Süleyman Demirel Üniversitesi,
Fen Bilimleri Enstitüsü Dergisi,
16-3 (2012), 260-263

Aritmetik Algoritmalar Kullanılarak Farklı Donanım Platformlarında Benchmark Testi Uygulaması

Arif KOYUN[1]*, Hüseyin Bilal MACİT[2]
[1] *Süleyman Demirel Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü / ISPARTA*
[2] *Süleyman Demirel Üniversitesi, Mühendislik Fakültesi, Elektronik ve Haberleşme Mühendisliği Bölümü / ISPARTA*
Alınış Tarihi:08.08.2012, Kabul Tarihi:15.11.2012

Özet: Bu çalışmada Borland Delphi yazılım geliştirme platformunda yeni bir aritmetik benchmark uygulaması geliştirilmiştir. Geliştirilen uygulama, 8 farklı algoritma test yaparak sistemin aritmetik işlem performansını ölçmekte, test sonuçları ile donanım bileşenlerine sayısal skor vermektedir. Geliştirilen uygulama, 10 farklı donanım platformunda incelenmiş ve skorlar elde edilmiştir. Üretilen skorlar, passmark skorları ile karşılaştırılıp, basit matematiksel işlemlerle oluşturulan stresle elde edilen skorların başarımı incelenmiştir.

Anahtar kelimeler: kıyaslama, algoritma, aritmetik, skor

Benchmark Test Application using Aritmethic Algorithms on Different Hardware Platforms

Abstract: This study developed a new arithmetic benchmark application with Borland Delphi software developing platform. This new application measures the system' s arithmetical processing performance with 8 different algorithms and gives numerical score. Application developed ran on 10 different hardware platforms and the scores obtained. Produced scores are compared with passmark scores and performance scores that obtained from the stress created with simple mathematical operations.

Keywords: benchmark, algorithm, arithmetic, score

Giriş

20. yüzyılın ortalarında bir odaya sığmayacak kadar büyük olan bilgisayarlar, çalışmak için binlerce watt gücünde elektrik kaynağına ihtiyaç duymakta, ancak sadece toplama çıkarma gibi basit matematiksel işlemleri yapabilmekteydi. Transistörlerin kullanılmasıyla bilgisayarların boyutu küçülmeye başlamış, üretim maliyetleri, enerji sarfiyatları azalmış, işlem kapasiteleri artmıştır. 21. Yüzyıl başlarında gelişmiş ülkelerde neredeyse her eve girmiş olan bilgisayarlar, kurumsal alanda da ciddi boyutta kullanılmaya başlamıştır. Günümüzde internetin de yaygınlaşması ile neredeyse kişi başına bir bilgisayar düşmektedir.

Bilgisayar ve bileşenlerinin üretiminde marka çeşitliliğinin artması rekabeti doğurmuş, rekabet de ürün çeşitliliğinin artmasını sağlamıştır. Tüketicinin sorunu; bilgisayarı veya bileşenlerini satın alırken hangi kriterlere göre ürün satın alacağını seçmek, üreticinin sorunu da ürününü nasıl pazarlayacağını ve bir sonraki ürünü bir öncekinden nasıl daha hızlı hale getireceğini bulmak olmuştur. Bu sorunun çözümünde hem tüketiciye hem üreticiye yardımcı olacak en iyi çözüm benchmark yazılımları olmuştur. Tüketici ürün satın alırken, satın almak istediği ürünün benchmark sonuçlarına göre karar vermekte, üretici ürün geliştirirken ve pazarlarken yine benchmark sonuçlarını baz almaktadır.

Bir bilgisayarın veya uygulamanın ne kadar başarılı olduğu, üç kritere göre nitelendirilir (Okulicka, 2010);

- Hız
- Fiyat
- Verimlilik

Benchmarking sözcüğü, benchmark' tan gelmektedir. Benchmark bir ölçü birimi ve benchmarking ise bu ölçümün yapılmasıdır. Henüz Türkçe' de karşılığı tam olarak benimsenmemiş olsa da, benchmarking; referans alma, örnek alma ve en çok da kıyaslama olarak kullanılmaktadır. İşletme biliminde benchmark' ın tanımı; seçilen konuda en iyi olabilmek için kendi ürünlerini, hizmetlerini ve uygulamalarını, rakipleri ya da endüstrinin önde gelen diğer şirketleri ile beraber değerlendirerek gelişme sürecidir (Baş, 2012).

