

## Penyelesaian FSSP dengan objektif minimasi Total Flow Time menggunakan Tabu Search - Jonathan Henry - 6182101047

Kasus kecil yang digunakan untuk mengetes program minimasi Total Flow Time menggunakan Tabu Search :

```
SA_kasuskecil.txt x tai20_5.txt Individu.java SimulatedAnnealing.java FSS.java
SA_kasuskecil.txt
1 number of jobs, number of machines, initial seed, upper bound and lower bound :
2 20 5 88325120 1108 1082
3 processing times :
4 27 92 75 94 18 41 37 58 56 20 2 39 91 81 33 14 88 22 36 65
5 79 23 66 5 15 51 2 81 12 40 59 32 16 87 78 41 43 94 1 93
6 22 93 62 53 30 34 27 30 54 77 24 47 39 66 41 46 24 23 68 50
7 93 22 64 81 94 97 54 82 11 91 23 32 26 22 12 23 34 87 59 2
8 38 84 62 10 11 93 57 81 10 40 62 49 90 34 11 81 51 21 39 27
```

Method untuk menghitung Total Flow Time :

```
// Method untuk menghitung Total Flow Time
public int hitungTotalFlowTime(int[] urutanPekerjaan) {
    int[][] waktuSelesai = new int[jadwal.length][urutanPekerjaan.length];
    int totalFlowTime = 0;

    for (int i = 0; i < urutanPekerjaan.length; i++) {
        int pekerjaan = urutanPekerjaan[i] - 1;
        for (int j = 0; j < jadwal.length; j++) {
            int mesin = j;
            if (i == 0 && j == 0) {
                waktuSelesai[j][i] = jadwal[mesin][pekerjaan];
            } else if (i == 0) {
                waktuSelesai[j][i] = waktuSelesai[j-1][i] + jadwal[mesin][pekerjaan];
            } else if (j == 0) {
                waktuSelesai[j][i] = waktuSelesai[j][i-1] + jadwal[mesin][pekerjaan];
            } else {
                waktuSelesai[j][i] = Math.max(waktuSelesai[j-1][i], waktuSelesai[j][i-1]) + jadwal[mesin][pekerjaan];
            }
        }
        totalFlowTime += waktuSelesai[jadwal.length-1][i];
    }
    return totalFlowTime;
}
```

## Penyelesaian FSSP dengan objektif minimasi Total Flow Time menggunakan Tabu Search - Jonathan Henry - 6182101047

Method untuk implelementasi Tabu Search :

```
// Method untuk implementasi Tabu Search
public int[] tabuSearch(int[] solusiAwal, int maxIterasi, int ukuranTabu) {
    int[] solusiTerbaik = solusiAwal.clone();
    int[] solusiSekarang = solusiAwal.clone();
    int nilaiTerbaik = hitungTotalFlowTime(solusiTerbaik);

    List<int[]> tabuList = new ArrayList<>();
    Random random = new Random();

    System.out.println(x:"Memulai Tabu Search...");
    System.out.println("Solusi awal: " + Arrays.toString(solusiAwal));
    System.out.println("Nilai awal: " + nilaiTerbaik);

    for (int iter = 0; iter < maxIterasi; iter++) {
        int[] solusiTetangga = null;
        int nilaiTetangga = Integer.MAX_VALUE;
        int[] bestMove = null;

        // Generate semua tetangga dengan swap
        for (int i = 0; i < solusiSekarang.length; i++) {
            for (int j = i+1; j < solusiSekarang.length; j++) {
                int[] tetangga = solusiSekarang.clone();
                // Swap
                int temp = tetangga[i];
                tetangga[i] = tetangga[j];
                tetangga[j] = temp;

                // Periksa apakah move ini ada di tabu list
                boolean isTabu = false;
                for (int[] tabuMove : tabuList) {
                    if (tabuMove[0] == i && tabuMove[1] == j || tabuMove[0] == j &&
                        tabuMove[1] == i) {
                        isTabu = true;
                        break;
                    }
                }

                // Aspiration criteria: Jika tabu tapi lebih baik dari solusi terba
                int nilaiSekarang = hitungTotalFlowTime(tetangga);
                if (!isTabu || nilaiSekarang < nilaiTerbaik) {
                    if (nilaiSekarang < nilaiTetangga) {
```

## Penyelesaian FSSP dengan objektif minimasi Total Flow Time menggunakan Tabu Search - Jonathan Henry - 6182101047

Modifikasi Class Hasil.java dengan menambahkan method untuk menampilkan hasil

Tabu Search:

```
public Hasil(int[] bestSolution, int makespan, int totalFlowTime) {  
    initComponents();  
    this.tampung = "";  
    this.pilih = new JFileChooser();  
  
    tampung += "Urutan job terbaik:\n";  
    for (int job : bestSolution) {  
        tampung += job + " ";  
    }  
  
    tampung += "\n\nMakespan: " + makespan + "\n";  
    tampung += "Total Flow Time: " + totalFlowTime + "\n\n";  
    this.HasilTerurutTextArea.append(tampung);  
}
```

