T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

PARALEL GERÇEK ZAMANLI KIYASLAMA UYGULAMA TAKIMI

YÜKSEK LİSANS TEZİ

Sevil SERTTAŞ

Enstitü Anabilim Dalı	**:**	**BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Dr. Öğr. Üyesi Veysel Harun ŞAHİN**

Nisan 2019

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

PARALEL GERÇEK ZAMANLI KIYASLAMA UYGULAMA TAKIMI

YÜKSEK LİSANS TEZİ

Sevil SERTTAŞ

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ

Bu tez 05.04.2019 tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

Dr. Öğr. Üyesi	Prof. Dr.	Dr. Öğr. Üyesi
Veysel Harun ŞAHİN	Celal ÇEKEN	Fırat AYDEMİR
Jüri Başkanı	Üye	Üye

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Sevil SERTTAŞ

05.04.2019

TEŞEKKÜR

Yüksek lisans eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen değerli danışman hocam Dr. Öğr. Üyesi Veysel Harun ŞAHİN'e teşekkürlerimi sunarım.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

A	: Dizi
BLO	: Bit seviyesinde işlem
D	: Karar
DM	: Dinamik hafıza
EEMBC	: Gömülü mikroişlemci kıyaslama konsorsiyumu
ER	: Harici kütüphane
FPO	: Kayan noktalı işlem
IF	: Giriş dosyası
IVAL	: Giriş değeri
IVEC	: Giriş vektörü
L	: Döngü
MP	: Çok yollu
MT	: Çok iş parçacıklı
NL	: İç içe döngü
PBench	: Paralel kıyasma uygulama takımı
POSIX	: Unix için taşınabilir işletim sistemi arayüzü
R	: Özyineleme
RTEMS	: Çok işlemcili sistemler için gerçek zamanlı yürütme
SP	: Tek yollu
ST	: Tek iş parçacıklı
ULP	: Ultra düşük güç
WCET	: En kötü durum yürütme süresi

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Gerçek zamanlı sistemler, kıyaslama uygulama takımı, paralel programlama, yazılım mühendisliği

Günümüzde gerçek zamanlı sistemler yaygın olarak kullanılmaktadır. Otomotiv endüstrisi, havacılık endüstrisi, askeri sistemler bu alanlara örnek teşkil etmektedir. Gerçek zamanlı sistemler ile çalışan bilim insanları, kendi sistemlerini alternatif sistemlerle karşılaştırmaya ihtiyaç duymaktadır. Bu amaç doğrultusunda kıyaslama uygulama takımları kullanılmaktadır.

Gerçek zamanlı sistemler için en kötü durum yürütme süresi kavramı çok önemlidir. Bundan dolayı en kötü durum yürütme süresi analizi araçları ve yöntemleri geliştirilmektedir. Geliştirilen bu araç ve yöntemlerin de karşılaştırılmaları gerekmektedir. Bu sebeple, bu araç ve yöntemleri hedefleyen kıyaslama programlarına da ihtiyaç duyulmaktadır.

Çok işlemcili/çok çekirdekli sistemlerin yaygınlaşması ile birlikte paralel programlama kavramı önem kazanmıştır. Bu sebeple bu tür sistemler, paralel programlama bakış açısı ile de kıyaslanmalıdır. Bu tez çalışmasında, paralel, gerçek zamanlı bir kıyaslama uygulama takımı olan PBench tasarlandı ve gerçekleştirildi. PBench, farklı sorunlara çözüm sağlayan çok iş parçacıklı ve tek iş parçacıklı uygulamalar içermektedir.

A PARALLEL, REAL-TIME BENCHMARK SUITE

SUMMARY

Keywords: Real-Time systems, benchmark suite, parallel programming, software engineering

Today, real-time systems are widely used. Some of these areas are automotive industry, aviation industry, military systems etc. People who work on real-time systems should compare their systems with alternative systems. For this purpose, they use benchmark applications.

The worst-case execution time concept for real-time systems is critical. Therefore, worst-case execution time analysis tools and methods are developed. These tools and methods are also need to be compared. Therefore, benchmark applications are also needed that targets these tools and methods.

The concept of parallel programming has gained importance as multiprocessor/multi-core systems become widespread. Therefore, these systems should also be compared from the point of view of paralel programming. In this thesis, a parallel, real-time benchmark suite, PBench is designed and developed. The PBench includes multi-threaded and single-threaded applications that provide solution to different problems.

BÖLÜM 1. GİRİŞ

Kıyaslama uygulama takımları, başta gerçek zamanlı sistemler olmak üzere; bilgisayar sistemlerinde, veritabanı sistemlerinde ve daha birçok sistemin değerlendirilmesi ve karşılaştırılması amacıyla kullanılmaktadır. Araştırmacılar, geliştirilen kıyaslama uygulama takımı programları ile kendi sistemlerini alternatif sistemlere karşı değerlendirme imkânı bulurlar.

Günümüzde kıyaslama uygulama takımlarının farklı bilgisayar bilimleri disiplinlerinde kullanımı yaygınlaşmış ve özellikle yazılım mühendisliği alanında etkisi artmıştır. Kıyaslama uygulama takımlarının yazılım mühendisliği alanında bu denli bir etkiye sahip olmasındaki en önemli sebep gerçekleştirilen yazılımların daha tutarlı olmasına katkı sağlamasıdır [1].

Bilgisayar donanım teknolojisinin hızla gelişmesiyle çok işlemcili/çok çekirdekli sistemler yaygınlaşmıştır. Bu sistemlerin kullanımı ile daha fazla verim elde edilebilecek program sayısı gün geçtikçe artmaktadır. Bu programlar farklı çekirdekler üzerinde eşzamanlı çalışan birçok iş parçacığı (iplik, thread) içermektedir. Bu tür programlar paralel programlar olarak adlandırılır ve bu programlama tekniği ise paralel programlama olarak ifade edilir.

Paralel programlama için iş parçacıklarının oluşturulması ve yönetilmesini sağlayan birçok iş parçacığı kütüphanesi mevcuttur. Bunlardan en bilineni pthread.h kütüphanesidir. Paralel programlama için derleyicilerin, program geliştiricilerin iş parçacıklarını yönetme yükünü azalttığı farklı paralel programlama kütüphaneleri de mevcuttur. Bunlardan en yaygın olanlarından birisi ise OpenMP kütüphanesidir [2]–[4].

Günümüzde bilim insanları ve mühendisler, çok çekirdekli sistemler üzerinde çalışan gerçek zamanlı programlar geliştirmeye çalışmaktadırlar. Bu amaçla gerçek zamanlı sistemler üzerine birçok araştırma gerçekleştirilmiştirler [5]. Gerçek zamanlı sistemlerde araştırmalarını, paralel bakış açısıyla gerçekleştiren bilim insanlarının da çalışmalarını çok çekirdekli sistemlerde karşılaştırması gerektiği açıktır. Bu sebeple çok çekirdekli sistemler üzerinde ölçeklenebilen paralel kıyaslama uygulama takımlarına ihtiyaç duyulmaktadır.

Gerçek zamanlı sistemlerde en önemli kriter zaman kavramıdır. Bu sistemlerin devreye alınmasından önce geliştiricilerin, programın zaman limitini aşıp aşmadığından emin olması gerekir. Programın bir bölümü ya da tamamının yürütme süresi hakkında bilgi almak için WCET analizleri gerçekleştirilir [6], [7].

