COMPLETE
Journal of Computer, Electronic, and Telecommunication
ISSN 2723-4371, E-ISSN 2723-5912
doi.org/10.52435/complete.v1i2.74

Analisis Kualitas Model Proses dalam Implementasi Process Mining : Literature Review

Afina Lina Nurlaili [1]*, Agung Mustika Rizki[2]

[1] Informatika, UPN Veteran Jawa Timur; afina.lina.if@upnjatim.ac.id
[2] Informatika, UPN Veteran Jawa Timur; agung.mustika.if@upnjatim.ac.id
* Correspondence: afina.lina.if@upnjatim.ac.id;

Abstrak: Proses bisnis memiliki peran dalam mengatur pola kerja suatu organisasi. Kerumitan dalam membuat proses bisnis menjadi tantangan tersendiri, terutama jika pekerjaan yang dicakup relatif banyak. *Process mining* menjadi salah satu solusi yang paling handal dengan secara otomatis dapat menemukan model proses, menganalisis kualitas model proses hingga dapat meningkatkan kualitas model proses dari data *event log* yang tersimpan pada sistem informasi. Penelitian ini mengevaluasi tiga algoritma process mining, yaitu Alpha, Alpha++, dan Heuristic Miner dalam hal pembuatan model proses dan kualitasnya. Berdasarkan evaluasi yang dilakukan, algoritma Alpha, Alpha++, dan Heuristic Miner memiliki kecocokan tersendiri untuk kasusnya masing-masing.

Kata Kunci: Alpha; Alpha ++; Event Log Process mining; Heuristic Miner.

1. Pendahuluan

Proses bisnis berfungsi mengatur pola kerja suatu organisasi. Namun, membuat proses bisnis yang baik merupakan pekerjaan yang sulit dan rumit. Dalam menangani permasalahan tersebut *process mining* menjadi solusi. [1]. *Process mining* mampu memodelkan proses bisnis secara otomatis, dengan menggunakan data yang direkam pada system informasi[2]. Secara umum, *process mining* adalah disiplin ilmu baru yang menyediakan teknik yang dapat digunakan untuk menemukan, mengukur kualitas dan meningkatkan proses bisnis [3].

Data yang digunakan sebagai masukan *process mining* disebut dengan *event log*, yaitu kumpulan aktivitas yang direkam secara berurutan yang terjadi selama pelaksanaan proses tertentu [4]. Setiap *event* mengacu pada aktivitas (yaitu langkah yang didefinisikan dalam proses bisnis) dan terhubung dengan *case* tertentu (mis.sebuah contoh proses bisnis) [5]. Urutan aktivitas yang termasuk dalam satu *case* disebut dengan *trace*. Event log juga dapat menyimpan informasi tambahan, seperti *resource* atau sumber daya (yaitu orang atau perangkat) yang menjalankan atau memulai suatu kegiatan, waktu suatu peristiwa atau atribut lain yang terkait dengan suatu proses bisnis [6].

Terdapat tiga jenis *process mining*. Jenis pertama adalah penemuan model proses, di mana model proses dibuat hanya dengan menggunakan perilaku yang diamati dalam *event log* [7]. Jenis kedua dari *process mining* adalah pemeriksaan kesesuaian kualitas, di mana model proses yang ada dibandingkan dengan *event log* dari proses yang sama [8]. Jenis ketiga adalah peningkatan model proses, di mana model proses ditingkatkan menggunakan informasi yang diperoleh dari event log. Dengan menggabungkan ketiga jenis tersebut, *process mining* dapat digunakan untuk memeriksa kepatuhan, menganalisis kemacetan, dan memprediksi penundaan, dan merekomendasikan tindakan untuk meminimalkan waktu pengerjaan yang diharapkan.

Tujuan dari penelitian ini melakukan studi komparasi efektivitas algoritma penemuan model proses dalam *process mining*. Algoritma yang digunakan untuk diuji adalah algoritma Alpha, Alpha++, dan Heuristic Miner. Selanjutnya dari ketiga algoritma tersbut dievaluasi dan dihitung kualitas tiap model proses yang terbentuk.

