ACTA INFOLOGICA
dergipark.org.tr/acin

İSTANBUL UNIVERSITY PRESS

DOI: 10.26650/acin.879052

RESEARCH ARTICLE

Formel Yaklaşımlar ile Sürüm Değerlendirme

A Formal Methods Approach for Release Evaluation

Ebru Aydın Göl[1]

ÖZ

Bu çalışmada sürüm değerlendirme sürecini otomatikleştirmek için formel metotlar kullanılarak bir yöntem geliştirilmiştir. Sürüm değerlendirme sürecinde, bir sunucunun yeni ve eski sürümleri benzer (veya aynı) konfigürasyonlarda çalıştırılır ve sistemlerin ürettikleri izler karşılaştırılır. Bu karşılaştırma grafik incelenmesi, izlerin sabit ölçütler ile karşılaştırılması veya regresyon analizi tabanlı olabilir. Bu makalede ise, yapılan önceki çalışmalardan farklı olarak formel metotlar tabanlı bir analiz yöntemi sunulmaktadır. Bu yöntem karşılaştırılacak sistemlerden verilerin toplanması, her iki veri kümesi için bu kümeleri tanımlayacak Sinyal Zamansal Mantık (STL) formüllerinin üretilmesi ve son olarak da yüksek başarım ile kümeleri tanımlayabilen aynı yapıya sahip formüllerin karşılaştırılması ile sürüm değerlendirme sonucunun üretilmesi adımlarından oluşmaktadır. Bu yöntem ile sürüm değerlendirmede kullanılmak üzere, ölçütlerin STL formülü olarak ifade edilmesi ve bu ölçütlerin sistem izlerinden otomatik olarak üretilmesi sağlanmıştır. Zamansal mantıkların konuşma diline benzerlikleri sayesinde bu formüller açıklayıcıdır. Geliştirilen metot ile değerlendirme süreci otomatikleştirilmektedir. Elde edilen sonuçlar örnek veri kümeleri üzerinde incelenmiştir.

Anahtar Kelimeler: Formel Metotlar, Sürüm Değerlendirme, Formül Türetme, Zamansal Mantık

ABSTRACT

In this paper, a formal method-based release evaluation method was developed. During the release evaluation process, two versions of a server are run under similar (or the same) configurations and the system logs are compared. This comparison can be based on graphical analysis, applying fixed rules over logs or regression analysis. This paper presents a novel release evaluation approach based on formal methods. The proposed method consists of three main steps. The first step is the collection of data from both versions of the server. The second step is the synthesis of Signal Temporal Logic (STL) formulas for each dataset. The final step is the generation of the release evaluation result by comparing the formulas that have the same structure and that can represent both datasets with high accuracy. Thus, the proposed approach represents the release evaluation rules as STL formulas and generates such formulas from system logs in an automated way. Due to the resemblance of temporal logics to natural language, the resulting formulas explain the evaluation result. The proposed method automates the release evaluation process. The findings of the paper are shown over sample datasets.

Keywords: Formal Methods, Release Evaluation, Formula Synthesis, Temporal Logics

[1](Dr. Öğr. Üyesi), Orta Doğu Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü, Ankara, Türkiye

ORCID: E.A.G. 0000-0002-5813-9836

Corresponding author:
Ebru AYDIN GÖL
Orta Doğu Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü, Ankara, Türkiye
E-mail address: ebrugol@metu.edu.tr

Submitted: 12.02.2021
Revision Requested: 02.04.2021
Last Revision Received: 15.04.2021
Accepted: 17.05.2021
Published Online: 29.06.2021

Citation: Aydin Gol, E. (2021). Formel yaklaşımlar ile sürüm değerlendirme. *Acta Infologica, 5*(1), 129-140. https://doi.org/10.26650/acin.879052

This work is licensed under Creative Commons Attribution-NonCommercial 4.0 International License

1. GİRİŞ

Günümüzde e-devlet, e-posta, e-alışveriş ve e-gazete gibi kullanıcıya doğrudan hizmet veren birçok servis elektronik ortamda internet tabanlı olarak sunulmaktadır. Bu tip servisler için güvenilirlik ve ulaşılabilirlik kritik öneme sahiptir. Diğer taraftan, sunucu tabanlı servis sağlayan bu sistemler düzenli olarak güncellenmekte, yani sunuculara yeni versiyonlar sürülmektedir. Bu işlem, sürüm yönetimi ile idame ettirilmektedir (Bays, 1999). Bir sürüm yönetimi döngüsünde takip edilen ana adımlar yeni versiyona eklenecek özelliklerin belirlenmesi, geliştirme planının yapılması, geliştirme ve test faaliyetlerinin yapılması, sürüm değerlendirme ve son olarak bütün sunucuların güncellenmesi olarak sıralanabilir. Bu adımlar içinde, test ve sürüm değerlendirme adımları sistemin güvenilirlik ve ulaşılabilirlik gibi özelliklerinin korunması için çok önemlidir (Sommerville, 2015).

Sürüm değerlendirme işleminde hedef, sistemin bütünün veya bir bileşeninin yeni bir versiyonu geliştirildiğinde, bu versiyonda yapılan değişikliklerin beklenmeyen etkilerinin olup olmadığını saptamaktır. Bu amaçla, yeni sürüm ve eski sürüm benzer (veya aynı) konfigürasyonlarda çalıştırılır ve sistemlerin ürettikleri metrikler karşılaştırılır (Howard, 2016). Bu değerlendirmede, A ve B versiyonları birbirleri ile karşılaştırıldığı için "A/B testi" adı da verilmektedir (Chatterjee & Simonoff, 2013). Bu değerlendirme etkilenen kullanıcı sayısına göre farklı seviyelerde yapılmaktadır. Bu sıralamada amaç, yeni versiyonun sürülmesini engelleyecek bir durumu olabildiğince az kullanıcıyı etkileyerek saptamaktır. Şekil 1'de örnek bir sürüm değerlendirme süreci verilmiştir. Burada kırmızı kutular ile gösterilen adımlarda, gerçek kullanıcıya hizmet eden sunucular üzerinde değerlendirme yapılmaktadır. Sürüm değerlendirme süreçlerinde, genel olarak en son değerlendirme, en son geçerli sürümün bulunduğu sunucuların konfigürasyonuna benzer bir konfigürasyona sahip sunucu üzerinde yapılır ve buna kanarya testi adı verilir. Bu aşamalardan herhangi birinde bir problem gözlendiğinde ileriki aşamalara geçilmez, sürüm geri çekilir ve sorun kaynağının araştırılması süreci başlatılır. Sistemin özelliklerine ve sürüm güncelleme sıklığına göre bu aşamalardan bir kısmı uygulanmayabilir veya farklı aşamalar eklenebilir.



Şekil 1. Örnek bir sürüm değerlendirme süreci.

Her bir sürüm değerlendirme adımında, değerlendirilen yeni versiyon ve bir önceki versiyonu çalıştıran iki yazılımın ürettikleri metrikler belirli ölçütlere göre değerlendirilerek beklenmeyen durumlar saptanmaya çalışılır. Yaygın olarak, yazılımlardan zaman serisi verisi toplanır. Bu veri üzerinde ölçütler sinyal filtreleri (en büyük değer/ en küçük değer gibi) sonuçlarının karşılaştırılması veya istatistiksel özelliklerin (ortalama, süregelen ortalama, standart sapma gibi) karşılaştırılması olarak tanımlanır. Örneğin, bazı değişiklikler sonucu ortalama CPU kullanımının artması (veya azalması) beklenmez. Bunun yanı sıra sunucuların CPU kullanım miktarları gün içinde kullanıcı aktivitelerine göre değişebilmektedir. Bu nedenle, CPU kullanımında beklenmeyen bir değişiklik olup olmadığını kontrol edebilmek için 'iki sunucunun ortalama CPU kullanımı arasındaki fark 5 dakika boyunca %5'in üzerine çıkmaz' gibi bir ölçüt kullanılabilir. Yapılan karşılaştırmalar sonucunda, ulaşılan fark önceden belirlenen bir değerin üzerinde ise beklenmeyen bir durum oluştuğu sonucu çıkarılıp kök neden analizi adımına geçilir. Aksi durumda, sürüm değerlendirme sürecinin sonraki adımı ile devam edilir (bkz. Şekil 1).

