4. IMPLEMENTASI PROGRAM

4.1 Implementasi Sistem

Dalam penelitian ini ada beberapa fungsi yang digunakan dalam menjalankan algoritma ini. Fungsi-fungsi yang digunakan akan dijelaskan untuk mengetahui langkah-langkah dalam algoritma yang digunakan.

4.1.1 Memasukan Data Dosen dan Jadwal mengajar

Dalam proses ini, data dosen dan jadwal mengajarnya akan dimasukan dalam algoritma untuk diubah menjadi jadwal berhalangan dosen. Proses tersebut dilakukan dengan menggunakan program berikut.

```
<script>
import moment from 'moment'
import * as xlsx from 'xlsx'
import * as ExcelJS from 'exceljs'
import 'lodash.combinations'
import _ from 'lodash';
import router from '../router'
export default {
 data() {
  return {
   file_num: 1,
   jsp: [],
   sidang: '',
   mhs: '',
   start: '',
   finish: '',
   dosen: null,
   ruang: 0,
   addDosen: [],
   addDosennum: 1
  }
 },
 mounted() {
  this.ruang = 1
 },
 methods: {
  fileDosen(e) {
   this.dosen = e.target.files[0]
  },
  fileMhs(e) {
   this.mhs = e.target.files[0]
```

```
  },
  tbhDosen() {
   this.addDosennum++
  },
  delDosen() {
   if (this.addDosennum > 1) {
    this.addDosennum--
   }
  },
  addFile() {
   this.file_num++
  },
  deleteFile() {
   if (this.file_num > 1) {
    this.file_num--
   }
  },
  getFile() {
   let files = Array.from(document.getElementsByName('jsp')).map(v => v.files[0])
   this.jsp = files
  },
  fileSidang(e) {
   this.sidang = e.target.files[0]
  },
  splitName(string, array) {
   var list = [];
   let string2 = string;
   var newArray = _.cloneDeep(array);
   while (newArray.some((v) => string2.indexOf(v) != -1)) {
    let arrIndex = newArray.findIndex((v) => string2.indexOf(v) != -1);
    list.push(newArray[arrIndex]);
    string2 = string2.split(newArray[arrIndex])[1] == "" ?
string2.split(newArray[arrIndex])[0] : string2.split(newArray[arrIndex])[1]
   }
   return { list, remain: string2 };
  },
  async submit() {
   let datadosen = []
   let promises = []
   let arrlist = []
   let week = ["Minggu", "Senin", "Selasa", "Rabu", "Kamis", "Jumat", "Sabtu"]
   for (var datearr = [], dt = new Date(this.start); dt <= new Date(this.finish);
dt.setDate(dt.getDate() + 1)) {
    let date = new Date(dt)
    if (![0, 6].includes(date.getDay())) {
     datearr.push(date);
    }
```

```
  }
  for (const item of this.jsp) {
   if (item !== null) {
    let filePromises = new Promise(resolve => {
     var reader = new FileReader()
     reader.onload = async (e) => {
      const bstr = e.target.result;
      const wb = xlsx.read(bstr, { type: 'binary' });
      /* Get first worksheet */
      let buffer = xlsx.write(wb, { type: 'buffer', bookType: 'xlsx' })
      resolve(buffer)
     }
     reader.readAsBinaryString(item)
    })
    promises.push(filePromises)
   }

  }
  await Promise.all(promises).then(async (fileBuffer) => {
   for (const buffer of fileBuffer) {
    let workbook = new ExcelJS.Workbook()
    let data = await workbook.xlsx.load(buffer)
     .then(async function () {
      let data = []
      let worksheet = workbook.getWorksheet(1)
      worksheet.spliceRows(0, 1)
      worksheet.spliceRows(0, 1)
      worksheet.eachRow(function (row) {
       row.getCell(8).value = row.getCell(8).text.replace('\n', ',')
      })
      worksheet.spliceColumns(4, 4)
      worksheet.spliceColumns(5, 5)
      worksheet.spliceRows(worksheet.actualRowCount - 1, 1)
      worksheet.spliceRows(worksheet.actualRowCount, 1)
      worksheet.eachRow(function (row) {
       let hari = row.getCell(1).text
       let mulai = row.getCell(2).text
       let selesai = row.getCell(3).text
       let dosen = row.getCell(4).text.replace('\n', ',')
       data.push({ hari, mulai, selesai, dosen })
      })
      return data
     })
    datadosen.push(data)
    datadosen = datadosen.flat()
   }
  })
```

```
let list_dosen = []
for (const item of datadosen) {
  let dsn = item.dosen
  dsn = dsn.replace('\n', ',')
  while (dsn.match(',\\w[^.]') != null) {
    let i = dsn.match(',\\w[^.]')
    let index = i.index
    let res = dsn.slice(0, index)
    list_dosen.push(res)
    dsn = dsn.slice(index + 1)
  }
  list_dosen.push(dsn)
}