Paper ini disusun dengan susunan sebagai berikut: Bagian 1 merupakan pendahuluan berisi tentang latar belakang masalah. Bagian 2, tinjauan beberapa studi literatur dari beberapa peneliti sebelumnya dan metode penelitian yang dilakukan. Kemudian Bagian 3 membandingkan hasil percobaan menggunakan studi kasus dan Bagian 4 menyajikan kesimpulan.

2. Bahan dan Metode

2.1. Bahan

Penyusunan paper ini menggunakan proses studi literatur. Berikut ini akan dijelaskan studi literatur yang telah dilakukan.

2.1.1. Event Log

Paper ini menggunakan *event log* untuk proses analisis. Event log dapat menyimpan berbagai macam jenis informasi. *Event log* merupakan hasil dari rekaman aktivitas setiap kali seseorang mengerjakan sesuatu pada sistem informasi. *Event log* yang telah dikumpulkan lalu dianalisis sehingga dapat dijadikan sebagai pedoman pembuatan model proses bisnis dari suatu perusahaan untuk masa mendatang [9].

Untuk dapat menerapkan teknik *process mining*, hal yang pertama kali dilakukan adalah mengambil *event log* dari sumber data (misalnya: basis data, log transaksi, jejak audit, dll.). XES adalah format standar untuk *process mining* yang didukung oleh sebagian besar kakas bantu *process mining*. XES dibuat pada tahun 2010 oleh *IEEE Task Force on Process Mining*. Di samping XES (eXtensible Event Stream), format lain yang didukung oleh kakas bantu ProM adalah file MXML (Mining eXtensible Markup Language).

Event log memiliki beberapa istilah yang sering digunakan untuk dijadikan sebagai dasar analisis, yaitu *case*, *trace*, aktivitas, waktu, sumber daya. *Case* adalah satu rangkaian aktivitas yang saling terhubung dimulai dari aktivitas awal sampai aktivitas akhir untuk menyelesaikan masalah tertentu. *Trace* adalah kumpulan *case* yang memiliki jalur yang sama atau kesamaan rangkaian aktivitas. Aktivitas adalah pekerjaan yang telah dilaksanakan pada suatu proses bisnis. Waktu adalah waktu pelaksanaan yang dicatat pada setiap aktivitas. Sumber daya adalah orang atau perangkat yang bertugas menyelesaikan setiap aktivitas. Gambar 1 merupakan contoh event log dari sebuah perusahaan karaoke.

2.1.2. Penemuan Model Proses

Jenis pertama dan termasuk proses yang paling penting pada *process mining* adalah penemuan model proses. Penemuan model proses merupakan teknik pengambilan event log dari sistem informasi sebagai masukan dan menghasilkan model proses [10]. Analis menggunakan kemampuan penemuan model proses dengan mengumpulkan data event log dari sistem informasi dan membuat visualisasi alur kerja melalui sistem. Analisis dilakukan dengan memeriksa seluruh pola proses untuk mengungkap situasi yang menyebabkan kesalahan dan inefisiensi atau memperburuk risiko.

Penemuan model proses diawali dengan memeriksa seluruh aktivitas dan hubungan antar aktivitas yang paling sering muncul [11]. Analis dapat memeriksa varian proses atau *trace* yang dinotasikan dalam subset data *event log* dan menganalisis bagaimana *trace* dalam *event log* ini berbeda satu dengan yang lain.

Trace ID	Case ID	Activity	Time	Resource
1	1	memesan_ruangan	13/12/2015 08.00	Krisna
1	1	mengisi_formulir	13/12/2015 08.01	Kirana
1	1	menghitung_biaya	13/12/2015 08.02	Kirana
1	1	menghitung_diskon	13/12/2015 08.03	Kirana
1	1	bersiap_membayar	13/12/2015 08.04	Kirana
1	1	melakukan_pembaya	13/12/2015 08.05	Krisna
2	2	memesan_ruangan	13/12/2015 08.06	Paul
2	2	mengisi_formulir	13/12/2015 08.07	Kirana
2	2	mengisi_formulir	13/12/2015 08.08	Kirana
2	2	menghitung_biaya	13/12/2015 08.09	Kirana
2	2	bersiap_membayar	13/12/2015 08.10	Kirana
2	2	melakukan_pembaya	13/12/2015 08.11	Paul
1	3	memesan_ruangan	13/12/2015 08.00	Mila
1	3	mengisi_formulir	13/12/2015 08.01	Gina
1	3	menghitung_biaya	13/12/2015 08.02	Gina
1	3	menghitung_diskon	13/12/2015 08.03	Gina
1	3	bersiap_membayar	13/12/2015 08.04	Gina
1	3	melakukan_pembaya	13/12/2015 08.05	Mila

Gambar 1. Contoh event log yang memiliki *case*, *trace*, aktivitas, waktu, sumber daya (actor)

Empat perspektif utama yang dicakup oleh *process mining* selama penemuan model proses meliputi [12]:

1. Perspektif *control-flow* yang berfokus pada urutan aktivitas. Tujuan dari perspektif ini adalah untuk menemukan gambaran komprehensif dari semua kemungkinan urutan aktivitas. Hasil dari perspektif ini biasanya dinyatakan dalam bentuk notasi pemodelan proses (misalnya, diagram BPMN, Petri net, EPC atau UML).
2. Perspektif organisasi berfokus pada informasi sumber daya dalam suatu aktivitas, yaitu aktor (misalnya, orang, sistem, peran atau departemen) yang terlibat dan bagaimana mereka terkait. Tujuannya adalah untuk menyusun organisasi dengan mengatur orang dalam hal peran dan unit organisasi atau untuk menunjukkan jaringan sosial.
3. Perspektif *case* berfokus pada karakteristik *case*. Sebuah *case* bisa ditentukan dengan melihat alur model proses atau oleh aktor yang mengerjakannya.
4. Perspektif waktu berkaitan dengan waktu dan frekuensi kejadian. Saat peristiwa terjadi pada suatu waktu, dimungkinkan untuk menemukan kemacetan, mengukur tingkat layanan, memantau pemanfaatan sumber daya, dan memprediksi waktu pemrosesan yang tersisa dari case yang berjalan.

2.1.3. Pengukuran Kualitas Model Proses

Jenis kedua dari *process mining* adalah pengukuran kualitas model proses. Di sini, model proses yang ada dibandingkan dengan event log dari proses yang sama [13]. Pemeriksaan kesesuaian kualitas dapat digunakan untuk memeriksa apakah data *real-time*, seperti yang tercatat di log, sesuai dengan model dan sebaliknya.

Algoritma *process mining* menganalisis tingkat kesesuaian antara model proses dan data event log. Beberapa kakas bantu *process mining* menyediakan fitur untuk memeriksa dan menganalisis kinerja model proses berdasarkan identitas individu dan tim yang melaksanakan aktivitas.

Terdapat setidaknya dua jenis pengukuran model proses yang digunakan pada penelitian ini, yaitu fitness dan presisi. Pengukuran fitness memiliki nilai tinggi (atau setara dengan satu) apabila trace yang ada pada event log semakin banyak ditampilkan dalam model proses. Begitu juga sebaliknya nilai fitness akan semakin turun apabila trace yang ada pada event log semakin sedikit ditampilkan dalam model proses. Pengukuran presisi memiliki nilai tinggi apabila kemungkinan trace dari model proses yang terbentuk semakin sedikit dimana trace tersebut tidak terdapat pada event log. Begitu juga sebaliknya nilai presisi akan semakin turun apabila kemungkinan trace dari model proses yang terbentuk semakin banyak dimana trace tersebut tidak terdapat pada event log.

Nilai kualitas fitness dapat dilihat dinyatakan sebagai

$$Qf = \frac{CCE + LCE}{2} \quad (1)$$

dimana nilai LCE merupakan hasil perhitungan nilai kesesuaian *trace*, sedangkan nilai CCE merupakan hasil perhitungan nilai kesesuaian aktivitas. Nilai LCE dapat dihitung menggunakan

$$LCE = \frac{tmp}{tel} \quad (2)$$

dimana tmp merupakan jumlah *trace* dari *event log* yang muncul dalam model proses sedangkan tel merupakan jumlah keseluruhan *trace* yang terdapat pada *event log*. Nilai CCE dapat dihitung menggunakan

$$CCE = \frac{1}{2}\frac{(ael-amp)}{ael} + \frac{1}{2}\frac{(ael-atel)}{ael} \quad (3)$$

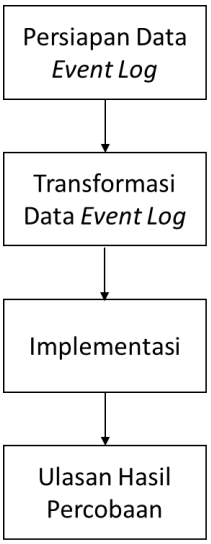
dimana ael merupakan total aktivitas yang terdapat pada event log, amp adalah jumlah aktivitas yang terdapat pada *event log* yang tidak muncul dalam model proses yang terbentuk, atel merupakan jumlah aktivitas yang ada pada *trace* yang terdapat pada event log setelah aktivitas terakhir yang muncul dalam model proses yang terbentuk. Nilai kualitas presisi dapat dinayatakan dengan

$$Qp = \frac{tmp}{tmp + ttmp} \quad (4)$$

dimana tmp merupakan trace yang terdapat pada event log dapat muncul dalam model proses yang terbentuk sedangkan ttmp merupakan trace yang terbentuk dari model proses yang tidak terdapat di *event log*.

2.2. Metode

Penelitian ini memiliki empat tahapan dasar: persiapan data event log, transformasi data event log dari CSV menjadi bentuk MXML menggunakan kakas bantu Disco, implementasi menggunakan kakas bantu ProM [14], dan ulasan hasil percobaan. Secara garis besar, tahapan proses yang dilaksanakan pada penelitian ini ditampilkan pada Gambar 2.



Gambar 2. Flowchart pengerjaan penelitian

2.2.1. Persiapan Data Event Log

Tahapan penting yang perlu disiapkan sebelum melakukan percobaan penemuan model proses yaitu persiapan data. Persiapan data dilakukan dengan mengambil data event log yang telah disimpan dalam sistem informasi. Event log yang baik apabila terdapat aktivitas, waktu, dan resource. Event log yang dijadikan sebagai masukan model proses terdiri dari tiga puluh case.

2.2.2. Transformasi Data Event Log

Transformasi data event log dilakukan agar saat proses implementasi ke dalam kakas bantu ProM dapat terbaca. Sebuah kakas bantu seperti Disco milik Fluxicon dapat secara otomatis mengubah data event log berupa CSV menjadi bentuk MXML. Bentuk MXML dapat dilihat pada Gambar 3. Tidak terdapat perbedaan mencolok pada bentuk MXML hanya saja bentuk ini merupakan bentuk narasi dari bentuk CSV dan terdapat penulisan complete untuk setiap aktivitas yang telah dikerjakan.

```
<!-- MXML version 1.0 -->
<!-- Created by Fluxicon Disco (fluxicon.com/disco) -->
<!-- (c) 2020 by Fluxicon BV, Bomanshof 259, 5611 NS Eindhoven, The Netherlands -->
<WorkflowLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://is.tm.tue.nl/research/processmining/WorkflowLog.xsd">
        <Source program="Fluxicon Disco"/>
        <Process id="case.Sheet1.mxml.gz" description="Converted to MXML by Fluxicon Disco">
                <ProcessInstance id="1">
                        <Data>
                                <Attribute name="Variant">Variant 1</Attribute>
                                <Attribute name="Variant index">1</Attribute>
                        </Data>
                        <AuditTrailEntry>
                                <WorkflowModelElement>ordering</WorkflowModelElement>
                                <EventType>complete</EventType>
                        </AuditTrailEntry>
                        <AuditTrailEntry>
                                <WorkflowModelElement>input_transaction</WorkflowModelElement>
                                <EventType>complete</EventType>
                        </AuditTrailEntry>
                        <AuditTrailEntry>
                                <WorkflowModelElement>calculating_cost</WorkflowModelElement>
                                <EventType>complete</EventType>
                        </AuditTrailEntry>
                        <AuditTrailEntry>
                                <WorkflowModelElement>calculating_discount</WorkflowModelElement>
                                <EventType>complete</EventType>
                        </AuditTrailEntry>
```

Gambar 3. Contoh hasil transformasi event log dalam bentuk MXML

2.2.3. Implementasi

Tahap implementasi dilakukan dengan melakukan ekstraksi event log menjadi sebuah model proses. Kakas bantu yang digunakan dalam penelitian ini adalah ProM 5.2.

Data event log yang telah disiapkan seperti pada Gambar dan Gambar selanjutnya dimasukkan ke dalam ProM untuk dianalisis kualitas model proses. Berikut merupakan algoritma yang digunakan:

- Algoritma Alpha

Algoritma Alpha mengurutkan aktivitas berdasarkan hubungan antar aktivitas satu dengan yang lain [7]. Maka dari itu, akan didapatkan informasi proses mana yang merupakan relasi berurutan dan proses mana yang memiliki relasi paralel. Alpha dalam penggambaran model proses menggunakan Petri net. Petri net adalah grafik dua arah yang terdiri dari place, transisi, dan garis panah yang menghubungkan keduanya [15]. Fungsi dari place adalah sebagai penanda input atau output suatu transisi sedangkan transisi merupakan aktivitas yang ada pada proses bisnis. Beberapa relasi pada Petri net dapat dijelaskan sebagai berikut.

1. Follows (>) apabila terdapat tuple aktivitas (A,B) saling terhubung
2. Causal (→) apabila terdapat aktivitas A > B dan B! > A
3. Parallel (||) apabila terdapat aktivitas A > B dan B > A
4. Unrelated (#) apabila terdapat aktivitas B! > B dan B! > A

- Algoritma Alpha ++

Algoritma Alpha ++ dibuat dengan tujuan menangani kekurangan algoritma Alpha dan Alpha + [16]. Pada algoritma Alpha memiliki kelemahan tidak dapat menemukan length-one-loop, length-two-loop, aktivitas tersembunyi, aktivitas ganda, ketergantungan implisit, dan keterikatan pemilihan aktivitas.

- Algoritma Heuristic Miner

Algoritma Heuristic Miner adalah algoritma penemuan model proses yang menangani permasalahan *spaghetti processes* dengan berfokus pada jumlah kemunculan relasi antar aktivitas satu dengan lain pada *event log* dalam membentuk model proses [17]. *Spaghetti processes* merupakan event log yang memiliki relasi terlalu banyak dan rumit sehingga sulit untuk dibaca [5]. Relasi antar aktivitas dengan jumlah sedikit tidak akan dimunculkan pada model proses yang dibentuk oleh algoritma Heuristic Miner. Berikut merupakan langkah-langkah yang digunakan dalam pembentukan model proses oleh Algoritma Heuristic Miner:

1. Membuat matrik dependency graph untuk menyimpan jumlah relasi kebergantungan antar dua aktivitas.
2. Menentukan Dependency threshold, Positive Observation threshold, Relative to best threshold sebagai dasar pemilihan relasi kebergantungan yang akan dimunculkan dalam model proses.
3. Memeriksa jika terdapat short loop (length-one-loop atau length-two-loop).
4. Menentukan relasi parallel (XOR atau AND) antar aktivitas.
5. Model proses dapat terbentuk.

