Uluslararası

maarif

Dergisi

Matematik

Ne İşe Yarar?

Teknoloji Çağında Okul Kütüphaneleri

Balkanlarda Türkçe Öğretimi ve Türkoloji Uluslararası Sempozyumu

PROF. DR. ÇETİN KAYA KOÇ

Bilgisayar, Matematik ve Eğitim

PROF. DR. ASIYE KAKIRMAN YILDIZ

Okuma Alışkanlığı Kazanmada Kütüphane ve Muhitin Önemi

Prof. Dr. Çetin Kaya KOÇ
Iğdır Üniversitesi

Bilgisayar, Matematik ve Eğitim

Teorem kanıtlayan yazılımlar, matematik eğitiminde gerçekten şaşırtıcı gelişmelere yol açmıştır. Bu Hilbert'in bile tasavvur edemeyeceği bir aşamadır. Matematik öğrencileri bu programlama platformlarını anlamak ve kullanmak için ciddi programcılar olmak zorunda kalacaklardır. Lisans öğrencileri, matematik öğrenmek için teorem kanıtlayan programlama dillerini ve platformlarını kullansınlar veya kullanmasınlar, artık matematikteki araştırma çabalarının küçük bir parçasıdırlar.

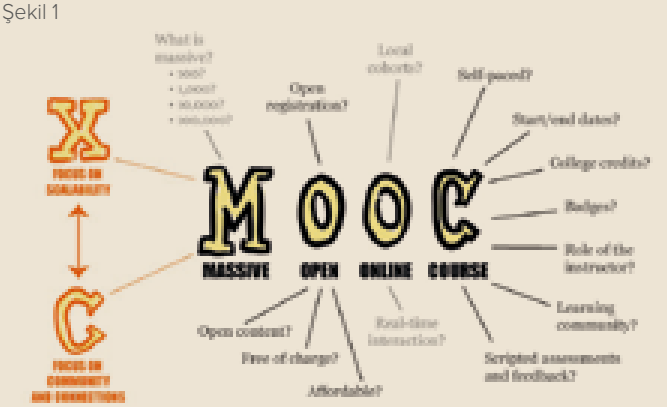
30 yıldır algoritmalar ve kodlama platformları üzerine yoğun bir çabanın ardından nihayet sisin dağıldığını ve karmaşanın ortadan kalkıp ufkun aydınlandığını görebiliyoruz. Meslekten olmayan terimlerle, bilgisayarların ve matematiğin ayrılmaz bir şekilde iç içe geçtiğini ve birbirlerinin ilerlemesini desteklediğini söyleyebiliriz. Başarı hikayeleri art arda geliyor ve bu iki alan tek bir noktada birleşiyor. Başlangıçta bilgisayarlar teoremleri kanıtlamak için kullanılıyordu şimdi ise bilgisayarlar büyük miktarda veri içinde kalıpları araştırıyor ve bunları kanıtlanabilir teoremlere dönüştürüyor. Ancak, bilinmeyen sularda yol alıyoruz ve yolumuz kayalar ve tehlikeli fırtınalarla dolu.

Ciddi sorunların üstesinden gelmek için yeterli yazılımlar henüz ortada yokken, hükümetler ve okul yöneticileri, eğitim krizini çözmek için bilgisayarları sınıflara yerleştirmeye başladılar [1]. Dünya çapında öğretmenlere düşük maaşlar ödendi ve öğrenciler bilgisayarlı eğitime geçmek için temel gereksinimlerden yoksundu, ancak yararlı yazılımlar olmadan yüksek maliyetli donanım getirmek ne öğretmenlerin ne öğrencilerin ne de ailelerinin sormadığı soruların cevabı gibi görünüyordu. (Resim 1) Öğretmenlerin donanımı kullanmaları bekleniyordu ama program dilinden anlamıyorlardı. Birkaç yıllık denemeden sonra, bilgisayarların eğitime uyarlanmasının güç olduğu en ateşli savunucular da dahil olmak üzere herkes için açıktı- en azından şimdilik.

