Ant Colony Optimization (ACO) untuk Permasalahan Multi Objective Flowshop Scheduling (MOFSP)

Kevin Jonathan–2014730020

1 Data Skripsi

Pembimbing utama/tunggal: **Cecilia E. Nugraheni**
Pembimbing pendamping: -
Kode Topik : **CEN4501**
Topik ini sudah dikerjakan selama : **1** semester
Pengambilan pertama kali topik ini pada : Semester **45 - Ganjil 18/19**
Pengambilan pertama kali topik ini di kuliah : **Skripsi 1**
Tipe Laporan : **B -** Dokumen untuk reviewer pada presentasi dan **review Skripsi 1**

2 Latar Belakang

Penjadwalan produksi merupakan aktivitas yang tidak terpisahkan dalam suatu perusahaan *manufakturing*. Penjadwalan (*scheduling*) sendiri didefinisikan sebagai suatu proses pengalokasian sumber daya atau mesin-mesin yang ada untuk melaksanakan tugas-tugas yang ada dalam suatu waktu tertentu (Baker, 1974). Sedangkan yang dimaksud dengan proses produksi adalah serangkaian langkah-langkah yang digunakan untuk mentransformasikan *Input* menjadi *Output*.

Proses penjadwalan *Flow Shop* adalah salah satu metode penjadwalan produksi di mana urutan mesin yang digunakan untuk setiap proses dalam seluruh pekerjaan harus sama. Dalam penelitian - penelitian penjadwalan sebelumnya hanya difokuskan pada satu kriteria saja (*single*) namun pada penelitian kali ini akan menggunakan lebih dari satu kriteria (*multiple*). Banyak algoritma yang dapat digunakan untuk menentukan urutan pengerjaan pekerjaan dalam proses penjadwalan produksi *Flow Shop*. Salah satu algoritma yang dapat digunakan dalam proses penjadwalan produksi *Multi Objective Flow Shop* adalah algoritma *Ant Colony Optimization*. Algoritma *Ant Colony Optimization* adalah algoritma yang mengadopsi perilaku koloni semut yang dikenal sebagai sistem semut. Algoritma ini menyelesaikan permasalahan berdasarkan tingkah laku semut dalam sebuah koloni yang sedang mencari sumber makanan.

Penelitian ini dibuat untuk mempelajari, mengaplikasikan, serta mengukur kinerja Algoritma *Ant Colony Optimization* pada proses penjadwalan *Multi Objective Flow Shop Scheduling* (MOFSP). Pada skripsi ini juga akan dibuat perangkat lunak yang dapat menerima n job yang masing-masing terdiri atas m buah operasi dan m buah mesin. Setiap operasi hanya ditangani oleh sebuah mesin dan setiap mesin hanya bisa menangani satu operasi. Urutan operasi dari setiap job adalah sama.

3 Rumusan Masalah

(a) Apa itu penjadwalan *Multi Objective Flowshop Scheduling* (MOFSP) ?

(b) Apa itu algoritma *Ant Colony Optimization* (ACO) ?

(c) Bagaimana cara kerja dan implementasi algoritma *Ant Colony Optimization* (ACO) dalam menyelesaikan permasalahan MOFSP ?

(d) Bagaimana kinerja algoritma *Ant Colony Optimization* (ACO) dalam menyelesaikan permasalahan MOFSP ?

4 Tujuan

(a) Menjelaskan penjadwalan *Multi Objective Flowshop Scheduling* (MOFSP).

(b) Menjelaskan algoritma *Ant Colony Optimization* (ACO) .

(c) Menampilkan cara kerja dan implementasi algoritma *Ant Colony Optimization* (ACO) dalam menyelesaikan permasalahan MOFSP .

(d) Mengetahui kinerja algoritma *Ant Colony Optimization* (ACO) dalam menyelesaikan permasalahan MOFSP dengan bantuan *benchmark* tertentu.

5 Detail Perkembangan Pengerjaan Skripsi

Detail bagian pekerjaan skripsi sesuai dengan rencan kerja/laporan perkembangan terkahir :

1. **Melakukan studi literatur : penjadwalan proses produksi secara umum, MOFSP, ACO, dan aplikasi ACO untuk masalah penjadwalan.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

   - **Definisi penjadwalan secara umum.**
     Secara umum penjadwalan menurut Baker (1974) didefinisikan sebagai proses pengalokasian sumber-sumber dalam jangka waktu tertentu untuk melakukan sekumpulan pekerjaan. Definisi ini mengandung dua arti yang berbeda, yaitu :

     (a) Penjadwalan merupakan fungsi pengambilan keputusan, yaitu menentukan jadwal.

     (b) Penjadwalan merupakan suatu teori, yaitu sekumpulan prinsip-prinsip dasar, model-model, teknik-teknik, dan kesimpulan-kesimpulan logis dalam proses pengambilan keputusan yang memberikan dalam fungsi penjadwalan (nilai konseptual).

     Menurut Conway (1967) Penjadwalan adalah proses pengurutan pembuatan produk secara menyeluruh pada beberapa mesin. Menurut Morton dan Pentico penjadwalan adalah proses pengorganisasian, pemilihan dan pemberian waktu dalam penggunaan sumber dayanya untuk melaksanakan aktivitas yang diperlukan dalam menghasilkan output yang diinginkan dengan memenuhi waktu yang diinginkan pula. Persoalan penjadwalan timbul apabila jumlah mesin dan peralatan yang dimiliki terbatas sedangkan terdapat beberapa pekerjaan yang dapat dikerjakan secara bersama. Untuk mendapat hasil yang optimal dengan keterbatasan sumber daya yang dimiliki, maka diperlukan adanya penjadwalan sumber-sumber tersebut secara efisien. Tujuan penjadwalan secara umum Baker (1974) adalah :

     (a) Meningkatkan produktivitas mesin, yaitu dengan mengurangi waktu menganggur mesin.

