Analysis of Quicksort

Rongen Lin

Nanjing University

April 27, 2020

Contents

Contents

Quicksort

```
procedure quicksort (integer value l, r);

    comment The array A is declared to be A [1 : N + 1], with A [N + 1] = ∞;
    if r − l ≧ M then
        i := l; j := r + 1; v := A [l];
        loop:
            loop: i := i + 1; while A [i] < v repeat;
            loop: j := j − 1; while A [j] > v repeat;
        until j < i:
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1
            then quicksort (l, j − 1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j − 1);
        endif;
    endif;

procedure insertionsort (l, r):
    loop for r − 1 > i > l:
        if A [i] > A [i + 1] then
            v := A [i]; j := i + 1;
            loop: A [j − 1] := A [j]; j := j + 1 while A [j] < v repeat;
            A [j − 1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

6 quantities

- A : the number of partitioning stages
- B : the number of exchanges during partitioning
- C : the number of comparisons during partitioning
- D : the number of insertions
- E : the number of keys moved during insertion
- S : the number of stack pushes

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A[1 : N+1], with A[N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A[l];
        loop:
            loop: i := i+1; while A[i] < v repeat;
            loop: j := j−1; while A[j] > v repeat;
        until j < i:
            A[i] := :A[j];
        repeat;
        A[l] := :A[j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

```
procedure insertionsort (l, r):
    loop for r−1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j−1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j−1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

6 quantities

- Instructions which do not reference memory cost one time unit
- Instructions which do reference memory cost two time units

the total running time is

$$24A + 11B + 4C + 3D + 8E + 9S + 7N$$

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = f_N + \frac{1}{N} \sum_{1 \le k \le N} (F_{k-1} + F_{N-k}) \qquad \text{for } N > M$$

F_N : the average value of some quantity for random arrangements.
f_N : the average value of some quantity for the first partitioning stage.

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = f_N + \frac{1}{N} \sum_{1 \le k \le N} (F_{k-1} + F_{N-k}) \qquad \text{for } N > M$$

F_N : the average value of some quantity for random arrangements.
f_N : the average value of some quantity for the first partitioning stage.

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1; while A [i] < v repeat;
            loop: j := j−1; while A [j] > v repeat;
        until j < i:
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = f_N + \frac{1}{N} \sum_{1 \le k \le N} (F_{k-1} + F_{N-k}) \qquad \text{for } N > M$$

F_N : the average value of some quantity for random arrangements.
f_N : the average value of some quantity for the first partitioning stage.

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1; while A [i] < v repeat;
            loop: j := j−1; while A [j] > v repeat;
        until j < i:
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

A : $f_N = f_{M+1} = F_{M+1} = 1.$

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = f_N + \frac{1}{N} \sum_{1 \le k \le N} (F_{k-1} + F_{N-k}) \qquad \text{for } N > M$$

F_N : the average value of some quantity for random arrangements.
f_N : the average value of some quantity for the first partitioning stage.

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1; while A [i] < v repeat;
            loop: j := j−1; while A [j] > v repeat;
        until j < i:
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

