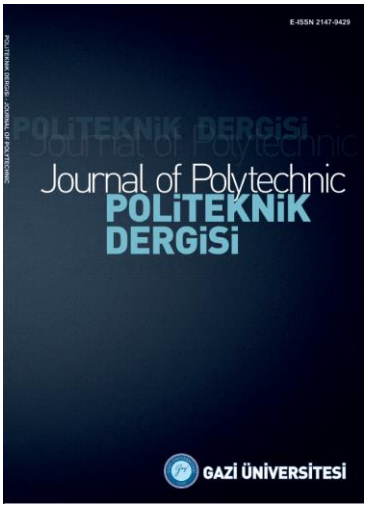


POLİTEKNİK DERGİSİ

JOURNAL of POLYTECHNIC

ISSN: 1302-0900 (PRINT), ISSN: 2147-9429 (ONLINE)
URL: http://dergipark.org.tr/politeknik

EMACrawler: web arama motoru veritabanı tazeliği optimizasyonu

EMACrawler: web search engine database freshness optimization

***Yazar(lar) (Author(s))**: Zülfü ALANOĞLU[1], M. Ali AKCAYOL[2]*

ORCID[1]: 0000-0001-9710-5658

ORCID[2]: 0000-0002-6615-1237

To cite to this article: Alanoğlu A. ve Akcayol M. A., "EMACrawler: Web Arama Motoru Veritabanı Tazeliği Optimizasyonu", ***Journal of Polytechnic***, 27(6): 2201-2214, (2024).

Bu makaleye şu şekilde atıfta bulunabilirsiniz: Alanoğlu A. ve Akcayol M. A., "EMACrawler: Web Arama Motoru Veritabanı Tazeliği Optimizasyonu", ***Politeknik Dergisi***, 27(6): 2201-2214, (2024).

Erişim linki (To link to this article): http://dergipark.org.tr/politeknik/archive

DOI: 10.2339/politeknik.1347054

EMACrawler: Web Arama Motoru Veritabanı Tazeliği Optimizasyonu

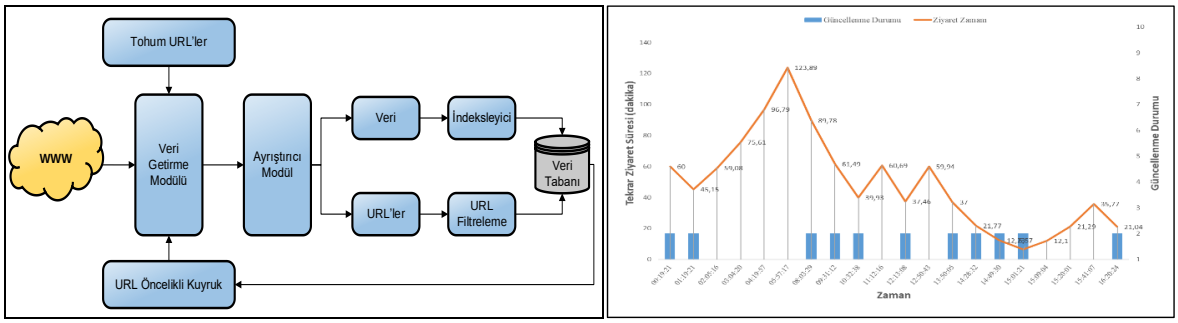
EMACrawler: Web Search Engine Database Freshness Optimization

Önemli noktalar (Highlights)

- *Web tarayıcıları / Web crawlers*
- *Web verilerini indeksleme / Indexing web data*
- *Web sayfalarının tekrar ziyaret zamanlarının belirlenmesi / Determining the revisit times of web pages*

Grafik Özet (Graphical Abstract)

Bu çalışmada, web tarayıcısı tarafından veri toplanması, ayrıştırma, indeksleme, filtreleme, depolama amacıyla kullanılacak, ayrıca tekrar ziyaret sürelerini tahmin ederek veri tabanı tazeliğini koruyacak bir yöntem geliştirilmiştir. / In this study, a method has been developed that will be used by web browser for data collection, parsing, indexing, filtering, storage, and also to protect the freshness of the database by estimating revisit times.



Şekil. Önerilen mimari ve web sayfası tekrar ziyaret zamanı **/ Figure.** The proposed architecture and web page revisit time

Amaç (Aim)

Bu çalışmanın amacı, web tarayıcısı için veri güncelleme optimizasyonunu gerçekleştirmektir. / The aim of this study is to perform data update optimization of the web browser.

Tasarım ve Yöntem (Design & Methodology)

Bu çalışmada, web tarayıcı veri güncelliği optimizasyonu için EMA yöntemi önerilmiştir. / In this study, EMA method is proposed for web browser data update optimization.

Özgünlük (Originality)

Web tarayıcıların kullandığı verinin değiştiğinde alınmasını ve tazeliğini korumak için yöntem geliştirilip uygulanmıştır. Çalışmada elde edilen başarı oranı göz önüne alındığında uygulanabilirliği oldukça yüksektir. / A method has been developed and implemented to ensure that web browsers' data is retrieved when changed and to maintain its freshness. Considering the experimental results obtained in the study, its applicability is quite high.

Bulgular (Findings)

Deneysel sonuçlar önerilem EMA yönteminin başarılı ve uygulanabilir olduğu göstermiştir. The experimental results have shown that the proposed EMA method is successful and applicable.

Sonuç (Conclusion)

Önerilen yöntem ile yapılan deneysel çalışmalarda diğer yöntemlerden daha başarılı sonuçlar elde edilmiştir. Önerilen yöntemin web tarayıcı veri güncelliğinin korunmasında daha başarılı olduğu görülmüştür. / The experimental studies with the proposed method yielded more successful results than other methods. It has been observed that the proposed method is more successful in maintaining web browser data up-to-date.

Etik Standartların Beyanı (Declaration of Ethical Standards)

Bu makalenin yazar(lar)ı çalışmalarında kullandıkları materyal ve yöntemlerin etik kurul izni ve/veya yasal-özel bir izin gerektirmediğini beyan ederler. / The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

EMACrawler: Web Arama Motoru Veritabanı Tazeliği Optimizasyonu

Araştırma Makalesi / Research Article

Zülfü ALANOĞLU[1*], M. Ali AKCAYOL[2]

[1]Antakya Meslek Yüksek Okulu, Bilgisayar Teknolojileri Bölümü, Hatay Mustafa Kemal Üniversitesi, Türkiye
[2]Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Gazi Üniversitesi, Türkiye

(Geliş/Received : 21.08.2023 ; Kabul/Accepted : 15.11.2023 ; Erken Görünüm/Early View : 12.03.2024)