list_dosen = _.uniqWith(list_dosen, _.isEqual)
list_dosen = _.filter(list_dosen, e => e !== "")
console.log(this.addDosen);
list_dosen.push(...this.addDosen)
console.log(list_dosen);
//Dosen
await fetch('http://localhost:3000/api/insertDosen',
  {
    headers: {
      'Accept': 'application/json',
      'Content-Type': 'application/json'
    },
    body: JSON.stringify(list_dosen),
    method: "POST"
  })
let dosen = await fetch('http://localhost:3000/api/getDosen')
  .then(response => { return response.json() })
  .then(data => { return data })// Karena Ambil dr DB
let namaDosen = dosen.map(v => v.nama_dosen)
for (const item of datadosen) {
  let res = this.splitName(item.dosen, namaDosen)
  item.dosen = res.list
}
//Jadwal Sidang per hari
var reader = new FileReader()
let json_jadwal = null
await new Promise(resolve => {
  reader.onload = (e) => {
    const bstr = e.target.result;
    let wb = xlsx.read(bstr, { type: 'binary' })
    let ws = wb.Sheets[wb.SheetNames[0]]
    var json_jadwal = Array.from(xlsx.utils.sheet_to_json(ws, {
      header: 1, blankrows: false
    }));
```

```
    json_jadwal.shift()
    json_jadwal = json_jadwal.map(v => {
      return { hari: v[0], mulai: v[1], selesai: v[2] }
    })
    json_jadwal = Array.from(json_jadwal)
    resolve(json_jadwal)
  }
  reader.readAsBinaryString(this.sidang)
}).then((res) => {
  json_jadwal = res
})
for (const item of namaDosen) {
  let jadwalhari = []
  for (const jadwal of json_jadwal) {
    jadwalhari.push({
      hari: jadwal.hari,
      mulai: jadwal.mulai,
      selesai: jadwal.selesai,
      penuh: false
    })
  }
  let obj = {
    nama: item,
    jadwal: jadwalhari
  }
  arrlist.push(obj)
}
for (const ajar in datadosen) {
  //each jadwal ajar
  if (Object.hasOwnProperty.call(datadosen, ajar)) {
    const listAjar = datadosen[ajar];
    for (const dosen of listAjar["dosen"]) {
      //each dosen di jadwal dosen
      var dosenAjar = Array.from(
        arrlist.find((x) => x.nama == dosen).jadwal
      ); //cari jadwal yg sesuai
      var indexDosenAjar = arrlist.findIndex((x) => x.nama == dosen);
      dosenAjar.forEach((element) => {
        let e = {
          hari: element.hari,
          mulai: element.mulai,
          selesai: element.selesai,
          penuh: element.penuh,
        };
        if (e.hari === listAjar.hari) {
          let start = moment(e.mulai, "HH:mm");
          let finish = moment(e.selesai, "HH:mm");
```

```
          let startLesson = moment(listAjar["mulai"], "HH:mm");
          let finLesson = moment(listAjar["selesai"], "HH:mm");
          if (startLesson.isBetween(start, finish, undefined, "[)")) {
            e.penuh = true;
          } else if (finLesson.isBetween(start, finish, undefined, "(]")) {
            e.penuh = true;
          } else if (
            start.isBetween(startLesson, finLesson, undefined, "()")
          ) {
            e.penuh = true;
          } else if (
            finish.isBetween(startLesson, finLesson, undefined, "()")
          ) {
            e.penuh = true;
          }
          let indexe = arrlist[indexDosenAjar].jadwal.findIndex(
            (v) =>
              v.hari === e.hari &&
              v.mulai === e.mulai &&
              v.selesai === e.selesai
          );
          arrlist[indexDosenAjar].jadwal[indexe] = e;
        }
      });
    }
  }
}
// console.log(arrlist);
for (const dosen of arrlist) {
  let newjadwal = []
  for (const jadwal of datearr) {
    let day = week[jadwal.getDay()]
    let jadwalhari = _.filter(dosen.jadwal, function (e) { return e.hari == day })
    for (const j of jadwalhari) {
      newjadwal.push({
        nama_dosen: dosen.nama,
        tanggal: moment(jadwal).format("DD/MM/yyyy"),
        jam_mulai: j.mulai,
        jam_selesai: j.selesai,
        penuh: j.penuh
      })
    }
  }
  dosen.jadwal = newjadwal
}

//Jadwal
```

```
    fetch('http://localhost:3000/api/insertJadwal',
     {
       headers: {
         'Accept': 'application/json',
         'Content-Type': 'application/json'
       },
       body: JSON.stringify(arrlist),
       method: "POST"
     })
    router.push('setjadwal')
  },
 }
}
</script>
```

