O Problema do Caminho dos Povos Originários

Professor Doutor Weslen Schiavon

Sumário

- Introdução ao Problema
- Formalização
- Algoritmos
- Complexidade do Problema

O Caminho dos Povos

- 📍 **Território:** na fronteira Brasil–Venezuela (estados de Roraima e Amazonas). A **Terra Indígena Yanomami** tem **9.664.975 ha (≈ 96.650 km²)** — maior que Portugal.
- 👥 **População: ~31.007 pessoas** no Brasil (estimativa de 2025 em estudo recente).
- 🗣️ **Línguas/subgrupos: Yanomae, Yanõmami, Sanima e Ninam** (mesma família linguística).
- 🌿 **Cultura e saberes: yãkoana** (rapé xamânico), **jenipapo** (pintura corporal), **tucumã** (alimento/óleo), **fibra de arumã** (cestaria).
- ⚠️ **Desafios atuais:** impactos do **garimpo ilegal** (malária, contaminação por mercúrio); governo relata **redução do garimpo** e **reabertura de polos de saúde** em 2025, mas **a contaminação persiste** em estudos independentes.

O Caminho dos Povos

- Entre os Yanomami, algumas aldeias estão **distribuídas pela floresta, conectadas por trilhas, rios e caminhos** usados para trocas de alimentos, ferramentas e informações.
- Um **mensageiro** precisa visitar todas as aldeias para **levar mensagens e pequenas oferendas** (como Yãkoana, Tucumã, Jenipapo, Fibra de Arumã), retornando à aldeia de origem após completar o trajeto.

O Caminho dos Povos

O Caminho dos Povos

- O mensageiro deve **visitar todas as 7 aldeias Yanomami** (Maturacá, Ariabü, Maiá, Paduarí, Nazaré, Watoriki, Haximu).
- Deve **passar por cada aldeia apenas uma vez** e **retornar à origem**.
- O objetivo é **minimizar a distância total perco**
- As **distâncias entre aldeias** são conhecidas (tabela fornecida).

O Caminho dos Povos

De/Para	Maturacá	Ariabú	Maiá	Paduarí	Nazaré	Watoriki	Haximu
Maturacá	-	9	15	21	19	13	10
Ariabú	9	-	11	17	20	15	12
Maiá	15	11	-	8	14	16	18
Paduarí	21	17	8	-	9	19	20
Nazaré	19	20	14	9	-	13	17
Watoriki	13	15	16	19	13	-	11
Haximu	10	12	18	20	17	11	-

Formalização

- Temos um conjunto de **aldeias** V={1, 2, ..., n}V = {1, 2, ..., n}V={1, 2, ..., n}.
- Cada par de aldeias (i,j) tem uma **distância** c_{ij}
- Queremos encontrar a **rota de menor distância total** que:
  - Visite **todas as aldeias uma única vez**;
  - **Retorne à aldeia de origem**.

