Applications of permutations and k-subsets

A. Topics and links extracted from personal communications with teachers at FEE CTU

Brute force decryption/cracking of simple codes (e.g. PIN)
Brute force solution of Travelling Salesman Problem
Dynamic Opponent Choice in Tournaments
Experimental assessment of sorting algorithms behaviour
Feature selection in machine learning
Gestalt Theory and Image Analysis
Graph isomorphism problem
Information compression in Rubik's cube solution and analysis
Interleaving in error correction codes
Linear-Time Ranking of Permutations
MinHash Algorithm for estimating the similarity between two sets
Morton Codes for High Performance Bounding Volume Hierarchy (BVH) Construction
Oriented Matroids
Quadratic assignment problem
RANSAC parameters estimation
Set covering/packing
Simulation and analysis of a combinatorial (e.g. card) game
Unique permutation hash function

Error correction codes -- interleaving strategy
https://en.wikipedia.org/wiki/Error_correction_code#Interleaving
Unique permutation hashing
https://www.sciencedirect.com/science/article/pii/S0304397513000133?via%3Dihub
Rank of a permutation is a compressed form of the permutation
https://github.com/benbotto/rubiks-cube-cracker/
Linear-Time Ranking of Permutations
https://ufal.mff.cuni.cz/~straka/papers/2007-ranking.pdf
Morton codes in High Performance Bounding Volume Hierarchy Construction (eg. in ray-tracing)
https://www.highperformancegraphics.org/wp-content/uploads/2017/Papers-Session3/HPG207_ExtendedMortonCodes.pdf

See also the references in particular papers and presentations linked above.

B. General overview topics listed by ChatGPT

Permutations:
1. Cryptography and Security
2. Combinatorial Optimization and Scheduling
3. Data Compression and Encoding
4. Biology and Chemistry
5. Computer Graphics and Image Processing
6. Game Theory and Puzzle Solving

k-subsets:
1. Data Mining and Machine Learning
2. Database Query Optimization
3. Graph Theory and Network Analysis
4. Cryptography
5. Combinatorial Optimization
6. Bioinformatics and Computational Biology
7. Error Detection and Correction in Coding Theory
8. Search and Enumeration Algorithms
9. Parallel and Distributed Computing
10. Game Development and AI

Permutations:

1. Cryptography and Security

Permutations in Key Generation: In cryptographic systems, permutations are used to generate keys or obfuscate data. Some encryption algorithms, especially block ciphers (like AES), involve permuting bits or bytes as part of their transformation steps to ensure security.

Permutation Ciphers: Some ciphers specifically use permutations as a core component to rearrange data in secure patterns, a concept dating back to early classical cryptography methods.

2. Combinatorial Optimization and Scheduling

Traveling Salesperson Problem (TSP): This is a classic optimization problem where we seek the shortest possible route that visits a set of cities and returns to the origin. Generating permutations of city orders allows algorithms to exhaustively (or heuristically) evaluate possible solutions. Although exact solutions use other methods due to the factorial growth, permutations play a critical role in approximation techniques.

Job Scheduling and Assignment Problems: In tasks like job-shop scheduling or employee shift planning, permutations help evaluate different orderings of tasks or assignments to optimize criteria such as time, cost, or resource usage.

3. Data Compression and Encoding

Burrows-Wheeler Transform (BWT): Widely used in data compression algorithms like bzip2, the BWT involves sorting rotations (or cyclic permutations) of a string to make it more compressible. The sorted positions are then indexed, which relates to finding the rank and permutation transformations.

Gray Codes: Generating a sequence of permutations where each subsequent permutation changes only slightly (like Gray codes) is useful in error detection and correction, especially for data sent over noisy channels.

4. Biology and Chemistry

Genome Sequencing: In bioinformatics, analyzing and comparing different possible arrangements (permutations) of genetic sequences is essential in genome sequencing. Permutations are used to evaluate different alignments or orderings of sequences, especially when dealing with structural variations.

Protein Folding: Permutation generation algorithms can help simulate different folding patterns of amino acid sequences in proteins to find stable configurations, which is crucial for drug design and understanding biological processes.

5. Computer Graphics and Image Processing

Rendering and Sampling: In computer graphics, permutations are often used to randomize sampling points to avoid artifacts, improving rendering quality in techniques like ray tracing. Permutations are also used in procedural generation of textures or in randomized patterns.

Image Permutations for Compression: Some image compression algorithms leverage permutations of pixel values to find more compressible representations, where lexicographical ranks might determine specific arrangements of data blocks.

6. Game Theory and Puzzle Solving

Puzzle Solving Algorithms: In games like the Rubik's Cube, permutations represent different states. Algorithms compute permutations and ranks to track progress toward solutions, and heuristics often rely on ordering states in specific patterns to find efficient paths.

Game AI: Permutations can model possible game moves or states in certain types of strategy games, which AI systems then use to explore game scenarios more effectively.

In all of these cases, efficient handling of permutations—whether through ranking, generating, or ordering—is critical to solving complex, real-world problems in a computationally feasible way. So, while it might seem like an abstract topic, permutations play a practical role in fields ranging from cryptography to bioinformatics and machine learning.

k-subsets (or combinations of a fixed size k-subsets chosen from a larger set):

1. Data Mining and Machine Learning

Feature Selection: In machine learning, selecting the best subset of features (or variables) from a larger pool is often critical for creating accurate models. When there are many potential features, selecting k-subsets allows researchers to evaluate combinations to find the subset that provides the best performance.

Association Rule Mining: In data mining, specifically in market basket analysis, algorithms like Apriori search for frequent itemsets of a fixed size k. These k-subsets represent combinations of products that frequently appear together, which can inform marketing strategies or product placement.

2. Database Query Optimization

Index Selection: Database systems use k-subsets to select the best combination of indexes for optimal query performance. Given a large number of possible indexes on a table, finding the best subset of k indexes is essential to improve search speed without excessive storage costs.

Join Optimization: For queries involving multiple tables, finding the optimal sequence and subset of joins is critical. Query planners consider subsets of tables to join in order to minimize processing time, especially when dealing with complex queries on large databases.

3. Graph Theory and Network Analysis

Cliques and Communities: Finding cliques (fully connected subgraphs) or k-cliques in a graph is important for identifying closely connected communities, whether in social networks or biological networks. Algorithms that identify these k-subsets of nodes help understand dense areas of connectivity.

Vertex Cover and Dominating Sets: In optimization problems like the vertex cover or dominating set problems, subsets of nodes of size k are chosen to satisfy certain graph properties, such as covering all edges or influencing the rest of the graph. These concepts have practical applications in network security and resource allocation.

4. Cryptography

Subset Sum Problem: Many cryptographic protocols are based on the difficulty of finding specific k-subsets that satisfy certain mathematical properties, such as subsets that sum to a particular value. This problem is the basis for several cryptographic systems and has implications for security protocols.

Secret Sharing: In schemes where a secret is divided among multiple participants (such as Shamir's Secret Sharing), certain k-subsets are chosen to reconstruct the secret, while smaller subsets reveal no information. This is widely used in secure data storage and multi-party security protocols.

5. Combinatorial Optimization

Knapsack and Subset-Sum Variants: In resource allocation problems, especially knapsack-type problems, finding optimal k-subsets of items that maximize value without exceeding a weight limit is crucial. These problems arise in logistics, budgeting, and planning applications.

Subset Selection in Portfolio Optimization: In finance, portfolio optimization involves selecting a k-subset of assets that balances risk and return. These combinatorial optimizations are crucial for managing investment portfolios under constraints, such as risk limits or diversification requirements.

6. Bioinformatics and Computational Biology

Genetic and Protein Interaction Networks: To understand genetic interactions, researchers look for significant k-subsets of genes or proteins that interact in meaningful ways. Identifying these subsets helps in understanding disease mechanisms or protein function.

Gene Expression Analysis: When analyzing gene expression data, researchers look for k-subsets of genes whose expression patterns are strongly correlated across samples. Identifying these subsets helps in clustering genes and understanding biological pathways.

7. Error Detection and Correction in Coding Theory

Error-Correcting Codes: In coding theory, subsets of codewords are carefully chosen to ensure that any k-bit error can be detected or corrected. Techniques like Reed-Solomon codes, for instance, rely on specific subsets of codewords to ensure data integrity in noisy transmission channels.

Parity Checks and Checksums: For simple error detection, selected k-subsets of data bits are grouped to compute parity bits. These are used in storage systems, networking, and distributed systems to ensure data consistency.

8. Search and Enumeration Algorithms

Backtracking Algorithms: In algorithms for constraint satisfaction problems (like Sudoku or n-queens), generating k-subsets allows systematic exploration of possible solutions while pruning impossible combinations.

Subset Enumeration for Exact Solutions: Some exact algorithms, like those solving the knapsack problem or set covering problem, enumerate k-subsets as part of their search for an optimal solution. These are used when approximation is not acceptable, such as in scheduling critical tasks or planning routes.

9. Parallel and Distributed Computing

Task Allocation: When distributing tasks across multiple processors, certain k-subsets of tasks may be allocated together to balance load or minimize inter-process communication. Choosing subsets of tasks optimally can maximize computational efficiency.

Cluster Analysis: In distributed systems, k-subsets can be used to create clusters of nodes or processes that work together efficiently on a subset of tasks, reducing communication costs and improving performance in large-scale computations.

10. Game Development and AI

Subset Exploration in Game Trees: AI algorithms in games often generate and evaluate k-subsets of potential moves to identify advantageous strategies. In card games, for example, k-subsets of potential hands are analyzed to make strategic decisions.

Randomized Subset Sampling for NPC Behavior: To make non-player characters (NPCs) behave in diverse ways, developers may sample k-subsets of possible actions or decisions to introduce variability in behavior patterns.

In these contexts, generating, managing, and optimizing over k-subsets are essential for balancing performance, accuracy, and efficiency, making them a powerful tool for solving both theoretical and practical problems across computer science and related fields.