Elaine de Castro Freire, Manuela Figueira, Raissa Brasil, Mariana Ramos, Giovanna Bembom, Luiggy Augusto, Maria Flach

Relatório Técnico: Solução Integral para o Planejamento do Mundo dos Blocos de Tamanho Variável via Programação Lógica e Verificação de Modelo

1. Introdução e Fundamentos Conceituais

1.1. Contextualização do Domínio e Limitações do Modelo Clássico

O problema clássico do mundo dos blocos estabeleceu-se como um paradigma canônico no campo do planejamento automatizado. Sua simplicidade abstrata permitiu o estudo claro de princípios de planejamento, como busca em espaço de estados e regressão de objetivos, tipicamente formalizados na arquitetura *Stanford Research Institute Problem Solver* (STRIPS) e implementados em Programação Lógica, como Prolog.[1]

O modelo canônico opera sob a suposição de **homogeneidade física**, tratando todos os blocos como entidades uniformes, e utiliza uma **representação puramente relacional**. O estado do mundo é capturado por um conjunto finito de fatos atômicos, principalmente envolvendo os predicados e .[1] Neste esquema, os "lugares" na mesa são meramente identificadores abstratos sem propriedades espaciais inerentes, como adjacência ou tamanho.[1]

Contudo, essa abstração de alto nível falha fundamentalmente quando confrontada com o domínio do **Mundo dos Blocos de Tamanho Variável (MBTV)**, conforme apresentado nos cenários que incluem restrições espaciais e blocos de larguras distintas. A inadequação é crítica, pois o modelo clássico não possui mecanismos para representar a **pegada física** de um bloco largo (ocupação espacial multiponto) ou para impor **restrições de estabilidade**, o que o levaria a gerar planos fisicamente instáveis, como empilhar um bloco largo sobre um estreito.[1] A solução para o MBTV exige, portanto, uma desconstrução sistemática do

formalismo STRIPS clássico e sua substituição por uma representação de conhecimento

fisicamente fundamentada.[1]

1.2. O Paradigma de Planejamento Integrado: Lógica de Primeira Ordem e NuSMV

A solução proposta para o planejamento no MBTV emprega uma abordagem de **planejamento integrado**, utilizando a Lógica de Primeira Ordem (F.O.L.) implementada em Prolog para a representação rica do conhecimento e o sistema NuSMV, baseado em Verificação de Modelo (*Model Checking*), para o processo de busca exaustiva (Planejamento como Verificação de Modelo - PAMC).[1]

O Prolog é ideal para definir os atributos estáticos e, crucialmente, para hospedar o **raciocínio derivado**. Predicados auxiliares complexos são necessários para calcular a validade física de uma ação sob demanda (por exemplo, a ocupação contígua e a estabilidade). Esses cálculos são consolidados no predicado central can/2.[1] O NuSMV, por sua vez, é empregado para mapear esse domínio complexo de regras para um

sistema de transição de estados finitos. A dinâmica complexa de transição, que em F.O.L. seria representada por axiomas de pré-condição e efeito, é traduzida em condições de TRANS no NuSMV. A busca do plano é então realizada verificando a negação da alcançabilidade do objetivo através da especificação **CTL** .[1] Se esta propriedade for falsa, o NuSMV produz um contraexemplo, que é o plano de ações desejado.[1]

2. Arquitetura de Representação de Conhecimento Estendida em Prolog

Para superar as limitações do modelo clássico, um novo esquema de representação é introduzido, que armazena explicitamente as propriedades físicas e o posicionamento espacial.

2.1. Conhecimento Estático: Atributos Físicos e Espaciais

O primeiro passo é formalizar as propriedades intrínsecas e imutáveis dos blocos, que são ignoradas no modelo clássico. A representação da largura horizontal é essencial para as novas regras de estabilidade e ocupação. Isso é alcançado através do predicado size(Block, Width).[1]

Para os cenários em análise (Situações 1, 2 e 3), os dados estáticos de tamanho inferidos são:

- size(a, 1).
- size(b, 1).
- size(c, 2).
- size(d, 2). [1]

Esta representação fornece os dados brutos necessários para que o planejador raciocine sobre estabilidade e restrições de espaço.

