BÖLÜM 27. YAKLAŞIM ALGORİTMALARI

Bilimsel ve mühendislik problemlerinin analitik çözümleri yapıldığı gibi pratikte uygulanması için çözümün sayısal olarak yapılması gerekmektedir. Sürekli ortamlarda bir problemi sayısal olarak çözmek için kesin çözüm yerine yaklaşık çözüm bulunması yeterlidir. Yaklaşık çözümün matematiksel anlamı, elde edilecek olan çözüm gerçek çözümün ε-komşuluğundaki bir değer olup ε değeri ne kadar küçük ise, çözüm o kadar iyi olur.

Yaklaşım yöntemi bir çok alana uygulanabilmektedir ve bu alanlar sürekli alanlar olabileceği gibi ayrık alanlar da olabilir. Yaklaşım yöntemlerinin gerekliliğini anlamak için Turing makinesinin kısaca ne olduğu üzerinde durulması gerekmektedir.

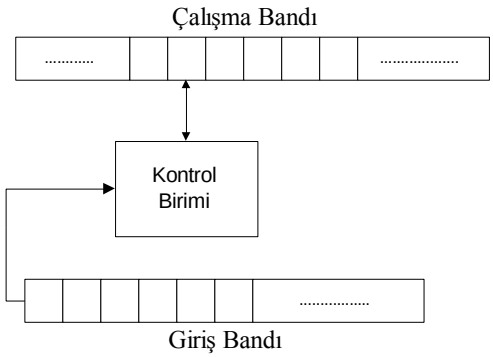
Turing Makinesi, sonlu durum kontrol kısmı, giriş bandı ve sınırsız bellekten (çalışma bandı) oluşan bir hesapsal modeldir (Şekil 27.1).

Tanım 27.1.
Kontrol kısmı, komutların icra edilmesini sağlayan (bantlardan simge okuma ve yazma, okuma/yazma başlıklarının hareket etmeleri) bir birimdir.

Tanım 27.2.
Bant, bir boyutlu, çift yönlü, genel amaçlı bir hafıza ortamı olup, giriş/çıkış işlemleri ve ara sonuçları saklamak için kullanılan bir bölgedir.

Bant (Bellek), hücrelerden oluşur ve bu hücrelere hem yazılabilmektedir ve hem de okunabilmektedir. Her hücre başlangıçta boşluk içerir ve boşluk B ile gösterilecektir.

Şekil 27.1' den de görüleceği gibi giriş bandının bir tarafı kapalı ve diğer tarafı açıktır. Çalışma (çıkış) bandının ise her iki tarafı açıktır. Bir Turing makinesinin çalışması bir durumdan başka bir duruma geçiş şeklinde olur. Bir geçiş



Şekil 27.1. *Bir Turing makinesinin ana kısımları.*

- Yeni duruma gitme
- O anda okuma/yazma başlığının altında bulunan hücreye yazma işlemi yapma
- Okuma/yazma başlığı bir hücrenin üzerinde sabit tutulabileceği gibi sağa veya sola da hareket edebilme

işlemlerinden oluşur. Bir durumdan başka bir duruma geçilecekse, gidilecek durumun seçimi aşağıdaki durumlara bağlıdır.

- O anki duruma
- Bir sonraki giriş simgesine
- O anki çalışma bandı üzerindeki simgeye

Turing makinesi, hesaplama modeli olduğuna göre hesaplama işlemi kısaca şöyledir. Sonlu sayıdaki giriş verileri hesaplama işlemi başlamadan önce banda yazılır. Hesaplama işlemine, verilen bir "durum" ile en soldaki hücreyi okumakla başlanır. Hücrede her ne yazılı ise silinir, yerine tekrar bir simge yazılır, komşu hücreye geçilir ve durum güncelleştirilir. Bu işlemler hesaplama bitene kadar bir döngü içerisinde devam eder. Eğer giriş bandı ile çıkış bandı farklı ise, okuma işlemleri giriş bandından yapılırken, yazma işlemleri çıkış bandı üzerine yapılır.

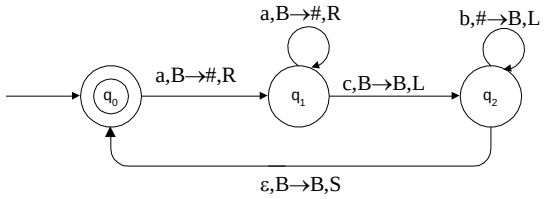
Bütün bunlardan sonra Turing makinesinin tanımı verilebilir.

Tanım 27.3.
Turing makinesi, algoritmalara ve mekaniksel yordamlara kesin bir tanım getirmek amacıyla ileri sürülmüş kuramsal bir hesaplama modelidir.

Bir Turing makinesi, bir dili tanıması için tasarlanabilir. Çünkü bir dili tanıma problemi karar verme problemi olup var olan problemlerin hepsinin karar verme şekilleri tanımlanabilmektedir. Bunun için Turing makinesinde birden fazla son durum olabilir ve çalışma sırasında Turing makinesi giriş verisini bitirdikten sonra son duruma ulaşıyorsa, bu Turing makinesi, girilen simge dizgesini tanımış olur. Herhangi bir dilin bütün kelimelerini tanıyan Turing makinesi, o dili tanımış olur. Bitime gitmeyen Turing makinesi giriş verisini tanımamış olur.

Bir Turing makinesi, sonuç hesaplayabilir ve birden fazla evet/hayır cevabı üretebilir. Bu durumda çalışma bandının son durumu ile ilgili bir cevap üretilmesi sağlanır.

Örnek olarak bir dili tanıyan Turing makinesi tasarlanabilir ve örnek dil $\{a^n cb^n \mid n \geq 0\}$ şeklindeki bir dili tanıyan Turing makinesi tasarlanabilir.



Şekil 27.2. *$\{a^n cb^n \mid n \geq 0\}$ dilini tanıyan Turing makinesi.*

x,y→u,v şeklinde bir geçişte x giriş simgesini, y o anda çalışma bandında okuma/yazma başlığının üzerinde bulunduğu hücrenin içermiş olduğu simgeyi, u okuma/yazma başlığının altında bulunan hücreye yazılan simgeyi gösterir ve son olarak v ise okuma/yazma başlığının hareket yönünü belirler. x,y→u,v gibi bir geçiş için eğer giriş simgesi x ve çalışma bandı simgesi y ise, bu geçiş için şartlar sağlanmış olur ve bu geçiş yapılır. B simgesi boşluk anlamında kullanılmaktadır ve ε simgesi ise hiçbir şey anlamında

kullanılmaktadır. Şekil 27.2' deki makinenin nasıl çalıştığına bakılabilir. Giriş bandından gelen her a için çalışma bandındaki hücreye # simgesi yazılmaktadır ve sağda ve bitişik olan hücreye geçilmektedir. Giriş bandından c simgesi geldiğinde çalışma bandında okuma/yazma başlığının üzerinde bulunduğu hücrede boşluk olduğundan herhangi bir şey yazılmaz ve bir soldaki hücreye tekrar dönülür ve giriş bandından gelen her b simgesi için ise, okuma/yazma başlığının üzerinde olduğu hücreye B simgesi yazılır ve bitişik solda bulunan hücreye gidilir. b simgeleri bittiğinde eğer okuma/yazma başlığı üzerinde bulunduğu hücrede B varsa, tanıma işlemi gerçekleşir. Bu işlem için hiçbir şey gelmediği halde okuma/yazma başlığının üzerinde bulunduğu hücrede B varsa, son duruma geçiş yapılmaktadır ve okuma/yazma başlığı yerinde kalmaktadır.

Tanım 27.4.

Turing makinesi, bir hesapsal model olduğundan $M=(Q,\Sigma,\Gamma,\delta,q_0,B,F)$ *şeklinde tanımlanır ve*

Q: *sonlu durumlar kümesi.*
Γ: *izin verilen bant simgeleri kümesi.*
B: $B \in \Gamma$ *ve boşluk*
Σ: $\Sigma \subset \Gamma$ *ve* $B \notin \Gamma$ *ve giriş verileri kümesi.*
δ: *bir sonraki durum fonksiyonu ve* $\delta: Q\times\Gamma \rightarrow Q\times\Gamma\times\{L,R,S\}$
q_0: $q_0 \in Q$ *ve başlangıç durumu*
F: $F \subseteq Q$ *ve son durumlar kümesi.*

Bu tanımlamaya göre Şekil 27.2' deki makinenin geçiş fonksiyonu aşağıdaki gibi olur.

$\delta(q_0,a)=(q_1,\#,R)$
$\delta(q_1,a)=(q_1,\#,R)$
$\delta(q_1,c)=(q_2,B,L)$
$\delta(q_2,b)=(q_2,B,L)$
$\delta(q_2,\varepsilon)=(q_0,B,S)$

