KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

YÜKSEK LİSANS TEZİ

BÜYÜK DİKGEN GÖRÜNTÜLERİN ÖRÜLMESİ İŞLEMİNİN EŞLE/İNDİRGE TABANLI BÜYÜK VERİ MİMARİLERİNE UYARLANMASI

HAYRUNNİSA SARI

KOCAELİ 2017

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

BÜYÜK DİKGEN GÖRÜNTÜLERİN ÖRÜLMESİ İŞLEMİNİN EŞLE/İNDİRGE TABANLI BÜYÜK VERİ MİMARİLERİNE UYARLANMASI

HAYRUNNİSA SARI

Doç.Dr. Ahmet SAYAR
Danışman, Kocaeli Üniversitesi .....
Yrd.Doç.Dr. Orhan AKBULUT
Jüri Üyesi, Kocaeli Üniversitesi .....
Yrd.Doç.Dr. Seçkin ARI
Jüri Üyesi, Sakarya Üniversitesi .....

Tezin Savunulduğu Tarih: 19.12.2017

ÖNSÖZ VE TEŞEKKÜR

Bu tez çalışması, görüntü örme işlemi için Hadoop Eşle/İndirge mimarisine uygun olarak algoritma geliştirilmesine yönelik olarak gerçekleştirilmiştir.

Tez çalışmamda desteğini esirgemeyen, çalışmalarıma yön veren, rehberlik eden, bilgisini ve tecrübesini paylaşan, bana güvenen ve yüreklendiren danışmanım Doç. Dr. Ahmet SAYAR'a sonsuz teşekkürlerimi sunarım.

Eğitim hayatım boyunca manevi destekleriyle beni hiçbir zaman yalnız bırakmayan, her konuda destekleyen ve cesaretlendiren aileme teşekkür ederim.

Aralık – 2017 Hayrunnisa SARI

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar

HDFS : Hadoop Distributed File System (Hadoop Dağıtık Dosya Sistemi)
HIPI : Hadoop Image Processing Interface (Hadoop Görüntü İşleme Arayüzü)
LCS : Longest Common Subsequence (En Uzun Ortak Küme Algoritması)

BÜYÜK DİKGEN GÖRÜNTÜLERİN ÖRÜLMESİ İŞLEMİNİN EŞLE/İNDİRGE TABANLI BÜYÜK VERİ MİMARİLERİNE UYARLANMASI

ÖZET

Yapılan çalışmada, büyük görüntülerin işlenebilmesine yönelik olarak gerçekleştirilen çalışmalar derinlemesine incelenerek, büyük boyutlardaki dikgen iki resmin birleştirilmesi işleminin Hadoop Eşle/İndirge modeline uygun olarak uygulanması için geliştirilen algoritma ve sonuçlar sunulmuştur. Sunulan çalışmada, Hadoop Eşle/İndirge yaklaşımının metinsel veriler ile işlem gerçekleştirmesi nedeni ile resimler öncellikle bitmap formatına ardından "0" ve "255" değerlerinden oluşan string formatına dönüştürülerek HDFS üzerine metin olarak kaydedilmiştir. Resimlerin metinsel formatta HDFS üzerinden okunarak map fonksiyonuna girdi olarak verilmesi ile hesaplamalar gerçekleştirilmiştir. Görüntü birleştirme işlemi iki resmin birbiri üzerinde her üst üste çakışma durumunun göz önünde bulundurulması ile en büyük eşleşmeyi sağlayan indeksin referans alınması sonucu gerçekleştirilmiştir. Verilerin giderek büyümesi ile büyük boyutlu resimlerin işlenmesindeki performansın iyileştirilmesine yönelik duyulan ihtiyaç Hadoop Eşle/İndirge modelinin kullanılması ile giderilmeye çalışılmıştır. Yapılan çalışmada, büyük dikgen görüntülerin örülebilmesine yönelik mimari sunulmaktadır ve piksel bazlı resim görüntülerinin bitmap haritasının çıkarılıp kodlanmasıyla büyük parça görüntülerden nesne çıkarımı için altyapı gerçekleştirilmeye çalışılmıştır.

Anahtar kelimeler: Büyük Görüntü İşleme, Büyük Veri, Görüntü Örme, Hadoop Eşle/İndirge.

ADAPTATION OF ORTOGONAL IMAGE STITCHING TO BIG DATA FRAMEWORKS

ABSTRACT

In this study, the processing of large images has been analyzed in detail. Algorithm and results are presented in order to stitch two large orthogonal images in accordance with Hadoop MapReduce framework. In the presented work, the images are first converted to bitmaps and then to string format with values of "0" and "255" and recorded as text on HDFS because of the Hadoop MapReduce approach to processing textual data. Images are read in textual format via HDFS and given as input to the map function. The image stitching operation is performed by taking the reference of the index which provides the greatest match as a result of considering all possible overlapping situations on two images. It is aimed to improve the performance of processing large images with increasing data by using the Hadoop MapReduce framework. The infrastructure for image extraction of large size images has been realized by encoding as bitmaps of the pixel based image.

Key words: Big Image Processing, Big Data, Image Stitching, Hadoop MapReduce.

GİRİŞ

Görüntü örme işlemi, birbiri ile örtüşen bir dizi resimden tek bir görüntü oluşturma işlemidir. Verilerin giderek büyümesi geliştirilen algoritmaların performansının iyileştirilmesi gerekliliğini zorunlu kılmıştır.

Yapılan çalışmada büyük görüntülerin işlenmesine yönelik gerçekleştirilen performans odaklı yaklaşımlar derinlemesine incelenmiştir. Daha önce büyük görüntüler için yapılan performans odaklı çalışmalarda, büyük veri işleme kütüphanelerinden biri olan Hadoop ve Eşle/İndirge yaklaşımının olumlu performans etkisi göz önünde bulundurularak yapılan çalışmada büyük boyutlardaki dikgen resimlerin örülmesi işlemi için Hadoop kullanılmasıyla büyük boyuttaki görüntüler üzerinde işlem yapılabilirliğin sağlanması hedeflenmiştir. Bu çalışmada, iki resim üzerinde en büyük çakışma indeksinin olduğu noktanın referans alınması ile yeni tek bir resim oluşturulmasına yönelik geliştirilen algoritma ve sonuçlar sunulmuştur.

Bölüm 1'de, görüntü örme işlemi ve büyük veri işleme mimarileri üzerine temel konular üzerinde durulmuştur. Bölüm 2'de ise, Hadoop Eşle/İndirge yaklaşımı kullanılarak geliştirilen görüntü işleme çözüm yaklaşımları için daha önce yapılan 35 ayrı çalışma ele alınmıştır, Bölüm 3'te büyük görüntü örme üzerine yüksek başarım ve performans odaklı 10 çalışma incelenmiştir, Bölüm 4'te tez çalışması süresince geliştirilen algoritma açıklanarak probleme çözüm yaklaşımı ortaya konmuştur, Bölüm 5'te elde edilen sonuçlar yorumlanarak bir sonraki aşamada geliştirilebilecek nitelikler ortaya konmuştur.

1. BÜYÜK VERİ İŞLEME VE GÖRÜNTÜ ÖRME ÜZERİNE TEMEL KAVRAMLAR

Bu kısımda ilk olarak büyük veri kavramından bahsedilecek olup ardından büyük veri işleme kütüphanelerinden olan Hadoop üzerine değinilecektir. Büyük görüntüler ve büyük görüntülerin işlenebilmesine yönelik temel konulardan bahsedilip, büyük görüntüler üzerinde görüntü örme ve görüntü örme işlemi için Hadoop'un kullanılabilirliğine değinilecektir.

1.1. Büyük Veri

"Büyük Veri" terimi, geleneksel işlemler veya araçlar kullanılarak işlenemeyen veya analiz edilemeyen bilgiler için geçerlidir. Terabyte boyutunda petabyte boyutlarına kadar oluşan bir büyüklükte "hacme" sahip olan, çok farklı kaynaklardan ve formatlardan gelen verileri içeren bir "çeşitliliğe" sahip olan (örneğin web log dosyaları, sosyal medya etkileşimleri, e-ticaret ve çevrimiçi işlemler, finansal işlemler), çok büyük bir "hız" ile büyüyen veriler büyük veri olarak tanımlanabilir. Günümüzde giderek büyüyen veri ile ilgili zorluklarla karşı karşıya kalınmıştır [1]. Sosyal medya verileri, GPS verileri, sensör ağları, nesnelerin interneti gibi bir çok alandaki veriler ciddi boyutlarda artış göstermektedir. International Data Corporation(IDC), 2025'te küresel verinin 163 zettabayta (bir trilyon gigabayt) büyüyeceğini öngörmektedir. Bu 2016 yılında üretilen 16.1ZB'lık verinin on katına karşılık gelmektedir [2].Verilerin bu kadar büyük ölçekte artması, düşük maliyet ile veri işlenebilirliğine yönelik çalışmaların yapılmasını zorunlu kılmıştır.

1.2. Hadoop

Apache Hadoop büyük veri setlerinin bilgisayarlardan oluşan kümeler tarafından işlenebilmesini mümkün kılan java programlama dili ile geliştirilmiş yazılım kütüphanesidir. Hadoop çok büyük ölçekli veri işlemleri için özel olarak tasarlanmış dağıtılmış bir kümelenmiş dosya sisteminin üzerine kurulmuş veri işleme ortamı olarak düşünülebilir [1].

Hadoop, Google Dosya Sistemi (GFS) ve bir sunucu kümesinde depolanan verileri paralel olarak işlemek amacıyla Eşle/İndirge görevlerine ayrılan Eşle/İndirge programlama yaklaşımından ilham almıştır.

Bir sunucudan binlerce makineye, her bir makinenin kendi yerel hesaplamasını ve depolama alanını sunmasıyla ölçeklenebilir olarak tasarlanmıştır. Hadoop Eşle/İndirge büyük kümeler üzerinde paralel olarak çok büyük miktarda verileri işleyen uygulamaların kolaylıkla yazılabilmesi için geliştirilen yazılım modelidir [3].

Hadoop temel olarak HDFS dosya sistemi ve Hadoop Eşle/İndirge programlama yaklaşımından oluşmaktadır.

1.2.1. Hadoop dağıtık dosya sistemi (HDFS)

Hadoop Dağıtık Dosya Sistemi (HDFS), uygulama verisine yüksek verimli erişim sağlayan dağıtık dosya sistemidir. Büyük miktarlarda veri içeren uygulamalar için uygundur. Mevcut dağıtık dosya sistemlerinden farklı olarak yüksek hata toleransı ve düşük maliyetteki donanım üzerinde uygulamak için geliştirilmesi yönü ile ayrılır. Hataların tespit edilmesi ve hızlı, otomatik kurtarmanın sağlanması, HDFS'nin temel bir mimari hedefidir. Tipik bir HDFS dosyası gigabyte boyundan terabyte boyutuna ulaşan bir büyüklükte oluşabilir. Bu sebeple HDFS büyük dosyaları destekler nitelikte geliştirilmiştir. HDFS, dosyalar için bir kez yazma ve çoklu okuma modeli kullanır. Bu varsayım, veri tutarlılığı sorunlarını basitleştirir ve yüksek veri aktarımına olanak tanır. HDFS bir platformdan diğerine kolayca taşınabilir şekilde tasarlanmıştır. Bu, HDFS'nin büyük veri setleri ile geliştirilen uygulamalarda yaygın şekilde benimsenmesini kolaylaştırmaktadır. Bir uygulama tarafından talep edilen bir hesaplama, üzerinde çalıştığı verinin yanında yürütülürse daha verimlidir. Bu durum özellikle veri setinin çok büyük olması durumunda geçerlidir. Hadoop kullanmanın avantajlarından birisi dosyaların ilgili düğümün yerel diskinden okunmasıdır. Böylelikle ağ trafiğinin meşgul edilmesinin önüne geçilmiş olur. Hadoop ağ tıkanıklığını en aza indirir ve sistemin genel verimliliğini arttırır [3].

HDFS master/slave mimarisindedir. Bir HDFS kümesi dosya sistemi ad alanını yöneten ve istemcilere dosya erişimini düzenleyen bir tek ana sunucudan(namenode) ve birçok sayıda işçi sunucudan(datanode) oluşur. Namenode üzerinde verinin

kendisi değil, verinin adresi ile ilgili bilginin bulunduğu metadata tutulur. Namenode olmaksızın dosya sistemi kullanılamaz. Sistem çalışırken namenode kullanılamaz hale gelirse, dosya sistemindeki bütün dosyalar kaybolur ve datanode üzerindeki bloklardan dosyaların nasıl yeniden oluşturulacağını bilmenin hiçbir yolu kalmamış olur. Bu nedenle, namenodun hatalara karşı dayanıklı olması önemlidir ve Hadoop bunun için iki mekanizma sağlar. Birincisi dosya sistemi meta verisinin kalıcı durumunu oluşturan sistem dosyalarını yedeklemektir. Diğeri ise ikincil namenode çalıştırmaktır [4].

Veri, bloklara ayrılarak Hadoop işlem kümesi üzerinde dağıtılır. Böylece büyük veri map ve reduce fonksiyonlarında işletilebilir küçük parçalara bölünmüş olur ve bu işlem büyük veri işlemedeki ölçeklenebilirliği sağlamış olur. Verilerin bölündüğü blok boyutu başlangıç değeri olarak 64MB olarak belirlenmiştir.

1.2.2. Hadoop eşle/indirge programlama modeli

Verilerin çok büyük olması sonucu uygulamaların makul bir süre ile işlenebilmesi için hesaplamaların yüzlerce veya binlerce makineye dağıtılması ihtiyacını ortaya çıkarmıştır. Eşle/İndirge büyük veri setlerini, büyük bir küme üzerinde(cluster) işlemek amacıyla geliştirilen dağıtık programlama modelidir. Temel olarak Eşle/İndirge programlama modeli, map ve reduce aşamalarından oluşan iki kısımdan oluşmaktadır. Kullanıcılar, anahtar/değer(key/value) çiftlerinden oluşan bir map fonksiyonu ve aynı anahtar ile ilişkili tüm değerleri birleştiren bir reduce fonksiyonu tanımlar. Bu işlevsel stilde yazılan programlar otomatik olarak paralelleştirilir ve büyük bir küme üzerinde yürütülebilir [5].

Eşle/İndirge modeli, bir Hadoop kümesindeki yüzlerce veya binlerce sunucuda ölçeklenebilirlik sağlamaktadır.

Programlama modeli giriş değeri olarak almış olduğu anahtar/değer(key/value) çiftlerinden çıkış olarak anahtar/değer çiftlerinin üretilmesini sağlar. Bir Eşle/İndirge programında giriş ve çıkış değerleri incelenecek olursa ;

(giriş) <k1, v1> → map -> <k2, v2> → birleştir(combine) → <k2, v2> → reduce → <k3, v3> (çıkış)

Hadoop kümesinde, Eşle/İndirge programı bir işe (job) aittir. Bu iş, görev (task) adı verilen parçalara bölünür. JobTracker, iş için gereken tüm verilerin küme üzerinde nerede olduğunu bulmak için namenode ile iletişim kurar ve işi her düğüm için küme üzerinde çalışabilecek Eşle/İndirge görevlerine dönüştürür. Bu görevler, verilerin bulunduğu kümedeki düğümlerde programlanır. Bir görevin tamamlanmasının başarısız olması durumunda, başarısızlık durumu JobTracker'a bildirilir ve JobTracker başarısız görevi kümedeki başka bir düğümde yeniden planlar [1]. TaskTracker, küme üzerinde JobTracker'dan almış olduğu görevleri (map, reduce, shuffle işlemleri) kabul eden düğümdür. Her TaskTracker, kabul edebileceği görev sayısını belirten bir dizi yuva (slot) ile yapılandırılmıştır. JobTracker, Eşle/İndirge işlemlerinde bir görevi zamanlamak için bir yer bulmaya çalıştığında, önce aynı veriyi içeren Datanode'u barındıran sunucuda boş bir alan arar, bulamazsa aynı rafdaki bir makine üzerinde boş bir yuva arar. TaskTracker, üretilen süreçleri izleyerek çıkış ve çıkış kodlarını yakalar. İşlem başarıyla bitirildiğinde veya başarısız olduğunda, JobTracker'e bildirir [6].