Bu çalışmada, en kötü durum yürütme süresi analizine odaklanıldı ve en kötü durum yürütme süresi araçlarının karşılaştırılmasına yardımcı olacak kıyaslama uygulamaları gerçekleştirildi. Bu tez çalışması, Sakarya Üniversitesi Bilgisayar ve Bilişim Bilimleri Fakültesi, Yazılım Mühendisliği Bölümü, Gerçek Zamanlı Sistemler Araştırma Laboratuvarı bünyesinde gerçekleştirilmiş olup, web sitesi üzerinden erişilebilirdir [8].

Bu tezin bölümleri şu şekilde organize edilmiştir: 2. Bölüm WCET hakkında bilgi vermektedir. 3. Bölüm literatürdeki diğer kıyaslama uygulamalarından bahsetmektedir. 4. Bölüm gerçek zamanlı işletim sistemi kavramını açıklamaktadır. 5. Bölüm gerçekleştirilen PBench kıyaslama uygulama takımından bahsetmektedir. 6. Bölüm tartışma ve 7. Bölüm sonuç olarak yazılıp, tez tamamlanmıştır.

BÖLÜM 2. WCET

Zaman kavramı gerçek zamanlı sistemlerde en önemli kriterdir. Bilim insanları bu sistemlerin testlerini gerçekleştirirken en kötü durum yürütme süresi (worst-case execution time) (WCET) analizlerini kullanmaktadırlar. WCET analizi akademik ve endüstriyel alanda çalışılan bir konudur. Birçok araştırmacı gerçek zamanlı programların zaman davranışları hakkında çıkarım yapabilmek için yeni WCET süresi analiz yöntemleri ve araçları geliştirmektedirler.

WCET analizi yöntemlerine ölçüm tabanlı, statik ve hibrit yöntemler örnek verilebilir. Bu yöntemlerden ölçüm tabanlı olanı, program girdilerini ve program koşullarını gözden geçirir, WCET tahminlerinin bu bilgilere dayanarak nasıl elde edildiğini gözden geçirir. Statik yöntem, doküman tabanlı bir yaklaşımdır. Statik zamanlama analiz yönteminde bir cihazın, tasarım ve geliştirilmesi sürecindeki tüm adımlarında zamanlama ile ilgili dokümantasyonunun varlığı sistemi güvenilir kılar. Diğer bir yöntem ise hibrit tekniği yöntemidir. Bu yöntem ölçüm tabanlı ve dokümantasyon tabanlı yöntemin karması olarak nitelendirilmiştir [9].

Statik yöntemin bir çeşidi ise statik olasılıksal zamanlama analizi yöntemidir. Bu yöntem programın yapısını analiz ederek ve üzerinde çalıştığı donanımın davranışlarını modelleyerek WCET dağılımı oluşturur. Programın her yönergesi kendisiyle ilişkili bir olasılık dağılımına sahip olabilir. Bu dağılımların birleştirilmesiyle statik olasılıksal zamanlama analizi yöntemi ile program için WCET analizi gerçekleştirilmiş olur [10], [11].

Ölçüm tabanlı olasılıksal zamanlama analizi yöntemi ise ölçüm tabanlı yöntemin bir çeşididir ve uygulamanın yürütülmesinden elde edilen sonuçlara göre bir olasılık

dağılım oluşturulur ve bu dağılım üzerinden uygulamanın belli zaman sınırını aşma olasılığı hesaplanır [12], [13].

Son yıllarda birçok WCET analizi araçları geliştirilmiştir. Örneğin OTAWA (Open Tool for Adaptive WCET Analysis), uyarlanabilir WCET analizi için açık araç olarak ifade edilebilir [14]. Bu araç, kontrol akış grafikleri, döngü algılama gibi son teknoloji WCET analizleri sunmaktadır. PowerPC, ARM, Sparc gibi birçok bilgisayar mimarisini de desteklemektedir. Chronos, yönerge zamanlaması için başarım arttırıcı mikro mimari özelliklerin zamanlama etkilerini yakalamak için mikro mimari modelleme gerçekleştirir [15]. SWEET (SWEdish Execution Time), akış analizi ve WCET hesaplaması için bir araştırma aracıdır [16]. Bu araç 2001 yılında İsveç'te bir araştırmacı grubu tarafından geliştirilmiştir. Ana amacı akış analizleridir. Programın kullandığı yol, döngüler hakkında bilgi verir. RapiTime gömülü sistemler, otomotiv elektroniği gibi alanlar için yazılan yazılımlara doğrulama hizmeti sunan ticari bir araçtır [17]. aiT, uçuş kontrol yazılımları da dahil olmak üzere kritik aviyonik yazılımların zamanlama davranışlarının doğrulanması için kullanılan bir WCET aracıdır [18]. Bound-T, WCET üst sınırını hesaplamak için makine kodunun statik analizini kullanan bir yazılım aracıdır [19].

Diğer alanlarda olduğu gibi WCET analizi yöntemleri ve araçları konusunda da çalışmaların değerlendirilip karşılaştırılması gerekmektedir. Bu yüzden WCET analizleri için geliştirilen kıyaslama uygulama takımı programları kullanılır. WCET analizine özgü birçok kıyaslama uygulaması ve kıyaslama uygulama takımları mevcuttur. Bu kıyaslama uygulamalarıa ve uygulama takımları hakkında bölüm 3'te detaylı bilgi verilecektir.

BÖLÜM 3. KIYASLAMA UYGULAMALARI

Bilgisayar bilimlerinde kıyaslama, gerçek zamanlı sistem olsun ya da olmasın tüm sistemlerde önemli bir prosedürdür. Yeni geliştirilen araçlardan, yazılımdan ve donanımdan yeni önerilen yöntemlere kadar her türlü şey değerlendirilmeli ve karşılaştırılmalıdır. Bu amaçla kıyaslama programları kullanılmaktadır [1].

Literatürde farklı amaçlarda kullanılabilecek çok sayıda kıyaslama uygulama takımı mevcuttur. Örneğin Standard Performance Evaluation Corporation (Standart Başarım Değerlendirme Kurumu) (SPEC) [20] güç karakteristikleri (ör. SPECpower_ssj2008) ve merkezi işlem birimi başarımı (ör. SPEC CPU2017) gibi farklı bilgisayar sistemlerini değerlendirmek için SPEC kıyaslama uygulama takımı olarak adlandırılan farklı kıyaslama uygulama takımı geliştirmektedir.

Rodinia kıyaslama uygulama takımı, heterojen hesaplama için geliştirilmiş bir kıyaslama uygulama takımıdır [21]. Bu kıyaslama takımı çok çekirdekli merkezi işlem birimi (CPU) ve grafik işlem birimi (GPU) platformlarını hedef alan kıyaslama uygulamaları içerir. Rodinia'nın diğer kıyaslama uygulamalarından farkı, geleneksel merkezi işlem birimi mimarileri için değil, heterojen mimariler için geliştirilmiş olmasıdır. Bu kıyaslama uygulama takımındaki uygulamalar, çoklu işlemeyi destekleyen bir uygulama geliştirme arayüzü olan OpenMP ile geliştirilmiştir.

