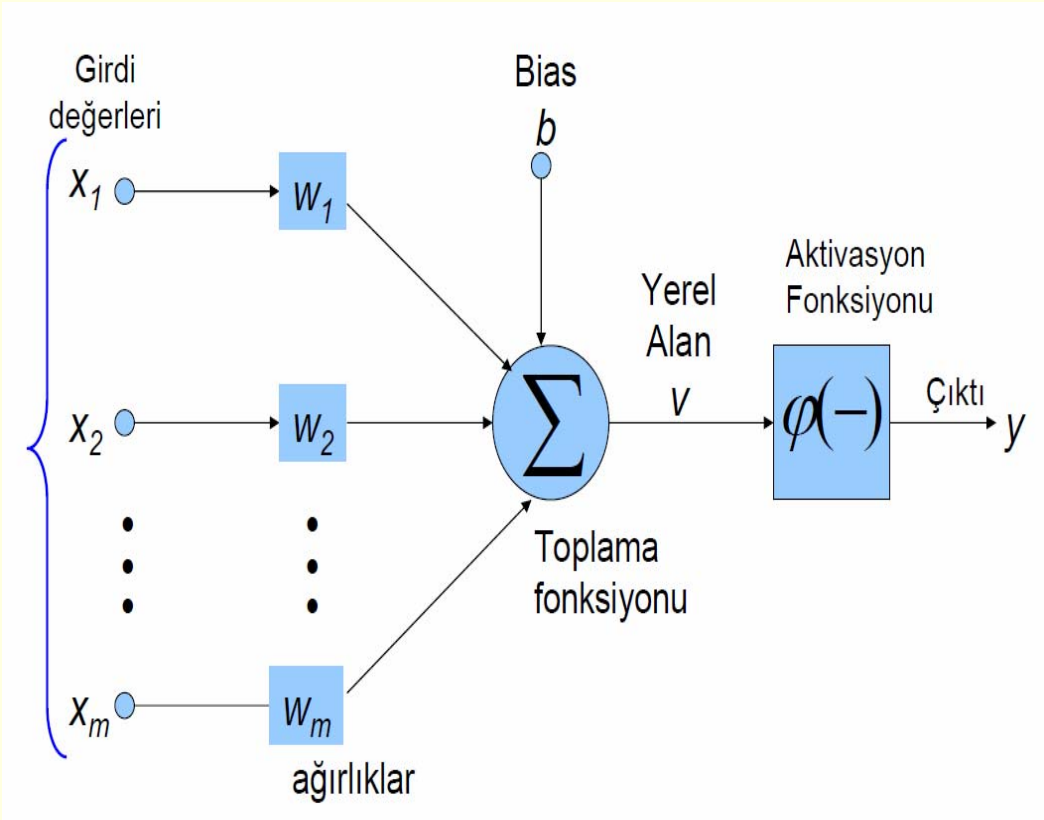
ÖZÖRGÜTLEMELİ YAPAY SİNİR AĞI MODELİ'NİN KULLANILDIĞI KUTUP DENGELEME PROBLEMİ İÇİN PARALEL HESAPLAMA TEKNİĞİ İLE BİR BAŞARIM ENİYİLEŞTİRME YÖNTEMİ

Bahadır KARASULU, Aybars UĞUR
Ege Üniversitesi,
Bilgisayar Mühendisliği Bölümü

Yol haritası

- Yapay sinir ağı ve kullanımı
- Özörgütlemeli Harita (SOM) nedir?
- Kutup dengeleme problemi ve çalışmamızdaki kullanımı
- Kullanılan paralel hesaplama tekniği ve ortam
- Başarım eniyileştirme için geliştirilen paralel hesaplama yöntemi
- Elde edilen deney sonuçları hakkında
- Çıkarımlar.

Yapay sinir ağı ve kullanımı (1/2)



Şekil 1: Yapay sinir hücresi şeması.

- YSA'lar, paralel çalışabilen birçok işlem elemanından oluşan yapılardır.
- Disiplinlerarası bir çok problemde kullanılırlar.
- Hata toleransları vardır.
- Çok Katmanlı Algılayıcı (MLP) sinir ağları, birçok sezim ve kestirim işlemlerini yürütmek için kullanılmakta olan parametrik olmayan bir yapay sinir ağı modelidir.