ÖZ

Günümüz bilgi ve teknoloji çağında arama motorları hayatımızın önemli bir parçası haline gelmiştir. Her ne kadar bilgiye erişimde ilk başvurulan arama motorları olsa da kullanıcılara sunulan içerikte eski ve gereksiz bilgiler yer almaktadır. Güncel verileri sağlamak açısından günümüzdeki arama motorları çoğunlukla istenen başarıyı sunamamaktadır. Web tarayıcılarının sunduğu verilerin güncelliğini sağlamak için tekrar ziyaret zamanının doğru tahmin edilmesi gerekmektedir. Bu çalışmada arama motorlarının performanslarını etkileyen en önemli özellik olan tekrar ziyaret zamanlarının belirlenmesi için üstel hareketli ortalamaya dayanan EMACrawler önerilmiştir. Önerilen yöntem kesinlik, toplam kapsama alanı ve verimlilik metrikleri kullanılarak test edilmiştir. EMACrawler'ın web sayfalarındaki güncel veriyi doğru tahmin zamanında ve hızlı bir şekilde elde ettiği görülmüştür. Yapılan deneysel çalışmaların sonucunda EMACrawler'ın güncel verilerin elde edilmesi ve tarayıcı veri tabanının tazeliğinin korunmasında diğer yöntemlerden daha başarılı olduğu görülmüştür.

Anahtar Kelimeler: Web tarayıcısı, güncelleme modülü, veri toplama, veri indeksleme.

EMACrawler: Web Search Engine Database Freshness Optimization

ABSTRACT

In today's information and technology age, search engines have become an important part of our lives. However, search engines are the first to be used to access information, old and unnecessary information is included in the content offered to users. Regarding providing up-to-date data, today's search engines often cannot offer the desired success. In order to keep the data presented by web browsers up-to-date, the time of return visits must be accurately estimated. In this study, EMACrawler based on exponential moving averages is proposed to determine the revisit times, which is the most important feature that affects the performance of search engines. The proposed method is tested using precision, total coverage, and efficiency metrics. It has been seen that EMACrawler obtains the current data on the web pages accurately and quickly. As a result of the experimental studies, it has been seen that EMACrawler is more successful than other methods in obtaining up-to-date data and maintaining the freshness of the browser database.

Keywords: Web crawler, update module, data collection, data indexing.

1. GİRİŞ (INTRODUCTION)

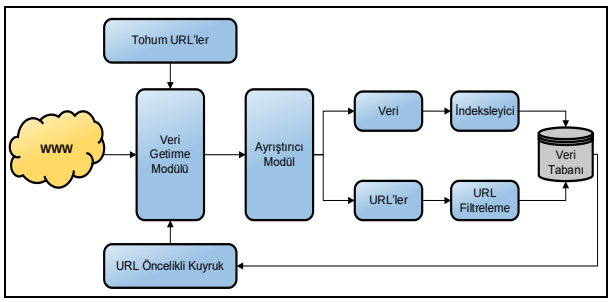
Günümüzde arama motorları yüz milyarlarca web sayfasını dizine eklemiştir [1]. Web, yapılandırılmış, yarı yapılandırılmış ve yapılandırılmamış veriler içeren, arama motorları aracılığı ile kullanıcılara sunulan birincil ve en büyük veri kaynağıdır [2, 3]. Arama motorlarının işleyişini kolaylaştıran en önemli bileşenler arasında web tarayıcıları vardır. Bu tarayıcılar, web sayfalarındaki verilerin ayrıştırılmasında, dizine eklenmesinde ve depolanmasında kritik bir rol oynar. Bir web tarayıcısının birincil işlevi, tohum URL seti olarak bilinen önceden belirlenmiş bir başlangıç URL grubundan başlayarak web sayfalarını tarayarak, indirerek ve ayrıştırarak sistematik olarak Web 'de gezinmeyi içerir. 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 verimli erişim için arama motorunun veri tabanında düzenlenir. Ayrıca, ayrıştırma işlemi sırasında web gezgini, ayrıştırılan sayfaya gömülü yeni URL'leri keşfeder. Bu yeni keşfedilen URL'ler daha sonra, gelecekte taranacakları sırayı belirleyen bir öncelik sırasına eklenir.

Tarayıcı, öncelik kuyruğundaki tüm URL'ler tükenene veya işlenene kadar bu işlemi tekrar tekrar izler. Şekil 1'de genel web tarayıcı mimarisi sunulmuştur.

Bir web tarayıcısının bileşenleri aşağıda açıklanmıştır.

Tohum URL'ler: Web tarayıcısının ilk taramaya başladığı URL'lerdir. Tohum URL'ler, tipik olarak, manuel olarak seçilen veya program kullanılarak üretilen, dikkatle seçilmiş URL'lerdir. Tohum URL'lerin kalitesi, tarama işleminin başlangıç kapsam ve yönünün belirlendiği için başarının artmasının başlıca nedenlerindendir.

*Sorumlu Yazar (Corresponding Author)
e-posta : zalanoglu@mku.edu.tr



Şekil 1. Genel web tarayıcı mimarisi (General web browser architecture)

Veri Getirme Modülü: Web sayfasının içeriğini taramaktan ve ardından içeriği işlemciye göndermekten sorumludur [4]. Tohum URL'lerden ve daha sonrasında öncelikli URL'lerden bilgi çıkarmak için kullanılır. URL'lerin barındırıldığı sunuculara http isteği gönderilir ve gelen veriyi ayrıştırıcı modüle gönderir.

Ayrıştırıcı Modül: Veri getirme modülünden gelen web sayfasını 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 depolanması için indekslenerek veri tabanında 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 depolanır. Veri tabanında veriler indekslenerek depolanmaktadır. 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 bir 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 bazı özelliklerine göre önceliklendirilir. Sıradaki her URL'e alaka düzeyi, tazeliği, önemi veya tarayıcının algoritması tarafından tanımlanan diğer kriterler gibi çeşitli faktörlere dayalı olarak bir öncelik değeri atanır.

Web tarayıcılarının yeni URL'leri keşfedip tarayarak dizini eklemenin yanı sıra değişen web sayfalarını tekrar taraması ve dizini güncel tutması gerekmektedir [5]. Tarayıcının hızla büyüyen ve değişen Web [6] ile yerel veri depolarında indekslenmiş web sayfaları koleksiyonlarını güncel tutması çok önemlidir. Bundan dolayı tarayıcılar Web sayfalarını düzenli olarak yeniden ziyaret eder ve indirir [7]. Sayfaların güncel olup olmadığının tespiti ve arama motorlarının veri depolarının güncellenmesi için birçok sayfa değişiklik tespit tekniği geliştirilmiştir. Web sayfalarının kendilerini özgü bir dinamizmi olduğu ve her birinin güncellenme süresinin farklı olduğu göz önüne alındığında tarayıcının hangi sayfaların ne sıklıkta ziyaret edilmesi gerektiğine 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.

