Tugas Kecil 3 IF2211 Strategi Algoritma
Implementasi Algoritma UCS dan A* untuk Menentukan Lintasan Terpendek

Ariel Jovananda
13521086 / K-02

Program Studi Sarjana Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

2023

DAFTAR ISI

BAB 1
DESKRIPSI PERSOALAN

Tugas kecil 3 ini ingin Anda membuat grafik segmen jalan di peta untuk menemukan jalur terpendek berdasarkan peta Google Map jalan-jalan di kota Bandung. Untuk menemukan jalur terpendek menggunakan algoritma UCS dan A*, komputer akan membuat grafik untuk merepresentasikan peta dan mengambil input dari node asal dan tujuan. Peta interaktif yang dapat diklik untuk membuat grafik juga dapat digunakan sebagai input program. Aplikasi akan meminta Anda untuk node asal dan tujuan serta strategi pencarian yang ingin Anda terapkan setelah menampilkan grafik. Aplikasi kemudian akan menampilkan jalur terpendek grafik yang dibangun.

BAB 2
KODE PROGRAM

2.1 Fungsi *UCS*

```
function UCS(start, goal, graph, vertices, nodeCoor){
 let visited = {}
 let activePaths = {}
 let endPath = {}
 let pathKe = 0;
 let startNode = 0
 let currNode = start
 console.log("start UCS search...")
 if(start === goal){
   endPath[0] = {
       path: [start],
       dist: 0,
       road: []
   }
 }
 activePaths[startNode] = {
     edgeNode: start,
     dist: 0,
     path: [start],
     road: []
 }
 while (!(Object.keys(visited).length === vertices.length - 1)){
     let lowestDist = Infinity
     let lowestDistNode = null
     let currPathKe = null
     for(let node in activePaths ){
         if(!visited[activePaths[node].edgeNode]){
             let pathDist = activePaths[node].dist;
             if (pathDist < lowestDist) {
                 lowestDist = pathDist;
                 lowestDistNode = activePaths[node].edgeNode;
                 currPathKe = node;
             }
         }
     }
```

```
    currNode = lowestDistNode
    if(currNode === null){
      break
    }
    let currPath = activePaths[currPathKe].path
    let currDist = activePaths[currPathKe].dist
    let currRoad = activePaths[currPathKe].road
    for(let branchNode in graph[currNode]){
        const x1 = nodeCoor[currNode].x;
        const y1 = nodeCoor[currNode].y;
        const x2 = nodeCoor[branchNode].x;
        const y2 = nodeCoor[branchNode].y;
        const dx = x2 - x1;
        const dy = y2 - y1;
        const result =  Math.sqrt(dx*dx + dy*dy);
        let newDist = currDist + result
        let newPath = currPath.concat(branchNode)
        let newRoad = currRoad.concat(graph[currNode][branchNode])
        if(!visited[branchNode] === true && branchNode != goal && branchNode !=
currNode){
          pathKe++
          activePaths[pathKe] = {
              edgeNode: branchNode,
              dist: newDist,
              path: newPath,
              road: newRoad
          }
        }
        if(branchNode == goal){
          pathKe++
          endPath[pathKe] = {
              dist: newDist,
              path: newPath,
              road: newRoad
          }
        }
    }
    visited[currNode] = true
    delete activePaths[currPathKe]
    let i = 0
    for (let node in activePaths) {
```

```
        i++
        console.log(`Path ${i}: [${activePaths[node].path}], cost:
${activePaths[node].dist}`);
    }
    lowestCost = Infinity
    lowestCostNode = null
    currPathKe = 0;
 }
 let [minPath, minDist] = Object.entries(endPath).reduce((acc, [path, {dist}]) => {
    if (dist < acc[1]) {
        return [path, dist];
    }
    return acc;
 }, ['', Infinity]);
 return endPath[minPath];

}
```

