Hibrit Olasılıksal Zamanlama Analizi

Hybrid Probabilistic Timing Analysis

Levent BEKDEMİR
Aselsan A.Ş.
P.K.1 06200, Yenimahalle (Ankara, Türkiye)
lvntbkdmr@gmail.com

Cüneyt F. BAZLAMAÇCI
İzmir Yüksek Teknoloji Enstitüsü
Urla 35430 (İzmir, Türkiye)
cuneytbazlamacci@iyte.edu.tr

***Özetçe* —Zaman-kritik sistemlerde mevcut güçlüklerin başında bir işin gereken son bitirilme zamanından önce tamamlanacak olmasının garanti edilmesi hususu gelir. Bu amaçla yazılımın tümünün davranışının çözümlenmesi gerekir. En kötü yürütme süresi bir yazılım biriminin olası en yüksek yürütme zamanını temsil eder ve bu metrik zaman-kritik sistemlerin kaynak planlamasında kullanılır. İlgili alandaki yakın zamanlı çalışmalar daha çok istatistiksel yaklaşımlara yer vermiştir. Bunlar, ölçüme dayalı zamanlama analizlerini stokastik yöntemler yardımıyla elde edilen olasılıksal güvenirlik seviyeleriyle tamamlamaktadırlar. En çok kullanılan yöntemlerde, ya uçtan uca ölümleri uç değer teorisi yardımıyla kullanarak ya da küçük program parçacıklarının ölçümlerini evrişim (convolution) teknikleri kullanarak tüm programın üst sınırı belirlenmeye çalışılmaktadır. Her iki yaklaşım da sorunludur. Bu çalışmada bir hibrid olasılıksal zamanlama analiz metodu önerilmektedir. Bu metod ile program yapı taşı olan alt birimler ayrı ayrı ve uç değer teorisi yardımıyla modellenerek en üst değerler yakalanabilmekte, kopulalar kullanarak da birimler arası bağımlılıklar modellenmekte ve sonuçta daha iyi bir sınır değer dağılımı bulunabilmektedir.**

Anahtar Kelimeler*—*en kötü yürütme süresi analizi, ölçüm-temelli olasılıksal zamanlama analizi, kopula teorisi, uç değer teorisi, istatistiksel zamanlama analizi.

***Abstract*—A major challenge in time-critical systems is ensuring that a job will be completed before the required deadline. For this purpose, the behavior of the entire software needs to be analyzed. Worst execution time represents the highest possible execution time of a software unit and this metric is used in resource planning of time-critical systems. Recent studies in the related field have mostly included statistical approaches. They complement their measurement-based timing analysis with probabilistic confidence levels obtained with the help of stochastic methods. In the most used methods, the upper limit of the whole program is tried to be determined either by using end-to-end measurements with the help of extreme value theory or by using convolution techniques to measure small program units. Both approaches are problematic. In this study, a hybrid probabilistic timing analysis method is proposed. With this method, the subunits that are the building blocks of the program can be modeled separately and the highest values can be captured with the help of extreme value theory, and then the dependencies between the units are modeled by using copulas and a better boundary value distribution can be found as a result.**

Keywords*—*worst execution time analysis, measurement-based probabilistic timing analysis, copula theory, extreme value theory, statistical timing analysis.

2021 15th Turkish National Software Engineering Symposium (UYMS) | 978-1-6654-1070-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/UYMS54260.2021.9659797

I. GİRİŞ

Güvenlik açısından kritik gerçek zamanlı sistemlerin en ayırt edici özelliği, yanıtlarını/sonuçlarını kesin olarak tanımlanmış bir süre içinde verebilmeleridir. Bu tür sistemlerin sistem seviyesi gereksinimleri yanında güvenlikle ilgili gereksinimlerini de karşılayabilmek için yazılım birimlerinin zamanlama özelliklerinin tahmin edilmesine ihtiyaç duyulur.

Teknolojideki son gelişmeler bilgisayar sistemlerinde zamanlama analizini güçleştirmiştir. *Çok çekirdekli yapılar*, *paylaşımlı veri yolları*, *boru hattı mimarileri*, *sıra-dışı-yürütme birimleri*, *dallanma tahmin birimleri* ve *ön bellek* gibi modern işlemcilerin performans artırıcı özellikleri yürütme süresini yürütme geçmişine bağımlı hale getirmiştir. Bu mimari iyileştirmeler, geleneksel statik analiz tekniklerinin uygulanabilirliğini azaltmış ve aynı kod için yürütme süresinin değişken olmasına neden olmuştur. Yürütme süresindeki bu değişkenlik öte yandan istatistiksel yöntemlerin en kötü yürütme süresi (EKYS) (*ing.* worst-case execution time (WCET)) analizinde uygulanabilmesine olanak sağlamıştır. Tüm olası yürütme yollarının kapsanmasına gereksinim duymadan, birkaç ölçüm ve ardından gelen istatistiksel analiz yardımıyla güvenli EKYS tahminleri elde etmek ve böylece sistemin en kötü zamanlama davranışını öngörmek mümkündür.

