Jurnal Komputer dan Teknologi Informasi
Vol. 3, No. 2, Juli 2025, pp. 53~58
E-ISSN: 2986-7592, DOI: 10.26714/jkti.v3i2.16604

Optimasi Penjadwalan Kerja dengan Kendala Operasional Menggunakan Pemrograman Linier

Sholikin[1], Ahmad Ilham, S.Kom., M.Kom. [2]
[1]Program Studi Informatika, Fakultas Teknik dan Ilmu Komputer Universitas Muhammadiyah Semarang, Semarang, Indonesia
[2]Program Studi Informatika, Fakultas Teknik dan Ilmu Komputer Universitas Muhammadiyah Semarang, Semarang, Indonesia

Info Artikel

Diterima 10 Mei 2025
Perbaikan 13 Juni 2025
Disetujui 28 Juli 2025

Keywords:

Optimasi penjadwalan,
Pemrograman Linier,
Efisiensi Biaya,
Python

ABSTRAK

Penjadwalan kerja karyawan yang optimal menjadi tantangan penting bagi perusahaan manufaktur untuk memastikan efisiensi operasional dan pengelolaan biaya. Penelitian ini bertujuan untuk mengembangkan model optimasi berbasis pemrograman linier guna meminimalkan biaya tenaga kerja dengan tetap memenuhi kebutuhan tenaga kerja harian dan kendala operasional lainnya, seperti jam kerja maksimum per karyawan. Studi kasus melibatkan 15 karyawan dengan tingkat upah yang bervariasi, yang dijadwalkan selama 7 hari kerja. Hasil penelitian menunjukkan bahwa solusi yang dihasilkan mampu meminimalkan biaya tenaga kerja dengan memenuhi semua kendala operasional yang ada. Implementasi model dilakukan menggunakan *Python* dengan pustaka *pulp*, dan analisis lebih lanjut menunjukkan efisiensi model dalam pengelolaan sumber daya manusia.

ABSTRACT

Optimal employee scheduling remains a critical challenge for manufacturing companies to ensure operational efficiency and cost management. This study aims to develop an optimization model based on linear programming to minimize labor costs while meeting daily workforce requirements and operational constraints, such as the maximum weekly working hours per employee. The case study involves 15 employees with varying wage rates scheduled over 7 workdays. The results show that the generated solution effectively minimizes labor costs while satisfying all operational constraints. The model is implemented using Python with the PuLP library, and further analysis highlights its efficiency in workforce resource management.

Ini adalah artikel akses terbuka di bawah lisensi CC BY-SA.

Penulis Korespondensi:

Sholikin
Program Studi Informatika, Fakultas Teknik Universitas Muhammadiyah Semarang
Alamat: Gedung FT-MIPA Lt. 7, Ruang 707, Jl.Kedungmundu Raya No.18, Semarang 50273, Indonesia
Email: sholikin.alafasy7@gmail.com

1. PENDAHULUAN

Penjadwalan karyawan merupakan aspek penting dalam manajemen operasional perusahaan, khususnya di sektor manufaktur yang memiliki kebutuhan tenaga kerja variabel setiap harinya. Proses penjadwalan yang tidak optimal dapat menyebabkan peningkatan biaya tenaga kerja dan ketidakefisienan operasional. Penjadwalan manual sering kali tidak memadai untuk menyelesaikan masalah ini karena kompleksitas kendala

yang harus dipertimbangkan, seperti kebutuhan minimal tenaga kerja harian, batas jam kerja mingguan, dan variasi tingkat upah karyawan.

Penelitian ini berfokus pada pengembangan model matematis berbasis pemrograman linier untuk memecahkan masalah penjadwalan tenaga kerja dengan kendala operasional tertentu. Model ini bertujuan meminimalkan total biaya tenaga kerja sambil memenuhi kebutuhan tenaga kerja harian perusahaan. Pendekatan berbasis teknologi, seperti pemrograman *Python* dengan pustaka optimasi, memungkinkan penyelesaian masalah secara efisien dan sistematis.

Pada studi kasus ini, model diterapkan pada perusahaan manufaktur dengan 15 karyawan untuk dijadwalkan selama satu minggu. Hasil dari model ini tidak hanya memberikan jadwal kerja yang optimal tetapi juga memberikan wawasan bagi pengambil keputusan terkait efisiensi sumber daya manusia.

Penelitian ini diharapkan dapat memberikan kontribusi praktis dalam manajemen penjadwalan tenaga kerja, serta menjadi referensi bagi penelitian serupa di masa depan.