2.2 Fungsi *AStar*

```
function AStar(start, goal, graph, vertices, nodeCoor){
   let visited = {}
   let activePaths = {}
   let endPath = {}
   let pathKe = 0;
   let startNode = 0
   let currNode = start
   console.log("start A* search...")
   if(start == goal){
     endPath[0] = {
        heuristik: 0,
        aStar: 0,
        path: [start],
        dist: 0,
        road: []
     }

   }
   activePaths[startNode] = {
```

```
      edgeNode: start,
      aStar: 0, //A star score
      dist: 0,
      heuristik: 0,
      path: [start],
      road: []
  }
  while (!(Object.keys(visited).length === vertices.length - 1)){
    let LowestAStar = Infinity
    let LowestAStarNode = null
    let currPathKe = null
    for(let node in activePaths ){
        if(!visited[activePaths[node].edgeNode]){
            let pathAStar = activePaths[node].aStar;
            if (pathAStar < LowestAStar) {
                LowestAStar = pathAStar;
                LowestAStarNode = activePaths[node].edgeNode;
                currPathKe = node;
            }
        }
    }
    currNode = LowestAStarNode
    if(currNode === null){
      break
    }
    let currPath = activePaths[currPathKe].path
    let currDist = activePaths[currPathKe].dist
    let currRoad = activePaths[currPathKe].road
    for(let branchNode in graph[currNode]){
        const x1 = nodeCoor[currNode].x;
        const y1 = nodeCoor[currNode].y;
        const x2 = nodeCoor[branchNode].x;
        const y2 = nodeCoor[branchNode].y;
        const x3 = nodeCoor[goal].x
        const y3 = nodeCoor[goal].y
        const dx = x2 - x1;
        const dy = y2 - y1;
        const dx1 = x3 - x2
        const dy1 = y3 - y2
        const distCurrBr =  Math.sqrt(dx*dx + dy*dy);
        const distBrGoal = Math.sqrt(dx1*dx1 + dy1*dy1)
```

```
        //let edgeWeight = graph[currNode][branchNode]
        let newHeu = distBrGoal
        let newDist = currDist + distCurrBr
        let newAStar = currDist + distCurrBr + newHeu
        let newPath = currPath.concat(branchNode)
        let newRoad = currRoad.concat(graph[currNode][branchNode])
        if(!visited[branchNode] === true && branchNode != goal && branchNode !=
currNode){
          pathKe++
          activePaths[pathKe] = {
              edgeNode: branchNode,
              heuristik: newHeu,
              dist: newDist,
              aStar: newAStar,
              path: newPath,
              road: newRoad
          }
        }
        if(branchNode == goal){
          pathKe++
          endPath[pathKe] = {
              heuristik: newHeu,
              aStar: newAStar,
              path: newPath,
              dist: newDist,
              road: newRoad
          }
        }
    }

    visited[currNode] = true
    delete activePaths[currPathKe]
    lowestCost = Infinity
    lowestCostNode = null
    currPathKe = 0;
  }
  console.log("end paths = ", endPath)
  let [minPath, minDist] = Object.entries(endPath).reduce((acc, [path, {dist}]) =>
{
      if (dist < acc[1]) {
          return [path, dist];
```

```
        }
        return acc;
    }, ['', Infinity]);
    console.log(`Minimal dist path: ${minPath}, dist: ${minDist}`);
    console.log(endPath)
    return endPath[minPath];
}
```

BAB 3
INPUT DAN OUTPUT

Penafian: Graf-graf yang divisualisasikan merupakan peta/graf khayalan, dan tidak menunjukkan peta/graf yang sesungguhnya. Hasil dari *UCS Search* dan *A* Search* selalu sama.

3.1 Peta jalan sekitar kampus ITB/Dago/Bandung Utara

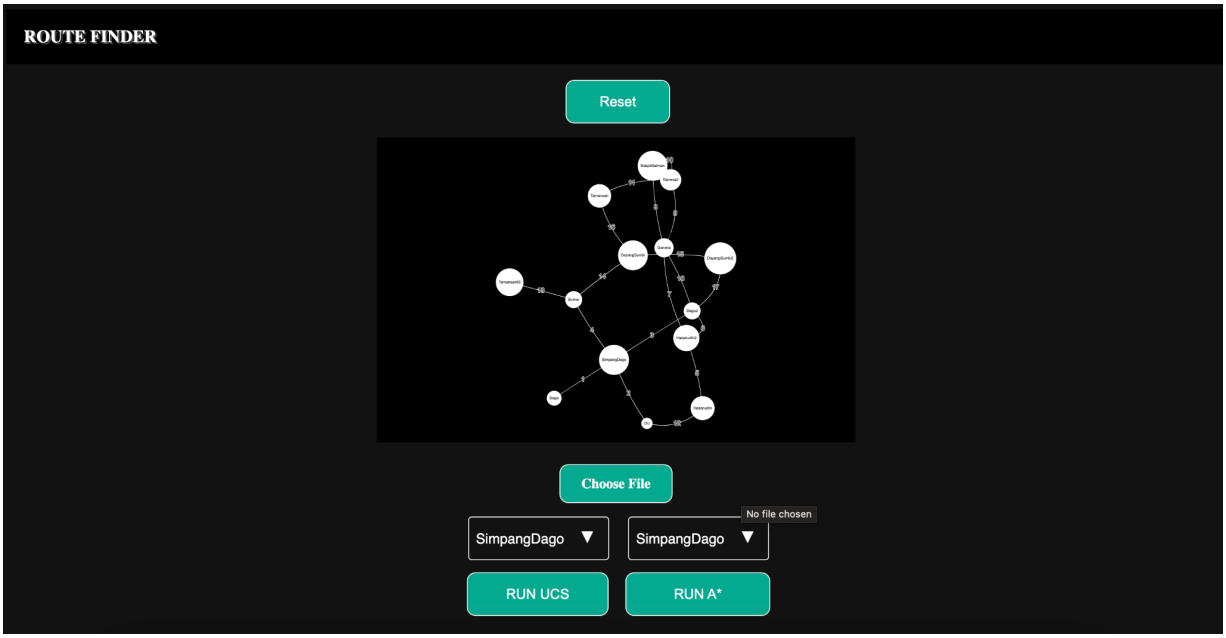
Input

```
14
SimpangDago -6.885223750760043 107.61368477392182
Dago -6.879069113409904 107.61650088612312
DU -6.8878107716607655, 107.61537986569274
Hasanudin -6.894683605168017 107.61738731025578
Hasanudin2 -6.89478162727485 107.61309777419027
Ganesa -6.893681600246217 107.61270282969319
MasjidSalman -6.893540012428024 107.61133149446633
Ganesa2 -6.893736057094537 107.60843523482107
Sumur -6.886371095950279 107.61183475051735
Tamansari65 -6.885508574323564 107.6115865252781
DayangSumbi -6.886973491077001 107.61154515440488
DayangSumbi2 -6.88738421458844 107.61339305340825
Dago2 -6.887397905366039 107.61364127864749
Tamansari -6.887548503893449 107.61093838159782

    SimpangDago Dago DU Hasanudin Hasanudin2 Ganesa MasjidSalman Ganesa2 Sumur
Tamansari65 DayangSumbi DayangSumbi2 Dago2 Tamansari
SimpangDago 0 1 2 0 0 0 0 0 4 0 0 0 3 0
Dago 1 0 0 0 0 0 0 0 0 0 0 0 0 0
DU 2 0 0 12 0 0 0 0 0 0 0 0 0 0
Hasanudin 0 0 12 0 5 0 0 0 0 0 0 0 0 0
Hasanudin2 0 0 0 5 0 7 0 0 0 0 0 0 6 0
Ganesa 0 0 0 0 7 0 8 9 0 0 0 0 18 0
MasjidSalman 0 0 0 0 0 8 0 10 0 0 0 0 0 0
Ganesa2 0 0 0 0 0 9 10 0 0 0 0 0 0 11
Sumur 4 0 0 0 0 0 0 0 0 13 14 0 0 0
Tamansari65 0 0 0 0 0 0 0 0 13 0 0 0 0 0
DayangSumbi 0 0 0 0 0 0 0 0 14 0 0 15 0 16
DayangSumbi2 0 0 0 0 0 0 0 0 0 0 0 0 15 17 0
Dago2 3 0 0 0 6 18 0 0 0 0 0 17 0 0
Tamansari 0 0 0 0 0 0 0 11 0 0 16 0 0 0
```