Ölçüte dayalı değerlendirme çok hızlı yapılabilse de ölçütlerin tanımlanması uzmanlık gerektiren zor bir iştir ve olası hataların maliyeti yüksek olur. Örneğin sürüm değerlendirme adımında yakalanamayan bir durum son kullanıcıyı etkileyebilir. Diğer taraftan fazla hassas olarak belirlenen ölçütler aslında gereği olmayan analiz çalışmalarının yapılmasına yol açıp, yazılım mühendislerine zaman kaybettirebilir ve sürüm planının aksamasına yol açabilir. Birçok uygulamada sürüm değerlendirme adımı ilgili mühendisler tarafından manuel olarak yapılmaktadır. Bu yaklaşımda, zaman serisi verisi olarak üretilen metriklerden oluşturulan grafikler incelenmekte ve incelemeyi yapan kişi tarafından sonuç belirlenmektedir. Uzmanlar, sadece bir metrikteki değişikliğe göre değil, takip edilen bütün metrikleri birlikte değerlendirerek ek çıkarımlar yapabilmektedir. Örneğin bir uygulama sunucusu için, sunucunun yeniden başlatıldığı durumlarda, diğer metriklerin değerlerine bakılmayabilir veya kullanıcı sayısında anlık değişiklikler oldu ise diğer metriklerdeki bazı değişiklikler için hata değerlendirmesi yapılmayabilir. Uzmanların yaptığı bu tip değerlendirmeleri, otomatik olarak yapabilmek için birden fazla metriğe dayalı ölçütler tanımlamak gerekmektedir. Diğer taraftan uzman incelemesi hem kişi hatasına açık hem de zaman alan bir adımdır.

Basit ölçüt fonksiyonları yerine, sürüm analizinde kullanılan bir diğer yaklaşım ise regresyon analizidir (Chatterjee & Simonoff, 2013). Burada da uygun analiz yöntemini (doğrusal, çok terimli), uygulanacak ön işlemi (ortalama alma, sapma hesaplama gibi) ve sonuçta elde edilecek değerler için limitleri seçmek uzmanlık gerektirir ve bu adımlar her bir metrik için tekrarlanmalıdır. Yukarıda verilen örnekteki gibi birçok metrikteki değişikliklere göre çıkarım yapılması regresyon analizine doğrudan eklenemez ve ikincil değerlendirmelerin tanımlanmasını gerekebilir. Sürüm değerlendirme problemi, yeni sürümde öncekine göre herhangi bir anomali var mı şeklinde ele alınarak, makine öğrenmesi tabanlı metotlar da uygulanabilir. Fakat bu yaklaşım, regresyon analizinde olduğu gibi, metriklerin karakteristik özellikleri göz önünde bulundurularak, her bir metrik için farklı bir model geliştirilmesini gerektirebilir.

Bu çalışmada, sürüm değerlendirme ölçütlerinin formel dillerde ifade edilmesi ve bu ölçütlerin, değerlendirme başarısını en iyileştirecek şekilde otomatik olarak üretilmesi amaçlanmaktadır. Sistemlerin ürettikleri izlerin, gerçek zamanlı ve gerçek değerli sinyallerin özelliklerini tanımlamak için geliştirilmiş olan Sinyal Zamansal Mantık (STL) formülleri ile tanımlanması (Donzé, 2013), formüllerin arasındaki farklar üzerinden sürüm değerlendirme sonucu yapılması hedeflenmektedir. Önerilen yöntem şu temel adımlar ile özetlenebilir. İlk adım, A ve B versiyonlarından veri toplanmasıdır. İkinci adım her iki veri kümesi için de bu kümeleri en uygun şekilde tanımlayacak STL formüllerinin bulunmasıdır. Üçüncü adım ise, bir önceki adımda her iki veri kümesini de başarı ile tanımlayan formül yapılarının belirlenip, optimize edilmiş formüllerin parametrelerinin karşılaştırılması ile sürüm değerlendirme sonucunun üretilmesidir.

STL formülleri ile sinyal özellikleri konuşma diline yakın bir yazım ile ifade edilebilir. Örneğin $F_{[0,10]}\, x>10$ formülü, takip eden on zaman biriminden en azından birinde x değerinin 10 üzerine çıkması gerektiğini, $x > 0\ U_{[0,10]}\, y = 1$ formülü ise takip eden on zaman biriminden birinde y'nin bir olması gerektiğini ve bu ana kadar x'in 0'dan büyük olması gerektiğini ifade eder. Zamansal ve Boole mantık operatörlerinin iç içe kullanılması ile daha karmaşık formüller elde edilebilir. Fakat, formül yapısının belirlenmesi ($x > p_1\ U_{[p2,p3]}\, y = p_4$) ve bu yapıdaki parametrelerin veriye uygun bir şekilde üretilmesi başlı başına zor bir problemdir.

Sistem izlerinden (zaman serisi verisi), bu izleri tanımlayacak STL formüllerinin üretilmesi üzerinde yakın zamanda birçok çalışma yapılmıştır (Aydin & Aydin Gol, 2020; Bartocci, Bortolussi & Sanguinetti, 2014; Bombara, Vasile, Penedo, Yasuoka & Belta, 2016; Ergurtuna & Aydin Gol, 2019; Hoxha, Dokhanchi & Fainekos, 2018; Jha, Tiwari, Seshia, Sahai & Shankar, 2019; Jin, Donzé, Deshmukh & Seshia, 2015; Ketenci & Aydin Gol, 2019; Kong, Jones, Ayala, Aydin Gol & Belta, 2014; Mohammadinejad, Deshmukh, Puranic, Vazquez-Chanlatte & Donzé, 2020; Yoo & Belta, 2017). Bu çalışmalar, veri kümesinin yapısı ve üretilen formül açısından farklılık göstermektedir. Veri kümesindeki bütün izleri tanımlayacak bir formül üretilmesi problemi gereksinim madenciliği olarak da tanımlanmaktadır: örneğin eski (legacy) bir sistemin sağladığı gereksinimin otomatik olarak bulunması (Hoxha et al., 2017; Jha et al., 2019; Jin et al., 2015). Bu konudaki bazı çalışmalarda, formül yapısı kullanıcı tarafından belirlenmiş ve belirlenen formül yapısı için parametre bulunmuştur (Hoxha et al., 2017; Jin et al., 2015). İyi ve kötü olarak etiketlenmiş sistem izlerinden, bu izleri doğru şekilde sınıflandırabilecek bir formül bulunması hem gerçek zamanlı sistem izleme hem de sınıflandırıcı üretmek uygulamaları için çalışılmıştır (Aydin & Aydin Gol, 2020; Bartocci et al., 2014; Bombara et al., 2016; Ergurtuna & Aydin Gol, 2019; Ketenci & Aydin Gol, 2019; Kong et al. 2014; Mohammadinejad et al., 2020; Yoo & Belta, 2017).