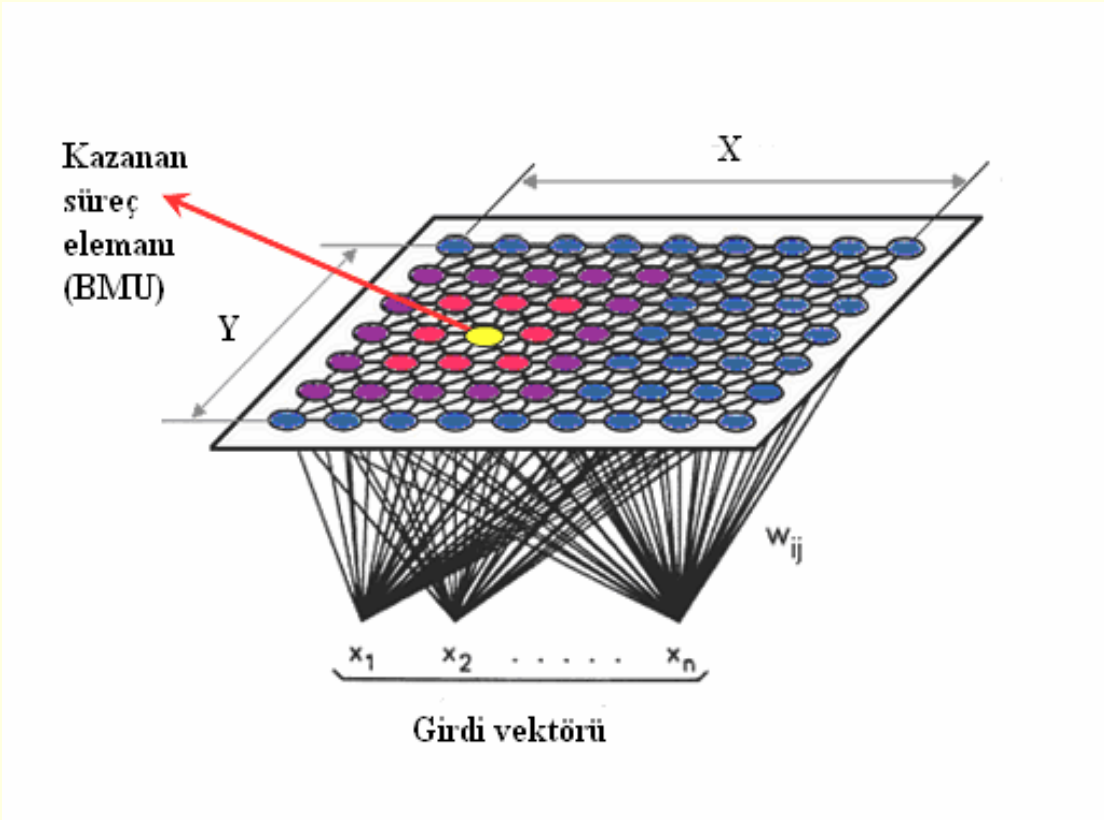
Yapay sinir ağı ve kullanımı (2/2)

- Bu çalışmada yazılım-tabanlı olarak Özörgütlemeli harita (Self-Organizing Map veya SOM) formundaki yapay sinir ağı model leri için basit bir asimetrik paralelleştirme yöntemi geliştirilerek, var olan seri biçimde tasarlanmış SOM algoritmaları/program larını hızlandıracak paralel uyarlama gerçekleştirilmeye çalışıl mıştır.
- Ele alınan kutup dengeleme probleminde seri olarak kalan kısmın haricinde **paralelleştirilebilecek olan** kısım ağın eğitimi ve komşuluk hesaplamaları göz önüne alınarak (öz olarak ağırlıkların ve yerleşimlerin ilgili süreç makineleri arasında trans ferleri ve bilgi paylaşımı) enyileştirilmiştir. Böylece elde edilen yeni algoritma sayesinde tek işlemcili seri algoritma/programın başarım oranı arttırılmıştır.
- Çalışmamızda öğrenme algoritması olarak **Destekleyici öğrenme** (reinforcement learning) yaklaşımı benimsenmiştir.

Özörgütlemeli harita (1/3)

- Kohonen Özörgütlemeli harita (SOM) topoloji-korumalı bir haritadır.
- Ana amacı, girdi uzayındaki komşuluk ilişkilerini mümkün olduğunca koruyan ve birimler arasındaki komşuluk ilişkilerine göre topoloji-korumalı bir harita yaratmaktır.
- Bir kazanan düğüm her girdi vektörü için en iyi uyumlu birim (Best Matching Unit veya BMU) şeklinde ifade edilir.