Başka bir örnek olarak BigDataBench-S [22] büyük bir veri kıyaslama uygulama takımıdır. Açık kaynak kodludur ve bilimsel alanlarda kullanıma yöneliktir.

PARSEC (Princeton Application Repository for Shared-Memory Computers) kıyaslama uygulama takımı ise örüntü tanıma, veri madenciliği, büyük ölçekli ve çok iş parçacıklı sistem uygulamaları içermektedir [23]. PARSEC, yeni nesil işlemcileri

tasarlamak için kullanılıbilecek bir kıyaslama uygulama takımı olması amacıyla geliştirilmiştir.

Gerçek zamanlı sistemler için en iyi bilinen kıyaslama uygulama takımlarından birisi Mälardalen WCET kıyaslama uygulama takımıdır [16], [24]. Bu kıyaslama uygulama takımı özellikle WCET analizine yönelik bir çalışmadır. C programlama dilinde yazılmış birçok program içerir. Programlar, bilim insanları ve geliştiricilerin yaptıkları çalışmaların farklı yönlerini test etmelerini sağlarlar. Bunun için programlarda dizi işlemleri, kayan nokta hesaplamaları gibi farklı özellikler kullanılmıştır. Mälardalen WCET kıyaslama uygulama takımı, akış analizine odaklanan çoğunlukla küçük programlardan oluşmaktadır.

DaCapo kıyaslama uygulama takımı, yüksek seviyeli bir dil olan Java ve yönetilen diğer diller için sistem tasarımı ve uygulamasında yeniliği teşvik etmek, kıyaslama uygulama takımlarının değerlendirilmesi için yeni metadolojilerin geliştirilmesine yönelik bir çalışmadır [25]. Bu çalışmada Java ile kıyaslama uygulamaları gerçekleştirmenin C, C++ ve Fortran gibi programlama dillerinden daha kapsamlı bir değerlendirme gerektiği vurgusu yapılmaktadır.

Java için Gerçek Zamanlı Spesifikasyon çalışması (Real-time Specification for Java), Java topluluğu süreci içinde ilk Java spesifikasyonu isteğidir (Java Specification Request / JSR-1). Amacı, java iş parçacıklarının oluşturulmasını, doğrulanmasını, analiz edilmesini ve yönetimini sağlayacak bir uygulama arabirimi sağlamaktır [26]. Gerçek zamanlı sistemlerin doğrulanması için açık zorluklar bulunmaktadır. Bu zorluklar; tip analizleri, hafıza analizleri, engelleme analizleri, döngü sınır analizleri ve istisnalardır. Bu konuda yardımcı olmak için geliştirilmiş olan kıyaslama bir kıyaslama uygulaması açık kaynak kodlu CDx yazılımıdır [27]. CDx, simüle edilmiş radar çerçevelerine dayanan uçak çarpışma algılamasını uygulayan, tek bir periyodik görevi olan bir uygulamadır.

PapaBench [28], WCET ve çizelgeleme analizine odaklanan paralel bir gerçek zamanlı kıyaslama uygulamasıdır. Paparazzi adı verilen, gerçek hayattaki bir İHA

kontrol yazılımı üzerine inşa edilmiştir ve bu özelliği ile gerçek zamanlı gömülü endüstriyel uygulamayı temsil eder.

TACLeBench [29], en kötü durum yürütme zamanı araştırmasına yardımcı olmayı amaçlayan bir kıyaslama uygulama kümesidir. Paralel uygulamalar dâhil olmak üzere farklı tiplerde 53 kıyaslama uygulama programı içerir.

Embedded Microprocessor Benchmark Consortium (Gömülü Mikroişlemci Kıyaslama Konsorsiyumu) (EEMBC) [30] özellikle gömülü sistemler için hedeflenen kıyaslama uygulamaları geliştirmektedir. EEMBC birkaç kategoride birçok kıyaslama uygulaması sunmaktadır. Ana kategoriler aşağıda sıralanmıştır;

- Ultra Düşük Güç (ULP) ve Nesnelerin İnterneti
- Heterojen Hesaplama
- Tek Çekirdekli İşlemci Başarımı
- Simetrik Çok Çekirdekli İşlemci Başarımı
- Telefon ve Tablet

MiBench kıyaslama uygulama takımı, otomotiv ve endüstriyel kontrol, tüketici cihazları, ofis otomasyonu, ağ, güvenlik ve telekomünikasyon gibi farklı gömülü sistem alanlarında çeşitli programlar içerir [31], [32]. Programlar platformdan bağımsızdır ve C programlama dili ile yazılmıştır.

BÖLÜM 4. GERÇEK ZAMANLI İŞLETİM SİSTEMİ

İşletim sistemleri, bilgisayar kullanıcısı ile bilgisayar donanımı arasında bir aracı olarak görev yapar. İşletim sistemlerinin amacı, kullanıcının programları uygun ve verimli bir şekilde yürütebilmesine imkân sağlamaktır. İşletim sistemleri, bilgisayar donanımını yöneten yazılımlardır. Farklı görevleri yerine getirmek üzere inşa edilmiş çeşitli yapıda işletim sistemleri vardır. Anasistem (mainframe) işletim sistemleri öncelikle donanım kullanımını optimize etmek için tasarlanmışlardır. Kişisel bilgisayarlarımızdaki işletim sistemleri ise karmaşık oyunları, iş uygulamalarını vb. herşeyi destekler. El bilgisayarları olarak adlandırılan bir diğer bilgisayarlar ise kullanıcının programları yürütmek için bilgisayarla kolayca iletişim kurabileceği bir ortam sağlamak üzere tasarlanmıştır. Bu nedenle bazı işletim sistemleri verimlilik, bazıları kullanılabilirlik diğerleri ise bu ikisinin kombinasyonu olarak tasarlanırlar [33].

Gerçek zamanlı işletim sistemleri ise sistem davranışının doğruluğunun sadece hesapların mantıksal sonuçlarına göre değil aynı zamanda bu sonuçların üretildiği fiziksel anın da hesaba katıldığı sistemlerdir. Gerçek zamanlı sistemler, ortamın belirlediği zaman aralıklarında sonuç vermelidirler. Bu sonucun üretildiği an, son teslim zamanı (deadline) olarak ifade edilir.

Gerçek zamanlı sistemler katı gerçek zamanlı sistemler (hard real-time systems) ve yumuşak gerçek zamanlı sistemler (soft real-time systems) olarak ikiye ayrılırlar. Eğer sistemin çıktı verme anı, son teslim zamanını aşması bir felaketle sonuçlanacak ise bu sistemler katı gerçek zamanlı sistemler olarak nitelendirilirler. Bu sistemlere örnek olarak; demiryollarındaki trafik ışıklandırması verilebilir. Tren gelmeden ışık kırmızı olarak değişmez ise bu durum bir felakete dönüşebilir. Öte yandan yumuşak

gerçek zamanlı sistemlerde son teslim zamanının aşılması bir felaketle sonuçlanmaz [34].

Gerçek zamanlı sistemlerin kullanım alanı oldukça yaygındır. Gerçek zamanlı hesaplama gerektiren uygulamalara örnek olarak nükleer santraller, demiryolu anahtarlama sistemleri, otomotiv ve aviyonik sistemler, hava trafik kontrolü, telekominikasyon, robotik ve askeri sistemler olarak sıralanabilir. Son yıllarda tıbbi cihazlar, multimedya sistemleri, uçuş simülasyon sistemleri, sanal gerçeklik ve etkileşimli oyunlar gibi yeni uygulama alanlarında da gerçek zamanlı sistemler yer almaktadır [35].