2. LİTERATÜR ARAŞTIRMASI (RELATED WORK)

Web tarayıcılarının web sayfalarını tekrar ziyaret etme sıklıklarının belirlenmesinde çeşitli yöntemler kullanılmıştır. Bu yöntemlerden biri veri madenciliğidir. Tazelik açısından veri madenciliğini kullanmak daha iyi sonuçlar vermesine rağmen verileri eğitilmesi ve ilkelerin belirlenmesi gibi süreçlerden dolayı performans önemli ölçüde düşmektedir. Bulloat ve arkadaşları [8] 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. Sunulan 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 [9] çalışmalarında, veri madenciliğini kullanan ve çok iş parçacıklı IFCrawler adını verdikleri web tarayıcısını geliştirmişlerdir. 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.

Web tarayıcılarının sayfaları tekrar ziyaret etmelerinin temel sebebi veri depolarının güncelliğini korumaktır. Han ve arkadaşları [10] , arama motorlarında bulunan veri depolarının güncelliğini korumada kullanılan stratejileri gözden geçirmiş ve sezgisel tazelik metriğini analiz etmişlerdir. Önerilen yöntemde web sayfasının değişim oranının yanı sıra kullanıcıların bakış açılarını da hesaba katan ağırlıklı tazelik metriği önerilmiştir. Çalışmada ağırlıklı tazelik metriğinin 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 [11], arama motorunun kullandığı veri deposunun güncellerken web tarayıcısının gereksiz isteklerini ortadan kaldırmak için trafiğe uyarlanabilir optimum güncelleme şemasını (Traffic Adaptive Optimum updating Scheme -TAOS) önermişlerdir. Çalışmadaki diğer bir yenilik ise

güncellenen belgelerin sadece güncellenen kısımlarının arama motoru veri deposuna yüklenmesi olarak belirtilmiştir. Dolayısı ile web sunucuları üzerindeki yükü azaltan ve ağ bant genişliğini minimum düzeyde kullanan otonom bilgi işlem mimarisi tasarlanmıştır. Zhu ve arkadaşları [12], Web gibi karmaşık bir ortamda bant genişliğini dikkate alan yeni bir bant genişliği tahsis yaklaşımını 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 [13] çalışmalarında, web sayfalarını yeniden ziyaret etmek ve yerel veri deposunu güncel tutmak için etkili bir zamanlama politikası önermişlerdir. Önerilen yaklaşım üç aşamadan oluşmaktadır. İlk aşama lineer olmayan tamsayı programının çözünürlüğü ile belirlenen sayfa başına güncelleme sayısı belirlenir. 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 [14] çalışmalarında, 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 [15], web sayfalarının değişiklik sıklıkları ile ilgili özellikleri tanımlamış ve web sayfalarının güncelleme geçmişlerini bu özellikler ile ilişkilendirerek kümelemeye dayanan bir tarama algoritması tasarlamışlardır. Algoritma her kümeden bir web sayfasını indirir ve bu web sayfası son tarama döngüsünde belli oranda güncellenip güncellenmeme durumuna bağlı olarak kümede bulunan diğer web sayfalarını da tarayıp taramayacağına karar vermektedir. Radinsky ve Bennett [16] çalışmalarında, web sayfalarının güncellenme geçmişlerine ek olarak sayfaların içerikleri, sayfalardaki değişiklik dereceleri ve ilişkileri, web sayfasının diğer sayfalar ile ilişkisi ve değişiklik türlerinin benzerliklerini incelemiş ve analiz etmişlerdir. Ayrıca ilgili web sayfalarını belirlemek ve içerik benzerliği ölçümünde kullanılan çok sayıda benzerlik metriği sunmuşlardır. Li ve arkadaşları [17], çalışmalarında taranacak web sayfalarını kendi önceliklerinin ölçüsünü hesaplamanın farklı yolları ile 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 [18] çalışmalarında, XML dosyalarını kullanarak (SiteMap.XML) etki alanlarına göre site haritası oluşturup 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 sayfaları tüm alt etiketleri, meta verileri ve şekil açıklamaları ile birlikte ayrıştırarak web sayfalarının 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 sistemi 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 aşmak için web sayfalarının uzak sunuculardan indirilip analiz edilmesi yerine, verinin bulunduğu sunucularda mobil aracılar kullanarak analiz işlemlerini yapılmaktadır. Kausar ve arkadaşları [19], Java applet'leri 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 [20] çalışmalarında, arama motorunun veri deposunu güncel tutmak için belge dizinine dayalı sayfa değişikliği algılama tekniği ve mobil aracılar kullanarak dağıtılmış dizin oluşturma yöntemini tanımlamışlardır. Önerilen yöntemde verilerin analizi, indekslenmesi ve sıkıştırılması sunucular üzerinde yapılmakta ve ağ yükü en aza indirilmektedir. Gupta ve arkadaşları [21] çalışmalarında, web sayfalarının değişim tespitini bant genişliğini daha verimli kullanmak adına sunucu tarafında gerçekleştiren yeni bir sayfa değişikliği algılama tekniği kullanılmıştır. 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 gönderilir. Tersi durumda değişiklik miktarları eşik değerin üzerinde ise yeni sayfa gönderilerek eski sayfa ile değiştirilir. Ayrıca çalışmada eşik değerler web sayfalarının değişim sıklığına göre değişebileceği belirtilmiştir. Zhu ve arkadaşları [12] çalışmalarında web ortamındaki web sitelerinden verilerin zaman kısıtlaması ile toplanması gereken bir web tarayıcı sistemi için bant genişliği ayırma sorununu araştırmışlardır. Bant genişliği ayırma sorununu farklı durumlarda ve farklı objektif fonksiyonlar ile formüle

Çizelge 1. Web tarama yöntemlerinin karşılaştırılması (Comparison of web crawling methods)

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 Metrikleri	Orta	Artımlı Mobil
Olasılık Tabanlı	Orta	Orta	İlk ve Yeniden Tarama Aralığı, Kesinlik, Geri Çağırma	Orta	Odaklı

etmişlerdir. Sonuçlar, önerilen yaklaşımın zaman kısıtlamalarını tatmin edici bir şekilde karşıladığını ve çeşitli senaryolarda karşılık gelen hedefler açısından arzu edilen performanslara ulaştığını göstermiştir.

Web tarayıcılarının web sayfalarını tekrar ziyaret zamanlarını belirlemede kullanılan bir başka yaklaşım da olasılık tabanlı yaklaşımdır. Sethi [22] yaptığı çalışmada, web sayfalarını tekrar ziyaret sıklıklarında değişiklik olma olasılığına dayalı olarak çeşitli web sayfaları için farklı ilk ve yeniden tarama aralığını dikkate alan bir yöntem önermiştir. 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 metriği olarak kesinlik ve geri çağırma olmak üzere iki standart metrik kullanılmış ve geleneksel web tarayıcı ile karşılaştırılarak sonuçlar analiz edilmiştir. Yapılan deneysel sonuçlara göre önerilen yöntemin, güncel verileri alma ile ağ yükü arasında önemli bir denge sağlayabildiğini göstermektedir. Santos ve arkadaşları [23], sayfa değişim sıklığının olasılığını bulmak için bir optimizasyon tekniği olan genetik programlama çerçevesini sunmuşlardır. Tarayıcı performans normalleştirilmiş indirimli kümülatif kazanç ölçütü kullanılarak diğer olasılık tabanlı temel algoritmalar ile karşılaştırılıp analiz edilmiştir. Önerilen çalışmanın tek odak noktası artımlı tarayıcının performansını iyileştirmektir.

