**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**INSTITUT TEKNOLOGI SEPULUH NOPEMBER**

**USULAN TUGAS AKHIR**

# IDENTITAS PENGUSUL

**NAMA : BAGUS ARDIANSYAH**

**NRP : 5110100194**

**DOSEN WALI : Prof. Dr. Ir. Joko Lianto Buliali, M.Sc.**

**DOSEN PEMBIMBING : 1. Prof. Ir. Drs.Ec. Riyanarto Sarno, M.Sc, Ph.D**

**2. Abdul Munif, S.Kom, M.Sc**

# JUDUL TUGAS AKHIR

“Rancang Bangun Aplikasi *Cost Constrained Process Miner* dengan Menggunakan Metode *Heuristic”*

# LATAR BELAKANG

Seperti yang telah diketahui secara umum, teknologi *workflow* diciptakan untuk mempermudah suatu pekerjaan yang kompleks. Dengan teknologi *workflow*, organisasi dapat menjalankan proses bisnisnya dengan lebih efisien dan lebih efektif. *Workflow* sendiri adalah fasilitas komputerisasi atau otomatisasi sebagian atau seluruhnya dari sebuah proses bisnis. Dokumen, informasi, atau task di proses sesuai dengan aturan prosedural yang berlaku. *Workflow* digunakan untuk koordinasi task antar user dengan tujuan utama adalah efesiensi, cepat dan menguntungkan. *Workflow* bertugas untuk mengatur aliran kerja dan aktifitas user. *Workflow* itu sendiri terdiri dari suatu set aktifitas [2].

Dalam dunia bisnis, suatu perusahaan atau organisasi akan berusaha semaksimal mungkin dalam mempertahankan keunggulan dan meningkatkan keuntungan bagi perusahaan atau organisasi tersebut dalam ruang lingkup bisnis yang fluktuatif. Sehingga untuk mendukung usaha tersebut, organisasi atau perusahaan kerap menggunakan *Business Project Management* (BPM) dalam proses bisnis mereka, khususnya dalam permodelan proses dan otomatisasi. Untuk permodelan dan otomatisasi proses bisnis, banyak perusahaan yang menggunakan *Workflow Management System* (WFMS), karena secara prinsip, proses bisnis bisa diekseskusi dengan lebih cepat dan efisien. Sebuah event log akan dihasilkan dari WFMS tersebut, dan memberikan history record proses yang dieksekusi pada WFMS [3]. Dari event log tersebut, kemudian dapat digunakan dan diolah untuk meningkatkan kemampuan dan efektifitas bisnis otomatisasi proses melalui process mining.

Dalam dunia bisnis, Organisasi atau suatu perusahaan berusaha keras untuk mempertahankan keunggulan kompetitif dan meningkatkan keuntungan mereka dalam lingkungan bisnis yang selalu berubah. Dalam rangka mencapai tujuan tersebut, organisasi menggunakan teknik BPM, khususnya pemodelan proses dan proses otomatisasi, untuk model dan mengotomatisasi proses bisnis mereka [5] . Pemodelan dan otomatisasi proses bisnis organisasi dapat dicapai menggunakan WFMS, karena secara prinsip, proses bisnis bisa dieksekusi dengan lebih cepat dan efisien menggunakan WFMS [5]. Sebuah event log memberikan *history record* proses yang dieksekusi oleh WFMS. *Event log* kemudian dapat digunakan untuk meningkatkan kemampuan dan efektifitas bisnis otomatisasi proses melalui *process mining*.

Dalam dunia *management accounting*, biaya operasional atau produksi dari proses bisnis suatu organisasi atau perusahaan pasti dicatat untuk pelaporan. Hasil pelaporan biaya ini sering kali digunakan untuk melakukan keputusan bisnis, teknik perhitungan yang digunakan secara umum ada 3, yaitu yang pertama adalah *Activity Based Costing* (ABC) yang mengidentifikasi aktifitas dari suatu organisasi dan meletakkan *cost product* pada setiap aktifitas tersebut. Yang kedua adalah *Time Driven Activity-Based Costing* (TDABC), yang menghitung *time-based cost* untuk setiap aktifitas. Hasil biaya akhir dari suatu produk atau servis dihitung berdasar aktifitas dan berapa banyak waktu yang dihabiskan untuk membuat produk atau menyediakan jasa. Terakhir ialah, *Resource Consumption Accounting* (RCA) menitik beratkan pada penggunaan biaya dari suatu *resource* dari suatu organisasi, dimana biaya produk atau servis dihitung berdasar dari *Resource Utilization Cost* [5]. Pada tugas akhir ini, menggunakan RCA dalam perhitungan biaya dalam masukan atau *input,* tetapi di dalam aplikasi ini tidak menghitung akuntansi nya.

Disatu sisi, WFMS dapat menyediakan bantuan otomatisasi dalam menjalankan operasi bisnis, seperti pencatatan *event log* , sedangkan dalam *management accounting*, lebih menitik beratkan dalam menganalisa biaya operasi bisnis. Sudah tentu kedua hal ini akan sangat berkaitan, perubahan dalam proses bisnis akan mempengaruhi biaya operasional. Namun, masih sedikit sekali *workflow management system* yang sudah *support* informasi seputar *cost* pada *event log* nya. Untuk dapat meningkatkan kemampuan dan efektifitas bisnis, *event log* yang sudah *support cost* tersebut dapat diolah kembali melalui *process mining* untuk mendapat permodelan yang lebih optimal, dengan menggunakan batasan biaya operasional.

Salah satu *petri net* yang mampu menyimpan *cost* dengan satuan waktu adalah *priced timed petri net* [6]. Di dalam *priced timed petri net*, mampu menyimpan waktu pada event dan transisi, dapat pula menyimpan *cost* berdasar satuan waktu. *Priced Timed Petri Net* (PTPN) adalah suatu gabungan N = (Q, P.T, *Cost*) dimana Q adalah bilangan terbatas dari *control-state* dan P adalah bilangan terbatas dari *place*. Konfigurasi N itu sendiri adalah gabungan (q, M). Dimana q € Q adalah *control state* dan M adalah penanda dari N. Penanda ini adalah *multiset* dari P x R >= 0, i.e.. M € (P x R >= 0). Penanda M ini mendefinisikan banyak dan umur dari *token* yang diletakkan di dalam net. *Token* ini dapat diidentifikasikan di dalam penanda M sebagai suatu pasangan (p, x) merepresentasikan *place* dan umur dari M. sehingga M (p, x) mendefinisikan banyaknya *token* dengan umur x dan place di p [6].

Hasil yang diharapkan adalah dengan mengolah *event log* yang diberikan sehingga, dapat membuat permodelan *workflow net* yang *cost aware*. *Workflow net* yang dihasilkan serupa *price timed petri net* yang mampu menyimpan informasi seputar *cost* pada *place* dan *event* yang dilakukan. Harapannya ialah mampu untuk mengatasi *cost constraint* yang diberikan pada awal pengolahan *event log* dan *cost optimation*, dalam artian mendapatkan permodelan *workflow net* yang mampu memberikan hasil *cost* yang paling optimal atau sesuai dengan batasan atau *constraint* yang diberikan. *Cost* yang tersimpan di dalam *workflow net* berupa *Resource Consumption Accounting* yang akan dijelaskan lebih lanjut pada bab metodologi.

# RUMUSAN MASALAH

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana mendapatkan model workflow net yang optimal sesuai dengan *cost constraint* yang ada.
2. Bagaimana cara memodifikasi algoritma *heuristic* di dalam *process miner* agar dapat mempertimbangkan biaya.
3. Bagaimana mengubah *heuristic net* yang merupakan hasil dari algoritma *heuristic* menjadi *workflow net* yang lebih umum digunakan.
4. Bagaimana menampilkan hasil solusi berupa workflow net sehingga memudahkan pengguna untuk mengerti dan memahami serta mengimplementasikannya.
5. Bagaimana memadukan hasil perhitungan dan permodelan sehingga dapat menjadi *plugin* pada Prom.

# BATASAN MASALAH

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Bahasa pemrograman menggunakan bahasa Java.
2. Data Uji yang akan digunakan berupa event log yang sudah support cost.
3. *Plugin* yang dihasilkan khusus untuk *Software Prom* 6.0.
4. Lingkup c*ost* dalam Tugas Akhir ini sebatas pada event log.
5. *Cost driver* dan *cost function* tidak di *generate* di dalam aplikasi ini, sudah merupakan *input* siap pakai.
6. Ruang lingkup permasalahan meliputi produksi.

# TUJUAN PEMBUATAN TUGAS AKHIR

Tujuan dari Tugas Akhir ini adalah sebagai berikut:

1. Membuat *plugin* Prom dengan input atau masukan berupa *event log* dan *cost constraint*.
2. Membuat *plugin* Prom yang dapat mencari model yang optimal dengan menggunakan algoritma *heuristic.*
3. Membuat *plugin* Prom yang mampu mengubah *heuristic net* menjadi *workflow net.*
4. Membuat *plugin* Prom yang menghasilkan *Workflow net model* yang *cost aware.*

# MANFAAT TUGAS AKHIR

Manfaat dari Tugas Akhir ini adalah sebagai berikut:

1. Dapat mempermudah dalam mengolah *input event log* dan *cost constraint.*
2. Dapat mempermudah dalam pembuatan model yang optimal dengan menggunakan algoritma *heuristic.*
3. Dapat mempermudah dalam pengkonversian *heuristic net* menjadi *workflow net.*
4. Dapat mempermudah dalam pembuatan *Workflow net model* yang *cost aware.*

# TINJAUAN PUSTAKA

## Algoritma Heuristic

Algoritma *heuristic* untuk *process mining*, menggali perspektif aliran control dari suatu process model. Sehingga, bisa dikatakan algoritma ini hanya mementingkan urutan dari suatu *event* dalam case [1].Terdapat beberapa tahap dalam menggunakan algoritma *heuristic,* yaitu:

1. Mining *dependency graph*

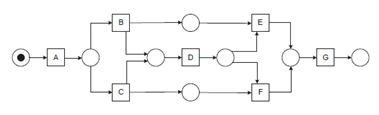
Langkah awal dari algoritma *heuristic* ialah dengan membuat *dependency graph.* Hitungan dasar berupa menghitung frekuensi digunakan untuk menunjukkan bagaimana hubungan dan keterkaitan antara dua buah *event* A dan B ( notasi A =>w B) nilai dari perhitungan =>w antara *event* dari suatu *event log* digunakan di dalam *heuristic search* untuk mendapatkan relasi keterkaitan.

Bagi sebagian relasi *dependency* atau keterkaitan tampaknya tidak perlu untuk selalu menggunakan nilai ambang batas atau *threshold*. Karena, kita tahu bahwa setiap kegiatan *non-initial* harus memiliki setidaknya satu kegiatan lain yang penyebabnya, dan setiap kegiatan *non-final* harus memiliki setidaknya satu kegiatan *dependent*. Dengan menggunakan informasi ini dalam apa yang disebut *all-activities-connected heuristic*, kita dapat mengambil calon terbaik (dengan A =>w B tertinggi). Heuristik yang sederhana ini membantu kita sangat besar dalam mencari hubungan kausalitas bahkan jika *event log* berisi *noise*[1].

1. *AND/XOR-split/join* dan *non-observable tasks*

Untuk dapat melakukan *mining* aktivitas yang *non-observable* *(AND-split* dan *AND-join*) sangat rumit, karena tidak muncul di dalam event log. Oleh karena itu untuk menghindari *modeling* dari aktivitas yang tak terlihat atau *invisible*, di dalam algoritma *heuristic* ini tidak menggunakan *petri net* sebagai reperesentasi dari *process model*, tapi menggunakan *Causal Matrix*. Penerjemahan dari suatu *petri net* menjadi *Causal Matrix* sangat jelas[1].

1. *Mining long distance dependencies*



Gambar 1. Contoh model yang menunjukkan *long distance dependency*.

Dalam beberapa process model, pilihan antara dua aktivitas tidak ditentukan di dalam suatu node di dalam *process model*, namun tergantung pada pilihan yang dibuat pada bagian lain dari *process model* tersebut. Seperti pada gambar diatas, menunjukkan *long distance dependency*. Setelah mengeksekusi aktivitas D, masi ada pilihan lain antara aktivitas E dan aktivitas F. Walaupun begitu, pilihan antara E dan F di kendalikan oleh pilihan sebelumnya antara B dan C. Sehingga jelas, sifat *non-local* ini susah untuk di gali dengan pendekatan mining yang biasa yang hanya berdasar informasi biner ( a>w b) [1].

## Cost Based Analysis

Fokus dari metode *cost based analysis* ialah dapat menghitung biaya dengan memasukkan informasi biaya yang terkait dalam *process mining.* Karya ini terdapat pada model data yang dihasilkan dari WFMS sesuai dengan ulasan temuan literatur, dan kemudian mengusulkan arsitektur untuk memungkinkan manajemen akuntansi laporan generasi melalui proses mining alat Prom. Tugas akhir ini menyajikan hasil implementasi pertama dengan menghasilkan berbagai jenis laporan akuntansi biaya manajemen [3].

Namun pada kali ini penggunaan metode ini hanya pada sebatas bagaimana dapat membaca *cost* yang ada pada *event log*. Karena *plugin* yang ada di dalam Prom sampai saat ini belum ada yang mampu membaca *cost*. *Cost* yang bisa dibaca dengan menggunakan *cost based analysis* sendiri harus sesuai dengan standar XES, oleh karena itu penambahan atribut *cost* pada *event log* tidak bisa sembarangan. Karena apabila penulisan tidak sesuai standar XES, tidak akan bisa dibaca sebagai *cost* menggunakan metode ini. Sampai saat ini belum ada metode lain yang mampu untuk mengenali *cost* dalam ruang lingkup Prom , hal ini dikarenakan masih terbatasnya metode dan *plugin* yang ada untuk mengenali *cost*. Dan dalam tugas akhir ini, *cost based analysis* yang digunakan adalah *Resource Consumption Accounting (*RCA), dimana RCA ini menitik beratkan pada penggunakan biaya dari suatu *resource* dari suatu organisasi, dimana biaya produk atau servis dihitung berdasar dari *Resource Utilization Cost.*

## *Plugin Prom*

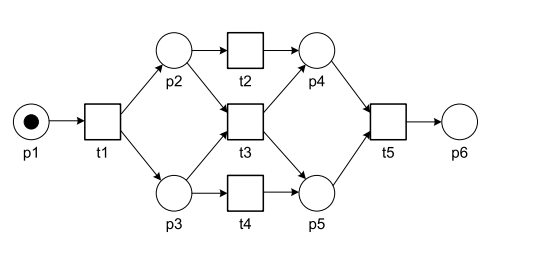
Sebuah *plugin* (atau *plugin* ekstensi) adalah komponen perangkat lunak yang menambahkan fitur khusus untuk aplikasi perangkat lunak yang ada. Bila aplikasi yang mendukung *plugin*, memungkinkan kustomisasi. Dalam tugas akhir ini, *plugin* yang dibuat khusus untuk *software* *process mining* bernama Prom [8]. *Plugin* prom ini akan muncul dalam pilihan ketika akan melakukan *process mining* menggunakan Prom.

## *Priced Timed Petri Net*

*Priced Timed Petri Net* (PTPN) Salah satu *petri net* yang mampu menyimpan *cost* adalah *priced timed petri net* [6]. Di dalam *priced timed petri net*, mampu menyimpan waktu pada *event* dan transisi dan dapat pula menyimpan cost berdasar satuan waktu. PTPN adalah suatu gabungan N = (Q, P.T, Cost) dimana Q adalah bilangan terbatas dari *control-state* dan P adalah bilangan terbatas dari *place*. Konfigurasi N itu sendiri adalah gabungan (q, M). Dimana q € Q adalah *control state* dan M adalah penanda dari N. Penanda ini adalah *multiset* dari P x R >= 0, i.e.. M € (P x R >= 0). Penanda M ini mendefinisikan banyak dan umur dari *token* yang diletakkan di dalam net. *Token* ini dapat diidentifikasikan di dalam penanda M sebagai suatu pasangan (p, x) merepresentasikan *place* dan umur dari M. sehingga M (p, x) mendefinisikan banyaknya *token* dengan umur x dan *place* di p [6].

## Petri net

Seperti yang dijelaskan pada [11], menunjukkan dasar dasar dari petri net terminologi dan notasi yang digunakan untuk tugas akhir ini. Petri net klasik ialah suatu triple (P,T,F) dimana P adalah set bilangan terbatas dari *place*. T adalah set bilangan terbatas dari transisi () dan adalah set dari *arcs* (relasi *flow*).



Gambar 2. *Petri net* dasar dengan *places* (p1,p2,p3,p4,p5,p6) dan transisi (t1,t2,t3,t4,t5).

## *Resource constrained workflow net*

Di dalam [10] menjelaskan perpanjangan dari *workflow net* yang disebut *resource-constrained workflow net* ( RCWF-net) yang dimana beberapa contoh dari *workflow* yang dieksekusi secara bersamaan, diasumsikan bahwa beberapa *instance* membagi *resource* global yang sama. Walaupun satu *instance* dari *resource cost workflow net* itu benar, beberapa *instance* bisa saja *deadlock* karena membagi *resource* yang sama.

## Workflow

Workflow terdiri dari urutan langkah-langkah yang terhubung di mana setiap langkah berikut tanpa penundaan atau kesenjangan dan berakhir tepat sebelum langkah berikutnya dapat dimulai. Ini adalah gambaran dari urutan operasi, dinyatakan sebagai karya seseorang atau kelompok, sebuah organisasi staf, atau satu atau lebih mekanisme sederhana atau kompleks. Workflow dapat dilihat sebagai salah abstraksi kerja nyata. Untuk tujuan kontrol, alur kerja mungkin pandangan karya nyata dalam aspek yang dipilih, sehingga berfungsi sebagai representasi virtual pekerjaan yang sebenarnya. Aliran yang dijelaskan bisa merujuk ke dokumen atau produk yang sedang ditransfer dari satu langkah ke yang lain.

# RINGKASAN ISI TUGAS AKHIR

*Plugin* Prom ini dibangun untuk membuat permodelan *workflow net model* yang *cost* *aware*, dalam artian dalam pembuatan *dependency graph* nya, memperhatikan *cost* yang ada pada setiap *event* dan juga memperhatikan batasan atau *constraint* yang di berikan oleh masukan pengguna. *Input* atau masukan pengguna terdapat tiga, yaitu *event* *log*, *threshold* dan *cost* *constraint*. Event log ini berisi *history* *record* proses yang dilakukan dalam *workflow management system* (WFMS) yang di dalamnya sudah memiliki *cost*. Sedangkan *threshold* merupakan batasan batasan terhadap algoritma *heuristic* itu sendiri, seperti batas maksimal *dependency*, *non* *observable*, atau *long* *dependency*. Yang terakhir adalah *cost* *constraint*, yaitu batasan *cost* sebagai penentu pembuatan *dependency* *graph*.

Sampai saat ini masih sangat sedikit *Workflow Management System* yang sudah support dengan *cost*, namun untuk standar penulisan *cost* sudah ada*, event log* yang memiliki *cost* tersebut memiliki ekstensi XES, Namun pada kali ini penggunaan metode ini hanya pada sebatas bagaimana dapat membaca *cost* yang ada pada *event* *log*. Karena *plugin* yang ada di dalam prom sampai saat ini belum ada yang mampu membaca *cost*. *Cost* yang bisa dibaca dengan menggunakan *cost based analysis* sendiri harus sesuai dengan standard XES, oleh karena itu penambahan atribut *cost* pada *event log* tidak bisa sembarangan. Karena apabila penulisan tidak sesuai standar XES, tidak akan bisa dibaca sebagai *cost* menggunakan metode ini.

Berikut arsitektur yang akan di bangun :

COST CONSTRAINT

EVENT LOG

THRESHOLD

RESOURCE COST ANALYSIS

HEURISTIC COST AWARE

DEPENDENCY GRAPH

CEK NON OBSERVABLE

CEK LONG DEPENDENCY

HEURISTIC NET

CONVERTER

WORKFLOW NET

Gambar 3. Arsitektur *Heuristic cost aware*.

*Event log* yang dimasukkan pengguna akan dihitung biayanya terlebih dahulu dengan menggunakan perhitungan *resource cost accounting.* Terdapat tiga fungsi utama yang terdapat pada *resource cost accounting* untuk dapat mengidentifikasi *cost* pada *event log*, yaitu *cost model*, *cost driver*, dan *cost function.* *Cost model* berisi data dan informasi yang dapat menganotasikan event log yang dimasukkan oleh user yang telah memiliki *cost,* sehingga dapat menghubungkan informasi yang terdapat di dalam event log dengan biaya sehingga tidak ada kerancuan di dalamnya.

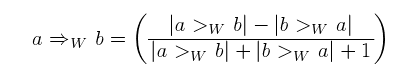
*Cost Driver* menjelaskan bagaimana *cost* yang ada di dalam *event* *log* diasosiasikan dengan proses dan *cost* yang dibutuhkan untuk setiap *event* yang ada dan definisikan sebagai *cost input,* sehingga *cost driver* dapat menyimpan *cost* yang diasosiasikan dengan diagram *task*. *Cost* *driver* juga memiliki jenis *costing*.

*Cost Function* menjelaskan bagaimana isi dari *report* *costing*, dan ia juga merupakan bagian dari *cost* *input*. Di dalamnya menjelaskan nama, deskripsi, ekspresi dan tipe pengembalian.

1. *Mining dependency graph*

Langkah awal dari Algoritma *heuristic* ialah dengan membuat *dependency* *graph*. Hitungan dasar berupa menghitung frekuensi digunakan untuk menunjukkan bagaimana hubungan dan keterkaitan antara dua buah *event* A dan B ( notasi A =>w B) nilai dari perhitungan =>w antara *event* dari suatu *event* log digunakan didalam *heuristic* *search* untuk mendapatkan relasi keterkaitan.

Didefinisikan W adalah sebuah *event log.* |A >w B| adalah jumlah dari A >w B terjadi, dan:



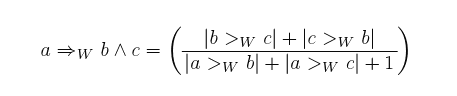
Perlu diperhatikan bahwa: -1 < a =>w b < 1. Nilai a =>w b yang tinggi dari merupakan indikasi yang kuat bahwa terdapat relasi ketergantungan antara aktivitas A dan B. Besarnya suatu nilai yang dihasilkan sangat bergantung pada *threshold* yang telah ditentukan sebelumnya dan juga tergantung pada *noise* yang ada pada *event* *log*.

Bagi sebagian relasi dependensi atau keterkaitan tampaknya tidak perlu untuk selalu menggunakan nilai ambang batas atau *threshold*. Karena kita tahu bahwa setiap kegiatan *non-initial* harus memiliki setidaknya satu kegiatan lain yang penyebabnya, dan setiap kegiatan *non*-*final* harus memiliki setidaknya satu kegiatan *dependent*. Dengan menggunakan informasi ini dalam apa yang disebu *tall-activities-connected heuristic,* kita dapat mengambil calon terbaik (dengan A =>w B tertinggi). Heuristik yang sederhana ini membantu kita sangat besar dalam mencari hubungan kausalitas bahkan jika *event* *log* berisi *noise*.

1. *AND/XOR-split/join* dan *non-observable tasks*

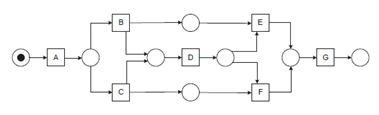
Untuk dapat melakukan *mining* aktivitas yang *non*-*observable* (*AND-split* dan *AND-join*) sangat rumit, karena tidak muncul di dalam *event* *log*. Oleh karena itu untuk menghindari *modeling* dari aktivitas yang tak terlihat atau *invisible*, di dalam algoritma *heuristic* ini tidak menggunakan *petri* *net* sebagai reperesentasi dari *process* *model*, tapi menggunakan *causal* *matrix*. Penerjemahan dari suatu *petri net* menjadi *causal* *matrix* sangat jelas.

Definisikan W adalah sebuah *event log* dari aktivitas T, {a,b,c} anggota T dan {b,c} memiliki relasi dependensi dengan a, maka:



|a >w b| |a>w c|adalah jumlah dari observasi positif dan |b >w c| + |c >w b| adalah jumlah b dan c muncul tepat setelahnya.

1. *Mining long distance dependencies*



Gambar 4. Model yang menunjukkan long *distance* *dependency*.

Dalam beberapa *process* *model*, pilihan antara dua aktivitas tidak ditentukan di dalam suatu node di dalam *process* *model*, namun tergantung pada pilihan yang dibuat pada bagian lain dari *process* *model* tersebut. Seperti pada gambar diatas, menunjukkan *long* *distance* *dependency*. Setelah mengeksekusi aktivitas D, masi ada pilihan lain antara aktivitas E dan aktivitas F. Walaupun begitu, pilihan antara E dan F di kendalikan oleh pilihan sebelumnya antara B dan C. Sehingga jelas, sifat *non-local* ini susah untuk digali dengan pendekatan mining yang biasa yang hanya berdasar informasi biner ( a>w b).

Gambar 5. Diagram Usecase.

Penjelasan dari masing masing fitur tersebut adalah:

1. Membuat *Heuristic* *net* model yang cost aware

Merupakan fitur utama dari plugin ini, yaitu membuat *heuristic* *net* model yang dalam pembuatannya , memperhitungkan batasan *cost* yang diberikan dan juga mampu untuk membaca *event* *log* yang memiliki *cost* sehingga bisa dijadikan penentu dalam pembuatan *dependency* *graph* sebelum akhirnya menjadi *heuristic* *net* model.

1. Menampilkan *event log* beserta biayanya

Fitur sampingan dari *plugin* ini adalah menampilkan *event log* yang merupakan masukan atau *input* dari aplikasi ini dan menampilkannya dalam keadaan terurut waktu dijalankannya event tersebut beserta biaya yang dikeluarkan setiap event.

1. Member *threshold* pada perhitungan heuristic

Ini merupakan fitur tambahan yang ada ketika akan membuat workflow net model menggunakan algoritma heuristic. Dengan fitur ini dapat memberikan batasan *dependency* yang ada, *non* *observable*, maupun *log* *dependency*. Fitur ini sifatnya tidak wajib, bila tidak dimasukkan akan secara otomatis menggunakan konfigurasi default.

1. Membuat *dependency* *graph* dengan *cost constraint*

Di dalam fitur ini ketika akan membuat *heuristic* *net model*, langkah awal adalah membuat *dependency* *graph*. Dengan masukan dari pengguna berupa *cost* *constraint*, akan menjadi penentu dalam pembuatan *dependency* *graph*. Karena hasil akhir yang diinginkan ialah *workflow net model* yang sesuai dengan batasan biaya yang ditetapkan pengguna.

1. Memberikan *constraint* biaya dalam pembentukan model

Dalam fitur ini dapat memberikan batasan biaya maksimal yang dapat dikeluarkan dari model yang ada. Dengan fitur ini model yang dibuat tidak akan melebihi batasan biaya yang telah diberikan.

1. Mengubah *heuristic net model* menjadi *workflow net model*

Satu hal yang menjadi kekurangan dari algoritma *heuristic* ialah, model yang dihasilkan tidak umum atau global, masih berupa *heuristic net model,* oleh karena itu di dalam aplikasi ini terdapat fitur untuk mengkonversi *heuristic net* menjadi *workflow net* agar dapat dengan mudah digunakan atau diolah kembali dengan aplikasi atau *plugin* lain.