Bu çalışmanın literatüre ana katkısı formel metotlara dayalı bir sürüm değerlendirme sistemi tanımlamasıdır. Bu sistemin tamamen otomatik olması, elde edilen sonuçların kullanıcıları için açıklayıcı nitelikte olması ve ulaşılan sonucun geçerliliğine dair nicel değerler sağlanması, geliştirilen yöntemin en önemli özellikleri arasındadır. Temel olarak, daha önce bahsedilen ölçüt belirlenmesi, regresyon analizi veya makine öğrenmesi tabanlı yaklaşımlar, her bir metrik için bir ön çalışma ile sürüm değerlendirmede uygulanacak olan modelin (ölçüt sınırları, regresyon parametreleri, anomali tanıma modeli gibi) belirlenmesini gerektirir. Diğer taraftan, geliştirilen değerlendirme sistemi bu tip ön çalışmalar gerektirmez. Sonuçlar formül tabanlı oluşturulduğu için, açıklayıcı niteliktedir. Bu da makine öğrenmesi tabanlı metotlara kıyasla bir avantaj sunmaktadır. Ek olarak, geliştirilen yöntem, iki sistemin karşılaştırılması üzerine formel yaklaşımlar tabanlı bir yöntem olarak daha geniş bir pencereden değerlendirildiğinde, sürüm değerlendirme dışında farklı problemlere de uygulanabilir (örneğin enjeksyon/ kötü niyetli sistem tespiti).

Makalenin yapısı. 2. Bölüm'de STL formüllerinin söz-dizim kuralları ve semantiği verilmiş, bu bilgiler ışığında formül sentezleme problemi anlatılmış ve literatürde bulunan yaklaşımlar özetlenmiştir. Takip eden kısımda ise, önerilen formel yaklaşımlar ile sürüm değerlendirme yöntemi anlatılmış ve bu yöntemde izlenecek her bir adım detaylandırılmıştır. 4. Bölüm'de, örnek veri kümeleri tanımlanmış ve bu örnek durumlar üzerinde elde edilen sonuçlar anlatılmıştır. Son kısımda ise, yapılan çalışma özetlenmiş ve ileriki çalışma yönleri sunulmuştur.

2. SİNYAL ZAMANSAL MANTIK VE FORMÜL SENTEZİ

Bu kısımda geliştirilen sürüm değerlendirme sisteminde kullanılan araçlara yönelik ön bilgiler sunulmuştur.

2.1 Sinyal Zamansal Mantık

Sonlu kesikli ve n-boyutlu bir sinyal $x = x_0, x_1,\ldots,x_N$ dizisi şeklinde gösterilir. Sinyalin *i.* değişkeni *t*-kesikli zamanında x_i^t notasyonu ile gösterilir.

Geçmiş zamanlı sinyal zamansal mantık (past time Signal Temporal Logic - ptSTL) için söz-dizim kuralları Denklem (1)'deki gibi tanımlanır (Donzé, 2013; Gabbay, 1989).

$$\phi = \boldsymbol{T} \left| x^i \sim c \right| \neg\phi | \phi \wedge \phi | \phi \vee \phi | \phi \boldsymbol{S}_{[a,b]} \phi \left| \boldsymbol{F}^{-}_{[a,b]} \phi \right| \boldsymbol{G}^{-}_{[a,b]} \phi \tag{1}$$

Bu denklemde $\boldsymbol{T}$ mantık sabiti "doğru", x^i sinyal değişkeni, $\sim \in \{<, \leq, >, \geq\}$ bir eşitsizlik sembolü, c sabit bir değer,¬-değil,∧-ve, ∨-veya ise standart mantık operatörleridir. $\boldsymbol{S}_{[a,b]}$ (olduğundan beri), $\boldsymbol{F}^{-}_{[a,b]}$ (geçmişte bir zaman) ve $\boldsymbol{G}^{-}_{[a,b]}$ (geçmişte her zaman) zaman aralıkları $[a,b]$ olan geçmiş zaman operatörleridir.

Bir ptSTL formülünün semantiği bir sinyal ve belirli bir zaman üzerinde tanımlanmıştır. Özetle, $\boldsymbol{T}$ formülü her zaman sağlanır; $x_t^i \sim c$ eşitsizliği sağlanıyor ise sinyal *t* anında $x^i \sim c$ formülünü sağlar. Mantık operatörleri de standart semantiklerine göre değerlendirilir: sinyal *t* anında ϕ formülünü sağlamıyor ise, $\neg\phi$ formülünü sağlar; ϕ_1 ve ϕ_2 formüllerini sağlıyorsa, $\phi_1 \wedge \phi_2$ formülünü de sağlar; ϕ_1 ve ϕ_2 formüllerinden en az birini sağlıyorsa, $\phi_1 \vee \phi_2$ formülünü de sağlar. Görüldüğü gibi mantık operatörlerinin anlamlandırmasında içerdikleri formüllerin o andaki değerleri kontrol edilmektedir. Diğer taraftan, zamansal operatörler için içerdikleri formüllerin belirli bir aralıktaki değerlerinden anlamlandırma yapılmaktadır. . ϕ formülü $[t - b, t - a]$ aralığında her an sağlanır ise, $\boldsymbol{G}^{-}_{[a,b]} \phi$ formülü *t* anında sağlanır. ϕ formülü $[t - b, t - a]$ aralığında herhangi bir anda sağlanır ise, $\boldsymbol{F}^{-}_{[a,b]} \phi$ formülü *t* anında sağlanır. ϕ_2 formülü $[t - b, t - a]$ aralığında bir t' anında sağlanır, ve ϕ^l formülü o andan itibaren, yani $[t', t]$ aralığında her an sağlanır ise, $\phi_1 \boldsymbol{S}_{[a,b]} \phi_2$ formülü *t* anında sağlanır. Bir $\boldsymbol{x}$ sinyalinin *t* anında ϕ formülünü sağlaması $\boldsymbol{x}, t \vDash \phi$ şeklinde gösterilir. Diğer yandan, $\boldsymbol{x}$ sinyalinin *t* anında ϕ formülünü sağlamaması, diğer bir deyişle ihlal etmesi, $\boldsymbol{x}, t \nvDash \phi$ şeklinde gösterilir.

Bir $\boldsymbol{x}$ sinyalinin bir ptSTL formülünü ne ölçüde sağlayıp sağlamadığına ise nicel değerlendirme ile karar verilir. Nicel değerlendirme hesabı Denklem (2)'de verilmiştir. Bir $\boldsymbol{x}$ sinyali ve ϕ formülü için, *t* anında nicel değerlendirme sonucu pozitif ise, yani $\rho(\phi, \boldsymbol{x}, t) > 0$ ise, o anda sinyal formülü sağlar. Diğer taraftan değerlendirme sonucu negatif ise, yani $\rho(\phi, \boldsymbol{x}, t) < 0$ ise, o anda sinyal formülü ihlal eder. Nicel değerlendirme sonucunun 0 olduğu durumlarda ise yukarıda özetlenen nitel değerlendirmeye bakılır.

$$
\begin{aligned}
\rho(\boldsymbol{T}, \boldsymbol{x}, t) &= T \\
\rho(x^i > c, \boldsymbol{x}, t) &= x_t^i - c \\
\rho(x^i < c, \boldsymbol{x}, t) &= c - x_t^i \\
\rho(\neg\phi, \boldsymbol{x}, t) &= -\rho(\phi, \boldsymbol{x}, t) \\
\rho(\phi_1 \wedge \phi_2, \boldsymbol{x}, t) &= \min\big(\rho(\phi_1, \boldsymbol{x}, t), \rho(\phi_2, \boldsymbol{x}, t)\big) \\
\rho(\phi_1 \vee \phi_2, \boldsymbol{x}, t) &= \max\big(\rho(\phi_1, \boldsymbol{x}, t), \rho(\phi_2, \boldsymbol{x}, t)\big) \\
\rho\big(\phi_1 \boldsymbol{S}_{[a,b]} \phi_2, \boldsymbol{x}, t\big) &= \max_{j \in [t-b, t-a]} \Big(\rho(\phi_2, \boldsymbol{x}, j), \min_{k \in [j+1, t]} \rho(\phi_1, \boldsymbol{x}, t)\Big) \\
\rho\big(\boldsymbol{F}^{-}_{[a,b]} \phi, \boldsymbol{x}, t\big) &= \max_{j \in [t-b, t-a]} \rho(\phi, \boldsymbol{x}, t) \\
\rho\big(\boldsymbol{G}^{-}_{[a,b]} \phi, \boldsymbol{x}, t\big) &= \min_{j \in [t-b, t-a]} \rho(\phi, \boldsymbol{x}, t)
\end{aligned}
\qquad (2)
$$

Nicel değerlendirme sonucu gürbüzlük olarak da isimlendirilir (Jin et al, 2015; Asarin, Donzé, Maler & Nickovic, 2012). Nicel değerlendirme sonucunun mutlak değerinden küçük bir ϵ değeri için, (yani $\epsilon < |\rho(\phi, \boldsymbol{x}, t)|$), sinyal ϵ kadar sarsılırsa da nitel değerlendirme sonucu değişmez.

Eşitsizliklerde ($x^i \sim c$) veya zaman aralıklarında (*[a,b]*) sabit sayılar yerine parametreler kullanılması ile parametrik ptSTL formülleri tanımlanır (Asarin et al., 2012). Parametrik bir ptSTL formülünün parametrelerine sabit değerler (*p*) atanarak bir ptSTL formülü ($\phi(p)$) elde edilir. Örneğin $\phi = \boldsymbol{F}^{-}_{[0,p1]} x^0 > p_2$ parametrik formülünden $p = (p_1 \leftarrow 5, p_2 \leftarrow 6)$ değerleri ile $\phi(p) = F^{-}_{[0,5]} x^0 > 6$ formülü elde edilir.

2.2 Formül Sentezi

Formül sentezi problemi, verilen sinyal kümesini en iyi tanımlayacak zamansal mantık formülünün türetilmesi olarak tanımlanır. Bu problem, kullanılan zamansal mantık (STL veya ptSTL), veri kümesinin yapısı, parametrik formüllerin problem girdisi olup olmaması açısından farklı biçimlerde çalışılmıştır (Aydin & Aydin Gol, 2020; Bartocci et al., 2014; Bombara et al., 2016; Ergurtuna & Aydin Gol, 2019; Hoxha et al., 2018; Jha et al., 2019; Jin et al., 2015; Ketenci & Aydin Gol, 2019; Kong et al., 2014; Mohammadinejad et al., 2020; Yoo & Belta, 2017).

Verilen bir sinyal kümesini en iyi şekilde tanımlayacak olan STL formülünü bulmayı hedefleyen formül sentezi problemi gereksinim türetme olarak da adlandırılmıştır (Jin et al. 2015). Bu çalışmada formül sentezleme problemi, verilen parametrik bir STL formülü için, veri kümesini olabildiğince sıkı şekilde tanımlayacak parametre değerlerinin bulunması olarak ele alınmıştır. Her bir sinyal için değerlendirme sinyalin başında yapılmış ve sinyal kümesi üzerinde elde edilen en küçük sinyal değerlendirmesini (gürbüzlük değerini) sıfıra yakın pozitif bir değer yapan parametreler üretilmiştir. Bu sayede veri kümesindeki sinyallerin hepsini olabildiğince net bir şekilde tanımlayan formüle ulaşılmıştır. Bu çalışmada görülen en önemli eksiklik, parametrik formül yapısının tanımlanmasının gerekmesidir. Ek olarak, parametreler arasında bir sıralama yapılması gerekmekte ve parametre optimizasyonu bu sıralamaya göre yapılmaktadır. Sıralamada önde yer alan parametreler aldıkları değerler ile diğer parametreleri kısıtlamakta, standart altı (sub-optimal) sonuçlar elde edilmektedir. Sadece pozitif örneklerden formül türetme üzerine yapılan diğer bir çalışmada bu kısıtların üzerine gidilerek, nicel değerlendirme sonuçlarına dayalı bir sıkılık ölçütü tanımlanmış ve bu ölçüt ile parametreler arası sıralama yapılmadan optimizasyon yapılmıştır (Jha et al., 2019). Ek olarak verilen bir parametrik formül kümesi için optimizasyon yapılmış ve bu formüllerin en uygun kombinasyonu üretilmiştir. Bu çalışmada (Jin et al. 2015)'de olduğu gibi tek bir parametrik formül ile kısıtlı kalınmasa da kullanıcının parametrik formül kümesinin tanımlaması gerekmektedir.

Formül sentezleme problemi, iyi ve kötü olarak etiketlenmiş olan sinyalleri ayırt edebilecek bir formül bulma olarak da ele alınmıştır. Bu alandaki çalışmalar da sinyal başına bir etiket olması (Bartocci et al., 2014; Bombara et al., 2016; Kong et al., 2014; Yoo & Belta, 2017) ve her sinyalin her noktası için bir etiket olması, yani etiketlerin zaman serisi verisi olması, olarak ayrışmaktadır (Aydin & Aydin Gol, 2020; Ergurtuna & Aydin Gol, 2019; Ketenci & Aydin Gol, 2019). Bu çalışmalar formül yapısının belirlenmesini, yani parametrik formül tanımlanmasını gerektirmemektedir. Formül yapısının üretilebilmesi için

karar ağacı tabanlı sentez metotları (Bombara et al., 2016; Ketenci & Aydin Gol, 2019; Yoo & Belta, 2017); yinelemeli formül birleştirme metotları (Kong et al., 2014; Ergurtuna & Aydin Gol, 2019); genetik algoritma tabanlı metotlar (Bartocci et al., 2014; Aydin & Aydin Gol, 2020); ve parametrik formül uzayının sıralanması ve her bir parametrik formül için parametre optimizasyonu yapılması yöntemleri (Aydin & Aydin Gol, 2020; Mohammadinejad et al., 2020) bulunmaktadır.

Bu çalışmada amaç, sürüm değerlendirme için formel metotlara dayalı bir yöntem geliştirmektir. Bu amaca ulaşmak için bir veri kümesini en iyi şekilde tanımlayacak formül sentezlenmiştir. Bu adımda, (Jha et al., 2019; Jin et al. 2015) çalışmalarında olduğu gibi sadece pozitif örnekler kullanılmıştır. Fakat bu çalışmalardan farklı olarak (Aydin & Aydin Gol, 2020; Mohammadinejad et al., 2020) çalışmalarında olduğu gibi parametrik formül uzayı sıralanmıştır ve her bir formül için parametre optimizasyonu yapılmıştır. Ek olarak, formülün veri kümesini tanımlamadaki başarısını ölçmek üzere nicel değerlendirmeye dayalı yeni bir ölçüt tanımlanmıştır. Bu bağlamda, pozitif örneklerden formül sentezi için parametrik formül tanımlanmasını gerektirmeyen özgün bir yöntem ortaya konmaktadır. Bu çalışmanın sonraki aşamalarında ise formüllerin karşılaştırılması için yeni bir formül farkı ölçüm metriği geliştirilmiştir. Bu metrik ile sürüm değerlendirme sonuçları üretilmiştir.

