In [1]:

```
import numpy as np
import pandas as pd
from scipy.stats import binom, norm


def analyze_doses(data_file):
    """
    Analyze the quality and curing time of vaccine doses using probab

    Parameters:
        data_file (str): Path to the dataset containing time of effect

    Returns:
        dict: Results of the analysis.
    """
    # Load dataset
    data = pd.read_csv(data_file)
    time_effects = data["time_of_effect"]

    # Probabilities for satisfactory and unsatisfactory doses
    p_satisfactory = 2 / 3
    p_unsatisfactory = 1 - p_satisfactory

    # Task 1: Analysis for 10 doses
    n1 = 10
    x_unsat = np.arange(0, n1 + 1)  # Possible outcomes for unsatisfac
    prob_unsat = binom.pmf(x_unsat, n1, p_unsatisfactory)  # Probabili
    task_1 = {
        "Probability Distribution (Unsatisfactory Doses)": dict(zip(x_
        "P(Exactly 3 Unsatisfactory)": binom.pmf(3, n1, p_unsatisfacto
        "P(At Most 3 Unsatisfactory)": binom.cdf(3, n1, p_unsatisfacto
        "P(More Than 8 Satisfactory)": 1 - binom.cdf(8, n1, p_satisfac
    }

    # Task 2: Analysis for 20 doses
    n2 = 20
    task_2 = {
        "P(At Least 11 Unsatisfactory)": 1 - binom.cdf(10, n2, p_unsat
        "P(At Most 5 Unsatisfactory)": binom.cdf(5, n2, p_unsatisfacto
        "P(At Least 13 Satisfactory)": 1 - binom.cdf(12, n2, p_satisfa
    }

    # Task 3: Time of effect analysis
    mean_time = time_effects.mean()
    std_time = time_effects.std()
    task_3 = {
        "P(Time < 11.5 hours)": norm.cdf(11.5, loc=mean_time, scale=st
        "P(Time > 10 hours)": 1 - norm.cdf(10, loc=mean_time, scale=st
        "90th Percentile of Time of Effect": norm.ppf(0.9, loc=mean_ti
    }

    # Combine results
    results = {"Task 1": task_1, "Task 2": task_2, "Task 3": task_3}
    return results


def format_output(results):
    """
    Format the results for user-friendly display.

    Parameters:
        results (dict): Results from the analysis.

    Returns:
        None
    """
    for task, details in results.items():
        print(f"\n{task}:")
        for description, value in details.items():
            if isinstance(value, dict):
                print(f"  {description}:")
                for outcome, prob in value.items():
                    print(f"    {outcome}: {prob:.4f}")
            else:
                print(f"  {description}: {value:.4f}")


if __name__ == "__main__":
    # File path to the dataset
    data_file = "Downloads/doses.csv"

    # Run analysis
    results = analyze_doses(data_file)

    # Format and display results
    print("=== Vaccine Dose Analysis Results ===")
    format_output(results)
```

```
=== Vaccine Dose Analysis Results ===

Task 1:
  Probability Distribution (Unsatisfactory Doses):
    0: 0.0173
    1: 0.0867
    2: 0.1951
    3: 0.2601
    4: 0.2276
    5: 0.1366
    6: 0.0569
    7: 0.0163
    8: 0.0030
    9: 0.0003
    10: 0.0000
  P(Exactly 3 Unsatisfactory): 0.2601
  P(At Most 3 Unsatisfactory): 0.5593
  P(More Than 8 Satisfactory): 0.1040

Task 2:
  P(At Least 11 Unsatisfactory): 0.0376
  P(At Most 5 Unsatisfactory): 0.2972
  P(At Least 13 Satisfactory): 0.6615

Task 3:
  P(Time < 11.5 hours): 0.3412
  P(Time > 10 hours): 0.7659
  90th Percentile of Time of Effect: 19.5236
```

In [ ]: