

Parte II: IA Simbólica ou Good Old-Fashioned Artificial Intelligence (GOFAI).

Prof. Fabiano Araujo Soares, Dr. / FGA 0221 - Inteligência Artificial

Universidade de Brasília

2025

Problemas de Satisfação de Restrições



Você já parou para pensar em quantas escolhas do nosso cotidiano dependem de restringirmos possibilidades até encontrar uma solução viável?

Problemas de Satisfação de Restrições



Você já parou para pensar em quantas escolhas do nosso cotidiano dependem de restringirmos possibilidades até encontrar uma solução viável?

A beleza dos problemas de satisfação de restrições está em transformar desafios complexos em buscas estruturadas por harmonia entre variáveis.

- Muitos problemas complexos em Inteligência Artificial podem ser formulados como Problemas de Satisfação de Condições (CSP).
- CSPs permitem representar o problema em termos de variáveis, domínios e restrições claras, facilitando a análise e solução sistemática.
- O modelo CSP possibilita o uso de técnicas gerais de busca e algoritmos de propagação de restrições, aumentando a eficiência da resolução.
- Aplicações práticas incluem:
 - Coloração de mapas (exemplo clássico: mapa da Austrália)
 - Jogos e quebra-cabeças como Sudoku e o problema das 8 damas
 - Escalonamento de tarefas e alocação de recursos em ambientes industriais e acadêmicos
 - Problemas de configuração, planejamento e design assistido por computador
- A estrutura comum dos CSPs permite o desenvolvimento de heurísticas eficazes que reduzem significativamente o espaço de busca, tornando possível resolver problemas que seriam intratáveis por métodos tradicionais.

Definição Formal de CSP

Um CSP é definido por:

- Um conjunto de variáveis: $X = \{X_1, X_2, \dots, X_n\}$
- Um domínio para cada variável: $D = \{D_1, D_2, \dots, D_n\}$
- Um conjunto de restrições: $C = \{C_1, \dots, C_m\}$

Solução: uma atribuição de valores às variáveis que satisfaz todas as restrições.

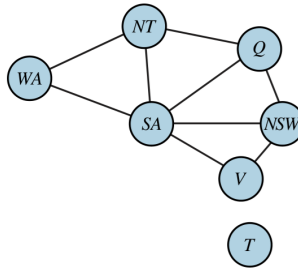
Exemplo: Coloração de Mapas

- Variáveis: regiões do mapa (e.g., estados da Austrália)
- Domínio: conjunto de cores possíveis (vermelho, verde, azul)
- Restrições: regiões vizinhas não podem ter a mesma cor



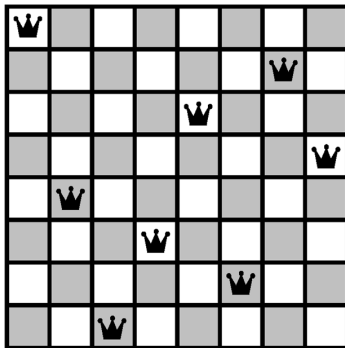
Representação em Grafo

- Variáveis como nós; restrições como arestas entre nós
- Facilita visualização e aplicação de algoritmos de propagação de restrições



Exemplo: Problema das 8 Damas

- Variáveis: colunas do tabuleiro
- Domínio: linhas possíveis
- Restrições: duas damas não podem estar na mesma linha ou diagonal



Exemplo CSP: Linha de Montagem de Carros

Tarefas:

- Instalação do eixo dianteiro (ExD) e traseiro (ExT)
- Instalação das rodas: RDD, RDE, RTD, RTE
- Parafusos das rodas: PRDD, PRDE, PRTD, PRTE
- Calotas das rodas: CDD, CDE, CTD, CTE
- Inspeção final (Ins)

Restrições/Condições:

- Eixos devem ser instalados antes das rodas.
- Rodas antes dos parafusos, parafusos antes das calotas.
- Inspeção somente ao final.
- Instalação dos eixos: 10 min cada, 1 por vez.
- Cada roda: 1 min. Cada jogo de Parafusos: 2 min. Cada Calota: 1 min. Inspeção: 3 min.
- 4 mecânicos, tempo máximo: 30 min.