3. ptSTL İLE SÜRÜM DEĞERLENDİRME

Bu çalışmada sürüm değerlendirme sürecini otomatikleştirmek için formel metotlar tabanlı bir yöntem geliştirilmiştir. Bu yöntemin ilk adımı, karşılaştırılacak olan A ve B versiyonlarından veri kümelerinin oluşturulmasıdır (3.1. Bölüm). İkinci adımda ise bu kümeleri en uygun şekilde tanımlayan ptSTL formülleri üretilmiştir (3.2. Bölüm). Üçüncü adım ise, bir önceki adımda her iki veri kümesini de başarı ile tanımlayan formül yapılarının belirlenip, optimize edilmiş formüllerin parametrelerinin karşılaştırılması ile sürüm değerlendirme sonucunun üretilmesidir (3.3. Bölüm).

3.1 Veri Kümesi Oluşturma

Önerilen sürüm değerlendirme metodunun ilk adımı, her bir sürüm için bir sinyal kümesinin oluşturulmasıdır. Bu adım için, öncelikle takip edilecek olan metriklerin belirlenmesi gerekmektedir. Örneğin istek cevap süresi (response time) ile ilgili yapılacak bir değerlendirme için, aynı sunucuya bir dakika içinde gelen istek sayısının da (request rate) göz önünde bulundurulması gerekir.

Takip edilecek metrikler $M = \{1,..., n\}$ kümesi ile ifade edilir. Burada her bir indeks $i \in M$ bir metrik için kullanılmaktadır (CPU kullanım yüzdesi gibi). Metrik-i'nin alabileceği değerler π^i ile gösterilir. Bu metriklerin belirlenmesinin ardından, değerlendirme yapılacak uygulamanın A ve B versiyonları, olabildiğince aynı konfigürasyonlarda çalıştırılır. Sistem günlüklerinden (loglar) belirlenen metriklerin anlık değerleri çıkarılarak Denklem (3)'deki gibi veri kümeleri elde edilir. Burada N bir sinyalin uzunluğunu ifade eder. Bu uzunluk, değerlendirme yapılan sistemin özelliklerine göre bir saat veya bir gün olarak seçilebilir. Farklı versiyonlardan oluşturulan veri kümeleri D_A ve D_B ile gösterilir. Sürüm versiyonları dışındaki faktörlerin etkisini istatistiksel olarak azaltmak için, bir veri kümesi oluşturulurken aynı konfigürasyona sahip birden fazla sunucudan izler kullanılabilir.

$$D = \{\boldsymbol{x} = x_0, x_1, \dots, x_N \mid her\ bir\ j \in M\ ve\ her\ bir\ i \in \{0, \dots, N\}\ için\ x_i^j \in \pi^j\} \quad (3)$$

3.2 Formül Sentezi

Önerilen metodun ikinci adımı, ilk adımda üretilen sinyal kümelerini (D_A ve D_B) en iyi şekilde tanımlayan ptSTL formüllerinin üretilmesidir. Bu amaçla, öncelikle bir formülün sinyal kümesine uyumunun tersini gösteren bir ölçüt tanımlanmıştır. Parametrik bir ptSTL formülü için, bu ölçütü en küçükleştiren parametre değerleri ızgara araması (grid search) metodu ile hesaplanmıştır. Veri kümesini en iyi tanımlayan ptSTL formüllerinin bulunması için ise, belirli bir operatör sayısına sahip ve değişken olarak takip edilen metrikler (M kümesi) kullanılarak oluşturulabilen bütün parametrik formüller tanımlanmış ve her biri için parametre optimizasyonu yapılmıştır.

3.2.1 Optimizasyon Ölçütü

Bir formülün bir sinyale olan uyumsuzluğu, sinyalin belirli bir noktasından (k) başlanarak elde edilen nicel değerlendirme sonuçlarının mutlak değerlerinin ortalaması olarak tanımlanmıştır.

$$C(\phi, \boldsymbol{x}) = \frac{\sum_{i=k}^{N} |\rho(\phi, \boldsymbol{x}, i)|}{N-k} \quad (4)$$

Bir formülün, bir sinyal üzerindeki uyumsuzluk değerinin düşük olması, formülün sinyalin *k*-*N* aralığındaki kısmını her noktada sıkı bir şekilde tanımladığını gösterir. Burada *k* sayısı, 0 alınabileceği gibi formülde geçen zaman aralıklarına göre de belirlenebilir. Örneğin, $\boldsymbol{F}^{-}_{[0,10]}\boldsymbol{G}^{-}_{[0,3]}x^0 > 6$ formülü için *k=13* değeri, değerlendirmenin ilk yapıldığı noktada da sinyalin uzunluğunun yeterli olmasını sağlar. Geliştirilen metotta elde edilen uyumsuzluk değeri farklı formüllerin karşılaştırılmasında kullanıldığı için, bütün formüllerde aynı *k* sabiti kullanılmıştır.

Bir formülün sinyal kümesi üzerindeki uyumsuzluk değeri bütün sinyaller üzerindeki uyumsuzluk değerlerinin toplamına eşittir.

$$C(\phi, D) = \frac{\sum_{x \in D} C(\phi, \boldsymbol{x})}{\sum_{x \in D} 1} \quad (5)$$

Denklem (5)'de tanımlanan başarım ölçütü, literatürde sadece pozitif örneklerden formül türetme çalışmalarında kullanılan ölçütlerden farklılık göstermektedir. Temel olarak, Jin et al. (2015) gelecek zamansal operatörlerini kullanmış ve sadece sinyalin başında değerlendirme yapmıştır. Ek olarak, bahsedilen çalışmada amaç bütün sinyallerin sıkı bir şekilde sağladığı bir formül türetmektir. Bu nedenle, pozitif bir *c* değeri için, $\wedge_{x \in D} \rho(\phi, \boldsymbol{x}, 0) > c$ koşulunu sağlayan ve *c* değerini en küçükleştiren parametreler aranmıştır. Bu çalışmada ise, değerlendirme sinyalin her noktasında yapılarak sürüm değerlendirmede kullanılmak üzere sinyal karakterini tanımlayan bir formül aranmıştır. Ek olarak, nicel değerlendirme sonucunun pozitif olması koşulu aranmamaktadır. Sinyalin gürültülü olabileceği de göz önünde bulundurularak, Denklem (4)'de gösterildiği gibi, sinyal boyunca birçok noktada nicel değerlendirme yapılmış ve değerlendirme sonuçlarının mutlak değerlerinin ortalaması hesaplanmıştır. Bu şekilde tanımlanan ölçüt, sinyali her noktada olabildiğince sıkı şekilde tanımlayan formüller için küçük değerler alacaktır.

3.2.2 Parametre optimizasyonu

Bu işlem, bir parametrik ptSTL formülü ϕ için, tanımlanmış parametre uzayı P^{ϕ} içinde, verilen ölçütü en iyileştiren parametre değerlerinin bulunmasını amaçlar. Bu çalışmada, parametre optimizasyonu, parametre uzayının taranması, yani ızgara araması metodu ile yapılmıştır. Örneğin parametrik $\phi = \boldsymbol{F}^{-}_{[0,p_1]}\boldsymbol{G}^{-}_{[0,p_2]}x^0 > p_3$ formülü ve $p_1, p_2 \in \{1,3, \ldots, 2m+1\}, p_3 \in \{1,2, \ldots, n\}$ parametre uzayı için, $m^2 n$ adet parametre değeri bulunmaktadır. Izgara araması metodunda, her bir parametre değeri $p \in P^{\phi}$ için bir ptSTL formülü $\phi(p)$ tanımlanır ve bu formülün veri kümesi uyumsuzluk değeri $C(\phi(p), D)$ Denklem (5)'deki gibi hesaplanır. Bu hesap sonucunda, uyumsuzluk değerini en küçükleştiren parametre değeri p* seçilir:

$$p^{\star} = \underset{p \in P^{\phi}}{\operatorname{argmin}}\, C(\phi(p), D) \quad (6)$$

3.2.3 Formül Sentezi

Bir sinyal kümesini en iyi şekilde tanımlayan ptSTL formülünü üretmek için, öncelikle sinyal değişkenleri (*M* kümesi) kullanılarak oluşturulabilen parametrik formüller tanımlanmıştır. Ardından her bir formül için Denklem (6)'da belirtilen parametre optimizasyonu yapılmıştır. Temel olarak, operatör limiti *c* ve değişken kümesi *M* olarak verildiğinde, *c* tane operatör içeren bütün ptSTL formülleri F^{c} ile gösterilmiş ve özyinelemeli olarak Denklem (7)'deki gibi tanımlanmıştır. Buna dayalı olarak, en çok *c* operatör içeren bütün ptSTL formülleri ise $F^{\leq c}$ ile gösterilmiş ve Denklem (8)'deki gibi tanımlanmıştır (Aydin & Aydin Gol, 2020).

$$\mathcal{F}^0 = \{x \sim p \,|\, x \in M, \sim \in \{<, >\}\} \quad (7)$$

$$\begin{aligned}\mathcal{F}^c = \{\neg\phi | \phi \in \mathcal{F}^{c-1}\} \cup \\ \left\{\boldsymbol{X}_{[a,b]}\phi \middle| \phi \in \mathcal{F}^{c-1}, \boldsymbol{X} \in \{\boldsymbol{F}^{-}, \boldsymbol{G}^{-}\}\right\} \\ \bigcup_{i=1}^{c-1} \left\{\phi_1 \; op \; \phi_2 \middle| \phi_1 \in \mathcal{F}^{i}, \phi_2 \in \mathcal{F}^{c-i-1}, op \in \{\wedge, \vee\}\right\} \\ \bigcup_{i=1}^{c-1} \left\{\phi_1 \, \boldsymbol{S}_{[a,b]} \, \phi_2 \middle| \phi_1 \in \mathcal{F}^{i}, \phi_2 \in \mathcal{F}^{c-i-1}\right\}\end{aligned}$$

$$\mathcal{F}^{\leq c} = \bigcup_{i=0}^{c} \mathcal{F}^{i} \quad (8)$$

Formül sentezi aşamasında öncelikle bir operatör üst limiti (c) belirlenmiştir. Ardından, takip edilen metrikler üzerinde bu limitten fazla operatör içermeyen bütün parametrik ptSTL formülleri $F^{\leq c}$ tanımlanmıştır (Denklem (8)). Her bir parametrik formül $\phi \in F^{\leq c}$ için, farklı versiyonlardan üretilmiş olan D_A ve D_B sinyal kümeleri üzerinde parametre optimizasyonu yapılmıştır. Bunun sonucunda Denklem (9)'da verilen parametrik formül ϕ, D_A kümesine göre optimum parametre $p_A^\star$, ve D_B kümesine göre optimum parametre $p_B^\star$ üçlülerinden oluşan bir küme tanımlanmıştır.

$$\mathcal{F}^p = \{(\phi, p_A^\star, p_B^\star) \mid \quad p_A^\star = \underset{p \in P^\phi}{\operatorname{argmin}} C(\phi(p), D_A), \quad p_B^\star = \underset{p \in P^\phi}{\operatorname{argmin}} C(\phi(p), D_B), \quad \phi \in \mathcal{F}^{\leq c} \} \tag{9}$$

Denklem (9)'da verilen kümenin oluşturulması için, $F^{\leq c}$ kümesindeki her bir parametrik formül için Denklem (6)'da verilen ızgara araması işlemi iki kere gerçekleştirilmiştir. $F^{\leq c}$ kümesinin boyutu takip edilen metrik sayısı (M kümesinin boyutu) ve operatör üst limiti (c) ile üstel olarak artmaktadır. Izgara aramasında kullanılan parametre değeri sayısı (P^ϕ kümesinin boyutu), formüldeki parametre sayısı ile üstel olarak artmaktadır. Bir formülde bulunan parametre sayısı ise formülün operatör sayısı ile doğru orantılıdır. Bu değerlendirme ışığında, metrik sayısının ve operatör üst limitinin arttırılmasının hesap süresini ciddi ölçüde arttıracağı görülmektedir. Bu süre köşegen taraması (Ergurtuna & Aydin Gol, 2019) veya ızgara aramasında bilinen en iyi sonuca göre aramanın erken sonlandırılması gibi keşifsel metotlar ile kısaltılabilir.

3.2 Sürüm Değerlendirme Kararı

Sunulan formel metotlara dayalı sürüm değerlendirme sürecinin son aşaması, A ve B versiyonlarını tanımlayacak şekilde üretilen formüllerin ve parametrelerinin karşılaştırılarak sürüm değerlendirme sonucunun üretilmesidir.

Öncelikle, her iki versiyonu da en iyi şekilde tanımlayan bir parametrik formül ile bu formülün D_A ve D_B sinyal kümelerine göre optimum parametreleri Denklem (10)'da verilen kritere göre seçilir.

$$(\phi^\star, p_A^\star, p_B^\star) = \underset{(\phi, p_A^\star, p_B^\star) \in \mathcal{F}^p}{\operatorname{argmin}} \max\left(C(\phi(p_A^\star), D_A), C(\phi(p_B^\star), D_B)\right) \tag{10}$$

Seçilen parametrik formül ve parametrelerden elde edilen formüller $\phi^\star\ (p_A^\star)$ ve $\phi^\star\ (p_B^\star)$, sırası ile D_A ve D_B sinyal kümelerini yüksek başarım ile tanımlamaktadır. Bu formüllerin yapısı aynıdır fakat eşitsizlik sabitleri veya zamansal parametreler farklılık gösterebilmektedir. Burada ortaya çıkan farklılıklara göre ise sürüm değerlendirme sonucu üretilmektedir. Temel olarak $\phi^\star$ formülünde geçen her bir parametre p_i için, bu parametrenin optimize edilmiş değerleri $p_A^\star(p_i)$ ve $p_B^\star(p_i)$ karşılaştırılır. Karşılaştırma sırasında zamansal ve metrik parametreler için değerlendirme yapılan sisteme göre önceden belirlenmiş olan sınırlar kullanılabilir. Bunun yanı sıra, sınırların otomatik olarak belirlenmesi gerektiği durumlarda ise, birbirlerine göre bağıl olarak kontrol edilebilirler. Örneğin, B versiyonu için bulunan parametrenin, A versiyonu için bulunan parametrenin yüzde 10 yakınında olması gibi, $p_B^\star(p_i) \in [0.9p_A^\star(p_i), 1.1p_A^\star(p_i)]$. Parametrelerden herhangi biri belirlenen sınırların dışında bir değer almış ise sürüm değerlendirme sonucu olumsuz, bütün parametrelerin belirlenen limitler içinde değer alması durumunda ise sürüm değerlendirme sonucu olumlu olarak üretilir. Bu yaklaşım $p_1,\ldots,p_n$ parametrelerini içeren $\phi^\star$ formülü için Denklem (11)'de özetlenmiştir.