## Penyelesaian FSSP dengan objektif minimasi Total Flow Time menggunakan Tabu Search - Jonathan Henry - 6182101047

Modifikasi Class Masukan.java dengan menambahkan parameter untuk Tabu Search:

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
    // Ambil input dari GUI  
    String lokasiTxt = this.LokasiField.getText();  
    int bnykIndividu = (int) this.BnykIndividuSpinner.getValue();  
    int maksIterasi = (int) this.MaksIterasiSpinner.getValue();  
    int soalTerpilih = Integer.parseInt(this.SoaTerpilih.getSelectedItem().toString()) - 1;  
  
    // Baca data dari file  
    TestRead pembaca = new TestRead(lokasiTxt);  
    int[][] jadwal = pembaca.getKumpulanSoal()[soalTerpilih].getSoal();  
  
    // Buat solusi awal random  
    int[] solusiAwal = new int[jadwal[0].length];  
    for (int i = 0; i < solusiAwal.length; i++) {  
        solusiAwal[i] = i + 1;  
    }  
  
    // Acak solusi awal  
    Random rng = new Random();  
    for (int i = solusiAwal.length - 1; i > 0; i--) {  
        int j = rng.nextInt(i + 1);  
        int temp = solusiAwal[i];  
        solusiAwal[i] = solusiAwal[j];  
        solusiAwal[j] = temp;  
    }  
  
    // Jalankan Tabu Search  
    FSS fss = new FSS(individu:null, jadwal);  
    int[] bestSolution = fss.tabuSearch(solusiAwal, maksIterasi, ukuranTabu:10); // Ukuran tabu list = 10  
    int makespan = fss.getMakespan();  
    int totalFlowTime = fss.hitungTotalFlowTime(bestSolution);  
  
    // Tampilkan hasil  
    Hasil h = new Hasil(bestSolution, makespan, totalFlowTime);  
    h.setVisible(b:true);  
    this.dispose();  
} // GEN-LAST:event_jButton2ActionPerformed  
// GEN-LAST:event_jButton2ActionPerformed
```

## Penyelesaian FSSP dengan objektif minimasi Total Flow Time menggunakan Tabu Search - Jonathan Henry - 6182101047

Hasil perhitungan program dengan Tabu Search:

```
Iterasi 0: Solusi baru terbaik = 16473
Iterasi 1: Solusi baru terbaik = 15361
Iterasi 2: Solusi baru terbaik = 14690
Iterasi 3: Solusi baru terbaik = 14103
Iterasi 4: Solusi baru terbaik = 13884
Iterasi 5: Solusi baru terbaik = 13748
Iterasi 1: Solusi baru terbaik = 15361
Iterasi 2: Solusi baru terbaik = 14690
Iterasi 3: Solusi baru terbaik = 14103
Iterasi 4: Solusi baru terbaik = 13884
Iterasi 5: Solusi baru terbaik = 13748
Iterasi 3: Solusi baru terbaik = 14103
Iterasi 4: Solusi baru terbaik = 13884
Iterasi 5: Solusi baru terbaik = 13748
Iterasi 5: Solusi baru terbaik = 13748
Iterasi 6: Solusi baru terbaik = 13659
Iterasi 7: Solusi baru terbaik = 13515
Iterasi 8: Solusi baru terbaik = 13454
Iterasi 9: Solusi baru terbaik = 13410
Iterasi 10: Solusi baru terbaik = 13359
Iterasi 11: Solusi baru terbaik = 13315
Iterasi 6: Solusi baru terbaik = 13659
Iterasi 7: Solusi baru terbaik = 13515
Iterasi 8: Solusi baru terbaik = 13454
Iterasi 9: Solusi baru terbaik = 13410
Iterasi 10: Solusi baru terbaik = 13359
Iterasi 11: Solusi baru terbaik = 13315
Iterasi 12: Solusi baru terbaik = 13278
Iterasi 7: Solusi baru terbaik = 13515
Iterasi 8: Solusi baru terbaik = 13454
Iterasi 9: Solusi baru terbaik = 13410
Iterasi 10: Solusi baru terbaik = 13359
Iterasi 11: Solusi baru terbaik = 13315
Iterasi 12: Solusi baru terbaik = 13278
Iterasi 14: Solusi baru terbaik = 13248
Iterasi 27: Solusi baru terbaik = 13241
Iterasi 12: Solusi baru terbaik = 13278
Iterasi 14: Solusi baru terbaik = 13248
Iterasi 27: Solusi baru terbaik = 13241
Iterasi 14: Solusi baru terbaik = 13248
Iterasi 27: Solusi baru terbaik = 13241
Tabu Search selesai
Solusi terbaik: [5, 11, 7, 12, 10, 9, 16, 18, 15, 13, 1, 19, 20, 4, 2, 8, 3, 14, 17, 6]
Nilai terbaik: 13241
```

# Penyelesaian FSSP dengan objektif minimasi Total Flow Time menggunakan Tabu Search - Jonathan Henry - 6182101047

Hasil Eksperimen				
Urutan job terbaik: 11 7 16 5 15 9 1 12 19 13 18 2 6 10 20 3 14 17 4 8				
Makespan: 0 Total Flow Time: 13083				
No. soal	Batas atas	Batas bawah	Makespan	Urutan pekerjaan