

Chapter 2: Performance

Data Structures in Java: From Abstract Data Types to the Java Collections Framework
by Simon Gray

A look at space/time complexity



Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Criteria

- Given the two algorithms that perform the same task, what criteria would you use to select one to use in a program?

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Consider

```

public NonNegIntSet union(NonNegIntSet s){
    NonNegIntSet result=new NonNegIntSet();
    int min=Math.min(this.contents.length,
        s.contents.length);
    for(int i=0;i<min; i++)
        if (this.isElement(i)||s.isElement(i))
            result.insert(i);
    if(min==this.contents.length) {
        for(int i=min;i<this.contents.length; i++)
            if (this.isElement(i)) result.insert(i);
        }
    else
        for(int i=min;i<s.contents.length; i++)
            if (s.isElement(i)) result.insert(i);
    return result;
}

public NonNegIntSet union2(NonNegIntSet s){
    NonNegIntSet result=this.copy();
    for (int i=0;i<s.contents.length;i++)
        if(s.isElement(i)) result.insert(i);
    return result;
}
    
```

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Strategy

- To compare the two methods what strategy would you employ?
- How does a given criteria influence the strategy adopted?

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Analysis and Measurement

An algorithm's performance can be described by its:

- **time complexity** – how long it takes to execute. In general, our preference is for shorter execution times rather than longer ones
- **space complexity** – how much additional space the algorithm needs. If the amount of additional memory needed is a function of the algorithm's input size, we prefer "smaller" functions

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Time Complexity: Count Instructions

	Statements	Cost
1	float findAvgLD(int [ja, int n) {	
2	float sum = 0;	1
3	int count = 0;	1
4	while (count < n) {	n
5	sum += grades[count];	$n-1$
6	count++;	$n-1$
7	}	
8	if (n > 0)	1
9	return sum / n;	
10	else	1
11	return 0.0f;	
12	}	
	TOTAL	$3n + 2$

How many times will each instruction execute?

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Theta (Θ) Notation

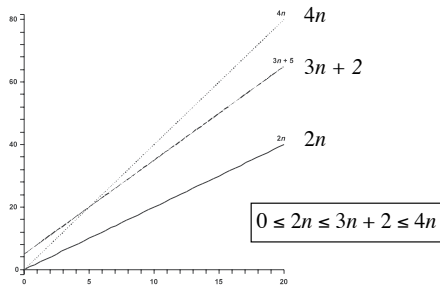
A function $f(n)$ is $\Theta(g(n))$ if there are positive constants c_1 , c_2 , and n_0 such that $0 \leq c_1g(n) \leq f(n) \leq c_2g(n)$ for all $n \geq n_0$.

This means that for all $n \geq n_0$, the graph of $f(n)$ falls between $c_1g(n)$ and $c_2g(n)$.

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Theta (Θ) Example: findAvg1d()

- $T_{\text{findAvg1D}}(n) = \Theta(n)$, for $c_1 = 2$, $c_2 = 4$, $n_0 = 5$



Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Big-Oh (O) Notation

A function $f(n)$ is $O(g(n))$ if there are positive constants c and n_0 such that $f(n) \leq cg(n)$ for all $n \geq n_0$.

What we are saying is that $f(n)$ will grow no faster than a multiple of $g(n)$; hence $g(n)$ is an **upper bound** on the growth rate of $f(n)$.

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

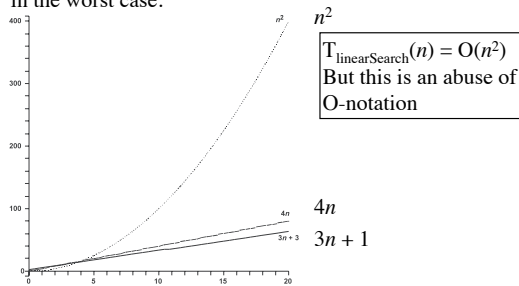
Linear Search

	Statements	Cost
1	int linearSearch(int [], int n, int target) {	
2	int i = 0;	1
3	while (i < n) {	n
4	if (target == array[i])	$n-1$
5	return i;	1
6	i++;	$n-1$
7	}	
8	return -1;	1
9	}	
	TOTAL	$3n + 1$

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Big-Oh (O): Linear Search

$T_{\text{linearSearch}}(n) = 3n + 1 = O(n)$ for $c = 4$ and $n_0 = 0$,
in the worst case.



Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Constant Time Algorithms: O(1)

```
int medianOf3( int [], int n ) {
  int v1 = a[0];
  int v2 = a[n/2];
  int v3 = a[n-1];
  if ( ( v1 < v2 ) && ( v1 < v3 ) ) // v1 is smallest
    if ( v2 < v3 )
      return n/2; // middle position
  else return n - 1; // last position
  else if ( ( v2 < v1 ) && ( v2 < v3 ) ) // v2 smallest
    if ( v1 < v3 )
      return 0; // first position
  else return n - 1; // last position
  else // v3 is smallest
    if ( v1 < v2 )
      return 0; // first position
    else return n / 2; // middle position
}
```

O(1): the number of instructions
executing is independent of
the size of the input

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Some Common Computing Times

$\log_2 n$	n	$n \log_2 n$	n^2	2^n
1	2	2	4	4
2	4	8	16	16
3	8	24	64	256
4	16	64	256	65,536
5	32	160	1,024	4,294,967,296
6	64	384	4,096	1.84×10^{19}
7	128	896	16,384	3.40×10^{38}
8	256	2,048	65,536	1.16×10^{77}
9	512	4,608	262,144	1.34×10^{154}
10	1,024	10,240	1,048,576	1.80×10^{308}

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Algorithm Measurement

Asymptotic analysis doesn't always tell the full story. Asymptotically, finding the largest value in an array of ints and an array of Integer objects is the same, but in reality...

Pseudocode: for timing an algorithm

1. *initialize the data structure*
2. *for n iterations*
3. *get the starting time*
4. *run the algorithm*
5. *get the finish time*
6. *totaltime = totaltime + (finish time - start time)*
7. *average time = total time / n*

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Some Astonishing Results?

n	array of int	array of Integer
4,000,000	11.363	21.739
8,000,000	22.727	42.958
800,000	2.314	4.329

Timings for findMax() on an array of ints and an array of Integers (times are in milliseconds)

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Components of Space Complexity

- **Instruction Space**
 - The space needed to store the program in memory for execution.
- **Data Space**
 - The space needed for constants and simple variables.
 - The space needed for objects.
- **Environment Stack Space**
 - Stores the information needed to resume execution of a method.

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Instruction Space

- The compiler and the compiler settings influence the size of the code produced.
- Settings may include optimization or debugging options.
- The space required is constrained based on the use of overlays and segments.

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Data Space

- The space required for simple variables and constants is defined by the language.
- The space required by an array is the product of the number of cells in the array and the size of the elements.
- The space required varies as items are introduced, used, then the memory is released as the item's scope ends.

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Scope

- The scope of an identifier is that portion of the program when the identifier has meaning.
- Java defines scope based on blocks, as defined by {}.

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Space Requirements

$c+S_p$

c represents the space required regardless of other method characteristics.

S_p represents the space requirements of the method that “depend” on method characteristics.

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Types and Space

Type	Space(bits)
boolean	1
char	16
byte	8
int	32
long	64
float	32
double	64
reference	32

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Example

```
public int compute (int a, int b, int c) {
    return a*b+c;
}
```

- How is the data space affected by the magnitudes of a, b, c?
- Pass by value versus pass-by-reference.

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Example

```
public int sequentialSearch (Object [] a, Object x) {
    int i;
    for(i=0;i<a.length && !x.equals(a[i]);i++);
    if (i==a.length) return -1;
    else return i;
}
```

How is the data space needed to run this method affected by the size of the array passed to the method?

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

Example

What is the space required to run the method defined below?

```
public void computeIt() {
    int i=4;
    int k;
    float x=4.0f;
    float y=3.0f;
    double z= (i+3)*x/y;
}
```

Copyright © 2007 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

What is the space required to run the method defined below?

```
public int computeIt(int value){
    if(10<value) {
        int total=0;
        for (int i=1;i<value;i++) total+=i;
        return total;
    }
    else {
        int sum=0;
        for (int i=value;i>value;i--) sum+=i;
        return sum;
    }
}
```