Özörgütlemeli harita (2/3)



Şekil 2: Kohonen Özörgütlemeli Haritası.

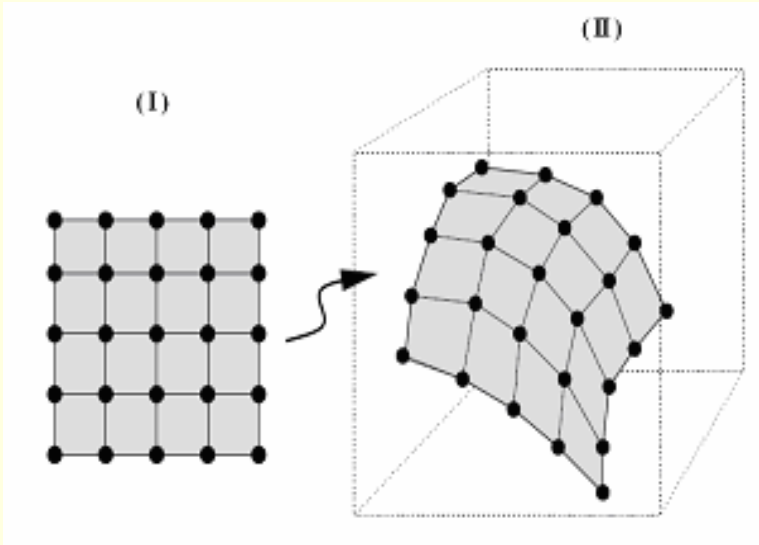
- Özörgütlemeli ağın eğitimi için her iterasyon aşağıda özetlendiği şekilde gerçekleşmektedir:

  1. Haritadaki düğümler arasından en yakın komşu (kazanan) her bir girdi örneği için bulunur.

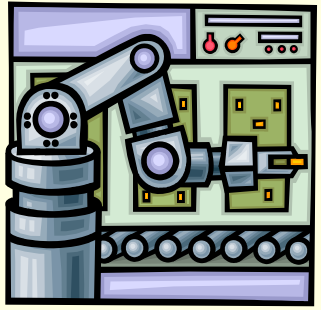
  2. Kazananın ve tüm komşularının ağırlıkları güncellenir.

Özörgütlemeli harita (3/3)

- En çok zaman harcanan kısım komşulukları bulurken geçen süredir. Komşuluk hesapları öklid mesafesi (uzaydaki iki nokta arasındaki mesafe) uyarınca hesaplanılır.
- Fiziksel uzayda iki boyutlu bir ızgara yapısı sergileyen SOM, Ağırlık/Girdi uzayında eğimli bir yapı sergilemektedir.



Şekil 3: (I) Fiziksel uzayda ve
(II) Ağırlık/Girdi uzayında Kohonen haritası.

