DISCUSSION #6
FRIDAY MAY 10TH 2007

Sophie Engle (Teacher Assistant)
ECS20: Discrete Mathematics

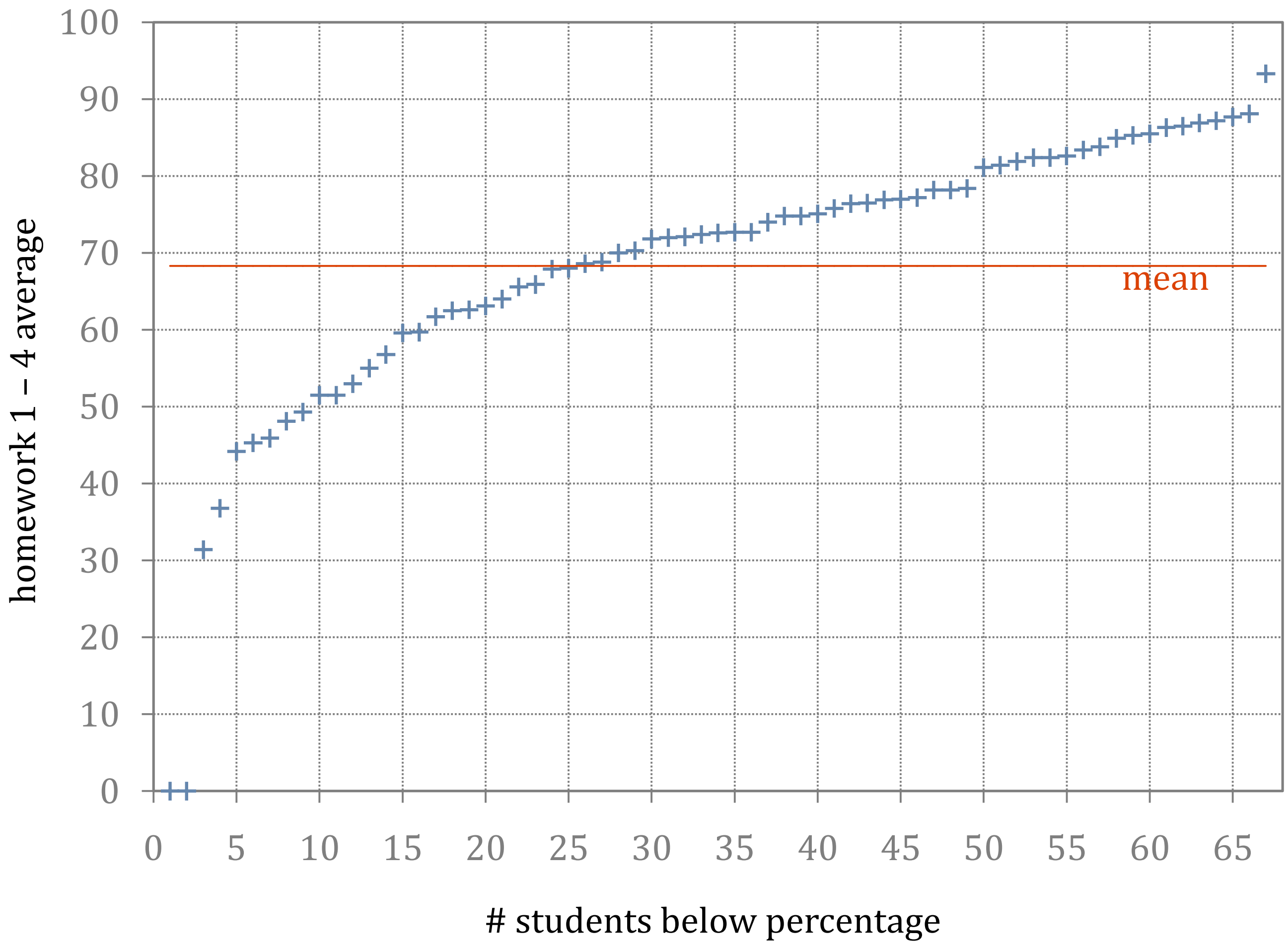
Homework Announcements

Homework Announcements

- Every homework is worth equal weight!
  - A homework worth 50 points affects your score equally as one worth 90 points.
  - Don’t skip small homework assignments!

- Homework 4 graded
- Homework 5 picked up
- Homework 6 posted
  - Posted sometime Thursday evening

Homework Average



Improving Your Homework Score

- How not to lose points:
  - READ THE DIRECTIONS!
    - Only awarded for work on problems that are graded!
      - You get 0 points for completing a wrong problem.
    - Make sure you answer the actual question.
      - If you are asked to provide the value of an exponent n, write "$n = 5$" on your homework, not $\mathcal{O}(x^5)$.

Improving Your Homework Score

- How not to lose points:
  - TURN IN HOMEWORK ON TIME!
    - This means BY 4:00PM. This is a DEADLINE, not an approximation.
    - Will not be making any future exceptions for late homework except in extreme cases (i.e. emergencies).

Improving Your Homework Score

- How to gain points:
  - TRY EVERY PROBLEM!
    - 99.99% chance that there will be a problem selected from every section.
    - Usually one easy and one difficult question is selected.
    - Therefore, at least do all of the easy problems from every section before tackling harder ones.

Improving Your Homework Score

- How to gain points:
  - INCLUDE YOUR WORK!
    - Correct answers get full credit…
    - Extremely rare that everyone gets all questions correct.

General Announcements

Announcements

- John will be leading discussion next week!
  - I will be out of town from May 18th – 20th

- The final score will be scaled.
  - Not intermediate scores however!

Discussion Outline

- Homework Announcements
- Recursion Topics:
  - Repeated Squares Algorithm
    - Not in your book!
  - Quick Sort Algorithm
- Common Midterm Mistakes
- Homework 4 Comments
- Homework 5 Comments
- No Homework 6 Hints :(

Repeated Squares Algorithm

Repeated Squares Algorithm

- Problem:
  - Compute x to the power n (i.e. find x^n)
  - Multiplying x a total of n times is expensive!

