SİMGESEL İŞLEM
(SYMBOUC MANIPULATION)

Prof. Dr. Avadis HACINLIYAN

Boğaziçi Üniversitesi Faik Bölümü

BİLİMSEL AMAÇLARLA kullanılan FORTRAN, BASIC, PASCAL ve benzeri diller sayısal işlemlere yöneliktir. Bu dillerde, belirli ölçülerde karakter türü verileri işleme olanağı varsa da, bu dilleri simgesel programlama dilleri adı ile bilinen dillerden ayıran en önemli husus, kullanılan her değişkenin değerinin kullanımdan önce tanımlı olması gereğidir. Bu tanımlama, atama, belirtme deyimi ile tanımlama, giriş, üst ve altprogramlar arasında iletişim gibi yöntemlerden biri ile o değişkene sayısal, mantıksal ya da karakter şeklinde bir değer vermek suretiyle gerçekleşir. İşlemler ise değişkenlere atanmış bulunan değerler arasında yapılır. Ticari uygulamalarda kullanılan veri işlemeye yönelik COBOL, genel amaçlı PL/1 yahut C gibi dillerde de veri yapılarının kullanımdan önce değer verilerek tanımlanması esası vardır, bu dillerde kendi kendine tanımlı veri yapıları yer almaz. Cebirde kullandığımız bir x değişkeni ise başka bir değer atanmamışsa x simgesi olarak tanımlıdır. Bu tür kendi kendisi olarak da tanımlı değişkenleri içeren, bu değişkenler üzerinde işlemlere olanak veren diller simgesel programlama dilleridir. Bu değişkenlere bir başka değer atanmadığı takdirde o değişken kendi adı ile tanımlanır, değişkene atanabilecek değerler ise yukarıda belirtilen sayısal ya da benzeri ifadelerle sınırlı olmayıp bir başka simgesel değer, yahut simgesel değerlerden oluşan bir ifade olabilir. Geleneksel programlama aritmetik gibi düşünülürse, simgesel programlama cebirin karşılığıdır. Bu nedenle simgesel programlamaya "bilgisayarlı cebir", simgesel programlama sistemlerine de "cebirsel programlama sistemleri" adı da verilir. Simgesel Programlama olarak çevirmiş olduğum "Symbolic Programming" deyimi bazan makine dili ile çevirici dili arasında bir adım olan, işlemleri çağrıştıran simgesel işlem kodları ve simgesel adresler içeren, ancak makro deyimleri içermeyen dil içinde kullanılır, burada deyim bu anlamda kullanılmayacaktır.

Bunu bir basit örnekle anlamaya çalışalım. N inci mertebeden Chebyshev polinomu şu tür bir ifade ile tanımlanabilir:

T(0)-1 :T(1) = X

FOR I - 2 TO N: T (I)= 2*X*T (1-1) - T (I-2) : NEXT I

Geleneksel bir programlama dilinde bu işlemin yapılabilmesi için önce N ve X' e birer sayısal değer atanmalıdır, bulunan sonuç da sayısal bir değerdir. Örneğin N=5, X=7 değerlerini atarsak işlem sonunda T (5)= 18817 şeklinde sayısal bir değer buluruz. Simgesel programlamada ise X'e herhangi bir değer atamazsak X kendisi yani X simgesi olarak tanımlanır. İşlemler ise cebir kuralları ile gerçekleştirilerek T (5) = $16*X^5-20'X^3+5*X$ cebirsel ifadesi simgesel bir sonuç olarak elde edilir. Bu sonucu bulduktan sonra X'e sayısal, simgesel bir değer, hatta y+3 gibi bir diğer simgesel bir ifade bile atayabiliriz. Bu nedenle de X'in bellekte kaplayacağı yer önceden kestirilemez, hatta işlemler sırasında değişebilir.

Simgesel programlama sistemlerinin gerçekleştirdiği başlıca işlemler şu şekilde özetlenebilir:

1. İstenen duyarlıkta tamsayı ve reel sayılı işlemler. Alışılagelmiş dillerde, tamsayı ve reel sayıların duyarlığı, kullanılan bilgisayar sistemindeki kelime boyutuna bağlı olarak bir yada birkaç sınırlı değer alabilir. Simgesel programlama dillerinde ise bu duyarlılık kullanıcı tarafından tanımlanabilmektedir. Bu sistemlerde 100! ifadesi 178 basamaklı bir tamsayı olarak hesaplanır, bilgisayarın tamsayı yahut reel sayı duyarlılık limiti bu sonucu etkilemez. Bu özellik nedeniyle normal dillerde yer alan sayı tiplerinin çoğuna burada ihtiyaç yoktur, tamsayı, rasyonel sayı ve reel sayı tipleri yeterlidir.

