Princípios de Contagem e Enumeração Computacional

Objetivos

Contar/listar o número de elementos de conjuntos finitos

Aplicações

Objetivos

Contar/listar o número de elementos de conjuntos finitos

Aplicações

- Determinar o número de operações realizadas por um algoritmo para determinar sua eficiência

Objetivos

Contar/listar o número de elementos de conjuntos finitos

Aplicações

- Determinar o número de operações realizadas por um algoritmo para determinar sua eficiência
- Ferramenta utilizada para resolver problemas estatísticos

Objetivos

Contar/listar o número de elementos de conjuntos finitos

Aplicações

- Determinar o número de operações realizadas por um algoritmo para determinar sua eficiência
- Ferramenta utilizada para resolver problemas estatísticos
- Desenvolver o raciocínio dos alunos de ED

Objetivos

Por que estudar problemas do tipo:

1. De quantas maneiras é possível dispor os alunos de uma turma em fila, sem que alunos com nomes consecutivos na ordem alfabética fiquem juntos?
2. Quantas permutações distintas tem a palavra MISSISSIPI?

Duas justificativas:

1. Estes problemas são fáceis de formular;
2. Algoritmo simples para resolver vários problemas: enumerar todas as possibilidades

Objetivos

Exemplo

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

Objetivos

Exemplo

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

P1) Quantos trajetos distintos o caminhão pode percorrer?

Objetivos

Exemplo

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

P1) Quantos trajetos distintos o caminhão pode percorrer?

P2) Qual o trajeto que minimiza o consumo do caminhão?

Exemplo

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

P1) Quantos trajetos distintos o caminhão pode percorrer?

– 12 possibilidades para a primeira parada, 11 para a segunda, etc.

P2) Qual o trajeto que minimiza o consumo do caminhão?

Objetivos

Exemplo

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

P1) Quantos trajetos distintos o caminhão pode percorrer?

– 12 possibilidades para a primeira parada, 11 para a segunda, etc.
– Total: $12! = 63.228.211.200$

P2) Qual o trajeto que minimiza o consumo do caminhão?

Objetivos

Exemplo

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ ao longo de um dia. Sabe-se que o consumo médio para ir da localidade L_i para localidade L_j é c_{ij}, e que o caminhão deve partir de sua garagem e retornar para mesma.

P1) Quantos trajetos distintos o caminhão pode percorrer?

– 12 possibilidades para a primeira parada, 11 para a segunda, etc.
– Total: $12! = 63.228.211.200$

P2) Qual o trajeto que minimiza o consumo do caminhão?

– Possível solução é gerar todos os possíveis trajetos

Objetivos

Esse é o problema Traveling Salesman Problem (TSP)

Objetivos

Esse é o problema Traveling Salesman Problem (TSP)

Não é conhecido nenhum algoritmo eficiente para resolver TSP

Objetivos

Esse é o problema Traveling Salesman Problem (TSP)

Não é conhecido nenhum algoritmo eficiente para resolver TSP

Existência implicaria $P = NP$. Vale USD 1.000.00

Objetivos

Esse é o problema Traveling Salesman Problem (TSP)

Não é conhecido nenhum algoritmo eficiente para resolver TSP

Existência implicaria P = NP. Vale USD 1.000.00

Melhores algoritmos utilizam enumeração parcial, de forma esperta (Programação Inteira)

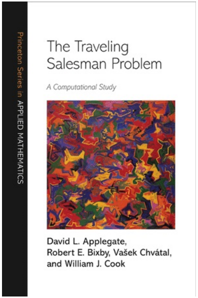
Objetivos

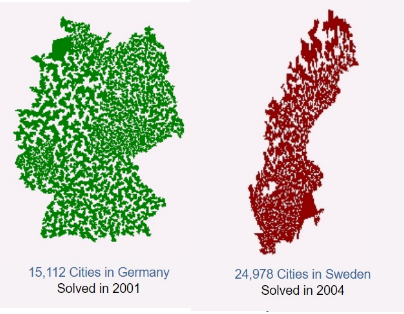
Esse é o problema Traveling Salesman Problem (TSP)