Açık kaynak kodlu bazı gerçek zamanlı işletim sistemleri aşağıda kısaca açıklanmıştır.

- Real-Time Executive for Multiprocessor Systems (RTEMS): POSIX gibi açık standart uygulama programlama arabirimlerini (API) destekleyen açık kaynak kodlu bir gerçek zamanlı işletim sistemidir [36].
- Zephyr: Birden fazla donanım mimarisini destekleyen, kaynak kısıtlı cihazlar için optimize edilmiş ve güvenlik göz önünde bulundurularak oluşturulmuş ölçeklenebilir bir gerçek zamanlı işletim sistemidir [37].
- eCos: Gömülü uygulamalar için açık kaynak kodlu gerçek zamanlı işletim sistemidir. Çok çeşitli pazarlarda ve cihazlarda konuşlandırılmıştır. Örnek olarak; Sirius uydu radyo alıcıları, Sony Playstation 3 Wi-fi modülleri, NETGEAR yönlendiricileri(routers) ve Alfa Manyetik Spektrometresi gibi büyük havacılık uygulamaları verilebilir [38].
- RIOT: Genellikle nesnelerin internetinde bulunan bir dizi cihazı destekleyen gerçek zamanlı, çok iş parçacıklı bir işletim sistemidir [39].
- ATK2: İsmi, otomotiv çekirdeği olarak Türkçe'ye çevirebileceğimiz Automotive Kernel ifadesinin kısaltmasından gelmektedir. Otomotiv sistemi kontrolü için gerçek zamanlı bir işletim sistemidir [40].
- Contiki OS: Nesnelerin interneti için açık kaynak kodlu bir gerçek zamanlı işletim sistemidir. Düşük maliyetli ve düşük güçlü mikrodenetleyicileri

internete bağlar. Karmaşık kablosuz sistemler oluşturmak için güçlü bir araçtır [41].

- MaRTE: Çok iş parçacıklı uygulamalar geliştirmek için kullanımı kolay ve kontrollü olan açık kaynak kodlu gerçek zamanlı işletim sistemidir [42].

Ek olarak, AliOS Things, Apache Mynewt, Atomthreads, BeRTOS, BitThunder, ChibiOS/RT, DuinOS, Erika EnterPrise, F9 Microlkernel, Femto OS, FreeRTOS, Freescale MQX, Frosted, FunkOS, Fusion Embedded RTOS, IntrOS, LibreRTOS, LiteOS, MARK3, MOE, Mongoose OS, Nut/OS, NuttX, Phoenix, Prex, Protothreads, RT-Thread, RTAI, StateOS, STratifyOS, TI-RTOS Kernel, TNKernel, TNeo, TinyOS, TizenRT, Tock, Trampoline, Xenomai, cocoOS, distortos, eChronos, embox, mbed OS, scmRTOS, sel4, uKOS, uSmartX diğer açık kaynak kodlu gerçek zamanlı işletim sistemlerine örnek verilebilir [40].

BÖLÜM 5. PBENCH

Bu tez çalışması için paralel kıyaslama uygulama takımı geliştirilmiştir. Geliştirilen kıyaslama uygulama takımına PBench ismi verilmiştir. “P” harfi programların paralel yapıda olduğunu temsil etmektedir. “Bench” kelimesi ise İngilizce kıyaslama anlamı taşıyan “benchmark” kelimesinin kısaltmasıdır.

Uygulamalar açık kaynak kodlu, MIT lisanslı ve ücretsiz olarak kullanıma açıktır. Uygulamalara Sakarya Üniversitesi Bilgisayar ve Bilişim Bilimleri Fakültesi, Yazılım Mühendisliği bölümü bünyesinde bulunan Gerçek Zamanlı Sistemler Araştırma Laboratuvarının internet sitesi üzerinden erişilebilirdir [8].

PBench çok iş parçacıklı programların yanı sıra aynı uygulamaların tek iş parçacıklı sürümlerini de içermektedir. Bunu yapmamızdaki amaç, ilgili araştırmacılara, çalışmalarının hem paralel programlama tekniği ile hem de sıralı programlama tekniği ile yazılmış uygulamaları yan yana karşılaştırma imkânı sağlamayı amaçlamış olmamızdır.

PBench yedi farklı uygulama içerir. Her uygulamanın çok iş parçacıklı ve tek iş parçacıklı sürümleri mevcuttur. Ek olarak her birisinin Linux ve RTEMS işletim sistemleri için de ayrı sürümleri bulunmaktadır. Böylece toplamda yirmi sekiz adet kıyaslama uygulaması bulunmaktadır. Programlar C programlama dilinde yazılmıştır. Yazılan programların Pardus 17.3 sürümünde, xubuntu 18.04 Linux dağıtımında ve RTEMS işletim sisteminde koşturulması sağlanmıştır.

5.1. Yöntem

5.1.1. Paralel Programlama

Bir problemin çözümünde kaynakların aynı anda kullanılmasına parallel hesaplama, bu hesaplama işlemini gerçekleştirmek için kullanılan programlama yöntemine ise parallel programlama denir. Paralel programlamada problem, aynı anda işlem yapmak üzere farklı parçalara bölünür. Bölünen her bir parça da farklı komut serilerine bölünür. Bu komut serileri farklı işlemci ya da çekirdekte aynı anda işletilirler.

Farklı parallel programlama modelleri bulunmaktadır. Bunlar; saf, heterojen, hibrit parallel programlama modelleridir. POSIX Threads, OpenMP ve MPI literatürde saf parallel programlama modeli olarak nitelendirlirler. CUDA, OPENCL, DirectCompute, Array Building Blocks ise heterojen parallel programlama modeli olarak belirtilmiştir. Hibrit parallel programlama ise Pthreads ve MPI, MPI ve OpenMP ya da CUDA ve Pthreads birleşimiyle oluşturulmuş parallel programlama modelidir [4]. Bu tez çalışmasında kullanılan parallel programlama yöntemi PThread'tir ve 5.1.2'de detaylandırılmıştır.

5.1.2. POSIX Thread (PThread) Yapısı

POSIX kelimesi ingilizce "Portable Operating System Interface for UNIX" ifadesinin kısaltmasıdır. Bilgisayar dünyasında iş parçacığı kavramının C dili ile kodlanabilmesi için genellikle UNIX ve benzeri işletim sistemlerinde (UNIX, Linux, Solaris, Mac OS X vb.) geliştirilen paralel programlama kütüphanesidir. POSIX kısaltmasının baş harfi ile thread kelimesinin birleşimi ile PThread adı verilen kütüphane ismi oluşmaktadır. C dili ile paralel program yazarken harici kütüphane olarak "pthread.h" kütüphanesi kullanılarak iş parçacıkları oluşturulup yönetilebilir. Kısaca C'nin iş parçacığı kullanımına imkan tanıyan POSIX kütüphanesidir [4].

5.2. PBench

PBench tasarımını gerçekleştirirken ilk önce karşılaştırmalar sırasında test edilmesi gereken özellikler belirlendi. Bu özellikler Tablo 5.1.'de gösterilmiştir. Özelliklerin kısaltmaları ingilizce yazılışlarının baş harfleri ile belirtilmiştir.