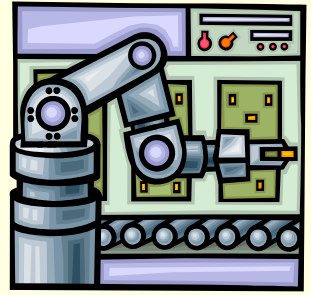


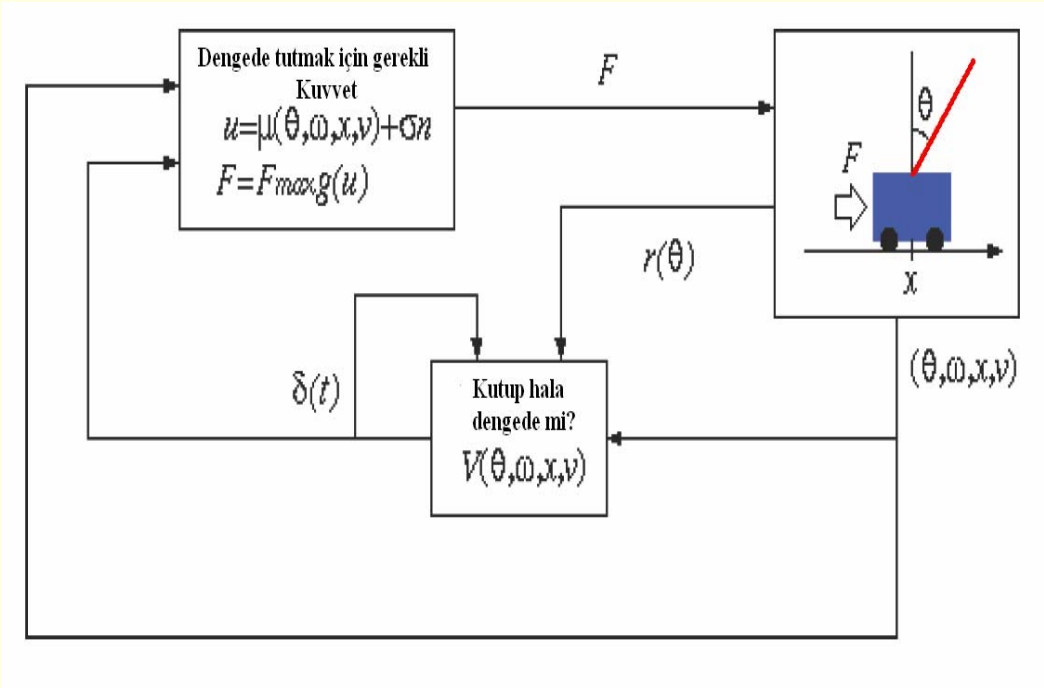
Kutup dengeleme problemi ve çalışmamızdaki kullanımı (1/2)

- Kutup dengeleme veya ters çevrilmiş sarkaç problemi uzun yıllardır yapay zeka ve yapay sinir ağları ile ilgilenen araştırmacıların (özellikle robotik) ortak bir karşılaştırma (benchmark) aracı olmuştur.

- Bu çalışmamızdaki yapay sinir ağı bir kutbu (burada örneğin 1 metre boyundaki bir çubuk gibi düşünebiliriz) temeline kuvvet uygulayarak dengede tutmayı öğrenmektedir. Kutbun davranışının Euler hareket yöntemine göre diferansiyel denklemlerin nümerik integrasyonu ile benzetimi yapılmıştır.

Kutup dengeleme problemi ve çalışmamızdaki kullanımı (2/2)





Şekil 4: Kutbu dengede tutmaya çalışan yapay sinir ağının çalışma şeması.

Euler Hareket denklemleri:

$$\ddot{x} = \frac{\left(F - M_p L\left(\dot{\omega}^2 \sin\omega - \ddot{\omega}\cos\omega\right)\right)}{M_c + M_p}$$

$$\ddot{\omega} = \frac{\left(G\sin\omega + \cos\omega\left(\frac{\left(-F - M_p L\left(\dot{\omega}^2 \sin\omega - \ddot{\omega}\cos\omega\right)\right)}{M_c + M_p}\right)\right)}{L\left(\frac{4}{3} - \frac{\left(M_p \cos^2\omega\right)}{M_c + M_p}\right)}$$

Problemde kullanılan parametreler

• Çalışmamızda sistem çalıştırılarak öğrenmeye bırakıldıktan sonra *Kutup Benzetimi* sırasındaki adımlar hem gerçek çalışma süresi tutularak hem de benzetim adımı için verilen zaman adımları olan 0.1 saniye aralıklarla ölçülmüştür.

• Kutup Benzetimi sırasında ilgili birimin girdi vektörüne olan benzerliği bir puanlama sistemiyle kontrol edilmektedir. Buna kutup puanı adı verilmektedir.

Paralel SOM Kutup Dengeleme programında kullanılan parametreler.

Parametre ismi:	**Açıklama:**	**Örnek:**
L	Kutbun (çubuğun) uzunluğu	1.0 metre
G	Yer çekimi ivmesi	9.81 m/sn2
Mc	Plakanın (cart) kütlesi	2.0 kg
Mp	Kutbun (pole) kütlesi	1.0 kg
F	Uygulanan kuvvet	10 Newton
$\ddot{x}$	İvme	m/sn^2
$\ddot{\omega}$	Açısal ivme	rad/sn^2

Kullanılan paralel hesaplama tekniği ve ortam

- Seri hesaptaki verim arttırılmak istenmektedir.
- Bunun için paralel bir algoritma geliştirilmiştir.
- Bu algoritmayla dayanılarak üretilen paralel program MPI (Mesaj Geçme Arayüzü) kütüphanesi kullanılarak çalıştırılmıştır.
- Temel olarak LAM-MPI ve ANSI (GNU) C dili kullanılmıştır.
- Özdeş makineler (her biri 256 MB RAM ‘e sahip 4 adet Intel Celeron mimarili makine) kullanılarak test yapılmıştır.
- Deneyler sırasında çeşitli büyüklüklerde (deneyde 25*25, 125*125, 250*250, 500*500 büyüklüğünde) YSA’lar denenerek, sistemin gösterdiği tepkiler ve algoritmanın güvenirliği sınanmıştır.

