

- Conjunto finito de estados
- Ambiente completamente observável
- Ações determinísticas
- Ambiente estático (sem eventos externos)
- Metas de alcançabilidade
- Planos seqüenciais
- Tempo implícito
- Planejamento *offline*

Suposições do Planejamento Clássico

Suposições restritivas feitas pelo Planejamento Clássico

- Conjunto finito de estados
- Ambiente completamente observável
- Ações determinísticas
- Ambiente estático (sem eventos externos)
- Metas de alcançabilidade
- Planos seqüenciais
- Tempo implícito
- Planejamento *offline*

Desafios:

- Problemas de pequeno porte já envolvem quantidades enormes de estados
- NP-completo, mesmo sob essas restrições

Busca Progressiva no Espaço de Estados

FCP (s_0, g, Σ)

$\pi \leftarrow \emptyset; s \leftarrow s_0$

loop

se s satisfaz g **então devolva**(π)

$A \leftarrow \{(s, a) \mid a \text{ é uma ação de } \Sigma \text{ aplicável a } s\}$

se $A = \emptyset$ **então devolva**(*falha*)

não-deterministicamente escolha $(s, a) \in A$

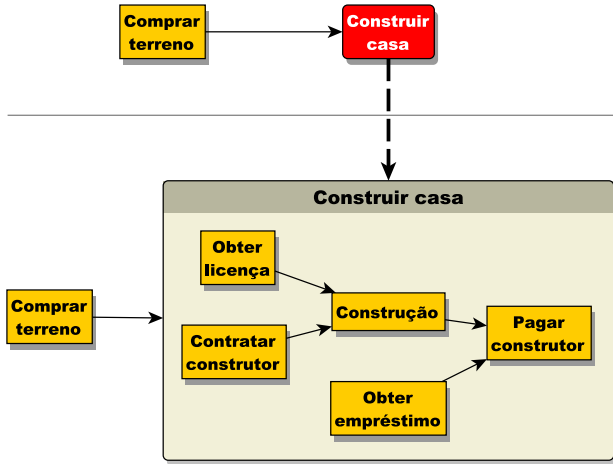
$\pi \leftarrow \pi \cup \{(s, a)\}$

$s \leftarrow \gamma(s, a)$

Planejamento Hierárquico

- Planejamento guiado por Redes Hierárquicas de Tarefas (HTNs)
- Projetista do domínio fornece diferentes **métodos** para decompor **ações compostas** (ou **tarefas**)
- Ações (ou tarefas) **compostas**, abstratas, representam sub-metas de alto nível
- Ações (ou tarefas) **primitivas** representam ações
- O estado corrente deve satisfazer um conjunto de **pré-condições** de um método para que este seja aplicável
- Mais expressivo que o Planejamento Clássico
- Faz uma busca no espaço de redes de tarefas através das diferentes decomposições
- Um problema de Planejamento Hierárquico consiste em, dado um estado inicial s_0 e uma ação composta, decompô-la através de métodos até o nível de ações primitivas, executáveis a partir de s_0

Exemplo de Planejamento Hierárquico



Redes Simples de Tarefas

Redes Simples de Tarefas (STNs):

- Caso especial de HTNs
- Efetua busca progressiva no espaço de estados
- Escolha de ações guiada pela decomposição de redes hierárquicas de tarefas
- Base para a família SHOP de planejadores:
 - SHOP - decomposição em ordem **total** (listas de tarefas)
 - SHOP2 - decomposição em ordem **parcial** (redes de tarefas)
 - JSHOP2 - compilação de planejadores específicos de domínio

