BAB I. PENDAHULUAN

1. Latar Belakang

DISC (*Data-Intensive Scalable Computing*) adalah sistem komputasi yang digunakan untuk mengumpulkan, mengkomputasi, dan mengelola kumpulan data berskala besar. DISC digunakan untuk memproses data besar untuk menghasilkan keputusan bisnis hingga penemuan ilmiah [1]. Karakteristik dari sistem DISC antara lain memproses *dataset* yang besar secara paralel dan menyimpannya dalam sistem penyimpanan bersama. Contoh pemrograman terdistribusi seperti ini dapat ditemukan pada *framework* seperti Apache Spark, Apache Hadoop, dan MapReduce.

Namun, dalam pengembangan sistem DISC, para pengembang sering kali menemui masalah pemrograman mulai dari kesalahan input, data yang tidak bersih, dan pengambilan keputusan yang salah karena kegagalan sistem dalam memproses data [1]. Hal ini dapat menjadi tantangan untuk melakukan *debug* pada pemrosesan data dalam sistem DISC, karena proses *debug* harus mengidentifikasi bagian data yang menyebabkan kegagalan, pengecualian, atau *crash* [2]. Cara lain untuk melakukan proses *debugging* ini adalah *trial and error debugging*, di mana pengembang mengulangi bagian dari prosedur pemrosesan data mereka pada subset data perantara, yang menghasilkan *outlier* atau *output* yang salah [2]. Sementara itu, *debugging* adalah aktivitas yang memakan waktu yang dapat menyumbang sebagian besar biaya pemeliharaan perangkat lunak [3].

Oleh karena itu, untuk memudahkan para pengembang ketika melakukan *debugging* pada sistem ini, mereka dapat menggunakan *debugging* otomatis. *Debugging* otomatis sistem DISC pada *framework* Apache Spark khususnya, dapat dilakukan dengan menggunakan *library data provenance* yang bernama Titian. Titian dapat menangkap *data provenance* pada sistem DISC. *Library* ini juga mendukung pelacakan *input* data yang bermasalah yang dapat memicu kegagalan dan kesalahan.

Saat ini, belum banyak penelitian yang menggunakan *library* Titian untuk melacak asal data di Spark. Penelitian pertama dilakukan oleh Interlandi [3], yang membuat *library* Titian dan menguji efisiensinya melalui perbandingan dengan NEWT dan RAMP menggunakan studi kasus *Word Count* dan *Grep*. Kemudian ditemukan bahwa dengan menggunakan Titian, *runtime overhead* tidak melebihi 30% dari *runtime program* dasar. Penelitian kedua dilakukan oleh Diestelkamper [4]. Hasil dari penelitian ini adalah bahwa perbandingan secara detail tidak dapat dilakukan karena Titian tidak mendukung *data provenance* untuk data bersarang dan beberapa skenario dalam penelitiannya. Namun, dia menyediakan data datar dan membuat skenario yang melibatkan komputasi *filter* dan *union*, *overhead* waktu dengan Titian adalah 5,89%, sementara menggunakan Pebble menghasilkan 6,98%.

Oleh karena itu, karena terbatasnya literatur tentang penggunaan Titian untuk melakukan pembuktian data, ada kebutuhan untuk menyelidiki penggunaan *library* Titian dalam studi kasus sistem DISC yang lain. Dalam konteks ini, *library* Titian digunakan dalam program analisis curah salju. Data klimatologi mengenai curah salju sering kali mengalami anomali, yang mengakibatkan kesenjangan data [5]. Pengamatan hujan salju modern dengan resolusi temporal yang tinggi diperoleh dari stasiun cuaca otomatis. Namun, stasiun-stasiun ini biasanya tidak mengukur kepadatan salju dan perubahan temporalnya secara langsung; sebagai gantinya, parameter-parameter ini sering kali diasumsikan atau dimodelkan [6]. Penelitian mengenai hujan salju ekstrem masih terbatas dan kurang dipahami, terutama dalam hal variabilitas spasial dan temporal serta penyebabnya [7]. Kesenjangan ini disebabkan oleh stasiun pengamatan yang langka, kurangnya indeks yang konsisten untuk perbandingan, dan mekanisme yang kompleks di balik curah salju ekstrem, terutama di *High Mountain Asia* (HMA) [7]. Selain itu, kegagalan untuk membedakan jenis curah hujan dalam pengamatan stasiun meningkatkan ketidakpastian dalam mengidentifikasi kejadian hujan salju [8]. Kesalahan data dalam analisis curah hujan, khususnya pada data curah hujan salju, disebabkan oleh berbagai faktor, termasuk kesalahan pengamat, kesalahan instrumen, dan masalah representasi data sebagai akibat dari keragaman curah hujan yang masih menjadi perhatian [9].