Tablo 5.1. Özellik Matrisi

Özellik	İngilizcesi	Kısaltması	Açıklaması
Tek İş Parçaçıklı	Singlethread	ST	Program sıralıdır.
Çok İş Parçacıklı	Multithread	MT	Program paraleldir.
Harici Rutin	External Routine	ER	Program harici kütüphane kullanır.
Tek Yollu	Single Path	SP	Programın her derlenmesinde, her zaman aynı yürütme yolunu izlemesidir.
Çok Yollu	Multi Path	MP	Programın her bir derlenmesinde farklı bir yürütme yolu izlemesidir.
Dinamik Bellek	Dynamic Memory	DM	Program, program verileri için belleği dinamik olarak ayırması ve serbest bırakmasıdır.
Döngü	Loop	L	Programın döngü içermesidir.
İç İçe Döngü	Nested Loop	NL	Programın iç içe döngü içermesidir.
Özyineleme	Recursive	R	Bir fonksiyonun program içerisinde kendini çağırmasıdır.
Karar	Decision	D	Program karar yapıları içerir.
Dizi	Array	A	Program dizi işlemleri gerçekleştirir.
Bit Düzeyinde İşlem	Bit Level Operation	BLO	Program bit düzeyinde işlemler gerçekleştirir.
Kayan Noktalı İşlem	Floating Point Operation	FPO	Program kayan noktalı işlemler gerçekleştirir.

Tablo 5.1. Devamı

Özellik	İngilizcesi	Kısaltması	Açıklaması
Tamsayı İşlem	Input Operation	IO	Program tamsayı işlemleri gerçekleştirir.
Giriş Vektörü	Input Vector	IVEC	Program harici giriş vektör verilerini parametre olarak alır.
Giriş Değeri	Input Value	IVAL	Program parametre olarak tek bir harici giriş değeri alır.
Giriş Dosyası	Input File	IF	Program harici giriş dosyasını parametre olarak alır.

Daha sonra farklı çalışma alanlarından örnek paralel problemler seçildi. Problem seçimini, daha önce belirlediğimiz özellikleri düşünerek gerçekleştirdik. Sonrasında seçilen problemleri çözen programları kodladık. Programlarımızın her birinde, paralel kıyaslama uygulamalarının özelliklerinin çoğunu kapsayacak şekilde bir dizi özellik kullandık. Programlar ve programlara ait özellikler xUbuntu 18.04 Linux dağıtımı için Tablo 5.2.'de, RTEMS için ise Tablo 5.3.'de gösterilmiştir. Tabloda programların desteklediği özellikler "+" işareti ile, programların desteklemediği özellikler ise "-" işareti ile gösterilmiştir.

Tablo 5.2. Kıyaslama Uygulama Programlarına Ait Özellikler (Linux)

Program / **Özellik**	p_matrix_multiplication	matrix_multiplication	p_selection_sort	selection_sort	p_array_search	array_search	p_factorial	factorial	p_binomial_distribution	binomial_distribution	p_prime_numbers	prime_numbers	p_linear_regression	linear_regression
ST	-	+	-	+	-	+	-	+	-	+	-	+	-	+
MT	+	-	+	-	+	-	+	-	+	-	+	-	+	-
ER	+	+	+	+	+	+	+	+	+	+	+	+	+	+
SP	-	-	-	-	+	+	-	-	+	+	+	+	+	+
MP	+	+	+	+	-	-	+	+	-	-	-	-	-	-
DM	-	-	+	-	-	-	-	-	-	-	-	-	-	-

Tablo 5.2. Devamı

Program / Özellik	p_matrix_multiplication	matrix_multiplication	p_selection_sort	selection_sort	p_array_search	array_search	p_factorial	factorial	p_binomial_distribution	binomial_distribution	p_prime_numbers	prime_numbers	p_linear_regression	linear_regression
L	+	+	+	+	+	+	+	-	+	+	+	+	+	+
NL	+	+	-	-	-	-	-	-	-	-	+	-	-	-
D	-	-	+	+	+	+	+	+	-	-	-	-	-	-
A	+	+	+	+	+	+	+	-	+	-	+	+	-	-
BLO	-	-	-	-	-	-	-	-	-	-	+	-	+	+
R	-	-	-	+	-	-	-	+	-	-	-	-	-	-
FPO	-	-	-	-	-	-	-	-	-	-	+	-	+	+
IO	+	+	+	+	+	+	+	+	+	+	+	+	-	-
IVEC	-	-	-	-	-	-	-	-	-	-	-	-	-	-
IVAL	-	-	-	-	-	-	+	+	-	-	-	-	-	-
IF	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tablo 5.3. Kıyaslama Uygulama Programlarına Ait Özellikler (RTEMS)

Program / Özellik	p_matrix_multiplication	matrix_multiplication	p_selection_sort	selection_sort	p_array_search	array_search	p_factorial	factorial	p_binomial_distribution	binomial_distribution	p_prime_numbers	prime_numbers	p_linear_regression	linear_regression
ST	-	+	-	+	-	+	-	+	-	+	-	+	-	+
MT	+	-	+	-	+	-	+	-	+	-	+	-	+	-
ER	+	+	+	+	+	+	+	+	+	+	+	+	+	+
SP	-	-	-	-	+	+	-	-	+	+	+	+	+	+
MP	+	+	+	+	-	-	+	+	-	-	-	-	-	-
DM	-	-	+	-	-	-	-	-	-	-	-	-	-	-

Tablo 5.3. Devamı

Program / Özellik	p_matrix_multiplication	matrix_multiplication	p_selection_sort	selection_sort	p_array_search	array_search	p_factorial	factorial	p_binomial_distribution	binomial_distribution	p_prime_numbers	prime_numbers	p_linear_regression	linear_regression
L	+	+	+	+	+	+	+	-	+	+	+	+	+	+
NL	+	+	-	-	-	-	-	-	-	-	+	-	-	-
D	-	-	+	+	+	+	+	+	-	-	-	-	-	-
A	+	+	+	+	+	+	+	-	+	-	+	+	-	-
BLO	-	-	-	-	-	-	-	-	-	-	+	-	+	+
R	-	-	-	+	-	-	-	+	-	-	-	-	-	-
FPO	-	-	-	-	-	-	-	-	-	-	+	-	+	+
IO	+	+	+	+	+	+	+	+	+	+	+	+	-	-
IVEC	-	-	-	-	-	-	-	-	-	-	-	-	-	-
IVAL	-	-	-	-	-	-	-	-	-	-	-	-	-	-
IF	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Uygulamalar bilgisayar bilimleri, matematik ve olasılık teorisi gibi yaygın bilinen problemlerden oluşmaktadır. Seçilen problemler sıralama algoritması, arama algoritması, matris çarpımı, faktöriyel hesaplama, asal sayı bulma, binom dağılımı ve lineer regresyon denklemi hesaplamalarıdır.