2. Polinomların açılımı ve terimlerinin kullanıcının belirteceği esaslara göre sıralanması, rasyonel fonksiyonların ortak payda üzerine alınması veya kısmi kesirlere ayrılması: Bu işlemler belki de simgesel programlama sistemlerinin sağladığı en önemli kolaylıktır, uzun hesapları kolaylaştırarak hatalardan arındırılmasına yardımcı olur.

3. Polinomların çarpanlara ayrılması: Bu işlem kesirlerin sadeleştirilmesi, entegrasyon gibi işlemler için de gereklidir. Son yıllarda, bu zor işlemin gerçekleştirilmesine yönelik yeni algoritmalar geliştirilmektedir.

4. Sadeleştirme, yerine yerleştirme, kalıp uydurma (pattern matching), işlemleri kullanıcının isteğine göre gerçekleştirilmekte, yerine yerleştirme işlemi yerel (örneğinx sin (x) ifadesinde x yerine y+4 değerini yerleştirme), global ancak belirli bir değişken için (örneğin tüm ifadelerde x yerine y+4 değerini yerleştirme), ya da tüm değerler için global olarak (örneğin tüm argümanlar için $Sin^2 + Cos^2$ ifadesinin Ve eşitlenmesi) gerçekleştirilebilir.

5. Analitik türev ve entegral almak amacı ile bu sistemlerde analizde bilinen türev alma formülleri yer almaktadır. Tipik bir uygulama, bir ifadenin, bağımlı olduğu ve bir başka değişken atanmadığı için kendi kendine tanımlı olan bir argümanına göre kısmi türevinin alınmasıdır. Bu işlem tekrarlanarak birden çok değişkene göre istenen mertebeden türevler alınabilir. Her fonksiyonun, belirli hesaplar sonucu bulunan ve aynı cinsten fonksiyonlar yardımı ile ifade edilebilen türevi olmasına karşın, aynı niteliklere sahip entegrali olmadığından entegral alma işlemi biraz daha zordur. Risch tarafından bulunan bir algoritma yardımı ile belirli tipte entegrallerin simgesel olarak hesaplanması sağlanmıştır. Tablolardan arama yolu ile entegrallerin hesaplanması da zaman zaman başvurulan bir yöntemdir. Öte yandan, simgesel yolla hesaplanan entegrallerin değerlerinin çoğu zaman entegrallerde yer alan parametrelerin değerine bağlı olduğu, simgesel programlama sistemlerinin bazan bu parametrelerin değerleri üzerinde belirli önkabuller yaptığı da bilinmektedir. Dikkat edilmezse, bu önkabuller hatalara yol açabilir.

6. Cebirsel programlama sistemlerinde belirli fonksiyonlar (örneğin trigonometrik, hiperbolik, üstel veya logaritmik fonksiyonlar, seriye açma, limit alma, rasyonel ifadelerin toplamının hesabı gibi prosedürler) yer almakta, fonksiyonların belirli özellikleri sistem tarafından kullanılabilmektedir. Bunun dışında yeni fonksiyonların tanımlanması için gereken olanaklar da sistemlerde yer almaktadır. Tüm sistemlerde, bu fonksiyonların sağladığı kuralların yerel yahut global olarak tanımlanması mümkündür. Bu sistemlerin çoğunda, prosedürel ve fonksiyonel mümkündür. Bu sistemlerin çoğunda, prosedüre! ve fonksiyonel programlama olanakları da mevcuttur.

7. Sembolik matrislerle hesap yapılabilmektedir. Matrislerle dört işlem, transpoze, determinant ve ters matris alma, mümkün olan durumlarda özdeğer ve özvektörlerin hesaplanması gibi işlemler bu sistemlerin tümünde tanımlıdır.

8. Sistemlerin çoğunda diferansiyel denklemlerin entegralini alma, özel fonksiyonlarla hesap yapma, iki yahut üç boyutta grafik çizme, kısmi diferansiyel denklemlerin simetri gruplarını hesaplama, diferansiyel geometri hesapları yapma, Pade yaklaştırımı oluşturma, analitik olarak hesaplanamayan sonuçları nümerik olarak hesaplama, yüksek enerji fiziği, optimizasyon gibi alanlardaki hesaplar için olanaklar mevcuttur. Bu dillerin tümünde ku"anıcı fonksiyonları tanımlamak suretiyle sistemin uygulama alanı genişletilebilir. Bir çoğunda da metin düzenleyici programlarına (TeX veya LafeX gibi) grafik ve simgesel sonuç aktarma, sayısal hesaplama amaçlı dillere (özellikle FORTRAN yahut C) formül aktarma olanağı yer alır.