A : $f_N = f_{M+1} = F_{M+1} = 1.$

C : $f_N = N + 1, f_{M+1} = F_{M+1} = M + 2.$

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

$$F_N = f_N + \frac{1}{N} \sum_{1 \leq k \leq N} (F_{k-1} + F_{N-k})$$

Analysis of the Basic Algorithm

$$F_N = f_N + \frac{1}{N} \sum_{1 \leq k \leq N} (F_{k-1} + F_{N-k})$$

$$NF_N = Nf_N + 2 \sum_{1 \leq k \leq N} F_{k-1}$$

Analysis of the Basic Algorithm

$$F_N = f_N + \frac{1}{N} \sum_{1 \le k \le N} (F_{k-1} + F_{N-k})$$

$$NF_N = Nf_N + 2 \sum_{1 \le k \le N} F_{k-1}$$

$$NF_N - (N-1)F_{N-1} = \nabla(Nf_N) + 2F_{N-1}$$

Analysis of the Basic Algorithm

$$F_N = f_N + \frac{1}{N} \sum_{1 \le k \le N} (F_{k-1} + F_{N-k})$$

$$NF_N = Nf_N + 2 \sum_{1 \le k \le N} F_{k-1}$$

$$NF_N - (N-1)F_{N-1} = \nabla(Nf_N) + 2F_{N-1}$$

$$\frac{F_N}{N+1} = \frac{F_{N-1}}{N} + \frac{\nabla(Nf_N)}{N(N+1)}$$

Analysis of the Basic Algorithm

$$F_N = f_N + \frac{1}{N} \sum_{1 \le k \le N} (F_{k-1} + F_{N-k})$$

$$NF_N = Nf_N + 2 \sum_{1 \le k \le N} F_{k-1}$$

$$NF_N - (N-1)F_{N-1} = \nabla(Nf_N) + 2F_{N-1}$$

$$\frac{F_N}{N+1} = \frac{F_{N-1}}{N} + \frac{\nabla(Nf_N)}{N(N+1)}$$

$$F_N = (N+1)\left(\frac{F_{M+1}}{M+2} + \sum_{M+2 \le k \le N} \frac{\nabla(kf_k)}{k(k+1)} \right)$$

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

$$\sum_{M+2\leq k\leq N} \frac{\nabla(kf_k)}{k(k+1)} = \sum_{M+2\leq k\leq N} \frac{f_k - f_{k-1}}{k+1} + \sum_{M+2\leq k\leq N} \frac{f_{k-1}}{k} - \sum_{M+2\leq k\leq N} \frac{f_{k-1}}{k+1}$$

Analysis of the Basic Algorithm

$$\sum_{M+2\le k\le N} \frac{\nabla(kf_k)}{k(k+1)} = \sum_{M+2\le k\le N} \frac{f_k - f_{k-1}}{k+1} + \sum_{M+2\le k\le N} \frac{f_{k-1}}{k} - \sum_{M+2\le k\le N} \frac{f_{k-1}}{k+1}$$

$$= 2 \sum_{M+2\le k\le N} \frac{\nabla f_k}{k+1} + \frac{f_{M+1}}{M+2} - \frac{f_N}{N+1}$$

Analysis of the Basic Algorithm

$$\sum_{M+2\le k\le N} \frac{\nabla(kf_k)}{k(k+1)} = \sum_{M+2\le k\le N} \frac{f_k - f_{k-1}}{k+1} + \sum_{M+2\le k\le N} \frac{f_{k-1}}{k} - \sum_{M+2\le k\le N} \frac{f_{k-1}}{k+1}$$
$$= 2 \sum_{M+2\le k\le N} \frac{\nabla f_k}{k+1} + \frac{f_{M+1}}{M+2} - \frac{f_N}{N+1}$$

Substituting, we have

Analysis of the Basic Algorithm

$$\sum_{M+2\le k\le N} \frac{\nabla(kf_k)}{k(k+1)} = \sum_{M+2\le k\le N} \frac{f_k - f_{k-1}}{k+1} + \sum_{M+2\le k\le N} \frac{f_{k-1}}{k} - \sum_{M+2\le k\le N} \frac{f_{k-1}}{k+1}$$
$$= 2 \sum_{M+2\le k\le N} \frac{\nabla f_k}{k+1} + \frac{f_{M+1}}{M+2} - \frac{f_N}{N+1}$$

Substituting, we have

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2\le k\le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

A :

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

A :

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A[1 : N+1], with A[N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A[l];
        loop:
            loop: i := i+1; while A[i] < v repeat;
            loop: j := j−1; while A[j] > v repeat;
        until j < i;
            A[i] := :A[j];
        repeat;
        A[l] := :A[j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

A :

$$f_N = f_{M+1} = F_{M+1} = 1, \nabla f_N = 0 \Rightarrow A_N = 2\frac{N+1}{M+2} - 1$$

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1; while A [i] < v repeat;
            loop: j := j−1; while A [j] > v repeat;
        until j < i;
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

C :

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

C :

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1; while A [i] < v repeat;
            loop: j := j−1; while A [j] > v repeat;
        until j < i;
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i+1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

C :

$$f_N = N+1, f_{M+1} = F_{M+1} = M+2, \nabla f_N = 1$$

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1 while A [i] < v repeat;
            loop: j := j−1 while A [j] > v repeat;
        until j < i:
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

C :

$$f_N = N+1, f_{M+1} = F_{M+1} = M+2, \nabla f_N = 1$$
$$C_N = (N+1)(2H_{N+1} - 2H_{M+2} + 1)$$