- Tüm uygulamalar için oldukça basit bir isimlendirme şeması belirledik. Uygulamalar ilgili problemlerin isimlerinden oluşmaktadır. Örneğin matris çarpımı yapan programın adı matrix_multiplication şeklindedir. Ayrıca her bir program hem tek iş parçacıklı hem de çok iş parçacıklı sürümleri ile yazıldığı için programın paralel ve sıralı sürümlerini "p" harfiyle ayırt ettik. Örneğin sıralı matris çarpım programının ismi "matrix_multiplication" olurken, paralel kodlanan matris çarpımı işlemi "p_matrix_multiplication" olarak adlandırılmıştır. Uygulamaların kısa açıklamaları aşağıdaki gibidir:

- p_selection_sort: Bu program paralel seçim sıralama işlemi gerçekleştirir. Program bir dizinin elemanlarını artan düzende sıralar. Her bir dizi elemanın değeri rastgele atanır.
- selection_sort: Bu program tek iş parçacığı ile seçim sıralama işlemi gerçekleştirir. Program bir dizinin elemanlarını artan düzende sıralar. Her bir dizi elemanın değeri rastgele atanır.
- p_array_search: Bu program dört iş parçacığı yardımıyla arama işlemini gerçekleştirir. Dizi dört parçaya bölünür ve her bir iş parçacığı dizinin bir parçası üzerinde çalışır. Bu programda aranacak olan dizi elemanları program içine gömülmüştür. Böylelikle program tek yol (single path) olarak çalışır.
- array_search: Bu program tek iş parçacığı ile dizi içerisinde arama işlemi gerçekleştirir. Bu programda aranacak olan dizi elemanları program içine gömülmüştür. Böylelikle program tek yol (single path) olarak çalışır.
- p_matrix_multiplication: Bu program matris çarpma programının paralel sürümüdür. Program dört iş parçacığı kullanarak iki adet 4x4 lük matrisin çarpma işlemini gerçekleştirir. Matrisin elemanları rastgele üretilmektedir.
- matrix_multiplication: Bu program 4x4 lük iki matrisin rastgele üretilen elemanlarını çarpma işlemini gerçekleştirir.
- p_factorial: Bu program iki iş parçacığı ile harici girdi olarak belirlenen sayının faktöriyel hesaplama işlemini gerçekleştirir.
- factorial: Bu program harici girdi olarak belirlenen sayının faktöriyelini hesaplama işlemi yapar.
- p_prime_numbers: Bu program bilinen iki değer arasında kalan sayılardan asal sayı olanları dört iş parçacığı kullanarak bulur.
- prime_numbers: Bu program bilinen iki değer arasında kalan sayılardan asal sayı olanları tek iş parçacığı kullanarak bulur.
- p_binomial_distribution: Bu program paralel binom dağılımını hesaplar. Bu hesaplama işlemi için üç iş parçacığı kullanılmıştır.
- binomial_distribution: Bu program tek iş parçacıklı olarak binom dağılımı hesaplaması yapar.
- p_linear_regression: Bu program paralel lineer regresyon denklemini hesaplar. Bu hesaplama işlemi için altı adet iş parçacığı kullanılmıştır.

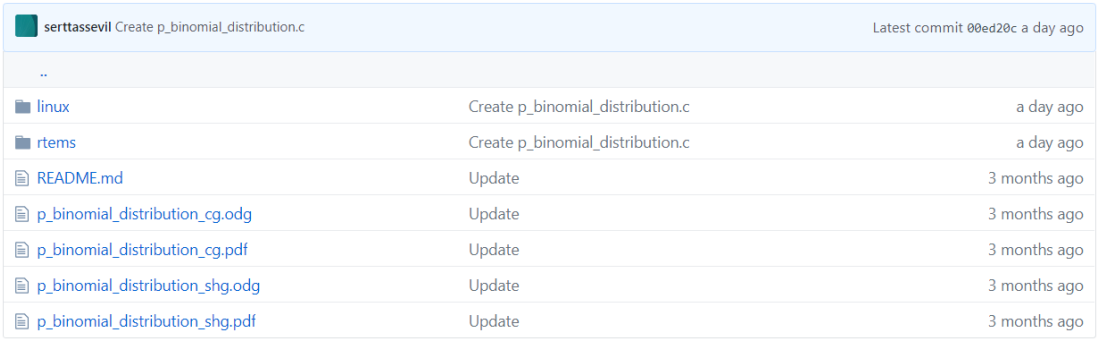
- linear_regression: Bu program tek iş parçacıklı olarak lineer regresyon denklemi hesaplaması yapar.

Geliştirme platformu için xubuntu 18.04 Linux dağıtımını kullandık. Programlarımızı C programlama dili ile geliştirdik ve GNU Compiler Collection (GCC) 6.3.0 sürümü ile tüm uygulamalarımızı başarıyla derledik. Programların tümü RTEMS gerçek zamanlı işletim sisteminde de derlenmiştir. Emülatör, SPARC ERC32 BSP'dir (Board Support Package). Çok iş parçacıklı paralel programalar için Pthreads API'si kullanılmıştır.

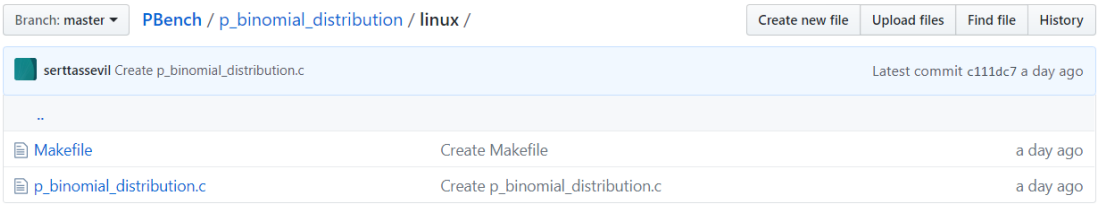
5.3. Dizin Yapısı

PBench'teki her bir uygulamanın kendine ait bir dizin yapısı mevcuttur. Aşağıda bu dizin yapısı detaylandırılmıştır.

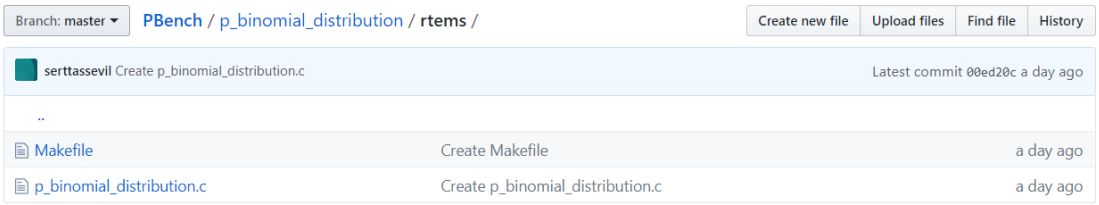
- Programın Linux dağıtımı için C kaynak kodu dosyası
- Programın Linux dağıtımı Makefile dosyası
- Programın RTEMS için C kaynak kodu dosyası
- Programın RTEMS için Makefile dosyası
- Programın README.md dosyası
- Programın çağrı grafiği dosyası (.pdf)
- Programın kapsam hiyerarşi grafiği dosyası (.pdf)



Şekil 5.1. p_binomial_distribution programı için genel dizin yapısı



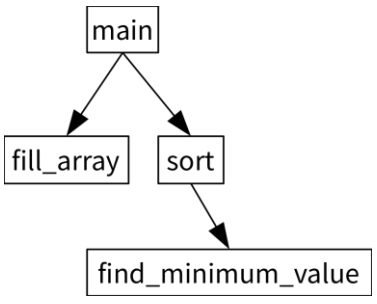
Şekil 5.2. p_binomial_distribution programının linux klasörünün dizin yapısı



