BAB I
PENDAHULUAN

1.1 Latar Belakang

Keberhasilan dan kegagalan sebuah bisnis bergantung pada penempatan fasilitas bisnis tersebut pada sebuah lokasi. Fasilitas tersebut dapat berupa kantor, gedung, pabrik, gudang penyimpanan barang, menara pemancar, dan sebagainya. Penempatan fasilitas yang efisien akan meningkatkan profit, meningkatkan pangsa pasar, mengurangi biaya operasional, dan meningkatkan kepuasan pelanggan. Permasalahan penempatan fasilitas pada suatu lokasi dapat didefinisikan sebagai penempatan beberapa fasilitas pada sejumlah lokasi yang tersedia agar seluruh pelanggan bisa terlayani dengan biaya minimal. Secara umum, permasalahan penempatan fasilitas diklasifikasikan menjadi *Capacitated Facility Location Problem* (CFLP) dan *Uncapacitated Facility Location Problem* (UFLP). *Facility Location Problem* pada beberapa literatur disebut juga dengan *Plant Location Problem*, *Warehouse Location Problem*, *The Location of Bank Account Problem*, dan sebagainya **(Guner & Sevkli, 2008)**.

Pada *Uncapacitated Facility Location Problem*, sebuah fasilitas diasumsikan dapat melayani semua pelanggan. **Lazic, dkk (2010)** menyebutkan masalah *Uncapacitated Facility Location* merupakan salah satu masalah lokasi yang paling banyak diteliti. **Lazic, dkk (2010)** juga menyebut pengaplikasian UFLP sangat beragam termasuk *Distribution System Design*, *Wireless Sensor Networks*, dan *Computer Vision*. **Galvao (2003)** menuliskan bahwa pendistribusian manfaat dari jaminan sosial di brazil dapat menggunakan model ULFP, khususnya di kota-kota kecil. **Cesarano, dkk (2021)** menggunakan model UFLP untuk mengembangkan manajemen *Road Side Unit* (RSU) pada lingkungan urban dinamis secara *real-time* dan hemat energi dimana hasil penelitian tersebut akan sangat bermanfaat pada *Internet of Vehicles* (IoV) yang merupakan pengembangan lanjutan dari *Internet of Things* (IoT).

Berbagai algoritma telah diterapkan untuk menyelesaikan *Uncapacitated Facility Location Problem* antara lain, *Simplified Binary Artificial Fish Swarm*

(**Azad, 2013**), *Artificial Bee Colony Optimization* (**Tuncbilek dkk, 2012**), *Ant Colony Optimization* (**Kole dkk, 2014**), *Cuckoo Search Algoritm via Levy Flight* (**Mesa, 2018**), *Particle Swarm Optimization* (**Guner & Sevkli, 2008**), *Firefly Algorithm* (**Tsuya dkk, 2017**), *Bat Algorithm* (**Babaoglu, 2016)**, *Genetic Algorithm* (**Tohyama dkk, 2011**), dan *Tabu Search Algorithm* (**Sun, 2005**).

Algoritma *Cuckoo Search* diperkenalkan Xin-She Yang dan Suash Deb pada tahun 2009. Algoritma *Cuckoo Search* terinspirasi dari perilaku burung *Cuckoo* dalam berkembang biak dengan cara meletakkan telurnya pada sarang burung inang dari spesies lain. Beberapa spesies burung *Cuckoo* seperti *Ani* dan *Guira* akan membuang telur burung inang saat meletakkan telurnya. Burung inang terkadang dapat mengetahui keberadaan telur burung Cuckoo dalam sarang, sehingga burung inang akan membuang telur burung Cuckoo atau memilih meninggalkan sarang dan membuat sarang baru di tempat lain. Beberapa spesies, misalnya *Tapera*, bahkan mampu meniru warna dan pola telur burung inang sehingga menurunkan kemungkinan telur burung *Cuckoo* dibuang **(Yang & Deb, 2009)**.

Algoritma *Cuckoo Search* menurut **Yang dan Deb (2009)** memiliki tiga aturan ideal. Pertama, burung *Cuckoo* hanya meletakkan satu telur pada satu waktu, sementara sarang yang menjadi tempat peletakan telur dipilih secara acak. Kedua, sarang terbaik dengan telur yang berkualitas dipilih sebagai solusi untuk generasi selanjutnya. Ketiga, jumlah sarang tetap dan probabilitas burung inang dapat menemukan telur burung *Cuckoo* adalah $pa \in [0, 1]$. Burung inang selanjutnya dapat membuang telur burung *Cuckoo* atau meninggalkan sarang dan membangun sarang baru. Dalam algoritma *Cuckoo Search*, solusi diasumsikan sebagai sebuah sarang.

Algoritma *Cuckoo Search* menggunakan *Levy Flight* untuk pencarian langkah. *Levy Flight* merupakan metode pencarian langkah yang terinspirasi dari pola pergerakan beberapa hewan dalam mencari makanan. Menurut **Yang dan Deb (2009)**, algoritma *Cuckoo Search* lebih efisien dalam mencari solusi optimal global dibandingkan dengan algoritma *Particle Swarm Optimization* dan *Genetic Algorithm*. **Zeng dan Chou (2012)** memberikan variasi algoritma *Cuckoo Search*

dengan menggunakan *Gaussian Distribution* sebagai pengganti *Levy Flight* yang menggunakan distribusi *Levy*. Menurut **Zheng dan Chou (2012)**, algoritma *Cuckoo Search* berdasarkan distribusi Gauss (GCS) mampu mengatasi kelemahan *Cuckoo Search* yaitu *covergence speed* dan akurasi yang rendah. Berdasarkan paparan diatas, maka dibuatlah penyelesaian *Uncapacitated Facility Location Problem* dengan menggunakan algoritma *Cuckoo Search* dengan *Gaussian Distribution*.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah di atas, maka rumusan masalah dalam penelitian ini adalah bagaimana penerapan algoritma *Cuckoo Search* berdasarkan Distribusi Gauss untuk penyelesaian *Uncapacitated Facility Location Problem*.

1.3 Tujuan

Tujuan dari penelitian ini adalah menerapkan algoritma *Cuckoo Search* berdasarkan Distribusi Gauss untuk penyelesaian *Uncapacitated Facility Location Problem*.

1.4 Manfaat

Memberikan alternatif penyelesaian masalah *Unlimited Facility Location* menggunakan algoritma *Cuckoo Search* berdasarkan Distribusi Gauss. Sehingga, hasil dari penelitian ini dapat dimanfaatkan lebih lanjut baik dalam hal pengembangan ilmu pengetahuan maupun dalam hal lainnya.

1.5 Batasan Masalah

Adapun batasan-batasan masalah dalam penelitian ini adalah sebagai berikut.

a. Permasalahan *Uncapacitated Facility Location* pada penelitian ini hanya menghitung biaya total, yaitu jumlah total biaya yang digunakan untuk membangun fasilitas dan biaya sebuah fasilitas melayani seorang pelanggan, tanpa memperhatikan jarak tempat tinggal pelanggan dengan fasilitas yang

akan dibangun, preferensi pelanggan untuk dilayani oleh fasilitas tertentu, dan lain-lain.

b. Pada satu lokasi hanya dapat dibangun satu fasilitas.

c. Data yang dipakai disimpan dalam *file* dengan tipe *comma separated values* (.csv) dalam aturan tertentu.