İlk simgesel programlama sistemleri büyük bilgisayarlara yönelikti. Kişisel bilgisayarların gelişmesi nedeniyle simgesel programlama sistemleri günümüzde kişisel bilgisayarlardan süperbikjisayarlara kadar her tür sistemde uygulanmıştır. 32 bitlik iç mimari, büyük miktarda (mümkünse görüntü) bellek adresleyebilme yeterli olmaktadır. Özelilkle 80386SX ya da üstü işlemcisi olan IBM uyumlu kişisel bilgisayarlarla Macintosh sınıfı bilgisayarlar, bilinen tüm simgesel programlama sistemlerini destekleyebilmektedir.

Simgesel programlama sistemlerinde bir sonuca varmak için gereken adım sayısını önceden kestirmek mümkün değildir. Ayrıca, başlangıç değerleri ve son sonuç basit olmakla birlikte sonuca erişmek için karmaşık, uzunluğu ve dolayısı ile bellekte kaplayacağı yeri önceden kestirilemeyen ara değerlerin hesaplanması gerekebilir. Bu nedenle, simgesel programlama uygulamaları yüksek bellek sığaları ile dinamik bellek ata-

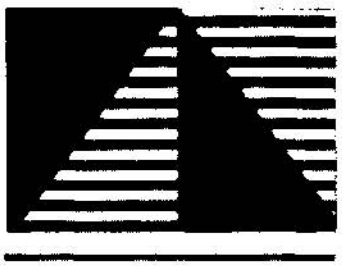
ma ve serbest bırakma, uzunluğu dinamik olarak değiştirilebilen yiğit (stack), değişken uzunluklu verileri temsil edebilen ilişkilendirilmiş liste (linked üst), ikili ya da çoklu ağaç gibi yapılara ihtiyaç gösterir. En başarılı simgesel programlama sistemleri bu yapıların en doğal olarak yer aldığı LISP ve C gibi dillerle geliştirilmiştir. Nitekim, bilinen genel amaçlı simgesel programlama sistemlerinin tümü bu iki dilden biri ile yazılmıştır. Yerine koyma, çarpanlara ayrıma, simgesel entegral alma, kalıp uydurma, sadeleştirme algoritmaları önemli ölçüde yapay us kabiliyetine ihtiyaç gösterir, bu nedenle simgesel programlama sistemleri yapay ustakine paralel bir gelişme göstermiştir. Simgesel programlama sistemlerinden birçoğunun temeli olan LISP, yaygın olarak yapay us uygulamalarında kullanılan bir dildir. Başarılı bir simgesel hesap, kullanıcıya her an müdahale ve yönlendirme imkanı tanıyan etkileşimli bir uygulama da gerektirdiği için bu sistemlerin hemen tümü etkileşimli bir ortam sağlar, çoğunda "batch job" niteliğindeki komut dizilerini de birlikte işleme olanağı vardır.

Simgesel programlama sistemlerini üç sınıfta toplamak mümkündür, bunlardan ilki LISP diline dayalı genel amaçlı sistemlerdir, örnek olarak kişisel bilgisayarlarda uygulanan MUMATH ve bu sistemden geliştirilen DERIVE, Hearn tarafından geliştirilen, bazı eksikliklerine rağmen yaygın bir kullanım alanı bulan REDUCE, bu tür sistemlerin en büyüğü ve genel amaçlısı olarak bilinen MACSYMA verilebilir. İkinci sınıfta C diline dayalı genel amaçlı sistemler toplanabilir, bunun en yaygın iki örneği de Wolfram tarafından geliştirilen SMP ve MATHEMATICA ile Waterloo Üniversitesi tarafından geliştirilen MAPLE'dir. Genel sistemler, çeşitli uygulamalara açık olduklarından bazı özel hesaplar için yeterli hız yahut kapasite sağlayamamaktadırlar. Genel amaçlı dillerdeki kısıtlar nedeniyle özel uygulamaların hesaplama gereksinmelerini kolayca karşılamaya yönelik, ancak sınırlı işlemleri gerçekleştirebilen özel amaçlı sistemler de geliştirilmiştir, bunlar üçüncü sınıfta toplanabilir, örnek olarak sayısal uygulamalara bazı simgesel olanaklar eklemek amacı ile geliştirilen FORTRAN ve PL/1 tabanlı FORMAC, teorik fizikte yer alan Feynman diyagramlarının hesabı için geliştirilen Fortran tabanlı ASHMEDAI ve CDC Çevirici dili (COMPASS) tabanlı SCHOONSHIP, tensör hesapları için geliştirilen LISP tabanlı ORTHOCARTAN, relativite ve gökmekaniği uygulamalarına yönelik CAMAL ve ALAM sayılabilir. Günümüzde teorik fizikte önem kazanan Virasoro Cebirlerinin temsillerini bulmak için gereken büyük tamsayıların çarpanlara ayrılması, nonkomütatif hesapların yapılması işlemleri genel amaçlı simgesel programlama dilleriyle yeterince hızlı bir şekilde yapılamadığından C diliyle özel programlar yazılmaktadır.