Başarım eniyileştirme için geliştirilen paralel hesaplama yöntemi - Algoritma

1. Başla,
2. İlgili dizi ve değişkenleri tanımla,
3. Gerekli dizi elemanları ve/veya değişkenlerin değerlerini sıfırla,
4. MPI'ı başlat,
5. İşlemci sayısı ve kimliklerini tespit et,
6. Euler Hareket denklemlerinde kullanılacak ilgili parametreleri dosyalardan oku,
7. Hesaplamada geçen sürenin tespiti için **saat tutmaya başla,**
8. **AG** isimli yapı bloğundan yeni bir **DenekAgi** isimli Kohonen SOM ağı oluştur,
9. SansSayilariniOlustur() ve RasgeleAgirliklar() isimli prosedürler ile rasgele sayı ureteci ile rasgele agirliklari olustur,
10. Ağın eğitimini başlat,
11. Ağın eğitimi sırasında ilgili girdi ve çıktı verilerini ve ağırlıkları işlemciler üzerine MPI_Send() fonksiyonu ile dağıt,
12. Ağın eğitimi boyunca ilgili hesaplamalar işlemcilere eşit miktarda (**Formül = [Toplam YSA birimi (düğüm sayısı) / toplam işlemci sayısı * (geçerli işlemcinin sırası +1)]**) dağıtılması yoluyla uygun hesaplamayı yaptır,
13. Elde edilen yerel sonuçları MPI_Recv() fonksiyonu ile işlemci sırası (rank) sıfır olan (yani ilk makine veya yönetici makine olarak adlandırılır) üzerinde toparla,
14. Adım 11 ile 13 arasındaki algoritma adımlarını eğitim adımları bitene kadar tekrarla,
15. Her adımda elde edilmiş olan Simulasyon zaman adımı değeri (0.1 saniyelik adımlar) , o adımda kutup çubuğunun y ekseni ile arasındaki açı (kutbun hala dengeli olup olmadığının tespiti için) ve uygulanan optimum kuvveti içeren 3 kolonluk bilgiyi ilgili dosyaya yazdır,
16. Saat tutmayı bitir.
17. Ağın en son halini ilgili dosyaya yazdırarak hesap kısmını sonlandır.
18. MPI'ı bitir.
19. Dur.

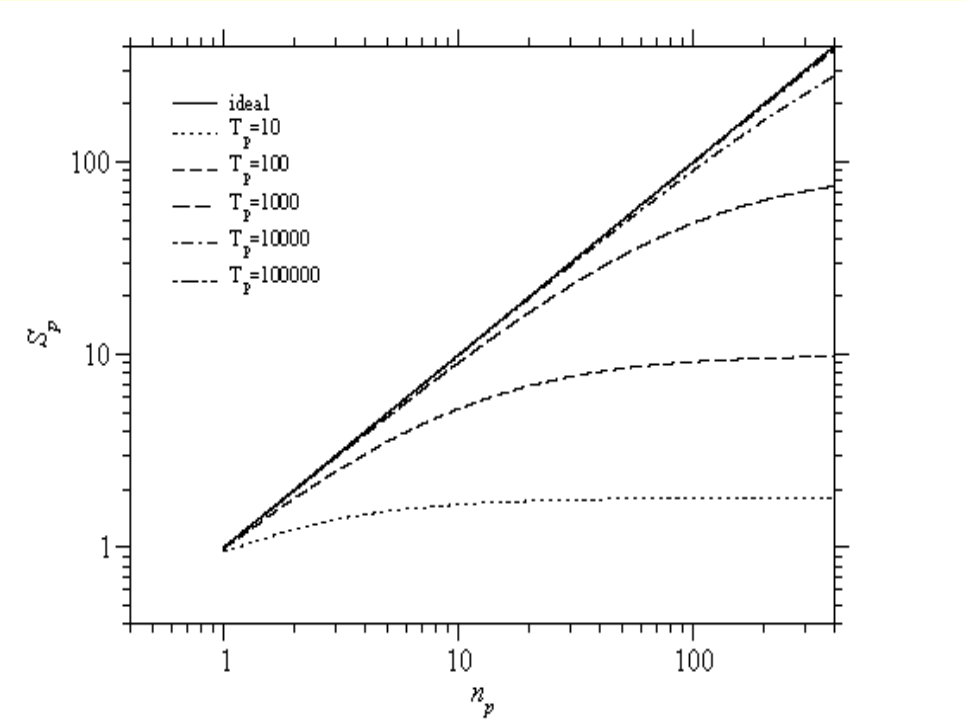
Elde edilen deney sonuçları (1/3)