$$\text{OLUMLU: } Eğer\ her\ bir\ i = 1, \ldots, n\ \ için \qquad p_B^\star(p_i) \in [(1-r)p_A^\star(p_i), (1+r)p_A^\star(p_i)] \tag{11}$$

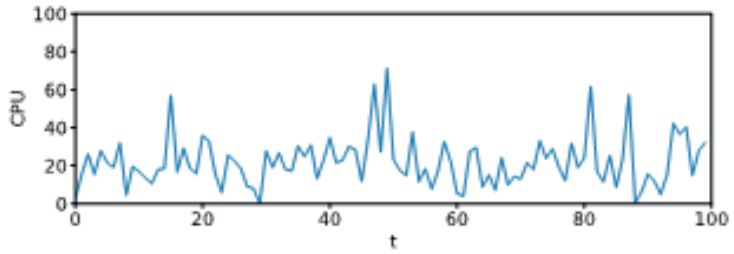
$$\text{OLUMSUZ: } Eğer\ en\ azından\ bir\ i = 1, \ldots, n\ \ için \qquad p_B^\star(p_i) \notin [(1-r)p_A^\star(p_i), (1+r)p_A^\star(p_i)]$$

Sürüm değerlendirme sonucunun olumsuz olması durumuna, $\phi^\star\ (p_A^\star)$ ve $\phi^\star\ (p_B^\star)$ formülleri ile Denklem (11)'de verilen koşulu sağlamayan parametreler kullanıcıya verilerek değerlendirmeye yönelik anlaşılır bir analiz sunulur.

4. ÖRNEK SİSTEM ÜZERİNDE SONUÇLAR

Bu bölümde geliştirilen sürüm değerlendirme sistemi merkezi işlemci birimi (central processing unit) kullanım oranına dayalı sürüm değerlendirme yapılması amacı ile kullanılmıştır. Çok çekirdekli CPU'larda, her çekirdekteki kullanım oranı ayrı bir metrik olarak tanımlanarak çok boyutlu bir sinyal üzerinde değerlendirme yapılabilir. Bu çalışmada ise sadece bir metrik, birleştirilmiş CPU kullanım oranı, kullanılmıştır. Belirlenen metrik üzerine iki deney yapılmıştır. Deneylerde

kullanılan veri kümelerindeki sinyaller yapay olarak Gaus dağılımlarından örneklemler alınarak oluşturulmuştur. Tüm sinyaller 100 birim uzunluğundadır. Bir sinyal örneği Şekil 2'de görülebilir. Deneylerde kullanılan sinyal kümelerinde 50 adet sinyal bulunmaktadır. Bütün uyumsuzluk hesaplarında, değerlendirmenin başladığı noktayı gösteren k sabiti (bkz. Denklem (4)) 20 olarak alınmıştır. Yani her bir ptSTL formülü için $(100\text{-}20)x50=4000$ adet nicel değerlendirme hesabı yapılmıştır.



Şekil 2. CPU Sinyal kümesinden örnek bir sinyal.

Ortalaması m sapması s olan bir Gaus dağılımı $G(m,s)$ ile gösterilmiştir. Bir veri kümesi üretilirken iki adet Gaus dağılımı kullanılmıştır: normal durum için $G(m^n,s^n)$ ve nadiren oluşan (gürültülü) durumu için $G(m^g,s^g)$. Buna ek olarak bir gürültü oranı parametresi $g \in [0,1)$ kullanılmıştır. Sinyal üretilirken, her bir zaman birimi için $1\text{-}g$ olasılık ile $G(m^n,s^n)$ dağılımından ve g olasılık ile $G(m^g,s^g)$ dağılımından örneklem alınmıştır. Ek olarak, CPU kullanım oranını modelleyebilmek için elde edilen veriler $[0,100]$ arasında tutulmuştur.

Yapılan deneylerde operatör üst limiti 1 olarak belirlenmiştir. Bir metrik için, toplam 8 parametrik ptSTL formülü tanımlanmıştır (Denklem (8)). Izgara araması için zamansal operatörlerin parametrelerin alt ve üst limitleri $\{0,5,10,15,20\}$ kümesinden, sinyal değişkeninin (CPU) karşılaştırma eşik değerleri ise $\{0,5,\ldots,30\}$ kümesinden alınmıştır.

Yapılan ilk deneyde A ve B sürümleri için sinyal kümeleri D_A ve D_B aynı profil ile oluşturulmuştur. Her bir sinyal kümesi için $G(20,10)$ dağılımı normal durumlar için ve $G(60,10)$ dağılımı gürültülü durumlar için kullanılmıştır. Gürültü oranı parametresi ise $g=0.02$ olarak belirlenmiştir. Şekil 2'de D^4 kümesinden bir sinyal gösterilmiştir. Geliştirilen sürüm değerlendirme metodunun uygulanması sonucunda, her iki veri kümesini birden en iyi şekilde tanımlayabilen parametrik formül $\phi^\star=\boldsymbol{G}^{-}_{[p_1,p_2]}\, x > p_3$ olmuştur. Buna ek olarak, iki veri kümesi içinde parametre optimizasyonu sonucunda aynı parametrelere $p_A^\star=p_B^\star=[0,15,5]$ ulaşılmıştır. Bu parametreler ile, veri kümeleri üzerinde elde edilen uyumsuzluk değerleri $C(\phi^\star(p_A^\star),D_A)=3.6907$ ve $C(\phi^\star(p_A^\star),D_B)=3.6917$ olmuştur. Buradaki farklılık, veri kümelerinin rastgele üretilmiş olmasından kaynaklanmaktadır. Bu değerler ile nicel değerlendirme değerinin $(|\rho(\phi,\boldsymbol{x},i)|)$ her iki veri kümesi için de ortalama 3.7 olduğu görülmektedir. Şekil 2'de örneklendirilen $[0,100]$ aralığında değer alabilen bir sinyal kullanıldığı da hesaba katıldığında, elde edilen formüllerin veri kümelerini sıkı bir şekilde tanımladığı görülmektedir. Elde edilen formüller tamamen aynı olduğu için, Denklem (11)'de özetlenen sürüm değerlendirme sonucu olumludur.

İkinci deneyde ise A sürümü normal durumu $G(20,10)$ dağılımı ile B sürümü normal durumu ise $G(25,10)$ dağılımı ile üretilmiştir. Her iki sürüm için de $G(60,10)$ dağılımı ile gürültülü durumlar üretilmiş ve gürültü oranı parametresi $g = 0.02$ olmuştur. Yani, B sürümü için ortalama CPU kullanım oranı beş birim artarak 20'den 25'e çıkmıştır. Yeni sürümde bu artışı tetikleyecek bir değişiklik yapılmamış ise bu beklenmeyen bir durum olarak görülür ve sürüm değerlendirme sisteminin bu konuda bir uyarı üretmesi beklenir.

Geliştirilen sürüm değerlendirme metodu uygulandığında, ilk deneyde olduğu gibi $\phi^\star = G^{-}_{[p_1,p_2]}\, x > p_3$ parametrik ptSTL formülü, her iki sinyal kümesini de en iyi şekilde tanımlayabilen formül olmuştur. Fakat bu deneyde, parametre optimizasyonu sonucunda $p_A^\star = [0,15,5]$ ve $p_B^\star = [0,15,10]$ parametrelerine ulaşılmıştır. Parametrelerden bir tanesi (p_3), B sürümünde A sürümüne oran ile iki katına çıkmıştır. Bu nedenle Denklem (11)'de özetlenen sürüm değerlendirme sonucu olumsuzdur. Olumsuz değerlendirme için kullanıcıya verilen $\phi^{A,\star} = \boldsymbol{G}^{-}_{[0,15]}\, x > 5$ ve $\phi^{B,\star} = \boldsymbol{G}^{-}_{[0,15]}\, x > 10$ formülleri de CPU kullanım oranında ciddi bir artış olduğunu net olarak göstermektedir.

