# Roteiro de Aula – Consistência de Arcos em CSPs (Poole & Mackworth, Cap. 4)

Prof. Rodrigo Silva

## Objetivo da Aula

Compreender o conceito de **consistência de restrições** em problemas de satisfação de restrições (CSPs), introduzir o algoritmo **GAC – Generalized Arc Consistency**, e discutir sua aplicação prática e eficiência.

## 1 Introdução

**Problemas de Satisfação de Restrições (CSPs)** são definidos por um triplo:

$$\langle V, D, C\rangle$$

onde $V$ é o conjunto de variáveis, $D$ seus domínios, e $C$ o conjunto de restrições.

O método *generate and test* é ineficiente, pois repete verificações. O *backtracking search* melhora, mas ainda reavalia inconsistências já conhecidas.
**Ideia central:** eliminar valores inconsistentes dos domínios antes ou durante a busca.

**Exemplo 4.13.** Se $A < B$ e $dom(B) = \{1, 2, 3, 4\}$, então $A = 4$ é inconsistente. Podemos eliminar 4 de $dom(A)$ antes de buscar soluções.

## 2 Redes de Restrições

Um CSP pode ser representado como uma **rede bipartida de restrições**:

- Nós circulares: variáveis ($A$, $B$, $C$);
- Nós retangulares: restrições ($A < B$, $B < C$);
- Arcos: conexões $\langle X, c\rangle$ entre variáveis e restrições.

**Exemplo 4.14.** CSP com $A, B, C \in \{1, 2, 3, 4\}$ e restrições $A < B$, $B < C$. A rede contém quatro arcos:

$$\langle A, A < B\rangle,\ \langle B, A < B\rangle,\ \langle B, B < C\rangle,\ \langle C, B < C\rangle$$

## 3 Consistência de Domínio e de Arco

**Consistência de domínio:** toda atribuição possível de uma variável satisfaz suas restrições unárias.

**Exemplo 4.16.** Para $B \neq 3$ e $dom(B) = \{1, 2, 3, 4\}$, o domínio não é consistente. Removendo 3, torna-se consistente.

**Consistência de arco:** para cada $x \in dom[X]$, deve haver pelo menos uma atribuição às variáveis relacionadas que satisfaça a restrição.

**Exemplo 4.17.** No CSP $A < B$, $B < C$, o arco $\langle A, A < B\rangle$ é inconsistente se $A = 4$, pois não há valor de $B$ que satisfaça $A < B$. Assim, removemos 4 de $dom(A)$.

## 4 O Algoritmo GAC

**Objetivo:** tornar a rede de restrições arc-consistente.

Listing 1: Pseudocódigo do algoritmo GAC (Poole & Mackworth, 2023)

```
procedure GAC(Vs, dom, Cs, to_do):
    while to_do not empty do
        select and remove (X, c) from to_do
        let {Y1, ... , Yk} = scope(c) \ {X}
        ND = { x $\in$ dom[X] , exists y1 ... ,yk $\in$ dom[Y1] ... dom[Yk]
                such that c(X=x,Y1=y1,...,Yk=yk) holds }
        if ND not equal dom[X] then
            dom[X] = ND
            to_do = to_do U {(Z, c_p) , {X,Z} $\subset$ scope(c_p, c_p != c, Z != X)}
    return dom
```

**Interpretação:**

- Iterativamente remove valores inconsistentes dos domínios.
- Se o domínio de $X$ é reduzido, outros arcos que dependem de $X$ são reavaliados.

**Exemplo 4.18.** Para $A < B$ e $B < C$, inicialmente:

$$dom(A) = dom(B) = dom(C) = \{1, 2, 3, 4\}$$

Após aplicar o GAC:

$$dom(A) = \{1, 2\}, \quad dom(B) = \{2, 3\}, \quad dom(C) = \{3, 4\}$$

## 5 Casos de Término e Interpretação

Ao final do algoritmo:

1. Algum domínio vazio $\Rightarrow$ sem solução.
2. Todos os domínios unitários $\Rightarrow$ solução única.
3. Domínios múltiplos não-vazios $\Rightarrow$ problema reduzido, mas requer busca.

**Exemplo 4.19.** Rede de agendamento: $A = 4, B = 2, C = 3, D = 4, E = 1$. **Conclusão:** todas as variáveis possuem domínios únicos; o CSP tem solução única.

**Exemplo 4.20.** CSP com $A = B$, $B = C$, $A \neq C$. Mesmo sendo arc-consistente, não há solução global.

## 6 Complexidade e Extensões

Para restrições binárias:

$$O(c\, d^3)$$

onde $c$ é o número de restrições e $d$ o tamanho médio dos domínios.

Espaço: $O(n\, d)$, com $n$ variáveis.

Extensões:

- Domínios infinitos (restrições intensionais);
- **Path consistency**: analisa trios de variáveis;
- **k-consistency**: generalização para $k$ variáveis.

## 7 Atividades

1. Construa o grafo de restrições para o CSP $X < Y$, $Y < Z$ e identifique os arcos inconsistentes.
2. Aplique o algoritmo GAC passo a passo sobre $A < B$, $B < C$ com domínios $\{1, 2, 3, 4\}$.
3. Implemente em Python uma versão simples do GAC e teste-a em pequenos CSPs.
4. Explique por que o GAC não garante a existência de solução, mesmo quando todos os arcos são consistentes.

## 8 Discussão Final

- A consistência de arcos é uma forma de **propagação de restrições**.
- Reduz o espaço de busca de forma sistemática, sem gerar atribuições completas.
- É frequentemente usada em conjunto com **busca com retrocesso**.

**Leitura Recomendada:**


- Poole, D.L. & Mackworth, A.K. (2023). *Artificial Intelligence: Foundations of Computational Agents*, 3rd ed., Cap. 4.
- Russell, S. & Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*, 4ª ed., Seção 6.2.