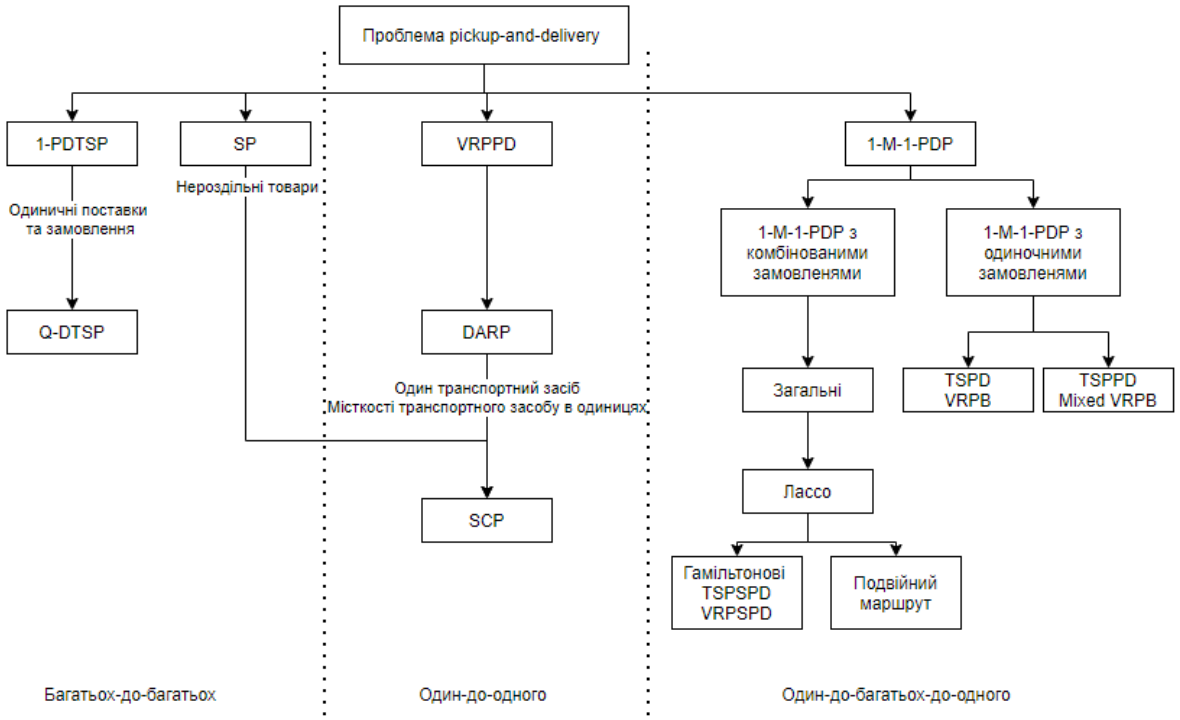
Додаток А

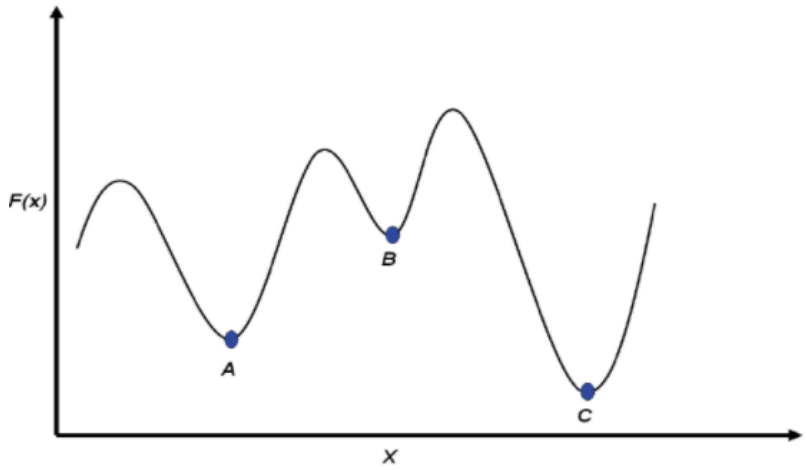
Графічний матеріал атестаційної роботи

КЛАСИФІКАЦІЯ PDP



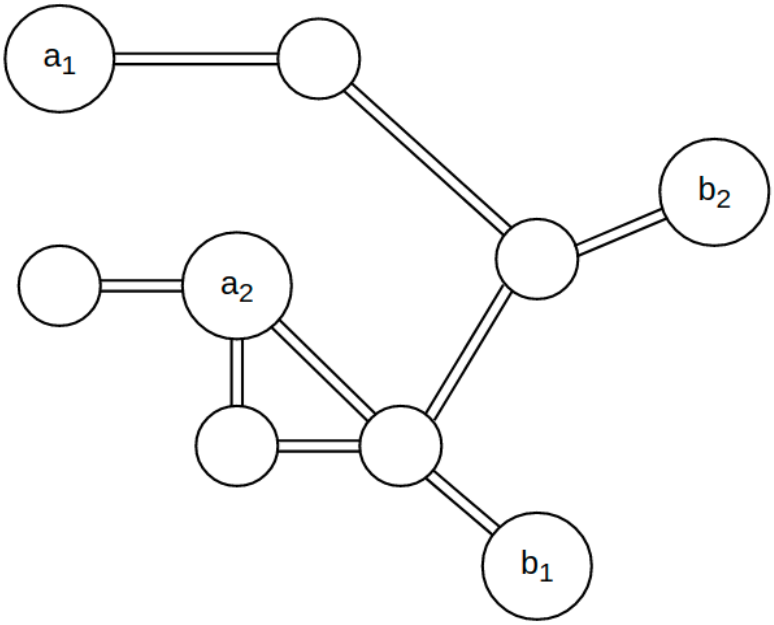
Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