Program 4.1 Program Frontend Input Dosen dan Jadwal Mengajarnya

```
app.post('/api/insertJadwal', async (req, res) => {
 let jadwal = req.body
 await client.$queryRaw`Truncate jadwal_halang restart identity cascade`
 const dosen = await client.dosen.findMany()
 for (const d of jadwal) {
  for (const jadwal of d.jadwal) {
   await client.jadwal_halang.create({
    data: {
     dosen: dosen.find((e) => e.nama_dosen == jadwal.nama_dosen).id,
     tanggal: jadwal.tanggal,
     jam_mulai: jadwal.jam_mulai,
     jam_selesai: jadwal.jam_selesai,
     penuh: jadwal.penuh
    }
   })
  }
 }
 res.send({ data: 'Hello, World!' })
})
```

Program 4.2 Program Input Jadwal Berhalangan Dosen ke Database

```
app.post('/api/insertDosen', async (req, res) => {
 let mhs = req.body
 await client.$queryRaw`Truncate dosen restart identity cascade`
 for (const m of mhs) {
  await client.dosen.create({
   data: {
    nama_dosen: m
   }
  })
```

```
 }
 res.send({ data: 'Hello, World!' })
})
```

Program 4.3 Program Input Dosen ke Database

Dalam kode diatas, input dapat menerima jadwal yang berjumlah lebih dari satu dimana setiap jadwal akan dimasukan menjadi jadwal berhalangan sidang proposal.

4.1.2 Input Data Mahasiswa

Dalam proses ini, data mahasiswa dimasukan dengan menggunakan file excel data seprti nrp, nama mahasiswa, judul skripsi, dosen pembimbing dosen pembimbing, dan dosen penguji. Proses tersebut dilakukan dengan menggunakan kode dibawah ini

```
//Mhs
    reader = new FileReader()
    let mhs = null
    await new Promise(resolve => {
     reader.onload = (e) => {
       const bstr = e.target.result;
       let wb = xlsx.read(bstr, { type: 'binary' })
       let ws = wb.Sheets[wb.SheetNames[0]]
       var mhs = Array.from(xlsx.utils.sheet_to_json(ws, {
         header: 1, blankrows: false
       }));
       mhs.shift()
       json_jadwal = json_jadwal.map(v => {
         return { hari: v[0], mulai: v[1], selesai: v[2] }
       })
       mhs = Array.from(mhs)
       resolve(mhs)
     }
     reader.readAsBinaryString(this.mhs)
    }).then((res) => {
     mhs = res
    })
    fetch('http://localhost:3000/api/insertMhs',
     {
       headers: {
         'Accept': 'application/json',
         'Content-Type': 'application/json'
       },
       body: JSON.stringify(mhs),
       method: "POST"
 })
```

Program 4.4 Program Frontend Input Mahasiswa

```
app.post('/api/insertMhs', async (req, res) => {
 let mhs = req.body
 await client.$queryRaw`Truncate mahasiswa restart identity cascade`
 const dosen = await client.dosen.findMany()
 for (const m of mhs) {
  await client.mahasiswa.create({
   data: {
    nrp: m[0],
    nama_mhs: m[1],
    judul: m[2],
    dosbing1: dosen.find((e) => e.nama_dosen.toLowerCase() == m[4].toLowerCase()).id,
    dosbing2: m[5]
     ? dosen.find((e) => e.nama_dosen.toLowerCase() == m[5].toLowerCase())
      ? dosen.find((e) => e.nama_dosen.toLowerCase() == m[5].toLowerCase()).id
      : null
     : null,
    kpenguji: dosen.find((e) => e.nama_dosen.toLowerCase() == m[6].toLowerCase()).id,
    apenguji: dosen.find((e) => e.nama_dosen.toLowerCase() == m[7].toLowerCase()).id
   }
  })
 }
 res.send({ data: 'Hello, World!' })
})
```