Yapılan çalışmalarda kullanılan 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'de gösterilmiştir.

3. VERİLERİN İNDEKSLENMESİ (INDEXING OF DATA)

Büyük verilerin yapısı ve hızlı bir şekilde çeşitlenip artması nedeni ile bu verilerin veri depolarına entegrasyonu ve indekslenmesi için geleneksel yöntemler yetersiz kalmaktadır. Bu nedenle verilerin indekslenmesinde özellikle verilerin saklanması, yapılandırılması, sorgu yanıt süresi ve sorgu doğruluğu gibi işlemlerin başarılı olması için etkili yöntemlerin kullanılması zorunludur. Web verileri gibi büyük verilerin indekslenmesindeki temel amaç veri sorgularında kullanılacak kıstaslara göre verileri ayrıştırmaktır. Ayrıştırılan her bir veri kullanıcı sorgularını karşılayan değeri içerir. Verileri indekslenmesiyle depolama işlemleri daha organize ve bilgi alımı da daha kolay olmaktadır. Fasolin ve arkadaşları çalışmalarında veri indeksleme yaklaşımlarını yapay zekâ ve yapay zekâ olmayan yaklaşımlar olarak gruplandırmışlardır [24]. Büyük verilerdeki bilinmeyen davranışların tespit edilmesi için yapay zekâ indeksleme yaklaşımları kullanılır ve bu yaklaşımlar modelleri gözlemleyerek benzer özelliklere sahip nesneler arasında ilişki kurarlar. Bu özelliklerinden dolayı yapay zekâsız indeksleme yaklaşımlarına göre avantajlı olsa da, verilerin eğitimi, tutarsız davranışlar vb. gibi durumlardan dolayı performans bakımından daha verimsiz olarak kabul edilirler [25]. Yapay zekâ olmayan yaklaşımlarda, veri öğelerinin anlamı ve aralarındaki ilişkinin indekslerin oluşumunda etkisi yoktur. Bunların yerine verideki 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.

3.1. Tersine Dizinleme (Inverted Indexing)

Tersine çevrilmiş indeksleme günümüzde özellikle büyük miktardaki verilerin saklanmasında arama motorları tarafından yaygın olarak kullanılan bir standart haline gelmiştir. Bir sözcük ile o sözcüğün bulunduğu belge arasında bir eşleme oluşturmak için kullanılmaktadır. Yapısının basit olması, kolay yönetilebilmesi ve daha az bellek tüketmesi gibi özelliklerinden dolayı yaygın olarak kullanılmaktadır. Temelde ileri ve ters çevrilmiş olmak üzere iki tür dizin

vardır. İleri dizinde her bir belge bir kelime listesi olarak saklanmaktadır. Tersine çevrilmiş dizinde ise kelimelerin verildiği belgelerin listesi saklanmaktadır [26]. En basit haliyle bir dizi belgeyi ileri ve tersine çevrilmiş dizinde saklamak Çizelge 2 ve 3' deki gibidir.

Çizelge 2: İleri Dizin (Advanced Index)

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

Çizelge 3: Tersine Dizin (Inverted Index)

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

Büyük veri işlemlerinde ters 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[27]. Her bir kayıt şunlardan oluşur;

A) Belge Tanımlayıcı d(t, i): t terimini içeren Ft belgelerinin i.inci sıra tanımlayıcısı

B) Belge içi frekans verisi f(t, i): t'nin d(t, i) de kaç kere tekrarlandığını kaydeden belge içi frekans verisi

C) d(t, i) belgesinde t'nin göründüğü kelime veya bayt konumlarını kaydeden f(t, i) tamsayılarının bir listesi

Önerilen tarayıcının zamanlama mekanizmasının başlatılabilmesi için öncelikle kullanılacak URL'lerin indekslenmesi gerekmektedir. Verilerin ilk defa web ortamından alınıp indekslenmesi adımları Algoritma 1'de gösterilmiştir.

Algoritma 1: Verilerin ilk defa web ortamından alınıp indekslenmesi (Indexing data from the web for the first time)

1. Başla
2. Dizine eklenecek URL'leri veri tabanından al
3. For tüm URL'ler:
4. Sayfayı indir ve ayrıştır
5. If Başlık ve Gövde metini alındı ise:
6. Web sitesi ID, URL ID ve sayfada kaç defa kullanıldığını tespit et
7. If Web Sitesi ID adında XML dosyası var mı
8. XML dosyasını oluştur ve URL ID adındaki etikete verileri yaz
9. Else
10. URL ID adındaki etikete verileri yaz
11. 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

Çizelge 4. Web sayfalarındaki verilerin indekslenmesi (Indexing data on web pages)

Sıra No	Kelime	Dokuman
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

sayfa içindeki tüm kelimeler ayrıştırılmış ve indekslenmiştir. Bu veriler veri tabanı tablosunda sırası ile ana sayfanın kimlik tanımı (ID- Identity Definition), URL 'in ID' si ve sayfada kelimenin kaç adet kullanıldığı verisi ile birlikte virgülle ayrılan değerler (CSV-Comma Separated Values) formatına uygun olarak indekslenmektedir. Çizelge 4'de kelimelerin indekslenmesi ile ilgili veri tabanı tablo görüntüsü gösterilmiştir.

Çizelge 4 'de görüldüğü gibi taranan web sayfalarından elde edilen her bir kelime ters indeks yaklaşımı kullanılarak indekslenmiştir. Dokuman sütununda bulunan ve virgül ile ayrılmış her bir üçlü gösterim A-B-C olsun;

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

4. VERİLERİN DEPOLANMASI (STORAGE OF DATA)

Web sayfalarından alınan veriler indekslenerek veri tabanında depolanmaktadır. 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 doküman nesne modeli (DOM-Document Object Model) yapısında oluşturulmuştur ve <URL> etiketlerindeki bulunan "id" değeri kullanılarak güncelleme ve silme işlemleri yapılabilmektedir. Sayfa