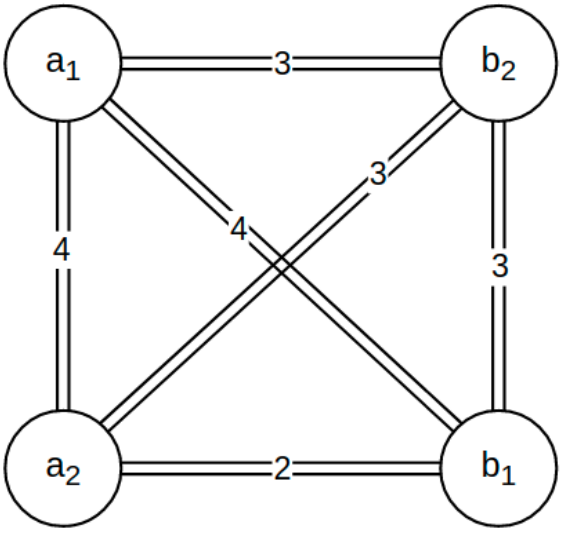
ПОШУКОВИЙ ПРОСТІР: ЛОКАЛЬНІ ТА ГЛОБАЛЬНИЙ МІНІМУМ



Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

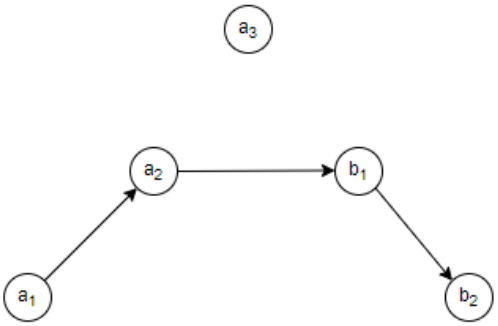
ЗВЕДЕННЯ ДО ЗАДАЧІ PDP

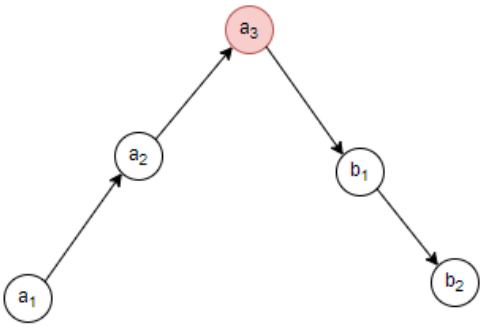




Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