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1; while A [i] < v repeat;
            loop: j := j−1; while A [j] > v repeat;
        until j < i;
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

D :

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

D :

```
procedure insertionsort (l, r):
    loop for r−1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j−1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j−1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

D :

$$D_N = N - H_N \qquad \text{for } N \le M$$

```
procedure insertionsort(l, r):
    loop for r−1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j−1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j−1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

D :

$$D_N = N - H_N \qquad \text{for } N \le M$$
$$f_N = f_{M+1} = \nabla f_k = 0$$

```
procedure insertionsort(l, r):
    loop for r−1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j−1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j−1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

D :

$$D_N = N - H_N \qquad \text{for } N \le M$$

$$f_N = f_{M+1} = \nabla f_k = 0$$

$$D_{M+1} = \frac{2}{M+1} \sum_{1 \le k \le M+1} (k - 1 - H_{k-1}) = M + 2 - 2H_{M+1}$$

```
procedure insertionsort(l, r):
    loop for r−1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j−1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j−1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

D :

$$D_N = N - H_N \qquad \text{for } N \le M$$

$$f_N = f_{M+1} = \nabla f_k = 0$$

$$D_{M+1} = \frac{2}{M+1} \sum_{1 \le k \le M+1} (k - 1 - H_{k-1}) = M + 2 - 2H_{M+1}$$

$$D_N = \frac{N+1}{M+2} D_{M+1} = (N+1)\left(1 - 2\frac{H_{M+1}}{M+2}\right)$$

```
procedure insertionsort (l, r):
    loop for r−1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j−1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j−1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

E :

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

E :

```
procedure insertionsort(l, r):
    loop for r−1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j−1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j−1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

E :

$$E_N = \frac{N(N-1)}{4} \qquad \text{for } N \le M$$

```
procedure insertionsort(l, r):
    loop for r−1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j−1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j−1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

E :

$$E_N = \frac{N(N-1)}{4} \qquad \text{for } N \le M$$
$$f_N = f_{M+1} = \nabla f_k = 0$$

```
procedure insertionsort(l, r):
    loop for r-1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j-1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j-1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

E :

$$E_N = \frac{N(N-1)}{4} \qquad \text{for } N \le M$$

$$f_N = f_{M+1} = \nabla f_k = 0$$

$$E_{M+1} = \frac{2}{M+1} \sum_{1 \le k \le M+1} \frac{(k-1)(k-2)}{4} = \frac{M(M-1)}{6}$$

```
procedure insertionsort(l, r):
    loop for r-1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j-1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j-1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = 2(N+1) \sum_{M+2 \le k \le N} \frac{\nabla f_k}{k+1} + \frac{N+1}{M+2}(f_{M+1} + F_{M+1}) - f_N \qquad \text{for } N > M$$

E :

$$E_N = \frac{N(N-1)}{4} \qquad \text{for } N \le M$$

$$f_N = f_{M+1} = \nabla f_k = 0$$

$$E_{M+1} = \frac{2}{M+1} \sum_{1 \le k \le M+1} \frac{(k-1)(k-2)}{4} = \frac{M(M-1)}{6}$$

$$E_N = \frac{N+1}{M+2} E_{M+1} = (N+1)\frac{M(M-1)}{6(M+2)}$$

