BİRİNCİ MERTEBEDEN DOĞRUSAL DİFERENSİYEL DENKLEMLERE DEDAKTİF YAKLAŞIM

A DEDUCTIVE APPROACH TO FIRST ORDER LINEAR DIFFERENTIAL EQUATIONS

TOLGA GÜYER

Hacettepe Üniversitesi
Fen Bilimleri Enstitüsü Yönetmeliğinin
Matematik Anabilim Dalı İçin Öngördüğü
BİLİM UZMANLIĞI TEZİ
olarak hazırlanmıştır.

1996

Fen Bilimleri Enstitüsü Müdürlüğüne

Bu çalışma jürimiz tarafından **MATEMATİK ANABİLİM DALI**'nda BİLİM UZMANLIĞI TEZİ olarak kabul edilmiştir.

Başkan :.....Doç.Dr. L.M.BROWN.....

Üye :.....Prof.Dr. Şeref MİRASYEDİOĞLU.....

Üye :.....Doç.Dr. Aydın TİRYAKİ.....

ONAY

Bu tez 10 /09 / 1996 tarihinde Enstitü Yönetim Kurulunca belirtilen yukarıdaki jüri üyeleri tarafından kabul edilmiştir.

10 /09/ 1996

Prof. Dr. Gültekin GÜNAY
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRÜ

ÖZET

Birinci mertebeden doğrusal diferensiyel denklemler, birinci dereceden mantık yapısı içinde incelenerek, sembolik çözümleri dedaktif yaklaşımla araştırılmıştır. Dedaktif yaklaşımda, yönlü graf tekniğine dayalı olarak oluşturulan clause kümesinden, resolution ilkesi ile elde edilen sonuçların doğruluğu, geliştirilen önteorem ve teoremlerle ispatlanmıştır.

ABSTRACT

First order linear differential equations have been studied within the context of first order logic and a deductive approach to their symbolic solutions has been investigated. The correctness of the deductive approach, which obtains solutions by applying the resolution principle to a clause set obtained using directed graph techniques, has been established in a Lemma and Theorem.

TEŞEKKÜR

Çalışmam süresince yardımlarını gördüğüm, bilgi ve deneyiminden yararlandığım, tez yöneticim ve danışmanım sayın Prof. Dr. Şeref Mirasyedioğlu'na, çalışmalarımıza gösterdiği yakın ilgi ve önerilerinden ötürü sayın Doç. Dr. L. M. Brown'a ve program geliştirme aşamasındaki yardımları için sayın Araş. Gör. Onur Kıymaz'a içtenlikle teşekkür ederim.

İÇİNDEKİLER DİZİNİ

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

1960'lı yılların ikinci yarısına kadar bilgisayarın matematik alanındaki işlevi, mühendislik dallarında karşılaşılan temel problemlerin çözümü ve karmaşık hesaplamaların yapılması ile sınırlıydı. Ancak 1930 yılında Herbrand'ın, teoremlerin mantıksal ortamda mekanik olarak ispatlanmasına yönelik yaptığı bir çalışma, 1960 yılında Gilmore'un dijital bilgisayarlar üzerinde geliştirdiği algoritmalara taban olması özelliği ile önem kazanmıştır. Özellikle bu tür çalışmalar, sembolik heaplama tekniklerine yeni yaklaşımlar sağlamıştır (Davis and Putnam 1960).

1965 yılında Robinson'un geliştirdiği bir sonuç çıkarma mekanizması olan *resolution kuralı*'nın ortaya çıkması ile mekanik teorem ispatlama ve sembolik hesaplama konularındaki çalışmalar hız kazanmıştır. Günümüzde de bu alandaki çalışmalar, gelişen bilgisayar teknolojisine paralel olarak devam etmektedir.

Sembolik hesaplama, matematiksel kavramlar üzerinde analitik olarak işlem yapılabilmesini sağlayan yöntemleri kapsar. Bu yöntemler, temelde *resolution ilkesi* ve *unification teoremi* üzerine kurulmuştur. Tez çalışmamızda, birinci mertebeden doğrusal diferensiyel denklemlerin dedaktif olarak sembolik hesaplama tekniği ile çözülmeleri gerçekleştirilmiştir. Bu amaçla;

i. Birinci mertebeden doğrusal diferensiyel denklemin genel formu, birinci dereceden mantık yapısı içinde tanımlanmıştır.

ii. Bu yapı içersinde, diferensiyel denklemin çözümleri sınıflandırılmıştır.

iii. Çözümlere karşılık gelen temel clause'lar yazılarak, diferensiyel denklemi temsil eden bir clause kümesi oluşturulmuştur.

iv. Oluşturulan temel clause kümesinden resolution sonucunda elde edilen çözümlerin doğruluğu, kurulan önteorem ve teoremlerle ispatlanmıştır.

Ele alınan konular, beş bölüm altında incelenmiştir.

Birinci bölüm, giriştir.

İkinci bölümünde, birinci dereceden mantık tanıtılmış ve Herbrand Teoremi ile Skolem standart formu üzerinden resolution ilkesine geçiş yapılmıştır.

Üçüncü bölümde, birinci dereceden mantığın program analizine uygulanması ele alınarak, bir programın sonlanmasına ilişkin bir önteorem (Program Termination Lemma) ile sonlandığı anda sonuç elde edilmesine ilişkin bir teorem (Program Response Theorem) verilmiştir.

Dördüncü bölümde, ele alınan diferensiyel denklemin matematiksel tanımı yapılarak genel çözümünün elde edilmesinde kullanılan teoremler verilmiştir.

Beşinci bölümde, adı geçen denklemin çözümüne ilişkin elde edilen sonuçlar birinci dereceden mantık yapısına taşınmış ve resolution örnekleri verilmiştir.

Ekler bölümünde ise birinci mertebeden doğrusal diferensiyel denklemler için, geliştirilen sembolik hesaplama tekniğine dayalı olarak çözüm üretebilen PROLOG programının listesi verilmiştir.

2. BİRİNCİ DERECEDEN MANTIK

2.1. Temel Kavramlar

Birinci dereceden mantığın bir konu üzerine uygulanmasında kullanılan dilin sembolleri, konunun niteliğine göre değişkenlik göstermektedir. Ancak kullanılan temel kavramlar, *birinci dereceden dil* olarak adlandırılan bu dillerin tümünde ortaktır. Bu kavramları aşağıdaki gibi sıralayabiliriz:

(i) Değişken sembolleri.

(ii) Sabit sembolleri.

(iii) Fonksiyon sembolleri.

(iv) Predicate sembolleri.

(v) Bağlaç sembolleri : "$\neg$", "$\wedge$", "$\vee$", "$\rightarrow$", "$\leftrightarrow$".

(vi) Niceleyici sembolleri : "$\forall$", "$\exists$".

(vii) Noktalama sembolleri : "(", ")", ",".

Değişkenler ve sabitler, birinci dereceden mantığın en küçük birimleridir. Fonksiyon ve predicate sembolleri, değişken, sabit ya da fonksiyon sembollerini bileşen olarak bulundurabilirler.

Bağlaç sembollerinin anlamları, önermeler mantığındaki ile aynıdır. "$\neg$" : 'değil' sembolü, "$\wedge$" : 've' sembolü, "$\vee$" : 'veya' sembolü, "$\rightarrow$" : 'gerektirme' sembolü ve "$\leftrightarrow$" : 'çift yönlü gerektirme' sembolüdür. Niceleyici sembollerinden "$\forall$" evrensel niceleyiciyi, "$\exists$" ise varlık niceleyicisini sembolize etmektedir.

Bu bölümde verilen tanımlarda, predicate sembolü olarak "p", "q", "r"; fonksiyon sembolü olarak "f", "g", "h", "k" ,"j"; değişken sembolü olarak "x", "y", "z", "u", "v" ve sabit sembolü olarak "a", "b", "c" harfleri kullanılacaktır.

Çalışmamıza özgü olan diğer predicate, fonksiyon, değişken ve sabit sembolleri daha sonra verilecektir.

2.1.1. Tanım : *Terim,* özyinelemli olarak aşağıdaki gibi tanımlanır:

(i) Bir sabit, bir terimdir.

(ii) Bir değişken, bir terimdir.

(iii) f bir n-bileşenli fonksiyon sembolü ve $t_1, \ldots, t_n$ terimler olmak üzere $f(t_1, \ldots, t_n)$ bir terimdir.

Eğer bir terim hiçbir değişken içermiyorsa *kapalı terim* adını alır.

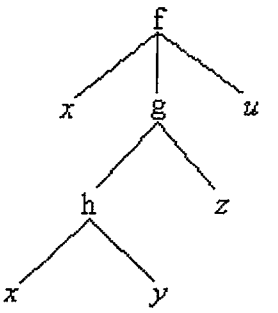
2.1.1. Örnek : f ve g iki fonksiyon sembolü; a, b sabitler ve x, y değişkenler olmak üzere aşağıdakiler birer terimdir:

$$g(f(x), g(x,y))$$

$$g(a,g(a,g(a,b))).$$

Terimlerin gösteriminde karşılaşılan en önemli güçlük, büyük terimlerde iç içe yazılmalardan kaynaklanan parantez karmaşasıdır. Bu gibi durumlarda, terimlerin ağaç yapısı biçimindeki gösterimleri ile, onların oldukça açık ve anlaşılabilir olmalarını sağlayabiliriz. Örnek olarak, $f(x,g(h(x,y),z),u)$ teriminin ağaç yapısı biçimindeki gösterimi Şekil 2.1.'de verilmiştir.

Ağaç yapısıyla gösterilen bir t terimini gözönüne alırsak, yapının en üstünde yeralan terim, t'nin *kökü* adını alır. Kökün alt kısımlarında yeralan diğer terimler ise *düğüm* adını alır. Terimin özyinelemli tanımını düşünecek olursak, bu yapıda bulunan düğümler de, kendi altlarında yeralan düğümlerin oluşturdukları ağaç yapılarının kökleri olacaktır. Diğer bir deyişle, bir terimin ağaç yapısındaki gösteriminde, uç noktalarda yeralan düğümlerin dışındaki tüm düğümler, aynı zamanda birer köktür.



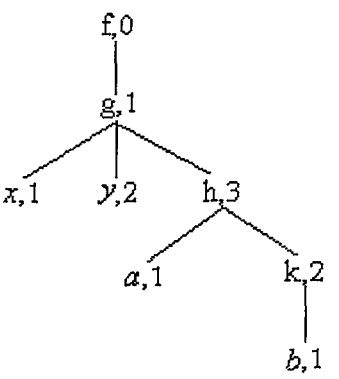
Şekil 2.1. Terimlerin ağaç yapısı biçimindeki gösterimleri.

2.1.2. Tanım : Ağaç yapısında verilen bir terimin, kökünün ve herbir düğümünün aşağıdaki kurallar yardımı ile numaralandırılmasıyla oluşan gösterimine o terimin *belirli ağaç gösterimi* adı verilir.

1. Terimin kökü 0 tamsayısı ile numaralandırılır. Kökün altında yeralan herbir düğüm soldan sağa doğru artan sıra ile 1'den başlayan tamsayılarla numaralandırılır.

2. Bir n tamsayısı ile numaralandırılmış olan bir düğümün altında yeralan diğer düğümler, soldan sağa doğru artan sıra ile 1'den başlayan tamsayılarla numaralandırılır.

2.1.3. Örnek : f(g(*x*,*y*,h(*a*,k(*b*)))) teriminin belirli ağaç gösterimi Şekil 2.2.'de verilmiştir.



Şekil 2.2. Terimlerin belirli ağaç gösterimleri.

2.1.3. Tanım : Bir (*iyi tanımlı*) *formül* özyinelemli olarak aşağıdaki gibi tanımlanır :

(i) p bir n-bileşenli predicate sembolü ve $t_1,...,t_n$ terimler olsun. Bu durumda $p(t_1,...t_n)$ bir formüldür. (Bu formüle *atomik formül* adı verilir.)

(ii) F ve G iki formül olsun. Bu durumda $(\neg F)$, $(F \wedge G)$, $(F \vee G)$, $(F \rightarrow G)$ ve $(F \leftrightarrow G)$ birer formüldür.

(iii) F bir formül ve x bir değişken olsun. Bu durumda $\forall x(F)$ ve $\exists x(F)$ birer formüldür.

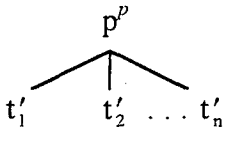
2.1.3. Örnek : p ve q predicate sembolleri, f bir fonksiyon sembolü ve x, y değişkenler olmak üzere aşağıdakiler birer formüldür :

$$\forall x \exists y (p(f(x),y) \rightarrow q(y))$$

$$\neg (\exists x (p(x,y) \wedge q(f(y))))$$

Terimlerin olduğu gibi atomik formüllerin de belirli ağaç gösterimleri tanımlanabilir.

2.1.4. Tanım : p bir n-bileşenli predicate sembolü ve $t_1, t_2, \ldots, t_n$ terimler olmak üzere $p(t_1, t_2, \ldots, t_n)$ atomik formülünü gözönüne alalım. $t_1', t_2', \ldots, t_n'$ sırasıyla $t_1, t_2, \ldots, t_n$ terimlerinin belirli ağaç gösterimleri olsun. Bu durumda, Şekil 2.3.'de verilen yapıya, $p(t_1, t_2, \ldots, t_n)$ atomik formülünün *belirli ağaç gösterimi* denir. Bu gösterimde p^p, belirli ağaç yapısının *kök predicate*'i adını alır. Burada p'nin üzerinde yeralan "p" sembolü, p'nin bir predicate olduğunu vurgulamaktadır.



Şekil 2.3. Atomik formüllerin belirli ağaç gösterimleri.

2.1.5. Tanım : F bir formül olmak üzere, $\forall x$'in -$\exists x$'in- $\forall x(F)$ -$\exists x(F)$- içindeki *alanı* F'dir. Bir değişkenin bir formül içersinde *sınırlı* olarak bulunması demek, ya o değişkenin bir niceleyicinin ardından gelmesi, ya da bir niceleyicinin alanı içersinde bulunması demektir.

Eğer bir değişken bir formül içersinde sınırlı değilse *serbest* adını alır.

2.1.6. Tanım : Hiçbir serbest değişken içermeyen formüle *kapalı formül* adı verilir.

2.1.7. Tanım : Bir atom ya da o atomun değiline bir *literal* adı verilir. *Pozitif literal* atomdur. *Negatif literal* ise atomun değilidir.

2.1.8. Tanım : Her bir L_i bir literal ve $x_1,...,x_s$; $L_1 \vee L_2 \vee ... \vee L_m$ içindeki değişkenler olmak üzere,

$$\forall x_1 ... \forall x_s (L_1 \vee ... \vee L_m)$$

biçimindeki formüle bir *clause* adı verilir.

2.1.4. Örnek : Aşağıdaki formüller birer clause'dur:

$$\forall x \forall y \forall z \; (p(x,z) \vee q(x,y) \vee \neg r(y,z))$$

$$\forall x \forall y (\neg p(x,y) \vee r(f(x,y),a))$$

2.2. Birinci Dereceden Mantıkta Bir Formülün Yorumlanması

2.2.1. Tanım : Boş olmayan bir D kümesine bir *yorumlama bölgesi* adı verilir. { **T** , **F** } kümesi ise sırasıyla "doğru" ve "yanlış" doğruluk değerlerinin kümesini göstermektedir.

2.2.2. Tanım : Birinci dereceden mantıkta bir F formülünün *yorumlanması*, bir D yorumlama bölgesi ile F içindeki her sabit, fonksiyon sembolü ve predicate sembolüne aşağıdaki biçimde bir 'değer' atanması biçiminde gerçekleştirilir :

(i) Her sabite D'nin bir elemanı atanır.

(ii) $D^n = \{(x_1,...,x_n) \mid x_1 \in D,...,x_n \in D\}$ olmak üzere, her n-bileşenli fonksiyon sembolüne D^n'den D'ye bir dönüşüm atanır.

(iii) Her n-bileşenli predicate sembolüne D^n'den { **T** , **F** } kümesine bir dönüşüm atanır.

Bu tanımdan sonra, niceleyicilerin kazandıkları anlamlar daha açık olarak şu şekilde ifade edilebilir : Bir F formülünün, bir D yorumlama bölgesi üzerinde yorumlanmasında $\forall x$, "D içindeki her x elemanı için" ve $\exists x$, "D içinde bir x elemanı vardır." anlamını taşır.

Genel olarak bir D yorumlama bölgesi üzerinde bir formülün herhangi bir yorumlamasında formüle **T** ya da **F** değerlerinden birisi aşağıdaki kurallar yardımı ile atanır :

1. Eğer G ve H formüllerinin doğruluk değerleri **T** ya da **F** olarak hesaplanmışsa, $\neg$G, G$\wedge$H, G$\vee$H, G$\rightarrow$H ve G$\leftrightarrow$H formüllerinin doğruluk değerleri Çizelge 2.1. kullanılarak hesaplanabilir.

2. Eğer G formülünün doğruluk değeri D içindeki her d elemanı için **T** oluyorsa, $\forall x$G formülünün doğruluk değeri **T**'dir; aksi durumda $\forall x$G formülü **F** değerini alır.

3. Eğer G formülünün doğruluk değeri D içindeki en az bir d elemanı için **T** oluyorsa, $\exists x$G formülünün doğruluk değeri **T**'dir; aksi durumda $\exists x$G formülü **F** değerini alır.

2.2.4. Tanım : Kapalı bir G formülü, bir I yorumlamasında **T** değerini alıyorsa I'ya G'nin *modeli* adı verilir.

G	H	$\neg$G	G$\wedge$H	G$\vee$H	G$\rightarrow$H	G$\leftrightarrow$H
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

Çizelge 2.1. Formüllerin doğruluk değerlerinin hesaplanması.

2.2.4. Tanım : G bir formül ve I bir yorumlama olsun. Bu durumda;

- Eğer $\exists$(G) I'da doğru ise G formülü I'da *sağlanabilir* denir.
- Eğer $\forall$(G) I'da doğru ise G formülü I'da *geçerlidir* denir.
- Eğer $\exists$(G) I'da yanlış ise G formülü I'da *sağlanamaz* denir.
- Eğer $\forall$(G) I'da yanlış ise G formülü I'da *geçersizdir* denir.

2.2.1. Örnek : $\forall x \exists y$ p(x,y) formülünü gözönüne alalım. I yorumlamasını, D bölgesi negatif olmayan tamsayıların kümesine ve p predicate'i "<" bağıntısına karşılık gelecek biçimde seçelim. Verilen formülü I yorumlamasına göre "her negatif olmayan tamsayı için bu tamsayıdan daha büyük olan en az bir negatif olmayan tamsayı vardır." biçiminde ifade edebiliriz. Bu durumda I'nın verilen formül için bir model olacağı açıktır. Diğer yandan I, $\exists y \forall x$ p(x,y) formülü için bir model değildir.

2.2.5. Tanım : S kapalı formüllerin bir kümesi ve I bir yorumlama olsun. Eğer I, S'deki her formül için bir model oluyorsa I'ya S'nin bir *modeli* denir.

2.2.6. Tanım : S kapalı formüllerin bir kümesi olsun. Bu durumda;

- Eğer S için model olan en az bir I yorumlaması varsa S *sağlanabilir* denir.
- Eğer S için her I yorumlaması bir model oluyorsa S *geçerlidir* denir.
- Eğer S için model olan hiçbir I yorumlaması yoksa S *sağlanamaz* denir.
- Eğer S için model olmayan en az bir I yorumlaması varsa S *geçersizdir* denir.

2.2.7. Tanım : S kapalı formüllerin bir kümesi ve F kapalı bir formül olsun. Eğer her I yorumlaması için, I'nın S için bir model olması, I'nın F için de bir model olmasını gerektiriyorsa, F'ye S'nin *mantıksal sonucu* denir.

2.2.1. Önerme : S kapalı formüllerin bir kümesi ve F kapalı bir formül olsun. Bu durumda F'nin S'nin mantıksal sonucu olması için gerekli ve yeterli koşul, $S \cup \{\neg F\}$ kümesinin sağlanamaz olmasıdır. (Chang and Lee, 1973)

2.2.2. Örnek : S={p(a), $\forall x$ (p(x)→q(x))} ve F=q(a) olsun. S'nin mantıksal sonucunun F olduğunu göstereceğiz. I, S için model olan herhangi bir yorumlama olsun. Bu durumda p(a), I'da doğrudur. Bununla beraber $\forall x$ (p(x)→q(x))'de I'da doğru olduğundan, p(a)→q(a)'da doğru olacaktır. Buradan q(a)'nın da I'da doğru olduğu sonucu çıkar. O halde, Tanım 2.2.7. gereğince, F, S'nin mantıksal sonucu olur.

2.3. Prenex Normal Formu

2.3.1. Tanım : Birinci dereceden mantıkta bir F formülünün *Prenex normal formu*'nda olması için gerekli ve yeterli koşul F'nin

$$Q_1x_1 \ldots Q_nx_n \text{ (M)}$$

biçiminde olmasıdır. Burada herbir (Q_ix_i), i=1,...,n, ($\forall x_i$) ya da ($\exists x_i$) niceleyicilerinden birisi ve M hiçbir niceleyici içermeyen bir formüldür. Burada, $Q_1x_1 \ldots Q_nx_n$ bölümüne formülün *niceleyiciler bölümü*, M'ye ise formülün *matris bölümü* adı verilir.

