

# BSM 420 – BİLGİSAYAR MİMARİLERİ

Değerlendirme Tekniği ve Ölçütlerin (Metrics) Seçimi

# DEĞERLENDİRME TEKNİĞİ SEÇİMİ

# Giriş

- Bir veya daha fazla sistemimiz var, gerçek veya varsayımsal
- Performanslarını değerlendirmek istiyoruz
- Hangi tekniği seçmeliyiz?
  - Analitik Modelleme?
  - Simülasyon?
  - Ölçüm?

# Değerlendirme Tekniği Seçimi

- Sistem hangi yaşam döngüsü aşamasında?
  - Bir şey var olduğunda >> ölçüm
  - Eğer yeni durumunda ise, analitik modelleme veya simülasyon
- Sonuçlar ne zaman gereklidir? (genellikle dün!)
  - Analitik modelleme tek seçenek
  - Simülasyonlar ve ölçüm aynı olabilir
- Hangi araçlar ve yetenekler mevcut?
  - Simülasyonu destekleyen diller
  - Ölçümü destekleyen araçlar (ör. Paket izleyiciler, izleme kodu)
  - Analitik modelleme becerileri (ör: kuyruk teorisi)

# Değerlendirme Tekniği Seçimi

- İstenen Doğruluk seviyesi?
  - Analitik modelleme kaba (doğru olduğu ortaya çıkarsa, analistler bile şaşırır!)
  - Simülasyonun daha fazla ayrıntısı vardır, ancak temel sistem ayrıntılarını soyutlayabilir
  - Ölçüm kulağa daha gerçekçi gelebilir, ancak iş yükü, yapılandırma vb. eksik olabilir
    - Doğru tasarım olmadan doğruluk yüksek olması bir anlam içermez
  - Doğru verilerle, doğru sonuçlar çıkarmanız gerekir
    - Örn: cevap süresi% 90 güvenle 10.2351'dir. Ne olmuş yani? Bu ne demek?

# Değerlendirme Tekniği Seçimi

- Alternatifler neler?
  - Analitik modeller - en kolay
  - Simülasyon - orta
  - Ölçüm - en zor
- Maliyet?
  - Ölçüm genellikle en pahalı
  - Analitik modelleme en ucuz (kalem ve kağıt )
  - Simülasyon ucuz ama bazı araçlar pahalı olabilir
    - Trafik üreticileri, ağ simülatörleri

# Değerlendirme Tekniği Seçimi

- Satabilirlik?
  - İnsanları ölçüm ile ikna etmek çok daha kolay
  - Sonuçları anlamak zor olduğu için insanlar analitik modelleme sonuçları hakkında şüphecidirler
    - Kullanımdan önce çoğu kez simülasyon ile geçerli kılınır
- İki veya daha fazla teknik kullanabilir miyiz?
  - Birini diğeriyle geçerli kıl
  - Birçok kaliteli performans değerlendirme makalesi analitik model + simülasyon veya ölçüm tekniğini kullanır

# Değerlendirme Tekniği Seçimi

<u>Kriterler</u>	<u>Modelleme</u>	<u>Simulation</u>	<u>Ölçüm</u>
1. Aşama	Tüm	Tüm	Prototype+
2. Gerekli zaman	Az	Orta	Değişir
3. Araç	Analist	Bazı diller	Enstuman
4. Doğruluk	Az	Orta	Değişir
5. Değerlendirme kolaylığı	Kolay	Orta	Zor
6. Maliyet	Düşük	Orta	Yüksek
7. Satılabilirlik	Düşük	Orta	Yüksek



# PERFORMANS ÖLÇÜTÜ SEÇİMİ

# Performans Ölçütleri

- Performans ölçütü nedir?
- İyi ölçütlerin karakteristikleri
- Standart işlemci ve sistem ölçütleri
- Hızlanma (Speedup) ve bağıl değişim



# Performans ölçütleri nedir?

- Sayı
  - Bir olayın kaç kere olduğu
- Süre
  - Bir zaman aralığı
- Boyut
  - Bir parametre
- Bu temel ölçümlerden türetilen bir değer.

