DAĞITIK WEB ÖNBELLEK SUNUCUSU TASARIMI VE GERÇEKLENMESİ

F. Erdoğan SEVİLGEN [1] **Kemal YILMAZ** [2]

Bilgisayar Mühendisliği Bölümü,
Mühendislik Fakültesi
Gebze Yüksek Teknoloji Enstitüsü, 41400, Gebze, Kocaeli

[1]e-posta: sevilgen@gyte.edu.tr [2] e-posta:kyilmaz@uekae.tubitak.gov.tr

Anahtar sözcükler:Dağıtık Web Ön Bellek Sunucular

ÖZET

Günümüzde Internet üzerinde bulunan ve ulaşılan bilgi miktarı çok büyüktür ve her geçen gün dramatik olarak artmaktadır. Bu bilgilere ulaşma hızını artırmak için bilinen metotlardan biri gerekli dosyaların bir çok kopyasını dünyanın değişik yerlerinde tutmaktır. Diğer bir metot ise web bilgilerinin daha sonra kullanmak için geçici olarak saklanmasını sağlayan web önbellek sunucularıdır. Web önbellek sunucular tek başına çalıştıkları gibi sunucuların kendi bilgi ve kaynaklarını başka sunucular ile paylaşabilmeklerini sağlayan dağıtık web önbellek sunucuları olarak da çalışabilmektedirler. Dağıtık web önbellek sunucuların tasarımda tam olarak dağıtık ve hiyerarşik yaklaşımlar ön plana çıkmaktadır. Her iki metodundan avantajları ve dezavantajları dikkate alınarak hibrit dağıtık web cache sunucu tasarımı ve gerçeklenmesi yapılmıştır.

1. GİRİŞ

Internet üzerinde iletilen bilgi miktarının çok hızlı artmasından dolayı, sitelere erişim yavaşlamaktadır. Bu iletişim hızını artırmak için kullanılan ilk metot ağ altyapısını güçlendirmektir, fakat bu maliyetleri yükselten bir metottur. Ağ bant genişliği daha verimli kullanmanın ve cevap süresini azaltmanın bir diğer metodu da web önbelleklemedir. Web önbellekleme, web bilgilerinin daha sonra kullanmak üzere geçici süre saklanmasıdır. Web önbellek sunucular tek başına çalışabildikleri gibi birbirleriyle yardımlaşarak dağıtık olarak da çalışabilmektedirler.

Dağıtık web önbellek sunucu mimarileri konusunda iki ana yaklaşım popülerdir. Birinci metot olan hiyerarşik metotta önbellek sunucular önbellek ağacı şeklinde yapılandırılmıştır. İstemciler sadece en alt seviyede bulunan sunuculara bağlanırlar. Eğer sunucu istenilen bilgiyi deposunda bulamazsa aynı seviyede bulunan diğer sunuculara gönderir, bu sunucular dan olumlu bir cevap dönmezse, aynı isteği ebeveyn sunucusuna yönlendirir. Bu işlem kök sunucuya kadar aynı şekilde devam eder, kök sunucu bilgiyi doğrudan kaynak sunucudan getirir ve önbellek sunuculara yönlendirir. Kullanıcının bağlı olduğu önbellek sunucu da bilgiyi depolar ve istemciye döner.

Diğer metot ise önbellek sunucularının tam manası ile dağıtık olarak çalışmasıdır. Böyle bir yapıda sunucular diğer sunucuların durumlarının bilmek zorundadırlar. Kullanıcı aradığı bilgiyi bağlı olduğu sunucuya sorar, eğer sunucu bu bilgiyi kendi deposunda bulamazsa kendisine yakın olan sunuculardan bu bilgiyi sorgular. Eğer sunucular belli bir zaman içinde bu bilgiyi sorguyu yapan sunucuya dönemezlerse, bilgi doğrudan kaynak sunucudan getirilir [1].