```
procedure insertionsort(l, r):
    loop for r-1 > i > l:
        if A[i] > A[i+1] then
            v := A[i]; j := i+1;
            loop: A[j-1] := A[j]; j := j+1 while A[j] < v repeat;
            A[j-1] := v;
        endif;
    repeat;
```

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = (N+1)\left(\frac{F_{2M+2}}{2M+3} + \sum_{2M+3\leq k\leq N} \frac{\nabla(kf_k)}{k(k+1)}\right) \qquad \text{for } N \geq 2M+2$$

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = (N+1)\left(\frac{F_{2M+2}}{2M+3} + \sum_{2M+3\le k\le N} \frac{\nabla(kf_k)}{k(k+1)}\right) \qquad \text{for } N \ge 2M+2$$

S :

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = (N+1)\left(\frac{F_{2M+2}}{2M+3} + \sum_{2M+3\le k\le N} \frac{\nabla(kf_k)}{k(k+1)}\right) \qquad \text{for } N \ge 2M+2$$

S :

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A[1 : N+1], with A[N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A[l];
        loop:
            loop: i := i+1 while A[i] < v repeat;
            loop: j := j−1 while A[j] > v repeat;
        until j < i:
            A[i] := :A[j];
        repeat;
        A[l] := :A[j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = (N+1)\left(\frac{F_{2M+2}}{2M+3} + \sum_{2M+3\le k\le N} \frac{\nabla(kf_k)}{k(k+1)}\right) \qquad \text{for } N \ge 2M+2$$

S :

$$j \in [M+2, N-M-1] \Rightarrow f_N = \frac{N-2M-2}{N}, \nabla(kf_k) = 1$$

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A[1 : N+1], with A[N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A[l];
        loop:
            loop: i := i+1 while A[i] < v repeat;
            loop: j := j−1 while A[j] > v repeat;
        until j < i;
            A[i] := :A[j];
        repeat;
        A[l] := :A[j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = (N+1)\left(\frac{F_{2M+2}}{2M+3} + \sum_{2M+3\le k\le N} \frac{\nabla(kf_k)}{k(k+1)}\right) \qquad \text{for } N \ge 2M+2$$

S :

$$j \in [M+2, N-M-1] \Rightarrow f_N = \frac{N-2M-2}{N}, \nabla(kf_k) = 1$$
$$F_{2M+2} = 0$$

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A[1 : N+1], with A[N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A[l];
        loop:
            loop: i := i+1 while A[i] < v repeat;
            loop: j := j−1 while A[j] > v repeat;
        until j < i;
            A[i] := :A[j];
        repeat;
        A[l] := :A[j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

A common form of recursions

$$F_N = (N+1)\left(\frac{F_{2M+2}}{2M+3} + \sum_{2M+3\le k\le N} \frac{\nabla(kf_k)}{k(k+1)}\right) \qquad \text{for } N \ge 2M+2$$

S :

$$j \in [M+2, N-M-1] \Rightarrow f_N = \frac{N-2M-2}{N}, \nabla(kf_k) = 1$$

$$F_{2M+2} = 0$$

$$S_N = \frac{N+1}{2M+3} - 1$$

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A[1 : N+1], with A[N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A[l];
        loop:
            loop: i := i+1 while A[i] < v repeat;
            loop: j := j−1 while A[j] > v repeat;
        until j < i;
            A[i] := :A[j];
        repeat;
        A[l] := :A[j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

B :

Analysis of the Basic Algorithm

B :

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N + 1], with A [N + 1] = ∞;
    if r − l ≧ M then
        i := l; j := r + 1; v := A [l];
        loop:
            loop: i := i + 1 while A [i] < v repeat;
            loop: j := j − 1 while A [j] > v repeat;
        until j < i:
            A [i] := : A [j];
        repeat;
        A [l] := : A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j − 1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j − 1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

B :

$A[1]$ (pivot) is the k-th smallest element, t (exchanges during partitioning) is the number of keys among $A[2], \ldots, A[k]$ which are $> A[1]$.

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N + 1], with A [N + 1] = ∞;
    if r − l ≧ M then
        i := l; j := r + 1; v := A [l];
        loop:
            loop: i := i + 1 while A [i] < v repeat;
            loop: j := j − 1 while A [j] > v repeat;
        until j < i:
            A [i] := : A [j];
        repeat;
        A [l] := : A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j − 1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j − 1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

B :

$A[1]$ (pivot) is the k-th smallest element, t (exchanges during partitioning) is the number of keys among $A[2], \ldots, A[k]$ which are $> A[1]$.

$$f_N = \frac{1}{N} \sum_{1 \le k \le N} \sum_{0 \le t \le k-1} t \frac{\binom{N-k}{t}\binom{k-1}{k-1-t}}{\binom{N-1}{k-1}}$$

$$= \frac{1}{N} \sum_{1 \le k \le N} \frac{N-k}{\binom{N-1}{k-1}} \sum_{0 \le t \le k-1} \binom{N-k-1}{t-1}\binom{k-1}{k-1-t} = \frac{N-2}{6}$$

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1 while A [i] < v repeat;
            loop: j := j−1 while A [j] > v repeat;
        until j < i:
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

B :

$A[1]$ (pivot) is the k-th smallest element, t (exchanges during partitioning) is the number of keys among $A[2], \ldots, A[k]$ which are $> A[1]$.

$$f_N = \frac{1}{N} \sum_{1 \le k \le N} \sum_{0 \le t \le k-1} t \frac{\binom{N-k}{t}\binom{k-1}{k-1-t}}{\binom{N-1}{k-1}}$$

$$= \frac{1}{N} \sum_{1 \le k \le N} \frac{N-k}{\binom{N-1}{k-1}} \sum_{0 \le t \le k-1} \binom{N-k-1}{t-1}\binom{k-1}{k-1-t} = \frac{N-2}{6}$$

$$f_N = \tfrac{1}{6}(N+1) - \tfrac{1}{2} = \tfrac{1}{6} f_N - \tfrac{1}{2} f_N \Rightarrow B_N = \tfrac{1}{6} C_N - \tfrac{1}{2} A_N$$

