HESAPLANABİLİRLİK

Urfat G. NURİYEV

Ege Üniversitesi, Fen Fakültesi, Matematik Bölümü, urfat@sci.ege.edu.tr

Halide G. SADIGOVA

Azerbaycan Bilimler Akademisi, Sibernetik Enstitüsü, F. Agaev 9, 370141, Bakü, AZERBAYCAN

"Gerçek tehlike bilgisayarın insan gibi düşünmeye başlaması değil, insanların bilgisayar gibi düşünmeye başlamasıdır."

Sydney J. Harris

1. Giriş

Bilim, yeni kavramların bulunmasıyla ilerler. Bazen yeni bir kavramın bir süre sonra doğal olmadığı ya da işlerliği bulunmadığı anlaşılır. Diğer biri ise beklenenden daha da önemli ya da yararlı çıkabilir. Çağdaş bilimin başarılı kavramlarından hesaplanabilirlik kuramı diğer işlevlerin yanısıra karmaşıklık konusuna da ışık tutmuştur.

Bu yazıda hesaplanabilirlik kuramının gelişim tarihinden, algoritmalardan, Turing makinesi, P, NP ve NP-tam sınıf kavramları ve bu alandaki son gelişmelere yer verilmiştir.

2. Hilbert'in Matematik Programı

19. yüzyılın sonlarında matematikçiler, matematiksel kanıtlama yöntemlerinin giderek daha da güçlenmesinin etkisiyle büyük gelişmeler kaydetmişlerdir. Böylece matematikçiler bu güçlü yöntemleri giderek artan bir güvenle uygulamaya başlamışlardı. Ancak bu güven 1902'de İngiliz mantık bilimcisi ve felsefecisi Bertrand Russell'in ünlü paradoksunu ortaya atmasıyla sarsıldı. O zamanlarda matematikçiler aksiyomlardan ve yöntemsel kurallardan oluşan son derece biçimsel bir matematik sistemini geliştirmeye girişti. Büyük Alman matematikçisi David Hilbert daha uygulanabilir ve kapsamlı bir projeye başladı. Hilbert programının amacı matematiğin herhangi bir iyi tanımlanmış alan ile ilgili her çeşit doğru matematiksel uslamlama yöntemini içine alacak kadar geniş kapsamlı ve yöntemsel kurallar listesi yaratmaktı. Ayrıca Hilbert projenin çelişkiden uzak olduğunu kanıtlamanın da mümkün olacağını düşünüyordu. Böylece matematikçiler sonsuza kadar, sarsılmaz bir temel üzerine oturabileceklerdi. Ancak onların umutları 1931'de, 25 yaşlarında zeki bir Avusturyalı matematik mantıkçısı olan Kurt Gödel'in, Hilbert'in programını altüst eden teoremiyle söndü.

Gödel Teoremi: Aksiyomlardan ve çıkarım kurallarından oluşan herhangi bir kesin ("biçimsel") matematik sistemi, basit aritmetik teoremlerinin tanımlamalarını kapsayacak kadar geniş kapsamlı olması ve çelişkisiz olması koşuluyla, sistemin kapsamına alınan yöntemlerle ne kanıtlanabilir ne de kanıtlanamaz bazı önermeleri içermelidir. Buna göre bu gibi önermelerin doğruluğu hakkında, onaylı yöntemlerle "karar verilemez". Gödel, gerçekte, uygun bir aritmetik teoremi şeklinde kodlandığında aksiyom sisteminin tutarlılığının bildirimini "karar verilmez" yöntem olduğunu kendiliğinden kanıtladığını göstermiştir.

3. Turing Makinesinin meydana gelişi

Bir Turing makinesi hakkında aklımızda tutulacak en önemli nokta bunun bir fiziksel nesne değil bir "Soyut Matematik" ürünü olduğudur. Bu kavram ilk kez, İngiliz matematikçisi, ünlü şifre uzmanı ve ilk bilgisayar bilimcilerinden Alan Turing tarafından 1935-36 yıllarında daha geniş kapsamlı bir probleme yanıt getirebilmek amacıyla ortaya konmuştur.