Endüstride, iki firma arasında kıyaslama uygulaması yapabilmek için, iki şirketin işlem ve yöntemlerine ilişkin bilgileri paylaşma konusunda önceden anlaşmış olmaları gerekir. Her iki şirket de bilgi değişiminden bazı kazançlar beklediği için dürüst sonuçlar vermek durumundadır.

Bilgisayar dünyasında benchmark, bir bileşenin başarımını, o bileşen üzerinde çeşitli sınamalar yaparak ölçmek veya tüm sistemin başarımını, farklı algoritmalar kullanarak ölçmek ve başka bileşenlerle kıyaslanabilecek şekilde skorlar üretmektir.

Tüketici, bir bilgisayar bileşenini satın alırken farklı marka ve modeller arasında tercih yapmak durumundadır. Bu tercihi yaparken kullanılan kritere fiyat/performans denir. Fiyat her zaman ulaşılabilir bilgi olduğuna göre, önemli olan bileşenin performansını ölçebilmektir. Bunun için firmaların kendi verdiği performans değerlerine güvenmek yerine tarafsız test yazılımlarının sonuçlarına göre hareket etmek daha doğru olmaktadır (Kapoor, 2012).

* *arifkoyun@sdu.edu.tr*

Materyal ve metot

Uygulama geliştirilirken, programlama dili olarak Borland Delphi yazılımının 7. sürümü kullanılmıştır. Programın çalıştırılması ve test aşamasında 10 farklı bilgisayar kullanılmıştır. Sonuçların test edilmesi aşamasında güvenilirlik ölçümü için passmark cpu benchmark sonuçları baz alınmıştır.

Aritmetik test aşamasında 8 farklı benchmark algoritması uygulanmaktadır. Her algoritma tamamlandıktan sonra ürettiği sayısal skoru herhangi bir log dosyasına kaydetmektedir.

Asal sayı testi

Asal sayı bulma algoritması, neredeyse tüm CPU benchmark yazılımlarında kullanılan basit ve etkili bir algoritmadır. Amacı ardışık sayıların içinden asal olanları bulmaktır. Algoritma bir sayının asal olup olmadığını anlamak için o sayısı kendisinden küçük tüm sayılara bölmektedir. İstenilen sayıda üst üste döngü ve kıyaslama işleminden oluştuğu için işlemci üzerinde stres oluşturan bir algoritmadır.

```
gecensure:=gettickcount; // Süreyi kaydet   } (t)
for i:=1 to 50000 do
// 1 den 50,000'e kadar asalları bulunsun
begin
  carpan:=0;
  For j:=1 To i Do                          } (n)
    If (i mod j=0) Then carpan:=carpan+1;
    If carpan=2 Then asal:=True
    else asal:=false;
end;
gecensure:=(gettickcount - gecensure);      }
// Süreyi kaydet                            } (t)
```

Bu algoritma 1'den 50.000'e kadar olan asal sayıları bulmak için 50.000 faktöriyel bölme işlemi yapmaktadır. Büyük O notasyonu kuralına göre bu algoritmanın çalışma zamanı;

O(N)=n+t+t
O(N)=n

't' (satır tek işlem) değerleri büyük O notasyonunda göz ardı edildiği için algoritmanın hızı 'n' olarak gösterilecektir. 'n' değeri kısaca dışarıdaki for döngüsünün başlangıcından bitimine kadar geçen zamandır. Algoritma dizi üzerinde işlem yapmadığı için her çalışmada aynı performansı gösterir, hızı logaritmik olarak değişmez.

Çizelge 1'de gösterildiği gibi, asal sayı ve diğer algoritmaların çalıştırılması ile donanım platformuna skor verebilemek için 'n' değeri yani ölçülebilir çalışma zamanı ve kullanılan bellek miktarı kullanılmaktadır.