- Algorithm:
  - "Repeated Squaring"
    - Requires only $\Theta(\log n)$ steps
    - More information at: www.nist.gov/dads/HTML/repeatedSquaring.html

Repeated Squares: Toolset

- To understand this algorithm:

  - Must be able to convert to binary
    - $13 = 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = (1101)_b$

  - Must know basic algebra rules
    - $x^{a+b} = x^a * x^b$
    - $(x^a)^2 * (x^b)^2 = (x^a * x^b)^2$

  - Must understand recursion

Repeated Squares: Basic Idea

- Suppose we want to find x^{13}:
  - Step 1: Convert exponent to binary
    $= x^{1101_b}$
  - Step 2: Expand binary representation
    $= x^{1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0}$
  - Step 3: Break into product terms
    $= x^{1\cdot 2^3}\ x^{1\cdot 2^2}\ x^{0\cdot 2^1}\ x^{1\cdot 2^0}$
  - Step 4: Simplify
    $= x^8\, x^4\, x^0\, x^1 = x^8\, x^4\, x^1$

Repeated Squares: Basic Idea

- Suppose we want to find x^{13}:
  - Step 5: Observe that we can reuse results

    $x^2 = (\ x\ \ )^2 = x\ \ * x$ $\quad$ $x^4 = (\ x^2\ )^2 = x^2 * x^2$

    $x^8 = (\ x^4\ )^2 = x^4 * x^4$

  - Step 6: Pull it all together and collapse squares

    $= x^8 * x^4 * x^1$

    $= (x^4)^2 * (x^2)^2 * x$

    $= (x^4 * x^2)^2 * x$

    $= (\ (x^2)^2 * x^2\ )^2 * x$

    $= (\ (x^2 * x)^2\ )^2 * x$

Repeated Squares: Basic Idea

- Suppose we want to find x^{13}:
  - Recap:

$$x^{13} = x^8 * x^4 * x^1 = (\,(x^2 * x)^2\,)^2 * x$$

  - Uses only 5 multiplications!

$$\begin{aligned}
\text{Find } x^2\text{:}\quad & \text{result} = x * x \\
\text{Find } x^2 * x\text{:}\quad & \text{result} = \text{result} * x \\
\text{Find } (x^2 * x)^2\text{:}\quad & \text{result} = \text{result} * \text{result} \\
\text{Find } (\,(x^2 * x)^2\,)^2\text{:}\quad & \text{result} = \text{result} * \text{result} \\
\text{Find } (\,(x^2 * x)^2\,)^2 * x\text{:}\quad & \text{result} = \text{result} * x
\end{aligned}$$

Repeated Squares: Basic Idea

- Suppose we want to find x^{13}:
  - For example, let $x = 3$:

result $= x * x$	result $= 3 * 3 = 9$
result $=$ result $* x$	result $= 9 * 3 = 27$
result $=$ result $*$ result	result $= 27 * 27 = 729$
result $=$ result $*$ result	result $= 729 * 729 = 531441$
result $=$ result $* x$	result $= 531441 * 3 = 1594323$

$$\mathbf{3^{13} = 1{,}594{,}323}$$

  - Work out on your own:
    - $x^{11} = x^8 * x^2 * x^1 = (\,(x^2)^2 * x\,)^2 * x$

Repeated Squares: Algorithm

- To take x to the power of n:
  - Convert n into its binary representation b
  - Initialize result to 1.
  - For each digit of b from left to right:
    - If the digit is 0, then result $=$ result $*$ result.
    - If the digit is 1, then result $=$ result $*$ result $* x$
  - Return result

Repeated Squares: Algorithm

- Repeated Squares Algorithm:
  - Iterative Version
  - Recursive Version

- Working Code Examples in R:
  - R is a free language and environment for statistical computing and graphics
  - Don’t worry about learning R!
  - Examples just given to show working code.

Repeated Squares: Algorithm

- Iterative Repeated Squares Algorithm

```
iPower <- function( x, n )
{
    b = int2bin( n )

    digits = length( b )
    result = 1

    for( i in 1:digits )
    {
        if( b[i] == 0 ) result = result * result
        else result = result * result * x
    }

    return( result )
}
```

Repeated Squares: Algorithm

- Recursive Repeated Squares Algorithm

```
rPower <- function( x, n, i=1, result=1 )
{
    b = int2bin( n )

    digits = length( b )

    if( i > digits ) return( result )

    if( b[i] == 0 ) result = result * result
    else result = result * result * x

    return( rPower( x, n, i+1, result ) )
}
```

Repeated Squares: Algorithm

- Which version is better?
  - Iterative Version?
  - Recursive Version?

- Better to write **iterative** rather than recursive algorithms when possible.

Quick Sort Algorithm

Quick Sort Algorithm

- Efficient Sorting Algorithm
  - On average makes $\Theta(n \log n)$ comparisons
  - Worst case makes $\Theta(n^2)$ comparisons

- Recursive Implementation
- Iterative Implementation

- Visualization:
  - www.wanginator.de/studium/applets/quicksort_en.html

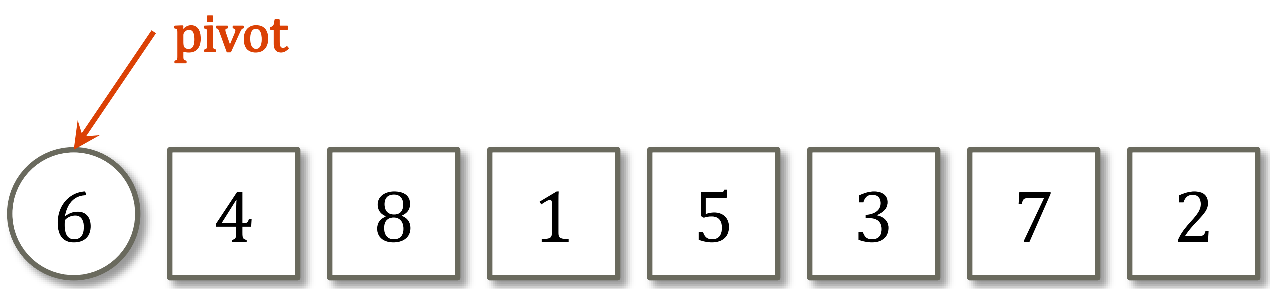
Quick Sort: General Idea

- Given a list of numbers $\{ a_1, a_2, \ldots a_n \}$
  - Choose a pivot, a_1
  - For every element a_i, where $1 < i \leq n$
    - If $a_i < a_1$ then place a_i in sublist_1
    - Else, place a_i in sublist_2
  - Then we have $\{ \text{sublist}_1, a_1, \text{sublist}_2 \}$
  - Repeat algorithm on sublist_1 and sublist_2
  - Complete once all sublists are of size 1

