Minimasi Jarak Tempuh *Order Picking* pada Gudang dengan Karakteristik *Two-Cross Aisle Layout*

Minimizing Order Picking Travel Distance on Two-Cross Aisle Warehouse Layout

Eka Kurnia Asih Pakpahan, Roland Silitonga, Anggoro Prasetyo Utomo
Program Studi Teknik Industri, Institut Teknologi Harapan Bangsa
E-mail: eka@ithb.ac.id, roland@ithb.ac.id, anggoro@ithb.ac.id

Abstrak

Kegiatan order picking mengkonsumsi hampir 60% dari total ongkos gudang. Faktor utama yang membuat hal ini terjadi adalah jarak perpindahan saat order picking dilakukan. Penelitian ini mengambil satu kasus khusus dimana tata letak gudang merupakan two-cross aisle. Untuk menemukan rute order picking dengan jarak perpindahan terpendek, lima algoritma heuristik diterapkan dan dievaluasi berdasarkan jarak perpindahan yang diberikannya. Hasil dari penelitian ini menunjukkan peningkatan efisiensi gudang melalui penurunan jarak perpindahan selama melakukan order picking.

Kata kunci: algoritma heuristik, order picking, minimasi jarak perpindahan,

Abstract

Order picking activities consumes almost 60% of the total warehousing cost. The primary factor affecting this large share is the pick-up travel distance. We take a specific case where the layout of a warehouse is a two-cross aisle and developed five heuristic algorithms to determine pick-up routes. The associated travel distance generated by each algorithm were then logged and compared to determine which algorithm generates the shortest pick-up travel distance. By doing this we managed to improve warehouse efficiency by decreasing the pick-up travel distance.

Keywords: heuristic algorithm, order picking, minimizing travel distance.

1. Pendahuluan

Kegiatan pergudangan memainkan peranan penting dalam memastikan efisiensi sebuah sistem rantai pasok, karena pergerakan barang antar rantai, membutuhkan kecepatan informasi persediaan berbagai produk yang disimpan di gudang (Lambert dan Stock, 2001), juga kecepatan keluar masuk barangnya. Pengelolaan gudang umumnya mengatur pelaksanaan lima kegiatan berikut: penerimaan (*receiving*), pengantaran barang untuk disimpan (*put-away*), penyimpanan (*storage*), pengambilan barang untuk dikeluarkan (*order picking*), dan pengiriman (*shipping*) (Bartholdi dan Hackman, 2011). Di antara lima kegiatan ini, *order picking* adalah kegiatan paling penting dari sisi efektifitas dan biaya (Habazin, Glasnovic dan Bajor, 2017). Kegiatan ini menghabiskan hampir 60% dari total biaya pergudangan akibat jarak perpindahan selama kegiatan berlangsung. Melihat signifikansinya peningkatan efisiensi pada kegiatan ini perlu diberi perhatian khusus.

Aktivitas *order picking* yaitu suatu aktivitas pengambilan produk dari lokasi penyimpanan berdasarkan pesanan pelanggan (Dukic dan Oluic, 2007). Jarak yang ditempuh saat *order picking* ditentukan oleh keputusan rute, yang sangat penting dalam mengurangi waktu berjalan (De Koster dan Roodbergen, 2007). Keputusan ini dapat diperoleh melalui metode analitis serta metode heuristik (Roodbergen, 2001). Banyak penelitian menemukan bahwa keputusan *routing* sulit diselesaikan karena bersifat *non polynomial* (*NP) hard*, jikapun dapat diselesaikan, waktu pencarian solusi sangatlah lama. Untuk alasan ini, metode heuristik dikembangkan dan digunakan untuk mendapatkan keputusan *routing* (Hall, 1993). Penelitian (Oktarina dan Wini, 2008) secara khusus berfokus pada permasalahan penentuan *routing* untuk *order picking*, mereka merancang algoritma heuristik dan mengimplementasikannya pada gudang berkarakteristik *three-cross aisle*.

