# OPTIMASI PENJADWALAN PENUGASAN CRANE DENGAN ALGORITMA BRANCH AND PRICE

**Yudhi Purwananto, Chastine Fatichah, Anna Kholilah, dan Rully Soelaiman**
*Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember (ITS), Surabaya, 60111*
*E-mail: yudhi@its-sby.edu*

**ABSTRAKSI**

*Perusahaan persewaan crane merupakan perusahaan jasa yang berprinsip memenuhi pesanan dari pelanggan. Karena keterbatasan sumber daya yang dimiliki, maka tidak semua pesanan yang datang dapat diterima. Pesanan dapat diterima apabila sebuah perusahaan sanggup memenuhinya tanpa mengabaikan pesanan-pesanan yang datang terlebih dahulu, karena pesanan yang datang terlebih dahulu seharusnya dilayani terlebih dahulu. Untuk menyelesaikan pesanan-pesanan yang diterimanya, sebuah perusahaan persewaan memiliki seorang planner yang bertugas untuk mendistribusikan semua sumber daya yang ada. Salah satu biaya operasional adalah biaya pengoperasian sumber daya yang dimiliki perusahaan persewaan. Untuk dapat memaksimalkan keuntungan, maka salah satunya adalah dengan meminimalkan biaya operasional yang dikeluarkan.*

*Permasalahan tersebut dapat dirumuskan sebagai permasalahan integer programming yang dapat diselesaikan dengan menggunakan algoritma branch and price. Dua proses yang akan dilakukan yaitu percabangan pada variabel keputusan yang belum integer dan pengecekan adanya reduced cost negatif pada tiap variabel keputusannya. Pengecekan reduced cost ini dilakukan agar tidak melakukan penelusuran pada semua kemungkinan solusi integer, seperti yang dilakukan pada algoritma branch and price. Hasil optimal dicapai pada solusi integer yang semua variabel keputusannya tidak memiliki reduced cost negatif.*

*Uji coba dan evaluasi dilakukan dengan menggunakan data yang didapatkan dari sebuah perusahaan persewaan crane pada NRL. Dari beberapa hasil uji coba yang dilakukan menunjukkan bahwa aplikasi dapat merumuskan integer programming dan dengan algoritma branch and price permasalahan penjadwalan penugasan crane dapat diselesaikan. Dari hasil uji coba yang dilakukan pada data NRL dibuktikan adanya peningkatan solusi optimal lebih dari 10%.*

***Kata kunci:*** *Graph, Integer Programming, Branch and Price, Column Generation*

## 1. PENDAHULUAN

Crane merupakan sebuah alat berat yang memiliki lengan panjang yang digunakan untuk menaikkan atau menurunkan beban yang berat dan dalam keadaan tergantung, memindahkannya secara menyamping [1]. Alat ini biasa digunakan dalam sebuah area konstruksi bangunan. Crane sendiri memiliki beberapa jenis sesuai dengan kapasitas maksimal yang dapat diangkutnya. Karena harganya yang mahal maka ada beberapa tempat yang menyewakan alat ini. Seperti halnya dalam sebuah perusahaan jasa persewaan, maka perusahaan persewaan crane juga berlaku demikian. Order datang dari pelanggan yang membutuhkan crane pada jam tertentu, kapasitas tertentu dan dalam jangka waktu tertentu pula. Semua variabel tersebut dispesifikasikan sendiri oleh pelanggan. Order yang dapat diterima oleh perusahaan ini disebut dengan job.

Penugasan crane dilakukan berdasarkan rangkaian job yang diberikan. Pengoperasian crane-crane ini akan mengeluarkan biaya sesuai dengan job-job yang diselesaikannya dalam satu hari. Biaya ini merupakan biaya operasional perusahaan persewaan crane. Untuk dapat memaksimalkan keuntungan yang didapat perusahaan, maka salah satunya adalah dengan meminimalkan biaya operasional yang dikeluarkan. Meminimalkan biaya inilah yang digunakan sebagai fungsi tujuan yang akan dicapai.

Permasalahan ini kemudian dapat dirumuskan sebagai permasalahan integer programming dan menyelesaikannya dengan menggunakan algoritma branch and price

Tujuan penelitian ini adalah menghasilkan aplikasi optimasi penjadwalan penugasan crane untuk mendapatkan biaya operasional minimal dengan menerapkan algoritma branch and price dalam suatu perusahaan persewaan crane.

## 2. INTEGER PROGRAMMING

Integer programming merupakan linear programming, dimana beberapa atau semua variabelnya dibatasi bernilai integer atau bilangan bulat.

Terdapat tiga tipe dasar model pemrograman integer [2], yaitu:

a. Model Integer Total, semua variabel keputusan harus bernilai bilangan bulat.
b. Model Integer Biner, semua variabel keputusan harus bernilai nol (0) atau satu (1). Dan model ini yang akan digunakan untuk menyelesaikan penjadwalan crane.
c. Model Integer Campuran, dari semua variabel keputusan, hanya beberapa (tidak semua), yang harus bernilai bilangan bulat.