Decomposição Hierárquica em Ordem Parcial

PFD ($s, w, \mathcal{D}, \mathcal{M}$)

se $w = \emptyset$ **então devolva**(π)

não-deterministicamente escolha $u \in w$, sem predecessores em w

se t_u é uma ação primitiva **então**

$A \leftarrow \{(a, \sigma) \mid a \text{ é uma ação de } \mathcal{D}, \sigma \text{ é uma substituição tal que } nome(a) = \sigma(t_u) \text{ e } a \text{ é aplicável a } s\}$

se $A = \emptyset$ **então devolva**(falha)

não-deterministicamente escolha $(a, \sigma) \in A$

$\pi \leftarrow \text{PFD}(\gamma(s, a), \sigma(w \setminus \{u\}), \mathcal{D}, \mathcal{M})$

se $\pi = \text{falha}$ **então devolva**(falha)

senão devolva($a.\pi$)

senão

$M \leftarrow \{(m, \sigma) \mid m \text{ é um método de } \mathcal{M}, \sigma \text{ é uma substituição tal que } nome(m) = \sigma(t_u) \text{ e } m \text{ é aplicável a } s\}$

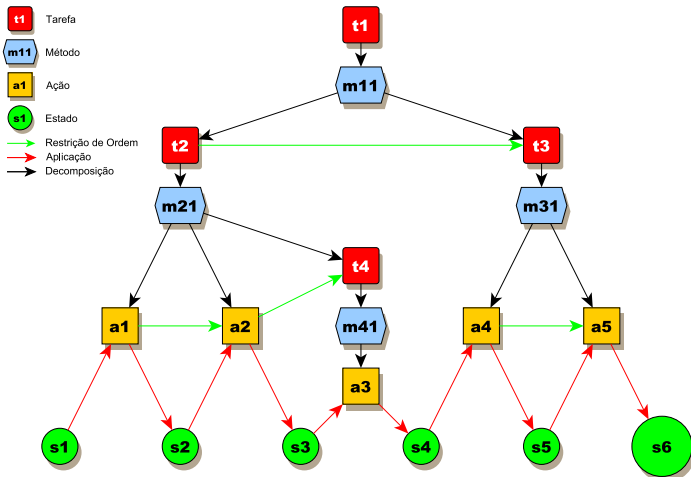
se $M = \emptyset$ **então devolva**(falha)

não-deterministicamente escolha $(m, \sigma) \in M$

não-deterministicamente escolha $w' \in \delta(w, u, m, \sigma)$

devolva PFD($s, w', \mathcal{D}, \mathcal{M}$)

Exemplo de Métodos de Decomposição



1 Introdução

- Planejamento Clássico e Hierárquico
- Planejamento Não-Determinístico

2 Planejamento Não-Determinístico Hierárquico

- Técnica de “Não-determinização”
- Planejador ND-SHOP2
- Implementação
- Testes Comparativos

Planejamento Não-Determinístico

Motivação:

- Ambientes práticos não são tão bem comportados como os do Planejamento Clássico
- Ações podem falhar de modo previsível
- É possível representar a **incerteza** nos efeitos de ações
- Agentes mais informados podem planejar para contingências

Estratégia:

- Planejador deve considerar todos os possíveis caminhos de execução diferentes, para poder encontrar um plano que funcione, apesar do não-determinismo

Problemas:

- Ainda mais difícil que o Planejamento Clássico
- O tamanho do plano condicional obtido pode crescer exponencialmente

Não-determinismo Limitado

Ações podem ter efeitos não-determinísticos, mas os efeitos possíveis podem ser descritos através de uma linguagem de ações

Planejamento Conformante, ou planejamento sem sensores, constrói planos seqüenciais que podem ser executados sem percepção

Planejamento Contingente, ou condicional, lida com o não-determinismo limitado construindo um plano condicional com diferentes ramificações que prevêm as diferentes contingências que possam surgir, ou uma *política*

Planos como Políticas

- Um plano seqüencial não é capaz de representar ações para diferentes evoluções do mundo
- Outro tipo de formalismo mais expressivo para representar planos deve ser utilizado em Planejamento Não-Determinístico

Definição

Uma **política** π :

- Função $\pi : \mathcal{S} \mapsto \mathcal{A}$, diz qual ação executar em um determinado estado
- Geralmente um conjunto de pares de estado e ação (s, a)
- Conjunto de estados em π , $S_\pi = \{s \mid (s, a) \in \pi\}$
- Política determinística: apenas uma ação por estado