2.3.1. Örnek : Aşağıdaki formüller Prenex normal formundadır :

$$\forall x \, \forall y \, (p(x) \rightarrow q(x,y))$$

$$\forall x \, \forall y \, \exists z \, (q(x,f(y)) \wedge r(z))$$

Birinci dereceden mantıkta herhangi bir formül Prenex Normal Formuna dönüştürülebilir. Bunun için, F ve G, x değişkenini bulunduran formüller, H, x değişkenini bulundurmayan bir formül, z, F ve G formülleri içinde bulunmayan bir değişken, Q, $\forall$ ya da $\exists$ niceleyicilerinden birisi ve Q_1 ve Q_2, ikisi birbirinden farklı olmak koşuluyla, $\forall$ ya da $\exists$ niceleyicilerinden birisi olmak üzere aşağıda verilen kurallar kullanılır:

1. $F \leftrightarrow G = (F \rightarrow G) \wedge (G \rightarrow F)$

2. $F \rightarrow G = \neg F \vee G$

3. $\neg (\neg F) = F$

4. $\neg(F \vee G) = \neg F \wedge \neg G$

5. $\neg(F \wedge G) = \neg F \vee \neg G$

6. $\neg(\forall x (F)) = \exists x (\neg F)$

7. $\neg(\exists x (F)) = \forall x (\neg F)$

8. $\forall x (F) \wedge \forall x(G) = \forall x (F \wedge G)$

9. $\exists x (F) \vee \exists x(G) = \exists x (F \vee G)$

10. $Qx (F) \vee H = Qx (F \vee H)$

11. $Qx (F) \wedge H = Qx (F \wedge H)$

12. $Q_1x (F) \vee Q_2x (G) = Q_1x\ Q_2z (F \vee G)$

13. $Q_1x (F) \wedge Q_2x (G) = Q_1x\ Q_2z (F \wedge G)$

Bu kurallarda kullanılan eşitlik simgesi, eşitliğin sol ve sağ tarafındaki formüllerin aynı doğruluk değerini taşıdıklarını göstermektedir. Diğer bir deyişle, F ve G iki formül olmak üzere, F=G olması için gerekli ve yeterli koşul, F ve G'nin her yorumlama altında aynı doğruluk değerine sahip olmalarıdır.

2.3.2. Örnek : $\forall x\forall y(\exists z\ p(x,y,z)\wedge(\exists u\ q(x,u)\rightarrow\exists y\ r(x,y)))$ formülünü aşağıdaki biçimde Prenex normal formuna dönüştürebiliriz:

$\forall x\forall y(\exists z\ p(x,y,z)\wedge(\exists u\ q(x,u)\rightarrow\exists y\ r(x,y)))$

$= \forall x\forall y(\exists z\ p(x,y,z)\wedge(\neg(\exists u\ q(x,u))\vee\exists y\ r(x,y)))$ *Kural 2'nin uygulanması.*

$= \forall x\forall y(\exists z\ p(x,y,z)\wedge(\forall u(\neg q(x,u))\vee\exists y\ r(x,y)))$ *Kural 7'nin uygulanması.*

$= \forall x\forall y(\exists z\ p(x,y,z)\wedge(\forall u\exists y(\neg q(x,u)\vee r(x,y))))$ *Kural 10'un uygulanması.*

$= \forall x \forall y \exists z \exists u \exists v(p(x,y,z) \wedge (\neg q(x,u) \vee r(x,v)))$ *Kural 12'nin uygulanması.*

Bulunan son formül, verilen formülün Prenex normal formunda ifade edilmiş biçimidir.

2.4. Skolem Standart Formu

2.4.1. Tanım : F, Prenex normal formunda bir formül olsun. F formülünün *arakesit normal formunda* olması için gerekli ve yeterli koşul F'nin $F_1 \wedge F_2 \wedge ... \wedge F_n$ biçiminde olmasıdır.

2.4.1. Örnek : Örnek 2.3.2.'de elde edilen Prenex normal formundaki

$$\forall x \forall y \exists z \exists u \exists v(p(x,y,z) \wedge (\neg q(x,u) \vee r(x,v)))$$

formülünü $\wedge$ bağlacının $\vee$ bağlacına dağılması özelliğini kullanarak,

$$\forall x \forall y \exists z \exists u \exists v((p(x,y,z) \wedge \neg q(x,u)) \vee (p(x,y,z) \wedge r(x,v)))$$

biçiminde arakesit normal formunda yazabiliriz.

2.4.2. Tanım : F, arakesit normal formunda bir formül olsun. F'nin *Skolem standart formunda* olması için gerekli ve yeterli koşul, Q_1, Q_2,...,Q_n'lerin herbiri birer evrensel niceleyici olmak üzere F'nin,

$$Q_1x\ Q_2x\ ...\ Q_nx\ (F)$$

biçiminde olmasıdır.

Bundan sonra Skolem standart formu yerine kısaca 'standart form' deyimi kullanılacaktır.

Birinci dereceden mantıkta herhangi bir formül standart forma dönüştürülebilir. Bunun için, formülün varlık niceleyicilerinden bağımsız olarak ifade edilmesi gerekmektedir. Bunu, verilen formülün niceleyiçiler bölümünde herbir varlık niceleyicisi için, o varlık niceleyicisinin alanında bulunan değişkeni, bu niceleyiciden

önce gelen evrensel niceleyicilerin alanlarında bulunan değişkenlerin bir fonksiyonu ile değiştirerek yapabiliriz. Burada seçilen fonksiyon sembolü, formülün matris bölümünde bulunmayan bir fonksiyon sembolü olmalıdır. Bu biçimdeki fonksiyonlara *Skolem Fonksiyonları* adı verilir. (Chang and Lee, 1973)

2.4.2. Örnek : $\forall x\forall y\exists z\exists u\forall v((\neg p(h(x),y,z)\vee q(x,u))\wedge r(u,v))$ formülünü aşağıdaki biçimde standart forma dönüştürebiliriz:

$\forall x\forall y\exists z\exists u\forall v\ ((\neg p(h(x),y,z)\vee q(x,u))\wedge r(u,v))$

$=\forall x\forall y\exists z\forall v\ ((\neg p(h(x),y,z)\vee q(x,f(x,y)))\wedge r(f(x,y),v))$

$=\forall x\forall y\forall v\ ((\neg p(h(x),y,g(x,y))\vee q(x,f(x,y)))\wedge r(u,v))$

Bulunan son formül, verilen formülün Skolem standart formunda ifade edilmiş biçimidir. Burada f ve g fonksiyonları birer Skolem fonksiyonudur.

2.4.3. Tanım : Standart formdaki $F = \forall x_1\forall x_2...\forall x_n\ (F_1\wedge F_2\wedge...\wedge F_k)$ formülünü gözönüne alalım. $S^F=\{ F_1,F_2,...,F_k\}$ kümesine F formülünün *clause kümesi* adı verilir.

2.4.1. Teorem : S^F, standart formdaki F formülünün clause kümesi olsun. Bu durumda F formülünün sağlanamaz olması için gerekli ve yeterli koşul S^F'nin sağlanamaz olmasıdır. (Chang and Lee, 1973)

2.5. Herbrand Teoremi

Mekanik teorem ispatlama konusunda oldukça önemli bir atılım, 1930 yılında Herbrand tarafından yapılmıştır. Tanım 2.2.6. gereğince, geçerli bir formülün tüm yorumlamalar altında doğru değerini aldığını biliyoruz. Herbrand, verilen formülün yanlış değerini alacağı bir yorumlama arayan algoritmayı geliştirmiştir. Eğer formül geçerli ise, bu biçimde bir yorumlama elde edilememekte ve algoritma sonlu bir adımda durmaktadır. Bu yöntem, birçok modern otomatik ispatlama prosedürüne temel oluşturmuştur. (Loveland, 1978)

2.5.1. Tanım : Verilen bir S clause kümesinin H(S) ile gösterilen *Herbrand evrenseli,* indirgemeli olarak aşağıdaki biçimde tanımlanır:

(i) S'deki herhangi bir clause'da yeralan herhangi bir sabit sembolü, H(S)'nin bir elemanıdır. Eğer S'deki hiçbir clause sabit sembolü bulundurmuyorsa H(S), *"a"* sabitini içerir.

(ii) Eğer f, S'de n-bileşenli bir fonksiyon sembolü ve $t_1, t_2, ..., t_n$, H(S)'nin elemanları iseler, $f(t_1, t_2, ..., t_n)$'de H(S)'nin bir elemanıdır.

2.5.1. Örnek : $S=\{p(f(x)), q(g(b,x))\}$ için $H(S)=\{b, f(b), g(b,b), f(f(b)), g(b,f(b)),...\}$ biçimindeki sonsuz küme olur.

2.5.2. Tanım : S bir clause kümesi olmak üzere, H(S)'nin bir elemanına S'nin *Herbrand terimi* ya da *temel terimi* denir. S'nin bir *temel clause'u* ise, S'deki bir clause'un değişkenin S'nin temel terimleri ile düzgün olarak değiştirilmesi ile elde edilir. Burada düzgün kelimesi ile anlatılmak istenen, değiştirme işlemi sırasında, farklı konumlarda bulunan aynı değişkenlerin, aynı temel terimlerle değiştirilmeleridir. Bu biçimde, bir S clause kümesinden türetilen tüm temel clause'ların kümesine S'nin *temel clause kümesi* denir ve S_H ile gösterilir. Bir F formülünün temel clause kümesi ise F'nin clause kümesinin temel clause kümesidir ve $(S^F)_H$ ile gösterilir.

2.5.3. Örnek : $S=\{p(x),q(x,y)\}$ olarak veriliyor. Bu durumda $H(S)=\{a\}$ ve $S_H=\{p(a),q(a,a)\}$ olacaktır.

2.5.4. Teorem : (Herbrand Teoremi) F Skolem standart formunda bir formül olsun. F'nin sağlanamaz olması için gerekli ve yeterli koşul F'nin sağlanamaz olan sonlu bir temel clause kümesinin bulunmasıdır. (Yani $(S^F)_H$'nin T gibi sonlu bir altkümeye sahip olmasıdır.)

2.5.5. Örnek : $F=\forall z((p(a)\vee p(b))\wedge(p(a)\vee q(b))\wedge\neg p(z))$ formülü veriliyor. Buradan $S^F=\{(p(a)\vee p(b)),(p(a)\vee q(b)),\neg p(z)\}$ olarak bulunur. $T\subset(S^F)_H$ olmak üzere $T=\{(p(a)\vee p(b)),\neg p(a),\neg p(b)\}$ kümesinin sağlanamaz olduğu açıktır. O halde Herbrand Teoremi gereğince S^F kümesi sağlanamazdır.

2.6. Dönüşümler

2.6.1. Tanım : i=1,...,n için v_i'ler birbirlerinden farklı değişkenler ve herbir t_i bir terim olmak üzere $\{t_1 / v_1 ,..., t_n / v_n\}$ biçimindeki sonlu kümeye bir *dönüşüm* adı verilir. Dönüşümler σ_i , i=1,..,n , sembolleri ile gösterilecektir.

Eğer $t_1,...,t_n$ temel terimler ise dönüşüm *temel dönüşüm* adını alır. Hiçbir eleman içermeyen dönüşüme ise *boş dönüşüm* adı verilir ve $\varnothing$ simgesi ile gösterilir.

2.6.1. Örnek : Aşağıdakiler birer dönüşümdür:

$$\{u/x,\ f(y)/z\}$$

$$\{g(f(a,b))/x,\ a/y,\ h(a,y)/z\}$$

2.6.2. Tanım : $\sigma = \{ t_1 / v_1 ,..., t_n / v_n\}$ bir dönüşüm ve F bir formül olsun. Bu durumda $F\sigma$ formülü, F içindeki herbir v_i , i=1,..,n , değişkeninin t_i terimi ile değiştirilmesi ile elde edilen formüldür ve F'nin *örneklemesi* adını alır.

2.6.3. Örnek : $F = \neg q(x,f(x,z))\vee p(u,g(h(x,y)),z)\vee r(u,v)$ formülü ile $\sigma_1 = \{k(a,z)/x,\ y/z\}$ ve $\sigma_2 = \{h(c,k(v))/u,\ a/x\ ,\ g(x,z)/v\}$ dönüşümleri için $F\sigma_1$ ve $F\sigma_2$ aşağıdaki gibi olur:

$$F\sigma_1 = \neg q(k(a,y),f(x,y))\vee p(u,g(h(k(a,y),y)),y)\vee r(u,v)$$

$$F\sigma_2 = \neg q(a,f(a,z))\vee p(h(c,k(g(a,z))),g(h(a,y)),z)\vee r(h(c,k(g(a,z))),\ g(a,z))$$

2.6.4. Tanım : $\sigma_1 = \{ t_1 / x_1 ,..., t_n / x_n\}$ ve $\sigma_2=\{e_1 / y_1 ,..., e_m / y_m\}$ iki dönüşüm olsun. $\sigma_1\circ\sigma_2$ ile gösterilen σ_1 ve σ_2 dönüşümlerinin bileşkesi,

$$\{t_1\sigma_2 / x_1 , \dots , t_n\sigma_2 / x_n , e_1 / y_1 , \dots , e_m / y_m\}$$

kümesinden, $t_i\sigma_2=x_i$ olan $t_i\sigma_2 / x_i$ elemanları ile $y_j\in\{x_1,...,x_n\}$olan e_j / y_j elemanlarının silinmesi ile elde edilir.

2.6.1. Önerme : σ bir dönüşüm olsun. Bu durumda $\sigma\circ\varnothing = \varnothing\circ\sigma = \sigma$ olur.

2.6.3. Örnek : $\sigma_1=\{g(h(x))/y, u/v\}$, $\sigma_2=\{f(u)/x, c/y\}$ ve $\sigma_3=\{z/u, g(z)/v\}$ olmak üzere, $(\sigma_1 \circ \sigma_2) \circ \sigma_3$ dönüşümünü aşağıdaki biçimde bulabiliriz :

$$(\sigma_1 \circ \sigma_2) \circ \sigma_3 = \{g(h(f(u)))/y, u/v, f(u)/x, \underline{c/y}\} \circ \sigma_3$$

$$= \{g(h(f(u)))/y, u/v, f(u)/x\} \circ \{z/u, g(z)/v\}$$

$$= \{g(h(f(z)))/y, z/v, f(z)/x, z/u, \underline{g(z)/v}\}$$

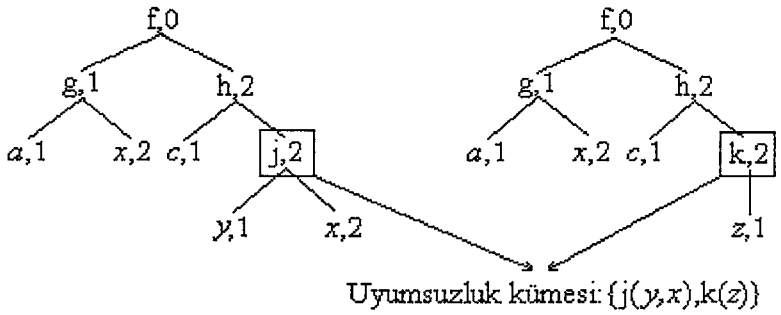
$$= \{g(h(f(z)))/y, z/v, f(z)/x, z/u\}$$

Burada altı çizilerek belirtilen elemanlar, tanım gereğince dönüşüm kümesinden silinen elemanlardır.

2.7. Unification Algoritması ve Unification Teoremi

2.7.1. Tanım : t_1 ve t_2 iki terim olsun. d_1 ve d_2, sırasıyla t_1 ve t_2'nin belirli ağaç gösterimlerinde birbirlerinden farklı iki düğüm olsun. Eğer d_1 ve t_1'in kökünü birleştiren yol ile d_2 ve t_2'nin kökünü birleştiren yollar aynı yollar ise, t_1 teriminin d_1 düğümünün kök durumunda olduğu alt terimi ile t_2 teriminin d_2 düğümünün kök durumunda olduğu alt teriminin oluşturdukları kümeye, t_1 ve t_2'nin *uyumsuzluk kümesi* adı verilir. (Fitting, 1990)

2.7.1. Örnek : f(g(*a*,*x*),h(*c*,j(*y*,*x*))) ve f(g(*a*,*x*),h(*c*,k(*z*))) terimlerinin uyumsuzluk kümesini Şekil 2.4.'deki gibi bulabiliriz.



Şekil 2.4. Uyumsuzluk kümesinin bulunması.

2.7.2. Tanım : σ bir dönüşüm ve t_1, t_2 iki terim olsun. σ'nın, t_1 ve t_2'nin bir *unifier*'i olması için gerekli ve yeterli koşul $t_1\sigma=t_2\sigma$ olmasıdır.

Tanım : σ, t_1 ve t_2 terimlerinin bir unifier'i olsun. σ'nın t_1 ve t_2'nin *en genel unifier*'i olması için gerekli ve yeterli koşul t_1 ve t_2'nin her σ_1 unifier'i için $\sigma_1=\sigma\circ\sigma_2$ olacak biçimde bir σ_2 dönüşümünün bulunmasıdır.

Pratik olarak tanım, verilen terimlerin en genel unifier'inin bulunmasında yetersiz kalacaktır. Bunun için, *unification algoritması* adı verilen aşağıdaki algoritma kullanılabilir :

Unification Algoritması

t_1 ve t_2 terimler olmak üzere, $W=\{t_1, t_2\}$ kümesini gözönüne alalım.

1.Adım : $k=0$, $W_k=W$ ve $\sigma_k=\varnothing$ olarak alınır.

2.Adım : Eğer W_k bir tek terim içeriyorsa algoritma sonlanır; σ_k, W için en genel unifier'dir. Aksi durumda W_k'nın uyumsuzluk kümesi D_k bulunur.

3.Adım : D_k kümesi; v_k, e_k teriminde bulunmayan bir değişken sembolü olmak üzere, v_k ve e_k elemanlarını bulunduruyorsa 4'ncü adıma geçilir. Aksi durumda, yani D_k içersindeki e_k terimi, yine D_k içersinde bulunan bir v_k değişkenini bulunduruyorsa, algoritma sonlanır. Bu durumda W kümesinin en genel unifier'i yoktur.

4.Adım : $\sigma_{k+1}=\sigma_k\circ\{e_k / v_k\}$ ve $W\sigma_{k+1}=W_{k+1}=W_k\{e_k / v_k\}$ olarak alınır.

5. Adım : $k=k+1$ alınır ve 2'nci adıma dönülür.

2.7.2. Örnek : $W=\{p(a,x,f(g(y))) , p(z,f(z),f(u))\}$ kümesinin en genel unifier'ini bulalım:

1) $k\leftarrow 0$, $W_0=W=\{p(a,x,f(g(y))) , p(z,f(z),f(u))\}$, $\sigma_0=\varnothing$.

2) W_0 birden fazla terim içerdiğinden işlem sonlanmaz; W_0'ın D_0 uyumsuzluk kümesi bulunur: $D_0=\{a,z\}$.

Bir $v_0=z\in D_0$ değişkeni var ve bu değişken $t_0=a\in D_0$ teriminde bulunmamaktadır. Dolayısıyla bir sonraki adıma geçilebilir.

4) $\sigma_1=\sigma_0\circ\{t_0/v_0\}= \varnothing\circ\{a/z\}=\{a/z\}$,

$W_1=W_0\sigma_1=\{p(a,x,f(g(y))) , p(z,f(z),f(a))\}\circ \{a/z\}=\{p(a,x,f(g(y))) , p(a,f(a),f(u))\}$

5) k←1 ve 2'nci adıma dönülür.

6) W_1 birden fazla terim içerdiğinden işlem sonlanmaz; W_1'in D_1 uyumsuzluk kümesi bulunur: $D_1=\{x,f(a)\}$.

7) Bir $v_1=x\in D_1$ değişkeni var ve bu değişken $t_1=f(a)\in D_1$ teriminde bulunmaz. Dolayısıyla bir sonraki adıma geçilebilir.

8) $\sigma_2=\sigma_1\circ\{t_1/v_1\}= \{a/z\}\circ\{f(a)/x\}=\{a/z, f(a)/x\}$,

$W_2=W_1\sigma_2=\{p(a,x,f(g(y))),p(a,f(a),f(u))\}\circ \{f(a)/x\}=\{p(a,f(a),f(g(y))),p(a,f(a),f(u))\}$

9) k←2 ve 2'nci adıma dönülür.

10) W_2 birden fazla terim içerdiğinden işlem sonlanmaz; W_2'in D_2 uyumsuzluk kümesi bulunur: $D_2=\{g(y),u\}$.

11) Bir $v_2=u\in D_2$ değişkeni var ve bu değişken $t_2=g(y)\in D_2$ teriminde bulunmaz. Dolayısıyla bir sonraki adıma geçilebilir.

12) $\sigma_3=\sigma_2\circ\{t_2/v_2\}= \{a/z,f(a)/x\}\circ\{g(y)/u\}=\{a/z, f(a)/x, g(y)/u\}$,

$W_3=W_2\sigma_3=\{p(a, f(a),f(g(y))),p(a,f(a),f(u))\}\circ \{g(y)/u\}=\{p(a,f(a),f(g(y)))\}$

13) k←3 ve 2'nci adıma dönülür.

14) W_3 tek terim içerdiğinden σ_3, W'nin en genel unifier'idir.

Sonuç olarak, $W=\{p(a,x,f(g(y))) , p(z,f(z),f(u))\}$ kümesinin en genel unifier'i;

$$\{a/z, f(a)/x, g(y)/u\}$$

olarak bulunur.