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>
```

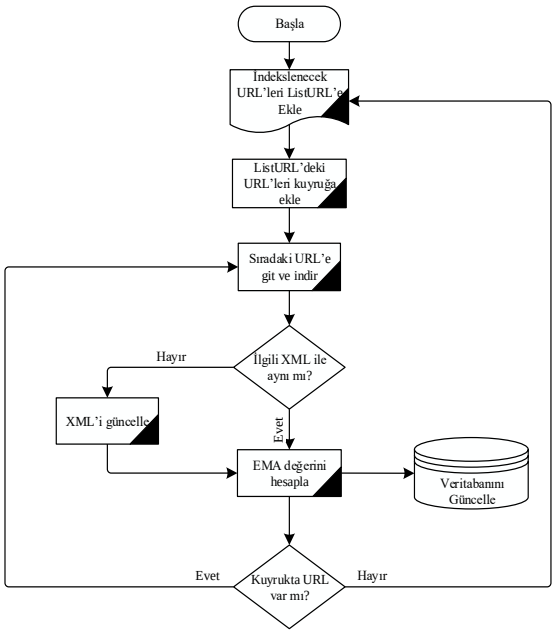
Şekil 2. Web sayfalarından alınan verilerin XML belgesine yazımı (Writing the data received from web pages to XML document)

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 2'de ID değeri 1 olan ana sayfanın ve bağlı URL'lerindeki verilerin bulunduğu 1.xml adına sahip dosyanın genel yapısı gösterilmiştir.

5. ÖNERİLEN GÜNCELLEME MODÜLÜ (RECOMMENDED UPDATE MODULE)

Web üzerinde [28]'e göre, şu anda dizine alınmış 4,15 milyar sayfa bulunmaktadır. Web'in bu derece büyümesine paralel olarak sayfaların taranması kadar sayfaların güncelliğini kontrol etmek de güçleşmektedir. Genel tarayıcılar belirli bir periyotta güncelleme yapmaktadır. Bu durum güncellenmeyen sayfaları tekrar taramak, sık güncellenen sayfalardaki verileri eşzamanlı alamamak vb. gibi birçok 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 zamanı, önceki ziyaretlere bakılarak tahmin edilebilir. Önceki ziyaretlerde sayfa sık değişmişse tekrar ziyaret sıklığı arttırılabilir ya da tersi durumda 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 akış şeması Şekil 3'de ve algoritması Algoritma 2'de gösterilmiştir.



Şekil 3. Önerilen güncelleme modülü algoritmasının akış şeması (Flowchart of the proposed update module algorithm)

Algoritma 2. Önerilen güncelleme modülünün algoritması (Algorithm of the proposed update module)

1. Başla
2. ListURL= Sonrakiindexzamanı şimdiki zamandan küçük olan URL'ler
3. Foreach(ListURL)
4. URL'e git ve indir
5. İlgili XML ile karşılaştır
6. If değişiklik var ise
7. XML i güncelle
8. EMA değerini hesapla
9. Veri Tabanını güncelle
10. Else
11. EMA değerini hesapla
12. Veri Tabanını Güncelle
13. Adım 2'ye git

Geliştirilen güncelleme modülü yönteminde sayfaları yeniden ziyaret etme zamanı, son dokuz ziyaret temel alınarak revize edilmiş 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 [29]. 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 [30]. Bir SMA denklem 1'deki gibi hesaplanır.

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

Burada P_M veri noktalarının değerini, 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 denklem 2'de gösterilmiştir.

$$SMA_{yeni} = \frac{P_M}{n} + SMA_{son} + \frac{P_{M-n}}{n} \quad (2)$$

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 denklem 3'de gösterilmiştir.

$$EMA_{yeni} = \left[V_b \left(\frac{y}{1+n}\right)\right] + EMA_{son}\left[1 - \left(\frac{y}{1+n}\right)\right] \quad (3)$$

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 5'deki gibidir.

Çizelge 5' 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 hesaplanır.

Her bir güncelleme işleminden sonra elde edilen kayıtlar saklanmaktadır. Çizelge 6'da veri tabanında rastgele seçilen 21 ID numaralı domaine sahip 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 6' da görüldüğü gibi güncellenme durumuna göre tekrar ziyaret süresi son güncellenme durumu daha etkili olacak şekilde üstel olarak artmakta ve azalmaktadır. Örnek olarak seçilen URL'in güncellenme durumuna göre tekrar ziyaret zamanı ve tekrar ziyaret süresini gösteren grafik Şekil 4' de gösterilmiştir.

5.1. Performans Değerlendirmesi (Performance Evaluation)

Önerilen EMA yöntemi ile karşılaştırma için gerçek zamanlı bir tarayıcı olan Real-Time [31] web tarayıcısı kullanılmıştır. Real-Time tarayıcı algoritması temel olarak, sayfaların en son değiştirilme zamanı ile bir sonraki değiştirilme zamanı arasındaki farkı tahmin etme üzerine kurulmuştur. Sayfaların yeniden ziyaret süresi, Real-Time tarayıcının sayfaları zamanında tarayıp tarayamayacağına göre belirlenir. Real-Time tarayıcısı, sayfaları güncellenir güncellenmez ziyaret edebilse de, tahminlerin doğruluğu vb. nedeniyle yeniden ziyaret zaman noktası, yenileme zaman noktasına eşit olmayabilir. Bu nedenle sayfaların tekrar ziyaret edilme süreleri denklem 4'deki gibi hesaplanır.

$$t_v = t_p + \Delta t + t_i \quad (4)$$

t_v : Bir sayfanın bir sonraki tekrar ziyaret zamanı
t_p : Bir sayfanın son ziyaret zaman noktası
Δt :Bir sayfanın son ziyaret zamanı ile yenilenme zamanı arasındaki zaman dilimi
t_i : Bir sayfanın yenileme süresi ile tekrar ziyaret süresi arasındaki ağırlıklı fark

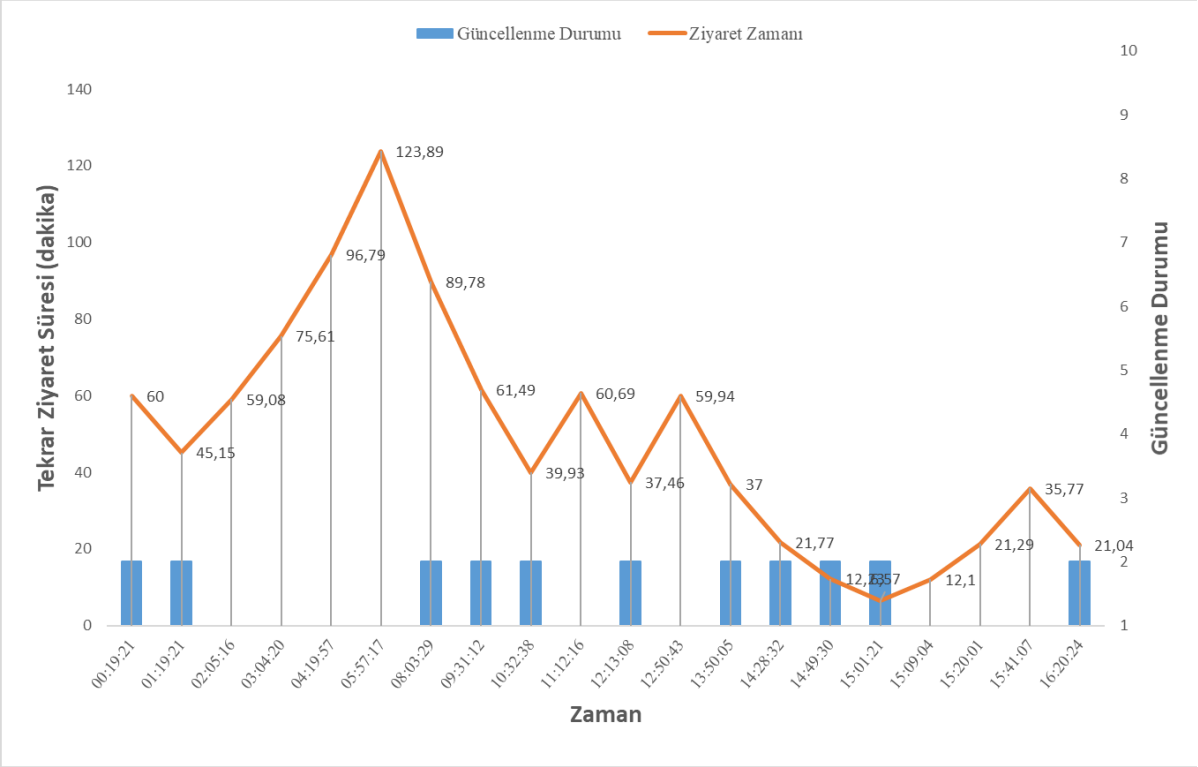
Önerilen web tarayıcı mekanizması MySQL veri tabanı kullanılarak Python programlama dili ile uygulanmıştır. Deneysel çalışmalar Windows 11 işletim sistemi, i5 2,90 GHz ve 16 GB RAM'e sahip iki bilgisayarda eş zamanlı gerçekleştirilmiştir.

Çizelge 5. Güncelleme modülü ile oluşturulan veri tabanı tablosu (Database table created with the update module)