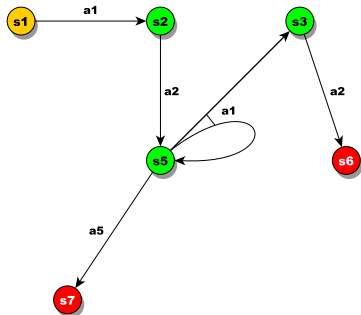
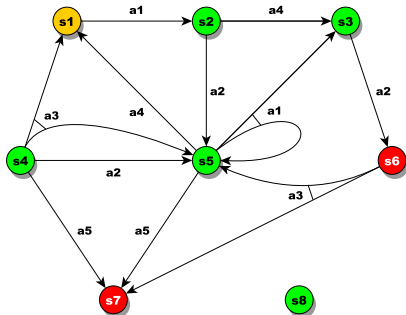
Estrutura de Execução de Políticas

Definição

A **estrutura de execução** Σ_π de uma política π , dado o sistema de transição de estados Σ é:

- $\Sigma_\pi \subseteq \Sigma$ é um grafo direcionado, onde:
 - Nós são estados alcançáveis através de ações de π
 - Arestas são possíveis transições de estado causadas por π
- Caminho em Σ_π de s_1 até $s_2 \iff s_1$ é π -ancestral de s_2 e s_2 é π -descendente de s_1

Estrutura de Execução de Políticas



- Sistema de Transição de Estados Σ

- Estrutura de Execução Σ_π
- s_2 é π -ancestral de s_7

Classes de Soluções

Diferentes níveis de garantia de planos como políticas

Fracos Para cada $s \in S_0$, existe pelo um caminho em Σ_π que alcança um estado meta

Fortes Qualquer estado de S_π alcança a meta sem que estados sejam visitados novamente (sem ciclos)

Fortes Cíclicos Qualquer estado de S_π alcança a meta, porém podem existir ciclos em Σ_π

Classes de Soluções

Diferentes níveis de garantia de planos como políticas

Fracos Para cada $s \in S_0$, existe pelo um caminho em Σ_π que alcança um estado meta

Fortes Qualquer estado de S_π alcança a meta sem que estados sejam visitados novamente (sem ciclos)

Fortes Cíclicos Qualquer estado de S_π alcança a meta, porém podem existir ciclos em Σ_π

- Nível crescente de garantia: Fracos $<$ Fortes Cíclicos $<$ Fortes
- Algumas vezes planos fortes não existem mas soluções fortes cíclicas são satisfatórias
- Como último recurso, caso não exista outra solução melhor, um plano fraco pode vir a funcionar

Algoritmos Tradicionais

- Tradicionalmente, utiliza-se algoritmos de Verificação de Modelos:
 - Técnica de verificação formal
 - Baseada na exploração exaustiva de um sistema de transição de estados Σ
- Utiliza busca em largura regressiva, a partir dos estados meta, até os estados iniciais
- Baseia-se em uma função de **pré-imagem**, que computa um conjunto de predecessores de S

1 Introdução

- Planejamento Clássico e Hierárquico
- Planejamento Não-Determinístico

2 Planejamento Não-Determinístico Hierárquico

- Técnica de “Não-determinização”
- Planejador ND-SHOP2
- Implementação
- Testes Comparativos

Planejamento Não-Determinístico Hierárquico

Suposições do Planejamento Não-Determinístico Hierárquico

- Conjunto finito de estados
- Ambiente completamente observável
- **Ações não-determinísticas**
- Ambiente estático (sem eventos externos)
- **Metas como decomposição de tarefas e alcançabilidade**
- **Planos como políticas**
- Tempo implícito
- Planejamento offline

No contexto deste trabalho, Planejamento Não Determinístico Hierárquico denota **Planejamento Hierárquico com Ações Não-Determinísticas sob Observabilidade Total**

“Não-determinização” de Planejadores Progressivos

Motivação:

- Muito trabalho tem sido feito para melhorar a eficiência de planejadores que efetuam busca progressiva no espaço de estados
- É desejável poder transportar estas técnicas para o Planejamento Não-Determinístico

Características:

- Transformação de modo sistemático (FCP \rightarrow ND-FCP)
- Estende planejadores progressivos para lidar com não-determinismo
- Preserva corretude e completude

Suposições:

- Σ com estados totalmente observáveis

Planejamento Progressivo Determinístico

FCP $(s_0, g, \mathcal{D}, \alpha)$

$\pi \leftarrow \emptyset; s \leftarrow s_0$

loop

se s satisfaz g **então devolva** (π)

$A \leftarrow \{(s, a) \mid a \text{ é uma instância total de um operador em } \mathcal{D}, a \text{ é aplicável a } s, \text{ e } a \in \alpha(s)\}$

se $A = \emptyset$ **então devolva** $(falha)$

não-deterministicamente escolha $(s, a) \in A$

$\pi \leftarrow \pi \cup \{(s, a)\}$

$s \leftarrow \gamma(s, a)$

Planejamento Progressivo Não-Determinístico

ND-FCP $(S_0, g, \mathcal{D}', \alpha')$

$\pi \leftarrow \emptyset; S \leftarrow S_0; \text{resolvidos} \leftarrow \emptyset$

loop

se $S = \emptyset$ **então devolva** (π)

selecione um estado $s \in S$ e remova-o de S

se s satisfaz g **então insira** s em *resolvidos*

senão se $s \notin S_\pi$ **então**

$A \leftarrow \{(s, a) \mid a \text{ é uma instância total de um operador em } \mathcal{D}', a \text{ é aplicável a } s, \text{ e } a \in \alpha'(s)\}$

se $A = \emptyset$ **então devolva** $(falha)$

não-deterministicamente escolha $(s, a) \in A$

$\pi \leftarrow \pi \cup \{(s, a)\}$

$S \leftarrow S \cup \gamma(s, a)$

senão se s não tem π -descendentes em $(S \cup \text{resolvidos}) \setminus S_\pi$

então devolva $(falha)$

Caracterização de Planos de ND-FCP

ND-FCP gera planos **fortes cíclicos**, mas pode ser adaptado para gerar também planos:

- Fortes** , trocando o teste por π -descendentes por *falha* assim que um ciclo for detectado
- Fracos** , removendo, cada vez que um estado meta é gerado, todo estado não-inicial do conjunto S , forçando o planejador a encontrar um caminho a partir de cada estado inicial até uma meta

1 Introdução

- Planejamento Clássico e Hierárquico
- Planejamento Não-Determinístico

2 Planejamento Não-Determinístico Hierárquico

- Técnica de “Não-determinização”
- Planejador ND-SHOP2
- Implementação
- Testes Comparativos

ND-SHOP2

Planejador ND-SHOP2 [Kuter & Nau 2004]

- Técnica de “não-determinização” aplicada a SHOP2
- α representa a escolha de ações geradas pela decomposição hierárquica
- Alguns resultados empíricos mostram condições em que o tempo de execução cresce apenas polinomialmente em relação à versão determinística do problema

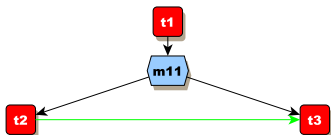
Permite combinar:

- Poder de controle de busca específico de domínio do planejamento hierárquico
- Capacidade de lidar com ações não-determinísticas