Her iki metodunda da dezvantajları vardır. Dağıtık yaklaşım ağ altyapısının fiziksel olarak varolan hiyerarşik yapısını göz ardı etmektedir, bu yüzden zaten varolan sistemin getirdiği avantajlarını kullanamamaktadır. Hiyeraşik yaklaşım ise eğer *d* seviye bir ağaç şeklinde tasarlanmışsa, önbellek ağacında bulunmayan bilgiyi getirmek için *d* bekleme süresine ihtiyacı vardır. Bu çalışmada, her iki metodun da avantajlarına sahip hibrit bir tasarım yapılmıştır. Ayrıca ağ ve CPU kullanımını önbellek sunucular arasında eşit dağıtmak için geliştirilen yeni yaklaşım üzerinde durulmuştur.

2. BENZER ÇALIŞMALAR

Tasarım konusunda yapılan çalışmalar sonucunda büyük ölçekli dağıtık önbellek sunucu sistemlerinin tasarımı için dört temel kriter ortaya çıkmıştır. Birincisi aranılan bilginin ulaşmak için gerekli olan hop sayısını minimize etmek, ikincisi çok fazla sayıda önbellek sunucuya kadar ölçeklenebilir olmak, üçüncüsü önbelleklerdeki başarısız aramaların sistemi yavaşlatmasını engellemek ve sonuncusu ise bilgileri kullanıcılara yakın yerde depolamaktır [1].

Dağıtık web önbellek mimariler arasından hiyerarşik yaklaşıma uygun olarak yapılan çalışmalardan Harvest [4], ve uyarlanabilir web önbellek

sunucularda sayılabilir [3]. Özet web önbellek [7] ve Cachemesh [2] mimarileri ise tam manası ile dağıtık çalışan mimari çeşitleri arasındadır.

Bu mimarilerden cachemesh ve uyarlanabilir web önbellek sistemlerine daha detaylı bakılırsa; Cachemesh sistemi önbellek sunucularının efektif bir şekilde yardımlaşabilmeleri için sunucular arasında önbellek yönlendirme sistemi kullanmaktadır. Bu sistemde, önbellek sunucular aralarında önbellek yönlendirme tablosu oluşturulur. Böylece her bir sunucu belirli sayıda web sitesi için gerekli bilgileri saklar. Kullanıcı istekleri uygun olan önbellek sunucusuna iletilir [2].

Uyarlanabilir web önbellekleme sisteminde ise popüler sayfalar kullanıcıya daha yakın yerlerde tutulurken diğerleri hiyerarşide daha yukarılarda tutuluyor. Popüler sayfalar sayfanın başarı oranına göre belirlenir. Buradaki problem ise her kullanıcının belli öncelikleri olduğu için herkesin popüler sitesinin farklı olması durumunda ne yapılacağı karar verilmesi gerekiyor. Popülarite oranı her zaman değişeceği için bunun hiyerarşideki önbellek sunuculara haber verilmesi gerekmektedir [3].

3. TASARIM

3.1 Mimari

Sistem tasarımında önbellek sunucuların birbirlerine çok uzak olmadığı kabul edildi, böylece önbellek sunucularının bilgilerini hızlı bir şekilde paylaşabilmeleri sağlanmaktadır. Eğer sistem birbirine çok uzak sunuculardan oluşursa, önbellek sunucuların bilgi paylaşımı yavaş olacaktır ve önbellek sunucuların cevap süresi eşik değerinden fazla olacağından birbirlerinin bilgilerinden faydalanamayacaklardır. Bu durumda bize tekil çalışan sunuculardan daha kötü performans verecektir, çünkü bekleme süresi ve ağ trafiği artacaktır.

Sisteminde iki tip önbellek sunucu vardır:

- Web erişimi olan sunucular: Bu sunucular bilgileri doğrudan kaynak sunucudan getirebilirler.
- Web erişimi olmayan sunucular: Bu sunucular bilgiyi doğrudan kaynak sunucudan getiremezler, kendilerine gelen isteği bağlı oldukları önbellek sunucuya yönlendirmeleri gerekmektedir.

Sunucular arasında böyle bir ayırımın olmasının avantajını bir örnek ile açıklayalım. A ve B önbellek sunucular aynı YAA (Yerel Alan Ağı) üzerinde ve B makinesi aynı zaman YAA için ağ geçidi (gateway) olsun. A bilgiyi kaynak sunucudan getirmek istediğinde istek B makinesi üzerinden geçmek zorundadır. Eğer A makinesi gelen isteği B makinesine yönlendirirse, B makinesi de fazla iş yapmadan aynı bilgiye sahip olur ve sistemin başarı oranı artar. Sistem içinde sadece web erişimi olmayan sunucular kullanılırsa Harvest önbellek sistemine benzer [4].

3.2 Önbellek Yerleştirme

Her önbellek sunucu sadece istemcilerin istedikleri bilgileri saklarlar. Bu yüzden 'pushing' ve 'prefetching' sistemde bulunmamaktadır. Aynı zamanda sistemde kaynakları verimli kullanmak için çok büyük bilgiler saklanmamaktadır [5]. Bunun sebebini şöyle açıklayabiliriz: S büyüklüğünde ve F sıklığında erişilen O bilgisi olduğunu düşünelim. Aynı zamanda o1 ve o2 olan iki küçük bilgi olsun ve bu bilgilerin büyüklüğü s1,s2 ve sıklıkları f1,f2 olsun. Ayrıca s1+s2=S olarak kabul edelim, eğer S eşik değerinden büyükse f1>F ve f2>F olma olasılığı çok büyüktür. Eğer O bilgisi depolanmışsa bunların yerine getirilecek olan küçük bilgilerin ihtiyacı olan ağ genişliği s1*f1+s2*f2 olur. Eğer O bilgisi depolanmamışsa O bilgisini getirmek için gerekli olan ağ genişliği S*F dir. Eğer f1>F ve f2>F olarak kabul edersek;

$$S*F = (s1+s2)*F = s1*F+s2*F$$

$$f1 > F \Rightarrow s1*f1 > s1 * F$$

$$f2 > F \Rightarrow s2*f2 > s2 * F$$

$$s1*f1 + s2*f2 > S*F$$

Buradan eşik değerinden büyük bilgilerin saklanmamasının daha uygun olduğu görülmektedir.

3.3 Önbellek Sunucu Çözümleme

Önbellek sunucudaki bütün adresler için en azından bir ipucu vardır. İpuçları bilginin kaynak adresini (URL) ve bilginin bulunduğu önbellek sunucu adresini içeren bir veri yapısıdır. İpuçları istenilen bilginin hangi sunucuda olabileceğini söyler, fakat hangi sunucuda olmayacağını söylemez. Bilgi önbellek sunucuda başarılı bir şekilde saklandığı takdirde sadece o URL ile ilgili bir ipucu oluşturulur. Bu yüzden HTTP hata mesajları ve saklanması uygun olmayan bilgiler için ipucu oluşturulmaz.

Eğer bir önbellek sunucu kullanıcının istediği bilgiyi kendi deposunda bulamazsa, bu isteği daha sonra anlatılacak hız kriterine göre en hızlı komşularına yönlendirir. Eğer belli zaman içinde hiçbir komşusu bu bilgiyi ona dönmezse, önbellek sunucu bilgiyi sunucu tipine göre ebeveyn sunucudan veya kaynak sunucudan getirir.

Hiyerarşide yüksek seviyede olan sunucular çocuklarından gelen istekleri sadece kendi depolarda ararlar. Eğer böyle bir sunucu isteği kendi deposunda bulamazsa kendisinin bir üst seviyesindeki sunucudan veya kaynak sunucudan ister. Burada hiyerarşi seviyesi ne kadar büyürse bekleme süresi o kadar

artacaktır. Ancak hop sayısında bir değişiklik olmayacaktır, çünkü zaten istemci bilgiyi doğrudan kaynak sunucudan istediği zaman da aynı sunuculardan geçmesi gerekir.

İsteği yönlendiren komşu sunucu sayısı o sunucunun derecesi olarak nitelendirilir. Eğer ağ içinde kullanılan ortalama derece artarsa başarı oranı da artacaktır, çünkü önbellek sunucular toplamda daha büyük önbellek bilgisi arasından arama yapacaktır. Her bir önbellek sunucunun kapasitesini 2 GB olarak alırsak ve sunucuların derecesini 2'den 3'e çıkarırsak, önbellek sunucuların arama yaptıkları bilgi miktarını 2 GB artırmış oluruz. Aynı zamanda sunucular arasında paylaşılan ipucu sayısı da artar, bu da sunucuların aranılan URL adresinin bulunduğu sunucuyu bulma ihtimalini artırır. Fakat sunucuların derecesini artırmak aranılan her bilgi için gerekli olan ağ mesaj miktarını artırmaktadır ve bu da ağ kullanım miktarını artırır.

İstek yönlendirme kısmının çalışması şöyle anlatılabilir. Sunucu 's', URL adresi 'u' yu kendi deposunda bulamadığında u' yu elinde bulunan ipucu listesinde arar. Bu listede aynı adres ile ilgili birden fazla sunucu bulabilir. Komşu sunucuların listesi hem run-time zamanında da hemde konfigurasyon dosya sayesinde oluşturulur. Her sunucu çalışmaya başladığında diğer önbellek sunucuların özelliklerini bir konfigürasyon dosyasından okur. Bu dosya bulunan sunucuların sırası isteklerin gönderilme sırasını doğrudan etkiler. Eğer dosyada A sunucusunun ismi B sunucusunun isminin üstünde ise istek öncelikle A sunucusuna, daha sonra B sunucusuna gönderilir. Fakat, bu sıralama statik değildir, s sunucusu A ile iletişim kurduğunda A sunucusundan gelen data miktarını ve geçen süreyi hesaplar. Belli zaman aralıkları ile bütün sunucular için toplanan bilgiler ile sunucuların performansı hesaplanır. Bu performans için aşağıdaki metot kullanılır.

Hız[x]=Hız[x]*Gecikme + Okunan_Data[x] / Geçen_Süre[x] *(1-Gecikme)

Gecikme 0 ile 1 arasında seçilen bir sabitidir. Eğer gecikme değeri çok küçük olursa, sunucuların yerleri çok hızlı değişir, çünkü sadece o anlık performanslarına göre değerlendirilir. Eğer gecikme değeri çok büyük seçilirse, çalışma sırasında sunucularda karşılaşılan sorunlar performansları etkilemez.

Eğer sunucu k eşik sürede sunucu s'ye cevap veremezse, s sunucusu k sunucusunun hızını negatif değere eşitler. Bu yüzden k sunucusu isteklerin gönderilmesi sırasında en sona gideceği için yeni istekler ona gönderilmez. Burada kullanılan eşik değeri de önemlidir. Eğer eşik değeri küçük seçilirse, s sunucu bazı ağ hatalarından dolayı başarılı olan k sunucusunu cezalandırabilir. Eğer büyük seçilirse, s sunucusu k sunucusunun çalışmadığını anlayamayacağı için gereksiz bir çok mesajı k sunucusuna göndermeye çalışır.

3.4 Önbellek Bilgi Değişim Protokolü

Daha önce anlatıldığı gibi s sunucusu aranılan u adresini kendi deposunda bulamadığında bu internet adresini diğer sunuculardan sorgular. Bu sorgulama zamanında aynı anda ipucu paylaşımı da yapılır. Bunun için verilen bütün cevaplar -olumlu veya olumsuz- ipucu içerir. Bu ipucu paylaşımı sayesinde her sunucu diğerlerinin ipucu bilgilerine de aynı zamanda sahip olur. İpucu paylaşımı sırasında 'hafıza' alanı kullanılır. Bu alan s sunucusu ile k sunucusunun en son ipucu değişim zamanını gösterir. Bu ipucular 'en son eklenenler' adında bir bağlı liste yapısında tutulur. Yeni ipucu geldiğinde listenin en başına eklenir. Eğer s sunucusu k sunucusu ile iletişim kurmaya karar verirse, bu listeden k sunucusu ile yaptığı en son ipucu değişimi zamanından sonra eklenen ipuçlarını gönderir. Sistem 2 çeşit sorgu ve 2 çeşit cevap vardır.

