دراسة وتطوير طريقة معالجة تفرعية لحذف البيانات المكررة في النظم الموزعة

د.م. عمار علي زقزوق

أستاذ مساعد في قسم هندسة التحكم الآلي والحواسيب

كلية الهندسة الميكانيكية والكهربائية

جامعة البعث

ملخص البحث

البيانات الضخمة هي مجموعة كبيرة من البيانات المعقدة التي تم تجميعها من مصادر مختلفة، وهذا التعقيد يأتي من كون البيانات ترد من مصادر مختلفة وبكمية كبيرة، وكلما زاد حجم البيانات وتعقيدها، كلما قلت كفاءة أدوات إدارة قواعد البيانات العادية أو تطبيقات معالجة البيانات التقليدية في معالجة هذه البيانات وقل الأداء. نظام Hadoop هو نظام مفتوح المصدر لا يعتمد التفرع فقط في معالجة البيانات، وإنما يتصف بخاصية التخزين الموزع والتي تمكننا من تخزين ملف كبير على أجهزة حاسوبية عدة ومعالجة بيانات الملف الموزعة على العقد الحاسوبية وكأنها ملفاً واحداً. مع ازدياد كمية البيانات يوماً بعد يوم، فمن الأفضل الحفاظ على بيانات قليلة أو حتى معدومة إن كانت بدون فائدة، وذلك لحفظ مساحة التخزين وزيادة سرعة معالجة البيانات.

نقدم في هذا البحث تقنية جديدة لإزالة الصور المتكررة من مجموعة كبيرة من الصور والحفاظ على البيانات الفريدة في نظام التخزين. وقد تم استخدام تقنية MD5 القائمة على تجزئة البيانات وتطبيقها لإلغاء البيانات المكررة في نظام Hadoop لتوفير المساحة التخزينية. كما تم تطبيق عملية إزالة التكرار على مستوى البايت، والتي تحقق كفاءة عالية بالكشف عن البيانات المكررة وحذفها، للحصول على البيانات الفريدة.

كلمات مفتاحية: البيانات الضخمة، المعالجة الموزعة، التخزين الموزع، خوارزمية MD5، إلغاء البيانات المكررة، Hadoop.

1- المقدمة (Introduction):

بسبب النمو المتسارع في استخدام التكنولوجيا الناشئة مثل الحوسبة السحابية والبيانات الضخمة، فإن معدل نمو البيانات يتزايد بشكل متسارع، وبسبب التطور السريع لكل مصادر البيانات، فإننا نحو توسع سريع للبيانات وفق الجدول (1).

وحدة القياس	الحجم
Byte	8 Bit
Kilo Byte	1024 Byte
Mega Byte	1024 Kilo Byte
Giga Byte	1024 Mega Byte
Tera Byte	1024 Giga Byte
Peta Byte	1024 Tera Byte
Exa Byte	1024 Peta Byte
Zetta Byte	1024 Exa Byte
Yotta Byte	1024 Zetta Byte
Xenotta Byte	1024 Yotta Byte
Shilentno Byte	1024 Xenotta Byte
Domegemegrotte Byte	1024 Shilentno Byte

الجدول (1): واحدات قياس حجم البيانات.

تستثمر العديد من المؤسسات الكثير من الأموال لتخزين مثل هذه البيانات الضخمة للنسخ الاحتياطي، لكن حل النسخ الاحتياطي التقليدي لا يوفر أية وسيلة لمنع النظام من تخزين البيانات المكررة، ما يزيد من تكلفة التخزين ووقت النسخ الاحتياطي، والذي يقلل بدوره من أداء النظام. تعد تقنية إلغاء البيانات المكررة (Deduplication) تقنية ضغط جديدة تمنع تخزين البيانات المتماثلة على أجهزة التخزين، وبالتالي تمكن من الاستخدام الفعال لمساحة التخزين.

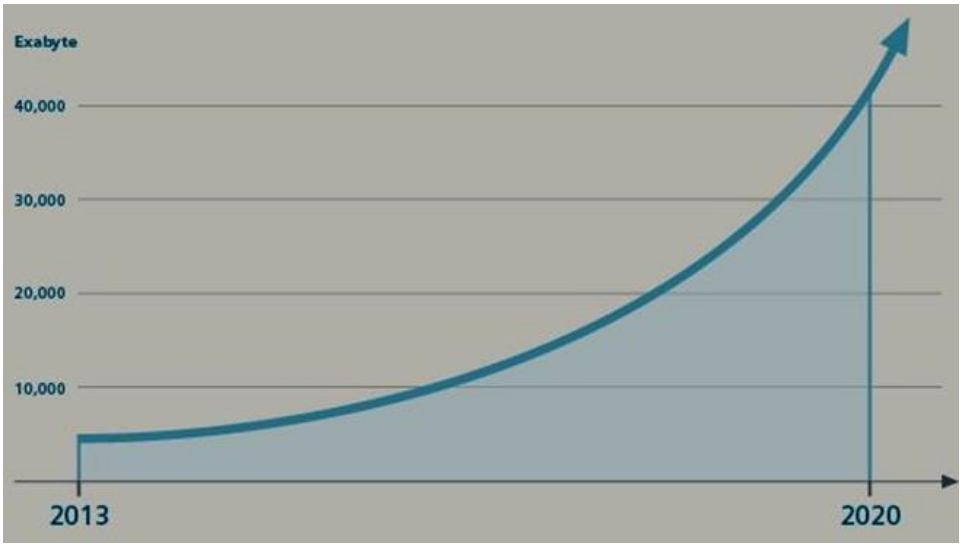
1.1- البيانات الضخمة:

1.1.1- ظهورها:

حوالي 90% من البيانات المتوفرة حالياً تكونت في السنين القليلة الماضية، فالبيانات التي تم توليدها حتى عام 2003 كانت حوالي 5 بليون غيغا بايت، وهذه الكمية نفسها من

البيانات أصبحت تتولد كل يومين في عام 2011، وكل عشر دقائق في عام 2013، وهذا المعدل في ازدياد مستمر وبصورة كبيرة. قامت شركة البيانات العالمية (International Data Corporation) بدراسة أوردت فيها أنه من عام 2005 وحتى عام 2020 سينمو العالم الرقمي بمعدل 300 ضعفاً، أي من 130 Exa Byte حتى 40000 Exa Byte. ومن المتوقع أن يتضاعف حجم البيانات كل سنتين حتى العام 2020، وأن يستفاد من 33 بالمائة من هذه البيانات الرقمية إذا تم تحليلها، مقارنة بنسبة 25% من البيانات المستفاد منها اليوم.

يبين الشكل (1) مخطط زيادة حجم البيانات، إذ من المتوقع أن يصبح حجم البيانات 44 Zeta Byte في العام 2020.



الشكل (1): مخطط زيادة حجم البيانات.

2.1.1- خصائصها:

البيانات الضخمة لها ثلاث صفات هي:

1- الحجم: يقاس حجم البيانات بشكل عام بوحدات القياس المعطاة في الجدول (1)، وكلما كان حجم البيانات كبيراً، كلما كانت الفائدة أكبر في تحليلها والعثور على معلومات مفيدة منها.

