CS 441 Discrete Mathematics for CS
Lecture 14

Counting

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Course administration

- **Homework 6 due today**
- **Homework 7 is out and due on October 29, 2009**

Course web page:
http://www.cs.pitt.edu/~milos/courses/cs441/

Counting

- Assume we have a set of **objects with certain properties**
- **Counting** is used to determine **the number of these objects**

Examples:

- Number of available phone numbers with 7 digits in the local calling area
- Number of possible match starters (football, basketball) given the number of team members and their positions

Basic counting rules

- Counting problems may be very hard, not obvious
- **Solution:**
  - **simplify the solution by decomposing the problem**
- **Two basic decomposition rules:**
  - **Product rule**
    - A count decomposes into a sequence of dependent counts ("each element in the first count is associated with all elements of the second count")
  - **Sum rule**
    - A count decomposes into a set of independent counts ("elements of counts are alternatives")

Product rule

A count can be broken down into a sequence of dependent counts

- "each element in the first count is associated with all elements of the second count"

Example:

- Assume an auditorium with a seat labeled by a letter and numbers in between 1 to 50 (e.g. A23). We want the total number of seats in the auditorium.
- 26 letters and 50 numbers
- How to count?

Product rule

A count can be broken down into a sequence of dependent counts

- "each element in the first count is associated with all elements of the second count"

Example:

- Assume an auditorium with a seat labeled by a letter and numbers in between 1 to 50 (e.g. A23). We want the total number of seats in the auditorium.
- 26 letters and 50 numbers
- How to count?
- **One solution: write down all seats (objects) and count them**

A-1 A-2 A-3 …A-50 B-1… Z-49 Z-50

1 2 3 50 51 … (n-1) n ← eventually we get it

Product rule

A count can be broken down into a sequence of dependent counts

- "each element in the first count is associated with all elements of the second count"

Example:

- assume an auditorium with a seat labeled by a letter and numbers in between 1 to 50 (e.g. A23). We want the total number of seats in the auditorium.
- 26 letters and 50 numbers
- A better solution?

Product rule

A count can be broken down into a sequence of dependent counts

- "each element in the first count is associated with all elements of the second count"

Example:

- assume an auditorium with a seat labeled by a letter and numbers in between 1 to 50 (e.g. A23). We want the total number of seats in the auditorium.
- 26 letters and 50 numbers
- A better solution?
- For each letter there are 50 numbers
- So the number of seats is 26*50 = 1300
- **Product rule:** number of letters * number of integers in [1,50]

Product rule

A count can be broken down into a sequence of dependent counts

- "each element in the first count is associated with all elements of the second count"

- **Product rule:** If a count of elements can be broken down into a sequence of dependent counts where the first count yields $n1$ elements, the second $n2$ elements, and kth count nk elements, by the product rule the total number of elements is:
    - $n = n1*n2* \ldots * nk$

Product rule

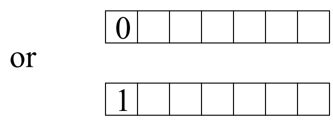
Example:

- How many different bit strings of length 7 are there?
    - E.g. 1011010
- Is it possible to decompose the count problem and if yes how?

Product rule

Example:

- How many different bit strings of length 7 are there?
  - E.g. 1011010
- Is it possible to decompose the count problem and if yes how?
- **Yes.**
  - Count the number of possible assignments to bit 1



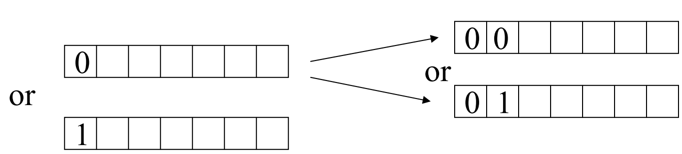
Product rule

Example:

- How many different bit strings of length 7 are there?
  - E.g. 1011010
- Is it possible to decompose the count problem and if yes how?

Yes.

- Count the number of possible assignments to bit 1
- For the first bit assignment (say 0) count assignments to bit 2



Total assignments to first 2 bits: 2*2=4

Product rule

Example:

- How many different bit strings of length 7 are there?
  - E.g. 1011010
- Is it possible to decompose the count problem and if yes how?
- **Yes.**
  - Count the number of possible assignments to bit 1
  - For the specific first bit count possible assignments to bit 2
  - For the specific first two bits count assignments to bit 3
  - Number of assignments to the first 3 bits: ?

Product rule

Example:

- How many different bit strings of length 7 are there?
  - E.g. 1011010
- Is it possible to decompose the count problem and if yes how?
- **Yes.**
  - Count the number of possible assignments to bit 1
  - For the specific first bit count possible assignments to bit 2
  - For the specific first two bits count assignments to bit 3
  - Number of assignments to the first 3 bits: 2*2*2=8

Product rule

Example:

- How many different bit strings of length 7 are there?
  - E.g. 1011010
- Is it possible to decompose the count problem and if yes how?
- **Yes.**
  - Count the number of possible assignments to bit 1
  - For the specific first bit count possible assignments to bit 2
  - For the specific first two bits count assignments to bit 3
  - Gives a sequence of n dependent counts and by the product rule we have:

    $\mathbf{n = 2 * 2 * 2 * 2 * 2 * 2 * 2 = 2^7}$

Product rule

Example:

The number of subsets of a set S with k elements.