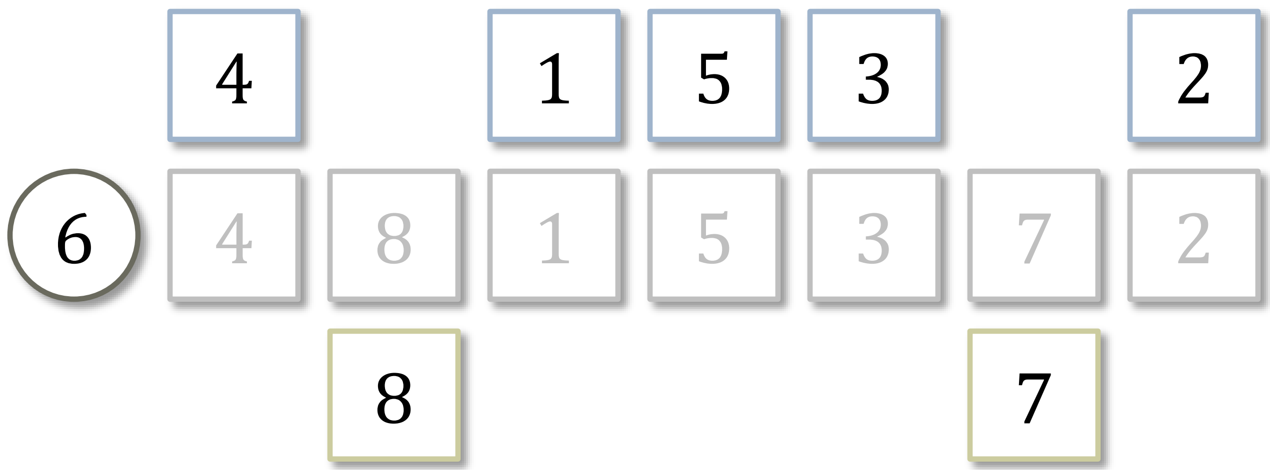
Quick Sort: General Idea

6	4	8	1	5	3	7	2

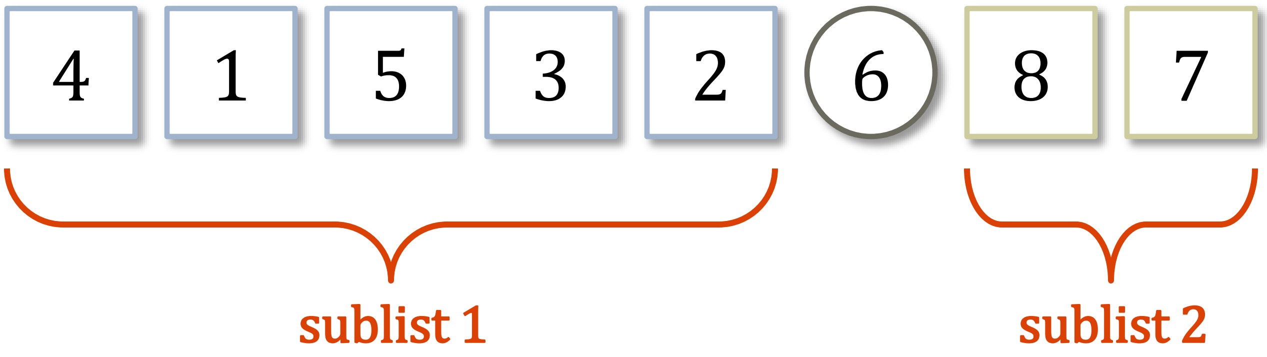
Quick Sort: General Idea



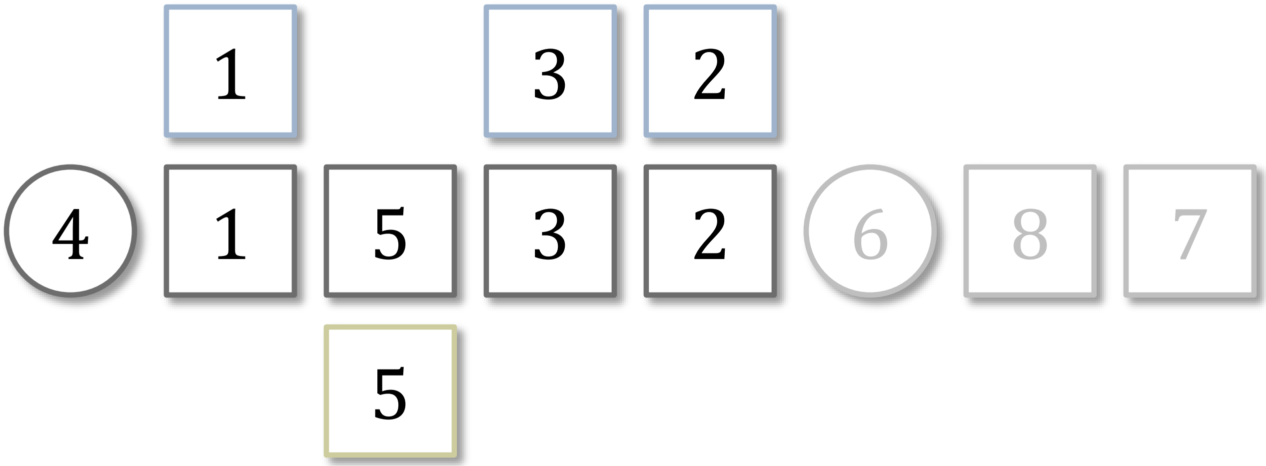
Quick Sort: General Idea



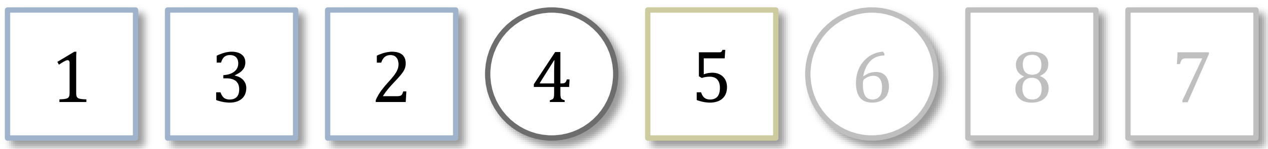
Quick Sort: General Idea



Quick Sort: General Idea



Quick Sort: General Idea



keep repeating until all squares are circles

Quick Sort: Recursive Algorithm


```
qSort <- function( a )
{
    n = length( a )
    if( n <= 1 ) return( a )

    pivot = a[1]
    less = c() # all elements < pivot
    more = c() # all elements >= pivot

    for( i in 2:n )
    {
        if( a[i] < pivot ) less = c( less, a[i] )
        else more = c( more, a[i] )
    }

    less = qSort( less )
    more = qSort( more )
    return( c( less, pivot, more ) )
}
```