Olasılık dağılımlarının evrişimi (*ing.* convolution) ve uç değer teorisi (UDT) (*ing.* extreme value theory (EVT)), istatistiksel zamanlama analizi alanında kullanılan iki ana yaklaşımdır. Mevcut evrişim yaklaşımları, gerçekte olması mümkün olmayan yürütüm yollarını da ürettiği için sonuçlarda gereğinden büyük tahminlere yol açmaktadır. Öte yandan, literatürdeki çalışmaların çoğu, yürütme yolu kapsama probleminden muzdarip olan programların uçtan uca ölçümlerine UDT uygulamaktadır. Diğer bir gözlemimiz de, literatürde rafta-hazır (*ing.* common-of-the-shelf (COTS)) platformlar için EKYS analizi alanında çok sınırlı sayıda çalışma bulunmasıdır.

Bu makalenin amacı, ticari kullanıma uygun COTS platformlar için endüstriyel olarak da uygulanabilir ve güvenilir bir hibrit olasılıksal zamanlama analizi (*ing.* hybrid probabilistic timing analysis (HYPTA)) yaklaşımı önermektir. Bu yaklaşımımızda analiz edilen program fonksiyonel bloklara bölünerek statik yapısal bilgi çıkarılmakta, bu sayede programları temel bloklara bölerek yapılan ölçümlere kıyasla oluşan olumsuz yan etkiler azaltılabilmektedir. Toplanan ölçümlerle birlikte bu yapısal bilgi tüm olası yürütme senaryolarını üstten sınırlayabilmek amacıyla sanal yolla yeni yürütme yolları üreterek bir olasılıksal EKYS (*ing.* probabilistic worst-case execution time

(pWCET)) dağılımı oluşturmak amacıyla kullanılmaktadır.

Mevcut son teknoloji çözümler ya program alt parçaları (bloklar, kapsamlar) arasında bağımsızlık varsayar ya da bloklar arasındaki olası tüm bağımlılık türlerini modelleyebilmek ve üstten sınırlayabilmek amacıyla görece muhafazakar bir evrişim yaklaşımı kullanır. Esasen rastgele değişkenler arasındaki bağımlılığı kopulalar kullanarak modellemek de mümkündür. Kopulalar, tekdüze marjinallere sahip ortak olasılık dağılım fonksiyonlarıdır. Bu nedenle, Monte-Carlo simülasyon tekniği ile rastgele değişkenlerin ortak davranışını simüle etmek mümkündür ve bu sayede rastgele değişkenlerin toplamlarının olasılık dağılımı türetilebilir.

n boyutlu olasılık dağılımının her bir marjinali, mümkün olduğu durumlarda birer parametrik sürekli uç değer dağılımı ile temsil edilebilir. Önerdiğimiz yöntemimizde programları esasen birden fazla yollar içerebilen işlevsel bloklara böldüğümüz ve bu işlevleri UDT ile modellemek mümkün olduğu için olasılık dağılımın kuyruk bölümündeki davranışa ilişkin olası nadir olayları tahmin etmek de mümkün olmaktadır.

Önerilen HYPTA yöntemimiz, Rapita Inc. tarafından geliştirilmiş olan ve muhafazakar bir evrişim mekanizması kullanan RapiTime [1] ticari ürününe kıyasla da daha iyi sonuç vermektedir. Vaka çalışmalarımızın sonuçları, RapiTime sonuçları ve yaygın olarak kabul edilen diğer yaklaşımlarla karşılaştırılmış ve yöntemimizin daha sıkı sınırlar sağladığı ve gereğinden büyük tahminleri azalttığı gösterilmiştir.

Makalenin geri kalanı şu şekilde düzenlenmiştir: 2. Bölümde arka plan bilgileri ve literatüre genel bir bakış sunulmaktadır. UDT ve kopulalar kullanan hibrit yöntemimiz ve vaka çalışmalarının sonuçları Bölüm 3'te verilmektedir. Bölüm 4, çalışmanın kısa bir özetini ve sonuçlarını içermektedir.

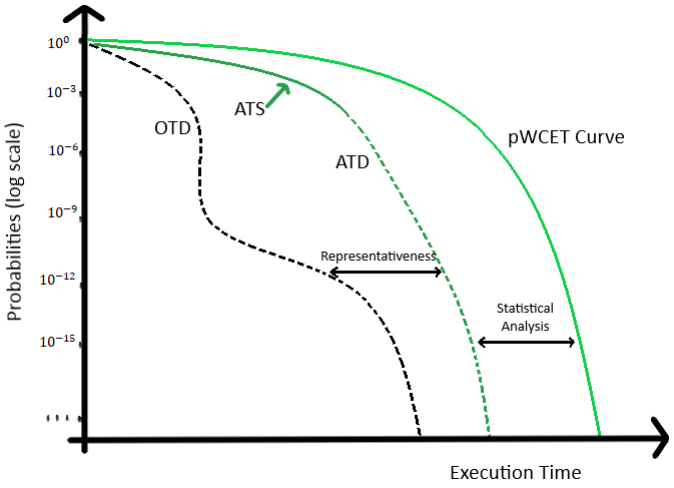
II. Temel Kavramlar

Statik olasılıksal zamanlama analizi (SPTA) teknikleri, platform hakkında eksik bilgiden kaynaklanan muhafazakarlığı, bazı platform davranışlarını olasılıklı olarak ifade ederek azaltmayı amaçlar. Ölçüme dayalı olasılıksal zamanlama analizi (MBPTA), programın yürütme süresi davranışını yaygın olarak kabul edilen istatistiksel yöntemlere dayalı olarak modelleyerek analiz aşamasında yakalanan en kötü durumlar hakkında bilimsel biçimde akıl yürütmeyi amaçlar. MBPTA'nın çıktısı, tek değerli bir EKYS değil, programın yürütme süresi profilinin, tüm analiz süresi gözlemlerini üstten sınırlayacağı garanti bir olasılık dağılımıdır. Şekil 1 MBPTA'nın temel kavramlarını sunmaktadır.

MBPTA'e girdi olarak, çalışma zamanı dağılımını (OTD) üstten sınırlayan bir analiz zamanı dağılımı (ATD) verilir. Her iki dağılım arasındaki farkı en aza indirgemeye *temsiliyet* denir. MBPTA yöntemlerinin uygulanabilirliği, az sayıda analiz zamanı örneği (ATS) ile tüm ATD'yi temsil edebilmeyi gerektirir. Görece küçük bir analiz zamanı örnekleri (ATS) kümesi kullanılarak, hem OTD hem de ATD için bir üst sınır pWCET dağılımı hesaplanır.

A. Uç Değer Teorisi

UDT aşırı büyük sel, kasırga ve depremler gibi olağandışı doğal olayları tahmin etmek için tasarlanmıştır ve MBPTA'nın da yapı taşı olarak bilinir. Bu uç olaylar, olasılık dağılımları



Şekil 1: Ölçüme dayalı olasılıksal zamanlama analizi. (MBPTA) temel kavramları [2].

olarak modellenir ve UDT, ilgilenilen olaya ilişkin gözlemlerin medyanından aşırı sapmalarla ilgilenir. Tipik bir UDT uygulaması adım adım şu şekilde yapılabilir [3]. 1) *Uygulanabilirlik Kanıtı*: Aynı şekilde dağıtılmış bağımsız (*ing.* independent and identically distributed (iid)) girdi gereksiniminin sağlanıp sağlanmadığı kontrol edilir ve elde edilen örneklerin kabul edilip edilmeyeceği belirlenir. 2) *Veri Seçimi*: Veri setinden dağılımın kuyruğunu temsil eden değerler, blok maxima ya da eşik üstü zirve (*ing.* peak-over-threshold) yaklaşımları [4] kullanılarak seçilir. 3) *Model Uydurma*: veri seçim mekanizmasına bağlı olarak, filtrelenen değerler ya genelleştirilmiş uç değer dağılımı (GEV) ya da genelleştirilmiş pareto dağılımı (GPD) biçiminde modellenir. 4) *Kuyruk Uzantısı*: model parametrelerini kullanarak örnek kuyruğun dağılımı bulunur. Verilen bir aşma olasılığına karşılık gelen uç değerleri hesaplamak için tahmin edilmiş olan olasılık dağılımının ters kümülatif dağılım fonksiyonu (ICDF) kullanılır. p aşma olasılığı olarak alındığında, $ICDF(p)$, aşılma olasılığı p olan üst sınır değeri x'i verir.

UDT ile tahmin edilen dağılımın kalitesi değerlendirilmesi zor bir husustur. Olasılıksal EKYS analizi hakkındaki son kapsamlı araştırmalar bu konuda evrensel bir fikir birliği olmadığına işaret etmektedir [2]. Güvenilirliği bazı istatistiksel testler ile değerlendirmek mantıklı olsa da UDT'den elde edilen tahminlerin yalnızca UDT'nin her hipotezinin doğrulanması durumunda güvenilir olduğunu düşünen yazarlar mevcuttur [5]. Tahmin edilen dağılımların güvenilirliğini değerlendirmek için quantile-quantile ya da mean-excess gibi standart istatistiksel araçları kullanmayı öneren yazarlar da vardır [6].

B. Kopula Teorisi

İstatistikte, birkaç rastgele değişkenin bağımlılığını modellemek için kopulalar da kullanılabilmektedir. Bir kopula temel olarak tekdüze marjinallere sahip çok değişkenli bir olasılık dağılımıdır.

X_1 ve X_2, kümülatif dağılım fonksiyonları (CDF) F_1 ve F_2 olarak tanımlanan iki farklı rastgele değişken olsun:

$$F_1(x_1) = P[X_1 \leq x_1]$$

$$F_2(x_2) = P[X_2 \leq x_2]$$

H de X_1 ve X_2 aşağıdaki gibi tanımlanmış ortak kümülatif dağılım fonksiyonu olsun:

$$H(x_1, x_2) = F_{X_1,X_2}(x_1, x_2) = P[X_1 \leq x_1, X_2 \leq x_2]$$

Burada H, rastgele değişkenlerin ortak davranışının tüm yönlerini temsil eder, ancak bağımlılık yapısını H'den çıkarmak zordur. Kopulalar, $F_1(x_1)$ ve $F_2(x_2)$ tarafından tanımlanan marjinallerin bağımlılık yapısını ve davranışını ayırdetmeyi mümkün kılar. Korelasyon, iki rastgele değişken arasında düz bir çizgi biçimindeki yalnızca bir tür bağımlılıktır.

I, $[0, 1]$ aralığını temsil etsin. d-boyutlu bir kopula C, I^d üzerinde tekdüze marjinallere sahip ve aşağıdaki genel bir notasyonla ifade edilen kümülatif bir dağılım fonksiyonudur.

$$C(\mathbf{u}) = C(u_1, u_2, ..., u_d)$$

Sklar teoremi [7], kopulalarla ilgili en önemli sonuçtur ve H ortak dağılımı ile bir kopula C arasındaki ilişkiyi açıklar.

F, marjinalleri $F_1, ..., F_d$ olan d boyutlu bir ortak dağılım fonksiyonu olsun. O zaman, ${F_1}^{(-1)}, ..., {F_d}^{(-1)}$ marjinal dağılım fonksiyonlarının sözde-tersini (*ing.* quasi-inverse) temsil etmek üzere, I^d üzerinde bir C kopulası vardır ki, tüm $x_1, ..., x_d \in R$ için

$$F(x_1, ..., x_d) = C(F_1(x_1), ..., F_d(x_d))$$

veya

$$C(u_1, ..., u_d) = F({F_1}^{(-1)}(u_1), ..., {F_d}^{(-1)}(u_d))$$

C. Ölçüme Dayalı Zamanlama Analizi Literatür İncelemesi

Ölçüme dayalı zamanlama analizi (MBTA)ile ilgili çok sayıda araştırma vardır. [3] nolu kaynak MBPTA'nın temeli olarak kabul edilir.

Tüm programın EKYS'sini küçük blokların gözlemlerinden türetmek fikri ile taraflı evrişim (*ing.* biased convolution) yönteminin kullanımı [1] önemli bir dönüm noktası olup, bu çalışma daha sonra ilk endüstriyel pWCET aracına evrilmiştir [8]. MBTA'nın olasılıksal versiyonu MBPTA hakkında kapsamlı inceleme çalışmaları da mevcuttur [2], [9].

III. UDT ve Kopula ile Hibrit Olasılıksal Zamanlama Analizi

A. Mevcut Teknolojik Durum

HYPTA alanındaki çalışmalar görece az sayıdadır [10]. Rapita Systems Ltd. tarafından geliştirilen RapiTime aracı [2]'da hibrit bir MBPTA olarak değil, hibrit bir MBTA çözümü olarak tanımlanmıştır ve aracın pWCET kavramını öngördüğü, ancak dağılımı bulmak için tahmine dayalı bir model uygulamadığı söylenmekte ve MBPTA'dan ziyade ölçüme dayalı bir SPTA olarak düşünülmelidir denmektedir. Bununla birlikte, [11], test edilen programın statik yapısal özelliklerini ölçümlerle birleştirdiği için bu aracın olasılıksal yaklaşım kısmı tartışmalı olarak HYPTA kategorisine girdiği de belirtilmiştir [8].

[12]'de, EKYS'yi ağaç düğümlerinin bir fonksiyonu olarak değerlendirmek için programın *zamanlama şeması* olarak adlandırılan bir dizi kural tanımlanmış ve [13]'de bunun olasılıksal bir versiyonu sunulmuştur. Düğümlerin yürütme süresi X_i rasgele değişkenleri ile temsil edildiğinde, problem sıralı bloklar için $(X_1 + X_2 + .. + X_n)$ hesaplamasına indirgenir. X_is'nin birbirinden bağımsız olduğu varsayılırsa, sonuç da dağılımların standart evrişimi ile kolayca elde edilir. Ancak bu varsayım, özellikle tam pozitif (komonotonik) veya tam negatif (karşı-komonotonik) bağımlı durumlar için gerçekte geçerli değildir.

[13] bu sorunu kopula kullanarak çözmeyi amaçlamış ve bloklar arasında komonotoniklik varsayımı ile evrişim yaklaşımı kullanmıştır. Ancak temel bloklar arasındaki tüm bağımlılık türleri için komonotonik evrişim kullanıldığı için karşı-komonotonik bir durum için (mükemmel negatif bağımlılık), komonotoniklik varsayımı aşırı büyük bir tahminle sonuçlanabilmektedir.

Uygulamada da, her küçük temel blok için ölçüm almak mümkün değildir. COTS platformları için makul çözüm, blokların büyüklüğünü artırmaktır, bu da büyük blokların olası dallanmalar içermesi nedeniyle varyans sorununu beraberinde getirir.

Fonksiyonel seviyede ölçüm noktaları eklendiğinde komonotoniklik varsayımından ve nadir olay yakalama hususundaki eksiklikten kaynaklanan pWCET'in gereğinden fazla tahmin edilmesi sorunu mevcut çalışmanın ana araştırma odağı olmuştur.

B. Önerilen Yöntem

UDT ve kopulalar kullanarak ampirik bir portföyün riskemaruz değer (*ing.* value-at-risk (VaR))'ini tahmin etmek için prosedürel bir yaklaşım önerilmiştir [14]. Bu yöntemde izlenen adımlar şu şekildedir: 1) her bir portföy unsurunun yatırım getirisini, kuyruk bölümlerini GPD, ara kısmı çekirdek dağılımı ile temsil ederek yarı parametrik parçalı dağılım (SPD) şeklinde modelle, 2) her bir dağılımı düzgün aralığa dönüştür, 3) birbiçimli marjinal dağılımlara birer t-kopula bul, 4) t-kopula üretecinden çok sayıda birbiçimli değer üret, 5) birbiçim değişkenleri orijinal hallerine geri dönüştür (ters dönüşüm örneklemesi [15]), 6) toplam VaR'ı elde etmek için ağırlıklı bir toplam gerçekleştir, 7) Son olarak, istenen VaR değerini, verilen α güven düzeyi ile hesapla.