Çizelge 1. *Algoritma sonucu oluşturulan skor*

Test	Skor	Süre	Bellek
Asal Sayı	1920	521,1ms	5764KB

Matris çarpım testi

Matris çarpım da, asal sayı bulma gibi işlemci üzerinde stres oluşturabilecek, iç içe tekrarlanan döngüler ve matematiksel işlemlerden oluşan bir aritmetik algoritmadır. Amacı yüksek eleman sayısına sahip iki boyutlu iki matrisin defalarca çarpımı ile işlemci üzerinde stres oluşturmaktır. İki matrisi 1000 defa çarpan kod satırı aşağıda gösterilmiştir;

```
for sayac:=1 to 1000 do                                   } (n)
begin
  for i := 0 to satir do                                  }
    for j := 0 to sutun do                                } (n)
      for k := 0 to index do                              }
        C[ i, j ] := C[ i, j ] +                          } (n)
        ( A[ i , k] * B[k, j                              } (n)
end;
```

Büyük O notasyonuna göre algoritmanın çalışma zamanı;

O(N)=n*n*n*n*t
$O(N)=n^4$

olarak hesaplanır. Son döngü içindeki satırın çalışma zamanı 't', büyük O notasyonuna göre ihmal edilmektedir.

Metin ve imaj sıkıştırma-çözme testi

Zip kelimesi; bir dosya uzantısını ifade etmekle beraber, bir veya birkaç dosyanın bir paket haline sıkıştırılması ile oluşan dosyalara verilen isimdir. Microsoft'un Windows Vista ve 7 adlı işletim sistemlerinin kütüphane dosyalarının arasında, zip (sıkıştır) ve unzip (çöz) işlemlerini yapan dll kütüphane dosyaları bulunmaktadır. Aşağıdaki Delphi kod kümesi kısaca zip işlemini yapmaktadır;

```
gecensure:=gettickcount; // Süre hesapla
lboFilesToZip.Items.Add('c:\hbm\PCBench\Temp\text.txt
'); // Ziple
ZipFiles('c:\hbm\PCBench\Temp\text.zip',
lboFilesToZip.Items); // Zip'i kaydet
gecensure:=(gettickcount - gecensure); // Süre hesapla
```

Aşağıdaki kod satırı unzip işlemini yapmakta olan kod satırının parçasıdır;

```
edtFileToUnzip.Text := 'c:\hbm\PCBench\Temp\text.zip';
edtUnzipToDir.Text := 'c:\hbm\PCBench\Temp\';
```

Zip ve unzip işlemleri harddisk-bellek-işlemci ve tam tersi yönde akış kullandığı için aynı platformda bile her çalıştırıldığında farklı sonuç üretir. Çünkü çalışma zamanı sonucu, donanıma bağlı olduğu kadar işletim sistemi mimarisi, harddisk dosya sistemi, işlemci önbellekleme algoritması gibi yazılımsal değerlere bağlıdır.

Sıralama testi

Bu testte; 0-10000 arasında rasgele sayısal değerlerden oluşan on bin haneli karakter dizisi tanımlanmaktadır. On bin haneli bu dizi, yerleştirmeli sıralama, kabarcık sıralama ve hızlı sıralama adlı üç farklı algoritmanın her biri ile 10'ar kez sıralanmaktadır.

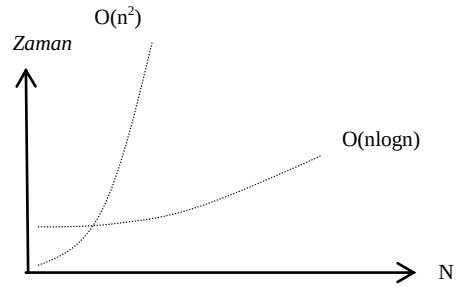
Büyük O notasyonu ile kabarcık sıralama incelenirse;

- Kabarcık sıralama iki adet for döngüsü kullanır.
- Dıştaki döngü, işlemin (n-1) defa tekrarlanmasını sağlar.
- İçteki döngü (n-1) kıyaslama yapar.
- n=10000 (döngü sayısı) ve işlem çalışma zamanı 't' varsayılmaktadır.

Algoritmada, büyük o notasyonu ile çalışma zamanı hesaplanması;

$O(N)=(n-1)*(n-1)*t$
$O(N)=n^2-2n+1+t$
$O(N)=n^2$

Şeklinde gösterilecektir. O notasyonundan farklı olarak, pratikte dizi sıralandıkça çalışma zamanı logaritmik olarak azalacaktır. Ω notasyonu ile algoritmanın çalışma zamanı Şekil 1' de gösterildiği gibi log(n) olarak ifade edilecektir.