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

Çizelge 6. Bir URL'in güncellenme durumuna göre tekrar ziyaret süresinin belirlenmesi (Determining the revisit time according to the update status of the URL)

URL ID	Dok. ID	Son İndeks Zamanı	SMA	SMA ort.	EMA (dakika)	Sonraki İndeks Zamanı
1041	21	6.06.2023 01:19	1,1,1,1,1,1,1,1,2	2	45,15	6.06.2023 02:05
1041	21	6.06.2023 02:05	1,1,1,1,1,1,1,2,1	1	59,08	6.06.2023 03:04
1041	21	6.06.2023 03:04	1,1,1,1,1,1,2,1,1	1	75,61	6.06.2023 04:19
1041	21	6.06.2023 04:19	1,1,1,1,1,2,1,1,1	1	96,79	6.06.2023 05:56
1041	21	6.06.2023 05:57	1,1,1,1,2,1,1,1,1	1	123,89	6.06.2023 08:00
1041	21	6.06.2023 08:03	1,1,1,2,1,1,1,1,2	2	89,78	6.06.2023 09:30
1041	21	6.06.2023 09:31	1,1,2,1,1,1,1,2,2	2	61,49	6.06.2023 10:31
1041	21	6.06.2023 10:32	1,1,2,1,1,1,2,2,2	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



Şekil 4. URL'in güncellenme durumuna göre ziyaret zamanı ve tekrar ziyaret süresi (Visit time and revisit time according to the update status of the URL)

5.2. Performans Metrikleri (Performance Metrics)

Karşılaştırılan EMA ve Real-Time tarayıcılarının performansları kesinlik, toplam kapsama oranı ve verimlilik olmak üzere 3 standart metrik kullanılarak ölçülmüştür. Kesinlik, önerilen tarayıcı tarafından yeniden taranan tüm sayfaların değişiklikle karşılaşan kısmı üzerinden hesaplanır [22]. Kesinlik (K), değişiklikle karşılaşılan sayfalar (D) ve tüm sayfalar (T) olmak üzere denklem 5'deki gibi hesaplanır.

$$K = \frac{D}{T} \tag{5}$$

Toplam kapsama oranı (TKO), taranan benzersiz kayıt sayısının veri tabanında bulunan tüm kayıtlara oranını olup denklem 6 ile gösterilmektedir.

$$TKO = \frac{Benzersiz\ kayıt\ sayısı}{Veri\ tabanındaki\ Tüm\ Kayıtlar} \tag{6}$$

Belirli bir zaman örneğinde (ti) bir veri tabanında N_c, artımlı tarayıcının tarayabileceği kayıt sayısını temsil etmektedir. N_s 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 [32]. Bu verilere dayanarak verimlilik denklem 7'da gösterilmiştir.

$$Verimlilik = \frac{N_s}{N_c} \tag{7}$$

5.3. Deneysel Yöntem (Experimental Method)

Deneysel çalışma aynı özelliklere sahip iki makinede aynı zaman aralığında gerçekleştirilmiştir. Tarayıcıların tarama işlemlerine başlamaları için kullanılan tohum URL setleri Sumrush [33] 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 7'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 aşağıdaki formülle hesaplanmıştır.

$$D = \frac{Değişiklik\ Sayısı}{Ziyaret\ Sayısı} \tag{8}$$

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 8'de gösterilmiştir.

Çizelge 7. Çalışmada kullanılan web sayfaları ve URL sayıları (Web pages and URLs used in the study)

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

Çizelge 8. Dinamiklik durumuna göre URL sayısı (Number of URLs based on dynamism)

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

Çizelge 8'de görüldüğü gibi her iki tarayıcı da dinamik olmayan sayfaları birbirine yakında değerlerde tespit etmiştir. Az dinamik olan sayfalar Real-Time tarayıcısı tarafından daha fazla kategorize edilirken orta ve yüksek dinamik sayfalar EMA tarayıcısı tarafından daha fazla işaretlenmiştir. Bu dinamiklik durumlarının kategorilere göre dağılımı Çizelge 9'da gösterilmiştir.

Çizelge 9. Her bir kategorinin dinamiklik durumlarına göre URL sayıları (Her kategorinin dinamikliğine göre URL sayısı)

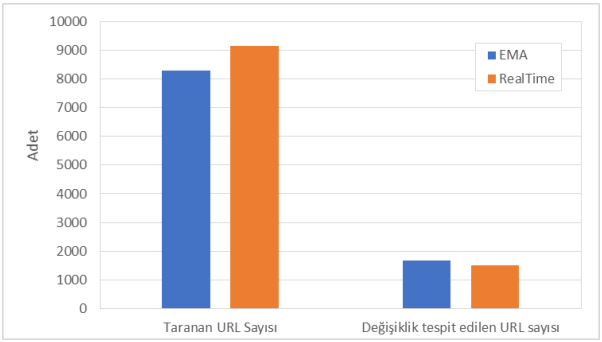
Kategoriler	Dinamiklik Durumu	URL Sayısı	
		EMA	Real-Time
E-ticaret ve Perakende	Dinamik olmayan	139	142
	Az	21	17
	Orta	17	24
	Yüksek	6	2
Yolculuk ve Turizm	Dinamik olmayan	200	208
	Az	16	17
	Orta	22	14
	Yüksek	6	6
Eğitim	Dinamik olmayan	106	104
	Az	2	34
	Orta	55	24
	Yüksek	0	0
Finans	Dinamik olmayan	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

EMA ve Real-Time tarayıcıları eş zamanlı olarak 12 saat boyunca 5 farklı kategoride seçilen 1046 farklı URL için çalıştırılmıştır. Her iki tarayıcı için yapılan tarama sonuçları Çizelge 10'da gösterilmiştir. Taranan ve değişiklik elde edilen toplam URL sayıları incelendiğinde önerilen EMA tarayıcısının belirtilen süre içerisinde yaptığı toplam tarama sayısı Real-Time tarayıcısına göre daha az iken değişiklik tespit edilen URL sayısı daha fazladır. EMA tarayıcısının, daha az performans ile daha verimli bir tarama gerçekleştirildiği görülmektedir.

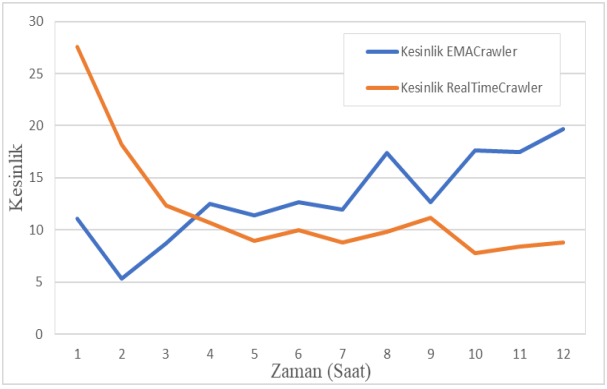
Çizelge 10. URL'lerin taranması sonuçları ve performans metrikleri (URLs crawling results and performance metrics)

	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	100,00	100,00	2,11	2,28