Dari sebuah Integer Programming (IP) dapat dihasilkan sebuah Linear Programming Relaxation (LPR) [3] dari IP dengan menggunakan fungsi obyektif dan batasan yang sama tetapi dengan syarat variabel yang bernilai integer diganti dengan batasan

berlanjut (continous constraint). Sebagai contoh variabel $x_i = 0$ atau 1 dapat diganti dengan dua continous constraint $x_i > 0$ dan $x_i < 1$ atau $0 < x_i < 1$.

## 3. ALGORITMA BRANCH AND PRICE

Algoritma branch and price memodifikasi strategi dasar branch and bound dengan usaha untuk memperkuat linear programming relaxation dari integer progrramming dengan pertidaksamaan baru sebelum melakukan percabangan sebuah solusi fractional. Prosedur branch and price berfokus pada pembangkitan kolom (*column generation*).

Branch and price merupakan penurunan dari branch and bound dengan relaksasi linear programming (LP relaxation) memperbolehkan adanya pembagian dan pricing digunakan pada tree branch and bound. Dalam branch and price, kolom disisihkan dari relaksasi LP karena terdapat terlalu banyak kolom yang ditangani secara efisien. Kemudian untuk memeriksa keoptimalan dari sebuah solusi LP, subproblem yang disebut *pricing problem*, yang merupakan bagian permasalahan untuk dual LP, diselesaikan untuk mencoba mengidentifikasi kolom yang akan memasuki basis. Jika terdapat kolom yang dimaksud tersebut, maka solusi LP dioptimasi kembali. Percabangan terjadi ketika tidak terdapat kolom yang dinilai dapat memasuki basis dan solusi LP tidak memenuhi batasan integral.

Column generation menyediakan penguraian dari sebuah permasalahan menjadi master dan sub-problem. Penguraian ini memungkinkan adanya penafsiran dalam pengaturan kontektual untuk penambahan beberapa batasan penting. Terdapat beberapa kesulitan dalam penggunaan teknik column generation untuk linear programming dalam solusi integer programming, yaitu:

- Integer programming konvensional bercabang pada variabel yang mungkin tidak efektif karena memperbaiki variabel dapat merusak struktur pricing problem. Menyelesaikan LP dan subproblem menjadi optimal mungkin menjadi tidak efisien, dimana beberapa aturan berbeda akan digunakan untuk menangani tree branch-and-price.
- LP relaxation dari master problem diselesaikan yang dengan column generation, mungkin tidak memiliki solusi optimal integral dan mengunakan prosedur standar branch and bound untuk master problem pada kolom yang ada tidak seperti halnya mencari solusi optimal, atau yang baik, atau mungkin feasible terhadap permasalahan sebenarnya.

Standar percabangan pada variabel menimbulkan sebuah permasalahan pada sebuah cabang dimana sebuah variabel telah diset nol. Daripada melakukan percabangan pada dalam master problem, kita menggunakan aturan percabangan pada variabel aslinya dalam formulasi program linier aslinya. Misalkan saja variabel $z_{ij}$ yang merupakan variabel 0-1 yang mengindentifikasikan task i akan dilakukan mesin j. Ketika $z_{ij} = 1$, semua kolom yang ada pada master problem yang tidak menugaskan task i pada mesin j didelete dan secara tetap task i akan diberikan pada mesin j. dan variabel tugas xi diremove dari jth knapsack. Setiap knapsack problem meliputi satu variabel setelah percaangan dilakukan.

Perlu diperhatikan, ketika memproses suatu node tidak dibutuhkan untuk menyelesaikan LP sampai optimal. Alasan utama untuk menyelesakan LP sampai optimal adalah adanya kemungkinan untuk memotong node berdasarkan bound. Pemangkasan masih mungkin terjadi tanpa menyelesaikan LP sampai optimal jika kita dapat menghitung bound yang tepat. Berikut ini merupakan ringkasan pengertian mengenai algoritma *branch and price*:

- Branch and price menggabungkan metode branch and price dan column generation untuk menyelesaikan program integer berskala besar.
- Pada tiap node dari tree branch and bound, kolom dapat dibangkitkan untuk memperkuat linear programming relaksasi.
- Pada branch and price, kumpulan kolom dibuang dari linear programming relaxation sebuah program integer berskala besar karena terdapat terlalu banyak kolom yang harus ditangani secaraefisien dan kebanyakan dari kolom itu menghasilkan variabel yang bernilai nol padaa solusi optimal.
- Kemudian cek keoptimalan, sebuah subproblem yang disebut juga dengan ‘pricing problem’ diselesaikan untuk mengidentifikasi adanya kolom yang memasuki basis. Pricing problem dilakukan untuk mengetahui adanya variabel keputusan yang memiliki reduced cost negatif.
- Jika kolom tersebut ditemukan, linear programming kembali dioptimasi.
- Percabangan terjadi ketika tidak terdapat kolom yang melalui proses pricing dan solusi program integer tidak memenuhi batasan kondisi integral.

## 4. PERUMUSAN MODEL MATEMATIKA

Dari deskripsi operasional persewaan crane yang telah dijelaskan sebelumnya, maka ada beberapa asumsi yang digunakan dalam menyelesaikan perumusan masalah. Asumsi-asumsinya adalah sebagai berikut:

1. Perencanaan berlaku untuk satu hari. Hal ini dikarenakan hampir tidak ada satu tugas (job) yang membutuhkan waktu lebih dari satu hari.
2. Tiap crane memiliki kecepatan yang sama.
3. Memiliki waktu mulai yang tetap.