# Zaman-normalize ölçütler

- “Oran” ölçütleri
  - Ortak zamana göre normalize edilen ölçüt
    - Saniye başına işler(Transactions)
    - Saniye başına byte
  - $(\text{Olay sayısı}) \div (\text{olayların olduğu zaman aralığı})$
- “Çıkış-Throughput”
- Farklı zaman aralıklarındaki ölçümleri karşılaştırmada kullanışlı

# «İyi» bir ölçütün faydaları

- Doğru ve detaylı karşılaştırma sağlar
- Doğru sonuçlara götürür
- Herkes tarafından anlaşılır
- Nicel bir temele sahiptir
- İyi bir metrik hatalı sonuçlardan kaçınmaya yardımcı olur

# İyi ölçütler...

- Doğrusal
  - Sezgimizle uygun olmalı
  - Eğer ölçüt 2x artarsa, performans ta iki kat artmalı
  - Mutlak bir gereklilik değil, ancak faydalı
    - Sesi ölçmek için dB ölçeği doğrusal değildir

# İyi ölçütler...

- Güvenilir
  - Eğer A ölçütü  $>$  B ölçütü ise
    - O zaman, Performance A  $>$  Performance B
- Ancak,
  - MIPS(A)  $>$  MIPS(B), ancak
  - Çalışma Zamanı (A)  $>$  Çalışma Zamanı (B)

# İyi ölçütler...

- Tekrarlanabilir
  - Deney ne zaman tekrarlansa aynı değer ölçülür
  - Deterministik olmalı
  - Her zaman doğru değil
    - Ör: İşletim sistemindeki zaman paylaşımı çalışma zamanına etki eder



# İyi ölçütler

- Kullanımı kolay
  - Eğer ölçümü zor ise kimse kullanmaz
  - Ölçümü / türetimi zor
    - → doğru bir şekilde ölçülmesi daha az muhtemel
- Yanlış bir değer kötü bir ölçütten daha kötü

# İyi ölçütler...

- Tutarlı
  - Birimler ve tanım sistem boyunca değişmez
  - Anlaşılır, ama her zaman doğru değil...
    - Ör. MIPS, MFLOPS
  - Tutarsız → karşılaştırma yapmak imkansız

# İyi ölçütler...

- Bağımsız
  - Performans sonuçları üzerine çok fazla yatırım yapılıyor
  - Belirli bir ölçütü optimize etme konusunda üreticilere yönelik baskı
  - Bir ölçütün tanımını etkileme baskısı
  - Ancak iyi bir ölçüt bu baskıdan bağımsızdır

# İyi ölçütler...

- Doğrusal -- *opsiyonel*
- Güvenilir-- *gerekli*
- Tekrarlanabilir-- *gerekli*
- Kullanımı kolay-- *opsiyonel*
- Tutarlı-- *gerekli*
- Bağımsız-- *gerekli*

# Saat Hızı (Frekansı)

- Daha hızlı saat == daha yüksek performans
  - 2 GHz işlemci her zaman 1 GHz den daha iyidir
- Ancak bu orantılı bir artış mı?
- Mimari farklılıklar?
  - Çevrim başına yerine getirilen gerçek işlem
  - Talimat başına çevrim (Clocks per instruction - CPI)
  - Boru hattı (pipeline) derinliği nedeniyle dallanmalardaki kayıp
- İşlemci darboğazda değilse ne olur?
  - Bellek ve I/O gecikmesi

# Saat Hızı

- (Daha hızlı saat)  
≠ (daha iyi performans)
- İyi bir birinci dereceden ölçüt

Linear	
Reliable	
Repeatable	😊
Easy to measure	😊
Consistent	😊
Independent	😊

# MIPS

- Hesaplama «hızı» ölçümü
- *Saniye başına çalışan milyon talimat (Millions of instructions executed per second)*
- $MIPS = n / (T_e * 1000000)$ 
  - $n$  = talimat sayısı
  - $T_e$  = çalışma zamanı
- Fiziksel benzerlik
  - Birim zamanda katedilen mesafe

# MIPS

- Ancak talimat başına hesaplama ne kadar gerçek?
  - E.g. CISC vs. RISC
  - Talimat başına çevrim (Clocks per instruction - CPI)
- MIPS = *Meaningless Indicator of Performance*
- Anlamsız Performans Göstergesi

Linear	
Reliable	
Repeatable	☺
Easy to measure	☺
Consistent	
Independent	☺



# MFLOPS

- «kat edilen mesafe» için daha iyi bir tanım
- 1 hesap birimi (~mesafe)  $\equiv$  1 kayan-noktalı işlem
- *Millions of floating-point ops per second*
- *Saniyede milyon kayan-noktalı işlem*
- $\text{MFLOPS} = f / (T_e * 1000000)$ 
  - $f$  = kayan-noktalı talimat sayısı
  - $T_e$  = çalışma zamanı
- GFLOPS, TFLOPS,...

