
GAZİ ÜNİVERSİTESİ
Gazi
1926

İNTERNET'TE HETEROJEN VERİ KAYNAKLARINDAN VERİ TOPLANMASI, ENTEGRASYONU VE GÜNCELLENMESİ

Zülfü ALANOĞLU

DOKTORA TEZİ
BİLİŞİM SİSTEMLERİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

OCAK 2024

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

İmza

Zülfü ALANOĞLU

...../……/……

İNTERNET'TE HETEROJEN VERİ KAYNAKLARINDAN VERİ TOPLANMASI, ENTEGRASYONU VE GÜNCELLENMESİ

(Doktora Tezi)

Zülfü ALANOĞLU

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Aralık 2023

ÖZET

Günümüzde Web hızla büyüyen ve çok çeşitli verinin bulunduğu en büyük veri kaynağıdır. Kullanıcılar bu veri kaynağından istedikleri veriyi almak için genellikle arama motorlarını kullanmaktadırlar. Arama motorları bu verileri Web tarayıcıları aracılığı ile elde ederler. Web tarayıcıları Web sayfalarındaki verileri toplar, ayrıştırır ve indeksleyip saklarlar. Web tarama sürecindeki en önemli konular; hangi URL'lerden başlanacağı, taramanın kapsamı ve veri deposunun güncel tutulması için kullanılan güncelleme algoritmasıdır. Bu tez çalışmasında ilk olarak kapsamı tüm Web olan genel bir tarayıcının tohum URL seçim ve kapsam genişletme yöntemleri sunulmuştur. Tohum URL seçiminde bazı ölçütlere dayanarak üç farklı tohum URL veri kümesi oluşturulmuş ve performansları detaylı bir şekilde analiz edilmiştir. Ayrıca kapsamı hızlı bir şekilde genişletmek için link skoruna dayalı yeni bir tarama algoritması önerilmiştir. Tohum URL kümelerinden başlanarak taramalar yapılmış, sonuçlar karşılaştırılmış ve detaylı analizleri yapılmıştır. İkinci olarak arama motorlarının performanslarını etkileyen en önemli özellik olan tekrar ziyaret sürelerinin belirlenmesi amacıyla EMA yöntemi önerilmiştir. EMA yöntemi kullanılarak geliştirilen EMACrawler'ın kesinlik, toplam kapsama alanı ve verimlilik ölçütleri kullanılarak test ve analiz işlemleri gerçekleştirilmiştir. Yapılan deneysel çalışmaların sonucunda, EMACrawler'ın güncel verilerin elde edilmesi ve veri ambarlarının tazeliğinin korunmasında başarılı olduğu görülmüştür.

Bilim Kodu	:	92408
Anahtar Kelimeler	:	Web Tarayıcıları, Tohum URL Seçimi, Kapsam Genişletme, Link Skoru Hesaplama
Sayfa Adedi	:	91
Danışman	:	Prof. Dr. M. Ali AKCAYOL

COLLECTING, INTEGRATING AND UPDATING DATA FROM HETEROGENOUS DATA SOURCES ON THE INTERNET

(P.h.D. Thesis)

Zülfü ALANOĞLU

GAZİ UNIVERSITY

INSTITUTE OF INFORMATICS

December 2023

ABSTRACT

Today, the Web is the largest source of data that is rapidly growing and contains a wide variety of data. Users typically use search engines to retrieve the data they desire from this data source. Search engines obtain this data through Web crawlers. Web crawlers collect, parse, index and store data on Web pages. The most important issues in the Web crawling process are which URLs to start from, the scope of the crawl, and the update algorithm used to keep the data repository up-to-date. In this thesis study, firstly, methods for seed URL selection and scope expansion of a general Web crawler with a scope covering the entire Web are presented. Three distinct sets URL datasets were created based on specific criteria for seed URL selection, and their performances were thoroughly analyzed. Additionally, a novel crawling algorithm, grounded on link score, was proposed for the swift expansion of scope. Crawls were initiated from seed URL sets, results were compared, and detailed analyses were performed. Secondly, the EMA method was proposed to determine revisit times, which is the most important feature affecting the performance of search engines. Testing and analysis were carried out using the accuracy, total coverage and efficiency criteria of EMACrawler, which was developed using the EMA method. As a result of the experimental studies, it has been observed that EMACrawler is successful in obtaining up-to-date data and maintaining the freshness of data warehouses.

Science Code	:	92408
Key Words	:	Web Crawler, Seed URL Selection, Scope Expansion, Link Score Calculation
Page Number	:	91
Supervisor	:	Prof. Dr. M. Ali AKCAYOL

TEŞEKKÜR

Doktora eğitimim süresince bilgi ve birikiminden yararlandığım, öğrencisi olmaktan büyük onur ve gurur duyduğum, karşılaştığım zorluklarda bana yol gösterip güç veren, akademik duruşu ile her daim örnek olan ve böyle bir çalışmanın yürütülmesi boyunca şahsıma duyduğu güven ve sağladığı destek için değerli hocam ve danışmanım Prof. Dr. M. Ali AKCAYOL'a teşekkür eder, saygı ve şükranlarımı sunarım.

Tez çalışmasını yaptığım süre boyunca benden desteklerini esirgemeyen, yapıcı görüş ve önerileri ile tezimin gelişmesine katkı sağlayan değerli tez izleme komitesi üyeleri Prof. Dr. Fırat HARDALAÇ ve Dr. Öğr. Üyesi Tülin ERÇELEBİ AYYILDIZ'a, tez ve proje çalışması boyunca her türlü desteği veren değerli arkadaşım Arş. Gör. KUBİLAY AYTURAN'a içten teşekkürlerimi sunarım.

İlk günden beri desteğini, sabrını, fedakârlığını ve sevgisini esirgemeyen değerli eşim Reyhan ALANOĞLU'na, motivasyon kaynağım, hayat enerjim ve varlığına şükrettiğim oğullarım Ali Ensar ALANOĞLU ve Ahmet Hamza ALANOĞLU'na, bugünlere gelmemde emeklerini, sevgisini ve desteğini eksik etmeyen ve evlatları olmaktan mutluluk duyduğum sevgili annem Kezibe ALANOĞLU ve sevgili babam Zülküf ALANOĞLU'na, ihtiyacım olduğunda her daim yanımda olan, aynı ailede olduğumuz için kendimi çok şanslı hissettiğim kardeşlerim ve yeğenlerime sonsuz teşekkürlerimi sunarım.

Son olarak doktora öğrenimim boyunca TÜBİTAK BİDEB- 2244 / Üniversite Sanayi İşbirliği Modeli, 118C127 numaralı "İnternet'te Heterojen Veri Kaynaklarından Veri Toplanması, Doğrulanması ve Sorgulanması" isimli proje kapsamında burs desteğini sağlayan Türkiye Bilimsel ve Teknolojik Araştırma Kurumu'na (TÜBİTAK) ve Huawei Telekomünikasyon Ltd. Şti.'ye teşekkür ederiz.

İÇİNDEKİLER

Sayfa

ÇİZELGELERİN LİSTESİ

Sayfa

ŞEKİLLERİN LİSTESİ

Sayfa

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
GHz	GigaHertz
GB	GigaByte

Kısaltmalar	**Açıklamalar**
1DCNN	Tek boyutlu Evrişimli Sinir Ağları (1 Dimensional Convolutional Neural Network)
ABC	Yapay Arı Kolonisi (Artificial Bee Colony)
API	Uygulama Programlama Arabirimleri (Application Programming Interface)
BFS	Geniş Öncelikli Arama (Breadth First Search)
CPU	Merkezi İşlem Birimi (Central Process Unit)
CSV	Virgülle Ayrılan Değerler (Comma-separated Values)
DFS	Derin Öncelikli Arama (Depth First Search)
DOM	Belge Nesne Modeli (Document Object Model)
DRUM	Güncelleme Yönetimi ile Disk Deposu (Disk Repository with Update Management)
EMA	Üstel Hareketli Ortalama (Exponential Moving) Average
FIFO	İlk Giren İlk Çıkar (First İn First Out)
FTP	Dosya Transfer Protokolü (File Transfer Protokol)
HITS	Bağlantı Dahil Metin Araması (Hyperlink Included Text Search)
HTML	Hiper Metin İşaretleme Dili (HyperText Markup Language)

Kısaltmalar	Açıklamalar
ID	Kimlik Tanımı (Identity Definition)
IoT	Nesnelerin İnternet'i (Internet of Things)
IP	İnternet Protokolü (Internet Protokol)
LSTM	Uzun Kısa Süreli Bellek (Long Short-Term Memory)
PMF	Olasılık Kütle Fonksiyonunu (Probability Mass Function)
RAM	Rastgele Erişimli Bellek (Random Access Memory)
RSS	Çok Basit Besleme (Really Simple Syndication)
SMA	Basit Hareketli Ortalama (Simple Moving Avarage)
TAOS	Trafiğe Uyarlanabilir Optimum Güncelleme Şemasını (Traffic Adaptive Optimum updating Scheme)
UPWPR	Kullanıcı Tercihi Tabanlı Ağırlıklı Sayfa Sıralaması (Preference Based Weighted Page Ranking)
URL	Tekdüzen Kaynak Bulucu (Uniform Resource Loader)
XML	Genişletilebilir İşaretleme Dili (Extensible Markup Language)
ZMQ	Sıfır Mesaj Kuyruğu (Zero Message Queue)

1. GİRİŞ

Teknoloji geliştikçe İnternet'teki veri miktarı, veri türleri ve verilerin bulunduğu veri kaynaklarının sayısı hızla artmaktadır. Bu büyümeye paralel olarak saklanan verilerin boyutu çok artmış ve "Büyük Veri" olarak isimlendirilen bir araştırma alanı oluşmuştur [1]. Veri günümüzde tüm alanlarda en değerli kaynaklardan biridir. Çeşitli kaynaklardan devasa verileri elde etmek için büyük veri teknolojileri ve yöntemleri kullanılmaktadır. Özellikle son yıllarda İnternet ve bulut bilişimin hızla gelişmesiyle birlikte her alanda ve sektörde büyük bir değişim ortaya çıkmıştır. Gartner'a göre büyük veri, gelişmiş karar verme, iç görü keşfi ve süreç optimizasyonu sağlamak için yeni işleme biçimleri gerektiren yüksek hacimli, yüksek hızlı ve/veya çok çeşitli bilgi varlıklarıdır [2].

We Are Social and Hootsuite'in en son Global Digital 2022 raporları [3], İnternet kullanıcılarının son 10 yılda iki kattan fazla artarak 2012 yılında 2.18 milyar seviyesindeyken 2022 yılında 4.95 milyara çıktığını ortaya koymaktadır. En son veriler, İnternet kullanıcılarının son bir yılda 192 milyon arttığını ve bunun 2021 yılında yıllık yüzde dörtlük büyümeyle sonuçlandığını göstermektedir. Bu büyüme sonucunda veri tabanlarındaki veri miktarı da hızla artmaktadır. Her geçen gün kamu kurumları, sağlık kuruluşları, eğitim kurumları gibi veri kaynakları İnternet'e devasa miktarda ve heterojen yapıda veri yüklemektedir [4]. İnternet'i sadece insanlar değil aynı zamanda cihazlar da kullanmaktadır. Nesnelerin İnternet'i (Internet of Things - IoT) önemli bir araştırma alanı olarak karşımıza çıkmaktadır [5]. İnternet'e bağlı cihazlar algılayıcılar yardımı ile her an veri akışı gerçekleştirmektedir. Bunun yanı sıra çeşitli sosyal süreçlerin dijitalleşmesi Web verilerinin muazzam derecede artmasına neden olmuş, hayatımıza büyük veri kavramı girmiş ve son zamanlarda önemli bir araştırma alanı haline gelmiştir [6].

Büyük veri, sosyal medya, meteoroloji, finans, deneysel fizik, telekomünikasyon, askeri gözetim, iş bilişimi ve çevre araştırmaları gibi çok çeşitli alanlarda kullanılmaktadır. Büyük veri genellikle yapılandırılmamış ya da yarı yapılandırılmış veriye sahiptir. Büyük veri kaynaklarından alınan bu verilerin katma değerli servisler oluşturabilmesi ve hızlı analiz edilmesi için uygun bir yapıda depolanması gerekmektedir. Pääkkönen ve Pakkala çalışmalarında belirli bir bilgi alanı, iş veya endüstriyel süreçte kullanılan büyük veri sistemleri için referans mimari önermişlerdir [7]. Önerilen mimaride veri kaynakları, veri

çıkarma, veri yükleme ve ön işleme, veri işleme, veri analizi, veri dönüştürme, görüntüleme ve görselleştirme, veri depolama ve model belirleme gibi bir takım özellik de açıklanmıştır [7].

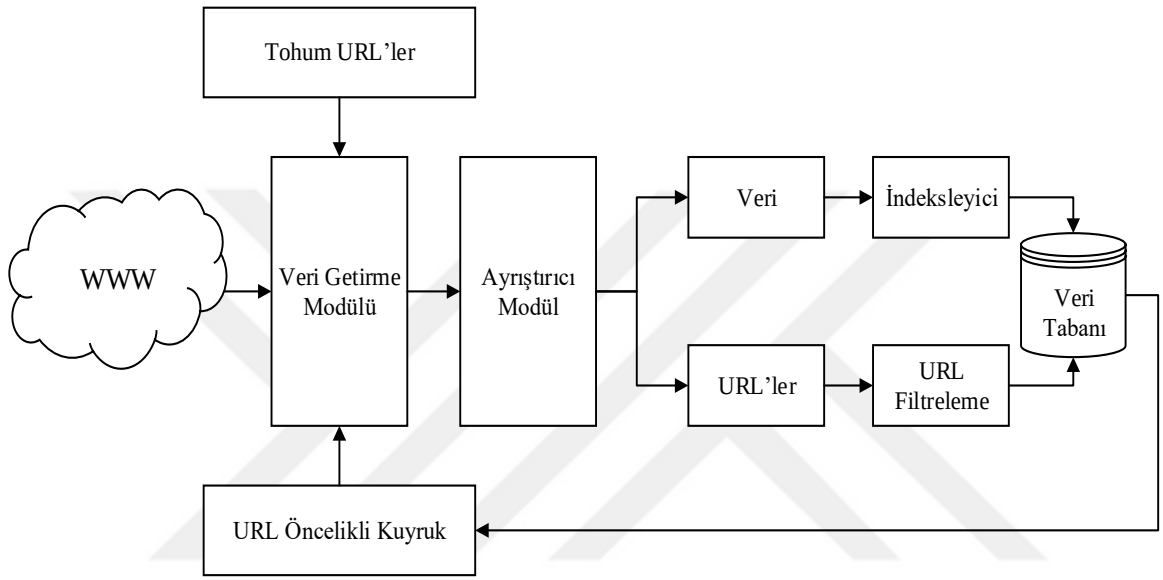
Assunção ve arkadaşları, herhangi bir büyük veri mimarisinde bulunması gereken bazı bileşenleri açıklamış ve mimaride olması gereken en önemli 4 aşamayı belirlemişlerdir [8]. Bu aşamalar veri kaynakları, veri yönetimi, modelleme, sonuç analizi ve görselleştirme olarak ifade edilmiştir. Ayrıca önerilen mimarinin en önemli özelliği, büyük miktarda veri depolama, hesap yapma becerisi ve avantajlarından dolayı büyük veri mimarisine dâhil edilmesini önerdikleri bulut bilişim ile ilişkilendirmesidir [8].

Zhang ve arkadaşları endüstriyel verileri kullanan ve dört ana aşamada çalışan büyük veri analitiği mimarisi önermişlerdir [9]. Bu aşamalardan birincisi tasarım ve geliştirme gibi ürün yaşam döngüsü hizmeti sunmaktadır. İkinci aşama algılayıcılar gibi farklı endüstriyel kaynaklardan büyük veriyi almakta ve sisteme entegre etmektedir. Üçüncü aşama, büyük veriyi yapısına göre işlemekte ve saklamaktadır. Son olarak dördüncü aşamada büyük veri madenciliği ve bilgi keşfi gerçekleştirilmektedir [9].

Blazquez ve Domenech sosyal ve ekonomik davranışları, eğilimleri ve değişiklikleri tahmin etmek amacıyla tasarlanmış büyük veri mimarisi geliştirmişlerdir [10]. Bu mimari geleneksel olmayan bilgi kaynaklarının ve veri analiz yöntemlerinin çoğunu uygun şekilde sisteme entegre etmektedir [10].

Büyük verinin en önemli veri kaynaklarından birisi de Web veri kaynaklarıdır. Günümüzde arama motorları yüz milyarlarca Web sayfasını dizinlemiştir[11]. Web, yapılandırılmış, yarı yapılandırılmış ve yapılandırılmamış veriler içeren ve arama motorları aracılığı ile kullanıcılara sunulan birincil ve en büyük veri kaynağıdır [12, 13]. Arama motorlarının işleyişini kolaylaştıran en önemli bileşenler arasında Web tarayıcıları gelmektedir [14]. Bu tarayıcılar, Web sayfalarındaki verilerin ayrıştırılmasında, dizine eklenmesinde ve depolanmasında kritik rol oynar. Bir Web tarayıcısının birincil işlevi, tohum tek düzen kaynak belirleyici (Uniform Resource Locator - URL) kümesi olarak bilinen önceden belirlenmiş bir başlangıç URL listesinden başlayarak Web sayfalarını tarayıp indirerek ve ayrıştırarak sistematik olarak Web üzerinde gezinmektir. Tarayıcı bir Web sayfasını ziyaret ettiğinde, yalnızca içeriğini indirmekle kalmaz, aynı zamanda ilgili bilgileri çıkarmak için

gerekli ayrıştırma işlemlerini de gerçekleştirir. Çıkarılan bu veriler daha sonra indekslenir ve etkin erişim için arama motorunun veri tabanında düzenlenir. Ayrıca, ayrıştırma işlemi sırasında Web gezgini, ayrıştırılan sayfada yer alan yeni URL'leri keşfeder. Bu yeni keşfedilen URL'ler, gelecekte taranacakları sırayı belirleyen öncelik sırasına sahip bir kuyruğa eklenir. Tarayıcı, öncelik kuyruğundaki tüm URL'ler bitene veya işlenene kadar bu işlemi tekrar tekrar gerçekleştirir. Şekil 1.1'de genel Web tarayıcı mimarisi sunulmuştur.



Şekil 1.1. Genel Web tarayıcı mimarisi

Bir genel Web tarayıcısı çok sayıda bileşene sahiptir.

Tohum URL'ler:

Web tarayıcısının ilk taramaya başladığı URL'lerdir. Tohum URL'ler, tipik olarak, manuel olarak seçilen veya program kullanılarak belirlenen URL'lerdir. Tohum URL'lerin kalitesi, tarama işleminin başlangıç, kapsam ve yönünü belirlediğinden dolayı taramanın başarısının artmasını belirlemektedir.

Veri Getirme Modülü:

Tohum URL'lerden ve ardından öncelikli URL'lerden bilgi çıkarmak için kullanılır. URL'lerin tutulduğu sunuculara hiper metin aktarım protokolü (HyperText Transfer Protocol - HTTP) isteği gönderilir ve gelen veriyi ayrıştırıcı modüle gönderir.

Ayrıştırıcı Modül:

Veri getirme modülünden gelen Web sayfasının içeriklerini veriler ve URL'ler şeklinde ayrıştırır. Bu veriler metin olabileceği gibi resim, video, belge vb. heterojen veriler de olabilir.

İndeksleyici:

Ayrıştırıcı modül tarafından ayrıştırılan veriler veri tabanında saklanması için indekslenerek ilgili tablolara eklenir.

URL Filtreleme:

Ayrıştırıcı modül tarafından ayrıştırılan URL'ler filtrelenerek ve önceliği belirlenerek veri tabanlarında ilgili tablolara eklenir.

Veri Tabanı:

Web sayfasındaki veriler ve URL'ler veri tabanında indekslenerek saklanır. URL'ler de veri tabanlarında benzersiz, alan adına bağlı ve öncelikli değeri ile birlikte saklanır.

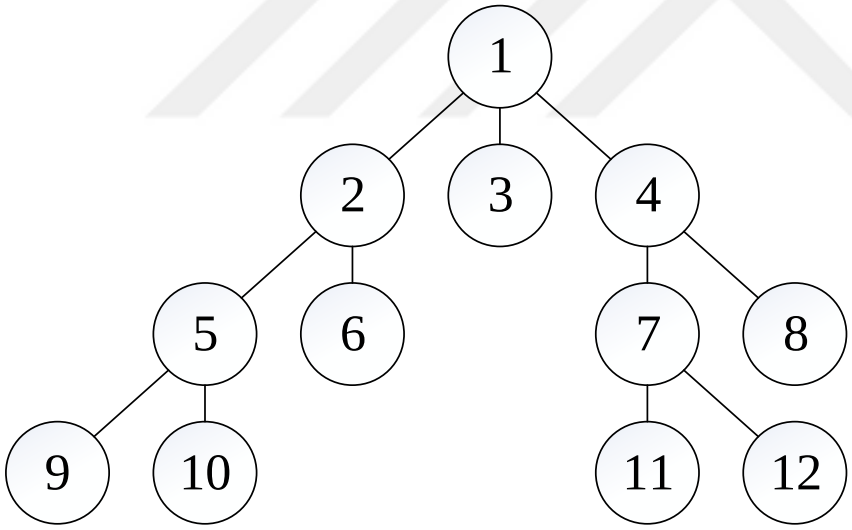
Öncelikli URL Kuyruğu:

Web tarayıcıları tarafından taranması gereken URL'leri yönetmek ve önceliklendirmek için kullanılan veri yapısıdır. Tohum URL'ler tarandıktan sonra elde edilen Web sayfalarından çıkarılan yeni URL'ler veri tabanlarında saklanır. Bu URL'ler hem kapsamı hızlı bir şekilde genişletmek hem de kaliteli sayfalara en kısa sürede ulaşmak için önceliklendirilir. Sıradaki her URL'e ilgi düzeyi, tazeliği, önemi veya tarayıcı algoritması tarafından tanımlanan diğer faktörlere dayalı olarak öncelik değeri atanır.

Web tarayıcısı kullanılarak Web sayfası üzerindeki bilgilerin toplanması ve alınması işlemine Web scraping denir [15]. Web scraping ile bir Web sayfasına HTTP üzerinden erişilir. Daha sonra belirli veriler toplanır ve Web sayfasından yerel bir depolama alanına veya bir tabloya kopyalanır. Son olarak da bu verileri üzerinde gerekli manipülasyon ve analiz işlemleri gerçekleştirilir [16].

Web scraping insani çaba gerektiren manuel sistemlerden otomatik sistemlere kadar çeşitli çalışma alanına sahiptir. Genel Web tarayıcılarının en önemli özelliklerinden biri kapsamlarının tüm Web olmasıdır. Bu durum veri toplama, indeksleme ve güncelleme

işlemlerinde birtakım zorlukları da beraberinde getirmektedir. Öncelikle tekrarlanan URL'lerin tespit edilmesi, sonsuz döngüden kaçınılması ve yeni bulunan URL'lere öncelik verilmesi gerekmektedir. Bunlara ek olarak literatürde toz (dust) olarak adlandırılan [17, 18], aynı metne sahip farklı URL'lerin de bulunup engellenerek verimliliği arttırmak gerekmektedir. Bu tür yinelenen verileri taramak, depolamak ve erişmek hem kaynakların verimsiz kullanılmasına hem de düşük kaliteli sıralamaya neden olmaktadır [18]. Web veri kaynaklarından veri toplamak amacıyla çeşitli algoritmalar geliştirilmiştir. Bu algoritmalardan genel Web tarayıcılarında en çok kullanılanı geniş öncelikli arama (Breadth First Search - BFS) algoritmasıdır. BFS ağaç ya da grafik yapılarında arama yapmak için düğümler arası geçiş yapan bir algoritmadır. Ağaç kökünden veya daha önce tanımlanan düğümden başlayarak bir alt düğümü yatay olarak tarar, hedefi bulursa durur hedef bulunamazsa bir alt seviyeye inip yatay olarak taramaya devam eder [19]. Şekil 1.2'de görüldüğü gibi 1 ile belirtilen başlangıç düğümünden itibaren bir alt düğüm (2,3,4) yatay olarak tarandıktan sonra bir alt seviyeye inerek taramaya devam eder.

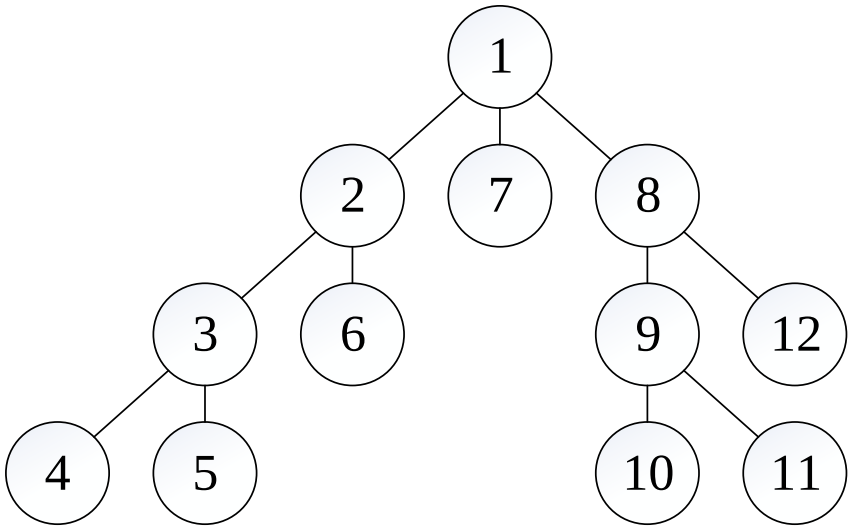


Şekil 1.2. BFS algoritması

BFS algoritmasında hedef, başlangıç düğümüne ne kadar yakın ise başarı o kadar yüksektir. Ancak, büyük Web sayfalarında hedef derinlere indikçe, BFS algoritmasının başarısı düşmektedir. Bu durum, BFS algoritmasını bir tür kör arama yöntemi olarak değerlendirilebileceği bağlamını ortaya koymaktadır. Bir diğer önemli arama algoritması derin öncelikli arama (Depth First Search - DFS) algoritmasıdır. DFS algoritması ağaç yapıları veya grafikler üzerinde gezinme ve arama amacıyla kullanılan bir algoritmadır.

Arama kökten başlar (grafik durumunda kök olarak bir düğüm seçer) ve geriye doğru izlemeye başlamadan önce her dal boyunca mümkün olduğunca derine arama yapar. Genel olarak, DFS, mevcut arama ağacının ilk alt düğümünü genişleterek ve bir hedef düğüm bulunana kadar veya alt düğüme sahip olmayan bir düğüme ulaşana kadar daha da derinlere inen kör bir arama stratejisidir. Ardından, arama, henüz keşfedilmemiş en son düğüme geri döner. Yinelemeli olmayan bir uygulamada, tüm yeni genişletilmiş düğümler keşif için bir yığına eklenir [20].

DFS algoritması, küçük derinliklere odaklanan aramalarda tercih edilebilir. Ancak, Web'in karmaşıklığı ve derinliği göz önüne alındığında, DFS'nin genel taramalarda sürekli olarak derinlere inmeye devam etmesi, bazı sorunlara yol açabilir. Web sayfalarının karmaşık ve derin bir yapıya sahip olması, DFS'nin genel Web tarayıcıları için uygun bir seçenek olmamasına neden olabilir. DFS, bir dalı mümkün olduğunca derine kadar keşfetme eğiliminde olduğu için, genel Web taramalarında tüm yapıyı tam olarak keşfetmeden önce hedefe ulaşabilir. Bu durum, Web'in geniş ve karmaşık yapısı içinde kaybolma veya daha yüzeysel bilgileri göz ardı etme riskini artırabilir. Şekil 1.3'de görüldüğü gibi 1 ile belirtilen başlangıç düğümünden itibaren alt düğümler (2,3,4) dikey olarak tarandıktan sonra keşfedilmemiş en son düğüm olan 5 ve sonrasında 6 ve 7. Düğümler taranarak taramaya devam eder.



Şekil 1.3. DFS algoritması

Her iki algoritmanın avantajları ve dezavantajları, uygulanacak özel bir problem bağlamına ve hedeflere bağlı olarak değerlendirilmelidir. BFS'nin yakınlık odaklı başarı eğilimine karşın DFS'nin derinlikte odaklanma yeteneği, kullanılacak algoritmanın seçimini belirleyen önemli faktörler arasında yer almaktadır.

Üçüncü temel arama algoritması, Google tarafından da kullanılan PageRank algoritmasıdır. Bu algoritma, Web sayfalarının önemini değerlendirmek amacıyla tasarlanmıştır. PageRank algoritması Web sayfasındaki domain içi bağlantıları (inlink) veya domain dışı bağlantıları (backlink) hesaplar. Daha sonra her Web sayfasına hesaplanan değere göre bir sayfa sıralaması verilir. Bu sıralama belirlendikten sonra, bir Web sitesinin sayfa sıralamasına dayanarak bir indeks oluşturulur. Oluşturulan indeks, aranan anahtar kelimeleri içeren bir Web sitesinin ne kadar ilgili olduğunu göstermek için kullanılmaktadır [21].

Bu temel arama algoritmalarının yanı sıra, literatürde çeşitli arama algoritmaları ve melez yaklaşımlar da önerilmiştir. Örneğin, Baker ve Akcayol'un gerçekleştirdiği çalışmada, bir öncelik kuyruğu kullanarak bir Web tarayıcı algoritması geliştirilmiştir. Geliştirilen algoritma, Web sayfası içindeki URL'leri InterDomain ve IntraDomain olarak iki kategoriye ayırmaktadır. InterDomain URL'leri için 2/3, IntraDomain URL'leri için ise 1/3 değerleri atanarak, bulunan URL'lerle öncelikli bir kuyruk yapısı oluşturulmaktadır. Bu sayede InterDomain URL'ler yüksek öncelikli değere sahip olup IntraDomain URL'lere göre öncelikli taranmaktadır. Bu tercih, özellikle domain içindeki bağlantı döngülerini önlemek amacını taşımaktadır. Ayrıca, önerilen algoritma, öncelikli kuyruk içindeki URL'leri tararken uzun bekleme durumlarını önlemek ve öncelikli kuyruk hafızasını yeni gelen yüksek öncelikli URL'lerde kullanmak için bir zamanlama mekanizması kullanmaktadır. Bu mekanizma, kuyruk içindeki maksimum bekleme süresini aşan URL'leri kuyruktan çıkarmaktadır. Yapılan deneysel sonuçlara göre, önerilen algoritma etkili bir tarama performansı sergilemektedir [22].

Thangaraj ve Sivagaminathan'ın yaptıkları çalışma [23], geniş çaplı bir Web araması için BFS algoritmasını geliştirmeye odaklanmıştır. Geliştirilen algoritma, BFS'de sıkça kullanılan sabit seviyedeki derinlik yerine olasılık tabanlı dinamik rastgele derinlik kavramını önermiştir. Bu öneri, tarayıcının daha fazla arama ve veri toplama işlemleri için sınırsız genişlikte ve rastgele derinlikte gezinmesine olanak tanımaktadır. Mevcut tarayıcıyı iyileştirmek amacıyla, BFS algoritması üzerine eklenen dinamik derinlik özelliği için bir

Poisson olasılık dağılımı veya optimize edilmiş rastgele bir sayı üretmek üzere olasılık kütle fonksiyonu (Probability Mass Function - PMF) kullanılmıştır. PMF, ayrı bir rastgele değişkenin bir değere tam olarak eşit olma olasılığını veren bir fonksiyondur. Web tarama sürecinin sonsuz değerlere yol açabileceği ve hiç bitmeyebileceği göz önüne alındığında, PMF, 0 ile 35 arasında değişen rastgele derinlik değerleri üretmektedir. Bu sayede, her ziyaret edilen sayfadan indirilebilir sayfaların maksimum düzeye çıkarılması amaçlanmaktadır. Yapılan deneysel sonuçlar, algoritmanın 0 ila 35 arasında değişen bir derinlik aralığı seçtiğini ve mevcut BFS algoritmasına kıyasla daha başarılı olduğunu kanıtlamaktadır [23]. Bu geliştirme, Web tarama sürecini daha etkili ve verimli hale getirerek geniş çaplı aramalarda daha başarılı sonuçlar elde edilmesine olanak tanımaktadır.

Lee ve arkadaşları çalışmalarında [24], milyarlarca Web sayfasını tek bir sunucu üzerinden indirebilen ve performansı modelleyebilen IRLbot adını verdikleri bir Web tarayıcısı önermişlerdir. IRLbot, her etki alanına algılanan itibarına dayalı olarak belirli bir değer vermektedir. Her zaman aralığında yalnızca belirli bir değeri karşılayan URL'leri taramaktadır. Yenilikçi bir özellik olarak, önerilen yöntem bekleyen kuyrukları sıralamak yerine, URL'leri karıştıran doğrusal taramalar kullanarak değerlerini bir dizi öncelik sırasına yerleştirir ve başarısız olanları ayrı bir biriktirme dosyasına aktarır. Önerilen yöntemin temel avantajlarından biri, tarama boyutunun arttıkça sağladığı iyi ölçeklenebilirliktir, bu da performansın birikmiş bağlantı sayısından etkilenmediği anlamına gelmektedir. Bunlara ek olarak yazarlar güncelleme yönetimi ile disk deposu (Disk Repository with Update Management - DRUM) olarak adlandırdıkları genel bir çerçeve sunan URL kontrol algoritmasını tanıtmışlardır. DRUM'a göre anahtar, bazı verilerin benzersiz bir tanımlayıcısıdır ve değer ise anahtara eklenen isteğe bağlı bir bilgidir. Bu anahtar değer çiftlerinde desteklenen “Kontrol et”, “Güncelle” ve “Güncellemeyi kontrol et” olmak üzere üç işlem bulunmaktadır. İlk durumda, gelen veri kümesi, disk önbelleğinde saklanan ve kopya veya benzersiz olarak sınıflandırılanlara karşı kontrol edilmesi gereken anahtarları içermektedir. Yinelenen anahtarlar için, her bir anahtarla ilişkili değer, isteğe bağlı olarak diskten alınabilir ve bazı işlemler için kullanılabilir. İkinci durumda, gelen liste, mevcut disk önbelleğiyle birleştirilmesi gereken <anahtar, değer> çiftlerini içermektedir. Belirli bir anahtar varsa değeri güncellenmektedir, yoksa disk dosyasında yeni bir giriş oluşturulmaktadır. Üçüncü durumda ise disk önbelleğinden tek geçişte hem kontrol hem de güncelleme gerçekleştirilir. Yapılan deneysel testler, önerilen yöntemin, daha önce literatürde öne çıkan yöntemlere kıyasla önemli ölçüde daha iyi ölçeklenebilirlik sağladığını

göstermiştir. Bu geliştirilmiş algoritma, İnternet üzerinde veri toplama süreçlerini daha etkili ve verimli hale getirerek başarılı sonuçlar elde etmiştir [24]. Web'in yapısı ve veri türleri değiştikçe verilerin toplanması için çeşitli arama algoritmaları geliştirilmektedir.