Entscheidungs problemi adıyla bilinen bu problem Hilbert tarafından 1900 Paris Uluslararası Matematikçiler kongresinde tanımlanmıştı. Turing'in ilgilenmiş olduğu Hilbert problemi Matematiğin

herhangi bir aksiyom sistemi cinsinden yazılmasından daha derine inen bir problemdir. Sorun şudur: *Matematiğin tüm problemlerini birbiri peşine çözebilecek genel bir mekanik yöntem ilke olarak var mıdır?* Turing bu soruyu *m sayısına uygulanan Turing makinesinin gerçekten DUR konumuna geçip geçmeyeceğine karar verme problemi* olarak yorumlamıştır. Bu problem DURMA problemi adıyla anılır.

Turing Gödel'in yapıtını inceledikten sonra DURMA probleminin çözülemeyeceğine dair teoremini geliştirmiştir.

4. Algoritmalar teorisi

Algoritma kavramı matematiğin en temel kavramlarından biridir. Algoritmalar ilk çağdan itibaren bilinip kullanılmakla beraber yirminci asırda gelişmiştir.

Algoritmalar ard arda gelen sonlu işlemlerle problem çözme yollarıdır. Bir *problemi* yanıt isteyen genel bir soru olarak kabul edebiliriz. *Soru* tanımı içinde (i) ilişkin tüm parametrelerin belirlenmesi ve (ii) yanıtın(çözümünün) taşıması gereken tüm özellikler olmalıdır. Parametrelere özgül değerler atandığında problemin bir *örneğini (bireysel problem)* elde ederiz.

Örnek-1: Genel bir 0-1 değişkenli denklem aşağıdaki gibidir:

$$\sum_{i=1}^{n} a_i x_i = b, \quad a_i, b \in \mathrm{N}, \quad x_i = 0 \; ya \; da \; 1, \quad i = \overline{1,n}.$$

Buna karşılık, bir 0-1 değişkenli denklem örneği şöyle verilebilir:

$$4x_1 + 3x_2 + 2x_3 + 5x_4 = 8, \qquad x_1, x_2, x_3, x_4 = 0 \; ya \; da \; 1.$$

$$x_1 = 0, \quad x_2 = 1, \quad x_3 = 0, \quad x_4 = 1$$

bu problemin çözümüdür.

Herhangi bir algoritmanın bir P problemini çözdüğünü söyleyebilmemiz için bu algoritma P probleminin örneklerinin hepsi için çözümü daima verebilmelidir.

Teoride esas olan algoritmik olarak çözülebilme veya çözülmeme durumudur. Belirsiz algoritma kavramından kesin Turing makinası kavramına geçilip verilen bir problem sınıfının bir algoritma ile çözülüp çözülemeyeceği sorusu kesinleştirilmiştir.

Turing makinasında belli bir sayıda "çalışma" ile bir "dinlenme" durumundan oluşan sonlu sayıda bir *içdurum* bulunmaktadır. Makina karelere bölünmüş sonsuz uzunlukta bir kağıt şerit üzerinde çalışır(şerit bellek görevini yapmaktadır). Şeridin her karesinde özel simgeler bulunmakta olup bu simgelerden bir boş anlamını taşır. Çalışma durumlarından birinde olduğu zaman üzerinde bulunduğu kareyi okur ve bir yandan hangi iç durumunda olduğuna ve diğer yandan da okuduğu simgeye bağlı olarak aşağıdaki işlemleri yapar: (i) Karede yazılı olanı siler ve başka bir simge (ya da yine aynı simgeyi) yazar. (ii) Bir kare ileri ya da geri gider. (iii)Yeni bir iç durumuna geçer.

Kısacası bir Turing makinesi normal bir bilgisayarın yaptığı herhangi bir işi yapabilir.