Common Midterm Mistakes

Common Midterm Mistakes

- Overuse of $\rightarrow$ versus $\wedge$
- Mixing set notation with quantifier notation
  - Symbols $\{ : \}$ are not to be used with quantifiers
- Confusion on countable and uncountable sets
- Confusion using $\{ \}$ versus $\{\{ \}\}$ in set notation

Common Midterm Mistakes

- Translate: "There exists a lion that drinks tea."
  - Let $L(x)$ indicate that x is a lion.
  - Let $T(x)$ indicate that x drinks tea.
  - Does $\exists x(\ L(x) \rightarrow T(x)\ )$ work?
    - Nope.
    - Try plugging in a giraffe for x!
  - Should have $\exists x(\ L(x) \wedge T(x)\ )$ instead.

Common Midterm Mistakes

- Countable Set:
  - Finite, or one-to-one correspondence with $\mathbb{N}$
- Uncountable Set:
  - Infinite, and no such correspondence exists

Common Midterm Mistakes

- Set of Binary Strings:
  - Countable!
  - Each binary string represents a number in $\mathbb{N}$

- Set of all books writeable in English language:
  - Countable!
  - Translatable into a finite binary string.

Common Midterm Mistakes

- Using Set Notation:
  - The element of set $\{1\}$ is the number 1.
  - The element of set $\{\{1\}\}$ is the set $\{1\}$.

  - Therefore, $1 \notin \{\{1\}\}$.
  - Therefore, $\{1\} \nsubseteq \{\{1\}\}$.

Homework 4 Review

Solutions will be removed before being posted on the TA website.

See my.ucdavis.edu for full solutions!

Section 3.1 #58

- WALL OF TEXT
  - Difficult to parse and understand.
  - Will go over how to tell what we need to know, and what we need to do for this problem.

Section 3.1 #58

- First, you were given a paragraph of text describing the situation:

> Suppose we have s men $m_1, m_2, \ldots, m_s$ and s women $w_1, w_2, \ldots, w_s$. We wish to match each person with a member of the opposite gender. Furthermore, suppose that each person ranks, in order of preference, with no ties, the people of the opposite gender. We say that an assignment of people of opposite genders to form couples is **stable** if we cannot find a man m and a woman w who are not assigned to each other such that m prefers w over his assigned partner and w prefers m to her assigned partner.

Section 3.1 #58

- First, you were given a paragraph of text describing the situation:

> Suppose we have s men $m_1, m_2, \ldots, m_s$ and s women $w_1, w_2, \ldots, w_s$. We wish to match each person with a member of the opposite gender. Furthermore, suppose that each person ranks, in order of preference, with no ties, the people of the opposite gender. We say that an assignment of people of opposite genders to form couples is **stable** if we cannot find a man m and a woman w who are not assigned to each other such that m prefers w over his assigned partner and w prefers m to her assigned partner.

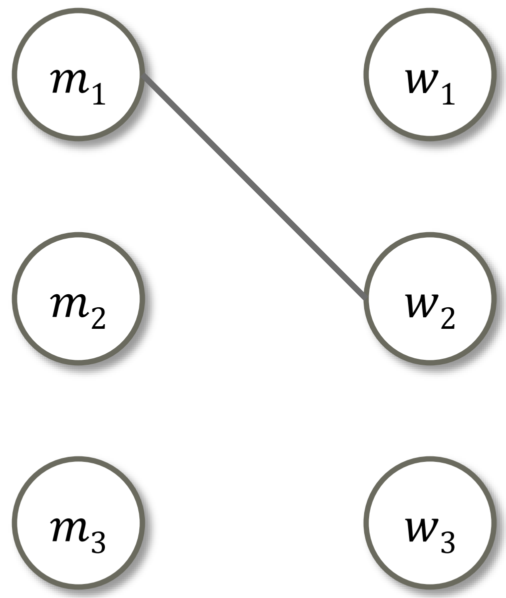
So, a **match** is a pair (m_i, w_j). (i.e., a couple.)

Section 3.1 #58

- First, you were given a paragraph of text describing the situation:

> Suppose we have s men $m_1, m_2, \ldots, m_s$ and s women $w_1, w_2, \ldots, w_s$. We wish to match each person with a member of the opposite gender. Furthermore, suppose that each person ranks, in order of preference, with no ties, the people of the opposite gender. We say that an assignment of people of opposite genders to form couples is **stable** if we cannot find a man m and a woman w who are not assigned to each other such that m prefers w over his assigned partner and w prefers m to her assigned partner.

match (m_1, w_2)



Section 3.1 #58

- First, you were given a paragraph of text describing the situation:

> Suppose we have s men $m_1, m_2, \ldots, m_s$ and s women $w_1, w_2, \ldots, w_s$. We wish to match each person with a member of the opposite gender. Furthermore, suppose that each person ranks, in order of preference, with no ties, the people of the opposite gender. We say that an assignment of people of opposite genders to form couples is **stable** if we cannot find a man m and a woman w who are not assigned to each other such that m prefers w over his assigned partner and w prefers m to her assigned partner.

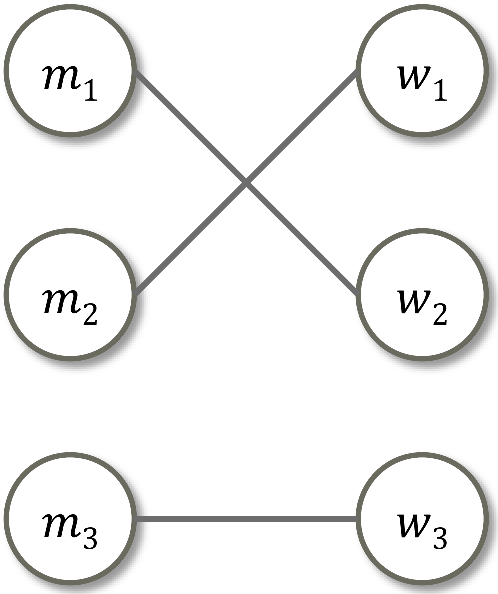
An assignment is a set of matches (a set of couples).