Şekil 1. *Kabarcık sıralamada büyük O ve Ω notasyonu gösterimi.*

Yerleştirmeli sıralama algoritması, $O(n^2)$ tipinde bir algoritmadır. Seçimli sıralama algoritması gibi iki döngüye sahiptir. for döngüsü, (n-1) kere çalışır ve elemanı sıralanmış elemanlar içindeki doğru yerine yerleştirir. While döngüsü, dizinin kalan elemanları üzerinde aynı işlemini gerçekleştirir. İlk aşamada, (n-1) işlem gerçekleştirilir, ikinci aşamada (n-2) işlem gerçekleştirilir ve bu şekilde $O(n^2)$ tipinde bir yapı oluşmaktadır.

$O(N)=(n-1)*(n-2)$
$O(N)=n^2-3n+2$
$O(N)=n^2$

Hızlı sıralama algoritması, sıralanacak bir sayı dizisini daha küçük iki parçaya ayırıp oluşan bu küçük parçaların kendi içinde sıralanması mantığıyla çalışır. İlk önce sayı dizisinden herhangi bir sayı pivot eleman olarak seçilir. Sayı dizisi, pivot sayıdan küçük olan tüm sayılar pivotun önüne, pivottan büyük olan tüm sayılar pivotun arkasına gelecek biçimde tekrar düzenlenir. Bu bölümlendirme işleminden sonra eleman sıralanmış son dizide olması gerektiği yere yerleştirilir. Pivotun sol ve sağında olmak üzere oluşan iki ayrı küçük sayı dizisi üzerinde, hızlı sıralama algoritması yinelemeli olarak yeniden çağrılır. Algoritma; eleman sayısı sıfır olan bir alt diziye ulaştığında bu dizi sıralanmıştır.

Hızlı sıralama algoritması n adet sayıyı, ortalama veya en iyi durumda O(n log n) karmaşıklığıyla, en kötü durumda ise $O(n^2)$ karmaşıklığıyla sıralar. $O(n^2)$ olan durumlarda, hızlı sıralama algoritması, diğer basit işleyişe sahip sıralama algoritmalarından daha hızlı değildir.

Sıralama algoritmalarında yalnızca n değerlerinin ölçümü ile net sonuç elde edilemediğinden, tekrarlanan döngülerin toplam zamanı uygulamada aşağıdaki kodlarla hesaplanmıştır.

```
gecensure:=gettickcount; // Süre hesapla
Sıralama Algoritmasını Uygula
gecensure:=(gettickcount - gecensure); // Süre hesapla
```

Bmp' den jpg' ye çevrim testi

JPEG standardında görüntü saklayan dosya biçimi de çoğunluk tarafından JPEG olarak adlandırılır. Bu dosyalar genellikle *.jpg*, *.jpe* ya da *.jfif* uzantılıdır, ancak çoğunlukla *.jpg* uzantısı kullanılmaktadır. (Patel vd., 2009) JPEG; dijital kameralarda ve diğer fotoğrafik görüntü yakalama cihazları tarafından kullanılan en yaygın görüntü biçimidir. Her renk bileşeni, 8x8 bloklar halinde ayrık kosinüs dönüşümü ile dönüştürülür, bu sayede resmin enerjisi az sayıda dönüşüm uzayındaki pikselde yoğunlaştırılır.

Bulgular

Geliştirilen uygulama, algoritmaları kullanarak ürettiği skorları Çizelge 2' deki gibi kullanıcıya sunmaktadır.