2.7.1. Teorem : (Unification Teoremi) t_1 ve t_2 iki terim olsun. Eğer t_1 ve t_2 bir unifier'e sahip değilse algoritma 3'ncü adımda başarısızlıkla sonuçlanır. Aksi durumda

algoritma herzaman 2'nci adımda sonlanacaktır. Bu durumda elde edilen en son σ_k, t_1 ve t_2'nin en genel unifier'i olur. (Chang and Lee, 1973)

Yukarıda verilen unification algoritması, sonuç olarak iki terimin en genel unifier'ini vermektedir. Bu sebeple, *ikili unification* olarak adlandırılabilir. Ancak bu işlem, *katlı unification* adı verilen yöntemle sonlu sayıda terim içeren bir kümenin en genel unifier'ini bulmaya yönelik olarak genelleştirilebilir. Bu, ikili unification'ın ardışık olarak uygulanması ile yapılır.

Terimlerden oluşan $T=\{t_0, t_1, t_2, \ldots, t_n\}$ kümesini ve bir σ dönüşümünü gözönüne alalım. Eğer $\sigma t_0=\sigma t_1=\sigma t_2=\ldots=\sigma t_n$ oluyorsa, σ, T için bir *unifier* olur. T kümesinin böyle bir unifier'e sahip olduğunu varsayalım. Bu durumda, herbiri ikili unification algoritması kullanılarak hesaplanabilecek elemanlardan oluşmak üzere, bir dönüşümler dizisini aşağıdaki biçimde oluşturabiliriz :

σ_1, t_0 ve t_1 terimlerinin en genel unifier'idir.

σ_2, $t_0\sigma_1$ ve $t_2\sigma_1$ terimlerinin en genel unifier'idir.

σ_3, $t_0(\sigma_1 \circ \sigma_2)$ ve $t_3(\sigma_1 \circ \sigma_2)$ terimlerinin en genel unifier'idir.

.

.

.

σ_n, $t_0(\sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_{n-1})$ ve $t_n(\sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_{n-1})$ terimlerinin en genel unifier'idir.

Bu durumda $(\sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_{n-1} \circ \sigma_n)$, T kümesi için *en genel unifier* olacaktır.

2.8. Resolution İlkesi

Verilen bir S clause kümesinin sağlanamazlığının gösterilmesinde Herbrand Teoreminin uygulanması, S'deki eleman sayısı arttıkça güçleşmektedir. Çünkü böyle bir S kümesi için S_H'nin sağlanamaz olan sonlu bir altkümesinin sezgisel olarak bulunması oldukça güçtür. Böyle bir altkümenin elde edilmesi için kullanılacak

algoritmalardan da, işlem sayısının yine S'nin eleman sayısı ile doğru orantılı olarak üstel bir biçimde artması sebebiyle, etkin olarak sonuç alınamamaktadır.

1965 yılında Robinson tarafından tanıtılan Resolution ilkesi, bir S clause kümesinin sağlanamaz olduğunun gösterilmesi için doğrudan S'nin kendisine uygulanan bir yöntemdir. Temelde resolution ilkesi, Davis ve Putnam (1960) tarafından verilen *bir literal kuralı*'nın bir uzantısı niteliğindedir.

2.8.1. Tanım : Bir C clause'unu gözönüne alalım. Eğer C içinde, aynı işareti taşıyan iki ya da daha fazla literal σ gibi bir en genel unifier'e sahipse, $C\sigma$'ya C'nin bir *çarpanı* adı verilir.

2.8.2. Tanım : C_1 ve C_2 ortak değişken bulundurmayan iki clause olsunlar. L_1 ve L_2 sırasıyla C_1 ve C_2 içersinde iki literal olsun. Eğer L_1 ve $\neg L_2$, σ gibi bir en genel unifier'e sahip ise, bu durumda,

$$(C_1\sigma - L_1\sigma) \cup (C_2\sigma - L_2\sigma)$$

clause'una C_1 ve C_2'nin *ikili resolventi* denir.

2.8.3. Tanım : C_1 ve C_2 clause'larının *resolventi*, aşağıdaki ikili resolventlerden birisidir:

4. C_1 ile C_2'nin bir ikili resolventi,

5. C_1 ile C_2'nin bir çarpanının ikili resolventi,

6. C_1'in bir çarpanı ile C_2'nin ikili resolventi,

7. C_1'in bir çarpanı ile C_2'nin bir çarpanının ikili resolventi.

Resolution ilkesi, verilen bir clause kümesinden bir sonuç elde etmek amacıyla ardışık olarak resolventlerin üretildiği bir yöntemdir. Elde edilecek bu sonuç, amaca göre değişkenlik gösterir. Eğer resolution ilkesi, bir teoremin ispatlanmasında kullanılıyorsa, bulunacak sonuç, □ simgesi ile gösterilen boş clause olacaktır. Bunun, verilen clause kümesinin sağlanamaz olması ile ilgisi, daha sonra verilecek

resolution'un tamlığı teoremi ile anlaşılacaktır. Diğer yandan, eğer resolution, sonuç olarak bir cevabın beklendiği bir probleme uygulanıyorsa, bu cevap bir *sonlandırma clause*'u ile elde edilecektir.

2.8.4. Tanım : Resolution ilkesi kullanılarak, bir S clause kümesinden, sonuç olarak bir C clause'unun elde edilmesi işlemine bir *dedüksiyon* adı verilir.

2.8.1. Teorem : (Resolution İlkesinin Tamlığı) Bir S clause kümesinin sağlanamaz olması için gerekli ve yeterli koşul, boş clause'a bir dedüksiyonun bulunmasıdır. (Chang and Lee, 1973)

3. PROGRAM ANALİZİ

Birinci dereceden mantıkta bir problemin çözülmesi ya da bir soruya cevap alınmasına ilişkin çalışmalar, 1960'lı yılların sonlarında ortaya konan "program" kavramı ile başlamıştır. Mantıksal formüllerin oluşturdukları programların analiz edilmesinde sembolik mantığı kullanan Floyd (1967)'u, bir programın sonlanması, doğruluğu ve programların denkliği konularındaki çalışmaları ile Manna (1969) izlemiştir. Sonraki dönemlerde Kowalski (1974) ve Clark (1978), programlar ve programların amaçlarını (goal) bazı özel durumlarda incelemişler, bu çalışmalar daha sonra, Lloyd ve Topor (1984) tarafından genelleştirilmiştir.

En genel tanımıyla bir program deyimi, A bir atomik formül ve F herhangi bir formül olmak üzere, F→A biçimindedir. Bir program, sonlu sayıdaki program deyiminin bir araya gelmesi ile oluşur. Ancak programların daha etkili bir şekilde incelenebilmeleri için, daha detaylı olarak tanımlanmaları gerekmektedir.

3.1. Programın Tanımı ve Yönlü Graflar

3.1.1. Tanım : Bir *yönlü graf*, boş olmayan bir V kümesi, V ile ayrık olan bir A kümesi ve A'dan V×V'nin içine bir D dönüşümünden oluşur. Burada V ve A'nın elemanları sırasıyla *verteksler* ve *arklar* olarak adlandırılırlar. D dönüşümüne ise *yönlendirme etkisi* adı verilir. Buna göre, $a \in A$ ve $u, u' \in V$ olmak üzere, eğer $D(a)=(u,u')$ oluyorsa, a arkı bir u *başlangıç verteksi*'ne ve bir u' *uç verteksi*'ne sahiptir denir. (Busacker and Saaty, 1965)

Bu şekilde tanımlı V, A ve D birimlerinden oluşan bir yönlü graf, graf(V,A,D) biçiminde gösterilecektir.

3.1.2. Tanım : Bir graf(V,A,D) yönlü grafında bir *yol*, her bir a_i (i=1, 2, ... ,n) arkı, bir u_i başlangıç verteksi ve bir u_{i+1} uç verteksine sahip olmak üzere, arkların bir (a_i), i=1,2,...,n, dizisi ile tanımlanır.

3.1.3. Tanım : $v_1,v_2,...,v_n$'ler değişkenler olmak üzere, $\bar{v}=(v_1,v_2,...,v_n)$ biçiminde tanımlanan $\bar{v}$, $v_1, v_2, ..., v_n$ değişkenlerinin *vektörü* adını alır.

3.1.4. Tanım : Bir *program*, aşağıdaki koşullar sağlanacak biçimde, bir $\bar{x}=(x_1, x_2, ... x_m)$ *girdi vektörü*, bir $\bar{y}=(y_1, y_2, ... y_n)$ *program vektörü*, bir $\bar{z}=(z_1, z_2 ,..., z_k)$ *çıktı vektörü* ve bir graf(V,A,D) yönlü grafından oluşur :

1. graf(V,A,D) yönlü grafında kesin olarak iki verteks bulunur: hiçbir arkın uç verteksi olmayan *başlama verteksi* $S\in V$ ve hiçbir arkın başlangıç verteksi olmayan *durdurma verteksi* $H\in V$. Ayrıca her $u\in V$ için u, S'den H'ye bir yol üzerinde bulunur.
2. graf(V,A,D) yönlü grafında, H'ye bağlanmayan her a arkı, niceleyiciden bağımsız (standart formda) bir $p_a(\bar{x},\bar{y})$ formülü ile ve bir $\bar{y}\leftarrow f_a(\bar{x},\bar{y})$ değer ataması ile birlikte tanımlıdır. Aynı şekilde, H'ye bağlanan her a arkı, standart formda bir $p_a(\bar{x},\bar{y})$ formülü ile ve bir $\bar{z}\leftarrow f_a(\bar{x},\bar{y})$ değer ataması ile birlikte tanımlıdır. Burada p_a, a arkının *denetim predicate*'i, $p_a(\bar{x},\bar{y})$ ise a arkının *denetim formülü* adını alır.
3. Her $u\in V$, $u \neq H$, için, $a_1, a_2, ..., a_r$, u'den çıkan arklar ve sırasıyla $p_{a_1}, p_{a_2}, ..., p_{a_r}$ bu arkların denetim predicate'leri olsunlar. Bu durumda her $\bar{x}$ ve $\bar{y}$ için, **T** değerini alan bir ve yalnız bir $p_{a_i}(\bar{x},\bar{y})$, $i\in\{1,2,..,r\}$, formülü vardır.

3.1.1. Örnek : Yorumlama bölgemiz pozitif tamsayılar kümesi olmak üzere, x_1 ve x_2 tamsayılarının ortak katlarının en küçüğünü hesaplayacak programı aşağıdaki gibi ifade edebiliriz :

$y_1\leftarrow 1$.

$y_2\leftarrow x_1$.

1 : Eğer $x_2>y_2$ ise [2'ye git.];

aksi durumda [3'e git].

2 : Eğer $y_2|x_2$ ise [$z\leftarrow x_2$, dur.];

aksi durumda [$y_1\leftarrow y_1+1$, $y_2\leftarrow x_1.y_1$, 1'e git.].

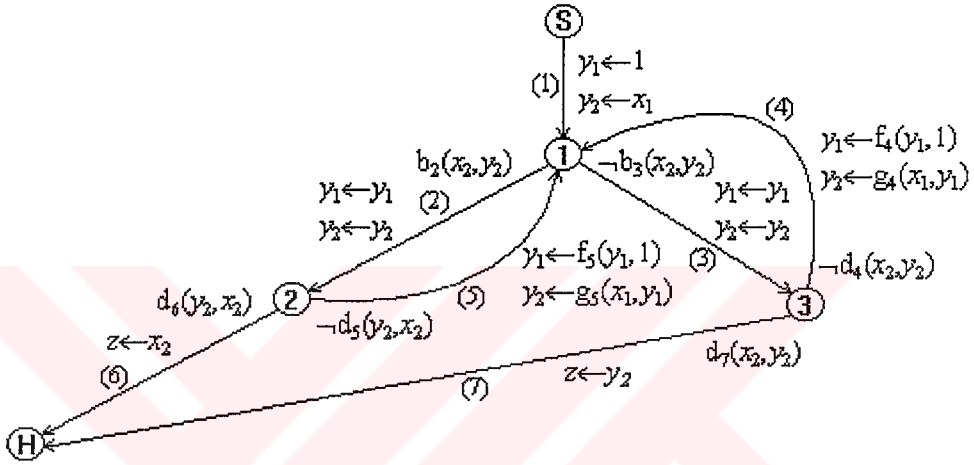
3 : Eğer $x_2|y_2$ ise [$z \leftarrow y_2$, dur.];

aksi durumda [$y_1 \leftarrow y_1+1$, $y_2 \leftarrow x_1.y_1$, 1'e git.].

Programın yönlü graf gösterimi, Şekil 3.1.'de görüldüğü gibi olacaktır. Bu gösterimde kullanılan denetim predicate'lerini ve fonksiyonları aşağıdaki biçimde tanımlayabiliriz:

$b_i(x,y) : x>y$ (i=2,3); $d_i(x,y) : x|y$ (i=4,5,6,7);

$f_i(x,y) : x+y$ (i=4,5); $g_i(x,y) : x.y$ (i=4,5).



Şekil 3.1. Programların yönlü graf gösterimleri.

Bir programın yönlü graf üzerindeki işleyişi, program denetiminin bir verteksten grafın yönü doğrultusunda diğer bir vertekse geçmesi biçiminde gerçekleşir. İşleyiş herzaman S verteksinden başlar ve H verteksinde son bulur. Aradaki işlemlerde ise aşağıdaki kurallar geçerli olacaktır:

1. Program denetiminin bir α_j verteksinde olduğunu varsayalım. a_j, başlangıç verteksi α_j ve uç verteksi α_{j+1} olan bir ark ve $\alpha_{j+1} \neq H$ olsun. Bu durumda, eğer $p_{a_j}(\bar{x}, \bar{y}_j)$ predicate'i **T** değerini alıyorsa $\bar{y}_{j+1} \leftarrow f_{a_j}(\bar{x}, \bar{y}_j)$ değer ataması gerçekleşir ve program denetimi α_j verteksinden α_{j+1} verteksine geçer.

2. Program denetimi yine bir α_j verteksinde ve a_j, başlangıç verteksi α_j ve uç verteksi H olan bir ark olsun. Eğer $p_{a_j}(\bar{x},\bar{y}_j)$ predicate'i **T** değerini alıyorsa program, $\bar{z} \leftarrow f_{a_j}(\bar{x},\bar{y}_j)$ değer ataması ile sonlanır.

3.2. Bir Programın Tanımlama Formülleri

Bir programın yönlü graf gösteriminden elde edilecek mantıksal formüller, onun resolution ile analiz edilmesine olanak verecektir. Bu şekilde, programın sonlandığı anda ürettiği mantıksal sonuçlar bir dedüksiyon sonucunda elde edilmiş olacaktır.

3.2.1. Tanım : Bir P programının graf(V,A,D) yönlü graf gösterimini gözönüne alalım. P'deki her $\alpha_i \in V$ verteksine karşılık, programın graf(V,A,D) üzerindeki işleyişi sırasında aşağıdaki biçimde değer alan bir $q_i(\bar{x},\bar{y})$ *erişim koşulu* bulunmaktadır:

(i) Program denetimi S verteksinden başladığı anda $q_S(\bar{x},\bar{y})$, **T** değerini alır.

(ii) Program denetimi bir α_i verteksine eriştiği anda $q_i(\bar{x},\bar{y})$, **T** değerini alır. Eğer α_i=H ise **T** değerini alacak erişim koşulu $q_H(\bar{x},\bar{z})$ olacaktır.

Burada $q_i(\bar{x},\bar{y})$'lere P'nin *erişim formülleri*, q_i'lere ise P'nin *erişim predicate*'leri denir. Özel olarak $q_S(\bar{x},\bar{y})$, P'nin *başlama formülü*, q_S *başlama predicate*'i; $q_H(\bar{x},\bar{z})$, *durdurma formülü* ve q_H *durdurma predicate*'i adını alır.

3.2.2. Tanım : Bir P programının graf(V,A,D) yönlü graf gösterimini gözönüne alalım. $\alpha_i,\alpha_j \in V$, bir $a \in A$ arkının sırasıyla başlangıç ve uç verteksleri olsunlar. $p_a(\bar{x},\bar{y})$, a'nın denetim predicate'i ve $f_a(\bar{x},\bar{y})$, a arkında program vektörüne atanan değeri alacak fonksiyon olmak üzere, a arkının *tanımlama formülü*;

$$w_a : q_i(\bar{x},\bar{y}) \wedge p_a(\bar{x},\bar{y}) \rightarrow q_j(\bar{x}, f_a(\bar{x},\bar{y}))$$

biçiminde tanımlanır.

3.2.3. Tanım : a_1, a_2, ..., a_r bir P programının arkları ve sırasıyla $w_{a_1}, w_{a_2}, \ldots, w_{a_r}$,bu arkların tanımlama formülleri olsunlar. Bu durumda P programının tanımlama formülü;

$$W_P : \forall \bar{y} \, (w_{a_1} \wedge w_{a_2} \wedge \ldots \wedge w_{a_r})$$

biçiminde tanımlanır.

3.2.4. Tanım : $W_P : \forall \bar{y} \, (w_{a_1} \wedge w_{a_2} \wedge \ldots \wedge w_{a_r})$, bir P programının tanımlama formülü olsun. $1 \leq i \leq r$ için, w'_{a_i}, w_{a_i} formülünün standart formu olmak üzere;

$$A_P = \{ w'_{a_1}, w'_{a_2}, \ldots, w'_{a_r} \}$$

kümesine, P programının *tanımlama clause'larının kümesi* adı verilir.

3.2.1. Örnek : Örnek 3.1.1.'de verilen programın her bir arkına karşılık gelen tanımlama formüllerini aşağıdaki gibi elde edebiliriz:

$w_1 : q_1(x_1,x_2,1,x_1)$

$w_2 : q_1(x_1,x_2,y_1,y_2) \wedge b_2(x_2,y_2) \rightarrow q_2(x_1,x_2,y_1,y_2)$

$w_3 : q_1(x_1,x_2,y_1,y_2) \wedge \neg b_3(x_2,y_2) \rightarrow q_3(x_1,x_2,y_1,y_2)$

$w_4 : q_1(x_1,x_2,y_1,y_2) \wedge q_3(x_1,x_2,y_1,y_2) \wedge \neg d_4(x_2,y_2) \rightarrow q_1(x_1,x_2,f_4(y_1,1),g_4(x_1,y_1))$

$w_5 : q_1(x_1,x_2,y_1,y_2) \wedge q_2(x_1,x_2,y_1,y_2) \wedge \neg d_5(y_1,x_2) \rightarrow q_1(x_1,x_2,f_5(y_1,1),g_5(x_1,y_1))$

$w_6 : q_1(x_1,x_2,y_1,y_2) \wedge q_2(x_1,x_2,y_1,y_2) \wedge d_6(y_2,x_2) \rightarrow q_H(x_1,x_2,x_2)$

$w_7 : q_1(x_1,x_2,y_1,y_2) \wedge q_3(x_1,x_2,y_1,y_2) \wedge d_7(x_2,y_2) \rightarrow q_H(x_1,x_2,y_2)$

Bu formüller standart forma getirilirlerse, tanımlama clause'ları;

(1) $q_1(x_1,x_2,1,x_1)$

(2) $\neg q_1(x_1,x_2,y_1,y_2) \vee \neg b_2(x_2,y_2) \vee q_2(x_1,x_2,y_1,y_2)$

(3) $\neg q_1(x_1,x_2,y_1,y_2) \vee b_3(x_2,y_2) \vee q_3(x_1,x_2,y_1,y_2)$

(4) $\neg q_1(x_1,x_2,y_1,y_2) \vee \neg q_3(x_1,x_2,y_1,y_2) \vee d_4(x_2,y_2) \vee q_1(x_1,x_2,f_4(y_1,1),g_4(x_1,y_1))$

(5) $\neg q_1(x_1,x_2,y_1,y_2)\vee\neg q_2(x_1,x_2,y_1,y_2)\vee d_5(y_1,x_2)\vee q_1(x_1,x_2,f_5(y_1,1),g_5(x_1,y_1))$

(6) $\neg q_1(x_1,x_2,y_1,y_2)\vee\neg q_2(x_1,x_2,y_1,y_2)\vee\neg d_6(y_2,x_2)\vee q_H(x_1,x_2,x_2)$

(7) $\neg q_1(x_1,x_2,y_1,y_2)\vee\neg q_3(x_1,x_2,y_1,y_2)\vee\neg d_7(x_2,y_2)\vee q_H(x_1,x_2,y_2)$

biçiminde olacaktır.

3.2.1. Teorem : A_P, bir P programının tanımlama clause'larının kümesi olsun. Bu durumda A_P sağlanabilirdir.(Chang and Lee, 1973)

3.3. Resolution ile Program Analizi

Bir P programı verildiğinde, programın yönlü graf gösteriminden elde edilecek tanımlama clause'ları kümesine resolution uygulanması ile programın analiz edilmesi, diğer bir deyişle programın girdi vektörü ile program tarafından üretilecek çıktı vektörü arasındaki ilişkinin bulunması mümkündür.

3.3.1. Tanım : $A_P=\{C_{a_1},C_{a_2},\ldots,C_{a_r}\}$, bir P programının tanımlama clause'larının kümesi olsun. R, A_P kümesinden üretilen bir resolvent olsun. R'nin, P programının *durdurma clause*'u olması için gerekli ve yeterli koşul, R'nin aşağıdaki koşulları sağlamasıdır:

(i) R'nin içerdiği tek erişim predicate'i q_H'dir.

(ii) R'deki hiçbir formül, program vektörüne ait bir değişken içermez.

(iii) R, S verteksinden H verteksine bir yola karşılık gelir.

