Runtime Functions

Monday, September 30, 2019 5:35 PM

getting a Runtime function from an Algorithm or program.

Basic idea: Count operations:

$- + * / = \quad == \quad [] =$ Cost of 1

```
x[4] = 7 * y[5] + 4;
```

Cost = 5

Rule 1)

```
S1;
S2
```

cost = cost(S1) + cost(S2)

Rule 2) if:

```
if (test)
    S1;
else
    S2;
```

cost = cost(test) + max(cost(S1), cost(S2))

Rule 3) for-loops

```
for(init, test, update)
    S1;
```

$$cost = cost(init) + \sum_{index=start}^{end} \left(cost(S1) + cost(test) + cost(update) \right)$$

Rule 4) while-loops

```
while (test)
    S1;
```

- Assume it terminates

Consider worst case number of repetitions: K

cost = K * (cost(S1) + cost(test))

$3n^2+8$; $7n^2+3n+2$; $99n^2+128 \Rightarrow \mathcal{O}(n^2)$

Applying to Code

EX #1

```
void Arraylist::swap(int i, int j)
   T tmp;

   tmp = data[i];
   data[i] = data[j];
   data[j] = tmp;

   return;
```

2 + 3 + 2 = 7 is $\Theta(1)$

EX #2

```
foo(int n, int k)
  int x;

  if (n == 0)
    x = 0;
  else {
    x = k * k;
    x = x / n;
  }
  return x;
;
```

1 + (2 + 2) = 4 + 1 = 5 is $\Theta(1)$

EX #3

```
sum(int a[], int n)
  int s = 0;

  for (int k=0; k<n; k++){
    s = s + a[k];
  }

  return s;
;
```

$1 + \sum_{k=0}^{n-1}(3+1+1) = 1 + 5n$ is $\Theta(n)$

EX #4

```
LL* LinkedList::find( T x )
  LL* p = this;

  while (p->next != NULL){
    if (p->data == x){
```

test

```
while (p->next != NULL){
  if (p->data == x){
    return p;
  }
  p = p->m_next;
}
return NULL;
;
```

2 + 2 = 4

test: $2 + 4n + 1$ is $\Theta(n)$ (size of list)

EX #5

```
sum_sqrm(int a**, int n)
  int s = 0;

  for (int k=0; k<n; k++)
  {
    for(int j=0; j<n; j++)
    {
      s = s + a[k][j];
    }
  }

  return s;
;
```

$1 + \sum_{j=0}^{n-1} (6) = 1 + 6n$

$1 + \sum_{k=0}^{n-1} (1 + 6n + 1 + 1)$

$\equiv$

$1 + \sum_{k=0}^{n-1} (6n + 3)$

$\equiv$

$1 + n(6n + 3)$

$\equiv$

$6n^2 + 3n + 1$ is $\Theta(n^2)$

EX #6

```
int x = 1;

  for (int i=0; i<n; i++)
  {
    for(int j=0; j< n*n; j++)
    {
      x = x + K * i;
    }
  }
;
```

$1 + \sum_{j=0}^{n^2-1} (3 + 1 + 1)$

$5n^2 + 1$

$1 + \sum_{i=1}^{n-1} (5n^2 + 1) + 1$

$\equiv$

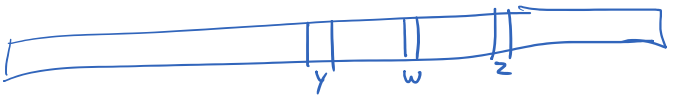
$5n^3 + n + 2$ is $\Theta(n^3)$

EX #7

$\log n$?

$\log_2 n$?

- given a sorted array



find(x) in the array?

How many times can you split array in half?
to which power to raise 2 to match size of array?