Dengan adanya penjelasan dan asumsi yang digunakan pada penjadwalan crane, maka untuk implementasi dibutuhkan inputan pada variabel-variabel berikut ini:

Untuk tipe crane, maka variabel-variabelnya:

- $t$ = tipe *crane* dimana $t = 1, \ldots, m$
- $m_t$ = jumlah *crane* untuk tiap tipe yang tersedia.
- $cap_t$ = kapasitas untuk tiap *crane* tipe $t$
- $r_t$ = tarip biaya untuk *crane* tipe $t$.

Untuk setiap *job* yang akan dijadwalkan (dimana $i = 1, 2, 3, \ldots, n$)

- $s_i$ = jam mulai untuk *job i*.
- $p_i$ = jangka waktu yang dibutuhkan untuk melakukan *job i*.
- $w_i$ = kapasitas yang dibutuhkan untuk *job i*.
- $d_{0i}$ $(d_{i0})$= waktu yang dibutuhkan untuk melakukan perjalanan dari depot ke lokasi *i* dan sebaliknya dari lokasi *job* ke depot.

Dan untuk setiap hubungan antar *job* dimana $i, j = 1, 2, 3, \ldots, n$, maka:

- $d_{ij}$ = waktu yang dibutuhkan dari lokasi *job i* ke lokasi *job j*.

Semua input yang dibutuhkan bernilai positif integer. Dari penjelasan variabel yang dijelaskan, maka:

1. untuk setiap job i digunakan crane tipe t yang memiliki: $w_i < cap_t$.
2. untuk 2 job (i, j) yang dilakukan oleh satu crane secara berurutan maka $s_i + p_i + d_{ij} + s_{ij} < s_j$.
3. tidak melebihi jumlah $m_t$ untuk tipe crane t yang tersedia
4. meminimalkan total biaya yang dikeluarkan

Untuk menghitung total biaya yang dikeluarkan, misalkan crane tertentu menyelesaikan job $i_1, i_2, i_3, \ldots, i_l$ secara berurutan, maka biaya yang harus dikeluarkan adalah: $r_t (p_{i1} + \ldots + p_{il} + d_{0,i1} + \ldots + d_{il-1,il} + d_{il,0})$. Total biaya yang dikeluarkan didapatkan dari jumlah biaya seluruh operasi crane yang ditugaskan.

Permasalahan penjadwalan crane ini dimodelkan sebagai sebuah directed graph G = (V, A) dengan bobot. Setiap vertex merupakan tugas untuk $i = 1, \ldots, n$ dan $V = \{1, \ldots, n\} \cup \{s, f\}$. Vertek s dan f merupakan depot dan dapat digunakan sebagai source dan sink pada graph G. Terdapat arc dari vertex i ke j dalam A apabila $s_i + p_i + d_{ij} < s_j$ untuk semua $I \in V$. Terdapat arc untuk {s, i }dan {i, f} pada A untuk semua $I \in V$. Kemudian terdapat bobot vektor ctij yang dihubungkan pada setiap arc $\{i, j\} \in A$. Bobot tersebut didapatkan sebagai berikut:

$$c^t_{ij} \begin{cases} r_t (d_{ij} + p_j) & \text{jika } w_j \le cap_t \\ M & \text{jika } w_j \ge cap_t \text{ untuk semua } t, \{i, j\} \in A \\ & M \text{ merupakan sejumlah biaya yang sangat besar.} \end{cases} \quad (1)$$

M jika $w_j > capt$ untuk semua t, $\{i, j\} \in A$ M merupakan sejumlah biaya yang sangat besar.

Dan berikut merupakan parameter-parameter yang digunakan:

- R = kumpulan path dalam graph G dari s ke f. Untuk

$$\delta_{ir} \begin{cases} 1 \text{ jika vertek } i \text{ terdapat pada } path\ r \\ 0 \text{ jika terjadi sebaliknya.} \end{cases} \quad (2)$$

- $c_{tr}$ = merupakan biaya yang dikeluarkan ketika sebuah crane tipe t mengambil path r. Perhitungan biaya dapat dengan mudah didapat dengan menggunakan (1). Perlu diperhatikan bahwa apabila terdapat sebuah vertek yang tidak dapat dilayani oleh crane tipe t maka $c_{tr}$ akan bernilai angka yang sangat besar. Hal ini dilakukan karena fungsi obyektif yang akan dicapai merupakan fungsi minimal.