2- التنوع: تأتي البيانات من مصادر متنوعة وفي أشكال مختلفة، فهي قد تكون نصوصاً أو أصواتاً أو فيديوهات أو صوراً أو بيانات واردة من أجهزة رصد وحساسات، والكثير من المصادر.

3- التسارع: هو معدل تدفق البيانات التي قد ترد على شكل حزم (Batch)، وقد ترد أسبوعيا أو يوميا أو كل ساعة أو كل دقيقة أو كل ثانية أو كل ميلي ثانية.

3.1.1- مصادرها:

البيانات الضخمة تضم كل ما يتم انتاجه عبر الأجهزة الالكترونية والميكانيكية والآلات والتطبيقات المختلفة، وقد تختلف نوعية البيانات حسب نوع المصدر. ويندرج تحت البيانات الضخمة البيانات التجارية والصحية والاجتماعية والعلمية والسياسية وغيرها. فيمايلي أمثلة لبعض مصادر البيانات:

1- بيانات الصندوق الأسود في الطائرات: يقوم بالتقاط أصوات طاقم الطائرة وتسجيل المحادثات وأداء طاقم الطائرة.

2- بيانات شبكات التواصل الاجتماعي: إذ يقوم ملايين المستخدمين بنشر معلومات وتعليقات ووجهات نظر حول مواضيع مختلفة، وينتج عن ذلك كمية ضخمة من البيانات اليومية.

3- بيانات تبادل السلع: تشمل هذه البيانات معلومات حول البيع والشراء التي يقوم بها عملاء الشركات المختلفة.

4- بيانات شبكات الكهرباء والمياه: معلومات عن استهلاك نقطة معينة اعتماداً على محطة معينة.

5- بيانات النقل: بيانات عن أنواع الناقلات مثل موديلها وسعتها والمسافة التي تقطعها وتوفرها.

6- بيانات محركات البحث: تقوم محركات البحث باستخراج كثير من البيانات من قواعد بيانات مختلفة.

7- مشاريع دراسة الظواهر الطبيعية: يتم توليد كمية ضخمة من البيانات من التجارب التي تُجرى، مثلاً ينتج مشروع مصادم الهدرونات الكبير حوالي 15 بيتابايت يومياً.

8- الحساسات: تلتقط الحساسات المرتبطة بالكثير من الأجهزة كمية كبيرة من البيانات.

4.1.1- فوائدها:

تتمثل فوائد البيانات الضخمة في نقاط عدة منها:

1- اتخاذ القرارات الصحيحة: تستطيع أية جهة سواءً أكانت حكومية أم خاصة أم ربحية أم تطوعية توفير الكثير من المال بالاستفادة من البيانات الضخمة في التخطيط السليم والصائب لقراراتها وخططها المستقبلية.

2- زيادة المبيعات: تستطيع الشركات المنتجة والمسوقون الاستفادة من المعلومات المحفوظة في شبكات التواصل الاجتماعي لمعرفة مدى الاستجابة لعروضها وحملاتها الإعلانية، وغيرها من الأمور التي تساعدها في التخطيط لمنتجاتها وزيادة مبيعاتها.

3- خدمات صحية أفضل: يمكن الاستفادة من بيانات المرضى المسجلة بالمشفى مثل السجل الطبي السابق للمريض في توفير خدمات أسرع وأفضل للمرضى.

4- تقليل تكلفة الإعلانات: تستطيع أية جهة إرسال إعلانات مخصصة للعملاء حسب اهتماماتهم موجهة لهم شخصياً عبر ما يسمى بالاستهداف الجزئي.

5.1.1- أنواعها:

يمكن تصنيف البيانات إلى ثلاثة أنواع رئيسة هي:

1- البيانات المهيكلة: مثل البيانات العلائقية.

2- البيانات شبه المهيكلة: مثل بيانات XML.

3- البيانات غير المهيكلة: مثل ملفات Word و PDF و Text.

من التحديات التي تجابه التعامل مع البيانات الضخمة هذا التنوع والاختلاف، فالتطبيقات والأدوات التقليدية صممت لتتعامل مع نوع واحد من البيانات.

2.1- الدراسات السابقة:

📖 K.A. Ramya [1]: قام هذا الباحث بإجراء دراسة حول طرائق ضغط البيانات ودورها في تقليل مساحة التخزين، حيث يتم حفظ البيانات الكبيرة في مساحة تخزين محدودة، إذ تعمل هذه التقنية على إزالة البيانات المكررة من المستوى الثنائي. يتم اتباع طرائقاً عدة في ضغط البيانات منها الآتي:

1- Lossless: لا تؤدي إلى فقدان المعلومات، وبالإمكان استرجاع الملفات الأصلية بعد ضغطها، فبعد فك الضغط نحصل على نسخة مطابقة للملفات الأصلية. لذلك فهي تعتبر مناسبة جداً للملفات النصية.

2- Lossy: تؤدي إلى فقدان بعض معلومات الملفات الأصلية، كأن نقوم بإزالة البكسلات ذات الدقة العالية في صورة، وبالتالي عند فك الضغط نحصل على نسخة مشابهة للملفات الأصلية وليست مطابقة لها، ولكن هذه الطريقة غير مناسبة للملفات النصية، لأن كل محرف في النص مهم ويؤثر على المحتوى بشكل مباشر.

كما تطرق الباحث إلى التقنيات المستخدمة في كلتا الطريقتين ومزايا كل منهما.

📖 Amarasinghe & Kodituwakku [2]: قام هذان الباحثان بدراسة البارامترات المتعلقة بقياس أداء تقنيات الضغط كمعدل الضغط ومعامل الضغط ونسبة مساحة التخزين المحفوظة وغيرها من البارامترات، واستخدما هذه البارامترات في مقارنة تقنيات الضغط.

📖 Kailas Devadkar & Supriya Milind More [3]: قدم هذان الباحثان نموذجاً يبين التقنيات المختلفة لإزالة البيانات المكررة:

1- استنادا على طريقة التقطيع (الكتل):

- التقطيع على مستوى الملف: لا يقسم الملف إلى كتل أصغر، وإنما يعتبر ملفاً كاملاً كقطعة واحدة، وبالتالي في هذا الأسلوب يتم إنشاء قيمة فهرس (دليل) وحيد فقط للملف بأكمله. هذه الطريقة، تفشل عندما يكون هناك تغيير طفيف في بيانات الملف، لأنها سوف تولد فهرساً للملف بأكمله بدلاً من توليد فهرس للبيانات التي تم تغييرها.
- التقطيع على مستوى الكتلة:

• التقطيع بحجم ثابت: يتم تقسيم الملف إلى كتل بحجم ثابت، تحل هذه الطريقة مشكلة التقطيع على مستوى الملف، لأنها تولد مؤشراً للتغييرات وليس للملف بأكمله، ولكن مع ملف كبير سيكون هناك عدد كبير من القطع، وبالتالي سيتطلب مساحة تخزينية أكبر.

• التقطيع بحجم متغير: يتم تقسيم الملف إلى كتل بحجم متغير، وبالتالي تحل هذه الطريقة مشكلة التقطيع بحجم ثابت.