3.3.1. Örnek : Yorumlama bölgemiz IR olmak üzere, girilen x_1, x_2 ve x_3 değerlerini kullanarak;

$$z=\frac{1}{\sqrt{x_1-x_2}-x_3}$$

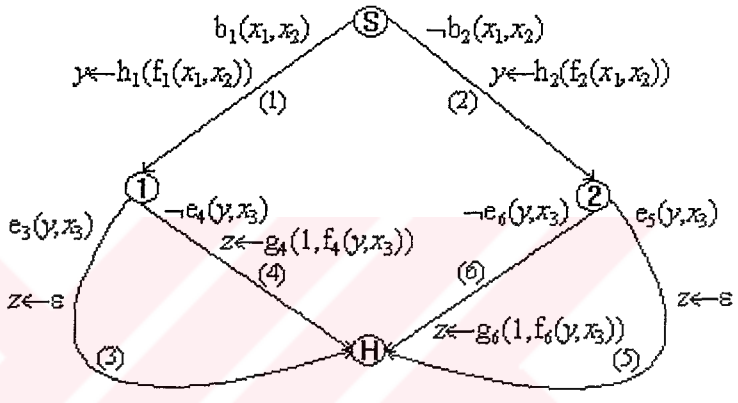
eşitliğinden z değerini hesaplayacak programın yönlü grafi Şekil 3.2.'de verilmiştir. Grafda, aşağıdaki denetim predicate'leri ve fonksiyonlar kullanılmıştır:

$b_i(x,y) : x>y$ (i=1,2); $e_i(x,y) : x=y$ (i=3,4,5,6);

$f_i(x,y) : x-y$ (i=1,2,4,6); $g_i(x,y) : x/y$ (i=4,6);

$h_i(x) : \sqrt{x}$ (i=1,2).

Programda, $\sqrt{x_1 - x_2} = x_3$ olması durumunda z'nin değer almayacağı açıktır. Bu durum, z'ye ε simgesinin atanmasıyla gösterilmiştir.



Şekil 3.2. Resolution ile program analizi

Yönlü grafdan elde edilen tanımlama formülleri:

$w_1 : b_1(x_1,x_2) \rightarrow q_1(x_1,x_2,x_3,h_1(f_1(x_1,x_2)))$

$w_2 : \neg b_2(x_1,x_2) \rightarrow q_2(x_1,x_2,x_3,h_2(f_2(x_1,x_2)))$

$w_3 : b_1(x_1,x_2) \wedge q_1(x_1,x_2,x_3,y) \wedge e_3(y,x_3) \rightarrow q_H(x_1,x_2,x_3,\varepsilon)$

$w_4 : b_1(x_1,x_2) \wedge q_1(x_1,x_2,x_3,y) \wedge \neg e_4(y,x_3) \rightarrow q_H(x_1,x_2,x_3,g_4(1,f_4(y,x_3)))$

$w_5 : \neg b_2(x_1,x_2) \wedge q_2(x_1,x_2,x_3,y) \wedge e_5(y,x_3) \rightarrow q_H(x_1,x_2,x_3,\varepsilon)$

$w_6 : \neg b_2(x_1,x_2) \wedge q_2(x_1,x_2,x_3,y) \wedge \neg e_6(y,x_3) \rightarrow q_H(x_1,x_2,x_3,g_6(1,f_6(y,x_3)))$

olarak bulunur. Programın tanımlama clause'ları ise aşağıdaki biçimde elde edilirler:

(1) $\neg b_1(x_1,x_2) \vee q_1(x_1,x_2,x_3,h_1(f_1(x_1,x_2)))$

(2) $b_2(x_1,x_2)\vee q_2(x_1,x_2,x_3,h_2(f_2(x_1,x_2)))$

(3) $\neg b_1(x_1,x_2)\vee\neg q_1(x_1,x_2,x_3,y)\vee\neg e_3(y,x_3)\vee q_H(x_1,x_2,x_3,\varepsilon)$

(4) $\neg b_1(x_1,x_2)\vee\neg q_1(x_1,x_2,x_3,y)\vee e_4(y,x_3)\vee q_H(x_1,x_2,x_3,g_4(1,f_4(y,x_3)))$

(5) $b_2(x_1,x_2)\vee\neg q_2(x_1,x_2,x_3,y)\vee\neg e_5(y,x_3)\vee q_H(x_1,x_2,x_3,\varepsilon)$

(6) $b_2(x_1,x_2)\vee\neg q_2(x_1,x_2,x_3,y)\vee e_6(y,x_3)\vee q_H(x_1,x_2,x_3,g_6(1,f_6(y,x_3)))$

Programın başlangıç koşullarına dikkat edilirse, x_1 ve x_2 için yalnızca iki durumun varlığı sözkonusudur : $x_1 \not> x_2$ ve $x_1 > x_2$. $x_1 = x_2$ durumu gözardı edilmiştir. Biz $x_1 \neq x_2$ olduğunu varsayalım ve matematiksel eşitlik ve büyüklük bağıntılarının sağlayacakları aşağıdaki formülü verelim:

$$\forall x \forall y\ (x \neq y \wedge x > y \rightarrow y \not> x) \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

Bu durumda, yapılan varsayımdan elde edilecek formülden ve (1) formülünden, aşağıdaki clause'ları elde ederiz :

(7) $\neg e_3(x_1,x_2)$

(8) $e_3(x,y)\vee\neg b_1(x,y)\vee\neg b_1(y,x)$

Resolution'a geçmeden önce, bir noktanın açıklanması gerekmektedir. Kullanılan tüm denetim predicate'lerinin ve fonksiyonların indisleri, onların yönlü graf üzerinde ilişkili oldukları arkları belirlemek içindir. Bu predicate'ler ve fonksiyonlar, anlamsal olarak denk olduklarından, resolution açısından da denk olacaklardır. Örneğin, $e_3(x_1,f_2(x_2,x_3))$ ve $\neg e_6(x_1,f_4(x_2,x_3))$ clause'larından resolution sonucu üretilen clause, □ olacaktır.

Şimdi, bulunan tanımlama clause'ları ile 7. ve 8. clause'lardan, aşağıdaki resolventleri elde edebiliriz :

(9) $\neg b_1(x_1,x_2)\vee\neg b_1(x_2,x_1)$

(7) ve (8)'den; $\sigma_1=\{x_1/x, x_2/y\}$.

(10) $\neg b_1(x_1,x_2)\vee\neg e_3(h_1(f_1(x_1,x_2)),x_3)\vee q_H(x_1,x_2,x_3,\varepsilon)$

(1) ve (3)'den; $\sigma_2=\{h_1(f_1(x_1,x_2))/y\}$.

(11) $\neg b_1(x_1,x_2)\vee e_4(h_1(f_1(x_1,x_2)),x_3)\vee q_H(x_1,x_2,x_3,g_4(1,f_4(h_1(f_1(x_1,x_2)),x_3)))$

(1) ve (4)'den; $\sigma_3=\sigma_2$.

(12) $b_2(x_1,x_2)\vee\neg e_5(h_2(f_2(x_1,x_2)),x_3)\vee q_H(x_1,x_2,x_3,\varepsilon)$

(2) ve (5)'den; $\sigma_4=\{\ h_2(f_2(x_1,x_2))/y\ \}$.

(13) $b_2(x_1,x_2)\vee e_6(h_2(f_2(x_1,x_2)),x_3)\vee q_H(x_1,x_2,x_3,g_6(1,f_6(h_2(f_2(x_1,x_2)),x_3)))$

(2) ve (6)'dan; $\sigma_5=\sigma_4$.

(14) $\neg b_1(x_2,x_1)\vee e_6(h_2(f_2(x_1,x_2)),x_3)\vee q_H(x_1,x_2,x_3,g_6(1,f_6(h_2(f_2(x_1,x_2)),x_3)))$

(9) ve (12)'den; $\sigma_6=\varnothing$.

(15) $\neg b_1(x_2,x_1)\vee e_6(h_2(f_2(x_1,x_2)),x_3)\vee q_H(x_1,x_2,x_3,g_6(1,f_6(h_2(f_2(x_1,x_2)),x_3)))$

(9) ve (13)'den; $\sigma_7=\varnothing$.

Bulunan (10), (11), (14) ve (15)'nci resolventler, birer durdurma clause'udurlar ve programın ürettiği sonuçları verirler. Bu clause'ları, predicate ve fonksiyonların anlamsal karşılıkları ile mantıksal formül biçiminde yazacak olursak, programın analizinden elde edilen sonuçları daha açık bir şekilde ifade etmiş oluruz :

1) $x_1>x_2 \wedge \sqrt{x_1 - x_2} = x_3 \rightarrow z = \varepsilon.$ (10. Resolvent)

2) $x_1>x_2 \wedge \sqrt{x_1 - x_2} \neq x_3 \rightarrow z = \dfrac{1}{\sqrt{x_1 - x_2} - x_3}.$ (11. Resolvent)

3) $x_2>x_1 \wedge \sqrt{x_1 - x_2} = x_3 \rightarrow z = \varepsilon.$ (14. Resolvent)

4) $x_2>x_1 \wedge \sqrt{x_1 - x_2} \neq x_3 \rightarrow z = \dfrac{1}{\sqrt{x_1 - x_2} - x_3}.$ (15. Resolvent)

3.4. Bir Programın Sonlanması ve Programdan Sonuç Alınması

Verilen bir P programının sonlanması ve sonlandığı anda sonuç vermesi, program analizinin en önemli aşamasıdır. Bu kesimde verilecek iki teoremle, bir programın sonlanması ve sonlandığı anda sonuç vermesi, belirli koşullar altında garanti edilmektedir. Ancak öncelikle bazı gösterimlerin verilmesi gerekmektedir.

Bir programın resolution ile analizinde, bazı durumlarda programın tanımlama clause'ları yeterli olmamaktadır. Bu gibi durumlarda, Örnek 3.3.1.'de olduğu gibi,

matematiksel olarak doğruluğu ispatlanmış bazı mantıksal formüllerden ya da kullanılan denetim predicate'lerinin bilinen bazı özelliklerinden elde edilecek clause'lar kullanılabilirler. Ayrıca, program içersinde yeralan her bir döngü için, yine verilen yorumlama bölgesi üzerinde matematiksel olarak doğruluğu bilinen bir indirgeme formülünün kullanılması gerekmektedir. Bunun gibi, programın tanımlama clause'larına ek olarak elde edilen clause'ların kümesi A_S ile gösterilecektir.

Programın girdi vektörünün bileşenleri olan değişkenlerin türleri ile ilgili aksiyomlardan gelen clause'ların kümesi ise A_I ile gösterilecektir.

3.4.1. Teorem : S, verilen bir P programının $A_P \cup A_S \cup A_I$ biçimindeki clause kümesi olsun. S sağlanabilirdir. (Chang and Lee, 1973)

3.4.1. Önteorem : (Program Termination Lemma) A_T, verilen bir P programında, A_P'den q_H predicate'ini içeren her bir literalin silinmesiyle elde edilen clause kümesi olsun. Bu durumda, P'nin durması için gerekli ve yeterli koşul, $A_T \wedge A_S \wedge A_I$ formülünün sağlanamaz olmasıdır. (Chang and Lee, 1973)

3.4.2. Teorem : (Program Response Theorem) P bir program, $S=A_P \cup A_S \cup A_I$ olsun. Bu durumda P'nin durması için gerekli ve yeterli koşul, S'den durdurma clause'una bir dedüksiyonun bulunmasıdır. (Chang and Lee, 1973)

4. BİRİNCİ MERTEBEDEN DOĞRUSAL DİFERENSİYEL DENKLEMLER

Diferensiyel denklemlerin çözümlerinin elde edilmesi gibi konularda sembolik hesaplama yapılabilmesi için, öncelikle kurulan algoritmik yapı içersinde yer alacak tüm aksiyom ve sonuçlar, sağlam matematiksel temellere dayandırılmalıdırlar. Dolayısıyla bu bölümde, ele alınan diferensiyel denklemin genel formu tanıtılarak, elde edilecek çözüm, katsayıların alacakları özel değerlere göre sınıflandırılmıştır.

4.1. Doğrusal Denklemler

4.1.1. Tanım : Genel formu,

$$\frac{dy}{dx} + p(x)y = q(x)$$

biçiminde olan diferensiyel denkleme *birinci mertebeden doğrusal diferensiyel denklem* adı verilir.

4.1.1. Teorem : $\frac{dy}{dx} + p(x)y = q(x)$ biçimindeki bir doğrusal diferensiyel denklem,

$$\lambda(x) = e^{\int p(x)dx}$$

olarak tanımlanan bir integral çarpanına sahiptir. Bu durumda sözkonusu denklemin bir parametreli çözüm ailesi,

$$y = \frac{1}{\lambda(x)}\left(\int \lambda(x).q(x).dx + c\right)$$

olur. Ayrıca bu çözüm ailesi, tüm çözümleri içerir.

4.2. Bernoulli Denklemi

4.2.1. Tanım : Genel formu,

$$\frac{dy}{dx} + p(x)y = q(x)y^n$$

biçiminde olan diferensiyel denkleme *Bernoulli denklemi* adı verilir.

4.2.1. Teorem : $n \neq 0$ ve $n \neq 1$ için,

$$\frac{dy}{dx} + p(x)y = q(x)y^n$$

biçimindeki Bernoulli denklemi, $u=y^{1-n}$ dönüşümü ile u'ya göre doğrusal diferensiyel denkleme indirgenir.

4.3. Çözümlerin Sınıflandırılması

4.3.1. Gösterim : Genel olarak $\frac{dy}{dx} + A(x)y = B(x)y^n$ biçiminde ifade edilen diferensiyel denklemden, $A(x)$, $B(x)$ ve n'nin alacakları özel değerlere göre elde edilen tüm diferensiyel denklemler, DE[$A(x)$, $B(x)$, n] sembolü ile gösterilecektir.

4.3.1. Tanım : DE[$A(x)$, $B(x)$, n] diferensiyel denklemini gözönüne alalım. $A(x)$ ve $B(x)$ fonksiyonlarının her ikisinin de sabit fonksiyondan farklı olduğu durumlara, DE[$A(x)$,$B(x)$,n] denkleminin *genel durumları* denir. Bunun dışındaki tüm alt durumlar, DE[$A(x)$, $B(x)$, n] denkleminin *temel durumları* adını alır.

4.3.1. Sonuç : (i) DE[$A(x)$, $B(x)$, n] diferensiyel denkleminin temel durumlarındaki çözümleri, aşağıdaki gibi elde edilebilir:

1. DE[0, 0, n] için genel çözüm : $y = c$.

2.1. DE[0, k, 0] için genel çözüm : $y = kx+c$.

2.2. DE[0, k, 1] için genel çözüm : $y = ce^{kx}$.

2.3. DE[0, k, n], n>1 için genel çözüm : $y = (1-n)(c+kx)$.

3. DE[k, 0, n], n≥0 için genel çözüm : $y = \frac{c}{e^{kx}}$.

4.1. DE[k, r, 0] için genel çözüm : $y = \frac{r}{k} + \frac{c}{e^{kx}}$.

4.2. DE[k, r, 1] için genel çözüm : $y = ce^{(r-k)x}$.

4.3. DE[k, r, n], n>1 için genel çözüm : $y^{1-n} = \frac{r}{k} + ce^{k(n-1)x}$.

5.1. DE[0, $B(x)$, 0] için genel çözüm : $y = \int B(x)dx + c$.

5.2. DE[0, $B(x)$, 1] için genel çözüm : $y = ce^{\int B(x).dx}$.

5.3. DE[0, B(x), n], n>1 için genel çözüm : $y^{1-n} = (1-n)\int B(x)dx + c.$

6. DE[A(x), 0, n], n≥0 için genel çözüm : $y = \dfrac{c}{e^{\int A(x)dx}}$.

7.1. DE[k, B(x), 0] için genel çözüm : $y = \dfrac{1}{e^{kx}}\int B(x)e^{kx}dx + \dfrac{c}{e^{kx}}$

7.2. DE[k, B(x), 1] için genel çözüm : $y = ce^{\int B(x)dx - kx}$.

7.3. DE[k, B(x), n], n>1 için genel çözüm : $y^{1-n} = (1-n)e^{k(n-1)x}\int B(x)e^{k(1-n)x}dx + ce^{k(n-1)x}$.

8.1. DE[A(x), k, 0] için genel çözüm : $y = \dfrac{k}{e^{\int A(x)dx}}\left(\int e^{\int A(x)dx} + c\right)$.

8.2. DE[A(x), k, 1] için genel çözüm : $y = ce^{kx - \int A(x)dx}$.

8.3. DE[A(x), k, n], n>1 için genel çözüm :

$$y^{(1-n)} = k(1-n)\,e^{(n-1)\int A(x)dx}\int e^{(1-n)\int A(x)dx}dx + ce^{(n-1)\int A(x)dx}.$$

(ii) DE[A(x), B(x), n] diferensiyel denkleminin genel durumlarındaki çözümleri, aşağıdaki gibi elde edilebilir:

1. DE[A(x), B(x), 0] için genel çözüm : $y = \dfrac{1}{e^{\int A(x)dx}}\int B(x)e^{\int A(x)dx}dx + \dfrac{c}{e^{\int A(x)dx}}$.

2. DE[A(x), B(x), 1] için genel çözüm : $y = c\,e^{\int (B(x)-A(x))dx}$.

3. DE[A(x), B(x), n], n>1 için genel çözüm :

$$y^{1-n} = (1-n)\,e^{(n-1)\int A(x)dx}.\int B(x)e^{(1-n)\int A(x)dx}dx + c(1-n)e^{(n-1)\int A(x)dx}$$

İspat **:** Her iki şıkta yer alan tüm sonuçlar, Teorem 4.1.1. ve Teorem 4.2.1.'in özel durumları olarak elde edilebilirler.

5. DEDAKTİF YAKLAŞIM

5.1. Genel Tanımlar

Bu kesimde genel olarak, kullanılan değişken, sabit ve parametre sembolleri verilecek, ele alınan diferensiyel denklem ve çözümü ile türev ve integrasyon kurallarının birinci dereceden mantık ilkeleri içersinde ifade edilmesinde kullanılacak predicate ve fonksiyonlar tanımlanacaktır.

Öncelikle matematiksel değişken sembolleri ile unification dönüşümlerinin uygulanabileceği değişken sembollerinin birbirlerinden ayrılmaları gerekmektedir. Çalışmamızın bundan sonraki bölümlerinde x ve y matematiksel değişken sembolleri olarak kullanılacaklardır. Bunlardan x serbest, y ise bağımlı değişkendir. Diğer değişkenler A, B, U, V ve F_i (i=1, 2, ..., n) sembolleri ile gösterileceklerdir. Sabitlerin gösteriminde ise c_i , i=1, 2, ..., n , sembolleri kullanılacaktır.

5.1.1. Tanım : Bir *Sabit Terimi* özyinelemli olarak aşağıdaki gibi tanımlanır :

(i) Bir sabit, bir terimdir.

(ii) f bir n-bileşenli fonksiyon sembolü ve $s_1,...,s_n$ sabit terimleri olmak üzere $f(s_1,...,s_n)$ bir sabit terimidir.

5.1.2. Tanım : i=1,...,n için p_i'ler birbirlerinden farklı değişkenler ve her bir s_i bir sabit terimi olmak üzere $\{s_1/p_1 ,..., s_n /p_n\}$ biçimindeki sonlu kümeye bir *parametre dönüşümü* adı verilir. Özel olarak, p_i değişkenleri de *parametre* adını alırlar.
Parametre dönüşümleri ϕ_i, i=1,..,n , sembolleri ile gösterilecektir. Parametrelerin gösteriminde ise n, k ve r harfleri kullanılacaktır.

5.1.3. Tanım : v bir değişken, c bir sabit, f bir fonksiyon sembolü, p bir predicate sembolü, E_1, E_2,...,E_n'ler matematiksel ifadeler ve *op*, "+", "*", "-" ya da "/" işlemlerinden herhangi birisini göstersin. Bu durumda MD *mantıksal dönüşüm operatörü*, özyinelemli olarak aşağıdaki gibi tanımlanır :

(i) MD(v) = v

(ii) $MD(c) = c$

(iii) $MD(E_1\ op\ E_2) = op(E_1, E_2)$

(iv) $MD(f(E_1, E_2,...,E_n)) = f(MD(E_1),MD(E_2),...,MD(E_n))$

(v) $MD(p(E_1, E_2,...,E_n)) = p(MD(E_1),MD(E_2),...,MD(E_n))$

5.1.4. Tanım : E_1 ve E_2 matematiksel ifadeler olmak üzere, *eşitlik* ve *türev* predicate'leri aşağıdaki gibi tanımlanır :

- $=(E_1,E_2)$: $E_1 = E_2$
- $d(E_1,E_2)$: $\frac{d(E_1)}{dx} = E_2$

5.1.5. Tanım : x ve y sırasıyla serbest ve bağımlı değişken sembolleri, E_1 ve E_2 matematiksel ifadeler olmak üzere *diferensiyel*, *integral*, *kuvvet* ve *kök* fonksiyonları aşağıdaki gibi tanımlanır :

- $diff(y,x)$: dy/dx
- $intg(E_1)$: $\int E_1 dx$
- $pow(E_1,E_2)$: $E_1^{E_2}$
- $root(E_1,E_2)$: $\sqrt[E_2]{E_1}$

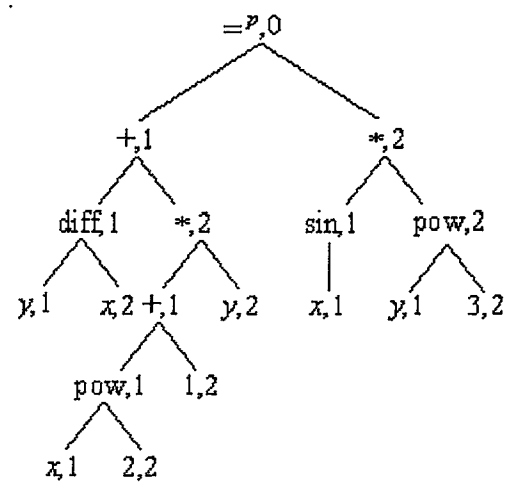
5.1.1. Örnek : $\frac{dy}{dx} + (x^2+1)*y = (\sin(x))*y^3$ diferensiyel denklemini, Tanım 5.1.4. ve Tanım 5.1.5.'i kullanarak;

$$=(diff(y,x) + (pow(x,2)+1)*y) , \sin(x)*pow(y,3))..... (1)$$

biçiminde yazabiliriz (1) ifadesine MD operatörünün uygulanması sonucunda, verilen diferensiyel denklemin atomik formül olarak karşılığı,

$$=(+(diff(y,x),*(+(pow(x,2),1),y)),*(\sin(x),pow(y,3)))..... (2)$$

olarak elde edilmiş olur. (2) formülünün belirli ağaç gösterimi ise Şekil 5.1.'de görüldüğü gibi olacaktır.