Adicionalmente, o ambiente da mesa é formalizado, deixando de ser uma coleção de identificadores abstratos para se tornar uma **grade linear e discretizada**. A definição formal da grade, através de fatos como table_slot(N) e table_width(W), estabelece um quadro de referência espacial coerente, que é essencial para validar as coordenadas e verificar os limites do movimento.[1] Para os cenários, a mesa é definida por slots de 0 a 6, com uma largura total de 7.[1]

2.2. Modelagem Dinâmica de Estado: O Predicado Unificado

O predicado relacional do modelo clássico é substituído pelo predicado , que unifica a representação da posição vertical e horizontal e remove a ambiguidade semântica anterior.[1]

A nova estrutura de estado permite distinguir explicitamente entre os tipos de suporte:

1. **:** O bloco está sobre a mesa, com sua borda mais à esquerda alinhada com a coordenada horizontal . Esta forma captura a posição absoluta.
2. **:** O bloco está empilhado sobre outro bloco. Sua posição horizontal é relativa ao bloco de suporte.

A adoção do leva a uma **assimetria explícita** no domínio. Ao contrário do STRIPS clássico, que trata o movimento para a mesa e o movimento para outro bloco de forma simétrica, o novo modelo exige lógica separada para cada tipo de destino. Mover para implica verificações complexas de (largura e contiguidade de slots), enquanto mover para implica verificações de e (estabilidade).[1] Essa diferença semântica é crucial e precisa ser tratada

rigorosamente nas pré-condições dos operadores.

2.3. Camada de Conhecimento Derivado para Raciocínio Espacial

Para manter a representação do estado dinâmica (a lista de fatos e) mínima e eficiente, informações espaciais complexas são tratadas como **conhecimento derivado**, calculado por predicados auxiliares sob demanda. Isso evita a redundância e a potencial inconsistência que adviriam da inclusão de fatos como ou na lista de estados.[1]

Os principais predicados auxiliares incluem:

- **:** Calcula a coordenada absoluta mais à esquerda de um bloco. Se o bloco estiver sobre outro, a coordenada é calculada recursivamente a partir da posição da base.[1]
- **:** Determina a lista completa de slots de tabela ocupados por um bloco específico, combinando sua posição absoluta () e sua largura () para gerar o intervalo contíguo $$.[1]
- **:** Predicado fundamental para o planejamento, ele determina se um determinado de tabela está desocupado por qualquer bloco, consultando os de todos os blocos no estado atual.[1]

Esta camada de abstração permite que o planejador formule perguntas espaciais complexas (e.g., "O intervalo de slots está livre?") sem sobrecarregar a representação do estado.[1]

3. Redefinindo Operadores de Planejamento e Impondo Leis Físicas

O operador de planejamento é simplificado para , onde é ou .[1] A inteligência física do domínio é inteiramente codificada nas pré-condições do predicado

.

3.1. Pré-condições de Validação do Bloco e Mobilidade

A restrição mais básica é a **Mobilidade do Bloco**, herdada do modelo clássico: um bloco só

pode ser movido se for o bloco do topo e, portanto, estiver limpo.

- **Condição:** O planejador deve verificar se é um membro do estado atual.[1]

3.2. Restrições de Empilhamento ()

Quando o destino é outro bloco, três condições devem ser satisfeitas para garantir uma operação válida:

1. **Validade Lógica:** O bloco não pode ser empilhado sobre si mesmo ().[1]
2. **Acessibilidade Vertical:** O bloco alvo deve estar livre para receber o novo bloco ().[1]
3. **Restrição de Estabilidade (Física):** Esta é a imposição crucial das leis físicas. Para garantir uma torre estável, o bloco sendo movido () não pode ser mais largo que o bloco de suporte ().
    - **Implementação:** O planejador consulta os fatos estáticos e e impõe a restrição .