Não é conhecido nenhum algoritmo eficiente para resolver TSP

Existência implicaria P = NP. Vale USD 1.000.00

Melhores algoritmos utilizam enumeração parcial, de forma esperta (Programação Inteira)





Princípio da Multiplicação

Técnicas elementar: princípio da multiplicação

Princípio da Multiplicação

Técnicas elementar: princípio da multiplicação

- Considere eventos $E_1, E_2, \ldots, E_k$

Princípio da Multiplicação

Técnicas elementar: princípio da multiplicação

- Considere eventos $E_1, E_2, \ldots, E_k$
- Evento E_i pode ocorrer de n_i formas diferentes, independente dos outros eventos

Princípio da Multiplicação

Técnicas elementar: princípio da multiplicação

- Considere eventos $E_1, E_2, \ldots, E_k$
- Evento E_i pode ocorrer de n_i formas diferentes, independente dos outros eventos
- Então a sequência de eventos $E_1 E_2 E_3 \ldots E_k$ pode ocorrer de

$$n_1 \times n_2 \times n_3 \times \ldots \times n_k$$

formas diferentes

Princípio da Multiplicação

Exemplo

Em uma placa de carro, as três primeiras posições são letras e as quatro restantes são dígitos. Quantas placas distintas são possíveis?

Princípio da Multiplicação

Exemplo

Em uma placa de carro, as três primeiras posições são letras e as quatro restantes são dígitos. Quantas placas distintas são possíveis?

Evento E_i: atribuir letra/digito a i-ésima posição

Princípio da Multiplicação

Exemplo

Em uma placa de carro, as três primeiras posições são letras e as quatro restantes são dígitos. Quantas placas distintas são possíveis?

Evento E_i: atribuir letra/digito a i-ésima posição

Número de possibilidades do evento E_i é independente dos outros eventos

Princípio da Multiplicação

Exemplo

Em uma placa de carro, as três primeiras posições são letras e as quatro restantes são dígitos. Quantas placas distintas são possíveis?

Evento E_i: atribuir letra/digito a i-ésima posição

Número de possibilidades do evento E_i é independente dos outros eventos

Temos 26 possibilidades para eventos E_i com $i = 1, 2, 3$
10 possibilidades para eventos E_i com $i = 4, 5, 6, 7$

Princípio da Multiplicação

Exemplo

Em uma placa de carro, as três primeiras posições são letras e as quatro restantes são dígitos. Quantas placas distintas são possíveis?

Evento E_i: atribuir letra/digito a i-ésima posição

Número de possibilidades do evento E_i é independente dos outros eventos

Temos 26 possibilidades para eventos E_i com $i = 1, 2, 3$
10 possibilidades para eventos E_i com $i = 4, 5, 6, 7$

Pelo princípio da multiplicação, o número de placas é $26^3 \cdot 10^4$.

Princípio da Multiplicação

Exemplo (TSP)

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

Princípio da Multiplicação

Exemplo (TSP)

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

Quantos possíveis trajetos?

Princípio da Multiplicação

Exemplo (TSP)

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

Note que não queremos revisitar a mesma localidade (sem repetição)

Princípio da Multiplicação

Exemplo (TSP)

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ ao longo de um dia. Sabe-se que o consumo médio para ir da localidade L_i para localidade L_j é c_{ij}, e que o caminhão deve partir de sua garagem e retornar para mesma.