2- استناداً على موقع إلغاء البيانات المكررة:

- إلغاء البيانات المكررة المستندة إلى المصدر: هنا يتم التخلص من البيانات المكررة قبل الإرسال إلى الجهاز الهدف، تقلل هذه الطريقة من النطاق الترددي لأنها تحذف البيانات المكررة قبل الإرسال، ولكن تحقق فعالية أقل في إزالة البيانات المكررة.
- إلغاء البيانات المكررة المستندة إلى الهدف: هنا يتم التخلص من البيانات المكررة في الجهاز الهدف، وتزيد هذه الطريقة من النطاق الترددي، لأن البيانات المكررة تصل مع البيانات المراد تخزينها، ولكنها تحقق فعالية كبيرة في إزالة البيانات المكررة.

3- استناداً على وقت إلغاء البيانات المكررة:

- إلغاء البيانات المكررة المضمنة: يتم تخزين البيانات أولاً في جهاز العميل، ثم يتم إزالة البيانات المكررة، وبالتالي ترسل فقط البيانات الفريدة إلى الهدف.
- إلغاء البيانات المكررة في مرحلة ما بعد المعالجة: يتم تخزين كل البيانات في الجهاز الهدف، ثم يتم إزالة البيانات المكررة.

📖 K. Siva Sanka & A. Venish [4]: قام هذان الباحثان بتعريف عملية إلغاء البيانات المكررة على أنها مكونة من أربع مراحل:

1- التقطيع: تقسيم البيانات الواردة إلى أجزاء.

2- التجزئة: إنشاء معرف خاص بكل جزء.

3- التطابق: مقارنة كل معرف لكل كتلة مع المعرفات المخزنة في جدول المعرفات.

4- الحذف أو التخزين: إذا كان هناك تطابق فالكتلة مكررة، ولا يتم تخزينها، ويجري تخزنها في حال عدم التطابق مع أحد المعرفات.

📖 Ruey-Kai Sheu وآخرون [5]: قاموا بتصميم نظام لإلغاء البيانات المكررة. اعتمد هذه التصميم على عملية إلغاء البيانات المكررة على مستوى الملف، أي تم توليد مفتاح مختزل للملف وتخزينه في نظام التخزين الموزع لـ Hadoop، ففي حال تكررت

عملية تخزين الملف نفسه مرة أخرى لن يتم تخزينه، وبالتالي حققت فعالية كبيرة في إلغاء البيانات على مستوى الملف. كما اعتمدوا على خوارزمية SHA2-512، والتي تحقق أماناً كبيراً في إرسال البيانات من الجهاز العميل إلى نظام تخزين Hadoop.

2- مشكلة البحث:

أثناء إدارة عمليات الملفات وتنفيذها على الحاسوب أو على أجهزة التخزين الأخرى، يمكن جمع العديد من الملفات المكررة ذات الحجم الكبير. إن تراكم هذه المستويات الرقمية غير المرغوب فيها يمكن أن يكون السبب الرئيس لنقص مساحة التخزين وانخفاض في أداء الحاسوب. لذلك، نحتاج إلى البحث عن الملفات المكررة ومحوها من القرص الصلب. ولكن، هل سيكون من السهل على أي شخص البحث عن جميع الملفات المكررة؟ وماذا لو لم يكن لدينا أية معلومات حول هذه الملفات التي لديها أيضاً نسخة طبق الأصل؟ هل سنقوم بالبحث عن نسخة مكررة من كل ملف واحد تلو الآخر؟ لأن هذا يحتاج إلى الكثير من الوقت. إذا تحدثنا بشكل خاص عن مستخدمي الحاسوب، فيجب أن يعرفوا كيف يمكن أن يؤثر الوصول إلى الملفات المكررة على وظائفهم! فوجود عدد كبير من الملفات المكررة يؤدي إلى أن أخذ نظام التشغيل وقتاً إضافياً للبحث عن ملف معين وتحديده. وبالتالي، بسبب حجم البيانات المتزايد وصعوبة تحديد ما هي الملفات المكررة، وعدد النسخ الموجودة منها، لا بد من البحث عن طرائق فعالة لحل هذه المشكلة.

3- هدف البحث:

عندما يريد مستخدم أو مستخدمون عدة في شركة ما بتخزين ملف كبير بحجم وليكن 5 Tera Byte، وهل من الممكن تخزين هذا الملف على حاسوب واحد بالنسبة للمستخدم؟ وبالنسبة لمستخدمي الشركة، إذا قمنا بتقسيم الملف إلى أجزاء عدة وتخزين هذه الأجزاء في أقسام عنقود (Cluster)، هل من الممكن الحفاظ على التنسيق بين أجزاء هذا الملف؟ من هنا أتت فكرة التخزين الموزع في نظام Hadoop التي كانت حلاً فعالاً لمشكلة تخزين البيانات الكبيرة ومعالجتها. إذ يقوم نظام Hadoop الموزع بتقسيم الملف إلى أجزاء وتخزينها في الأجزاء الموجودة، ولكن يتعامل معها كأنها ملفاً واحداً، وبالتالي يمكن إجراء عمليات على الكتلة الواحدة من الملف أو على الملف بأكمله. كما

يقوم افتراضياً بنسخ كل جزء من الملف ثلاث نسخ في الأجهزة الموجودة، ما يؤمن حماية البيانات الموجودة والحفاظ عليها. لكن نحن بصدد عدد كبير من البيانات، وبالنتيجة استغلال البيانات المفيدة هو الهدف الأهم. فلو فرضاً أن هذا الملف ذو حجم 5 Tera Byte يحوي ملفات مكررة بحجم حوالي 1 Tera Byte، أي يحوي ملفات غير مفيدة تأخذ فراغاً من مساحة التخزين، عوضاً عن أن نظام Hadoop يقوم بعملية النسخ الاحتياطي للبيانات، ما دعا إلى العمل لاكتشاف تقنيات تعمل بسرعة وفعالية عالية لحذف الملفات المتكررة، الأمر الذي وجهنا للقيام بهذا البحث الذي يحقق في مضمونه أهدافاً عدة، وهي:

1- الاستفادة الكاملة من مساحة التخزين عن طريق تخزين المعلومات الفريدة وحذف المتكررة.

2- تحسين أداء النظام الذي يرتبط ارتباطاً موثوقاً بمساحة التخزين.

3- سهولة إدارة الملفات وهيكلتها وسرعة البحث عن المعلومات من قبل المستخدم.

4- نظام Hadoop البرمجي:

أصبح اسم Hadoop مرادفاً للبيانات الضخمة، فهو برنامج مفتوح المصدر بإطار تخزين موزع لمجموعات كبيرة من البيانات في تجمعات حواسيب (Computer Clusters)، ما يعطيه خاصية التوسعية (Scalability) والاعتمادية (Reliability)، والعمل دون القلق من فشل الأجهزة. ولديه المقدرة على تخزين بيانات كبيرة من جميع أنواع البيانات، مع القدرة على التعامل مع وظائف متزامنة لا حدود لها.

1.4- خصائصه:

يتميز Hadoop بخصائص عدة منها:

1- قليل الكلفة: يمكنك بناء تجمع هدوب من جهاز واحد أو أجهزة قليلة، ثم كلما احتجنا إلى مساحة ومعالجة أكبر يمكننا إضافة أجهزة أخرى دون أن يؤثر ذلك على النظام (خاصية التوسعية).