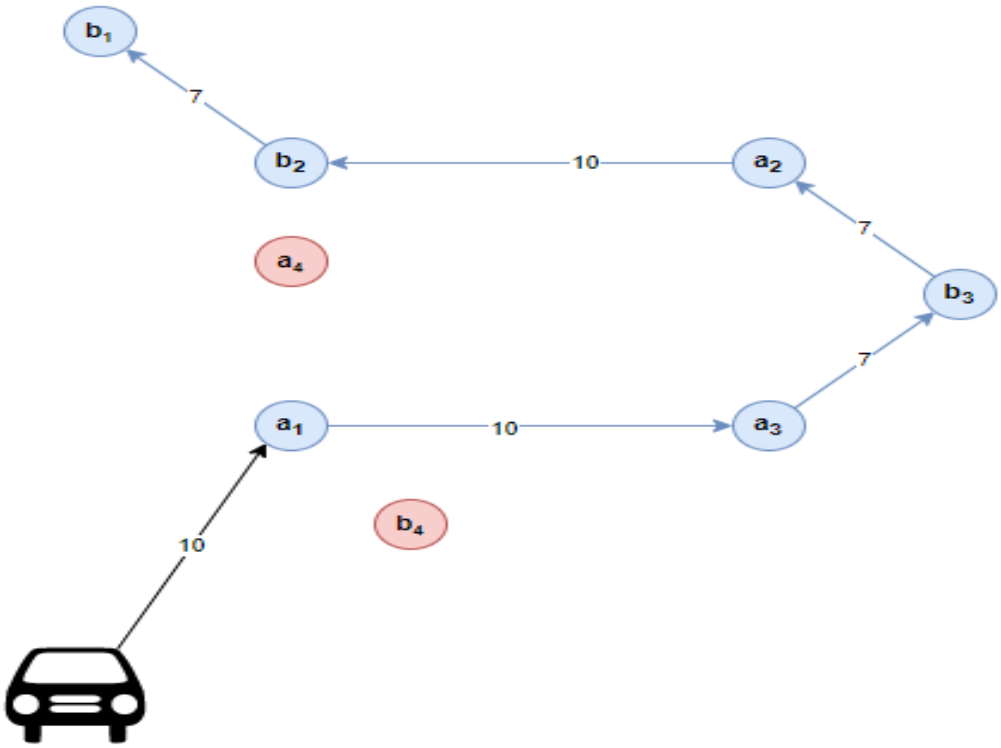
ДОДАВАННЯ НОВОЇ ТОЧКИ ДО ШЛЯХУ





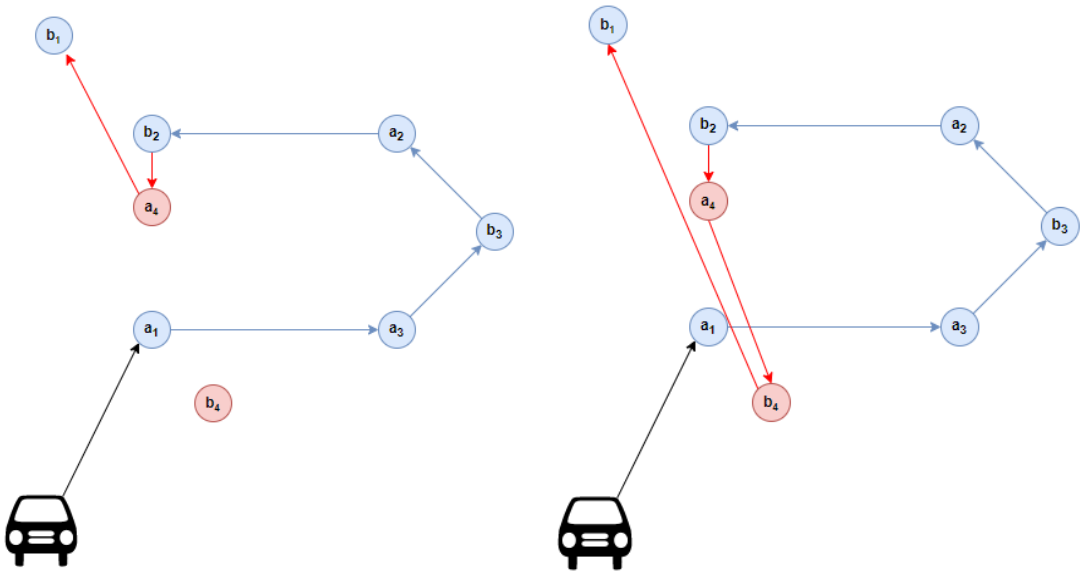
Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

ДОДАВАННЯ НОВОГО ЗАМОВЛЕННЯ



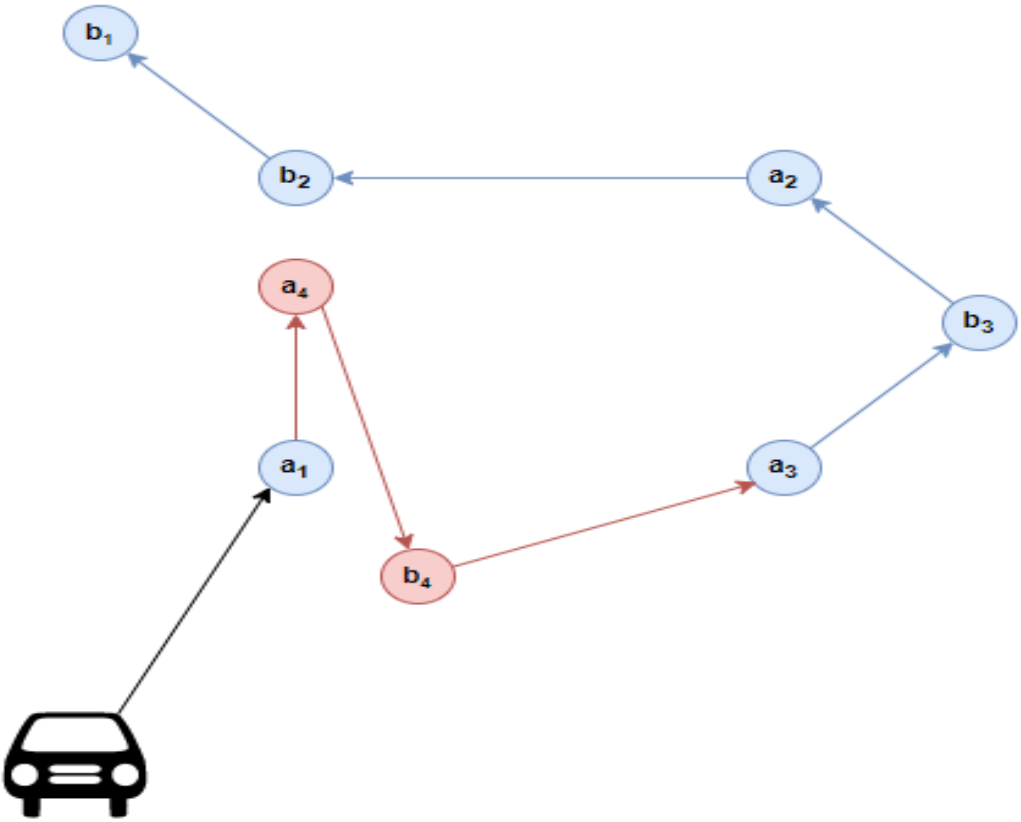
Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