Modelagem CSP: Variáveis e Domínios

- **Variáveis:** Tempo de início de cada tarefa (T_i), Mecânico responsável (M_i)
- **Domínios:**
 - $T_i \in [0, 30]$
 - $M_i \in \{1, 2, 3, 4\}$
- **Restrições:**
 - Precedência: $T_{ExD} + 10 \leq T_{RDD}, T_{RDE}$; $T_{ExT} + 10 \leq T_{RTD}, T_{RTE}$ etc.
 - Não sobreposição do mesmo mecânico (M_i)
 - Tempo total ≤ 30 min

Resumo do CSP da Linha de Montagem

- Encontrar uma sequência e alocação de tarefas aos mecânicos, respeitando restrições e o tempo máximo.
- Demonstra como CSPs modelam problemas reais e complexos em indústrias.

Tipos de Restrições em CSPs

Em Problemas de Satisfação de Condições, as restrições regulam as combinações de valores possíveis das variáveis.

Tipos:

- Unária
- Binária
- Ternária
- Global
- Obrigatória (dura)
- De preferência (suave)

- Afeta uma única variável.
- Exemplo: No problema de colorir mapas, “ $SA \neq \text{Verde}$ ”, restringe o domínio da variável SA.
- Representa condições de validade de valores individuais.

Restrição Binária

- Afeta duas variáveis simultaneamente.
- Exemplo: No Sudoku, " $A1 \neq B1$ "; na coloração de mapas, " $SA \neq NSW$ " (regiões vizinhas).
- Permite construir grafos de restrições.

Restrição Ternária

- Afeta três variáveis.
- Exemplo: Em um agendamento, “X, Y e Z não podem ocupar o mesmo horário”.
- São menos frequentes, mas importantes em problemas mais complexos.

- Envolvem subconjuntos maiores de variáveis (podendo ser todas).
- Exemplo: “Todos os valores em uma linha de Sudoku devem ser diferentes” (restrição **all-different**).
- Facilitam expressar limitações amplas do problema.

Obrigatória (dura) e de Preferência (suave)

- **Obrigatória (dura):** Deve ser satisfeita obrigatoriamente. Se violada, solução é inválida.
- **Preferência (suave):** Idealmente satisfeita, mas pode ser relaxada se necessário.
Exemplo: “Professor prefere dar aulas pela manhã.”
- CSPs flexíveis permitem soluções parciais, balanceando restrições obrigatórias e preferenciais.

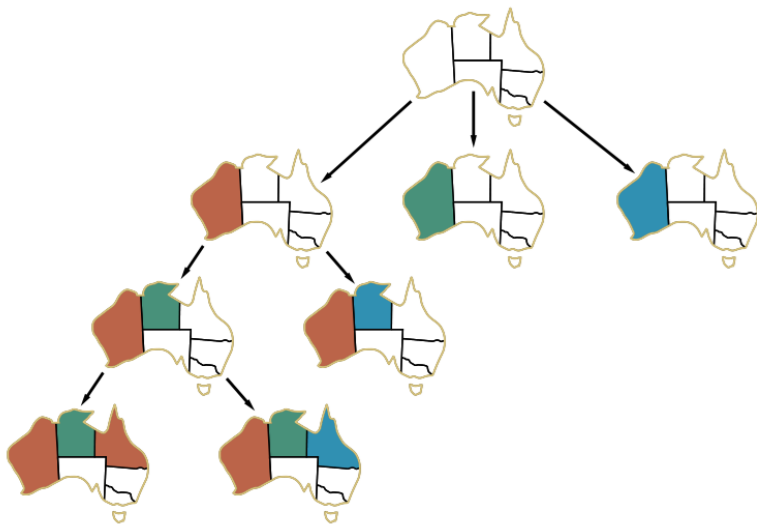
- Backtracking
- Forward checking
- Algoritmo AC-3 de consistência de arco

