Randomized Turing Machines

Deterministic Algorithm

- Is the algorithm which, given a particular input, will always produce the same output, with the underlying machine always passing through the same sequence of states.
- Is the algorithm that always solve the problem correctly.
Randomized Algorithm

- Is the algorithm that employs a degree of randomness as part of its logic. The algorithm typically uses a sequence of random bits as an auxiliary input to guide its behavior, in the hope of achieving good performance.
- So, in addition to input the algorithm takes a source of random numbers and makes random choices during execution.
- Its behavior can vary even on a fixed input.
- Different runs of a randomized algorithm may give different results.

Randomized vs. Deterministic Algorithm

There are two principal advantages to randomized algorithms.
- The first advantage is performance; randomized algorithms run faster than the best-known deterministic algorithms for many problems.
- The second advantage is that many randomized algorithms are simpler to describe and implement than deterministic algorithms of comparable performance.
Monte Carlo algorithm

- A Monte Carlo algorithm is a randomized algorithm that its running time is deterministic, but its output may be incorrect with a certain (typically small) probability.
- The concept of a Monte Carlo algorithm applies in situations where the algorithm makes a decision, its output is either yes or no. If the answer is yes, then it confirms it with probability larger than 1/2, but if the answer is no, then it simply remains silent.

Las Vegas algorithm

- Las Vegas algorithm is a randomized algorithm that always gives correct results or it informs about the failure. In other words, Las Vegas algorithm does not gamble with the correctness of the result; it gambles only with the resources used for the computation.
- Las Vegas algorithm as an algorithm that runs for an unpredictable amount of time but always succeeds.
- A simple example is randomized quicksort, where the pivot is chosen randomly, but the result is always sorted.
Quicksort: A Randomized Algorithm

- Given a list of elements, randomly select one element.
- Split the list into those above and those below the selected element.
- Recursively repeat the process on each new list.
- Put the new lists back together, resulting in a completely sorted list.
- Complexity Range: $O(n \log n)$ to $O(n^2)$.

A Turing Machine Using Randomization

Use a multi tape Turing Machine
- The first tape contains the input.
- The second tape is called the random tape. When blanks are scanned, they are filled with randomly chosen 0's and 1's with a probability of ½ (“coin- flip”) and never changed.
- The third and subsequent tapes are blank scratch tapes.
QuickSort on A Randomized Turing Machine

1) Select Pivot (Random Aspect of The Algorithm) The first tape contains the delineated sub-list of length m. Use $O(\log m)$ new random bits from tape 2 to select a random number. Since the value may not be a power of 2, we may not be able to select every integer between 1 and m with equal probability. Take $2 \log_2 m$ bits from tape 2, divide by m, and add 1. This yields a value between 1 and m with a probability close to $1/m$. 
Quicksort On A Randomized Turing Machine

2) Place Pivot on Tape 3
3) Copy Values <= Pivot To Tape 4 (Sub-Sublist)
4) Copy Values > Pivot To Tape 5 (Sub-Sublist)
5) Copy The Sub-Sublists From Tapes 4 & 5 Back To The Space on Tape 1 That Held The Sublist, Placing A Marker Between The Two Sub-Sublists
6) If The Size of either or both Sub-Sublists > 1, Recursively Perform Quicksort on those Sub-Sublists.

Language of A Randomized Turing Machine

- Every Turing machine, FA, or PDA accept some language, even if that language is empty set. But Randomized Turing machine may accept no languages at all.

- When considering what randomized TM M does in response to an input w, all possible contents of the random tape must also be considered. Some of the strings on the random tape may result in an accept state while others a reject state.
Cont.

- Each input $w$ to a randomized Turing machine $M$ has some probability of acceptance, which is the fraction of the possible contents of the random tape that lead to acceptance.

- Any sequence of moves that leads to acceptance uses only a finite section of the infinite random tape. If $m$ is the number of random tape cells scanned, the probability of what is seen there is $2^{-m}$.

Example of a randomized Turing Machine

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
<th>B0</th>
<th>B1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$q_00RS$</td>
<td>$q_001SR$</td>
<td>$q_010RS$</td>
<td>$q_011SR$</td>
<td>$q_1B0SS$</td>
<td>$q_1B1SS$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_00RS$</td>
<td>$q_001SR$</td>
<td>$q_010RS$</td>
<td>$q_011SR$</td>
<td>$q_1B0SS$</td>
<td>$q_1B1SS$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_00RR$</td>
<td>$q_001SR$</td>
<td>$q_010RS$</td>
<td>$q_011RR$</td>
<td>$q_1B0SS$</td>
<td>$q_1B1SS$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_00RR$</td>
<td>$q_001SR$</td>
<td>$q_010RS$</td>
<td>$q_011RR$</td>
<td>$q_1B0SS$</td>
<td>$q_1B1SS$</td>
</tr>
</tbody>
</table>

Transition Function of a Randomized Turing Function

Each row corresponds to a state and each column to a pair of XY symbols where $X$ is from the input tape and $Y$ is from the random tape. The table entries qUVDE means the Turing enters state $q$, writes $U$ on the input tape, writes $V$ on the random tape, moves the input head in direction $D$, and moves the random head in direction $E$. 
THANK YOU