Kullanılan formüller:

Hızlanma (Speedup) formülü:

$$S_p(n_p) = \frac{T(1)}{T(n_p)}$$

n_p: İşlemci Sayısı



Şekil 5: Amdahl Kanunu grafiği. Burada $T(n_p)$ = 10, 100, 1000, 10000 ve 100000 için hızlanmalar görülmektedir.

Verim formülü:

$$\varepsilon = \frac{T(1)}{n_p T(n_p)}$$

Elde edilen deney sonuçları (2/3)

Kullanılan parametreler:

Açıklama:	Ağ 25*25	Ağ 125*125	Ağ 250*250	Ağ 500*500
SOM Satır Sayısı	25	125	250	500
SOM Sutun Sayısı	25	125	250	500
Eğitim adımları	100000	100000	100000	100000
Dengeleme Süresi (sn)	240	240	240	240
Benzetim AdımSayısı (Iterasyon)	1000	1000	1000	1000
Kutbun DengeAçısı (y_ekseni ile)	75	75	75	75
Kutup Çubuğu uzunluğu (metre)	3.0	3.0	3.0	3.0
Plakanın Ağırlığı (kg)	2.0	2.0	2.0	2.0
Kutbun Ağırlığı (kg)	1.0	1.0	1.0	1.0
Yerçekimi	9.81	9.81	9.81	9.81
Benzetim Zaman Adımları (saniye)	0.1	0.1	0.1	0.1

Elde edilen deney sonuçları (3/3)

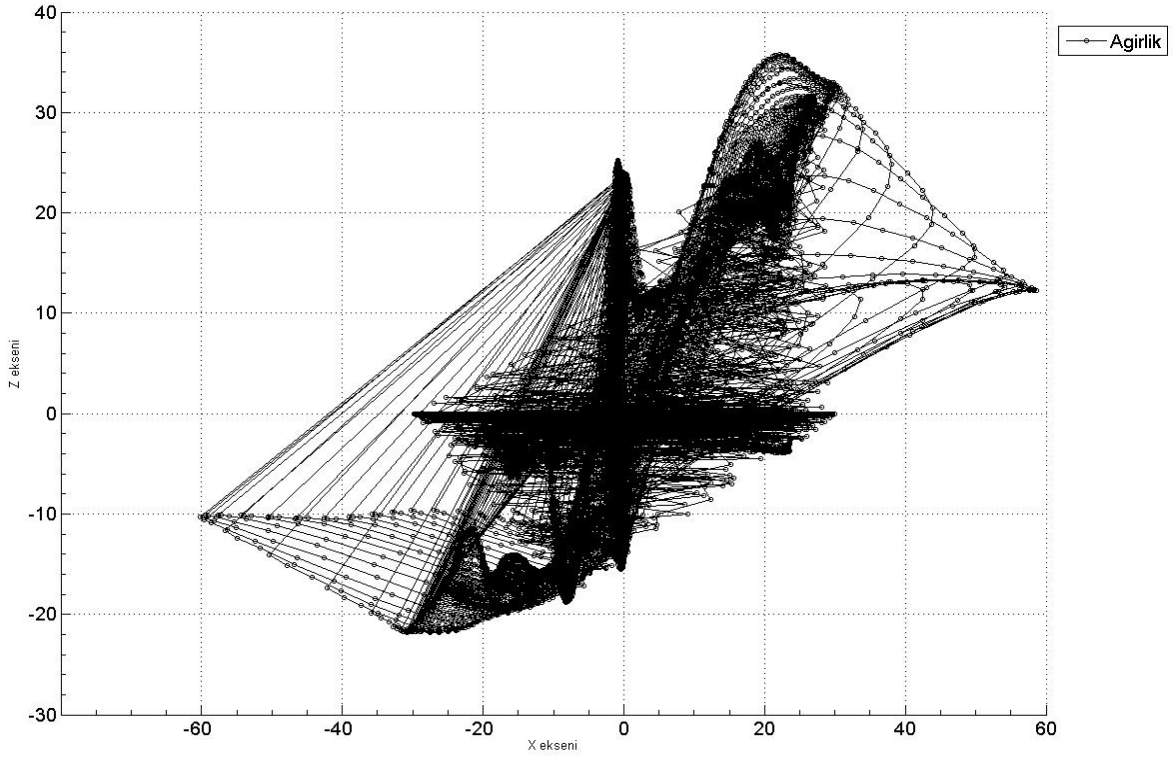
Elde edilen değerler:

Simülasyonun Tek ve Çok-işlemcili sistemde çalıştırılması sonucu elde edilen geçen süre değerleri, hızlanma ve verimler.

Açıklama:	İşlemci Sayısı (n_p)	Ağ (25* 25)	Ağ (125* 125)	Ağ (250* 250)	Ağ (500* 500)
Geçen Süre (sn)	1	215	4545	17254	72543
	2	245	5370	18816	83732
	3	271	5564	19507	80767
	4	290	5337	19658	78640
Hızlanma Oranı	1	1	1	1	1
	2	0.8775	0.8463	0.9169	0.8663
	3	0.7962	0.8168	0.9053	0.8981
	4	0.7413	0.8516	0.8777	**0.9224**
Verimlilik (% olarak)	1	100	100	100	100
	2	43	42	45	43
	3	26	27	30	29
	4	18	21	21	**23**

Ağ büyütülürse veya işlemci sayısı daha da arttırılırsa bu değerler de artış olacaktır.

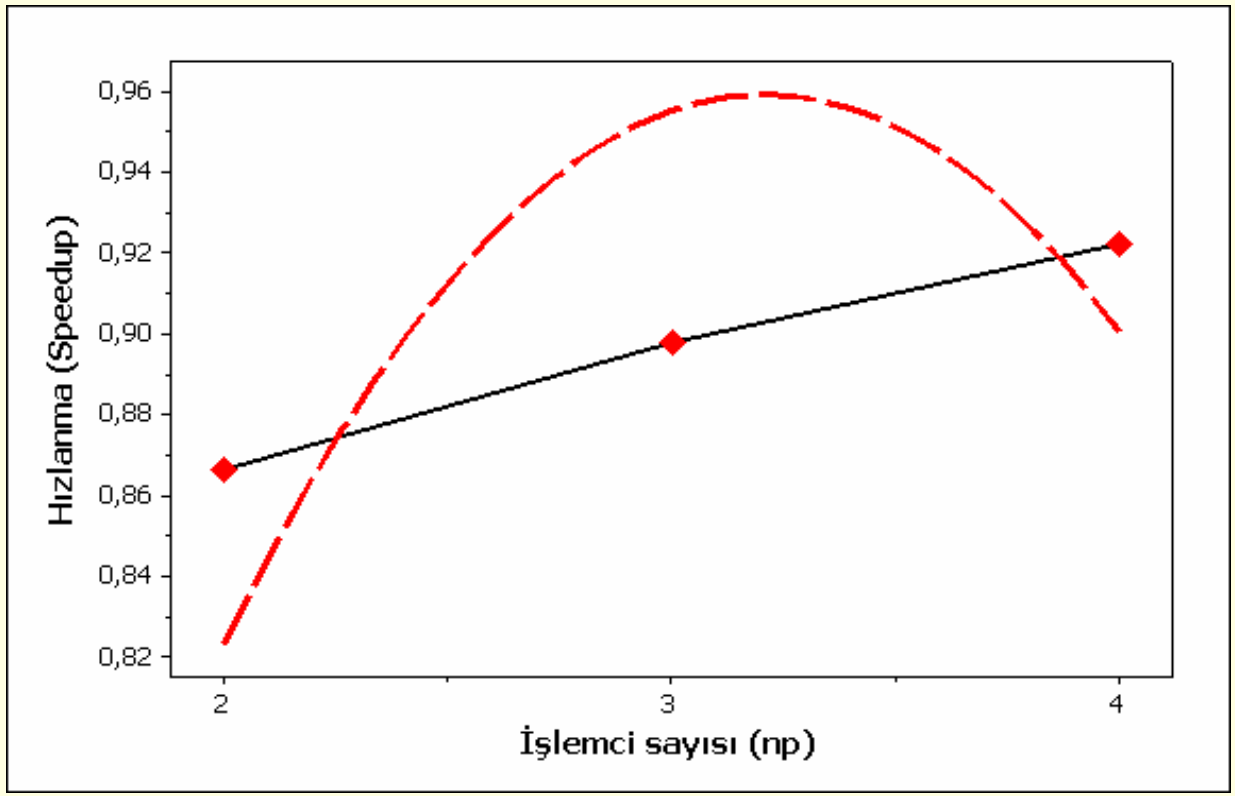
Sonuçların tutarlılığı (1/2)