Untuk, variabel keputusan:

$$y_{tr} = \begin{cases} 1 \text{ sebuah } crane \text{ tipe } t \text{ mengambil } path\ r \\ 0 \text{ jika terjadi sebaliknya.} \end{cases} \quad (3)$$

***Model Matematika***

Min $$\sum_{t=1}^{m} \sum_{r \in R} c_{tr} y_{tr}$$

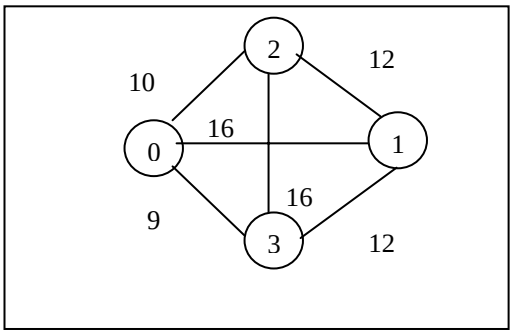
Batasan

$$\sum_{t=1}^{m} \sum_{r \in R} \delta_{ir} y_{tr} = 1 \text{ untuk i = 1,…,n} \quad (4)$$

$$\sum_{r \in R} y_{tr} \le m_t \text{ untuk t = 1,…,m} \quad (5)$$

$$y_{tr} \in \{0,1\} \text{ untuk t =1,…,m,} \quad (6)$$

Batasan (4) menyatakan bahwa setiap vertek harus terlibat satu kali dalam path yang terpilih, pertidaksamaan (5) menyatakan bahwa crane yang digunakan tidak boleh melebihi crane yang tersedia dan batasan (6) merupakan batasan integral.




**Gambar 1.** Graph Jarak Tempuh

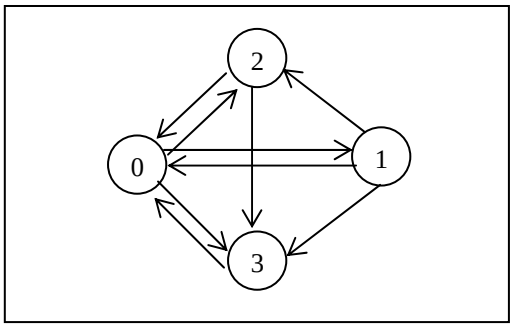
Terdapat rangkain 3 job sebagai berikut $s_1 = 20$, $s_2 = 40$, $s_3 = 60$, $p_1 = p_2 = p_3 = 0$, $w_1 = 1$, $w_2 = w_3 = 0$ .Dan waktu jarak tempuh dari tiap lokasi ke lokasi dij telah dideskripsikan pada gambar 1, dimana node 0, merupakan lokasi depot, node 1 lokasi job 1 dan seterusnya.

Proses pembentukan directed *graph* pada gambar 2 dilakukan jika terdapat *arc* untuk 2 job *(i, j)* yang dapat dilakukan oleh satu crane secara berurutan dengan syarat $s_i + p_i + d_ij < s_j$.

*Path* merupakan rute yang akan dilalui untuk pengoperasian suatu *crane* tertentu. Penentuan *path* yang memungkinkan diperoleh dari *directed graph* yang telah didapatkan pada proses sebelumnya. Dari

*graph* pada gambar 2 terdapat 7 *path* yang terdapat pada graph dari *s* ke *f*:

1. s – 1 - f = a
2. s – 2 – f = b
3. s – 3 – f = c
4. s – 1 – 2 – f = d
5. s – 1 – 3 – f = e
6. s – 2 – 3 – f = f
7. s – 1 – 2 – 3 – f = g




**Gambar 2.** Directed Graph

Dari 7 path yang disebutkan pada proses penentuan path, dengan R = a, …, g, maka dapat diperoleh penurunan model matematikanya sebagai berikut:

- Fungsi Obyektif

$$\text{Min} \quad \sum_{t=1}^{m}\sum_{r\in R} c_{tr}\, y_{tr}$$

$$min \quad c_{1a}y_{1a} + c_{1b}y_{1b} + c_{1c}y_{1c} + c_{1d}y_{1d} + c_{1e}y_{1e} + c_{1f}y_{1f} + c_{1g}y_{1g} + c_{2a}y_{2a} + c_{2b}y_{2b} + c_{2c}y_{2c} + c_{2d}y_{2d} + c_{2e}y_{2e} + c_{2f}y_{2f} + c_{2g}y_{2g}$$

Menerapkan aturan (1) dengan tarif biaya $r_1$ = 1 dan $r_2$ = 3, maka didapatkan perhitungan biaya sebagai berikut:

$$Min \quad M\,y_{1a} + 20\,y_{1b} + 18\,y_{1c} + M\,y_{1d} + M\,y_{1e} + 35\,y_{1f} + M\,y_{1g} + 96\,y_{2a} + 60\,y_{2b} + 54\,y_{2c} + 114\,y_{2d} + 111\,y_{2e} + 105\,y_{2f} + 159\,y_{2g}$$

Agar mempermudah pengidentifikasian, maka dengan mengganti variabel keputusan $y_{tr}$ dengan $x_i$, didapatkan:

$$Min \quad M\,x_1 + 20\,x_2 + 18\,x_3 + M\,x_4 + M\,x_5 + 35\,x_6 + M\,x_7 + 96\,x_8 + 60\,x_9 + 54\,x_{10} + 114\,x_{11} + 111\,x_{12} + 105\,x_{13} + 159\,x_{14}$$