Algoritma heuristik yang dikembangkan adalah: (1) *s-shaped*, (2) *largest gap*, (3) *return-to-home*, (4) *middle point* dan (5) *aisle-by-aisle.*

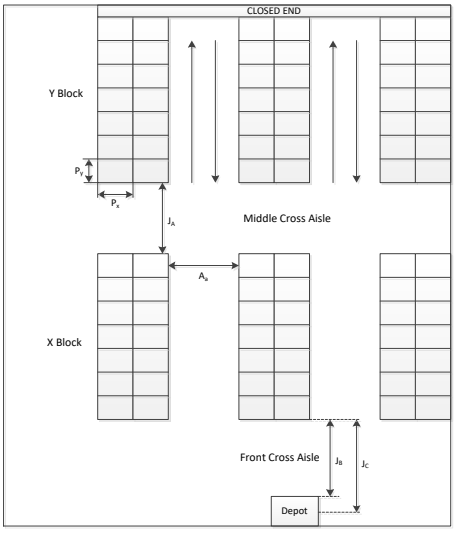
Penelitian ini mengambil pendekatan yang sama dengan penelitian (Oktarina dan Wini, 2008) dan menerapkannya pada gudang berkarakteristik *two-cross aisle*. Penelitian ini melakukan penyesuaian terhadap lima algoritma heuristik dan mengimplementasikannya pada gudang dengan *two-cross aisle*. Kinerja masing-masing algoritma (dalam ukuran jarak perpindahan) kemudian dicatat dan dibandingkan satu sama lain.

Tulisan ini kami sajikan dalam empat bagian. Bagian 1 berisi pendahuluan dan pengenalan masalah; bagian 2 membahas deskripsi sistem dan desain algoritma; bagian 3 menjelaskan implementasi algoritma serta hasilnya; bagian 4 menyajikan kesimpulan dan bagian 5 menyajikan daftar pustaka yang digunakan dalam penelitian ini.

2. Deskripsi Sistem dan Rancangan Algoritma

2.1 Deskripsi Sistem

Tata letak gudang merupakan faktor penting yang harus dipertimbangkan ketika mengambil keputusan *routing*. Pada penelitian ini, kami mengambil kasus di mana tata letak gudang adalah *two-cross aisle*. Tata letak tersebut diilustrasikan pada Gambar.1.



Gambar 1. Gudang dengan Tata Letak *Two-Cross-Aisle*

Implikasi dari *two-cross aisle* adalah bahwa pada blok gang tertentu, salah satu ujung lorongnya tertutup. Dengan demikian, peralatan pemindahan material harus berbelok 180 derajat untuk keluar dari lorong.

2.2 Desain Algoritma

Pada tahap ini, dikembangkan 5 algoritma heuristik untuk penentuan keputusan *routing*. Algoritma tersebut adalah *s-shaped*, *largest gap*, *return-to-home*, *middle point* dan *aisle-by-aisle.* Jarak yang ditempuh pada setiap rute dihitung menggunakan metode *rectilinier*. Penjelasan untuk setiap algoritma kami berikan sebagai berikut:

2.2.1 Algoritma *S-Shaped*

Berdasarkan algoritma heuristik s-*shaped* rute *order picking* ditentukan melalui langkah-langkah berikut:

1. Rute dimulai dari lokasi depot menuju lorong ujung kanan atau kiri berdasarkan jarak terdekat yang memiliki setidaknya satu lokasi pengambilan.
2. Menentukan blok terdekat (blok X) dan terjauh (blok Y) yang dipisahkan oleh *middle-cross aisle*.