1.3. Büyük Görüntü Verileri ve Büyük Görüntü İşleme

İnternet üzerine her gün artan miktarda fotoğraf verisi yüklenirken uygulamalar bu verileri işleyebilmekte yetersiz kalmaktadır [7]. Sosyal medya platformu olan Facebook istatistiklerine göre her gün ortalama 350 milyon, her saat 14,58 milyon, her dakika 243,000, her saniye 4000 fotoğraf yüklenmektedir. Aynı zamanda günümüzdeki hastanelerin büyük bölümü, veri merkezlerinde patolojik görüntüler, BT taramaları ve X-ışınları gibi medikal görüntüleme biçiminde petabaytlık hasta kayıtları üretip depoladığı için büyük veri sorunu ile karşılaşmaktadır [8]. Petabyte boyutlarına kadar ulaşan bu kadar büyük görüntüler ile kabul edilebilir bir işleme zamanı problemi ile karşılaşılmaktadır. İşleme zamanı zorluğu işlemlerin dağıtık ve ölçeklenebilir bir şekilde gerçekleştirilmesi ihtiyacını ortaya çıkarmıştır. Hadoop genellikle metinsel verilerin işlenmesinde kullanılmış olsa da görüntü verileri üzerinde de kullanılmaktadır [9]. Bu kısımda Hadoop'un büyük görüntü işleme ara yüzü olan HIPI'den bahsedilecektir.

1.3.1. Hipi - Hadoop görüntü işleme arayüzü (Hadoop image processing interface

HIPI, Apache Hadoop Eşle/İndirge paralel programlama yapısı kullanımına uygun olarak geliştirilen yazılım kütüphanesidir. HIPI, Eşle/İndirge programlama yaklaşımı ile paralel olarak geliştirilen (özellikle bir küme üzerinde çalışan) uygulamalarda yüksek verimli görüntü işleme işlemlerinin uygulanabilmesinde kolaylık sağlar. Büyük bir görüntü kümesinin HDFS üzerinde depolanması ve dağıtık programlanabilmesi için bir çözüm sunar. HIPI aynı zamanda birçok görüntü işleme algoritmasını içeren popüler görüntü işleme kütüphanesi olan OpenCv ile birlikte kullanım da sağlamaktadır [10].

HIPI'nin amacı, araştırmacıları ve öğrencileri kolaylıkla uygulamalar oluşturmalarını sağlayacak olan büyük ölçekli görüntü işleme ve görme projelerinin geliştirilmesini son derece erişilebilir hale getirecek bir araç oluşturmaktır [7].

1.4. Görüntü Örme Üzerine Temel Kavramlar

Görüntü örme, birbiri ile ortak alanlara sahip görüntülerden tek bir görüntü elde etme amacıyla görüntülerin birleştirilmesi işlemini kapsamaktadır.

Örülen görüntüler, resimlerin panoramik görüntülenmesinde, uydu görüntülerinde ve dijital haritalarda mozaik görüntülerin yüksek çözünürlüklü görüntülenmesinde, tıbbi görüntülemede, bir mekânın üç boyutlu sanal turunun oluşturulmasında, gerçek dünyadan edinilen görüntüleri kullanarak 3D ortamı modelleme ile ilişkili diğer uygulamalarda kullanılabilir [11-12].

Görüntü örme işleminde temel olarak kullanılan terimler incelenecek olursa [13] ; özellik, görüntü üzerinde algılanması kolay olan, eşsiz, belirgin ve istenen kısımdır, görüntü kaydı (image registration), farklı veri kümelerinin tek bir koordinat sistemine dönüştürüldüğü bir işlemdir. Kayıt, farklı boyutlardan elde edilen verileri ilişkilendirmek veya dahil etmek için gereklidir. Bir görüntünün referans alınmasıyla, bütün noktaların bire bir aynı hizada çakıştırılmasını temel alan bir işlemdir. Özellik eşleme (feature matching) iki özellik kümesi arasında benzerlik oluşturmaktır.

Görüntü örme yaklaşımları ağırlıklı olarak doğrudan ve özellik tabanlı olmak üzere iki grupta sınıflandırılır.

1.4.1. Doğrudan teknikler

Girdi görüntülerinin piksel yoğunluklarını değerlendirir. Bir görüntünün, her bir piksel yoğunluğunu diğerinin piksel yoğunluğuyla karşılaştırma yolunu izler. Bu yaklaşımda her bir pikselin karşılaştırılması karmaşık bir işlemdir. Bu yaklaşımda, resimler pikseller arasındaki benzerlik derecesini bulmak için birbirlerine göre kaydırılır. Piksel-piksel eşleştirme kullanan bu metodolojiler yaygın olarak doğrudan yöntemler olarak bilinir [13].

Tez kapsamında geliştirilen algoritmada doğrudan teknik kullanılmıştır.

1.4.2. Özellik tabanlı teknikler

Girilen görüntülerin çıkarılan özelliklerine dayalı olarak görüntüler arasındaki bir ilişkiyi çözer [13 - 14]. Bu yaklaşım, görüntü rotasyonuna karşı daha dayanıklı, hızlı ve düzensiz görüntü dizileri arasındaki örtüşen ilişkileri otomatik olarak belirleme yeterliliğine sahip olması açısından doğrudan tekniklere göre daha avantajlıdır [13].

Bu yaklaşım için; özellik çıkarımı, görüntü kaydı ve görüntü harmanlama takip eden çeşitli aşamalardır. Özellik tabanlı yöntemler, noktalar, çizgiler, kenarlar, köşeler veya diğer şekiller arasında denklikler kurarak kullanılır.

Özellik tabanlı birleştirme işleminin gerçekleştirilebilmesi için çok sayıda özellik algılama yöntemi bulunmaktadır. Bu yöntemlere SURF, SIFT, MSER [15], Harris, FAST [13] örnek olarak verilebilir.

2. HADOOP EŞLE/İNDIRGE KULLANILARAK GELİŞTİRİLEN GÖRÜNTÜ İŞLEME ÇÖZÜM YAKLAŞIMLARI

2.1. Temel Görüntü İşleme Algoritmalarının Hadoop Eşle/İndirge Kullanılarak Geliştirilmesi Üzerine Yaklaşımlar

Epanchintsev T. ve diğerleri [16] yapmış oldukları çalışmada, görüntü işleme için Hadoop Eşle/İndirge tabanlı dağıtık bir altyapı sunmuşlardır. Java görüntü işleme kütüphanesi olan OpenIMAJ ve Java2D kullanarak SIFT, kenar tespiti gibi temel görüntü işleme işlemlerini dağıtık olarak gerçekleştirmişlerdir. Yaptıkları çalışmada algoritmaları tekrar yazmak yerine var olan kütüphaneleri dağıtık olarak kullanabilir duruma getirmişlerdir.

Vemula S. ve Crick C. [17] yapmış oldukları çalışmada, büyük ölçekte görüntülerin işlenebilmesini mümkün kılan Hadoop Eşle/İndirge temelli kütüphane geliştirmişlerdir. Yapılan çalışmayı, Hadoop'un güçlü Eşle/İndirge sisteminin teknik detaylarını soyutlamak ve kullanıcıların büyük resim veri setlerini işlemek için kolay bir mekanizma sağlamak üzere tasarlamışlardır. Laplacian filtreleme, Canny kenar tespiti ve k-means görüntü segmentasyonu algoritmaları için dağıtık bir sistem geliştirmişlerdir. Yaptıkları çalışma, Hadoop'un Eşle/İndirge çerçevesinin üstünde büyük ölçekli görüntü işleme uygulamaları oluşturmak için yeni bir şeffaflık ve basitlik düzeyi sağlamıştır.

Dong ve diğerleri [18] bulut resim işleme(ICP) üzerine büyük resim verileri ile başa çıkabilecek nitelikte çalışma gerçekleştirmiştir. Önerdikleri ICP sistemi için statik ICP (SICP) ve dinamik ICP(DICP) olmak üzere 2 farklı yaklaşım ile sunmuşlardır. SICP yaklaşımını daha önceden dağıtık dosya sistemi üzerinde saklanan büyük resim verilerini işlemek için geliştirirken, DICP yaklaşımını dinamik giriş verileri için geliştirmişlerdir. Elde ettikleri deneysel sonuçlarda, Eşle/İndirge ile birlikte geliştirdikleri sistemin zamansal olarak etkin bir yaklaşım olduğuna ve kaliteli sonuçlar ürettiğine değinmişlerdir. Geliştirmiş oldukları ICP yaklaşımının Hadoop kümesi üzerinde çalıştırılması yönü ile güçlü olduğuna değinmişlerdir.

Bednarz ve diğerleri [19] yapmış oldukları çalışmada, Hadoop dosya sistemi ve Eşle/İndirge programlama yaklaşımı kullanan JAI kütüphanesinden yararlanan (HIPI görüntü işleme için Java ile geliştirilen açık kaynak yazılım kütüphanesi) bir sistem geliştirmişlerdir.

Wu ve diğerleri [20] yüksek karmaşıklık ve hesaplama yükü olan, çok kullanıcılı sistem üzerinde, görüntü işleme işlemlerini map fonksiyonu ile map fonksiyonu ile işlenen ara veriyi birleştirerek son çıktıyı üreten reduce fonksiyonu tanımlamışlardır. Çalışmalarında, DSRF(Dinamik Anahtar Dönüştürme İşlevi) algoritması ile başarıyla tamamlanan görevlerin yüzdesine göre dinamik olarak görevler arasında geçiş yapan bir sistem önermişlerdir. Bekleme süresi ve hesaplama zamanının azaltılması ile daha yüksek performanslı bir Hadoop Eşle/İndirge sistemi elde etmeyi hedeflemişlerdir. Elde ettikleri deneysel sonuçlarda, Eşle/İndirge işlemini verimli bir şekilde hızlandırmakta ve geleneksel Hadoop uygulamasından daha iyi bir performans elde ettiklerine değinmişlerdir.

Almeer [21] Sobel filtreleme, resmi yeniden boyutlandırma, resim format dönüşümü, otomatik zıtlık, resim keskinleştirme, resim gömme, metin gömme, görüntü kalitesi denetimi gibi temel görüntü işleme algoritmalarını Hadoop Eşle/İndirge kullanarak 112 çekirdekten oluşan bir bulut hesaplama sistemi üzerinde gerçekleştirmiştir. Çalışmasında Katar Yarımadası'nın uydulardan alınan büyük ölçekli(166.698.335 piksel çözünürlüğünde) TIFF görüntüleri kullanılmıştır. Gerçekleştirmiş olduğu deneysel sonuçlar, temel görüntü işleme algoritmalarının uzaktan algılama görüntülerine uygulandığında kabul edilebilir çalışma süreleri ile etkili bir şekilde paralel olabileceğini göstermiştir.

Barapatre H. ve diğerleri [22] yapmış oldukları çalışmada, fotografik mozaik oluşturma konusunda büyük bir veri görüntü işleme probleminin çözümü için Hadoop temelli uygulama sunmuştur. Geliştirmiş oldukları Eşle/İndirge temelli yaklaşımda, ilk olarak map aşamasında girdi imgenin tüm grid hücreleriyle mesafesini (veya aynılığı) hesaplamışlardır. Reduce aşamasında, her grid hücresi için en uygun görüntüyü bulmuşlardır.

Kaneko K. ve Yamamoto M. [23] yapmış oldukları çalışmada, video kamera ile çekilmiş video görüntüler üzerinde çalışmışlardır. Video çerçevelerinden gri

görüntüler oluşturmak ve video görüntüler üzerinden özellik çıkarımı için Eşle/İndirge programlama yaklaşımının kullanılması ile görüntü işleme gerçekleştirmişlerdir.

Bajcsy P. ve diğerleri [24] yapmış oldukları çalışmada, her biri yaklaşık yarım Terabyte boyutundan büyük üç mikroskobik veri seti üzerinde Hadoop Eşle/İndirge temelli yaklaşım ile çalışarak bir Hadoop kümesi üzerinde resim verisi ile çalışmanın faydalarını paylaşmak istemiştir. Çalışmalarında Hadoop ile kullanılan mevcut kütüphanelerin(NIH ImageJ/Fiji, ImageJDev, BioImageXD, ImageJ) yetersiz kaldıklarına değinerek kendi görüntü işleme kütüphanelerini geliştirdiklerini belirtmişlerdir.

White B. ve diğerleri [25] yapmış oldukları çalışmada, Eşle/İndirge yaklaşımının çeşitli pratik bilgisayar görme algoritmalarına (sınıflandırıcı eğitimi, kayan pencereler, kümeleme, bag-of-features, arka plan çıkarma ve image registration) nasıl uygulanacağını göstermişlerdir.

Sayar A. ve diğerleri [26] yapmış oldukları çalışmada, HDFS'in özellikle büyük boyutlu dosyalar için tasarlandığına değinerek çalışmalarında küçük olan resim dosyalarının birleştirilmesiyle performansın iyileştirmesine yönelik çalışma sunmuştur. Resimlerin sayısının azalmasına bağlı olarak JobTracker ve TaskTracker üzerindeki yükün de azalması ve alınan girdi verisinin boyutunun artmasından dolayı task başına düşen tek seferde görüntü işleme adımı sayısı artışını performans üzerinde olumlu etki olarak değerlendirmişlerdir.

Ramos Pollan ve diğerleri [27] heterojen hesaplama kaynakları üzerinde büyük ölçekli görüntü işleme ve analizi için bir yazılım çerçevesi olan BIGS'i önermiştir. HBase ve Amazon'un DinamoDB hizmeti için BIGS desteği de dahil olmak üzere, tüm işçiler tarafından gerekli paylaşılan bilgi kaynaklarının ölçeklenebilirliğini sağlamak için bir NoSQL depolama modeli kullanmayı tercih etmişlerdir.

Malakar R. ve Vydyanathan N. [28] yapmış oldukları çalışmada, yüksek performanslı heterojen bir görüntü işleme sistemi oluşturmak için CUDA(genel amaçlı hesaplamalar için GPU'nun paralel hesaplama yeteneklerini kullanmak için NVIDIA tarafından önerilen bir C tabanlı programlama modelini Hadoop dağıtık

işleme çerçevesine entegre etmeye çalışmıştır. Adaboost tabanlı bir yüz algılama algoritması kullanan deneysel değerlendirmelerine göre, bir Hadoop kümesini etkinleştiren CUDA'nın, düşük uçlu(low-end) ekran kartlarını kullanırken bile %25'lik bir iyileşme sağlayabileceğini gözlemlemişlerdir.

Moise D. ve Shestakov D. [29] yapmış oldukları çalışmada, Eşle/İndirge paradigmasının açık kaynaklı uygulaması olan Hadoop üzerinde terabayt boyutunda görüntü verisi işleme ve ilgili sonuçları sunmuştur. Hadoop'un performans avantajlarına değinmenin yanında büyük veri işlemede yaygın olarak bulunan kaynak heterojenliği ve veri yönetimi sorunlarını ele almışlardır. Çalışmalarında yetersiz disk alanı, iletişim zaman aşımı vb. sebeplerle görevlerin başarısız olabildiğini gözlemlediklerini belirmişlerdir.

Ali M. ve Kumar J. [30] gri tonlamalı, sobel kenar algılama, Gauss bulanıklığı ve hızlı corner_9 algılama gibi görüntü işleme teknikleri için Hadoop ve bulut hesaplama yöntemlerinin kullanılması ile etkin zaman performansı elde etmeyi hedeflemiştir. Mevcut görüntü işleme algoritmaları için optimum süre performansı elde edilmesi hedeflenmiştir.

Wang W. ve diğerleri [31] Hadoop Eşle/İndirge tabanlı görüntü estetik analiz yaklaşımı yöntemini önermiştir. C++ ile kodlanan estetik analiz algoritmalarını JNI kullanarak Java platformu üzerinde Eşle/İndirge yaklaşımına uyarlamaya çalışmışlardır. Küçük boyutlarda olan çok fazla resmin Hadoop Eşle/İndirge yaklaşımı ile işlenme verimliliğini arttırmayı hedeflemişlerdir. Hadoop Eşle/İndirge programlama yaklaşımın çok büyük dosyalar için verimli olduğuna değinip kendi sistemlerindeki küçük boyutlu fakat çok fazla miktarda olan resim dosyaları için bellek tasarrufu ve JVM yeniden kullanma stratejisini önermişlerdir. Birden çok kullanıcının eşzamanlı istekleri için, varsayılan FIFO (İlk Giren İlk Çıkanlar) zamanlayıcının talepleri karşılayamadığına değinerek Eşle/İndirge tasarımını optimize ettiklerine değinmişlerdir. Büyük miktarda küçük dosyaların içeriğini birkaç büyük dosyaya birleştirmek için kullanmış oldukları Hadoop Arşivlerinin (HAR) orijinal HDFS'den küçük dosyaların depolanmasında çok daha iyi verimliliğe sahip olduğuna değinmişlerdir. Geliştirmiş oldukları paralel hesaplama ve optimizasyon tekniklerinin performans artışını önemli ölçüde etkin hale getirdiğine

değinmişlerdir. Hadoop kümesinde işçi düğüm sayısındaki artışın geliştirdikleri sistem performansını olumlu etkilediğine değinmişlerdir.

Wu ve diğerleri [32] çalışmalarında büyük resim verilerin etkin bir şekilde işlenebilmesi için Eşle/İndirge tabanlı çözüm üretmiştir. Çalışmalarını MATLAB'ın dağıtık hesaplama altyapısını kullanarak gerçekleştirmişlerdir. Elde ettikleri sonuçlarda Eşle/İndirge yaklaşımının performans olarak oldukça etkin olduğuna değinmişlerdir.