Okullar kısa sürede geleneksel yöntemlere geri döndü ancak bu süreçte edinilen tecrübenin hepsi de heba olmadı. Bu tecrübeden kurs içeriklerini web üzerinden ve en önemlisi video aracılığıyla nasıl aktaracağımızı öğrendik. Bunlar, hızlı eğitim için ek yöntemler olarak daima var olacaklar. Akabinde yirmi yıl boyunca sınıflarda öğretmenleri bilgisayarlarla desteklemek için birkaç girişim daha oldu. Bunların içinde en çok göze çarpan ***MOOC (Kitlesel Açık Çevrimiçi Kurslar)*** adı verilen girişimdir. (Şekil 1) Eğitimden ziyade ölçeklenebilirlik ve maddi kazanç için tasarlanan MOOC, birkaç yıl içinde mutlak bir başarısızlıkla sonuçlandı. -Bununla birlikte bize bazı uygulanabilir modeller de bıraktı, örneğin kursların bir kısmını çevrimiçi açık öğretim yönteminin kullanılabildiği bir ortama taşıdı.- En azından şimdilik, mobil cihazların ve internetin yaygınlaşması ve mekâna bağlı olmadan erişim rahatlığı sağlaması gibi etkenler nedeniyle sınıflarda bilgisayar konusunda ısrar edilmiyor.

Resim 1

Şekil 1



PROGRAMLAMA VE MATEMATİK

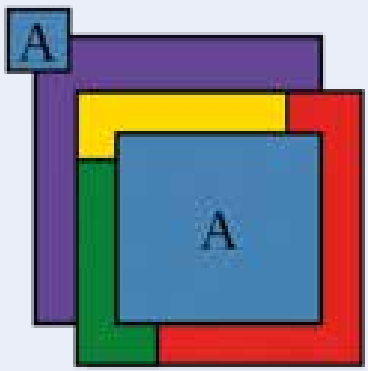
Sınıflardaki bilgisayarların, web ve video aracılığıyla çevrimiçi sunulduğu ve mevcut müfredatla entegre MOOC'un bir kombinasyonunun kullanılması dünyanın üzerinde uzlaştığı genel yaklaşımmış gibi görünüyor. Ancak, bu arada sessiz bir devrim yaşanmakta. Otuz yıl boyunca matematik ve bilgisayar bilimleri eğitimi iç içe geçti. Zaten matematik bölümlerinde hesaplama teorisi, algoritmalar ve hesaplanabilirlik üzerine dersler vardı. Ek olarak, matematikçiler programlama öğrenmeye ve çeşitli programlama platformlarında yetkinleşmeye, hatta cebirsel hesaplama için kendi programlama dillerini ve platformlarını geliştirmeye başladılar. ***Maple, Mathematica, Octave, Sage*** gibi açık kaynak kodlu ve tescilli yazılımlar, hatalı el hesaplamalarından kurtulmak ve egzotik cebirsel yapılarda aritmetik yapmak için geliştirildi. Bu araçların ***Euler*** ve ***Gauss'***u kıskandıracağını hayal edebiliyorum! Bu imkanlara sahip olsalardı belki daha egzotik özdeşlikler ve kanıtlar keşfederlerdi. Tarih kitaplarında Gauss'un bir dizi aritmetik hesaplama yapması için genç matematikçiler tuttuğu ve ofisinde bunlardan daha genel kurallar ve ilişkiler tahmin etmelerini istediğini öğreniyoruz. Şimdi bir ***Python for-döngüsü*** aynı şeyi bir zamanın milyarda birinde yapabilir.

PROF. DR. ÇETİN KAYA KOÇ
Iğdır Üniversitesi

İTÜ Elektronik ve Haberleşme Mühendisliği'ni 1980 yılında birincilikle bitirdi. Doktorasını Kaliforniya Üniversitesi'nde yaptı.

ABD'deki Oregon Eyalet Üniversitesi'nde (Oregon State University) Bilgi Güvenliği Merkezi kurmuş. Buradaki çalışmaları sonucunda 'Olağanüstü ve Sürdürülebilir Araştırma Liderliği' ödülüne layık görüldü. Dünyanın ikinci büyük kriptografi konferansı olan "Cryptographic Hardware and Embedded Systems (CHES)"in kuruculuğunu üstlendi. Kriptografi (şifreleme) mühendisliğine yaptığı katkılarından dolayı 2007 yılında "IEEE Fellow" (alanında çok değerli işler yapmış bilim insanları) unvanına layık görüldü. 50'den fazla kriptografik cihaz, yazılım ve donanımın tasarım ile geliştirilmesine katkıda bulundu. Üzerinde çalıştığı şifreleme yazılımlarından BSAFE, 2003 yılında yapılan tahmine göre 4.5 milyardan fazla cihazda yer aldı. Profesör Çetin Kaya Koç'un tasarladığı algoritmalar, Intel, Samsung ve Huawei gibi firmaların cihazlarında kullanılıyor. Stanford PLOS 2019 araştırmasına göre en çok atıf alan 17 bin 80 bilgisayar mühendisi arasında 103'üncü sırada. Kriptoloji ve şifreleme alanında en çok doktora öğrencisi yetiştiren dünyadaki üç akademisyeninden biri.

