IMPLEMENTATION OF SHORTEST PATH FINDER ALGORITHMS FOR UBER

A PROJECT REPORT

Submitted by

Abhaya Shrestha 16BCE2291

Nitesh Dhital 16BCE2295

Aayush Mallik 16BCE2296

CSE4014 - High Performance Computing (EPJ)

Project Guide

MANJULA V

Associate Professor

School of Information Technology and Engineering

B-Tech

IN

Computer Science and Engineering

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Fall Semester 2019-20

Abstract

The project deals with implementation of all Pair Shortest Path. This algorithm is implemented using parallel programming concept for faster solution. The purpose of developing this project is to find the shortest path between all the present nodes in a graph. This project can be implemented for Airline Systems, Transportation services(Uber), Courier Services, Networking etc. From the results it is observed that parallel algorithm is considerably effective for large graph sizes and MPI implementation is better in terms of performance over serial implementation.

Table of Contents

Introduction

Finding the shortest path between two objects or all objects in a graph is a common task in solving many day to day and scientific problems. The algorithms for finding shortest path find their application in many fields such as social networks, bioinformatics, aviation, routing protocols, Google maps etc. Shortest path algorithms can be classified into two types: single source shortest paths and all pair shortest paths. There are many different algorithms for finding the all pair shortest paths. Some of them are Floyd-Warshall algorithm and Johnson's algorithm. All pair shortest path algorithm which is also known as Floyd-Warshall algorithm was developed in 1962 by Robert Floyd. This algorithm follows the methodology of the dynamic programming. The algorithm is used for graph analysis and finds the shortest paths (lengths) between all pair of vertices or nodes in a graph. The graph is a weighted directed graph with negative or positive edges. The algorithm is limited to only returning the shortest path lengths and does not return the actual shortest paths with names of nodes.

Problem Statement and Objective

The major problem is to resolve the issues that rise by using the Dijkstar's Algorithm and Bellman Ford Algorithm . The problem with Dijkstar's Algorithm is that it does not work for negative nodes and the Bellman Ford is that it won't work for loops. The primary goal of this project is to implement the shortest path finder algorithms in Parallel to find the shortest distance between two nodes in map.

Literature Review

1. Shortest Transportation Route Network in Nigeria Using FloydWarshall's Algorithm

Esuabana, Ita Micah Ikpang, Ikpang Nkereuwem Okon, Ekom-obong Jackson (2015)

Abstract

This study presents the application of Floyd – Warshall algorithm, which is an all pairs shortest path algorithm in finding the shortest route network for major cities in Nigeria. Twenty one (21) city route networks in Nigeria showing distances (km) are considered with Calabar, Cross River State as the origin node and Kaduna, Kaduna State as the sink node. Distance and precedence matrices are computed for all iterations to obtain the weight between nodes in the network and the shortest route respectively. The optimal route for all pairs in the network and the total distance travelled from one node to another are obtained respectively from the precedence and distance matrices of the final iteration. Detailed results showed that the algorithm is efficient. The designed route network shows the shortest route for all pairs of nodes in the network and also exposes hidden shortest routes. These routes are recommended for inter-city transportation in Nigeria.

Problem

In this research paper they presented the application of Floyd – Warshall algorithm in finding the shortest route network for major cities in Nigeria. But the problem is once they set the data about the route into the system they have to modify every time manually if there is any constructions happening in a particular

route . So, this is not real-time based idea. For that mobile application should be made with GPS enabled feature. And by implementing this idea into that application the shortest path can be find in Nigeria as well as other countries.

Methodology Adopted

Intercity distances (km) for twenty one (21) cities with respect to the origin node used in this study were collected and the Floyd-Warshall algorithm was employed to find the optimal routes for all pairs in the network.

Performance Limitations

Here, the performance limitations is soley based upon the GPS system's fault. If we can resolve it with alternatives no such limitations are there for the proposed paper.

