**UNIVERSITY OF BARISHAL**

PROJECT

**TITLE**  : Critical Path Method (CPM) on Building a

House Using Graph Data Structure.

**COURSE NAME :** Data Structures and Algorithms.

**COURSE CODE :** CSE – 1201

***SUBMITTED BY* -**

**NAME : Humayra Alamgir Nuha.**

**ROLL : 230102002**

**DEPARTMENT : Computer Science and Engineering.**

**SEMESTER : 2nd**

**YEAR : 1st**

***SUBMITTED TO* –**

**Md. Erfan**

**Assistant Professor & Chairman**

**Department of Computer Science and Engineering**

**University of Barishal.**

**DATE OF SUBMISSION:**

**Title:** Critical Path Method (CPM) on building a house.

**INTODUCTION:**

The Critical Path Method (CPM) is a fundamental project management technique widely used in construction, software development, and various engineering fields to optimize scheduling and resource allocation. This project explores the application of CPM in the context of building a house, utilizing graph data structures and computational algorithms to determine the most efficient sequence of tasks.

By implementing the forward pass and backward pass algorithms, the project identifies the critical path—the longest sequence of dependent tasks that dictate the project's minimum completion time. Through this approach, potential delays, slack times, and resource constraints are analyzed to enhance project efficiency and risk mitigation.

This study not only demonstrates the practical application of CPM but also highlights the computational significance of graph theory in project scheduling. The findings from this project can serve as a framework for optimizing real-world construction projects, improving planning accuracy, and ensuring timely project completion.

**OBJECTIVE:**

* To apply the Critical Path Method (CPM) for efficient project scheduling in house construction.
* To utilize graph data structures for representing task dependencies.
* To implement forward pass and backward pass algorithms for critical path identification.
* To analyze slack times and potential delays for risk mitigation.
* To enhance resource allocation and project efficiency through computational analysis.

**TOOLS & TECHNOLOGIES:**

Programming Language: C.

Software: CodeBlocks.

**DATA STRUCTURE:** Graph (Directed Acyclic Graph).

**ALGORITHMS:** Forward Pass Algorithm, Backward Pass Algorithm.

**METHODOLOGY:**

To calculate the critical path, we will follow the following steps:

STEP 1: Calculate the Earliest Start Time (EST) and Latest Start Time (LST) for each activity.

STEP 2: Calculate the Earliest Finish Time (EFT) and Latest Finish Time(LFT) for each activity.

STEP 3: Calculate slack time.

The activities where there is no slack are the ones making up a critical path.

**FORWARD PATH ALGORITHM:**.

1. Initialize est[0] = 0 for the starting vertex .

2.For each activity(edge)(u,v):

1.Set est[u][v] = max(est[u], eft[u]), where eft[u] is early finish time of all preceding activities ending at vertex u.

2.Set eft[u][v] = est[u][v] + duration[u][v].

3.Repeat until all the vertices and edges are processed.

**BACKWARD PASS ALGORITHM:**

1. Initialize lft[n-1] = eft[n-1] for the last vertex .

2.For each activity(edge)(u,v):

1.Set lft[u][v] = min(lft [u], lst[u]), where lst[v] is the start time of all activities starting at vertex v.

2.Set lst[u][v] = lft[u][v] - duration[u][v].

3.Repeat until all the vertices and edges are processed in reverse topological order.

**CALCULATION:**

Duration(t): Indicates the time to complete an activity.

EST: This is the earliest time that an activity can be started assuming all previous activities have been completed beforehand. For activities that have more than one precedent, The EST is the greatest of the completion time of their precedents.

EFT: This is the earliest time that an activity can finish. It is equal to the EST + its estimated duration.

EFT = EST + t

LST: It is the latest time and activity can begin without delaying the whole project. It is equal to the Latest Finish Time (LFT) minus the expected duration of the activity (t).

LST = LFT – t

LFT:This is the latest time at which an activity can be completed without delaying the entire project. It is obtained by equaling the latest start time of the activity that immediately follows. It activities have more than one task immediately following them, the elective will be least of the start time of those activities**.**

Slack: A period of time when an activity can be delayed without causing the entire project to be delayed. All activities content in the critical path have zero slack.

S = LS - ES = LF - EF

**CONCLUSION:**

The implementation of the Critical Path Method in house construction demonstrates its effectiveness in optimizing project schedules and resource allocation. By leveraging graph data structures and computational algorithms, this project provides valuable insights into task dependencies, critical path identification, and delay mitigation. The findings highlight the importance of structured project management techniques in ensuring timely and cost-effective project completion, offering a practical approach for real-world applications.