     (b) Mengurangi terhadap persediaan barang setengah jadi, dengan mengurangi rata-rata pekerjaan yang menunggu dalam antrian karena mesin sibuk oleh pekerjaan lain.

     (c) Mengurangi keterlambatan (tardiness). Dalam banyak hal, beberapa atau semua pekerjaan mempunyai batas waktu penyelesaian (duedate). Apabila suatu pekerjaan melewati batas waktu tersebut, maka akan dikenai pinalti. Keterlambatan dapat diperkecil dengan mengurangi maksimal tardiness atau mengurangi pekerjaan yang terlambat (number of tardy job).

Pada saat merencanakan suatu jadwal produksi, yang harus dipertimbangkan adalah ketersediaan sumber daya yang dimiliki baik berupa tenaga kerja, peralatan/prosesor ataupun bahan baku. Karena sumber daya yang dimiliki dapat berubah-ubah (terutama operator dan bahan baku), maka penjadwalan dapat kita lihat merupakan proses yang dinamis. Masalah penjadwalan muncul karena keterbatasan :

- Waktu
- Tenaga Kerja
- Jumlah Mesin
- Sifat dan syarat pekerja.

- **Klasifikasi Masalah Penjadwalan**
  Permasalahan penjadwalan dapat dilihat dari :

  (a) Mesin :
  - Mesin Tunggal
  - Mesin ganda (2 mesin)
  - M mesin

  (b) Aliran proses
  - *Job Shop*
  - *Flow Shop*

  (c) Pola Kedatangan
  - Statis
  - Dinamis

  (d) Elemen Penjadwalan
  - Deterministik
  - Stokastik

  Metode - metode penyelesaian masalah penjadwalan yaitu :

  (a) Heuristic

  (b) Matematis

  (c) Simulasi

- **Penjadwalan** *Flow Shop*
  Menurut Baker (1974) model penjadwalan dapat dibedakan menjadi 4 jenis keadaan, yaitu :

  (a) Mesin yang digunakan, dapat berupa proses dengan mesin tunggal atau proses dengan mesin majemuk.

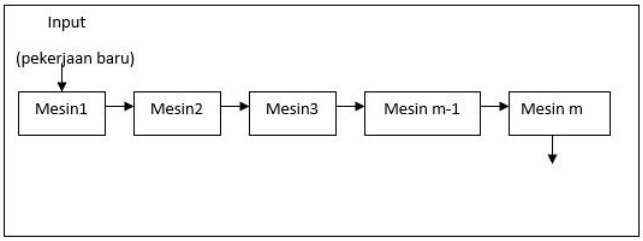
  (b) Pola aliran proses, dapat berupa aliran identik atau sembarang.

  (c) Pola kedatangan pekerjaan, Statis atau Dinamis.

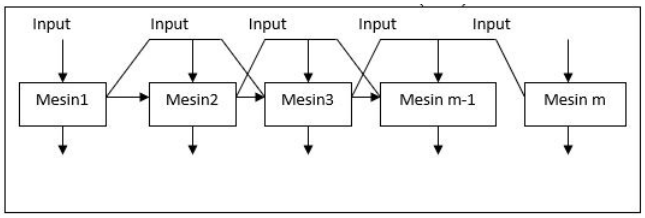
  (d) Sifat informasi yang diterima, dapat berupa Deterministik atau Stokastik.

  Pada jenis keadaan pertama, jumlah mesin dapat dibedakan atas mesin tunggal dan mesin majemuk. Model mesin tunggal merupakan model dasar dan biasanya dapat diterapkan dalam kasus mesin majemuk. Pada model kedua, pola aliran dapat dibedakan atas Flow Shop dan Job Shop. Pada Flow Shop dijumpai pola aliran pemrosesan dari suatu mesin ke mesin yang lain dalam urutan (routing) tertentu. Semua pekerjaan yang mengalir pada saat produksi yang sama tanpa boleh melewatinya disebut dengan pure Flow Shop. Tetapi jika pekerjaan yang datang kedalam

Flow Shop tidak harus dikerjakan pada semua mesin, jenis Flow Shop ini disebut dengan General Flow Shop. Contoh pola aliran Pure Flow Shop dan contoh pola aliran Gereral Flow Shop ditunjukkan pada Gambar 1 dan Gambar 2

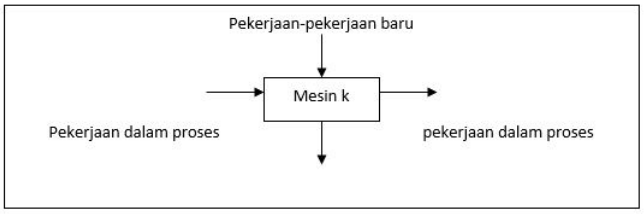


Gambar 1: Pola aliran *pure flow shop*



Gambar 2: Pola aliran *general flow shop*

Pada Job Shop setiap pekerjaan mempunyai routing yang berbeda. Alir proses yang tidak searah ini mengakibatkan setiap pekerjaan yang akan diproses pada suatu mesin dapat merupakan pekerjaan baru atau pekerjaan yang sedang dikerjakan (work in proses).



Gambar 3: Pola aliran *job shop*

Pada model ketiga, pola kedatangan pekerjaan dapat dibedakan atas pola kedatangan Statis dan Dinamis. Pada pola Statis, pekerjaan datang secara bersamaan pada waktu nol, siap dikerjakan pada mesin-mesin yang juga sudah siap untuk bekerja atau kedatangan pekerjaan yang tidak bersamaan tetapi saat kedatangan telah diketahui sejak waktu nol. Sedangkan pola Dinamis mempunyai kedatangan pekerjaan tidak menentu, dijumpai adanya variable waktu sebagai faktor pengaruh.

Pada model keempat, perilaku elemen-elemen penjadwalan dapat dibedakan atas Deterministik dan Stokastik. Model Deterministik dapat dilihat dari adanya kepastian atas informasi tentang beberapa aspek. Sedangkan pada model Stokastik, mengandung unsur ketidakpastian. Aspek yang dimaksud adalah :

(a) Karakteristik pekerjaan dari segi kedatangan, jumlah (kuantitas) pekerjaan, batas waktu penyelesaian (duedate) dan perbedaan kepentingan antar pekerjaan.

(b) Karakteristik pekerjaan dari segi banyaknya operasi, susunan mesin dan waktu proses.

(c) Karakteristik mesin dari segi jumlah dan kapasitas mesin, kemampuan dan kecocokan tiap mesin dengan pekerjaan yang diberikan.

Terdapat target utama yang ingin dicapai melalui penjadwalan flow shop ini yaitu jumlah output yang dihasilkan (throughput) berupa makespan. Penjadwalan flow shop didefinisikan sebagai penjadwalan dimana setiap job mempunyai pola aliran atau rute proses yang tetap pada seluruh mesin.

- **Beberapa Istilah dalam Penjadwalan Flow Shop**
  Penjadwalan Flow shop dapat dijelaskan sebagai berikut. Jika terdapat n job $\{j1, j2, \ldots jn\}$, maka harus diproses pada m mesin $\{m1, m2, \ldots, mm\}$. Waktu yang diperlukan untuk memproses job i pada mesin j adalah tij. Jadi permasalahan penjadwalan adalah menentukan urutan job yang memberikan solusi terbaik berdasarkan kriteria tertentu. Beberapa istilah yang digunakan dalam masalah penjadwalan yaitu :

  (a) Waktu proses (processing time = tj) : yaitu rentang waktu yang dibutuhkan untuk menyelesaikan suatu operasi pada job j.

  (b) Ready Time (rj), yaitu saat mulai suatu job j dapat dikerjakan.

  (c) DueDate (dj), yaitu batas waktu akhir suatu job harus sudah terselesaikan. Bila melewati batas ini, suatu job dikatakan terlambat (tardy).

  (d) Waktu penyelesaian (completion time = Cj) : saat job j telah selesai dikerjakan.

  (e) Waktu tinggal (flow time = Fj) : lamanya job j berada dilantai pabrik (shop). Flow time dihitung sejak job siap dijadwalkan sampai job selesai dikerjakan.

  (f) Lateness (Lj), yaitu merupakan penyimpangan waktu penyelesaian saat job terhadap duedate job yang bersangkutan. Lateness dihitung dengan persamaan Lj = Cj - dj. Lj $<$ 0, saat penyelesaian memenuhi batas akhir (earliness). Lj $>$ 0, saat penyelesaian melewati batas akhir (tardiness).

  (g) Slack (SLj), yaitu waktu yang tersedia bagi suatu pekerjaan. SLj = dj - tj.

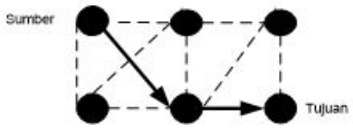
  (h) Tardiness (Tj), yaitu merupakan keterlambatan penyelesaian suatu job terhadap duedate job tersebut. Tj = max 0,Lj.

  (i) Makespan (Ms), yaitu waktu dimana semua pekerjaan terakhir selesai (MaxLj).

- **Multi Objective Flow Shop Scheduling**
  Pada dasarnya MOFSP adalah bagian dari *flow shop* itu sendiri. MOFSP menjelaskan kriteria optimasi yang akan diselesaikan menggunakan algoritma *Ant Colony Optimization*. Kriteria tersebut bersifat jamak *Multi Objective*. Kriteria tersebut misalnya saja minimasi waktu proses keseluruhan, minimasi waktu tunggu job, minimasi waktu nganggur mesin, dan sebagainya. Pemilihan kriteria optimasi yang akan diselesaikan pada skripsi ini akan ditentukan lebih lanjut pada saat pengerjaan skripsi 2.

- **Ant Colony Optimization**
  Any Colony Optimization (ACO) awalnya dikembangkan oleh Marco Dorigo et. al. (1996). Algoritma Ant Colony Optimization (ACO) adalah algoritma yang didasarkan pada cara kerja semut untuk menentukan jarak terpendek dari sarang menuju sumber makanan. Semut dapat menemukan jarak terpendek dengan memanfaatkan jejak pheromone (air liur semut) yang dimanfaatkan sebagai komunikasi tidak langsung antar semut. Ketika semut berjalan, ia meninggalkan pheromone dalam jumlah tertentu pada jalur yang dilewatinya. Semut dapat mencium pheromone dan ketika memilih jalur mereka cenderung untuk memilih jalur dengan konsentrasi pheromone yang lebih besar adalah jarak terpendek.

Saat seekor semut yang terisolasi bergerak secara acak, semut ini akan mengikuti jejak yang telah ditinggalkan sebelumnya yang dapat dideteksi dan mempunyai tingkat probabilitas yang tinggi untuk diikuti dan melanjutkan jejak sebelumnya dengan pheromone baru. Tingkah laku kolektif yang muncul disebut dengan tingkah laku Autocalystic, dimana semut yang lain dapat mengikuti jejak yang ada dan jejak yang semakin jelas akan memudahkan bagi semut yang lain untuk mengikutinya. Proses ini secara khusus terjadi melalui kumpulan umpan balik yang positif, dimana kemungkinan semut untuk memilih pola meningkat seiring dengan jumlah semut yang sebelumnya mengikuti pola yang sama.



Gambar 4: Jalur Solusi Semut *Ant Colony Optimization*

Marco Dorigo et. al. (1996) mengatakan bahwa Ant Colony Optimization adalah algoritma heuristik yang serba guna untuk memecahkan berbagai masalah optimasi. Algoritma ACO memiliki karakteristik sebagai berikut :

(a) Serba guna (versatile), dapat dipakai untuk memecahkan masalah dengan versi yang sama, seperti TSP dan Asymetric Travelling Salesman Problem (ATSP).

(b) Sempurna (robust), dapat diterapkan untuk memecahkan dengan hanya perubahan sedikit terhadap masalah optimasi yang lain, seperti Quadratic Assigment Problem dan Job shop Scheduling Problem (JSP).

(c) Pendekatan yang berbasis populasi.

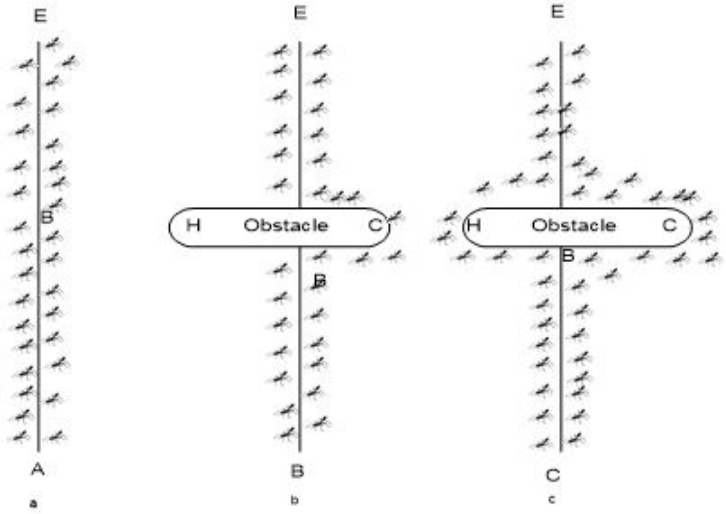
- **Algoritma Ant Colony Optimization pada penjadwalan flow shop**

(a) Karakteristik Semut

Sesuai dengan contoh eksperimen pada gambar 5 terdapat pola saat sekelompok semut berjalan (contoh, dari sumber makanan A ke sarang E, dan sebaliknya). Secara tiba-tiba rintangan muncul dan pola menjadi terpotong. Pada posisi B, semut berjalan dari A ke E (atau pada posisi D yang bergerak dengan arah yang berlawanan) harus memutuskan apakah harus bergerak ke kiri atau ke kanan. Pilihan ini dipengaruhi oleh intensitas jejak pheromone yang ditinggalkan oleh semut sebelumnya. Tingkat pheromone yang lebih tinggi pada pola sebelah kanan memberikan semut rangsangan yang lebih kuat dan kemungkinan yang lebih tinggi untuk berbelok ke kanan. Semut pertama mencapai titik B atau D mempunyai kemungkinan yang sama untuk belok ke kiri atau ke kanan (karena tidak terdapat pheromone pada dua pola alternatif tersebut).

Karena pola BCD lebih pendek dibandingkan dengan pola BHD, semut pertama yang mengikuti ini akan mencapai D sebelum semut yang mengikuti pola BHD. Hasilnya adalah semut yang bergerak dari E ke D akan mendapatkan jejak yang lebih jelas pada pola DCB, karena setengah dari semut tersebut yang memilih untuk mendekati rintangan melalui DCBA dan dengan segera akan sampai melalui BCD, mereka akan melalui pola memilih pola DCB dibandingkan pola DHB. Sebagai Konsekuensi, jumlah semut yang mengikuti pola BCD per unit waktu akan lebih banyak dibandingkan dengan jumlah semut yang mengikuti pola BHD. Hal ini menyebabkan jumlah pheromone pada pola yang lebih pendek akan muncul lebih cepat dibandingkan dengan pola yang lebih jauh, dan oleh karena itu kemungkinan semut

yang memilih pola yang diikuti mempunyai bias terhadap pola yang lebih pendek. Hasil akhir yang akan dipilih secara cepat akan ditunjukkan pada pola yang lebih pendek.
Algoritma yang akan dibahas pada bagian selanjutnya adalah model yang berasal dari kumpulan kehidupan nyata semut. Selanjutnya hal ini disebut dengan Sistem Semut dan algoritma yang akan dibahas dikenal dengan algoritma semut. Karakteristik semut asli untuk ketika sedang bergerak dari satu titik ke titik tujuan dapat dilihat pada kedua ilustrasi gambar berikut ini.



Gambar 5: Contoh karakteristik semut 1

Penjelasan karakteristik semut (1) :

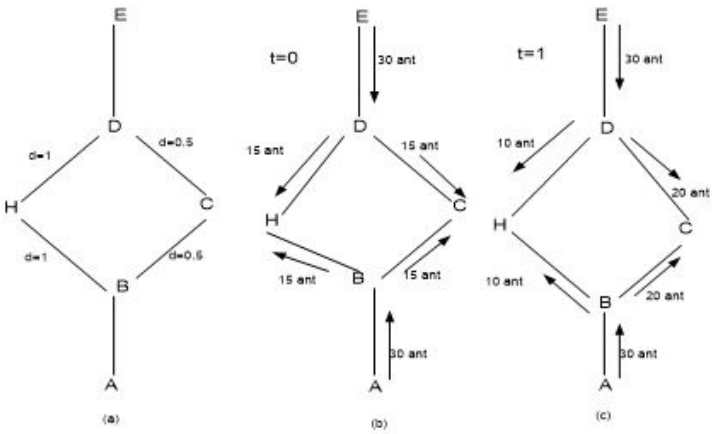
i. Semut-semut mengikuti jalur dari titik A ke titik E.

ii. Sebagai rintangan maka diletakkan hambatan pada jalur lintasan ; semut-semut memilih untuk bergerak memutari salah satu sisi dengan probabilitas yang sama.

iii. Di jalur yang lebih pendek ditinggalkan lebih banyak pheromone.

Oleh karena itu, bahwa metafora kumpulan semut dapat digunakan untuk menjelaskan model ini. Sesuai dengan gambar 6 yang merupakan interpretasi sistem semut pada situasi gambar 6. Untuk menyempurnakan hal ini, misalkan jarak antara D dan H, diantara B dan H, dan diantara B dan D melalui C sama dengan 1, dan posisi C adalah setengah jalan antara D dan B. Sekarang hal ini dipertimbangkan sebagai interval yang berlainan pada waktu : t (0, 1, 2) .
Misalkan bahwa 30 semut yang baru datang ke B dari A, dan 30 semut ke D dari E pada tiap unit waktu, kecepatan gerak semut pada kecepatan 1 per unit waktu, dan saat bergerak semut akan meninggalkan jejak pheromone pada intensitas 1, dan membuat hal ini menjadi lebih mudah, penguapan secara sempurna dan berurutan terjadi pada pertengahan interval waktu (t+1, t+2).
Pada saat t = 0 belum terdapat jejak, tetapi 30 semut berada di B dan 30 di D. Pilihan mereka tentangt jalan mana yang akan ditempuh adalah acak. Selanjutnya rata-rata 15 semut dari setiap bagian akan bergerak ke arah H dan 15 ke arah C. Pada t = 1, 30 semut baru akan muncul ke B dari A akan menemukan jejak pada intensitas 15 dan pola mengarah ke H, yang ditinggalkan oleh 15 semut yang bergerak dari B, dan jejak pada intensitas 30 pada pola C, akan mencapai kumpulan jejak yang ditinggalkan oleh 15 semut yang bergerak dari B

dan 15 semut yang mencapai B datang dari D melalui C. Kemungkinan dalam memilih pola selanjutnya bisa menjadi berat sebelah dan sejumlah semut diharapkan bergerak kearah C akan meningkat dua kali ke arah H : 20 versus 10. Hal yang sama berlaku untuk 30 semut pada D yang datang dari E. Proses berlanjut sampai semua semut akan memilih pola yang paling pendek.



Gambar 6: Contoh karakteristik semut 2

Penjelasan karakteristik semut (2) :

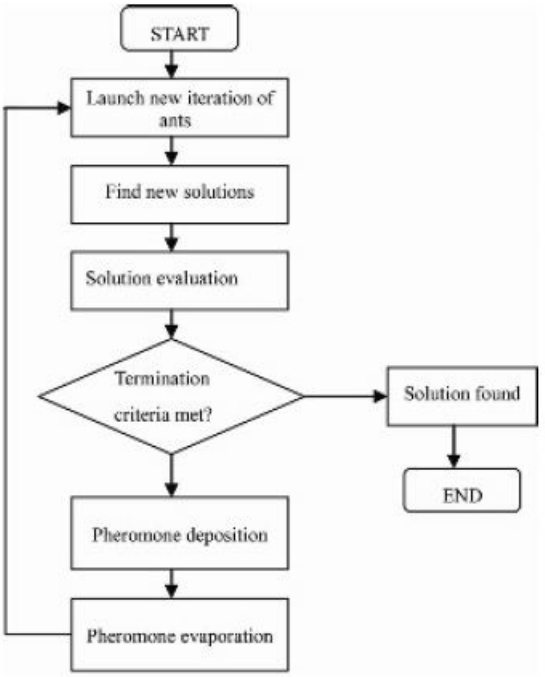
i. Graf dengan jarak-jaraknya.

ii. Pada saat t = 0 tidak ada jejak dijalur-jalur graf ; oleh karena itu semut-semut akan memilih apakah akan belok kekiri atau kekanan dengan probabilitas yang sama.

iii. Pada saat t = 1 jejak lebih banyak pada jalur yang lebih pendek, karena itu jalur tersebut akan ditinggalkan lebih banyak pheromone.

Ide yang muncul adalah jika kita memberikan nilai pada semut yang harus memilih diantara beberapa pola, yang secara kuat dipilih berdasarkan semut yang sebelumnya (dengan tingkat jejak tertinggi) dipilih dengan probabilitas yang lebih tinggi. Selanjutnya tingkat jejak yang lebih tinggi sejalan dengan pola yang pendek. Karakteristik semut tersebut digunakan oleh *Ant Colony Optimization* . Tetapi pada *Ant Colony Optmization* dimodifikasi dengan mengubah bahwa pada awalnya setiap jalur yang ada memiliki sejumlah pheromone awal, semut buatan akan memilih jalur-jalur yang ada dengan probabilitas yang tergantung pada jarak dan intensitas pheromone.

(b) Cara kerja Algoritma
Gambar 7 berikut adalah flow chart dari algoritma ant colony.



Gambar 7: Flowchart algoritma ant colony optimization

Berdasarkan Gambar 7, untuk menjalankan proses pelatihan dengan algoritma ant colony mempersiapkan jalur-jalur yang ada perlu dilakukan. Jalur-jalur tersebut merupakan kemungkinan solusi-solusi yang mungkin dipilih oleh semut. Kemungkinan dipilihnya suatu jalur akan ditentukan berdasarkan nilai feromon, oleh karena itu perlu dibentuk pula tempat penyimpanan feromon untuk setiap jalur yang mungkin dilalui. Pada awalnya, algoritma ant colony akan membentuk sekumpulan semut, semut-semut tersebut akan disebar secara acak pada jalur-jalur yang ada. Proses penyebaran semut tersebut akan memperhatikan nilai feromon dari masing-masing jalur. Semakin besar nilai feromon dari suatu jalur, maka semakin besar pula kemungkinan dipilihnya jalur tersebut untuk dilalui semut. Jalur-jalur yang dipilih oleh semut akan disimpan dan dibandingkan dengan jalur-jalur yang telah dipilih oleh semut-semut lainnya.
Dari jalur-jalur yang telah dipilih akan dibentuk nilai feromonnya berdasarkan tingkat keoptimalan dari jalur tersebut. Nilai feromon-feromon tersebut akan ditambahkan pada jalur-jalur yang bersangkutan untuk memperbesar kemungkinan dipilihnya jalur tersebut sesuai dengan tingkat keoptimalannya. Semakin optimal suatu jalur, semakin besar pula jumlah feromon yang akan ditambahkan pada suatu jalur. Tidak lupa pula, proses evaporasi feromon akan dilakukan. Proses evaporasi feromon ini akan mengurangi jumlah feromon yang disimpan. Proses evaporasi ini bertujuan untuk mengurangi jumlah feromon dari jalur-jalur yang dianggap kurang optimal. Proses-proses tersebut akan terus dilakukan hingga suatu kondisi berhenti ditemui.

```
membentukDataFeromon()
solusiTerbaik = null
while(kondisi berhenti belum ditemui)
    koloniSemut = bentuk kumpulan semut kosong
    for(1 sampai jumlah semut yang akan disebar)
        solusiLokal = pilihSolusiSecaraAcakBerdasarkanNilaiFeromon()
        If(solusiLokal solusi yang valid)
            (opsional) localSearch(solusiLokal)
            if(hasil(solusiLokal) lebih baik dari hasil(solusiTerbaik))
                simpan solusiLokal pada solusiTerbaik
            end if
            simpan solusiLokal pada koloniSemut
        end if
    end for
    updateNilaiFeromon(koloniSemut)
end while
output : solusiTerbaik
```

Gambar 8: Pseudocode algoritma ant colony optimization

Berdasarkan flow chart pada gambar 8, dibentuklah pseudocode di atas. Baris pertama dari pseudocode tersebut merupakan proses pembentukan data feromon. Proses pembentukan data feromon tersebut akan memperhatikan kasus yang ingin dicari hasil optimalnya. Nilai masingmasing jalur dan cara mengartikan nilai feromon tersebut diatur berdasarkan kasus yang ingin dicari hasil optimalnya. Jumlah feromon pada masing-masing jalur pada awalnya bernilai sama. Setelah data feromon selesai dibuat, proses pelatihan dengan menggunakan algoritma ant colony dimulai. Baris ketiga pada pseudocode menunjukkan kondisi berhenti untuk proses optimisasi dari algoritma ant colony.

Seluruh potongan kode di dalam kondisi while tersebut merupakan satu fase pelatihan dari algoritma ant colony. Fase pelatihan tersebut terdiri dari proses pembentukan jalur, penentuan solusi optimal, dan perubahan nilai feromon (penambahan dan evaporasi feromon). Fase pelatihan dari algoritma ant colony bertujuan untuk membentuk data feromon yang mampu menghasilkan solusi yang paling optimal. Pada setiap fase pelatihan ini akan dibentuk sejumlah semut yang bertugas untuk menyimpan solusi acak/solusi lokal yang telah dipilih pada suatu fase pelatihan. Semut-semut ini akan dianggap memilih sebuah jalur/solusi secara acak berdasarkan nilai yang disimpan pada data feromon. Jika solusi acak tersebut lebih baik dari solusi optimal yang diketahui pada saat ini, maka solusi acak tersebut akan dianggap sebagai solusi optimal yang baru.

Pada saat pemilihan solusi secara acak, perlu dipastikan bahwa solusi tersebut merupakan solusi yang valid dari suatu kasus. Pada saat pemilihan solusi kita juga dapat melakukan proses local search. Proses ini bertugas untuk membandingkan solusi yang dibentuk dengan solusi yang mirip. Jika solusi lain tersebut lebih baik, maka solusi yang dipilih akan diganti menjadi solusi yang mirip tersebut. Proses ini bersifat opsional, karena tidak semua permasalahan dapat diaplikasikan dengan proses ini.

