CSI5165 COMBINATORIAL ALGORITHMS

Prof. Lucia Moura

Fall 2005
INTRODUCTION TO COMBINATORIAL ALGORITHMS
Introduction to Combinatorial Algorithms

What are:

- Combinatorial Structures?
- Combinatorial Algorithms?
- Combinatorial Problems?
Combinatorial Structures

Combinatorial structures are collections of $k$-subsets/$K$-tuple/permutations from a parent set (finite).

Examples:

- **Undirected Graphs:**
  Collections of 2-subsets (edges) of a parent set (vertices).

  \[
  V = \{1, 2, 3, 4\} \quad E = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{3, 4\}\}
  \]

- **Directed Graphs:**
  Collections of 2-tuples (directed edges) of a parent set (vertices).

  \[
  V = \{1, 2, 3, 4\} \quad E = \{(2, 1), (3, 1), (1, 4), (3, 4)\}
  \]

- **Hypergraphs or Set Systems:**
  Similar to graphs, but (hyper) edges may be sets with more than two elements.

  \[
  V = \{1, 2, 3, 4\} \quad E = \{\{1, 3\}, \{1, 2, 4\}, \{3, 4\}\}
  \]
Building blocks: finite sets, finite lists (tuples)

• Finite Sets

\[ X = \{1, 2, 3, 5\} \]
- unordered structure, no repeats
  \[ \{1, 2, 3, 5\} = \{2, 1, 5, 3\} = \{2, 1, 1, 5, 3\} \]
- cardinality (size) = number of elements \(|X| = 4\).

A \textit{k-subset} of a finite set \(X\) is a set \(S \subseteq X, |S| = k\).
For example: \(\{1, 2\}\) is a 2-subset of \(X\).

• Finite Lists (or Tuples)

\[ L = [1, 5, 2, 1, 3] \]
- ordered structure, repeats allowed
  \[ [1, 5, 2, 1, 3] \neq [1, 1, 2, 3, 5] \neq [1, 2, 3, 5] \]
- length = number of items, length of \(L\) is 5.

An \textit{n-tuple} is a list of length \(n\).
A \textit{permutation} of an \(n\)-set \(X\) is a list of length \(n\) such that every element of \(X\) occurs exactly once.

\[ X = \{1, 2, 3\}, \quad \pi_1 = [2, 1, 3] \quad \pi_2 = [3, 1, 2] \]
Graphs

**Definition.** A *graph* is a pair $(V, E)$ where:

- $V$ is a finite set (of vertices).
- $E$ is a finite set of 2-subsets (called *edges*) of $V$.

Example: $G_1 : V = \{0, 1, 2, 3, 4, 5, 6, 7\}$

$E = \{\{0, 4\}, \{0, 1\}, \{0, 2\}, \{2, 3\}, \{2, 6\}, \{1, 3\}, \{1, 5\}, \{3, 7\}, \{4, 5\}, \{4, 6\}, \{4, 7\}, \{5, 6\}, \{5, 7\}, \{6, 7\}\}$

Complete graphs: graphs with all possible edges.

Examples:

- $K_1$
- $K_2$
- $K_3$
- $K_4$

Substructures of a graph:

1. A **hamiltonian circuit** (hamiltonian cycle) is a closed path that passes through each vertex once.

The following list describes a hamiltonian cycle in $G_1$:

$[0, 1, 5, 4, 6, 7, 3, 2]$ (different lists may describe the same cycle).
Traveling Salesman Problem: given a weight/cost function 
\( w : E \to R \) on the edges of \( G \), find a smallest weight 
hamiltonian cycle in \( G \).

2. A clique in a graph \( G = (V, E) \) is a subset \( C \subseteq V \) such that 
\( \{x, y\} \in E \), for all \( x, y \in C \) with \( x \neq y \).
(Or equivalently: the subgraph induced by \( C \) is complete).

Example:
\( G_2 : \)

Some cliques of \( G_2 : \)

Maximum cliques of \( G_2 : \)

Famous problems involving cliques:
- Maximum clique problem: find a maximum clique in a graph.
- All cliques problem: find all cliques in a graph without repetition.
Set systems/Hypergraphs

Definition. A set system (or hypergraph) is a pair \((X, \mathcal{B})\) where:
\(X\) is a finite set (of points).
\(\mathcal{B}\) is a finite set of subsets of \(X\) (blocks).