Sistem analisis curah salju disaring oleh fungsi '*failure*' yang mendefinisikan nilai mustahil dari curah salju. Delta curah salju yang melebihi nilai mustahil curah salju dikategorikan sebagai data kegagalan. Kemudian, Titian akan melacak *input* asli yang memicu kegagalan dan menunjukkannya kepada kami dengan segera.

Hasilnya, penelitian ini memberikan kontribusi yang signifikan pada bidang ini dengan memberikan evaluasi empiris yang komprehensif terhadap akurasi dan efisiensi Titian dibandingkan dengan *debugging* manual, khususnya dalam *debugging* otomatis untuk sistem analisis curah salju. Aspek akurasi dan efisiensi ini belum pernah diteliti dalam penelitian sebelumnya. Dengan mendemonstrasikan bagaimana Titian dapat menyederhanakan proses *debugging*, penelitian kami dapat menjadi dasar bagi penelitian di masa depan untuk mengeksplorasi analisis akar masalah dan tindakan pencegahan.

2. Topik dan Batasannya

Penelitian ini mengangkat topik pengujian keakurasian dan keefisienan Titian dibandingkan manual *debugging*. Studi kasus yang kami gunakan adalah program curah salju yang memiliki objektif utama yaitu mengklasifikasikan curah salju dari beberapa area di United States perharinya. Kami menggunakan 5 *dummy*

datasets yang memiliki ukuran yang berbeda-beda. Setiap dataset terdiri dari data tanggal, kode pos area, dan jumlah curah salju perhari dalam ukuran *feet*/mm.

Batasan pada penelitian ini adalah jumlah isi *dataset* yang digunakan untuk pengujian keakurasian dibatasi hanya dengan 3 bulan untuk setiap 4 tahun terakhir. Pembatasan ini dilakukan dikarenakan pengujian keakurasian dilakukan secara manual dengan melakukan *crosscheck* antara *output* Titian dengan data sebenarnya yang ada di *dataset*. Selain itu, penelitian ini dilakukan pada komputer yang tidak memiliki GPU *Processor*, sehingga untuk waktu pemrosesan kemungkinan berbeda berdasarkan kemampuan masing-masing komputer.

3. Tujuan

Tujuan dari penelitian ini adalah untuk mencari keakurasian Titian dalam melacak asal-usul data pada sistem DISC. Selain itu, dengan penelitian ini, kami akan membuktikan keefisienan Titian yang dibandingkan dengan *manual debugging* berdasarkan waktu yang dibutuhkan dalam mencari data yang menyebabkan *incorrect output* pada program DISC.

4. Organisasi Tulisan

Laporan tugas akhir ini berisi 5 BAB utama yaitu: BAB 1 berisi pendahuluan yang menjelaskan latar belakang penelitian, topik dan batasannya, tujuan penelitian, dan organisasi tulisan; BAB II memaparkan terkait studi literatur dari penelitian terdahulu; Kemudian, BAB III berisi metode penelitian yang kami gunakan untuk merealisasikan riset; Setelah riset berjalan, kami menuliskan hasil penelitian dan analisisnya pada BAB IV; Kemudian, diakhiri dengan kesimpulan pada BAB V.