Proposed alternative methodology

The problem heavily depends on the use of GPS system. It has been found that the GPS cannot be trusted always. It has been linked with inaccuracy multiple number of instances. GLONASS is a new technology being developed by Russia. This can be implemented for the project.

2. Floyd-warshall algorithm to determine the shortest path based on android.

Author- Ramadiani, D Bukhori, Azainil and N Dengen(2018)

Abstract

The development of technology has made all areas of life easier now, one of which is the ease of obtaining geographic information. The use of geographic information may vary according to need, for example, the digital map learning, navigation systems, observations area, and much more. With the support of adequate infrastructure, almost no one will ever get lost to a destination even to foreign places or that have never been visited before. The reasons why many institutions and business entities use technology to improve services to consumers and to streamline the production process undertaken and so forth. Speaking of the efficient, there are many elements related to efficiency in navigation systems, and one of them is the efficiency in terms of distance. The shortest distance determination algorithm required in this research is used Floyd-Warshall Algorithm. Floyd-Warshall algorithm is the algorithm to find the fastest path and the shortest distance between 2 nodes, while the program is intended to find the path of more than 2 nodes.[1]

Problem

In this research paper they made an application based android donation pickup using the application of digital maps. Particularly in this research, the shortest route selection algorithm named Floyd-Warshall Algorithm is applied to a digital map of donation pickup program, which is expected to streamline the distance that must be passed by donation pickup institution to donor location. The concept used in this research paper can only useful to android user but not to ios user . As , nowadays people are using ios devices just like android. Production is also same like android devices.

Methodology Adopted

The focus of the discussion in this study using the Floyd-Warshall algorithm is the shortest path selection between 2 or more marker positions or location markers in a digital map. The Floyd-Warshall algorithm compares each route variation with the distance from each point to the destination point and accumulates the distance between the nodes (intersection points) that are passed. The line that connects between nodes has a value in the magnitude of the number indicating the distance weight. The Floyd-Warshall algorithm will take each line as a calculation material in determining the shortest route.

Performance Limitations

The softwares that are used for developing apps are limited to the flexibility provided by the development kit. The user can only implement the algorithm according to the tools that have been provided previously in the development kit.

Proposed alternative methodology

The proposed solution for this limitation is to develop application for ios user too. IOS applications can be made using Swift, Ruby Rail etc software. So that this whole system works perfect for all the users .

3. Parallelization of Shortest Path Finder on GPU: Floyd-Warshall

[Dhananjay Kulkarni, Neha Sharma, Prithviraj Shinde and Vaishali Varma - Department of Computer Engineering, Nasik(2015)]

Abstract

This project deals with implementation of Floyd Warshall Algorithm i.e. All Pair Shortest Path and is implemented using parallel programming concept for faster solution. This is a research-based project in which the serial and parallel computations are compared. Floyd Warshall algorithm has overcome the drawbacks of Dijkstra's and Bellman Ford Algorithm. For parallel programming, the project is implemented using NVIDIA GPU (NVIDIA GeForce 820M, 410M) for which CUDA (CUDA Toolkit 6.0) is used. The purpose of developing this project is to find the shortest path between all the present nodes in a graph. This system is designed to work on a large dataset (set of 48 or 72 or 100 cities). This project can be implemented for Airline Systems, Transportation services, Courier Services, Networking.[3]

Problem:

Problems that are not well mapped are usually too small or too unpredictable. These very small problems lack the parallelism required to use all the threads on the GPU and/or may be suitable for low-level caching on the CPU, which greatly improves CPU performance. Unpredictable problems have too many meaningful branches, which may prevent data from being efficiently streamed from the GPU memory to the kernel or reduce parallelism.

Methodology Adopted

The Project was divided into six modules.The modules are as follows: Read Data from file. Convert raw file data into matrix form. Launch GPU Kernel. Compute shortest Path. Return Output from GPU to CPU. Display output on CPU.
Performance Limitations

The performance limitations are based on the current state of the GPU's. Depending on the system and the GPU being used, the result can be vastly affected.

Proposed alternative methodology

The project should be made dynamic such that it can handle large datasets as well as small datasets by being capable of deciding when to use the GPU and CPU without external influence.