3. adımdan sonrası Monte-Carlo Simülasyonunu temsil eder. Yukarıdaki VaR analizi ile bizim hedefimiz olan EKYS analizi arasında bir analoji kurulabileceğini; yatırımların getirilerinin blokların yürütme zamanına, VaR'ın da programımızın olasılıksal EKYS'ine karşılık geldiğini gözlemlemiş bulunuyoruz.

Yaklaşımımızdaki en önemli adım, analiz edilen programın blokları arasındaki bağımlılığı temsil eden kopula modelinin türetilmesidir. Bir kopula elde etmek için, marjinallerin aynı boyutlara sahip olması gerekir, yani her bloğun gözlemleri aynı anda alınmalıdır. Bu, özellikle koşullu ve yinelemeli bloklar için önemli bir kriterdir, çünkü programın tek bir çalışmasında sıralı bloklar bir kez ziyaret edilirken koşullu veya yinelemeli bloklar sıfır veya birkaç kez ziyaret edilebilir, bu nedenle koşullu ve yinelemeli bloklar için özel bir değerlendirme yapılması şarttır. Bir kapsam ya tüm işlev gövdesinin kendisi ya da bir işlevin koşullu veya yinelemeli parçaları

olabilir. Şekil 2, "if-else bloğunun" tamamının bir kapsamı ve "for bloğu"nun tamamının da başka bir kapsamı temsil edebileceğini göstermektedir.

Şekil 2: Örnek bir programda olası kapsamlar.

Hibrit metodumuzun adımları aşağıda verilmiştir:

1) Kaynak kodundaki fonksiyonların giriş ve çıkış noktalarına ölçüm noktaları (*ing.* instrumentation points (IPoint ya da IP)) yerleştir.
2) IP'leri kullanarak [16]'da açıklandığı gibi programın zamanlama şemasını türet.
3) Yinelemeli ve koşullu bloklar içinde çağrılan fonksiyonları işaretle ve kapsamları belirle.
4) İşlevsel blokların (IP'ler ağacı içindeki her düğüm) yürütme süresini temsil eden rastgele değişkenleri belirle.
5) En içteki kapsamdan en dıştakine olacak şekilde, sıralı bloklar için toplanması gereken rastgele değişkenleri içeren her bir kapsam için uygun kopula bul.
6) Monte-Carlo yaklaşımını ve kopulaları kullanarak kapsamların sonraki n yürütmelerini simüle et.
7) Her rastgele değişkenin ICDF'ini türet.
8) Simülasyonlardan üretilen her tekdüze marjini, ICDF'leri kullanarak orijinal haline dönüştür.
9) Kapsamın genel dağılımını elde etmek amacıyla tüm marjinaller için bir toplam işlemi gerçekleştir.
10) Analiz edilecek hiçbir kapsam kalmayıncaya kadar $2-7$ arasındaki adımları tekrarla.

Yaklaşımla ilgili ayrıntılar ve ek bilgiler [17] ve [18]'de sunulmuştur.

IV. Değerlendirme

A. Deney Düzeneği

Deneysel değerlendirme aşamasında LEON3 SPARC V8 işlemcisinin hataya dayanıklı bir versiyonu olan LEON3FT tabanlı bir ASIC platformu kullanılmıştır.

LEON3FT, yüksek performans, düşük karmaşıklık ve düşük güç tüketimi gerektiren gömülü uygulamalar için tasarlanmıştır. 64 MHz'de çalışır ve standart LEON3 işlemcinin, çip üzerindeki RAM belleklerinde hata algılama ve düzeltme dahil olmak üzere birçok işlevselliğini destekler. Platformun yazılım tarafında, gerçek zamanlı işletim sistemi (RTOS) olarak RTEMS bulunmaktadır. Ölçüm amacıyla, program her ölçüm noktasına geldiğinde özel GPIO sinyalleri üretilir ve bu sinyaller özel olarak geliştirdiğimiz bir harici donanım cihazı tarafından yakalanarak zaman damgalanır. Uçtan uca ölçümler yerine her bir kapsam için ayrıntılı ölçümler alınması gerektiğinden ve prob etkilerini de azaltmak için böyle bir çözümün kullanılması zorunlu olmuştur.

İstatistiksel hesaplama ve grafikler için ise güçlü görselleştirme yeteneklerine sahip R dili ve ortamı kullanılmıştır.

IP'lerin nasıl uygulanacağı hususu analiz ortamına bağlıdır. Çalışmamızda IP bir makro olarak gerçeklenmiştir ve test edilen program ilgili IP noktasına geldiğinde, platformun bir genel amaçlı giriş çıkış (GPIO) portuna IP'yi tanımlayan kimlik numarası yazılmaktadır. Ayrıca bu çıktının en soldaki biti değiştirilerek tasarlamış olduğumuz harici zamanlama donanımının bu bilgiyi zaman etiketleyerek kaydetmesine yönelik sinyalleşme sağlanmaktadır.

Şekil 3'te ölçüm noktaları (IP'ler) yerleştirilmiş basit bir örnek program sunulmuştur. İlgili rastgele değişkenler ve bunlara karşılık gelen IP'lerin zaman şeması da aşağıda verilmiştir.

```
void insertSortAsc(x) {
  IPoint(29);
  //sort x in ascending order
  IPoint(28);
}
void insertSortDesc(x) {
  IPoint(27);
  //sort x in descending order
  IPoint(26);
}
void testProgram() {
  IPoint(31);
  insertSortAsc(x);
  insertSortDesc(x);
  IPoint(30);
}
```

Şekil 3: Örnek ölçüm programı.

$$W(insertSortAsc) = W(IPoint_{29-28}) \quad (1)$$
$$W(insertSortDesc) = W(IPoint_{27-26}) \quad (2)$$
$$W(testProgram_{entry}) = W(IPoint_{31-29}) \quad (3)$$
$$W(insertSortAsc_{ret}) = W(IPoint_{28-27}) \quad (4)$$
$$W(insertSortDesc_{ret}) = W(IPoint_{26-30}) \quad (5)$$

$$W(testProgram) = \underbrace{W(testProgram_{self})}_{\text{X}} + \underbrace{W(testProgram_{sub})}_{\text{Y}} \quad (6)$$

$$W(testProgram_{self}) = \underbrace{W(testProgram_{entry})}_{\text{X}} + \underbrace{W(insertSortAsc_{ret})}_{\text{Y}} + \underbrace{W(insertSortDesc_{ret})}_{\text{Z}} \quad (7)$$

$$W(testProgram_{sub}) = \underbrace{W(insertSortAsc)}_{\text{X}} + \underbrace{W(insertSortDesc)}_{\text{Y}} \quad (8)$$

B. Örnek Vaka Analizi

Örnek vaka analizi kapsamında ilk olarak ilgili fonksiyonları Mälardalen WCET Benchmarks'tan [19] seçerek aşağıdaki sentetik ve tekrarlanabilir kıyaslama programı oluşturulmuştur (Şekil 4). Programın yürütme süresine ait denklemler de 9 - 14 arasında gösterilmiştir.

```
void testProgram(void)
{
   float result = 0;
   int swapCnt = 0;
   vector vecA_tmp;

   vecA_tmp = f1(vec_A, 10); //select

   swapCnt = f2(vecA_tmp); //insertsort

   f3(vec_A); //insertsort (reverse)

   if (_scale > 50)
   {
      f4(vec_A, vec_C, vec_B, _scale); //fir
   }
   else
   {
      f5(mat_A, mat_B, mat_C); //matmult
   }

   int maxCnt = 100 - (int)(swapCnt/51);
   for (int i = 0; i < maxCnt; i++)
   {
      f6(vec_B[i]); //sqrt
   }
}
```

Şekil 4: Vaka analizi için örnek program.

vec_A, vec_B, mat_A ve mat_B girdileri rastgele oluşturulmaktadır. vec_i ve mat_i aynı nesneler olup, boyutları farklıdır.

Bu programın toplam yürütme süresi de 6 denklemi ile gösterilebilir. Ancak, $W(testProgram_{self})$ ve $W(testProgram_{sub})$ ifadeleri bu durumda 7 ve 8 denklemlerinde olduğu gibi kolayca ayrıştırılamaz çünkü dikkatli incelenmesi gereken koşullu ve yinelemeli bloklar içerir.

Yöntemimizde koşullu ve yinelemeli bloklar, programın en üst seviyesinde tek bir kapsam olarak ele alınmaktadır. Bu yaklaşım, alt kapsamlar arasındaki bağımlılığı modelleyen kopulaları türetmek için gereklidir.

$testProgram_{sub}$ için A, B, C, D ve E'yi oluşturduktan sonra, sırasıyla karşılık gelen alt kapsam gözlemleri (u, v, k, m ve n) arasındaki korelasyon Şekil 5'de gösterilmiş olup, aynı prosedürler $testProgram_{self}$ ve $testProgram$ için de uygulanmıştır.

$testProgram$, $testProgram_{self}$ ve $testProgram_{sub}$ kapsamlarına, R'nin `RVineStructureSelect` işlevi kullanılarak bir Vine Kopula modeli uyarlanmış ve her birine uyum testleri uygulanmış ve tüm bulunan kopulaların geçerli olduğu gözlenmiştir.