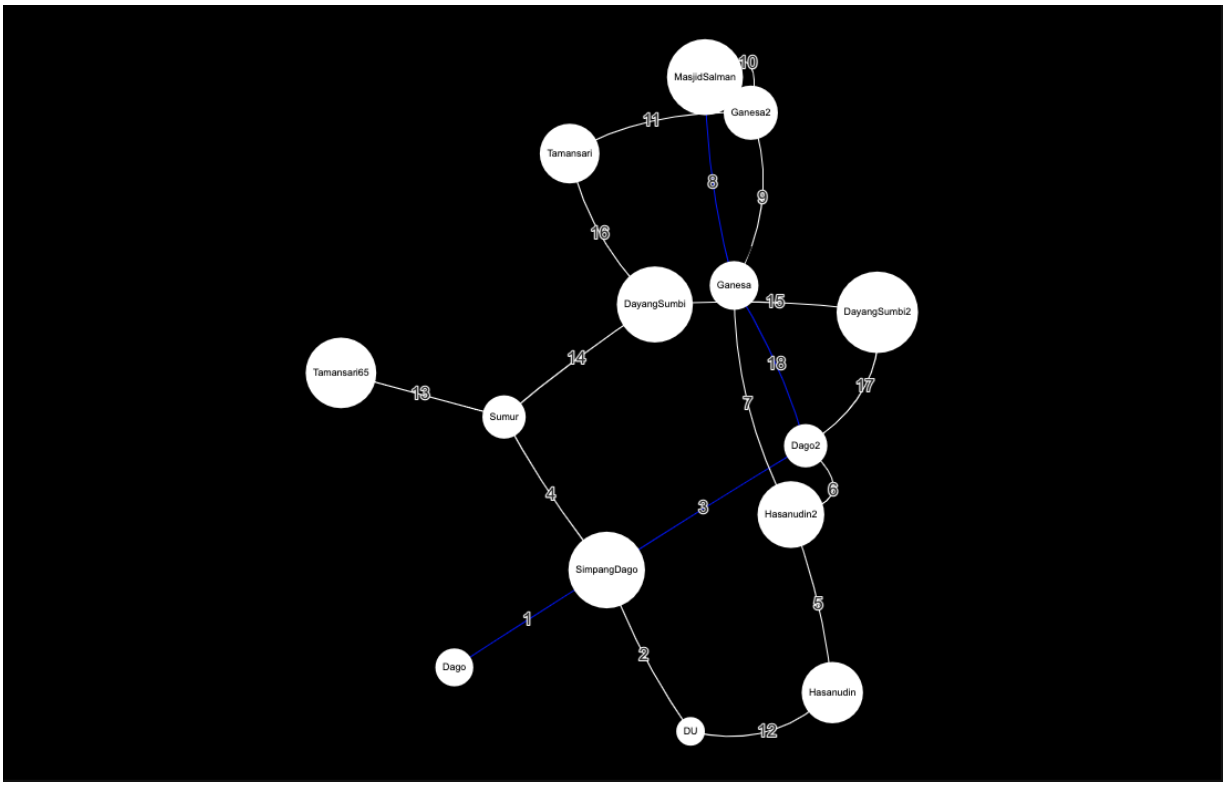
Output

State Awal



Gambar 3.1.1 State Awal

State setelah dilakukan *search*



Gambar 3.1.2 State setelah dilakukan *search* dengan awal di “Dago” dan akhir di “MasjidSalman