```
procedure quicksort (integer value l, r);
    comment The array A is declared to be A [1 : N+1], with A [N+1] = ∞;
    if r − l ≧ M then
        i := l; j := r+1; v := A [l];
        loop:
            loop: i := i+1 while A [i] < v repeat;
            loop: j := j−1 while A [j] > v repeat;
        until j < i:
            A [i] := :A [j];
        repeat;
        A [l] := :A [j];
        if j − l < r − i + 1        both subfiles have more than M elements
            then quicksort (l, j−1); quicksort (i, r);
            else quicksort (i, r); quicksort (l, j−1);
        endif;
    endif;
```

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

$$A = 2\frac{N+1}{M+2} - 1$$

$$B = (N+1)\left(\frac{1}{3}H_{N+1} - \frac{1}{3}H_{M+2} + \frac{1}{6} - \frac{1}{M+2}\right) + \frac{1}{2}$$

$$C = (N+1)(2H_{N+1} - 2H_{M+2} - 1)$$

$$D = (N+1)\left(1 - 2\frac{H_{M+1}}{M+2}\right)$$

$$E = (N+1)\frac{M(M-1)}{6(M+2)}$$

$$S = \frac{N+1}{2M+3} - 1$$

Analysis of the Basic Algorithm

$$A = 2\frac{N+1}{M+2} - 1$$

$$B = (N+1)\left(\frac{1}{3}H_{N+1} - \frac{1}{3}H_{M+2} + \frac{1}{6} - \frac{1}{M+2}\right) + \frac{1}{2}$$

$$C = (N+1)(2H_{N+1} - 2H_{M+2} - 1)$$

$$D = (N+1)\left(1 - 2\frac{H_{M+1}}{M+2}\right)$$

$$E = (N+1)\frac{M(M-1)}{6(M+2)}$$

$$S = \frac{N+1}{2M+3} - 1$$

Corollary

The total average running time of Quicksort $(N > 2M+1)$ is

$$\frac{35}{3}(N+1)H_{N+1} - \frac{69}{2} + \frac{1}{6}(N+1)\left(8M + 71 - 70H_{M+2} + \frac{270}{M+2} + \frac{54}{2M+3} - 36\frac{H_{M+1}}{M+2}\right)$$

Analysis of the Basic Algorithm

Analysis of the Basic Algorithm

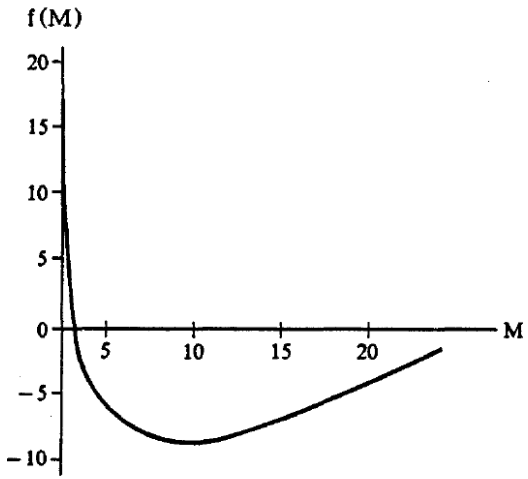

Fig. 2a. Contribution of M in Quicksort

Analysis of the Basic Algorithm

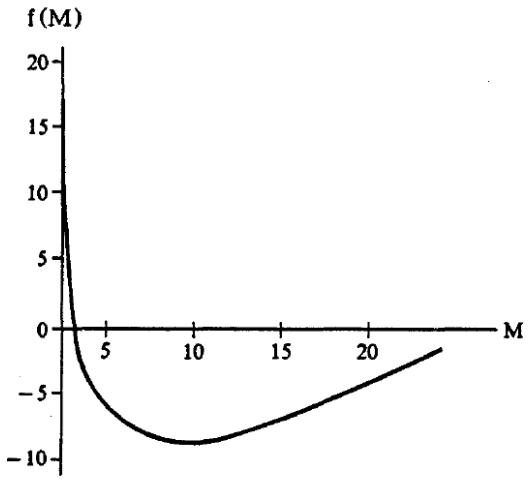