- **Formulação:**

$$\min_{\pi} \sum_{k=1}^{n} c_{\pi(k),\pi(k+1)}, \quad \pi(n+1) = \pi(1)$$

O Caminho dos Povos

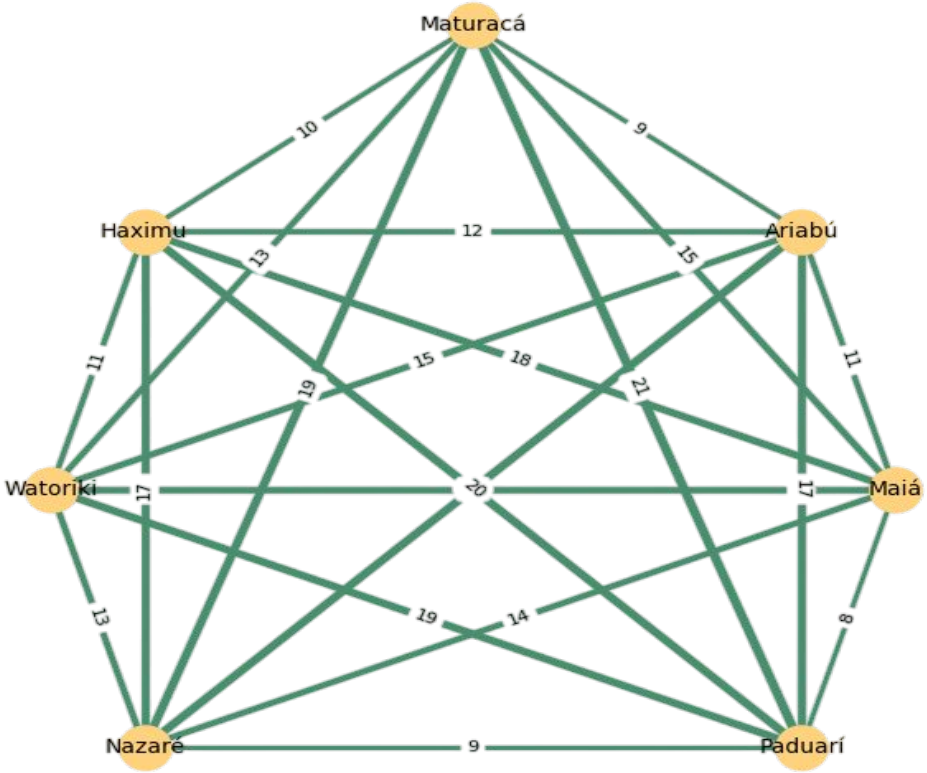
Qual rota seria a Melhor escolha?

O Caminho dos Povos

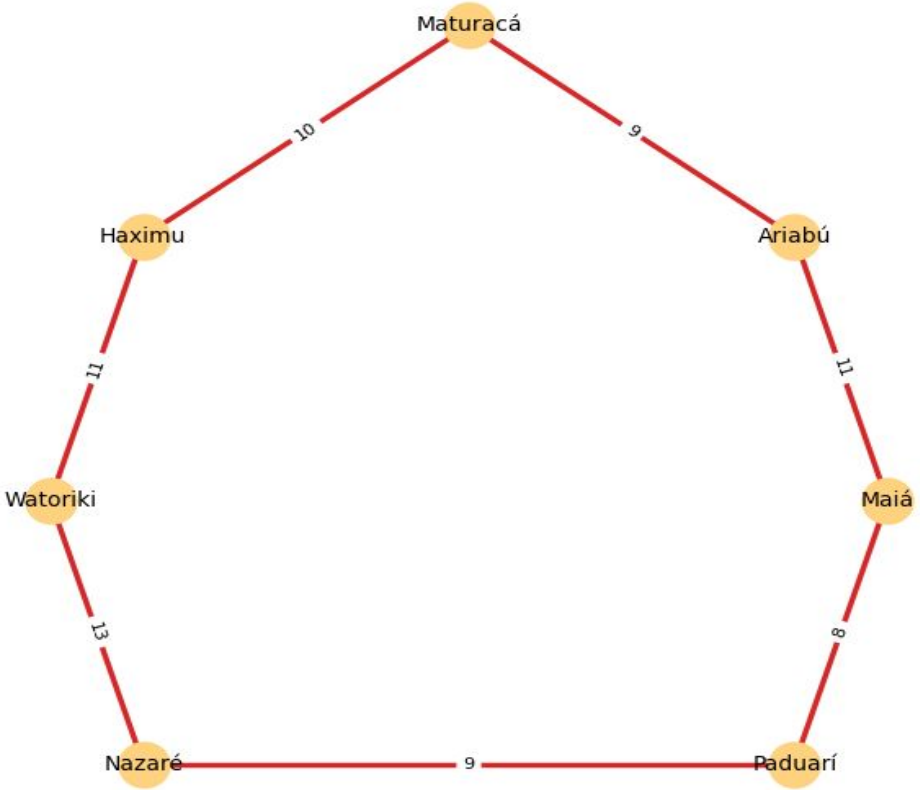
De/Para	Maturacá	Ariabú	Maiá	Paduarí	Nazaré	Watoriki	Haximu
Maturacá	-	9	15	21	19	13	10
Ariabú	9	-	11	17	20	15	12
Maiá	15	11	-	8	14	16	18
Paduarí	21	17	8	-	9	19	20
Nazaré	19	20	14	9	-	13	17
Watoriki	13	15	16	19	13	-	11
Haximu	10	12	18	20	17	11	-

O Caminho dos Povos

O Caminho dos Povos



O Caminho dos Povos



Solução Ótima

Qual algoritmo pode nos dar a solução ótima?