Heurísticas usadas na busca:

- Menor valor restante (MRV)
- Grau da variável (degree heuristic)
- Valor menos restritivo

- Algoritmo recursivo que tenta construir uma solução atribuindo valores às variáveis uma a uma.
- Ao encontrar conflito, desfaz atribuição (backtrack) e tenta outro valor.
- **Exemplo:** No problema das 8-damas, tenta posicionar cada rainha em uma coluna, e volta atrás se encontrar ataque entre elas.

Exemplo: Backtracking



Forward Checking

- Após cada atribuição, elimina valores conflitantes dos domínios das variáveis ainda não atribuídas.
- Antecipação de falhas: detecta mais cedo quando não há solução possível, podendo a busca.
- **Exemplo:** Ao colorir o mapa da Austrália, se SA recebe “vermelho”, elimina a opção “vermelho” dos vizinhos (WA, NSW, NT), e detecta se algum fica sem cores disponíveis.

Exemplo: Forward Checking





























	<i>WA</i>	<i>NT</i>	<i>Q</i>	<i>NSW</i>	<i>V</i>	<i>SA</i>	<i>T</i>
Initial domains							
After <i>WA</i> =red							
After <i>Q</i> =green							
After <i>V</i> =blue							

Figure 6.7 The progress of a map-coloring search with forward checking. *WA* = red is assigned first; then forward checking deletes red from the domains of the neighboring variables *NT* and *SA*. After *Q* = green is assigned, green is deleted from the domains of *NT*, *SA*, and *NSW*. After *V* = blue is assigned, blue is deleted from the domains of *NSW* and *SA*, leaving *SA* with no legal values.

Algoritmo AC-3 (Consistência de Arco)

- Propaga restrições entre pares de variáveis para reduzir os domínios antes/durante a busca.
- Remove valores dos domínios onde não existe valor compatível na variável associada.
- **Exemplo:** Em Sudoku, se uma célula recebe o valor “5”, o AC-3 garante que “5” seja removido das células na mesma linha, coluna e região, mantendo consistência.

Exemplo AC-3 com exclusão de valores em domínios

Variáveis e Domínios:

- A, B, C
- Domínios: $D(A) = D(B) = D(C) = \{1, 2, 3\}$

Restrições:

- $A > B$
- $B = C$

Passo 1: Inicializamos a fila com arcos:

$$(A, B), (B, A), (B, C), (C, B)$$

Exemplo AC-3 com exclusão de valores em domínios (cont.)

Passo 2: Revisão dos arcos:

- Para arco (A, B) , removemos de $D(A)$ valores que não tem suporte em $D(B)$ para $A > B$.

$$D(B) = \{1, 2, 3\} \implies D(A) \text{ deve ter ao menos um valor maior que um valor em } D(B)$$

Valores removidos de $D(A)$: 1, não existe valor em $D(B)$ maior que 1.

$$D(A) = \{2, 3\}$$

- Para arco (B, A) , removemos de $D(B)$ valores que não tem suporte em $D(A)$ para $A > B$.

$$D(A) = \{2, 3\}$$

$B = 3$ não tem $A > 3$, logo removemos 3 de $D(B)$.

$$D(B) = \{1, 2\}$$

Exemplo AC-3 com exclusão de valores em domínios (cont.)

- Para arco (B, C) , removemos de $D(B)$ valores que não tem suporte em $D(C)$ para $B = C$.

$$D(C) = \{1, 2, 3\}, D(B) = \{1, 2\}$$

Agora $D(C)$ pode ser reduzido a $\{1, 2\}$, pois para $B = 1$ ou 2 , só 1 e 2 são possíveis para C .

$$D(C) = \{1, 2\}$$

- Para arco (C, B) , o domínio de C atual não necessita redução.

