การจัดกลุ่มข้อมูล เป็นงานรวมกลุ่มข้อมูลด้วยการพิจารณาความคล้ายคลึงกัน อัลกอริทึม k-means เป็นอัลกอริทึมมาตรฐานที่ทำการรวมกลุ่มข้อมูล โดยเริ่มต้นกระบวนการด้วยการคำนวณระยะห่างเพื่อกำหนดข้อมูลแต่ละตัวเข้ากลุ่มที่อยู่ใกล้ที่สุด จากนั้นคำนวณค่าจุดกึ่งกลางของแต่ละกลุ่มโดยใช้ค่าเฉลี่ยของข้อมูลทั้งหมดในกลุ่ม อัลกอริทึมจะทำสองขั้นตอนนี้ซ้ำจนกระทั่งค่าจุดกึ่งกลางไม่มีการเปลี่ยนแปลงและข้อมูลไม่มีการเปลี่ยนกลุ่ม ประสิทธิภาพของอัลกอริทึมจะขึ้นกับจำนวนข้อมูล จำนวนกลุ่ม และจำนวนรอบในการวนซ้ำ ผู้วิจัยได้พัฒนาวิธีการสุ่มตามความหนาแน่นเพื่อลดจำนวนข้อมูลที่จะใช้ในการจัดกลุ่ม ซึ่งจะช่วยให้อัลกอริทึมจัดกลุ่มข้อมูลทำงานได้เร็วขึ้น การสุ่มข้อมูลจะใช้วิธีการคัดเลือกข้อมูลตัวแทนจากบริเวณที่มีความหนาแน่นสูง อัลกอริทึมที่พัฒนาขึ้นนี้ได้รับการแปลงเป็นโปรแกรมที่เขียนด้วยภาษาเออแลง ซึ่งเป็นภาษาเชิงฟังก์ชันที่ช่วยให้การพัฒนาโปรแกรมต้นแบบทำได้อย่างรวดเร็ว และจากผลการทดสอบโปรแกรมพบว่าโปรแกรมสุ่มข้อมูลและจัดกลุ่มข้อมูลตามความหนาแน่นสามารถทำงานได้ดีกับข้อมูลที่มีการกระจายแบบซิฟ ผลการจัดกลุ่มด้วยข้อมูลสุ่มพบว่าจุดกึ่งกลางกลุ่มเบี่ยงเบนไปจากจุดกึ่งกลางที่แท้จริงเพียงเล็กน้อย เมื่อวัดประสิทธิภาพการใช้เนื้อที่หน่วยความจำของโปรแกรมที่พัฒนาขึ้นพบว่าใช้เนื้อที่หน่วยความจำน้อยกว่าโปรแกรมจัดกลุ่มข้อมูลตามปกติถึง 60% ดังนั้นเทคนิคและโปรแกรมที่พัฒนาขึ้นจึงมีศักยภาพที่จะพัฒนาให้ทำงานกับข้อมูลสตรีมได้ นอกจากนี้ภาษาเออแลงยังมีขีดความสามารถในการโปรแกรมแบบพร้อมกันกับหลายหน่วยประมวลผลได้ ซึ่งจะช่วยให้โปรแกรมทำงานได้เร็วขึ้นและรองรับข้อมูลปริมาณมากขึ้นได้

239013

Clustering is a task of grouping data based on similarity. A standard k-means algorithm groups data by firstly assigning all data points to the closest clusters, then determining the new cluster mean of each cluster based on the average value of its members. The algorithm repeats these two steps until it converges; that is until there is no change in cluster means and cluster assignment among data points. Performance of the algorithm depends on the number of data points, number of data clusters, and number of iterations. To speed up the clustering process, we develop the density-biased reservoir sampling algorithm as an efficient data reduction technique. Instead of simply randomly selecting data for clustering, we evaluate data density and draw samples from the dense area. The proposed algorithm has been implemented with a functional programming paradigm using the Erlang language. The declarative style of Erlang facilitates a rapid prototyping. Our experimental results reveal the effectiveness of drawing samples of high density from the Zipf distributed data. The shift of cluster means is minimal, whereas the decrease in memory usage is significant. The proposed density-biased reservoir sampling technique thus shows a great potential on dealing with large and streaming data. Moreover, the Erlang language itself has an important feature of concurrent programming on a multi-core architecture that can help speed up the computation of large and continuously generated data.