Şekil 5.1. Örnek 5.1.1.'deki atomik formülün belirli ağaç gösterimi.

5.2. Çözümlerin Mantıksal Formları

Genel olarak, ele aldığımız diferensiyel denklem ve onun çözümlerine ilişkin matematiksel ifadeleri, Kesim 5.1.'de verilen tanımları kullanarak mantıksal formda, yani birinci dereceden mantığın dilsel yapısı içersinde bir gerektirme olarak yazabiliriz. Örneğin, DE[2,12,0] diferensiyel denklemini gözönüne alalım. Bu denklemin çözümü $y=6+c_1/e^{2x}$ biçimindedir. Matematiksel olarak doğruluğu kanıtlanmış olan bu yargıyı, daha açık olarak, "Eğer $(dy/dx)+2y=12$ ise $y=6+c_1/e^{2x}$ olur." şeklinde ifade edebiliriz. Aynı yargıyı, verilen tanımları ve MD operatörünü kullanarak;

$$\forall x\ (=(+(\text{diff}(y,x)*(2,y)),12) \rightarrow =(+(6,/(c_1,\exp(*(2,x))))))$$

mantıksal formülü ile gösterebiliriz.

5.2.1. Tanım : A ve B iki fonksiyon, $n\geq 0$ bir tamsayı olmak üzere, DE[A,B,n] diferensiyel denklemini gözönüne alalım. $y=F$ $(F=F(x))$ bu denklemin genel çözümü olsun. Bu durumda, belirtilen diferensiyel denklemin çözümüne ilişkin *mantıksal form*;

$$\forall x\ (\text{MD}(\text{DE}[A,B,n]) \rightarrow =(y,\text{MD}(F)))$$

biçiminde olur. Burada, gerektirme okunun sol tarafındaki formül mantıksal formun *denklem bölümü*, sağ tarafındaki formül ise *çözüm bölümü* adını alır.

Bu tanımdan sonra, daha önce elde ettiğimiz temel durumlardaki çözümler ile genel durumlardaki çözümlerin mantıksal formlarını oluşturabiliriz.

5.2.1. Önteorem : (i) DE[A,B,n] diferensiyel denkleminin temel durumlarındaki çözümlerine ilişkin mantıksal formlar aşağıdaki biçimde ifade edilirler:

(Mt1) $\forall x\ (=(\text{diff}(y,x),0) \rightarrow =(y,c_1))$.

(Mt2.1) $\forall x\ (=(\text{diff}(y,x),k) \rightarrow =(y,+(*(k,x),c_1)))$.

(Mt2.2) $\forall x\ (=(\text{diff}(y,x),*(k,y)) \rightarrow =(y,*(c_1,\exp(*(k,x)))))$.

(Mt2.3) $\forall x\ (=(\text{diff}(y,x),*(k,\text{pow}(y,n))) \rightarrow =(\text{pow}(y,-(1,n)),*(-(1,n),+(c_1,*(k,x)))))$.

(Mt3) $\forall x\ (=(+(\text{diff}(y,x),*(k,y)),0) \rightarrow =(y,/(c_1,\exp(*\ (k,x)))))$.

(Mt4.1) $\forall x\ (=(+(\text{diff}(y,x),*(k,y)),r) \rightarrow =(y,+(/(r,k),/(c_1,\exp(*(k,x))))))$.

(Mt4.2) $\forall x\ (=(+(\text{diff}(y,x),*(k,y)),*(r,y)) \rightarrow =(y,*(c_1,\exp(*(-(r,k),x)))))$.

(Mt4.3) $\forall x\ (=(+(\text{diff}(y,x),*(k,y)),*(r,\text{pow}(y,n))) \rightarrow =(\text{pow}(y,-(1,n)),+(/(r,k),$
$*(c_1,\exp(*(k,*(-(n,r)\ ,x)))))))$.

(Mt5.1) $\forall x\ (=(\text{diff}(y,x),B) \rightarrow =(y,*(c_1,\text{intg}(B))))$.

(Mt5.2) $\forall x\ (=(\text{diff}(y,x),*(B,y)) \rightarrow =(y,+(\exp(\text{intg}(B)),c_1)))$.

(Mt5.3) $\forall x\ (=(\text{diff}(y,x),*(B,\text{pow}(y,n))) \rightarrow =(\text{pow}(y,-(1,n)),+(*(-(1,n),\text{intg}(B)),c_1)))$.

(Mt6) $\forall x\ (=(+(\text{diff}(y,x),*(A,y)),0) \rightarrow =(y,/(c_1,\exp(\text{intg}(A)))))$.

(Mt7.1) $\forall x\ (=(+(\text{diff}(y,x),*(k,y)),B) \rightarrow =(y,+(*(/(1,\exp(*(k,x))),$
$\text{intg}(*(B,\exp(*(k,x)))),/(c_1,\exp(*(k,x))))))$.

(Mt7.2) $\forall x\ (=(+(\text{diff}(y,x),*(k,y)),*(B,y)) \rightarrow =(y,*(c_1,\exp(-(\text{intg}(B),*(k,x))))))$.

(Mt7.3) $\forall x\ (=(+(\text{diff}(y,x),*(A,y)),*(B,\text{pow}(y,n))) \rightarrow =(\text{pow}(y,-(1,n)),+(*(*(-(1,n),$
$\exp(*(*(k,-(n,1)),x))),\text{intg}(*(B,\exp(*(*(k,-(1,n)),x))),*(c_1,\exp(*(*(k,-$
$(n,1)),x)))))$.

(Mt8.1) $\forall x(=(+(\text{diff}(y,x),*(A,y)),k) \rightarrow =(y,*(/(k,\exp(\text{intg}(A))),$
$+(\text{intg}(\exp(\text{intg}(A))),c_1))))$.

(Mt8.2) $\forall x\ (=(+(\text{diff}(y,x),*(A,y)),*(k,y)) \rightarrow =(y,*(c_1,\exp(-(*(k,x),\text{intg}(A))))))$.

(Mt8.3) $\forall x\ (=(+(\text{diff}(y,x),*(A,y)),*(k,\text{pow}(y,n))) \rightarrow =(\text{pow}(y,-(1,n)),+(*(*(*(k,-(1,n)),$
$\exp(*(-(n,1),\text{intg}(A)))),\text{intg}(\exp(*(-(1,n),\text{intg}(A)))),*(c_1,\exp(*(-$
$(n,1),\text{intg}(A))))))$.

(ii) DE[A,B,n] diferensiyel denkleminin genel durumlarındaki çözümlerine ilişkin mantıksal formlar aşağıdaki biçimde ifade edilirler:

(Mg1) $\forall x$ (=(+(diff(y,x),*(A,y)),B) → =(y,+(*(/(1,exp(intg(A))),
intg(*(B,exp(intg(A))))),/(c_1,ex p(intg(A))))).

(Mg2) $\forall x$ (=(+(diff(y,x),*(A,y)),*(B,y))→ =(y,*(c_1,exp(intg(-(B,A)))))).

(Mg3) $\forall x$ (=(+(diff(y,x),*(A,y)),*(B,pow(y,n)))→=(pow(y,-(1,n)),+(*(*(-
(1,n),exp(*(- (n,1),intg(A)))),intg(*(B,exp(*(-(1,n),intg(A)))))),
((-(1,n),c_1),exp(*(-(n,1),intg(A)))))).

***İspat* : (i)** Sonuç 4.3.1.'in (i) şıkkı gereğince, DE[0,0,n] diferensiyel denkleminin genel çözümünün $y=c_1$ olduğunu biliyoruz. Diğer yandan, Tanım 5.1.4. ve Tanım 5.1.5. kullanılarak bu diferensiyel denklem,

=(diff(y,x),0)

olarak yazılabilir. MD(=(diff(y,x),0)) = =(diff(y,x),0) ve $MD(c_1) = c_1$ olduğundan, Tanım 5.2.1.'den, (Mt1) mantıksal formu,

$\forall x$ (=(diff(y,x),0) → =(y,c_1))

olarak elde edilmiş olur. Diğer mantıksal formlar da benzer biçimde elde edilebilirler.

(ii) Sonuç 4.3.1.'in (ii) şıkkından, DE[A,B,0] diferensiyel denkleminin genel çözümünün,

$$y = \frac{1}{\exp\left(\int A dx\right)} * \int B * \exp\left(\int A dx\right) dx + \frac{c}{\exp\left(\int A dx\right)}. = F$$

olduğunu biliyoruz. Yine, Tanım 5.1.4. ve Tanım 5.1.5. kullanılarak sözkonusu diferensiyel denklem,

=(diff(y,x)+A*y,B)

olarak yazılabilir. Tanım 5.1.5.'i kullanarak F ifadesini,

$$\frac{1}{\exp(intg(A))} * intg(B * \exp(intg(A))) + \frac{c}{\exp(intg(A))}$$

biçiminde yazabiliriz. MD(=(diff(y,x)+A*y,B)) = =(+(diff(y,x),*(A,y)),B) ve

MD(F) = +(*(/(1,exp(intg(A))),intg(*(B,exp(intg(A))))),/(c_1,exp(intg(A)))) olduğuna göre, Tanım 5.2.1. gereğince (Mg1) mantıksal formu,

$\forall x$ (=(+(diff(y,x),*(A,y)),B) $\rightarrow$ =(y,+(*(/(1,exp(intg(A))),

intg(*(B,exp(intg(A))))),/(c_1,exp(intg(A)))))

olarak elde edilir. Diğer mantıksal formların elde edilmesi de benzer biçimde olacaktır.

Temel durumlardaki çözümlerin mantıksal formlarından, (Mt5.1)'den (Mt8.3)'e kadar olan mantıksal formlarla, genel durumlardaki çözümlerin mantıksal formlarının çözüm bölümlerine bakılacak olursa, bu bölümlerin özel olarak, integral fonksiyonunu içerdikleri görülür. Resolution sırasında, çözümde bulunan bir integralin hesaplanabilmesi için, çözüm ve ilgili integrasyon kuralı arasında bir ikili resolution'un gerçekleşmesi gerekmektedir. Bu resolution'dan elde edilecek resolvent, denklemin genel çözümü olacaktır.

5.2.2. Tanım : =(y,MD(F)) atomik formülü, verilen bir DE[A,B,n] diferensiyel denkleminin çözümüne ilişkin herhangi bir mantıksal formda yer alan çözüm bölümü olsun. Bu formülü K ile gösterelim. Eğer MD(F) terimi **intg** fonksiyonunu bulunduruyorsa K'ya *integral içeren çözüm* adı verilir.

5.2.3. Tanım : K bir integral içeren çözüm ve intg($f_1(x)$), intg($f_2(x)$),..., intg($f_n(x)$)'ler K içersinde yer alan intg fonksiyonları olsunlar. Bu durumda K'ya karşılık gelen mantıksal form,

$$\forall x\ (K\wedge=(\text{intg}(f_1(x)),F_1)\wedge=(\text{intg}(f_2(x)),F_2)\wedge\ldots\wedge=(\text{intg}(f_n(x)),F_n))\rightarrow K^1$$

olarak tanımlanır. Burada K^1, K formülüne

$$\alpha=\{F_1/\ \text{intg}(f_1(x)),F_2/\ \text{intg}(f_2(x)),\ldots,F_n/\text{intg}(f_n(x))\}$$

özel dönüşümünün uygulanmış biçimidir.

Bu formlar, *integral içeren çözümlere karşılık gelen mantıksal formlar* olarak adlandırılırlar.

5.2.2. Önteorem : DE[A,B,n] diferensiyel denkleminin integral içeren çözümlerine karşılık gelen mantıksal formlar aşağıdaki biçimde elde edilirler:

(Mç1) $\forall x\ (=(y,+(\text{intg}(B),c_1))\wedge=(\text{intg}(B),F_1) \rightarrow =(y,+(F_1,c_1)))$.

(Mç2) $\forall x\ (=(y,*(c_1,\exp(\text{intg}(B))))\wedge=(\text{intg}(B),F_1) \rightarrow =(y,*(c_1,\exp(F_1))))$.

(Mç3) $\forall x(=(\text{pow}(y,-(1,n)),+(*(-(1,n),\text{intg}(B)),c_1)\wedge=(\text{intg}(B),F_1)\rightarrow$
$=(\text{pow}(y,-(1,n)),+(*(-(1,n),F_1),\ c_1))$.

(Mç4) $\forall x\ (=(y,/(c_1,\exp(\text{intg}(A))))\wedge=(\text{intg}(A),F_1) \rightarrow =(y,/(\ c_1,\exp(F_1))))$.

(Mç5) $\forall x\ (=(y,+(*(/(1,\exp(*(k,x))),\text{intg}(*(B,\exp(*(k,x))))),/(c_1,\exp(*(k,x)))))\wedge$
$=(\ \text{intg}(*(B,\exp(*(k,x)))),F_1) \rightarrow =(y,+(*(/(1,\exp(*(k,x))),F_1),/(c_1,\exp(*(k,x))))))$.

(Mç6) $\forall x\ (=(y,*(c_1,\exp(-(\text{intg}(B),*(k,x)))))\wedge=(\text{intg}(B),F_1) \rightarrow$
$=(y,*(c_1,\exp(-(F_1,*(k,x))))))$.

(Mç7) $\forall x\ (=(\text{pow}(y,-(1,n)),+(*(*(-(1,n),\exp(*(*(k,-(n,1)),x))),$
$\text{intg}(*(B,\exp(*(*(k,-(1,n)),x)))),*(c_1,\ \exp(*(*(k,-(n,1)),x))))\wedge$
$=(\ \text{intg}(*(B,\exp(*(*(k,-(1,n)),x)))),F_1) \rightarrow =(\text{pow}(y,-(1,n)),+(*(*(-$
$(1,n),\exp(*(*(k,-(n,1)),x))),F_1),*(c_1,\ \exp(*(*(k,-(n,1)),x)))))$.

(Mç8) $\forall x(=(y,*(/(k,\exp(\text{intg}(A))),+(\text{intg}(\exp(\text{intg}(A))),c_1)))\wedge=(\text{intg}(A),F_1)\wedge$
$=(\text{intg}(\exp(F_1)),F_2) \rightarrow =(y,*(/(k,\exp(F_1)),+(F_2,c_1))))$.

(Mç9) $\forall x\ (=(y,*(c_1,\exp(-(*(k,x),\text{intg}(A)))))\wedge=(\text{intg}(A),F_1) \rightarrow$
$=(y,*(c_1,\exp(-(*(k,x),F_1)))))$.

(Mç10) $\forall x\ (=(\text{pow}(y,-(1,n)),+(*(*(*(k,-(1,n)),\ \exp(*(-(n,1),\text{intg}(A)))),$
$\text{intg}(\exp(*(-(1,n),\text{intg}(A)))),*(c_1,\exp(*(-(n,1),\text{intg}(A))))))),$
$*(c_1,\exp(*(*(k,-(n,1)),x)))))\wedge=(\text{intg}(A),F_1)\wedge=(\ \text{intg}(\exp(*(-(1,n),F_1))),F_2)) \rightarrow$
$=(\text{pow}(y,-(1,n)),+(*(*(*(k,-(1,n)),\ \exp(*(-(n,1),F_1))),F_2,$
$*(c_1,\exp(*(-(n,1),F_1))))))),*(c_1,\exp(*(*(k,-(n,1)),x))))))$.

(Mç11)$\forall x\ (=(y,+(*(/(1,\exp(\text{intg}(A))),\text{intg}(*(B,\exp(\text{intg}(A))))),/(c_1,\exp(\text{intg}(A)))))$
$\wedge=(\text{intg}(A),F_1)\wedge=(\text{intg}(*(B,\exp(F_1))),F_2)\rightarrow=(*(\exp(F_1),y),+(F_2,/(c_1,$
$\exp(F_1))))))$.

(Mç12) $\forall x\ (=(y,*(c_1,\exp(\text{intg}(-(B,A)))))\wedge=(\text{intg}(-(B,A)),F_1) \rightarrow =(y,*(c_1,\exp(F_1))))$.

(Mç13) $\forall x\ (=(\text{pow}(y,-(1,n)),+(*(*(-(1,n),\exp(*(-(n,1),\text{intg}(A)))),\text{intg}(*(B,\exp(*(-$
$(1,n),\ \text{intg}(A))))))),*(*(-(1,n),c_1),\ \exp(*(-(n,1),\text{intg}(A))))))\wedge$
$=(\text{intg}(A),F_1)\wedge=(\text{intg}(*(B,\exp(*(-(1,n),F_1)))),F_2) \rightarrow =(\text{pow}(y,-(1,n)),$
$+(*(*(-(1,n),\exp(*(-(n,1),\ F_1))),F_2),*(*(-(1,n),c_1),\exp(*(-(n,1),F_1))))))$.

İspat : Önteorem 5.2.1.'de elde edilen, diferensiyel denklemin temel durumlarındaki çözümlerinden (Mt5.1)'de çözüm bölümü,

$$=(y,+(\text{intg}(B),c_1))$$

biçiminde integral içeren bir çözümdür.Dolayısıyla Tanım 5.2.3.'den, (Mt5.1)'den elde edilecek integral içeren çözüme karşılık gelen mantıksal form,

$$\forall x\ (=(y,+(\text{intg}(B),c_1))\wedge=(\text{intg}(B),F_1) \rightarrow =(y,+(F_1,c_1)))$$

biçiminde olur.

Yine, diferensiyel denklemin temel durumlarındaki çözümlerinden (Mt5.2)'de çözüm bölümü,

$$=(y,+(\exp(\text{intg}(B)),c_1))$$

biçiminde integral içeren çözümdür. Buradan, Tanım 5.2.3. kullanılarak (Mt5.2)'in integral içeren çözüm bölümüne karşılık gelen mantıksal form,

$$\forall x\ (=(y,+(\exp(\text{intg}(B)),c_1))\wedge=(\text{intg}(B),F_1) \rightarrow =(y,+(\exp(F_1),c_1)))$$

olur.

(Mt5.2), (Mt6),..., (Mg1),..., (Mg3) formüllerinden elde edilecek integral içeren çözümlere karşılık gelen mantıksal formların sırasıyla (Mç3), (Mç4),...,(Mç13) oldukları benzer şekilde ispatlanabilir.

5.3. Program Clause'larının Oluşturulması

Kesim 5.2.'de elde edilen mantıksal formların standart forma getirilmesi ile oluşturulan clause'lar, bu mantıksal formlar aynı zamanda birer program deyimi de olacaklarından, program clause'u olarak adlandırılabilirler. Bu clause'lara, türev ve integrasyon aksiyomlarından üretilecek clause'larıda ekleyerek, P_{dif} programını tanımlayabiliriz.

5.3.1. Gösterim : Türev ve integrasyon kurallarından elde edilecek aksiyomların standart forma getirilmeleri ile oluşturulan clause kümesi, $\Re$ sembolü ile gösterilecektir.

5.3.1. Tanım : C_i'ler (i=1,2,...,n) herhangi bir diferensiyel denklemi temsil eden clause'lar olmak üzere, belirtilen diferensiyel denkleme karşılık gelen P_{dif} programı,

$$P_{dif} = \{C_1, C_2, \ldots, C_n\} \cup \Re$$

olarak tanımlıdır.

5.3.1. Önteorem : C_1, C_2, ... ,C_{34} clause'ları aşağıdaki gibi verilmiş olsunlar. Bu durumda birinci mertebeden doğrusal diferensiyel denklemlere karşılık gelen P^L_{dif} programı,

$$P^L_{dif} = \{C_1, C_2, \ldots, C_{34}\} \cup \Re$$

biçiminde olur.

C_1 $\neg$=(diff(y,x),0)$\vee$=(y,c_1).

C_2 $\neg$=(diff(y,x),k)$\vee$=(y,+(*(k,x),c_1)).

C_3 $\neg$=(diff(y,x),*(k,y))$\vee$=(y,*(c_1,exp(*(k,x)))).

C_4 $\neg$=(diff(y,x),*(k,pow(y,n)))$\vee$=(pow(y,-(1,n)),*(-(1,n),+(c_1,*(k,x)))).

C_5 $\neg$=(+(diff(y,x),*(k,y)),0)$\vee$=(y,/(c_1,exp(* (k,x)))).