Exemplo AC-3 com exclusão de valores em domínios (cont.)

Resumo: Após o processo AC-3,

$$D(A) = \{2, 3\}, \quad D(B) = \{1, 2\}, \quad D(C) = \{1, 2\}$$

Os valores inconsistentes foram removidos antecipadamente.

Maintaining Arc Consistency (MAC)

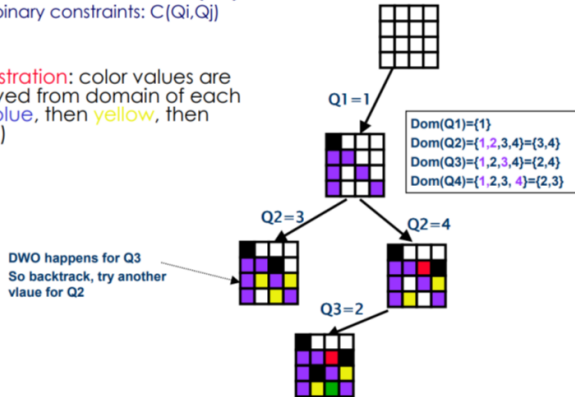
- Depois que uma variável X_i recebe um valor,
- o processo de **INFERÊNCIA** chama **AC-3**,
- mas em vez de uma fila de todos os arcos no CSP, começamos apenas com os arcos (X_j, X_i) para todos os X_j que são variáveis não atribuídas vizinhas de X_i .

Heurísticas de Busca para CSPs

- **MRV (Menor valor restante):** Escolhe primeiro a variável com menos opções disponíveis, aumentando chance de conflito precoce e podando o ramo mais cedo.
- **Grau da variável:** Prioriza variáveis com mais restrições (mais vizinhos), para controlar o espaço de busca.
- **Valor menos restritivo:** Ao atribuir valor, prefere o que menos reduz os domínios das variáveis futuras.
- **Exemplo prático:** No Sudoku, escolha primeiro as células com menos opções possíveis e atribua o número que impacta menos as outras variáveis.

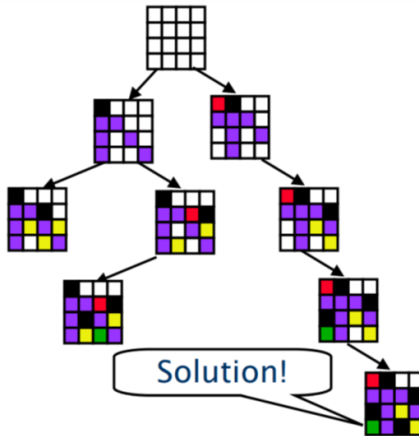
Exemplo Final: 4-Damas

- 4X4 Queens
 - $Q1, Q2, Q3, Q4$ with domain $\{1..4\}$
 - All binary constraints: $C(Q_i, Q_j)$
- FC illustration: color values are removed from domain of each row (blue, then yellow, then green)



Exemplo Final: 4-Damas (cont.)

- 4X4 Queens continue...



- CSPs são fundamentais em IA para problemas estruturados
- Abordam desde tarefas de planejamento até jogos e lógica
- Permitem uso de algoritmos genéricos e heurísticas eficientes

- Russell, S. J., & Norvig, P. (2021). *Artificial Intelligence: A Modern Approach*. 4^a ed.
- Slides oficiais: aima.cs.berkeley.edu
- Outros: materiais complementares em IA.

Obrigado!

E-mail: fabiano-soares@unb.br

LinkedIn: <https://www.linkedin.com/in/fabiano-soares-06b6a821a/>

Site do curso: <https://www.fabiano-soares.eng.br/fga0221-inteligência-artificial>