80386 tabanlı IBM uyumlu veya Macintosh uyumlu, en az 4 MByte gerçek belleği olan kişisel bir bilgisayar, bu dillerin hemen hemen tümünün kullanılması için yeterlidir. Ayrıca, iş istasyonları, "mainframe" ve süper bilgisayarlarda da simgesel programlama uygulamaları yapılmaktadır. Simgesel programlamanın ilk uygulamaları için başlangıçta ancak büyük çaptaki bilgisayarlar yeterli olabiliyordu. Bilgisayar donanımındaki gelişmeler, simgesel programlamanın genellikle vektör veya paralel programlamaya uygun olmaması gibi nedenler uygulamaları iş istasyonlarına ve kişisel bilgisayarlara kaydırmıştır.

Uluslararası literatürde MATHEMATICA, MACSYMA ve MAPLE dillerindeki uygulamalara sık sık rastlanmaktadır. Bu diller kaynak kodu seviyesinde kullanıcıya açık değildir. Geliştirilen ilk genel amaçlı dillerden biri olan REDUCE ise, grafik özelliklerinin bulunmaması nedeniyle, kullanımı yukarıdaki üç dil kadar yaygınlaşamamış, buna karşı, kaynak kodunun kullanıcıya açık olması, diğer diller gibi simgesel programlama mantığına daha uygun olan (C ye benzer) fonksiyonel programlama yerine (PASCAL ve FORTRAN'a benzer) prosedürel programlamaya yakınlığı nedeniyle belirli kullanıcı gruplarında kabul görmüştür. Ülkemizde de en yaygın kullanılan sistem REDUCE'dir, hatta bu dilde programlamayı tanıtan Türkçe bir kitap bile yayınlanmıştır. REDUCE, 640 K Byte'lik IBM uyumlu kişisel bilgisayarlardan CRAY süperbilgisayarlarına kadar uzanan bir platforma uygulanmıştır. CP/M işletim sistemini kullanan 64 KByte'lik bilgisayarlar için geliştirilen, MULISP diline dayalı olan MUMATH, daha sonra kişisel bilgisayarlara uygulanmıştır. Bu sistemin grafik özellikler ve bazı sayısal hesap olanakları da içeren, ancak programlama olanakları sınırlı olan bir şekli DERIVE adı ile geliştirilmiştir, kullanılan simgesel hesap algoritmaları bakımından etkin, küçük sistemlere uygunluğu ve ucuzluğu nedeniyle çok yaygın olarak kullanılan bir diğer sistemdir.

Simgesel programlama sistemleri, uzun ve karmaşık hesapların yapılması, cebirsel, sayısal ve grafik özelliklerinin birleştirilmesi nedeniyle araştırmalarda önemli kolaylıklar sağlamış, eğitimde çığır açmış, mühendislik ve fen disiplinlerindeki bilgisayar öğretimini programlama eğitimi yerine paket programlardan yararlanma gibi daha çağdaş bir yaklaşıma yönlendirirken programlama mantığını da basit bir düzeyde verilmesine olanak sağlamıştır. Giderek artan bir kullanım alanı bulacağı açıktır.



KAYNAKLAR

LAnthony C. Hearn, "REDUCE User's Manual Version 3.4" CP78 Sayılı RAND Yayını, Santa Monica, California (Temmuz 1991 baskısı)

2. Gerhard Rayna, "REDUCE, Sohvvare lor Algebraic Computation" Springer Verlag Yayınevi, New York, (1987).

3. Stephen Vvolfram, "Mathematica, A System lor Doing Mathematics by Computer" 2. Baskı, Addison Vvesley Yayınevi, Redvvood City, California, (1991).

4. Roman Maeder, "Programming in Mathematica" Addison Vvesley Yayınevi, Redvvood City, California (1989).

5. J. F. Ogilvie "Computer Algebra İn Modern Physlcs" Computers in Physics, Cilt 3, Sayı 1, s 66-74 (1989).

6. V. Haktanır, B. Şahin ve E. Kıral, "MATHCAD ve REDUCE" Ekonomist Yayınevi, Ankara (1991).