3. *Picker* melintasi lorong blok X hingga sampai pada titik-titik lokasi pengambilan barang dan melakukan pengambilan barang.
4. *Picker* berjalan terus sepanjang lorong hingga mencapai *middle-cross aisle*. Terdapat 2 kemungkinan yang dapat terjadi: (a) Terdapat barang yang harus diambil di lorong yang bersesuaian dengan lorong blok X di blok Y, sehingga *picker* menyeberang melalui *middle-cross aisle* memasuki blok Y ke lorong yang bersesuaian tersebut. (b) Tidak terdapat barang yang harus diambil di lorong yang bersesuaian dengan lorong blok X di blok Y, sehingga *picker* tetap di blok X dan berpindah lorong di blok tersebut melalui *middle-cross aisle*.
5. *Picker* berpindah lorong ke lorong terdekat di blok Y. Mendatangi lokasi pengambilan tersebut melalui *middle cross aisle* dan berbalik arah setelah mencapai ujung lorong. Ulangi langkah ini hingga seluruh barang yang harus diambil dari blok Y terpenuhi.
6. Pada lorong paling kiri di blok Y, *picker* kemudian dapat berpindah ke blok X dengan melintasi *middle cross aisle*, sesampainya di blok X, picker dapat menyusuri lorong demi lorong di blok X melalui *middle cross aisle* dan keluar dari *front cross aisle* atau sebaliknya hingga seluruh lokasi pengambilan di blok X selesai.
7. Kembali ke depot.

2.2.2 Algoritma *Return-to-home*

Berdasarkan algoritma heuristik *return-to-home,* rute *order picking* ditentukan melalui langkah-langkah berikut:

1. Rute dimulai dari lokasi depot menuju lorong ujung kanan atau kiri berdasarkan jarak terdekat yang memiliki setidaknya satu lokasi pengambilan.
2. Menentukan blok terdekat (blok X) dan terjauh (blok Y) yang dipisahkan oleh *middle-cross aisle*.
3. *Picker* melintasi lorong blok X hingga sampai pada titik-titik lokasi pengambilan barang dan melakukan pengambilan barang.
4. *Picker* berjalan terus sepanjang lorong blok X hingga mencapai *middle cross aisle*. Di titik ini, terdapat 2 kemungkinan yang muncul: (a) *Picker* memasuki blok Y apabila masih terdapat lokasi pengambilan di lorong tersebut dan berbalik arah kembali lalu berpindah ke lorong sebelahnya melalui *middle-cross aisle*. (b) Langsung berpindah ke lorong sebelah di blok X melalui *middle-cross aisle* bila tidak ada lokasi pengambilan di lorong terdekat di blok Y.
5. *Picker* berpindah lorong ke lorong terdekat di blok Y. Mendatangi lokasi pengambilan tersebut lalu berbalik arah menuju *middle-cross aisle* dan berpindah lorong di blok Y. Ulangi langkah ini hingga seluruh lokasi pengambilan di blok Y terpenuhi.
6. *Picker* sekarang berada pada lorong ujung kiri yang memiliki lokasi pengambilan, dan akan berpindah ke blok X.
7. *Picker* memasuki lorong di blok X yang memiliki lokasi pengambilan. Setelah sampai di titik tersebut, berbalik arah dan berpindah ke lorong sebelahnya melalui *middle-cross aisle*.
8. *Picker* berpindah lorong demi lorong yang mempunyai lokasi pengambilan di blok X. melalui *middle-cross aisle*. Ulangi langkah ini hingga seluruh lokasi pengambilan di blok X terambil.
9. Kembali ke depot.

2.2.3 Algoritma *Mid-point*

Berdasarkan algoritma heuristik *mid-point,* rute *order picking* ditentukan melalui langkah-langkah berikut:

1. Rute dimulai dari lokasi depot menuju lorong ujung kanan atau kiri berdasarkan jarak terdekat yang memiliki setidaknya satu lokasi pengambilan.
2. Menentukan blok terdekat (blok X) dan terjauh (blok Y) yang dipisahkan oleh *middle-cross aisle*.
3. *Picker* melintasi lorong blok X hingga sampai pada titik lokasi pengambilan.
4. *Picker* berjalan terus sepanjang lorong hingga mencapai *middle-cross aisle*.