Büyük verilerin hızla çeşitlenip artmasının ve geleneksel indeksleme yöntemlerinin yetersiz kalmasının bir sonucu olarak, verilerin etkili bir şekilde indekslenmesi büyük bir öneme sahiptir. Bu indeksleme sürecinde, verilerin saklanması, yapılandırılması, sorgu yanıt süresi ve sorgu doğruluğu gibi kritik işlemlerin başarılı bir şekilde gerçekleştirilmesi gerekmektedir. Özellikle Web verileri gibi büyük verilerin indekslenmesinde temel amaç, verileri belirli sorgu kıstaslarına göre ayrıştırmak ve her bir ayrıştırılmış verinin, kullanıcı sorgularını karşılayan değerleri içermesini sağlamaktır [25]. Verilerin etkili bir şekilde indekslenmesi, depolama işlemlerini daha düzenli ve bilgi alımını daha kolay hale getirir.

Fasolin ve arkadaşları, veri indeksleme yaklaşımlarını yapay zekâ ve yapay zekâ olmayan yaklaşımlar olarak iki ana gruba ayırmışlardır [25]. Yapay zekâ indeksleme yaklaşımları, büyük verideki bilinmeyen davranışları tespit etmek amacıyla kullanılmaktadır. Bu yaklaşımlar, modelleri gözlemleyerek benzer özelliklere sahip nesneler arasında ilişki kurmaktadır. Yapay zekâ indeksleme yaklaşımları, birçok avantaja sahip olmasına rağmen verilerin eğitimi ve tutarsız davranışlar gibi zorluklar nedeniyle performans bakımından yapay zekâ olmayan yaklaşımlara göre daha verimsiz olarak değerlendirilmektedir [26]. Yapay zekâ olmayan yaklaşımlarda, veri öğelerinin anlamı ve aralarındaki ilişkinin indekslerin oluşumunda etkisi yoktur. Bunların yerine, en çok sorgulanan veya aranan öğelere göre indeksleme yapılmaktadır. Yapay zekâ olmayan indekslemede en çok tercih edilen yaklaşımlar tersine çevrilmiş indeksleme, ağaç tabanlı indeksleme, karma indeksleme ve özel indeksleme yaklaşımları olarak sıralanabilir. Söz konusu veri metinlerden oluştuğunda tersine çevrilmiş indeks yapısı en başarılı sonucu vermektedir. Tersine çevrilmiş indeks yapısı, özellikle metin verilerinden oluşan büyük veri setlerinin saklanması ve indekslenmesi için yaygın olarak kullanılan bir yöntemdir. Bu indeksleme yaklaşımı, her bir belgenin bir kelime listesi olarak saklandığı ileri dizin ve kelimelerin bulunduğu belgelerin listesinin saklandığı tersine çevrilmiş dizin olmak üzere iki temel türde dizini içerir. Tersine çevrilmiş indeksleme, arama motorları tarafından geniş bir şekilde benimsenmiş ve kullanılan bir standart haline gelmiştir, çünkü basit yapısı, kolay yönetilebilirliği ve düşük bellek tüketimi gibi avantajlara sahiptir [27]. En basit haliyle bir

dizi belgeyi ileri ve tersine çevrilmiş dizinde saklamak Çizelge 1.1 ve Çizelge 1.2'deki gibidir.

Çizelge 1.1. İleri dizin

Doküman Numarası	Kelime
1	Hızlı erişim
2	Hızlı sonuç
3	Yavaş erişim
4	Yavaş sonuç

Çizelge 1.2. Tersine çevrilmiş dizin

Kelime	Doküman Numaraları
Hızlı	1 - 2
Yavaş	3 - 4
Erişim	1 - 3
Sonuç	2 - 4

Büyük veri işlemlerinde tersine çevrilmiş dizin dosyaları oluşturmak için birçok yöntem önerilmiştir. Veri koleksiyonundaki herhangi bir belgede görülen her bir terim (t) bir sözlükte kaydedilir. Daha sonra t içeren Ft belgelerinin her biri için bir kayıt içeren kayıt listesi ile ilişkilendirilir [28]. Her bir kayıt şunlardan oluşur;

- Belge Tanımlayıcı d(t, i): t terimini içeren Ft belgelerinin i'ninci sıra tanımlayıcısı
- Belge içi frekans verisi f(t, i): t'nin d(t, i) de kaç kere tekrarlandığını kaydeden belge içi frekans verisi
- d(t, i) belgesinde t'nin göründüğü kelime veya bayt konumlarını kaydeden f(t, i) tamsayılarının bir listesi

Web tarayıcılarının yeni URL'leri keşfedip tarayarak dizine eklemesinin yanı sıra değişen Web sayfalarını tekrar taraması ve dizini güncel tutması gerekmektedir [29]. Web sayfalarının güncel olup olmadığının tespiti ve arama motorlarının veri depolarının tazelenmesi için birçok sayfa değişiklik tespit tekniği geliştirilmiştir. Web sayfalarının kendilerine özgü bir dinamizmi olduğu ve her birinin güncellenme süresinin farklı olduğu göz önüne alındığında, tarayıcının hangi sayfaların ne sıklıkta ziyaret edileceğine etkin bir yöntemle karar vermesi gerekmektedir. Tarayıcının, sık güncellenen sayfaları sıklıkla

ziyaret ederken nadiren değişen ya da statik olan sayfaları daha az sıklıkta ziyaret etmesi gerekmektedir.

Web tarayıcılarının Web sayfalarını tekrar ziyaret etme sürelerinin belirlenmesinde çeşitli yöntemler kullanılmıştır. Bu yöntemlerden biri veri madenciliğidir. Tazelik açısından veri madenciliğini kullanmak daha iyi sonuçlar vermesine rağmen verilerin eğitilmesi ve ilkelerin belirlenmesi gibi süreçlerden dolayı performans önemli ölçüde düşmektedir. Bulloat ve arkadaşları [30], çalışmalarında, Web tarayıcısının Web sayfalarını yeniden ziyaret etme sürelerini ve sıklığını veri madenciliği yöntemlerini kullanarak araştırmışlardır. Web sayfalarının tıklama sıklığı, URL'in konumu, sonraki cezalandırma ve sayfanın kategorisi gibi verileri kullanarak tarama sıklığını belirlemişlerdir. Önerilen yöntemde tarayıcının sayfa güncelleme sıklığını kullanıcılar belirlemektedir. Kullanıcı tercihlerine bağlı olarak popüler sayfalar sık güncellenirken popüler olmayan bağlantılar daha az sıklıkta güncellenmektedir. Kharazmi ve arkadaşları IFCrawler adını verdikleri, veri madenciliğini kullanan ve çok iş parçacıklı Web tarayıcısını önermişlerdir. IFCrawler tarama işlemi sırasında bilgi toplayan WebInfoCollector ve toplanan verilerden ilkeleri çıkarmak için kullanılan PolicyRepository adını verdikleri iki bileşenden oluşmaktadır. IFCrawler genel amaçlı bir Web tarayıcısı ile bir milyon Web sayfası kullanılarak karşılaştırılmıştır. Önerilen tarayıcının tazelik performansının önemli derecede iyi olduğu belirtilmiştir [31].

Web tarayıcılarının sayfaları tekrar ziyaret etmelerinin temel sebebi veri depolarının güncelliğini korumaktır. Han ve arkadaşları [32], arama motorlarında bulunan veri depolarının güncelliğini korumada kullanılan stratejileri gözden geçirmiş ve sezgisel tazelik ölçütünü analiz etmişlerdir. Önerilen yöntemde Web sayfasının değişim oranının yanı sıra kullanıcıların bakış açılarını da hesaba katan ağırlıklı tazelik ölçütü önerilmiştir. Çalışmada ağırlıklı tazelik ölçütünün hesaplanmasında Web sayfasının öneminin nasıl belirleneceği ele alınmıştır. Yazarlar en iyi sezgisel tazeliği elde etmek için kaynak sayfaların değişim oranının Web sayfasının öneminden daha büyük olması gerektiği sonucuna varmışlardır.

Veri depoları güncellenirken dikkat edilmesi gereken en önemli konu bant genişliğinin verimli bir şekilde kullanılmasıdır. Bu konuda çalışma yapan Amudhan ve Thirupathi [33], arama motorunun kullandığı veri deposunu güncellerken Web tarayıcısının gereksiz isteklerini ortadan kaldırmak için trafiğe uyarlanabilir en uygun güncelleme şemasını (Traffic Adaptive Optimum updating Scheme -TAOS) önermişlerdir. Çalışmadaki diğer bir

yenilik ise güncellenen belgelerin sadece güncellenen kısımlarının arama motoru veri deposuna yüklenmesidir. Dolayısı ile Web sunucuları üzerindeki yükü azaltan ve ağ bant genişliğini minimum düzeyde kullanan otonom bilgi işlem mimarisi tasarlanmıştır. Zhu ve arkadaşları [34], Web gibi karmaşık bir ortamda bant genişliğini dikkate alan yeni bir bant genişliği tahsis yaklaşımı önermişlerdir. Önerilen yaklaşım yeni Web sitelerinin sayısının sistemin eşzamanlı tarayabileceği Web tarayıcıları sayısını aşması ve aşmaması olmak üzere iki senaryo için tasarlanmıştır. Web tarayıcılarının sırasını belirlemek ve bant genişliği tahsisini programlamak için döngü tabanlı yeniden tahsis yaklaşımı önerilmiştir. Yapılan kapsamlı simülasyonlar sonucunda önerilen sistemin tüm senaryolarda arzu edilen performansa ulaştığı ve zaman kısıtlamalarını iyi bir şekilde karşıladığı görülmüştür. Souza ve arkadaşları [35] çalışmalarında, Web sayfalarını yeniden ziyaret etmek ve yerel veri deposunu güncel tutmak için etkili bir zamanlama politikası önermişlerdir. Önerilen yaklaşım üç aşamadan oluşmaktadır. İlk aşamada lineer olmayan tamsayı programının çözünürlüğü ile belirlenen sayfa başına güncelleme sayısı belirlenmektedir. İkinci aşamada her sayfada nezaket politikası çerçevesinde güncelleme zamanlarının bir listesini belirlemek için zaman algoritması kullanılmıştır. Son aşamada ise basit ve verimli bir ağ akış şeması kullanılarak tarayıcılara güncellemeler atanmaktadır. Yapılan deneylerde elde edilen bulgulara göre önerilen yöntemin veri deposunu önemli derecede güncel tutabildiğini göstermektedir. Kausar ve arkadaşları [36], Web sayfalarının depolanan önceki sürümlerinin değiştirilip değiştirilmediğini belirlemek için Hash değerini kullanan değişiklik algılama sistemi önermişlerdir. Çalışmanın en önemli katkısı eski ile yeni sayfa arasındaki değişiklikleri tespit etmede kullanılan zamandan tasarruf sağlamasıdır.

Bir Web sayfasının en iyi güncelleme sıklığını yakalamak için bu sayfaların güncellenme geçmişlerinin incelenmesine, Web sayfalarının yapısına ya da etki alanlarına dayanan, kategori ya da kümeleme yöntemlerini kullanan çalışmalarda yapılmıştır. Tan ve Mitra [37], Web sayfalarının değişiklik sıklıkları ile ilgili özellikleri tanımlamış ve Web sayfalarının güncelleme geçmişlerini bu özellikler ile ilişkilendirerek kümelemeye dayanan bir tarama algoritması önermişlerdir. Algoritma her kümeden bir Web sayfasını indirir ve bu Web sayfasının son tarama döngüsünde güncellenme oranına bağlı olarak kümede bulunan diğer Web sayfalarını da tarayıp taramayacağına karar vermektedir. Radinsky ve Bennett [38], çalışmalarında Web sayfalarının güncellenme geçmişlerine ek olarak sayfaların içerikleri, sayfalardaki değişiklik dereceleri, Web sayfasının diğer sayfalar ile ilişkisi ve değişiklik türlerini inceleyerek analiz etmişlerdir. Ayrıca ilgili Web sayfalarını belirleyen ve içerik

benzerliği ölçümünde kullanılan çok sayıda benzerlik ölçütü sunmuşlardır. Li ve arkadaşları [39], taranacak Web sayfalarını ana sayfa, geçiş sayfaları ve içerik sayfaları olmak üzere 3 kategoriye ayırmışlardır. Ana sayfaların her zaman öncelikli olduğu belirtilmiştir. Geçiş sayfaları ve içerik sayfaları arasındaki önceliği belirlemek için ise ağırlık katsayısını hesaplamış ve buna göre önceliklendirmişlerdir. Mor ve arkadaşları [40], XML dosyalarını kullanarak (SiteMap.XML) etki alanlarına göre site haritası oluşturarak bu haritaya göre yeni eklenen sayfaları indirmeye dayanan bir tarayıcı geliştirmişlerdir. Geliştirilen tarayıcı, SiteMap.XML dosyasına yeni bir URL eklenmemişse eski dosyaları tekrar taramaktadır. Tarayıcı Web sayfalarını tüm alt etiketleri, meta verileri ve şekil açıklamaları ile birlikte ayrıştırarak sayfaların kaç kez ziyaret edildiğini saymakta ve bu verilere göre sıralamaktadır. Bunun yanı sıra sayfalardaki değişiklikleri sayan, sayısını tutan ve değişiklik belirli bir eşiğin üstündeyse sayfayı indiren eşzamanlı değişim hesaplayıcı bir sistem kullanılmıştır.

Web tarayıcılarının güncellemeleri zamanında yakalamak için sayfaları tekrar ziyaret etmelerindeki bir diğer sorun da özellikle bant genişliği olmak üzere ağ kaynağının tüketilmesidir. Bu sorunu aşmanın bir yolu da Web sayfalarının uzak sunuculardan indirilip analiz edilmesi yerine, verinin bulunduğu sunucularda mobil aracılar kullanarak analiz işlemlerinin yapılmasıdır. Kausar ve arkadaşları [41], Java agletleri kullanarak sunucu bilgisayar üzerinde analiz işlemlerini gerçekleştiren mobil tarayıcıyı önermişlerdir. Önerilen tarayıcı veri kaynağında verileri analiz etmekte, indekslemekte ve sıkıştırmaktadır. Böylelikle verimlilik ve performans artmakta olup ağ yükü ve veri trafiği önemli ölçüde azalmaktadır. Bedevi ve arkadaşları [42], arama motorunun veri deposunu güncel tutmak için belge dizinine dayalı sayfa değişikliği algılama tekniği ve mobil aracılar kullanarak dağıtılmış dizin oluşturma yöntemini önermişlerdir. Önerilen yöntemde verilerin analizi, indekslenmesi ve sıkıştırılması sunucular üzerinde yapılmakta ve ağ yükü en aza indirilmektedir. Gupta ve arkadaşları [43], Web sayfalarının güncelliğini, bant genişliğini daha verimli kullanmak adına sunucu tarafında gerçekleştiren yeni bir sayfa değişikliği algılama tekniği önermişlerdir. Temelde önerilen teknik eski Web sayfasının etiket ve metin kodunu yeni Web sayfası ile karşılaştırmaya dayanmaktadır. Web sayfalarının değişiklik miktarları belirli bir eşik değerden az ise değişiklikler bir metin dosyasına kaydedilerek tüm Web sayfaları yerine bu metin dosyası gönderilmektedir. Tersi durumda değişiklik miktarları eşik değerin üzerinde ise yeni sayfa gönderilerek eski sayfa ile değiştirilir. Ayrıca çalışmada, eşik değerlerin Web sayfalarının değişim sıklığına göre değişebileceği belirtilmiştir.

Web tarayıcılarının Web sayfalarını tekrar ziyaret zamanlarını belirlemede kullanılan bir başka yaklaşım da olasılık tabanlı yaklaşımdır. Sethi [44], Web sayfalarını tekrar ziyaret sürelerinde değişiklik olma olasılığına dayalı, her bir Web sayfaları için farklı, ilk ve yeniden tarama aralığını dikkate alan bir yöntem önermişlerdir. Önerilen yöntem ayrıca talep edilen bilgiyi koruma amacı ile Web içindeki bağlantı yapılarına göre ziyaret edilecek URL'lere öncelik atamaktadır. Çalışmada önerilen tekrar ziyaret zamanlayıcı modülü Markov modeline dayanmaktadır. Performans ölçütü olarak kesinlik ve geri çağırma olmak üzere iki standart ölçüt kullanılmış ve geleneksel Web tarayıcı ile karşılaştırılarak sonuçlar analiz edilmiştir. Yapılan deneysel sonuçlar önerilen yöntemin, güncel verileri alma ile ağ yükü arasında önemli bir denge sağlayabildiğini göstermektedir. Santos ve arkadaşları [45], sayfa değişim sıklığının olasılığını bulmak için bir optimizasyon tekniği olan genetik programlama çerçevesini önermişlerdir. Tarayıcı performansı normalleştirilmiş indirimli kazanç ölçütü kullanılarak diğer olasılık tabanlı temel algoritmalar ile karşılaştırılıp analiz edilmiştir. Önerilen çalışmanın tek odak noktası artımlı tarayıcının performansını iyileştirmektir.

Yapılan çalışmalarda önerilen yöntemlerde kullanılan tazelik performansı, genel performans, kullanılan parametreler, bant genişliği kullanımı ve tarayıcı modeli gibi hayati derecede önemli parametreler temelinde karşılaştırmalı analizi yapılmıştır. Analizi yapılan Web tarama yöntemlerinin karşılaştırılması Çizelge 1.3'de gösterilmiştir.

Çizelge 1.3. Web tarama yöntemlerinin karşılaştırılması

Yöntem	Tazelik	Genel Performans	Kullanılan Parametreler	Bant Genişliği Kullanımı	Tarayıcı Modeli
Veri Madenciliği	Yüksek	Düşük	Yeniden Ziyaret Etme Tıklama Sıklığı URL Konumu Sayfa Kategorisi	Orta	Odaklı
Güncelleme Sıklığı	Orta	Orta	Sezgisel Tazelik Ağırlıklı Tazelik Kullanıcı Bakış Açısı Hash Değeri Hesaplaması	Orta/Düşük	Periyodik Genel
Kümeleme	Orta	Yüksek	Değişiklik Sıklıkları Güncelleme Geçmişi Sayfa İçerikleri Diğer Sayfalar ile İlişki Benzerlik ölçütleri	Orta	Artımlı Mobil
Olasılık Tabanlı	Orta	Orta	İlk ve Yeniden Tarama Aralığı Kesinlik Geri Çağırma	Orta	Odaklı

2. İNTERNET'TEKİ HETEROJEN VERİ KAYNAKLARI

Heterojen veri kaynakları, çeşitli tipteki yapılar, formatlar ve çeşitli platformlar üzerinde bulunan veri kaynaklarını ifade etmektedir. İnternet'e her geçen gün kullanıcılar, akıllı cihazlar, robotlar, algılayıcılar vb. birçok farklı kaynaktan veri aktarılmaktadır. İnternet'teki bu çok kaynaklı heterojen veriler arasında multimedya verileri, Web içerikleri, tabu formattaki veriler vb. bulunmaktadır. Web ortamındaki bu veriler çeşitli yapılarda bulunmaktadır. Bu yapılar, yapılandırılmış, yarı yapılandırılmış ve yapılandırılmamış verilerden oluşmaktadır. Bu heterojen veri kaynakları, farklı paylaşım standartları ve iletişim protokolleri kullanmaktadır. Heterojen veri kaynaklarından gelen verilere erişme, bunları alma ve kullanma ihtiyacı bir takım zorlukları da beraberinde getirmektedir. Bu zorluklardan bazıları, veri kaynakları arasında benzer özelliklerin eşleştirilmesi, bir kullanıcı sorgusu ile ilgili farklı veri kaynaklarından belirli özelliklerin alınması, sistemdeki tüm bileşenlerin istediği şekilde diğer veri kaynaklarından gelen verileri farklı formatlarda alma vb. sayılabilir [46].

Bu bağlamda, çok kaynaklı heterojen veri tabanları, bu tür zorlukları ele almak ve farklı veri kaynakları arasında uyumluluğu sağlamak adına önemli bir rol oynamaktadır. Bu veri tabanları, farklı kaynaklardan gelen verileri entegre etme, standardize etme ve uygun formatta sunma yeteneğine sahiptir. Bu, kullanıcılara daha etkili ve kapsamlı veri erişimi sağlama amacını taşımaktadır.

Yeni Web teknolojilerinin ortaya çıkması, sosyal medya ağlarının yaygınlaşması ve IoT ile üretilen verilerin artmasıyla insanlığın büyük veri çağına girdiği kabul edilmektedir. Büyük veri hacim, hız, çeşitlilik, değer ve doğruluk gibi özelliklere sahip bir veri kümesidir [47]. Büyük verinin ölçeklenebilirliği, karmaşıklığı ve büyüme hızı nedeni ile klasik ilişkisel veri tabanlarında depolanması, işlenmesi ve analiz edilmesi zor bir süreçtir [48]. Web'de bulunan heterojen büyük veriler yapılandırılmış, yarı yapılandırılmış ve yapılandırılmamış olmak üzere üç farklı biçimde bulunmaktadır. Yapılandırılmış veriler önceden tanımlanmış bir veri modeline bağlı, saklanabilen, erişilebilen ve işlenebilen her türlü veri olarak adlandırılmaktadır [49]. İlişkisel veri tabanlarında saklanan veriler yapılandırılmış veriye örnek olarak gösterilebilir. Bunun aksine, yapılandırılmamış veriler bilinmeyen yapı ve forma sahip olan ve önceden tanımlanmış bir veri modeline sahip olmayan verilerdir [50]. Metin dosyası, video, resim vb. veriler ve bunların birleşimini içeren heterojen veri

kaynakları yapılandırılmamış veri olarak gösterilmektedir. Yarı yapılandırılmış veri ise her iki veri türünü de içeren bir yapıya sahiptir. Yarı yapılandırılmış veriler açık bir veri modeline sahip olmayan ve eksik olabilen düzensiz verilerdir. HTML Web sayfaları, Web sayfaları için gerçekten basit dağıtım (Really Simple Syndication - RSS) verileri ve genişletilebilir işaretleme dili (Extensible Markup Language - XML) dosyaları yarı yapılandırılmış veri olarak temsil edilmektedir [49, 51]. Web ortamında bulunan büyük verinin toplanması ve indekslenmesi için temel araç Web tarayıcılarıdır. Web tarayıcıları Web ortamında bulunan metinlerin yanı sıra resim, video, doküman vb. gibi heterojen verileri de toplamaktadırlar. Web tarayıcıları ile toplanan veriler yapılandırılmış formda ilişkisel veri tabanlarında saklanabilmektedir Yarı yapılandırılmış ve yapılandırılmamış verilerin son derece yoğun bilgi işlem ve depolama sorunları nedeni saklanmasında ilişkisel veri tabanları yetersiz kalmaktadır. Bu sorunlarla başa çıkmak için literatürde NoSQL veri tabanları ve bulut tabanlı mimariler kullanılmaktadır [52].

Günümüzde en büyük heterojen veri kaynaklarının başında sosyal medya verileri gelmektedir. Web günlükleri ve sosyal medya platformları tarafından oluşturulan Web verileri, geleneksel Web sayfası içeriklerinden farklılık göstermektedir. Sosyal medya platformları (Facebook, Twitter, Instagram vb.), kullanıcıların paylaştığı metinler, resimler, videolar, etiketler ve bağlantılar gibi çeşitli verileri barındırmaktadır. Bu veriler farklı format ve yapıda olabilirler. Bu platformlarda zamansal olarak sürekli bir veri akışı gerçekleştirilmektedir. Bu akış verisini elde etmek için Web tarayıcısının düşük gecikme süresi, yüksek veri kalitesi, uygun ağ nezaketi ve yüksek ölçeklenebilirlik gereksinimi karşılaması gerekmektedir [53]. Twitter, Flickr, Youtube vb. gibi birçok sosyal medya platformları uygulama programlama arabirimleri (Application Programming Interface - API) aracılığıyla kullanıcılar ve oluşturulan içerikler ile ilgili yapılandırılmış verilere erişim izni sağlamaktadır. Bunun yanı sıra belirli bir kullanıcı topluluğunun Twitter verilerini toplayan ve analiz eden [54], Twitter akışlarında konu algılamayı ele alan [55] ve birbirine bağlı Web ve sosyal medya verilerini entegre bir şekilde toplayan [56] çalışmalarda yapılmıştır.

Web tarayıcılarını güncel teknolojiler ile birleştiren çalışmalarda yapılmıştır. Blok zinciri (Blockchain) Nakamoto tarafından 21. Yüzyılın başlarında önerilmiştir [57]. Blok zinciri, büyük miktarda veriyi merkezi olmayan bir şekilde düzenleyebilen, mutabakat sürecini basitleştiren, veri güvenliğini sağlayan ve bilgi paylaşımını etkin bir şekilde kullanan

dağıtılmış bir defter teknolojisidir. Wang ve arkadaşları, blok zinciri tekniklerini kullanarak Web sunucularının iş yükünü hafifletmek ve belirli kurallara göre veri toplamaya izin vermek için blok zinciri tabanlı bir odaklanmış Web tarayıcısı geliştirmişlerdir [58]. Görüntü işleme konusunda Web ortamınki görüntüleri toplayıp görüntü veri tabanı oluşturan Web tarayıcıları geliştirilmiştir. Kalmukov ve Valova, içerik tabanlı görüntü alma tekniklerini, yöntemlerini ve algoritmalarını test etmek için kullanacakları veri kümesini oluşturan Web tarayıcısının mimarisini sunmuşlardır [59]. Ali ve arkadaşları, Hadoop YARN kullanarak büyük ölçekli görüntü veri kümesi oluşturma amacı ile Web sistematik bir şekilde tarayan bir tarayıcı geliştirmişlerdir. Geliştirilen tarayıcı, küçük resim ve simgeler gibi görüntüleri azaltmak için bazı teknikler kullanmakta ve açık kaynak Web tarayıcısı olan Apache Hadoop ve Apache Nutch'a dayanmaktadır [60].

Web tarayıcıları konusunda önemli bir araştırma alanı da anti-tarayıcı teknolojileridir. Anti-tarayıcılar, kötü niyetli kişilerin veya sistemlerin bazı teknik araçları kullanarak toplu olarak Web sitesi bilgilerini elde etmelerini önleme amacı ile geliştirilmişlerdir. Tarayıcı ne kadar başarılı olursa olsun gelişmiş anti-tarayıcılar tarafından keşfedilebilmektedir. Aynı şekilde anti-tarayıcı ne kadar iyi olursa olsun gelişmiş Web tarayıcıları ile bozulabilmektedir. Anti-tarayıcılar İnternet protokolü (Internet Protocol – IP) kısıtlamaları, kullanıcı erişim kontrolü, oturum erişim kısıtlamaları, insan ve bilgisayar ayrımı amaçlı tam otomatik genel turing testi (Completely Automated Public Turing test to tell Computers and Humans Apart - CAPTCHA) ile doğrulama gibi kısıtlamaları kullanmaktadır. Bunlara ek olarak en başarılı sistem, insanların göremediği ve asla tıklayamayacağı, yalnızca Web tarayıcılarının ziyaret edebileceği Web sayfalarına kasıtlı olarak bağlantı bırakan bal küpü (Honeypot) teknolojisidir [61]. Tarayıcılar ve anti-tarayıcılar zıt amaçlar için çalışır ve birbirlerinin düşmanı gibidir.

Coğrafi veriler de günümüzde sürekli artan heterojen yapıdaki veri türlerini oluşturmaktadır. Haritalama hizmetleri (Google Maps, Yandex Maps vb.) ve küresel konumlama sistemi (Global Positioning System – GPS) cihazları ürettikleri coğrafi konum verilerini farklı formatlarda oluşturmaktadır. Yer işareti, rota bilgisi, coğrafi sınırlar gibi çeşitli bilgiler içermektedir. Zhang ve arkadaşları, coğrafi konum verisi toplamada yeni bir metodoloji keşfetmek için, Web tarama araçlarını kullanarak resmi olmayan coğrafi konum verisi toplamanın teknik yolunu önermiş ve bunun fizibilitesini test etmişlerdir. Çalışma, yüksek, orta ve düşük olmak üzere üç derecede ifade edilen birkaç farklı coğrafi konum verisi

toplama yönteminin ekonomik maliyeti, zaman maliyeti, teknik zorluğu ve fizibilite farklılıklarını incelemekte ve analiz etmektedir. Tarayıcı yazılımlarından biri olan GooSeeker ile veri toplamanın diğer tüm veri toplama yöntemlerinden daha ekonomik, başarılı ve hızlı olduğunu göstermişlerdir [62].

Web ortamına her an milyarlarca algılayıcıdan veriler aktarılmaktadır. IoT, algılayıcılar yardımı ile çevresel faktörler hakkında veriler üretmektedir. Shemshadi ve arkadaşları çalışmalarında, Web üzerinden IoT veri kaynaklarını belirlemiş ve ilk kez büyük ölçekli gerçek dünya IoT verilerini toplamak için bir Web tarayıcısı geliştirmişlerdir. İnternet'e bağlı yaklaşık iki milyon nesnenin verilerini toplamışlardır. Bir IoT arama motorundan gerçek dünya sorgu kümesi kullanarak IoT'deki eğilimleri incelemişlerdir [63]. Liang ve arkadaşları, IoT arama motoru kavramlarına, araştırmalarına ve ilerlemelerine genel bir bakış sağlamak için IoT arama ve arama motoru tekniklerini incelemişlerdir. Öncelikle, temsili bir örnek olarak geleneksel Web arama motorlarının gelişimini kısaca gözden geçirmiş ve IoT arama motorlarının gelişim sürecine odaklanmışlardır. Ayrıca, Web aramasının ana tekniklerini incelemiş ve geleneksel Web ile yeni IoT arama motorları arasındaki farkları tartışarak mevcut IoT arama tekniklerini sistematik olarak incelemişlerdir [64]. Shemshadi ve arkadaşları, İnternet'teki IoT verilerini yakalamak için tasarlanmış bir Web tarayıcısı olan ThingSeek'i önermişlerdir. Temel olarak Web haritalama aracılığıyla yayınlanan algılayıcı verilerine odaklanmışlardır. ThingSeek çerçevesi, hem insan hem de makine kullanıcılarını kapsayan bir tarayıcı ve iki sorgulama arabirimini içermektedir. IoT verilerinin büyük bir bölümünün Web haritalama ara yüzleri kullanılarak paylaşıldığını keşfetmişlerdir [65].

Hava durumu servisleri, çeşitli zaman aralıklarında hava şartları ile ilgili hava durumu verileri sunmaktadır. Bu veriler sıcaklık, nem, yağış miktarı, rüzgâr hızı gibi çeşitli türlerden oluşmaktadır. Uteuov ve arkadaşları, insanların ev cihazlarından gözlemlerini paylaştığı "kitle kaynaklı kişisel hava istasyonları" adı verilen veri kaynağını kullanmışlardır. Bu verilerin barındırıldığı weathermap.netatmo.com Web sitesi üzerinden tarama gerçekleştirmişlerdir. Yaşadıkları şehir için genel netatmo API'sini ve Python tarayıcı uygulamasını kullanarak verileri taramışlardır. Şehir konumunda bir yıl boyunca 5 dakikada bir veri toplayan zaman serisi kullanılarak verileri taramışlardır [66].

Finansal verilerde farklı formatta ve heterojen yapıda olabilmektedir. Finansal veriler hisse senetleri, döviz, emtia ve tahvil vb. işlemlerin yapıldığı borsalardan gelmektedir. Bu veriler sayısal veriler, grafiksel veriler, görseller ve benzeri heterojen verilerden oluşmaktadır. Razzaq ve Yang çalışmalarında, 2008'den 2019'a kadar Çin'in şehir düzeyindeki verilerini kullanarak dijital finansın yeşil büyüme üzerindeki etkisini incelemişlerdir. Kapsayıcı dijital finansı ve yeşil büyümeyi ölçmek için Web tarayıcısı teknolojisinden yararlanılmıştır [67]. Yi ve arkadaşları, finans piyasasının merkezi olan borsa ile ilgili forum sitelerindeki metinlerden yararlanarak duygu analizi yapmışlardır. Metin duygularının kusurları ve yatırım sorunları göz önüne alındığı bu çalışmada, borsayı değerlendirmek için tek boyutlu evrişimli sinir ağları (1 Dimensional Convolutional Neural Network - 1DCNN) ve uzun kısa süreli bellek (Long Short-Term Memory - LSTM) dâhil olmak üzere Web tarayıcısı ve derin öğrenme teknolojilerine dayalı bir çerçeve önermektedir. Bunlar arasında çıkarılan özellikler sadece hisse senedi fiyatını değil aynı zamanda metin bilgisini de içermektedir. Bu analizleri yapabilmek için forum sitelerindeki kullanıcı yorumlarını içeren metinleri kullanmışlardır. İnternet'ten büyük ölçekli metin verilerini almak için Web tarayıcı teknolojisini kullanmış ve ilgili finansal bilgileri analiz ederek duyguları manuel olarak etiketlemişlerdir [68]. Raicu çalışmasında, çeşitli makine öğrenimi yöntem ve tekniklerini kullanarak duygu sınıflandırması için kullanılmak üzere bir finansal bankacılık veri kümesinin oluşturulmasına yönelik bir araştırma çerçevesi sunmaktadır. Veri kümesi, Romanya finansal bankacılık sosyal medyasından Web tarayıcıları ile toplanan 2234 finansal bankacılık yorumunu içermektedir [69].