$\log_2 n$ n: size of array

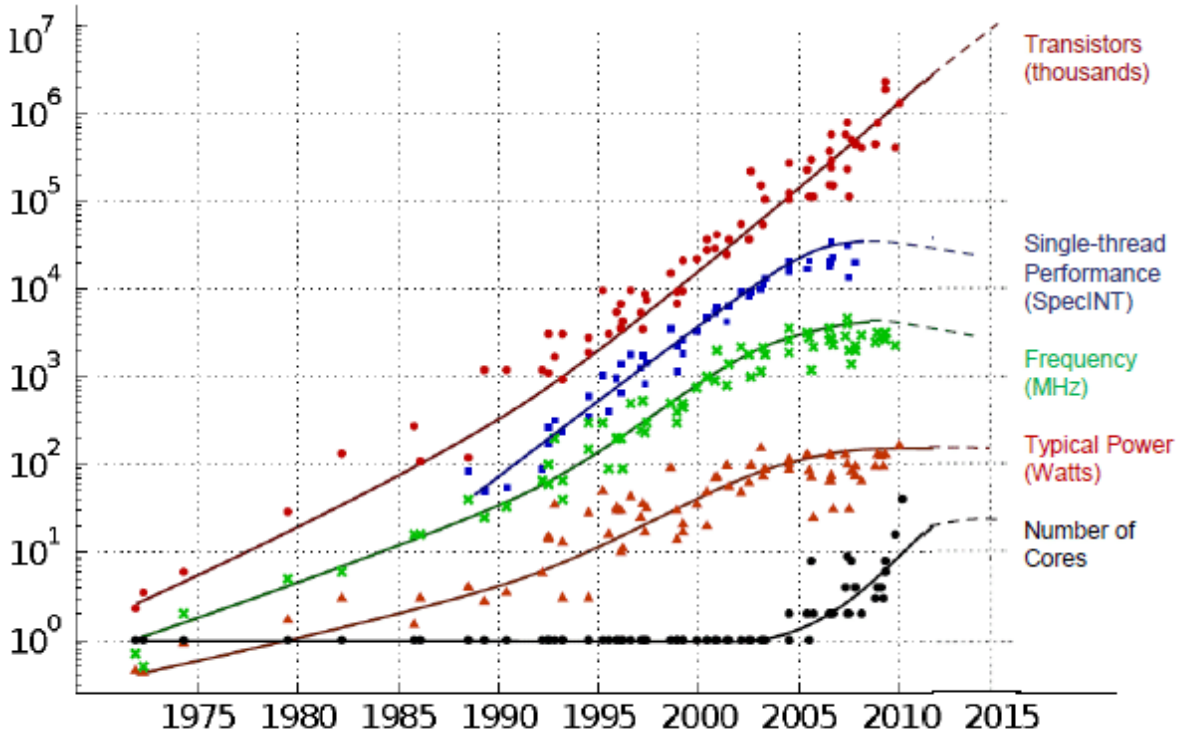
;

- Apply to List Implementations (Data Structures)

	ArrayList	LinkedList	Enc. LinkedList
front	1	1	1
back	1	n	1
at	1	n	n
insert_back	n	n	1
insert_front	n	1	1
insert	n	1	1
remove	n	1	1
find	n	n	n

Importance

Moore's Law



Wirth's Law: Software becomes slower faster than hardware becomes faster…

10^x

Programs	100 /sec 6,000/min	1,000 /sec 60,000 /min	
$125n$	48	480	10x
$12n^2$	22	70	3x
$\frac{1}{3}n^3$	26	56	2x
2^n	12	15	1.2x
$n!$	7	8	+1

- Problem complexity:

"This program has complexity n^2"

"This problem has complexity n^2"

"the best program known to solve such problem has complexity n^2"

Sorting : n^2 C.A.R. Hoare $n.\log_2 n$ "Quicksort"

Matrix multiplication: n^3 $n^{2.56}$

SAT $(a \vee b \vee \neg c) \wedge (c \vee b) \wedge (\neg a)$

2^n

a = off
b = on
c = off

Cryptography $p \overset{\text{msg}}{\Rightarrow} q \overset{\text{cypher}}{}$ prime factors of q

Prime factorisation 2^n