5. Untuk blok Y, *picker* mengunjungi lokasi pengambilan melalui *middle-cross aisle*. Ulangi langkah tersebut hingga seluruh lokasi pengambilan blok Y terpenuhi.
6. Untuk blok X, *picker* mengunjungi lokasi pengambilan yang berada pada setengah bagian belakang blok melalui *middle-cross aisle* dan mengunjungi lokasi pengambilan yang berada pada setengah bagian depan blok melalui *front-cross aisle*. Ulangi langkah tersebut hingga seluruh lokasi pengambilan di blok X terpenuhi.
7. Kembali ke depot.

2.2.4 Algoritma *Largest Gap*

Berdasarkan algoritma heuristik *largest gap,* rute *order picking* ditentukan melalui langkah-langkah berikut:

1. Rute dimulai dari lokasi depot menuju lorong ujung kanan atau kiri berdasarkan jarak terdekat yang memiliki setidaknya satu lokasi pengambilan.
2. Menentukan blok terdekat (blok X) dan terjauh (blok Y) yang dipisahkan oleh *middle-cross aisle*.
3. *Picker* melintasi lorong blok X hingga sampai pada titik lokasi pengambilan.
4. *Picker* berjalan terus sepanjang lorong hingga mencapai *middle-cross aisle*. Di titik ini, terdapat dua kemungkinan: (a) *Picker* memasuki blok Y apabila terdapat lokasi pengambilan di lorong tersebut, berbalik arah lalu berpindah ke lorong sebelahnya melalui *middle-cross aisle*. (b) *Picker* tetap di blok X dan berpindah ke lorong sebelahnya melalui *middle-cross aisle*.
5. Saat *picker* berada di blok Y, perpindahan antar lorong selalu dilakukan melalui *middle-cross aisle.*
6. Saat *picker* berada di blok X, *picker* harus selalu membandingkan jarak perpindahan antar lokasi pengambilan pada lorong (gap) blok X dengan mempertimbangkan dua akses jalan, yaitu *middle-cross aisle* dan *front-cross aisle*. Terdapat tiga kemungkinan yang dapat terjadi: (a) Nilai *largest gap* berada pada *middle-cross aisle,* dengan begitu *picker* harus mengakses lorong berikutnya melalui *front-cross aisle*, (b) Nilai *largest gap* berada pada *front-cross aisle,* dengan begitu *picker* harus mengakses lorong berikutnya melalui *middle-cross aisle,* (c) Gap keduanya sama, sehingga *picker* dapat memilih mengakses lorong berikutnya baik melalui *front-cross aisle* ataupun *middle-cross aisle.*
7. Kembali ke depot.

2.2.5 Algoritma *Aisle-by-aisle*

Rute *order picking* yang dihasilkan algoritma ini hanya akan melewati setiap lorong satu kali. Programa dinamis digunakan untuk menentukan rute terbaik untuk berpindah dari satu lorong pengambilan ke lorong pengambilan yang lain dengan mempertimbangkan jarak terpendek.

Rute *order picking* dimulai dari depot menuju lorong terdekat. Di lorong tersebut, *picker* mengambil seluruh barang yang terdapat dalam daftar pengambilan dan keluar melalui *front-cross aisle* ataupun *middle-cross aisle* ke lorong pengambilan ke-dua. Pengambilan keputusan didasari jarak tempuh terpendek. Proses ini terus berulang sampai seluruh lokasi pengambilan dikunjungi. Kemudian order picker kembali ke titik awal.

Dalam algoritma *aisle-by-aisle* ini dikenal istilah transisi untuk menggambarkan cara *picker* untuk melakukan perpindahan dan pengambilan barang. Transisi menggunakan perhitungan dengan program dinamis. Roodbergen (2001) mengelompokkan transisi ini ke dalam dua kelompok, yaitu: (a) Transisi untuk berpindah dari satu lorong ke lorong lainnya dan (b) Transisi untuk mengambil barang di lorong.