Şekil 5.3. p_binomial_distribution programının RTEMS klasörünün dizin yapısı

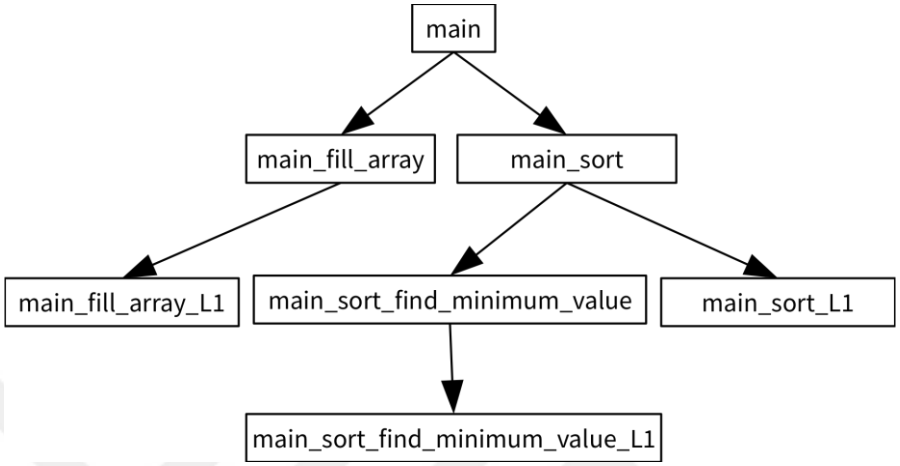
Programın linux dizini içerisindeki C kaynak kodu dosyası xubuntu 18.04 üzerinde derlenmiş C kodlarının olduğu dosyadır. Programın linux dizini içerisindeki Makefile dosyası xubuntu 18.04 üzerinde derleme sürecine yardımcı olmak için oluşturulmuş dosyadır. Programın rtems dizini içerisindeki C kaynak kodu dosyası rtems üzerinde derlenmiş C kaynak kodlarını temsil eder. Aynı şekilde rtems dizini içerisindeki Makefile dosyası rtems üzerinde derleme sürecine yardımcı olması için oluşturulmuştur.

Programın ana dizinindeki README.md dosyası program hakkında genel bilgileri içermektedir. Programın ana dizinindeki çağrı grafiği dosyası program akışını kolayca anlamayı sağlayan bir grafiktir. Programın ana dizinindeki kapsam hiyerarşi grafiği dosyası ise programın fonksiyon çağırımlarını ve döngü girişlerini kolayca anlamamızı sağlayan grafiktir. Aşağıda çağrı grafiği ve kapsam hiyerarşi grafiği şekiller ile detaylandırılmıştır.



Şekil 5.4. selection_sort programı için çağrı grafiği

Programın çağrı grafikleri ile hangi fonksiyonun hangi fonksiyon çağırımlarında bulunduğunu kolaylıkla anlayabiliriz. Şekil 3.4.' te main fonksiyonu fiil_array ve sort isminde iki fonksiyona sahiptir. Aynı şekilde sort fonksiyonu da find_minimum_value isminde bir fonksiyona sahiptir.



Şekil 5.5. selection_sort programı için kapsam hiyerarşi grafiği

Programın kapsam hiyerarşi grafikleri ile fonksiyonların içerdikleri döngüleri ve yaptıkları fonksiyon çağırımlarını kolaylıkla anlayabiliriz. Şekil 3.5.'te main fonksiyonu fiil_array ve sort isminde iki fonksiyona sahiptir. Fill_array fonksiyonu ise main_fill_array_L1 isminde bir döngüye sahiptir. Sort fonksiyonu find_minimum_value isminde bir fonksiyona ve main_sort_L1 isminde bir döngüye sahiptir. Benzer şekilde find_minimum_value fonksiyonu da main_find_minimum_value_L1 isminde bir döngüye sahiptir.

BÖLÜM 6. TARTIŞMA

PBench kıyaslama uygulama takımındaki uygulamaların RTEMS sürümlerine ilişkin iki özel durumla karşılaşılmıştır. Bu durumlar aşağıda açıklanmaktadır.

rand() fonksiyonu: matrix_multiplication ve p_matrix multiplication uygulamaları rastgele sayı üretmektedir. Dolayısıyla rastgele sayı üretmek için rand() fonksiyonu kullanmaktadır. Uygulamaların RTEMS sürümlerinin koşturuluduğu SPARC emülatörü bir emülatör olduğundan dolayı her çalışma başında sistem zamanını sıfırlamaktadır. Uygulamanın makine kodlarında değişiklik yapılmadığından dolayı rand() fonksiyonu kodları her koşturmada aynı noktada bulunmakta ve dolayısıyla aynı zamanı temel olarak almaktadır. Bundan dolayı rand() fonksiyonu ilgili program için sürekli olarak aynı değeri üretmektedir. Diğer bir deyişle rastgele sayı üretmek için bir rastgelelik kaynağı bulunamamaktadır.

Bundan dolayı bu programların RTEMS sürümlerinde rastgele sayı üretimi gerçekleştirilmedi. Bunun yerine sabit değerler uygulamanın kaynak kodu içerisine el ile yazıldı.

Değer Atama: Programın çalıştırılmasına başlanırken komut satırından değer alan factorial ve p_factorial uygulamalarında da sorun yaşanmıştır. Bir gerçek zamanlı işletim sistemi olan RTEMS işletim sistemi doğası gereği komut satırından değer almak için tasarlanmadığından dolayı bu özelliği desteklememektedir. Dolayısıyla bu uygulamaların RTEMS sürümlerinde komut satırından değer alma işlemi yapılmamış; bunun yerine ilgili değerler programın kaynak koduna el ile yazılmıştır.

BÖLÜM 7. SONUÇ

Bu tez çalışması ile birlikte çok işlemcili veya çok çekirdekli donanımlar üzerine inşa edilmiş, gerçek zamanlı sistemlerin WCET analizi için paralel gerçek zamanlı kıyaslama uygulama takımı gerçekleştirilmiştir. Bilim insanları, araştırmacılar, mühendisler, gerçek zamanlı sistem testleri ile uğraşan herkes bu uygulama takımından faydalanarak sistem testlerini gerçekleştirebilirler. PBench adını verdiğimiz bu uygulama takımı aynı programın hem sıralı programlama tekniği ile hem de paralel programlama tekniği ile kodlanmış sürümlerini içermektedir. Bu sayede WCET analizi ve gerçek zamanlı sistem testlerini gerçekleştirecek olan kişiler hem uygulamalarını hem de programlama tekniğinin uygulamalar üzerindeki etkisini inceleyebilme şansı bulabilirler. Uygulamaların tümü Linux dağıtımında ve RTEMS gerçek zamanlı işletim sisteminde koşturulmuştur. Uygulamlarda kullanılan yazılımsal çeşitlilik, (döngüler vb.) gerçek zamanlı sistem testlerindeki başarımı artıracaktır.

KAYNAKLAR

[1] S. E. Sim, S. Easterbrook, and R. C. Holt, "Using benchmarking to advance research: a challenge to software engineering," *25th Int. Conf. Softw. Eng. 2003. Proceedings.*, pp. 74–83, 2003.