Çizelge 2. *Uygulamanın örnek sonuç gösterimi*

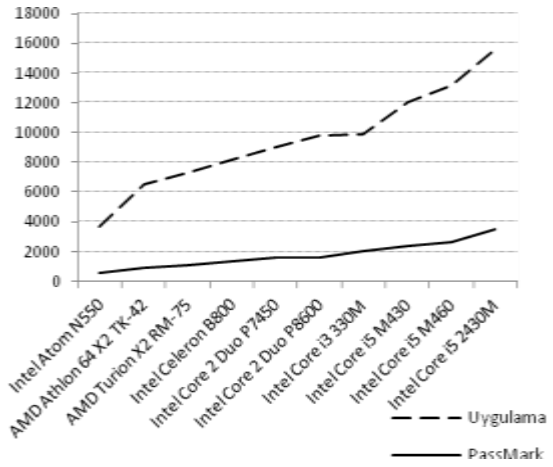
Benchmark	Skor	Süre	Bellek	Hız
Asal sayı Sıralama Matris çarpım	3068	381ms	5759KB	18832392 döngü/sn
Zip / Unzip Bmp->Jpg	2861	392ms	6850KB	19032180 döngü/sn

Elde edilen sonuçların güvenirliğini kanıtlamak için, 10 farklı platformda yapılan testlerin sonucu, passmark benchmark sonuçları ile kıyaslanmış ve Şekil 2' de ifade edilmiştir. Passmark yazılım, merkezi Avusturalya, Sydney' de bulunan bir yazılım geliştirme firmasıdır. 1998 yılında kurulan firma, geliştirdiği benchmark araçları ile, belirli bir standart içinde bugüne kadar, 300 ün üzerinde farklı işlemcinin kullanıldığı 20.000'den fazla

sistemin benchmark işlemini yapmış ve benchmark sonuçlarını web sitesinde yayınlamıştır.

Çizelge 3. *Oluşturulan skorların passmark sonuçlarına göre yanılma yüzdesi*

İşlemci Modeli	Yanılma Yüzdesi
Intel Atom N550	%5,48
AMD Athlon 64 X2 TK-42	%6,25
AMD Turion X2 RM-75	%5,59
Intel Celeron B800	%4,93
Intel Core 2 Duo P7450	%4,75
Intel Core 2 Duo P8600	%5,03
Intel Core i3 330M	%3,95
Intel Core i5 M430	%4,03
Intel Core i5 M460	%3,96
Intel Core i5 2430M	%3,48



Şekil 2. *Sonuçların passmark test sonuçları ile kıyaslaması.*

Tartışma ve Sonuç

Benchmark yazılımlarının birçoğu internet üzerinde ücretsiz temin edilebilmektedir. Üreticilerin baz aldığı benchmark yazılımları ise genellikle yüksek ücretler karşılığında satılmaktadır. Dolayısıyla, tüketici, kendisi yapmadığı bir testi, üreticinin verdiği sonuçlara inanmak durumunda kalarak kabul etmekte ve bu sonuçlara göre ürün satın alma tercihi yapmaktadır. Bu durum haksız rekabete yol açmaktadır.

Bu çalışma ile geliştirilen uygulamanın ürettiği skorlar, passmark skorları ile kıyaslandığında, Çizelge 3' te gösterildiği gibi maksimum %6,25 yanılma payına ulaşmıştır. Bu değerler göstermektedir ki, sıralama, matris çarpım, zip, unzip, asal sayı gibi aritmetik algoritmalar, yüksek fiyatlara satılmakta olan benchmark yazılımları ile yakın sonuçlar üretmektedir.

Kaynaklar

Okulicka, F. 2010. Paralel Programming Lecture 14, Computer Performance. http://lib.bioinfo.pl/files/courses/pdfcache/lecture_461.pdf (Erişim Tarihi: 10.04.2012).

Baş, M. 2012. Çağdaş Yönetim Kuramlarından Biri Benchmarking, Gazi Üniversitesi İ.İ.B.F. http://www.mehmetbas.com/cagdas-yonetim-kuramlarindan-biri-bencmarking/ (Erişim Tarihi 12.04.2012)

Kapoor, M. 2012. Benchmarks Software –Computer Benchmarks. http://hamiltonfinancials.com/11_11/Benchmarking_Software-Computer_Benchmarking-PC_Benchmarking-Benchmarking_Computer.html. (Erişim Tarihi: 15.05.2012)

Patel, P., Wong, J., Tatikonda, M., Marczewski, J., 2009. JPEG Compression Algorithm Using CUDA, Department of Computer Engineering, University of Toronto, Course Project for ECE 1724.

Dongarra, J. J., 1983. Performance of various computers, Computer Architecture News.

Mah Ung, Gordon., Case, Loyd, 2010. How to Properly Benchmark Your PC.