- Batasan (4)

$$\sum_{t=1}^{m}\sum_{r\in R} \delta_{ir}\, y_{tr} = 1 \qquad \text{untuk i = 1,…,n}$$

$$\sum_{r\in R} \delta_{ir}\, y_{1r} + \delta_{ir}\, y_{tr} = 1 \quad \text{untuk i = 1,…,n}$$

untuk i = 1 :

$$\delta_{1a}y_{1a} + \delta_{1b}y_{1b} + \delta_{1c}y_{1c} + \delta_{1d}y_{1d} + \delta_{1e}y_{1e} + \delta_{1f}y_{1f} + \delta_{1g}y_{1g} + \delta_{1a}y_{2a} + \delta_{1b}y_{2b} + \delta_{1c}y_{2c} + \delta_{1d}y_{2d} + \delta_{1e}y_{2e} + \delta_{1f}y_{2f} + \delta_{1g}y_{2g} = 1$$

Dengan aturan (2) dan mengganti variabel keputusan $y_{tr}$ dengan $x_i$, didapatkan:

$$x_1 + 0 + 0 + x_4 + x_5 + 0 + x_7 + x_8 + 0 + 0 + x_{11} + x_{12} + 0 + x_{14} = 1$$

untuk i = 2 :

$$\delta_{2a}y_{1a} + \delta_{2b}y_{1b} + \delta_{2c}y_{1c} + \delta_{2d}y_{1d} + \delta_{2e}y_{1e} + \delta_{2f}y_{1f} + \delta_{2g}y_{2g} + \delta_{2a}y_{2a} + \delta_{2b}y_{2b} + \delta_{2c}y_{2c} + \delta_{2d}y_{2d} + \delta_{2e}y_{2e} + \delta_{2f}y_{2f} + \delta_{2a}y_{2g} = 1$$

Dengan aturan (2) dan mengganti variabel keputusan $y_{tr}$ dengan $x_i$, didapatkan:

$$0 + x_2 + 0 + x_4 + 0 + x_6 + x_7 + 0 + x_9 + 0 + x_{11} + 0 + x_{13} + x_{14} = 1$$

untuk i = 3 :

$$\delta_{3a}y_{1a} + \delta_{3b}y_{1b} + \delta_{3c}y_{1c} + \delta_{3d}y_{1d} + \delta_{3e}y_{1e} + \delta_{3f}y_{1f} + \delta_{3g}y_{1g} + \delta_{3a}y_{2a} + \delta_{3b}y_{2b} + \delta_{3c}y_{2c} + \delta_{3d}y_{2d} + \delta_{3e}y_{2e} + \delta_{3f}y_{2f} + \delta_{3g}y_{2g} = 1$$

Dengan aturan (2) dan mengganti variabel keputusan $y_{tr}$ dengan $x_i$, didapatkan:

$$0 + 0 + x_3 + 0 + x_5 + x_6 + x_7 + 0 + 0 + x_{10} + 0 + x_{12} + x_1 + x_{14} = 1$$

- Batasan (5)

$$\sum_{r\in R} y_{tr} \le m_t \qquad \text{untuk t = 1,…,m}$$

untuk *t = 1* :

$$y_{1a} + y_{1b} + y_{1c} + y_{1d} + y_{1e} + y_{1f} + y_{2g} \le 1$$

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \le 1$$

untuk *t* = 2 :

$$y_{2a} + y_{2b} + y_{2c} + y_{2d} + y_{2e} + y_{2f} + y_{2g} \le 1$$

$$x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} + x_{14} \le 1$$

- Batasan (6)

$y_{tr} \in \{0,1\}$ untuk t =1,…,m,

Untuk mempermudah penyelesaian, maka model matematika integer programming tersebut diubah menjadi sebuah permasalahan linear programming relaxation, oleh karena itu batasan (6) diubah menjadi:

$$x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14} \in \{0,1\}$$

Dari model matematika tersebut, solusi optimal dari linear programming relaxation dengan nilai variabel $x_6$, $x_{11}$, $x_{12}$ = ½ atau $y_{1,\,s-2-3-f}$, $y_{2,\,s-1-2-f}$, $y_{2,\,s-1-3-f}$ = ½ dan variabel yang lain bernilai nol. Sedangkan z optimal = 130

Dari bentuk linear programming yang didapat berdasarkan contoh permasalahan, maka dengan diberikan variabel dual $u_i$ pada batasan (4) dan

variabel dual $e_t$ pada batasan (5) didapatkan dual problem sebagai berikut:

Fungsi Obyektif:

Max $z = u_1 + u_2 + u_3 + e_1 + e_2$

Batasan:

$u_1 + 0 + 0 + e_1 + 0 \le 1000$
$0 + u_2 + 0 + e_1 + 0 \le 20$
$0 + 0 + u_3 + e_1 + 0 \le 18$
$u_1 + u_2 + 0 + e_1 + 0 \le 1000$
$u_1 + 0 + u_3 + e_1 + 0 \le 1000$
$0 + u_2 + u_3 + e_1 + 0 \le 35$
$u_1 + u_2 + u_3 + e_1 + 0 \le 1000$
$u_1 + 0 + 0 + 0 + e_2 \le 96$
$0 + u_2 + 0 + 0 + e_2 \le 60$
$0 + 0 + u_3 + 0 + e_2 \le 54$
$u_1 + u_2 + 0 + 0 + e_2 \le 114$
$u_1 + 0 + u_3 + 0 + e_2 \le 111$
$0 + u_2 + u_3 + 0 + e_2 \le 105$
$u_1 + u_2 + u_3 + 0 + e_2 \le 159$
$u_1, u_2, u_3$ *unrestricted*
$e_1, e_2 \ge 0$

