Praveen Sagar Karri
UTD ID: 2021220422

Home Work 4

Discrete Structures (CS 5333)

Page 184 - Exc.04: Find the product **AB**, where

a) $A = \begin{vmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ -1 & 1 & 0 \end{vmatrix}$ $\quad B = \begin{vmatrix} 0 & 1 & -1 \\ 1 & -1 & 0 \\ -1 & 0 & 1 \end{vmatrix}$

c) $A = \begin{vmatrix} 0 & -1 \\ 7 & 2 \\ -4 & -3 \end{vmatrix}$ $\quad B = \begin{vmatrix} 4 & -1 & 2 & 3 & 0 \\ -2 & 0 & 3 & 4 & 1 \end{vmatrix}$

Answer:

a) $AB = \begin{vmatrix} -1 & 1 & 0 \\ 0 & 1 & -1 \\ 1 & -2 & 1 \end{vmatrix}$

b) $AB = \begin{vmatrix} 2 & 0 & -3 & -4 & -1 \\ 24 & -7 & 20 & 29 & 2 \\ -10 & 4 & -17 & -24 & -3 \end{vmatrix}$

Page 185 - Exc.24:

a) Show that the system of simultaneous linear equations

$a_{11}x_1 + a_{12}x_2 + \cdot \quad \cdot \quad \cdot + a_{1n}x_n = b_1$

$a_{21}x_1 + a_{22}x_2 + \cdot \quad \cdot \quad \cdot + a_{2n}x_n = b_2$

...

$a_{n1}x_1 + a_{n2}x_2 + \cdot \quad \cdot \quad \cdot + a_{nn}x_n = b_n.$

in the variables $x_1, x_2, \ldots, x_n$ can be expressed as **AX** = **B**, where **A** = $[a_{ij}]$, **X** is an $n \times 1$ matrix with x_i the entry in its ith row, and **B** is an $n \times 1$ matrix with b_i the entry in its ith row.

Answer:

If $A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$ And $X = \begin{vmatrix} x_1 \\ \vdots \\ x_n \end{vmatrix}$ then AX = $\begin{vmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n \end{vmatrix}$

If $B = \begin{vmatrix} b_1 \\ \vdots \\ b_n \end{vmatrix}$ and AX = B then $\begin{vmatrix} b_1 \\ \vdots \\ b_n \end{vmatrix} = \begin{vmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n \end{vmatrix}$ which implies

$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$

$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$

......

$a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n$

We simply note that under the given definitions of A, X, and B, the definition of matrix multiplication is exactly the system of equations shown.

b) Show that if the matrix **A** = $[a_{ij}]$ is invertible (as defined in the preamble to Exercise 18), then the solution of the system in part (a) can be found using the equation **X** = $\mathbf{A}^{-1}\mathbf{B}$.

Answer:

The given system is the matrix equation AX = B. If A is invertible with inverse A^{-1}, then we can multiply both sides of this equation by A−1 to obtain A^{-1}AX = A^{-1}B. By the definition of inverse the left-hand side simplifies to IX and this is simply X. Thus the given system is equivalent to the system X = A^{-1}B, which obviously tells us exactly what X is (and therefore what all the values xi are).

Page 202 - Exc.8: Describe an algorithm that takes as input a list of *n* distinct integers and finds the location of the largest even integer in the list or returns 0 if there are no even integers in the list.

Answer:

Procedure greatest even integer($a_1, a_2, a_3, \ldots\ldots\ldots a_n$, distinct integers)

g := 0
m := a_1
for i:= 1 to n
    if a_i is even and $a_i > m$ **then**
        g := i
        m := a_i
return g.

Page 216 - Exc.24: Suppose that you have two different algorithms for solving a problem. To solve a problem of size *n*, the first algorithm uses exactly $n^2 2^n$ operations and the second algorithm uses exactly $n!$ operations. As *n* grows, which algorithm uses fewer operations?

Answer:

The first algorithm uses fewer operations because $n^2 2^n$ is O(n!) but n! is not O($n^2 2^n$). In fact, the second function overtakes the first function for good at n = 8, when 82 · 28 = 16,384 and 8! = 40,320.

Page 216 - Exc.34. a) Show that $3x^2 + x + 1$ is $\Theta(3x^2)$ by directly finding the constants k, C_1, and C_2 in Exercise 33.

Answer:

If f (x) and g(x) are functions from the set of real numbers to the set of real numbers, then f (x) is O(g(x)) if and only if there are positive constants k, C1, and C2 such that $C1|g(x)| \le |f(x)| \le C2|g(x)|$ whenever x > k.

$c1|3x^2| \le |3x^2 + x + 1| \le c2|3x^2|$

$|3x^2| \le |3x^2 + x + 1|$ for x > 0. and $|3x^2 + x + 1| \le 2|3x^2|$ for all x > 1. Which implies that $1|3x^2| \le |3x^2 + x + 1| \le 2|3x^2|$ for all x > 1; in other words, C1 = 1 and C2 = 2 k=1.

Page 229 - Exc.8: Given a real number x and a positive integer k, determine the number of multiplications used to find x^{2^k} starting with x and successively squaring (to find x^2, x^4, and so on). Is this a more efficient way to find x^{2k} than by multiplying x by itself the appropriate number of times?