Section 3.1 #58

- First, you were given a paragraph of text describing the situation:

> Suppose we have s men $m_1, m_2, \ldots, m_s$ and s women $w_1, w_2, \ldots, w_s$. We wish to match each person with a member of the opposite gender. Furthermore, suppose that each person ranks, in order of preference, with no ties, the people of the opposite gender. We say that an assignment of people of opposite genders to form couples is **stable** if we cannot find a man m and a woman w who are not assigned to each other such that m prefers w over his assigned partner and w prefers m to her assigned partner.

assignment



Section 3.1 #58

- First, you were given a paragraph of text describing the situation:

> Suppose we have s men $m_1, m_2, \ldots, m_s$ and s women $w_1, w_2, \ldots, w_s$. We wish to match each person with a member of the opposite gender. Furthermore, suppose that each person ranks, in order of preference, with no ties, the people of the opposite gender. We say that an assignment of people of opposite genders to form couples is **stable** if we cannot find a man m and a woman w who are not assigned to each other such that m prefers w over his assigned partner and w prefers m to her assigned partner.

Therefore…

Section 3.1 #58

- An assignment is a set of matches.
- An assignment is **stable** if we can NOT find:
  - A pair (m_i, w) such that all of the following hold:
    - m_i and w_j are not matched
    - m_i prefers w_j to his partner
    - w_j prefers m_i to her partner
  - If any of these is false, then we are still okay.

Section 3.1 #58

- Second, you are given a paragraph for the actual exercise you must complete:

Suppose we have 3 men m_1,m_2, and m_3 and 3 women w_1, w_2, and w_3. Furthermore, suppose that the preference rankings of the men for the 3 women, from highest to lowest, are m_1: w_3, w_1, w_2; m_2: w_1, w_2, w_3; m_3: w_2, w_3, w_1; and the preference rankings of the women for the three men, from highest to lowest, are w_1: m_1, m_2, m_3; w_2: m_2, m_1, m_3; w_3: m_3, m_2, m_1. For each of the six possible assignments of men and women to form three couples, determine whether this assignment is stable.

preferences

	m_1	m_2	m_3
Top:	w_3	w_1	w_2
Mid:	w_1	w_2	w_3
Last:	w_2	w_3	w_1

Section 3.1 #58

- Second, you are given a paragraph for the actual exercise you must complete:

Suppose we have 3 men m_1,m_2, and m_3 and 3 women w_1, w_2, and w_3. Furthermore, suppose that the preference rankings of the men for the 3 women, from highest to lowest, are m_1: w_3, w_1, w_2; m_2: w_1, w_2, w_3; m_3: w_2, w_3, w_1; and the preference rankings of the women for the three men, from highest to lowest, are w_1: m_1, m_2, m_3; w_2: m_2, m_1, m_3; w_3: m_3, m_2, m_1. For each of the six possible assignments of men and women to form three couples, determine whether this assignment is stable.

preferences

	w_1	w_2	w_3
Top:	m_1	m_2	m_3
Mid:	m_2	m_1	m_2
Last:	m_3	m_3	m_1

Section 3.1 #58

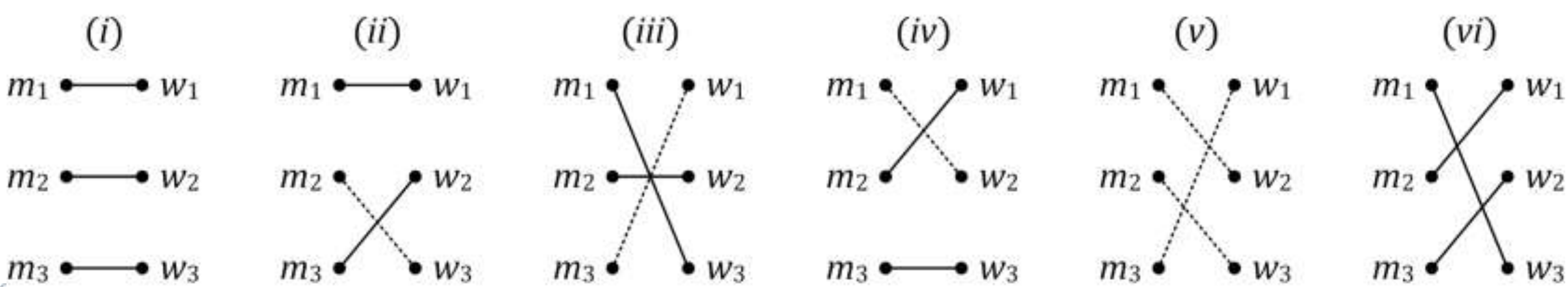
- Second, you are given a paragraph for the actual exercise you must complete:

> Suppose we have 3 men m_1,m_2, and m_3 and 3 women w_1, w_2, and w_3. Furthermore, suppose that the preference rankings of the men for the 3 women, from highest to lowest, are m_1: w_3, w_1, w_2; m_2: w_1, w_2, w_3; m_3: w_2, w_3, w_1; and the preference rankings of the women for the three men, from highest to lowest, are w_1: m_1, m_2, m_3; w_2: m_2, m_1, m_3; w_3: m_3, m_2, m_1. For each of the six possible assignments of men and women to form three couples, determine whether this assignment is stable.

Finally! This is what we need to do for this problem.

Section 3.1 #58

- Therefore:
  - For 6 possible assignments:
    - Must provide “Yes this is stable” or
    - Must provide “No this is not stable”
- First step? List the assignments!