A verificação de estabilidade em é um mecanismo poderoso de otimização da busca. Ao impor a restrição física na pré-condição, o planejador realiza o **pruning físico**. Isto significa que ramos de busca que levariam a estados fisicamente impossíveis são eliminados na expansão do nó, em vez de serem explorados, o que aumenta drasticamente a eficiência da busca ao reduzir o fator de ramificação efetivo.[1]

3.3. Restrições de Posicionamento na Mesa ()

Quando o destino é a mesa, o foco muda para garantir espaço horizontal contíguo.

1. **Validade da Coordenada:** deve ser um slot válido ().[1]
2. **Disponibilidade de Espaço Contíguo (Espacial):** Esta é a checagem mais complexa ().
    - O sistema consulta a largura do bloco, determina o intervalo de slots $$ que será ocupado e, em seguida, utiliza o predicado derivado para confirmar que *todos* os slots contidos nesse intervalo estão desocupados. Deve-se também garantir que a coordenada final não exceda o .[1]

Esse conjunto rigoroso de pré-condições garante que cada ação validada pelo planejador seja **logicamente e fisicamente válida** no domínio do MBTV.[1]

3.4. Atualização de Estado (e)

Após a validação por , a transição de estado é aplicada atualizando a lista de fatos dinâmicos:

- **Delete List:** Remove-se o antigo fato e, se o antigo suporte () era um bloco, remove-se o status do antigo destino do novo bloco.[1]
- **Add List:** Adiciona-se o novo fato e . Se o movimento liberou um bloco anterior, o status desse bloco () deve ser adicionado para permitir que ele seja usado como destino ou movido em ações futuras.[1]

4. Análise Situacional e Definição de Estados Lógicos

As três situações apresentadas ilustram os desafios que validam a necessidade do modelo estendido.

4.1. Situação 1: Pilha Simples ()

A Situação 1 demonstra a transição de uma distribuição inicial () para uma torre vertical ().[1] Assume-se que os blocos

e possuem e e possuem .

- **Estado Inicial (Lógica):** Bloco sobre , e isolados na mesa.
  - Exemplo de fatos : .[1]
  - Blocos livres: .
- **Estado Objetivo (Exemplo Lógico de uma Torre, e.g., sobre sobre sobre):**
  - Exemplo de fatos : .
  - O desafio primário aqui é desempilhar para mover , e então construir a nova torre, garantindo que o bloco largo () esteja na base para satisfazer a estabilidade.

4.2. Situação 2: Remanejamento com Estruturas ()

A Situação 2 foca no remanejamento de pilhas pré-existentes e na restrição de **ocupação espacial contígua**.[1]

- **Estado Inicial (Lógica):** Duas pilhas separadas.
  - Configuração: sobre , e sobre , com e na mesa.
- **Desafio:** O movimento de blocos largos, como (size 2), exige uma verificação rigorosa do . Se o planejador propõe , deve-se garantir que os slots e estejam livres e contíguos simultaneamente. A lógica e é ativada em toda a sua complexidade para podar tentativas de colocar um bloco em slots parcialmente ocupados ou que excedam os limites.[1]

4.3. Situação 3: Reempilhagem Complexa (ou)

A Situação 3, que envolve a desmontagem e reempilhagem complexa de duas pilhas [1], fornece o teste de conceito mais claro para a restrição de estabilidade.

- **Estado (Lógica):** sobre , e sobre .
  - Exemplo: .
- **Teste de Estabilidade Definitivo:** Se o planejador, em , tentasse mover o bloco largo () para o topo do bloco estreito () via , a pré-condição falharia imediatamente porque .[1]

Essa falha imediata, ditada pela física codificada no , demonstra a capacidade do formalismo estendido de **podar planos fisicamente impossíveis** na raiz da expansão do estado, validando a eficácia do modelo.[1] O objetivo final é construir uma nova torre, como

(D no topo) ou (C no topo), o que requer sequências de desempilhamento e reposicionamento que respeitem rigorosamente o a cada etapa.[1]

5. Modelagem Comparativa de Conceitos de Planejamento (Tabelas Mandatórias)

As tabelas a seguir comparam como os conceitos centrais do planejamento são abordados no modelo clássico (STRIPS), no modelo estendido (Prolog) e na implementação para verificação de modelo (NuSMV), conforme as exigências das Situações 1, 2 e 3.