2	965	1003	992	1105	56	190	936	915	5,35	18,16	92,26	95,89	2,14	2,32
3	588	806	659	884	91	129	568	755	8,70	12,33	56,21	77,06	2,93	3,37
4	627	552	759	736	131	112	628	624	12,52	10,71	59,94	52,77	2,88	2,73
5	789	434	926	759	119	94	807	665	11,38	8,99	75,43	41,49	2,48	1,99
6	305	306	455	671	132	104	323	567	12,62	9,94	29,16	29,25	4,86	4,77
7	648	456	769	650	125	92	644	558	11,95	8,80	61,95	43,59	2,81	2,46
8	428	396	522	586	182	103	340	483	17,40	9,85	40,92	37,86	3,87	3,61
9	372	378	466	511	132	117	334	394	12,62	11,19	35,56	36,14	4,17	4,14
10	728	485	828	632	184	81	644	551	17,59	7,74	69,60	46,37	2,69	2,21
11	154	203	297	572	183	88	114	484	17,50	8,41	14,72	19,41	8,98	8,68
12	374	195	535	667	206	92	329	575	19,69	8,80	35,76	18,64	4,35	3,56

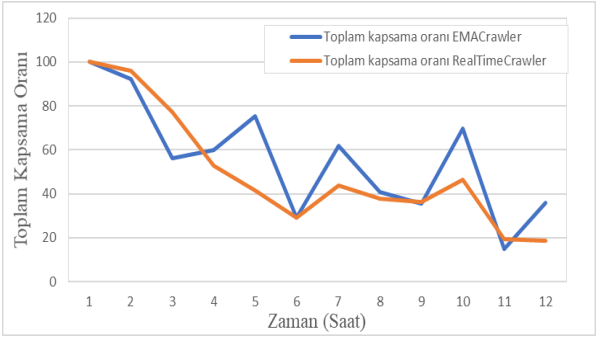
Çizelge 10'da gösterilen tarama sonucunda elde edilen sonuçlar kullanılarak kesinlik, toplam kapsama oranı ve verimlilik metrikleri hesaplanmıştır. Elde edilen sonuçlar Şekil 5, Şekil 6, Şekil 7 ve Şekil 8'de grafiksel olarak gösterilmiştir.



Şekil 5. Taranan ve değişiklik elde edilen URL sayıları (Number of URLs crawled and changes obtained)



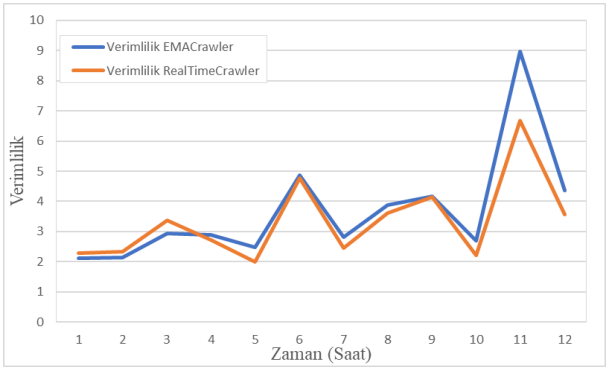
Şekil 6. Tarayıcıların kesinlik metriklerinin karşılaştırılması (Comparison of accuracy metrics of crawlers)



Şekil 7. Tarayıcıların toplam kapsama oranlarının karşılaştırılması (Comparison of total coverage rates of crawlers)

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.

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 8. Tarayıcıların verimliliklerinin karşılaştırılması (Comparison of the efficiency of crawlers)

Genel ve artımlı web tarayıcılarında en önemli performans metriklerinden biri de verimliliktir. Çizelge 8 incelendiğinde önceki performans metriklerine paralel olarak taramanın başlarında EMA tarayıcısının verimliliği Real-Time tarayıcısına göre başlangıçta daha az iken 3. saatten sonra daha iyi sonuçları elde etmiştir.

6. SONUÇ (CONCLUSION)

Bu çalışmada arama motorlarının performanslarını etkileyen en önemli özellik olan tekrar ziyaret zamanlarının belirlenmesi için üstel hareketli ortalamaya dayanan EMACrawler önerilmiştir. Önerilen yöntem kesinlik, toplam kapsama alanı ve verimlilik metrikleri kullanılarak diğer yöntemlerle karşılaştırılmıştır. EMACrawler'ın web sayfalarındaki verinin güncellenme zamanını doğru tahmin ederek hem web trafiğini azaltmak hem de tarayıcı veri tazeliğini sağlamada daha başarılı olduğu görülmüştür. Böylelikle, web tarayıcılardaki veri kalitesinin en önemli bileşeni olan veri tazeliğinin sağlanması ile birlikte veri kalitesinin iyileştirilmesi sağlanmıştır.

Gelecekte, metin tabanlı taramaya ek olarak multimedya ve doküman tabanlı taramalar yapan ve özellikle derin Web'i tarayabilen tarayıcıların tasarlanması yararlı olacaktır. Böylelikle Web'in önemli bir kısmındaki verilere ulaşılacak ve tüm veri türlerinde kullanıcıların talep ettikleri verilerin kalitesinin iyileştirilmesi sağlanabilecektir.

TEŞEKKÜR (ACKNOWLEDGEMENT)

Bu çalışma, TÜBİTAK tarafından BİDEB-2244 Sanayi Doktora Programı kapsamında 118C127 numara ile desteklenen "İnternette Heterojen Veri Kaynaklarından Veri Toplanması, Doğrulanması ve Sorgulanması" başlıklı projenin bir parçasıdır. Sağladığı destek için TÜBİTAK'a teşekkür ederiz.

ETİK STANDARTLARIN BEYANI (DECLARATION OF ETHICAL STANDARDS)

Bu makalenin yazar(lar)ı çalışmalarında kullandıkları materyal ve yöntemlerin etik kurul izni ve/veya yasal-özel bir izin gerektirmediğini beyan ederler.

YAZARLARIN KATKILARI (AUTHOR'S CONTRIBUTIONS)

Zülfü ALANOĞLU : Deneyleri yapmış, sonuçları analiz etmiş ve makalenin yazım işlemleri gerçekleştirmiştir.

M. Ali AKCAYOL : Sonuçları analiz etmiş, makalenin kontrol ve danışmanlık işlemini gerçekleştirmiştir.

ÇIKAR ÇATIŞMASI (CONFLICT OF INTEREST**)**

Bu çalışmada herhangi bir çıkar çatışması yoktur.

KAYNAKLAR (REFERENCES)

[1] Google,"How Google Search Works", ***www.google.com***, [Erişim Tarihi: 10/08/2022).

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

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

[4] Li K., Fei J., and Fan C., "Optimization and application of web crawler architecture," ***SPIE,*** 12506: 151-155, (2022).

[5] Patil T. A. and Chobe S., "Web Crawler for searching Deep web sites," in ***2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)***, Pune, India,1-5 (2017).

[6] Avrachenkov K., Borkar V., and Patil K., "Deep reinforcement learning for web crawling," in ***Seventh Indian Control Conference (ICC)***, Mumbai, India:201-206 (2021).