2. MODEL MATEMATIS

a. Fungsi Objektif

$$\text{Minimize: } Z = \sum_{k=1}^{15} \sum_{h=1}^{7} \text{upah}_k \cdot x_{kh}$$

b. Kendala

Total jam kerja harian:

$$\sum_{k=1}^{15} x_{kh} \geq 64, \quad \forall h \in \{1, 2, \ldots, 7\}$$

Total jam kerja mingguan per karyawan:

$$\sum_{h=1}^{7} x_{kh} \leq 48, \quad \forall k \in \{1, 2, \ldots, 15\}$$

Jam kerja harian maksimum:

$$0 \leq x_{kh} \leq 8, \quad \forall k \in \{1, 2, \ldots, 15\}, \forall h \in \{1, 2, \ldots, 7\}.$$

3. IMPLEMENTASI PYTHON

Inisialisasi Model

```
model = pulp.LpProblem("Optimasi_Penjadwalan_Karyawan",
pulp.LpMinimize)
```

Keterangan :
Membuat model optimasi linear bernama "Optimasi_Penjadwalan_Karyawan" dengan tujuan meminimalkan biaya tenaga kerja.

Definisi Variabel Keputusan

```
x = pulp.LpVariable.dicts("JamKerja", ((k, h) for k in karyawan
for h in range(1, hari_dalam_minggu + 1)),
                          lowBound=0, upBound=max_jam_per_hari,
cat="Continuous")
```

Keterangan :
- Variabel keputusan x[k, h] merepresentasikan jumlah jam kerja karyawan k pada hari h.
- Batasan:

- Minimum: 0 (tidak bekerja).
- Maksimum: 8 jam per hari.

Fungsi Objektif

```
total_biaya = pulp.lpSum(upah_per_jam[k] * x[k, h]
                         for k in karyawan for h in range(1,
hari_dalam_minggu + 1))
model += total_biaya, "Total Biaya Tenaga Kerja"
```

Keterangan :
- Menetapkan tujuan model untuk meminimalkan total biaya tenaga kerja.
- Biaya dihitung dengan mengalikan upah per jam karyawan k dengan jam kerja x[k, h] pada semua hari.

Kendala 1: Minimal 8 Karyawan Per Hari

```
for h in range(1, hari_dalam_minggu + 1):
    model += pulp.lpSum(x[k, h] for k in karyawan) >=
minimal_karyawan_per_hari * max_jam_per_hari,
f"Kebutuhan Hari {h}"
```

Keterangan :
- Setiap hari harus ada minimal 8 karyawan yang bekerja selama 8 jam total.
- Kendala ini memastikan kebutuhan tenaga kerja harian terpenuhi.

Kendala 2: Maksimal Jam Kerja Per Minggu

```
for k in karyawan:
    model += pulp.lpSum(x[k, h] for h in range(1,
hari_dalam_minggu + 1)) <= max_jam_per_minggu, f"Maks_Jam_{k}"
```

Keterangan :
- Setiap karyawan tidak boleh bekerja lebih dari 48 jam dalam seminggu.
- Kendala ini mencegah beban kerja berlebihan pada karyawan.

Pemecahan Model

```
model.solve()

if model.status == pulp.LpStatusOptimal:
    print("Solusi Optimal Ditemukan:\n")
    total_biaya_kerja = pulp.value(model.objective)
    print(f"Total Biaya Tenaga Kerja: ${total_biaya_kerja}")
else:
    print("Tidak ada solusi yang memenuhi semua kendala.")
```

Keterangan :
- Model diselesaikan menggunakan model.solve().
- Jika solusi optimal ditemukan total biaya tenaga kerja dihitung dari nilai fungsi objektif, jadwal kerja optimal per karyawan dan hari ditampilkan.
- Jika tidak ada solusi, pesan kesalahan akan ditampilkan.

4. HASIL DAN ANALISIS

4.1 Pemenuhan Kendala

Solusi yang dihasilkan memenuhi semua kendala yang telah ditentukan. Setiap hari terdapat 8 karyawan yang masing-masing bekerja selama 8 jam, sehingga total jam kerja kumulatif per hari adalah 8×8=648, sesuai dengan kebutuhan minimal. Selain itu, tidak ada karyawan yang bekerja lebih dari 48 jam per minggu, dan jam kerja harian setiap karyawan tidak melebihi 8 jam. Dengan demikian, solusi ini memenuhi semua batasan yang telah ditetapkan dalam model.

Table 1. Skema jadwal optimal

Hari	K1	K2	K3	K4	K7	K8	K9	K11	K12	K13	K14
Hari 1		8	8	8	8		8	8	8	8	
Hari 2	8	8	8			8	8	8	8	8	
Hari 3		8		8	8	8	8	8	8	8	
Hari 4			8	8	8	8	8		8	8	8
Hari 5		8	8	8	8	8	8	8	8		
Hari 6		8	8	8	8	8	8	8		8	
Hari 7		8	8	8	8	8		8	8	8	

Keterangan :

- Setiap kolom mewakili tugas (K1, K2, dst.).
- Setiap baris mewakili hari (Hari 1, Hari 2, dst.).
- Angka 8.0 menunjukkan waktu yang dihabiskan (jam) pada tugas tertentu di hari tersebut.
- Kosong berarti tugas tersebut tidak dilakukan pada hari itu.