Terdapat satu transisi yang dapat digunakan untuk berpindah dari lorong satu ke lorong lain (t_a), yaitu: melewati *front-cross aisle* (*node a*) atau *middle-cross aisle* (*node b*). Sementara untuk melakukan pengambilan *item* pada lorong terdapat tiga transisi yang dapat digunakan, yaitu: memasuki dan meninggalkan lorong dari *middle-cross aisle* atau *node b* (t_1); memasuki dan

meninggalkan lorong dari *front-cross aisle* atau *node a* (t_2); memasuki lorong dari *front-cross aisle* atau *node a* dan meninggalkan lorong dari *middle-cross aisle* atau *node b* atau sebaliknya (t_3).
Misalkan ditentukan notasi-notasi berikut:

t_1	:	Jarak tempuh jika memasuki dan meninggalkan lorong melalui *middle-cross aisle*
t_2	:	Jarak tempuh jika memasuki dan meninggalkan lorong melalui *front-cross aisle*
t_3	:	Jarak yang ditempuh jika memasuki lorong melalui *front-cross aisle* dan meninggalkan lorong melalui *middle-cross aisle* atau sebaliknya.
t_a	:	Jarak yang ditempuh jika melewati *front-cross aisle* atau *middle-cross aisle*
L_a^j	:	Posisi akhir picker di *front-cross aisle* untuk lorong j
L_b^j	:	Posisi akhir picker di *middle-cross aisle* untuk lorong j

Penentuan rute dilakukan dengan langkah-langkah berikut:
Langkah 1:
L_a^j: rute yang dimulai dari node a dan berakhir di node a untuk lorong j dan mengandung transisi t_2
L_b^j: rute yang dimulai dari node a dan berakhir di node b untuk lorong j dan mengandung transisi t_3

Langkah 2:
Lorong selanjutnya ditentukan melalui model matematika berikut:

L_a^{j+1} = min (1)
- $L_a^j + t_a + t_2$
- $L_b^j + t_a + t_3$

L_b^{j+1} = min
- $L_b^j + t_a + t_1$
- $L_a^j + t_a + t_3$

Langkah 3:
Lorong terakhir ditentukan melalui model matematika berikut:

L_a^{j+x} = min (2)
- $L_b^{j+x} + t_a + t_3$
- $L_a^{j+x} + t_a + t_2$

3. Implementasi Algoritma dan Perbandingan Kinerjanya

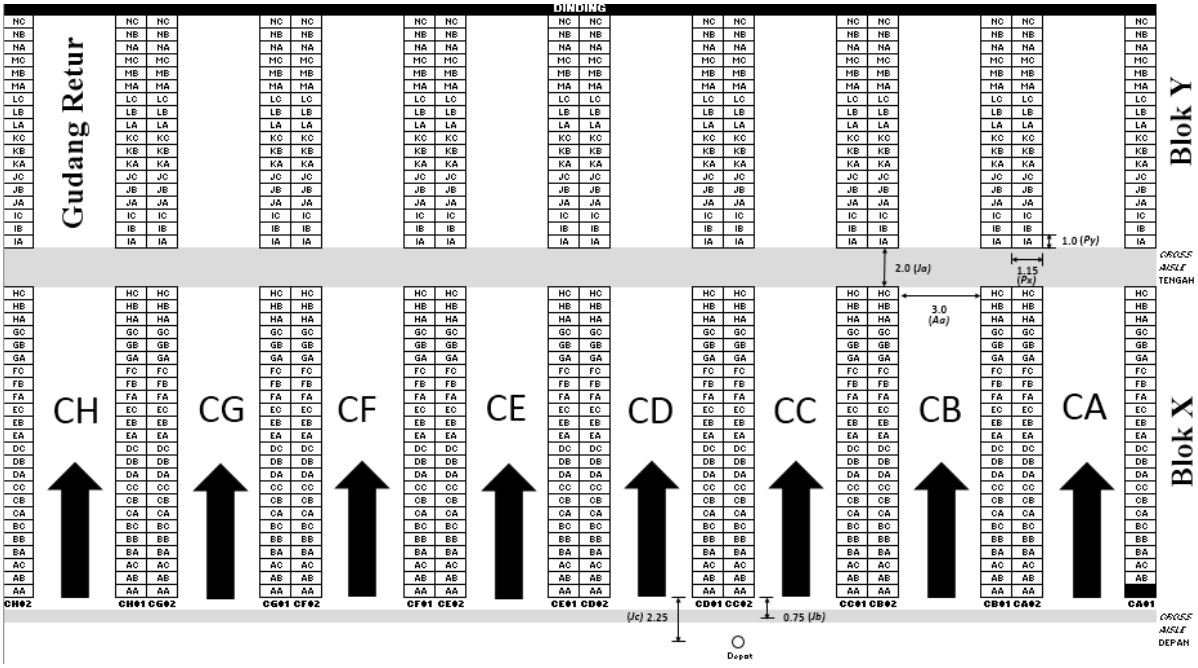
Kelima algoritma heuristik yang sudah dijelaskan pada sub-bagian sebelumnya kemudian diimplementasikan pada gudang dengan karakteristik seperti pada Tabel 1.