Q sorgusu:
Bu sorgu bir sunucudan diğer sunucuya bilgi istemek için gönderilir. Önce sorgu tipi 'Q' kullanılır, sonra bilginin URL adresi ve en sonuna da ipuçları eklenir. Bu sıralama, sunucu mesaj tipini ve URL adresini aldıktan hemen sonra cevap verebilmesi için önemlidir. Cevap verdikten sonra gelen ipucu bilgilerine göre kendi ipucu listesini güncelleştirebilir.
Format: Q URL İpucu1 İpucu2 ... İpucuN "."\r\n\r\n

P sorgusu:
P sorgusu internet bağlantısı olmayan sunuculardan onların ebeveynlerine gönderilen sorgulardır.
Format: P URL İpucu1 İpucu2 ... İpucuN "."\r\n\r\n

Y cevabı:
Bu cevaplar aranılan bilginin sunucunun bilgi deposunda bulunduğunu gösterir. Bu cevaplar hem ipuçlarını hem de istenilen bilgiyi içerir.
Format: Y İpucu1 İpucu 2 ... İpucuN "."\r\n\r\n BİLGİ

N cevabı:
Bu cevaplar aranılan bilginin sunucunun deposunda bulunmadığını gösterir ve cevaplar sadece ipucu içerir.
Format: N İpucu1 İpucu2 ... İpucuN "."\r\n\r\n

İsteklerin, bilgilerin ve ipuçlarını bir mesaj toplanması ağda dolaşan mesaj sayısını azaltmaktadır. Birçok mesajı bir tek mesaj içinde toplamak ağ trafiğini azaltacaktır.

3.5 Önbellek Yer değiştirme

Önbellek yer değiştirme için LRU algoritması kullanılmaktadır. Bütün yer değiştirmeler yerel

yapılmaktadır ve diğer sunucular haberdar edilmemektedir. Ayrıca ipucu listesinin boyutuyla sınırlı olduğu için bu listede yer değiştirme yapmak gereklidir. İpucu yer değiştirme algoritması olarak da LRU kullanılmaktadır

4. SONUÇ

4.1 Performans

Sistemin performansını ölçmek için DEC firmasından alınan erişim kayıt bilgileri kullanıldı [6]. Bu testin sonucunda %14 başarı elde edildi. Fakat elimizdeki kayıtta istenilen bilgilerin %70'nin tekil bilgiler olduğundan ideal önbellek sunucunun başarısının %30 olması gerekmektedir. Fakat varolan metotta ipuçları kullanıldığı için sunuculardaki bulunan bütün bilgilere tam olarak ulaşılamamıştır. Bunun yanında önbellek hacmi sınırlı olduğu için bazı bilgiler daha önce getirilmesine rağmen silinmiştir. Bütün bunlara karşılık sistemin başarısı %47'dir. Ayrıca testler sırasında sisteminin yeni eklenen sunucuları tanıdığı ve çöken sunucuları da kotardığı gözlemlenmiştir.