Deneylerde kullanılan sinyal kümeleri üzerinde regresyon analizine dayalı bir sürüm değerlendirme metodu da uygulanmıştır. Bu metotta, öncelikle benzetimli tavlama algoritması kullanılarak D_A kümesindeki sinyalleri tanımlayabilecek 3. dereceden bir sinyal üretilmiştir. Üretilen tanımlayıcı sinyalin sinyal kümesine uyumluluğunu ölçmek için ortalama artık kareler toplamı kullanılmıştır (r_A). Ardından, üretilen bu sinyalin D_B kümesi üzerindeki ortalama artık kareler toplamı r_B hesaplanmıştır. Bu metotta sürüm değerlendirme sonucu r_A ve r_B arasındaki farka göre üretilir. İlk deneyde r_A ve r_B değerleri, sırası ile, 124 ve 125 olarak hesaplanmıştır. İkinci deneyde ise $r_A = 129$ ve $r_B = 176$ olarak hesaplanmıştır. İlk deneyde birbirine yakın değerler gözlenirken, ikinci deneyde r_B'de görülen artış olumsuz bir sürüm değerlendirme sonucu üretilmesi gerektiğini göstermektedir. Geliştirilen metoda göre özetlenen regresyon analizi tabanlı sürüm değerlendirme yaklaşımı, kullanıcıya üretilen sonuç ile ilgili açıklayıcı bir bilgi sunamamaktadır. Bunun yanı sıra, farklı karakterlerdeki sinyaller için tanımlayıcı sinyalin derecesi (3.) ve artık kareler toplamları arasındaki farklar için sınır belirlenmelidir.

5. SONUÇ

Bu çalışmada formel metotlara dayalı bir sürüm değerlendirme metodu sunulmuştur. Önerilen sürüm değerlendirme metodu, takip edilen metrikler ile ilgili yazılabilecek bütün parametrik ptSTL formüllerinin sıralanması ve her bir parametrik formül için A ve B versiyonlarından üretilmiş olan sinyal kümeleri üzerinde parametre optimizasyonu yapılması adımlarını içermektedir. Son olarak da iki versiyonu da en iyi şekilde tanımlayabilen parametrik formül için üretilen parametre değerleri karşılaştırılarak sürüm değerlendirme sonucu üretilmiştir. Önerilen metot ile sürüm değerlendirmede kullanılmak üzere ptSTL formüllerinin otomatik olarak üretilmesi sağlanmıştır.

Hakem Değerlendirmesi: Dış bağımsız.
Çıkar Çatışması: Yazar çıkar çatışması bildirmemiştir.
Finansal Destek: Bu çalışma, TÜBİTAK tarafından desteklenmiştir. (Proje No: 117E242)

Peer-review: Externally peer-reviewed.
Conflict of Interest: The author has no conflict of interest to declare.
Grant Support: This study was supported by TÜBİTAK. (Project No: 117E242)

Kaynaklar/References

Asarin E., Donzé A., Maler O. & Nickovic D. (2012). Parametric Identification of Temporal Properties. In: Khurshid S., Sen K. (eds). *Lecture Notes in Computer Science, vol 7186: Proceedings of Runtime Verification* (pp. 147-160). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-29860-8_12.

Aydin, S. K. & Aydin Gol, E. (2020). Synthesis of monitoring rules with STL. *Journal of Circuits, Systems, and Computers*, *29(11)*, 1-26. https://doi.org/10.1142/S0218126620501777.

Bartocci E., Bortolussi L. & Sanguinetti G. (2014) Data-Driven statistical learning of temporal logic properties. In: Legay A., Bozga M. (eds). *Lecture Notes in Computer Science, vol 8711: Proceedings of Formal Modeling and Analysis of Timed Systems* (pp 23-37). Springer, Cham. https://doi.org/10.1007/978-3-319-10512-3_3.

Bays, M. E. (1999). *Software Release Methodology*. USA: Prentice-Hall.

Bombara, G., Vasile, C.-I., Penedo, F., Yasuoka, H. & Belta, C. (2016). A decision tree approach to data classification using signal temporal logic. *Proceedings of the Hybrid Systems: Computation and Control*, 1–10. https://doi.org/10.1145/2883817.2883843.

Chatterjee, S. & Simonoff, J. S. (2013). *Handbook of Regression Analysis*. Wiley.

Donzé A. (2013). On signal temporal logic. In: Legay A., Bensalem S. (eds). *Lecture Notes in Computer Science, vol 8174: Proceedings of Runtime Verification* (pp 382-383). Berlin: Springer. https://doi.org/10.1007/978-3-642-40787-1_27.

Ergurtuna, M. & Aydin Gol E. (2019). An efficient formula synthesis method with past signal temporal logic. *Proceedings of the IFAC Conference on Intelligent Control and Automation Sciences (ICONS)*, 43-48. https://doi.org/10.1016/j.ifacol.2019.09.116.

Gabbay D. (1989) The declarative past and imperative future. In: Banieqbal B., Barringer H., Pnueli A. (eds) Temporal Logic in Specification. Lecture Notes in Computer Science, vol 398 (pp. 409-448). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-51803-7_36.

Howard, D. (2016). *IT Release Management*. O'Reilly.

Hoxha, B., Dokhanchi, A. & Fainekos, G. (2018). Mining parametric temporal logic properties in model-based design for cyber-physical systems. *Int J Softw Tools Technol Transfer 20*, 79–93. https://doi.org/10.1007/s10009-017-0447-4 .

Jha, S., Tiwari, A., Seshia, S.A., Sahai, T. & Shankar, N. (2019). TeLEx: learning signal temporal logic from positive examples using tightness metric. *Formal Methods in System Design 54*. 364–387. https://doi.org/10.1007/s10703-019-00332-1.

Jin, X., Donzé, A., Deshmukh, J. V. & Seshia, S. A. (2015). Mining requirements from closed-loop control models. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 34, 11.* 1704-1717. https://doi.org/10.1109/TCAD.2015.2421907.

Ketenci, A. & Aydin Gol, E. (2019). Synthesis of monitoring rules via data mining. *Proceedings of the IEEE American Control Conference (ACC)*, 1684-1689. https://doi.org/10.23919/ACC.2019.8815002.

Kong, Z., Jones, A., Ayala, A. M., Aydin Gol, E. & Belta, C. (2014). Temporal logic inference for classification and prediction from data. *Proceedings of the Hybrid Systems: Computation and Control*, 273–282. https://doi.org/10.1145/2562059.2562146.

Mohammadinejad, S., Deshmukh, J. V., Puranic, A. G., Vazquez-Chanlatte, M. & Donzé, A. (2020). Interpretable classification of time-series data using efficient enumerative techniques. *Proceedings of the Hybrid Systems: Computation and Control*, 1–10. https://doi.org/10.1145/3365365.3382218.

Sommerville, I. (2015). *Software Engineering (10th ed.)*. Pearson.

Yoo, C & Belta, C. (2017). Rich time series classification using temporal logic. *Proceedings of Robotic: Science and Systems.* 1-9.