E-ticaret çağımızın temel unsurlarından biri olarak hayatımızın önemli bir yer kaplamaktadır. E-ticaret verileri, ürünler ile ilgili bilgileri, fiyatları, yorumları, görselleri ve videoları içeren çeşitli formatlarda barındırılmaktadır. Onyenwe ve arkadaşları, çalışmalarında, ürün güncellemelerini belirlemek ve HTML verilerini elde etmek amacıyla bir e-ticaret sitesinde Web tarama ve kazıma yöntemlerini başarıyla uygulamışlardır [70]. HTML verileri, kullanıcılara yönelik ad, fiyat, gönderi tarihi, gönderi saati gibi ürün ayrıntılarını çıkarmak üzere tarayıcı tarafından toplanmış ve bu bilgiler, kullanıcılar için anlamlı bilgiler sunma amacıyla kullanılmıştır. Web tarayıcısı komut dosyaları, Python programlama dili kullanılarak yazılmış ve tarayıcı, HTML etiketleri temel alınarak etkili bir şekilde çalıştırılmıştır [70]. Mai ve arkadaşları ise çalışmalarında, e-ticaret Web sitelerindeki sağlamlık ve ölçeklenebilirlik değerlendirmesine odaklanarak önemli bir katkı sağlamayı amaçlamışlardır [71]. Bu bağlamda, e-ticaret Web sitelerinde açık kaynaklı Web

tarayıcılarının yenilik çerçevesini önermişlerdir. Çoklu test ortamları oluştururken, Web tarayıcılarının sağlamlığını ve ölçeklenebilirliğini değerlendirmek için sağlamlık testi ve ölçeklenebilirlik testi uygulamışlardır. Sağlamlık testi, girişimsizlik testi ve girişim testi olmak üzere iki deney için tasarlanmıştır. Sağlamlık başarısızlık oranı, sağlamlığı ölçmek için kullanılmıştır. Tarayıcılar arasındaki önemli farklılıkları analiz etmek amacıyla ise Friedman testi ve Nemenyi testi gibi istatistiksel yöntemler kullanılmıştır [71].

Haber siteleri, İnternet üzerinde metin, video, ses gibi çeşitli formatlarda güncel haberleri sunan önemli veri kaynakları arasında yer almaktadır. Shi ve arkadaşları, artımlı Python Web tarayıcısı ve Scrapy tarayıcı çerçevesini kullanarak ana akım Web sitelerinden haber sayfalarını gerçek zamanlı olarak tarayıp bu verileri veri tabanında depolamışlardır. Bloom filtresini kullanarak Web bağlantılarındaki tekrarı önleyen artımlı Web tarayıcısı, haber sayfalarını etkili bir şekilde takip edebildiğini göstermektedir [72]. Lu ve arkadaşları, düzenli ifadeler ve Xpath, Web sayfası analizi ve Web Magic tarayıcı çerçevesi gibi Web tarayıcı teknolojilerini kullanarak Java tabanlı yapılandırılabilir bir haber toplama sistemini gerçekleştirmişlerdir. Önerilen sistem, veri yakalama, bilgi çıkarma ve haberlerin depolanması gibi işlevleri başarıyla yerine getirebilmektedir. Sistem yüksek düzeyde yapılandırılabilir olup çok kaynaklı haber verilerini tarayabilmektedir [73]. Mondoza ve arkadaşları, haber makalelerini toplayan bir tarayıcı ve bu makalelerin ait olduğu bölümü tanımlayan bir sınıflandırıcı sunmuşlardır. Çeşitli bilgi kaynaklarının bulunması sebebiyle, belirli ilgi alanlarına odaklanarak haber makalelerini toplamak ve filtrelemek amacıyla bir araç tasarlamışlardır. Bu çalışmada, farklı Web sitelerinden otomatik olarak haber makaleleri toplamak için bir tarayıcı kullanılmış, her haber makalesinin bölümünü belirleyen bir sınıflandırıcı entegre edilmiş ve nihayetinde ilgili bölümle eşleşen haber makalelerinin görüntülendiği bir Web uygulaması önerilmiştir [74].

Mobil uygulama verileri, kullanıcı tarafından oluşturulan içerik, kullanım istatistikleri ve kullanıcı etkileşimleri gibi çeşitli veri türlerinden oluşmaktadır. Bilimsel veriler ve akademik makalelerde ise metin, grafik, tablo ve diğer formatlarda ve bilimsel veri tabanlarında bulunmaktadır. İnternet'teki heterojen veri kaynakları, bu farklı türdeki verilerin toplanması, işlenmesi ve analizini gerektiren karmaşık süreçleri içermektedir. Bu süreçler, veri entegrasyonu, veri dönüşümü, anlamlı bilgi çıkarma ve veri madenciliği gibi adımları içermektedir. Heterojen veri kaynakları, yapısal farklılıklar, veri türlerinin çeşitliliği veya kaynakların farklılığı gibi nedenlerden ötürü büyük miktarda veri işleme, entegrasyon ve

analiz zorlukları ortaya çıkarabilir Bu tür veri türleriyle başa çıkabilmek için uygun araçlar, algoritmalar ve yöntemlerin kullanılması önemlidir. Veri analitiği ve yapay zekâ teknikleri, bu çeşitli veri kaynaklarından anlamlı bilgi elde etme konusunda önemli rol oynamaktadır.

İnternet'teki heterojen kaynaklardan veri toplamak için kullanılan yöntemler arasında en yaygın olanı Web tarayıcılarıdır. Web tarayıcıları, Python ve BeautifulSoup gibi çeşitli diller ve kütüphaneleri kullanarak Web sayfalarının HTML içeriğini analiz edebilir, belirli etiketleri ve veri alanlarını tanımlayabilir ve bu verileri toplayabilirler. Web tarayıcıların dışında birçok hizmet sağlayıcı, verilerine program ile erişim sağlamak için API'ler sunar. Özellikle sosyal medya platformlarının API'leri, kullanıcı profilleri, gönderiler ve diğer verilerin çekilmesine olanak tanır. RSS beslemeleri ise belirli bir Web sitesinin içeriğini otomatik olarak takip edip güncellemeleri almanıza olanak sağlar. Bu yöntem genellikle haber portalları, bloglar ve diğer içerik sağlayıcılarından veri toplamak amacıyla kullanılmaktadır.

Web tarayıcılarının etkili bir şekilde çalışabilmesi için, kullanıcı isteklerine bağlı olarak mümkün olan en geniş kapsamda ve farklı sayfalara en kısa sürede ulaşma yeteneği önemlidir. Ancak, Web tarama sürecinde bağlantıların takip edilmesi nedeniyle, hiçbir Web sayfasına ulaşılamayan veya farklı yapılarla tasarlanmış sayfalara erişim sağlanamayan durumlar ortaya çıkabilir. Web ortamındaki Web sitelerinin sayısındaki üstel artış, etkili bir tarama için uygun bir tarama stratejisinin belirlenmesini daha da kritik hale getirmiştir.

3. İNTERNET'TE HETEROJEN VERİ KAYNAKLARINDAN VERİ TOPLANMASI, ENTEGRASYONU VE GÜNCELLENMESİ

Günümüzde İnternet'i kullanarak Web üzerindeki verilere erişmek hayatımızın önemli bir parçası haline gelmiştir. Şuanda mevcut dünya nüfusu 7,9 milyar olup 5,2 milyar (%66.2) İnternet kullanıcısı mevcuttur [75]. Bu sayı 2012'de 2,4 milyar [76] iken 2022'de 5,2 milyara yükselmiş, yani yaklaşık %116 artmıştır.

İnternet kullanıcı sayısındaki artış, Web üzerindeki veri miktarının artması anlamına gelmektedir. Web, her geçen gün hızla büyüyen ve çeşitli veri türlerini içeren devasa bir veri deposudur. Bu devasa miktardaki veri kümesi kullanılarak istenilen bilgilerin çekilmesi, veri tabanlarına entegre edilmesi ve güncellenmesi gibi süreçlerde bir dizi zorluğu beraberinde getirmektedir.

3.1. Heterojen Veri Kaynaklarından Veri Toplanması

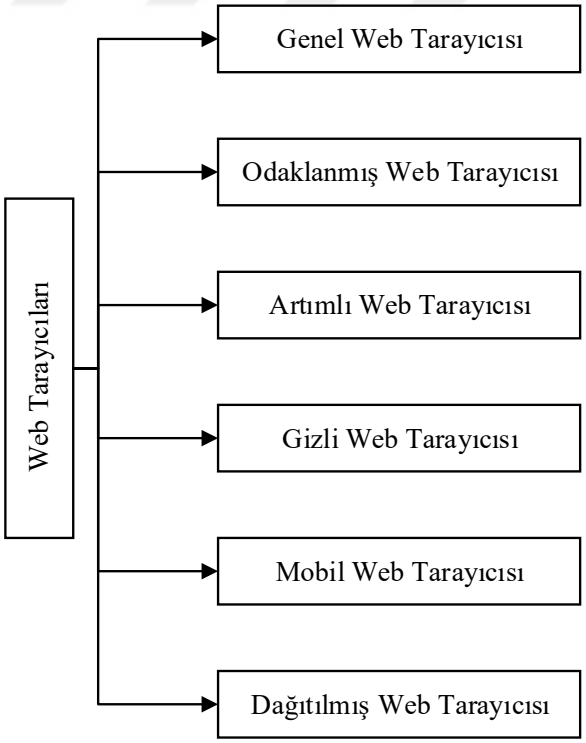
Günümüzde, İnternet gibi devasa bir veri deposundan istenilen bilgilere zamanında ve doğru bir şekilde erişmek, hayati bir öneme sahiptir. Ancak, bu büyük veri havuzundaki bilgilere ulaşmak her geçen gün daha da zorlaşmaktadır [77]. Bu zorlukları aşabilmek, Bu zorluğun üstesinden gelmek ve Web üzerindeki verilere erişim sağlamak için arama motorları yaygın bir şekilde kullanılmaktadır. Web sayfalarını tek tek ziyaret etmek ve her ziyarette istenilen bilgileri elle aramak neredeyse imkânsızdır. Bu nedenle arama motorları bu süreci kolaylaştırmaktadır.

Farklı tarayıcı türleri olmasına rağmen temelde genel ve odaklı olmak üzere iki ana tarayıcı türü vardır. Odaklı tarayıcılar, belirli bir konu ya da alan ile ilgili sayfaları tarama eğilimindedirler. Bu tarayıcılar, önceden belirlenmiş bazı verileri kullanarak erişim sayısını sınırlarlar [78]. Diğer yandan, genel tarayıcılarda ise böyle bir sınırlama yoktur. Arasu ve arkadaşları [79] tarafından yapılan çalışmada, odaklı taramanın kullanıcılar arasında değişen ön bilgilere göre değişiklik gösterebileceği vurgulanmış ve genel tarayıcıların gerçek dünyada daha önemli olduğu özellikle belirtilmiştir. Bu iki temel tarayıcı türünün yanı sıra çeşitli Web tarayıcı türleri de geliştirilmiştir.

3.1.1. Web tarayıcılarının sınıflandırılması

Web sayfalarının sayıları ve boyutları sürekli olarak artmaktadır, bu da Web sayfalarını arama ve indeksleme işlemlerinin giderek karmaşık hale gelmesine neden olmaktadır. Bu karmaşıklık, arama motorlarının, Web'in devasa veri kaynağını tamamen kapsayan bir tarama gerçekleştirmekte zorlanmalarına yol açmaktadır. Günümüzde tüm Web'i tarayan ve indeksleyen bir tarayıcı yoktur. Hedefi tüm Web olan tarayıcılar, genel tarayıcı olarak adlandırılır. Ancak, genel tarayıcılar, kullanıcı taleplerine uygun verilere ulaşmada yetersiz kalmaktadır. Bu nedenle, çeşitli özel tarayıcı türleri geliştirilmiştir, bu türler belirli ihtiyaçlara odaklanarak daha etkili ve verimli bir tarama sağlamayı amaçlamaktadır.

Web tarayıcıları, kullanım amacı, kapsam, tarama politikası ve kullanılan stratejilere bağlı olarak çeşitli araştırmacılar tarafından farklı yöntemlerle sınıflandırılmıştır. Bu sınıflandırmalar, tarayıcıların özelliklerini ve işlevselliğini anlamak, karşılaştırmak ve belirli ihtiyaçlara uygun tarayıcıları seçmek için önemli bir temel sağlamaktadır. Şekil 3.1'de Web tarayıcılarının kapsamlı bir sınıflandırılması yapılmıştır.



Şekil 3.1. Web tarayıcılarının sınıflandırılması

Genel Web tarayıcıları

Genel Web tarayıcıları, kapsamı tüm Web olan tarayıcılardır. Belirli bir tohum URL kümesi ile taramaya başlar ve BFS ile taramaya devam eder. Google, Bing, Yandex vb. gibi ticari tarayıcıların çoğu genel tarayıcılardır. Bu ticari tarayıcıların mimari ve algoritmaları genellikle sınırlı bir şekilde açıklanmıştır, çünkü çoğunlukla ticari amaçlarla kullanıldıkları için detaylı araştırmalara konu olmamışlardır.

Heydon ve Najork çalışmalarında, Mercator ismini verdikleri ölçeklenebilir ve genişletilebilir genel bir Web tarayıcısını geliştirmişlerdir. Çalışmada, ölçeklenebilir bir tarayıcının ana bileşenleri açıklanmış ve tasarım alternatifleri tartışılmıştır. Yazarlar, Mercator'u 8 günlük bir çalışma süresince, Google ve İnternet Arşiv Tarayıcısı ile karşılaştırmışlardır. Tarayıcının kapsamı, simülasyon çalışmalarında sınırlı tutulmuş fakat tüm Web olacak şekilde genişletilebileceği belirtilmiştir [80].

Günümüzde en popüler genel Web tarayıcısı olan Google'ın kurucularından Page ve arkadaşları, Web sayfalarının göreceli önemini ölçmek için PageRank adlı bir sıralama hesaplama yöntemini önermiştir. PageRank, Web'in grafiğine dayalı olarak her Web sayfası için bir sıralama oluşturan bir algoritmadır. PageRank'in arama, göz atma ve trafik tahmininde bulunan uygulamaları bulunmaktadır. Günümüzde en popüler ve etkili arama motorlarından biri olan Google, bu algoritmayı temel alarak geliştirilmiştir. [81].

Ticari tarayıcılara alternatif olarak, açık kaynak kodlu tarayıcılar da geliştirilmiştir. Bu tarayıcılar genellikle farklı platformlarda çalışabilir ve çeşitli tarayıcı türlerini destekleyebilir, bu da kullanıcılara çeşitli seçenekler sunar. Ayrıca, lisans bilgileri, kullanıcıların yazılımın kullanım şartlarına dair bilgi edinmelerine yardımcı olur. Bu açık kaynak tarayıcılar, Web tarayıcısı çeşitliliğini artırarak kullanıcı deneyimini zenginleştirir ve Web üzerinde özgür bir gezinme imkânı sunar.

Çizelge 3.1, Github'ın yıldız referans sistemine göre tercih edilen 15 açık kaynak tarayıcının çeşitli özelliklerini sunmaktadır. Bu özellikler programlama dili, uyumlu işletim sistemi, tarayıcı türü ve lisans bilgilerinden oluşmaktadır.

Çizelge 3.1. Açık kaynak tarayıcılar

Açık Kaynak Tarayıcı	Programlama Dili	İşletim Sistemi	Tarayıcı Türü	Lisans
Scrapy	Python	Çapraz Platform	Genel	BSD
Apache Nutch	Java	Çapraz Platform	Dağıtılmış	Apache
Heritrix	Java	Linux	Dağıtılmış	Apache
Crawl4J	Java	Çapraz Platform	Genel	Apache
PySpider	Python	Windows	Genel	Apache
Cola	Python	Çapraz Platform	Dağıtılmış	---
BUbiNG	Java	Linux	Dağıtılmış	Apache
WebMagic	Java	Çapraz Platform	Dağıtılmış	Apache
Portia	Python	Çapraz Platform	Dağıtılmış	BSD
Gnu Wget	C	Çapraz Platform	Genel	GNU
WebSphinix	Java	Çapraz Platform	Genel	GNU
WebCollector	Java	Çapraz Platform	Dağıtılmış	GPL
Node-Crawler	JavaScript	Windows	Genel	MIT
HAWK	C#	Windows	Genel	Apache
Goutte	PHP	Çapraz Platform	Genel	MIT

Çizelge 3.1'de görüldüğü üzere geliştirilen açık kaynak tarayıcılarda Java ve Python programlama dilleri tercih edilmiş, büyük çoğunluğu platformdan bağımsız çapraz platformda inşa edilmiş ve en çok lisanslama Apache tarafından gerçekleştirilmiştir.

Odaklanmış Web tarayıcıları

Odaklanmış Web tarayıcıları, belirli bir konu ile ilgili arama yapan tarayıcılardır. Bu tarayıcı, genel tarayıcılar gibi tüm bağlantılar yerine, kullanıcıların tercihine göre belirli bir alanı ya da konuyu taramak için kullanılır.

İlk odaklanmış Web tarayıcısı, Chakrabarti ve arkadaşları tarafından geliştirilmiştir [82]. Önerilen odaklanmış Web tarayıcısı, belirlenen bir dizi konu ile ilgili Web araması gerçekleştirmektedir. Yazarlara göre odaklı tarayıcı, uzmanlığı örneklerden öğrenen ve Web'i keşfeden bir sistemdir [82]. Odaklı tarayıcılar, Web'in konu ile ilgili olmayan bölgelerinden kaçınarak konu ile ilgili olabilecek sayfaları bulur. [83]'de yazarlar beş kategoride sınıflandırılan odaklanmış tarayıcı yaklaşımlarının bir incelemesini sunmuştur. Bu yaklaşımlar öncelik tabanlı tarayıcı, yapı tabanlı tarayıcı, öğrenme tabanlı tarayıcı, bağlam tabanlı tarayıcı ve diğer odaklı tarayıcılardır. [84]'da ise yazarlar odaklanmış

tarayıcıları klasik, semantik ve öğrenen odaklanmış tarayıcılar olmak üzere üç gruba ayırmışlardır. İlgi alanına ve aranan kelimelere göre istenilen bilgileri hızlı ve doğru bir şekilde getirdiği için odaklı tarama önemli bir araştırma alanı olmuştur. [85]'de odaklı tarayıcıların aranan konu ile ilgili tahminlerini iyileştirmek için önlemler geliştirilmiş ve [86]'de odaklı taramada kullanılan yöntemler tartışılmıştır.

Web'deki konuyla ilgili verileri almak ve yeni URL'leri çıkarmak için anahtar kelime veya arama ölçütlerine dayalı anahtar kelime tabanlı yaklaşımlar kullanılmaktadır [87]. Anahtar kelime tabanlı yaklaşımda, anahtar kelime içeren Web sayfalarından veriler ve yeni URL'ler çıkarılmakta ve arama ile ilgisiz sayfalar görmezden gelinmektedir [27]. Kumar ve arkadaşları, anahtar kelime sorgusu tabanlı odaklı tarayıcıyı tartışmışlardır. Sorgu tabanlı tarayıcının, harcanan zaman ve hassasiyet açısından önceki geniş öncelikli tarayıcılardan daha verimli olduğu sonucuna varmışlardır. Çalışmada tohum URL'ler, anahtar veri kümeleri ile Google API'leri ve tohum arama ara yüzü kullanılarak elde edilmiştir [88].

Web'in belirli bir bölümünü almak için kümeleme, sınıflandırma, bulanık küme, Naive Bayes vb. gibi veri madenciliğini kullanan yaklaşımlar kullanılmaktadır [89-91]. Safran ve arkadaşları çalışmalarında, ziyaret edilmeyen URL'lerin ilgi düzeyini tahmin etmek için dört ilgi özniteliği kullanan, öğrenmeye dayalı yeni bir odaklı tarama yaklaşımı sunmuşlardır [92]. Kullanılan özellikler; sayfada bulunan URL sözcükleri, URL'lerdeki bağlantı metinleri, URL sözcüklerinin bağlı olduğu alan adlarını işaret eden ana sayfalar ve bu URL'leri çevreleyen metinlerdir. Deneysel çalışmalarında Naïve Bayesian sınıflandırma modeli benimsenmiştir. Tohum URL seçim işlemlerinde, ilk olarak konu kelimelerinden olan "Borsa" kelimesi Google, Yahoo ve MSN arama motorlarına sorgu olarak gönderilmiştir. Aday tohum URL'ler, belirlenen arama motorlarından elde edilen en iyi k-URL'dir Son olarak tohum URL'ler, aday k-URL'lerin üç arama motorunun en az ikisinin sonuç listesinde görünen URL'ler alınarak oluşturulmuştur [92].

Odaklanmış Web tarayıcılarının performanslarını iyileştirmek için ontoloji tabanlı yaklaşımlar kullanılmaktadır. Ontoloji, alanın öğeleri arasında var olabilecek kavramları ve ilişkileri tanımlayan bir kavramsallaştırmanın özelliğidir [93]. Bu yaklaşımda ontoloji, bilgi deposunu yapılandırmak ve ilgisiz verileri filtrelemek amacı ile kullanılmaktadır [94, 95]. Agre ve Dongre çalışmalarında, ontoloji tabanlı İnternet tarayıcısı için yalnızca konu ile ilgili Web sitelerini alan, taramada en iyi tahmini kullanan ve tarama performansının

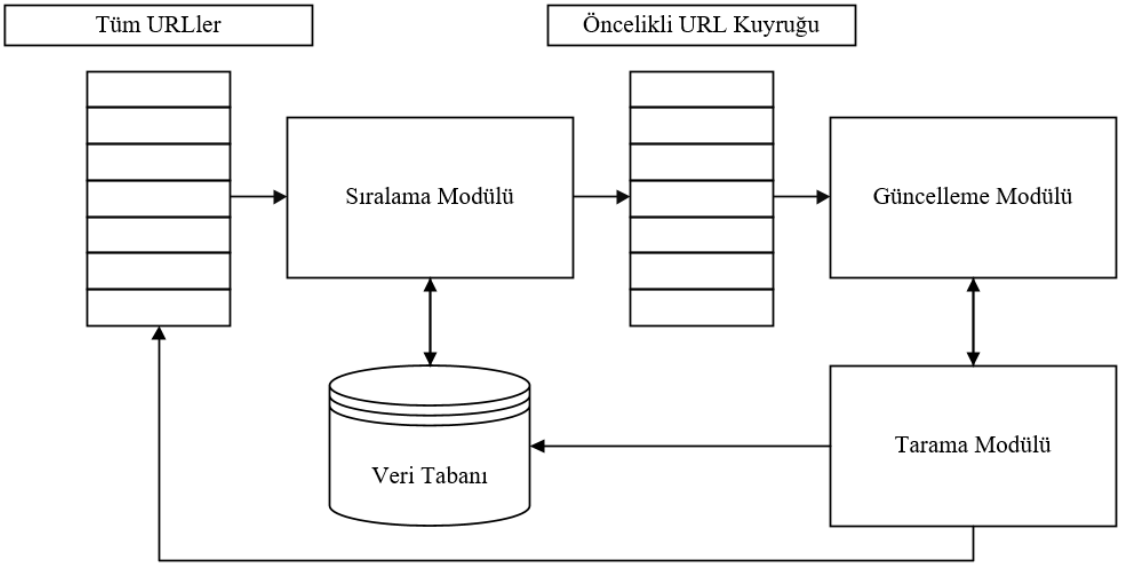
iyileştirilmesine yardımcı olan bir odaklanmış tarayıcı kullanmışlardır [96]. Du ve arkadaşları, odaklı Web tarayıcının konusu ile ilgili ontolojiden yararlanarak, kullanıcıların ilgi alanlarına dayalı olan tohum URL'leri seçmek için yeni bir yöntem geliştirmişlerdir. Bu çalışmada ontoloji oluşturma yaklaşımının üç adımını tanımlamışlardır. Bunlar çalışmada kavram seçimi, optimize edilmiş kavram kafesinin oluşturulması ve kavram kafesinin kullanıcı-çıkar ontolojisine eşlenmesi sürecidir. Bu kullanıcı ilgi ontolojisi, örtük kavramları ve aralarındaki ilişkiyi tanımlamıştır. Yazarlara göre etkili tohum URL seçimi ile tarayıcı, arama sorgusuyla ilgili daha verimli sonuçlar üretmektedir [97].

Odaklanmış Web tarayıcılarında, sayfalar arasındaki anlamsal benzerlikleri belirlemenin temel amacı, arama konusuyla ilişkilendirilmiş sayfaların birbirleriyle olan semantik benzerliklerini tespit etmektir. Semantik tabanlı yaklaşımda anlamsal benzerlik oranını belirlemekteki temel şart içerik analizinin yapılmasıdır [98]. İçerik analizi, sayfa içeriğinin anlamını çıkarmak için çeşitli algoritmaları içerir. İçerik analizi yapılırken en çok kullanılan algoritmalardan biri olan PageRank [81], sayfalar arasındaki bağlantı yapılarına dayalı olarak sayfaların önemini değerlendiren bir algoritmadır. Görsel tabanlı sayfa segmentasyonu [99], sayfaların görsel içeriğini analiz ederek, görsel benzerliklere dayalı olarak sayfaları anlamsal olarak sınıflandıran bir yaklaşımdır. SiteRank [100], bir Web sitesinin genel önemini değerlendirmek için kullanılan bir algoritmadır. Sayfaların bir Web sitesinin toplam etkileşimini yansıtan SiteRank, anlamsal benzerlik analizinde kullanılabilir. Son olarak densometrik segmentasyon [101], sayfa içeriğinin yoğunluk analizini kullanarak, sayfalar arasındaki anlamsal benzerlikleri belirleyen bir yaklaşımdır. Bu algoritmalar, odaklanmış Web tarayıcılarında semantik benzerlik analizine temel oluşturur.

Odaklanmış Web tarayıcılarında literatürde en çok kullanılan performans ölçütleri hasat oranı, kesinlik ve ilişki oranıdır. Hasat oranı odaklı tarayıcılarda kullanılan bir performans ölçütü olup ilgili Web sayfaları ile taranan Web sayfaları arasındaki oranı temsil etmektedir. Kesinlik oranı ya da diğer adıyla hassasiyet oranı, odaklanmış tarayıcılarda verimliliği değerlendirmek için elde edilen verilerin kalitesinin ölçüsüdür. Kesinlik değerinin yüksek olması odaklı Web tarayıcısının ilgisiz sayfaları o derecede iyi reddettiğini göstermektedir. Kesinlik, temel olarak, döndürülen sonuçların toplam sonuçlara oranını göstermektedir. Geri çağırma oranı ise bir arama talebine cevap olarak alınan ilgili sayfaların oranını ifade etmektedir.

Artımlı Web tarayıcıları

Artımlı Web tarayıcısı, tarama işlemini her defasında yeniden başlatmak yerine, sayfaları düzenli olarak güncelleyen geleneksel bir tarayıcı türüdür. Artımlı tarayıcı Web'i sürekli tarar ve sayfaları periyodik olarak yeniden ziyaret eder [102]. Artımlı Web tarayıcısının temel hedefi yerel koleksiyonu taze tutmak ve yerel koleksiyonun kalitesini artırmaktır [103]. Odaklanmış taramada, artımlı Web tarayıcılardan toplanan verilerin ilgi düzeyi diğer tarayıcılara kıyasla az olabilmektedir. Ayrıca artımlı tarayıcılar gizli Web'i taramada başarısız olurlar. Artımlı Web tarayıcısının mimarisi Şekil 3.2'de gösterilmiştir [104].



Şekil 3.2. Artımlı Web tarayıcı mimarisi

Gupta ve arkadaşları, artımlı Web tarayıcısında kullanmak için geliştirdikleri mekanizma ile her bir URL'e sayfa sıralaması, sayfa yenileme sıklığının hesaplanması ve Web sayfasının içeriğinin kullanıcıların ilgi alanlarına uygun olup olmadığı durumuna göre bir öncelik vermektedirler. Tohum URL'ler eğlence, turizm, cep telefonları, spor, mülk olmak üzere beş farklı alandan olup daha az orta düzeyde ve oldukça dinamik Web sayfalarının bir karışımından oluşmaktadır. Kapsam olarak da belirtilen beş farklı alan ile ilgili Web sayfalarını içermektedir [105].

Pavai ve Geetha, geliştirdikleri artımlı Web tarayıcısı algoritması ile bir Web sayfasının, gelecekteki değişim zamanını tahmin etmeyi amaçlayan ve veri madenciliğine dayalı yaklaşımı kullanmışlardır. Yüzey Web sayfalarını ve derin Web veri tabanlarını aşamalı olarak taramak için Bayes olabilirlik tahmincisini kullanan yeni bir istatistiksel yaklaşım sunmuşlardır. Tohum URL'leri almak için çapraz dil bilgi erişimi (Cross Lingual Information Access - CLIA) projesinin tohum URL'lerini (20 Ağustos 2015'te taranmış) ve sorgu koleksiyonlarını kullanmışlardır. Çalışmanın kapsamı da film, iş, kitap ve uçak bileti rezervasyonu olmak üzere dört alan ile sınırlandırılmıştır [106].

Santos ve arkadaşları, sayfa güncelliğine dayalı yaklaşımı benimsemiş ve artımlı Web tarayıcısının zamanlama mekanizması için yeni bir algoritma geliştirmişlerdir. Geliştirilen algoritma ile Web sayfalarını, son taranmalarından bu yana değiştirilme olasılıklarına göre sıralamak ve Web tarayıcılarının planlayıcıları tarafından kullanılacak puan işlevlerini otomatik olarak oluşturmak için bir genetik programlama çerçevesi sunmuşlardır. Çalışmada, tarayıcının tazelik oranına odaklanılmış ve kapsam olarak bir sınırlama verilmediğinden tüm Web hedeflenmiştir [45].

Tan ve Mitra, geliştirdikleri artımlı Web tarayıcısında kümeleme yöntemini kullanmışlardır. Mevcut Web sayfalarını, hem statik hem de dinamik özelliklerini kullanarak 100 kümeye ayırmışlardır. Çalışmaya göre tarayıcı, sayfaların son indirilmelerinden bu yana değişip değişmediğini kontrol etmek için bir kümeden bir Web sayfası örneği getirmektedir. Bir kümedeki önemli sayıda sayfa değiştiyse, kümedeki diğer Web sayfaları da karşıdan yüklenmektedir. Web sayfalarının kümeleri farklı değişiklik sıklıklarına sahiptir. Çalışma artımlı taramayı aynı Web sayfaları ve sınırlı kapsamda gerçekleştirmiştir. Belirlen Web sayfalarının dışındaki sayfalar taranmamıştır [107].

Shi ve arkadaşları çalışmalarında, Scrapy adlı tarayıcı mimarisi temelinde artımlı bir Web tarayıcısı geliştirmişlerdir. Tohum URL olarak belirledikleri haber sitelerinin bağlantılarını kullanmışlardır. Geliştirilen tarayıcı, Web sitesindeki haber bilgilerini başarılı bir şekilde tarayarak artımlı tarama gerçekleştirmiştir. Buna ek olarak tarayıcı, Bloom filtresi ile haber sitesini izlemiş ve güncellenen haberleri veri tabanında saklamıştır. Ancak Web tarayıcısının en büyük dezavantajı, kapsamı haber siteleri olduğu için genel tarama yapmamaktadır [72].

Nagar ve Singhal, artımlı bir Web tarayıcısının tekrar ziyaret sıklığı problemi üzerinde durmuş ve Web sayfalarına ziyaret sıklığını yönetmek için yeni bir yaklaşım önermişlerdir. Çalışmaya göre sunucudaki bir log dosyası veri tabanı, Web sayfası kimliğini, sayfadaki URL'leri ve başarılı bağlantı sayılarını kaydetmektedir. Başarılı bağlantı sayısı belli bir değeri geçen URL'lerin sıralaması yükseltilmektedir. Başarılı bağlantı sayısı bir log dosyasında tutulmakta ve bu bilgi, farklı Web sayfalarının öncelikli sıralamasını oluşturmada kullanılmaktadır. Bu öncelikli sıralama bilgisi kullanıcıların arama geçmişinden oluşan log dosyası veri tabanına eklenmektedir. Bunun sonucunda Web'den, yüksek dereceli sayfalar, kullanıcıların arama geçmişine göre tarayıcı tarafından indirilerek veri tabanına kaydedilmektedir [108].

Artımlı Web tarayıcılarının en önemli özelliği farklı zaman dilimlerinde Web sayfalarını tarayıp değişiklikleri tespit etmektir. Artımlı Web tarayıcılarında başarı oranı, toplam denemeler arasındaki başarı yüzdesi olarak tanımlanmaktadır. Başarı oranının yüksek olması kapsamın geniş olması ve daha az gereksiz istek içermesi anlamına gelmektedir. Artımlı Web tarayıcılarındaki temel prensip ziyaret edilen Web sayfalarını güncellenme sıklıklarına bağlı olarak tekrar ziyaret etmektir. Kullanılan tarama algoritmasına göre ziyaret sıklıkları sabit ya da değişken aralıkta olup farklılık göstermektedir. Buradaki en önemli konu sık güncellenen sayfaları nadiren ya da hiç güncellenmeyen sayfalara göre daha kısa süre sonra ziyaret etmektir.