4. Parallelization of Shortest Path Algorithm Using OpenMP and MPI

[Rajashri Awari, Dept.of Info.Tech. MTech., Yeshwantrao Chavan College of Engineering,2017]

Abstract

Graph problems are solved by using the standard graph Algorithm. Example is Large matrix problems, graphs solving problems, equations etc. There is a representation of two algorithms. The algorithm to find all pair shortest path is Floyd Warshall algorithm, single source shortest path is Dijkstra's algorithm. These two algorithms implement in serial formulation. Parallel algorithm is considerably effective for large graph size. By using these algorithms find the shortest distance, such as city to city, routing, networking, social media. Parallel algorithm used for calculating or finding shortest path of graph. With the help of graph algorithm these operations can be done in parallel and reduce the computation time and efficiency. All pair shortest path problem applies for directed and undirected graph whose finite nodes and edges in un-weighted and undirected graph.[4]

Problem

OpenMP and MPI can be considered sufficient when the data sets are small or even moderate but when the data sets are large, the performance might decline significantly.

Methodology Adopted

The two algorithms used in this paper as Floyd Warshall and Dijkstra's Algorithm. These Algorithms implemented in c, c++ for serial formulation. For large graph matrix problem solve these two algorithms as serially. By using dataset as large matrix got from search engine which consist node to node and its weight. Large matrix size consists as 500*500, 1000*1000, 2000*2000. Using large size matrix converted algorithm in serial form implanted in c, c++. And the serial form can be converted into the parallel form using OpenMP and MPI for reducing or consuming time and the efficiency. Steps involved in this project are ; 1. Serial Implementation of Floyd and Dijkstra's algorithm. 2. Parallel Implementation of Floyd and Dijkstra's Algorithm. 3. Comparison of serial and parallel.

Performance Limitations

Here, again the major problem is dependent on the system being implemented. If we are to perfrom tasks on large data sets, a better GPU or a system that enables user to process large set of datas is required.

Proposed alternative methodology

If the problem is small enough to accommodate and enough computing resources such as core and memory, then OpenMP is a good choice. And when the data size is moderate and the problem is computationally intensive, MPI can be considered a framework. Hence, MapReduce can be used as it is an excellent framework when the data size is large and the task does not need to be iteratively processed.

5. Comparative Analysis of Floyd Warshall and Dijkstras Algorithm using Opencl

[Asad Mohammad, Vikram Garg - Gyan Ganga College of Technology, International Journal of Computer Applications (0975 – 8887),(2015)]

Abstract

Shortest path algorithms find applications in large real-world domains. All pair shortest path (APSP) and single source shortest path (SSSP) both have their special applications domains. All though every SSSP can be applied for all vertices to calculate APSP. But APSP can't. In this paper heterogeneous implementation of Floyd Warshalls algorithm and Dijkstra's algorithm is compared on dense graphs have positive edge weights ranging from 1 to 10. It is found that Dijkstra's algorithm is better than Floyd Warshall algorithm in sequential implementation. But as there is less parallelism identified in Dijkstra algorithm as compared to parallel to parallel FW gives less execution time as compared to Dijkstra's.[5]

Problem

TPU is a better solution than GPU. The usage of Tensor Processing Unit results in the enhancement of the conclusions that were drawn from the research paper. But, TensorFlow has limited support for OpenCL and AMD GPUs. We can build Tensorflow with SYCL (single source OpenCL) support, but performance might not be as good as with NVIDIA GPUs.

Methodology Adopted

In this paper Dijkstra algorithm is compared with BellmanFord algorithm on which three techniques Workfront Sweep, Near-Far and Bucketing are applied. All these algorithms are studied for different data structures and traversal techniques. In Floyd Warshall and Dijkstra algorithm are compared by applying divide and conquer on FW to make use of multi GPU cluster using OpenCL. In this paper Floyd Warshall algorithms is compared with Dijkstra algorithm for dense graphs and Dijkstra algorithm is also compared with FW with its APSP implementation.