Algoritmo Ótimo

```c
#include <stdio.h>
#include <limits.h>

#define N 7

char *aldeias[N] = {
    "Maturacá", "Ariabü", "Maiá", "Paduarí", "Nazaré", "Watoriki", "Haximu"
};

int dist[N][N] = {
    {0, 9, 15, 21, 19, 13, 10},
    {9, 0, 11, 17, 20, 15, 12},
    {15, 11, 0, 8, 14, 16, 18},
    {21, 17, 8, 0, 9, 19, 20},
    {19, 20, 14, 9, 0, 13, 17},
    {13, 15, 16, 19, 13, 0, 11},
    {10, 12, 18, 20, 17, 11, 0}
};

int min(int a, int b) { return a < b ? a : b; }

```

- **Inicialização de variáveis** pertinentes ao problema
- **Distâncias** entre comunidades
- **Retorna menor valor**

Algoritmo Ótimo

```
22  int tsp(int visited[], int pos, int count, int cost, int start, int path[], int bestPath[], int *bestCost) {
23      if (count == N && dist[pos][start] > 0) {
24          int totalCost = cost + dist[pos][start];
25          if (totalCost < *bestCost) {
26              *bestCost = totalCost;
27              for (int i = 0; i < N; i++) bestPath[i] = path[i];
28          }
29          return totalCost;
30      }
31
32      for (int i = 0; i < N; i++) {
33          if (!visited[i] && dist[pos][i] > 0) {
34              visited[i] = 1;
35              path[count] = i;
36              tsp(visited, i, count + 1, cost + dist[pos][i], start, path, bestPath, bestCost);
37              visited[i] = 0;
38          }
39      }
40      return *bestCost;
41  }
```

- Verifica se já **percorreu todas comunidades**

- **Salva caminho percorrido e custo atual** e chama recursivamente a função

Algoritmo Ótimo

```
43  int main() {
44      int visited[N] = {0}, path[N], bestPath[N];
45      int bestCost = INT_MAX;
46      visited[0] = 1; path[0] = 0;
47
48      tsp(visited, 0, 1, 0, 0, path, bestPath, &bestCost);
49
50      printf(" • Melhor rota (solução ótima):\n");
51      for (int i = 0; i < N; i++) printf("%s -> ", aldeias[bestPath[i]]);
52      printf("%s\n", aldeias[0]);
53      printf(" • Distância total: %d km\n", bestCost);
54      return 0;
55  }
```

- seta **cidade inicial** como a **0** e chama função

Complexidade do Problema

- Em um primeiro momento podemos qualquer roda
  - N possibilidades

- A partir de cada iteração passamos por uma cidade diferente
  - N - 1 possibilidades