$\alpha_1 q \alpha_2$ Turing makinesi için bir komut olup $q_0 \in Q$ ve $\alpha_1,\alpha_2 \in \Gamma^*$ olur. $Q \cap \Gamma = \varnothing$ olduğu kabul edilir. M bir Turing makinesi olmak üzere M' nin bir komutu şöyle tanımlanır. $X_1X_2...X_{i-1}qX_i...X_n$ bir komut olsun

ve $\delta(q,X_i)=(p,Y,L)$ için eğer i-1=n ise, $X_i=B$ olduğu kabul edilir. Eğer i=1 ise, bir sonraki komut yoktur. Eğer i>1 ise,

$$X_1X_2...X_{i-1}qX_i...X_n \vdash_M X_1X_2...X_{i-2}X_{i-1}YpX_{i+1}..X_n$$

olur ve $\vdash_M$ simgesinin anlamı, M makinesi kullanılarak $X_1X_2...X_{i-1}qX_i...X_n$ dizgesi ve q durumundan bir geçiş sonucunda $X_1X_2...X_{i-2}X_{i-1}YpX_{i+1}..X_n$ dizgesine ulaşılır. Eğer $X_1X_2...X_{i-2}X_{i-1}YpX_{i+1}..X_n$ dizgesinin son elemanı B ise, bu eleman silinir. Bu tanımlamadan sonra geçiş fonksiyonu ve bu geçiş fonksiyonu kullanılarak makinenin çalışması Şekil 27.2' deki makine için gösterilebilir.

Durum	a	b	c	ε
q_0	$(q_1,\#,R)$	----	----	----
q_1	$(q_1,\#,R)$	----	(q_2,B,L)	----
q_2	----	(q_2,B,L)	----	(q_0,B,S)

Şekil 27.3. *Şekil 27.2' de verilen Turing makinesi için geçiş fonksiyonu.*

M gibi bir Turing makinesi tarafından tanınan L dili L(M) şeklinde gösterilir. Daha matematiksel olarak $M=(Q,\Sigma,\Gamma,\delta,q_0,B,F)$ tarafından tanınan dil $\{w|w\in\Sigma^*$ ve $q_0w \vdash_M \alpha_1p\alpha_2$, $p\in F$ ve $\alpha_1,\alpha_2\in\Gamma^*\}$ şeklinde tanımlanır.

Örnek olarak $L=\{0^n1^n|n\geq1\}$ dilini tanıyan Turing makinesi tasarlanabilir. $Q=\{q_0,q_1,q_2,q_3\}$ ve $F=\{q_3\}$, $\Sigma=\{0,1\}$ için geçiş fonksiyonu Şekil 27.4' te görüldüğü gibi olur.

Durumlar	1	0	ε
q_0	$(q_1,1,R)$	----	----
q_1	$(q_1,1,R)$	(q_2,B,L)	----
q_2	----	(q_2,B,L)	(q_3,B,S)
q_3	$(q_1,1,R)$	----	----

Şekil 27.4. *$L=\{0^n1^n|n\geq1\}$ dilini tanıyan Turing makinesi.*

Normal bir belirli (deterministic) Turing makinesi, herhangi bir durumda gelen giriş simgesine karşılık sadece ve sadece bir geçiş yapar (eğer bu durum ve giriş simgesine göre geçiş tanımlı ise) ve bu nedenle izlenecek yol bellidir. Fakat belirsiz Turing makinesinde ise, herhangi bir durumda bir giriş simgesine karşılık birden fazla geçiş yapılabilmektedir. Bundan dolayı bu makinelerin çalışması bir ağaç yapısı olarak ele alınabilir, çünkü izlenecek yol sayısı birden fazladır. Belirsiz Turing makinesi ile belirli Turing makinesi arasında bir denklik söz konusu değildir. Diğer bir deyişle belirsiz Turing makinesini belirli Turing makinesine dönüştürecek bir algoritma günümüze kadar geliştirilmiş değildir. Belirli Turing makinesi ile çözülebilen problemlere polinomsal problemler denip bunlar P sınıfı problemlerdir. Bunun dışındakiler NP veya co-NP sınıfı problemlerdir. P sınıfı problemler için etkili algoritmalar vardır. NP ve co-NP sınıfı problemler için çoğu zaman yaklaşım yöntemleri kullanılır. Pratikte sürekli ortamların herhangi bir problemini çözmek sürekli değer gerektireceğinden yaklaşım yöntemi gerektirecektir.

27.1. Newton-Rapson Yöntemi

f(x)=0 denkleminin çözümü Newton-Rapson yöntemi ile yapılabilir. Bunun için f(x) bağıntısının Taylor açılımı elde edilir (sıfır noktası civarında).

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{f''(x_0)h^2}{2} + \cdots$$

Verilen bu denklemde üçüncü terimden sonraki terimler değer olarak küçüleceklerinden dolayı yaklaşım yönteminde çözüme yaklaştıkça bu terimlerin değeri gittikçe küçülecektir, bundan dolayı çözüm yapılırken

$f(x_1)=f(x_0+h)=f(x_0)+hf'(x_0)$

eşitlikleri kullanılabilir. Buradan $f(x_0)+hf'(x_0)=0$ alındığında

$$h = x_1 - x_0 = -\frac{f(x_0)}{f'(x_0)}$$

eşitliği elde edilir ve bu eşitlikte görüleceği gibi h hatayı temsil eder. Çözümü elde etmek için h değişkeninin değerinin sıfıra gitmesi gerekmektedir.