Zhang ve diğerleri [33] Hadoop Streaming teknolojisini kullanarak Hadoop Eşle/ İndirge temelli büyük ölçekte görüntü işleme yaklaşımı önermiştir. Çalışmalarının ikili görüntü işleme için geliştirildiğine değinmişlerdir. Elde ettikleri deneysel sonuçlarda, tek düğümlü makine ile karşılaştırdıklarında büyük görüntü işleme için önemli performans kazanımları elde ettiklerine değinmişlerdir. Ayrıca artan görüntü verisi ile gerçekleştirdikleri deneylerde, paralel işlemenin avantajlarının belirgin bir şekilde ortaya konduğuna değinilmiştir. Deneylerini. bmp formatında, 256x256'lık ikili resimler ile gerçekleştirmişlerdir.

2.2. HIPI ile Görüntü İşleme

Chris Sweeney ve diğerleri [34] Eşle/İndirge yaklaşımı için açık kaynak kodlu görüntü işleme ara yüzü sunmuştur. Geliştirmiş oldukları sistemin amacı büyük ölçekteki görüntü işleme projelerinin geliştirilmesinde kolaylık sağlayacak bir araç oluşturmaktır. Sunmuş oldukları ara yüz ile kullanıcıyı Eşle/İndirge yaklaşımının karmaşık detaylarından gizlemektedir. HIPI kullanıcıları için yüksek derecede paralelleştirilmiş ve dengeli uzaktan algılamada için bir format ve bu dosyaları oluşturmak için basit yöntemler sunmuşlardır.

Sunny B. ve diğerleri [35] yapmış oldukları çalışmada, ameliyat sonrası incelemeleri mümkün kılan cerrahi işlem sırasında kaydedilen boyutu GB seviyesinde olan yüksek tanımlı (HD) videolar üzerinde analiz gerçekleştirebilmek için Hadoop'un görüntü işleme ara yüzü olan HIPI kullanarak Hadoop temelli yaklaşım önermiştir. Sunmuş oldukları çalışmada cerrahi videolardan aletleri tespit etmek için SIFT, SVM ile SURF ve SVM ile Haralick Özellikleri algılama yöntemlerini dağıtık olarak tercih etmişlerdir.

2.3. Uzaktan Algılama ve Uydu Görüntülerinin İşlenmesi

Pan ve Zang [36] uzaktan algılama görüntülerinin işlenmesi için Hadoop temelli yaklaşım önermiştir. Elde ettikleri sonuçlarda bulut ortamında HDFS kullanılarak test etmişler ve geliştirdikleri yaklaşımın hızlı olduğu sonucuna varmışlardır.

Golpayegani ve Halem [37] Lv ve arkadaşları [38] Hadoop ile görüntüleri ham olarak kullanmadan önce metinsel formata ardından ikili formata dönüştürerek bazı uydu görüntü işleme algoritmalarını Hadoop'un eşle/indirge modelini kullanarak gerçekleştirdiler. Görüntülerin ham olarak kullanılmaması nedeniyle ön işleme uzun süren bir hesaplama zamanı almıştır. Ermias da [39] yaptığı çalışmada eşle/indirge temelli büyük boyutlu uydu görüntülerinin işlenmesini sunmuş; Sobel, Laplacian ve Canny gibi kenar bulma algoritmaları üzerinde çalışmıştır.

Cavallaro vd. ve diğerleri [40] uzaktan algılamada kanıtlanmış istatistiksel veri madenciliği yöntemlerinin, veri analiz süreçlerine ölçeklenebilir ve paralel işleme teknikleri geliştirebilmek üzerine çalışmıştır. Çalışmalarını, büyük uzaktan algılama veri kümelerini ölçekleyebilen ve özellik çıkarımı yöntemleriyle yüksek doğruluk sağlayan sınıflandırma için hızlı bir SVM yaklaşımı geliştirilmesi üzerine yürütmüşlerdir. SVM'in büyük veriler üzerinde işlenebilmesi üzerine yapmış oldukları araştırmalarda, Java platformunda Hadoop temelli Apache Mahout (0,9 sürümü) kullanımın SVM için paralelleştirme teknolojisine sahip olmadığını bu yüzden bilimsel çalışma için tercih edilen bir araç olmadığını, Java platformunda Spark temelli ApacheSpark/MLib (1.1 sürümü) yalnızca doğrusal SVM'leri desteklediğini belirli problem alanıyla birlikte tercih edilen bir teknoloji olmadığını, C ve MPI tabanlı pSVM, MPI'ın eski ve beta sürümünün kullanılmasından kaynaklı olarak bilimsel çalışma bağlamında çalışabilecekleri bir araç olmadığını belirtmişlerdir. Çalışmalarına PiSVM (1.2 sürümü ve 1.3 sürümü) libSVM'e dayandığı, güncel MPI standartlarına dayandığı, açık kaynak kodlu olduğu için tercih ettiklerini belirtmişlerdir. PiSVM uygulamasını kullanarak, paralelleştirme tekniklerini uygulamak, çapraz doğrulama ve her bir eğitim ve test işlemi için önemli hızlanma elde edildiği sonucuna vardıklarından bahsetmişlerdir.

Chebbi I. ve diğerleri [41] yapmış oldukları çalışmada, Eşle/İndirge programlama modelini uygulayan, büyük ölçekli uzaktan algılama görüntülerinin sınıflandırmasını

geliştirebilen ve mekânsal büyük veriler için Hadoop konseptinin gücünden faydalanan bir sistem önermiştir. Çalışmalarında Hadoop Eşle/İndirge ve HDFS kullanarak büyük uydu görüntüleri depolamak ve işlemek için bir yaklaşım sunulmuştur. Yapmış oldukları deney sonucunda, önerilen sistemin büyük veri hacmi ile bile iyi sonuç alabildiğini gözlemlemişlerdir.

Sarade D. ve diğerleri [42] çalışmalarında, TIF formatında uzaktan algılama görüntülerinin işlenebilmesi için Hadoop Eşle/İndirge temelli yaklaşım önermiştir. Temel görüntü işleme algoritmalarının uzaktan algılama görüntüleri için Hadoop yaklaşımı ile makul işlem sürelerinde hesaplanabileceğine değinmişlerdir. Metinsel verinin işlenmesi için tasarlanan Hadoop yaklaşımın TIF resimler ile de başarılı olarak gerçeklenebildiğine değinmişlerdir.

Eldawy [43] ve diğerleri, NASA arşivinde bulunan uydu veri setlerini indekslemek, sorgulamak ve görselleştirmek için etkili bir yol sağlayan interaktif bir sistem olan Shahed'i sunmaktadır. Kullanıcıların haritayı kullanarak mevcut veri kümelerine göz atabilmelerini, zamansal olmayan sorguları yayınlamalarını ve sonuçları görüntü veya video olarak görselleştirebilmelerini sağlayan, harita tabanlı bir web arabirimi aracılığıyla Shahed'le gerçek bir deneyim sunmaktadır. Kullanıcıların uydu verileri verimli bir şekilde kullanıcı dostu bir ara yüzle sorgulamasına ve görüntülemesine olanak tanıyan bir sistem geliştirmişlerdir. Geliştirdikleri sistemde, büyük miktarda veriyi işlemek için mekânsal veriler için bir Eşle/İndirge çerçevesi olan SpatialHadoop kullanılmıştır. SpatialHadoop, Eşle/İndirge'i mekânsal veri işleme için ana programlama paradigması olarak kullanmaktadır. Geliştirdikleri sistemde, resimler ve videolar için sorgu sonuçlarını Eşle/İndirge kullanarak ısı haritaları olarak görselleştirmek için seçenekler sunmuşlardır.

2.4. CBS Verilerinin İşlenmesi

Park J. ve diğerleri [44] , büyük boyutlara ulaşan 3D Coğrafi Bilgi Sistemi versisini paralel işlemeyebilmek adına Hadoop Eşle/İndirge tabanlı ve MPI tabanlı iki farklı yaklaşım önermiştir. MPI ile üretmiş oldukları çözüm yaklaşımının Hadoop yaklaşımına göre daha hızlı olduğu sonucuna varmışlardır. Eşle/İndirge yaklaşımın ise MPI'a göre hata toleransı yüksek ve daha kararlı olması yönü ile önde olduğuna değinmişlerdir.

Zhao L. ve diğerleri [45] çalışmalarında Yüksek Performans Hesaplama(HPC) kümesi ve Hadoop kümesi üzerinde paralel GIS mimarisi yaklaşımını incelemişlerdir. İncelemiş oldukları GridGIS mimarisinin heterojen sistemler arasındaki birlikte çalışabilirliği çözmek ve mekânsal veri CBS servislerini paylaşmak için internetten dağıtılan kaynak ve verilerin avantajlarından yararlanırken geliştirici için karmaşık olduğuna ve ağ kaynaklarının tam olarak kullanılamadığına; Cloud GIS mimarisinin kullanıcının iş talebi ve geliştiricinin yükünü azaltması, yüksek kaynak ve düşük ağ kullanımı avantajlarına karşılık güvenlik ve gizlilik eksikliğinin olduğuna, çevrimiçi işlemler için daha yüksek ağ gereksinimlerinin olduğuna; HPC temelli Paralel GIS mimarisinin güçlü hesaplama gücü, karmaşık işlemler için uygun olması, GIS hesaplama görevlerinin çeşitli özelliklerine erişilebilmesi yönü ile avantajlıyken geliştiriciler için yüksek gereksinimlerin olduğuna, yüksek maliyet ve düşük ölçeklenebilirlik olduğuna; Hadoop temelli Paralel GIS mimarisi için düşük maliyet ile yüksek performans elde edilebilmesi, yüksek ölçeklenebilirlik ve güvenilirliğe sahip olması, geliştiriciler için gereksinimlerin düşük olması yönü ile avantajlıyken yalnızca büyük veriler için uygun ve gerçek zamanlı uygulamalar için uygun olmadığına değinmişlerdir.

2.5. Mekânsal Verinin İşlenmesi

Eldawy ve diğerleri [46] mekânsal veriler için geliştirdikleri açık kaynak kodlu ve dağıtık Eşle/İndirge temelli SpatialHadoop çerçevesini sunmuştur. SpatialHadoop'un mevcut Hadoop programlarını olduğu gibi çalıştırırken mekânsal veriyle uğraşırken Hadoop'tan daha iyi bir performans gösterdiğine değinmişlerdir. 70M uzamsal nesneler için sorgu, 20 düğümlü bir Hadoop kümesinde 200 saniye çalışırken, aynı özelliklerdeki kümede SpatialHadoop 2 saniyede gerçekleştirmektedir.

Aji ve diğerleri [47] çalışmalarında yüksek performanslı mekânsal sorgular üretmek için mekânsal veri depolama sistemi olan Hadoop GIS i sunmuştur. Geliştirdikleri Hadoop GIS bildirilen mekânsal sorguları desteklemek için Hive'a entegre edilmiştir. Patoloji görüntüleme ve Açık Sokak Haritası veri setlerinde yapmış olduklarını deneysel sonuçlarda Hadoop-GIS'in verimli olduğu sonucunu belirtmişlerdir.

2.6. Gökbilimi ile İlgili Resimlerin İşlenmesi

Wiley ve diğerleri [48] multiterabyte boyutunda SDSS görüntü veri tabanını kullanarak Hadoop Eşle/İndirge tabanlı ölçeklenebilir bir görüntü işleme çalışması yürütmüştür.400 düğümden oluşan bir Hadoop kümesi üzerinde yapmış oldukları testlerde 100.000 dosyayı (300 milyon piksel) 3 dakikada işlediklerini belirtmişlerdir. Hadoop kullanımın çok büyük veri setlerini işlemek için uygun olduğuna değinmişlerdir.

Chang G. ve diğerleri [49] yüzlerce gigabayt boyutuna ulaşabilen görüntü dosyalarına(The Lunar Mapping and Modeling Project (LMMP)) Eşle/İndirge tabanlı yaklaşımı önermiştir. Bulut bilgi işlem araçlarını kullanarak, donanım ve bilgisayar altyapı maliyetlerini azaltmayı amaçlamışlardır. Çalışmaları için Elastic Compute Cloud (EC2) ve Amazon'un Basit Depolama Hizmetini (S3) kullanmayı tercih etmişlerdir.

Banda J. ve diğerleri [50] güneş görüntü verilerinin işlenmesi için büyük veri yaklaşımını inceleyen bir çalışma sunmuştur. Her gün terabyte boyutundan daha büyük bir ölçekte artan güneş verisi üretildiğine değinerek odaklandıkları Hadoop yaklaşımın güneş fiziği için gelecek olduğuna değinmişlerdir.

3. BÜYÜK GÖRÜNTÜ ÖRME ÜZERİNE YÜKSEK BAŞARIM VE PERFORMANS ODAKLI ÇALIŞMALARI

3.1. Hadoop Eşle/İndirge Temelli Yaklaşımlar

Sayar A. ve diğerleri [51] mozaik uydu görüntülerinin örülmesi ve nesne çıkarımı için Eşle/İndirge tabanlı dağıtık ve ölçeklenebilir bir mimari oluşturmayı hedeflemiştir.

Sayar A. ve diğerleri [52] uydu görüntülerinin birleştirilmesinde yüksek performansın Hadoop Eşle/İndirge mimarisine uygun olarak gerçekleştirilebilmesine yönelik vektör tabanlı mozaik örme durum çalışması gerçekleştirmiştir.

Rajak R. ve diğerleri [53] yapmış oldukları çalışmada, Hadoop kullanarak paralel olarak uzaktan algılama uydu veri işleme gerçekleştirmek amacıyla ve program çıktısını HBase'de saklamak üzere Eşle/İndirge tabanlı mimari sunmuştur. Image registration, Water shed görüntü segmentasyonu, Image Mosaicing ve Gauss Filtresi sunmuş oldukları çalışmada Eşle/İndirge çözümü önerdikleri algoritmalardır. Yapmış oldukları deneyleri, Landsat görüntülerinden uydu verileri üzerinde gerçekleştirmişlerdir. Bu çalışmada, optimize HDFS depolama alanına sahip yüksek çözünürlüklü uydu görüntü verilerini işleme için Hadoop küme çerçevesini kullanmanın verimliliği getirdiği sonucuna varılmıştır. Dört düğümlü bir kümenin kullanılmasıyla karmaşık görüntü işleme algoritmaları için bile en az 7X hız elde edilebileceği sonucuna varmışlardır.

3.2. Diğer Performans Yaklaşımları

Kubiasa A. [54] ve diğerleri, GPU üzerinde etkili bir 2B / 3B görüntü kaydı gerçekleştiren bir yöntem önermiştir. Görsel olarak x-ray resimlerine benzer gerçekçi DRR (digitally reconstructed radiography)(dijital olarak yeniden yapılandırılmış radyografi) üretmek için etkin bir metot önerdiklerine değinilmiştir. Çalışmalarında GPU, paralel olarak DRR ve röntgen görüntüsü arasında yoğunluk temelli benzerlik ölçütü(intensity-based similarity measures) hesaplamak için kullanılmıştır. GPU

programlama için Open GL ve Cg kullanımını tercih etmişlerdir. Benzerlik ölçütünü elde edebilmek için gerçekleştirdikleri deneysel sonuçlarda GPU algoritmalarının CPU algoritmalarına göre üç ile altı kat daha hızlı olduğu gözlenmiştir.

Shamonin D. [55] ve diğerleri medikal görüntüler üzerinde tıbbi görüntü analizinde önemli, fakat zaman alıcı bir görev olan görüntü kaydı için paralelleştirme yöntemleri önermiştir. Seçmiş oldukları Gauss piramitleri yönteminin görüntü üzerinde bağımsız satır-satır işlenebileceğine bu nedenle de her satır veya sütunu farklı bir iş parçacığına atayarak GPU için uygun bir yöntem geliştirebildiklerine değinmişlerdir. Paralellik için, kullanılan thread sayısı CPU çekirdeği sayısına ulaşana kadar performansın arttığını gözlemlenmiştir. Uygulamış oldukları paralelleştirme ve optimizasyonun, 8 çekirdekli bir makinede 8 iş parçacığı kullanarak 4-5x'lik bir hızlanma ile sonuçlandığına değinilmiştir. Çalışmalarında GPU, algoritmanın bir parçası olarak hesaplama açısından maliyeti yüksek bileşenleri hızlandırmak için kullanılmıştır.

Kim H. ve Parashar M. [56] çalışmalarında, tıbbi görüntü kayıt uygulaması için yerel hesaplama ortamlarını (veri merkezleri) ve genel bulut hizmetlerini bütünleştiren CometCloud çerçevesini kullanan bulut sistemi araştırmıştır. Görüntü kaydında her işçinin bütün bir görüntüyü işlediğine bu nedenle işlenecek görüntü sayısı kadar görev tanımlandığına değinmişlerdir. Çalışmaları için Rutgers Üniversitesi New Jersey Kanser Enstitüsü'nde özel bulut ve Amazon EC2 üzerinde genele açık bulut ortamı kullanılmıştır.