Şekil 2



Bu yeni oyunun adı hesaplamalı matematiktir. Bunun tamamen matematik olduğunu söylemiyorum, çünkü o sadece bunun bir dalıdır, hatta sadece cebirin bir dalıdır. 1970'lerin başlarında, matematikçiler kombinatoryal yapıları denemek için bilgisayar gücünü (for-loop'un gücü) test etmeye başladılar. İlk atılım, herhangi bir düzlemsel haritanın bölgelerini renklendirmek için daha fazla dört rengin gerekli olmadığını belirten dört renkli harita teoreminin kanıtıydı, böylece iki bitişik bölgenin aynı renge sahip olmadığı kanıtlandı. Kanıt, insanlar tarafından tespiti imkânsız tüm durumların kontrol edilmesini gerektiriyordu. (Şekil 2) Bu, insan yeteneğiyle doğrulanamadığı için bazılarının şüphe ettiği ilk bilgisayar destekli kanıttı. Bazı şüpheleri ortadan kaldırmak için, bir grup matematikçi teoremin daha basit bir versiyonunu ortaya koydu, 1997'de bilgisayar destekli araçlarla yine kanıtlandı. Dört renkli harita teoreminin üçüncü kanıtı 2005'te ***Georges Gonthier*** tarafından genel bir teorem kullanılarak geldi -***Proving software*** kanıtlayan yazılım-. Bu tür yazılım paketleri 2000 yılından beri matematikçi-bilgisayarcı bilim insanları tarafından oluşturulmuştur. İnsan-makine iş birliği bilgisayar destekli ispatların adımlarını otomatikleştirerek teoremleri kanıtlamak için çok faydalı sonuçlar ortaya koymaktadır. [3]

Yeni bir matematik geliştirmenin temel adımları tanım, problemin ifadesi ve ispattır. Üniversitede matematik okuyan bizler, en yaratıcı unsurlar olarak göründükleri için genellikle tanım ve problemin ifadesi boyutunu büyüleyici buluruz. Çoğumuzda matematiği dâhilerin işi olduğuna dair bir algı var ve yaratıcı çalışmalar tanım ve problem ifadelerinde çok daha belirgin. Öte yandan, her biri bir ön-

Otuz yıl boyunca matematik ve bilgisayar bilimleri eğitimi iç içe geçti. Zaten matematik bölümlerinde hesaplama teorisi, algoritmalar ve hesaplanabilirlik üzerine dersler vardı. Ek olarak, matematikçiler programlama öğrenmeye ve çeşitli programlama platformlarında yetkinleşmeye, hatta cebirsel hesaplama için kendi programlama dillerini ve platformlarını geliştirmeye başladılar.

Şekil 3

Show that the premises "A student in this class has not read the book," and "Everyone in this class passed the first exam" imply the conclusion "Someone who passed the first exam has not read the book."

Solution: Let $C(x)$ be "x is in this class," $B(x)$ be "x has read the book," and $P(x)$ be "x passed the first exam." The premises are $\exists x(C(x) \wedge \neg B(x))$ and $\forall x(C(x) \rightarrow P(x))$. The conclusion is $\exists x(P(x) \wedge \neg B(x))$. These steps can be used to establish the conclusion from the premises.

Step	Reason
1. $\exists x(C(x) \wedge \neg B(x))$	Premise
2. $C(a) \wedge \neg B(a)$	Existential instantiation from (1)
3. $C(a)$	Simplification from (2)
4. $\forall x(C(x) \rightarrow P(x))$	Premise
5. $C(a) \rightarrow P(a)$	Universal instantiation from (4)
6. $P(a)$	Modus ponens from (3) and (5)
7. $\neg B(a)$	Simplification from (2)
8. $P(a) \wedge \neg B(a)$	Conjunction from (6) and (7)
9. $\exists x(P(x) \wedge \neg B(x))$	Existential generalization from (8)

Genel algoritmalar ve teorem kanıtlayan yazılım Lean'den bazı örnekler [4]'tedir. Tipik bir Yalın kodu Şekil 4'te gösterilmektedir.