Fig. 2a. Contribution of M in Quicksort

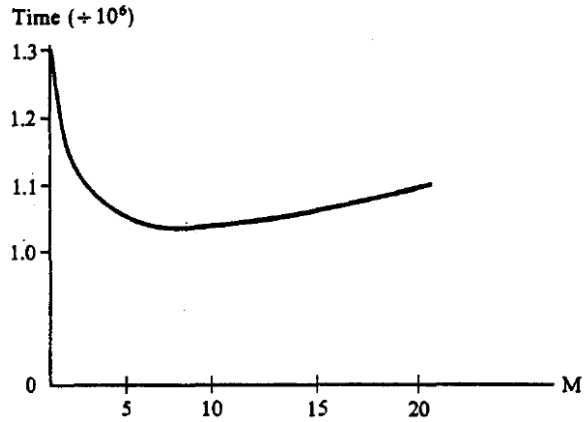

Fig. 2b. Total running time of Quicksort for $N = 10{,}000$

Analysis of the Basic Algorithm

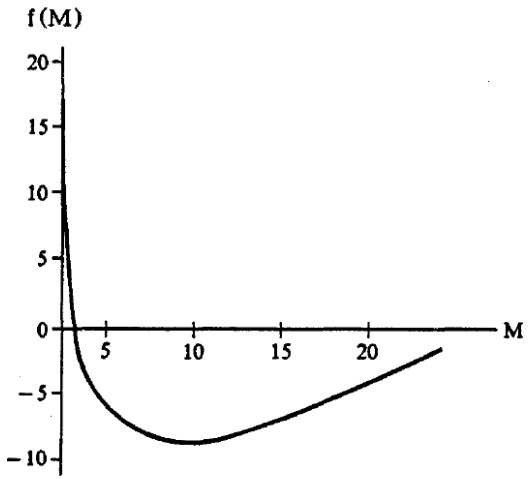

Fig. 2a. Contribution of M in Quicksort

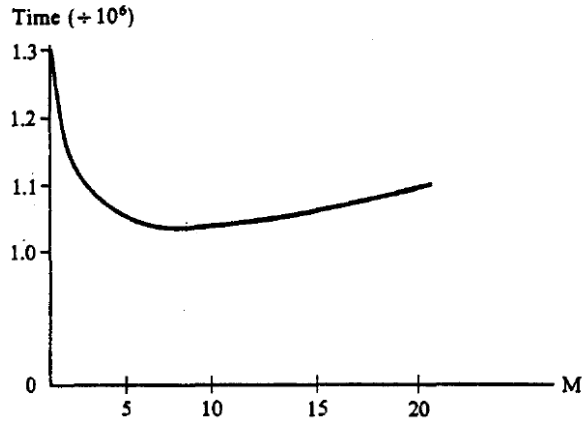

Fig. 2b. Total running time of Quicksort for $N = 10{,}000$

The best value of M is 9, then we have

Analysis of the Basic Algorithm

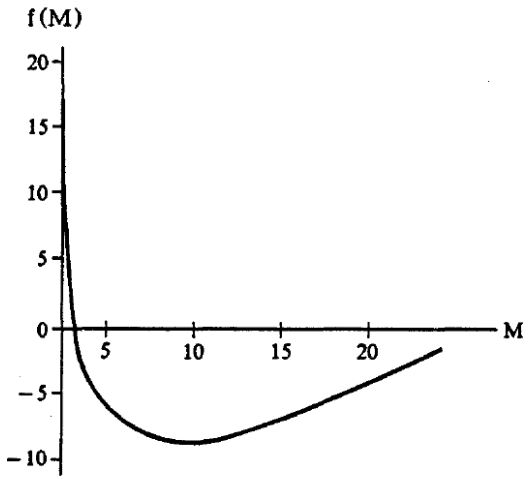

Fig. 2a. Contribution of M in Quicksort

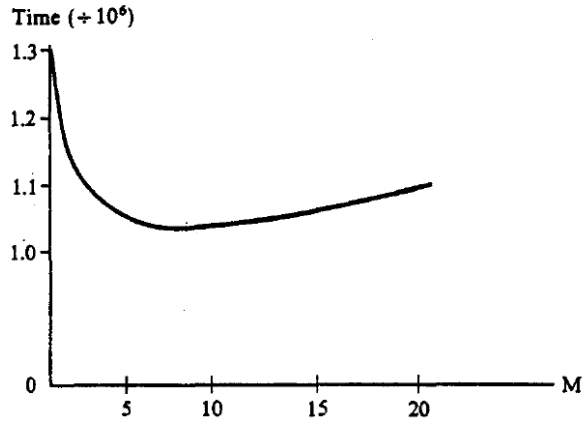

Fig. 2b. Total running time of Quicksort for $N = 10{,}000$

The best value of M is 9, then we have

$$T(N) = 11.6667(N+1)\ln N - 1.732N - 19$$

Contents

Analysis of the Median-of-Three Modification

Analysis of the Median-of-Three Modification

$$F_N = f_N + \sum_{1 \le k \le N} \frac{(N-k)(k-1)}{\binom{N}{3}} (F_{k-1} + F_{n-k})$$

Analysis of the Median-of-Three Modification

Analysis of the Median-of-Three Modification

$$F_N = f_N + \sum_{1 \le k \le N} \frac{(N-k)(k-1)}{\binom{N}{3}} (F_{k-1} + F_{n-k})$$

Analysis of the Median-of-Three Modification

$$F_N = f_N + \sum_{1 \le k \le N} \frac{(N-k)(k-1)}{\binom{N}{3}} (F_{k-1} + F_{n-k})$$

$$\binom{N}{3} F_N = \binom{N}{3} f_N + 2 \sum_{M < k < N} (N-k-1) k F_k$$

Analysis of the Median-of-Three Modification

$$F_N = f_N + \sum_{1 \le k \le N} \frac{(N-k)(k-1)}{\binom{N}{3}} (F_{k-1} + F_{n-k})$$

$$\binom{N}{3} F_N = \binom{N}{3} f_N + 2 \sum_{M < k < N} (N-k-1) k F_k$$

Define $F(z) = \sum_{N>M} F_N z^N$ and $f(z) = \sum_{N>M} f_N z^N$, then we have

Analysis of the Median-of-Three Modification

$$F_N = f_N + \sum_{1 \le k \le N} \frac{(N-k)(k-1)}{\binom{N}{3}} (F_{k-1} + F_{n-k})$$