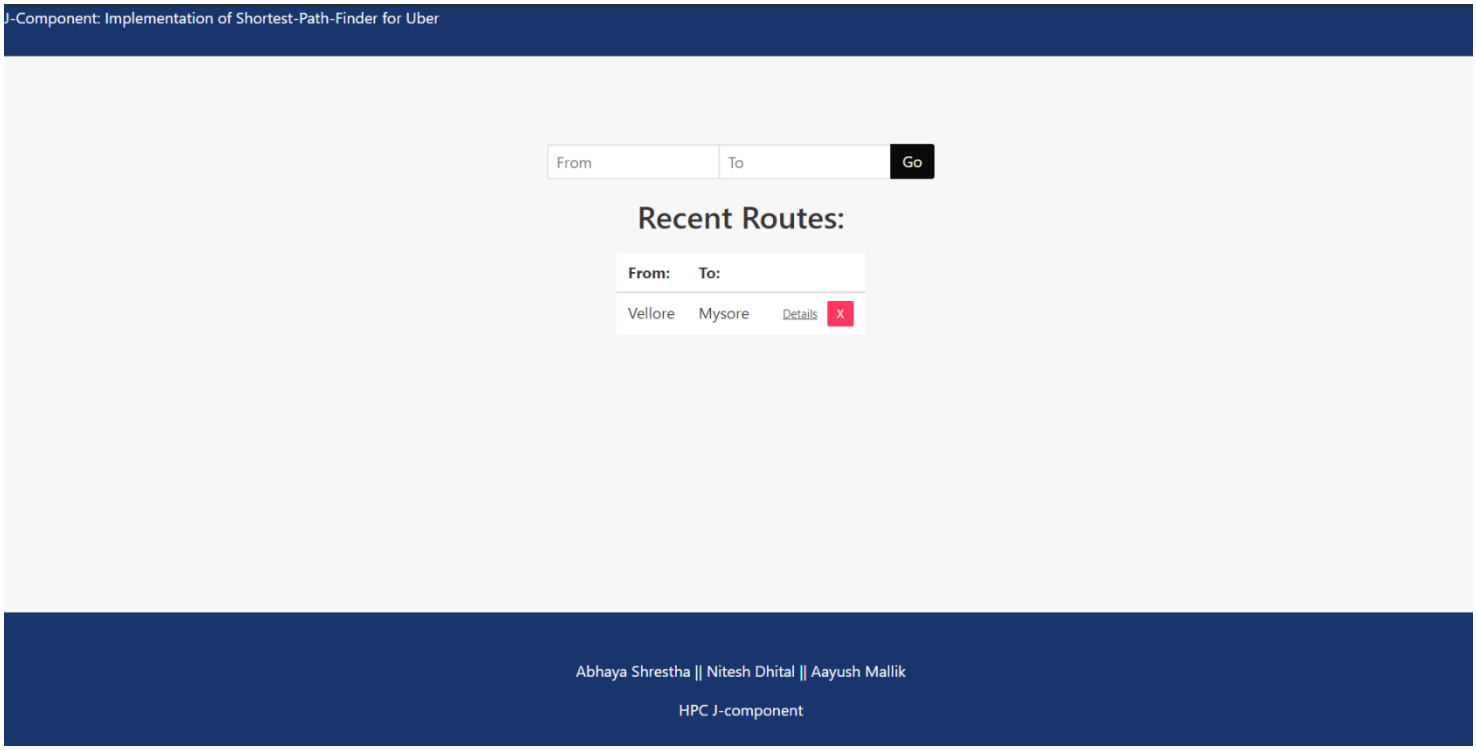
Performance Limitations

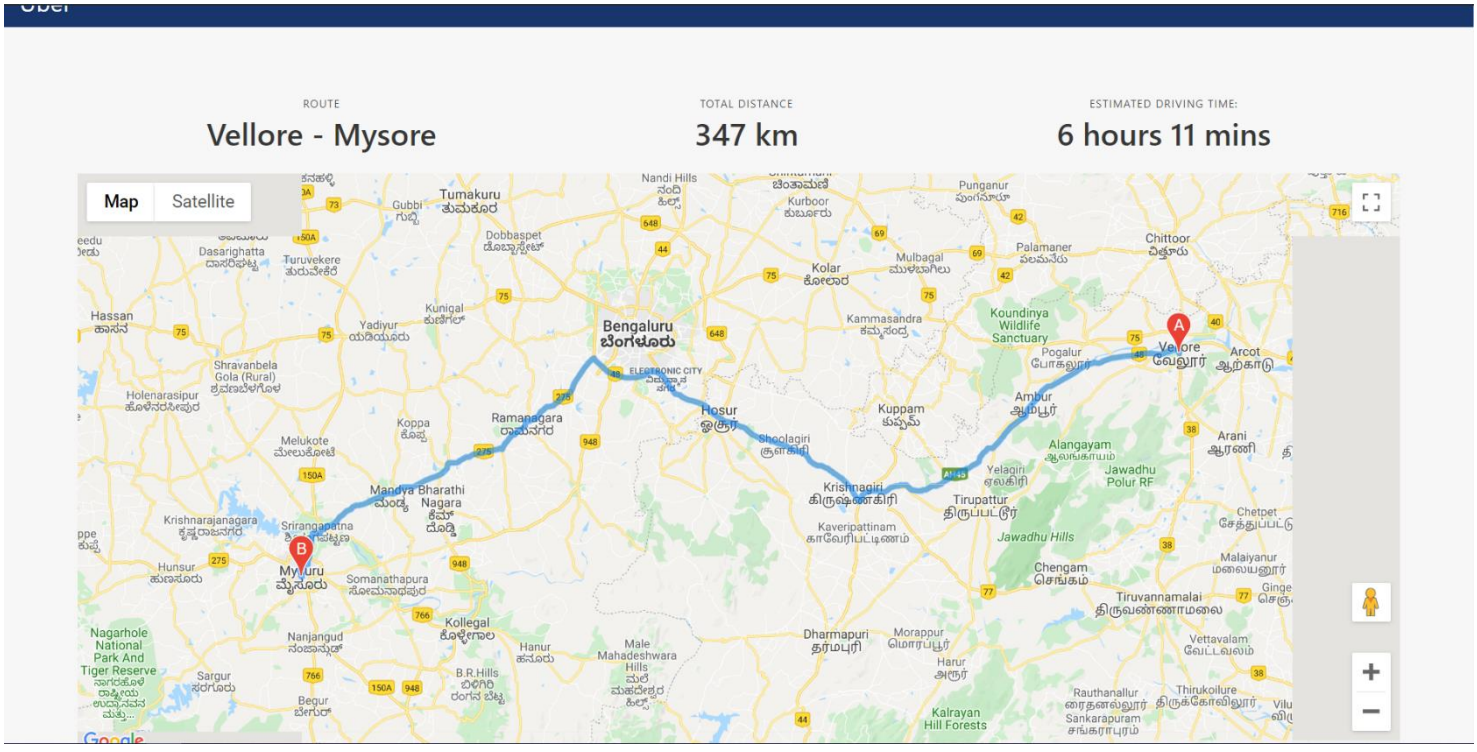
The GPU's have be subjected to start slower and stop slower, though it does more tasks than the normal CPU. Without special connectors, the communication in the desktop system has been found to be pretty slow.

Proposed alternative methodology

Since, TPU provides better conclusions compared to the use of GPU but has less compatibility. Better support should be developed for TPU such that we can obtain optimal results.

Implementation





Testing and Performance Evaluation

We tested the proposed system and we calculated the shortest distance between nodes. We have implemented the distance matrix which is one of the most used algorithms used by Google for finding the distance between two nodes. The matrix itself is used for calculating the distance so that the maximum speed up is already achieved. There could be further time improvement but that would depend upon the high performance clusters used for image processing of the satellite images.