2- معالجة الأعطال تلقائياً: بما أن Hadoop يستخدم عدداً كبيراً من الأجهزة، ويكرر نسخ البيانات في هذه الأجهزة (Replication)، فلن يتسبب تعطل بعض الأجهزة في

إيقاف العمل، فإذا تعطل جهاز سيقوم Hadoop تلقائياً بتكرير نسخ البيانات التي كانت بالجهاز المعطل (خاصية الاعتمادية).

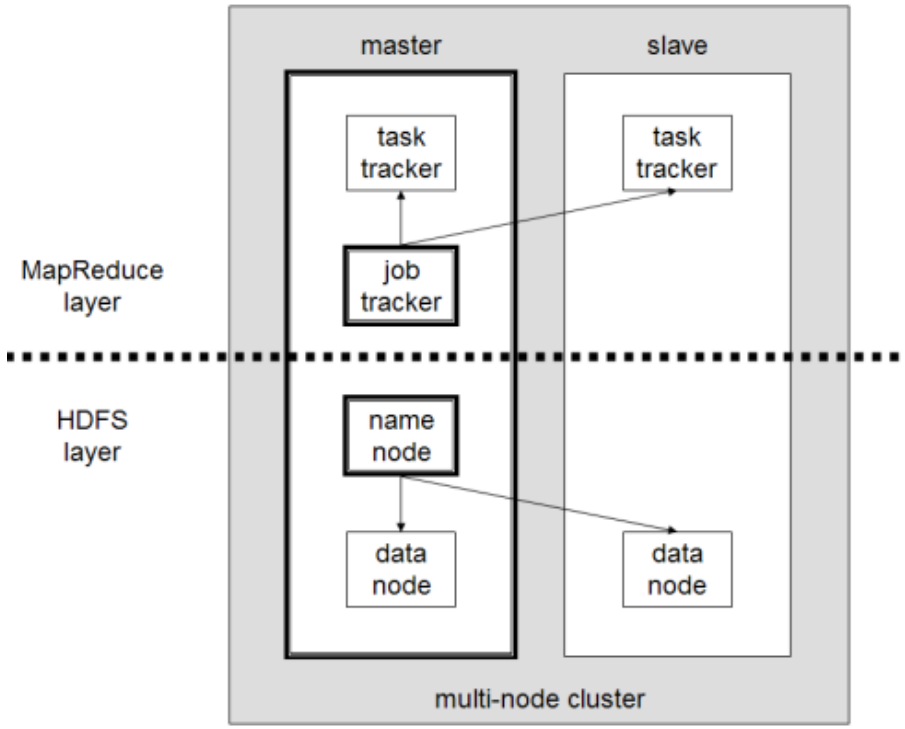
3- سريع في تنفيذ العمليات: يقوم Hadoop بمعالجة البيانات في أماكن تواجدها بأجهزة التجمع، إذ يقوم كل حاسوب في التجمع بمعالجة البيانات المخزنة فيه، ويتم كل ذلك في وقت واحد.

2.4- مكوناته الأساسية:

يتكون Hadoop من مكونين أساسيين، الشكل (2)، هما:

1- HDFS (Hadoop Distributed File System): يستخدم كنظام ملفات لتخزين البيانات في أوعية تخزينية موزعة.

2- Map Reduce: بيئة برمجة تستخدم المعالجة المتوازية في معالجة البيانات.



الشكل (2): عنقود Hadoop متعدد العقد.

5- النظام المقترح:

قبل البدء بوضع الخوارزمية المقترحة، لا بد من التعريف بالبرمجيات المستخدمة:

1- VMWare Workstation Pro: برنامج الآلة الافتراضية.

2- Linux: نظام تشغيل إصدار Ubuntu 16.04.

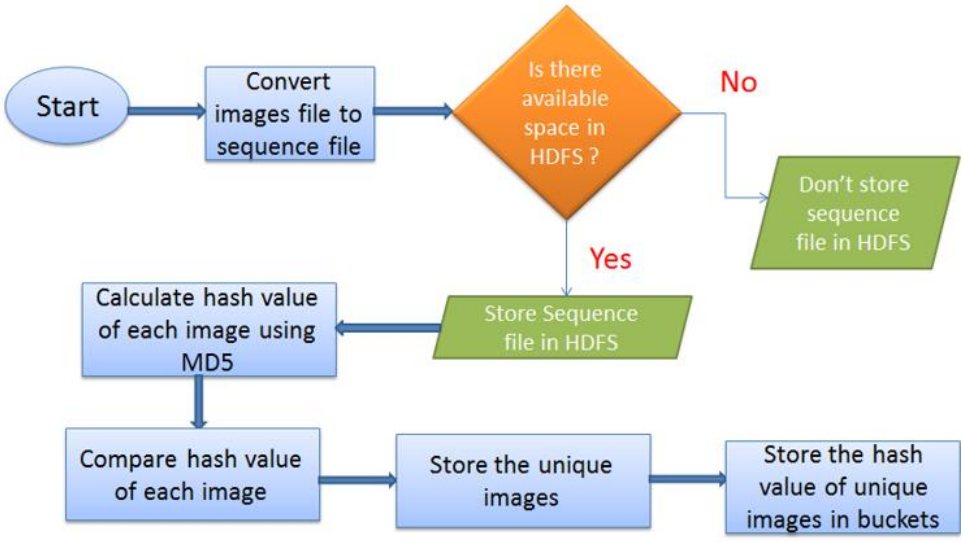
3- Hadoop 2.7.3.

4- Java 8 Oracle: البيئة المستخدمة في كتابة البرامج.

5- جهاز Namenode افتراضي: يمثل العقدة الرئيسة في نظام Hadoop، والتي تكون متوفرة على جميع المعلومات المتعلقة بالأجهزة الفرعية كالمساحة الكلية والمساحة الموفرة واسم الملف المراد تخزينه وامتداده، وكذلك المواقع المتاحة في Datanode لتخزين البيانات فيها.

6- جهاز Datanode افتراضي: يمثل العقدة التي يتم تخزين البيانات فيها.

يبين الشكل (3) الخوارزمية المتبعة في عملية إلغاء البيانات المكررة.



الشكل (3): الخوارزمية المتبعة في عملية إلغاء البيانات المكررة.

I- مرحلة تحويل ملف الصور إلى ملف تسلسلي: الملف التسلسلي هو ملف مبسط يقوم بتخزين البيانات على شكل أزواج (مفتاح، قيمة). فلو قمنا بعملية معالجة لكل صورة، والتي تعتبر كملف، فذلك سيستغرق وقتاً طويلاً، فالملف التسلسلي يحل هذه المشكلة عن

طريق تخزين جميع الصور في ملف تسلسلي واحد، وكل صورة تمثل على شكل (مفتاح, قيمة). يشكل المفتاح اسم الصورة، والقيمة تشكل (محتوى الصورة) البيانات الثنائية للصورة. عند المعالجة، سيتم تقسيم الملف المتسلسل بوساطة Mapreduce لتوليد المفتاح المختزل لكل صورة.

II- مرحلة تخزين الملف التسلسلي في نظام التخزين الموزع HDFSK والذي يشغل مساحة أقل من تخزين الصور في نظام Hadoop.