Program 4.5 Program Input Data Mahasiswa ke Database

4.1.3 Pengaturan Jadwal Dosen

Setelah memasukan input jadwal dosen, dosen dapat mengganti jadwal berhalangan jika ada perubahan jadwal. Kode perubahan jadwal tersebut adalah sebagai berikut.

```
<script>
import _ from 'lodash';
export default {
  data() {
    return {
      curDosen: '',
      oldDosen: '',
      dosenList: [],
      arrlist: '',
      hari: null,
      checks: Array(),
      curJadwal: [],
      checkbox: [],
      i: 0
    }
  },
```

```
async created() {
   this.dosenList = await fetch('http://localhost:3000/api/getDosen')
   .then(response => { return response.json() }).then(data => { return data })
},
mounted() {
},
methods: {
   getVal() {
      let tempcheck = this.checkbox
      for (const i in this.checks) {
         for (const j in this.checks[i]) {
            if (this.checks[i][j] !== undefined) {
               this.checks[i][j].penuh = tempcheck[i][j]
            }
         }
      }
   },
   async submit() {
      this.updateTable()
      this.$refs.loading.hidden = false
      var res= await fetch('http://localhost:3000/api/GA')
      this.$refs.loading.hidden = true
      this.$router.push("hasil")
   },
   async updateTable() {
      if (this.checks != []) {
         this.getVal()
         let temp_checks = _.zip(...this.checks)
         temp_checks = _.flatten(temp_checks)
         temp_checks = _.compact(temp_checks)
         await fetch('http://localhost:3000/api/updateJadwalDosen',
            {
               headers: {
                  'Accept': 'application/json',
                  'Content-Type': 'application/json'
               },
               body: JSON.stringify(temp_checks),
               method: "POST"
            })
      }

   }
},
watch: {
   async curDosen(val) {
      this.updateTable()
      this.checks = null
```

```
        this.curJadwal = await fetch('http://localhost:3000/api/getJadwalDosen/' + val)
          .then(response => { return response.json() }).then(data => { return data })
        this.curJadwal = _.chain(this.curJadwal).groupBy('tanggal').map((val, tanggal) => ({
tanggal, val })).value()
        this.hari = this.curJadwal.map(e => e.tanggal)
        this.checks = this.curJadwal.map(e => e.val);
        this.checks = _.zip(...this.checks)
        this.checkbox = this.checks.map((e) => e.map((f) => f ? f.penuh : null))
      }
    }
}
</script>
<template>
  <div class="container-fluid d-xxl-flex justify-content-center">
    <h1 class="mb-3">Pengaturan Jadwal</h1>
    <form method="post" name="formAlg" @submit.prevent="submit">
      <div class="mb-3">
        <label for="dosen" class="form-label">Nama Dosen : </label>
        <select name="dosen" id="dosen" class="form-select mb-3" v-model="curDosen">
          <option disabled value="">
            Pilih Dosen
          </option>
          <option v-for="item in dosenList" :key="item" :value="item.id">
            {{ item.nama_dosen }}
          </option>
        </select>
        <p v-if="curDosen">Jadwal {{ curDosen }} :</p>
        <table class="table table-bordered">
          <thead>
            <tr>
              <th v-for="Ehari in hari" :key="Ehari">{{ Ehari }}</th>
            </tr>
          </thead>
          <tbody>
            <tr v-for="(Ejam, index) in checks" :key="index">
              <th v-for="n, index2 in Ejam" :key="index2">
                {{ n ? n.jam_mulai + '-' + n.jam_selesai : "" }}
                <div v-if="n !== undefined" class="d-inline-block float-right">
                  <input type="checkbox" v-model="checkbox[index][index2]"
:onChange="getVal">
                </div>
              </th>
            </tr>
          </tbody>
        </table>
      </div>
      <div class="mb-3" ref="submit">
```

```
            <button type="submit" class="btn btn-primary" ref="submit_text">Submit
            </button>
          </div>
        </form>
    </div>
    <div class="overlay justify-content-center" ref="loading" hidden>
      <div class="d-flex align-items-center ">
        <div class="spinner-border">
        </div>
        <p class="text-center align-middle m-0 ps-3">Loading ...</p>
      </div>
    </div>
</template>
```