ДОДАВАННЯ НОВОГО ЗАМОВЛЕННЯ НЕ ОПТИМАЛЬНИЙ ВАРІАНТ



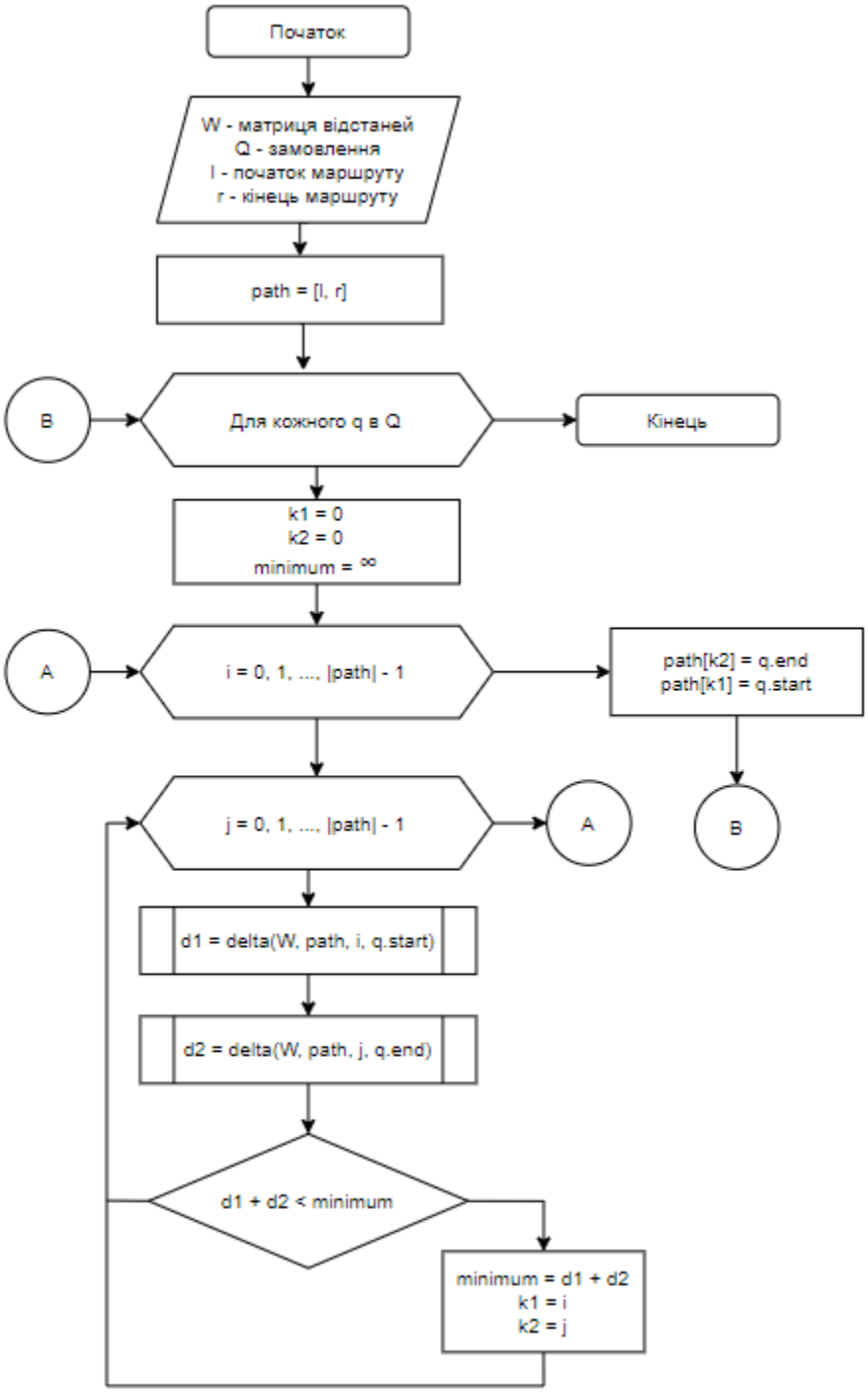
Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