$$\binom{N}{3} F_N = \binom{N}{3} f_N + 2 \sum_{M<k<N} (N-k-1) k F_k$$

Define $F(z) = \sum_{N>M} F_N z^N$ and $f(z) = \sum_{N>M} f_N z^N$, then we have

$$F'''(z) = f'''(z) + 12 \frac{F'(z)}{(1-z)^2}$$

Analysis of the Median-of-Three Modification

Analysis of the Median-of-Three Modification

$$F'''(z) = f'''(z) + 12\frac{F'(z)}{(1-z)^2}$$

Analysis of the Median-of-Three Modification

$$F'''(z) = f'''(z) + 12\frac{F'(z)}{(1-z)^2}$$

By $\theta F(z) = -(1-z)F'(z)$, it becomes

Analysis of the Median-of-Three Modification

$$F'''(z) = f'''(z) + 12\frac{F'(z)}{(1-z)^2}$$

By $\theta F(z) = -(1-z)F'(z)$, it becomes

$$-\theta(\theta-1)(\theta-2)F(z) = -12\theta F(z) + (1-z)^2 f'''(z)$$

Analysis of the Median-of-Three Modification

$$F'''(z) = f'''(z) + 12\frac{F'(z)}{(1-z)^2}$$

By $\theta F(z) = -(1-z)F'(z)$, it becomes

$$-\theta(\theta-1)(\theta-2)F(z) = -12\theta F(z) + (1-z)^2 f'''(z)$$

This corresponds to

Analysis of the Median-of-Three Modification

$$F'''(z) = f'''(z) + 12\frac{F'(z)}{(1-z)^2}$$

By $\theta F(z) = -(1-z)F'(z)$, it becomes

$$-\theta(\theta-1)(\theta-2)F(z) = -12\theta F(z) + (1-z)^2 f'''(z)$$

This corresponds to

$$-\theta U(z) = (1-z)^2 f'''(z)$$
$$(-2-\theta)T(z) = U(z)$$
$$(5-\theta)F(z) = T(z)$$

Analysis of the Median-of-Three Modification

Analysis of the Median-of-Three Modification

$$-\theta U(z) = (1-z)^2 f'''(z)$$
$$(-2-\theta) T(z) = U(z)$$
$$(5-\theta) F(z) = T(z)$$

Analysis of the Median-of-Three Modification

$$-\theta U(z) = (1-z)^2 f'''(z)$$
$$(-2-\theta) T(z) = U(z)$$
$$(5-\theta) F(z) = T(z)$$

Define $U(z) = \sum_{N>M} U_N z^N$ and $T(z) = \sum_{N>M} T_N z^N$, then it corresponds to

Analysis of the Median-of-Three Modification

$$-\theta U(z) = (1-z)^2 f'''(z)$$
$$(-2-\theta) T(z) = U(z)$$