Gizli Web tarayıcıları

Web sayfalarındaki bilgiler genellikle yüzeysel ve gizli olmak üzere iki ana kategoride saklanır. Yüzeysel Web'deki verilere herhangi bir tarayıcı ulaşabilir. Gizli Web'deki verilere ise oturum açma formu, arama veya sorgu arabirimleri ile ulaşıldığından, arama motorlarının kapsamı dışındadır. Gizli Web'deki Web sayfaları, bir veri tabanındaki sorgu arabirimleri aracılığıyla gönderilen sorgulara yanıt olarak birleştirilir. Web'de formların, ara yüzlerin vb. arkasına gizlenmiş ve dizine eklenemeyen büyük miktarda veri bulunmaktadır [109]. Bu verilere ulaşan tarayıcılar gizli Web tarayıcılarıdır. Literatürde bu tarayıcılar derin Web tarayıcıları olarak da anılmaktadır.

Zerfos ve arkadaşları, gizli Web'deki verilere sorgu ile ulaşmak için üç farklı sorgu oluşturma politikasını önermişlerdir. Bunlar anahtar kelime listesinden rastgele sorguları

seçen bir politika, genel bir metin koleksiyonunda sorguları sıklıklarına göre seçen bir politika ve uyarlanabilir bir şekilde iyi bir sorguyu temel alan bir politikadır. Yapılan deneysel çalışmalar, bu politikaların kullanıldığı dört gizli Web sitesinde, yaklaşık 100 sorgu sonrasında %90'dan fazla başarı elde edildiğini göstermiştir [110].

Kaur ve Geetha, gizli Web'i taramak için SIMHAR adını verdikleri, ağaç tabanlı yaklaşımı benimseyen, dağıtılmış bir Web tarayıcısı önermiş ve uygulanmışlardır. Scrapy Framework ve Redis sunucusunu birleştirerek taramayı üç aşamaya bölmüştür: uyarlama, ilgili kaynak seçimi ve temel içerik çıkarma. Tarayıcı, aranabilir formları doğru bir şekilde algılamış ve göndermiştir. Çalışmada önerilen tarayıcı, gizli Web'den belirli bir konu ile ilgili sonuçları aradığından odaklanmış tarayıcı gibi çalışmaktadır. Tohum URL'ler ise sisteme yazarlar tarafından konu ile ilişkili olarak DMOZ veri kümesinden seçilerek eklenmiştir. Veri kümesi, konuyla ilgili 260,000'den fazla URL içeren 6 farklı alandan seçilmiş tohum URL'leri içermektedir [111].

Gupta ve Bhatia çalışmalarında, HiCrawl adını verdikleri alana özgü yaklaşımı benimseyen ve tıp alanındaki siteleri taramayı hedefleyen odaklanmış gizli bir Web tarayıcısını önermişlerdir. Önerilen tarayıcı beş temel bölüme sahiptir. Bunlar indirici, Web sayfası çözümleyicisi, form çözümleyicisi, etki alanına özgü bir veri havuzu kullanan form işlemcisi ve etki alanına özgü sınıflandırma hiyerarşisidir. HiCrawl tohum URL kümesi, tıp alanına ait Web sayfalarının URL'lerinden oluşmaktadır. Bu tohum URL kümesi, Google, Yahoo vb. gibi herhangi bir güvenilir Web arama motorunun dizinlerinden elde edilmekte ve bir FIFO kuyruğu olan tohum URL sınırında saklanmaktadır [112].

Bhatia ve arkadaşları çalışmalarında, etki alanına özgü gizli Web tarayıcısı AKSHR'ı önermişlerdir. Bu tarayıcının özellikleri arasında arama ara yüzlerinin otomatik indirilmesi, etki alanına özgü ara yüz yaklaşımı kullanarak anlamsal eşlemelerin tanımlanması ve arama ara yüzlerini otomatik doldurma yeteneği bulunmaktadır. Çalışmanın deneysel değerlendirmesi için kitaplar, havayolu, elektronik, emlak ve filmler olmak üzere beş alan dikkate alınmış ve arama ara yüzü tarayıcısına her etki alanı için bir tohum URL sağlanmıştır. Önerilen tarayıcının ortalama tarama hassasiyeti %76,9 ile %93,1 arasında değiştiği görülmüştür. Genel hassasiyet ortalaması, geri çağırma ortalaması ve genel ortalama F-ölçüsünün sırasıyla %85.23, %86.32 ve %85.44 olduğu ve AKSHR'nin hem tutarlı hem de verimli olduğunu belirtilmiştir [113].

Gizli Web tarayıcılarında performans ölçütü olarak genel tarayıcılarda kullanılan ölçeklenebilirlik, tazelik ve benzeri ölçütler kullanılmaktadır. Ayrıca hedef gizli Web sitesinde bulunan toplam sayfa sayısının, getirilen sayfa sayısına oranı olan hasat oranı da önemli bir ölçüttür. [114]'de yazarlara göre tarayıcı tarafından taranan toplam sayfa sayısı P_c, toplanan derin Web veri tabanının alana özgü toplam sayfa sayısı N_f ve tarayıcının arama alanındaki alan özgü aranabilir sayfaların toplam sayısı N_{cf} ise hasat ve kapsama oranı Eş. 3.1 ve Eş. 3. 2'deki gibi ifade edilebilir.

$$Hasat\ Oranı = \frac{N_{cf}}{P_c} \tag{3.1}$$

$$Kapsama\ Oranı = \frac{N_{cf}}{N_f} \tag{3.2}$$

Burada hasat oranı ve kapsama oranı N_{cf} arttığında artmaktadır. P_c ve N_f arttığında ise azalmaktadır. Web'de bulunan kaliteli verilerin büyük çoğunluğu çeşitli kurumlar ve özel firmalar tarafından eklenmektedir. Bundan dolayı gizli Web tarayıcılarının kapsamı yüzey Web tarayıcılarına göre daha geniştir ve bu da gizli Web tarayıcılarının önemini arttırmaktadır. Gizli Web tarayıcıları, verileri form doldurma, üye girişi ve sorgulamalar sonucu almaktadır. Gizli Web tarayıcılarının, verilere direk ulaşamaması beraberinde birçok dezavantajı getirmektedir. Bunlar ölçeklenemez olması, ağ üzerinde ağır yük oluşturması, yarı otomatik olması ve belirli bir formatta arama yapamaması olarak sıralanabilir.

Mobil Web tarayıcılar

Mobil Web tarayıcılarında sayfa seçim ve filtreleme işlemleri, geleneksel tarayıcılardan farklı olarak arama motorları tarafından değil, sunucu tarafından gerçekleştirilir. Mobil taramada, verilerin koda taşınması yerine kodun verilere taşınması söz konusudur. Bu durum, geleneksel Web tarayıcılarının neden olduğu ağ yükünü azaltarak daha etkili bir mobil tarama deneyimi sağlar [109]. Web taraması bağlamında, bir tarayıcının, veri kaynağına yani Web sunucusuna geçiş yapabilme yeteneği mobilite olarak adlandırılmaktadır [115].

Kumar ve Bhatia çalışmalarında, gizli Web'den veri almak için mobil taramanın çeşitli avantajlarını kullanan bir yaklaşım önermişlerdir. Tarama sırasında geleneksel olarak

verileri koda taşımak yerine, kodu verilere taşıma kavramını kullanmışlardır. Önerilen yaklaşım, yinelemeli olarak sorguları çalıştırarak yerel veri tabanından verileri alır. Sorgular, potansiyel anahtar kelimeler kullanılarak, minimum sayıda sorguda maksimum veri kümesini kapsayacak şekilde oluşturulmuştur. Yapılan çalışma, odaklı bir tarama yapmak için Hindistan dışında yaşayan Hint kökenli akademisyenlerin verilerini kapsamaktadır. Ayrıca, tohum URL'leri yazarlar tarafından belirlenmiş sitelerden manuel olarak alınmıştır [115].

Anbukodi ve Manickam yaptıkları çalışmada, genel tarayıcıların ağ bant genişliği kullanımı, merkezi işlem birimi (Central Process Unit - CPU) döngülerinin ve belleğinin kullanımı, işletim sistemlerinin sınırları vb. nedenler ile uzak sunucuda yüke sebep olan durumların ortadan kaldırılması için mobil tarayıcı mimarisi önermişlerdir. Önerilen mobil taramanın avantajı, tarama işlevselliğini tüm Web üzerinde dağıtmamıza izin vermesi ve son taramadan bu yana değiştirilmemiş sayfaları filtreleme yeteneğidir. Yerelleştirilmiş veri erişimi, uzak sayfa seçimi, filtrelenmesi ve sıkıştırılması gibi özelliklere sahiptir. Bu tür bir taramada, denemeler sanal bir sistem üzerinde gerçekleştirilmiş ve kapsam bir veya birkaç sunucuyla sınırlıdır, tohum URL'leri de yazarlar tarafından belirlenmiş birkaç Web sitesinden oluşmaktadır [116].

Nath ve Bal, mobil Web tarayıcılarının etkili bir şekilde çözebildiği Web'in katlanarak büyümesi ve buna bağlı olarak ortaya çıkan trafik ve bant genişliğinin tüketimi sorununu ele almışlardır. Çalışmalarında mobil tarayıcıya dayalı verimli bir indeksleme sistemi sunarak bant genişliği tüketim sorununa çözüm önerileri getirmişlerdir. Yazarlar tarafından geliştirilen tarayıcı, uzak sayfayı ziyaret ederek son taramadan sonra değişiklik için sayfayı analiz eder ve sadece değiştirilmiş olanları döndürür. Yazarlar tarafından belirlenen 100 Web sayfasında 30 gün boyunca denemeler yapılmıştır. Önerilen mobil tarayıcı sistemi, geleneksel tarayıcı sistemiyle sayfa değiştirme davranışı, ağdaki yük ve bant genişliği olmak üzere üç parametre ile karşılaştırılmıştır. Deneysel sonuçlar, önerilen mobil tarayıcının geleneksel tarayıcı sistemlerinden daha verimli olduğunu göstermiştir [117].

Mobil tarayıcıların temel özelliği, sayfaların değişip değişmediğini sunucu üzerinde kontrol ederek sadece değişen sayfaların alınmasını sağlamaktır. Bu sayede bant genişliğinden ve işlemci performansından kazanç sağlanmaktadır.

Dağıtılmış Web tarayıcıları

Dağıtılmış Web tarayıcıları, coğrafi olarak farklı konumlar üzerinde çalışan tarayıcılardır. Farklı konumlarda çalışması, ağ trafiğini azaltma, yüksek ölçeklenebilirlik ve yükü dengeli bir şekilde dağıtma eğilimi gibi avantajları beraberinde getirir. Ancak, bu avantajlarına rağmen, verileri yönetme ve gizli ağın taranması konularında yetersizdir [118]. Dağıtılmış Web tarayıcısı, geleneksel tarayıcılar temel alınarak geliştirildiğinden dolayı çalışma prensibi ve temel yapısı geleneksel Web tarayıcısına benzemektedir [119].

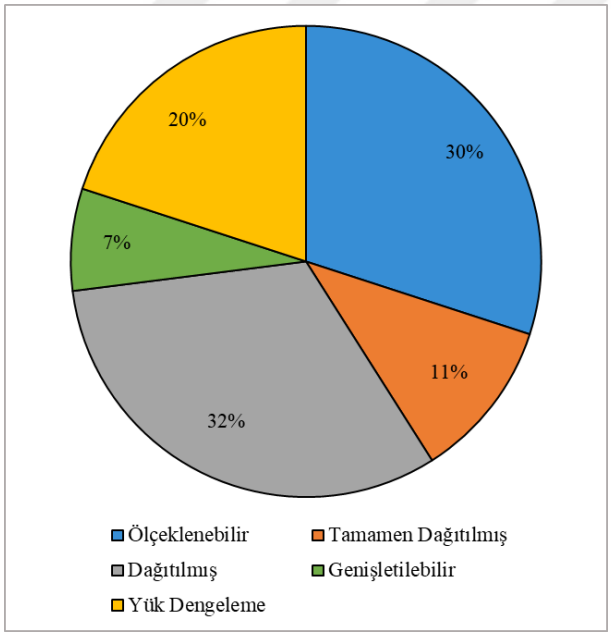
Cai ve Zhang çalışmalarında, dinamik Web sayfalarının ayrıştırılmasındaki zorluklar nedeniyle eksik bilgi alınması ve tarayıcı verimliliği gibi sorunları çözmek amacıyla "Dis-Dyn" adını verdikleri bir dağıtılmış Web tarayıcısı önermişlerdir. Yazarlar geliştirdikleri Dis-Dyn tarayıcısında, dinamik sayfaların ayrıştırılması için HtmlUnit kullanılmış ve tarayıcının verimliliğini artıran dağıtım özelliğini gerçekleştirmek için Redis ve sıfır mesaj kuyruğu (Zero Message Queue - ZMQ) tercih edilmiştir. Denemeler, sadece bir dinamik film sitesinde gerçekleştirilmiş ve bu nedenle kapsam bu siteyle sınırlı kalmıştır [120].

Yu ve arkadaşları çalışmalarında, Hadoop platformunda çalışan dağıtılmış bir Web tarayıcı modeli önermişlerdir. Önerilen tarayıcı, geleneksel dağıtık tarama sisteminde var olan uzantı ve yük dengeleme problemini çözmeyi amaçlayan, bulut bilişime dayalı, dağıtık bir Web tarayıcısıdır. Yazarlar tarafından seçilen tohum URL'ler ile başlayıp tüm Web'i hedef alan genel bir tarama yapılmıştır. Tarama sonuçları Nutch ile karşılaştırılmış ve önerilen tarayıcının daha iyi performans sergilediği belirtilmiştir [121].

Le Quoc ve arkadaşları, coğrafi olarak dağıtılmış ve harita azaltma tabanlı bir tarayıcı olan UniCrawl'ı sunmuşlardır. UniCrawl, coğrafi olarak dağıtılmış birkaç siteyi düzenler ve her site bağımsız bir tarayıcı çalıştırarak çeşitli teknikler kullanarak Web içeriğini alır ve ayrıştırır. UniCrawl, taranan etki alanını siteler arasında böler ve siteler arası iletişim maliyetini en aza indirirken depolama ve bilgi işlem kaynaklarını birleştirir. Belirli bir bölgeye yayılmış ve yazarlar tarafından belirlenmiş üç Web sitesi üzerinde yapılan deneylerde önerilen tarayıcının ağ tüketimi açısından %93,6'lık bir performans artışı ve 1,75'lik bir hızlanma faktörü gösterdiği belirtilmiştir [121].

Pu çalışmasında, dağıtılmış Web tarayıcılarının etkin bir şekilde kullandığı Hadoop teknolojisine dayalı bir Web tarayıcı sistemi önermiştir. Bu dağıtılmış Web tarayıcısının temel özellikleri yapılandırılabilir, verimli ve ölçeklenebilir olmasıdır. Tohum URL kümesi olarak 20 haber Web sitesi URL'i kullanılmıştır. Çalışmada test ortamının ağ bant genişliği saniyede 2 MB olduğu belirtilmiş ve URL bağlantısının tarama derinliği 2 olarak ayarlanmıştır. Tarayıcı yapılan 3 farklı denemede yaklaşık 11000 Web sayfasını indirmeyi başarmıştır. Önerilen tarayıcının gerçek ortamdaki kapsamının ise tüm Web olduğu belirtilmiştir [122].

Liu ve Jin yaptıkları çalışmalarında, ChainMR Crawler adlı dağıtık bir dikey tarayıcı önermektedir. Önerilen tarayıcı, veri madenciliği tabanlı bir tarayıcı olup URL yönetim modülü, indirme modülü, HTML bölme modülü ve depolama modülünden oluşmaktadır. Her modül URL'i yönetmek için Redis bellek veri tabanını kullanmış ve yinelenen URL'leri filtrelemek için BloomFilter algoritması benimsenmiştir. Nutch ile yapılan karşılaştırmalı taramalar sonucunda ChainMR Crawler'ın verimliliğinin daha iyi olduğu kanıtlanmış ve çeşitli ihtiyaçlara genişletilebilir olduğu belirtilmiştir [123].



Şekil 3.3. Dağıtılmış Web tarayıcılarında farklı özelliklerinin payı

Dağıtılmış Web tarayıcılarında performans ölçütleri; ölçeklenebilirlik, hataya karşı dayanıklılık, dağıtım derecesi, koordinasyon, bölümleme teknikleri, kapsam, örtüşme ve

iletişim yüküdür [124]. Şekil 3.3'de genel olarak dağıtılmış Web tarayıcısında bulunan farklı özelliklerinin payı gösterilmiştir [125]. Dağıtılmış Web tarayıcısının performansı dağıtılmıştık derecesi, tarama süresi, bant genişliği ve kullanılan sistemin özelliklerine göre farklılık göstermektedir.

Çizelge 3.2'de, genel, odaklanmış, artımlı, gizli, mobil ve dağıtılmış Web tarayıcılarının kapsamlı bir karşılaştırması sunulmuş ve bu tarayıcılar çeşitli özelliklere göre analiz edilmiştir. Karşılaştırma, tarayıcıların kapsamları, kullanılan algoritmalar, sayfa seçim yöntemleri, tohum URL kaynakları, örnek tarayıcılar ve ölçeklenebilirlik özelliklerine odaklanarak gerçekleştirilmiştir.

Kapsam açısından genel Web tarayıcıları, tüm Web'i kapsayan bir perspektife sahiptir. Odaklanmış Web tarayıcıları ise belirli bir konu veya alan üzerinde yoğunlaşır. Mobil tarayıcılar belirli Web sayfalarına odaklanırken, gizli ve dağıtılmış Web tarayıcıları genellikle genel veya odaklı tarayıcılar gibi geniş bir kapsamı hedefler. Kapsamı tüm Web olan tarayıcılar genellikle BFS algoritmasını, kapsamı Web'in bir bölümü ya da belirli bir konu olan tarayıcılar ise DFS algoritmasını birincil olarak kullanmaktadır. Genel olarak, Web tarayıcılarında tarama işlemi tohum URL'lerden başlamaktadır. Odaklanmış ve gizli Web tarayıcılarında hedef sayfalar belli olduğundan konu, anahtar kelime ve form doldurma yöntemleri ile tohum URL'leri seçip aramalarını gerçekleştirmektedirler.

Web tarayıcılarının tohum URL kaynakları, türlerine, kapsamlarına ve araştırma alanlarına göre manuel, yarı otomatik veya otomatik olarak seçilebilmektedir. Literatürde yapılan çalışmalarda ve ticari olarak geliştirilen arama motorlarında farklı özellikte Web tarayıcıları geliştirilmiştir. InfoSpiders, OntoCrawler vb. bazı Web tarayıcıları belirli bir türe sahip iken SIMHAR, SmartCrawler vb. bazı tarayıcılar ise birden farklı türde tarayıcı özelliği göstermektedir. Son olarak ölçeklenebilirlik özellikleri açısından, genel, artımlı ve gizli Web tarayıcıları genellikle kapsamlarının önceden bilinmediği için ölçeklenebilirlik özelliklerine sahip değildir. Bunların dışındaki tarayıcılar ise belirli bir kapsam, anahtar kelime, sorgu sonucunda tarandığından ölçeklenebilir özelliklere sahiptir. Bu analizler, Web tarayıcılarının farklı özellikleri ve kullanım senaryoları arasındaki önemli farkları vurgulamakta ve her birinin belirli bir amaç için uygun olduğunu göstermektedir.

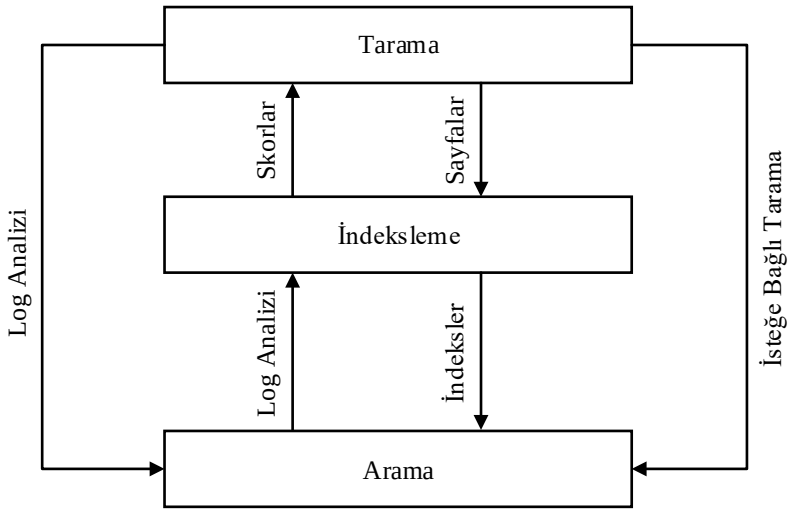
Çizelge 3.2. Web tarayıcılarının karşılaştırılması

Tarayıcı Türü	Kapsam	Kullanılan Algoritmalar	Sayfa Seçimi	Tohum URL Kaynağı	Örnek Tarayıcı	Ölçeklene-bilirlik
Genel Web Tarayıcı	Tüm Web	BFS, Page Rank, HITS	Tohum URL	Page Rank ve HITS değeri en yüksek sayfalar	Google, Mercator, Yahoo, Bing, Yandex vs. Açık Kaynak Tarayıcılar	Hayır
Odaklanmış Web Tarayıcı	Web'in konu ya da alan tabanlı belli bölümü	DFS, Köpek Balığı Arama, Vektör Uzay Modeli, En İyi İlk, En İyi N İlk, Page Rank HITS	Konu veya Anahtar kelime	Kelime veri kümelerinin arama sonuçları DMOZ veri kümesi, Belli konudaki Web sayfaları	InfoSpiders, OntoCrawler, LSCrawler,	Evet
Artımlı Web Tarayıcıları	Tüm Web, Web'in konu ya da alan tabanlı belli bölümü	BFS, DFS, En iyi ilk arama, Page Rank, HITS	Tohum URL, Öncelik sırasına bağlı URL	Page Rank ve HITS değeri en yüksek sayfalar Belirli koleksiyonlar	SmartCrawler RCrawler WebMiner	Hayır
Gizli Web Tarayıcıları	Belirli Web sayfaları	Uyarlanabilir algoritma, Genel-frekans algoritması, rastgele16K, rastgele-1M	Anahtar Kelime sorgusu veya form doldurma	Kelime tabanlı sorgulama, Form tabanlı sorgulama, özellik ve etiket çıkarmaya dayalı sorgulama, DMOZ veri kümesi, arama motorları.	SIMHAR SmartCrawler HiCrawl AKSHR	Hayır
Mobil Web Tarayıcılar	Belirli Web sayfaları	BFS, DFS	Tohum URL	Kelime tabanlı sorgulama		Evet
Dağıtılmış Web Tarayıcıları	Tüm Web, Web'in konu ya da alan tabanlı belli bölümü	BFS	Tohum URL	Tohum URL Kümesi	UbiCrawler SIMHAR Dis-Dyn UniCrawl Nutch	Evet

3.1.2. Önerilen Web tarayıcısı

Arama motorları, temel olarak 3 bölümden oluşmaktadır. Bu bölümler sırasıyla Web sayfalarını tarama, verileri indeksleme ve bu indeks içinde gerçekleştirilen aramayı içermektedir [126]. Web sayfalarını tarama işlemi, Web tarayıcıları (örümcek, tarama botları vb.) tarafından gerçekleştirir. Web tarayıcıları tarama işlemine belirli bir başlangıç noktası olan tohum URL'ler ile başlar. Web tarayıcıları, ziyaret edilen her Web sayfasının içeriğini alır ve bu verileri indeksleme sürecine tabi tutar. Aynı zamanda, sayfa içindeki diğer URL'leri çıkararak öncelikli kuyruğa ekler.

Çizelge 3.4'de döngüsel yapı kullanarak veri tabanını güncel tutan bir tarayıcı mimarisi gösterilmiştir [126]. Bu mimari, sürekli bir döngü içinde çalışarak veri tabanını güncellemeyi sağlar. İlk aşamada, tohum URL'ler kullanılarak başlanır ve bu URL'lerden elde edilen sayfalar taranır. Taranan her sayfanın içeriği alınıp indekslenir ve sayfa içindeki diğer URL'ler öncelikli kuyruğa eklenir. Ardından, öncelikli kuyruktan sırası gelen her URL taranır ve bu süreç tekrarlanır. Bu sayede arama motoru, Web üzerindeki verileri düzenli bir şekilde tarama ve indeksleme yeteneğine sahip olur.



Şekil 3.4. Tarayıcı mimarisi

Tarama işlemi sırasında, belirli politikalara göre URL'ler tekrar taranarak güncellik sağlanır. Bu politikalar, belirli bir sayfanın ne sıklıkta taranacağı, hangi durumlarda tekrar taranacağı gibi faktörlere dayanabilir. Böylece arama motoru, Web'in dinamik yapısına uygun bir

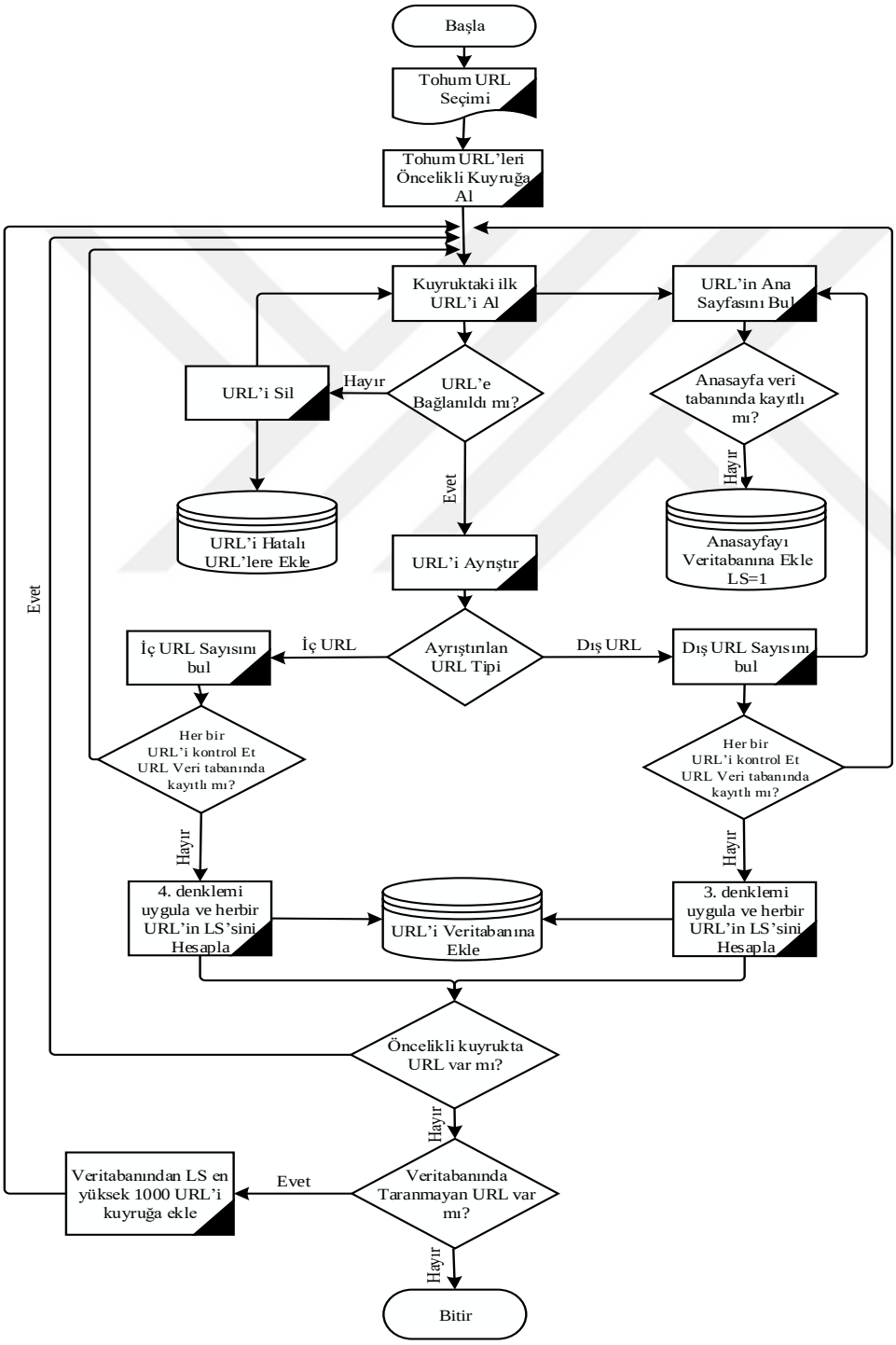
şekilde güncel verilere erişim sağlar. Bu tarama ve indeksleme döngüsü, kullanıcıların arama sorgularına hızlı ve güncel cevaplar verebilmek adına sürekli olarak devam eder.

Web'in tamamının taranması devasa hacmi nedeniyle uzun bir süreçtir ve neredeyse imkânsızdır. Mevcut arama motorları, örneğin Google, Baidu, Yahoo gibi büyük platformlar, tüm Web'in sadece yaklaşık olarak %5'ini tarayabilmektedir [127]. Bu durum, arama motorlarının en önemli kısıtlamalarından biri olan kapsam genişliği problemini ortaya çıkarmaktadır. Kapsam genişliğini artırmak için özellikle ticari arama motorları, çeşitli algoritmalar ve genellikle gizli tutulan özellikler kullanmaktadır. Kapsam genişletme sürecindeki ilk adım genellikle tohum URL'lerin seçimiyle başlar. İkinci adım ise etkili bir tarama algoritması kullanmaktır. Bu adımların her biri, arama motorlarının daha fazla sayfa ve kaynağa erişim sağlaması için önemlidir.

Bu çalışmada önerilen Web tarayıcısı, genel bir tarayıcı olma özelliği taşıdığından kapsamı tüm Web'i içerecek şekilde tasarlanmıştır. Genel tarayıcı, sırası gelen URL'leri ayrıştırarak Web üzerindeki tüm URL'lere ulaşmaya çalışır. Belirli dil ve bölgelere bağlı kalmamak adına, tohum URL kümesi oluşturulurken dünya genelinde en popüler ve en çok ziyaret edilen Web sayfalarından seçimler yapılmıştır. Bu strateji, tarayıcının geniş bir coğrafi ve kavramsal kapsama sahip olmasını hedefler ve bu sayede farklı dil ve kültürlerden gelen İnternet kullanıcılarının çeşitli içeriklere erişimini mümkün kılar. Ayrıca, tohum URL'lerin seçimindeki dikkatli yaklaşım, tarayıcının başlangıçta daha zengin ve çeşitli bir veri kümesiyle çalışmasını sağlar. Bu da tarayıcının daha etkili ve kapsamlı bir şekilde Web'i taramasına olanak tanır.

Önerilen algoritmaya göre tohum URL kümesi öncelikli kuyruğa alınarak her bir URL sırasıyla taranmıştır. Tarayıcı, herhangi bir sebep nedeniyle ulaşılamayan URL'lerle karşılaştığında, ilgili hata kodu, hata mesajı ve hatalı URL bilgilerini içeren veriyi bir veri tabanı tablosuna kaydederek bu URL'leri taranacak URL'ler tablosundan silmektedir. URL'e başarılı bir şekilde bağlantı kurulduğunda iki ana işlem gerçekleştirilmiştir. İlk olarak, URL'in ana sayfası belirlenerek ilgili veri tablosuna kaydedilmiş ve link skoru (LS) en yüksek değer olan 1'e atanmıştır. İkinci olarak, sayfa içerisinde ayrıştırılan URL'ler, alan içi (IntraDomain) ve alan dışı (InterDomain) olmak üzere iki gruba ayrılmış ve bu grupların sayıları belirlenmiştir. Daha sonra, veri tabanından kontrol edilerek, daha önce kaydedilmeyen URL'lerin Eş. 3.5 ve 3.6'ya göre link skorları hesaplanır. Bu süreç sonunda

elde edilen URL'ler, link skorları ve tarama zaman damgası veri tabanına kaydedilmiştir. Bu işlemler, kuyrukta taranacak URL olduğu sürece tekrarlanmıştır. Kuyruktaki son URL tarandıktan sonra, veri tabanındaki en yüksek link skoruna sahip bin URL öncelikli kuyruğa alınarak keşfedilen tüm URL'ler tamamlanana kadar bu işlemler devam etmiştir. Geliştirilen Web tarayıcısının akış şeması Şekil 3.5'te gösterilmiştir.



Şekil 3.5. Geliştirilen Web tarayıcı algoritmasının akış şeması

Literatürdeki çalışmaları iki ana grupta değerlendirebiliriz: birincisi tohum URL seçimi üzerine yapılan çalışmalar, diğeri ise kapsam genişletme metotlarına odaklanan çalışmalardır.

3.1.3. Tohum URL seçimi

Tohum URL seçimi, bir Web tarayıcısının tarama kapsamını ve topladığı veri içeriğini belirleyen temel bir adımdır. Bu başlangıç URL kümesi, tarayıcıların arama süreçlerini başlattığı giriş noktasını oluşturur. Bir Web tarayıcısının en önemli sorunlarından biri en uygun sayfaları elde etmek için hangi URL'lerden başlaması gerektiğidir [128]. Dolayısıyla, tohum URL'lerin dikkatlice seçilmesi, geniş bir tarama kapsamı sağlamak açısından kritik öneme sahiptir. Nispeten küçük bir tohum URL kümesi kullanarak Web'in önemli bir bölümünü keşfetmek mümkündür. Ancak rastgele bir küme seçmek, Web'in önemli bir kısmını keşfedilmemiş bırakabilir [129]. Tohum URL'lerin seçimi, manuel, yarı otomatik ve otomatik olarak farklı yöntemlerle gerçekleştirilebilir. Web'in dinamik yapısı nedeniyle, en iyi tohum URL'lerin zaman içinde değişim göstermesi kaçınılmazdır. Tohum URL seçimi konusunda çok fazla çalışma yapılmamıştır.

Kleinberg [130] Web sayfalarını, merkez ve otorite adını verdiği iki ana grupta toplayan bağlantı dâhil metin araması (Hyperlink-Induced Topic Search - HITS) algoritmasını önermiştir. HIST algoritması yalnızca Web sayfaları arasından köprüleri dikkate almaktadır. Özellikle çok sayıda dış bağlantı sağlayan merkez Web sayfalarının iyi bir tohum URL olduğu belirtilmiştir. Önerilen yöntemde yalnızca Web sayfasının içerdiği köprüler dikkate alınmış ve köprülerin işaret ettiği sayfaların durumu göz ardı edilmiştir. Yapılan çalışmaya göre, tohum URL seçiminde merkez Web sayfalarının seçimi, kapsamın genişlemesini sağlamaktadır.

Daneshpajouh ve arkadaşları [131], tohum kümesi oluşturmak amacıyla yeni ve hızlı bir algoritma sunmuşlardır. Araştırmacılar, farklı topluluklardan gelen tohum URL'lerini tanımlayan ve çıkaran ilk tohum çıkarma algoritmasını önermişlerdir. Çalışmada, HITS algoritmasına dayanan bu algoritma, tohum URL kümesi oluşturmak için O(n) süresinde çalışan bir yöntem sunmaktadır. Algoritma, tohum URL'lerin farklı topluluklardan düğümler içermesini garanti etmek için seçilen tohum URL'ler arasındaki mesafeyi ölçmektedir. Düşük çalışma süresi, sunulan algoritmayı benzersiz kılmıştır. Çalışmada,

PageRank, Trawling, HITS ve ağ akış tabanlı keşif algoritmaları üzerinde çalışılmıştır. Yaptıkları deneylerde, Web tarayıcısı önerilen yöntem ile tanımlanan tohum URL'leri taramaya başlarsa, rastgele tohum kümesi başlatmaktan daha az yinelemede ve farklı topluluklardan daha yüksek PageRank değerine sahip daha fazla sayfa tarandığını göstermişlerdir

Zheng ve arkadaşları [132], tohum URL seçimi için rastgele, en yüksek PageRank değeri ve en çok alan dışı bağlantıya sahip k sayfaya dayalı tohum seçim stratejilerini kullanan grafik tabanlı bir yaklaşım önermişlerdir. Farklı tohum URL'lerin "iyi" ya da "kötü" olarak adlandırılan koleksiyonlar ile sonuçlanabileceğini belirterek, tohum URL seçiminin önemini vurgulamışlardır. Kullanılan tohum seçim stratejilerinin etkinliğini, gerçek Web verileri üzerindeki deneysel sonuçlar ile kanıtlamışlardır. Çalışmada, Web tarayıcısının mevcut tohum URL'leri nasıl kullanabileceğini belirtilmiş, ancak tohum URL'lerin nasıl tanımlanması gerektiği ile ilgili bilgi verilmemiştir. Önerilen yaklaşımın performansını değerlendirmek için her biri en az 100 sayfa içeren ve en az bir harici bağlantıya sahip olan 2000 Web sitesinden rastgele örnekler seçerek tohum URL kümesini oluşturmuşlardır.

Nwala ve arkadaşları [133], Web arşivi koleksiyonları için tohum URL elde etme amacıyla sosyal medya gönderilerini kullanarak kapsamlı bir çalışma gerçekleştirmişlerdir. Bu çerçevede, tohum URL'lerine 10 temel kriter üzerinden (popülerlik, coğrafi konum, konu uzmanlığı, güvenilirlik, itibar vb.) bir kalite puanı atama yöntemini benimsemişlerdir. Çalışmaları kapsamında, referans koleksiyonlarından 1552 ve Twitter mikro koleksiyonlarından 4.209 adet tweet içeriğinden toplamda 2.027 tohum URL elde etmişlerdir.

Tohum URL seçiminde kullanılan temel yöntemler, uzman tarafından manuel seçim, yarı otomatik seçim ve otomatik seçim olmak üzere üç ana kategoride incelenebilir. Genellikle tarayıcılar için tohum URL çıkarma işlemi manuel olarak gerçekleştirilir. Manuel seçimde [134-136], bir veya birkaç konu hakkında yapılan taramalarda konunun uzmanları tarafından tohum URL'ler seçilmekte, önceliklendirilmekte ve tarama kuyruğuna eklenmektedir.

Yarı otomatik seçimde DMOZ ve curlie.org gibi açık kaynak dizinlerden, belirli özelliklere göre tohum URL'ler seçilmektedir. Chan ve Yamana [137], DMOZ üzerinde bulunan URL'leri belirli alan adları (.com, .net, cn, .tw, .jp, .kr) ve dillere göre (Çince, Japonca ve Korece) ayıklandıktan sonra tohum URL olarak kullanmışlardır. Menczer ve Monge [138],

InfoSpider adını verdikleri bir tarayıcı geliştirmişlerdir. InfoSpider, kullanıcı sorgularını genel bir arama motoruna göndermekte ve sonuç olarak dönen URL'leri tohum URL olarak kullanmaktadır.

Otomatik seçimde ise tohum URL'ler, sosyal medya platformları gibi kaynaklardan elde edilir. Örneğin, Twitter gibi sosyal medya kullanıcılarının paylaştığı URL'ler otomatik olarak tohum URL olarak kullanılabilir. Priyatam ve arkadaşları [139], Twitter'da paylaşılan URL'lerin her bir köşe noktası olan bir grafik oluşturmuş ve benzer köşeleri birbirine bağlamışlardır. Taramaya köşe noktalarını oluşturan ve benzersiz olan URL'ler ile başlanmıştır. Sanagavarapu ve arkadaşları [140], Wikipedia ve Twitter'ı kullanarak tohum URL'lerin otomatik olarak çıkarılması için puanlama (SeedRel) ölçütü ve URL'lerin ilgi düzeyini belirlemek için çeşitlilik indeksi kullanan bir yaklaşım önermişlerdir. Buna ek olarak Sanagavarapu ve arkadaşları [141], tohum ve alt URL'lerin tanımlanması ve puanlanması için yapay arı kolonisi (Artificial Bee Colony - ABC) algoritmasını önermişlerdir. Önerilen algoritmayı güvenlik alanına uygulamış ve Wikipedia üzerinden 34.007 tohum URL çıkarmışlardır. Bu çeşitli tohum URL seçim yöntemleri, tarayıcıların başlangıç noktalarını belirleme sürecini optimize etmeye yönelik çeşitli yaklaşımları temsil etmektedir.

Geliştirilen tarayıcıda kullanılan tohum URL'ler Alexa[142] tarafından sağlanmıştır. Alexa bir Amazon şirketidir ve içerik araştırması, rekabet analizi, anahtar kelime araştırması için başvuru kaynağı olarak kullanılmaktadır. Alexa, Web sitelerini, ziyaretçilerin günlük harcadıkları saat, ziyaretçi başına günlük sayfa görüntüleme sayısı, arama trafiği yüzdesi ve toplam bağlantılı site sayısı gibi metrikler temel alarak sıralamaktadır.

Bu çalışmada 102 faklı ülkenin her birinde sıralamaya giren 50 Web sayfası alınmıştır. Sayfa üzerinden veri çekme işlemi için Python programlama dili kütüphanesi olan BeautifulSoup kullanılmıştır. Alexa'dan toplam 5079 Web sitesi alınmış ve kayıt edilmiştir. Google, Youtube, Facebook vb. gibi Web siteleri neredeyse tüm ülkelerde ilk sıralarda yer almakta ve veri tabanında tekrar etmektedir. Bu Web sayfalarından benzersiz olan 2502 Web sayfası tespit edilmiştir. Sadece 1 ülkede listeye giren 1920 Web sayfası tohum URL olarak kullanılmıştır.

3.1.4. Kapsam genişletme

Kapsam, Web tarayıcılarının performansını belirleyen ve kullanıcı isteklerine uygun verileri tarama oranını belirten önemli bir ölçüttür. Arama motorlarının en zorlandıkları konu, istenilen bilgilerin tamamına nasıl ulaşılacağı ile ilgilidir. Arama motorunun, Web tarayıcısı ve tarayıcının kullandığı tarama tekniklerine bağlı olarak kapsamı ve etkinliği artar [77]. Web'in grafik yapısı karmaşıktır ve içerik ile köprülere erişmek için verimli bir algoritma gerekir. Kapsam, öncelikle seçilen tohum URL'lere ve ardından taramanın genişleyebilmesi için URL'lerin öncelik durumuna göre sıralanmasına bağlıdır. Web tarayıcıları için Web sayfalarının önemleri farklıdır ve hangi sayfaların önce taranması gerektiği ile ilgili çeşitli algoritmalar geliştirilmiştir [143]. Web tarayıcıları türüne bağlı olarak, kapsam sadece URL'leri takip ederek genişletilebileceği gibi, bağlantı metinleri analizi veya sayfadaki verilerin analizi gibi yöntemlerle de genişleyebilir. Kapsam, tarayıcı çeşidine göre farklılık gösterir. Genel tarayıcılarda kapsam tüm Web iken, odaklı tarayıcılarda belli bir konu ya da belirli bir etki alanı olabilir. Bunun dışında artırılmış, gizli, mobil ve dağıtılmış Web tarayıcıları, genel ve odaklanmış tarayıcıların özelliklerini taşırlar.

Page ve arkadaşları [144], Google arama motorunun temel algoritmalarından biri olan PageRank algoritmasını önermişlerdir. Hangi Web sayfalarının öncelikli taranması gerektiğini belirlemek için farklı ölçütler kullandılar. Bu ölçütler geniş öncelikli, geri bağlantı sayısı ve PageRank olup tarayıcıyı önemli sayfalara yönlendirme açısından bu üç ölçüt karşılaştırılmıştır. Sonuç olarak PageRank'in diğerlerine göre önemli sayfaları daha erken taradığı görülmüştür.

Prakash ve Kumar [145], çalışmalarında PageRank ve köpekbalığı arama (Shark-Search) algoritmasının birleştirerek PageRank algoritmasını kullanan köpekbalığı algoritmasını önermişlerdir. Yapılan ön deneylerde orijinal sayfa sıralaması algoritmasına göre önemli gelişmelerin gözlendiği belirtilmiştir.

Cao ve arkadaşları [146], RankCompede adını verdikleri yeni bir model önermişlerdir. Bu model, aynı ağda rastgele yürüyüşe izin veren ve rekabet kavramlarını birleştirerek kümeleme ve sıralama işlemlerini aynı anda gerçekleştiren bir yöntem sunmaktadır. Geleneksel grafik kümeleme yaklaşımları ile karşılaştırıldığında, önerilen yöntemin ağ düğümlerini gruplamada daha hızlı ve sezgisel olduğu belirtilmiştir.

Najork ve Wiener [147], önerdikleri yöntemde, 328 milyon benzersiz sayfa içeren bir tarama sırasında, taranan sayfaların zaman içerisindeki ortalama sayfa kalitesini incelemişlerdir. Taramaya başlandığında yüksek kalitedeki Web sayfalarını seçme eğiliminde olan BFS yöntemini seçmişlerdir. Sayfaların kalitesini ölçmek ve sıralamak için PageRank algoritmasını kullanmışlardır.

Nisreen ve Elsheh [143] çalışmalarında, Web sayfalarının benzerliği ve dinamikliği kullanılarak Web sayfalarına öncelik veren bir yöntem önermişlerdir. Çalışmada dinamik ve statik URL'ler için ayrı iki öncelikli kuyruk kullanılmıştır. Dinamik sayfalar yüksek önceliğe sahip olduğundan dolayı öncelikli taranmaktadır. Elde edilen bulgulara göre Web sayfalarının dinamikliğini kullanmanın, URL'lerin taranma sırasının belirlenmesinde etkili bir yol olduğu belirtilmiştir.

Gupta ve Singh [148] tarafından yapılan çalışmada, kullanıcı tercihi tabanlı sayfa sıralaması adı verilen yeni bir sayfa sıralama algoritması önerilmiştir. Önerilen algoritmanın sayfa içerik ilgi düzeyini belirlemek için aracılar kullandığı ve sıralamada kullanıcı davranışlarının da dikkate alındığı vurgulanmıştır. Yine kullanıcı davranışı ile ilgili olarak Alhaidari ve arkadaşları [149], kullanıcı davranışı ve tercihine odaklanan sayfa sıralama algoritmaları, PageRank, Ağırlıklı PageRank ve HITS algoritmalarını tartışmışlardır. Bu algoritmaların birleşiminden oluşan kullanıcı tercihi tabanlı ağırlıklı sayfa sıralama algoritması (Preference Based Weighted Page Ranking Algorithm - UPWPR) önerilmiştir. UPWPR algoritması Web içerik madenciliği ve Web kullanım madenciliğini kullanarak kullanıcı tercihlerine göre arama sonuçlarını sıralamaktadır.

Baker ve Akcayol [150], URL'leri alan içi ve alan dışı olarak sınıflandıran ve bir öncelik sırası kullanan tarayıcı algoritmasını sundular. Önerilen algoritma alan dışı bağlantılar için 2/3 ve alan içi bağlantılar için 1/3 değerin verilerek alan dışı bağlantılara öncelik verilmektedir. Bu, kapsamı genişletmek ve aynı etki alanı içindeki bağlantı döngülerinden kaçınmak amacını taşımaktadır. Ayrıca, öncelikli kuyruğa eklenen URL'ler için bir zamanlama mekanizması kullanılmakta ve belirli bir süre boyunca taranmayan URL'ler kuyruktan çıkartılmaktadır.

Tüm Web'in taranması olası olmadığından, temel amacımız, hem lokasyon olarak hem de taranmamış Web sayfalarına öncelik vererek kapsamı genişletmektir. Kapsamı genişletmek için tohum URL seçimi ile birlikte tarama yöntemi hayati derecede önemlidir. Bir Web sitesinin ana omurgasının ve zengin URL içeriğinin bulunduğu sayfa ana sayfa (domain adresleri) olduğu gözlemlenmiştir. Web sayfası içerisindeki URL ağaç yapısının en üstünde ana sayfa bulunmaktadır. Bu nedenle, kapsamı genişletmek için öncelikli olarak ana sayfaların taranması gerekmektedir. Bir URL kuyruktan alındığında ve URL'e bağlantı sağlandığında, sayfadaki her bir InterDomain için ayrıştırma işlemi gerçekleştirilerek URL'in ana sayfasının veri tabanında kayıtlı olup olmadığı sorgulanır. Eğer ana sayfa veri tabanında kayıtlı değilse, URL'e öncelikli bir taranma için link skoru en büyük değer olan 1 değeri atanır. Diğer URL'ler için ise link skoru hesaplanacaktır. Baker ve Akcayol'un [150] geliştirdikleri öncelikli kuyruk yapısında InterDomain linkin skoru (LS_{max}) ve IntraDomain linkin skorunun (LS_{min}) hesaplanması sırasıyla Eş. 3.3 ve Eş. 3.4'deki gibidir.

$$LS_{max} = \frac{\sum(\propto_{inter}+\propto_{intra}) - \beta_{intra}}{\sum(\propto_{inter}+\propto_{intra})} * 0.66 \tag{3.3}$$

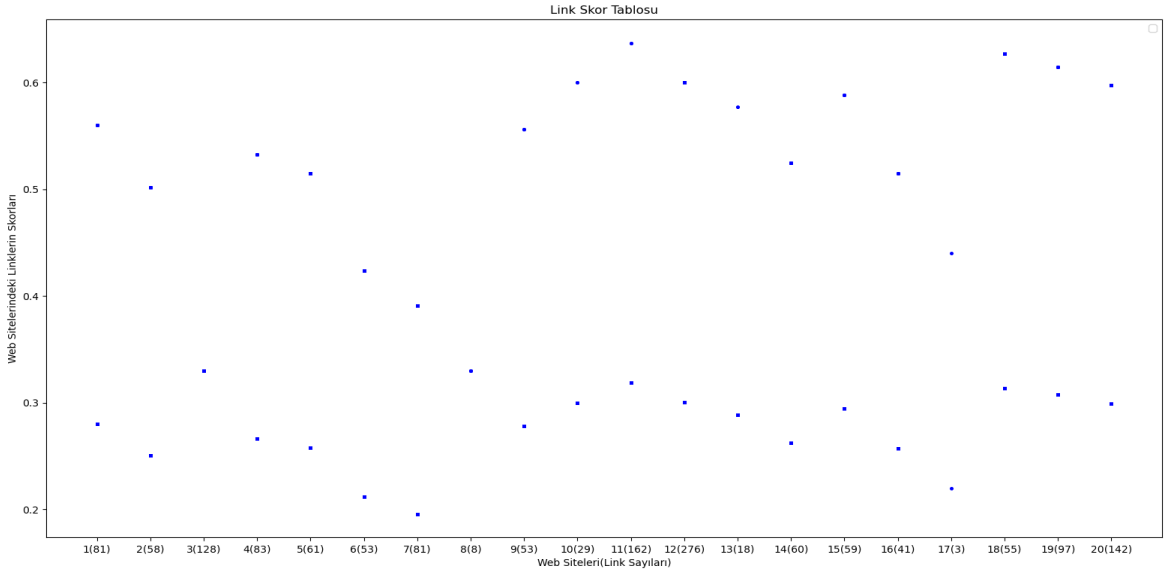
$$LS_{min} = \frac{\sum(\propto_{inter}+\propto_{intra}) - \beta_{intra}}{\sum(\propto_{inter}+\propto_{intra})} * 0.33 \tag{3.4}$$

Eşitliklerde kullanılan $\propto_{inter}$ InterDomain linklerin toplamını, $\propto_{intra}$ IntraDomain linklerin toplamını ve β_{intra} InterDomain ve IntraDomain arasındaki minimum değeri göstermektedir. Bu hesaplama ile bir Web sayfasındaki tüm IntraDomainler ile InterDomainler kendi içerisinde aynı değerleri alır. Bunun sonucunda öncelikli taramada artarda tarandıkları için iki temel sorun ile karşılaşılmaktadır. İlk sorun nezaket politikasına aykırı olarak bant genişliği sık kullanmış olur. İkinci sorun ise LS aynı olduğundan kapsam hızlı bir şekilde genişlememektedir. Tohum URL'ler arasından InterDomain ve IntraDomain sayıları toplamı 200 den az olan rastgele seçilmiş 20 Web sitesinin Eş. 3.3 ve 3.4'e göre dağılımı Şekil 3.6'de gösterilmiştir.

Bu sorunları çözmek için hesaplanan değeri sabit bir sayı ile çarpmak yerine her bir URL için rastgele bir sayı ile çarpmanın kapsamı ve dallanmayı arttırdığını söyleyebiliriz. Bunun için geliştirilen eşitliklerimiz sırasıyla Eş 3.5 ve 3.6'deki gibidir

$$LS_{max} = \frac{\sum(\propto_{inter}+\propto_{intra}) - \beta_{intra}}{\sum(\propto_{inter}+\propto_{intra})} * rand(x_{min}, x_{max}) \quad (3.5)$$

$$LS_{min} = \frac{\sum(\propto_{inter}+\propto_{intra}) - \beta_{intra}}{\sum(\propto_{inter}+\propto_{intra})} * rand(y_{min}, y_{max}) \quad (3.6)$$



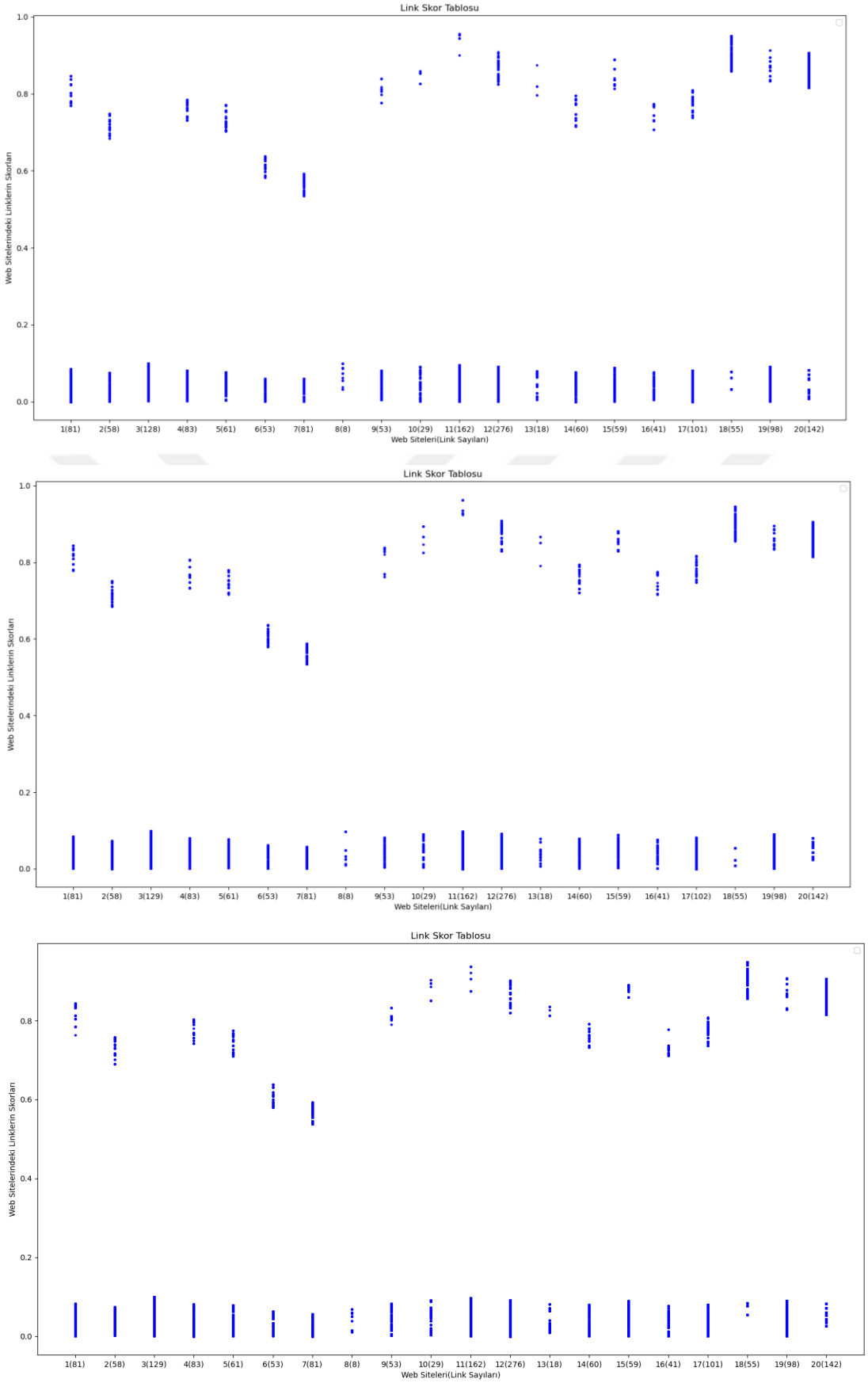
Şekil 3.6. Baker ve Akcayol'a göre LS dağılımı

Eş. 3.5 ve 3.6'da görüldüğü üzere LS hesaplanırken, eşitlik sabit bir sayı yerine LS_{max} x_{min}, x_{max} aralığındaki rastgele bir sayı ile LS_{min}'de y_{min}, y_{max} aralığındaki rastgele bir sayı ile çarpılmıştır. En uygun x_{min}, x_{max} ve y_{min}, y_{max} değerlerini tespit etmek için Çizelge 3.3'deki 5 farklı değer kullanılarak link skorları ayrı ayrı hesaplanmıştır.

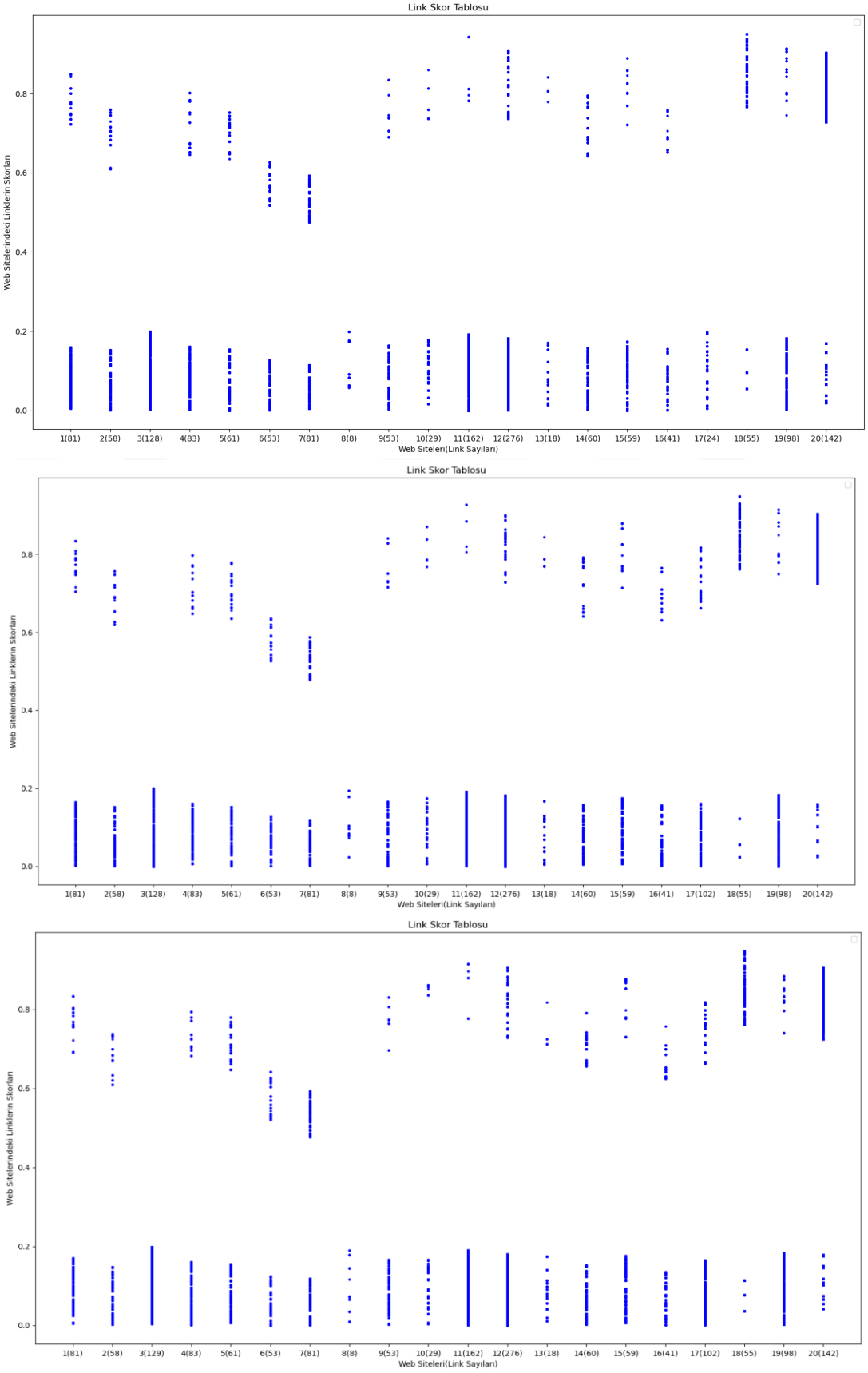
Çizelge 3.3. Kullanılan x_{min}, x_{max} ve y_{min}, y_{max}değerleri

Aralık Kümeleri	x_{min}	x_{max}	y_{min}	y_{max}
1	0	0.1	0.9	1
2	0	0.2	0.8	1
3	0	0.3	0.7	1
4	0	0.4	0.6	1
5	0	0.5	0.5	1

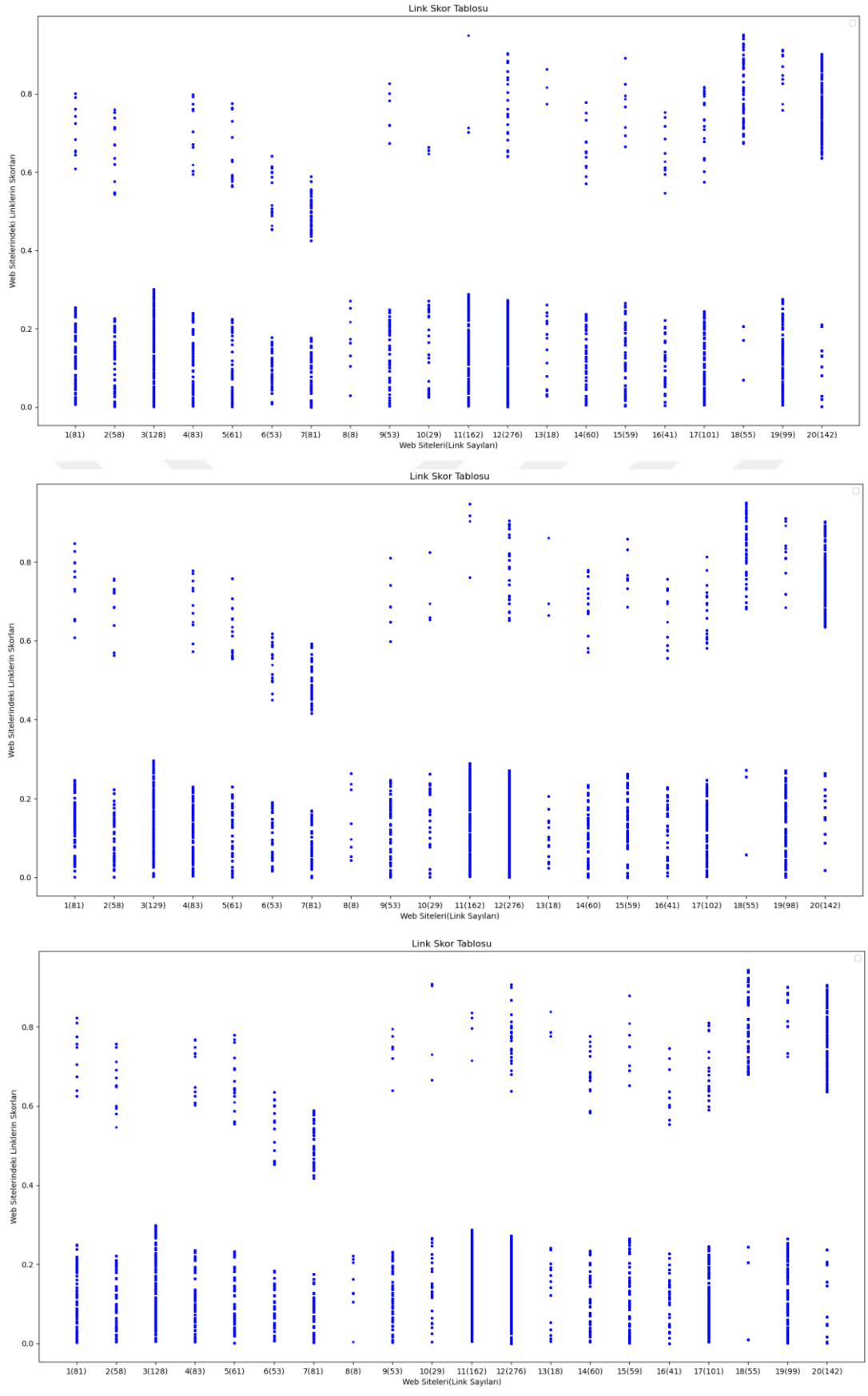
Şekil 3.6'da kullanılan 20 Web sitesi üzerinde Çizelge 3.4'deki 5 faklı değer aralık kümesi kullanılarak LS_{max} ve LS_{min} dağılımı her bir aralık kümesi için 3 defa test edilmiştir. Hesaplanan LS dağılımları Şekil 3.7, 3.8, 3.9, 3.10 ve 3.11'de gösterilmiştir.



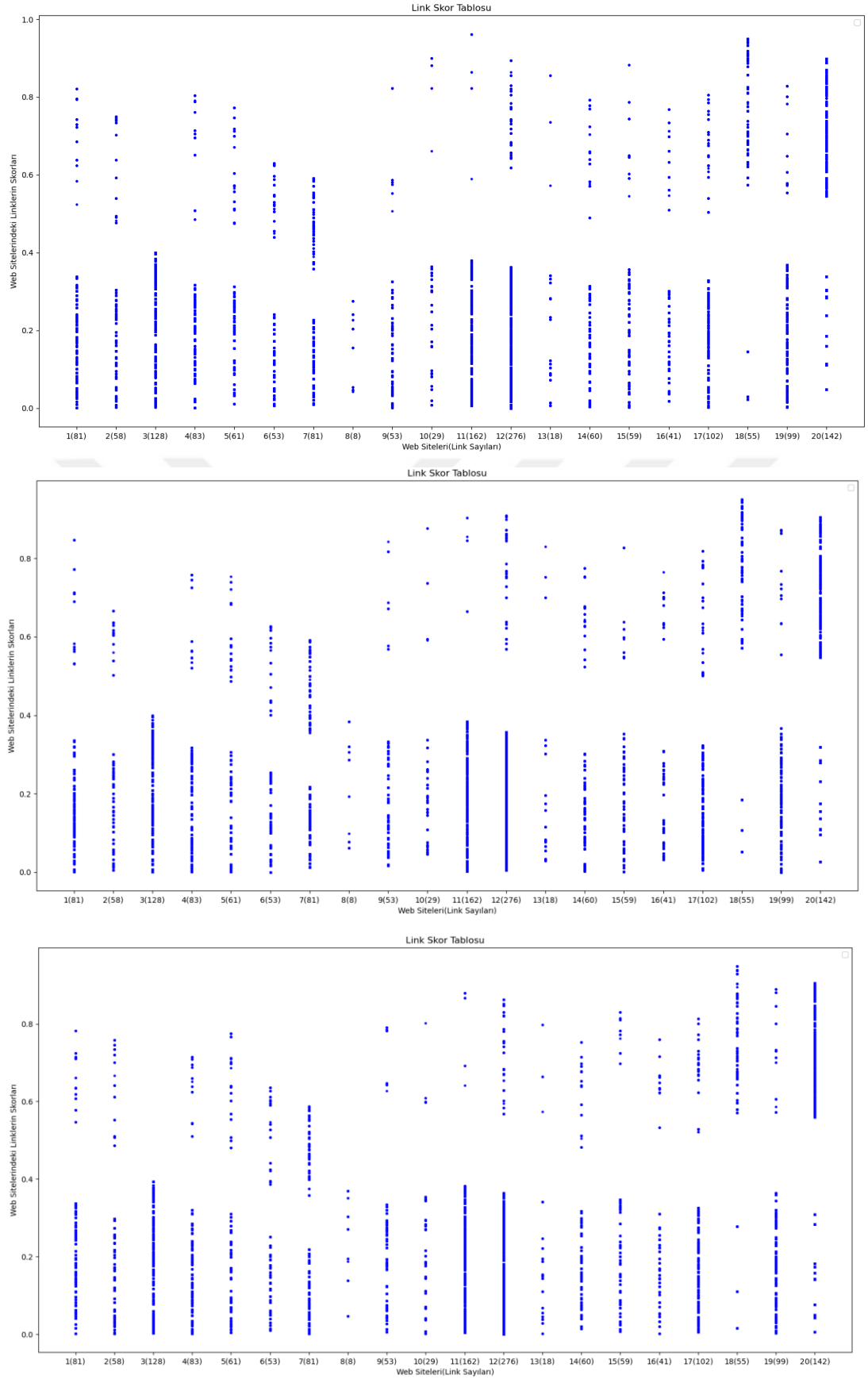
Şekil 3.7.(0,0.1-0.9,1) aralığı için LS dağılımı.