ДОДАВАННЯ НОВОГО ЗАМОВЛЕННЯ ОПТИМАЛЬНИЙ ВАРІАНТ



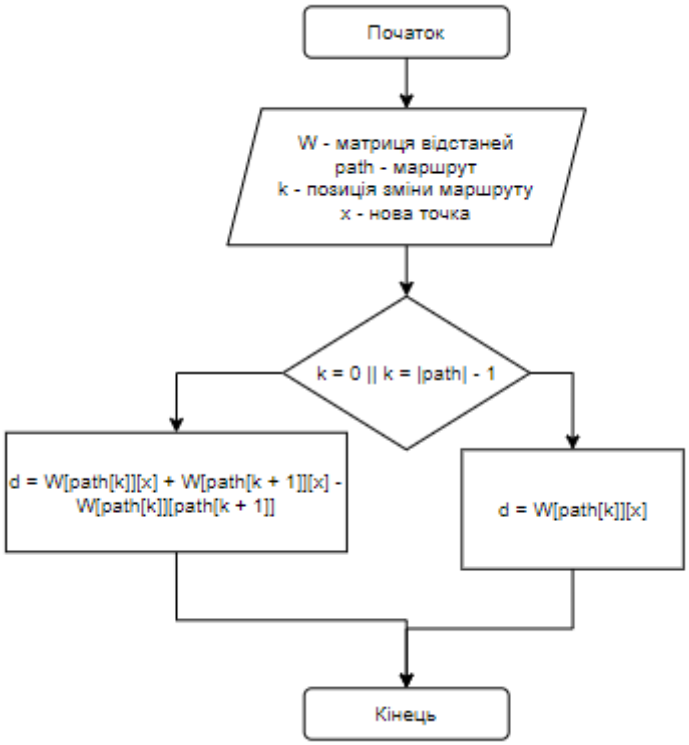
Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

СХЕМА РОЗРОБЛЕНОГО АЛГОРИТМУ



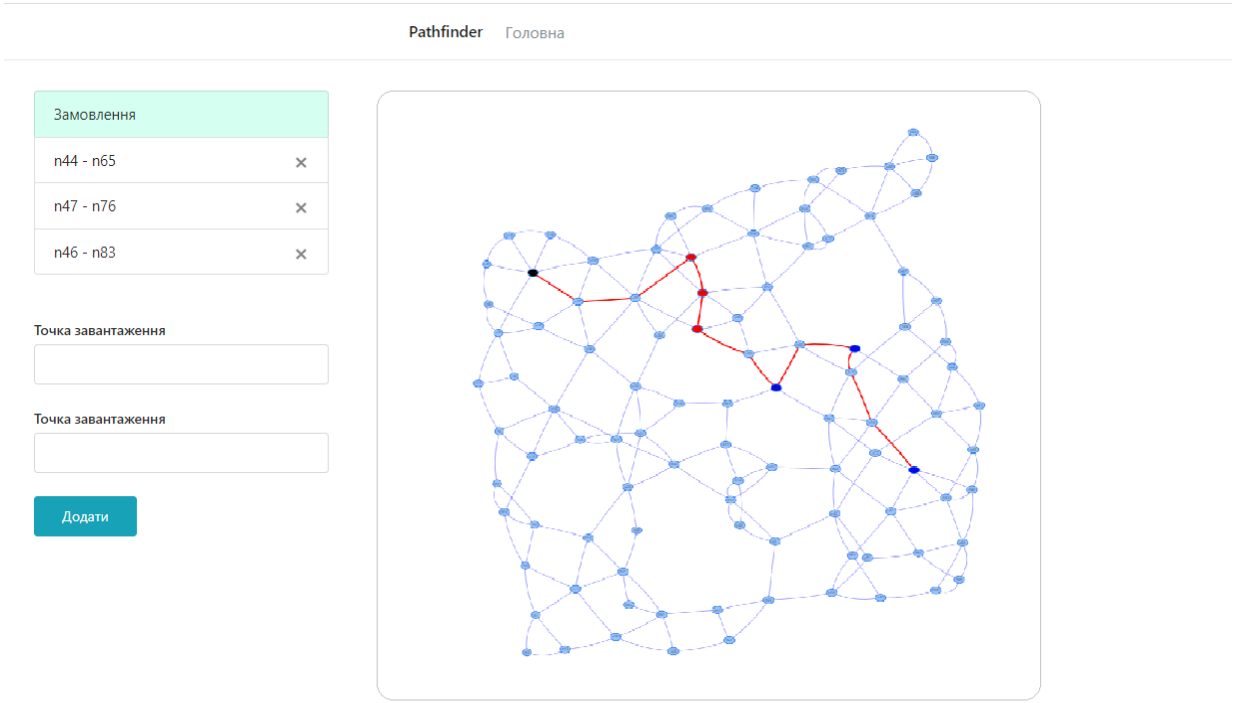
Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