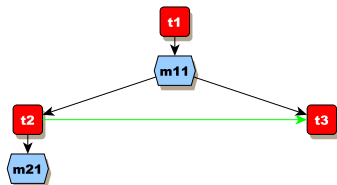
Simulação



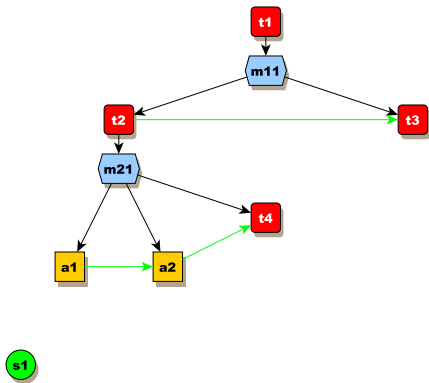
Simulação



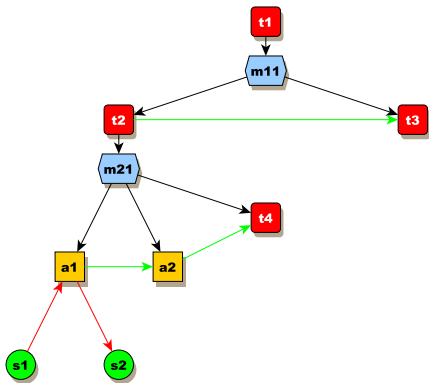
Simulação



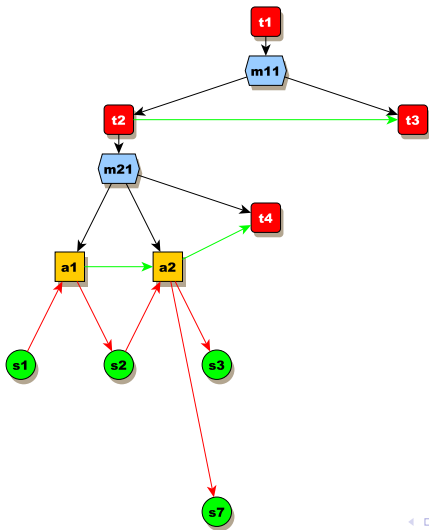
Simulação



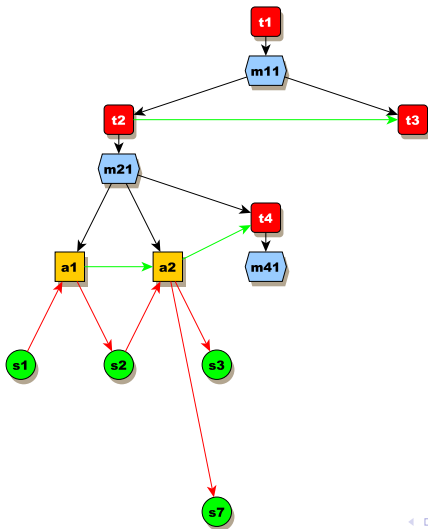
Simulação



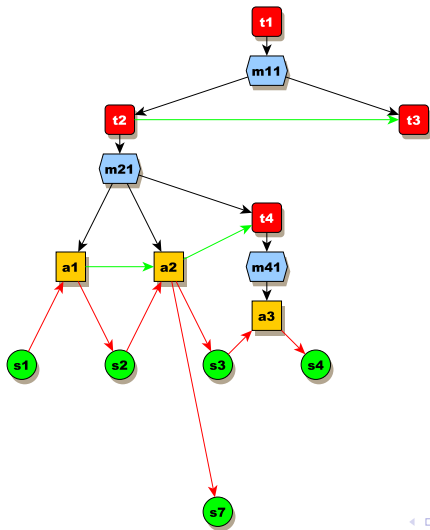
Simulação