5. Hesaplanabilirlik

Hesaplanabilirlik gerçek bir "mutlak" matematik kavramıdır. İlk kez 1930'larda bu kavram gibi temel nitelikli diğer kavramlarla birlikte matematik bilimine girdiği için aynı zamanda oldukça yeni bir fikirdir ve matematiğin tüm alanlarını kapsar.

Eski sayılar ve kesirler, pay ve paydalar Turing makinelerince ne boyutta olursa olsunlar, hesaplanabilirler. π, $\sqrt{2}$ gibi diğer pek çok irrasyonel sayı Turing makinesi ile üretilebilirler. Turing makinesinin üretemeyeceği bazı irrasyonel sayılar da vardır. Turing makinesince üretilebilen sayılara *hesaplanabilir sayılar* adı verilirken, üretilemeyen sayılar (büyük çoğunluğu oluşturan sayılar!) hesaplanamaz sayılar olarak adlandırılır.

Hesaplanabilirlik fikrini açıklayan başka yollar da vardır. Bunlardan, tarih bakımından birincisi, Amerikalı mantıkçı Alonza Church'ün, Stephen C. Kleene'in yardımıyla geliştirdiği "Lambda Hesabı"dır. Church'ün yöntemi Turing'in yönteminden oldukça farklı ve daha soyuttur.

Hesaplanabilirlik, genellikle matematikte önemli bir konudur ve sadece sayılara uygulanabilir bir konu olarak değerlendirilmemelidir. Hesaplanabilirlik fikrinin gücü kısmen bazı iyi tanımlanmış matematik işlemlerinin aslında hesaplanamaz olmasından kaynaklanır. Çünkü, böyle hesaplanamaz işlemler olmasaydı, hesaplanabilirlik kavramı matematiğin ilgisini çekmezdi.

6. Algoritmik çözülmeyen problemler

Algoritmik çözülmezlik ilk olarak matematiksel mantıktaki problemler (sonuç çıkarabilme problemi) ve algoritmalar teorisindeki problemler (örneğin, kendi kendine hesap edilebilme problemi) için düşünülmüştü. Daha sonra ise matematiğin başka birçok dallarındaki benzer fakat daha az genel problemlere yöneltildi. Bunlardan biri de Hilbert'in onuncu problemidir: *Verilen her hangi bir Diophant denkleminin tamsayılı çözümünün olup olmadığının bulunması.* 10. problem genel halde uzun süre çözülemedi ve 1970 yılında Y.V.Matiyosevich bu problemin çözülemez olduğunu ispatladı. Aşağıdaki 3 problem de çözümsüzlüğü ispatlanmış problemlerdendir:

1. Bir açının pergel ve cetvelle 3 eşit parçaya bölünmesi.
2. Sadece pergel ve cetvel yardımıyla, alanı verilmiş bir dairenin alanına eşit olan karenin çizilmesi.
3. Yalnız pergel ve cetvelle, verilmiş olan bir küpün 2 katı hacminde bir küpün 1 ayrıtın çizilmesi.

7. Algoritmik karmaşıklık

Matematikteki formalizm algoritmik olarak çözülemeyen problemlerin araştırılmasını gündeme getirdi. Bilgisayar teknolojisindeki gelişmeler ise, başka tür problemlerin, ilke olarak değil, sadece uygulamada zor olan problemlerin öğrenilmesini öne çıkardı. Bu problemlerin çoğunu tam çözebilmek için bilgisayarları yüzlerce yıl çalıştırmak gerekebilir.