Kelebihan Algoritma Heuristic Miner yaitu mampu menampilkan aktivitas berulang berupa length-one-loop dan length-two-loop apabila dalam event log terdapat relasi tersebut serta hasil model proses dapat diubah ke dalam bentuk Petri net. Petri net dapat menampilkan relasi paralel seperti XOR dan AND. Kekurangan dari algoritma Heuristic Miner adalah jumlah relasi antar aktivitas yang tingkat kemunculannnya sedikit pada event log dapat dianggap sebagai gangguan sehingga relasi tersebut tidak dapat tampil pada model proses.

2.2.4. Ulasan Hasil Percobaan

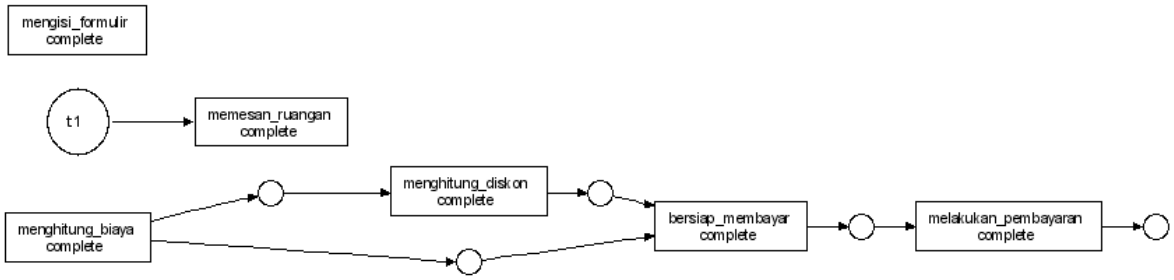
Tahap akhir dari penelitian ini adalah mengulas hasil dari setiap hasil percobaan penggunaan algoritma. Pengukuran kualitas model proses menjadi parameter penilaian yang dikaji dan dibandingkan, sehingga diketahui perbedaan pengukuran kualitas model proses mana yang lebih unggul dari setiap algoritma yang digunakan pada percobaan penelitian.

3. Hasil

3.1. Model Proses

3.1.1. Alpha

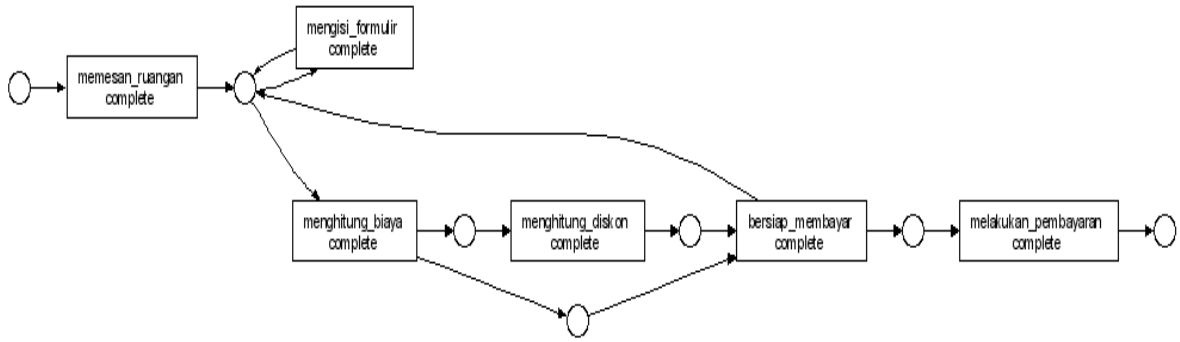
Model proses yang terbentuk dengan *event log* pernelitian menghasilkan relasi perulangan length-one-loop ditampilkan pada Gambar 4. Berdasarkan model proses tersebut, kekurangan utama dari algoritma Alpha adalah tidak mampu menampilkan relasi perulangan seperti length-one-loop dan length-two-loop.