# MFLOPS

- Tamsayı programı= 0 MFLOPS
  - Ancak hala faydalı işler yapar
  - Sıralama, arama, vs.
- FLOP nasıl sayılır?
  - Ör. transcendental ops, kökler
- FLOP tanımındaki esneklikler
- Makineler boyunca tutarlı değil

Linear	
Reliable	
Repeatable	☺
Easy to measure	☺
Consistent	
Independent	

# SPEC

- System Performance Evaluation Coop
- Sistem Performans Değerlendirme Birliği
- Bilgisayar üreticileri «temsili» programları seçer
- Standartlaşmış yöntem
  1. Çalışma zamanlarını ölçer
  2. Temel bir makineye normalize et
  3. SPECmark = normalize değerlerin geometrik ortalaması

# SPEC

- Geometrik ortalama uygun değil
- SPEC derecesi, SPEC dışı programların çalışma zamanlarına karşılık gelmez
- Subject to tinkering
  - Komite hangi programların süitin parçası olacağını belirler
  - Derleyici optimizasyonlarına odaklanır

Linear	
Reliable	
Repeatable	☺
Easy to measure	½☺
Consistent	☺
Independent	

# QUIPS

- HINT kıyas setinin (benchmark) bir parçası olarak geliştirilmiştir
- Harcanan çaba yerine çözüm kalitesini kullanır
- Kalite kesin bir matematiksel tanım
- QUIPS = Saniye başına Kalite İyileşmesi  
(*Quality Improvements Per Second*)