Tabel 1. Karakteristik Gudang Kajian

Deskripsi		Dimensi (meter)
Lokasi Penyimpanan	Panjang rak (P_x)	1.15
	Lebar rak (P_y)	1.00
Jarak	Blok X dan blok Y (J_a)	2.00
	Blok X dengan *cross aisle* depan (J_b)	0.75
	Blok X dengan lokasi depot (J_c)	2.25
Lorong	Lebar lorong (A_a)	3.00

Gudang yang akan dikaji berbentuk persegi panjang dan dipisahkan menjadi dua buah blok, yang terdekat dengan depot dinamakan blok X dan yang terjauh dinamakan blok Y. Gudang tersebut memiliki 8 buah lorong penyimpanan barang dan 636 titik lokasi pengambilan. Titik lokasi pengambilan diberi kode secara berurutan mulai dari lorong terkanan: CA-CB hingga CH. Dalam sebuah lorong, tersusun rak yang berada di sebelah kanan dan kiri lorong. Bagian kanan dinotasikan dengan angka 01 dan bagian kiri dinotasikan dengan angka 02. Setiap rak memiliki 3

titik pengambilan dan penamaannya secara berurutan adalah AA-AB-AC lalu BA-BB-BC, dan seterusnya. Saat pengambilan barang, *picker* berada pada posisi tengah dari rak.



Gambar 2. Tata Letak Gudang Kajian

Task order yang diberikan kepada *picker* berisi dua puluh jenis barang. Kapasitas *pallet* diasumsikan tak terhingga sehingga memungkinkan *picker* mengambil seluruh barang hingga *task order* terselesaikan. Jarak tempuh *picker* dihitung dengan metode *rectilinear* yaitu menurut perpindahan horisontal dan vertikal.

Kelima algoritma kemudian diuji kinerjanya dalam empat kali percobaan. Empat kali percobaan merupakan jumlah sampel yang minimum bagi pengujian ini. Perbedaan antar percobaan terletak pada isi *task order* yang harus dilakukan *picker*. Setiap isi *task order* dibangkitkan secara random dengan bantuan MS. Excel. Mengingat jumlah *barang* yang harus diambil per *task order* umumnya adalah 20, maka untuk setiap percobaan dibangkitkan 20 angka random antara 1-636 (menggambarkan lokasi pengambilan barang), spesifikasi isi *task order* untuk setiap percobaan diberikan pada Tabel 2 hingga Tabel 5.