Gambar 4. Hasil penemuan model proses algoritma Alpha

3.1.2. Alpha ++

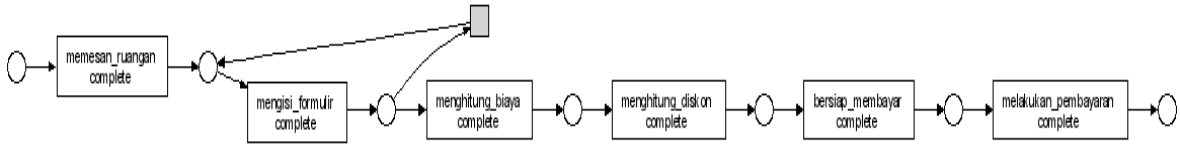
Algoritma Alpha ++ adalah pengembangan dari algoritma Alpha. Keunggulan Algoritma Alpha ++ adalah kemampuan menampilkan relasi perulangan seperti length-one-loop dan length-two-loop. Hasil dari penemuan model proses terlihat pada Gambar 5.



Gambar 5. Hasil penemuan model proses algoritma Alpha ++

3.1.3. Heuristic Miner

Meskipun algoritma Heuristic Miner dapat menangani kasus *spaghetti processes* akan tetapi relasi yang memiliki frekuensi kecil dalam event log tidak tergambarkan pada model proses yang dibentuk. Model proses yang dibentuk oleh algoritma Heuristic Miner ditampilkan pada Gambar 6.



Gambar 6. Hasil penemuan model proses algoritma Heuristic Miner

3.2. Kualitas Model Proses

Kualitas model proses yang terbentuk dihitung dari nilai fitness, didapatkan bahwa algoritma Alpha dan Heuristic Miner memiliki nilai fitness yang lebih tinggi karena karena tidak terdapat relasi perulangan. Sehingga banyak kemungkinan *trace* yang terbentuk. Nilai presisi Alpha lebih rendah dibanding Alpha ++ dan Heuristic Miner karena trace yang terbentuk tidak sesuai dengan *event log* yang ada. Tabel perbandingan nilai kualitas model proses ditampilkan pada Tabel 1.

Tabel 1. Tabel perbandingan nilai kualitas model proses

	Alpha	Alpha ++	Heuristic Miner
Fitness	0.96	0.94	0.96
Presisi	0.66	0.84	0.93

Pada penemuan model proses, algoritma Alpha ++ memiliki keunggulan dalam menangani proses perulangan baik length-one-loop maupun length-two-loop dibandingkan algoritma Alpha dan Heuristic Miner. Tetapi, dalam hal nilai fitness, algoritma Alpha dan Heuritic Miner lebih unggul dibandingkan algoritma Alpha++. Kemudian, dalam hal presisi, algoritma Heuristic Miner paling unggul dibandingkan algoritma Alpha dan Alpha++.

4. Kesimpulan

Algoritma Alpha, Alpha++, dan Heuristic Miner memiliki kecocokan tersendiri untuk kasusnya masing-masing. Heuristic miner lebih cocok digunakan untuk *spaghetti processes* karena *trace* yang memiliki frekuensi kecil dapat dianggap sebagai gangguan. Alpha dan Alpha ++ lebih cocok untuk kasus yang kecil. Kemudian, dalam hal terdapat perulangan, algoritma Alpha++ adalah alternatif terbaik.