Dari persamaan dual diatas didapatkan hasil optimal maksimal dengan nilai 131 dengan variabel keputusan:

$u_1 = 96 \;\; u_2 = 18 \;\; u_3 = 15 \;\; e_1 = 2 \;\; e_2 = 0$

Dari persamaan dual di atas didapatkan reduced cost sebagai berikut:

$$c_{tr} - \sum_{i=1}^{n} \delta_{ir} u_i - e_t$$

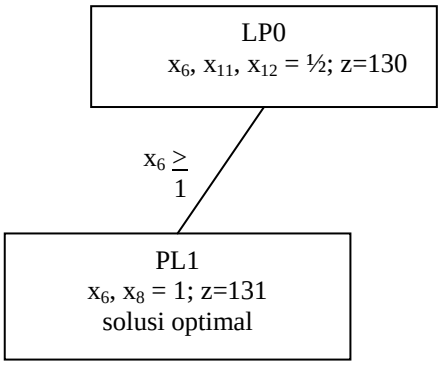
Hasil solusi linear programming dan berkaitan dengan variabel dualnya maka pricing problem akan memberlakukan pertanyaan sebagai berikut:

Price $\exists t, r \in R$ dengan batasan

$$c_{tr} - \sum_{i=1}^{n} \delta_{ir} u_i - e_t < 0 ?$$

Dari perhitungan pada aturan percabangan dan pricing pada subbab sebelumnya, maka algoritma branch and price dapat diterapkan pada contoh permasalahan sebagai berikut ini:

1. Pada LP0 didapatkan solusi optimal $x_6$, $x_{11}$, $x_{12}$ = ½ atau $y_{1, s-2-3-f}$, $y_{2, s-1-2-f}$, $y_{2, s-1-3-f}$ = ½ dengan nilai z optimal = 130.
2. Dilakukan percabangan pada variabel $x_6$ seperti yang ditunjukkan pada gambar 3. menghasilkan LP1 dan LP2. Dari LP1 didapatkan solusi optimal $x_6$, $x_8$ = 1 dengan nilai z =131.




**Gambar 3.** Percabangan pada variabel x6

3. Karena penyelesaian PL1 merupakan solusi integral, maka dilanjutkan dengan penyelesaian pricing sebagai berikut ini:

- $x_6$ atau $y_{1, s-2-3-f}$.
  *t=1* dan *r = f = s-2-3-f*

$$c_{tr} - \sum_{i=1}^{n} \delta_{ir} u_i - e_t$$
$$= c_{1f} - \sum_{i=1}^{3} \delta_{ir} u_i - e_t$$
$$= c_{1f} - \left(\delta_{1f} u_1 + \delta_{2f} u_2 + \delta_{3f} u_3\right) - e_1$$

= 35 – (0 + 1(18) + 1(15) - 2)
= 35 – 34
= 1 ≠ 0

- $x_8$ atau $y_{2, s-1-f}$.
  *t=2* dan *r = f = s-1-f*

$$c_{tr} - \sum_{i=1}^{n} \delta_{ir} u_i - e_t = c_{2f} - \sum_{i=1}^{3} \delta_{ir} u_i - e_t$$
$$= c_{2f} - \left(\delta_{1f} u_1 + \delta_{2f} u_2 + \delta_{3f} u_3\right) - e_2$$

= 96 – (96 + 0 + 0) - 2
= 96 – 96
= 0 ≠ 0

Karena reduced costs yang dihasilkan dari tiap variabel tidak bernilai negatif maka solusi integer ini merupakan solusi optimal dan permasalahan terselesaikan.

## 5. HASIL UJI COBA

Pada uji coba ini akan diberikan beberapa skenario dari beberapa *job* yag akan diberikan untuk mengetahui kebenaran proses-proses pada aplikasi yang telah dibuat. Terdapat 3 skenario yang akan diuji coba dengan kombinasi *job* yang berbeda, yaitu:

1. Skenario pertama dilakukan pada kombinasi *job* yang tidak membutuhkan proses percabangan dan pricing, karena solusi yang dihasilkan pada linear programming sudah merupakan solusi integral dan optimal.
2. Skenario kedua dilakukan pada kombinasi *job* yang tidak membutuhkan proses percabangan tetapi membutuhkan proses pricing, karena

solusi yang dihasilkan pada linear programming merupakan solusi integral tetapi ditemukan adanya variabel keputusan $x_i$ yang memiliki reduced cost < 0, sehingga solusi belum mencapai optimal.

3. Skenario ketiga dilakukan pada kombinasi *job* yang membutuhkan proses percabangan dan pricing, karena solusi yang dihasilkan pada linear programming belum mencapai solusi integral dan optimal.