- How to count them?
- **Hint:** think in terms of bitstring representation of a set?
- Assume each element in S is assigned a bit position.
- If A is a subset it can be encoded as a bitstring: if an element is in A then use 1 else put 0
- How many different bitstrings are there?
  - $n = \underbrace{2 * 2 * \ldots 2}_{\text{k bits}} = 2^k$

Sum rule

A count decomposes into a set of independent counts

- "elements of counts are alternatives", they do not depend on each other

Example:

- You need to travel in between city A and B. You can either fly, take a train, or a bus. There are 12 different flights in between A and B, 5 different trains and 10 buses. How many options do you have to get from A to B?

Sum rule

A count decomposes into a set of independent counts

- "elements of counts are alternatives", they do not depend on each other

Example:

- You need to travel in between city A and B. You can either fly, take a train, or a bus. There are 12 different flights in between A and B, 5 different trains and 10 buses. How many options do you have to get from A to B?
- We can take only one type of transportation and for each only one option. The number of options:
  - n = 12+5+10

Sum rule:

- n = number of flights + number of trains + number of buses

Sum rule

A count decomposes into a set of independent counts

- "elements of counts are alternatives"

- **Sum rule:** If a count of elements can be broken down into a set of independent counts where the first count yields $n1$ elements, the second $n2$ elements, and kth count nk elements, by the sum rule the total number of elements is:
  - $n = n1+n2+ \ldots + nk$

Beyond basic counting rules

- **More complex counting problems** typically require a combination of the sum and product rules.

Example: A login password:

- The minimum password length is 6 and the maximum is 8. The password can consist of either an uppercase letter or a digit. There must be at least one digit in the password.
- How many different passwords are there?

Beyond basic counting rules

Example: A password for the login name.

- The minimum password length is 6 and the maximum is 8. The password can consist of either an uppercase letter or a digit. There must be at least one digit in the password.
- How to compute the number of possible passwords?

Step 1:

- The password we select has either 6,7 or 8 characters.
- So the total number of valid passwords is by the sum rule:
  - P = P6+P7+P8

    The number of passwords of length 6,7 and 8 respectively

Beyond basic counting rules

Step 1:

- The password we select has either 6,7 or 8 characters.
- So the total number of valid passwords is by the sum rule:
  - P = P6+P7+P8

    The number of passwords of length 6,7 and 8 respectively

Step 2

- Assume passwords with 6 characters (upper-case letters):
- How many are there?
- If we let each character to be at any position we have:
  - P6-nodigits = 26^6 different passwords of length 6

Beyond basic counting rules

Step 1:

- The password we select has either 6,7 or 8 characters.
- So the total number of valid passwords is by the sum rule:
  - P = P6+P7+P8

    The number of passwords of length 6,7 and 8 respectively

Step 2

- Assume passwords with 6 characters (either digits + upper case letters):
- How many are there?
- If we let each character to be at any position we have:
  - P6-all = $(26+10)^6=(36)^6$ different passwords of length 6

Beyond basic counting rules

Step 2

But we must have a password with at least one digit. How to account for it?

A trick. Split the count of all passwords of length 6 into to two mutually exclusive groups:

- **P6-all = P6-digits + P6-nodigits**

1. P6-digits – count when the password has one or more digits
2. P6-nodigits – count when the password has no digits

- We know how to easily compute P6-all and P6-nodigits
  - **P6-all = 36^6 and P6-nodigits = 26^6**
  - Then **P6-digits = P6-all – P6-nodigits**

Beyond basic counting rules

Step 1:

the total number of valid passwords is by the sum rule:

- **P = P6+P7+P8**
- The number of passwords of length 6,7 and 8 respectively

Step 2

The number of valid passwords of length 6:

P6= P6-digits = P6-all – P6-nodigits

$= 36^6 - 26^6$

Analogically:

P7= P7-digits = P7-all – P7-nodigits

$= 36^7 - 26^7$

P8= P8-digits = P8-all – P8-nodigits

$= 36^8 - 26^8$

Inclusion-Exclusion principle

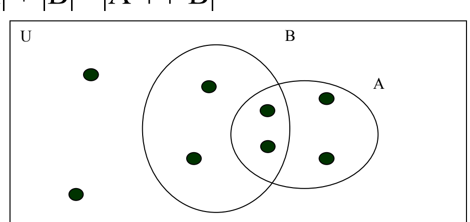
Used in counts where the decomposition yields two count tasks with overlapping elements

- If we used the sum rule some elements would be counted twice

Inclusion-exclusion principle: uses a sum rule and then corrects for the overlapping elements.

We used the principle for the cardinality of the set union.

- $|A \cup B| = |A| + |B| - |A \cap B|$



Inclusion-exclusion principle

Example: How many bitstrings of length 8 start either with a bit 1 or end with 00?

- It is easy to count **strings that start with 1**:
- How many are there? 2^7
- It is easy to count the **strings that end with 00**.
- How many are there? 2^6
- Is it OK to add the two numbers to get the answer? $2^7 + 2^6$
- **No. Overcount**. There are some strings that can both start with 1 and end with 00. These strings are counted in twice.
- How to deal with it? How to correct for overlap?
- How many of strings were counted twice? 2^5 (1 xxxxx 00)
- Thus we can correct for the overlap simply by using:
- $2^7 + 2^6 - 2^5 = 128 + 64 - 32 =$ **160**

Tree diagrams

Tree: is a structure that consists of a root, branches and leaves.

- Can be useful to represent a counting problem and record the choices we made for alternatives. The count appears on the leaf nodes.

Example:

What is the number of bit strings of length 4 that do not have two consecutive ones.

Tree diagrams

Example:

What is the number of bit strings of length 4 that do not have two consecutive ones?