Referensi

1. W. Van Der Aalst, T. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," IEEE Trans. Knowl. Data Eng., vol. 16, no. 9, pp. 1128–1142, 2004, doi: 10.1109/TKDE.2004.47.
2. L. Wen, W. M. P. van der Aalst, J. Wang, and J. Sun, "Mining process models with non-free-choice onstructs," Data Min. Knowl. Discov., vol. 15, no. 2, pp. 145–180, 2007, doi: 10.1007/s10618-007-0065-y.
3. M. Song and W. M. P. van der Aalst, "Towards comprehensive support for organizational mining," Decis. Support Syst., vol. 46, no. 1, pp. 300–317, 2008, doi: 10.1016/j.dss.2008.07.002.
4. A. K. Alves de Medeiros, A. J. M. M. Weijters, and W. M. P. van der Aalst, "Genetic process mining: A basic approach and its challenges," Bus. Process Manag. Work. (BPM 2005), vol. 3812, no. task C, pp. 203–215, 2006, doi: 10.1007/11678564_18.
5. A. K. A. De Medeiros et al., "Process mining based on clustering: A quest for precision," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 4928 LNCS, pp. 17–29, 2008, doi: 10.1007/978-3-540-78238-4_4.
6. A. Rozinat, R. S. Mans, M. Song, and W. M. P. van der Aalst, "Discovering colored Petri nets from event logs," Int. J. Softw. Tools Technol. Transf., vol. 10, no. 1, 2008, doi: 10.1007/s10009-007-0051-0.
7. D. Informatika, F. Teknik, U. T. Madura, and D. Informatika, "a More Efficient Deterministic Algorithm in Process," vol. 14, no. 3, pp. 971–995, 2018.
8. A. Burattin, F. M. Maggi, and A. Sperduti, "Conformance checking based on multi-perspective declarative process models," Expert Syst. Appl., vol. 65, pp. 194–211, 2016, doi: 10.1016/j.eswa.2016.08.040.
9. W. M. P. Van Der Aalst, Process Mining Discovery, Conformance and Enhancement of Business Processes. Germany: Springer, 2011.
10. Y. A. Effendi and R. Sarno, "Discovering process model from event logs by considering overlapping rules," Int. Conf. Electr. Eng. Comput. Sci. Informatics, vol. 2017-Decem, no. September, pp. 19–21, 2017, doi: 10.1109/EECSI.2017.8239193.
11. S. J. J. Leemans, D. Fahland, and W. M. P. van der Aalst, "Discovering block-structured process models from event logs containing infrequent behaviour," Lect. Notes Bus. Inf. Process., vol. 171, pp. 66–78, 2014, doi: 10.1007/978-3-319-06257-0_6.
12. M. Weske, "Business Process Management," Bus. Process Manag., no. Bpm 2007, pp. 24–28, 2015, doi: 10.1007/978-3-642-28616-2.
13. Y. Amelia Effendi and R. Sarno, "Conformance Checking Evaluation of Process Discovery Using Modified Alpha++ Miner Algorithm," Proc. - 2018 Int. Semin. Appl. Technol. Inf. Commun. Creat. Technol. Hum. Life, iSemantic 2018, pp. 435–440, 2018, doi: 10.1109/ISEMANTIC.2018.8549770.
14. F. M. Maggi, C. Di Francescomarino, M. Dumas, and C. Ghidini, "Predictive Monitoring of Business Processes," 2013, [Online]. Available: http://arxiv.org/abs/1312.4874.
15. B. Vázquez-Barreiros, M. Mucientes, and M. Lama, "A genetic algorithm for process discovery guided by completeness, precision and simplicity," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 8659 LNCS, pp. 118–133, 2014, doi: 10.1007/978-3-319-10172-9_8.

16. D. Rahmawati, M. A. Yaqin, and R. Sarno, "Fraud detection on event logs of goods and services procurement business process using Heuristics Miner algorithm," in 2016 International Conference on Information Communication Technology and Systems (ICTS), 2016, pp. 249–254, doi: 10.1109/ICTS.2016.7910307.
17. R. Sarno, Y. A. Effendi, and F. Haryadita, "Modified Time-Based Heuristics Miner for Parallel Business Processes," IRECOS (International Rev. Comput. Software), vol. 11, no. 3, pp. 249–260, 2016, doi: https://doi.org/10.15866/irecos.v11i3.8717.