Program 4.6 Program Frontend Pengaturan Jadwal Dosen

```
app.get('/api/getJadwalDosen/:id', async (req, res) => {
  const result = await client.jadwal_halang.findMany({
    where: {
      dosen: {
        equals: Number(req.params.id)
      }
    },
    orderBy: [
      {
        id: 'asc'
      },
      {
        jam_mulai: 'asc'
      }
    ]
  })
  res.send(result)
})
app.post('/api/updateJadwalDosen', async (req, res) => {
  let json = req.body
  for (const j of json) {
    await client.jadwal_halang.update({
      where: {
        id: j.id
      },
      data: {
        penuh: j.penuh
      }
    })
  }
  res.send('')
```

```
})
```

Program 4.7 Program Backend Jadwal Dosen

4.1.4 Penambahan Mahasiswa Berhalangan

Setelah mengatur jadwal dosen, pengguna dapat memasukan daftar mahasiswa yang berhalangan pada hari tertentu. Proses penambahan tersebut dapat dilihat dengan kode berikut.

```
<script>
import 'lodash.combinations'
export default {
  data() {
    return {
      msh: [],
      tgl_sdg: [],
      halang_num: 1,
      mhs: [],
      tgl: []
    }
  },
  async mounted() {
    this.msh = await fetch('http://localhost:3000/api/getMhs')
      .then(response => { return response.json() }).then(data => { return data })
    console.log(this.msh);
    this.tgl_sdg = await fetch('http://localhost:3000/api/getTgl')
      .then(response => { return response.json() }).then(data => { return data })
    console.log(this.tgl_sdg);
  },
  methods: {
    addFile() {
      this.halang_num++
    },
    deleteFile() {
      if (this.halang_num > 1) {
        this.halang_num--
      }
    },
    //Mhs
    async submit() {
      let hlg = []
      for (const i in this.mhs) {
        hlg.push({ mhs: this.mhs[i], tgl: this.tgl[i].tanggal })
      }
      fetch('http://localhost:3000/api/insertMhsHalang',
        {
```

```
            headers: {
               'Accept': 'application/json',
               'Content-Type': 'application/json'
            },
            body: JSON.stringify(hlg),
            method: "POST"
          })
        this.$router.push('setjadwal')
      }

  }
}
```

Program 4.8 Program Frontedn Penambahan Mahasiswa Berhalangan

```
app.post('/api/insertMhsHalang', async (req, res) => {
 let jadwal = req.body
 await client.$queryRaw`Truncate jadwal_halang_mhs restart identity cascade`
 for (const d of jadwal) {
  await client.jadwal_halang_mhs.create({
   data: {
    mhs: d.mhs,
    tanggal: d.tgl
   }
  })
 }
 res.send({ data: 'Hello, World!' })
})
```

Program 4.9 Program Input Mahasiswa Berhalangan

```
app.get('/api/getMhs', async (req, res) => {
 const mhs = await client.mahasiswa.findMany()
 let result = Array.from(mhs)
 res.send(result)
})
```

Program 4.10 Program Mengambil Daftar Mahasiswa dari Database

```
app.get('/api/getTgl', async (req, res) => {
 const mhs = await client.$queryRaw`select distinct tanggal from jadwal_halang`
 let result = Array.from(mhs)
 console.log(result[0].tanggal)
 result = result.sort(function (a, b) {
  return moment(a.tanggal, 'DD/MM/yyyy') > moment(b.tanggal, 'DD/MM/yyyy')
   ? 1
   : moment(a.tanggal, 'DD/MM/yyyy') == moment(b.tanggal, 'DD/MM/yyyy')
   ? 0
```

```
    : -1
  })
  res.send(result)
})
```