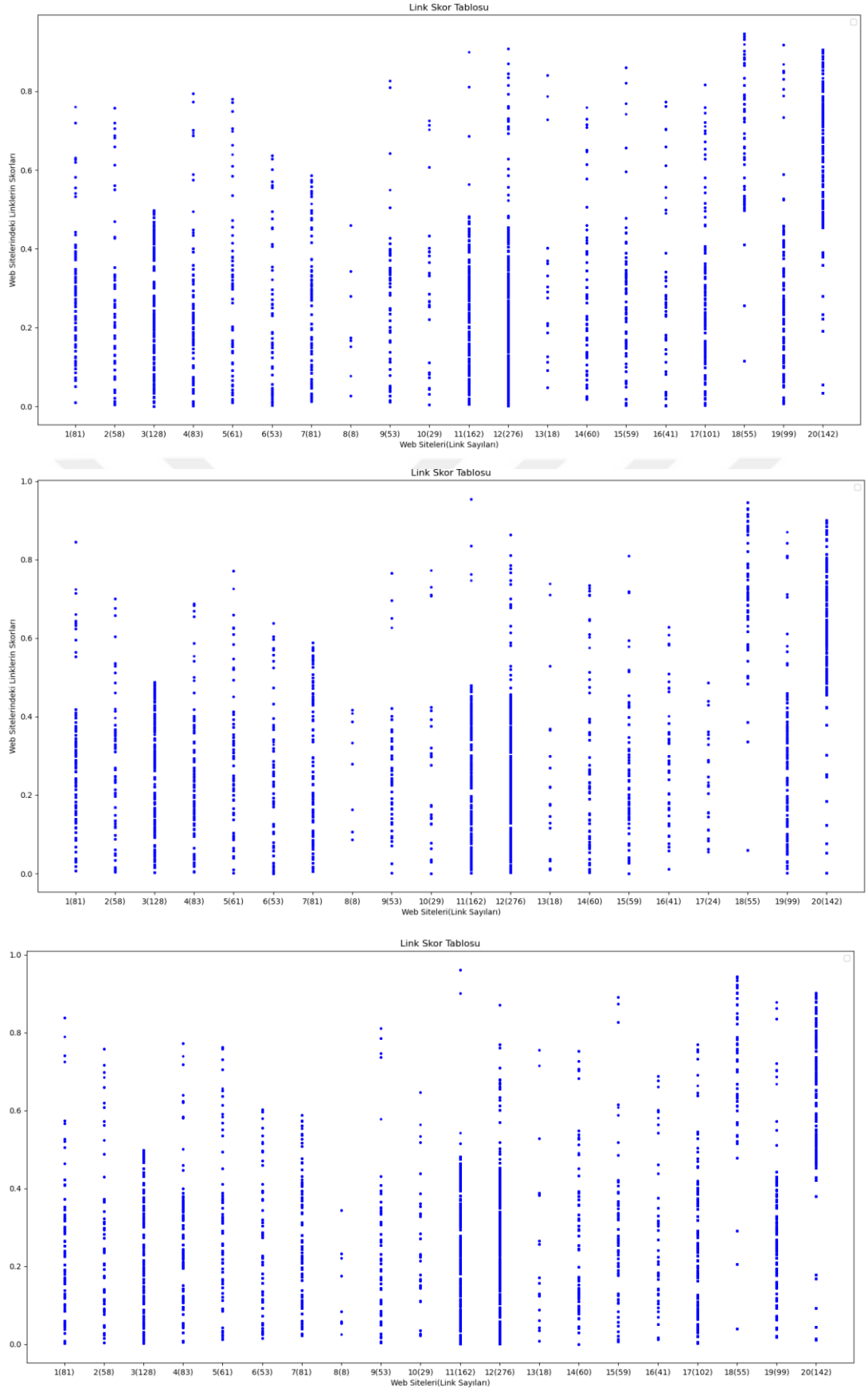
Şekil 3.8.(0,0.2-0.8,1) aralığı için LS dağılımı



Şekil 3.9.(0,0.3-0.7,1) aralığı için LS dağılımı



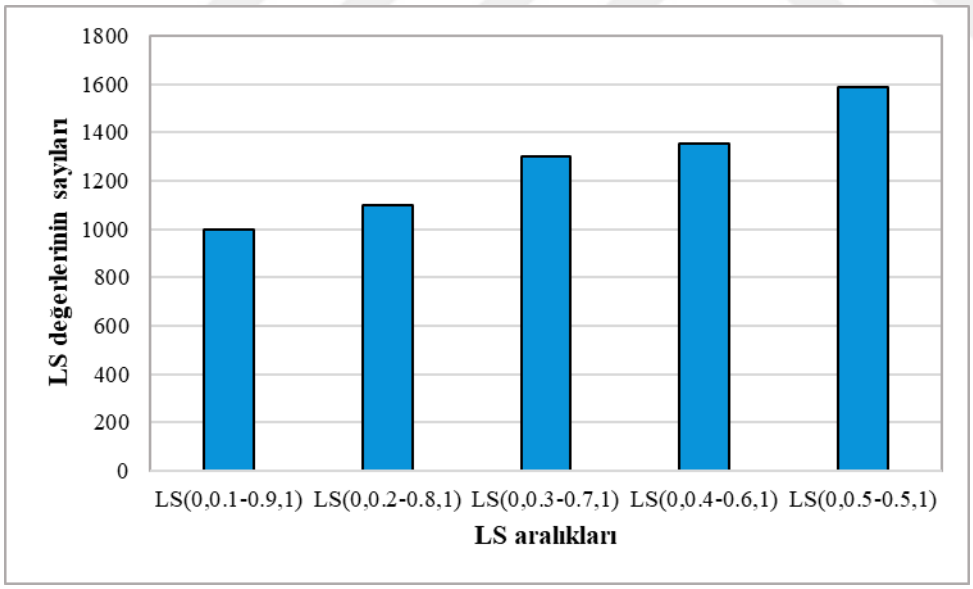
Şekil 3.10. (0,0.4-0.6,1) aralığı için LS dağılımı



Şekil 3.11. (0,0.5-0.5,1) aralığı için LS dağılımı

Şekil 3.7, 3.8, 3.9, 3.10 ve 3.11'de görüldüğü gibi LS değeri IntraDomainler ile InterDomainler URL'ler için hesaplandığında, sadece 2 farklı değerle sınırlı kalmamakta, aksine 0 ile 1 arasında homojen bir dağılıma sahip olmaktadır. Bu durum, sistem tarafından test edilen 5 farklı aralıkta incelendiğinde, aralık değeri arttıkça farklı LS sayıları buna paralel olarak arttığını göstermektedir. 20 Web sayfasından alınan 1648 farklı URL'in ayrı ayrı link skorları hesaplanmıştır. Alınan 5 farklı aralıkta URL'lerin link skorlarının benzersiz olanlarının sayısı Şekil 3.12'de gösterilmiştir.

Şekil 3.12'deki grafik incelendiğinde, LS hesaplanırken aralık değeri arttıkça URL'lerin LS değerlerinin dağılımının homojen bir yapı sergilediği gözlemlenmektedir. LS_{min} hesaplanırken aralık 0 ile 0.5, LS_{max} hesaplanırken de aralığın 0.5 ile 1 arasında alınması en uygun sonucu vermektedir. LS'ye göre öncelikli kuyruk oluşturulduğunda aynı domainde tarama olasılığı azalırken farklı domainlerde artmaktadır. Bunun sonucunda hem farklı sayfalarda işlem yapıp bant genişliğini ihlal etmeyecek hem de hızlı bir şekilde keşfedilmemiş yeni Web sayfalarını keşfedecektir.



Şekil 3.12. LS aralıklarına göre URL'lerin benzersiz LS dağılımları.

Kapsam genişletme amacıyla gerçekleştirilen deneysel çalışmada, 3 farklı tohum URL kümesi kullanılmıştır. Bunlardan birinci tohum URL kümesi farklı ülkeler ile kesişimi olmayan tüm URL'lerdir. Özellikle popüler ve yaygın olarak bilinen Web siteleri (Google, Twitter, Instagram vb.) diğer çoğu URL'lerle sıkça bağlantılı olduğu için, bu popüler siteler

tohum kümesine dâhil edilmemiştir. İkincisi her bir ülkede toplam bağlı site sayısı en fazla olan URL'lerdir. Seçilen URL'ler, HITS algoritmasına göre merkezi sayılan Web sitelerini yansıtmak üzere seçilmiştir. Son olarak üçüncüsü her bir ülkede InterDomain sayısı en fazla olan 5 URL kümesidir. URL'ler, diğer Web sayfalarına bağlantı URL sayısına göre sıralanmış ve her bir ülkede en fazla bağlantıya sahip olan 5 URL seçilmiştir.

Çizelge 3.4'de, üç farklı tohum URL kümesi kullanılarak, 3 saatlik zaman periyoduna göre yapılan 3 farklı taramada elde edilen sonuçlar ayrıntılı bir şekilde listelenmiştir. Çizelge 3.5'de de yine üç farklı tohum URL kümesi kullanılarak, 3 saatlik zaman periyoduna göre yapılan 3 farklı taramada elde edilen hatalı URL sayıları verilmiştir. Deneyler, Intel Xeon CPU E5-2650 2.00 GHz işlemci, 8 GB RAM ve Windows Server 2019 işletim sistemine sahip bir sunucu bilgisayar üzerinde yapılmıştır. Geliştirilen tarayıcı, Python programlama dili ve ilgili kütüphaneler kullanılarak tasarlanmış ve MySQL veri tabanı üzerinde uygulanmıştır.

Çizelge 3.4. Üç farklı tohum URL kümesinde üç saatlik zaman periyoduna göre yapılan tarama sonuçları

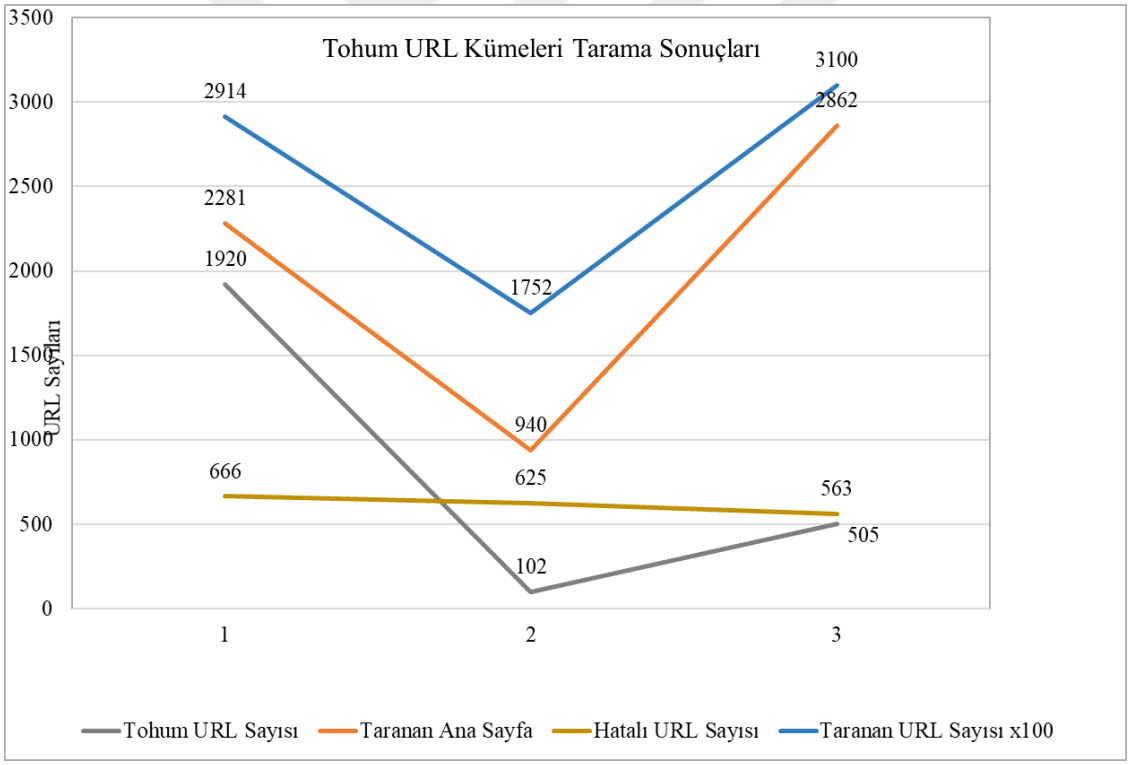
Tohum URL Kümesi	Tohum URL Sayısı	Taranan URL Sayısı				Taranan Ana Sayfa			
		1. Tarama	2. Tarama	3. Tarama	Ortalama	1. Tarama	2. Tarama	3. Tarama	Ortalama
1	1920	298655	285647	290087	291463	2301	2245	2297	2281
2	102	176681	180934	168175	175263,3	940	955	925	940
3	505	311654	312586	306057	310099	2883	2901	2802	2862

Çizelge 3.5. Üç farklı tohum URL kümesinde üç saatlik zaman periyoduna göre yapılan taramada elde edilen hatalı URL sayıları

Tohum URL Kümesi	Tohum URL Sayısı	Hatalı URL Sayısı				Hatalı URL Oranı
		1. Tarama	2. Tarama	3. Tarama	Ortalama	Ortalama
1	1920	674	686	638	666	0,228502
2	102	616	618	641	625	0,356606
3	505	548	597	544	563	0,181555

Taramalar sonucunda elde edilen sayısal verilerden tohum URL sayıları, taranan ana sayfa sayıları ile hatalı URL sayıları sol eksende ve taranan URL sayıları sağ eksende olmak üzere Şekil 3.13'de gösterilmiştir.

Tarama sonucundan da görüldüğü gibi tohum URL sayısının taranan URL, ana sayfa ve hatalı URL sayısı üzerinde etkisi yoktur. Performansı etkileyen en önemli gösterge kullanılan tohum URL kümesinin kalitesidir. Tarama sonuçlarına göre en çok tohum URL kullanan (1) ve en az tohum URL kullanan (2) numaralı tohum URL kümelerine göre (3) numaralı tohum URL kümesinin tüm sonuçlarda daha iyi performans gösterdiği görülmüştür. Özellikle 3 numaralı tohum URL kümesi ile tarama başlatıldığında diğer kümelere göre hem hatalı sayfa sayısı ve oranı daha düşük, hem de taranan ana sayfa ve toplam URL sayıları daha fazla çıkmıştır.



Şekil 3.13. Üç farklı tohum URL kümesinde üç saatlik zaman periyoduna göre yapılan tarama sonuçları

Şekil 3.13'te görüldüğü gibi, (2) numaralı tohum URL kümesi en düşük performansı sergilemektedir. Bu durumun başlıca nedeni, tohum URL kümesinin az sayıda URL içermesidir. Bir değer etken ise toplam bağlı site sayısının yüksek olmasıdır. Bu durum, [10] tarafından tanımlanan otorite Web sayfaları kategorisinde bulunan bilgilendirici içeriklere

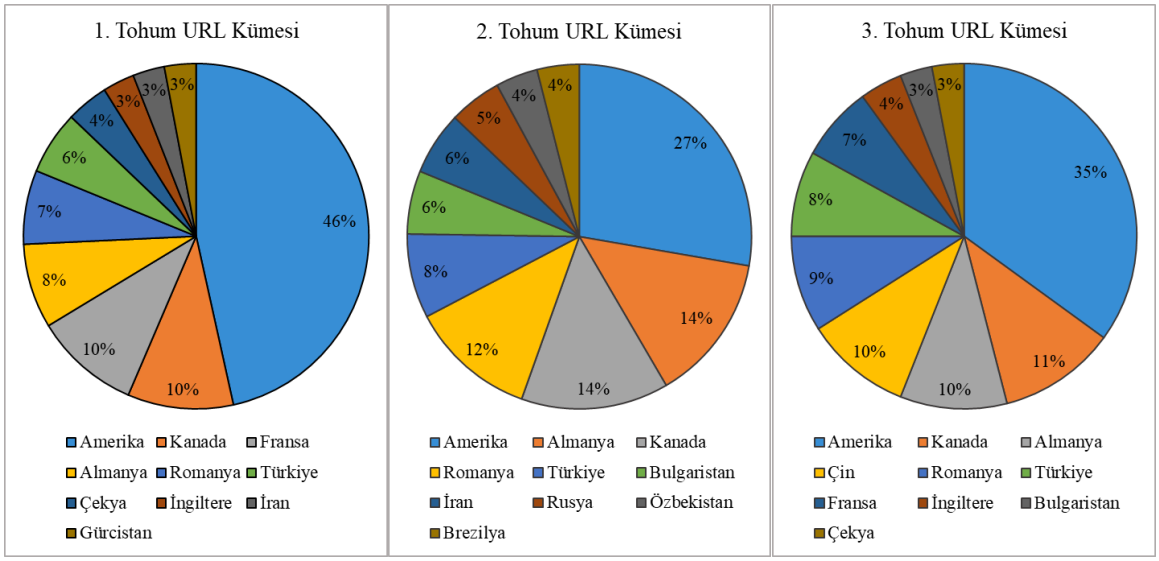
sahip sitelerin, InterDomain sayısının az olması nedeniyle kapsamın hızlı bir şekilde genişlememesine sebep olmaktadır. (1) ve (3) numaralı tohum URL kümeleri arasında benzer performans gözlendiği dikkat çekmektedir. (1) numaralı kümenin, (3) numaralı kümeden yaklaşık 4 kat daha fazla tohum URL içerdiği görülmektedir. Ancak (1) numaralı kümenin (3) numaralı kümeden daha kötü performans gösterdiği görülmektedir. Bu durumun temel nedeni, (1) numaralı kümenin içerdiği URL'lerin hem merkez hem otorite sayfalarından oluşması, (3) numaralı kümenin ise ağırlıklı olarak merkez sayfalardan oluşmasından kaynaklanmaktadır. Bu durum, tohum URL kümesinin bileşenlerinin tipinin, kapsam genişletme performansını etkileyebileceğini göstermektedir.

Kapsam genişliği bakımından bir diğer performans ölçütü taranan URL'lerin bulundukları ülkeye göre dağılımlarıdır. Çizelge 3.5'da üç farklı tohum URL kümesi ile yapılan taramada elde edilen ana sayfaların sayısının ülkelere göre dağılımları gösterilmektedir.

Çizelge 3.6. 3 farklı tohum URL kümesi ile yapılan taramada elde edilen ana sayfaların sayısının ülkelere göre dağılımları

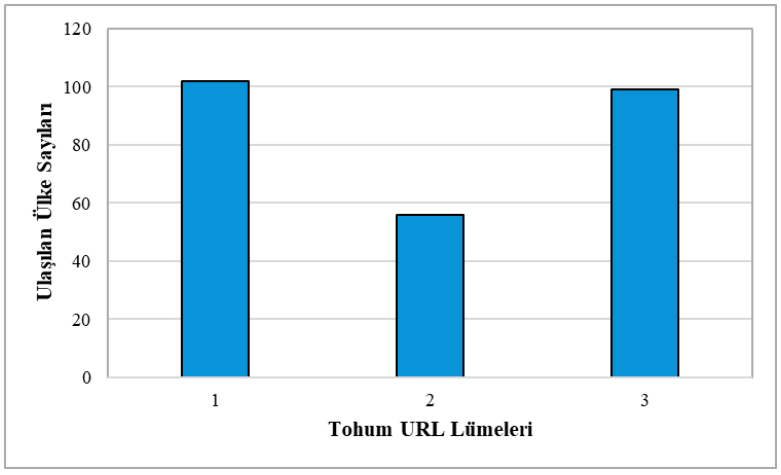
1.Tohum URL Kümesi		2.Tohum URL Kümesi		3.Tohum URL Kümesi	
Web Sayfası Sayısı	Ülke	Web Sayfası Sayısı	Ülke	Web Sayfası Sayısı	Ülke
776	Amerika	180	Amerika	651	Amerika
161	Kanada	90	Almanya	207	Kanada
158	Fransa	86	Kanada	186	Almanya
130	Almanya	74	Romanya	175	Çin
109	Romanya	54	Türkiye	159	Romanya
100	Türkiye	35	Bulgaristan	142	Türkiye
61	Çekya	35	İran	137	Fransa
54	İngiltere	31	Rusya	79	İngiltere
46	İran	27	Özbekistan	64	Bulgaristan
45	Gürcistan	24	Brezilya	60	Çekya
626	Diğer	300	Diğer	886	Diğer

Taranan ana sayfaların ülkelere göre dağılımları incelendiğinde 2 ve 3 numaralı tohum URL kümelerinin 1 numaralı tohum URL kümesine göre daha homojen dağıldığı görülmüştür. Toplam taranan URL sayısı göz önünde bulundurulduğunda 3 numaralı tohum URL kümesinin 2 numaralı tohum URL kümesine göre yüzdesel olarak daha başarılı sonuç verdiği tespit edilmiştir. Şekil 3.14'de bu dağılımların yüzdesel olarak grafiği gösterilmiştir.



Şekil 3.14. Ana sayfaların sayısının ilk 10 ülkeye göre dağılımları

Şekil 3.15'de görüldüğü gibi, 3 tohum URL kümesi ile yapılan taramalar sonucunda ulaşılan ülke sayıları sunulmuştur. Grafik üzerinde özellikle dikkat çeken durum, 2 numaralı tohum URL kümesinin daha az ülkeye ulaşmış olmasıdır. Bu durumun ana sebebi, her bir ülkeden sadece bir URL'nin seçilmesidir. Ayrıca, seçilen bu URL'in başka bir ülkeye ait ana bilgisayarlar üzerinde barındırılmasından kaynaklanmaktadır. Öte yandan, 1 ve 3 numaralı tohum URL kümeleriyle gerçekleştirilen taramalarda ise neredeyse tüm ülkelerin tarandığı gözlemlenmiştir. Böylece, bölgesel olarak taramanın dağılımının, kapsamın genişlemesi ve performansın artırılması üzerinde olumlu bir etkisi olduğu söylenebilir. Bu bulgu, tohum URL kümesinin içeriğinin ve dağılımının, taranan coğrafi bölgelerin kapsam genişletme stratejilerini etkileyebileceğini göstermektedir.



Şekil 3.15. Üç farklı tohum URL kümesi ile yapılan taramalar sonucunda ulaşılan ülke sayıları

Çizelge 3.4'te belirtildiği üzere, her bir ülkede IntraDomain sayısı en fazla olan 5 Web sitesi, toplamda 505 Web sitesini içeren bir tohum URL kümesi oluşturulmuştur. Bu tohum URL kümesi kullanıldığında, hem keşfedilen URL sayısı hem de taranan ana sayfa sayısı en yüksek seviyededir. Ayrıca, oransal olarak hatalı sayfa sayısı da diğer tohum URL kümelerine kıyasla daha düşüktür. Çizelge 3.6'da, bu hataların ve bu hataların tespit edildiği URL sayıları detaylı bir şekilde gösterilmiştir.

Tarama işleminin yapıldığı sunucu bilgisayar üniversite içerisinde barındırıldığından ve bazı Web sitelerine (illegal, bahis vb.) giriş izni bulunmadığından dolayı 403 hata kodlu bağlantı sorununun yüksek olduğu tahmin edilmektedir. İkinci en yüksek sayıda verilen hatanın sebebi de Web sitelerinin otomatik tarama yapan tarayıcıları engellemesi ve bu engelin tarayıcı tarafından aşılamamasından kaynaklanmaktadır. Diğer hataların oranlarının ise taranan toplam URL sayısına göre oransal olarak makul seviyede olduğu görülmüştür.

Çizelge 3.7. Üç tohum URL kümesi ile yapılan taramada elde edilen hatalar ve sayıları

Hata Kodu	Hata	Hatalı URL Sayıları		
		1.Tohum URL Kümesi	2.Tohum URL Kümesi	3.Tohum URL Kümesi
403	Client Error: Forbidden for url	213	184	156
11001	Getaddrinfo failed	76	66	36
10060	Connection Timed Out Error	77	63	74
	SSL: certıfıcate_verıfy_faıled	49	72	31
	Timeout=30	35	41	23
406	Client Error: Not Acceptable for url	44	48	24
	Max retries	36	1	10
10054	ConnectionResetError	25	19	36
404	Client Error: Not Found for url	20	12	39
503	Server Error:Service Unavailable for url	22	8	22
500	Server Error:Internal Server Error for url	18	23	20
405	Client Error: Not Allowed for url	2	19	41
	Diğerleri	49	69	51
	TOPLAM	**666**	**625**	**563**

3.2. Heterojen Veri Kaynaklarından Toplanan Verilerin Entegrasyonu

Geliştirilen tarayıcı mimarisi, Web sayfalarından alınan verileri indeksleyerek bir veri tabanında saklamaktadır. Bu Web sayfalarının güncellenip güncellenmediğinin kontrolünün yapılabilmesi için indekslenmiş kelimelerden sorgu yapmak, hem karmaşık sorgular gerektirir hem de performansı düşürmektedir. Bundan dolayı Web sayfalarının barındırdığı verilerin tamamının XML dosyalarında depolanması, güncelleme kontrolü için kolaylık sağlayacaktır. Geliştirilen tarayıcı mimarisinin gereği olarak, bir URL'i taramaya başladığında öncelikle ana sayfasını tarayıp kaydetmekte ve daha sonra URL taranmaktadır. Tarayıcı verilerin depolanması için ilk olarak taranan bir ana sayfa ile karşılaştığında ana sayfanın ID'si XML dosyasının adı olacak şekilde yeni bir XML dosyası oluşturmaktadır. Eğer taranan URL'in ana sayfası daha önce taranmış ve XML dosyası oluşturulmuş ise yeni taranan veriler bu dosya içerisinde <URL> etiketi içerisine yeni bir kayıt olarak eklenmektedir. XML dosyasının ana etiketi (<root>) altında, ID değeri taranan URL olan <URL> etiketi bulunmaktadır. <URL> etiketi altında ise Web sayfasının başlık (<title>) verisinin bulunduğu <baslik> ve gövde (<body>) metninin bulunduğu <icerik> etiketi bulunmaktadır.

XML belgeleri belge nesne modeli (Document Object Model - DOM) yapısında oluşturulmuştur ve <URL> etiketlerinde bulunan "id" değeri kullanılarak güncelleme ve silme işlemleri yapılabilmektedir. Sayfa güncellenmiş ise <baslık> ve <etiket> verileri de güncellenmektedir. Sayfaya ulaşılamazsa ve veriler alınamazsa URL'in ID değeri veri tabanından alınarak ilgili belgenin <URL> etiketinin "id" değeri ile eşleşen etiket silinmektedir. Şekil 3.16'da ID değeri 1 olan ana sayfanın ve bağlı URL'lerdeki verilerin bulunduğu 1.xml adına sahip dosyanın genel yapısı gösterilmiştir.