C_6 $\neg$=(+(diff(y,x),*(k,y)),r)$\vee$=(y,+(/(r,k),/(c_1,exp(*(k,x))))).

C_7 $\neg$=(+(diff(y,x),*(k,y)),*(r,y))$\vee$=(y,*(c_1,exp(*(-(r,k),x)))).

C_8 $\neg$=(+(diff(y,x),*(k,y)),*(r,pow(y,n))$\vee$=(pow(y,-(1,n)),+(/(r,k),
(c_1,exp((k,*(-(n,r) ,x)))))).

C_9 $\neg$=(diff(y,x),B)$\vee$=(y,+(intg(B),c_1)).

C_{10} $\neg$=(diff(y,x),*(B,y))$\vee$=(y,*(c_1,exp(intg(B)))).

C_{11} $\neg$=(diff(y,x),*(B,pow(y,n)))$\vee$=(pow(y,-(1,n)),+(*(-(1,n),intg(B)),c_1)).

C_{12} $\neg$=(+(diff(y,x),*(A,y)),0)$\vee$=(y,/(c_1,exp(intg(A)))).

C_{13} $\neg$=(+(diff(y,x),*(k,y)),B)$\vee$=(y,+(*(/(1,exp(*(k,x))),
intg(*(B,exp(*(k,x)))),/(c_1,exp(*(k,x))))).

C_{14} $\neg$=(+(diff(y,x),*(k,y)),*(B,y))$\vee$=(y,*(c_1,exp(-(intg(B),*(k,x)))))).

C_{15} $\neg$=(+(diff(y,x),*(A,y)),*(B,pow(y,n)))$\vee$=(pow(y,-(1,n)),+(*(*(-(1,n),
exp(*(*(k,-(n,1)),x))),intg(*(B,exp(*(*(k,-(1,n)),x))),*(c_1,exp(*(*(k,-
(n,1)),x)))).

C_{16} $\neg$=(+(diff(y,x),*(A,y)),k)$\vee$=(y,*(/(k,exp(intg(A))),+(intg(exp(intg(A))),c_1))).

C_{17} $\neg$=(+(diff(y,x),*(A,y)),*(k,y))$\vee$=(y,*(c_1,exp(-(*(k,x),intg(A))))).

C_{18} $\neg$=(+(diff(y,x),*(A,y)),*(k,pow(y,n)))$\vee$=(pow(y,-(1,n)),+(*(*(*(k,-(1,n)), exp(*(-(n,1),intg(A)))),intg(exp(*(-(1,n),intg(A)))),*(c_1,exp(*(-(n,1),intg(A))))).

C_{19} $\neg$=(+(diff(y,x),*(A,y)),B)$\vee$=(y,+(*(/(1,exp(intg(A))), intg(*(B,exp(intg(A))))),/(c_1,exp(intg(A))))).

C_{20} $\neg$=(+(diff(y,x),*(A,y)),*(B,y))$\vee$=(y,*(c_1,exp(intg(-(B,A))))).

C_{21} $\neg$=(+(diff(y,x),*(A,y)),*(B,pow(y,n)))$\vee$=(pow(y,-(1,n)),+(*(*(-(1,n), exp(*(-(n,1),intg(A)))),intg(*(B,exp(*(-(1,n),intg(A)))))),*(*(-(1,n),c_1), exp(*(-(n,1),intg(A))))).

C_{22} $\neg$=(y,+(intg(B),c_1))$\vee\neg$=(intg(B),F_1)$\vee$=(y,+(F_1,c_1)).

C_{23} $\neg$=(y,*(c_1,exp(intg(B))))$\vee\neg$=(intg(B),F_1)$\vee$=(y,*(c_1,exp(F_1))).

C_{24} $\neg$=(pow(y,-(1,n)),+(*(-(1,n),intg(B)),c_1)$\vee\neg$=(intg(B),F_1)$\vee$ =(pow(y,-(1,n)),+(*(-(1,n),F_1), c_1)).

C_{25} $\neg$=(y,/(c_1,exp(intg(A))))$\vee\neg$=(intg(A),F_1)$\vee$=(y,/(c_1,exp(F_1))).

C_{26} $\neg$=(y,+(*(/(1,exp(*(k,x))),intg(*(B,exp(*(k,x))))),/(c_1,exp(*(k,x)))))$\vee$ $\neg$=(intg(*(B,exp(*(k,x)))),F_1)$\vee$=(y,+(*(/(1,exp(*(k,x))),F_1),/(c_1,exp(*(k,x))))).

C_{27} $\neg$=(y,*(c_1,exp(-(intg(B),*(k,x)))))$\vee\neg$=(intg(B),F_1)$\vee$=(y,*(c_1,exp(-(F_1,*(k,x))))).

C_{28} $\neg$=(pow(y,-(1,n)),+(*(*(-(1,n),exp(*(*(k,-(n,1)),x))), intg(*(B,exp(*(*(k,-(1,n)),x)))),*(c_1, exp(*(*(k,-(n,1)),x))))$\vee$ $\neg$=(intg(*(B,exp(*(*(k,-(1,n)),x)))),F_1)$\vee$=(pow(y,-(1,n)),+(*(*(-(1,n),exp(*(*(k,-(n,1)),x))),F_1),*(c_1, exp(*(*(k,-(n,1)),x))))).

C_{29} $\neg$=(y,*(/(k,exp(intg(A))),+(intg(exp(intg(A))),c_1)))$\vee\neg$=(intg(A),F_1)$\vee$ =(intg(exp(F_1)),F_2)$\vee$=(y,*(/(k,exp(F_1)),+(F_2,c_1))).

C_{30} $\neg$=(y,*(c_1,exp(-(*(k,x),intg(A)))))$\vee\neg$=(intg(A),F_1)$\vee$=(y,*(c_1,exp(-(*(k,x),F_1))))).

C_{31} $\neg$=(pow(y,-(1,n)),+(*(*(*(k,-(1,n)), exp(*(-(n,1),intg(A)))), intg(exp(*(-(1,n),intg(A)))),*(c_1,exp(*(-(n,1),intg(A)))))))),

$*(c_1,exp(*(*(k,-(n,1)),x)))) \vee \neg=(intg(A),F_1) \vee \neg=(intg(exp(*(-(1,n),F_1))),F_2)) \vee =(pow(y,-(1,n)),+(*(*(*(k,-(1,n)),exp(*(-(n,1),F_1))),F_2,*(c_1,exp(*(-(n,1),F_1)))))),*(c_1,exp(*(*(k,-(n,1)),x))))).$

C_{32} $\neg=(y,+(*(/(1,exp(intg(A))),intg(*(B,exp(intg(A))))),c_1))) \vee \neg=(intg(A),F_1) \vee \neg=(intg(*(B,exp(F_1))),F_2) \vee =(*(exp(F_1),y),+(F_2,c_1)))$.

C_{33} $\neg=(y,*(c_1,exp(intg(-(B,A))))) \vee \neg=(intg(-(B,A)),F_1) \vee =(y,*(c_1,exp(F_1)))$.

C_{34} $\neg=(pow(y,-(1,n)),+(*(*(-(1,n),exp(*(-(n,1),intg(A)))),intg(*(B,exp(*(-(1,n),intg(A)))))),*(*(-(1,n),c_1),exp(*(-(n,1),intg(A)))))) \vee \neg=(intg(A),F_1) \vee \neg=(intg(*(B,exp(*(-(1,n),F_1)))),F_2) \vee =(y,+(*(*(-(1,n),exp(*(-(n,1),F_1))),F_2),*(*(-(1,n),c_1),exp(*(-(n,1),F_1)))))$.

İspat : Önteorem 5.2.1.'in (i) şıkkı gereğince,

$$\forall x\ (=(diff(y,x),0) \rightarrow =(y,c_1)) \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

formülünün geçerli olduğunu biliyoruz. (1) formülü standart forma getirilirse,

$$\neg=(diff(y,x),0) \vee =(y,c_1)$$

clause'u elde edilir. Bu ise C_1 clause'udur.

Aynı şekilde, Önteorem 5.2.2.'den,

$$\forall x\ (=(y,+(intg(B),c_1)) \wedge =(intg(B),F_1) \rightarrow =(y,+(F_1,c_1))) \ldots\ldots\ldots\ldots\ldots\ldots (2)$$

formülü geçerlidir. (2) formülü standart forma getirilirse,

$$\neg=(y,+(intg(B),c_1)) \vee \neg=(intg(B),F_1) \vee =(y,+(F_1,c_1))$$

biçiminde C_{22} clause'u elde edilir.

Diğer clause'lar da benzer şekilde Önteorem 5.2.1. ve Önteorem 5.2.2.'nin birer sonucu olarak elde edilebilirler.

5.3.1. Örnek : DE[$x*cos(x^2),0,0$] diferensiyel denkleminin çözümü için P^L_{dif} programının dedüksiyonu aşağıdaki gibi olacaktır:

(1) $=(+(diff(y,x),*(*(x,cos(pow(x,2))),y),0)$

Verilen diferensiyel denkleme MD operatörü uygulanarak elde edilen formül.

(2) $=(y,/(c_1,exp(intg(*(x,cos(pow(x,2)))))))$

(1) ve C_{12} clause'larının resolventi; $\sigma_1=\{*(x,cos(pow(x,2)))/A\}$.

(3) $\neg=(intg(*(x,cos(pow(x,2)))),F_1) \vee =(y,/(c_1,exp(F_1)))$

(2) ve C_{25} clause'larının resolventi; $\sigma_2=\sigma_1$.

Matematiksel olarak,

$$\frac{dU(x)}{dx} = kV(x) \rightarrow \int V(x)\cos(U(x))dx = \frac{\sin(U(x))}{k}$$

formülü geçerli olacağından,

(4) $\neg d(U,*(k,V))\vee=(intg(*(V,cos(U))),/(sin(U),k))$.

(5) $\neg d(pow(x,2),*(k,x))\vee=(y,/(c_1,exp(/(sin(pow(x,2)),k))))$

(3) ve (4) clause'larının resolventi; $\sigma_3=\{x/V, pow(x,2)/U, /(sin(U),k)/F_1\}$.

ve buradan, $\frac{d(x^2)}{dx} = 2x$ olduğu bilindiğine göre,

(5) $d(pow(x,2),*(2,x))$

(6) $=(y,/(c_1,exp(/(sin(pow(x,2)),2))))$

(4) ve (5) clause'larının resolventi $\phi_1=\{2/k\}$.

biçiminde, $y = \frac{c_1}{e^{\sin(x^2)/2}}$ çözümü elde edilmiş olur.

5.4. P^L_{dif} Programının Yönlü Graf Gösterimi

Önteorem 5.3.1. ile oluşturulan P^L_{dif} programını, Tanım 3.3.1.'i kullanarak Şekil 5.2.'deki gibi bir yönlü grafla gösterebiliriz. Bu gösterimde, Tanım 5.1.4.'de verilen predicate'lere ek olarak, t_1 ve t_2 iki terim olmak üzere;

$$b_i(t_1,t_2) : t_1 > t_2 \quad i\in\{1,2,...,27\}$$

predicate'i kullanılmıştır. Ayrıca, arklar üzerinde program değişkenlerine atanan m_i değerleri, Çizelge 5.1.'de görüldüğü gibi olacaklardır.

Yönlü grafdan elde edilecek tanımlama clause'ları, $A_{P^L_{dif}}$ kümesini oluştururlar. Bu durumda $A_{P^L_{dif}}$ kümesi;

$A_{P^L_{dif}} = \{(1)\ \neg=_1(x_1,0)\vee\neg=_1(x_2,0)\vee\neg=_1(x_3,n)\vee q_H(x_1,x_2,x_3,m_1),$

(2) $\neg=_2(x_1,0)\vee\neg=_2(x_2,k)\vee\neg=_2(x_3,n)\vee q_1(x_1,x_2,x_3,n),$

(3) $\neg q_1(x_1,x_2,x_3,y)\vee\neg=_3(y,0)\vee q_H(x_1,x_2,x_3,m_2),$

(4) $\neg q_1(x_1,x_2,x_3,y)\vee\neg=_4(y,1)\vee q_H(x_1,x_2,x_3,m_3),$

(5) $\neg q_1(x_1,x_2,x_3,y)\vee\neg b_5(y,1)\vee q_H(x_1,x_2,x_3,m_4)$,

(6) $\neg\neg_6(x_1,k)\vee\neg\neg_6(x_2,0)\vee\neg\neg_6(x_3,n)\vee q_H(x_1,x_2,x_3,m_5)$,

(7) $\neg\neg_7(x_1,k)\vee\neg\neg_7(x_2,r)\vee\neg\neg_7(x_3,n)\vee q_2(x_1,x_2,x_3,n)$,

(8) $\neg q_2(x_1,x_2,x_3,y)\vee\neg\neg_8(y,0)\vee q_H(x_1,x_2,x_3,m_6)$,

(9) $\neg q_2(x_1,x_2,x_3,y)\vee\neg\neg_9(y,1)\vee q_H(x_1,x_2,x_3,m_7)$,

(10) $\neg q_2(x_1,x_2,x_3,y)\vee\neg b_{10}(y,1)\vee q_H(x_1,x_2,x_3,m_8)$,

(11) $\neg\neg_{11}(x_1,0)\vee\neg\neg_{11}(x_2,B(x))\vee\neg\neg_{11}(x_3,n)\vee q_3(x_1,x_2,x_3,n)$,

(12) $\neg q_3(x_1,x_2,x_3,y)\vee\neg\neg_{12}(y,0)\vee q_H(x_1,x_2,x_3,m_9)$,

(13) $\neg q_3(x_1,x_2,x_3,y)\vee\neg\neg_{13}(y,1)\vee q_H(x_1,x_2,x_3,m_{10})$,

(14) $\neg q_3(x_1,x_2,x_3,y)\vee\neg b_{14}(y,1)\vee q_H(x_1,x_2,x_3,m_{11})$,

(15) $\neg\neg_{15}(x_1,A(x))\vee\neg\neg_{15}(x_2,0)\vee\neg\neg_{15}(x_3,n)\vee q_H(x_1,x_2,x_3,m_{12})$,

(16) $\neg\neg_{16}(x_1,k)\vee\neg\neg_{16}(x_2,B(x))\vee\neg\neg_{16}(x_3,n)\vee q_4(x_1,x_2,x_3,n)$,

(17) $\neg q_4(x_1,x_2,x_3,y)\vee\neg\neg_{17}(y,0)\vee q_H(x_1,x_2,x_3,m_{13})$,

(18) $\neg q_4(x_1,x_2,x_3,y)\vee\neg\neg_{18}(y,1)\vee q_H(x_1,x_2,x_3,m_{14})$,

(19) $\neg q_4(x_1,x_2,x_3,y)\vee\neg b_{19}(y,1)\vee q_H(x_1,x_2,x_3,m_{15})$,

(20) $\neg\neg_{20}(x_1,A(x))\vee\neg\neg_{16}(x_2,k)\vee\neg\neg_{16}(x_3,n)\vee q_5(x_1,x_2,x_3,n)$,

(21) $\neg q_5(x_1,x_2,x_3,y)\vee\neg\neg_{21}(y,0)\vee q_H(x_1,x_2,x_3,m_{16})$,

(22) $\neg q_5(x_1,x_2,x_3,y)\vee\neg\neg_{22}(y,1)\vee q_H(x_1,x_2,x_3,m_{17})$,

(23) $\neg q_5(x_1,x_2,x_3,y)\vee\neg b_{23}(y,1)\vee q_H(x_1,x_2,x_3,m_{18})$,

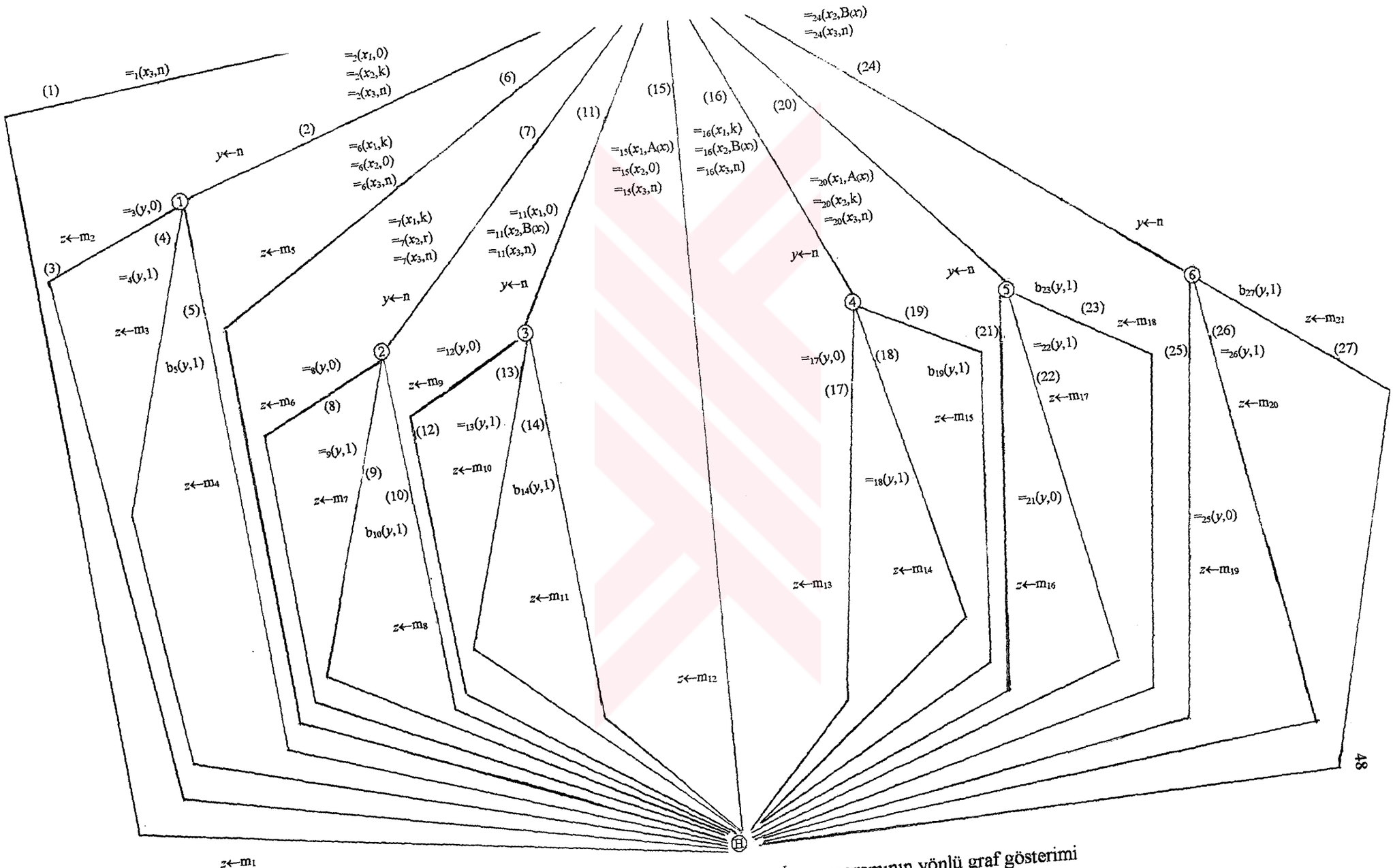
(24) $\neg\neg_{24}(x_1,A(x))\vee\neg\neg_{24}(x_2,B(x))\vee\neg\neg_{24}(x_3,n)\vee q_6(x_1,x_2,x_3,n)$,

(25) $\neg q_6(x_1,x_2,x_3,y)\vee\neg\neg_{25}(y,0)\vee q_H(x_1,x_2,x_3,m_{19})$,

(26) $\neg q_6(x_1,x_2,x_3,y)\vee\neg\neg_{26}(y,1)\vee q_H(x_1,x_2,x_3,m_{20})$,

(27) $\neg q_6(x_1,x_2,x_3,y)\vee\neg b_{27}(y,1)\vee q_H(x_1,x_2,x_3,m_{21})$ }

olarak bulunabilir.



Şekil 5.2. DE[x_1,x_2,x_3] diferensiyel denklemi için P^{L}_{dif} programının yönlü graf gösterimi

m_1 : c_1

m_2 : +(*(k,x),c_1)

m_3 : *(c_1 ,exp(*(k,x)))

m_4 : root(*(-(1,n),+(c_1 ,*(k,x))),-(1,n))

m_5 : /(c_1 ,exp(* (k,x)))

m_6 : +(/(r,k),/(c_1 ,exp(*(k,x))))

m_7 : *(c_1 ,exp(*(-(r,k),x)))

m_8 : root(+(/(r,k),*(c_1 ,exp(*(k,*(-(n,r) ,x))))),-(1.n))

m_9 : +(intg(B(x)),c_1)

m_{10} : *(c_1,exp(intg(B(x))))

m_{11} : root(+(*(-(1,n),intg(B(x))),c_1),-(1,n))

m_{12} : /(c_1 ,exp(intg(A(x))))

m_{13} : +(*(/(1,exp(*(k,x))),intg(*(B(x),exp(*(k,x))))./(c_1 ,exp(*(k,x))))

m_{14} : *(c_1 ,exp(-(intg(B(x)),*(k,x))))

m_{15} : root(+(*(*(-(1,n),exp(*(*(k,-(n,1)),x))),intg(*(B(x),exp(*(*(k,-(1,n)),x))),*(c_1 ,exp(*(*(k,-(n,1)),x))), -(1,n))

m_{16} : *(/(k,exp(intg(A(x)))),+(intg(exp(intg(A(x)))),c_1))

m_{17} : *(c_1 ,exp(-(*(k,x),intg(A(x)))))

m_{18} : root(+(*(*(*(k,-(1,n)),exp(*(-(n,1),intg(A(x))))),intg(exp(*(-(1,n),intg(A(x))))),*(c_1,exp(*(-(n,1),intg(A(x)))),-(1,n))

m_{19} : +(*(/(1,exp(intg(A(x)))),intg(*(B(x),exp(intg(A(x)))))./c_1 (exp(intg(A(x)))))

m_{20} : *(c_1 ,exp(intg(-(B(x),A(x)))))

m_{21} : +(*(*(-(1,n),exp(*(-(n,1),intg(A(x))))),intg(*(B(x),exp(*(-(1,n),intg(A(x)))))),*(*(-(1,n),c_1), exp(*(-(n,1),intg(A(x))))))

Çizelge 5.1. P^L_{dif} programının yönlü grafinda kullanılan m_i değerleri.