III- مرحلة map: يتم في هذه المرحلة تطبيق خوارزمية MD5 لإيجاد المفتاح المختزل.

تستخدم هذه الخوارزمية في تشفير كلمات المرور في أنظمة التشغيل وقواعد البيانات. تعتبر المصداقية من أهم أسباب ابتكار هذه الخوارزمية، إذ عن طريق نتيجة التشفير باستخدام هذه الخوارزمية نعرف هل البيانات المستقبلة صحيحة أم خاطئة؟ يقوم مبدأ عمل هذه الخوارزمية على استقبال بيانات بحجم 512 بت وتوليد مفتاح اختزال بطول 128 بت، يرسل مع الرسالة وعند استقبال الرسالة من قبل المستقبل يقوم بتوليد مفتاح اختزال للرسالة نفسها، فإذا تطابق المفتاحان، عندها تكون البيانات صحيحة.

يوضح الشكل (4) مخطط آلية عمل خوارزمية MD5.

1- إضافة بتات الحشو: في حال كان حجم البيانات المراد تطبيق خوارزمية MD5 عليها أقل من 512 بت، يتم إضافة 1 يتبع بأصفار عدة، حتى يصبح طول الرسالة مساوياً لـ 448 بت، أي أقل من 512 بت بـ 64 بت.

2- إلحاق طول الرسالة: إلحاق طول الرسالة بـ 64 بت التي تشكل المرتبة الدنيا من الرسالة الأصلية حتى يصبح طول الرسالة 512 بت.

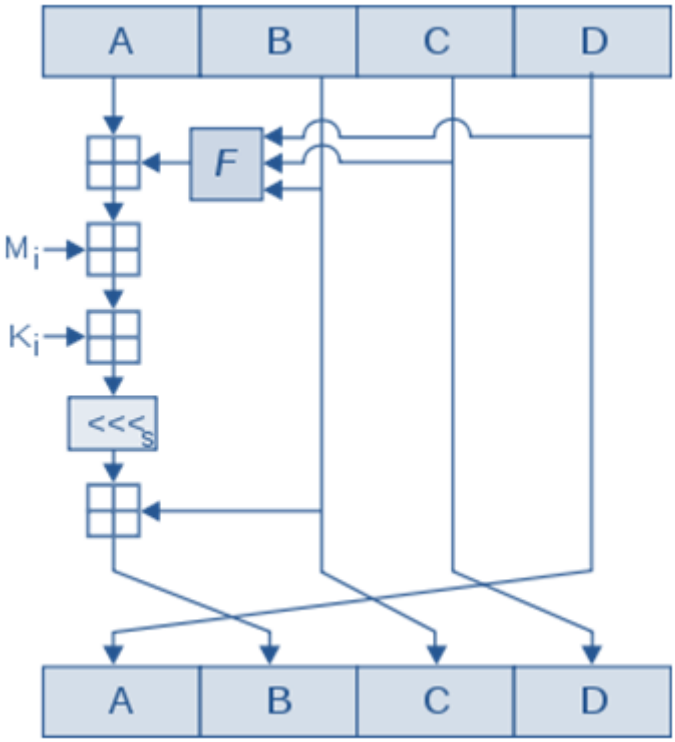
3- تهيئة حافظات MD5: تعريف بطول 4 كلمات (four-word buffer) طول كل واحدة منها 32 بت، تُعرّف مسبقًا بالقيم الآتية:

(A: 01 23 45 67)، (B: 89 ab cd ef)، (C: fe dc ba 98)، (D: 76 54 32)

4- التنفيذ: تقسيم الرسالة إلى أربع علب حجم كل علبة 128 بت، تشكل كلمات العلبة الأولى كلمات المرحلة الأولى، وكلمات العلبة الثانية كلمات المرحلة الثانية، وهكذا. ثم

نُعرّف 4 دوال مساعدة تأخذ كل منها مدخلاً مكوناً من 3 كلمات، كل كلمة عبارة عن 32 بت، وتُخرج كلمة واحدة مكونة من 32 بت أيضاً:

$$F(X,Y,Z) = XY \text{ v not}(X)\ Z)$$
$$G(X,Y,Z) = XZ \text{ v } Y \text{ not}(Z)$$
$$H(X,Y,Z) = X \text{ xor } Y \text{ xor } (Z)$$
$$I(X,Y,Z) = Y \text{ xor } (X \text{ v not}(Z))$$



الشكل (4): مخطط آلية عمل خوارزمية MD5.

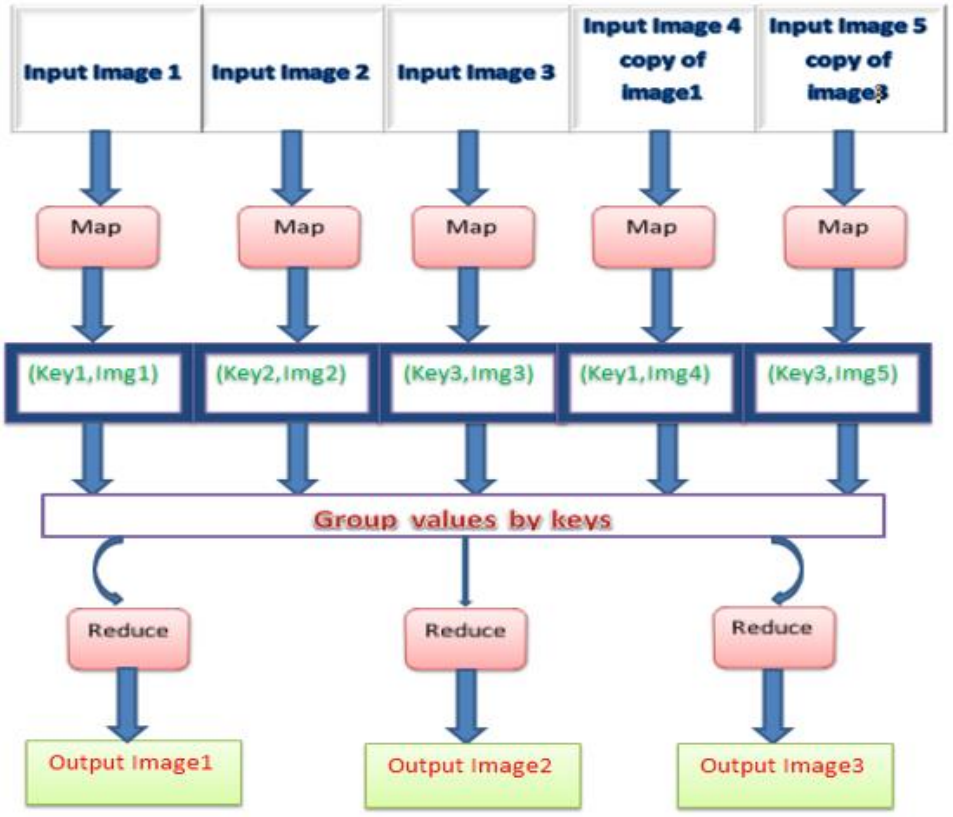
5- توليد الخرج: الحصول على المفتاح المختزل بطول 128 بت.

وبالتالي، سيتم تطبيق خوارزمية MD5 على البيانات الثنائية للصورة في مرحلة map، ويكون خرج مرحلة map مجموعة من القيم، حيث المفتاح يمثل المفتاح المختزل المولد عن طريق خوارزمية MD5، والقيمة تمثل اسم الصورة.