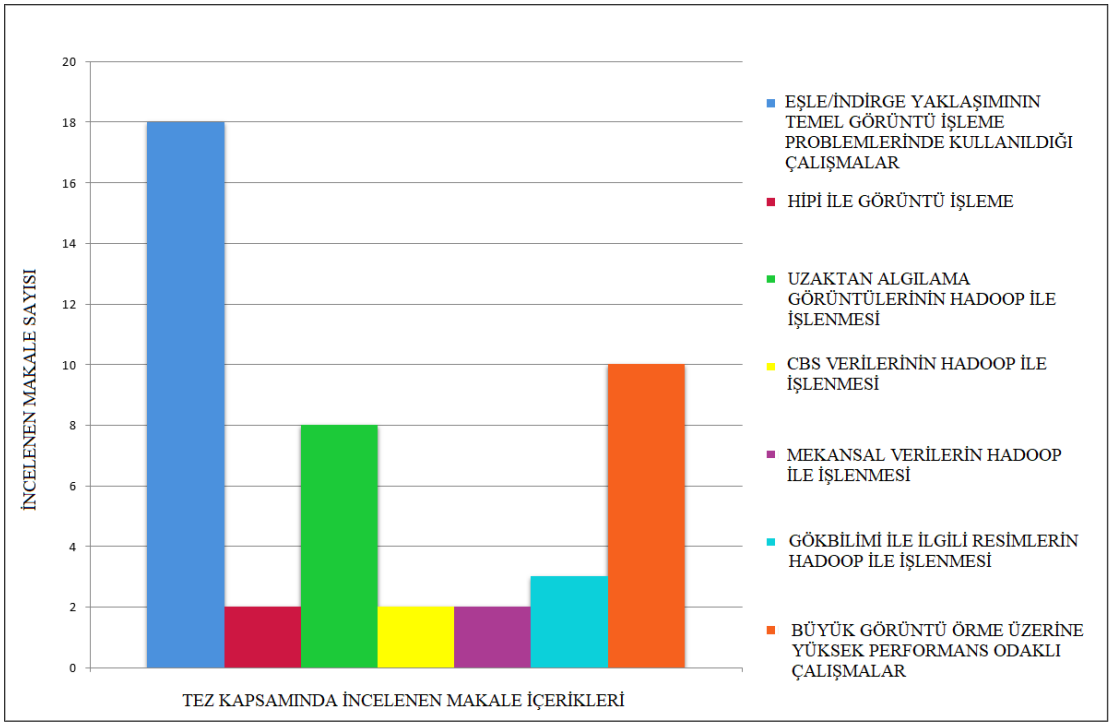
Diaz J. ve diğerleri [57] kullanıcılara grid, bulut ve yüksek performans hesaplama altyapısı sunan FutureGrid (FG)'i destekleyen resim oluşturma ve resim kaydetme işlemlerini içeren kullanıcıdan soyut resim yönetim platformu geliştirmeyi hedeflemiştir. Geliştirdikleri görüntü kaydı altyapısı için Nimbus, Eucalyptus, OpenStack, Moab/xCAT kullanıldığına değinilmiştir. Resim kaydını desteklemek için görüntüyü bulut ortamına kaydetmek ve HPC altyapılarına kaydetmek üzere iki ayrı hizmet sunulmuştur.

Warfield S. ve diğerleri [58] tıbbi görüntülerin kaydı için paralel algoritma geliştirerek kayıt algoritmalarının performans özelliklerini araştırmıştır. Çalışmalarında her düğüm, POSIX iş parçacıklarıyla gerçekleştirilen

paralelleştirilmiş bir yeniden örnekleme ve karşılaştırma işlemi gerçekleştirmektedir ve MPI ile gerçekleştirilen iletişimle kümede dinamik olarak yük dengelemesi sağlanmıştır.

Adam M. [59] ve diğerleri çalışmalarında, yüksek çözünürlüklü görüntüleri örmeye ve büyük HD panoramaları oluşturmaya olanak tanıyan gerçek zamanlı GPU tabanlı uygulama önermiştir. Önermiş oldukları yaklaşımın görüntülerde karşılık gelen noktalar arasındaki farklılıkların hesaplanmasına dayandığını belirtilmiştir. Sunmuş oldukları görüntü örme çalışmasının, MRF(Markov random fields) yaklaşımına dayandığını ve yaklaşık çıkarım için yaygın olarak kullanılan BP(max-product belief propagation) algoritmasının kullanıldığına değinilmiştir. NVIDIA CUDA çerçevesini kullanan GPU donanımı üzerinde gerçeklendiğini belirtilmiştir.

Ino F. [60] ve diğerleri, çok işlemcili sistemler sade görüntü kaydı için paralel algoritma önermiştir. Ölçeklenebilir performans ile büyük ölçekli kayıt gerçekleştirmek için, geliştirdikleri algoritmanın veri dağıtım, veri-paralel işleme ve yük dengelemesi olmak üzere üç ayrı adım izlemişlerdir. Elde ettikleri deneysel sonuçlarda, yük dengeleme tekniği ile işlem süresinin yarıya indiğine değinilmiştir.



Şekil 3.1. Büyük Görüntülerin Performans Odaklı İşlendiği İncelenen Çalışma Sayıları

Bu kısımda tez konusu ile ilgili literatür araştırması kapsamında Hadoop ile Büyük Görüntü İşleme başlığı altında 45 ayrı çalışma incelenmiştir. Yapılan çalışmaların kategorik olarak sınıflandırma grafiği Şekil 3.1 ile gösterilmektedir. İncelen çalışmalarda, Hadoop Eşle/İndirge programlama yaklaşımının büyük görüntülerin işlenmesi konusunda olumlu etkileri üzerinde durulması sebebiyle bu tez çalışmasında dikgen görüntülerin örülmesi işlemi için Hadoop Eşle/İndirge temelli bir yaklaşım önerilmiştir.

4. BÜYÜK GÖRÜNTÜLER ÜZERİNDE GÖRÜNTÜ ÖRME İŞLEMİ İÇİN GELİŞTİRİLEN EŞLE-İNDİRGE TABANLI MİMARİ

Tez süresince büyük dikgen görüntülerin birleştirilmesine yönelik çalışma gerçekleştirilmiştir. Öncelikle Hadoop üzerine uyarlanabilir basit tek makine çözümlerinin üretilmesine yönelik çalışmalar yapılmıştır. Ardından Hadoop kullanılarak görüntü birleştirme için çözüm üretilmiştir.

Bu kısımda öncelikle tez süresince üretilen tek makine çözümüne değinilecek olup ardından geliştirilen Hadoop Eşle/İndirge çözümünden bahsedilecektir.

4.1. Tek Makinede Üretilen Çözümler

Geliştirilen algoritmada her bir resim pikseli için string kodlama işlemi gerçekleştirilerek pikseller arasındaki eşleşme durumlarının pikseller için üretilen kod değerleri arasında LCS algoritmasının uygulanması ile bulunması hedeflenmiştir.

Bu kısımda ilk bölümde LCS algoritmasından bahsedilecek olup, ikinci bölümde LCS algoritmasının görüntüler arasındaki kesişimleri bulma işleminde kullanılmasına değinilecektir.

4.1.1. En uzun ortak küme (LCS) algoritması

Verilen iki dizi arasında, her iki dizide de ortak bulunan alt diziyi bulan algoritmadır. İki dizi arasında bulunan alt dizi, dizilerin elemanlarının birbiri ardında olma zorunluluğu bulunmayan ve sıralı olan ortak elemanlarından oluşmaktadır.

abcdef” ve “acdefghjkl” dizileri arasındaki en uzun ortak küme 5 elemanlı “acdef” alt kümesi olarak bulunur.

LCS yaklaşımı için dinamik programlama ile geliştirilen algoritma çözümüne ait sözde kod Şekil 4.1 ile gösterilmiştir.

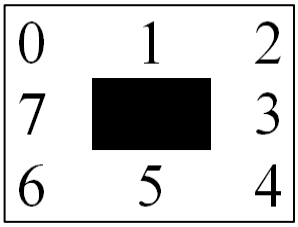
Algoritma 1 LCS Algoritması - Dinamik Programlama Çözümü

```
1:  function LCSALG(input1,input2)          ▷ input1 - dizi, input2 - dizi
2:      LCS ← dizi(0..m,0..n)
3:      for i = 0 to m do
4:          LCS[i,0] ← 0
5:      end for
6:      for j = 0 to n do
7:          LCS[0,j] ← 0
8:      end for
9:      for i = 1 to m do
10:         for j = 1 to n do
11:             if input1[i] ==input2[i] then
12:                 LCS[i,j] ⇐ 1+LCS[i,j]
13:             else
14:                 LCS[i,j] ← max(LCS[i−1,j],LCS[i,j−1])
15:             end if
16:         end for
17:     end for
18: end function
```

Şekil 4.1. LCS Algoritması - Dinamik Programlama Çözümü

4.1.2. İki görüntünün birleştirilmesi işleminin LCS algoritmasının kullanılması ile çözülmesi

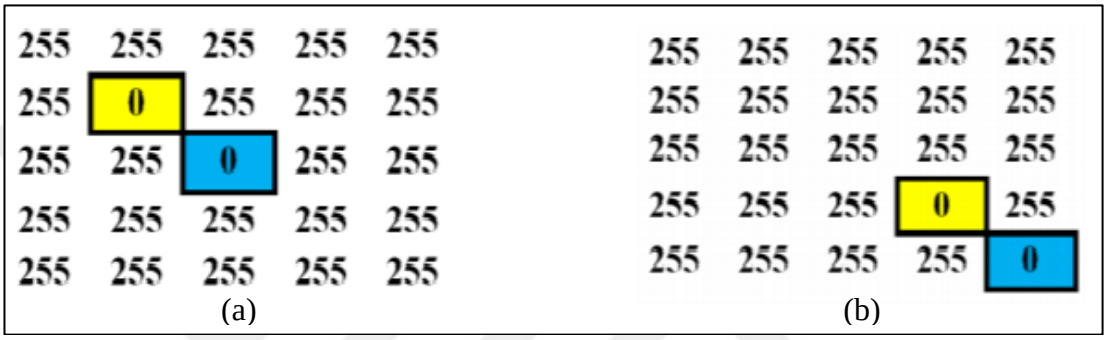
Algoritma siyah-beyaz resimler üzerinde işlem yapılabilir olarak tasarlanmıştır. Gri resimler ‘128’ eşik değerinin referans alınması ile siyah-beyaz resme dönüştürülmüştür. Siyah piksel değeri ‘0’ , beyaz piksel değeri ‘255’ olarak kabul edilmiştir. Geliştirilen algoritmada her bir siyah piksel için kod üretilerek siyah pikseller arasında üretilen kodlar arasında LCS algoritmasının uygulanması ile resimlerin birbiri üzerinde en çok ortak eleman içerdiği noktanın hesaplanması gerçekleştirilmiştir. Hesaplanan bu noktanın referans olarak alınması ile görüntülerin birleştirilmesi işlemi gerçekleştirilmiştir.



Şekil 4.2. Bir Pikselin Komşuluk Değerleri

Bir siyah piksele ait kod değeri, pikselin komşuluğunda bulunan siyah piksellerin konumuna göre üretilmiştir. Şekil 4.2 ile bir piksele ait komşuluk değerleri gösterilmektedir. Bir pikselin komşuları [0,7] aralığındaki değerlerden birini almaktadır. Üretilen piksel koduna, komşuluğundaki piksellerde siyah piksel olması durumunda komşuluk değeri eklenmektedir.

Geliştirilen algoritmanın siyah pikseller için kod üretme işlemleri adım adım incelenecek olursa;



Şekil 4.3. Resimler İçin Tanımlı Matris Elemanları a) Resim1 b) Resim2

Şekil 4.3.a ile gösterilmekte olan Resim1 için matris elemanları üzerinde sarı ile işaretli [1,1] indeksindeki “0” elemanının (siyah piksel) “4” komşuluğunda şekil üzerinde mavi ile işaretlenmiş [2,2] indeksinde “0” değeri bulunmaktadır. [1,1] koordinatındaki “0” elemanı için “4” komşuluk değeri üretilen string kod değerine eklenmiştir.(Komşuluk değerleri Şekil 4.2 ile gösterilmiştir.)

Üretilen Kod = [1,1] : 4

[1,1] indeksindeki “0” elemanının “0” değerinde olan mavi ile işaretlenmiş [2,2] indeksindeki elemanın “0” komşuluk değerinde”1” elemanı bulunmaktadır. String kod değerine 0 değeri Üretilen Kod = [1,1] : 4 0 eklenmektedir.

Üretilen Kod = [1,1] : 4 0

Şekil 4.3.b ile gösterilmekte olan Resim2 için matris elemanları üzerinde, [3,3] indeksinde sarı ile işaretli “0” elemanının “4” komşuluğunda şekil üzerinde [4,4] indeksinde mavi ile işaretlenmiş “0” değeri bulunmaktadır.[3,3] koordinatındaki “0” elemanı için “4” komşuluk değeri üretilen string kod değerine eklenmektedir.

Üretilen Kod = [3,3] : 4

[3,3] indeksindeki “0” elemanının “0” değerinde olan mavi ile işaretlenmiş [4,4] indeksindeki elemanın “0” komşuluk değerinde”0” elemanı bulunmaktadır. String kod değerine 0 değeri eklenmektedir.

Üretilen Kod = [3,3] : 4 0

Uygulamada üretilen piksel kod değerleri arasında LCS algoritmasının uygulanması ile ortak “0” değerlerinin yakalanması beklenmektedir.

Algoritma 2 Rekürsif Piksel Kodlama Algoritması

```
1:  function PIKSELKODLA(resim,koordX,koordY,kod,siyahPikselListesi)
2:      pikselRengi ← resim[koodrX,koordY]
3:      h ← resim.yukseklikDegeri
4:      w ← resim.genislikDegeri
5:      if pikselRengi < 128 then
6:          kod ← kod + "[" + koordX + "," + "koordY" + "]"
7:          siyahPikselListesi.ekle(kod)
8:          if ((koordY + 1) < h) and ((koordX + 1) < w) then
9:              PikselKodla(resim,koordX + 1,koordY + 1,kod,SiyahPikselListesi
10:         end if
11:         if ((koordX + 1) < w) then
12:             PikselKodla(resim,koordX + 1,koordY,kod,SiyahPikselListesi
13:         end if
14:         if ((koordY + 1) < h) then
15:             PikselKodla(resim,koordX,koordY + 1,kod,SiyahPikselListesi
16:         end if
17:     else
18:         if ((koordX + 1) < w) then
19:             PikselKodla(resim,koordX + 1,koordY,"",SiyahPikselListesi
20:         else if ((koordX + 1) == w) and ((koordY + 1) < h) then
21:             PikselKodla(resim,0,koordY + 1,"",SiyahPikselListesi
22:         else if ((koordX + 1) == w) and ((koordY + 1) == h) then
23:             return;
24:         end if
25:     end if
26: end function
```

Şekil 4.4. Rekürsif Piksel Kodlama Algoritması

[1,1] ve [3,3] elemanlarının kod değerleri arasında LCS değeri uygulandığında “4 0” değeri elde edilmektedir. Bunun anlamı ikisi arasındaki ortak 1 lerin komşuluklarının

“4 0” kod değerinde olmasıdır.

Resimlerin birleştirilme işlemi Resim1’in [1,1] noktasının Resim2 üzerinde [3,3] noktası ile çakıştırılması ile gerçekleştirilmektedir.

Geliştirilen rekürsif piksel kodlama algoritmasının sözde kodu Şekil 4.4 ile gösterilmektedir.

4.1.3. LCS algoritması kullanılarak geliştirilen algoritma ile ilgili elde edilen sonuçlar

Tek makinede çalışmak üzere resimler arasındaki eşleşmeleri bulabilmek adına algoritma sunulmuştur. Algoritma bütün üst üste eşleşme durumlarını bulabilir nitelikte geliştirilmiş olsa da tanımlanan kesişim bulma fonksiyonun rekürsif olması performansı olumsuz yönde etkilemiştir.

Tek makinede çalışmak üzere geliştirilen algoritmanın bir sonraki adımda Hadoop Eşle/İndirge mimarisine uyarlanması hedeflenmektedir.

4.2. Hadoop Eşle/İndirge Mimarisine Uygun Olarak Geliştirilen Çözümler

Yapılan çalışmada iki resmin birbiri ile en çok ortak kesiştiği koordinatın referans alınması ile dikgen resimlerin birleştirilmesi gerçekleştirilmiştir. Geliştirilen algoritma tek bir makine üzerinde çalışmaktadır.

Algoritma Eşle/İndirge mimarisine uygun olarak java programlama dili geliştirilmiştir. Algoritma geliştirilirken açık kaynak kodlu javacv kütüphanesinden yararlanılmıştır.