Note que não queremos revisitar a mesma localidade (sem repetição)

Evento E_i: atribuir i-ésima localidade a ser visitada

Princípio da Multiplicação

Exemplo (TSP)

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

Note que não queremos revisitar a mesma localidade (sem repetição)

Evento E_i: atribuir i-ésima localidade a ser visitada

Número de possibilidades do evento E_i ainda é independente dos outros eventos

Princípio da Multiplicação

Exemplo (TSP)

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

Note que não queremos revisitar a mesma localidade (sem repetição)

Evento E_i: atribuir i-ésima localidade a ser visitada

Número de possibilidades do evento E_i ainda é independente dos outros eventos

Número de possibilidade evento E_1 é 12, E_2 é 11, etc.

Princípio da Multiplicação

Exemplo (TSP)

Considere um caminhão que necessita entregar mercadorias em 12 localidades $\{L_1, \ldots, L_{12}\}$ *ao longo de um dia. Sabe-se que o consumo médio para ir da localidade* L_i *para localidade* L_j *é* c_{ij}*, e que o caminhão deve partir de sua garagem e retornar para mesma.*

Note que não queremos revisitar a mesma localidade (sem repetição)

Evento E_i: atribuir i-ésima localidade a ser visitada

Número de possibilidades do evento E_i ainda é independente dos outros eventos

Número de possibilidade evento E_1 é 12, E_2 é 11, etc.

Pelo princípio da multiplicação, número de trajetos é
$12 \cdot 11 \cdot \ldots \cdot 1 = 12!$

Princípio da Multiplicação

Exemplo

Considere o trecho de código abaixo

Para i=1,…ℓ
    Para j=1,…m
        Para k=1,…n
            PRINT(“OI”)

Q: Quantas vezes o código imprime ‘OI’?

Princípio da Multiplicação

Exemplo

Considere o trecho de código abaixo

```
Para i=1,...ℓ
    Para j=1,...m
        Para k=1,...n
            PRINT("OI")
```

Q: Quantas vezes o código imprime 'OI'?

A: $\ell \cdot m \cdot n$

Gerando todas as palavras de um alfabeto

Agora queremos enumerar todas as possibilidades

Gerando todas as palavras de um alfabeto

Agora queremos enumerar todas as possibilidades

Por exemplo, gerar todas as palavras de n letras a partir de um alfabeto com s letras diferentes

Gerando todas as palavras de um alfabeto

Agora queremos enumerar todas as possibilidades

Por exemplo, gerar todas as palavras de n letras a partir de um alfabeto com s letras diferentes

(O número de palavras possíveis é s^n)

Gerando todas as palavras de um alfabeto

Utilizamos algoritmo recursivo baseado em “indução” em n

Gerando todas as palavras de um alfabeto

Utilizamos algoritmo recursivo baseado em “indução” em n

Caso base: $n = 0$**.** Não gera nada

Gerando todas as palavras de um alfabeto

Utilizamos algoritmo recursivo baseado em "indução" em n

Caso base: $n = 0$**.** Não gera nada

Passo indutivo. Suponha que saiba gerar todas as palavras de tamanho n. Como usar isso para gerar palavras de tamanho $n+1$?

$\Rightarrow$ Gerar todas possibilidades de ultima letra,
chama recursivamente para gerar primeiras n posições

Gerando todas as palavras de um alfabeto

Vetor P: vai guardando palavra parcialmente preenchida

```
PROCEDIMENTO GeraPalavras(i,P)   #P[i+1], P[i+2],..., P[n] estão preenchida
    If  i = 0    #P está totalmente preenchido
        Imprima a palavra P
    Else
        For  j = 1 to s
            P[i] ← letra Letra[j]
            GeraPalavras(i-1, P)
        End For
    End If

MAIN
    Letra ← vetor com s posições contendo as letras possiveis
    P ← vetor de n posições
    GeraPalavras(n, P)
```

Exercicios

Exercício 1: Quantas funções existem de um conjunto com n elementos a um conjunto com m elementos?

Exercício 2: Use a regra do produto para mostrar que o número de subconjuntos de um conjunto S é $2^{|S|}$

Exercício 3: Escreva um código recursivo para gerar todas as palavras com n letras minusculas sem repeticao