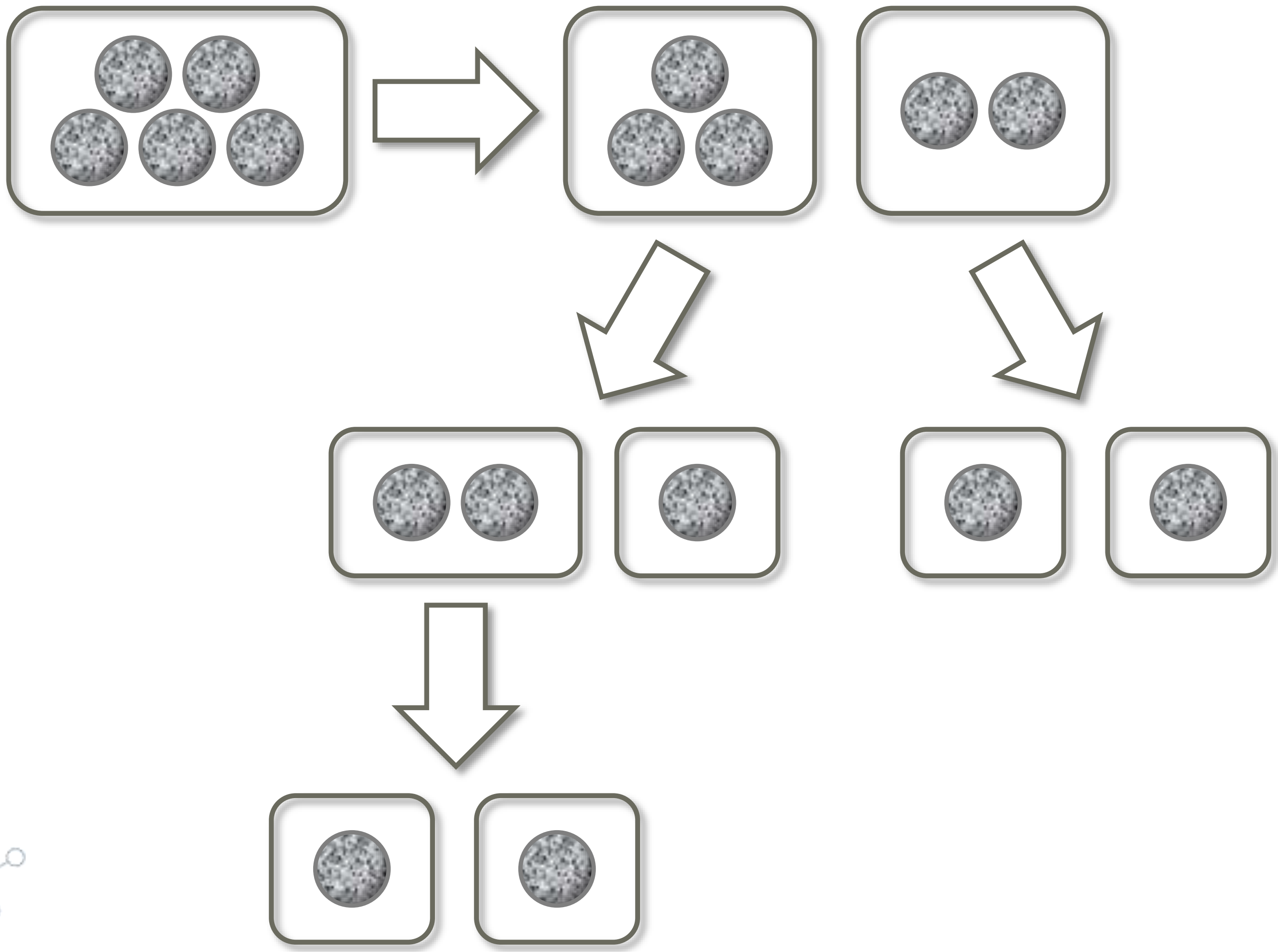


Homework 5 Review

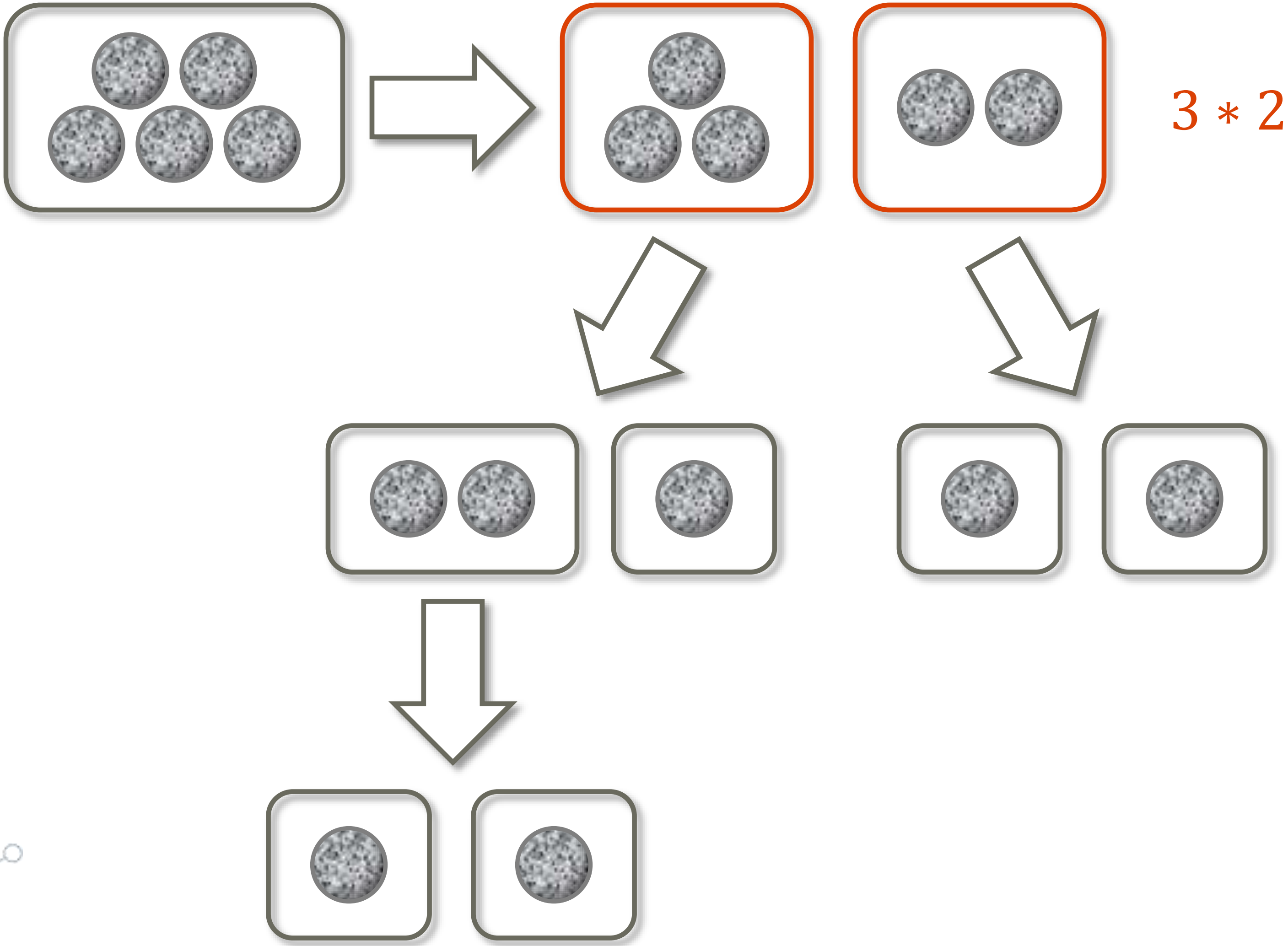
Strong versus Normal Induction

- Only major difference:
  - Inductive Hypothesis is different!
    - Normal induction:
      Assume $P(k)$ is true.
    - Strong induction:
      Assume $P(k)$ is true, and $P(k-1)$ is true, and …, and $P(2)$ is true and $P(1)$ is true.