N! Combinações

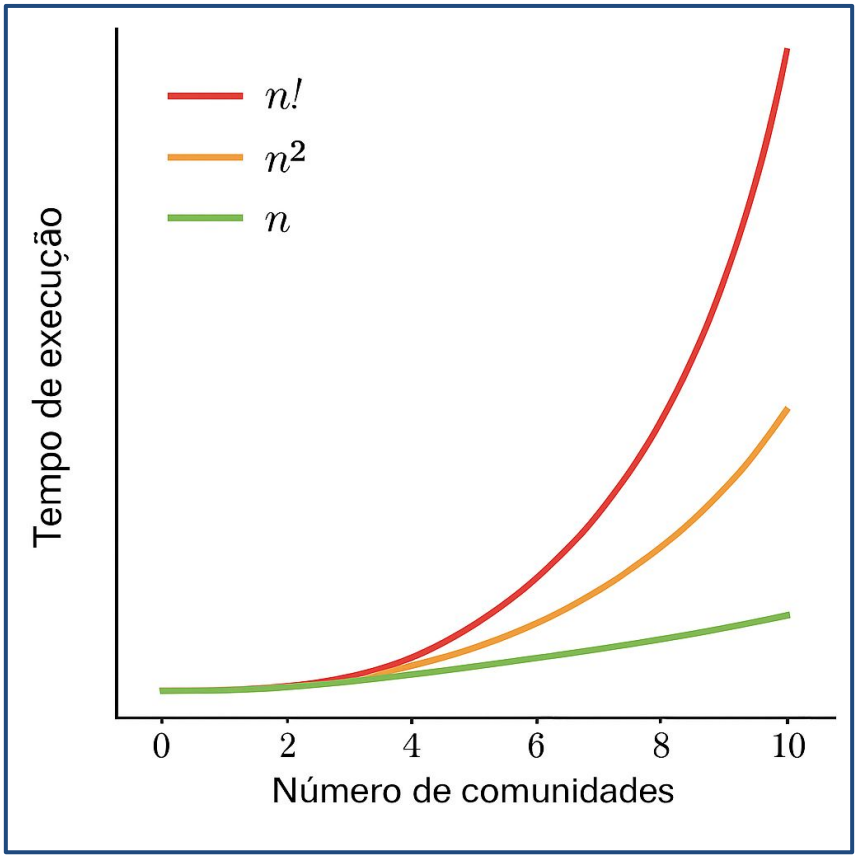
Complexidade do Problema

- Podemos escolher qualquer rota
  - N possibilidades
- Devemos escolher a rota com a menor distância final

- Para N =
  - 5 = 120
  - 10 = 3.628.800
  - 20 = 2.432.902.008.176.640.000

N! Combinações

Complexidade do Problema



Complexidade do Problema

- Solução por **Força Bruta:**
  - Sempre fornece a **solução ótima**
  - Mas nem sempre em tempo viável
  - Inviável em problemas com muitos dados e complexos

Qual abordagem podemos usar?

Solução Sub-Ótima?

O Caminho dos Povos

De/Para	Maturacá	Ariabú	Maiá	Paduarí	Nazaré	Watoriki	Haximu
Maturacá	-	9	15	21	19	13	10
Ariabú	9	-	11	17	20	15	12
Maiá	15	11	-	8	14	16	18
Paduarí	21	17	8	-	9	19	20
Nazaré	19	20	14	9	-	13	17
Watoriki	13	15	16	19	13	-	11
Haximu	10	12	18	20	17	11	-

Algoritmo Sub-Ótimo

1. **Escolha uma comunidade inicial** (ex.: Maturacá).
2. **A partir dela**, procure **a comunidade mais próxima** que ainda não foi visitada.
3. **Vá até essa comunidade** e marque-a como “visitada”.
4. **Repita o processo**, sempre indo para a comunidade mais próxima ainda não visitada.
5. **Quando todas tiverem sido visitadas**, **retorne à comunidade inicial**.

Algoritmo Sub-Ótimo

```c
#include <stdio.h>
#include <limits.h>

#define N 7

char *aldeias[N] = {
    "Maturacá", "Ariabü", "Maiá", "Paduarí", "Nazaré", "Watoriki", "Haximu"
};

int dist[N][N] = {
    {0, 9, 15, 21, 19, 13, 10},
    {9, 0, 11, 17, 20, 15, 12},
    {15, 11, 0, 8, 14, 16, 18},
    {21, 17, 8, 0, 9, 19, 20},
    {19, 20, 14, 9, 0, 13, 17},
    {13, 15, 16, 19, 13, 0, 11},
    {10, 12, 18, 20, 17, 11, 0}
};
```

- **Inicialização de variáveis** pertinentes ao problema
- **Distâncias** entre comunidades