Yapılan çalışmada resimler Hadoop dosya sistemi üzerinden string formatında okunmuştur. Her bir resim string formatında ifade edilmiştir. Oluşturulan Map fonksiyonlarına resimler string formatında girdi olarak verilmiştir.

Uygulama mimarisi art arda çalışan iki mapper fonksiyonunun tanımlanması ile oluşmaktadır. 1.Mapper fonksiyonunun çıktısı, 2. Mapper fonksiyonunda girdi olarak kullanılmıştır. 1.Mapper fonksiyonu resimlerin birebir eşleştirilebileceği yeni bir uzaya geçirilmesi işlemini kapsarken, 2.Mapper fonksiyonu resimler arasındaki ortak

kesişimlerin hesaplanması ve hesaplanan en çok ortak kesişim koordinatı referans alınarak iki görüntünün birleştirilmesi işlemini kapsamaktadır.

4.2.1. Resimlerin string formatında ifade edilmesi

Yapılan çalışmada oluşturulan map fonksiyonlarında metinsel olarak girdi uygulanabilmesi amacı ile resimlerin string formatında ifade edilmesi işlemi gerçekleştirilmiştir.

255	255	255	255	255		255	255	255	255	255
255	0	255	255	255		255	255	255	255	255
255	255	0	0	0		255	255	255	255	255
255	255	255	255	255		255	0	0	0	255
255	255	255	255	255		255	255	255	255	0
(a)						(b)				

Şekil 4.5. Resimler İçin Tanımlı Matris Elemanları a) Resim1 b) Resim2

Uygulamada kullanılan resimlerin pikselleri siyah – beyaz olmak üzere iki değer ile tanımlanmıştır. Siyah pikseller “0” değeri ile, beyaz pikseller “255” değeri ile ifade edilmiştir. Şekil 4.5.a ve b ile iki farklı 5x5 görüntü için piksel değerleri matris formatında gösterilmiştir.

Resimlerin string formatına dönüştürülmesi, resim matrislerinin sütunlarının “,” karakteri ile resim matrislerinin sütunlarının satırlarının “;” karakteri ile ayrılacak şekilde sıralanması ile gerçekleştirilmiştir.

Şekil 4.5.a ile gösterilmekte olan Resim1 string formatında ifade edilecek olursa elde edilen değer, “ 255 , 255 , 255 , 255 , 255 ; 255 , 0 , 255 , 255 , 255 ; 255 , 255 , 0 , 0 , 0 ; 255 , 255 , 255 , 255 , 255 ; 255 , 255 , 255 ,255 , 255 ; ” olur.

Şekil 4.5.b ile gösterilmekte olan Resim2 string formatında ifade edilecek olursa elde edilen değer, “ 255 , 255 , 255 , 255 , 255 ; 255 , 255 , 255 , 255 , 255 ; 255 , 255 , 255 , 255 , 255 ; 255 , 0 , 0 , 0 , 255 ; 255 , 255 , 255 , 255 , 0 ; ” olur.

4.2.2. Resimlerin yeni uzayda tanımlanması

Resimlerin yeni bir uzaya geçilerek tanımlanması iki resim arasındaki olası tüm çakışma durumlarının hesaplanabilmesi için uygulanmıştır. Resimlerin yeni bir uzaya dönüştürülmesi işlemi uygulamada tanımlanan ilk map fonksiyonudur.

Giriş : String formatında Resim1 <tab> String formatında Resim2
Çıkış : String formatında Resim1 <tab> String formatında yeni uzayda tanımlı Resim2

Dosya üzerinden okunan her bir resim ikilisi için Resim1'in genişlik ve yükseklik değeri kadar beyaz pikselin Resim2'nin sol ve üst kısmına eklenmesi gerçekleştirilmiştir.

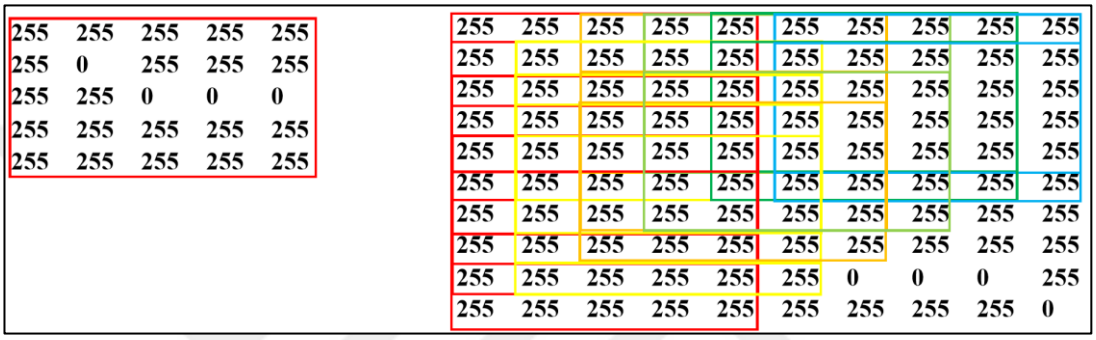
Şekil 4.5.a ve b ile gösterilmekte olan 5x5 boyutundaki resimler yeni uzaya geçildiğinde, Resim1'in genişliği olan 5 değeri kadar beyaz piksel Resim2'nin sol kısmına eklenmiştir, Resim1'in yükseklik değeri olan 5 değeri kadar beyaz piksel Resim2'nin üst kısmına eklenmiştir. Böylece 10x10'luk yeni bir resim elde edilmiştir. Şekil 4.5.b ile gösterilmekte olan resim için elde edilen yeni resim Şekil 4.6 ile gösterilmiştir.

255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	0	0	0	255
255	255	255	255	255	255	255	255	255	0

Şekil 4.6. Resimlerin Yeni Uzayda Tanımlanması

4.2.3. Resimler arasındaki en büyük kesişme sayısının hesaplanması

İki resim arasında birinci matrisin boyutu referans alınarak oluşturulan blok boyutu ile resimlerin birbiri üzerinde gezdirilmesi ve çakışan siyah piksellerin sayılması işlemi gerçekleştirilmiştir. Resim1'in elemanları Resim2 üzerinde kayan bir çerçeve gibi düşünülerek Resim2'nin tüm elemanları ile birebir eşleşme sağlanacak şekilde ortak indisteki piksellerin siyah olması durumu sayılmıştır. İfade edilen kayan pencere durumu Şekil.4.7 ile gösterilmiştir.



Şekil 4.7. Resimlerin Birbiri Üzerinde Gezdirilmesi

Şekil.4.7 ile gösterilmekte olan ilk resim matrisinin elemanları ikinci resim matrisinde gösterilmekte olan her biri farklı renk ile işaretlenen 5x5'lik matrisler ile eşleştirilmiştir. Bu işlem Resim2 üzerinde tanımlı tüm 5x5'lik matrisler için gerçekleştirilmiştir ve her eşleşme durumu için kesişen siyah pikseller sayılmıştır. Elde edilen kesişme sayılarının en yüksek olduğu kesişim koordinatı tutularak bir sonraki adım olan resimlerin birleştirilmesi işleminde kullanılmıştır. Resimler arasındaki en büyük eşleşme sayısının hesaplanması işleminin matematiksel olarak ifadesi Denklem (4.1) ile gösterilmiştir.

$$\text{En büyük kesişim sayısı} = \sum_{row=0}^{n+a-1} \sum_{col=0}^{m+b-1} \text{en büyük seç} \left(\sum_{i=0}^{n} \sum_{i=0}^{m} 1\,[Resim1_{ij} = YeniResim2_{i+row,j+col} = 0] \right) \qquad (4.1)$$

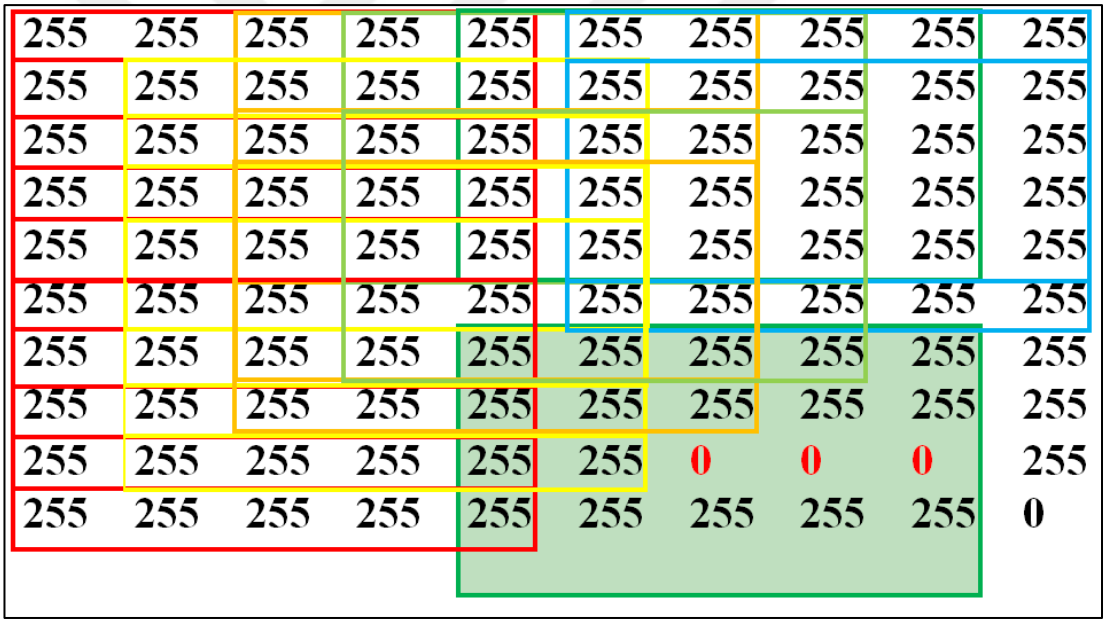
Burada, n Resim1'in satır sayısı, n + a Resim2'nin yeni uzayda tanımlı satır sayısı, m Resim1'in sütun sayısı, m + b Resim2'nin yeni uzayda tanımlı sütun sayısı, a Resim2'nin satır sayısı, b Resim2'nin sütun sayısı olarak ifade edilmiştir.

İşlemler algoritmada tanımlanan 2.map fonksiyonu üzerinde gerçekleştirilmiştir. Map fonksiyonun girdisi 1.map fonksiyonun çıktısı olarak tanımlanmıştır.

Giriş: String formatında resim1 <tab> String formatında yeni uzayda tanımlı resim2
Çıkış: Resim1 ve Resim2'nin birleşmesi ile string formatında tanımlanan yeni resim

4.2.4. Resimlerin birleştirilmesi

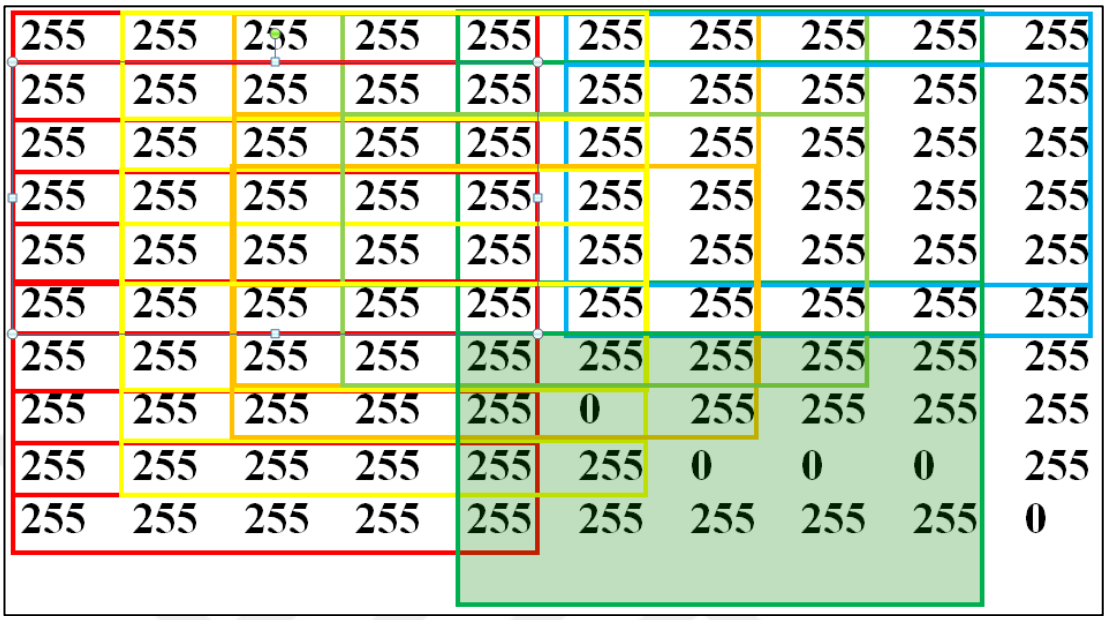
Resimler, bir önceki adımda hesaplanan iki resim arasındaki en büyük kesişme koordinatının referans alınmasıyla birleştirilmiştir. Şekil 4.5.a ile gösterilen resim ile Şekil 4.6 ile gösterilen yeni uzayda ifade edilen ikinci resmin birleştirilme işlemi için elde edilen en büyük kesişme koordinatı ikinci resim üzerinde (4,6) noktası olarak hesaplanır.



Şekil 4.8. En Büyük Kesişme İndeksinin Gösterilmesi

(4,6) noktası referans alınarak Şekil 4.5.a ile gösterilen matris boyutu olan 5x5 birimlik matris Şekil 4.8 ile gösterilen yeşil taralı bölge ile ifade edilen matristir. Resim1'in matris elemanları ile Şekil 4.8 ile gösterilmekte olan yeşil bölge ile taranmış matris elemanları üst üste çakıştırıldığında 3 siyah piksel üst üste denk gelmekte (kesişen siyah piksel değerleri şekil üzerinde kırmızı ile gösterilmiştir) ve resimlerin birleştirilmesi sırasında iki resim arasındaki en büyük kesişme miktarını göstermektedir. Resim 1'in matris elemanları Resim 2 üzerine (4,6) noktası referans alınarak yerleştirilmiştir. Elde edilen birleştirilmiş resim Şekil 4.9 ile

gösterilmiştir. Geliştirilen birleştirme algoritmasının sözde kodu Şekil 4.10 ile gösterilmiştir.



Şekil 4.9. İki Resmin Birleştirilmesi İşleminin Şekil Olarak İfade Edilmesi

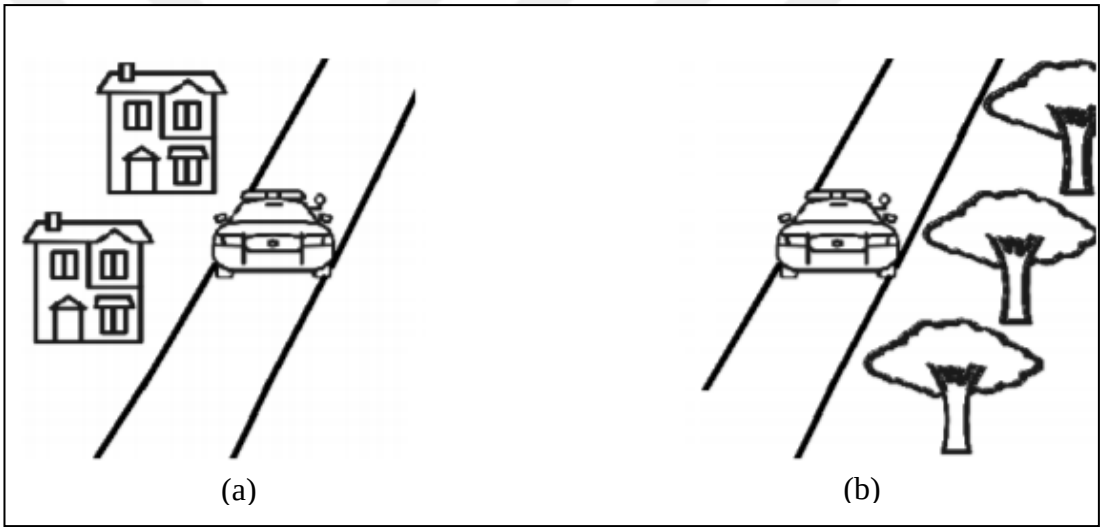
Algoritma 3 Resimleri Birleştirme Algoritması

```
1:  function MAP(resim1,resim2)          ▷ resim1(nxm boyutunda)2 - resim2(axb
    boyutunda) - String formatında tanımlı resimler
2:      for row = 0 to n + a − 1 do
3:          for col = 1 to m + b − 1 do
4:              EslesmeSayisi ← 0
5:              for i = 0 to m do
6:                  for i = 0 to n do
7:                      if resim1[i, j] ==resim2[i,j] then
8:                          EslesmeSayisi ←EslesmeSayisi + 1
9:                      end if
10:                 end for
11:             end for
12:         end for
13:     end for
14: en büyük eslesme sayısını ve eslesme koordinatını sec (x,y)
15:     for i = 0 to m do
16:         for i = 0 to n do
17:             resim2[i + x] ←resim1[i+j]
18:         end for
19:     end for
20: end function
```