IV- مرحلة Reduce: تستقبل هذه المرحلة خرج المرحلة map وتطابق المفتاح لكل القيم. في حال وجود تطابق في المفتاح، والذي يمثل المفتاح المختزل من MD5،

وبالتالي يوجد تكرار بين الصور، ومن ثم سيتم نسخة واحدة من الصور في نظام Hadoop وتجاهل البقية، وبالنتيجة الحصول على الصور الفريدة، يتم أيضاً حفظ مفتاح الاختزال لكل الصور الفريدة في Buckets، ففي حالة تخزين صور أخرى في نظام التخزين، يتم مقارنتها مع المفاتيح المخزنة للتأكد من التكرار.

يبين الشكل (5) مخطط آلية عمل Map Reduce.



الشكل (5): مخطط آلية عمل Map Reduce.

6- النتائج ومناقشتها:

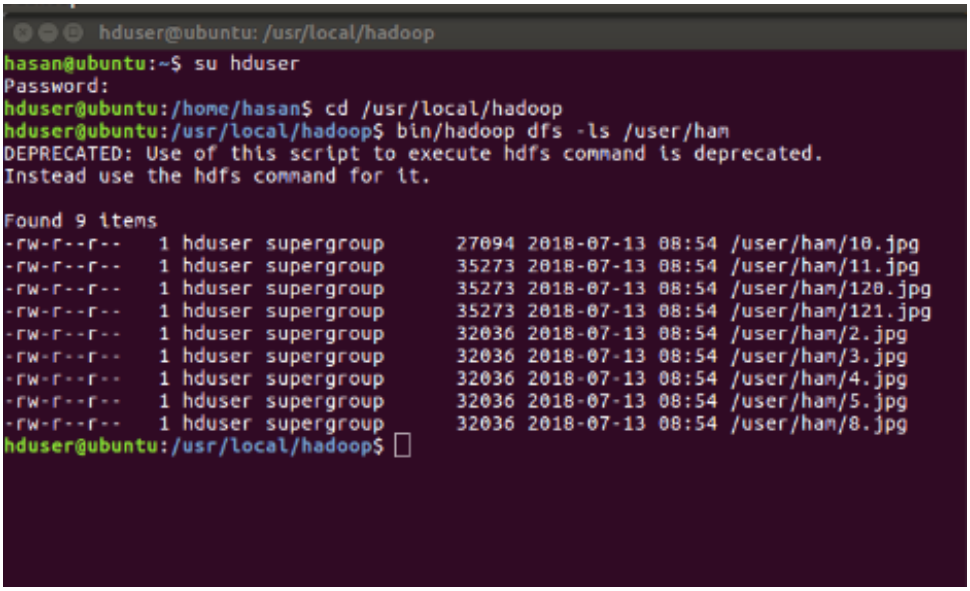
يتضمن النظام المقترح تطبيق عملية إلغاء البيانات المتكررة على ملف صور من صيغة jpeg يحوي نسخة متكررة عن كل صورة، ومن ثم الحصول على الصور الفريدة.

يظهر الشكل (6) الملف المخزن في نظام Linux قبل تطبيق عملية إلغاء البيانات المكررة.

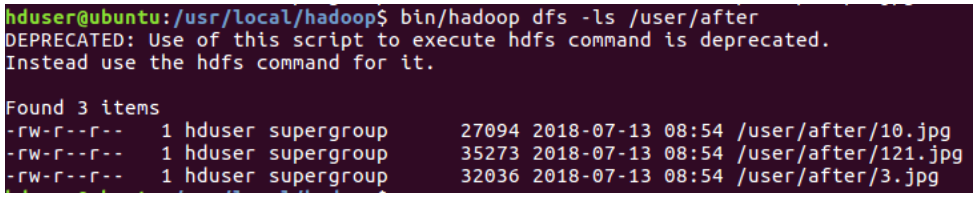
كما نلاحظ أن هذا الملف يحوي 9 صور من بينها الصور المكررة. بعد تطبيق الخوارزمية المتبعة نحصل على الملف المبين في الشكل (7).

وبالتالي، هناك ثلاث صور فريدة فقط.

إن حجم الملف قبل عملية إلغاء البيانات المكررة: 286 KB، وبعد عملية إزالة الملفات المتكررة هو: 92 KB.



الشكل (6): الملف المخزن في نظام Linux قبل تطبيق عملية إلغاء البيانات المكررة.



الشكل (7): الملف المخزن في نظام Linux بعد تطبيق عملية إلغاء البيانات المكررة.

تطرق الباحثان Kodituwakku و Amarasinghe في بحثهما عن تقنيات الضغط على أهم البارامترات، وهو معدل الضغط الذي يعطى بالعلاقة (1).

$$compression\ Ratio = \frac{size\ after\ compression}{size\ before\ compression} \qquad (1)$$

أي هو النسبة بين حجم الملفات بعد ضغطها وحجمها قبل عملية الضغط، وذلك لحساب فعالية عملية الضغط في تحسين التخزين. وبالتالي، لو قمنا بعملية ضغط للملف الذي يحتوي على البيانات المكررة سيكون حجم الملف المضغوط: 277 KB، أي يكون معدل الضغط وفق الآتي:

$$compression\ Ratio = \frac{277}{286} = 0.968$$

وكلما كان معدل الضغط قريباً من الصفر كلما دل على فعالية أكبر. وبالمقارنة مع ذلك، يمكن حساب معدل إلغاء البيانات المكررة وفق الآتي:

$$Deduplication\ Ratio = \frac{size\ after\ deduplication}{size\ before\ deduplication} = \frac{92}{286} = 0.321$$

أي أن معامل إلغاء البيانات المكررة للنموذج المقترح أفضل من معامل الضغط، وبالتالي توفير مساحة تخزينية أكبر.

تُحسب النسبة المئوية للمساحة التخزينية التي تم توفيرها بالعلاقة (2).

$$saving\ percentage = \frac{size\ before\ compression - size\ after\ compression}{size\ before\ compression} \% \quad \textbf{(2)}$$

وبالتالي، يمكن حساب المساحة التخزينية التي تم توفيرها عن طريق ضغط الملف وفق الآتي:

$$saving\ percentage = \frac{286-277}{286} \% = 0.3146\ \%$$

وهي نسبة صغيرة.

باستخدام تقنية إلغاء البيانات المكررة فإن المساحة التخزينية التي توفيرها تعطى بالعلاقة (3).