$$(5-\theta) F(z) = T(z)$$

Define $U(z) = \sum_{N>M} U_N z^N$ and $T(z) = \sum_{N>M} T_N z^N$, then it corresponds to

$$(N+1) U_{N+1} = N U_N + 6\nabla^3 f_{N+3} \binom{N+3}{3}$$
$$(N+1) T_{N+1} = (N+2) T_N + U_N$$
$$(N+1) F_{N+1} = (N-5) F_N + T_N$$

Analysis of the Median-of-Three Modification

Analysis of the Median-of-Three Modification

By some techniques, we get

Analysis of the Median-of-Three Modification

By some techniques, we get

$$F_N = \frac{12}{7}\frac{N+1}{M+2}f_{M+1} - \frac{5}{7}f_N + \frac{12}{7}(N+1)\sum_{M+2\le k\le N}\frac{\nabla f_k}{k+1} - \frac{2}{7}\frac{1}{\binom{N}{6}}\sum_{M+1\le k\le N-1}\binom{k}{5}f_k$$

Analysis of the Median-of-Three Modification

By some techniques, we get

$$F_N = \frac{12}{7}\frac{N+1}{M+2}f_{M+1} - \frac{5}{7}f_N + \frac{12}{7}(N+1)\sum_{M+2\le k\le N}\frac{\nabla f_k}{k+1} - \frac{2}{7}\frac{1}{\binom{N}{6}}\sum_{M+1\le k\le N-1}\binom{k}{5}f_k$$

The best value of M is again 9, then we have

Analysis of the Median-of-Three Modification

By some techniques, we get

$$F_N = \frac{12}{7}\frac{N+1}{M+2}f_{M+1} - \frac{5}{7}f_N + \frac{12}{7}(N+1)\sum_{M+2\le k\le N}\frac{\nabla f_k}{k+1} - \frac{2}{7}\frac{1}{\binom{N}{6}}\sum_{M+1\le k\le N-1}\binom{k}{5}f_k$$

The best value of M is again 9, then we have

$$T'(N) = 10.6286(N+1)\ln N + 2.116N - 71$$

Analysis of the Median-of-Three Modification

By some techniques, we get

$$F_N = \frac{12}{7}\frac{N+1}{M+2}f_{M+1} - \frac{5}{7}f_N + \frac{12}{7}(N+1)\sum_{M+2\le k\le N}\frac{\nabla f_k}{k+1} - \frac{2}{7}\frac{1}{\binom{N}{6}}\sum_{M+1\le k\le N-1}\binom{k}{5}f_k$$

The best value of M is again 9, then we have

$$T'(N) = 10.6286(N+1)\ln N + 2.116N - 71$$
$$T(N) = 11.6667(N+1)\ln N - 1.732N - 19$$