[2] A. Grama, A. Gupta, G. Karypis, and V. Kumar, "Introduction to Parallel Computing, Second Edition," Addison-Wesley, 2003, p. 656.

[3] B. K. David and H. Wen-mei W., *Programming Massively Parallel Processors, 2nd Edition*. Morgan Kaufmann, 2012.

[4] J. Diaz, C. Muñoz-Caro, and A. Niño, "A survey of parallel programming models and tools in the multi and many-core era," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 8, pp. 1369–1386, 2012.

[5] H. Kopetz, *Real-Time Systems*. Boston, MA: Springer US, 2011.

[6] R. Wilhelm *et al.*, "The worst-case execution-time problem—overview of methods and survey of tools," *ACM Trans. Embed. Comput. Syst.*, vol. 7, no. 3, pp. 1–53, Apr. 2008.

[7] J. Abella *et al.*, "WCET analysis methods: Pitfalls and challenges on their trustworthiness," in *10th IEEE International Symposium on Industrial Embedded Systems (SIES)*, 2015, pp. 1–10.

[8] http://rtsrlab.sakarya.edu.tr., Erişim Tarihi: 13.08.2018.

[9] J. Abella, D. Hardy, I. Puaut, E. Quinones, and F. J. Cazorla, "On the comparison of deterministic and probabilistic WCET estimation techniques," in *Proceedings - Euromicro Conference on Real-Time Systems*, 2014, pp. 266–275.

[10] S. Altmeyer, L. Cucu-Grosjean, and R. I. Davis, "Static probabilistic timing analysis for real-time systems using random replacement caches," *Real-Time Syst.*, vol. 51, no. 1, pp. 77–123, 2015.

[11] S. Altmeyer and R. I. Davis, "On the correctness, optimality and precision of Static Probabilistic Timing Analysis," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014*, 2014, pp. 1–6.

[12] L. Kosmidis *et al.*, “Fitting processor architectures for measurement-based probabilistic timing analysis,” *Microprocess. Microsyst.*, vol. 47, pp. 287–302, 2016.

[13] F. J. Cazorla *et al.*, “PROXIMA: Improving Measurement-Based Timing Analysis through Randomisation and Probabilistic Analysis,” in *2016 Euromicro Conference on Digital System Design (DSD)*, 2016, pp. 276–285.

[14] http://www.otawa.fr/., Erişim Tarihi: 12.02.2019.

[15] X. Li, Y. Liang, T. Mitra, and A. Roychoudhury, “Chronos: A timing analyzer for embedded software,” *Sci. Comput. Program.*, 2007.

[16] http://www.mrtc.mdh.se/projects/wcet/sweet/index.html.,Erişim Tarihi:12.01.2019.

[17] https://www.rapitasystems.com., Erişim Tarihi: 10.02.2019.

[18] https://www.absint.com., Erişim Tarihi: 13.09.2018.

[19] http://www.bound-t.com/., Erişim Tarihi: 12.02.2019.

[20] https://www.spec.org., Erişim Tarihi: 13.08.2018.

[21] S. Che *et al.*, “Rodinia: A Benchmark Suite for Heterogeneous Computing.”

[22] X. Tian *et al.*, “BigDataBench-S: An open-source scientific big data benchmark suite,” in *Proceedings - 2017 IEEE 31st International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2017*, 2017, pp. 1068–1077.

[23] C. Bienia, S. Kumar, J. P. Singh, and K. Li, “The PARSEC Benchmark Suite: Characterization and Architectural Implications,” in *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques*, 2008, pp. 72–81.

[24] J. Gustafsson, A. Betts, A. Ermedahl, and B. Lisper, “The mälardalen WCET benchmarks: Past, present and future,” in *10th International Workshop on Worst-Case Execution Time Analysis, WCET 2010*, 2010, vol. 15, pp. 136–146.

[25] S. M. Blackburn *et al.*, “The DaCapo Benchmarks: Java Benchmarking Development and Analysis,” in *Proceedings of the 21st Annual ACM SIGPLAN Conference on Object-oriented Programming Systems, Languages, and Applications*, 2006, pp. 169–190.

[26] G. Bollella *et al.*, *The {Real}-time {Specification} for {Java}*. Addison-Wesley, 2000.

[27] P. Parizek, G. Haddad, G. T. Leavens, J. Vitek, and T. Kalibera, "Challenge benchmarks for verification of real-time programs," in *ACM SIGPLAN Notices*, 2010, vol. 44, no. 11, p. 7.

[28] F. Nemer, H. Cassé, P. Sainrat, J.-P. Bahsoun, and M. De Michiel, "PapaBench: a Free Real-Time Benchmark," in *6th International Workshop on Worst-Case Execution Time Analysis (WCET'06)*, 2006, vol. 4.

[29] H. Falk *et al.*, "TACLeBench: A Benchmark Collection to Support Worst-Case Execution Time Research," *16th Int. Work. Worst-Case Exec. Time Anal. (WCET 2016)*, vol. i, no. 2, p. 10, 2016.

[30] https://www.eembc.org., Erişim Tarihi: 21.02.2019.

[31] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown, "MiBench: A free, commercially representative embedded benchmark suite," *2001 IEEE Int. Work. Workload Charact. WWC 2001*, pp. 3–14, 2001.

[32] http://vhosts.eecs.umich.edu/mibench/., Erişim Tarihi: 09.01.2019.

[33] A. G. G. A. Silberschatz, P. B. Galvin, "Operating System Concepts," *John Wiley Sons, Inc.*, vol. 7, no. 3, pp. 783–837, 2005.

[34] E. Applications, *Real Time Systems : Design Principles for Distributed Embedded Applications*. .

[35] G. C. Buttazzo, *Hard Real-Time Computing Systems*, vol. 24. Boston, MA: Springer US, 2011.

[36] https://www.rtems.org/., Erişim Tarihi: 21.02.2019.

[37] https://www.zephyrproject.org/., Erişim Tarihi: 21.02.2019.

[38] https://www.ecoscentric.com/ecos/., Erişim Tarihi: 21.02.2019.

[39] https://riot-os.org/., Erişim Tarihi: 21.02.2019.

[40] https://www.osrtos.com/., Erişim Tarihi: 21.02.2019.

[41] http://www.contiki-os.org/., Erişim Tarihi: 21.02.2019 .

[42] https://marte.unican.es/., Erişim Tarihi: 21.02.2019.

ÖZGEÇMİŞ

Sevil Serttaş, 22.01.1992'de Sakarya'da doğdu. İlk, orta ve lise eğitimini Sakarya'da tamamladı. 2010 yılında Tes-İş Adapazarı Anadolu Lisesi'nden mezun oldu. 2010 yılında başladığı Sakarya Üniversitesi Bilgisayar Mühendisliği Bölümü'nü 2014 yılında bitirdi. 2015 yılında Sakarya Üniversitesi Bilgisayar ve Bilişim Mühendisliği Bölümü'nde yüksek lisans eğitimine başladı. 2018 yılında Kütahya Dumlupınar Üniversitesi'nde araştırma görevlisi olarak çalışmaya başladı. Halen Kütahya Dumlupınar Üniversitesi Bilgisayar Mühendisliği Bölümü'nde Araştırma Görevlisi olarak görev yapmaktadır.