Program 4.11 Program Mengambil Tanggal Berhalangan

4.1.5 Algoritma Genetika

Setelah mengatur jadwal berhalangan, maka data mahasiswa dan jadwal berhalangan diolah dalam algoritma genetika untuk mencari jadwal sidang yang sesuai dengan keinginan. Proses algoritma tersebut dapat dijelaskan dengan kode berikut.

```
app.get('/api/GA', async (_req, res) => {
  var mhs = await client.mahasiswa.findMany()
  const hlg = await client.jadwal_halang_mhs.findMany()
  var jadwal = await client.jadwal_halang.findMany({
    orderBy: {
      id: 'asc'
    }
  })
  let halanglist = lodash
    .chain(jadwal)
    .map((e) => {
      return { tanggal: e.tanggal, jam_mulai: e.jam_mulai, jam_selesai: e.jam_selesai }
    })
    .uniqWith(lodash.isEqual)
    .map((e) => {
      return { ...e, dosen: [] }
    })
    .value()
  let sorted_jadwal = lodash
    .chain(jadwal)
    .groupBy('dosen')
    .map((val, dsn) => ({ dosen: dsn, val }))
    .value()
  for (const sidang of halanglist) {
    for (const item of sorted_jadwal) {
      let jadwal = Array.from(item.val)
      let res = jadwal.find(
        (x) =>
          x.tanggal == sidang.tanggal &&
          x.jam_mulai == sidang.jam_mulai &&
          x.jam_selesai == sidang.jam_selesai
      )
      if (res.penuh) {
        halanglist.find((x) => x == sidang).dosen.push(Number(item.dosen))
```

```
    }
  }
}
let popnumber = 100
let sessionnum = halanglist.length
let ruang = 3
let mutateChance = 0.2
//Generate Chromosome
var gene = []
for (let i = 0; i < popnumber; i++) {
  let chromo = []
  for (let j = 0; j < sessionnum; j++) {
    chromo.push(Array(ruang).fill(0))
  }
  gene.push(chromo)
}
console.log('Empty gene created')
for (let index = 0; index < gene.length; index++) {
  for (const skripsi of mhs) {
    let sesi = mathjs.randomInt(0, sessionnum)
    let indexZero = gene[index][sesi].findIndex((value) => value == 0)
    while (indexZero == -1) {
      sesi = mathjs.randomInt(0, sessionnum)
      indexZero = gene[index][sesi].findIndex((value) => value == 0)
    }
    gene[index][sesi][indexZero] = skripsi.id
  }
}
console.log('Gene Created')
let final_weight = []
let epoch = 0
let limit = 10000
let weight = []
while (lodash.max(weight) !== 1 && epoch < limit) {
  //Reset Weight
  final_weight = []
  weight = []
  //Fitness Function
  for (const chromo of gene) {
    let num = (await fitnessFunction(chromo, mhs, halanglist, hlg)).weight
    weight.push(1 / num)
  }
  //Selection
  let indexes = gene.map((_val, index) => index)
  let index = []
  let numOfWin = 10
  while (index.length < numOfWin) {
```

```
  let num = mathjs.pickRandom(indexes, 1, weight)[0]
  if (!index.includes(num)) {
   index.push(num)
  }
 }
 index = lodash.shuffle(index)
 index = lodash.chunk(index, 2)
 //Crossover
 for (const pair of index) {
  let ind1 = gene[pair[0]]
  let ind2 = gene[pair[1]]
  let indexes = mhs.map((value) => value.id)
  let objSwap = []
  // get pos of all index
  for (const ind of indexes) {
   let x1 = lodash.findIndex(ind1, function (val) {
    return lodash.includes(val, ind)
   })
   let x2 = lodash.findIndex(ind2, function (val) {
    return lodash.includes(val, ind)
   })
   let obj = { index: ind, x1: x1, x2: x2 }
   objSwap.push(obj)
  }
  //new ind
  for (let i = 0; i < 20; i++) {
   let newChromo1 = []
   let newChromo2 = []
   for (let j = 0; j < sessionnum; j++) {
    newChromo1.push(Array(ruang).fill(0))
    newChromo2.push(Array(ruang).fill(0))
   }
   for (const swap of objSwap) {
    let flip = Math.random()
    let x1 = swap.x1
    let x2 = swap.x2
    if (flip > 0.5) {
     //if flip >.5, x1 to new1 and x2 to new2
     let inZero1 = newChromo1[x1].findIndex((val) => val === 0)
     let inZero2 = newChromo2[x2].findIndex((val) => val === 0)
     while (inZero1 == -1) {
      let new_index = mathjs.randomInt(0, sessionnum)
      inZero1 = newChromo1[new_index].findIndex((val) => val == 0)
      x1 = new_index
     }
     newChromo1[x1][inZero1] = swap.index
     while (inZero2 == -1) {
```

```
        let new_index = mathjs.randomInt(0, sessionnum)
        inZero2 = newChromo2[new_index].findIndex((val) => val == 0)
        x2 = new_index
      }
      newChromo2[x2][inZero2] = swap.index
    } else {
      //if flip <.5, x1 to new2 and x2 to new1
      let inZero1 = newChromo1[x2].findIndex((val) => val == 0)
      let inZero2 = newChromo2[x1].findIndex((val) => val == 0)
      while (inZero1 == -1) {
        let new_index = mathjs.randomInt(0, sessionnum)
        inZero1 = newChromo1[new_index].findIndex((val) => val == 0)
        x2 = new_index
      }
      newChromo1[x2][inZero1] = swap.index
      while (inZero2 == -1) {
        let new_index = mathjs.randomInt(0, sessionnum)
        inZero2 = newChromo2[new_index].findIndex((val) => val == 0)
        x1 = new_index
      }
      newChromo2[x1][inZero2] = swap.index
    }
  }
  //Mutation Ind 1
  if (Math.random() <= mutateChance) {
    let index = lodash.sample(indexes)
    let x1 = lodash.findIndex(newChromo1, function (val) {
      return lodash.includes(val, index)
    })
    let y1 = newChromo1[x1].findIndex((val) => val === index)
    let x2 = mathjs.randomInt(0, sessionnum)
    while (x2 == index) {
      x2 = mathjs.randomInt(0, sessionnum)
    }
    let y2 = newChromo1[x2].findIndex((val) => val === 0)
    while (y2 === -1) {
      x2 = mathjs.randomInt(0, sessionnum)
      while (x2 == index) {
        x2 = mathjs.randomInt(0, sessionnum)
      }
      y2 = newChromo1[x2].findIndex((val) => val === 0)
    }
    newChromo1[x1].splice(y1, 1)
    newChromo1[x1].push(0)
    newChromo1[x2][y2] = index
  }
  //Mutation Ind 2
```

```
    if (Math.random() <= mutateChance) {
      let index = lodash.sample(indexes)
      let x1 = lodash.findIndex(newChromo2, function (val) {
        return lodash.includes(val, index)
      })
      let y1 = newChromo2[x1].findIndex((val) => val === index)
      let x2 = mathjs.randomInt(0, sessionnum)
      while (x2 == index) {
        x2 = mathjs.randomInt(0, sessionnum)
      }
      let y2 = newChromo2[x2].findIndex((val) => val === 0)
      while (y2 === -1) {
        x2 = mathjs.randomInt(0, sessionnum)
        while (x2 == index) {
          x2 = mathjs.randomInt(0, sessionnum)
        }
        y2 = newChromo2[x2].findIndex((val) => val === 0)
      }
      newChromo2[x1].splice(y1, 1)
      newChromo2[x1].push(0)
      newChromo2[x2][y2] = index
    }
    //Result put back in Pop
    gene.push(newChromo1)
    gene.push(newChromo2)
  }
}
for (const genes of gene) {
  let res = await fitnessFunction(genes, mhs, halanglist, hlg)
  final_weight.push(1 / res.weight)
  final_con.push(res.con)
}
epoch++
//Elimination
if (epoch > 0) {
  let sort = []
  for (let i = 0; i < gene.length; i++) {
    const element = gene[i]
    sort.push({ gene: element, weight: final_weight[i] })
  }
  sort.sort(function (a, b) {
    return a.weight > b.weight ? -1 : a.weight == b.weight ? 0 : 1
  })
  sort.splice(sort.length / 2)
  gene = sort.map((v) => v.gene)
  final_weight = sort.map((v) => v.weight)
}
```

```
  }
  if (lodash.max(weight) == 1) {
    await client.$queryRaw`Truncate jadwal_sidang restart identity cascade`
    let max = Math.max(...final_weight)
    let max_index = final_weight.findIndex((val) => val == max)
    let winner = gene[max_index]
    for (const [i, sesi] of winner.entries()) {
      let ruang = 1
      let newsesi = sesi
      let jdlHalang = halanglist[i]
      for (const ind of newsesi) {
        if (ind !== 0) {
          let Curmhs = mhs.find((v) => v.id === ind)
          await client.jadwal_sidang.create({
            data: {
              mhs: Curmhs.id,
              ruang: 'R' + ruang,
              tanggal: jdlHalang.tanggal,
              jam_mulai: jdlHalang.jam_mulai,
              jam_selesai: jdlHalang.jam_selesai
            }
          })
        }
        ruang++
      }
    }
    res.send({ data: 'Success' })
  } else {
    res.send({ data: 'Failed' })
  }
})
```