Şekil 4.10. Resimleri Birleştirme Algoritması

4.2.5. Geliştirilen uygulama

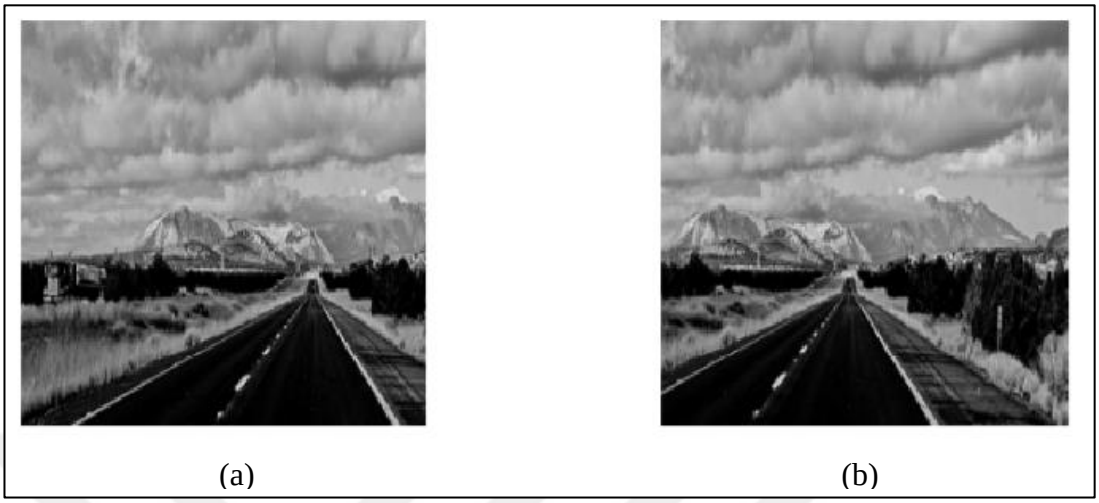
Geliştirilen uygulama kullanıcıdan giriş değerleri olarak iki ayrı resmin dosya yolunu alır. Resimleri dosya üzerinde okuma işlemini gerçekleştirmenin ardından resimleri string formatına dönüştürerek dosyaya kaydeder. Yazılan map fonksiyonunun girdisi resimlerin string formatında kaydedilmiş olduğu bu metin dosyasıdır. Resimler üzerinde yeni uzaya geçme map fonksiyonu ve maksimum eşleme/birleştirme işlemini gerçekleştiren map fonksiyonları birbiri ardına çalıştırılır. Yeni resim dosya sistemi üzerinde yazılır. String formatındaki resmi ".png" formatına dönüştürecek olan program fonksiyonu çalıştırılır ve resimlerin birleşmiş hali uygulama çıktısı olarak alınmış olur.



Şekil 4.11. Uygulamada Giriş Olarak Uygulanan Basit Resimler a)Resim1 b) Resim2

Şekil 4.12. Elde Edilen Yeni Resim

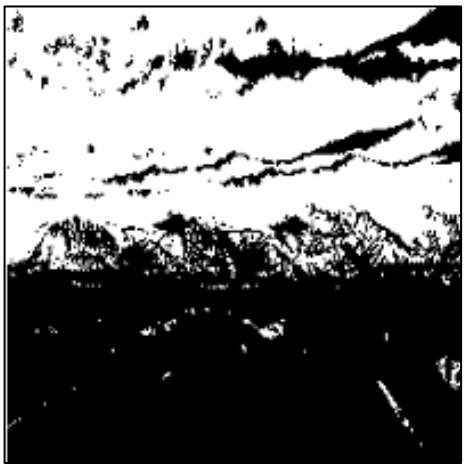
Şekil 4.11 ile gösterilen basit iki resmin alınması ile birleştirilme işlemi başarılı olarak gerçekleştirilmiştir ve Şekil 4.12 ile gösterilmiştir.



(a) (b)

Şekil 4.13. Uygulamada Giriş Olarak Uygulanan Gri Resimler a) Resim1 b) Resim2

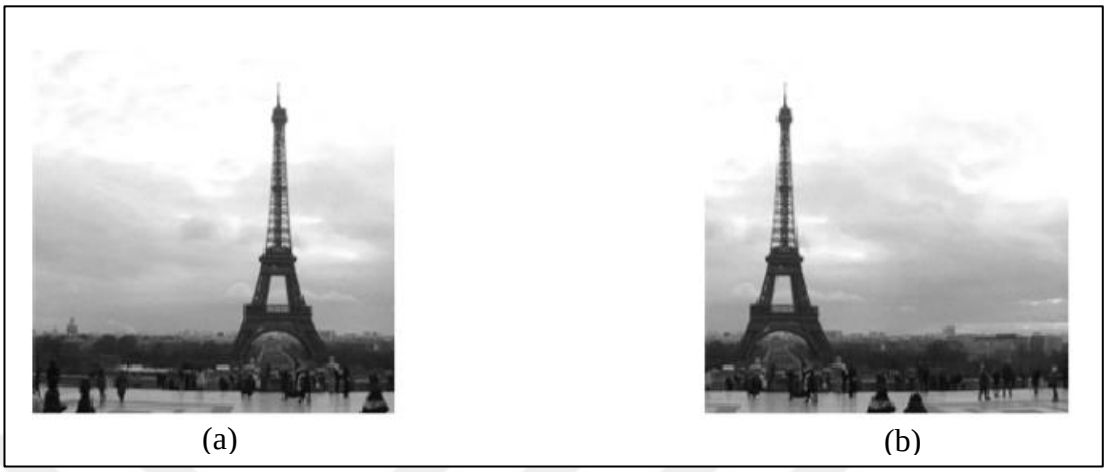
Şekil 4.13.a ve b ile gösterilmekte olan gri resimler üzerinde birleştirme algoritması uygulandığında resimleri birleştirme işlemi gerçekleştirilmiştir. Okunan resimlerin siyah beyaza dönüştürülürken eşik değerinin tüm resimler için 128 seçilmesi nedeniyle Şekil 4.14 ile gösterilmekte olan elde edilen yeni resim gözlemlenebilir bir nitelikte olmamıştır.



Şekil 4.14. Elde Edilen Yeni Resim

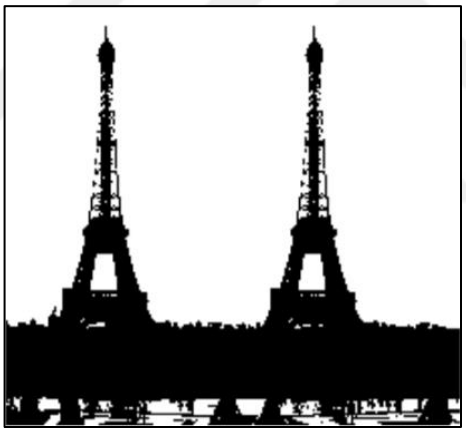
Resmin siyah beyaza dönüştürülürken resmin siyah - beyaz piksel dengesi göz önünde bulunduracak bir eşik değeri belirlenirse daha gözlemlenebilir sonuçlar elde edilebilir. Bu durumla ilgili farklı bir çözüm olarak resimdeki kenarları çıkarma ön işleme adımı eklenebilir. Böylece resmin içerdiği kenarlar önceden tespit edileceği

için resim başarılı olarak birleştirilen Şekil 4.11 ile benzer konuma gelecektir ve birleştirmede elde edilen kalite arttırabilmiş olacaktır.



(a) (b)

Şekil 4.15. Uygulamada Giriş Olarak Uygulanan Gerçek Resimler a) Resim1 b) Resim2

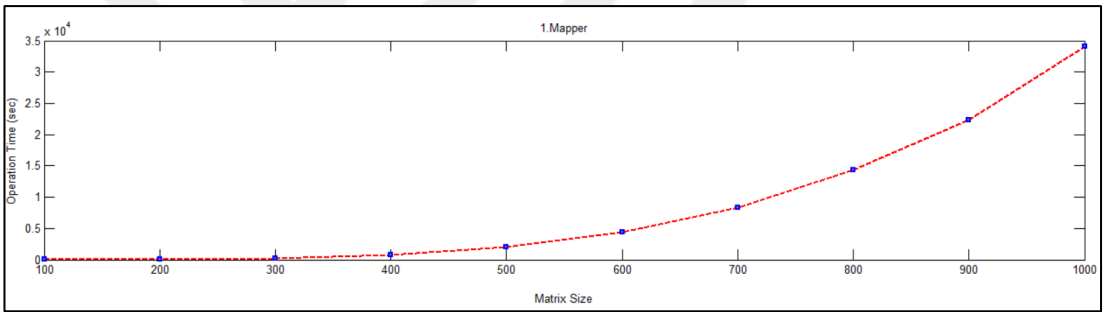


Şekil 4.16. Elde Edilen Yeni Resim

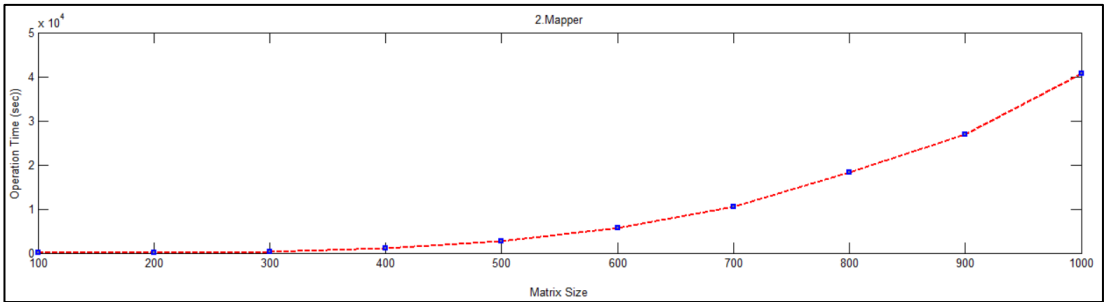
Uygulama gerçek resimler üzerinde uygulandığında birleştirme başarımı istenilen başarımı kaçırabilmektedir. Bu duruma örnek Şekil 4.15 ile gösterilmekte olan resimlerin (Şekil 4.15 a, b) birleştirilmesi işleminin sonucu (Şekil 4.16) ile verilmiştir. Resimlerin birleştirilme algoritmasının resimleri birbiri ile üst üste en çok çakışan noktanın referans alınmasıyla birleştirmesi her resim için doğruluğu sağlamamaktadır. Geliştirilen algoritmanın birleşme noktasını yakalayabilmek için daha etkin bir yaklaşıma ihtiyacı vardır.

5. SONUÇLAR VE ÖNERİLER

Yapılan çalışmada resimlerin Hadoop Eşle/İndirge modeline uygun olarak işlenebilmesine yönelik algoritma geliştirilmeye çalışılmıştır. Bir resmin metinsel olarak ifade edilerek map fonksiyonuna girdi olarak verilmesiyle çalışacak map fonksiyonunun performansı test edilmek istenmiştir. Resimlerin dosyadan okunması işlemi sırasında Hadoop dosya sisteminin kullanılması ile büyük boyutlardaki görüntüler için ölçeklenebilir bir sistem oluşturulması hedeflenmiştir. İki resmin birbiri ile en çok benzerlik içerdiği noktanın referans alınarak birleştirilmesi işlemi n^4 karmaşıklık ile çözülmeye çalışılmıştır.



Şekil 5.1. Resimlerin Yeni Uzaya Dönüşüm İşlemi İçin Matris Boyutu – İşlem Süresi (sn) Grafiği



Şekil 5.2. Resimlerin Birleştirilmesi İşlemi İçin Matris Boyutu – İşlem Süresi (sn) Grafiği

Yapılan testler uygulamanın büyük resimler için işlem yapılabilirliğini mümkün kılsa da geliştirilen algoritmanın n^4 karmaşıklığa sahip olması ve tek makine üzerinde çalışması nedeni ile süre bazındaki performansı doğrultusunda istenen performans elde edilememiştir. Şekil 5.1 ve Şekil 5.2 ile gösterilmekte olan uygulamada çalışan

map fonksiyonlarının süre performans grafiği incelendiğinde matris boyutu artışı ile işlem süresinde ciddi bir artış gözlemlenmiştir. Algoritmanın bir sonraki adımda bir küme üzerinde işlem yapılabilirliğinin sağlanması ile büyük resimler üzerinde kullanılabilirliğinin elverişli hale getirilmesi hedeflenmektedir.

Geliştirilen uygulamanın iyileştirilebilecek noktalardan bir diğeri ise resimlerin siyah beyaza dönüştürülürken eşik değerinin resmin siyah-beyaz piksel dengesine göre belirleyecek bir algoritma kullanılmasıdır. Resimlerin "128" eşik değerine göre siyah-beyaz olarak dönüştürülmesinin beyaz yoğunluğu fazla veya siyah yoğunluğu fazla olan gerçek resimlerin üzerindeki etkisinin eşleşmeleri yakalamaktaki zorluk veya eşleşmelerin bulunmasındaki doğruluğun azalması olarak ortaya çıkmasıdır.

Geliştirilen algoritma, siyah piksellerin sayılması üzerine tasarlanmıştır. Resimlerin siyah renk yoğunluğunun az olduğu durumlarda başarım sağlamıştır. Fakat resmin beyaz renk yoğunluğu fazla olduğu durumlarda kesişmelerin yakalanması zorlaşmıştır. Bir sonraki adımda resimlerin genel renk yoğunluğunu çıkaran bir ön işleme adımının eklenmesi ve kesişmelerin bu adımı referans almasıyla hesaplanması uygulamanın performansını olumlu yönde etkileyebilecek niteliktedir.

Resimlerde bulunan kenarların çıkarılması ardından resimler arasındaki eşleşme durumlarının hesaplanması resimlerin birleştirileceği noktanın tespit edilebilmesinde daha doğru ve hızlı bir sonuç üretmesinde etkili olabilir. Algoritma geliştirilirken bir sonraki adımda bu ön işleme adımlarına da yer verilebilir.

Algoritma görüntü örme işlemin dağıtık işlenebilmesine yönelik atılan küçük bir adım olarak geliştirilmiştir. "En çok kesişmenin olduğu bölge resimlerin birleştirilmesi gerektiği bölge midir ?" sorusu da algoritmanın işleyişinin güncellenmesi gerektiğini düşündürebilecek niteliktedir.

Yapılan çalışmada piksel bazlı büyük parça görüntülerinin bitmap haritasının çıkarılıp kodlanmasıyla büyük veri çatılarında işlenebilirliğinin mümkün olduğu ortaya konmuştur. Yapılan çalışma, dikgen görüntülerin örülmesi için geliştirilmiş olsa da resimlerin metinsel olarak HDFS üzerine kaydedilmesi ile diğer görüntü işleme algoritmalarının da metinsel verileri işlemek için geliştirilen Hadoop kullanılmasıyla çözülmesinin uygun olduğu gözlenmiştir.

KAYNAKLAR

[1] DeRoos R., Deutsch T., Eaton C., Lapis G, Zikopoulos P., *Understanding Big Data Analytics for Enterprise Class Hadoop and Streaming Data*, 1st ed., McGraw-Hill Osborne Media, USA, 2011.

[2] https://www.seagate.com/files/www-content/our-story/trends/files/Seagate-WP-DataAge2025-March-2017.pdf, (Ziyaret Tarihi: 25 Mayıs 2016).

[3] http://hadoop.apache.org/, (Ziyaret Tarihi: 25 Mayıs 2016).

[4] White T., *Hadoop: The Definetive Guide*, 4th ed., O'Reilly Media, Sebastopol, California, USA, 2015.

[5] Dean J., Ghemawat S., MapReduce: Simplified Data Processing on Large Clusters, *Proceedings of the 6th conference on Symposium on Opearting Systems Design and Implementation*, San Francisco, California, USA, 06-08 December 2004.

[6] https://wiki.apache.org/hadoop/TaskTracker, (Ziyaret Tarihi: 25 Mayıs 2016).

[7] Zhu H., Shen Z., Shang L., Zhang X., Parallel Image Texture Feature Extraction under Hadoop Cloud Platform, *Intelligent Computing Theory: 10th International Conference*, Taiyuan, China, 3-6 August 2014.

[8] Farahrahmand F., Homayoun H., Joshi R., Koohi A., Neshatpour K., Rafatirad S., Sasan A., Big Biomedical Image Processing Hardware Acceleration: A Case Study for K-means and Image Filtering., *Circuits and Systems (ISCAS), IEEE International Symposium on*, Montreal, Canada, USA, 22-25 May 2016.

[9] Bednarz T., Szul P., Productivity Frameworks In Big Data Image Processing Computations - Creating Photographic Mosaics With Hadoop And Scalding, *Procedia Computer Science*, 2014, **29**, 2306-2314.