**Tabel 1.** Data Job Skenario Pertama

| *Id Job* | *s* | *p* | *w* | *loc* |
|---|---|---|---|---|
| 100003537 | 27000 | 31500 | 35 | Vaals |
| 100006061 | 25200 | 25200 | 60 | Landgraaf |
| 100006191 | 25200 | 32400 | 30 | Heerlen |
| 100006290 | 27000 | 31500 | 90 | Herten |
| 100006475 | 27900 | 5400 | 35 | Maastricht |

Untuk skenario pertama rangkaian *job* didapatkan dari 5 *job* yang diambil dari Data_030697, seperti yang ditunjukkan pada tabel 1.

**Tabel 2.** Data Crane

| $t$ | $m_t$ | $cap_t$ | $r_t$ | $speed_t$ | $q_t$ |
|---|---|---|---|---|---|
| 5230 | 1 | 7 | 585 | 40 | 1 |
| 5330 | 2 | 20 | 675 | 40 | 2 |
| 5430 | 3 | 25 | 680 | 40 | 3 |
| 5530 | 11 | 30 | 715 | 40 | 11 |
| 5630 | 1 | 35 | 715 | 40 | 1 |
| 5730 | 4 | 40 | 735 | 40 | 4 |
| 5830 | 3 | 45 | 735 | 40 | 3 |
| 5930 | 0 | 50 | 778 | 40 | 2 |
| 6030 | 2 | 60 | 855 | 40 | 2 |
| 6130 | 2 | 70 | 980 | 40 | 2 |
| 6230 | 0 | 80 | 980 | 40 | 2 |
| 6330 | 2 | 90 | 990 | 40 | 2 |
| 6430 | 0 | 100 | 1030 | 40 | 2 |
| 6530 | 1 | 120 | 1555 | 40 | 1 |
| 6730 | 1 | 150 | 1610 | 40 | 1 |
| 6830 | 0 | 180 | 1640 | 40 | 1 |
| 7030 | 1 | 300 | 1670 | 40 | 1 |

Data crane untuk skenario pertama ditunjukkan pada tabel 2. Dan data distance pada tabel 3.

**Tabel 3.** Data Distance Skenario Pertama

| | *Sit* | *Hee* | *Gel* | *Lan* |
|---|---|---|---|---|
| **Sit** | 0 | 19 | 5 | 18 |
| **Hee** | 24 | 0 | 18 | 6 |
| **Gel** | 12 | 17 | 0 | 20 |
| **Lan** | 31 | 16 | 31 | 0 |

Keterangan:
Sit = Sittard
Hee = Heerlen
Gel = Geleen
Lan = Landgraaf

Solusi optimal didapatkan tanpa adanya percabangan dan proses *column generation* dalam waktu 0,015 detik.

**Tabel 4.** Data Job Skenario Kedua

| *IdJob* | *s* | *p* | *w* | *loc* |
|---|---|---|---|---|
| 100006062 | 24300 | 34200 | 25 | Sittard |
| 100006191 | 25200 | 25200 | 60 | Landgraaf |
| 100006503 | 25200 | 32400 | 30 | Heerlen |
| 100006629 | 26100 | 9900 | 30 | Sittard |
| 100006674 | 26100 | 31500 | 30 | Gelleen |

Untuk skenario kedua rangkaian *job* didapatkan dari 5 *job* yang diambil dari Data_030697, seperti yang ditunjukkan pada tabel 4. Data crane ditunjukkan pada tabel 2 dan datadistance pada tabel 5.

Solusi optimal didapatkan tanpa adanya percabangan tetapi dilakukan proses *column generation* dalam waktu 0,032 detik.

**Tabel 5.** Data Distance Skenario Kedua

| | *Sit* | *Maa* | *Hee* | *Her* | *Vaa* | *Lan* |
|---|---|---|---|---|---|---|
| **Sit** | 0 | 19 | 19 | 26 | 33 | 18 |
| **Maa** | 23 | 0 | 21 | 43 | 25 | 27 |
| **Hee** | 24 | 15 | 0 | 40 | 16 | 6 |
| **Her** | 24 | 29 | 56 | 0 | 54 | 37 |
| **Vaa** | 35 | 27 | 18 | 67 | 0 | 21 |
| **Lan** | 31 | 28 | 16 | 51 | 28 | 0 |

Keterangan:
Sit = Sittard
Maa = Maastricht
Hee = Heerlen
Her = Herten
Vaa = Vaals
Lan = Landgraaf

Untuk skenario ketiga rangkaian *job* didapatkan dari contoh permasalahan, seperti yang ditunjukkan pada tabel .6.

**Tabel 6.** Data *Job* Skenario Ketiga

| *IdJob* | *s* | *p* | *w* | *loc* |
|---|---|---|---|---|
| 1001 | 20 | 0 | 1 | job1 |
| 1002 | 40 | 0 | 0 | job2 |
| 1003 | 60 | 0 | 0 | job3 |

Dan data *crane* untuk skenario ketiga ditunjukkan pada tabel 7.

**Tabel 7.** Data Crane Skenario Ketiga

| $T$ | $m_t$ | $cap_t$ | $r_t$ | $speed_t$ | $q_t$ |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 1 | 1 |
| 2 | 1 | 1 | 3 | 1 | 1 |

Data *distance* untuk skenario ketiga ditunjukkan pada tabel 8.