Vaka analizi için nihai sonuç Şekil 6'da gösterilmektedir ve yöntemimizin diğer tüm yöntemlere göre en küçük tahmini verdiği görülmektedir.

$$W(testProgram_{sub}) = \underbrace{W(f1)}_{\text{A}} + \underbrace{W(f2)}_{\text{B}} + \underbrace{W(f3)}_{\text{C}} + \underbrace{W(cond_{sub})}_{\text{D}} + \underbrace{W(loop_{sub})}_{\text{E}} \quad (9)$$

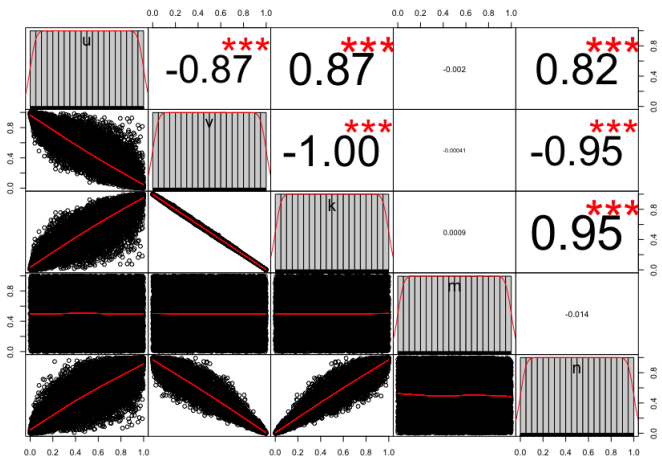
$$W(testProgram_{self}) = \underbrace{W(f1_{ret})}_{\text{A}} + \underbrace{W(f2_{ret})}_{\text{B}} + \underbrace{W(f3_{ret})}_{\text{C}} + \underbrace{W(cond_{self})}_{\text{D}} + \underbrace{W(loop_{self})}_{\text{E}} \quad (10)$$

$$W(cond_{sub}) = max(\underbrace{W(f4)}_{\text{A}}, \underbrace{W(f5)}_{\text{B}}) \quad (11)$$

$$W(cond_{self}) = max(\underbrace{W(f4_{ret})}_{\text{A}}, \underbrace{W(f5_{ret})}_{\text{B}}) \quad (12)$$

$$W(loop_{sub}) = \underbrace{\overbrace{W(f6) + ... + W(f6)}^{\text{A}}}_{\text{N}} \quad (13)$$

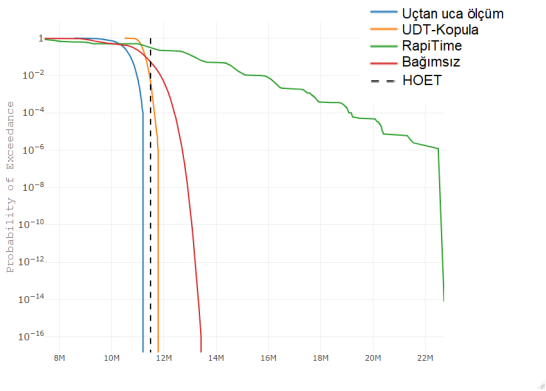
$$W(loop_{self}) = \underbrace{\overbrace{W(f6_{ret}) + ... + W(f6_{ret})}^{\text{A}}}_{\text{N}} \quad (14)$$



Şekil 5: $testProgram_{sub}$ için korelasyon çizimi.

Döngü bloğunun yürütme süresi dağılımını hesaplamak için standart evrişim kullanıyor olmamıza rağmen sonucumuz (UDT-Kopula), her kapsamı bağımsız varsayan yaklaşımın yine de altındadır. Kullandığı komonotonik varsayım nedeniyle ticari RapiTime aracının da, sonucu oldukça fazla biçimde tahmin ettiği gözlemlenmektedir.

Tablo I, tahmin edilen dağılımlardan elde edilen bazı pWCET değerlerini özetlemekte ve RapiTime yaklaşımının, pWCET(10^{-9}) durumu için uçtan uca yürütme süresini 100%'den fazla tahmin ettiğini göstermektedir. Bu fazlalık



Şekil 6: Örnek vaka çalışması için tahmini pWCET dağılımı.

sınırlı zamanlama kaynaklarına sahip gerçek zamanlı gömülü yazılım ortamları için kabul edilemez. Önerdiğimiz yöntemle elde ettiğimiz sonuç ise hem kuyruk modelleme metodolojisine (UDT) dayalı olarak güvenlidir, hem de önerilen bağımlılık modelleme yaklaşımının (kopulalar) yardımıyla daha sıkı bir değerdir.

TABLO I: ÖRNEK VAKA ÇALIŞMASI İÇİN HESAPLANAN PWCET DEĞERLERİ

pWCET	UDT-Kop	Bağımsız	RapiTime	Uçtan uca
10^{-2}	57.2	60.2	79.1	54.8
10^{-4}	58.1	63.7	95.8	56.3
10^{-6}	58.6	67.0	113.6	56.3
10^{-9}	58.7	68.3	113.6	56.3

Sonucumuzun sıkılığını ve güvenilirliğini görebilmek adına aynı deney düzeneğinden ilave 10^6 adet uçtan uca ölçüm alınmış ve bu vaka çalışması için gözlenen en yüksek yürütme süresi (HOET) 57.44 ms olmuştur. Yöntemimizin 'bağımsız've 'komonotonik' varsayımlara kıyasla hem daha küçük değerler sunduğu hem de güvenilir olduğu, RapiTime yaklaşımının ise büyük bir faktörle fazla tahmin ürettiği görülmektedir.

V. SONUÇ

Bu makale, COTS platformlarda çalışan zaman kritik endüstriyel uygulamalar için de uygun ve gelişmiş bir HYPTA metodu önermektedir.