- HINT kıyas programı

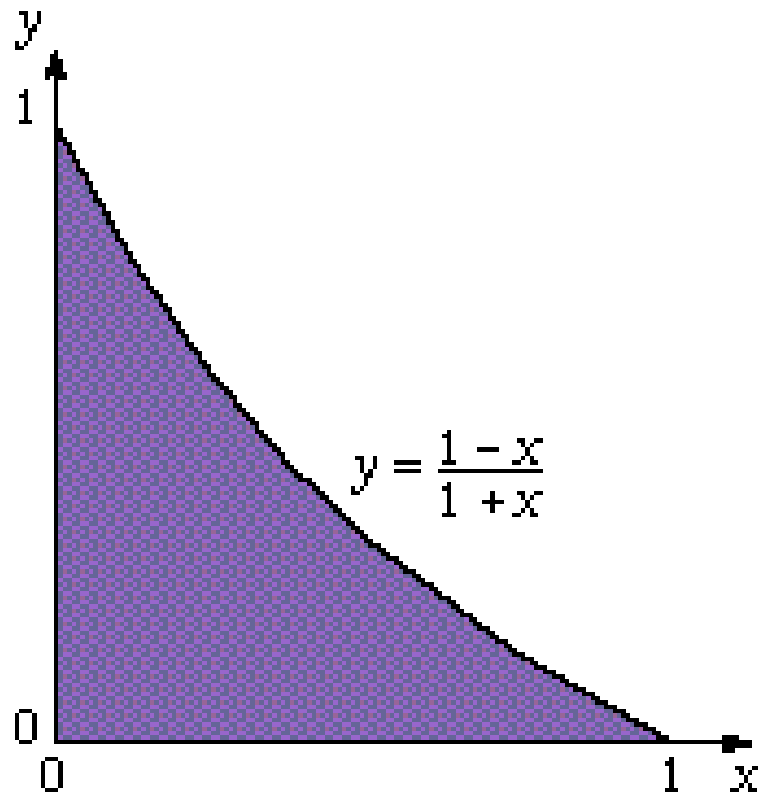
- Alt ve üst bağıl sınırları bul

$$\int_0^1 \frac{(1-x)}{(1+x)} dx$$

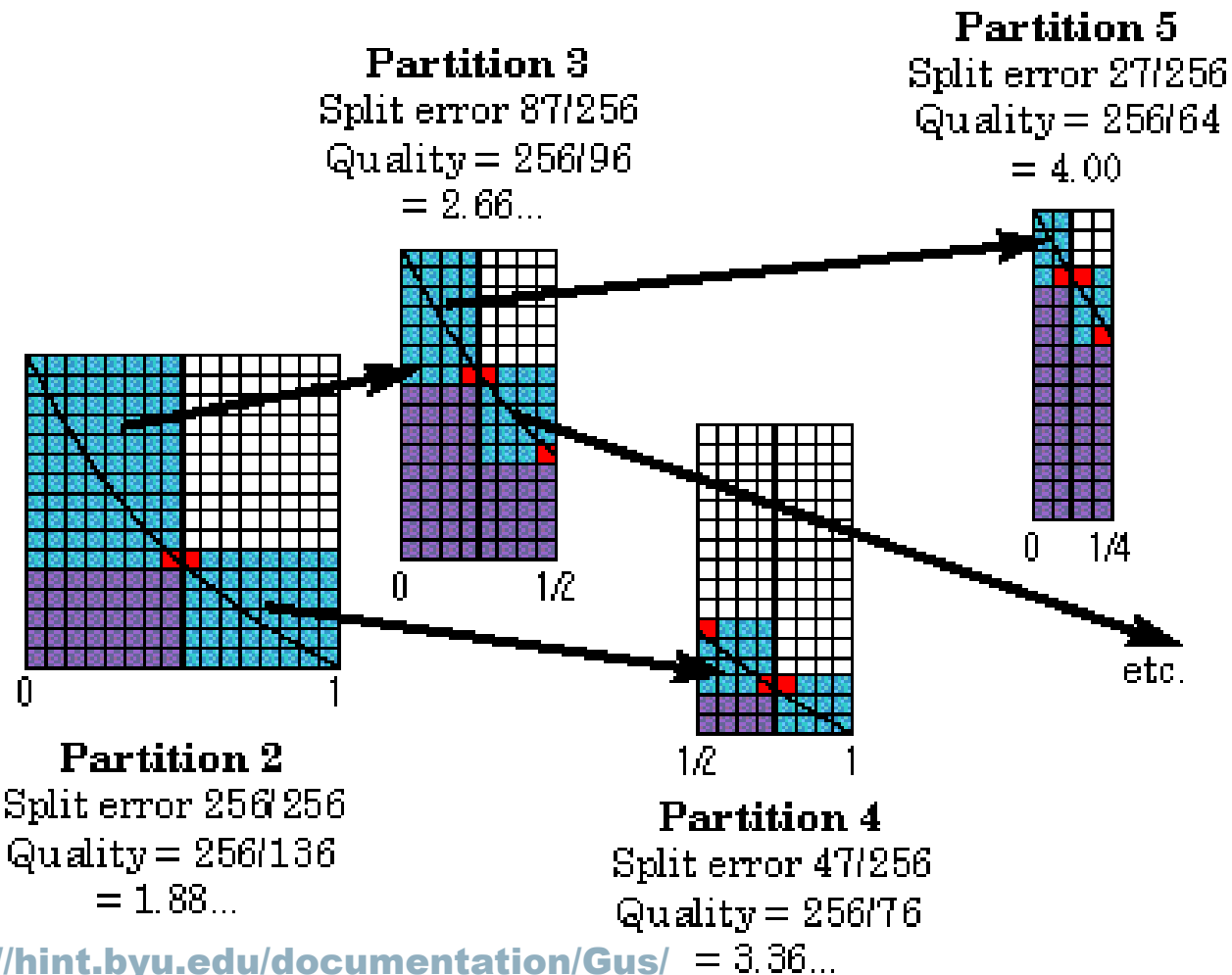
- Aralığı alt bölümlere ayırma tekniğini

- x,y yi 2'nin katları olan aralıklara böl
  - u ve l sınırları elde etmek için eğrinin tamamen üstündeki / altındaki karelerin sayısını hesapla
  - *Kalite* =  $1/(u - l)$

# QUIPS



# QUIPS



From: <http://hint.byu.edu/documentation/Gus/>

HINT/ComputerPerformance.html



# QUIPS

- Öncelik olarak
  - Kayan-nokta işlemlere
  - Bellek performansına odaklanır.
- Sayısal programların performansını hesaplamada iyi
- Ancak aşağıdakileri ihmal eder:
  - I/O performance
  - Çok-programlama
  - Talimat önbelleği
- *Kalite tanımı*

Linear	≈ 😊
Reliable	
Repeatable	😊
Easy to measure	😊
Consistent	😊
Independent	😊

# Çalışma Zamanı / yürütme süresi

- Programı yürütmek için gereken **zamanla** ilgilenir
- En küçük toplam yürütme süresi == en yüksek performans
- Süreler doğrudan karşılaştırılır
- Uygun oranla üretilir
- Zaman == temel performans metriği
  - Zamanı ölçemezseniz, hiçbir şey bilmezsiniz

# Çalışma Zamanı

- “Kronometre” ölçümü

Start\_count = read\_timer();

Ölçüm yapılacak program kısmı

Stop\_count = read\_timer();

Elapsed\_time = (stop\_count – start\_count)  
\* clock\_period;

- “Duvar Saati” ölçümü

- I/O beklemelerini kapsar, zaman-paylaşımı, OS maliyeti, ...