5.1. Situação 1 - Tabela de Conceitos

Conceito	STRIPS	Prolog Estendido	Proposta NuSMV	Justificativa
Block Properties	block(X)	size(X,W)	DEFINE size_a :=	Necessário p/ largura e estabilidade [1]
Mobility	clear(B), clear(Target)	can_move(B, Dest, S)	TRANS move->precon ditions	Evita ações inválidas [1]
Target Accessibility	clear(Target)	clear(Target), is_free/2	DEFINE clear b...	Necessário p/ destino livre [1]
Stability	não representado	size(B)<=size(T)	condição booleana size Evita	empilhar maior sobre menor [1]
Spatial Occupancy	não modelado	busy_slots, is_free	vetor/array occupied(i)	Checa espaço contiguo [1]
Logical Validity		Block \== Target		Evita auto-empilha mento [1]

5.2. Situação 2 - Tabela de Conceitos

Conceito	STRIPS	Prolog Estendido	Proposta NuSMV	Justificativa
Block	block(X)	size(X,W)	size	Invariantes

Properties			constantes	estáticos [1]
Mobility	move(B,Pi,Pj)	can_move(B, Dest,S)	TRANS move->clear&.. .	Prune ações inválidas [1]
Target Accessibility	clear(Target)	clear(Target), is_free	clear x macros	Disponibilidad e dinâmica [1]
Stability	não presente	size(B)<=size(T)	guard size<=...	Evita ações inválidas [1]
Spatial Occupancy	ausente	busy_slots	arrays/occupie d	Controle de slots [1]
Logical Validity		Block \== Target		Evita auto-empilha mento [1]

5.3. Situação 3 - Tabela de Conceitos

Conceito	**STRIPS**	**Prolog Estendido**	**Proposta NuSMV**	**Justificativa**
Block Properties	block(X)	size(X,W)	size_* constantes	Necessário p/ restrições físicas [1]
Mobility	move(B,From, To)	can_move/3	TRANS move->precon ditions	Planner usa precondições [1]
Target Accessibility	clear(Target)	clear(Target), is_free	clear macros	Necessário p/ desmontar torre [1]

Stability	não presente	size(B)<=size(T)	guard size<=...	Poda de planos inválidos [1]
Spatial Occupancy	ausente	busy_slots, is_free	arrays/occupied	Reempilhar exige espaço contiguo [1]
Logical Validity		Block \== Target		Evita auto-empilhamento [1]

6. Planejamento como Verificação de Modelo (NuSMV)

O mapeamento do domínio MBTV para o ambiente NuSMV, um verificador de modelo simbólico, exige a conversão das relações de F.O.L. para variáveis de estado com domínios finitos.

6.1. Mapeamento de Estado e Variáveis

A tradução exige um **grounding lógico** completo. Relações como em F.O.L. são convertidas em variáveis discretas, como on_a: {b, c, d, table_0,..., table_6}. O domínio deve enumerar todos os suportes possíveis para garantir a **finitude do espaço de estados**.[1] A discretização da mesa em 7 slots (0 a 6) é o fator que garante que o problema seja computacionalmente tratável via verificação de modelo. Se o espaço da mesa fosse infinito, o NuSMV não poderia ser aplicado.[1]

Variáveis auxiliares, como , são mapeadas para variáveis booleanas clear_a, e a ação é representada por uma variável enumerada move [1]:

Snippet de código

VAR

```
  on_a: {table_0,..., table_6, b, c, d};
  ...
  move: {none, move_c_b, move_a_2,...};
```

6.2. Codificação de Restrições Físicas e Invariantes (DEFINE)

As propriedades estáticas do Prolog são transcritas para constantes imutáveis no NuSMV, garantindo que as regras de estabilidade e ocupação tenham acesso a estes dados [1]:

Snippet de código

```
DEFINE size_a := 1;
DEFINE size_c := 2;
```

As complexas pré-condições de são encapsuladas em macros DEFINE que verificam o estado atual. Por exemplo, a macro que checa o de um bloco pode ser definida como a ausência de qualquer outro bloco sobre ele. A estabilidade é verificada através de uma condição booleana direta nas macros.