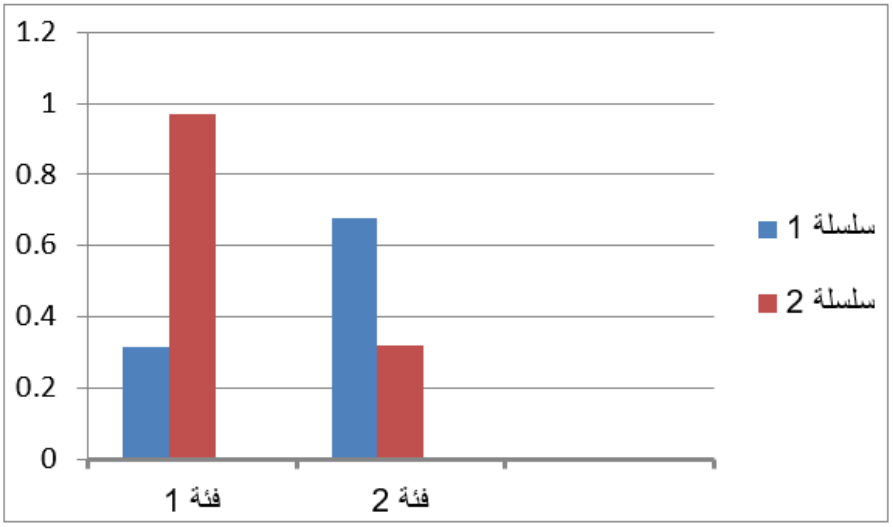
$$saving\ percentage = \frac{size\ before\ compression - size\ after\ compression}{size\ before\ compression} \% \quad \textbf{(3)}$$

وبالتالي:

$$saving\ percentage = \frac{286-92}{286} \% = 0.6783\%$$

وهذا يعني أن التقنية المقترحة لها فعالية أكبر في توفير مساحة تخزينية من تقنية الضغط.

يبين الشكل (8) مخطط فعالية كل من تقنيتي الضغط وتقنية إلغاء البيانات المكررة المقترحة، حيث تمثل الفئة الأولى تطبيق تقنية الضغط، والفئة الثانية التقنية المقترحة. السلسلة الأولى تمثل المساحة التخزينية التي تم توفيرها حسب كل تقنية، والسلسلة الثانية تمثل معدل الضغط ومعامل إلغاء البيانات المكررة.

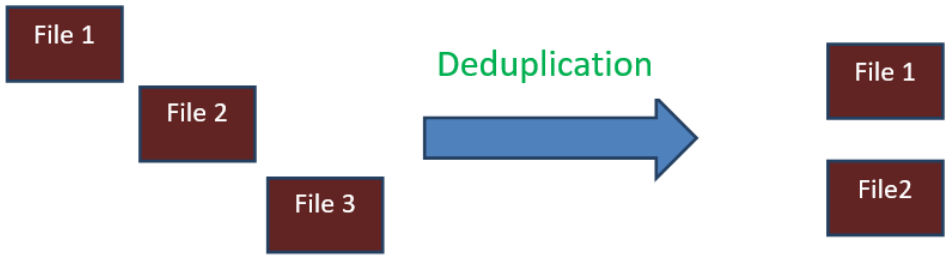


الشكل (8): مخطط فعالية كل من تقنيتي الضغط وتقنية إلغاء البيانات المكررة المقترحة.

حسب دراسة الباحثان Supriya Milind More و Kailas Devadkar، فإن عملية إلغاء البيانات على مستوى الكتل بعد تقسيم الملف إلى كتل بحجم ثابت أو متغير ذات فعالية أكبر من إلغاء البيانات على مستوى الملف، لأن عملية البحث عن التكرار ستكون أوسع لكنها تتطلب وقتاً أكبر في المعالجة، ففي حالة إلغاء التكرار على مستوى الملف سيتم المقارنة بين ملفين على أساس مفتاح التجزئة لكل ملف، وبالتالي في حال إعادة تخزين ملف يحوي بيانات كثيرة متطابقة مع ملف مخزن، ولكن الاختلاف في بيانات قليلة فسوف يتم تخزين الملف، لأن المطابقة في عملية إلغاء التكرار تتم على مستوى الملف، أي المطابقة بين الملفات، أما على مستوى الكتل فسوف يتم المطابقة بين الملفات على أساس أجزاء هذه الملفات. ولكن عملية المطابقة في نموذجنا المقترح، قمنا بإزالة التكرار

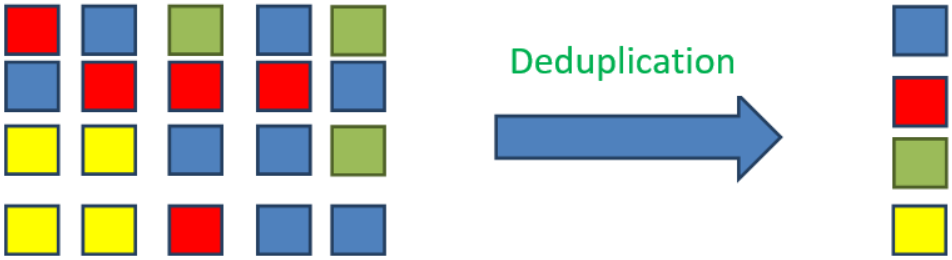
على مستوى البيانات الثنائية (على مستوى البايت)، أي على أساس كامل البيانات في الملف، أي هذه التقنية ذات فعالية أكبر في إزالة التكرار من التقنيتين السابقتين، ولكن على حساب الزيادة في وقت التنفيذ.

يظهر الشكل (9) كيفية إزالة البيانات المتكررة على مستوى الملف.



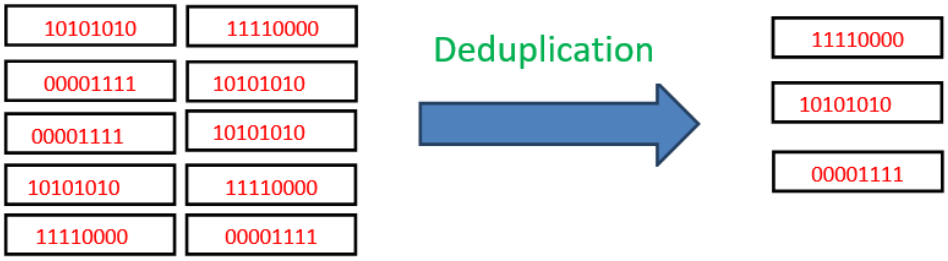
الشكل (9): كيفية إزالة البيانات المتكررة على مستوى الملف.

يعرض الشكل (10) كيفية إزالة البيانات المتكررة على مستوى الكتل.



الشكل (10): كيفية إزالة البيانات المتكررة على مستوى الكتل.

يبين الشكل (11) كيفية إزالة البيانات المتكررة على مستوى البايت.



الشكل (11): كيفية إزالة البيانات المتكررة على مستوى البايت.

– كما عرف الباحثان عملية إلغاء البيانات المكررة بالاعتماد على الموقع، إذ أن إلغاء البيانات المكررة عند المصدر (المستخدم) له فعالية أقل من إلغاء التكرار عند الجهاز

الهدف (HDFS)، إذ قد يقوم المستخدم بمعالجة الملف لإزالة التكرار ثم تخزينه في نظام Hadoop، لكن قد يقوم الملف بتخزين الملف نفسه مرة أخرى أو بيانات منه في نظام Hadoop، وبالتالي سيتم تخزينه لأن عملية إلغاء البيانات المكررة تتم من جهة المستخدم، ولكن من إيجابيات هذه التقنية أنها تقلل من عرض النطاق الترددي لأنها تحذف البيانات قبل الإرسال إلى نظام تخزين Hadoop. أما إزالة البيانات المكررة بالاستناد إلى الجهاز الهدف لها فعالية أكبر في إلغاء التكرار على حساب الزيادة في البيانات المرسلة من المستخدم.