СХЕМА АЛГОРИТМУ ОЦІНКИ ЗМІНИ МАРШРУТУ



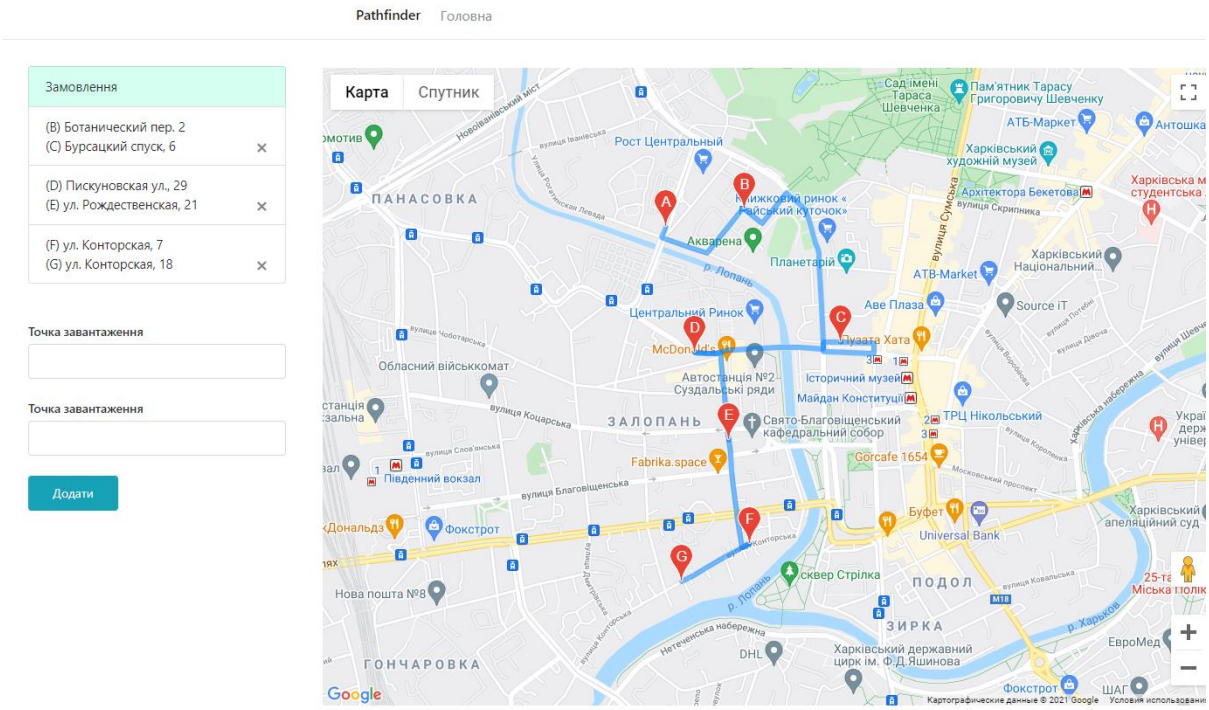
Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

ІНТЕФЕЙС РОБОТИ З ГРАФОМ



Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

ІНТЕРФЕЙС РОБОТИ З МАПОЮ



Розроб.	*Паречин В.П.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*	
Перевір.	*Проф. Калита Н.І.*				
Н. Контр.	*Проф. Калита Н.І.*				
				СПРм-19-2	*Лист 1*
Затверд.	*Гребеннік І.В.*			*СТ*	*Листів 1*

Додаток Б

Програмні документи

ЗАТВЕРДЖЕНО

ГЮИК.502520.008 – 01 12 01 – ЛУ

Дослідження та аналіз ефективності методів розв’язування задач на графах у системному проектуванні

Текст програми

ГЮИК.502520.008 – 01 12 01 – ЛУ

ЛИСТІВ 17

2021 р.

Міністерство освіти і науки України
Харківський Національний Університет Радіоелектроніки

«ЗАТВЕРДЖУЮ»
керівник кваліфікаційної роботи
проф. Калита Н. І.

Дослідження та аналіз ефективності методів розв’язування задач на графах у системному проектуванні

Текст програми

ЛИСТ ЗАТВЕРДЖЕННЯ

ГЮИК.502520.008 – 01 12 01 – ЛУ

РОЗРОБИВ:
ст. гр. СПРм-19-2
Паречин В. П.

2021 р

app.vue

```
<template>
  <div id="app">
    <header />
    <router-view></router-view>
  </div>
</template>

<script>
import Header from "@/components/Header";
export default {
  name: "App",
  components: {
    Header,
  }
};
</script>
```

Header.vue

```
<template>
  <nav>
     …….
  </nav>
</template>

<script>
import { mapGetters } from "vuex";
export default {
  name: "Header",
};
</script>
```

GraphView.vue

```
<template>
     …….
</template>

<script>
export default {
  name: "Graph",
  data() {
      return {
        nodes: [],
        edges: []
      };
  },
  props: {
    nodes: Array,
    edges: Array
  },
  emits: ['node:select'],
  methods: {
    render: function() {
        this.nodes = new vis.DataSet(this.nodes);
        this.edges = new vis.DataSet(this.edges);
        let data = {
            nodes: nodes,
            edges: edges
        };
        let options = {};
        let network = new vis.Network(container, data, options);
        network.on("selectNode", function (props) {
            $emit('node:select', props)
        });
    },
```

```
    setColor: function (n, color) {
      let node = this.nodes.get(n)
      node.color = {background: color};
      nodes.update(node)
    },

    setEdgeColor: function setEdgeColor(n, color, size) {
      let edge = this.edges.get(n)
      edge.color = color;
      edge.width = size;
      edges.update(edge)
    },

    colorPath: function colorPath(path, params) {
      for (var i = 0; i < path.length - 1; i++) {
        setEdgeColor(toEdgeId(path[i], path[i + 1]), params.color,
params.boldValue);
      }
    },

    clearPath: function colorPath(path) {
      for (var i = 0; i < path.length - 1; i++) {
        setEdgeColor(toEdgeId(path[i], path[i + 1]), "blue", 1);
      }
    }
  }
};</script>
```