Örnek-2: Örnek-1'deki problemi $n = 50$ için saniyede 1.000.000 işlem yapan bir bilgisayarda sayımlama yöntemi ile çözmesi için gereken zamanı hesaplayalım.

$$\sum_{i=1}^{50} a_i x_i = b, \quad a_i, b \in \mathbf{N}, \quad x_i = 0 \; ya \; da \; 1, \quad i = \overline{1,50}.$$

Burada $x_i = 0$ *ya da* 1 olmak üzere $(x_1, \cdots, x_{50})$ vektörlerini incelemek gerekecek ki, bunların da sayısı $N = 2^{50}$ olup,

$$N = 2^{50} = \left(2^{10}\right)^5 = (1024)^5 > (1000)^5 = \left(10^3\right)^5 = 10^{15}$$

işlem yapılacaktır. Bilgisayarımızın hızı $V = 10^6$ işlem / saniye olduğu için $N = 10^{15}$ işlem yapmak için gereken zaman $t = \frac{N}{V} = \frac{10^{15}}{10^6} = 10^9 \; san \; \approx \; 25 \, yıl$. Önceleri bu tür problemlerin çözüm tekniklerinin bilgisayar teknolojisindeki hızlı gelişmeler sayesinde yeterli gelecekleri düşüncesi sonradan bu iyimserliğe gölge düşürecek kuşkular ortaya çıkardı. Bu kuşkular problemlerin çözüm karmaşıklığı ile ilgilidir.

8. P ve NP sınıf

Genel olarak algoritmaları çözüm karmaşıklığı açısından iki sınıfta toplamak mümkündür: (i) Verimli (efficient) algoritmalar. (ii) Üssel zamanlı algoritmalar.

Verimli algoritmalar problemleri polinom zamanda çözebilen algoritmalardır. Diğer bir deyişle verimli algoritmalarda çözüm için gerekli işlem sayısı problemin karakteristik bir boyutu cinsinden bir polinom ile ifade edilebilir. Öte yandan işlem sayısı problemin karakteristik boyutunu üssel bir işlevi olarak ifade edilen algoritmalar ise *üssel zamanlı algoritmalar* olarak sınıflandırılmaktadır.

Örnek-1'deki problemin karakteristik boyutu değişkenlerin sayısı olan n'dir. Genel olarak *problemlerin karakteristik boyutuna* ilişkin verileri bilgisayarın belleğinde yazmak için gereken yerin büyüklüğü ile tanımlanabilir. Algoritmaların problemleri çözmede alacağı zaman bu büyüklük parametrelerinin bir işlevi olarak gösterilir. Bir algoritmanın, büyüklüğü verilen bir problem için maksimum kaç iterasyon süreceği ve her iterasyondaki toplam aritmetik işlem sayısı hesaplanabilir. Aritmetik işlemleri, toplama, çıkarma, çarpma, bölme ve karşılaştırma gibi basit işlemlerin bilgisayarda sabit zaman aldığını varsayarak algoritmanın alacağı zamanı yalnızca büyüklük parametreleri cinsinden yazabiliriz. Bu işlem 1. Örnekteki problem için $T(n)$ olarak yazıldığında sayımlama algoritmasının *süresel karmaşıklığı* $O\left(2^n\right)$ olur, yani C sabit bir sayı olmak üzere $T(n) < C \cdot 2^n$'dir. İlk olarak J.Edmonds tarafından ortaya atılan *iyi - kötü* algoritma ayrımının testi işte bu süresel karmaşıklığa dayandırılmaktadır.

Bir algoritma ancak eğer süresel karmaşıklığı büyüklük parametrelerinin bir polinomu ise iyi kabul edilmektedir. Örneğin $O(n)$, $O\left(n^2\right)$, $O\left(ln + m^3\right)$, $O\left(n^3 + m^3 + 5\right)$, vb. polinom sınırlı süresel karmaşıklıkları ifade ederler, $O\left(6^n\right)$, $O\left(n!m!\right)$, $O\left(n^m\right)$, ise polinom sınırlı olmayan süre karmaşıklıklarının, dolayısıyla kötü algoritmaları temsil eder. Polinom zamanlı çözer yöntemi bulunmuş problem sınıfı P ile gösterilir. Problemler sınıfını formal olarak tanımlamak için *gerekirci (deterministik) (DTM)* ve *gerekirci olmayan (nondeterministik) Turing makinesi (NTM)* kavramları kullanılır. Burada DTM'yi bildiğimiz bilgisayarlara eş bir makine, NTM'yi ise henüz eşi dünyada olmayan üstün bir makine olarak kabul edebiliriz. Bu öyle bir üstün makenedir ki, bir anda sonlu bir k kadar çözüm tahminlerinde bulunup her tahmin için ayrı bir bilgisayar gibi hesaplamalar yapabilir.