[7] Mallawaarachchi V., Meegahapola L., Madhushanka R., Heshan E., Meedeniya D., and Jayarathna S., "Change detection and notification of web pages: ***A survey," ACM Computing Surveys (CSUR)***, 1(53):1-35, (2020).

[8] Bullot H., Gupta S. K., and Mohania M. K., "A data-mining approach for optimizing performance of an incremental crawler," in ***Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)***, 13(17):610-615, (2003)

[9] Kharazmi S., Nejad A. F., and Abolhassani H., "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)***, London, UK,1-7, (2009).

[10] Jianchao H., Cercone N., and Xiaohua H., "A Weighted Freshness Metric for Maintaining Search Engine Local Repository," in I***EEE/WIC/ACM International Conference on Web Intelligence (WI'04),*** *Beijing, China,* 677-680, (2004).

[11] Amudhan V. and Thirupathi D., "Traffic Adaptive Optimum Updating Scheme for Search Engines," in ***2006 1st International Conference on Digital Information Management***, 6(6):395-403, (2007).

[12] Zhu W., Li Y., Li S., Xu Y., and Cui X., "Optimal bandwidth allocation for web crawler systems with time constraints," ***Journal of Ambient Intelligence and Humanized Computing***, 5(14):5279-5292, (2023).

[13] Souza C., Laber E., Valentim C., and Cardoso E., "A Polite Policy for Revisiting Web Pages," in ***2007 Latin American Web Conference (LA-WEB 2007)***, Santiago, Chile,128-135, (2007).

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

[15] Tan Q. and Mitra P., "Clustering-based incremental web crawling," ***ACM Trans. Inf. Syst.***,4(28):1-27, (2010).

[16] Radinsky K. and Bennett P. N., "Predicting content change on the web," presented at the ***Proceedings of the sixth ACM international conference on Web search and data mining***, Rome,415-424, (2013).

[17] Li H., Guo M., Cai L., and Yang Y., "An incremental update strategy in Deep Web," in ***2010 Sixth International Conference on Natural Computation***, Yantai, China, 131-134, (2010).

[18] Mor J., Rai D., and Kumar N., "An XML based Web Crawler with Page Revisit Policy and Updation in Local Repository of Search Engine," ***International Journal of Engineering & Technology***,7(3): 1119-1123, (2018).

[19] Kausar M. A., Nasar M., and Singh S. K., "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***, Kanjirapally, India, 1-6: (2013).

[20] Badawi M., Mohamed A., Hussein A., and Gheith M., "Maintaining the search engine freshness using mobile agent," ***Egyptian Informatics Journal***, 1(14):27-36, (2013)

[21] Gupta A., Dixit A., and Sharma A., "A Novel Web Page Change Detection Technique for Migrating Crawlers," In: ***Sensors and Image Processing: Proceedings of CSI***. Springer, Singapore, 49-57, (2018).

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

[23] Santos A. S. R., Carvalho C. R., Almeida J. M., Moura E. S. de, Silva A. S. da, and Ziviani N., "A genetic programming framework to schedule webpage updates," ***Information Retrieval Journal***, 1(18):73-94, (2015).

[24] Fasolin K. et al., "Efficient Execution of Conjunctive Complex Queries on Big Multimedia Databases," in ***2013 IEEE International Symposium on Multimedia***, Anaheim, CA, 536-543,(2013).

[25] Gani A., Siddiqa A., Shamshirband S., and Hanum F., "A survey on indexing techniques for big data: taxonomy and performance evaluation," ***Knowledge and Information Systems***, 2(46): 241-284 (2016).

[26] Shah S. and Shaikh A., "Hash based optimization for faster access to inverted index," in ***2016 International Conference on Inventive Computation Technologies (ICICT)***, Coimbatore, India,1-5,(2016).

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

[28] "World Wide Web Size", ***https://www.worldwidewebsize.com/*** [Erişim Tarihi : 18/8/2023].

[29] Burkov A. and Chaib-draa B., "Effective learning in the presence of adaptive counterparts," ***Journal of Algorithms***, 4(65):127-138, (2009).

[30] Hansun S., "A new approach of moving average method in time series analysis," in ***2013 Conference on New Media Studies (CoNMedia)***, Tangerang, Indonesia, 1-4, (2013).

[31] Zuo X. L., Wang W. Wang B., Y., and Zuo W. L., "Research and Implementation of Improved Real-Time Crawler Modeling," in ***Applied Mechanics and Materials***, vol. 312:791-795 (2013).

[32] Zerfos P., Cho J., and Ntoulas A., "Downloading textual hidden web content through keyword queries," in ***Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL '05***), Denver, CO:100-109, (2005).

[33] "Most Visited Websites in Turkey "***https://www.semrush.com/website/top/turkey/all/*** [Erişim Tarihi: 12/03/2023]