[10] http://hipi.cs.virginia.edu/index.html, (Ziyaret Tarihi: 25 Mayıs 2016).

[11] Chen C.Y., Klette R., Image Stitching — Comparisons and New Techniques, *Computer Analysis of Images and Patterns: 8th International Conference*, Ljubljana, Slovenia, 1–3 September 1999.

[12] https://en.wikipedia.org/wiki/Image_stitching, (Ziyaret Tarihi: 25 Mayıs 2016).

[13] Menaka R., Pravenaa S., A Methodical Review on Image Stitching and Video Stitching Techniques, *International Journal of Applied Engineering Research*, 2016, **11**(5), 3442-3448.

[14] Uddin M., Bonny M., Feature-Based Image Stitching Algorithms, *Computational Intelligence (IWCI)*, Dhaka, Bangladesh, 12-13 December 2016.

[15] Shaikh T., Patankar A., Multiple Feature Extraction Techniques in Image Stitching, *International Journal of Computer Applications*, *Intelligent Transportation Systems Conference*, 2015, **123**(15), 35–39.

[16] Sozykin A., Epanchintsev T., Mipr – A Framework For Distributed Image Processing Using Hadoop, *Application of Information and Communication Technologies (AICT 9th International Conference*, Rostov on Don, Russia, 14-16 October 2015.

[17] Vemula S., Crick C., Hadoop Image Processing Framework, *Big Data (BigData Congress)*, New York, USA, 27 June - 2 July 2015.

[18] Dong L., Lin Z., Liang Y., He L., Zhang N., Chen Q., A Hierarchical Distributed Processing Framework for Big Image Data, *IEEE Transactions on Big Data*, 2015, **2**(4) , 297 - 309.

[19] Bednarz T., Szul P., Productivity Frameworks In Big Data Image Processing Computations - Creating Photographic Mosaics With Hadoop And Scalding, Procedia Computer Science, 2014, **29**, 2306-2314.

[20] Wu T.Y., Chen C.Y., Kuo L.S., Lee W.T., Chao H.C.., Cloud Based Image Processing System With Priority Based Data Distribution Mechanism, *Computer Communications*, 2012, **35**(15), 1809–1818.

[21] Almeer M.H., Cloud Hadoop Map Reduce For Remote Sensing Image Analysis, *J Emerg Trends Comput Inf Sci.*, 2012, **3**(4), 637–644.

[22] Barapatre H., Nirgun V, Ginde J, Image Processing Using Mapreduce With Performance, *International Journal of Emerging Technology and Innovative Engineering*, 2015, **1**(4), 21-27.

[23] Yamamoto M., Kaneko K., Parallel Image Database Processing With Mapreduce And Performance Evaluation In Pseudo Distributed Mode, *International Journal of Electronic Commerce Studies*, 2012, **3**(2), 211-228.

[24] Bajcsy P., Vandecreme A., Amelot J., Nguyen P., Chalfoun J., Brady M., Terabyte-Sized Image Computations On Hadoop Cluster Platforms, *Big Data IEEE International Conference on*, California, USA, 6-9 October 2013.

[25] White B., Yeh T., Lin J., Davis L. Web-Scale Computer Vision Using Mapreduce For Multimedia Data Mining, *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, Washington, USA, 25 July 2010.

[26] Demir İ., Sayar A., Hadoop Plugin For Distributed And Parallel Image Processing, *Signal Processing and Communications Applications Conference*, Mugla, Turkey, 18-20 April 2012.

[27] Ramos-Pollán R., González F.A., Caicedo J.C., Cruz-Roa A., Camargo J.E., Vanegas J.A., Bigs: A Framework For Large-Scale Image Processing And Analysis Over Distributed And Heterogeneous Computing Resources, *E-Science (e-Science),* Chicago, USA, 8-12 October 2012.

[28] Malakar R., Vydyanathan N., A CUDA-Enabled Hadoop Cluster For Fast Distributed Image Processing, *Parallel Computing Technologies Parcomptech*, Bangalore, India, 21-23 February 2013.

[29] Moise D., Shestakov D., Gudmundsson G., Amsaleg L., Terabyte-Scale Image Similarity Search: Experience And Best Practice, *Big Data, 2013 IEEE International Conference on,* California, USA, 6-9 December 2013.

[30] Ali M., Kumar J., Implementation Of Image Processing System Using Handover Technique With Map Reduce Based On Big Data In The Cloud Environment, *The International Arab Journal of Information Technology*, 2016, **13**(2), 1067-1071.

[31] Wang W., Zhao, W., Cai C., Huang J., Xu X., Li, L., An Efficient Image Aesthetic Analysis System Using Hadoop, *Signal Processing: Image Communication*, 2015, **39**, 499–508.

[32] Wu D., Li Z., Bie R., Zhou M., Research on Database Massive Data Processing and Mining Method based on Hadoop Cloud Platform, *Identification, Information and Knowledge in the Internet of Things (IIKI),* Beijing, China, 17-18 October 2014.

[33] Zhang G., Wu Q., Zhuo Z., Wang X., Lin X., A Large Scale Images Processing Model Based on Hadoop Platform, *Proceedings of the Second International Conference on Innovative Computing and Cloud Computing*, Wuhan, China, 01-02 December 2013.

[34] Arietta S., Lawrence J., Liu L., Sweeney C., HIPI: A Hadoop Image Processing Interface for Image-based MapReduce Tasks, Undergraduate Thesis, Computer Science, University of Virginia, 2011.

[35] Sunny B., Ramesh R, Abraham V.,Vikas V, Map-Reduce Based Framework for Instrument Detection in Large Scale Surgical Videos, *Control Communication & Computing India (ICCC)*, Trivandrum, India, 19-21 November 2015.

[36] Pan X., Zhang S., A Remote Sensing Image Cloud Processing System Based On Hadoop, *Cloud Computing and Intelligent Systems (CCIS)*, Hangzhou, China, 30 October - 1 November 2012.

[37] Golpayegani N., Halem M., Cloud Computing for Satellite Data Processing on High End Compute Clusters, *IEEE International Conference on Cloud Computing,* Bangalore, India, 21-25 September. 2009.

[38] Lv Z., Hu Y., Zhong H., Wu J., Li B., Zhao H. Parallel K-Means Clustering of Remote Sensing Images Based on MapReduce, *International Conference On Web Information Systems And Mining*, Sanya, China, 23-24 October 2010.

[39] Ermias B.T., Distributed Processing Of Large Remote Sensing Images Using MapReduce: A case of Edge Detection, Master Thesis, Institute for Geoinformatics, Universität Münster, Münster, North-Rhine Westphalia, Germany, 2011.

[40] Cavallaro G., Riedel M., Richerzhagen, M.; Benediktsson J. A., Plaza A., On Understanding Big Data Impacts in Remotely Sensed Image Classification Using Support Vector Machine Methods, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2015, **8**(10), 4634 - 4646.

[41] Chebbi I., Boulila W.Farah I.R., Improvement Of Satellite Image Classification: Approach Based On Hadoop/Mapreduce, *Advanced Technologies for Signal and Image Processing (ATSIP)*, Monastir, Tunisia, 21-23 March, 2016.

[42] Sarade S.D., Ghule N.B., Disale S.P., Sasane S.R., Large Scale Satellite Image Processing Using Hadoop Distributed System, *International Journal of Advanced Research in Computer Engineering & Technology*, 2015, **3**(3), 731-735.

[43] Eldawy A., Alharthi S., Alzaidy A., Daghistani A., Ghani S., Basalamah S., A Demonstration Of Shahed: A Mapreduce-Based System For Querying And Visualizing Satellite Data, *Data Engineering (ICDE),* Seoul, South Korea, 13-17 April 2015.

[44] Park J.W., Lee Y.W., Yun C.H., Park H.K., Chang S.I., Lee I.P., Cloud Computing For Online Visualization Of Gis Applications In Ubiquitous City, *CLOUD COMPUTING 2010 : The First International Conference on Cloud Computing, GRIDs, and Virtualization,* Lisbon, Portugal, 21-26 November 2010.

[45] Zhao L., Chen L., Ranjan R., Choo K., He J., Geographical Information System Parallelization For Spatial Big Data Processing: A Review, *Cluster Computing,* 2016, **19**(139), 139–152.

[46] Eldawy A., Mokbel M.F., A Demonstration of SpatialHadoop: An Efficient Mapreduce Framework for Spatial Data, *Proc. VLDB Endow.*, 2013, **6**(12), 1230-1233.

[47] Aji A., Wang F., Vo H., Lee R., Liu Q., Zhang X., Hadoop GIS: A High Performance Spatial Data Warehousing System Over Mapreduce, *Proc. VLDB Endow.*, 2013, **6**(12), 1009-1020.

[48] Wiley K., Connolly A., Gardner J., Krughoff S., Balazinska M., Howe B., Kwon Y., Bu Y., Astronomy In The Cloud: Using Mapreduce For Image Co-Addition, *Publications of the Astronomical Society of the Pacific*, 2011, **123**(901), 366-380.

[49] Chang G., Malhotra S., Wolgast P., Leveraging The Cloud For Robust And Efficient Lunar Image Processing, *Aerospace Conference*, Big Sky, Montana, USA, 5-12 March 2011.

[50] Banda J.M., Schuh M.A., Angryk R.A., Big Data New Frontiers: Mining, Search And Management Of Massive Repositories Of Solar Image Data And Solar Events, *17th East European Conference on Advances in Databases and Information Systems*, 1-4 September 2013.

[51] Eken S., Sayar A., A MapReduce based Big Data Framework for Object Extraction from Mosaic Satellite Images, *International Conference on Internet of Things and Big Data (Doctoral Consortium)*, Roma, Italy, 23-25 April 2016.

[52] Eken S., Sayar A., Uydu Görüntülerinin Yüksek Performansta İşlenmesi Üzerine Bir İnceme: Vektör Tabanli Mozaik Örme Durum Çalışması, *6. Uzaktan Algılama ve CBS Sempozyumu*, Adana, Türkiye, 5-7 Ekim 2016.

[53] Rajak R., Raveendran D., Bh M.C., Medasani S.S., High Resolution Satellite Image Processing Using Hadoop Framework, *Cloud Computing in Emerging Markets (CCEM)*, Bangalore, India, 25-27 November 2016.

[54] Kubias A., Deinzer F., Feldmann T., Paulus D., Schreiber B., Brunner T., 2D/3D Image Registration On The GPU, *Pattern Recognit. Image Anal*, 2008, **18**(381), 381–389.

[55] Shamonin D.P., Bron E.E., Lelieveldt B.P.F., Smits M., Klein S., Staring M. , Fast Parallel Image Registration on CPU and GPU for Diagnostic Classification of Alzheimer's Disease, *Frontiers in Neuroinformatics*, 2014, **7**(50), 1-15.

[56] Kim H., Parashar M., Foran D. J., Yang L., Investigating the use of autonomic cloudbursts for high-throughput medical image registration, *Grid Computing, 2009 10th IEEE/ACM International Conference on*, Canada, USA,13-15 October 2009.

[57] Diaz J., Laszewski G., Wang F., Fox G., Abstract Image Management And Universal Image Registration For Cloud And HPC Infrastructures, *Cloud Computing (CLOUD), IEEE 5th International Conference on*, Honolulu, Hawaii, USA, 24-29 June 2012.

[58] Warfield S.K., Jolesz F.A., Kikinis R., A High Performance Computing Approach To The Registration Of Medical Imaging Data, *Parallel Comput*, 1998, **24**(9), 1345–1368.

[59] Adam M., Jung C., Roth S., Brunnett G., Real-time Stereo-Image Stitching Using GPU-Based Belief Propagation, Editors: Magnor M. A., Rosenhahn B., Theisel H., *VMV*, 2009.

[60] Ino F., Ooyama K., Hagihara K., A Data Distributed Parallel Algorithm for Nonrigid Image Registration, *Parallel Computing*, 2005, **31**(1), 19-43.

EKLER

EK-A

Hadoop Kurulumu

1. Ubuntu 14_04 LTS sürümü VirtualBox 5.0.10 üzerine HADOOP kurulmuştur.
2. Oracle Java 8 kurulumu gerçekleştirilmiştir. Bu işlem adımı için terminal üzerinde Şekil A.1 ile gösterilen komutlar çalıştırılmıştır.

```
hduser@hayrunnisa-VirtualBox:~$ sudo apt-get install software-properties-co
mmon

hduser@hayrunnisa-VirtualBox:~$ sudo add-apt-repository ppa:webupd8team/jav
a

hduser@hayrunnisa-VirtualBox:~$ sudo apt-get update

hduser@hayrunnisa-VirtualBox:~$ sudo apt-get install oracle-java8-installer
```

Şekil A.1. Oracle Java 8 kurulumu komutları

3. HDFS ve Mapreduce erişimi için bir grup ve kullanıcı oluşturulmuştur. Bu işlem için terminal üzerinde Şekil A2 ile gösterilen komutlar çalıştırılmıştır.

```
hduser@hayrunnisa-VirtualBox:~$ sudo addgroup hadoop

hduser@hayrunnisa-VirtualBox:~$ adduser --ingroup hadoop hduser
```

Şekil A.2. Terminal üzerinden kullanıcı ve grup oluşturma

4. Oluşturulan kullanıcıya sudo yetkisi vermek için terminal üzerinde Şekil A.3 ile gösterilen komut çalıştırılmıştır.

```
hduser@hayrunnisa-VirtualBox:~$ sudo adduser hduser sudo
```

Şekil A.3. Kullanıcıya sudo yetkisi verme

5. Hduser kullanıcısı için anahtar oluşturma işlemi gerçekleştirilir. Oluşturulan açık anahtar authorized_keys dosyasına yazılır. Master düğümün açık anahtarı slave düğümlerde ssh klasörünün altındaki authorized_keys dosyasına kaydedilir. Bu işlem Şekil A.4 ile gösterilmiştir. Bu işlem adımı düğümlerin birbirine erişebilmesi için gerçekleştirilir

Şekil A.4. Kullanıcı için terminal üzerinden anahtar oluşturma adımı

6. Hadoop kurulumunda ipv6 kapatmak için /etc/sysctl.conf dosysına Şekil A.5 ile gösterilen komut çalıştırılarak dosya içinde düzenleme işlemi gerçekleştirilmiştir.

```
hduser@hayrunnisa-VirtualBox:~$ sudo nano /etc/sysctl.conf

   # disable ipv6

   net.ipv6.conf.all.disable_ipv6 = 1

   net.ipv6.conf.default.disable_ipv6 = 1

   net.ipv6.conf.lo.disable_ipv6 = 1
```

Şekil A.5. Terminal üzerinden ipv6 kapatma işlemi

7. 6. Adımda uygulanan işlemin doğruluğu aşağıda bulunan satırlar terminal üzerinde yazılarak test edilmiştir. Şekil A.6 ile gösterilen komut çalıştırıldığında 1 sonucunu döndürmesi gerekmektedir.

```
cat /proc/sys/net/ipv6/conf/all/disable_ipv6
```

Şekil A.6. ipv6 kapatılma durumunu test etme komutu

8. http://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-2.7.1/hadoop-2.7.1-src.tar.gz linki üzerinden Apache Hadoop indirilir.
9. İndirilen hadoop-2.7.1-src.tar.gz dosyası /usr/local/hadoop dizinine çıkarılır. Bu işlem için terminal üzerinde Şekil A.7 ile gösterilen komutlar çalıştırılmıştır.

```
hduser@hayrunnisa-VirtualBox:/usr/local$ sudo tar -xzvf hadoop-2.7.1.tar.g
z

hduser@hayrunnisa-VirtualBox:/usr/local$ sudo mv hadoop-2.7.1 /usr/local/ha
doop
```

Şekil A.7. Hadoop dosyalarının /usr/local/hadoop dizinine çıkarılması

10. Namenode ve Datanode için hadoop temp dizini açılır. Bu işlem için terminal üzerinde Şekil A.8 ile gösterilen komutlar çalıştırılmıştır.

```
hduser@hayrunnisa-VirtualBox:/usr/local$ sudo mkdir -p /usr/local/hadoop_tm
p/hdfs/namenode

hduser@hayrunnisa-VirtualBox:/usr/local$ sudo mkdir -p /usr/local/hadoop_tm
p/hdfs/datanode
```

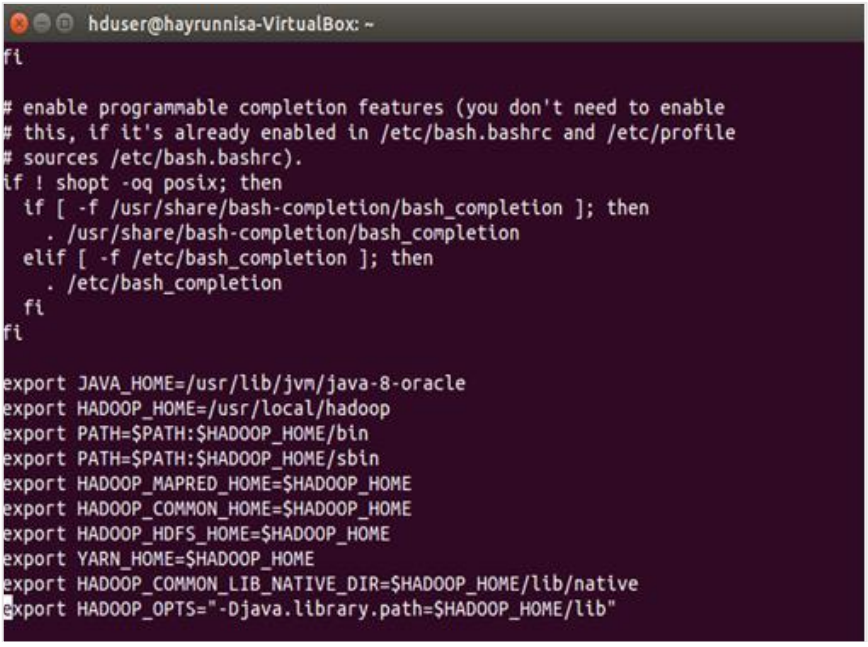
Şekil A.8. Namenode ve Datanode için hadoop_tmp dizininin açılması

11. Hadoop klasörünün sahiplik ayarlarının yapılması için Şekil A.9 ile gösterilen komutlar çalıştırılmıştır.

```
hduser@hayrunnisa-VirtualBox: /usr/local$ sudo chown hduser:hadoop -R /usr/
local/hadoop_tmp/
```

Şekil A.9. Hadoop klasörünün sahiplik ayarlarının düzenlenmesi

12. Hadoop içerisindeki dosyalarda konfigürasyon işlemleri gerçekleştirilir.
    a) $HOME/.bashrc dosyasının konfigürasyonu aşağıdaki Şekil A.10 ile gösterilmiştir.