Algoritmo Sub-Ótimo

```
20  int main() {
21      int visited[N] = {0};
22      int current = 0, next, total = 0;
23      visited[current] = 1;
24
25      printf("Rota (heurística gulosa):\n");
26      printf("%s ", aldeias[current]);
27
28      for (int step = 1; step < N; step++) {
29          int best = INT_MAX;
30          for (int j = 0; j < N; j++) {
31              if (!visited[j] && dist[current][j] < best) {
32                  best = dist[current][j];
33                  next = j;
34              }
35          }
36          visited[next] = 1;
37          total += best;
38          current = next;
39          printf("-> %s ", aldeias[current]);
40      }
```

- Escolhe **rota com menor distância** da localização **atual**

Algoritmo Sub-Ótimo

```
41
42      total += dist[current][0];
43      printf("-> %s\n", aldeias[0]);
44      printf("Distância total: %d km\n", total);
45      return 0;
46  }
```

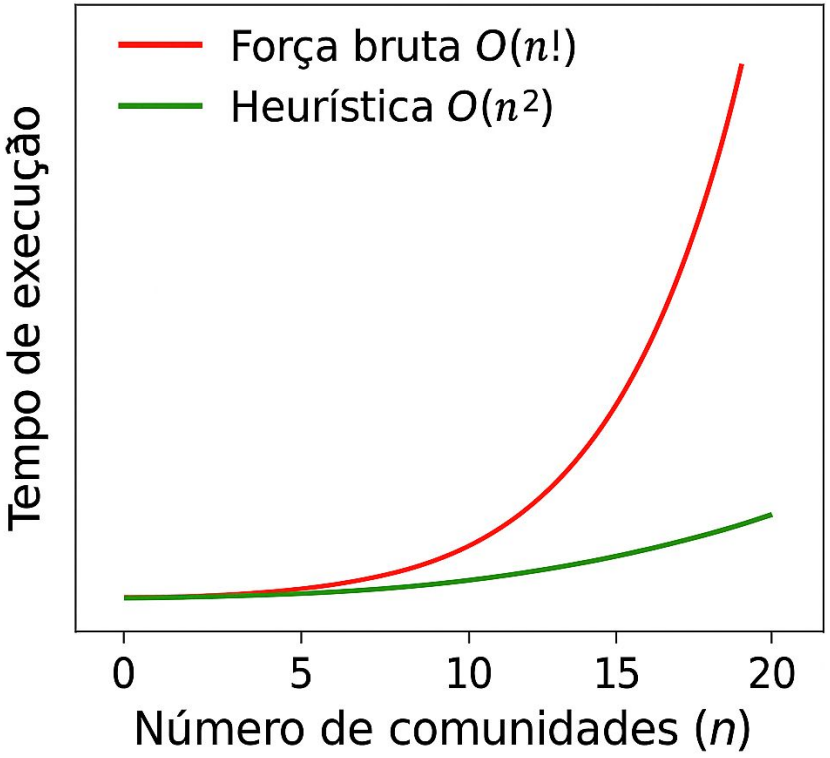
- Mostra resultado

Complexidade do Problema

- Cada passo compara a distância para todas as comunidades não visitadas → **O(n²)**.

- É **muito mais rápida** que a força bruta **O(n!)**.

Complexidade do Problema



Algoritmo Sub-Ótimo

Esse algoritmo sempre retorna a solução ótima?

Carapanaúba (*Aspidosperma nitidum*)

- Empregada pelos Yanomami e outros povos amazônicos como **remédio contra febres e malária**, além de dores no corpo [1].

- Preparada a partir da **casca**, geralmente em infusão ou decocção [1].

Yãkoana (*Virola elongata*, Myristicaceae)

- É a principal planta usada na produção do **rapé xamânico Yanomami (yãkoana ou epéna)** [4].
- O pó é preparado a partir da **casca interna e das sementes** da *Virola elongata* e misturado a outras substâncias vegetais [4].
- Utilizado em **rituais de cura e comunicação espiritual** pelos xamãs (*xapiri*), que o sopram nas narinas para fortalecer o corpo e o espírito [4].
- Também tem função terapêutica, sendo associado à purificação e tratamento de enfermidades espirituais [4].