3.2 Peta jalan sekitar Buahbatu atau Bandung Selatan

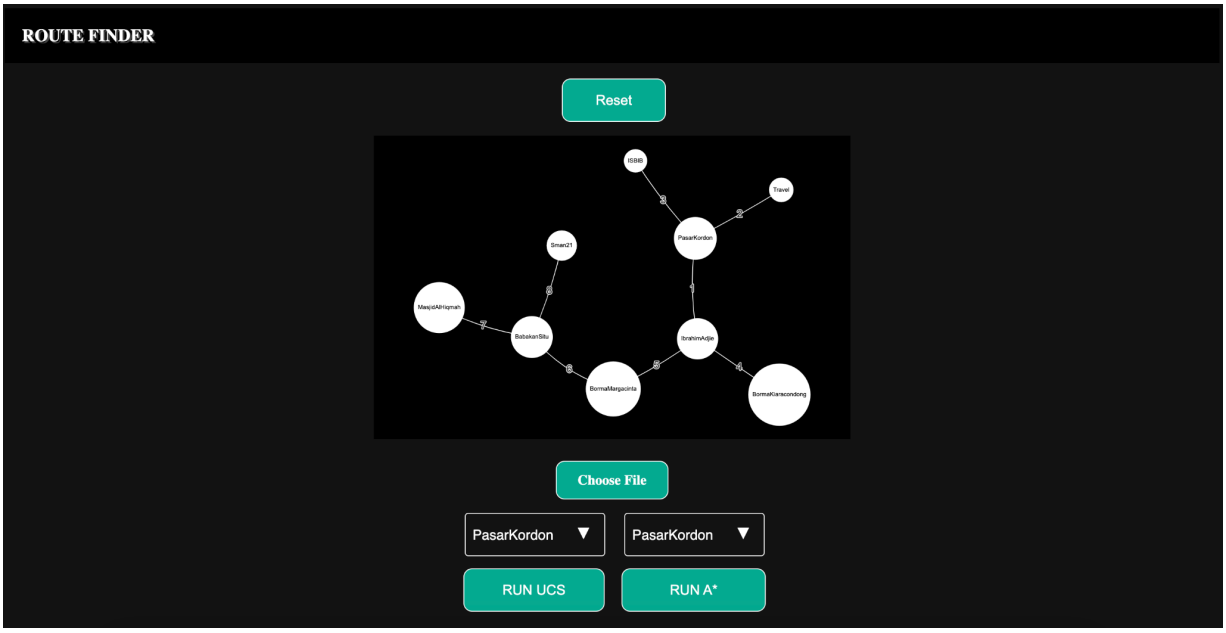
Input

```
9
PasarKordon -6.954389695054669 107.63905120601575
IbrahimAdjie -6.954051679462522 107.64025419496352
BormaKiaracondong -6.942861231636595 107.64208591465906
BormaMargacinta -6.9554479468862285 107.65139221105022
BabakanSitu -6.962041337131752 107.67188314155648
MasjidAlHiqmah -6.965008203863548 107.67120971335297
Sman21 -6.957160617474226 107.67233516872174
Travel -6.960639224482717 107.63900580576589
ISBIB -6.9432362333892526 107.62872377709571

  PasarKordon IbrahimAdjie BormaKiaracondong BormaMargacinta BabakanSitu
MasjidAlHiqmah Sman21 Travel ISBIB
PasarKordon 0 1 0 0 0 0 0 2 3
IbrahimAdjie 1 0 4 5 0 0 0 0 0
BormaKiaracondong 0 4 0 0 0 0 0 0 0
BormaMargacinta 0 5 0 0 6 0 0 0 0
BabakanSitu 0 0 0 6 0 7 8 0 0
MasjidAlHiqmah 0 0 0 0 7 0 0 0 0
Sman21 0 0 0 0 8 0 0 0 0
Travel 2 0 0 0 0 0 0 0 0
ISBIB 3 0 0 0 0 0 0 0 0
```

