CSE 21
Mathematics for Algorithm and System Analysis

Basic Information
Course website:
http://users.sdsc.edu/~jianwu/CSE21/
Instructor Team

- Instructor: Jianwu Wang, Ph.D.
- TA: Uday Khandelwal
- Tutor:
  - Ruiqing Qiu
  - Atyansh Jaiswal
  - Jiajie Duan
  - Justin W. Lai
What is this Class about?

• Discrete mathematics: fundamental mathematics to support many aspects of Computer Science, such as programming, data structure, database, artificial intelligence.

• The problems in this class will be solved by mathematically techniques and most of the solutions can be easily expressed in programming languages.
What is this Class NOT about?

• Not a programming class: no lab, no programming tests.
• Not pure mathematics: focus on application, not proof or inventing new theorems.
Course Website

- [http://users.sdsc.edu/~jianwu/CSE21/](http://users.sdsc.edu/~jianwu/CSE21/)
- Notes will be put online around one or two days before each lecture.
  - Print them by yourself if needed.
- Office hour schedule can be found online.
- Online forum [https://piazza.com/ucsd/summer2013/cse21/home](https://piazza.com/ucsd/summer2013/cse21/home)
Textbook and Syllabus

• Textbook: freely available electronic textbook by Edward A. Bender and S. Gill Williamson:
  – Basic Counting and Listing
  – Functions
  – Decision Trees
  – Basic Concepts in Graph Theory
• Each unit has four sections.
• Content marked with star will not be covered.
• Each class will cover one section with minor variations.
• Textbook and class schedule info is at course website.
Grading

- Grading: Check time schedule online.
  - Quizzes: 20%. 4 quizzes total, 15 minutes per quiz. For you and me to know how the study is going.
  - Midterm: 30%
  - Final: 50%
- The final grades will be curved. F is possible.
- Homework
  - No hand in; no grading.
  - Solutions of all exercises are available at course website.
  - Check whether you understand the knowledge.
  - TA will discuss it during discussion session.
How to Study this Class Well?

• Monday to Thursday
  – Study homework everyday.
  – Read the section for the next lecture (reading only the first a few pages is fine).

• Friday
  – Review the whole week’s study.
  – Prepare the quiz on Monday.
CSE 21
Mathematics for Algorithm and System Analysis

Unit 1: Basic Count and List
Section 1: Lists with Repetitions
Learning Outcomes

• By the end of this lesson, you should be able to
  – Master two concepts: List and Set, and tell their differences.
  – Understand and apply two theorems of List and Set: $k$-lists with repetitions, Rule of Product.
  – Understand and apply two operations of List and Set: Cartesian Product, Lexicographic order.
Why do we need to learn them?

• List and Set are two core data structures for a collection of elements in programming.

• The operations of List and Set are useful to understand the problem space.
  - Test all possible parameter combinations for a same scientific program to know which is the best.
List and Set

• List
  – Definition: an ordered collection. Later, we will indicate whether the elements must be distinct.
  – Example: (apple, banana, pear), (banana, apple, pear), (apple, peach, apple)

• Set
  – Definition: a collection of distinct objects where order does not matter.
  – Example: \{apple, peach, pear\}, \{peach, apple, pear\}
List and Set (2)

• Equality Check
  – List: if and only if they have the same elements in the same order.
  – Set: if and only if they have the same elements.

• Application in programming
  – List: tasks have to be processed based on its order, often sequentially.
  – Set: No order required. Potential to be parallelized.
## Comparison of List and Set

<table>
<thead>
<tr>
<th></th>
<th>List</th>
<th>Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commonality</td>
<td>They are both collections of objects</td>
<td></td>
</tr>
<tr>
<td>Notation</td>
<td>parenthesis</td>
<td>brace (curly brackets)</td>
</tr>
<tr>
<td>Order</td>
<td>matters</td>
<td>does not matter</td>
</tr>
<tr>
<td>Repetition</td>
<td>allowed</td>
<td>not allowed</td>
</tr>
</tbody>
</table>
Multiset, Cardinality and \( k \)-list

- **Multiset**: a collection of objects (repeats allowed) where order does not matter. Example: \( \{1, 2, 2\} \)
  - Some multisets are not sets.

- **Cardinality (or size)** of a set/list
  - Definition: the number of elements in a set/list
  - Notation: \(|A|\)
  - Example: \(|\{a, b, c\}| = 3\)

- **\( k \)-list \( L \) from an \( n \)-set \( S \)**
  - \(|L| = k; |S| = n\)
  - Every element of \( L \) is from \( S \)
  - \( L \) could have repetition
Theorem 1 : \( k \)-list with Repetitions

• Example
  – How many ways to construct 2-list with repetition from a 3-set \( \{x, y, z\} \)?
    • \( 9 = 3^2 \) ways total : \( xx, xy, xz, yx, yy, yz, zx, zy, zz \)

• \textit{Theorem 1} (\( k \)-lists with repetitions): There are \( n^k \) ways to construct a \( k \)-list from an \( n \)-set.
Example Problems of Theorem 1

• How many 5-letter long English names we could have?
  – 5-letter names: 5-list
  – English alphabet number: 26, 26-set
  – Total number: \(26^5 = 11,881,376\) (over 11 million)

• How many possible phone numbers in U.S.?
  – Each phone number in U.S. has 10 digits: 10-list
  – There are 10 Arabic numbers: 10-set
  – Total number: \(10^{10}, 10\) billion.
Generalization of Theorem 1

• Construct $k$-list not from one set, but multiple sets.