GoogleMapLoader.vue

```
<template>
  <div>
    <div
      class="google-map"
      ref="googleMap"
    ></div>
    <template v-if="Boolean(this.google) && Boolean(this.map)">
      <slot
        :google="google"
        :map="map"
      />
    </template>
  </div>
</template>

<script>
import GoogleMapsApiLoader from "google-maps-api-loader";

export default {
  props: {
    mapConfig: Object,
    apiKey: String
  },

  data() {
    return {
      google: null,
      map: null
    };
  },
```

```
 async mounted() {

    const googleMapApi = await GoogleMapsApiLoader({
      apiKey: this.apiKey
    });
    this.google = googleMapApi;
    this.initializeMap();
  },

  methods: {
    initializeMap() {
      const mapContainer = this.$refs.googleMap;
      this.map = new this.google.maps.Map(mapContainer,
this.mapConfig);
    }
  }
};
</script>
```

Marker.vue

```
<script>
import { cfg } from "@/constants/mapSettings";

export default {
  props: {
    google: {
      type: Object,
      required: true
    },
    map: {
      type: Object,
      required: true
```

```
    },
    marker: {
      type: Object,
      required: true
    }
  },

  mounted() {
    const { Marker } = this.google.maps;

    new Marker({
      position: this.marker.position,
      marker: this.marker,
      map: this.map,
      icon: cfg
    });
  },

  render() {}
};
</script>
```

PathRenderer.vue

```
<template>
  <GoogleMapLoader
    :mapConfig="mapConfig"
    apiKey="xxx"
  >

    <template slot-scope="{ google, map }">
      <GoogleMapMarker
        v-for="marker in markers"
        :key="marker.id"
```

```
        :marker="marker"
        :google="google"
        :map="map"
      />
      <GoogleMapLine
        v-for="line in lines"
        :key="line.id"
        :path.sync="line.path"
        :google="google"
        :map="map"
      />
    </template>
  </GoogleMapLoader>
</template>

<script>
import GoogleMapLoader from "./GoogleMapLoader";
import GoogleMapMarker from "./GoogleMapMarker";
import GoogleMapLine from "./GoogleMapLine";

import { mapSettings } from "@/constants/mapSettings";

export default {
  components: {
    GoogleMapLoader,
    GoogleMapMarker,
    GoogleMapLine
  },

  props: {
    markers: Array,
    lines: Array
  },
```

```
  computed: {
    mapConfig() {
      return {
        ...mapSettings,
        center: this.mapCenter
      };
    },

    mapCenter() {
      return this.markers[1].position;
    }
  }
};
</script>
```

Line.vue

```
<script>
import { LINE_PATH_CONFIG } from "@/constants/mapSettings";

export default {
  props: {
    google: {
      type: Object,
      required: true
    },
    map: {
      type: Object,
      required: true
    },
    path: {
      type: Array,
```

```
        required: true
      }
    },

    mounted() {
      const { Polyline } = this.google.maps;
      new Polyline({
        path: this.path,
        map: this.map,
        ...LINE_PATH_CONFIG
      });
    },

    render() {},
  };
  </script>
```

Api.js

```
  class Api {
    host = "localhost:4502/api";

    getGrapth(id) {
      return axios.get(host + "/graph", {
        params: {
          id: id
        }
      }).then(response => response.data)
    }

    uploadGrapth(data) {
      return axios.put(host + "/upload/graph", data, {
        headers: {
          'Content-Type': 'multipart/form-data'
```

```
      }
    }).then(response => response.data)
  }

  getSolution(id, location, targetLocation, orders) {
    return axios.get(host + "/solve", {
      params: {
        grapthId: id,
        currentLocation: location,
        targetLocation: targetLocation,
        orders: orders
      }
    }).then(response => response.data)
  }

  addOrderToSolution(id, path, newOrders) {
    return axios.get(host + "/isolve", {
      params: {
        grapthId: id,
        currentSolution: path,
        newOrders: newOrders
      }
    }).then(response => response.data)
  }
}
export default new Api();
```

Router.js

```
var express = require('express');
var fileupload = require("express-fileupload");
var graphStorage = require("graphStorage");
var solver = require("solver");
```

```
var app = express();
app.use(fileupload());

app.get('/graph', function(req, res) {
  res.send(graphStorage.get(req.query.id));
});

app.put('/upload/graph', function(req, res) {
  res.send(graphStorage.save(req.files.grapth));
});

app.put('/solve', function(req, res) {
  res.send(
    solver.solve({
      graph: graphStorage.get(req.query.id),
      startPoint: req.query.startLocation,
      endPoint: req.query.targetLocation,
      pairs: req.query.orders
    })
  );
});

app.put('/isolve', function(req, res) {
  res.send(
    solver.solve({
      graph: graphStorage.get(req.query.id),
      currentSolution: req.query.path,
      pairs: req.query.newOrders
    })
  );
});
```