Program 4.12 Program Algoritma Genetika

```
async function fitnessFunction(gene, ref, jadwal, halang) {
  let con = 0
  let genes = lodash.cloneDeep(gene)
  let jadwalHalang = lodash.cloneDeep(jadwal)
  let refs = lodash.cloneDeep(ref)
  let hlg = lodash.cloneDeep(halang)
  genes.forEach((value, i) => {
    let dna = value
      .map((val) => (val != 0 ? refs.find((value) => value.id == val) : null))
      .filter((v) => v !== null)
    let newDna = dna.map((val) => {
      return {
        index: val.id,
```

```
      mahasiswa: val.nama_mhs,
      dosen: [val.dosbing1, val.dosbing2, val.kpenguji, val.apenguji].filter((v) => v != '')
    }
  })
  let conflict = 0
  if (newDna.length > 1) {
    //count conflict
    let dosens = []
    for (const val of newDna) {
      dosens.push(val.dosen)
    }
    let combine = lodash.combinations(dosens, 2)
    for (const value of combine) {
      conflict += lodash.intersection(value[0], value[1]).length
    }
    con += conflict
  }
  //count konflik dengan jadwal
  let conflict2 = 0
  newDna.forEach((v) => {
    conflict2 = lodash.intersection(v.dosen, jadwalHalang[i].dosen).length
    con += conflict2
  })
  let res=hlg.filter((e)=>e.tanggal==jadwalHalang[i].tanggal)
  res=res?res.map((e)=>e.mhs):""
  let newDnaid=newDna.map((e)=>e.index)
  let conflict3=lodash.intersection(newDnaid,res).length
  con+=conflict3
 })
 return { weight: con + 1, con: con }
}
```