Tabel 2. Isi *Task Order* pada Skenario 1

562 (CG.02.FA)	358 (CE.01.HA)	407 (CE.02.JB)	140 (CB.02.EB)	555 (CG.02.CC)
466 (CF.02.BA)	237 (CC.02.IC)	441 (CF.01.GC)	265 (CD.01.EA)	414 (CE.02.LC)
385 (CE.02.CA)	28 (CA.01.JA)	106 (CB.01.HA)	305 (CD.02.DB)	271 (CD.01.GA)
479 (CF.02.FB)	230 (CC.02.GB)	85 (CB.01.AA)	48 (CA.02.BC)	165 (CB.02.MC)

Tabel 3. Isi *Task Order* pada Percobaan 2

553 (CG.02.CA)	251 (CC.02.NB)	37 (CA.01.MA)	326 (CD.02.KB)	407 (CE.02.JB)
213 (CC.02.AC)	191 (CC.01.HB)	438 (CF.01.FC)	30 (CA.01.JC)	452 (CF.01.KB)
279 (CD.01.IC)	159 (CB.02.KC)	361 (CE.01.IA)	118 (CB.01.LA)	239 (CC.02.JB)
441 (CF.01.GC)	274 (CD.01.HA)	501 (CF.02.MC)	55 (CA.02.EA)	573 (CG.02.IC)

Tabel 4. Isi *Task Order* pada Percobaan 3

139 (CB.02.EA)	298 (CD.02.BA)	274 (CD.01.HA)	54 (CA.02.DC)	181 (CC.01.EA)
561 (CG.02.EC)	72 (CA.02.JC)	556 (CG.02.DA)	95 (CB.01.DB)	409 (CE.02.KA)
276 (CD.01.HC)	369 (CE.01.KC)	616 (CH.02.BA)	566 CG.02.GB)	271 (CD.01.GA)
4 (CA.01.BA)	424 (CF.01.BA)	546 (CG.01.NC)	502 (CF.02.NA)	446 (CF.01.IB)

Tabel 5. Isi *Task Order* pada Percobaan 4

226 (CC.02.FA)	374 (CE.01.MB)	578 (CG.02.KB)	138 (CB.02.DC)	447 (CF.01.IC)
485 (CF.02.HB)	83 (CA.02.NB)	365 (CE.01.JB)	314 (CD.02.GB)	155 (CB.02.JB)
488 (CF.02.IB)	22 (CA.01.HA)	596 (CH.01.CB)	358 (CE.01.HA)	106 (CB.01.HA)
158 (CB.02.KB)	341 (CE.01.BB)	114 (CB.01.JC)	235 (CC.02.IA)	468 (CF.02.BC)

Kinerja masing-masing algoritma di setiap percobaan diberikan di Tabel 6.

Tabel 6. Perbandingan Kinerja Algoritma pada Empat Percobaan

Algoritma Heuristik	Jarak Tempuh Percobaan 1 (meter)	Jarak Tempuh Percobaan 2 (meter)	Jarak Tempuh Percobaan 3 (meter)	Jarak Tempuh Percobaan 4 (meter)	Rataan Jarak Tempuh (meter)
S-shape	353.70	365.60	375.30	408.30	375.725
Return	356.20	324.60	428.80	405.80	378.850
Mid-point	391.95	342.85	390.55	383.05	377.100
Largest gap	**324.70**	**312.10**	**363.30**	**345.80**	**336.475**
Aisle by aisle	347.95	337.45	376.05	364.05	356.375

Berdasarkan hasil perhitungan dari empat percobaan, ditemukan bahwa algoritma heuristik yang selalu menghasilkan jarak tempuh terpendek di gudang kajian adalah algoritma *largest gap*. Jika dibandingkan dengan metode yang saat ini digunakan perusahaan, yang secara rata-rata menghasilkan jarak tempuh sebesar 386.975 maka implementasi algoritma *largest gap* mengakibatkan penghematan jarak tempuh rata-rata sebesar 50.5 meter. Jika aktivitas *picking* ini dilakukan sebanyak 8-12 kali per hari, maka total jarak tempuh yang dapat dihemat per hari berkisar antara 404 – 606 meter. Jika menggunakan riset Bartholdi dan Hackman (2011), yang menyatakan bahwa *order picking* berperan pada 70% waktu dan 55% biaya, maka penghematan yang dilakukan ada di kisaran 7-8%, meskipun tidak diteliti secara detil pada studi ini.