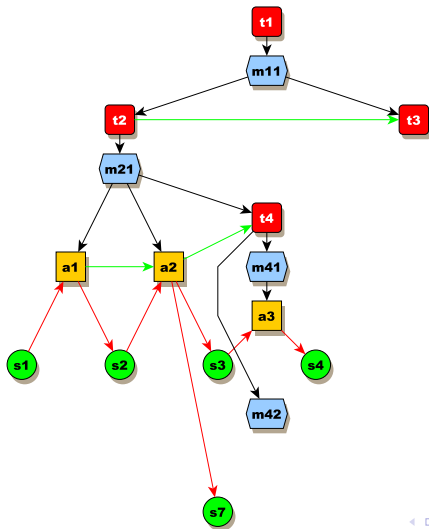
Simulação



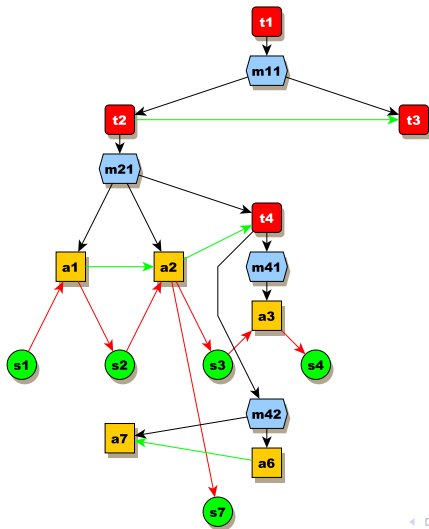
Simulação



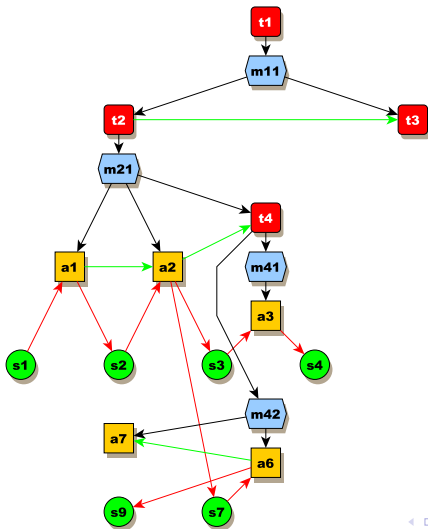
Simulação



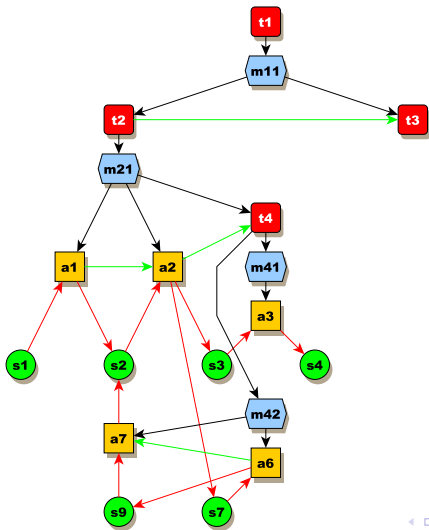
Simulação



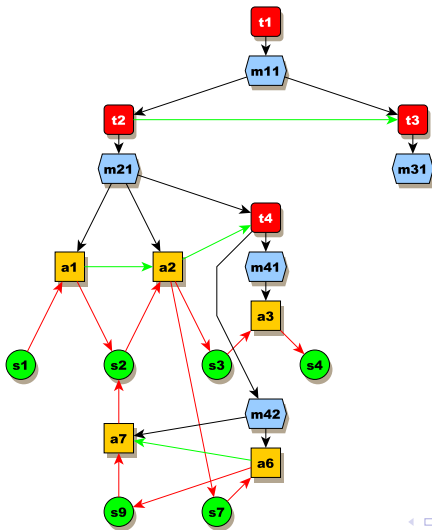
Simulação



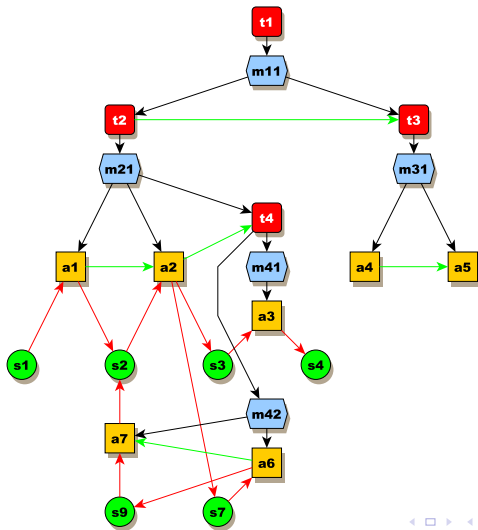
Simulação