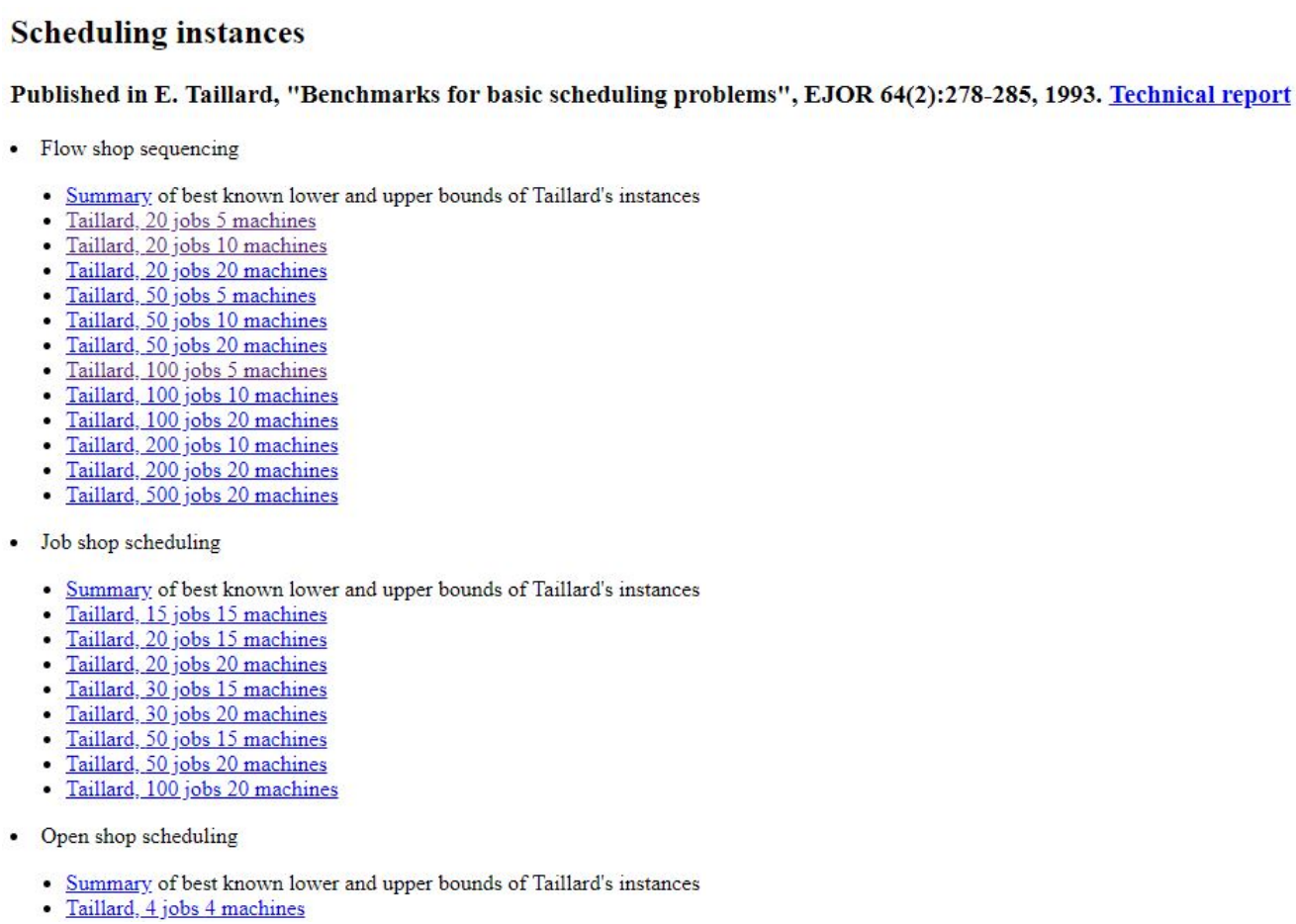
Setelah semut-semut telah selesai memilih solusi secara acak, proses perubahan nilai feromon akan dilakukan. Proses perubahan tersebut mencakup proses penambahan dan evaporasi nilai feromon. Proses penambahan nilai feromon akan memperhatikan tingkat keoptimalan dari solusi yang dipilih suatu semut. Semakin optimal suatu solusi, semakin banyak pula nilai feromon yang ditambahkan pada jalur/solusi tersebut. Setelah proses optimisasi selesai, solusi yang paling optimal akan diberikan sebagai data keluaran.

2. **Melakukan analisa aplikasi ACO pada masalah MOFSFP.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

3. **Mengembangkan perangkat lunak (analisis, desain, implementasi, dan pengujian).**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Akan dilakukan pada Skripsi 2.

4. **Melakukan eksperimen dengan sebuah** *benchmark*.
   Status : Ada sejak rencana kerja skripsi.
   Hasil :

   - **Data Pengujian**
     Dalam melakukan pengujian tingkat keoptimalan dari suatu algoritma, pengujian sebaiknya dilakukan dengan menggunakan data kasus standar. Data kasus standar tersebut akan dianggap sebagai suatu kasus unik yang memerlukan algoritma khusus untuk proses optimisasinya. Data standar tersebut akan digunakan sebagai pembanding dalam proses optimisasi. Dengan menggunakan data standar, penilaian kualitas dari suatu algoritma optimisasi dapat dilakukan. Dalam kasus penelitian ini, data kasus standar tersebut akan menggunakan data dari suatu benchmark yang bernama *Taillard Benchmark*. *Benchmark* tersebut dapat diakses lewat url berikut : `http://mistic.heig-vd.ch/taillard/problemes.dir/ordonnancement.dir/ordonnancement.html`.
     Berikut tampilan dari *Taillard Benchmark* :



Scheduling instances

Published in E. Taillard, "Benchmarks for basic scheduling problems", EJOR 64(2):278-285, 1993. Technical report