Tucumã (*Astrocaryum aculeatum*, Arecaceae)

- Palmeira amazônica amplamente utilizada pelos Yanomami como **alimento e fonte de óleo** [5].
- O fruto do tucumã é consumido cru ou em preparações e fornece energia em longas expedições de caça e coleta [5].
- As **fibras do tucumã** também são usadas para confecção de **cordas, arcos e adornos** [5].
- Possui importância cultural e simbólica, associada à subsistência e ao trabalho coletivo [5].

Jenipapo (*Genipa americana*, Rubiaceae)

- O **fruto do jenipapo** é usado para produzir um pigmento preto-azulado aplicado no corpo em **pinturas rituais** e cerimônias festivas [6].
- Entre os Yanomami, as pinturas com jenipapo e urucum representam **proteção espiritual**, beleza e pertencimento social [6].
- Também é empregado como **planta medicinal** por diferentes povos amazônicos, com propriedades antissépticas e calmantes [6].

Fibra de Arumã (*Ischnosiphon arouma*, Marantaceae)

- O **arumã** é amplamente usado na **cestaria Yanomami**, especialmente na confecção de **paneiros e balaios** usados para coleta e transporte de alimentos [3].
- As fibras são retiradas dos caules e trançadas por mulheres, representando um saber transmitido entre gerações [3].
- Além da utilidade prática, o trançado expressa **identidade e estética cultural** [3].

Referências

1. Milliken, William, and Bruce Albert. "The use of medicinal plants by the Yanomami Indians of Brazil." *Economic Botany* 50.1 (1996): 10-25.

2. Gallois, D. T. (2002). *Redes de Relações nas Cestarias Indígenas do Brasil.* Museu do Índio / FUNAI.

3. Rodrigues, Igor. "Corpos que emergem: vegetais trançados e sua persistência entre os povos do rio Mapuera." *Revista de Arqueologia* 34.3 (2021): 146-177.

4. Inoue, Cristina Yumie Aoki. "Worlding the study of global environmental politics in the Anthropocene: Indigenous voices from the Amazon." *Global Environmental Politics* 18.4 (2018): 25-42.

5. Costa, Joanne Regis da, Johannes Van Leeuwen, and Jarbas Anute Costa. "Tucumã-do-amazonas, Astrocaryum tucuma Martius." *Frutíferas e plantas úteis na vida amazônica., pgs. 221-228* (2005).

6. Smiljanic, Maria Inês. "A comemoração do dia do índio entre os Yanomami de Maturacá (AM)." *Faces da indianidade. Curitiba: Nexo Design* (2009): 155-165.

7. CORMEN, Thomas H. et al. Algoritmos de aproximação. In: CORMEN, Thomas H. et al. Algoritmos: teoria e prática. 4. ed. Rio de Janeiro: Elsevier, 2022. cap. 35, p. 774.

8. KOPENAWA, Davi; ALBERT, Bruce. A queda do céu: palavras de um xamã Yanomami. São Paulo: Companhia das Letras, 2015. 730 p.

Referências

9. **BRASIL. Ministério da Saúde.** Quantas pessoas vivem hoje no território Yanomami? Brasília, DF: Ministério da Saúde, 1 fev. 2023. Disponível em: https://www.gov.br/saude/pt-br/composicao/svsa/coes/coe-yanomami/faq/perguntas-frequentes-coe-yanomami/quantas-pessoas-vivem-hoje-no. Acesso em: 29 out. 2025. Serviços e Informações do Brasil

10. **INSTITUTO SOCIOAMBIENTAL (ISA).** Yanomami — Povos Indígenas no Brasil. São Paulo: ISA, [s.d.]. Disponível em: https://pib.socioambiental.org/pt/Povo%3AYanomami. Acesso em: 29 out. 2025.

O Problema do Caminho dos Povos Originários

Professor Doutor Weslen Schiavon

Contato: wsdsouza@inf.ufpel.edu.br