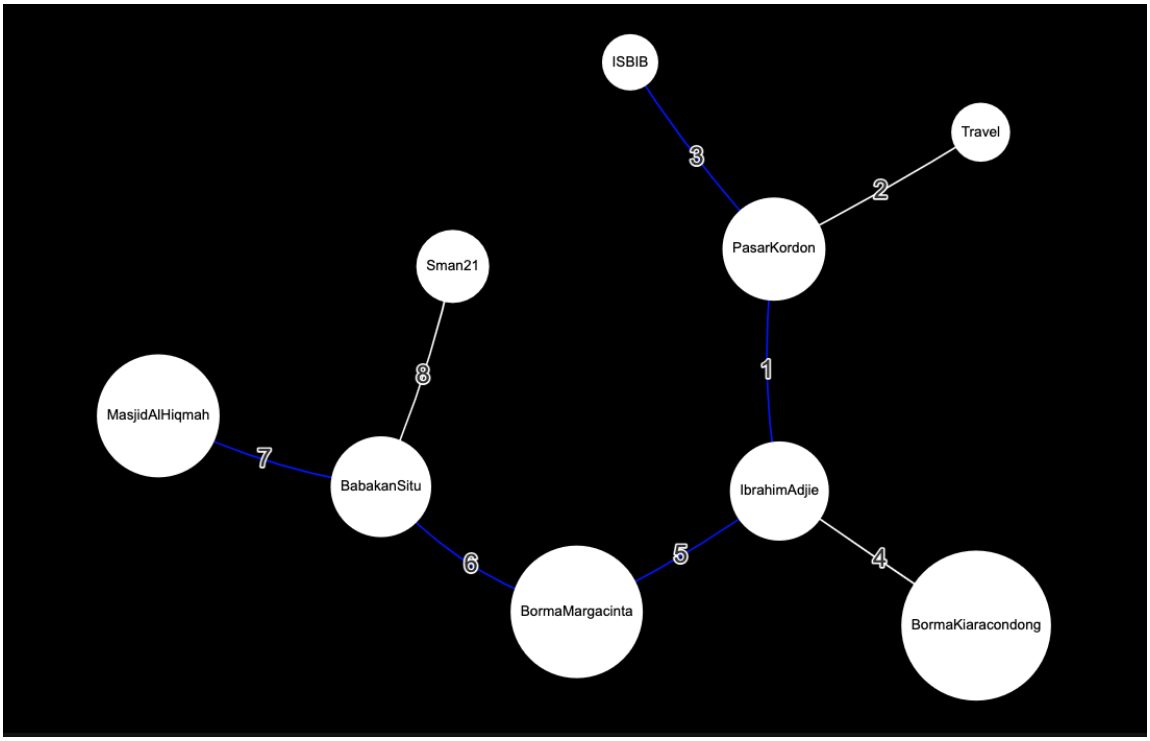
Output

State Awal



Gambar 3.2.1 State Awal

State setelah dilakukan *search*



Gambar 3.2.2 State setelah dilakukan *search* dengan awal di “ISBIB” dan akhir di “MasjidAlHiqmah”

3.3 Peta jalan sekitar Alun-alun Bandung

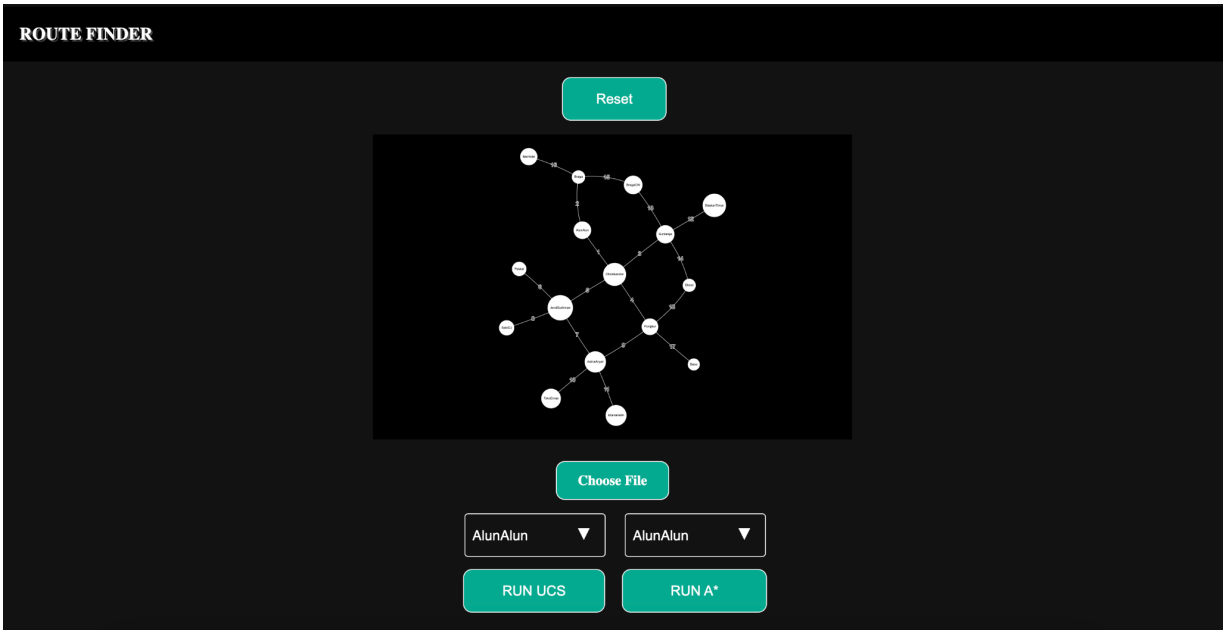
Input

```
16
AlunAlun -6.921157798480552 107.6070524318345
OttoIskandar -6.920806703565497 107.60409107161638
JendSudirman -6.920106386881566 107.59835704818326
AstnaAnyar -6.92692302818919 107.59987005282414
Pungkur -6.927200313207882 107.6036874799257
Suniaraja -6.915824370943439 107.60452450158039
StasiunTimur -6.91496668082747 107.60460549928135
BragaCW -6.916896481398406 107.60922236823588
Braga -6.921458998298462 107.60980426355675
IbisHotel -6.921859941232551 107.61285975272209
Baso -6.928104608880081 107.60903121502483
Dbest -6.929766906980908 107.60347951108317
Paskal -6.916501174100647 107.59824450658233
SateDJ -6.919578320606566 107.59383101063507
TokoEmas -6.9267529730421655 107.59847328284779
IstanaHelm -6.928244044543902 107.60034010020078

  AlunAlun OttoIskandar JendSudirman AstnaAnyar Pungkur Suniaraja StasiunTimur
BragaCW Braga IbisHotel Baso Dbest Paskal SateDJ TokoEmas IstanaHelm
AlunAlun 0 1 0 0 0 0 0 0 2 0 0 0 0 0 0 0
OttoIskandar 1 0 5 0 4 3 0 0 0 0 0 0 0 0 0 0
JendSudirman 0 5 0 7 0 0 0 0 0 0 0 0 6 8 0 0
AstnaAnyar 0 0 7 0 9 0 0 0 0 0 0 0 0 0 10 11
Pungkur 0 4 0 9 0 0 0 0 0 0 17 18 0 0 0 0
Suniaraja 0 3 0 0 0 0 12 13 0 0 0 14 0 0 0 0
StasiunTimur 0 0 0 0 0 12 0 0 0 0 0 0 0 0 0 0
BragaCW 0 0 0 0 0 13 0 0 15 0 0 0 0 0 0 0
Braga 2 0 0 0 0 0 0 15 0 16 0 0 0 0 0 0
IbisHotel 0 0 0 0 0 0 0 0 16 0 0 0 0 0 0 0
Baso 0 0 0 0 17 0 0 0 0 0 0 0 0 0 0 0
Dbest 0 0 0 0 18 0 0 0 0 0 0 0 0 0 0 0
Paskal 0 0 6 0 0 14 0 0 0 0 0 0 0 0 0 0
SateDJ 0 0 8 0 0 0 0 0 0 0 0 0 0 0 0 0
TokoEmas 0 0 0 10 0 0 0 0 0 0 0 0 0 0 0 0
IstanaHelm 0 0 0 11 0 0 0 0 0 0 0 0 0 0 0 0
```