Conclusion and Future Enhancements

In this project, we developed parallel implementations of shortest path graph algorithm known as Floyd-Warshall, which were used to find the all-pairs shortest path in transportation networks. Both the sequential and parallel implementation is done for the Floyd-Warshall algorithm. The implemented Floyd-Warshall algorithm had theoretical computational complexity, which was verified experimentally. Here we implemented Floyd-Warshall parallelly with the help of MPI Thus, the system implements Parallelization of shortest path finder algorithm: Floyd-Warshall. This system can be implemented in the areas where the operation is to be performed on large dataset and the computation time should be less hence to reduce the time of computation parallelization can be implemented using GPU.

Appendix

Coding

```
import { ADD_ROUTE, REMOVE_ROUTE } from
"./types";
// in this file are action creators, and thet are self
descriptive,
// so no need to add extra comments
export function addRoute(route) {
  return dispatch => {
    dispatch({ type: ADD_ROUTE, payload: route });
  };
}
export function removeRoute(id) {
  return dispatch => {
    dispatch({ type: REMOVE_ROUTE, payload: id });
  };
}
export const ADD_ROUTE = 'ADD_ROUTE';
export const REMOVE_ROUTE =
'REMOVE_ROUTE';
////
import React, { Component } from 'react';

import {BrowserRouter as Router, Route, Switch} from
'react-router-dom';



import LandingPage from '../containers/LandingPage';
import RouteDetails from './RouteDetails';
import Header from './Header';
```

```
 import Footer from './Footer';
export default class App extends Component {
  render() {
    return (
      <div>
        <Router>
          <div id="main-wrapper">
            <Header />
            <main>
              <Switch>
                <Route exact path="/"
 component={LandingPage} />
                <Route path="/route/:id"
 component={RouteDetails} />
                <Route render={() => <h3 className="title
 is-3 error-message">Not found</h3>} />
              </Switch>
            </main>
            <Footer />
          </div>
        </Router>
      </div>
    );
  }
 }

 Main.html
 <!DOCTYPE html>
 <html>
  <head>
    <meta name="viewport" content="width=device-
 width, initial-scale=1">
    <link rel="stylesheet"
 href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.5.2/
```

```
css/bulma.min.css">
    <script
src="https://maps.googleapis.com/maps/api/js?key=AIza
SyD-qK2ZPbet_B7BtUDOkU-
ChOKuArHr0ck"></script>
    <title>Pathfinder</title>
  </head>
  <body>
    <div id="root"></div>
  </body>
</html>
```

Javascript File

```
import React from 'react';
import ReactDOM from 'react-dom';
import { Provider } from 'react-redux';
import { createStore, applyMiddleware } from 'redux';
import reduxThunk from 'redux-thunk';


// including custom styles
require('./styles/style.css');

import App from './components/App';
import reducers from './reducers';

// creating the store with redux-thunk middleware
const createStoreWithMiddleware =
applyMiddleware(reduxThunk)(createStore);

ReactDOM.render(
  <Provider
store={createStoreWithMiddleware(reducers)}>
    <App />
  </Provider>
```

, document.getElementById('root'));

References

[1] Ramadiani, Ramadiani & Bukhori, D & Azainil, Azainil & Dengen, N. (2018). IOP Conference Series: Earth and Environmental Science. 144. 012019. 10.1088/1755-1315/144/1/012019.

[2] Esuabana, I.M., Ikpang, I.N., & Okon, E.J. (2015). Shortest Transportation Route Network in Nigeria Using Floyd- Warshall's Algorithm.

[3] Kulkarni, D., Sharma, N., Shinde, P., & Varma, V. (2015). Parallelization of Shortest Path Finder on GPU: Floyd-Warshall.

[4] R. Awari, "Parallelization of shortest path algorithm using OpenMP and MPI," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, 2017, pp. 304-309.

[5] Mohammad, Asad and Vikram Vinod Garg. “Comparative Analysis of Floyd Warshall and Dijkstras Algorithm using Opencl.” (2015).