4.2 Analizler

- **Önbellek Yer değiştirme:** Sistemde her önbellek sunucunun bağımsız çalışması ve birbirlerinin bilgi depolarına karışmamasına karar verilmiştir. Sistemin performansını artırmak için çocuk sunucuların ebeveyn sunucularında bazı bilgilerin saklanmasına izin vermesi gerekir. A ve B sistemde birer düğüm ve B sunucusu hiyerarşide A sunucusunun ebeveyni olsun. Varolan sistemde, A sunucusu B sunucusunun sahip olmağı herhangi bir bilgiye sahip olursa, (komşu sunucularında getirilmiş) gerektiğinde yer kazanmak için bu bilgiyi silmek zorunda kalır. Fakat eğer A bilgiyi silmeden B'den bilgiyi saklamasını isteyebilirse daha sonra tekrar kullanabilir ve sistemin başarı oranını artırır. Bu yaklaşım sisteme 'push' özelliğinin eklenmesi demektir.

- **Önbellek Yerleştirme:** Şu anki sistemde saklanılan bilginin içeriği ile ilgili herhangi bir işlem yapılmamaktadır. Web sayfaları resim veya Java programları gibi gömülü bilgiler içerebilir. Bu bilgiler sayfalar istendiğinde getirilmektedir. Gömülü bilgilerin istenildiği veya istenilmediği HTTP başlıklarının 'User-Agent' alanında belirtilir. Eğer önbellek sunucular bu alanı okuyup gömülü bilgilerin istenileceğini anlarsa gerekli olan bilgileri kullanıcı istemeden daha önce isteyebilir. Bu web sayfasını görmek için gönderilen mesaj sayısının azaltır, fakat HTML bilgilerini ayırmak zaman alan bir işlemdir.

- **Hata Toleransı:** Sistem genel olarak hata toleranslı tasarlandığı halde kullanıcıların kullandığı internet gezginleri hata toleranslı değildir. Sunucular internet bağlantısı kesildiği durumda -yavaş olmasına rağmen- çalışmaya devam ederler. Fakat birçok ticari internet gezgini her protokol için bir tane önbellek sunucuya izin vermektedir. Bu yüzden sistemde çalışan başka sunucular olmasına rağmen kullanıcılar internete bağlanamamaktadır. Bunu çözmek için web önbellek ağı kullanıcı makinelerini de kapsayacak şekilde genişletilebilir. Böylece kullanıcılardaki sunucular çöktüğünde kullanıcılar sistemi tekrar ayağa kaldırabilirler.

- **Parametrelerin Etkisi**: Sistemde sabit tutulan parametrelerin sistem üzerindeki etkisinin kontrol edilmesi gerekir. Gecikme değerinin küçük veya büyük olarak seçilmesinin ve saklanacak bilgilerin büyüklüğünün eşik değerinin değişik değerlerde seçilmesinin sistemin performansına olan etkilerinin gözlenmesi gerekir.

KAYNAKLAR

[1] Tewari, R., Dahlin, M., Vin, H. M., Kay, J. S., "Design Considerations for Distributed Caching on the Internet", Proceedings of the 1999 International Conference on Distributed Computing Systems, May 1999.

[2] Wang, Z., "Cachemesh: A Distributed Cache System For World Wide Web", Web Cache Workshop, June 1997.

[3] Michel, S., Nguyen, K., Rosenstein, A., Zhang, L., "Adaptive Web Caching: Toward A New Global Caching Architecture ", Computer Networks & ISDN Systems, November 1998

[4] Chankhunthod, A., Danzig, P. B., Neerdaels, C., Schwartz, M. F., Worrell, K. J. "A Hierarchical Internet Object Cache", Proceedings of the 1996 USENIX Technical Conference, pages 153-163, San Diego, CA, January 1996.

[5] Arlitt, M. , Williamson, C., "Web Server Workload Characterization: The Search for Invariants", In Proceedings of the SIGMETRICS Conference on Measurement and Modeling of Computer Systems, May 1996.

[6] Digital Equipment Corporation, "Digital's Web Proxy Traces", ftp://ftp.digital.com/pub/DEC/traces/proxy/webtracs.html, September 1996.

[7] Almeida, J., Broder, A. Z., Cao, P., Fan, L. "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol", IEEE/ACM Transactions On Networking, Vol.8, No.3, June 2000