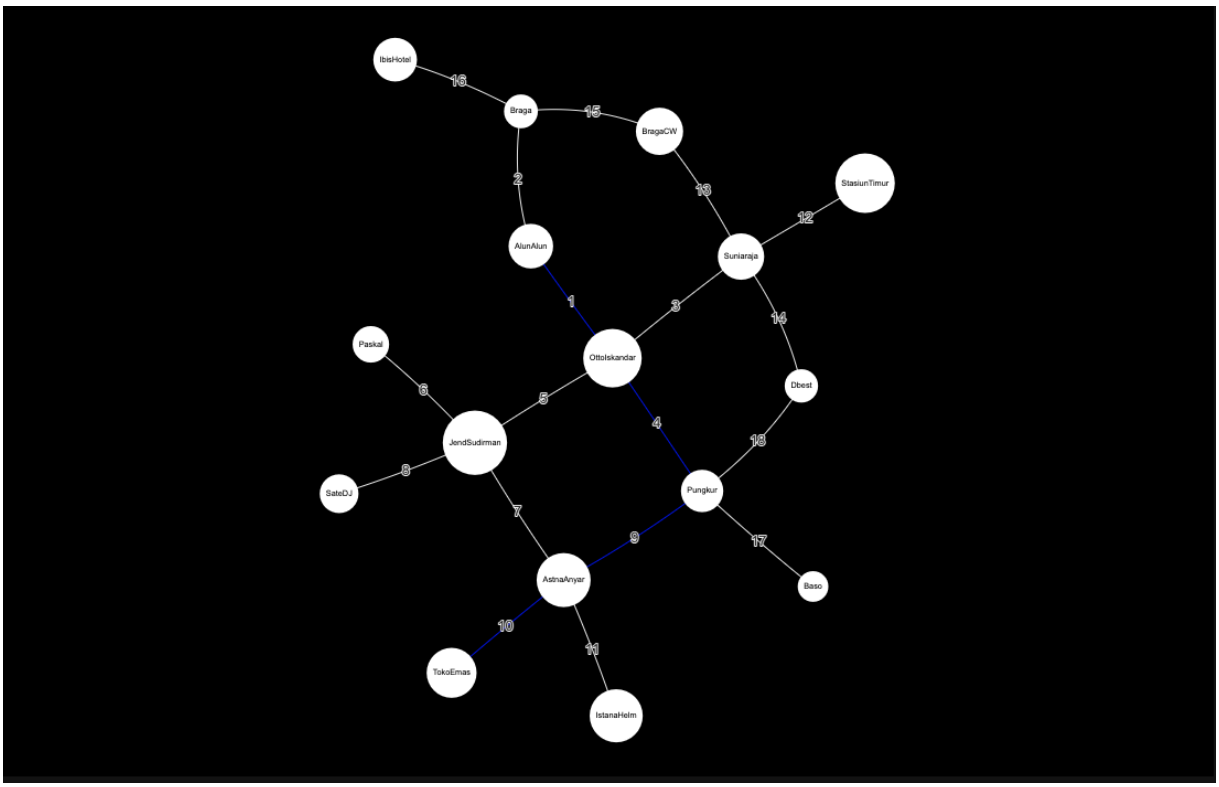
Output

State awal

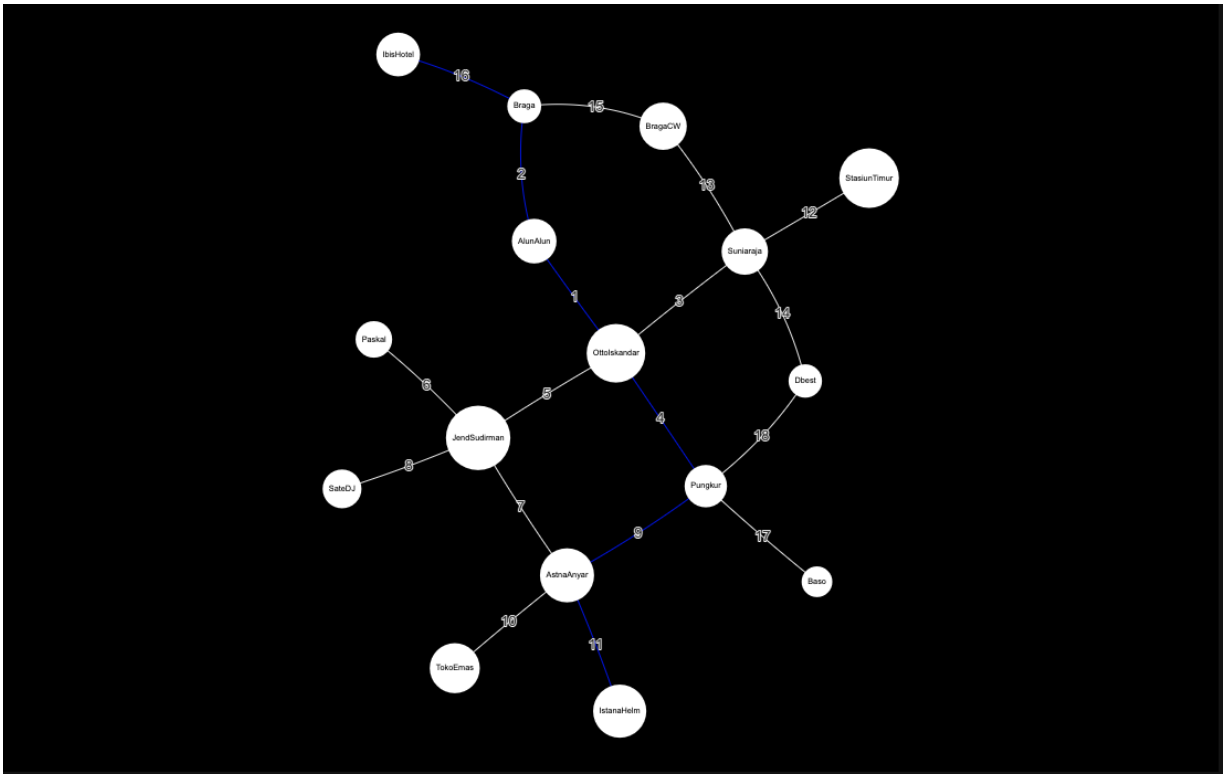


Gambar 3.3.1 State Awal

State setelah dilakukan *search*



Gambar 3.3.2 State setelah dilakukan *search* dengan awal di “TokoEmas” dan akhir di “AlunAlun”



Gambar 3.3.3 State setelah dilakukan *search* dengan awal di “IstanaHelm” dan akhir di “IbisHotel”

3.4 Peta jalan sebuah kawasan di kota asalmu

Input

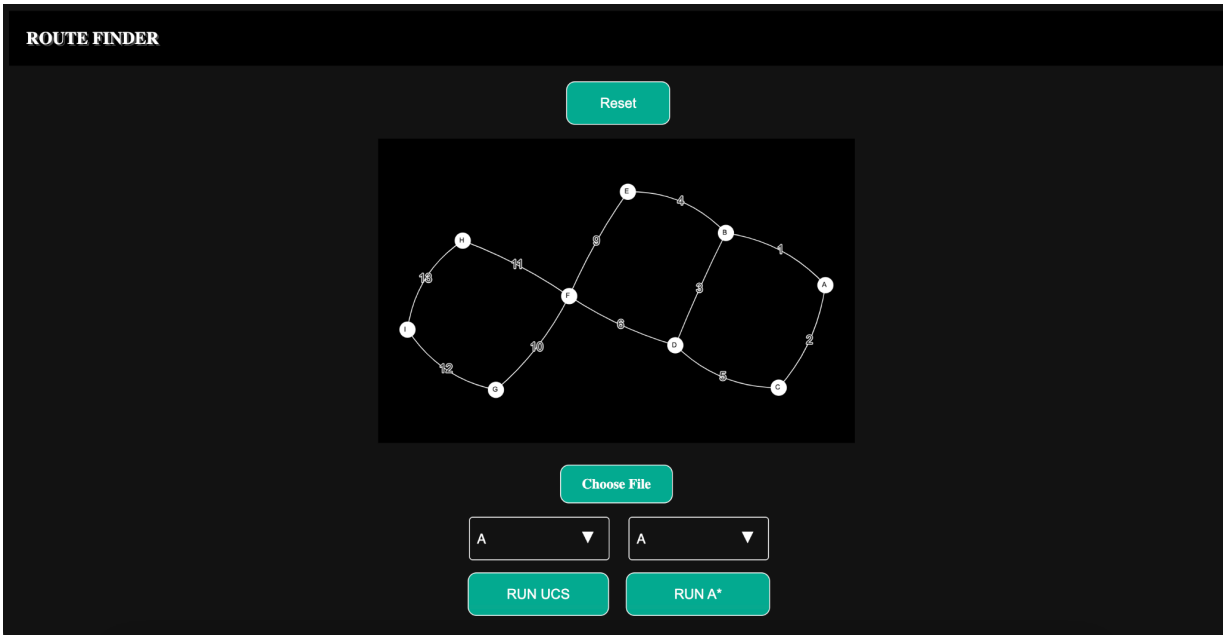
```
9
A -6.88247 107.60721
B -6.88234 107.60790
C -6.88344 107.60763
D -6.88307 107.60814
E -6.88229 107.60847
F -6.88274 107.60861
G -6.88302 107.60891
H -6.88237 107.60953
I -6.88306 107.60963

  A B C D E F G H I
A 0 1 2 0 0 0 0 0 0
B 1 0 0 3 4 0 0 0 0
C 2 0 0 5 0 0 0 0 0
D 0 3 5 0 0 6 0 0 0
E 0 4 0 0 0 9 0 0 0
F 0 0 0 6 9 0 10 11 0
G 0 0 0 0 0 0 11 0 0 12
```

```
H 0 0 0 0 0 11 0 0 13
I 0 0 0 0 0 0 12 13 0
```