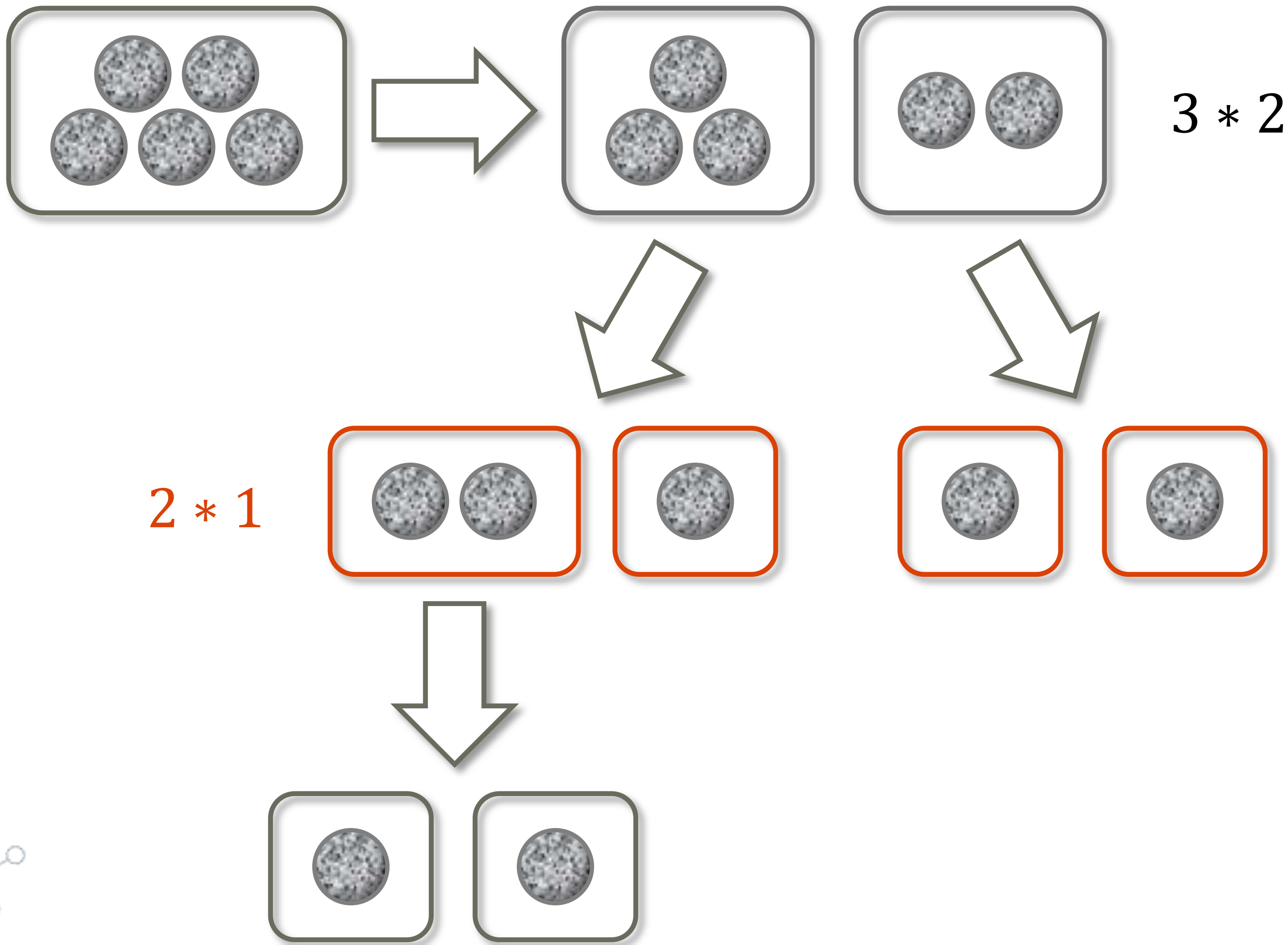
Section 4.2 #14



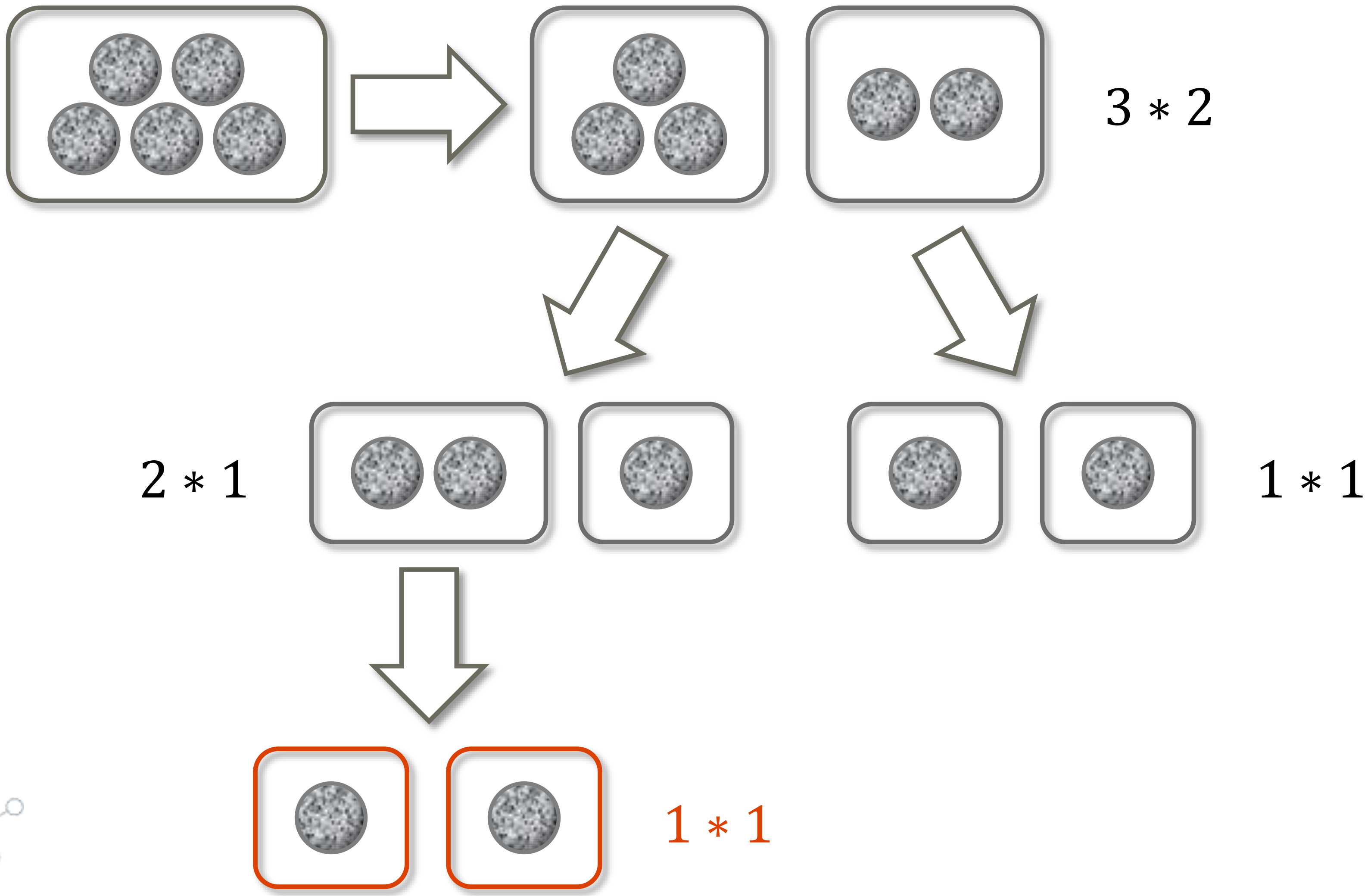
Section 4.2 #14



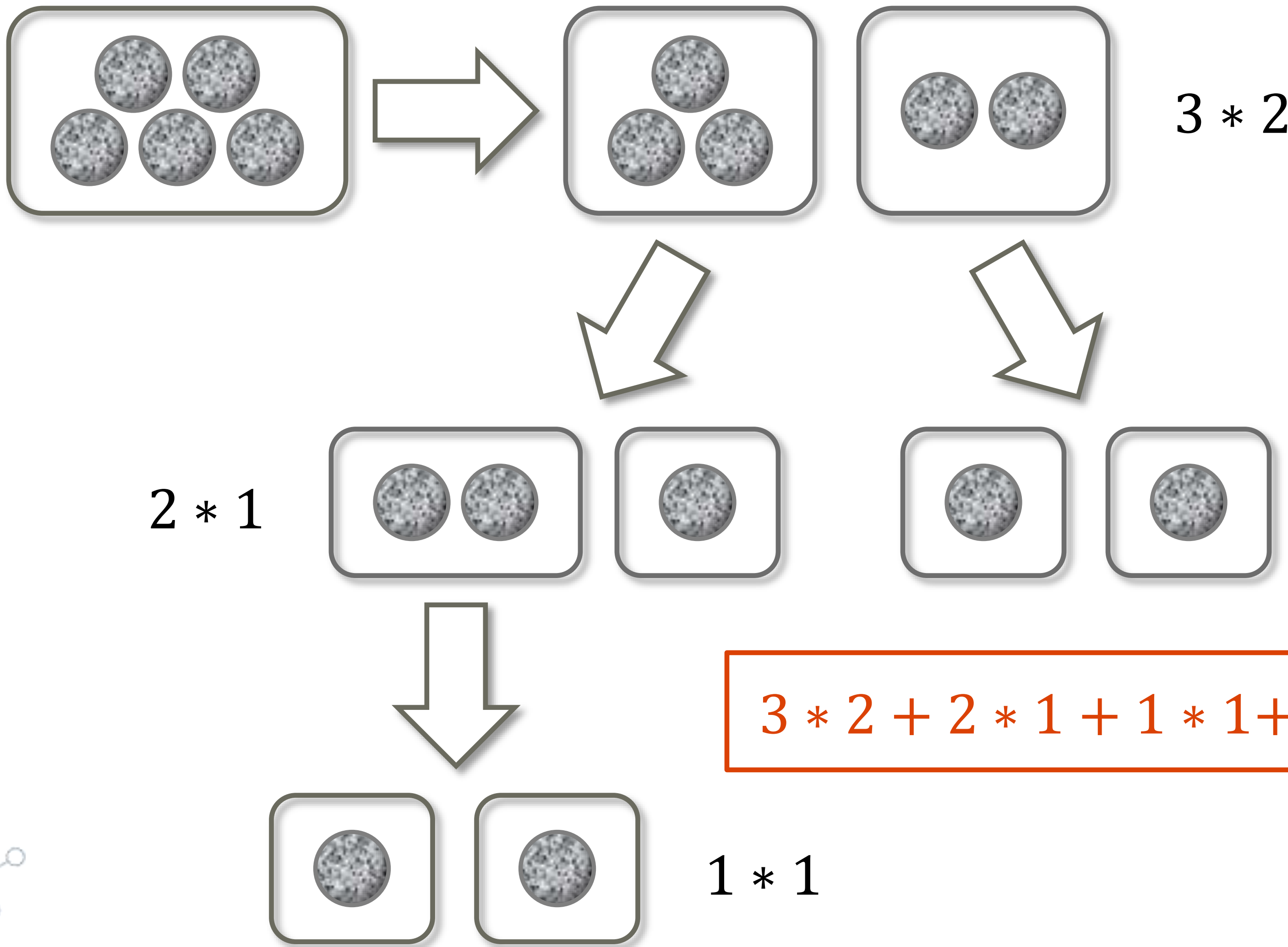
Section 4.2 #14



Section 4.2 #14



Section 4.2 #14



$$\boxed{3 * 2 + 2 * 1 + 1 * 1 + 1 * 1}$$

Section 4.2 #14

- Conjecture:
  - $P(n)$: No matter how you split the piles, the sum of products is $n (n - 1) / 2$
- Basis Case:
  - $P(1)$: One stone may be split $1 (1 - 1) / 2 = 0$ ways is true since there is no way to split a single stone.
- Inductive Hypothesis:
  - Assume $P(1)$, $P(2)$, ..., and $P(k)$ is true.