- “CPU zamanı” ölçümü

- Sadece işlemci zamanını kapsar

```
package com.tutorialspoint;

import java.lang.*;

public class SystemDemo {







    public static void main(String[] args) {

        // returns the current time in milliseconds
        System.out.print("Current Time in milliseconds = ");
        System.out.println(System.currentTimeMillis());
    }
}
```

Current Time in milliseconds = 1349333576093

# Çalışma Zamanı

- Hem duvar saati hem de CPU sürelerini bildirmek için en iyisidir
- Sistem gürültü etkilerini kapsar
  - Arka plan OS görevleri
  - Sanaldan fiziksel sayfa haritalama
  - Rastgele önbellek eşleme ve değiştirme
  - Değişken sistem yükü
- Ortalama ve varyansı bildirir

Linear	
Reliable	
Repeatable	$\approx$ 
Easy to measure	
Consistent	
Independent	

# Performans ölçütleri özeti

	Clock	MIPS	MFLOPS	SPEC	QUIPS	TIME
Linear					≈ 😊	😊
Reliable						≈ 😊
Repeatable	😊	😊	😊	😊	😊	😊
Easy to measure	😊	😊	😊	½ 😊	😊	😊
Consistent	😊			😊	😊	😊
Independent	😊	😊			😊	😊

# Diğer ölçütler

- Cevap Zamanı (Response time)
  - İstekten cevaba geçen süre
- Çıkış (Throughput)
  - Birim zamanda tamamlanan görev ve işlemler  
Ör. Saniye başına video çerçeveleri
- Bandgenişliği
  - Saniye başına bit
- *Ad hoc* ölçütler
  - Belirli bir ihtiyaç için tanımlanır