Tata letak gudang di perusahaan kajian memiliki dua blok yaitu blok X dan blok Y. Blok X memiliki 24 rak penyimpanan dengan total panjang sebesar 24 meter. Sedangkan blok Y memiliki 18 rak penyimpanan dengan total panjang sebesar 18 meter. Algoritma *largest gap* cenderung baik diterapkan pada gudang yang memiliki blok relatif panjang. *Largest gap* akan menentukan rute berdasarkan jarak terjauh lokasi pengambilan dalam sebuah lorong. Semakin panjang blok atau lorong yang harus dilalui *picker* maka semakin jelas gap yang terlihat antar lokasi pengambilan dalam sebuah lorong. Apabila lorong yang dilalui oleh *picker* tidak terlalu panjang, maka gap yang dihasilkan tidak signifikan sehingga penerapan metode *largest gap* menjadi kurang baik.

4. Kesimpulan

Beberapa kesimpulan yang dapat ditarik dari penelitian ini adalah:

1. Algoritma heuristik untuk gudang dengan karakteristik *two-cross aisles* telah berhasil dikembangkan dari algoritma heuristik untuk gudang dengan karakteristik *three-cross aisles.*
2. Algoritma heuristik yang dikembangkan telah diimplementasikan pada sebuah gudang dengan layout *two-cross aisles*. Hasil implementasi menunjukkan bahwa penerapan algoritma *largest gap* memberikan jarak perpindahan paling minimum. Jika dibandingkan dengan kebijakan penentuan rute yang digunakan saat ini, maka terdapat penghematan jarak dari sebelumnya rata-rata sejauh 386.975 meter, menjadi rata-rata sejauh 336.475 meter atau memiliki selisih rata-rata sejauh 50.5 meter. Jika diasumsikan penghematan jarak berbanding lurus dengan ongkos pemindahan material maka diperoleh penghematan dengan persentase yang sama.
3. Perlu dikaji konsistensi algoritma untuk macam-macam ukuran blok di gudang, mengingat karakteristik gudang kajian pada penelitian ini memiliki ukuran blok yang relatif panjang. Pada Gudang yang ukuran blok tidak terlalu panjang, besar *gap* saat penentuan *largest gap* dapat saja tidak signifikan, sehingga algoritma ini akan sama baiknya dengan algoritma lain.

5. Daftar Pustaka

Bartholdi, J.J dan Hackman, S.T, (2008), *Warehouse and Distribution Science*, Georgia Institute of Technology.

Bartholdi, J.J dan Hackman, S.T, (2011), *Warehouse and Distribution Science*, *The Supply Chain and Logistic Institute*. Atlanta: School of Industrial and System Engineering.

De Koster, R dan Roodbergen, K.J, (2007), Design and Control of Warehouse Order Picking: A Literature Review, *European Journal of Operation Research*, 182, pp. 481-501.

Dukic, G. dan Oluic, C, (2007), Order Picking Methods: Improving Order – Picking Efficiency, *International Journal of Logistics System and Management*, 3, pp. 451 – 460.

Habazin, J., Glasnovic, A., dan Bajor, I., (2017), Order Picking Process in Warehouse: Case Study of Dairy Industry in Croatia, *Promet Traffic & Transportation*, 29 (1), pp. 57-65.

Hall, R.W, (1993), Distance approximations for routing manual pickers in a warehouse. *IIE Transactions*, 25(4), 76-87.

Lambert, D.M dan Stock, J.R, (2001), Strategic *Logistic Management*, Edisi Keempat, New York.

Oktarina, R. dan Wini, (2008), *Penentuan Rute Order Picking dengan Menggunakan Metode Heuristic (Studi Kasus: PT"X")*. Tugas Akhir Teknik Industri, Universitas Widyatama.

Roodbergen K.J, (2001), Layout and Routing Methods for Warehouses, *Ph.D. Series Research in Management.*