6.3. Transição de Estado (TRANS)

O bloco TRANS define as transições válidas no modelo de Kripke. Cada cláusula em TRANS corresponde a um operador e deve impor todas as pré-condições geradas pelo , garantindo que a nova lógica de domínio esteja rigorosamente codificada.[1]

A codificação da estabilidade no TRANS atua como um *guard* que restringe as transições. Por exemplo, uma transição que representa o movimento de um bloco sobre um bloco só é permitida se a condição de estabilidade for satisfeita no estado atual:

Snippet de código

```
TRANS
(move = move_d_on_a) ->
   (on_d!= on_a) &
   clear_d &
   clear_a &
   (size_d <= size_a) &
   next(on_d) = a &
   next(clear_a) = FALSE &...
```

A tradução da restrição de ocupação espacial exige que o TRANS verifique se a matriz de *slots* ocupados do destino está livre antes de atualizar a posição do bloco, confirmando a disponibilidade contígua de *slots*.[1]

6.4. A Estratégia de Planejamento como Verificação de Propriedades

A estratégia de **Planejamento como Verificação de Modelo** formaliza a busca de um plano como a verificação de uma propriedade de alcançabilidade na Lógica Temporal da Árvore de Computação (CTL).[1]

O objetivo (goal) do planejamento é codificado como uma proposição sobre o estado final. A especificação submetida ao NuSMV é a negação da alcançabilidade desse objetivo:

O operador E (Existe um caminho) e F (Futuramente) combinados significam que existe um caminho para o estado objetivo. Ao negar essa expressão, a ferramenta é instruída a procurar um estado onde o objetivo não pode ser alcançado.1 Se o verificador determinar que esta especificação é
falsa (ou seja, o objetivo é, de fato, alcançável), ele produz um **contraexemplo** (ou). Esta sequência de estados e ações, do estado inicial ao estado final, representa o plano de solução.[1]

7. Conclusão, Implicações e Perspectivas Futuras

A solução proposta para o problema do Mundo dos Blocos de Tamanho Variável representa um avanço significativo em relação ao modelo clássico STRIPS. Ao desconstruir a

representação abstrata e integrar atributos físicos estáticos () com um formalismo de estado unificado (), foi possível criar um domínio de planejamento que impõe rigorosamente as leis da estabilidade e da ocupação espacial contígua.

O núcleo da eficácia reside na imposição dessas restrições físicas nas pré-condições do operador . Embora o cálculo dessas pré-condições seja computacionalmente mais custoso do que uma simples consulta de associação de lista no STRIPS clássico, o benefício é a **poda precoce** de ações inválidas. Esta eliminação na raiz da expansão do nó de busca compensa o custo computacional, resultando em uma redução drástica do fator de ramificação efetivo e melhorando a eficiência geral do planejamento.[1] O sistema resultante não apenas gera planos válidos, mas também garante que esses planos sejam fisicamente realizáveis.

Em termos de evolução do modelo, várias extensões podem aumentar seu poder e realismo:

- **Modelagem 3D e Coordenada Z:** O modelo atual opera principalmente em 2D (posição horizontal e ordem de empilhamento). Uma extensão natural exigiria a introdução de um atributo de altura e uma coordenada explícita. O predicado teria que verificar a folga vertical para o manipulador e raciocinar sobre volumes espaciais.[1]
- **Estabilidade Baseada em Centro de Massa:** A regra simplificada pode ser substituída por um cálculo explícito do **centro de massa** sobre a área de suporte. Isso permitiria modelar estruturas que são inerentemente instáveis, mas que ainda se mantêm (ou seja, o centro de massa está dentro da base de suporte).[1]
- **Planejamento Estratégico:** A utilização de especificações CTL/LTL mais avançadas no NuSMV, além da simples alcançabilidade (), pode ser explorada. Isso permitiria verificar propriedades estratégicas, como a existência de uma estratégia *não-perdedora* ou *não-empate* (como visto em domínios de jogos como Tic-Tac-Toe [1]), garantindo planos que são robustos contra incertezas ou ações adversárias em ambientes dinâmicos.

Referências citadas

1. Problema_Blocos_Tamanhos_Variaveis.docx (1).pdf