# Ortalama vs. Bitiş ölçütleri

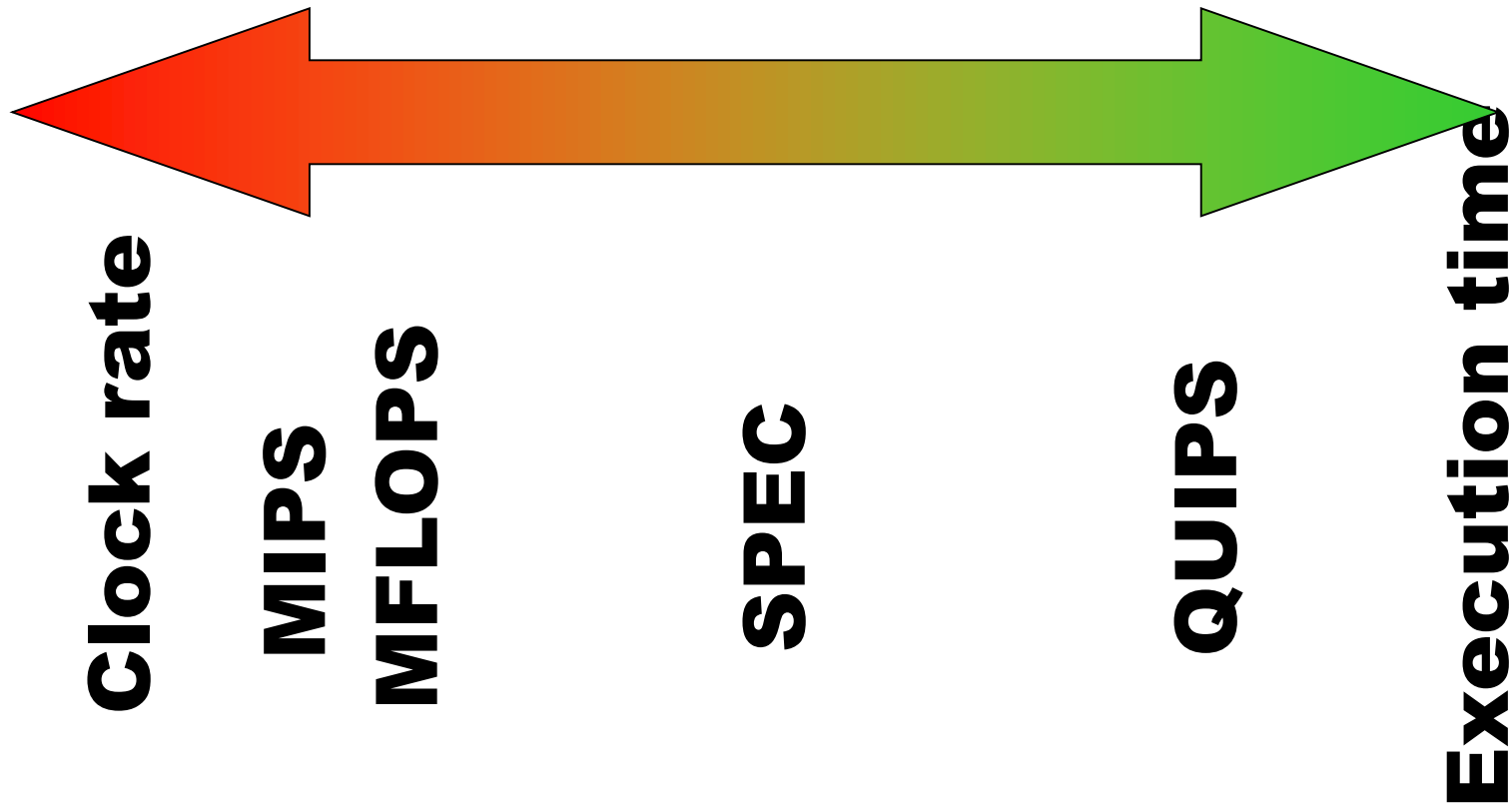
- Ortalamalara dayalı ölçütler
  - Ne yapıldığını ölçün
  - Yararlı olsun veya olmasın!
    - Talimatları say, sıfırla çarp,...
  - Güvenilmez ölçütler üretir
- Bitiş tabanlı metrikler
  - Hedefe doğru ilerlemeyi ölçer
  - Sadece gerçekte başarılmış olanları sayar



# Ortalama vs. Bitiş ölçütleri

**Means-based**

**Ends-based**



# Hızlanma(Speedup)

- “Hız” bir oran ölçütüdür
  - $R_i = D_i / T_i$
  - $D_i \sim$  “kat edilen mesafe ” by System i

# Hızlanma

$$S_{2,1} = \frac{R_2}{R_1} = \frac{D_2 / T_2}{D_1 / T_1} = \frac{D / T_2}{D / T_1}$$
$$= \frac{T_1}{T_2}$$

# Bağıl Değişim

- Yüzde değişim oranı olarak Sistem 2'nin Sistem 1'e göre performansı

$$\Delta_{2,1} = \frac{R_2 - R_1}{R_1}$$

# Bağıl Değişim

$$\begin{aligned}\Delta_{2,1} &= \frac{R_2 - R_1}{R_1} = \frac{D / T_2 - D / T_1}{D / T_1} \\ &= \frac{T_1 - T_2}{T_2} \\ &= S_{2,1} - 1\end{aligned}$$

# Bağıl Değişim

$\Delta_{2,1} > 0 \Rightarrow$  System 2 is faster than System 1

$\Delta_{2,1} < 0 \Rightarrow$  System 2 is slower than System 1

# Önemli Noktalar

## Ölçütler

- Sayılar
- Süreler
- Boyutlar
- Yukarıdakilerin kombinasyonu

olabilir

# Önemli Noktalar

- İyi ölçütler
  - Doğrusal
    - Daha doğal
  - Güvenilir
    - Karşılaştırma ve tahmin için kullanışlı
  - Tekrarlanabilir
  - Kullanımı kolay
  - Tutarlı
    - Farklı sistemlerde tanım aynıdır
  - Dış etkilere bağımsız



# Önemli Noktalar

**Not-so-good metric**

**Better metrics**



**Clock rate**

**MIPS**

**MFLOPS**

**SPEC**

**QUIPS**

**Execution time**

# Önemli Noktalar

- Hızlanma =  $T2 / T1$
- Bağlı Değişim = Hızlanma - 1