Şekil 4

```
-- Need access to many useful tactics
import tactic

-- Need injective functions
open function

-- Let X, Y, Z be types and
-- let f : X → Y and g : Y → Z be
-- functions between these types
variables {X Y Z : Type}
  {f : X → Y} {g : Y → Z}

-- Theorem: if f and g are
-- injective, then so is g ∘ f.
theorem injective_comp :
  injective f ∧ injective g →
  injective (g ∘ f) :=
begin
  -- assume f and g are injective.
  rintro ⟨f_inj, g_inj⟩,
  -- We want to prove g ∘ f is
  -- injective. So say a,b ∈ X and
  -- assume g(f(a))=g(f(b)).
  intros a b hgf,
  -- We want to prove that a = b. By
  -- injectivity of f, it suffices to
  -- prove that f(a)=f(b).
  apply f_inj,
  -- By injectivity of g, it suffices
  -- to prove g(f(a))=g(f(b)).
  apply g_inj,
  -- But this is an assumption.
  assumption,
end
```

cekinin doğruluğuna bağlı, genellikle önerme dizileri olan ispatların mekanik yönleri hakkında çoğu zaman kafamız karışır. Bunlar pek çoğumuza, anlaşılmaz sembol dizileri gibi görünürler, örneğin Şekil 3'teki basit bir ispata bakınız. Bunlar matematiksel ifadelerden çok bilgisayar programlarına benzemiyor mu?

TEOREM KANITLAYAN PROGRAMLAR

Teoremleri ispatlarken, şu adımları izleyen bir ***"makine"***ymişiz gibi çalışırız: P Doğruysa ve P Q'yu ima ediyorsa, Q Doğru olmalıdır. Bu aksiyomatik yaklaşım, grup teorisi, lineer cebir ve gerçek analizde önemli bir rol oynar. Mantığın çıkarım kurallarına uyan programlar yazarak kendimizi ispat adımlarını takip eden makinelerle donatıyoruz.

Yalın kod'un (Lean code) yanı sıra, çok derin matematik yapabilen ciddi teorem kanıtlayan ***Coq ve Isabelle/HOL*** de platformları da kullanılmaktadır. Bu paketler aynı zamanda lisans matematik öğretimi için de yararlıdır. Bu yazılımlar, matematik eğitiminde gerçekten şaşırtıcı gelişmelere yol açmıştır. Bu ***Hilbert***'in bile tasavvur edemeyeceği bir aşamadır. Matematik öğrencileri bu programlama platformlarını anlamak ve kullanmak için ciddi programcılar olmak zorunda kalacaklardır. Yine de tecrübelerimiz, matematik öğrencilerinin yalnızca küçük bir alt kümesinin bu yolu izlediğini göstermektedir. Lisans öğrencileri, ciddi matematik öğrenmek için teorem kanıtlayan programlama dillerini ve platformlarını kullansınlar veya kullanmasınlar, artık matematikteki araştırma çabalarının küçük bir parçasıdırlar. Bu sistemlerde bir ispat yazmak, insanların düşüncelerini netleştirmelerine yardımcı olur ve belki de onları daha iyi ve daha derin soyutlamalara yönlendirir.

Soyutlamalar ve teorem kanıtlama platformları hakkında bazı uyarılar da yok değil. Bu sistemler tarafından üretilen ispatlar uzadıkça uzuyor ve takip edilmesi çok zorlaşıyor. Ürettikleri kanıtları doğrulamak için bu sistemleri kullanabilir miyiz? Görünen o ki, ***Halting*** problemine benzer bir çelişkiye çok yakınız. Yakında bir bilgisayarın insanların ilgilendiği bir varsayımın kanıtını duyurduğu bir noktaya gelebiliriz ve eğer argüman yeterince derinse, o zaman bilgisayar argümanlarını insanların anlayabileceği bir dil kullanarak açıklayamayabilir [4]. Herhangi bir insan bir bilgisayarın "bu kanıt, benim için anlamanı bekleyemeyeceğim kadar önemli" böbürlenmesinden hoşnut olur mu, emin değilim. [5]

KAYNAKLAR

[1] Neal Koblitz. "The Case Against Computers in K-13 Math Education (Kindergarten through Calculus)", The Mathematical Intelligencer, 18(1):9-16, 1995.

[2] Kenneth Appel and Wolfgang Haken. "Every Planar Map is Four Colorable: I. Discharging & II. Reducibility", *Illinois Journal of Mathematics*, 21 (3):429-567, 1977.

[3] Stephen Ornes. "How Close Are Computers to Automating Mathematical Reasoning", *Quanta Magazine*, August 27, 2020.

[4] Kevin Buzzard. "Proving Theorems with Computers", Notices of American Mathematical Society, 67(11):1791-1799, December 2020.

[5] Arthur Charles Clarke and Stanley Kubrick. "2001: A Space Odyssey", 1968.