Program 4.13 Program Fitness Function

4.2 Implementai Sistem Validasi

Dalam algoritma ini akan dilakukan validasi untuk mengecek jika kromosom merupakan kromosom yang valid dan dapat dimasukan ke dalam populasi. Jika kromosom yang dibentuk tidak valid, maka kromosom akan dibuat lagi sampai kromosom tersebut valid. Kode sistem validasi dapat dilihat di program berikut.

```
async function validationTest(gene, ref, jadwal, halang) {
  let con = 0
  let genes = lodash.cloneDeep(gene)
  let jadwalHalang = lodash.cloneDeep(jadwal)
  let refs = lodash.cloneDeep(ref)
  let hlg = lodash.cloneDeep(halang)
  genes.forEach((value, i) => {
    let dna = value
      .map((val) => (val != 0 ? refs.find((value) => value.id == val) : null))
      .filter((v) => v !== null)
    let newDna = dna.map((val) => {
      return {
        index: val.id,
        mahasiswa: val.nama_mhs,
        dosen: [val.dosbing1, val.dosbing2, val.kpenguji, val.apenguji].filter((v) => v != '')
      }
    })
    let conflict = 0
    if (newDna.length > 1) {
      //count conflict
      let dosens = []
      for (const val of newDna) {
        dosens.push(val.dosen)
      }
      let combine = lodash.combinations(dosens, 2)
      for (const value of combine) {
        conflict += lodash.intersection(value[0], value[1]).length
      }
      con += conflict
    }
    //count konflik dengan jadwal
    let conflict2 = 0
    newDna.forEach((v) => {
      conflict2 = lodash.intersection(v.dosen, jadwalHalang[i].dosen).length
      con += conflict2
    })
    let res = hlg.filter((e) => e.tanggal == jadwalHalang[i].tanggal)
    res = res.map((e) => e.mhs)
    let newDnaid = newDna.map((e) => e.index)
    let conflict3 = lodash.intersection(newDnaid, res).length
    con += conflict3
  })
  return { weight: con + 1, con: con }
}
```

Program 4.14 Sistem Validasi

```
async function newFitnessfunction(gene) {
  let genes = lodash.cloneDeep(gene)
  let x = 0
```

```
  let y = 0
  let temp_y=0
  genes.forEach((value,i) => {
    if (value.some((el) => el > 0)) {
      x++
    }
    if (i>0) {
      if (genes[i-1].some((el) => el > 0)||temp_y>0) {
        if (value.every((el)=>el==0)) {
          temp_y++
        }
        else{
          y+=temp_y
          temp_y=0
        }
      }
    }
  })
  return { weight: 1/(x+y) }
}
```

Program 4.15 Program Fitness Function Baru