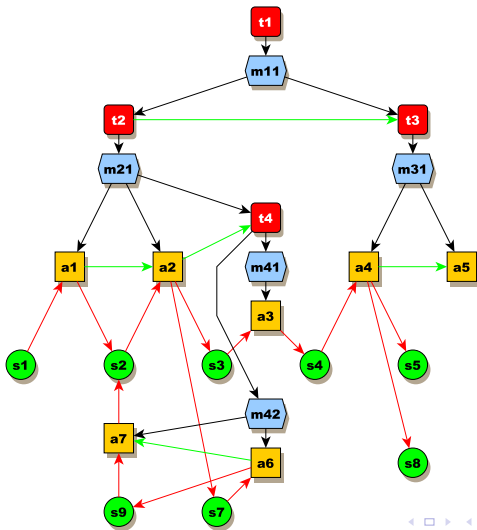
Simulação



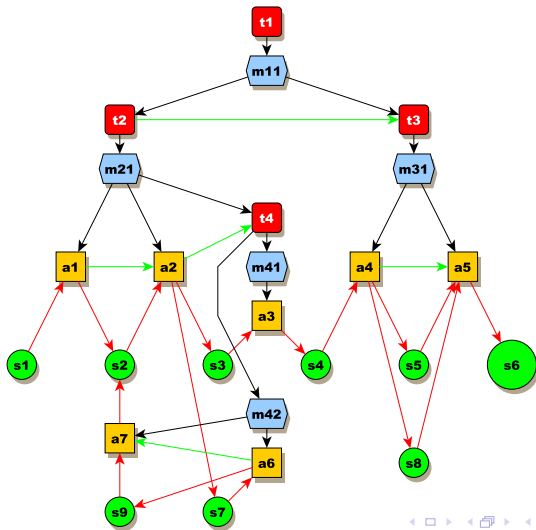
Simulação



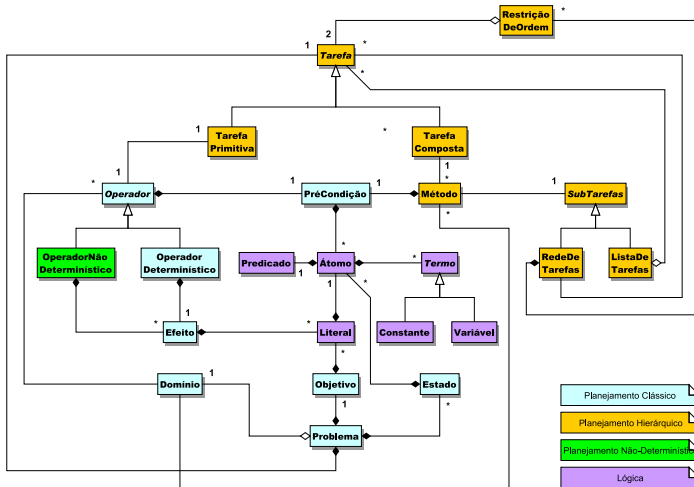
Simulação



Simulação



Modelo da Linguagem para ND-SHOP2



Exemplo Presas e Predador

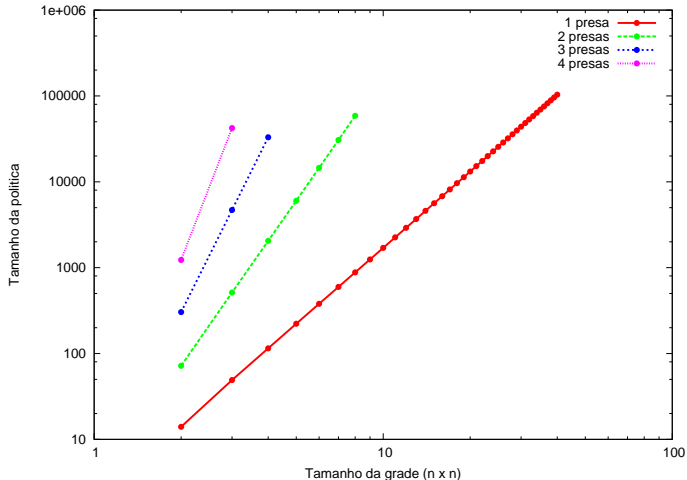
Métodos HTN para obter eficiência:

- Seguir presas primeiro na horizontal e depois na vertical
 - Diminui o número de ações de movimento a considerar
- Escolher uma presa específica, segui-la e agarrá-la, depois considerar outras presas
 - Escolha de ações independe do número total de presas

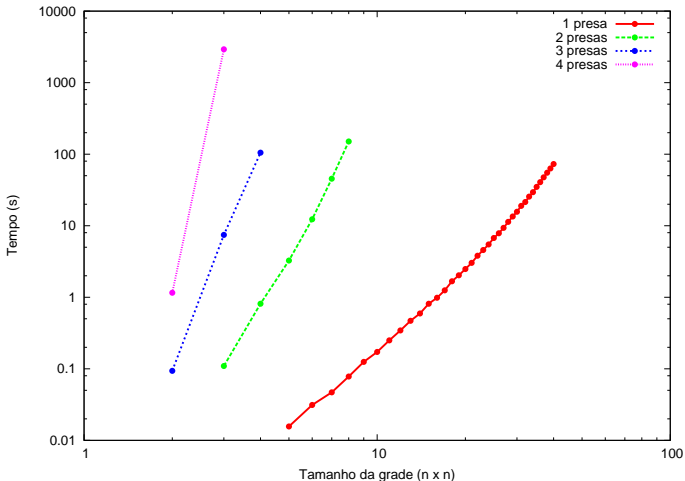
Efeito

Adotar estas duas estratégias combinadas reduz o número de ações aplicáveis a apenas **uma** por estado

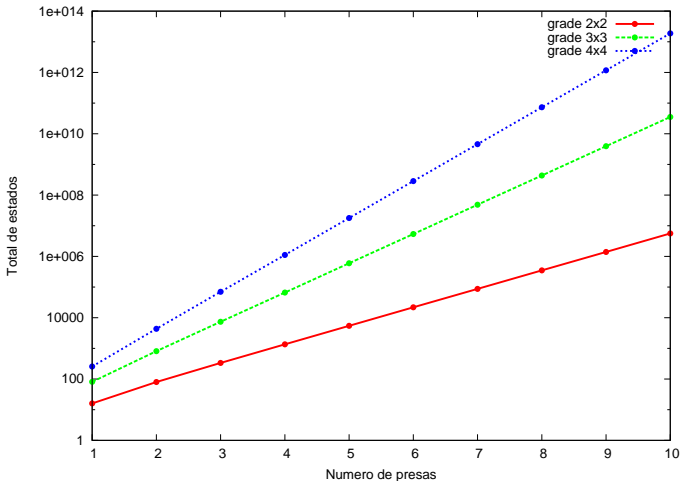
Tamanhos das Políticas Obtidas



Tempo Médio de Execução




Número Total de Estados



Conclusão

- ND-FCP permite que avanços nos planejadores com busca progressiva no espaço de estados possam ser utilizados para domínios não-determinísticos
- ND-SHOP2 permite controle sobre escolhas de ações mas ainda deve explorar todos os estados alcançáveis
- Meios de representação compacta de conjuntos de estados são necessários para a solução de problemas maiores
 - YoYo [Kuter et al. 2005] representa conjuntos de estados e transições através de BDDs

Referências

-  U. Kuter. (PhD thesis, 2006)
Planning under Uncertainty: Moving Forward
-  U. Kuter, D. Nau, M. Pistore, P. Traverso. (2005)
A Hierarchical Task-Network Planner based on Symbolic Model Checking
-  U. Kuter, D. Nau (2004)
Forward-Chaining Planning in Nondeterministic Domains
-  M. Ghallab, D. Nau, P. Traverso. (2004)
Automated Planning.
-  S. Russell, P. Norvig. (2002)
Inteligência Artificial.