Solusi optimal skenario ketiga didapatkan melalui proses percabangan dan proses column generation dalam waktu 0,109 detik.

Berdasarkan dari uji coba yang dilakukan untuk beberapa kasus, banyaknya *job* mempengaruhi penambahan waktu untuk menentukan *path*, tetapi tidak banyak mempengaruhi waktu yang dibutuhkan untuk penyelesaian dengan algoritma branch and price. Karena terdapat kasus rangkaian banyak *job*, namun tanpa dilakukan percabangan solusi optimal telah didapatkan pada solusi *linear programming*, dimana solusi yang dihasilkan merupakan solusi integral dan optimal karena tidak terdapat variabel $x_i$ yang memiliki *reduced costs* bernilai negatif seperti yang diberikan pada skenario pertama. Sedangkan pada skenario keduakombinasi *job* tidak membutuhkan adanya proses percabangan namun proses column generation dilakukan karena terdapat variabel $x_i$ yang memiliki reduced cost bernilai negatif. Sedangkan pada skenario ketiga dibutuhkan adanya percabangan dan column generation pada proses penyelesaian dengan *branch and price.*

**Tabel 8.** Data Distance Skenario Ketiga

| | ***depot*** | ***job1*** | ***job2*** | ***job3*** |
|---|---|---|---|---|
| **depot** | 0 | 16 | 10 | 9 |
| **job1** | 16 | 0 | 12 | 12 |
| **job2** | 10 | 12 | 0 | 16 |
| **job3** | 9 | 12 | 16 | 0 |

Dari uji coba dengan berbagai rangkaian data *input* menghasilkan peningkatan dibandingkan dengan solusi yang diberikan *planner* seperti yang ditunjukkan pada tabel 9.

## 6. SIMPULAN DAN SARAN

Dari beberapa proses yang dilakukan dalam aplikasi yang dibuat, maka dapat diambil kesimpulan sebagai berikut:

a. Aplikasi ini dapat diterapkan untuk berbagai kombinasi rangkaian *job* pada algoritma *branch and price*
b. Dengan aplikasi yang dibuat, didapatkan rute penugasan untuk *crane* tipe tertentu.
c. Dari hasil uji coba aplikasi pada rangkaian *job* yang didapatkan dari data NRL, dibuktikan adanya peningkatan keoptimalan solusi dibandingkan dengan solusi yang diberikan oleh *planner*.

Aplikasi ini digunakan untuk perusahaan persewaan crane yang hanya memiliki satu depot. Pengembangan dapat dilakukan dengan permasalahan perusahaan persewaan *crane* yang memiliki lebih dari satu *depot.*

## DAFTAR PUSTAKA

[1]. Faneyte, B. C., Diego; Spieksma, C. R., Frits; Woeginger, J., Gerhard. (2001). *A Branch-and-Price Algorithm for a Hierarchical Crew Scheduling Problem*.
[2]. Taylor III, W., Bernard; (1999). *Introduction to Management Science*, Sixth Edition, Prentice Hall, Virginia Polytechnic Institute and State University.
[3]. Taha, A., Hamdy. (2003). *Operations Research: An Introduction*, Seventh Edition, Prentice Hall, University of Arkansas.
[4]. Branhart, Cynthia; Johnson, L., Ellis; Nemhauser, L., George; Savelsbergh, W.P., Martin; Vance, H., Pamela. (1996). *Branch and Price: Column Generation for Solving Huge Integer Programs*.

**Tabel 9.** Rangkaian Data Input

| ***Rangkaian (NRL)*** | ***Jumlah job (NRL)*** | ***Solusi Planner (NRL)*** | ***Solusi IP*** | ***Pening-katan*** | ***Level Pencarian Tree*** | ***Banyak iterasi*** | ***Waktu (detik)*** |
|---|---|---|---|---|---|---|---|
| Data_970421 | 48 | 136024 | 111938 | 17.87% | 3 | 3 | 4.95 |
| Data_970424 | 40 | 134264 | 85503.6 | 36.32% | 0 | 0 | 0.66 |
| Data_970425 | 38 | 118416 | 72121.1 | 39.09% | 2 | 2 | 2.25 |
| Data_970515 | 30 | 121382 | 64742.3 | 46.66% | 3 | 3 | 0.98 |
| Data_970516 | 40 | 126667 | 67005.3 | 47.10% | 3 | 3 | 2.82 |
| Data_970603 | 40 | 95649.5 | 68381.5 | 28.51% | 0 | 0 | 1.51 |
| Data_970604 | 39 | 99653.4 | 81367.3 | 18.35% | 0 | 0 | 0.69 |
| Data_970605 | 31 | 120173 | 88892.9 | 26.3% | 0 | 0 | 0.41 |
| Data_970609 | 41 | 130928 | 116113 | 11.31% | 5 | 5 | 5.51 |
| Data_970610 | 39 | 142107 | 99337 | 30.01% | 0 | 0 | 0.72 |
| Data_970611 | 47 | 138518 | 101907 | 26.43% | 1 | 1 | 3.1 |