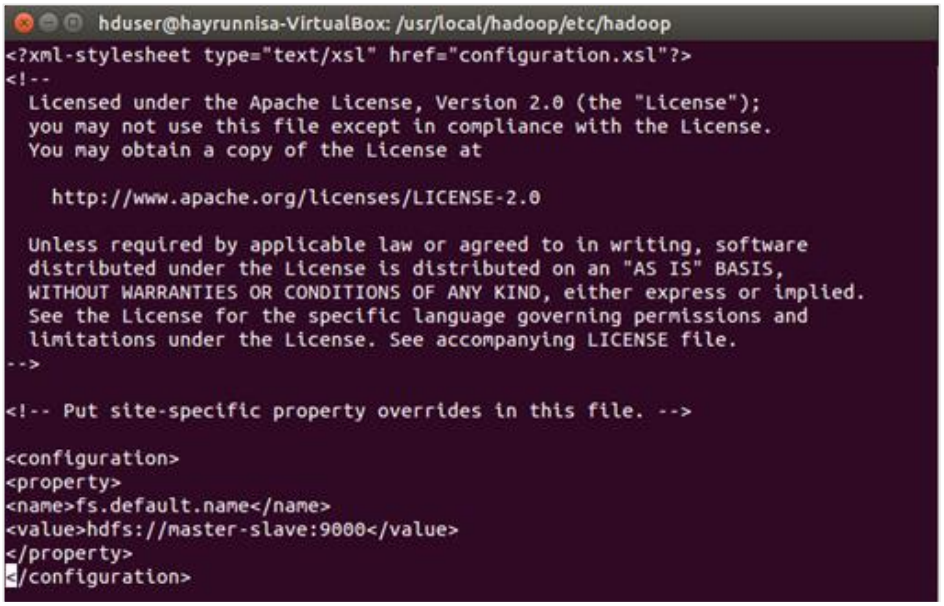


Şekil A.10. $HOME/.bashrc dosyasının konfigürasyonu

b) Hadoop tarafından kullanılan ortam değişkenlerini içerdiği dosya olan hadoop-env.sh dosyasının konfigürasyonu Şekil A.11 ile gösterilmiştir.

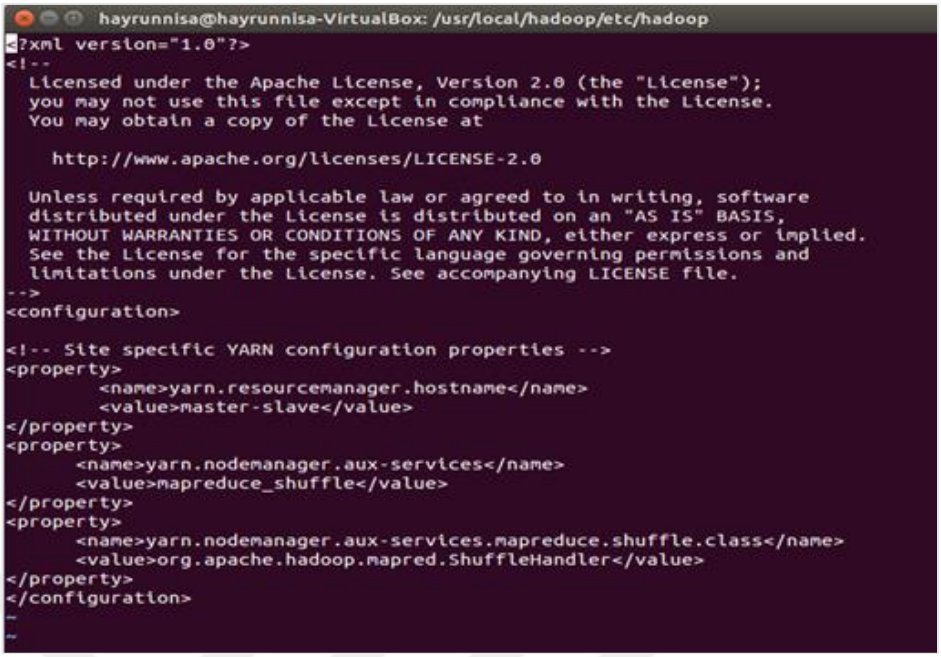
Şekil A.11. "hadoop-env.sh" dosyasının konfigürasyonu

c) Hadoop'un başlatıldığında kullanmış olduğu konfigürasyon özelliklerini içeren core-site.xml dosyasının konfigürasyonu Şekil A.12 ile gösterilmiştir.



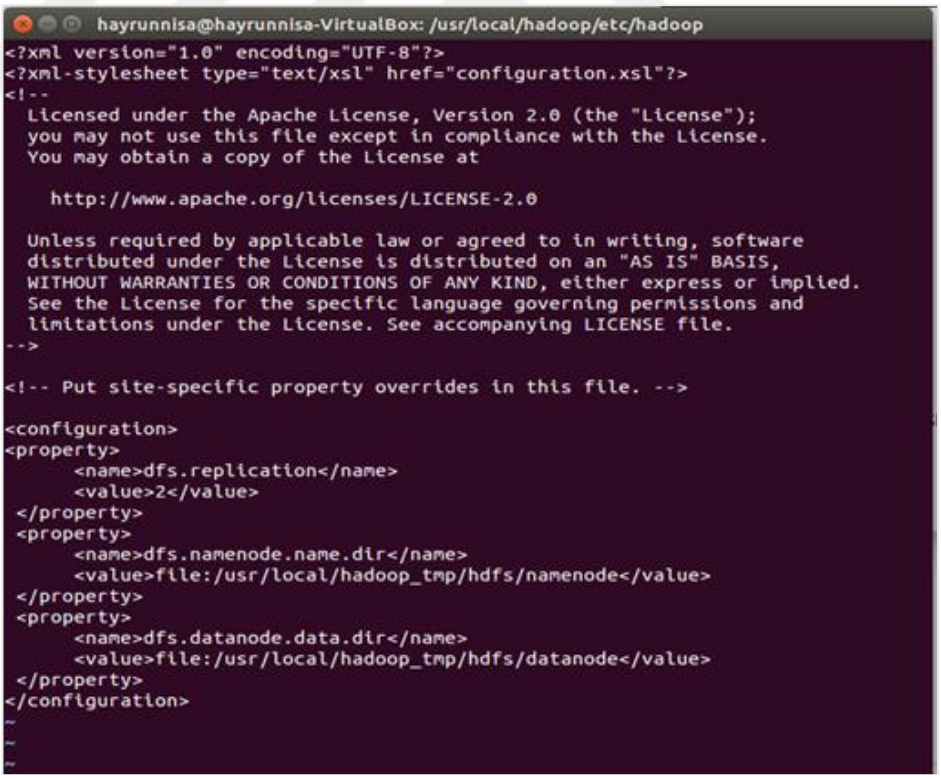
Şekil A.12. "core-site.xml" dosyasının konfigurasyonu

d) Yarn-site.xml dosyasının konfigürasyonu aşağıdaki Şekil A.13 ile gösterilmiştir.



Şekil A.13. “yarn-site.xml” dosyasının konfigürasyonu

e) Hdfs-site.xml dosyasının konfigürasyonu Şekil A.14 ile gösterilmiştir.



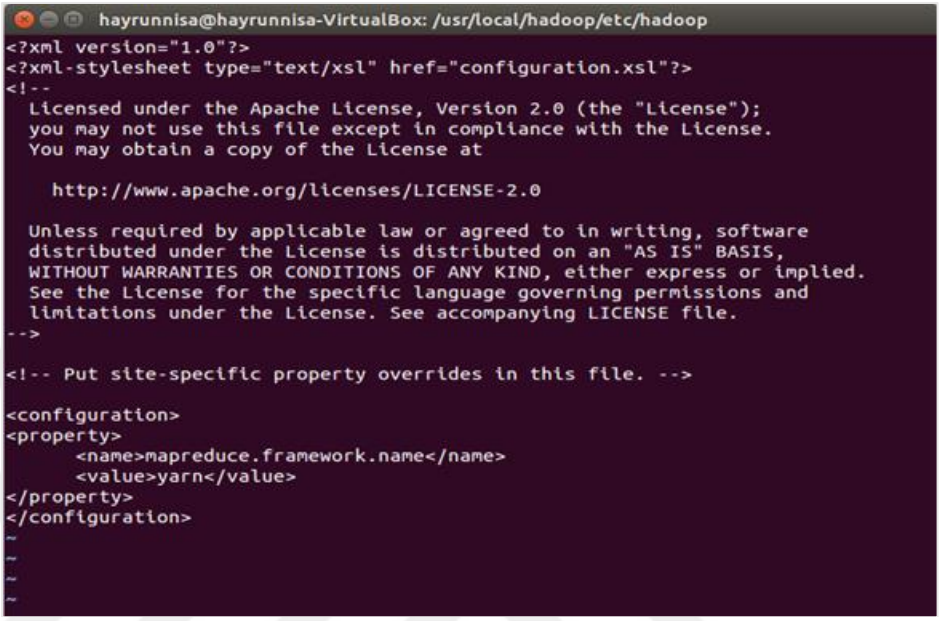
Şekil A.14. “hdfs-site.xml” dosyasının konfigürasyonu

f) Varsayılan olarak /usr/local/hadoop/etc/hadoop/ dosyası mapred-site. xml.template dosyasını içerir. Bu dosyanın adı mapred-site.xml olarak güncellenir. Bu işlem için gerekli kodlar Şekil A.15 ile gösterilmiştir.

```
hduser@hayrunnisa-VirtualBox:/usr/local$ sudo mv mapred-site.xml.template m
apred-site.xml
```

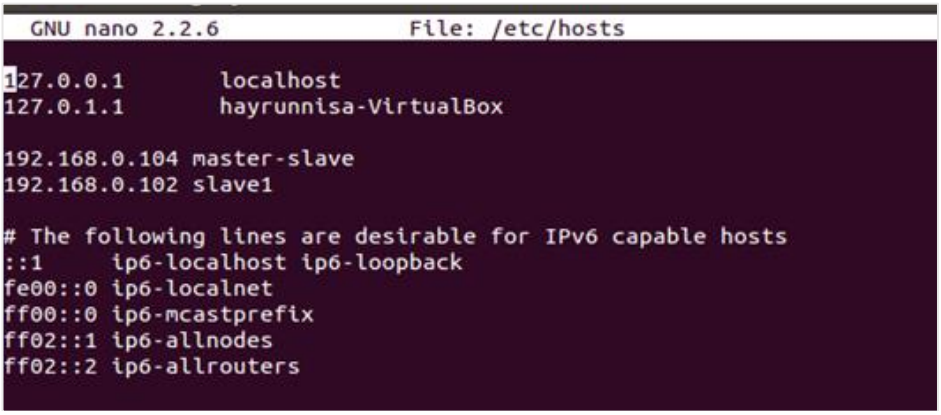
Şekil A.15. “mapred-site.xml.template” dosyasının adının güncellenmesi

g) Mapred-site.xml dosyasının konfigürasyonu Şekil A.16. ile gösterilmiştir.



Şekil A.16. “mapred-site.xml” dosyasının konfigürasyonu

h) “ssh” ile masterdan slavelere bağlanabilmek için /etc/hosts dosyasına master ve slavelerin ip adresleri kaydedilir. İşlem adımı Şekil A.17 ile gösterilmiştir.



Şekil A.17. /etc/hosts dosyasının güncellenmesi

13. Namenode’un formatlanması için terminal üzerinde Şekil A.18 ile gösterilen komutlar çalıştırılmıştır.

```
hduser@hayrunnisa-VirtualBox:~$ cd /usr/local/hadoop

hduser@hayrunnisa-VirtualBox:/usr/local/hadoop$ bin/hdfs namenode -format
```

Şekil A.18. Namenode format işlemi

14. Master node üzerinde uygulanan Hadoop konfigurasyon işleminin, slave'ler üzerinde de uygulanabilmesi için Şekil A.19. ile gösterilmekte olan komut terminal üzerinde çalıştırılmıştır.

```
/usr/local/hadoop/etc/hadoop$ for i in `cat /usr/local/hadoop/etc/hadoop/slaves` ; do echo $i;
rsync -avxP --exclude=logs /usr/local/hadoop/ $i:/usr/local/hadoop/ ; done
```

Şekil A.19. Master node üzerinde uygulanan Hadoop konfigurasyon işleminin, slave'ler üzerinde de uygulanması

15. Namenode ve datanode'un başlatılması için Şekil A.20. ile gösterilen komut satırı terminal üzerinde çalıştırılmıştır.

```
master@hayrunnisa-VirtualBox:/usr/local/hadoop$ sbin/start-dfs.sh
15/12/28 23:42:55 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-master-na
menode-hayrunnisa-VirtualBox.out
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-master-da
tanode-hayrunnisa-VirtualBox.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-ma
ster-secondarynamenode-hayrunnisa-VirtualBox.out
15/12/28 23:43:11 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
master@hayrunnisa-VirtualBox:/usr/local/hadoop$ sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-master-resource
manager-hayrunnisa-VirtualBox.out
localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-master-n
odemanager-hayrunnisa-VirtualBox.out
```

Şekil A.20. Namenode ve datanode'un başlatılması

16. Jps ile hadoop deamonları aşağıdaki Şekil A.21 ile gösterilmiştir.

```
master@hayrunnisa-VirtualBox:/usr/local/hadoop$ jps
8640 SecondaryNameNode
8800 ResourceManager
8441 DataNode
8937 NodeManager
8283 NameNode
13548 Jps
```

Şekil A.21. jps komutu

KİŞİSEL YAYIN VE ESERLER

[1] **Sarı H.**, Eken S., Sayar A., An Approach for Stitching Satellite Images in a Bigdata MapReduce Framework, *4th International Workshop on Geoinformation Science: GeoAdvances*, Karabuk, Turkey , 14-15 October 2017.

[2] Eken S., **Sarı H.**, Uysal O.**,** Sayar A., Distributed Approach for Point Set Pattern Matching: A Case study on Simple Objects, *5th High Performance Computing Conference (BAŞARIM 2017)*, Istanbul, Turkey, 14-15 September 2017.

[3] **Sarı H.**, Sayar A., A Study for Adaptation of Image Stitching to Big Data Frameworks, *International Conference On Mathematics And Engineering (ICOME 2017)*, Istanbul, Turkey, 10 - 12 May 2017.

[4] Eken S., Karabaş A.**, Sarı H.,** Sayar A., A Framework for Recognition and Animation of Chess Moves Printed on a Chess Book, *The International Arab Journal of Information Technology (IAJIT)*, 2018, **15**(1), 29-36.

ÖZGEÇMİŞ

Hayrunnisa SARI 1993'te İzmit'te doğdu. Lise öğrenimini Merkez Bankası Derince Anadolu Lisesi'nde tamamladı. 2011 yılında girdiği Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümü'nden 2015 yılında mezun oldu. Aynı yıl içinde Kocaeli Üniversitesi Bilgisayar Mühendisliği Anabilim Dalı'nda yüksek lisans eğitimine başladı. Yüksek lisans eğitiminde görüntü örme işleminin büyük veri işleme mimarisine uyarlanabilmesine yönelik algoritma geliştirme konusunda çalışmıştır.