Önerilen metodumuzda ana ilke, her bir rastgele değişkeni mümkün olduğunca UDT dağılımı ile rastgele değişkenler arasındaki bağımlılığı da kopulalar yardımıyla modellemektir. Temel olarak, bir program kapsamlara bölünmekte ve her bir kapsam önerilen metodolojimiz kullanılarak ayrı ayrı analiz edilmekte ve ardından birleştirilmektedir.

Deneylerimiz, önerilen UDT ile kopulalar birleşiminden oluşan metodun mevcut yaklaşımlara ve RapiTime adlı ticari araca kıyasla çok daha iyi sonuçlar sağlayabileceğini göstermiştir.

Bu çalışma, MATLAB ve R yardımıyla gösterimi yapılan bir metodoloji sunmaktadır. Gelecekte önerinin tam işlevsel otomatik bir araca dönüştürülmesi yararlı olacaktır.

TEŞEKKÜR

Çalışmanın ekipman desteği için Türk Havacılık ve Uzay Sanayi A.Ş. (TUSAŞ)'ye teşekkür ederiz.

VI. KAYNAKÇA

[1] G. Bernat, A. Colin, and S. Petters, "WCET analysis of probabilistic hard real-time systems," in *Proc. of the 23rd IEEE Real-Time Systems Symposium*. IEEE Comput. Soc, 2002, pp. 279–288.

[2] F. J. Cazorla, L. Kosmidis, E. Mezzetti, C. Hernandez, J. Abella, and T. Vardanega, "Probabilistic worst-case timing analysis," *ACM Computing Surveys*, vol. 52, no. 1, pp. 1–35, 2019.

[3] L. Cucu-Grosjean, L. Santinelli, M. Houston, C. Lo, T. Vardanega, L. Kosmidis, J. Abella, E. Mezzetti, E. Quinones, and F. J. Cazorla, "Measurement-based probabilistic timing analysis for multi-path programs," in *Proc. of Euromicro Conference on Real-Time Systems*. IEEE, 2012, pp. 91–101.

[4] B. Y. A. Ferreira and L. De Haan, "On the block maxima method in extreme value theory: PWM estimators," *Annals of Statistics*, vol. 43, no. 1, pp. 276–298, 2015.

[5] L. Santinelli, F. Guet, and J. Morio, "Revising measurement-based probabilistic timing analysis," in *Proc. of Real-Time and Embedded Technology and Applications Symposium*. IEEE, 2017, pp. 199–208.

[6] G. Lima, D. Dias, and E. Barros, "Extreme value theory for estimating task execution time bounds: a careful look," in *Proc. of the 28th Euromicro Conference on Real-Time Systems*. IEEE, 2016, pp. 200–211.

[7] A. Sklar, "Fonctions de répartition àn dimensions et leurs marges," *Publications de L'Institut de Statistique de L'Université de Paris*, vol. 8, pp. 229–231, 1959.

[8] G. Bernat, A. Colin, and S. Petters, "pWCET: A tool for probabilistic worst-case execution time analysis of real-time systems," *Unpublished Report - University of York*, pp. 1–18, 2003.

[9] R. I. Davis and L. Cucu-Grosjean, "A survey of probabilistic timing analysis techniques for real-time systems," *Leibniz Transactions on Embedded Systems*, vol. 6, no. 1, pp. 3:1–3:60, 2019.

[10] J. Abella, C. Hernandez, E. Quinones, F. J. Cazorla, P. R. Conmy, M. Azkarate-askasua, J. Perez, E. Mezzetti, and T. Vardanega, "WCET analysis methods: pitfalls and challenges on their trustworthiness," in *Proc. of the 10th International Symposium on Industrial Embedded Systems*. IEEE, 2015, pp. 1–10.

[11] R. I. Davis, I. Bate, I. Broster, A. Burns, S. Hutchesson, and R.-r. Plc, "Transferring real-time systems research into industrial practice: four impact case studies," vol. 106, no. 7, pp. 7:1–7:24, 2018.

[12] C. Park and A. Shaw, "Experiments with a program timing tool based on source-level timing schema," in *Proc. of the 11th Real-Time Systems Symposium*, 1990, pp. 72–81.

[13] G. Bernat, A. Burns, and M. Newby, "Probabilistic timing analysis: an approach using copulas," *Journal of Embedded Computing*, vol. 1, no. 2, p. 179, 2005.

[14] K. Avdulaj, "Value-at-Risk based on extreme value theory method and copulas. Empirical evidence from Central Europe," Master's thesis, Univerzita Karlova, 2010.

[15] F. Miller, A. Vandome, and M. John, *Inverse Transform Sampling*. VDM Publishing, 2010.

[16] A. Betts, "Hybrid measurement-based wcet analysis using instrumentation point graphs," Ph.D. dissertation, University of York, 2010.

[17] L. Bekdemir, "Hybrid probabilistic timing analysis with extreme value theory and copulas," Master's thesis, Middle East Technical University, 2019.

[18] L. Bekdemir and C. F. Bazlamaçcı, "Hybrid probabilistic timing analysis with extreme value theory and copulas," *in review in Microprocessors and Microsystems (Elsevier)*, 2021.

[19] J. Gustafsson, A. Betts, A. Ermedahl, and B. Lisper, "The Mälardalen WCET benchmarks: past, present and future," in *Proc. of Int. Workshop on Worst-Case Execution Time Analysis (WCET 2010)*, 2010, pp. 136–146.