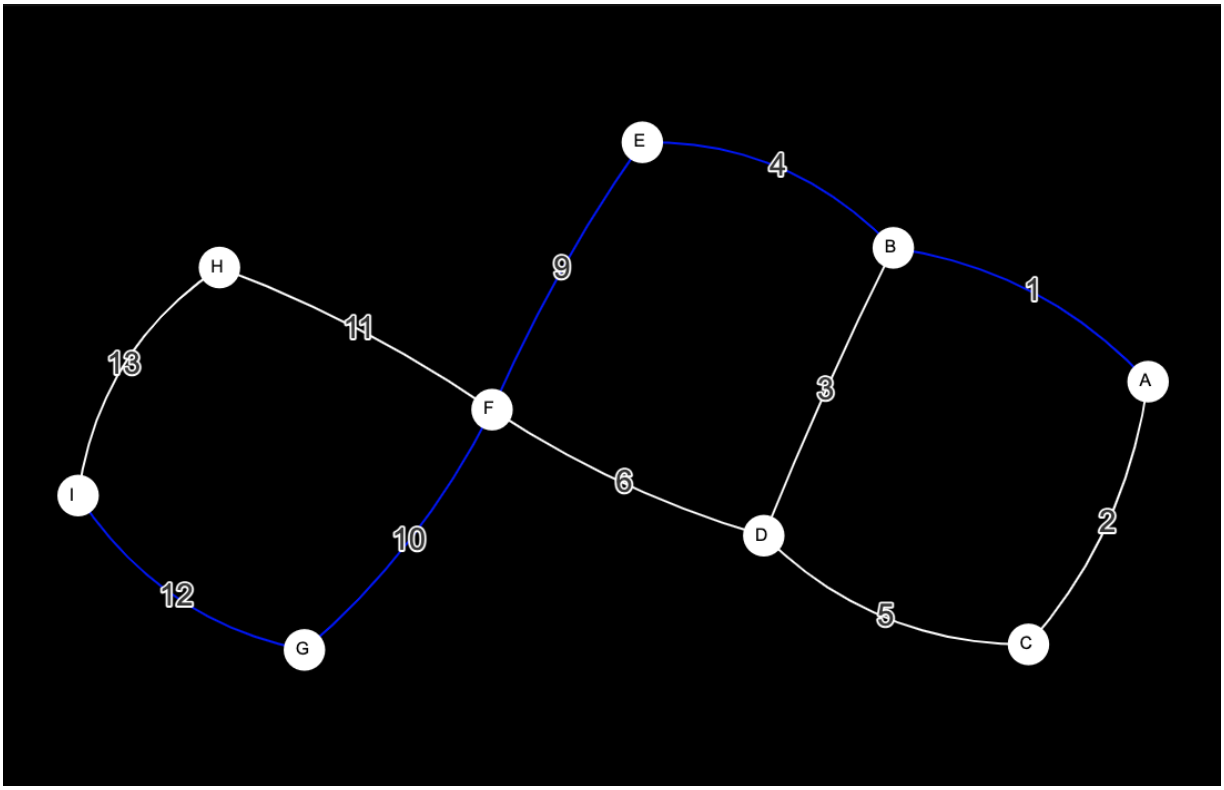
Output

State Awal



Gambar 3.4.1 State Awal

State setelah dilakukan *search*



Gambar 3.3.3 State setelah dilakukan *search* dengan awal di “A” dan akhir di “I”

BAB 4
KESIMPULAN DAN KOMENTAR

4.1 Kesimpulan

Demikian program yang saya buat untuk memenuhi tugas kecil ketiga dari mata kuliah Strategi Algoritma. Program masih jauh dari sempurna dan masih ada kekurangan-kekurangannya, seperti kurang modular, kurang efisien, input yang harus sangat sesuai, dan program masih belum bisa mengimplementasikan bonus. Maka dari itu saya berharap, saya bisa mendapatkan komentar atau saran agar program ini bisa menjadi program yang lebih baik dan benar.

4.2 Komentar

Dengan diberinya tugas kecil ini, saya mempelajari banyak hal mengenai algoritma *UCS* dan *A**, selain itu saya juga mempelajari lebih mengenai *web development*, saya sangat bersyukur dengan diberinya tugas ini, saya dapat mempelajari hal-hal tersebut dan mengembangkan diri sendiri. Tugas kecil ini merupakan suatu tantangan yang besar bagi saya, namun pada akhirnya berkat program bisa selesai dan berjalan dengan sebagaimana harusnya.

LAMPIRAN

Link Repository: https://github.com/arieljovananda88/Tucil3_13521086