Figure 2.2

QuickSort

Algorithm 2.2

QuickSort

2.3.2 The QuickSort Strategy

2.3.1 Introduction to Divide and Conquer

QuickSort and MergeSort: Divide and Conquer
In Section 2.4, we showed that if a sorting algorithm removes at most one inversion
from the permutation of the keys after each comparison, then it must do at least
\( n(n-1)/2 \) comparisons on the average (Theorem 2.1). QuickSort, however, does not
follow this rule. In this section, we describe how QuickSort can be implemented
in order to remove at most one inversion from the permutation of the keys after
each comparison. This results in a sorting algorithm that is more efficient than
many others.

2.4.3 Analysis of QuickSort

A small example is shown in Figure 2.7. The detailed operation of QuickSort is
shown only for the first line of code.

```
and [QuickSort] L[splitPoint)

| for i = 0 to end
| | if [L[splitPoint] > x]
| | | splitPoint = splitPoint + 1
| for i = 0 to end
| | | if [L[splitPoint] < x]
| | | | splitPoint = splitPoint - 1
| | | begin
| | | unknownIndex = x
| | | \textbf{for} x : Key
| | | | procedure SplitFirstLastIndex (splitPoint Index)\textbf{end} \textbf{if} [L[splitPoint] > x]
| | | begin
| | | | for i = 0 to end
| | | | | QSort (splitPoint + 1, last)
```

Figure 2.6 How Split Works: Initial, Intermediate, and Final Views.

The unknown group. At each iteration of the loop, QuickSort compares the next
unknown with \( x \). If the unknown is less than \( x \), it is placed in the
"less than \( x \)" group. If the unknown is greater than \( x \), it is placed in the
"greater than \( x \)" group. The other keys in the list before \( x \) are divided among
these two groups. Any key in the list between \( x \) and \( x \) is placed in the
"equal to \( x \)" group.
Expand the recurrence relation to get
\[ 0 = (1) \delta + (1 - u) \delta 2 + (1 - u) \delta 2 + (1 - u) \delta 0 = (u) \delta + (u) \delta \]

Decomposed by the recurrence relation is
\[ \delta \begin{cases} 0 & \text{for } n \geq 2 \\ (1) \delta + (1 - u) \delta \sum_{i=1}^{n-1} \left[ \left( \frac{1}{u} \right)^i \right] + (1 - u) \delta (u) \delta \end{cases} \]

Figure 2.7 Example of Quicksort

1 3 12 14 12 28 29

End line

Section with one key are sorted.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.

Split the first section.
This, of course, depends on $u$ in the worst case. Spill may spill off the entire array at
size of the stack depends on the number of spills, into which all the lists will be spills of
the other spills. We set the result to zero, if no other spills are made on stacks, and the
spill algorithm is working on sublists, the beginning and ending indexes (call them the
spill algorithm. If any stack that Q-factor is in-place sort. It is not, while the
AT first glance it may seem that Q-factor is in-place sort. It is not. While the

\textbf{Space Usage:

\[(1 - u) \otimes u \otimes u = (1 - u) V
\]

and therefore,

\[(1 - u) \otimes u \otimes u = (1 - u) V + (1 - u) g = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.

The recurrence relation for $g$ is

\[0 = (1 - u) V
\]

and

\[\frac{(1 - u) V - (1 - u) g}{V - u} = (u) g
\]

Now let

\[1 - (1 - u) V = \frac{(1 - u) V - (1 - u) g}{V - u}
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.

So

\[\frac{(1 - u) V - (1 - u) g}{V - u} = (u) g
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

Now let

\[1 - (1 - u) V = \frac{(1 - u) V - (1 - u) g}{V - u}
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.

So

\[\frac{(1 - u) V - (1 - u) g}{V - u} = (u) g
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.

So

\[\frac{(1 - u) V - (1 - u) g}{V - u} = (u) g
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.

So

\[\frac{(1 - u) V - (1 - u) g}{V - u} = (u) g
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.

So

\[\frac{(1 - u) V - (1 - u) g}{V - u} = (u) g
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.

So

\[\frac{(1 - u) V - (1 - u) g}{V - u} = (u) g
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.

So

\[\frac{(1 - u) V - (1 - u) g}{V - u} = (u) g
\]

and

\[\frac{1 + u - (1 - u) V}{V - u} = (u) g
\]

We have so far restricted the size of the stack to be small, thus we are
lightly dependent on the size of the stack. Thus, the size of the stack is
limited to the size of the stack, thus we are.
the algorithm described next can significantly reduce the maximum stack size.

The worst-case amount of space used by the stack is in \( O(n^2) \). One of the modifications to

This time in such a way that \( n-1 \) pairs of borders are stored on the stack.