Şekil 6: 125*125'lik Ağ'a ait ağırlıkların X ve Z ekseni düzleminde gösterimi (bu grafiğe ağırlık/girdi uzayında Kohonen SOM haritası da denilmektedir).

1. Deney sonucunda beklenildiği gibi Kohonen Ağırlık/Girdi uzayındaki gösterimi eğimli bir yapıyı sergilemektedir.
2. Grafiğin orta noktasındaki yoğunlaşmış (koyu) bölge ise sistemin öğrenme sırasında çubuğu dengede tutmak için uygulanacak optimal kuvveti denediği durumların en çok aynı bölgede yoğunlaşmasından kaynaklanmaktadır.

Sonuçların tutarlılığı (2/2)



Şekil 7: 500*500'lik Ağ'a ait hızlanma oranı grafiği.
(Grafikteki düz çizgi ile gösterilen eğri gerçek hızlanma değerlerini, kesikli çizgilerle gösterilen eğri ise 'eğri uydurma' ile elde edilen hızlanma grafiğini gösterir).

1. Grafikteki 'eğri uydurma' (curve fitting) ile elde edilen kırmızı kesikli çizgi Amdahl kanunu'na tutarlılık sergilemektedir.
2. Ağın büyüklüğü arttırıldıkça beklenen değerlere yaklaşım gözlenmektedir.
3. Verim, daha çok sayıda işlemci kullanılarak da artırılabilir.

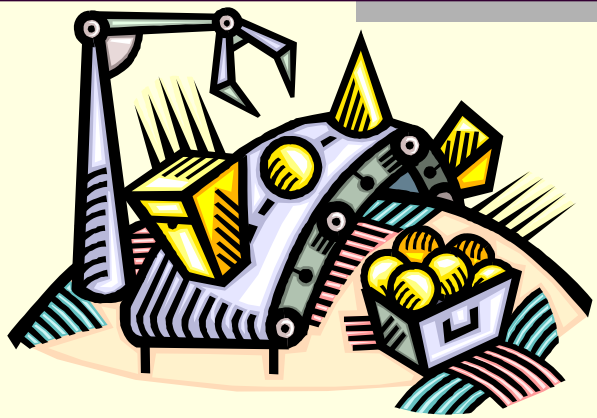
Çıkarımlar

- Çalışmada seri hesabın yanı sıra paralel hesap yapılması sonucu belirli bir hızlanma elde edilmiştir. Fakat bu hızlanma istenilen düzeyde olmamıştır. Bunun başlıca nedenleri problemin doğası gereği fazlaca paralelleştirmeye uygun olmamasıdır. Tepe performansı 3 makinenin aynı anda çalıştığı durumda hızlanma değeri olarak 0.9 civarında elde edilmiştir.

- Eğer paralelleştirme yüzdesi artırılamazsa problemin çözüm etkinliği artırılamayacak ve bir ilerleme kaydedilemeyecektir. Bunun için temel öneri olarak **SOM** ve **LVQ** (Learning Vector Quantization) tarzı ağ yapılarının kullanıldığı problemlerde Kohonen katmanı için ayrıca bir paralel hesaplama tekniği geliştirilmesi, girdi ve çıktı katmanlarının farklı bir yaklaşımla ele alınması daha uygun olacaktır.

Teşekkürler

- Sorularınız ???



- İletişim:

bahadir.karasulu@ege.edu.tr
aybars.ugur@ege.edu.tr
Ege Üniversitesi, Bilgisayar Mühendisliği.