Polinom sınırlı DTM ile çözülebilen tüm karar problemleri P *sınıfını* oluşturur. Polinom sınırlı NTM ile çözülebilen karar problemleri ise *NP sınıfındandır*. P, NP'nin bir alt kümesidir. Çünkü NTM'ler polinom sınırlı DTM ile çözülebilen her problemi çözebilir. Asıl sorun yalnız polinom sınırlı NTM ile çözülebilen karar problemleminin DTM ile polinom sınırlı sürede çözülüp çözülemeyeceğidir. Yani $P \stackrel{?}{=} NP$ sorusunun yanıtı Kapmaşıklık kuramının çözülmemiş en önemli problemidir.

9. NP-tamlık

Bir X problemi eğer, Y problemini çözen iyi bir algoritmadan yararlanılarak elde edilen diğer iyi bir algoritma ile çözülebiliyorsa, X, Y'ye (polinom) *indirgenebilir* denir.

Örnek-3: X ve Y problemi aşağıdaki gibi olsun:

$$Y:\ ax^2 + bx + c = 0, \qquad X:\ dx + q = 0.$$

X problemi Y probleminin özel durumu olduğu için Y problemini çözen algoritmadan yararlanarak X'i de çözebiliriz. Yani X, Y'ye indirgenebilir.

Burada X ya da Y için polinom sınırlı bir algoritmanın varlığı gerekli değildir. Yalnızca birini çözen bir iyi algoritma varsa, bunun diğeri için de iyi algoritma elde etme anlamına geleceğini söylüyoruz. Bunun için de X'i Y'nin özel bir durumuna polinom sınırlı sürede indirgeyecek bir algoritma bulmak yeterlidir. Eğer NP sınıfındaki her problem Y probleminin özel durumlarına indirgenebiliyorsa, Y problemi *NP-zor* ve aynı zamanda Y'nin kendisi de NP sınıfından ise Y'ye *NP-tam problem* denir.

Bir NP tam probleme verimli bir çözüm yöntemi bulunması halinde diğer NP tam problemler de verimli bir şekilde çözülebileceklerdir. Bu birbiri ile ilişkisiz gibi görünen problemlerin hepsinin matematiksel bir anlamda, aynı ailenin bireyleri oldukları, ilk olarak Toronto Üniversitesi'nden Stephen Cook tarafından 1971'de farkedildi. Daha önce matematikçiler bu problemlerin herbirini tek tek ele alıp, herbiri için ayrı bir "iyi" çözüm yöntemi arıyorlardı. Cook'un çalışmalarından sonra bilimciler daha önce bağlantısız olan birçok problemin NP tam olduğunu görmek için atağa geçti ve günümüzde yüzlerce bu tür problem saptanmıştır. Bu problemlerden belki de, en ünlüsü Gezgin Satıcı Problemidir .

Gezgin Satıcı Problemi: Bir gezgin satıcının bulunduğu bir şehirden başlayarak $(n-1)$ sayıda

şehrin her birine ancak ve ancak bir defa uğramak koşulu ile tekrar başladığı şehre dönmesini sağlayan en kısa turu bulma problemidir.