1. Menampilkan *workflow net model*

Setelah selesai membuat *workflow net model*, dapat ditampilkan dan dilihat modelnya, dengan atribut atributnya seperti nilai *cost* yang ada. Juga dapat menampilkan *non* *obvervable* *task* serta *log dependenc*y, yang merupakan kelebihan dari algoritma *heuristic*.

# METODOLOGI

## Penyusunan proposal tugas akhir

Pada bab satu identitas pengusul berisi nama mahasiswa yang mengusulkan beserta NRP dan nama dari dosen wali beserta dosen pembimbing. Pada bab dua judul tugas akhir berisi judul tugas akhir yang diajukan, nama judul dari tugas akhir yang diajukan adalah “Rancang Bangun Aplikasi *Cost Constrained Process Miner* dengan Menggunakan Metode *Heuristic*”. Pada bab tiga latar belakang berisi latar belakang dibuatnya tugas akhir ini. pada beberapa paragraf awal menjelaskan latar belakang dan penjelasan dari *workflow* , *Business Project Management*, dan *Workflow Management System* yang saat ini sedang banyak digunakan organisasi maupun perusahaan. Pada beberapa paragraf selanjutnya dijelaskan metodologi yang telah diimplementasikan maupun yang akan diimplementasikan seperti penggunaan algoritma heuristic untuk melakukan *mining* pada *event* *log*, kemudian penggunaan *Resource Cost Accounting* untuk menghitung cost yang ada pada *event* *log*, kemudian pada paragraf terakhir menjelaskan hasil yang diharapkan yaitu berupa *priced timed petri net* yang pada pelabelan *cost* menggunakan perhitungan *resource cost accounting* dan sudah optimal serta sesuai dengan *cost constraint* atau *cost threshold* yang telah ditetapkan.Pada bab empat rumusan masalah berisi masalah yang diangkat pada tugas akhir ini, masalah utama yang dihadapi adalah bagaimana cara mendapatkan *workflow net* yang *cost aware* dan optimal serta memenuhi *cost constraint* yang diberikan.

Pada bab lima batasan masalah berisi batasan yang membatasi permasalahan yang dijelaskan pada bab selanjutnya yaitu bahasa pemrograman yang digunakan adalah java, data uji yang digunakan *support cost*, dan hasil akhir berupa *plugin* khusus untuk *software* Prom.Pada bab enam berupa tujuan pembuatan tugas akhir, tujuan utama dari tugas akhir ini adalah membuat *plugin* prom yang menerima masukan berupa *event* *log* yang *support* *cost* dan menghasilkan hasil berupa *workflow* *net* yang *cost* *aware*. Pada bab tujuh berisi manfaat tugas akhir yang dibuat yaitu dapat mempermudah dalam pembuatan *workflow* *net* yang *cost* *aware* dan optimal sesuai dengan *cost contraint* yang diberikan. Pada bab delapan berisi tinjauan pustaka yaitu algoritma yang digunakan berupa algoritma *heuristic* untuk pembentukan model *workflow* *net*, kemudian *cost* *based* *analysis*, *plugin* prom, dan *Priced Timed Petri Net.*

Pada bab sembilan ringkasan tugas akhir yang menjelaskan inti dari tugas akhir yang akan dibuat.kemudian bab sepuluh metodologi yang terdiri dari beberapa sub bab yaitu penusunan proposal tugas akhir, studi literatur, analisis dan desain perangkat lunak, implementasi perangkat lunak, pengujian beserta evaluasi, dan penyusunan buku tugas akhir. Pada bab sebelas berisi jadwal kegiatan pembuatan tugas akhir yang dimulai pada bulan oktober dan diakhiri pada bulan januari.Pada bab duabelas ialah daftar pustaka.

## Studi literatur

Terdapat beberapa studi literatur yang akan dipelajari, Antara lain:

1. Algoritma *Heuristic.*
2. *Cost Based Analysis.*
3. *Plugin* Prom.
4. *Priced* *Timed* *Petri* *net.*
5. *Petri* *Net.*
6. *Workflow* *Net*.
7. *Resource* *Constrained* *Workflow* *Net.*

## Analisis dan desain perangkat lunak

Proses analisis dari perangkat lunak yang akan dibuat ini adalah dengan pemecahan masalah masalah yang telah dirumuskan dalam bab rumusan masalah. Seperti bagaimana cara menemukan permodelan *workflow* *net* yang optimal dari suatu *event* *log* dengan diberikan *cost* *constraint*, tentu sebelum masuk ke dalam tahap implementasi harus merumuskan dan menganalisis terlebih dahulu.