Examples:

- Graph: A graph is a set system with every block with cardinality 2.
- Partition of a finite set:
  A partition is a set system \((X, \mathcal{B})\) such that
  \(B_1 \cap B_2 = \emptyset\) for all \(B_1, B_2 \in \mathcal{B}, B_1 \neq B_2\), and
  \[\bigcup_{B \in \mathcal{B}} B = X.\]
- Steiner triple system (a type of combinatorial designs):
  \(\mathcal{B}\) is a set of 3-subsets of \(X\) such that for each \(x, y \in X, x \neq y\),
  there exists exactly one block \(B \in \mathcal{B}\) with \(\{x, y\} \subseteq B\).

Example:
\(X = \{0, 1, 2, 3, 4, 5, 6\}\)
\(\mathcal{B} = \{\{0, 1, 2\}, \{0, 3, 4\}, \{0, 5, 6\}, \{1, 3, 5\}, \{1, 4, 6\}, \{2, 3, 6\}, \{2, 4, 5\}\} \)
Combinatorial Algorithms

Algorithms for investigating combinatorial structures. Three types:

- **Generation**
  Construct all combinatorial structures of a particular type.
  - Generate all subsets/permutations/partitions of a set.
  - Generate all cliques of a graph.
  - Generate all maximum cliques of a graph.
  - Generate all Steiner triple systems of a finite set.

- **Enumeration**
  Compute the number of different structures of a particular type.
  - Compute the number of subsets/permutations/partitions of a set.
  - Compute the number of cliques of a graph.
  - Compute the number of maximum cliques of a graph.
  - Compute the number of Steiner triple systems of a finite set.

- **Search**
  Find at least one example of a combinatorial structures of a particular type (if one exists).

Optimization problems can be seen as a type of search problem.
  - Find a Steiner triple system on a finite set. (feasibility)
  - Find a maximum clique of a graph. (optimization)
  - Find a hamiltonian cycle in a graph. (feasibility)
  - Find a smallest weight hamiltonian cycle in a graph. (optimization)
Hardness of Search and Optimization

Many search and optimization problems are NP-hard, which means that

unless $P = NP$ (an important unsolved complexity question)

no polynomial-time algorithm to solve the problem would exist.
Approaches for dealing with NP-hard problems:

• Exhaustive Search
  – exponential-time algorithms.
  – solves the problem exactly

  (Backtracking and Branch-and-Bound)

• Heuristic Search
  – algorithms that explore a search space to find a feasible
    solution that is “close to” optimal, within a time limit
  – approximates a solution to the problem

  (Hill-climbing, Simulated annealing, Tabu-Search, Genetic
   Algorithms)

• Approximation Algorithms
  – polynomial time algorithm
  – we have a provable guarantee that the solution found is “close
    to” optimal.

  (not covered in this course)
## Types of Search Problems

1) **Decision Problem:**
A yes/no problem

**Problem 1:**
Clique (decision)
Instance: graph $G = (V, E)$, target size $k$

**Question:**
Does there exist a clique $C$ of $G$ with $|C| = k$?

2) **Search Problem:**
Find the guy.

**Problem 2:**
Clique (search)
Instance: graph $G = (V, E)$, target size $k$

**Find:**
a clique $C$ of $G$ with $|C| = k$, if one exists.

3) **Optimal Value:**
Find the largest target size.

**Problem 3:**
Clique (optimal value)
Instance: graph $G = (V, E)$,

**Find:**
the maximum value of $|C|$, where $C$ is a clique

4) **Optimization:**
Find an optimal guy.

**Problem 4:**
Clique (optimization)
Instance: graph $G = (V, E)$,

**Find:**
a clique $C$ such that $|C|$ is maximize (max. clique)
Plan for the Course

1. Generating elementary combinatorial objects
   Sequential generation (successor), rank, unrank.
   Algorithms for subsets, $k$-subsets, permutations.
   Reference: textbook chapter 2. [2 weeks]

2. Exhaustive Generation and Exhaustive Search
   Backtracking algorithms
   (exhaustive generation, exhaustive search, optimization)
   Brach-and-bound
   (exhaustive search, optimization)
   Reference: textbook chapter 4. [3 weeks]

3. Heuristic Search
   Hill-climbing, Simulated annealing, Tabu-Search, Genetic Algs.
   Applications of these techniques to various problems.
   Reference: textbook chapter 5. [3 weeks]

4. Computing Isomorphism and Isomorph-free Exhaustive Generation
   Graph isomorphism, isomorphism of other structures.
   Computing invariants.
   Computing certificates.
   Isomorph-free exhaustive generation.
   Example: Generate all trees on $n$ vertices, without isomorphic copies.
   Reference: textbook chapter 7, papers. [3 weeks]