- Flow shop sequencing
  - Summary of best known lower and upper bounds of Taillard's instances
  - Taillard, 20 jobs 5 machines
  - Taillard, 20 jobs 10 machines
  - Taillard, 20 jobs 20 machines
  - Taillard, 50 jobs 5 machines
  - Taillard, 50 jobs 10 machines
  - Taillard, 50 jobs 20 machines
  - Taillard, 100 jobs 5 machines
  - Taillard, 100 jobs 10 machines
  - Taillard, 100 jobs 20 machines
  - Taillard, 200 jobs 10 machines
  - Taillard, 200 jobs 20 machines
  - Taillard, 500 jobs 20 machines
- Job shop scheduling
  - Summary of best known lower and upper bounds of Taillard's instances
  - Taillard, 15 jobs 15 machines
  - Taillard, 20 jobs 15 machines
  - Taillard, 20 jobs 20 machines
  - Taillard, 30 jobs 15 machines
  - Taillard, 30 jobs 20 machines
  - Taillard, 50 jobs 15 machines
  - Taillard, 50 jobs 20 machines
  - Taillard, 100 jobs 20 machines
- Open shop scheduling
  - Summary of best known lower and upper bounds of Taillard's instances
  - Taillard, 4 jobs 4 machines

Gambar 9: Tampilan *Taillard Benchmark*

Terdapat berbagai macam data pengujian, diantaranya : *flow shop*, *job shop*, dan *open shop*. Misalnya kita ingin mendapatkan data kasus standar untuk mengerjakan 20 jobs pada 5 mesin pada penjadwalan *flow shop*. Maka tampilan data kasus standar tersebut sebagai berikut :

```
number of jobs, number of machines, initial seed, upper bound and lower bound :
          20           5   873654221        1278        1232
processing times :
 54 83 15 71 77 36 53 38 27 87 76 91 14 29 12 77 32 87 68 94
 79  3 11 99 56 70 99 60  5 56  3 61 73 75 47 14 21 86  5 77
 16 89 49 15 89 45 60 23 57 64  7  1 63 41 63 47 26 75 77 40
 66 58 31 68 78 91 13 59 49 85 85  9 39 41 56 40 54 77 51 31
 58 56 20 85 53 35 53 41 69 13 86 72  8 49 47 87 58 18 68 28
number of jobs, number of machines, initial seed, upper bound and lower bound :
          20           5   379008056        1359        1290
processing times :
 26 38 27 88 95 55 54 63 23 45 86 43 43 40 37 54 35 59 43 50
 59 62 44 10 23 64 47 68 54  9 30 31 92  7 14 95 76 82 91 37
 78 90 64 49 47 20 61 93 36 47 70 54 87 13 40 34 55 13 11  5
 88 54 47 83 84  9 30 11 92 63 62 75 48 23 85 23  4 31 13 98
 69 30 61 35 53 98 94 33 77 31 54 71 78  9 79 51 76 56 80 72
number of jobs, number of machines, initial seed, upper bound and lower bound :
          20           5  1866992158        1081        1073
processing times :
 77 94  9 57 29 79 55 73 65 86 25 39 76 24 38  5 91 29 22 27
 39 31 46 18 93 58 85 58 97 10 79 93  2 87 17 18 10 50  8 26
 14 21 15 10 85 46 42 18 36  2 44 89  6  3  1 43 81 57 76 59
 11  2 36 30 89 10 88 22 31  9 43 91 26  3 75 99 63 83 70 84
 83 13 84 46 20 33 74 42 33 71 32 48 42 99  7 54  8 73 30 75
number of jobs, number of machines, initial seed, upper bound and lower bound :
          20           5   216771124        1293        1268
processing times :
 53 19 99 62 88 93 34 72 42 65 39 79  9 26 72 29 36 48 57 95
 93 79 88 77 94 39 74 46 17 30 62 77 43 98 48 14 45 25 98 30
 90 92 35 13 75 55 80 67  3 93 54 67 25 77 38 98 96 20 15 36
 65 97 27 25 61 24 97 61 75 92 73 21 29  3 96 51 26 44 56 31
 64 38 44 46 66 31 48 27 82 51 90 63 85 36 69 67 81 18 81 72
number of jobs, number of machines, initial seed, upper bound and lower bound :
          20           5   495070989        1236        1198
processing times :
 61 86 16 42 14 92 67 77 46 41 78  3 72 95 53 59 34 66 42 63
 27 92  8 65 34  6 42 39  2  7 85 32 14 74 59 95 48 37 59  4
 42 93 32 30 16 95 58 12 95 21 74 38  4 31 62 39 97 57  9 54
 13 47  6 70 19 97 41  1 57 60 62 14 90 76 12 89 37 35 91 69
 55 48 56 84 22 51 43 50 62 61 10 87 99 40 91 64 62 53 33 16
number of jobs, number of machines, initial seed, upper bound and lower bound :
          20           5   402959317        1195        1180
```

Gambar 10: Tampilan *Taillard Benchmark*

5. **Menulis dokumen skripsi.**
   Status : Ada sejak rencana kerja skripsi.
   Hasil : Berikut ini adalah bagian dari dokumen skripsi yang telah ditulis pada Skripsi 1 :

   - Bab 1
   - Bab 2
   - Sebagian Bab 3

6 Pencapaian Rencana Kerja

Langkah-langkah kerja yang berhasil diselesaikan dalam Skripsi 1 ini adalah sebagai berikut:

1. Melakukan studi literatur : penjadwalan proses produksi secara umum, MOFSP, ACO, dan aplikasi ACO untuk masalah penjadwalan.
2. Melakukan analisa aplikasi ACO pada masalah MOFSFP.
3. Mempelajari *benchmark taliard*.
4. Menulis sebagian dokumen skripsi.

Bandung, 11/11/2018

Kevin Jonathan

Menyetujui,

Nama: Cecilia E. Nugraheni
Pembimbing Tunggal