Proses desain dari perangkat lunak yang akan dibuat ini adalah dengan mengolah hasil dari analisis yang telah dilakukan sebelumnya. Setelah masalah telah mampu dianalisis dan dipecahkan, maka dengan prosedur atau algoritma yang ada akan digunakan di dalam pengimplementasian sehingga mendapat hasil akhir yang diinginkan, dalam tugas akhir ini yaitu *workflow* *net* yang optimal secara *cost* sesuai dengan *cost* *constraint* yang diberikan.

## Implementasi perangkat lunak

Pengembangan aplikasi untuk tugas akhir ini akan berupa *plugin* untuk Prom 6.0. *Input* berupa *event* *log* yang telah memiliki *cost*, dan *output* yang dihasilkan berupa model *workflow* *net*. Beberapa hal yang diperlukan dalam implementasi ini adalah :

1. IDE *Eclipse Kepler.*
2. *Project* dasar Prom 6.0.
3. Bahasa Pemrograman Java
4. *Java Development Kit* 7 (JDK)
5. *Java Runtime Environment* (JRE)
6. *Library* Prom

## Pengujian dan evaluasi

Pengujian aplikasi ini akan menggunakan *black* *box* *testing*, dimana input berupa beberapa *event* *log* yang telah memiliki *cost*, dan akan dianalisis apakah hasil atau *output* yang dihasilkan memiliki hasil yang optimal. Dan juga mendeteksi apakah program dapat berjalan dengan seharusnya.

## Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Pendahuluan
   1. Latar Belakang
   2. Rumusan Masalah
   3. Batasan Tugas Akhir
   4. Tujuan
   5. Metodologi
   6. Sistematika Penulisan
2. Tinjauan Pustaka
3. Desain dan Implementasi
4. Pengujian dan Evaluasi
5. Kesimpulan dan Saran
6. Daftar Pustaka

# JADWAL KEGIATAN

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tahapan | Tahun 2013 | | | | | | | | | | | | | | | |
| Oktober | | | | November | | | | Desember | | | | Januari | | | |
| Analisa kebutuhan dan studi literatur |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Perancangan sistem |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Uji coba dan evaluasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Penyusunan buku |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | A. Weijters, W. v. d. Aalst and A. A. d. Medeiros, "Process Mining with Heuristic Miner Algorithm," *BETA Working Paper Series, WP 166, Eindhoven University of Technology, Eindhoven,* 2006. |
| [2] | W. v. d. Aalst and K. v. Hee, "Workfow Management: Models, Methods, and Systems," *MIT press, Cambridge, MA,* 2002. |
| [3] | T. Liu, Y. Cheng and Z. Ni, "Mining Event Logs to Support Workflow Resource Allocation," *Knowledge-based systems,* 2012. |
| [4] | W. Nauta, "Towards Cost-Awareness in Process Mining," Eindhoven University of Technology, Eindhoven, 2011. |
| [5] | P. A. Abdulla and R. Mayr, "Computing Optimal Coverability Costs in Priced Timed Petri Nets," *26th Annual IEEE Symposium on Logic in Computer Science (LICS 2011) IEEE,* pp. 399-408, 2011. |
| [6] | W. van der Aalst, K. van Hee, A. ter Hofstede, N. Sidorova, H. Verbeek and M. W. M. Voorhoeve, "Soundness of workflow nets: classification, decidability, and analysis," *Formal Asp. Comput.,* vol. 23, pp. 333-363, 2011. |
| [7] | J. Desel and J. Esparza, "Free Choice Petri Nets," *Cambridge Tracts in Theoretical Computer Science,* vol. 40, 1995. |
| [8] | W. Z. Low, "Cost-Aware Workflow Systems: Support for Cost Mining and Cost Reporting," Faculty of Science And Technology, Queensland University of Technology, Queensland,Australia, 2011. |
| [9] | M. Martos-Salgado and F. Rosa-Velardo, "Dynamic soundness in resource-constrained workflow nets," *Lecture Notes in Computer Science,* vol. 6722, pp. 259-273, 2011. |
| [10] | K. van Hee, A. Serebrenik, N. Sidorova and M. Voorhoeve, "Soundness of resourceconstrained workflow nets," *Lecture Notes in Computer Science,* vol. 3536, pp. 250-267, 2005. |
| [11] | W. v. d. Aalst, B. v. Dongen, C. Gunther, R. Mans, A. d. Medeiros, A. Rozinat, V. Rubin, M. Song, A. Weijters and H. Verbeek, "ProM 4.0 : comprehensive support for real process analysis," *26th International Conference on Applications and Theory of Petri Nets,* 2007. |