• Example
  – How many ways to construct 2-list from English alphabet letters and Arabic numbers where the first element has to be letters and the second element has to be number, e.g., (a, 1)?
  – $26 \times 10$
Theorem 2 : Rule of Product

- **Premise**: structures to be constructed by making a sequence of $k$ choices
- **Problem**: how many structures we could have?
- **Conditions**
  - the $i$th choice can be made in $c_i$ ways, a number independent of what choices were made previously
  - each structure arises in exactly one way in this process
- **Predicate**: if the two conditions are both true, then the number of structures is $c_1 \times \cdots \times c_k$
Example Problem of Rule of Product

- A California license plate is made of one number 0-9 followed by three letters A-Z, and then 3 numbers 0-9.
- How many distinct license plates we can have?
  - Total 7 digits. We will make 7 choices. Each choice is independent.
  - 10 ways for each of the four numbers, 26 ways for each of the three letters.
  - No duplication can happen during this process.
  - Based on Rule of Product: $10 \times 26 \times 26 \times 26 \times 10 \times 10 \times 10 = 118,813,760,000$ (over 100 billion).
Definition 1: Cartesian Product

- If \( C_1, \ldots, C_k \) are sets, the *Cartesian product* of the sets is a set written \( C_1 \times \cdots \times C_k \) and consists of all \( k \)-lists \((x_1, \ldots, x_k)\) with \( x_i \in C_i \) for \( 1 \leq i \leq k \).

- Example
  - \( \{1, 2\} \times \{x\} \times \{a, b, c\} = \{(1, x, a), (1, x, b), (1, x, c), (2, x, a), (2, x, b), (2, x, c)\} \)
  - The lists can be written simply by \(1xa, 1xb, 1xc, 2xa, 2xb, 2xc\)
Apply Rule of Product to Cartesian Product

- Suppose $C_i$ is the collection of $i$th choices,
- Condition Check
  - Independence
  - No duplication in the process
- Conclusion
  \[ |C_1 \times \cdots \times C_k| = |C_1| \times \cdots \times |C_k| \]
Example 3: Counting Names

- In a planet, the alphabet contains only 5 letters: A, I, L, S and T.
  - All names are 6 letters long,
  - begin and end with consonants,
  - and contain two vowels which are not adjacent to each other,
  - adjacent consonants must be different.

- How many possible names are there?
  - The list begins with LALALS, LALALT, LALASL, LALAST, LALATL, LALATS, LALILS and ends with TSITAT, TSITIL, TSITIS, TSITIT.
Solution of Example 3 using Rule of Product

• The possible positions for the two vowels are (2, 4), (2, 5) and (3, 5).

• Each of these results in two isolated consonants and two adjacent consonants.

• Result is based on the following factors:
  1. choose the vowel locations (3 ways);
  2. choose the vowels (2 × 2 = 4 ways);
  3. choose the isolated consonants (3 × 3 = 9 ways);
  4. choose the adjacent consonants (3 × 2 = 6 ways).

• Based on Rule of Product, result is 3 × 4 × 9 × 6
Solution of Example 3 using Cartesian Product

• This construction can be interpreted as a Cartesian product.
• \( C_1 \) is the set of lists of possible positions for the vowels: \( \{(2,4), (2,5), (3,5)\} \): 3-set.
• \( C_2 \) is the set of lists of vowels in those positions: \( \{AA, AI, IA, II\} \): 4-set.
• \( C_3 \) is the set of lists of in two isolated consonants: \( \{LL, LS, LT, SL, SS, ST, TL, TS, TT\} \): 9-set.
• \( C_4 \) is the set of lists of two adjacent consonants: \( \{LS, LT, SL, ST, TL, TS\} \): 6-set.
• Cardinality of Cartesian Product: \( 3 \times 4 \times 9 \times 6 \).
• Example: \( ((2,5), IA, SS, ST) \) in the Cartesian product corresponds to the name SISTAS.
Definition 2: Lexicographic Order

• Definition: For Cartesian product $P = C_1 \times \cdots \times C_k$, the lexicographic order on $P$ is defined by saying that $(a_1, \ldots, a_k) <_L (b_1,\ldots,b_k)$ if and only if there is some $t \leq k$ such that: 1) $a_i = b_i$ for $i < t$ and 2) $a_t < b_t$.

• How to check? (a good algorithm exercise)
  1. From the first element, keep checking whether $a_i = b_i$, until we find the $t$th element that $a_t \neq b_t$;
  2. If $a_t < b_t$, then $(a_1, \ldots, a_k) <_L (b_1,\ldots,b_k)$; otherwise $(b_1,\ldots,b_k) <_L (a_1, \ldots, a_k)$

• Examples: $AB < AC$ ($t=2$), $ATBT < ATUA$ ($t=3$).
Lexicographic Order for Numbers and Words

• The lists in the definition of lexicographic order have the same length: \(k\)-lists.

• Lexicographic order for numbers
  – Add zero for numbers with less digits at the beginning
  – Example: \(12 = 012 < 123\)

• Lexicographic order for words (dictionary order)
  – Add space for words with less letters in the end
  – Example: \(\text{BAT} = \text{BAT}(\ ) < \text{BATA}\)
Homework and Pre-Reading Assignment

• Homework:
  – Exercise 1.1, 1.2, 1.6, 1.8, 1.9, in page CL-7 to CL-9

• For next class, please read Section 1 (CL-1 to CL-7) and Section 2 (CL-9 to CL-12) of our textbook.
  – Try to understand and summarize the differences between List and Set from their definition, applicable operation and examples.
  – Try to understand and summarize the differences of List with and without repetition, and how these differences affect their operations.