Bu gezgin satıcı ilk durak olarak bu n kentin herhangi birini seçebilir. Daha sonra, ikinci durak yeri için, $(n-1)$ seçim vardır, sonra $(n-2)$ vb. Böylece farklı gezi programlarının toplam sayısı $(n-1)!$ dir. Ancak bu rotanın yarısı sadece ters yönde, yolun öteki yarısından oluştuğu için doğru sayı $(n-1)!$ in yarısıdır. Formal olarak bile, bu problem yeterince korkunçtur.

Eğer kentlerin sayısı fazla olursa, örneğin Türkiye'nin 79 il merkezi söz konusu olursa ne olur? Satıcının Ankara'dan yola çıkıp yol boyunca 78 il merkezini dolaşıp Ankara'ya döndüğünü düşünelim. Hiç düşünmeden problemin birbirinden farklı 78! in yarısı kadar, yani yaklaşık 10^{90} rota içerdiğini söyleyebiliriz. Saniyede bu tür bir milyon işlem yapabilen bir bilgisayar kullanırsak işlemi tamamlamak yaklaşık $4 \cdot 10^{76}$ yıllık bir zaman alacaktır.

Buradan neden satranç veya benzer oyunların büyük beceri ve marifet istediği anlaşılır. Bu tür oyunlarda algoritma ile verilen oyun stratejilerin pratik bakımdan imkansızlığı ile karşılaşmış oluruz. Böylece bu tür problemlerde ilke olarak, önce, olanaklı her farklı durum için hesaplama yapılarak (bilgisayarda hesaplama daha olası) sonra da (eğer varsa) verilen koşulları sağlayan en verimli sonuç seçilip tam olarak çözüme varılabilir. Ancak sorun, bu tür problemlerde farklı durumların sayısının akıl almaz büyüklükte olmasıdır.

10. Son durum ve gelişmeler

Günümüzde Matematiğin hemen-hemen tüm alanlarında karşımıza çıkan NP-tam problemlerin listesinde her geçen gün artan 1000'den fazla problem vardır (Bkz. Journal of Algorithms).

Bugüne kadar yapılan araştırmalarda NP-tam problemlerini çözebilen bir polinom zaman algoritması bulunamamıştır ve genel kanı böyle bir algoritmanın var olmadığı yönündedir. Fakat bu öngörü henüz kanıtlanmamıştır. Uzmanların çoğu Turing makinesine benzer bir aygıt kullanılarak, NP-tam probleminin polinom sürede çözülmesinin gerçekte olanaksız olduğunu ve sonuçta P ve NP-nin aynı olamadığına inanmaktadırlar. S.Smale'e göre "$P \neq NP$?" problemi XXI yüzyılın Matematiğin en önemli sorusudur.

Son yıllarda hesaplanabilirlik kuramında en önemli gelişme hesaplamaların ***kuantum modelinin*** yaratılmasıdır. Üzerinde en çok konuşulan ***kuantum bilgisayarı*** fikri de burada ele alınmaktadır. David Deutsch'e göre, bir "kuantum bilgisayarı" inşa etmek "ilke olarak" olasıdır. Deutsch bu bilgisayar için P'de yer almayan, ama bu aygıt sayesinde polinom sürede çözülebilen problemlerin (problem sınıfları) var olduğunu ileri sürmektedir.

Sonuç: Üssel sınırlı algoritmaların teknolojik gelişmelerden sağladıkları yarar önemsiz denecek kadar küçük olduğu kolayca gösterilebilir. Bu yüzden etkili algoritma geliştirme, bilgisayar teknolojisinin gelişiminden daha çok önem taşımaktadır.

KAYNAKLAR

[1] Cormen, T.H. , Leiserson, C.E. , Rivest R.L. : Introduction to Algorithms. McGraw - Hill Book Company. New York, 2001.

[2] Carey, M.R. , Johnson, D.S. : Computers and Intractability: A Guide ty the Thoery of NP-Completeness. W.H. Freeman, 1979.

[3] Devis M.D.,Sigal R.,Weyuker E.J.: Computability, Complexity, and Languages. Fundamentals of Computer Science. Academic press. New York, 1994