1.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<root>
    <URL id="503">
        <baslik>Yeni dijital sahtekârlar türemiş, uyanık olun!</baslik>
        <icerik> Biraz önce 850'li hatlardan arayan biri Superonline'daki abonelik
        süremin sona erdiğini ve sözleşmemi yenilemem gerektiğini söyledi...
        </icerik>
    </URL>
    <URL id="504">
        <baslik>Bakan Kasapoğlu'ndan milli boksörlere tebrik</baslik>
        <icerik>Gençlik ve Spor Bakanı Mehmet Muharrem Kasapoğlu, Hırvatistan'da
        düzenlenen 22 Yaş Altı Avrupa Boks Şampiyonası'nı 10 madalyayla tamamlayan
        milli sporcuları tebrik etti...
        </icerik>
    </URL>
    <URL id="505">
        <baslik>Kültür ve Turizm Bakanlığı Özel Ödülleri verildi | Kültür-Sanat
        Haberleri</baslik>
        <icerik>Kültür ve Turizm Bakanı Mehmet Nuri Ersoy, Kültür ve Turizm Bakanlığı
        Özel Ödülleri töreninde yaptığı konuşmada, bu ödüllerin kültür ve sanat
        dünyasına çok önemli bir katkı sağladığını söyledi...
        </icerik>
    </URL>
    <URL id="506">
        <baslik>JAPON İŞİ... Uçan araba için hazırlıklar tamam tarih netleşti!
        Birlikte geliştirecekler...</baslik>
        <icerik>Birlikte uçan otomobil geliştirecekler! 2025'te havalanacak 26.03.2022
        - 09:28 Güncelleme: 26.03.2022 - 13:31 SkyDrive ve Suzuki, uçan otomobil
        faaliyetleri ve teknolojisinde iş birliği yaptıklarını açıkladılar...</icerik>
    </URL>
</root>
```

Şekil 3.16. Web sayfalarından alınan verilerin XML belgesine yazımı

Önerilen tarayıcının zamanlama mekanizmasının başlatılabilmesi için öncelikle kullanılacak URL'lerin indekslenmesi gerekmektedir. Verilerin ilk defa Web ortamından alınıp indekslenmesi adımları Algoritma 3.1'de gösterilmiştir.

Algoritma 3.1. Verilerin ilk defa Web ortamından alınıp indekslenmesi

1. Başla
2. İndekslenecek URL'leri veri tabanından al
3. Tüm URL'ler için yap:
4. Sayfayı indir ve ayrıştır
5. Eğer Başlık ve Gövde metni alındı ise:
6. Web sitesi ID, URL ID ve sayfada kaç defa kullanıldığını tespit et
7. Eğer Web Sitesi ID adında XML dosyası var ise:
8. XML dosyasını oluştur ve URL ID adındaki etikete verileri yaz
9. Değilse URL ID adındaki etikete verileri yaz
10. Bitir

Çalışmamızda kullandığımız Web tarayıcısının kapsamı tüm Web'dir. Verilerin anlamları, birbirleri ile ilişkileri ve bilinmeyen özelliklerinden ziyade performans ön planda olduğundan, yapay zekâ olmayan indeksleme yaklaşımları üzerinde durulmuş ve ters indeksleme yaklaşımı benimsenmiştir. Veri tabanında taranan ana sayfalar ve diğer tüm URL'ler ayrı tablolarda saklanmaktadır. Sırası gelen URL tarandıktan sonra sayfa içindeki tüm kelimeler ayrıştırılmış ve indekslenmiştir. Bu veriler, veri tabanı tablosunda sırası ile ana sayfanın kimlik tanımı (Identity Definition - ID), URL'in ID'si ve sayfada kelimenin kaç adet kullanıldığı verisi ile birlikte virgülle ayrılan değerler (Comma-separated Values - CSV) formatına uygun olarak indekslenmektedir. Çizelge 3.7'de kelimelerin indekslenmesi ile ilgili veri tabanı tablo görüntüsü gösterilmiştir.

Çizelge 3.8'de görüldüğü gibi taranan Web sayfalarından elde edilen her bir kelime, ters indeks yaklaşımı kullanılarak indekslenmiştir. Doküman sütununda bulunan ve virgül ile ayrılmış her bir üçlü gösterim A-B-C olsun;

A- Taranan ana sayfanın IT'si olup aynı zamanda verilerin kaydedildiği XML dosyasının adını,
B- A ile belirtilen XML dosyasındaki <URL> etiketinin ID'sini,
C- Kelimenin taranan sayfadaki sayısını göstermektedir.

Çizelge 3.8. Web sayfalarındaki verilerin indekslenmesi

Sıra	Kelime	Doküman
31	Altın	1-1-8,1-3-8,62-3340-1,98-4642-3,144-5590-2
39	Anne	1-1-2,1-3-2,63-3740-1,63-3741-1,144-5587-2
41	Astroloji	1-1-1,1-3-1,62-3340-1,1-2-1
58	Basketbol	1-1-2,1-3-2,63-3738-1,62-3337-1,1-2-2
64	Bilişim	1-1-2,1-3-2,1-2-2
69	Burç	1-1-1,1-3-1,62-3340-10,62-3337-1,1-2-1
75	Cevap	1-1-2,1-3-2,16-548-1,116-4791-1,1-2-2,16-549-1
79	Cumhuriyet	1-1-1,1-3-1,116-4793-1,1-2-1,62-3339-3
89	Dosya	1-1-1,1-3-1,98-4642-1,138-5235-2,138-5239-1
97	Ekonomi	1-1-6,1-3-6,16-548-1,47-2818-2,47-2819-2,47-2820-2
113	Fikstür	1-1-2,1-3-2,62-3337-1,1-2-2
122	Gazetecilik	1-1-3,1-3-3,63-3739-2,63-3740-3,63-3741-2
149	Hafta	1-1-1,1-3-1,98-4642-3,98-4648-1,1-2-1,62-3339-9

224	Personel	1-1-2,1-3-2,62-3337-2,1-2-2
282	Takım	1-1-2,1-3-2,47-2818-1,144-5590-1,138-5235-3
294	Türkiye	1-1-5,1-3-5,47-2818-1,47-2819-1,47-2820-1,63-3740-1
307	Whatsapp	1-1-1,1-3-1,16-548-1,138-5237-1,138-5235-1
321	Yol	1-1-2,1-3-2,62-3336-1,63-3740-1,138-5235-5

3.3. Heterojen Veri Kaynaklarından Toplanan Verilerin Güncellenmesi

Web üzerinde şu anda dizine alınmış 4.15 milyar sayfa bulunmaktadır [151]. Web'in bu derece büyümesi sayfaların hem taranması hem de güncelliğinin kontrol edilmesini zorlaştırmaktadır. Genel tarayıcılar, belirli bir periyotta güncelleme yaparlar; ancak bu yaklaşım, güncellenmeyen sayfaların tekrar taranması, sık güncellenen sayfalardaki verilere anında erişememe gibi bir dizi dezavantajı beraberinde getirmektedir. Web'de bazı sayfalar dakika hatta daha az bir sürede (haber, spor vb.) güncellenirken bazı Web sayfaları ayda, yılda ve bazen hiç güncellenmemektedir. Bu durumda tarayıcımızın performansını arttırmak, bant genişliğini boşa harcamamak ve güncellenen sayfaların güncel halini en uygun zamanda yakalamak için her sayfaya özel, güncelleme zamanını ayarlayan bir model kullanılmalıdır.

Bir Web sayfasının tekrar ziyaret zamanını önceki ziyaretlere dayanarak tahmin etmek, sayfanın geçmiş güncelleme frekansına göre ayarlanabilir. Eğer sayfa geçmiş ziyaretlerde sıkça güncellenmişse, tekrar ziyaret sıklığı artırılabilir; aksine, sayfa daha önce nadiren güncellenmişse sıklığı azaltılabilir. Önceki ziyaretlerin ortalaması alındığında her veri noktası eşit derecede önemli olarak görülür. Fakat sürekli değişen bir sistem üzerinde son güncel değerler eski değerlere göre durumu daha iyi yansıtma eğilimindedir. Yani bir Web sayfası ilk zamanlarda hiç güncellenmemiş daha sonraki zamanlarda ise sık güncellenmiş olabilir ya da bunun tersi durum gözlenebilir. Bu nedenle son verilerin daha önemli olduğu eski verilerin ise giderek önemini kaybettiği bir sistem yararlı olabilir. Önerilen güncelleme modülünün algoritması Algoritma 3.2'de gösterilmiştir.

Algoritma 3.2. Önerilen Güncelleme Modülünün Algoritması

1. Başla
2. URL Listesi = Sonrakiindexzamanı şimdiki zamandan küçük olan URL'ler
3. URL Listesindeki her bir URL için yap:

4. URL'e git ve indir
5. İlgili XML ile karşılaştır
6. Eğer değişiklik var ise:
7. XML i güncelle
8. EMA değerini hesapla
9. Veri Tabanını güncelle
10. Yok ise:
11. EMA değerini hesapla
12. Veri Tabanını Güncelle
13. Adım 2'ye git

Geliştirilen güncelleme modülü yönteminde, sayfaların yeniden ziyaret edilme zamanı, son dokuz ziyaret temel alınarak üstel hareketli ortalama (Exponential Moving Average - EMA) ile hesaplanmaktadır. EMA, genelde finans ve teknik analizde zaman serisi verilerini analiz etmek için kullanılan bir istatistiksel teknikler ailesidir. Tipik olarak, EMA, son gözlemlerin eskilerden daha bilgilendirici olduğunu varsayar [152]. EMA'nın hesaplanabilmesi için kök değer olarak öncelikle basit hareketli ortalamanın (Simple Moving Avarage - SMA) hesaplanması gerekir. SMA bir veri serisindeki son n adet verinin ortalamasıdır. Serideki her bir noktanın ağırlığı eşittir [153]. Bir SMA eşitliği Eş 3.7'deki gibi hesaplanır.

$$SMA = \frac{P_M + P_{M-1} + \ldots\ldots + P_{M-(n-1)}}{n} \tag{3.7}$$

Burada P_M veri noktalarının değerini belirtirken M ve n hesaplamada kullanılan veri noktalarının sayısını ifade eder. Ardışık değerler hesaplanırken formüle yeni bir değer gelir ve en eski değer atılır. Bu durumda yeni değer geldiğindeki güncel SMA değerinin hesaplanması Eş. 3.8'de gösterilmiştir.

$$SMA_{yeni} = \frac{P_M}{n} + SMA_{son} + \frac{P_{M-n}}{n} \tag{3.8}$$

EMA ise SMA'dan farklı olarak özyinelemeli bir yapıya sahiptir. n adet veri noktasının EMA değeri hesaplanırken, ilk EMA değeri SMA'dır. Daha sonraki her EMA değerinin hesaplanması Eş. 3.9'da gösterilmiştir.

$$EMA_{yeni} = \left[V_b\left(\frac{y}{1+n}\right)\right] + EMA_{son}\left[1 - \left(\frac{y}{1+n}\right)\right] \tag{3.9}$$

EMA_{yeni} : Hesaplanacak olan EMA değeri

V_b : Bugünkü değer

EMA_{son} : Bir önce hesaplanan EMA değeri

y : Yumuşatma Faktörü

n : Hesaplanan veri noktası sayısını temsil eder.

Önerdiğimiz yöntemde ziyaret edilen sayfa güncellenmiş ise V_b=2, güncellenmemiş ise V_b=1 olarak alınır. V_b 1 ya da 2 değerlerini aldığı için hesaplanan EMA değerleri de 1 ile 2 arasında bir değer alır. İlk defa taranan bir sayfada SMA=1 olarak hesaplanır ve EMA'da 1 olur. Bundan sonraki EMA hesaplamaları da kendi kendini çağıran olarak devam edeceği için önceki EMA değerlerini saklamanın bir anlamı yoktur. Veri tabanında URL, son 10 tarama durumu, son 10 taramada güncellik durumu (SMA ortalama), EMA değeri ve bir sonraki tarama zamanı saklanır. Örnek veri tabanı tablosu Çizelge 3.8'deki gibidir.

Çizelge 3.8'de sütunlar sırası ile güncelliği kontrol edilen URL'in ID'sini, ana sayfasının ID'sini, son tarama zamanını, son 10 kontroldeki güncellik durumunu, basit hareketli ortalamasını, üstel hareketli ortalamasını ve bir sonraki ziyaret zamanını göstermektedir. Sayfasın bir sonraki ziyaret zamanını, son indeks zamanına EMA değeri eklenerek hesaplanmaktadır.

Çizelge 3.9. Güncelleme modülü veri tabanı tablosu

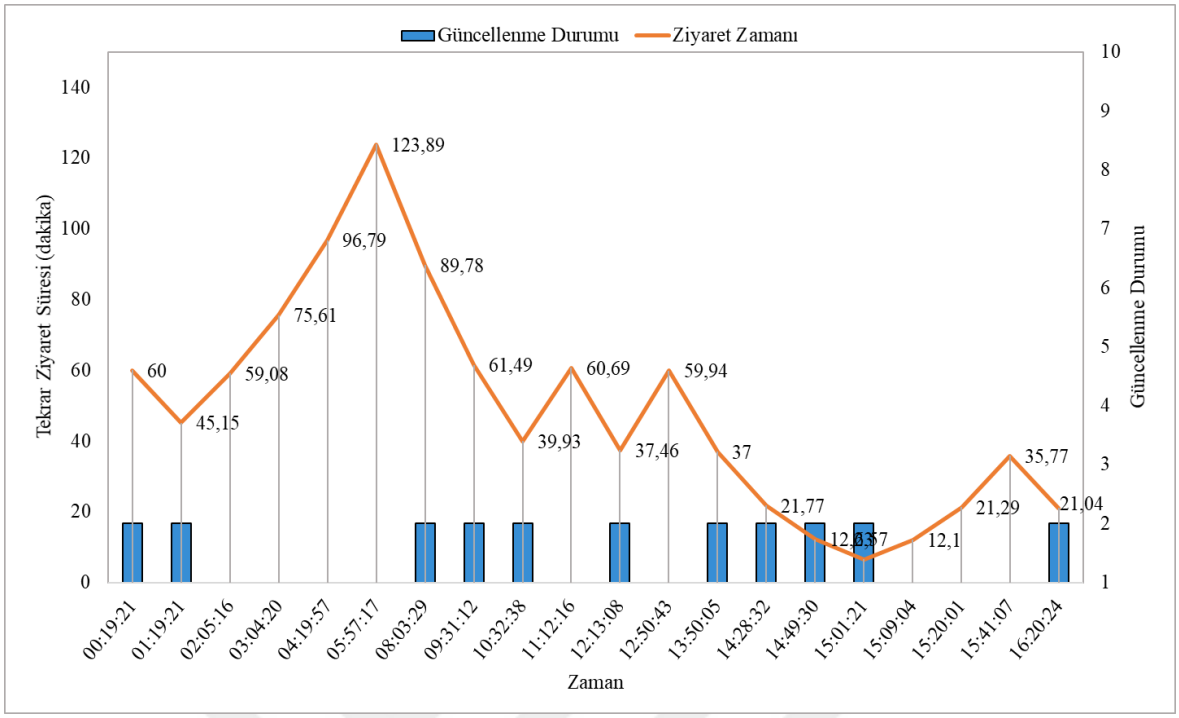
URL ID	Dok. ID	Son İndeks Zamanı	SMA	SMA ort.	EMA (dakika)	Sonraki İndeks Zamanı
383	8	6.06.2023 16:20	1,1,1,1,1,1,1,1,1,1	1	257,989	6.06.2023 20:36
958	20	6.06.2023 15:18	1,2,2,1,2,2,2,1,1,1	1,5	855,168	6.06.2023 16:42
1175	24	6.06.2023 13:22	1,1,1,1,1,1,1,1,1,1	1	214,991	6.06.2023 16:56
586	12	6.06.2023 14:05	1,2,1,1,1,1,2,1,1,1	1,2	225,831	6.06.2023 17:47
135	3	6.06.2023 18:50	1,1,1,1,1,1,2,1,1,2	1,2	136,35	6.06.2023 19:15
1078	22	6.06.2023 14:56	1,1,1,1,1,2,1,2,1,1	1,2	164,189	6.06.2023 17:40
335	7	6.06.2023 16:14	1,1,1,1,1,1,1,1,1,1	1	257,989	6.06.2023 20:32
1208	25	6.06.2023 13:25	1,1,1,1,1,1,1,1,1,1	1	214,991	6.06.2023 16:59
373	8	6.06.2023 16:20	1,1,1,1,1,1,1,1,1,1	1	257,989	6.06.2023 20:35
101	3	6.06.2023 18:50	1,2,1,1,1,1,1,1,1,1	1,1	332,572	6.06.2023 23:07
383	8	6.06.2023 16:20	1,1,1,1,1,1,1,1,1,1	1	257,989	6.06.2023 20:36
958	20	6.06.2023 15:18	1,2,2,1,2,2,2,1,1,1	1,5	855,168	6.06.2023 16:42

Her bir güncelleme işleminden sonra elde edilen kayıtlar saklanmaktadır. Aşağıdaki Çizelge 3.9'da veri tabanında rastgele seçilen 21 ID numaralı Web sitesine ait 1041 numaralı URL'in kaydı gösterilmiştir. Güncelleme durumunu gösteren SMA sütununda, yeni durum kuyruğa sağdan eklenmektedir. Bu durumda soldaki son güncelleme durumu kuyruktan atılarak tekrar hesaplanmaktadır. EMA sütunundaki değer son güncellemeden itibaren kaç dakika sonra tekrar ziyaret edileceğini göstermektedir.

Çizelge 3.10. Bir URL'in güncellenme durumuna göre tekrar ziyaret süresinin belirlenmesi

URL ID	Dok. ID	Son İndeks Zamanı	SMA	SMA ort.	EMA (dakika)	Sonraki İndeks Zamanı
1041	21	6.06.2023 01:19	1,1,1,1,1,1,1,1,1,2	2	45,15	6.06.2023 02:05
1041	21	6.06.2023 02:05	1,1,1,1,1,1,1,1,2,1	1	59,08	6.06.2023 03:04
1041	21	6.06.2023 03:04	1,1,1,1,1,1,1,2,1,1	1	75,61	6.06.2023 04:19
1041	21	6.06.2023 04:19	1,1,1,1,1,1,2,1,1,1	1	96,79	6.06.2023 05:56
1041	21	6.06.2023 05:57	1,1,1,1,1,2,1,1,1,1	1	123,89	6.06.2023 08:00
1041	21	6.06.2023 08:03	1,1,1,1,2,1,1,1,1,2	2	89,78	6.06.2023 09:30
1041	21	6.06.2023 09:31	1,1,1,2,1,1,1,1,2,2	2	61,49	6.06.2023 10:31
1041	21	6.06.2023 10:32	1,1,2,1,1,1,1,2,2,2	2	39,93	6.06.2023 11:11
1041	21	6.06.2023 11:12	1,2,1,1,1,1,2,2,2,1	1	60,69	6.06.2023 12:12
1041	21	6.06.2023 12:13	2,1,1,1,1,2,2,2,1,2	2	37,46	6.06.2023 12:50
1041	21	6.06.2023 12:50	1,1,1,1,2,2,2,1,2,1	1	59,94	6.06.2023 13:49
1041	21	6.06.2023 13:50	1,1,1,2,2,2,1,2,1,2	2	37	6.06.2023 14:26
1041	21	6.06.2023 14:28	1,1,2,2,2,1,2,1,2,2	2	21,77	6.06.2023 14:48
1041	21	6.06.2023 14:49	1,2,2,2,1,2,1,2,2,2	2	12,23	6.06.2023 15:00
1041	21	6.06.2023 15:01	2,2,2,1,2,1,2,2,2,2	2	6,57	6.06.2023 15:07
1041	21	6.06.2023 15:09	2,2,1,2,1,2,2,2,2,1	1	12,1	6.06.2023 15:19
1041	21	6.06.2023 15:20	2,1,2,1,2,2,2,2,1,1	1	21,29	6.06.2023 15:40
1041	21	6.06.2023 15:41	1,2,1,2,2,2,2,1,1,1	1	35,77	6.06.2023 16:16
1041	21	6.06.2023 16:20	2,1,2,2,2,2,1,1,1,2	2	21,04	6.06.2023 16:37

Çizelge 3.9'da görüldüğü gibi güncellenme durumuna göre tekrar ziyaret süresi son güncellenme durumu daha etkili olacak şekilde üstel olarak artmakta ve azalmaktadır. Çizelge 3.9'da örnek olarak seçilen URL'in güncellenme durumuna göre tekrar ziyaret zamanı ve tekrar ziyaret süresini gösteren grafik Şekil 3.17'de gösterilmiştir.



Şekil 3.17. URL'in güncellenme durumuna göre ziyaret zamanı ve tekrar ziyaret süresi

4. DENEYSEL ÇALIŞMALAR

EMA yönteminin etkinliğini değerlendirmek için gerçek zamanlı bir tarayıcı olan Real-Time [154] Web tarayıcısı kullanılmıştır. Önerilen Web tarayıcı mekanizması, MySQL veri tabanı kullanılarak Python programlama dili ile uygulanmıştır. Deneysel çalışmalar Windows 11 işletim sistemi, i5 2,90 GHz ve 16 GB belleğe (RAM) sahip iki bilgisayarda eş zamanlı olarak gerçekleştirilmiştir.

4.1.Veri Kümesi

Deneysel çalışma, aynı özelliklere sahip iki makinede aynı zaman aralığında gerçekleştirilmiştir. Tarayıcıların tarama işlemlerine başlamaları için kullanılan tohum URL kümeleri, Sumrush [155] üzerinden beş farklı alandan en çok ziyaret edilen Türkçe Web siteleri içinden seçilen 23 farklı Web sitesi domain adresleri kullanılarak oluşturulmuştur. Tohum URL'lerden çıkartılan 1046 adet URL her iki tarayıcı tarafından periyodik olarak taranıp sayfalardaki güncellemeler takip edilmiştir. Bu Web siteleri ile ilgili detaylı bilgi Çizelge 4.1'de gösterilmiştir.

Taranan URL'lerin güncellenme durumuna göre değişken periyotlarda yapılan tekrar ziyaret sayıları ve bu ziyaretlerde tespit edilen değişiklik sayıları veri tabanında saklanmıştır. Buna göre URL'ler dinamik olmayan, az dinamik, orta dinamik ve yüksek dinamik olmak üzere 4 farklı dinamiklik grubuna ayrılmıştır. Yapılan tarama süresince hiç değişmeyen URL'ler dinamik olmayan sayfalar grubuna eklenmiştir. Değişiklik tespit edilen URL'lerin ise dinamiklik (D) durumu Eş. 4.1 ile hesaplanmıştır.

$$D = \frac{Değişiklik\ Sayısı}{Ziyaret\ Sayısı} \tag{4.1}$$

Çizelge 4.1. Çalışmada kullanılan Web sayfaları ve URL sayıları

Kategoriler	Tohum URL'ler	Taranan URL Sayıları
E-ticaret ve Perakende	www.trendyol.com	50
	www.kitapyurdu.com	21
	www.akakce.com	21

Kategoriler	Tohum URL'ler	Taranan URL Sayıları
	www.amazon.com.tr	41
	www.vivense.com	50
Yolculuk ve Turizm	www.obilet.com	48
	www.hotels.com	47
	www.flypgs.com	50
	www.etstur.com	50
	www.booking.com	49
Eğitim	www.memurlar.net	23
	www.meb.gov.tr	35
	www.anadolu.edu.tr	60
	www.boun.edu.tr	45
Finans	www.haremaltin.com	17
	www.doviz.com	44
	www.borsagundem.com	70
	www.canlidoviz.com	44
Haber ve Medya	www.sondakika.com	83
	www.hurriyet.com	50
	www.ensonhaber.com	48
	www.haberturk.com	50
	www.haberler.com	50
TOPLAM	23	1046

Buna göre D değeri 0 olanlar dinamik olmayan, 0 dan büyük ve 0.25 den küçük olanlar az dinamik, 0.25 - 0.5 arasında olanlar orta dinamik ve 0.5 ile 1 arasında olanlar ise yüksek dinamik URL'ler grubuna eklenmiştir. Karşılaştırılan her iki tarayıcıda 4 gruba ait URL sayıları Çizelge 4.2'de gösterilmiştir.

Çizelge 4.2. Dinamiklik durumuna göre URL sayıları

Dinamiklik Durumu	URL Sayısı	
	EMA	Real-Time
Dinamik olmayan	745	758
Az dinamik	60	189
Orta Dinamik	173	81
Yüksek Dinamik	68	18

Çizelge 4.2'de görüldüğü gibi her iki tarayıcı da dinamik olmayan sayfaları birbirine yakın değerlerde tespit etmiştir. Az dinamik olan sayfalar, Real-Time tarayıcısı tarafından daha fazla kategorize edilirken, orta ve yüksek dinamik sayfalar ise EMA tarayıcısı tarafından daha fazla işaretlenmiştir. Bu dinamiklik durumlarının kategorilere göre dağılımı Çizelge 4.3'de gösterilmiştir.

Çizelge 4.3. Her bir kategorinin dinamiklik durumlarına göre URL sayıları

Kategoriler	Dinamiklik Durumu	URL Sayısı	
		EMA	Real-Time
E-ticaret ve Perakende	Dinamik olmayan	139	142
	Az	21	17
	Orta	17	24
	Yüksek	6	2
Yolculuk ve Turizm	Dinamik olmayan	200	208
	Az	16	17
	Orta	22	14
	Yüksek	6	6
Eğitim	Dinamik olmayan	106	104
	Az	2	34
	Orta	55	24
	Yüksek	0	0
Finans	Dinamik olmayan	120	122
	Az	15	42
	Orta	23	9
	Yüksek	17	4
Haber ve Medya	Dinamik olmayan	180	182
	Az	6	79
	Orta	56	10
	Yüksek	39	6
TOPLAM		1046	1046

4.2. Değerlendirme Ölçütleri

Karşılaştırılan EMA ve Real-Time tarayıcılarının performansları kesinlik, toplam kapsama oranı ve verimlilik olmak üzere üç standart ölçüt kullanılarak ölçülmüştür. Kesinlik, önerilen tarayıcı tarafından yeniden taranan tüm sayfaların değişiklikle karşılaşan kısmı üzerinden

hesaplanır [44]. Kesinlik (K), değişiklikle karşılaşılan sayfalar (D) ve tüm sayfalar (T) olmak üzere Eş. 4.2'deki gibi hesaplanır.

$$K = \frac{D}{T} \tag{4.2}$$

Toplam kapsama oranı, taranan benzersiz kayıt sayısının veri tabanında bulunan tüm kayıtlara oranını olup Eş. 4.3 ile gösterilmektedir.

$$Toplam\ Kapsama\ Oranı = \frac{Benzersiz\ kayıt\ sayısı}{Veri\ tabanındaki\ Tüm\ Kayıtlar} \tag{4.3}$$

Belirli bir zaman örneğinde (ti) bir veri tabanında N_c, artımlı tarayıcının tarayabileceği kayıt sayısını temsil etmektedir. N_c ise o anda veri tabanındaki kayıt sayısının, veri tabanında ti'den ti+1'e eklenen, silinen veya güncellenen kayıt sayısının ve artımlı tarayıcının tarayabileceği kayıt sayısının toplamını ifade etmektedir [110]. Bu verilere dayanarak verimlilik Eş.4.4'de gösterilmiştir.

$$Verimlilik = \frac{N_s}{N_c} \tag{4.4}$$

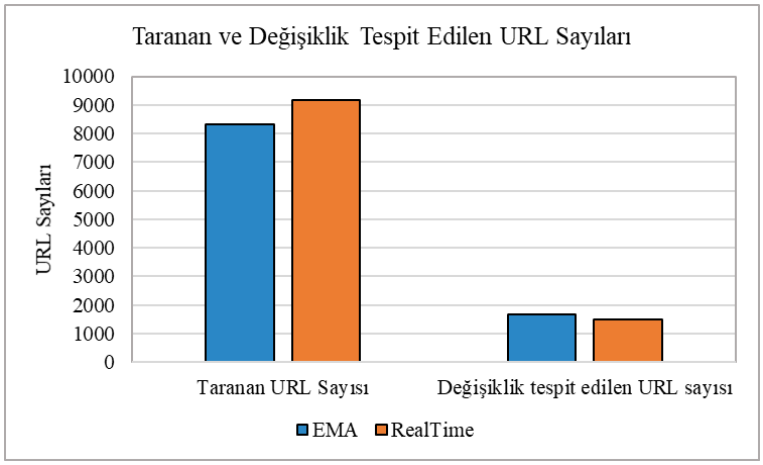
4.3. Deneysel Sonuçlar

EMACrawler ve Real-Time tarayıcıları, eş zamanlı olarak 12 saat boyunca, 5 farklı kategoride seçilen 1046 farklı URL için çalıştırılmıştır. Tarama sonucu taranan benzersiz URL sayısı, taranan URL sayısı, değişiklik tespit edilen URL sayısı, değişiklik tespit edilmeyen URL sayısı, kesinlik, toplam kapsama oranı ve verimlilik ölçütleri kullanılarak karşılaştırılmıştır. Her iki tarayıcı için yapılan tarama sonuçları Çizelge 4.4'de gösterilmiştir.

Taranan benzersiz URL sayısındaki değişimlerin incelenmesi, tarama işleminin kapsamını ve etkinliğini anlamak için önemlidir. Her aşamada değişiklik tespit algoritmalarıyla tespit edilen ve edilemeyen URL sayıları karşılaştırılmaktadır. Bu, tespit edilen değişikliklerin oranını ve algoritmaların başarısını göstermektedir.

Çizelge 4.4. URL'lerin taranması sonuçları ve performans ölçütleri

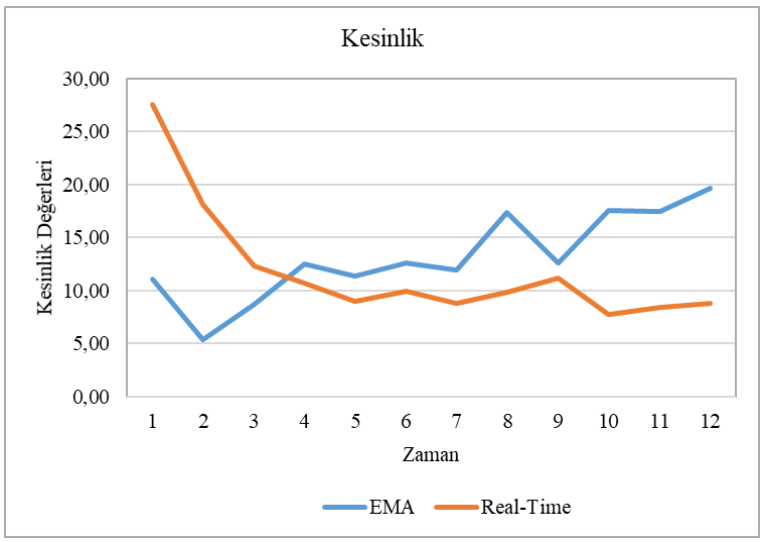
	Taranan benzersiz URL Sayısı		Taranan URL Sayısı		Değişiklik tespit edilen URL sayısı		Değişiklik tespit edilmeyen URL sayısı		Kesinlik		Toplam kapsama oranı		Verimlilik	
Tarama Zamanı(Saat)	EMA	Real-Time	EMA	Real-Time	EMA	Real-Time	EMA	Real-Time	EMA	Real-Time	EMA	Real-Time	EMA	Real-Time
1	1046	1046	1102	1395	116	288	986	1107	11,09	27,53	2,80	2,57	2,11	2,28
2	965	1003	992	1105	56	190	936	915	5,35	18,16	2,82	2,66	2,14	2,32
3	588	806	659	884	91	129	568	755	8,70	12,33	2,44	2,58	2,93	3,37
4	627	552	759	736	131	112	628	624	12,52	10,71	2,42	2,38	2,88	2,73
5	789	434	926	759	119	94	807	665	11,38	8,99	2,58	2,30	2,48	1,99
6	305	306	455	671	132	104	323	567	12,62	9,94	2,15	2,18	4,86	4,77
7	648	456	769	650	125	92	644	558	11,95	8,80	2,45	2,32	2,81	2,46
8	428	396	522	586	182	103	340	483	17,40	9,85	2,20	2,26	3,87	3,61
9	372	378	466	511	132	117	334	394	12,62	11,19	2,20	2,22	4,17	4,14
10	728	485	828	632	184	81	644	551	17,59	7,74	2,44	2,36	2,69	2,21
11	154	203	297	572	183	88	114	484	17,50	8,41	1,98	2,10	8,98	8,68
12	374	195	535	667	206	92	329	575	19,69	8,80	2,13	2,09	4,35	3,56



Şekil 4.1. Taranan ve değişiklik elde edilen URL sayıları

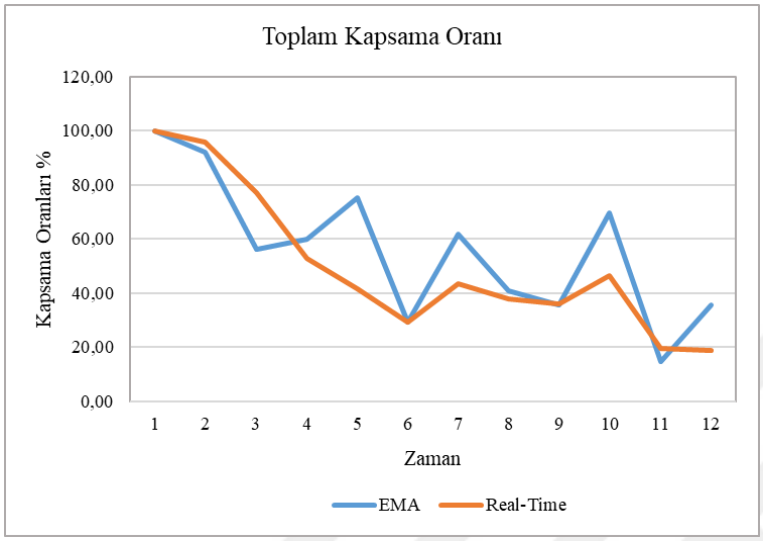
Taranan ve değişiklik tespit edilen toplam URL sayıları incelendiğinde, önerilen EMA tarayıcısının belirtilen süre içinde gerçekleştirdiği toplam tarama sayısının Real-Time tarayıcısına kıyasla daha düşük olduğu gözlemlenmiştir. Ancak, değişiklik tespit edilen URL sayısı açısından EMA tarayıcısının daha etkili bir performans sergilediği görülmektedir. EMA tarayıcısı, daha az tarama sayısıyla daha fazla değişiklik tespit etme kapasitesine sahiptir. Bu durum, EMA tarayıcısının daha verimli bir tarama gerçekleştirdiğini ve performans açısından avantajlı olduğunu ortaya koymaktadır.

Çizelge 4.4'de gösterilen tarama sonucunda elde edilen sonuçlar kullanılarak kesinlik, toplam kapsama oranı ve verimlilik ölçütleri hesaplanmıştır. Elde edilen sonuçlar Şekil 4.2, Şekil 4.3 ve Şekil 4.4'de grafiksel olarak gösterilmiştir.



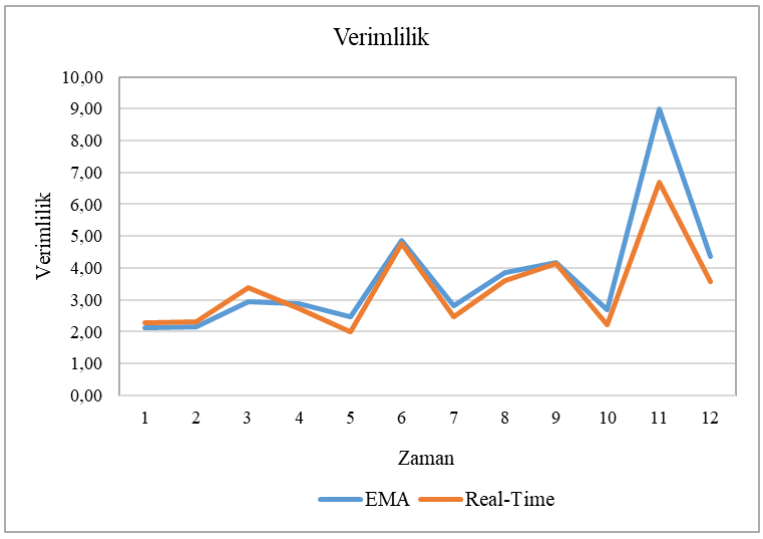
Şekil 4.2. Tarayıcıların kesinlik ölçütlerinin karşılaştırılması

Real-Time tarayıcısı, ilk taramada daha fazla güncellen sayfa tespit edip tarama süresini düşürdüğü için kesinlik değeri yüksektir. Fakat süre ilerledikçe kesinlik değeri azalan bir seyir izlemektedir. Buna karşın, EMA tarayıcısı ilk taramadan itibaren artan kesinlik oranı ile taramasına devam etmektedir.



Şekil 4.3. Tarayıcıların toplam kapsama oranlarının karşılaştırılması

Toplam kapsama oranları incelendiğinde, iki tarayıcı da tüm URL'ler ile %100 oranı ile taramaya başlamaktadır. Her iki tarayıcıda da zaman ilerledikçe kapsama oranları düşmekte olup, EMA tarayıcısı çoğunlukla Real-Time tarayıcısından daha iyi performans göstermektedir.



Şekil 4.4. Tarayıcıların verimliliklerinin karşılaştırılması

Genel ve artımlı Web tarayıcılarında en önemli performans ölçütlerinden biri de verimliliktir. Şekil 4.4 incelendiğinde, önceki performans ölçütlerine paralel olarak taramanın başlarında EMA tarayıcısının verimliliği Real-Time tarayıcısına göre başlangıçta daha az iken, 3. saatten sonra daha iyi sonuçları elde etmiştir.

5. SONUÇLAR VE ÖNERİLER

İnternet kullanıcıları, Web ortamındaki hedefledikleri verilere arama motorlarını kullanarak erişim sağlamaktadır. Bu kullanıcıların beklentileri arasında, arama motorlarının güncel ve doğru veriyi sağlama yükümlülüğü öne çıkmaktadır. Özellikle güncel verilere olan talepler karşısında, Web sayfalarının önemi ve uygunluğu ile ilgilenen arama motorları yetersiz kalmaktadır. Web gibi heterojen veri kaynaklarından veri toplama, bu verileri entegre etme ve güncelleme süreçleri, veri yönetimi süreçlerinde önemli zorluklar ve gereklilikler içermektedir.

Bu gereksinimler ışığında, bu tez çalışmasında, Web ortamındaki verilerin toplanması, entegrasyonu ve güncellenmesi konuları ele alınmış ve bu konuda yapılmış akademik çalışmalar incelenmiştir. Yapılan araştırmalarda Web tarayıcılarının performansını etkileyen faktörler detaylı bir şekilde incelenmiştir. Bu kapsamda Web tarayıcılarının, tohum URL seçim metotları, kapsam genişletme yöntemi, toplanan verilerin entegrasyonu ve güncellemeleri zamanında yakalamak için kullandığı algoritmalar detaylı bir şekilde incelenmiştir. Literatürde bu konular ile ilgili birçok çalışma yapılmış ve çeşitli çözüm önerileri getirişmiş olmasına rağmen, Web'in dinamik doğası gereği zaman geçtikçe eski sistemlerin performansı da düşmekte ve gelişen teknolojilere uygun olarak yeni algoritmaların geliştirilmesi gerekmektedir. Çalışmamızda Web tarayıcıları ile ilgili tespit edilen bu konularla ilgili eksikliklere odaklanılmış ve çözüm önerileri geliştirilmeye çalışılmıştır.

Bu tez çalışmamızda, ilk olarak, Web sayfalarını taramak amacıyla kullanılan tohum URL seçim metodu ve tarama algoritması tanıtılmıştır. Tohum URL'leri belirlemek için 102 farklı ülkede ziyaretçilerin günlük harcadığı saat, ziyaret başına günlük sayfa görüntüleme sayısı, arama trafiğinin yüzdesi ve toplam bağlı site sayısı gibi ölçütler temel alınmıştır. Bu kapsamda belirlenen toplam 5079 URL içerisinden bazı özelliklere göre seçilmiş 3 tohum URL kümesi kullanılarak sistem performansı test edilmiştir.

İkinci olarak, tohum URL kümeleri kullanılarak kapsamı hızlı bir şekilde genişletmek için InterDomain, IntraDomain ve ilk rastlanan ana sayfalara LS belirlenerek önceliklendirme yapılmaktadır. Ana sayfalara, en yüksek önem derecesine sahip oldukları için 1 değeri

atanmaktadır. Ayrıca, LS hesaplamasında farklı değer aralıklarıyla yapılan testler sonucunda, IntraDomain URL'ler için LS_{min}'in 0-0.5 ve InterDomain URL'ler için LS_{max}'ın 0.5 ile 1 arasında en uygun değer aralığı olduğu tespit edilmiştir. Algoritmanın 3 farklı tohum URL kümesi üzerinden gerçekleştirilen 12 saatlik taramalarda elde ettiği taranan URL sayısı, taranan ana sayfa sayısı ve hatalı URL sayıları detaylı bir şekilde incelenmiş ve karşılaştırılmıştır. Ayrıca, kapsamın dünya genelindeki bölgesel dağılımları incelenmiş ve taramalar sonucunda elde edilen ana sayfaların ülkelere göre dağılımları karşılaştırılmıştır. Bunlara ek olarak 3 tohum URL kümesi için, taramalar sırasında meydana gelen hatalar, hata sayıları ve hata içerikleri incelenmiş ve kapsamlı bir analiz edilmiştir. Önerilen algoritma, hem yeni sayfaları ve bu sayfalardaki URL'leri etkili bir şekilde elde edip öncelikli kuyruğa eklemekte, hem de alan içi döngülerden başarılı bir şekilde kaçınmaktadır. Yapılan deneysel çalışmalara göre, önerilen algoritma, kapsamı genişletme, yinelenen URL'ler ve alan içi döngülerden kaçınma konusunda iyi bir performans göstermektedir.

Bu tez çalışması, üçüncü odak noktası olarak, Web kullanıcılarına en güncel veriyi en uygun zamanda sunmak için yeni bir yöntem önermektedir. Bu amaç doğrultusunda, tekrar ziyaret zamanlarının belirlenmesi için üstel hareketli ortalama olan EMA yöntemi önerilmiştir. EMA yöntemi kullanılarak, EMACrawler adını verdiğimiz bir Web tarayıcısı geliştirilmiştir. Yapılan deneylerde elde edilen tarama sonuçları kesinlik, kapsama oranları ve verimlilik arasında önemli bir denge olduğunu göstermiştir. EMACrawler, yüksek kesinlik değeri, uygun kapsama oranı ve verimlilik değeri ile büyük ölçekli Web taramalarında etkin bir performans sergileyerek, güncel verilerin elde edilmesi ve veri ambarlarının tazeliğinin korunmasında arama motorlarına yardımcı olabileceği görülmüştür. Ayrıca, daha sonra ziyaret edilmesi gereken URL'ler için kullanılan zamanlama mekanizması ile arama motorunun, veri tabanında kaliteli ve güncel veriler barındırması sağlanmaktadır.

Bu tez çalışmasının, Web tarayıcılarının tohum URL kümesi seçimi, kapsamı, tarama algoritması ve veri güncelliğine odaklanarak, Web tarayıcı alanına önemli katkılarda bulunduğuna inanmaktayız. Ancak, Web hızlı bir şekilde gelişen ve değişen dinamik bir yapıya sahip olduğundan, veri toplama, entegrasyonu ve güncellenmesi ile ilgili yeni sorunların ortaya çıkması kaçınılmazdır. Verilerin toplanması ve veri depolarının güncelliği konusunda Web sayfalarının indirilip ayrıştırması önemli derecede performans kayıplarına neden olmaktadır. Bu bağlamda, sayfaların sadece güncellenen kısımlarının indirilmesi ve

bu işlemler için bilgi işlem sürecinin dağıtık bir yapıda sunucu bilgisayarlarda gerçekleştirilmesi, genel performansı artırabilir.

Yapılandırılmamış heterojen verilerin yapılandırılıp ilişkisel veri tabanlarında tutulması çeşitli sorunları beraberinde getirmektedir. Bunun yerine her türden verilerin saklanabileceği hem ilişkisel hem de ilişkisel olmayan veri tabanlarının kullanılması veri entegrasyon sorununu kolaylaştırabilir. Daha gelişmiş bir sistem kullanılarak, Web tarayıcıları gelişen yapay zekâ teknolojileriyle entegre edildiğinde, veri toplama ve işleme süreçlerinde önemli ölçüde verimlilik artışı sağlanabilir. Yapay zekâ tabanlı Web tarayıcıları, otomatik olarak Web sayfalarını tarama, içerik analizi yapma, önemli bilgileri çıkarma ve yapılandırılmış verilere dönüştürme yeteneklerini içerebilir.