GraphStorage.js

```
class GraphStorage {
  constructor() {
    this.storage = StorageUtils.initStorage();
  }

  get(id) {
    let graph = this.storage.get(id);
    return {
      nodes: graph.getNodes(),
      edges: graph.getEdges()
    };
  }

  save(file) {
    let costMatrix = CsvParser.parse(file);
    let nodes = [];
    let edges = [];

    for (let i = 0; i < costMatrix.length; i++) {
      nodes.push({
        name: costMatrix[0][i],
        index: i
      })
    }

    for (let i = 1; i < costMatrix.length; i++) {
      for (let j = 0; j < costMatrix[0].length; j++) {
        edges.push({
          from: i,
          to: j
        })
      }
```

```
    }

    this.storage.save(new Graph(nodes, edges));
  }
}

exports.graphStorage = new GraphStorage();
```

GraphSearch.js

```
class GraphSearch {

  constructor(graph) {
    this.distances = this.initArray(graph.length, Infinity);
    this.next = this.initArray(graph.length, null);

    for (let i = 0; i < graph.length; i++) {
      for (let j = 0; j < graph.length; j++) {
        if (graph[i][j]) {
          this.distances[i][j] = graph[i][j];
          this.next[i][j] = j;
        }
      }
    }

    for (let i = 0; i < graph.length; i++) {
      this.distances[i][i] = 0;
      this.next[i][i] = i;
    }

    for (let k = 0; k < graph.length; k++) {
      for (let i = 0; i < graph.length; i++) {
        for (let j = 0; j < graph.length; j++) {
```

```
          if  (this.distances[i][j] > this.distances[i][k] +
this.distances[k][j]) {
            this.distances[i][j] = this.distances[i][k] +
this.distances[k][j];
            this.next[i][j] = this.next[i][k];
          }
        }
      }
    }
  }

  getPath(u, v) {
    if (this.next[u][v] == null) {
      return []
    }

    var path = [u]
    while (u != v) {
      u = this.next[u][v]
      path.push(u)
    }
    return path
  }

  initArray(size, value) {
    let res = new Array(size);
    for (let i = 0; i < size; i++) {
      res[i] = new Array(size);
      for (let j = 0; j < size; j++) {
        res[i][j] = value;
      }
    }
    return res;
```

```
  }
}
```

Solver.js

```
class Solver {
  solve(graph, start, end, pairs) {
    let path = [start, end]
    if (!graph.solved) {
      graph.solved = new GraphSearch(graph);
    }

    pairs.forEach((pair) => {
      let k1 = 0
      let k2 = 0
      let minimum = Infinity
      for (let i = 0; i < path.length; i++) {
        for (let j = 0; j < path.length; j++) {
          delta = this.getDelta(path, pair.from, i, graph.solved) +
this.getDelta(path, pair.to, j, graph.solved);
          if (delta < minimum) {
            minimum = delta;
            k1 = i;
            k2 = j;
          }
        }
      }
    })
  }

  isolve(graph, path, pairs) {
    pairs.forEach((pair) => {
      let k1 = 0
```

```
      let k2 = 0
      let minimum = Infinity
      for (let i = 0; i < path.length; i++) {
        for (let j = 0; j < path.length; j++) {
          delta = this.getDelta(path, pair.from, i, graph.solved) +
this.getDelta(path, pair.to, j, graph.solved);
          if (delta < minimum) {
            minimum = delta;
            k1 = i;
            k2 = j;
          }
        }
      }
    })
  }

  getDelta(path, node, k, searchGraph) {
    if (k == 0 || k == path.length - 1) {
      return searchGraph.distances[path[k]][node.index]
    } else {
      return searchGraph.distances[path[k]][node.index] +
searchGraph.distances[path[k + 1]][node.index] -
searchGraph.distances[path[k]][path[k + 1]]
    }
  }
}

exports.solver = new Solver()
```

№	Позначення	Назва	Примітки
		Текстові документи	
1.	ГЮИК. 502520.018 ПЗ	Пояснювальна записка	64 с.
3.	ГЮИК.502520.008 – 01 12 01	Текст програми	17 с.
		Графічні документи	
		Класифікація PDP	Включено до ПЗ
4.		Пошуковий простір: локальний та глобальний мінімум	Включено до ПЗ
5.		Зведення до задачі PDP	Включено до ПЗ
6.		Додавання нової точка до шляху	Включено до ПЗ
7.		Додавання нового замовлення	Включено до ПЗ
8.		Додавання нового замовлення – неоптимальний варіант	Включено до ПЗ
9.		Додавання нового замовлення – оптимальний варіант	Включено до ПЗ
10.		Побудова маршуту на графі	Включено до ПЗ
11.		Схема розробленого алгоритму	Включено до ПЗ
12.		Схема алгоритму оцінки зміни маршруту	Включено до ПЗ
13.		Інтерфейс роботи з графом	Включено до ПЗ
14.		Інтерфейс роботи з мапою	Включено до ПЗ

	Лист	*№ докум*	*Підп.*		ГЮИК.507900.027 ДЗ			
Зм.	*Пареч н В.П.*			*Дата*				
Розроб.		*Проф. Калита Н.І.*			*Дослідження та аналіз ефективності методів розв'язування задач на графах у системному проектуванні*		Лист	Листів
Перевірив.		*Проф. Калита Н.І*					*1*	*1*
Н. Контр.		*Гребеннік І.В.*				*ХНУРЕ Кафедра СТ*		