Answer:

k multiplications are used when x is successively squared. Yes it is more efficient than multiplying x by itself as in this case 2^k multiplications are used.

Page 230 - Exc.14: There is a more efficient algorithm (in terms of the number of multiplications and additions used) for evaluating polynomials than the conventional algorithm described in the previous exercise. It is called **Horner's method**.

This pseudocode shows how to use this method to find the value of

$a_nx_n + a_{n-1}x_{n-1} + \cdot \quad \cdot \quad \cdot + a_1x + a_0$ at $x = c$.

procedure *Horner*(c, a_0, a_1, a_2, . . . , a_n: real numbers)

$y := a_n$

for $i := 1$ **to** n

$y := y * c + a_{n-i}$

return y $\{y = a_nc_n + a_{n-1}c_{n-1} + \cdot \quad \cdot \quad \cdot + a_1c + a_0\}$

a) Evaluate $3x2 + x + 1$ at $x = 2$ by working through each step of the algorithm showing the values assigned at each assignment step.

Answer:

On the initial call to the algorithm:

n = 2 c=2 $a_0 = 1$ $a_1 = 1$ $a_2 = 3$ $y = a_2 = 3$

First iteration (j=1)	second iteration (j=2)	FINAL ANSWER
$y = y*c + a_{n-1}$	$y = y*c + a_{n-2}$	y = 15
$y = 3*2 + a_1$	$y = 7*2 + a_0$	
y = 3*2 + 1	y = 7*2 + 1	
y = 7 y = 15		

Check: $3(2^2) + 2 + 1 = (3*4) + 2 + 1 = 12 + 2 + 1 = 15$

b) Exactly how many multiplications and additions are used by this algorithm to evaluate a polynomial of degree *n* at *x* = *c*? (Do not count additions used to increment the loop variable)

Answer:

The loop iterates n times. In 1 iteration of the loop, 1 multiplication and 1 addition is done. Therefore total number of multiplications and additions done is n*(1+1) = 2n operations total (n additions and n multiplications).

Page 230 - Exc.20: What is the effect in the time required to solve a problem when you double the size of the input from *n* to 2*n*, assuming that the number of milliseconds the algorithm uses to solve the problem with input size *n* is each of these function? [Express your answer in the simplest form possible, either as a ratio or a difference. Your answer may be a function of *n* or a constant.]

a) log log *n* **b)** log *n* **c)** 100*n*
d) *n* log *n* **e)** *n*2 **f)** n^3
g) 2^n

Answer:

a) log log n
log log 2n − log log n = log ((log 2 + log n) /log n) = log ((1 + log n)/ log n).If n is large, the fraction in this expression is approximately equal to 1, and therefore the expression is approximately equal to 0. In other words, hardly any extra time is required. For example, in going from
n = 1024 to n = 2048, the number of extra milliseconds is log 11/10 + 0.14.

b) log n
Here we have log 2n − log n = log 2n /n = log 2 = 1. One extra millisecond is required, independent of n.

c) 100n
Here 100n/100(2n) = 100n/200n = 1:2. Time required is doubled for the input 2*n*.

d) n log n
2n log 2n – n log n = 2n log 2 + 2n log n – n log n = 2n + n log n.
2n + n log n extra milliseconds are required when the input is changed to 2*n*.

e) n^2
Because $(2n)^2/n^2/$ = 4, we see that four times as much time is required for the larger problem.

f) n^3
$(2n)^3/n^3/$ = 8, we see that eight times as much time is required for 2*n* input.

g) 2^n
The relevant ratio is $2^{2n}/2^n$, which equals 2^n . If n is large, then this is a huge number. For example, in going from n = 10 to n = 20, the number of milliseconds increases over 1000-fold.

Page 231 - Exc.44: What is the best order to form the product **ABC** if **A**, **B**, and **C** are matrices with dimensions 3×9, 9×4, and 4×2, respectively?

Answer: We have two choices: (AB)C or A(BC). For the first choice, it takes $3 \cdot 9 \cdot 4 = 144$ multiplications to form the 3×4 matrix AB, and then $3 \cdot 4 \cdot 2 = 24$ multiplications to get the final answer, for a total of 168 multiplications. For the second choice, it takes $9 \cdot 4 \cdot 2 = 72$ multiplications to form the 9×2 matrix BC, and then $3 \cdot 9 \cdot 2 = 54$ multiplications to get the final answer, for a total of 126 multiplications. The second method uses fewer multiplications and so is the better choice.

Page 234 - Exc.24: Find an integer n with n > 2 for which $(\log n)^{2^{100}} < \sqrt{n}$.

Answer:

$(\log n)^{2^{100}} < \sqrt{n}$

⇨ $(\log n)^{2^{101}} < n$ Squaring on both sides

Let $n = 2^k$, then

⇨ $(k)^{2^{101}} < 2^k$

⇨ $2^{101} \log k < k$ Taking logarithmic on both sides

Let $k = 2^m$ then

⇨ $2^{101} m < 2^m$

m should be greater than 101 and also $m < 2^{m-101}$. m = 108 satisfies the conditions.

Therefore $n = 2^{2^{108}}$ satisfies the given equation.