Tabel 1 menunjukkan hasil implementasi model optimasi penjadwalan menggunakan pustaka PuLP, yang merupakan library Python untuk menyelesaikan masalah optimasi linier, diperoleh jadwal kerja optimal untuk memenuhi kebutuhan operasional perusahaan selama 7 hari. Jadwal ini memastikan pemenuhan jumlah minimal karyawan yang diperlukan setiap hari serta mengoptimalkan distribusi beban kerja untuk setiap karyawan dengan batas maksimal 48 jam per minggu.

4.2 Efisiensi Biaya Tenaga Kerja

Fungsi objektif model berhasil meminimalkan total biaya tenaga kerja, yang mencapai $5808.0. Biaya ini dihitung berdasarkan upah per jam setiap karyawan, dengan memprioritaskan karyawan yang memiliki upah lebih rendah untuk memenuhi kebutuhan harian. Oleh karena itu, solusi yang dihasilkan adalah solusi optimal dari segi biaya tenaga kerja.

4.3 Rekomendasi Alternatif Solusi

Meskipun solusi memenuhi kendala dan meminimalkan biaya, beberapa alternatif solusi dapat dipertimbangkan untuk meningkatkan aspek lain dari sistem penjadwalan, seperti:

1. Rotasi Karyawan: Beberapa karyawan digunakan lebih sering dibandingkan yang lain, seperti K2 dan K3, yang bekerja hampir setiap hari. Implementasi rotasi kerja yang lebih merata dapat meningkatkan kepuasan kerja karyawan.
2. Analisis Sensitivitas: Melakukan analisis sensitivitas pada struktur upah dapat membantu perusahaan memahami dampak perubahan tarif upah terhadap biaya tenaga kerja.
3. Peningkatan Produktivitas: Memberikan pelatihan atau adopsi teknologi untuk meningkatkan efisiensi karyawan dapat menjadi solusi jangka panjang untuk mengurangi kebutuhan jam kerja tanpa meningkatkan biaya.

5. KESIMPULAN

Solusi yang dihasilkan memenuhi semua kendala yang diberikan dan meminimalkan biaya tenaga kerja secara efektif. Alternatif solusi dapat diterapkan untuk meningkatkan efisiensi operasional atau kepuasan kerja karyawan.

Lampiran

Kode Python

```
import pulp

# Data masalah
karyawan = [f"K{i}" for i in range(1, 16)]
upah_per_jam = {
    "K1": 15, "K2": 12, "K3": 14, "K4": 13, "K5": 16,
    "K6": 15, "K7": 11, "K8": 14, "K9": 13, "K10": 15,
    "K11": 12, "K12": 13, "K13": 14, "K14": 15, "K15": 16
}
max_jam_per_hari = 8
max_jam_per_minggu = 48
hari_dalam_minggu = 7
minimal_karyawan_per_hari = 8

# Inisialisasi model
model = pulp.LpProblem("Optimasi_Penjadwalan_Karyawan",
pulp.LpMinimize)

# Definisi variabel keputusan
# x[k,h] adalah jumlah jam kerja karyawan k di hari h
x = pulp.LpVariable.dicts("JamKerja", ((k, h) for k in karyawan for
h in range(1, hari_dalam_minggu + 1)),
                          lowBound=0, upBound=max_jam_per_hari,
cat="Continuous")

# Fungsi objektif: Minimalkan total biaya tenaga kerja
total_biaya = pulp.lpSum(upah_per_jam[k] * x[k, h]
                         for k in karyawan for h in range(1,
hari_dalam_minggu + 1))
model += total_biaya, "Total_Biaya_Tenaga_Kerja"

# Kendala: Setiap hari minimal ada 8 karyawan bekerja maksimal 8
jam
for h in range(1, hari_dalam_minggu + 1):
    model += pulp.lpSum(x[k, h] for k in karyawan) >=
minimal_karyawan_per_hari * max_jam_per_hari, f"Kebutuhan_Hari_{h}"
```

```
# Kendala: Total jam kerja per karyawan dalam seminggu tidak
melebihi 48 jam
for k in karyawan:
    model += pulp.lpSum(x[k, h] for h in range(1, hari_dalam_minggu
+ 1)) <= max_jam_per_minggu, f"Maks_Jam_{k}"

# Pemecahan masalah
model.solve()

# Menampilkan hasil
if model.status == pulp.LpStatusOptimal:
    print("Solusi Optimal Ditemukan:\n")
    total_biaya_kerja = pulp.value(model.objective)
    print(f"Total Biaya Tenaga Kerja: ${total_biaya_kerja}")
    print("Jadwal Kerja Optimal:")
    for h in range(1, hari_dalam_minggu + 1):
        print(f"Hari {h}:")
        for k in karyawan:
            jam_kerja = x[k, h].varValue
            if jam_kerja > 0:
                print(f"  - {k}: {jam_kerja} jam")
else:
    print("Tidak ada solusi yang memenuhi semua kendala.")
```