Algoritma 27.1. Newton-Rapson

▷ f(x)=0 eşitliğinin çözümünün Newton-Rapson ile çözülmesi.

1- $x \leftarrow x_0$
2- h=f(x)/f'(x)
3- x=x-h
4- eğer $|h|<\varepsilon$
5- 8. adıma git ve işlemi bitir.
6- değilse
7- 2. adıma git
8- işlem sonu ve sonucu görüntüle.

Algoritma 27.1' de Newton-Rapson yönteminin f(x)=0 eşitliğini sağlayan x değerinin bulunması için kullanılabileceği gibi f(x) gibi bir bağıntının minimum veya maksimum noktasının bulunması için de kullanılabilir. Bu durumda f'(x)=0 olduğundan f'(x) bağıntısının kök değeri bulunur.

27.2. Kuadratik Tahmin Yöntemi

Diğer bir yaklaşım yöntemi de kuadratik bağıntıların minimum veya maksimum noktasının bulunmasında kullanılan yaklaşım yöntemidir. Bunun için serbest değişken için bir başlangıç değeri ve adım değeri

belirlenir (bu değerler gelişigüzel olarak belirlenir). Bu işlemi gerçekleştiren algoritma Algoritma 27.2' de görülmektedir.

Algoritma 27.2. Kuadratik_Yaklaşım

▹ f(x) kuadratik bir bağıntı olmak üzere bu bağıntının minimum veya maksimum noktasını bulur.

1- x_0 ve Δx gelişigüzel olarak belirlenir.
2- $k \leftarrow 1$
3- $x_k \leftarrow x_{k-1}+\Delta x$
4- $f_k \leftarrow f(x_k)$, $f_{k+1} \leftarrow f(x_{k+1})$
5- Eğer $f_{k+1}<f_k$ ise, minimuma yaklaşıyor.
6- Eğer $f_{k+1}<f_k$ ise
7- $x_{k+2} \leftarrow x_{k+1}+\Delta x$ olur
8- değilse
9- $x_{k+2} \leftarrow x_k+\Delta x$ olur
10- 3 nokta kullanılarak x* ve f* kuadratik polinom yaklaşımı ile hesaplanır.
11- $f_{min} \leftarrow \min(f_1,f_2,f_3)$ ve buna karşılık gelen x_{min} hesaplanır.
12- Eğer $\left|\frac{f^* - f_{min}}{f_{min}}\right| \le \varepsilon_f$ ise işlemi bitir.
13- $\{x_1,x_2,x_3,x_{min}\}$ kümesinde en büyük değer atılır ve 10. adıma gidilir.

27.3. Küme Kapsar

U bir n tane elemanın evreni olsun ve $S=\{S_1,S_2,...,S_k\}$ kümesi U kümesinin alt kümelerinin bir koleksiyonu olmak üzere $m:S \rightarrow Q$ bir maliyet fonksiyonu olsun. Amaç, U kümesinin bütün elemanlarını kapsayan ve S kümesinin minimum maliyetli alt koleksiyonunun bulunmasıdır.

Problemi çözmek için her elemanın frekansı kullanılabilir ve bir elemanın frekansı, o elemanın elemanı olduğu alt küme sayısı olarak tanımlanabilir. En sık kullanılan elemanın frekansı f olsun. f=2 olduğu durumda problem, düğüm kapsama problemine dönüşür.

Bu problemin çözümü için kaba seçim algoritması kullanılabilir. Mantık olarak maliyet-etkili küme seçilir ve bu kümenin içermiş olduğu elemanlar silinir; bu işleme bütün elemanlar kapsanıncaya kadar devam edilir. Iterasyon başında K kapsanan elemanları içeren küme olsun. Bu iterasyonda yeni eleman içeren bir maliyet-etkili S kümesinin ortalama maliyeti

$$\frac{f(K)}{|S - K|}$$

olur. Bir elemanın ortalama kapsanma maliyeti hesaplanabilir. Bir kümenin kapsanması durumunda maliyetin elemanlara eşit olarak dağıtıldığı kabul edilir.

Algoritma 27.3. Küme Kapsama

▹ f(x)=0 eşitliğinin çözümünün Newton-Rapson ile çözülmesi.

1- K←∅
2- K≠U olduğu sürece
3- Maliyet-etkili minimum olan S kümesini bul.
4- $\alpha \rightarrow \frac{f(R)}{|S -|}$