في نموذجنا المقترح، لم نقم بتخزين الملف في نظام Hadoop، ثم تطبيق تقنية إلغاء البيانات المكررة، ولا العكس من ذلك. إنما قمنا بتحويل الملف إلى ملف تسلسلي، والذي يشغل حجماً أقل من الملف ثم تخزين الملف التسلسلي في HDFS، وبالتالي حصلنا على بيانات أقل في الإرسال وفعالية عالية، لأن إلغاء التكرار يتم من قبل الجهاز الهدف (Hadoop).

– بالمقارنة مع النموذج المصمم من قبل الباحث Ruey-Kai Sheu وآخرين من حيث الخوارزمية المتبعة في توليد مفتاح التجزئة وكذلك إزالة البيانات المكررة بالاستناد على مستوى التقطيع، كما في الجدول (2).

النموذج المقترح	نموذج Ruey–Kai Sheu	
MD5	SHA2–512	خوارزمية توليد مفتاح التجزئة
على مستوى البيانات	على مستوى الملف	مستوى التقطيع

الجدول (2): مقارنة النتائج.

- كما ذكرنا، فإن التقطيع على مستوى الملف أقل فعالية في إلغاء التكرار من التقطيع على مستوى البايت (البيانات) لكن أسرع في التنفيذ.
- خوارزمية SHA2-512 تولد مفتاح تجزئة بحجم 512 بت، أما خوارزمية MD5 تولد مفتاح تجزئة بحجم 128 بت، وبالتالي فإن خوارزمية SHA2-512 ذات أمان أكبر، إذ يكون من الصعب جداً كسر مفتاح التشفير في حال إرسال البيانات، أما خوارزمية MD5 أقل أماناً لكنها أسرع في عملية توليد المفتاح المختزل.

– لا بد من الإشارة إلى أن تصميم نظام Hadoop والتحكم في برمجة نظامه يلعب دوراً كبيراً في التخزين ومعالجة البيانات. كما هو معروف، فإن تثبيت نظام Hadoop بشكل آلي يوفر تكرارية البيانات، وذلك من أجل الحفاظ على البيانات في حال ضياعها أو حذفها، إذ يقوم بإنشاء ثلاث نسخ من الملف الذي يتم تخزينه في HDFS، وفي حال تم فقدان إحدى النسخ يقوم بإعادة النسخ حتى يكتمل عدد النسخ إلى ثلاثة، ولكن تختلف متطلبات النظام من مبرمج لآخر أو من مستخدم لآخر. وفي ظل البحث في مجال إلغاء البيانات المكررة للحصول على أكبر فعالية ممكنة في إلغاء التكرار، يعتبر النسخ الاحتياطي للبيانات كإحدى السلبيات، وخاصة أنه يتم النسخ الثلاثي للبيانات، لأنه وفي هذه الحالة عند إلغاء التكرار قبل التخزين في نظام Hadoop ومن ثم إرسال البيانات إلى HDFS ستتكرر البيانات، ومن هذا المنحى اتبعنا في هذا البحث تهيئة نظام Hadoop ليقوم بالنسخ مرة واحدة فقط، أي سيكون هناك نسخة واحدة فقط من البيانات، وبالتالي تحقيق موازنة بين النسخ الاحتياطي لتأمين حماية البيانات وعملية إلغاء البيانات المكررة.

7- الخاتمة والآفاق المستقبلية:

إن عملية إلغاء البيانات المكررة تمر بمراحل أساسية عدة، إلا أن الاختلاف يكون في الطريقة المتبعة في كل مرحلة منها. في هذا البحث، تم تقديم تقنية MD5 القائمة على تجزئة البيانات وتطبيقها لإلغاء البيانات المكررة في نظام Hadoop لتوفير المساحة التخزينية. كما تم تطبيق عملية إزالة التكرار على مستوى البايت، والتي تحقق كفاءة عالية بالكشف عن البيانات المكررة وحذفها، للحصول على البيانات الفريدة. يمكن أن تتضمن الأبحاث المستقبلية بناء أنظمة لتحقيق موازنة أكبر بين عملية الكشف عن التكرار من جانب المصدر والهدف، وذلك بهدف الوصول إلى سرعة أكبر لإيجاد البيانات المكررة دون التأثير على سرعة عملية حذف التكرار، كما نقترح استخدام خوارزميات أخرى لإيجاد العناصر المتطابقة متضمنة أماناً كبيراً وسرعة عالية في توليد المفتاح المختزل.

8- المراجع:

1. K.A. Ramya, "A Survey on Lossless and Lossy Data Compression Methods", International Journal of Computer

Science and Engineering Communications, Vol.4, Issue.1, Page.1277-1280, (2016).

2. Kodituwakku, Amarasinghe, "Comparison of Lossless Data Compression Algorithms for Text Data", Vol 1 No 4 416-425.

3. Supriya Milind More, Kailas Devadkar, "A Comparative Survey on Big Data Deduplication Techniques for Efficient Storage System", IJIACS, ISSN 2347 – 8616, Volume 7, Issue 3, March 2018.

4. Venish and Siva Sankar, "Study of Chunking Algorithm in Data Deduplication", Proceedings of the international conference on soft computing systems ICSCS, Volume 2, 2016.

5. Ruey-KaiSheu and et al, "Design and Implementation of File Deduplication Framework on HDFS", International Journal of Distributed Sensor Networks, Volume 2014, Article ID 561340.

6. Shobha Rani, "MapReduce with Hadoop for Simplified Analysis of Big Data", International Journal of Advanced Research in Computer Science IJIACS, Volume 8, No. 5, May-June 2017.

7. Tom White," Hadoop: The Definitive Guide, Third Edition", ISBN: 978-1-449-31152-0 1327616795.

8. https://www.saylor.org/site/wp-content/uploads/2012/07/MD51.pdf accessed on 15/4/2018

9. https://comp.utm.my/marinamam/files/2016/11/Ch-12-Cyptographic-Hash-Function-student.pdf accessed on 24/6/2018

Develop a Parallel Treatment Method to Eliminate Duplicated Data in Distributed Systems

Dr. Eng. Ammar Ali Zakzouk

Assistant Professor in Automatic Control and Computers Engineering Department
Mechanical and Electrical Engineering Faculty
Albaath University

Abstract

Big data is a large collection of complex data collected from different sources. This complexity comes from the fact that the data comes from different sources and in large quantity. The larger the volume and complexity of the data, the less the efficiency of regular database management tools or traditional data processing applications in processing these data and less performance. The Hadoop system is an open-source system that not only supports parallel in data processing, but also has a distributed storage feature that enables us to store a large file on multiple computers and process file data distributed on the nodes as a single file. With the increasing amount of data day by day, it is better to keep little or even less data if it is useless to save storage space and increase data processing speed. In this research we present a new technique for removing redundant images from a wide range of images and preserving unique data in the storage system. MD5 based on data fragmentation and its application for deduplication in the Hadoop system was used to provide storage space. The byte-by-repeat removal process, which is highly efficient in detecting and deleting duplicate data, has been applied to obtain unique data.

Key-words: Big Data, Distributed Processing, Parallel Storage, MD5 Algorithm, Deduplication Data Cancelling, Hadoop.