KAYNAKLAR

1. Cofas, E., *The role of big data in digitalizing information* Scientific Papers Series Management, Economic Engineering in Agriculture & Rural Development, 2023. 23(3).

2. Philip Chen, C.L. and C.-Y. Zhang, *Data-intensive applications, challenges, techniques and technologies: A survey on Big Data.* Information Sciences, 2014. 275: p. 314-347.

3. İnternet: Hootsuite, W.A.S.v. *Digital 2022: Another Year Of Bumper Growth*. 2022 [cited 2023 11 Temmuz]; Available from: https://wearesocial.com/uk/blog/2022/01/digital-2022-another-year-of-bumper-growth-2/.

4. Hashem, I.A.T., et al., *The rise of "big data" on cloud computing: Review and open research issues.* Information Systems, 2015. 47: p. 98-115.

5. Hussein, A.H., *Internet of things (IOT): Research challenges and future applications.* International Journal of Advanced Computer Science and Applications, 2019. 10(6).

6. Abdalla, H.B., *A brief survey on big data: technologies, terminologies and data-intensive applications.* Journal of Big Data, 2022. 9(1): p. 107.

7. Pääkkönen, P. and D. Pakkala, *Reference architecture and classification of technologies, products and services for big data systems.* Big data research, 2015. 2(4): p. 166-186.

8. Assunção, M.D., et al., *Big Data computing and clouds: Trends and future directions.* Journal of parallel and distributed computing, 2015. 79: p. 3-15.

9. Zhang, Y., et al., *A big data analytics architecture for cleaner manufacturing and maintenance processes of complex products.* Journal of cleaner production, 2017. 142: p. 626-641.

10. Blazquez, D. and J. Domenech, *Big Data sources and methods for social and economic analyses.* Technological Forecasting and Social Change, 2018. 130: p. 99-113.

11. Google. *How Google Search Works*. 2022 [cited 2022 10 August].

12. Sadiku, M., S. Musa, and S.R. Nelatury, *Future Internet research.* International Journal of Advances in Scientific Research and Engineering (IJASRE), 2017. 3(2): p. 23-25.

13. Jaiganesh, S., P. Babu, and K.N. Satheesh, *Comparative study of various Web search algorithms for the improvement of Web crawler.* Int. J. Eng. Res. Technol.(IJERT), 2013. 2(4).

14. Almukhtar, F., N. Mahmoodd, and S. Kareem, *Search engine optimization: a review.* Applied Computer Science, 2021. 17(1): p. 70-80.

15. Boeing, G. and P. Waddell, *New Insights into Rental Housing Markets across the United States: Web Scraping and Analyzing Craigslist Rental Listings.* Journal of Planning Education and Research, 2017. 37(4): p. 457-476.

16. İnternet: Wikipedia. *Web scraping*. 2019 [cited 2019 3.12.2019]; Available from: https://en.wikipedia.org/wiki/Web_scraping.

17. Bar-Yossef, Z., I. Keidar, and U. Schonfeld, *Do not crawl in the DUST: Different URLs with similar text.* ACM Transactions on the Web (TWEB), 2009. 3(1): p. 1-31.

18. Rodrigues, K., et al., *Removing DUST Using Multiple Alignment of Sequences.* IEEE Transactions on Knowledge and Data Engineering, 2015. 27(8): p. 2261-2274.

19. Pavalam, S.M., et al., *A survey of Web crawler algorithms.* International Journal of Computer Science Issues (IJCSI), 2011. 8(6): p. 309.

20. İnternet: *Depth-first search*. 28.09.2020 [cited 2020 28 Septenber]; Available from: https://en.wikipedia.org/wiki/Depth-first_search#/media/File:Depth-first-tree.svg.

21. Jasani, B.M. and C.K. Kumbharana, *Analyzing Different Web Crawling Methods.* International Journal of Computer Applications, 2014. 975: p. 8887.

22. Baker, M. and M. Akcayol, *Priority queue based estimation of importance of Web pages for Web crawlers.* International Journal of Electrical and Computer Engineering, 2017. 9(1): p. 330-342.

23. Thangaraj, M. and P.G. Sivagaminathan, *An Improved Generic Crawler using Poisson Fit Distribution.* Communications. 6: p. 7-13.

24. Lee, H.-T., et al., *IRLbot: scaling to 6 billion pages and beyond.* ACM Transactions on the Web (TWEB), 2009. 3(3): p. 1-34.

25. Fasolin, K., et al. *Efficient Execution of Conjunctive Complex Queries on Big Multimedia Databases*. in *2013 IEEE International Symposium on Multimedia*. 2013.

26. Gani, A., et al., *A survey on indexing techniques for big data: taxonomy and performance evaluation.* Knowledge and Information Systems, 2016. 46(2): p. 241-284.

27. Shah, S. and A. Shaikh. *Hash based optimization for faster access to inverted index*. in *2016 International Conference on Inventive Computation Technologies (ICICT)*. 2016.

28. Petri, M. and A. Moffat, *Compact inverted index storage using general-purpose compression libraries.* Software: Practice and Experience, 2018. 48(4): p. 974-982.

29. Patil, T.A. and S. Chobe. *Web Crawler for searching Deep Web sites*. in *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*. 2017. IEEE.

30. Bullot, H., S.K. Gupta, and M.K. Mohania. *A data-mining approach for optimizing performance of an incremental crawler*. in *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*. 2003.

31. Kharazmi, S., A.F. Nejad, and H. Abolhassani. *Freshness of Web search engines: Improving performance of Web search engines using data mining techniques*. in *2009 International Conference for Internet Technology and Secured Transactions, (ICITST)*. 2009.

32. Jianchao, H., N. Cercone, and H. Xiaohua. *A Weighted Freshness Metric for Maintaining Search Engine Local Repository*. in *IEEE/WIC/ACM International Conference on Web Intelligence (WI'04)*. 2004.

33. Amudhan, V. and D. Thirupathi. *Traffic Adaptive Optimum Updating Scheme for Search Engines*. in *2006 1st International Conference on Digital Information Management*. 2007.

34. Zhu, W., et al., *Optimal bandwidth allocation for Web crawler systems with time constraints.* Journal of Ambient Intelligence and Humanized Computing, 2023. 14(5): p. 5279-5292.

35. Souza, C., et al. *A Polite Policy for Revisiting Web Pages*. in *2007 Latin American Web Conference (LA-WEB 2007)*. 2007.

36. Bhatia, S., M. Sharma, and K.K. Bhatia, *A Novel Approach for Crawling the Opinions from World Wide Web.* International journal of information retrieval research, 2016. 6(2): p. 1-23.

37. Tan, Q. and P. Mitra, *Clustering-based incremental Web crawling.* ACM Trans. Inf. Syst., 2010. 28(4): p. Article 17.

38. Radinsky, K. and P.N. Bennett, *Predicting content change on the Web*, in *Proceedings of the sixth ACM international conference on Web search and data mining*. 2013, Association for Computing Machinery: Rome, Italy. p. 415–424.

39. Li, H., et al. *An incremental update strategy in Deep Web*. in *2010 Sixth International Conference on Natural Computation*. 2010.

40. Mor, J., D. Rai, and N. Kumar, *An XML based Web Crawler with Page Revisit Policy and Updation in Local Repository of Search Engine.* International Journal of Engineering & Technology, 2018. 7: p. 1119.

41. Kausar, M.A., M. Nasar, and S.K. Singh. *Maintaining the repository of search engine freshness using mobile crawler*. in *2013 Annual International Conference on Emerging Research Areas and 2013 International Conference on Microelectronics, Communications and Renewable Energy*. 2013.

42. Badawi, M., et al., *Maintaining the search engine freshness using mobile agent.* Egyptian Informatics Journal, 2013. 14(1): p. 27-36.

43. Gupta, A., A. Dixit, and A. Sharma, *A Novel Web Page Change Detection Technique for Migrating Crawlers*. 2018. p. 49-57.

44. Sethi, S., *An optimized crawling technique for maintaining fresh repositories.* Multimedia Tools and Applications, 2021. 80(7): p. 11049-11077.

45. Santos, A.S.R., et al., *A genetic programming framework to schedule Webpage updates.* Information Retrieval Journal, 2015. 18(1): p. 73-94.

46. Jaybal, Y., C. Ramanathan, and S. Rajagopalan. *Hdsanalytics: a data analytics framework for heterogeneous data sources*. ACM.

47. Marr, B., *Why Only One of the 5 Vs of Big Data Really Matters IBM Big Data & Analytics Hub: IBM. 2015*.

48. Pol, U.R., *Big data analysis using Hadoop MapReduce.* Am. J. Eng. Res. AJER, 2016. 5: p. 146-151.

49. Eberendu, A.C., *Unstructured Data: an overview of the data of Big Data.* International Journal of Computer Trends and Technology, 2016. 38(1): p. 46-50.

50. Zhao, Y. and J. Chen, *A survey on differential privacy for unstructured data content.* ACM Computing Surveys (CSUR), 2022. 54(10s): p. 1-28.

51. Zhang, L., N. Li, and Z. Li. *An Overview on Supervised Semi-structured Data Classification*. in *2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)*. 2021.

52. Bahrami, M., M. Singhal, and Z. Zhuang. *A cloud-based Web crawler architecture*. in *2015 18th International Conference on Intelligence in Next Generation Networks*. 2015.

53. Hurst, M. and A. Maykov. *Social Streams Blog Crawler*. in *2009 IEEE 25th International Conference on Data Engineering*. 2009.

54. Boanjak, M., et al., *TwitterEcho - A distributed focused crawler to support open research with twitter data.* 2012.

55. Psallidas, F., A. Ntoulas, and A. Delis. *Soc Web: Efficient monitoring of social network activities*. in *International Conference on Web Information Systems Engineering*. 2013. Springer.

56. Gossen, G., E. Demidova, and T. Risse, *iCrawl: Improving the Freshness of Web Collections by Integrating Social Web and Focused Web Crawling*, in *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries*. 2015, Association for Computing Machinery: Knoxville, Tennessee, USA. p. 75–84.

57. Nakamoto, S., *Bitcoin: A peer-to-peer electronic cash system.* Decentralized Business Review, 2008: p. 21260.

58. Wang, J., et al. *FDataCollector: A Blockchain Based Friendly Web Data Collection System*. in *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*. 2021.

59. Kalmukov, Y. and I. Valova. *Design and development of an automated Web crawler used for building image databases*. in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. 2019.

60. Ali, A., et al. *Large Scale Image Dataset Construction Using Distributed Crawling with Hadoop YARN*. in *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*. 2018.

61. Zhou, F. and Y. Wang. *Exploring The Role of Web Crawler and Anti-Crawler Technology in Big Data Era*. in *2022 11th International Conference of Information and Communication Technology (ICTech))*. 2022.

62. Zhang, Z., et al. *A Study on Unofficial Geographic Location Data Acquisition Technology Path under the Background of Big Data Era*. in *IOP Conference Series: Materials Science and Engineering*. 2019. IOP Publishing.

63. Shemshadi, A., et al., *Searching for the internet of things: where it is and what it looks like.* Personal and Ubiquitous Computing, 2017. 21: p. 1097-1112.

64. Liang, F., et al., *Search engine for the internet of things: Lessons from Web search, vision, and opportunities.* IEEE Access, 2019. 7: p. 104673-104691.

65. Shemshadi, A., Q.Z. Sheng, and Y. Qin. *ThingSeek: A crawler and search engine for the Internet of Things*. in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 2016.

66. Uteuov, A., A. Kalyuzhnaya, and A. Boukhanovsky, *The cities weather forecasting by crowdsourced atmospheric data.* Procedia Computer Science, 2019. 156: p. 347-356.

67. Razzaq, A. and X. Yang, *Digital finance and green growth in China: Appraising inclusive digital finance using Web crawler technology and big data.* Technological Forecasting and Social Change, 2023. 188: p. 122262.

68. Yi, J., et al., *Analysis of Stock Market Public Opinion Based on Web Crawler and Deep Learning Technologies Including 1DCNN and LSTM.* Arabian Journal for Science and Engineering, 2023. 48(8): p. 9941-9962.

69. Raicu, I., *Financial banking dataset for supervised machine learning classification.* Informatica Economica, 2019. 23(1): p. 37-49.

70. Onyenwe, I., et al., *Developing products update-alert system for E-commerce Websites users using HTML data and Web scraping technique.* arXiv preprint arXiv:2109.00656, 2021.

71. Yang, D., *A comparative study of open source crawlers based on robustness and scalability testing on e-commerce Websites*. 2021, Chiang Mai: Graduate School, Chiang Mai University.

72. Shi, Z., M. Shi, and W. Lin. *The implementation of crawling news page based on incremental Web crawler*. in *2016 4th Intl Conf on Applied Computing and Information Technology/3rd Intl Conf on Computational Science/Intelligence and Applied Informatics/1st Intl Conf on Big Data, Cloud Computing, Data Science & Engineering (ACIT-CSII-BCD)*. 2016. IEEE.

73. Lu, M., et al. *The design and implementation of configurable news collection system based on Web crawler*. in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*. 2017. IEEE.

74. García-Mendoza, C.-V. and O. Juárez Gambino. *Web Crawler and Classifier for News Articles*. in *Mexican International Conference on Artificial Intelligence*. 2022. Springer.

75. *World Internet Usage And Population Statistics 2023 Year Estimates*. 2022 [cited 2023 21.07.2023]; Available from: https://www.internetworldstats.com/stats.htm.

76. Kausar, M.A., V. Dhaka, and S.K. Singh, *Web crawler: a review.* International Journal of Computer Applications, 2013. 63(2): p. 31-36.

77. Pavalam, S., et al., *A survey of Web crawler algorithms.* International Journal of Computer Science Issues (IJCSI), 2011. 8(6): p. 309.

78. Menczer, F., et al. *Evaluating topic-driven Web crawlers*. in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. 2001.

79. Arasu, A., et al., *Searching the Web.* ACM Transactions on Internet Technology (TOIT), 2001. 1(1): p. 2-43.

80. Heydon, A. and M. Najork, *Mercator: A scalable, extensible Web crawler.* World Wide Web, 1999. 2(4): p. 219-229.

81. Page, L., et al., *The PageRank citation ranking: Bringing order to the Web*. 1999, Stanford InfoLab.

82. Chakrabarti, S., M. Berg, and B. Dom, *Focused crawling: A new approach to topic-specific Web resource discovery.* Computer Networks, 2000. 31: p. 1623-1640.

83. Gupta, A. and P. Anand, *Focused Web crawlers and its approaches*. 2015. 619-622.

84. Batsakis, S., E.G.M. Petrakis, and E. Milios, *Improving the performance of focused Web crawlers.* Data & Knowledge Engineering, 2009. 68(10): p. 1001-1013.

85. Safran, M.S., A. Althagafi, and D. Che. *Improving Relevance Prediction for Focused Web Crawlers*. in *2012 IEEE/ACIS 11th International Conference on Computer and Information Science*. 2012.

86. Agre, G.H. and N.V. Mahajan. *Keyword focused Web crawler*. in *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*. 2015.

87. Age, S., et al., *A Self Adaptive Semantic Focused Web Crawler*. International Journal of Research In Science & Engineering, 2017. 1(6): p. 74-79.

88. Kumar, M., et al., *Keyword query based focused Web crawler.* Procedia Computer Science, 2018. 125: p. 584-590.

89. Mali, S. and B.B. Meshram. *Focused Web crawler with revisit policy*.

90. Safran, M.S., A. Althagafi, and D. Che. *Improving relevance prediction for focused Web crawlers*. IEEE.

91. Taylan, D., et al. *Intelligent focused crawler: learning which links to crawl*. IEEE.

92. Safran, M.S., A. Althagafi, and D. Che. *Improving relevance prediction for focused Web crawlers*. 2012. IEEE.

93. Gruber, T.R., *A translation approach to portable ontology specifications.* Knowledge acquisition, 1993. 5(2): p. 199-220.

94. Mukhopadhyay, D., A. Biswas, and S. Sinha. *A New Approach to Design Domain Specific Ontology Based Web Crawler*. in *10th International Conference on Information Technology (ICIT 2007)*. 2007.

95. Ehrig, M. and A. Maedche. *Ontology-focused crawling of Web documents*.

96. Agre, G. and S. Dongre, *A keyword focused Web crawler using domain engineering and ontology.* International Journal of Advanced Research in Computer and Communication Engineering, 2015. 4(3): p. 463-465.

97. Du, Y., et al., *An approach for selecting seed URLs of focused crawler based on user-interest ontology.* Applied Soft Computing, 2014. 14: p. 663-676.

98. Yu, Y.B., et al., *A survey about algorithms utilized by focused Web crawler*. Journal of Electronic Science and Technology, 2018. 16: p. 129-138.

99. Cai, D., et al., *Vips: a vision-based page segmentation algorithm.* 2003.

100. Wu, J. and K. Aberer, *Using SiteRank for Decentralized Computation of Web Document Ranking*. Vol. 3137. 2004.

101. Kohlschütter, C. and W. Nejdl, *A densitometric approach to Web page segmentation*, in *Proceedings of the 17th ACM conference on Information and knowledge management*. 2008, Association for Computing Machinery: Napa Valley, California, USA. p. 1173–1182.

102. Cho, J. and H. Garcia-Molina, *Estimating frequency of change.* ACM Transactions on Internet Technology (TOIT), 2003. 3(3): p. 256-290.

103. Sharma, S. and P. Gupta. *The anatomy of Web crawlers*. in *International Conference on Computing, Communication & Automation*. 2015.

104. Singh, M. and B. Varnica, *Web crawler: Extracting the Web data.* International Journal of Computer Trends and Technology, 2014. 13(3): p. 132-137.

105. Gupta, A. and A. Dixit, *A novel user trend-based priority assigner and URL scheduler for dynamic incremental crawling.* Concurrency and Computation: Practice and Experience, 2021. n/a(n/a): p. e6555.

106. Pavai, G. and T.V. Geetha, *Improving the freshness of the search engines by a probabilistic approach based incremental crawler.* Information Systems Frontiers, 2017. 19(5): p. 1013-1028.

107. Tan, Q. and P. Mitra, *Clustering-based incremental Web crawling.* ACM Transactions on Information Systems (TOIS), 2010. 28(4): p. 1-27.

108. Nagar, Y. and N. Singhal, *A users search history based approach to manage revisit frequency of an Incremental Crawler.* International Journal of Computer Applications, 2013. 63(3).

109. Kumar, M., R. Bhatia, and D. Rattan, *A survey of Web crawlers for information retrieval.* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2017. 7(6): p. e1218.

110. Zerfos, P., J. Cho, and A. Ntoulas. *Downloading textual hidden Web content through keyword queries*. in *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'05)*. 2005.

111. Kaur, S. and G. Geetha, *SIMHAR - Smart Distributed Web Crawler for the Hidden Web Using SIM+Hash and Redis Server.* IEEE Access, 2020. 8: p. 117582-117592.

112. Gupta, S. and K.K. Bhatia. *HiCrawl: A Hidden Web Crawler for Medical Domain*. in *2013 International Symposium on Computational and Business Intelligence*. 2013.

113. Bhatia, K.K., A.K. Sharma, and R. Madaan. *AKSHR: A novel framework for a Domain-specific Hidden Web Crawler*. in *2010 First International Conference On Parallel, Distributed and Grid Computing (PDGC 2010)*. 2010.

114. Raghavan, S. and H. Garcia-Molina, *Crawling the hidden Web*. 2000, Stanford.

115. Kumar, M. and R. Bhatia. *Design of a mobile Web crawler for hidden Web*. in *2016 3rd International Conference on Recent Advances in Information Technology (RAIT)*. 2016.

116. Anbukodi, S. and K.M. Manickam. *Reducing Web crawler overhead using mobile crawler*. in *2011 International Conference on Emerging Trends in Electrical and Computer Technology*. 2011.

117. Nath, R. and S. Bal, *A novel mobile crawler system based on filtering off non-modified pages for reducing load on the network.* Int. Arab J. Inf. Technol., 2011. 8(3): p. 272-279.

118. Deshmukh, S. and K. Vishwakarma. *A Survey on Crawlers used in developing Search Engine*. in *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*. 2021.

119. Shkapenyuk, V. and T. Suel. *Design and implementation of a high-performance distributed Web crawler*. 2002. IEEE.

120. Cai, J.F. and H. Zhang, *Dis-Dyn Crawler: A Distributed Crawler for Dynamic Web Page*, in *Proceedıngs Of The 4th Internatıonal Conference On Mechatronıcs, Materıals, Chemıstry And Computer Engıneerıng 2015 (ICMMCCE 2015)*. 2015. p. 2623-2626.

121. Quoc, D.L., et al. *UniCrawl: A Practical Geographically Distributed Web Crawler*. in *2015 IEEE 8th International Conference on Cloud Computing*. 2015.

122. Pu, Q. *The Design and Implementation of a High-Efficiency Distributed Web Crawler*. in *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*. 2016.

123. Liu, X.X. and Z.P. Jin, *ChainMR Crawler: A Distributed Vertical Crawler Based on MapReduce*, in *Securıty, Prıvacy And Anonymıty In Computatıon, Communıcatıon And Storage (SPACCS 2016)*. 2016. p. 33-39.

124. Boldi, P., et al., *Ubicrawler: A scalable fully distributed Web crawler*. Software: Practice and Experience, 2004. 34(8): p. 711-726.

125. Bal, S.K. and G. Geetha. *Smart distributed Web crawler*. in *2016 International Conference on Information Communication and Embedded Systems (ICICES)*. 2016.

126. Castillo, C., *Effective Web crawling.* SIGIR Forum, 2005. 39(1): p. 55–56.

127. Zhang, X. and K.P. Chow, *A Framework for Dark Web Threat Intelligence Analysis.* International Journal of Digital Crime and Forensics (IJDCF), 2018. 10(4): p. 108-117.

128. Henzinger, M.R., *Algorithmic challenges in Web search engines.* Internet Mathematics, 2004. 1(1): p. 115-123.

129. Stergiou, S. and K. Tsioutsiouliklis. *Set cover at Web scale*. 2015.

130. Kleinberg, J.M. *Authoritative sources in a hyperlinked environment*. 1998. Citeseer.

131. Daneshpajouh, S., M.M. Nasiri, and M. Ghodsi. *A Fast Community Based Algorithm for Generating Web Crawler Seeds Set*. in *WEBIST (2)*. 2008.

132. Zheng, S., P. Dmitriev, and C.L. Giles, *Graph-based seed selection for Web-scale crawlers*, in *Proceedings of the 18th ACM conference on Information and knowledge management*. 2009, Association for Computing Machinery: Hong Kong, China. p. 1967–1970.

133. Nwala, A.C., M.C. Weigle, and M.L. Nelson, *Garbage, Glitter, or Gold: Assigning Multi-dimensional Quality Scores to Social Media Seeds for Web Archive Collections*. 2021, Cornell University Library, arXiv.org: Ithaca.

134. Ganguly, B. and R. Sheikh, *A review of focused Web crawling strategies.* International Journal of Advanced Computer Research, 2012. 2(4): p. 261.

135. Shamrat, F.J.M., et al., *An effective implementation of Web crawling technology to retrieve data from the world wide Web (www).* International Journal of Scientific & Technology Research, 2020. 9(01): p. 1252-1256.

136. Jiang, L. and H. Zhang. *Multi-agent based individual Web spider system*. in *2010 World Automation Congress*. 2010. IEEE.

137. Chan, S.-B. and H. Yamana. *The method of improving the specific language focused crawler*. in *CIPS-SIGHAN Joint Conference on Chinese Language Processing*. 2010.

138. Menczer, F. and A.E. Monge, *Scalable Web search by adaptive online agents: An infospiders case study*, in *Intelligent Information Agents*. 1999, Springer. p. 323-347.

139. Priyatam, P.N., et al., *Seed selection for domain-specific search*, in *Proceedings of the 23rd International Conference on World Wide Web*. 2014, Association for Computing Machinery: Seoul, Korea. p. 923–928.

140. Sanagavarapu, L., et al., *Fine Grained Approach for Domain Specific Seed URL Extraction*. 2018.

141. Sanagavarapu, L.M., S. Sarangi, and Y.R. Reddy. *ABC Algorithm for URL Extraction*. in *ICWE Workshops*. 2017.

142. İnternet: Alexa. *The top 500 sites on the Web*. 2021 9.12. [cited 2021 9:12:2021]; Available from: https://www.alexa.com/topsites/countries.

143. Alderratia, N. and M. Elsheh. *Using Web Pages Dynamicity to Prioritise Web Crawling*. in *Proceedings of the 2019 2nd International Conference on Machine Learning and Machine Intelligence*. 2019.

144. Cho, J., H. Garcia-Molina, and L. Page, *Efficient crawling through URL ordering.* Computer networks and ISDN systems, 1998. 30(1-7): p. 161-172.

145. Prakash, J. and R. Kumar, *Web Crawling through Shark-Search using PageRank.* Procedia Computer Science, 2015. 48: p. 210-216.

146. Cao, L., et al., *Rankcompete: Simultaneous ranking and clustering of information networks.* Neurocomputing, 2012. 95: p. 98-104.

147. Najork, M. and J.L. Wiener, *Breadth-first crawling yields high-quality pages*, in *Proceedings of the 10th international conference on World Wide Web*. 2001, Association for Computing Machinery: Hong Kong, Hong Kong. p. 114–118.

148. Gupta, D. and D. Singh. *User preference based page ranking algorithm*. in *2016 International Conference on Computing, Communication and Automation (ICCCA)*. 2016.

149. Alhaidari, F., S. Alwarthan, and A. Alamoudi. *User Preference Based Weighted Page Ranking Algorithm*. in *2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)*. 2020.

150. Baker, M. and M. Akcayol, *Priority Queue Based Estimation of Importance of Web Pages for Web Crawlers.* International Journal of Computer Electrical Engineering, 2017. 9: p. 330-342.

151. İnternet: *World Wide Web Size.* 14.06.2021. Available from: https://www.worldwideWebsize.com/

152. Burkov, A. and B. Chaib-draa, *Effective learning in the presence of adaptive counterparts.* Journal of Algorithms, 2009. 64(4): p. 127-138.

153. Hansun, S. *A new approach of moving average method in time series analysis*. in *2013 Conference on New Media Studies (CoNMedia)*. 2013.

154. Zuo, X.L., et al. *Research and Implementation of Improved Real-Time Crawler Modeling*. in *Applied Mechanics and Materials*. 2013. Trans Tech Publ.

155. İnternet: Semrush. *Most Visited Websites in Turkey* 2023 [cited 2023 12/03/2023]; Available from: https://www.semrush.com/Website/top/turkey/all/.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : ALANOĞLU, Zülfü

Uyruğu : T.C.

Doğum tarihi ve yeri :

Medeni hali :

Telefon :

Faks :

e-mail :

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Doktora	Gazi Üniversitesi / Bilişim Ens. Bilişim Sis.	Devam ediyor
Yüksek lisans	Hatay Mustafa Kemal Üniversitesi / Bilgisayar Mühendisliği	2017
Lisans	Fırat Üniversitesi / Bilgisayar Öğretmenliği	2006
Lise	Elazığ Hulusi Sayın Lisesi	2001

İş Deneyimi

Yıl	Yer	Görev
2013-Halen	Hatay Mustafa Kemal Üniversitesi	Öğretim Görevlisi
2016-2013	Milli Eğitim Bakanlığı	Öğretmen

Yabancı Dil

İngilizce, ….

Yayınlar

1. Z. Alanoğlu and M. Akçayol , "Web Tarayıcıları için Etkili Tohum URL Seçimi ve Kapsam Genişletme Algoritması", International Journal of Advances in Engineering and Pure Sciences, vol. 35, no. 1, pp. 27-38, Mar. 2023, doi:10.7240/jeps.1174193

2. Z. Alanoğlu and M. A. Akcayol , "Web Tarayıcılarında Tohum URL Seçimi ve Performans Analizi: Kapsamlı Bir İnceleme", Düzce Üniversitesi Bilim ve Teknoloji Dergisi, vol. 11, no. 3, pp. 1399-1423, Jul. 2023, doi:10.29130/dubited.1097123

3. Z. Alanoğlu and M. A. Akcayol , “EMACrawler: Web search engine database freshness optimization” Journal of Polytechnic, (Accepted).

4. Sosyal ağ analizinde yapay zeka yaklaşımları
   Editörler: Parlar T., Esen F.S.
   Bölüm: Sosyal ağ uygulamalarında veri toplama, indeksleme ve içerik özgünlüğü Alanoğlu Z., Akcayol M.A., pp.75-90.
   Nobel Akademik Yayıncılık, ISBN: 978-625-427-965-2, Mart 2023.

5. Kutlu, Y., Alanoglu, Z., Gökçen, A., & Yeniad, M. (2019). Raspberry Pi Based Intelligent Robot that Recognizes and Places Puzzle Objects. *ArXiv, abs/2101.12584*.

GAZİLİ OLMAK AYRICALIKTIR.