5.4.1. Tanım : A_i, $1 \le i \le 34$, $A_{P^L_{dif}}$ kümesinin $q_H(x_1,x_2,x_3,m_j)$ predicate'ini bulunduran herhangi bir elemanı olsun. Eğer m_j $(1 \le j \le 21)$ formülü, **intg** fonksiyonunu içeriyorsa, A_i'ye $A_{P^L_{dif}}$ 'nin *integral içeren durdurma clause'u* adı verilir.

5.4.2. Tanım : A_i, $A_{P^L_{dif}}$ 'nin herhangi bir integral içeren durdurma clause'u ve $q_H(x_1,x_2,x_3,m_j)$ A_i'de bulunan durdurma predicate'i olsun. m_j'de bulunan **intg** fonksiyonlarının intg($f_1(x)$), intg($f_2(x)$),..., intg($f_n(x)$) biçiminde olduklarını varsayalım. Bu durumda, A_i'ye karşılık gelen *integrasyon clause'u*,

$$\neg q_H(x_1,x_2,x_3,m_j)\vee\neg=(intg(f_1(x)),F_1)\vee \ldots\vee\neg=(intg(f_n(x)),F_n)\vee q_H(x_1,x_2,x_3,m_j')$$

olarak tanımlanır. Burada m_j', m_j terimine,

$$\alpha=\{F_1/\ intg(f_1(x)),F_2/\ intg(f_2(x)),\ldots,F_n/intg(f_n(x))\}$$

özel dönüşümünün uygulanmış biçimidir.

5.4.3. Tanım : $A_{P_{dif}^L}$ 'nin tüm integrasyon clause'larının oluşturdukları kümeye $A_{P_{dif}^L}$ 'nin *integrasyon clause'larının kümesi* denir ve bu küme $I_{P_{dif}^L}$ ile gösterilir.

5.4.4. Tanım : $A_{P_{dif}^L}$ 'nin *çözüm optimizasyonu kümesi*

$$E_{P_{dif}^L} = I_{P_{dif}^L} \cup \Re$$

biçiminde tanımlıdır.

5.4.1. Teorem : $A_{P_{dif}^L} \cup E_{P_{dif}^L}$ kümesi sağlanabilirdir.

İspat : P_{dif}^L 'de bulunan her bir *a* arkının tanımlama clause'u, bir $q_i(\bar{x},\bar{y})$ ya da $q_H(\bar{x},\bar{z})$ pozitif literalini bulundurur. Dolayısıyla $A_{P_{dif}^L}$ kümesini oluşturan her bir clause, bu şekilde bir pozitif literali bulunduracaktır. $a_1,a_2,\ldots,a_m$; P_{dif}^L 'nin S verteksinden farklı tüm verteksleri olmak üzere, I_1, her bir $q_i(\bar{x},\bar{y})$ $(1 \le i \le m)$ ve $q_H(\bar{x},\bar{z})$ predicate'ine **T** değerinin atanacağı bir yorumlama olsun. Bu durumda $A_{P_{dif}^L}$, I_1 içinde **T** değerini alır. Tanım 5.4.2. gereğince, her $K \in I_{P_{dif}^L}$ için K, $q_H(\bar{x},\bar{z})$ predicate'ini pozitif olarak bulunduracağından, $I_{P_{dif}^L}$ kümesi de I_1 içinde **T** değerini alacaktır. O halde $A_{P_{dif}^L} \cup I_{P_{dif}^L}$ kümesi sağlanabilirdir. Öte yandan, $\Re$ kümesi sağlanabilir olduğundan, **T** değerini alacağı en az bir I_2 yorumlamasına sahiptir. $\Re$ kümesinin elemanlarından hiçbirisi, bir q_i erişim predicate'ini bulundurmayacağından, I yorumlamasını $I=I_1\cup I_2$ olarak tanımlayabiliriz. Bu durumda $A_{P_{dif}^L} \cup I_{P_{dif}^L} \cup \Re$ kümesi, I içinde **T** değerini alacaktır. O halde $A_{P_{dif}^L} \cup E_{P_{dif}^L}$ kümesi sağlanabilirdir

5.5. P^L_{dif} Programının Sonlanması

5.5.1. Önteorem : A_T, $A_{P^L_{dif}}$ kümesinin elemanlarından q_H predicate'inin içeren her bir literalin silinmesiyle elde edilen küme olmak üzere, P^L_{dif} programının sonlanması için gerekli ve yeterli koşul, $A_T \cup E_{P^L_{dif}}$ kümesinin sağlanamaz olmasıdır.

İspat :($\Rightarrow$) P^L_{dif} programının sonlandığını varsayalım. Bu durumda, $\bar{x}$'in herhangi bir değeri için, P^L_{dif} programında en az bir a arkı vardır; öyle ki, denetim bu a arkı üzerinden H verteksine geçer. ω_i, a arkının başlangıç verteksi olsun. Bu durumda, $p_a(\bar{x},\bar{y})$ ve $z \leftarrow f_a(\bar{x},\bar{y})$, sırasıyla a arkına karşılık gelen denetim predicate'i ve değer ataması olmak üzere, a arkının tanımlama clause'u;

$$\neg q_i(\bar{x},\bar{y}) \vee \neg p_a(\bar{x},\bar{y}) \vee q_H(\bar{x}, f_a(\bar{x},\bar{y}))$$

biçiminde olacaktır. Denetim a arkı üzerinden, ω_i verteksinden H verteksine geçtiğinden, $q_i(\bar{x},\bar{y})$ ve $p_a(\bar{x},\bar{y})$'nin her ikisinin de **T** değerini almış olmaları gerekir. Bu durumda, $\neg q_i(\bar{x},\bar{y}) \vee \neg p_a(\bar{x},\bar{y})$, **F** değerini alır. Öte yandan $\neg q_i(\bar{x},\bar{y}) \vee \neg p_a(\bar{x},\bar{y})$ aynı zamanda A_T içersinde bulunduğundan, $A_T \cup E_{P^L_{dif}}$ kümesi sağlanamaz olur.

($\Leftarrow$)$A_T \cup E_{P^L_{dif}}$ kümesi sağlanamaz olsun. Teorem 5.4.1.'den, $A_{P^L_{dif}} \cup E_{P^L_{dif}}$ kümesi sağlanabilirdir. Diğer yandan, aynı teoremin ispatından $A_{P^L_{dif}} \cup E_{P^L_{dif}}$ kümesi için, q_i erişim predicate'ini içeren herhangi bir atomun **T** değerini alacağı bir I yorumlamasının bulunabileceğini biliyoruz. $A_T \cup E_{P^L_{dif}}$ kümesi sağlanamaz olduğundan, I 'da **F** değerini alacaktır. Dolayısıyla $E_{P^L_{dif}}$, I 'da **T** değerini aldığına göre, A_T, I 'da **F** değerini alacaktır. I, hiçbir q_i predicate'inin değilini içermediğinden, I 'da **F** değerini alacak $C'_a \in A_T$ clause'u hiçbir pozitif literal

içermemelidir. Öte yandan C'_a , $C_a \in A_{P^L_{dif}}$ clause'undan q_H'yi içeren literallerin silinmesiyle elde edilmiş olduğundan,

$$\neg q_i(\bar{x}, \bar{y}) \vee \neg p_a(\bar{x}, \bar{y})$$

biçiminde olacaktır. C'_a, I'da **F** değerini aldığından, hem $q_i(\bar{x}, \bar{y})$ hem de $p_a(\bar{x}, \bar{y})$ **T** değerini alacaklardır. Bu durumda, denetim a_i verteksinden H verteksine geçer ve P^L_{dif} programı sonlanır.

5.5.1. Teorem : $S = A_{P^L_{dif}} \cup E_{P^L_{dif}}$ olsun. Bu durumda P^L_{dif}'nin durması için gerekli ve yeterli koşul, S'den durdurma clause'una bir dedüksiyonun bulunmasıdır.

İspat :($\Rightarrow$) $S' = A_T \cup E_{P^L_{dif}}$ olsun. P^L_{dif} durduğundan, Önteorem 5.5.1.'e göre S' sağlanabilirdir. Bu durumda S' den □ clause'una bir D' dedüksiyonu vardır. D, D' dedüksiyonundan q_H'yi içeren clause'ların tekrar yerlerine konmalarıyla elde edilecek dedüksiyon olsun. O halde D dedüksiyonu, sadece durdurma predicate'ini içeren bir clause ile, yani durdurma clause'u ile sonlanan bir dedüksiyon olacaktır.

($\Leftarrow$) S'den durdurma clause'una bir D dedüksiyonunun bulunduğunu varsayalım. S', S'deki tüm durdurma predicate'lerinin silinmesiyle elde edilen clause kümesi olsun. Bu durumda D dedüksiyonu, S' den □ clause'una bir dedüksiyon olacaktır. O halde S' sağlanamazdır. $S' = A_T \cup E_{P^L_{dif}}$ olarak yazılabileceğinden, Önteorem 5.5.1. gereğince P^L_{dif} sonlanır.

6. ÖNERİLER

Birinci mertebeden doğrusal diferensiyel denklemlerin dedaktif yaklaşımla sembolik hesapları için geliştirilen yöntemin, diğer diferensiyel denklem türlerinin sembolik çözümlerinde de etkin biçimde uygulama alanı bulacağı ümit edilmektedir.

KAYNAKLAR

Akyıldız, E., Alpay, Ş., Erkip, A., 1990, Differential Equations : METU, Ankara.

Bateman, S., O., Jr., and Danielopoulos, S., D., 1981, Computerized Analytic Solutions of Second Order Differential Equations : Computer J., 24, 180-183.

Busacker, R., G., and Saaty, T., L., 1965, Finite Graphs and Networks : an Introduction with Applications : McGraw-Hill, New York.

Chang, C., L., and Lee, R., C., T., 1973, Symbolic Logic and Mechanical Theorem Proving : Academic Press, London.

Clark, K., L., 1978, Logic and Data Bases : Plenum Press, New York, 293-322.

Davis, M., and Putnam, H., 1960, A Computing Procedure for Quantification Theory : J. Assoc. Comput. Mach., 7, 201-215.

Dewar, M., C., 1989, An Integrated Symbolic and Numeric Computation Environment : Proc. ISSAC'89, ACM, New York, 171-179.

Fitting, M., 1990, First Order Logic and Automated Theorem Proving : Springer-Verlag, 97-152.

Floyd, R., W., 1967, Assigning Meaning to Programs : Proc. Symp. Appl. Math., American Mathematical Society, 19, 19-32.

Gilmore, P., C., 1960, A Proof Method for Quantification Theory : IBM J. Res. Develop., 4, 28-35.

Harrington, S., J., 1979, A New Symbolic Integration System in Reduce : Computer J., 22, 2.

Hayes, P., J., 1973, Computation and Deduction : Proc. MFCS Conf., Czechoslovak Academy of Sciences.

Kowalski, R., A., 1974, Information Processing 74, Stockholm, North Holland.

Lloyd, J., W., and Topor, R., W., 1984, Making Prolog More Expressive : J. Logic Programming , 1, 3, 225-240.

Lloyd, J., W., 1987, Foundations of Logic Programming : Springer-Verlag, 4-26, 35-136.

Loveland, D., W., 1978, Automated Theorem Proving : A Logical Bases : North Holland.

Manna, Z., 1969, The Correctness of Programs : J. Comput. System Sci., 3, 119-127.

Mignotte, M., 1982, Some Useful Bounds. Symbolic & Algebraic Computation (Computing Supplementum 4) : Springer-Verlag, Wien, New York.

Robinson, J., A., 1965, A Machine-Oriented Logic Based on the Resolution Principle : Journal of the Association for Computing Machinery, 12, 1, 23-41.

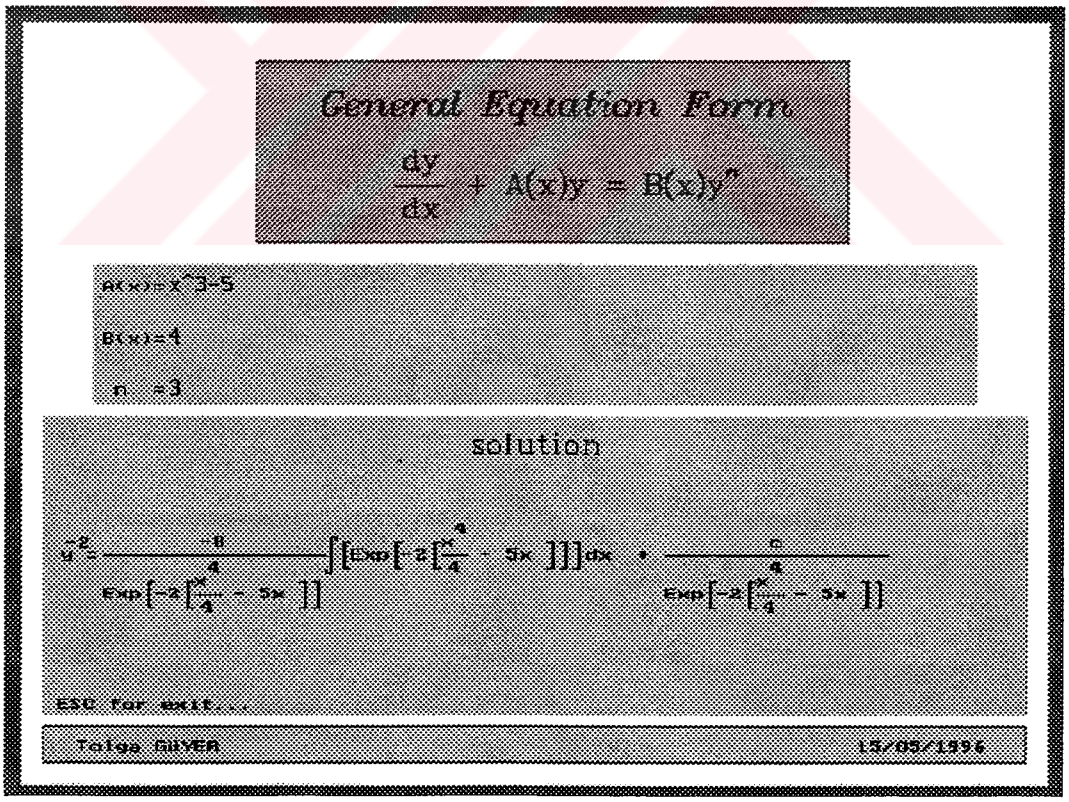
Schildt, H.,1987, Advanced Turbo Prolog : McGraw-Hill.

P^{L}_{dif} Programının PROLOG ile Simulasyonu EK-1

Listesi EK-2'de verilen PROLOG Programı, P^{L}_{dif} programı için elde edilen clause yapıları üzerinde sembolik hesaplama yaparak, integrasyon işleminde başarılı olduğu ölçüde sonuç vermektedir. Program, genel denklem yapısına uygun bir biçimde, $A_{(x)}$ ve $B_{(x)}$ katsayıları ile n üstel sabitini kullanıcıdan isteyerek genel çözümü, sadeleştirme aksiyomlarından geçirdikten sonra, matematiksel yazım formatında vermektedir.

İntegrasyonda karşılaşılan güçlükler nedeniyle $A_{(x)}$ ve $B_{(x)}$ katsayılarının türleri, polinom fonksiyonlarla sınırlandırılmıştır. n sabiti ise sıfır ya da pozitif bir tamsayı olarak belirlenmiştir.

Programın çalışmasından genel bir görünüm, aşağıda verilmiştir:



PROLOG Programının Listesi EK-2

```
"/*****************************************************************************/
/*                                                                           */
/*          Solutions of the Differential Equations of the Form:             */
/*                                                                           */
/*                          dy                 n                            */
/*                          ---- + A[x].y = B[x].y                           */
/*                          dx                                               */
/*                                                                           */
/*            with Symbolic Computation.                                     */
/*                                                                           */
/*       Tolga GÜYER                                                         */
/*       15/05/1996                                                          */
/*                                                                           */
/*****************************************************************************/

/*****************************************************************************/
/*                                                                           */
/*  Types of the Coefficients                                                */
/*  -------------------------                                                */
/*  A[x] : A Polynomial function.                                            */
/*  B[x] : A Polynomial function.                                            */
/*  n    : A pozitive integer or zero.                                       */
/*                                                                           */
/*****************************************************************************/

/*****************************************************************************/
/*                                                                           */
/*  Compiler            Best memory allocation for the compiling             */
/*  ----------          ----------------------------------------             */
/*  Turbo Prolog 2.0    Code array size  :5000                               */
/*                      Stack size       :1000                               */
/*                      Trail array size :10                                 */
/*                      Heap size        :0                                  */
/*                                                                           */
/*****************************************************************************/

code=5000

 DOMAINS

   FLIST=STRING*
   Fx =  var(STRING);
         int(INTEGER);
         i(Fx);
         pls(Fx,Fx);
         mlt(Fx,Fx);
         min(Fx,Fx);
         dvd(Fx,Fx);
         pts(Fx,Fx);
         exp(Fx)

 PREDICATES

   start
   eform
   name
   solution
   enter
   quit(char)
   getmin(integer,integer,integer)
```

```
  mwrite(Fx,integer,integer)
  ptscases(Fx,Fx,integer,integer,integer,integer,integer,integer)
  plmncases(Fx,Fx,integer,integer,integer,integer,integer,string)
  mltcases(Fx,Fx,integer,integer,integer,integer,integer,integer)
  dvdcases(Fx,Fx,integer,integer,integer,integer,integer,integer,integer,integer)
  fcases(Fx,integer,integer,integer,integer)
  specdvd(Fx,Fx,integer,integer,integer,integer,integer,integer)
  uxvx(integer,integer,integer,integer,integer,integer)
  len(Fx,integer)
  lenpls(integer,integer,integer,integer,integer)
  maxlen(integer,integer,integer)
  dvdline(integer,integer,integer)
  finddvd(Fx,integer)
  findpts(Fx,integer)
  d(Fx,Fx)
  intg(Fx,Fx,integer)
  reduce(Fx,Fx)
  plsr(Fx,Fx,Fx)
  minr(Fx,Fx,Fx)
  mltr(Fx,Fx,Fx)
  dvdr(Fx,Fx,Fx)
  ptsr(Fx,Fx,Fx)
  cases(Fx,Fx,integer,Fx,integer)
  capts(integer,Fx,Fx,Fx)
  writesol(Fx,integer)
  beginparsing(string,Fx)
  tolist(string,FLIST)
  parse(FLIST,FLIST,Fx)
  plus(FLIST,FLIST,Fx)
  subplus(FLIST,FLIST,Fx,Fx)
  mult(FLIST,FLIST,Fx)
  submult(FLIST,FLIST,Fx,Fx)
  expon(FLIST,FLIST,Fx)
  subexpon(FLIST,FLIST,Fx,Fx)
  subparse(FLIST,FLIST,Fx)
  first(string,FLIST,FLIST)

CLAUSES

start:-initgraph(0,5,_,_,"<Path for the BGI directory>"),
                                  /* Example : "C\\PROLOG\\BGI" */
       setbkcolor(1),
       setcolor(2),
       rectangle(5,5,635,475),
       name,
       solution,
       eform,
       setcolor(12),
       outtextxy(20,165,"->"),
       setcolor(15),
       setfillstyle(1,9),
       rectangle(50,155,590,240),
       outtextxy(55,165,"A(x)="),
       outtextxy(55,197,"B(x)="),
       outtextxy(55,228," n  ="),
       enter.

enter:-setcolor(12),
       outtextxy(20,165,"->"),
       setcolor(15),
       cursor(10,12),
       readln(A),
       setcolor(2),
       rectangle(5,5,635,475),
       setfillstyle(1,1),
```

```
        bar(10,160,35,175),
        setcolor(12),
        outtextxy(20,195,"->"),
        setcolor(15),
        rectangle(50,155,590,240),
        cursor(12,12),
        readln(B),
        setcolor(2),
        rectangle(5,5,635,475),
        setfillstyle(1,1),
        bar(10,190,35,205),
        setcolor(12),
        outtextxy(20,225,"->"),
        setcolor(15),
        rectangle(50,155,590,240),
        cursor(14,12),
        readint(N),
        setcolor(2),
        rectangle(5,5,635,475),
        setfillstyle(1,1),
        bar(10,220,35,235),
        rectangle(50,155,590,240),
        beginparsing(A,PA),
        beginparsing(B,PB),
        cases(PA,PB,N,Solution,P),
        reduce(Solution,RedSol),
        setcolor(10),
        writesol(RedSol,P).

writesol(S,1):-outtextxy(30,330,"y="),
                mwrite(S,47,330),
                setcolor(12),
                outtextxy(28,422,"ESC for exit..."),
                readchar(Ch),
                quit(Ch).
writesol(S,P):-outtextxy(30,330,"y ="),
                str_int(SP,P),
                outtextxy(33,323,SP),
                mwrite(S,56,330),
                setcolor(12),
                outtextxy(28,422,"ESC for exit..."),
                readchar(Ch),
                quit(Ch).

quit('\27'):-closegraph,
             exit.
quit(_):-start.

eform:-setfillstyle(1,11),
       setcolor(15),
        rectangle(148,28,512,142),
        bar(150,30,510,140),
        setcolor(12),
        settextstyle(7,0,3),
        outtextxy(190,40,"General Equation Form"),
        settextstyle(8,0,1),
        setcolor(1),
        line(235,106,265,106),
        outtextxy(230,90,"     + A(x)y = B(x)y "),
        outtextxy(230,75," dy"),
        outtextxy(230,105," dx"),
        settextstyle(0,0,1),
        outtextxy(277,94,"                  n").

solution:-setcolor(15),
```

```
            settextstyle(8,0,1),
            setfillstyle(1,9),
            bar(20,250,620,430),
            rectangle(18,248,622,432),
            outtextxy(283,250,"solution").

name:-setfillstyle(1,11),
        setcolor(15),
        rectangle(18,438,622,462),
        setcolor(1),
        bar(20,440,620,460),
        settextstyle(0,0,1),
        outtextxy(40,448,"Tolga GÜYER"),
        outtextxy(510,448," 15/05/1996").

/********************** EQUATION-CASES SECTION ***************/

/********************** Elementary Cases of the Equation ******/

cases(int(0),int(0),_,var("c"),1):-!.
cases(int(0),int(K),0,S,1):-
                S=pls(mlt(int(K),var("x")),var("c")),!.
cases(int(0),int(K),1,S,1):-
                S=mlt(var("c"),exp(mlt(int(K),var("x")))),!.
cases(int(0),int(K),N,S,P):-
                N>1,
                P=1-N,
                S=mlt(min(int(1),int(N)),pls(var("c"),mlt(int(K),var("x")))),!.
cases(int(K),int(0),_,S,1):-
                S=dvd(var("c"),exp(mlt(int(K),var("x")))),!.
cases(int(K),int(L),0,S,1):-
                S=pls(dvd(int(L),int(K)),dvd(var("c"),exp(mlt(int(K),var("x"))))),!.
cases(int(K),int(L),1,S,1):-
                F=L-K,
                S=mlt(var("c"),exp(mlt(int(F),var("x")))),!.
cases(int(K),int(L),N,S,P):-
                N>1,
                P=1-N,
                N1=N-1,
                KN1=K*N1,
S=pls(dvd(int(L),int(K)),mlt(var("c"),exp(mlt(int(KN1),var("x"))))),!.
cases(int(0),Bx,0,S,1):-
                intg(Bx,IBx,C),
                capts(C,Bx,IBx,I),
                S=pls(I,var("c")),!.
cases(int(0),Bx,1,S,1):-
                intg(Bx,IBx,C),
                capts(C,Bx,IBx,I),
                S=mlt(var("c"),exp(I)),!.
cases(int(0),Bx,N,S,P):-
                N>1,
                P=1-N,
                intg(Bx,IBx,C),
                capts(C,Bx,IBx,I),
                S=pls(mlt(int(P),I),var("c")),!.
cases(Ax,int(0),_,S,1):-
                intg(Ax,IAx,C),
                capts(C,Ax,IAx,I),
                S=dvd(var("c"),exp(I)),!.
cases(int(K),Bx,0,S,1):-
                reduce(mlt(Bx,exp(mlt(int(K),var("x")))),R),
                intg(R,I1,C),
                capts(C,mlt(Bx,exp(mlt(int(K),var("x")))),I1,I),
                S=pls(mlt(dvd(int(1),exp(mlt(int(K),var("x")))),I),
                dvd(var("c"),exp(mlt(int(K),var("x")))),!.
```

```
cases(int(K),Bx,1,S,1):-
                intg(Bx,IBx,C),
                capts(C,Bx,IBx,I),
                S=mlt(mlt(var("c"),exp(I)),dvd(int(1),exp(mlt(int(K),var("x")))),!.
cases(int(K),Bx,N,S,P):-
                N>1,
                P=1-N,
                Kp=K*P,
                reduce(mlt(Bx,exp(mlt(int(Kp),var("x")))),R),
                intg(R,I1,C),
                capts(C,mlt(Bx,exp(mlt(int(Kp),var("x")))),I1,I),
                S=pls(mlt(mlt(int(P),dvd(int(1),exp(mlt(int(Kp),var("x"))))),I),
                dvd(var("c"),exp(mlt(int(Kp),var("x")))),!.
 cases(Ax,int(K),0,S,1):-
                intg(Ax,IAx,C1),
                capts(C1,Ax,IAx,I1),
                S1=dvd(int(K),exp(I1)),
                intg(exp(I1),II,C2),
                capts(C2,exp(I1),II,I2),
                S=mlt(S1,pls(I2,var("c"))),!.
 cases(Ax,int(K),1,S,1):-
                intg(Ax,IAx,C),
                capts(C,Ax,IAx,I),
                S=mlt(mlt(var("c"),exp(mlt(int(K),var("x")))),dvd(int(1),exp(I))),!.
 cases(Ax,int(K),N,S,P):-
                N>1,
                P=1-N,
                Kp=K*P,
                intg(Ax,IAx,C1),
                capts(C1,Ax,IAx,I1),
                reduce(exp(mlt(int(P),I1)),R),
                intg(R,II,C2),
                capts(C2,exp(mlt(int(P),I1)),II,I2),
                S=pls(mlt(dvd(int(Kp),exp(mlt(int(P),I1))),I2),
                dvd(var("c"),exp(mlt(int(P),I1)))),!.

/********************** General Cases of the Equation ********/

cases(Ax,Bx,0,S,1):-
                intg(Ax,IAx,C1),
                capts(C1,Ax,IAx,I1),
                S1=dvd(int(1),exp(I1)),
                reduce(mlt(Bx,exp(I1)),R),
                intg(R,II,C2),
                capts(C2,mlt(Bx,exp(I1)),II,I2),
                S=pls(mlt(S1,I2),dvd(var("c"),exp(I1))),!.
cases(Ax,Bx,1,S,1):-
                intg(Bx,IBx,C1),
                capts(C1,Bx,IBx,I1),
                intg(Ax,IAx,C2),
                capts(C2,Ax,IAx,I2),
                S=dvd(mlt(var("c"),exp(I1)),exp(I2)),!.
cases(Ax,Bx,N,S,1):-
                N>1,
                P=1-N,
                intg(Ax,IAx,C1),
                capts(C1,Ax,IAx,I1),
                S1=dvd(int(P),exp(mlt(int(P),I1))),
                reduce(mlt(Bx,exp(mlt(int(P),I1))),R),
                intg(R,II,C2),
                capts(C2,mlt(Bx,exp(mlt(int(P),I1))),II,I2),
                S=mlt(S1,pls(I2,var("c"))),!.
                capts(1,_,I,RI):-reduce(I,RI),!.
                capts(0,B,_,i(B)):-!.
```

```
/************************ PARSING SECTION ***********************/

beginparsing(F,PF):-tolist(F,FLIST),
                    parse(FLIST,_,PF),!.

tolist(STR,[Head|Tail]):-fronttoken(STR,Head,STR1),
                         tolist(STR1,Tail),!.
tolist(_,[]).

parse(L1,L2,FX):-plus(L1,L2,FX),!.

plus(L1,L2,FX2):-mult(L1,L21,FX1),
                 subplus(L21,L2,FX1,FX2),!.
mult(L1,L2,FX2):-expon(L1,L21,FX1),
                 submult(L21,L2,FX1,FX2),!.

expon(L1,L2,FX2):-subparse(L1,L21,FX1),
                  subexpon(L21,L2,FX1,FX2),!.

subplus(["+"|L1],L2,FX1,FX3):-!,
                              mult(L1,L21,FX2),
                              subplus(L21,L2,pls(FX1,FX2),FX3).
subplus(["-"|L1],L2,FX1,FX3):-!,
                              mult(L1,L21,FX2),
                              subplus(L21,L2,min(FX1,FX2),FX3).
subplus(L1,L1,FX,FX).

submult(["*"|L1],L2,FX1,FX3):-!,
                             expon(L1,L21,FX2),
                             submult(L21,L2,mlt(FX1,FX2),FX3).
submult(["/"|L1],L2,FX1,FX3):-!,
                             expon(L1,L21,FX2),
                             submult(L21,L2,dvd(FX1,FX2),FX3).
submult(L1,L1,FX,FX).

subexpon(["^"|L1],L2,FX1,FX3):-!,
                              subparse(L1,L21,FX2),
                              subexpon(L21,L2,pts(FX1,FX2),FX3).
subexpon(L1,L1,FX,FX).

subparse(["exp","("|L1],L2,exp(FX)):-parse(L1,L21,FX),
                                     first(")",L21,L2),!.
subparse(["i","("|L1],L2,i(FX)):-parse(L1,L21,FX),
                                 first(")",L21,L2),!.
subparse(["("|L1],L2,FX):-parse(L1,L21,FX),
                          first(")",L21,L2),!.
subparse(["-",H2|L1],L1,int(INT)):-str_int(H2,INT1),
                                   INT=-INT1,!.
subparse([H|L1],L1,int(INT)):-str_int(H,INT),!.
subparse([N|L1],L1,var(N)).

first(H,[H|T],T).



/************************ DIFFERENTIATION SECTION ***************/

/************************ Derivation Rules **********************/

d(int(_),int(0)).
d(var(_),int(1)):-!.
d(pls(U,V),pls(U1,V1)):-
```

```
                                d(U,U1),
                                d(V,V1).
d(min(U,V),min(U1,V1)):-
                                d(U,U1),
                                d(V,V1).
d(mlt(int(I),U),mlt(int(I),U1)):-d(U,U1).
d(mlt(U,V),pls(mlt(U1,V),mlt(U,V1))):-
                                d(U,U1),
                                d(V,V1).
d(dvd(pts(var(X),int(N)),int(I)),mlt(int(N),dvd(pts(var(X),int(N1)),int(I)))):-
                                N1=N-1.
d(dvd(U,V),dvd(min(mlt(U1,V),mlt(U,V1)),mlt(V,V))):-
                                d(U,U1),
                                d(V,V1).
d(pts(E1,int(I)),mlt(mlt(int(I),pts(E1,int(I1))),EXP)):-
                                I1=I-1,
                                d(E1,EXP).
d(exp(U),mlt(U1,exp(U))):-
                                d(U,U1).

/************************ INTEGRATION SECTION ********************/

/************************ Integration Rules **********************/

getmin(X,Y,X):-X<Y.
getmin(_,Y,Y).

intg(int(I),mlt(int(I),var("x")),1).
intg(var(X),dvd(pts(var(X),int(2)),int(2)),1).
intg(mlt(int(I),U),mlt(int(I),U1),C):-
                                intg(U,U1,C).
intg(dvd(U,int(I)),dvd(U1,int(I)),C):-
                                intg(U,U1,C).
intg(pls(U,V),pls(U1,V1),C):-
                                intg(U,U1,C1),
                                intg(V,V1,C2),
                                getmin(C1,C2,C).
intg(min(U,V),min(U1,V1),C):-
                                intg(U,U1,C1),
                                intg(V,V1,C2),
                                getmin(C1,C2,C).
intg(pts(var(X),int(I)),dvd(pts(var(X),int(I1)),int(I1)),1):-
                                I1=I+1,!.
intg(exp(var(X)),exp(var(X)),1).
intg(exp(mlt(int(K),var(X))),dvd(exp(mlt(int(K),var(X))),int(K)),1).
intg(mlt(var(X),exp(var(X))),
        min(mlt(var(X),exp(var(X))),exp(var(X))),1).
intg(mlt(var(X),exp(mlt(int(K),var(X)))),
        min(dvd(mlt(var(X),exp(mlt(int(K),var(X)))),int(K)),
        dvd(exp(mlt(int(K),var(X))),int(K))),1).
intg(mlt(U,exp(var(X))),min(mlt(U,exp(var(X))),VDU),C):-
                                d(U,DU),
                                reduce(DU,RDU),
                                intg(mlt(RDU,exp(var(X))),VDU,C),!.
intg(mlt(DU,exp(U)),exp(U),1):-
                                d(U,DU).
intg(mlt(V,exp(U)),dvd(exp(U),int(K)),1):-
                                d(U,DU),
                                reduce(DU,RDU),
                                RDU=mlt(int(K),V).
intg(X,X,0):-!.

/************************ REDUCING SECTION **********************/

reduce(pls(X,Y),R):- !,
```

```
                    reduce(X,X1),
                    reduce(Y,Y1),
                    plsr(X1,Y1,R).
reduce(min(X,Y),R):-!,
                    reduce(X,X1),
                    reduce(Y,Y1),
                    minr(X1,Y1,R).
reduce(mlt(X,Y),R):-!,
                    reduce(X,X1),
                    reduce(Y,Y1),
                    mltr(X1,Y1,R).
reduce(dvd(X,Y),R):-!,
                    reduce(X,X1),
                    reduce(Y,Y1),
                    dvdr(X1,Y1,R).
reduce(pts(X,Y),R):-!,
                    reduce(X,X1),
                    reduce(Y,Y1),
                    ptsr(X1,Y1,R).
reduce(exp(U),exp(R)):-!,
                    reduce(U,R).
                    reduce(R,R).

/************************ Reduction of the additions ************/

plsr(int(0),X,X):-!.
plsr(X,int(0),X):-!.
plsr(int(X),int(Y),int(Z)):-!,
                    X+Y=Z.
plsr(X,X,mlt(int(2),X)):-!.
plsr(int(X),Y,Z) :-
                    X<0,
                    T=-X, !,
                    minr(int(T),Y,Z).
plsr(Y,int(X),Z) :-
                    X<0,
                    T=-X,!,
                    minr(int(T),Y,Z).
plsr(mlt(int(I),X),X,mlt(int(I1),X)):-!,
                    I+1=I1.
plsr(X,mlt(int(I),X),mlt(int(I1),X)):-!,
                    I+1=I1.
plsr(mlt(int(I1),X),mlt(int(I2),X),mlt(int(I3),X)):-!,
                    I1+I2=I3.
plsr(int(I),X,pls(X,int(I))):-!.
plsr(pls(X,int(I1)),int(I2),pls(X,int(I3))):-!,
                    I1+I2=I3.
plsr(pls(X,int(I1)),pls(Y,int(I2)),pls(R,int(I3))):-!,
                    I1+I2=I3,
                    plsr(X,Y,R).
plsr(pls(X,int(I)),Y,pls(R,int(I))):-!,
                    plsr(X,Y,R).
plsr(X,Y,pls(X,Y)).


/************************ Reduction of the subtractions *********/

minr(int(X),int(Y),int(Z)):-!,
                    Z=X-Y.
minr(X,int(0),X):-!.
minr(X,X,int(0)):-!.
minr(mlt(int(I1),X),mlt(int(I2),X),X):-
                    I1>I2,
                    1=I1-I2,!.
```

```
minr(mlt(int(I),X),X,X):-I=2,!.
minr(mlt(int(I1),X),mlt(int(I2),X),mlt(int(I),X)):-
                    I1>I2,
                    I=I1-I2,!.
minr(mlt(int(I),X),X,mlt(int(I1),X)):-
                    I1=I-1,!.
minr(X,Y,min(X,Y)).

/************************ Reduction of the multiplications ******/

mltr(int(X),int(Y),int(Z)):-!,
                    X*Y=Z.
mltr(int(0),_,int(0)):-!.
mltr(_,int(0),int(0)):-!.
mltr(int(1),X,X):-!.
mltr(X,int(1),X):-!.
mltr(int(I),dvd(X,int(I)),X):-!.
mltr(int(I1),dvd(X,int(I2)),mlt(int(I),X)):-
                    I1>I2,
                    I1 mod I2=0,
                    I=I1/I2.
mltr(int(I1),dvd(X,int(I2)),dvd(X,int(I))):-
                    I2>I1,
                    I2 mod I1=0,
                    I=I2/I1.
mltr(dvd(X,Y),dvd(Y,Z),dvd(X,Z)).
mltr(dvd(X,Y),dvd(Z,X),dvd(Z,Y)).
mltr(mlt(int(I1),X),int(I2),M1):-!,
                    I1*I2=I3,
                    mltr(int(I3),X,M1).
mltr(int(I1),mlt(int(I2),X),M1):-!,
                    I1*I2=I3,
                    mltr(int(I3),X,M1).
mltr(mlt(int(I1),X),mlt(int(I2),Y),mlt(int(I3),R)):-!,
                    I1*I2=I3,
                    mltr(X,Y,R).
mltr(mlt(int(I),X),Y,mlt(int(I),R)):-!,
                    mltr(X,Y,R).
mltr(X,int(I),mlt(int(I),X)):-!.
mltr(pts(X,int(I1)),pts(X,int(I2)),pts(X,int(I3))):-!,
                    I3=I1+I2.
mltr(X,pts(X,int(I)),pts(X,int(I1))):-!,
                    I1=I+1.
mltr(pts(X,int(I)),X,pts(X,int(I1))):-!,
                    I1=I+1.
mltr(X,X,pts(X,int(2))):-!.
mltr(X,Y,mlt(X,Y)).

/************************ Reduction of the divisions ************/

dvdr(int(0),_,int(0)):-!.
dvdr(X,int(1),X):-!.
dvdr(pts(X,int(I1)),pts(X,int(I2)),int(1)):-
                    I1=I2,!.
dvdr(pts(X,int(I1)),pts(X,int(I2)),pts(X,int(I))):-
                    I1>I2,
                    I=I1-I2,!.
dvdr(pts(X,int(I1)),pts(X,int(I2)),dvd(int(1),pts(X,int(I)))):-
                    I1<I2,
                    I=I2-I1,!.
dvdr(X,X,int(1)).
dvdr(pts(X,int(I1)),X,pts(X,int(I))):-
                    I=I1-1,!.
dvdr(X,pts(X,int(I1)),dvd(int(1),pts(X,int(I)))):-
                    I=I1-1,!.
```

```
dvdr(mlt(X,Y),X,Y).
dvdr(mlt(Y,X),X,Y).
dvdr(X,mlt(X,Y),dvd(int(1),Y)).
dvdr(X,mlt(Y,X),dvd(int(1),Y)).
dvdr(mlt(pts(X,int(I)),Y),X,mlt(pts(X,int(I1)),Y)):-
                    I1=I-1,!.
dvdr(mlt(Y,pts(X,int(I))),X,mlt(Y,pts(X,int(I1)))):-
                    I1=I-1,!.
dvdr(X,mlt(pts(X,int(I)),Y),dvd(int(1),mlt(pts(X,int(I1)),Y))):-
                    I1=I-1,!.
dvdr(X,mlt(Y,pts(X,int(I))),dvd(int(1),mlt(Y,pts(X,int(I1))))):-
                    I1=I-1,!.
dvdr(mlt(X,Y),pts(X,int(I)),dvd(Y,pts(X,int(I1)))):-
                    I1=I-1,!.
dvdr(mlt(Y,X),pts(X,int(I)),dvd(pts(X,int(I1)),Y)):-
                    I1=I-1,!.
dvdr(pts(X,int(I)),mlt(X,Y),dvd(pts(X,int(I1)),Y)):-
                    I1=I-1,!.
dvdr(pts(X,int(I)),mlt(Y,X),dvd(pts(X,int(I1)),Y)):-
                    I1=I-1,!.
dvdr(mlt(pts(X,int(I1)),Y),pts(X,int(I2)),mlt(pts(X,int(I)),Y)):-
                    I=I1-I2,!.
dvdr(mlt(Y,pts(X,int(I1))),pts(X,int(I2)),mlt(Y,pts(X,int(I)))):-
                    I=I1-I2,!.
dvdr(pts(X,int(I1)),mlt(pts(X,int(I2)),Y),mlt(pts(X,int(I)),Y)):-
                    I=I1-I2,!.
dvdr(pts(X,int(I1)),mlt(Y,pts(X,int(I2))),mlt(Y,pts(X,int(I)))):-
                    I=I1-I2,!.
dvdr(X,Y,dvd(X,Y)).

/************************ Reduction of the exponents ************/

ptsr(X,int(1),X):-!.
ptsr(_,int(0),int(1)):-!.
ptsr(X,Y,pts(X,Y)).

/************************ WRITING SECTION ***********************/

/************************ Length of an expression ***************/

maxlen(L1,L2,L1):-
                    L1>L2,!.
maxlen(_,L2,L2).

lenpls(L1,L2,0,0,L):-
                    L=L1+L2+1,!.
lenpls(L1,L2,_,_,L):-
                    L=L1+L2+4,!.
len(mlt(pls(K,F),min(M,N)),L):-
                    len(pls(K,F),L1),
                    len(min(M,N),L2),
                    L=L1+L2+4,!.
len(mlt(min(K,F),pls(M,N)),L):-
                    len(min(K,F),L1),
                    len(pls(M,N),L2),
                    L=L1+L2+4,!.
 len(mlt(min(K,F),min(M,N)),L):-
                    len(min(K,F),L1),
                    len(min(M,N),L2),
                    L=L1+L2+4,!.
len(mlt(pls(K,F),pls(M,N)),L):-
                    len(pls(K,F),L1),
                    len(pls(M,N),L2),
                    L=L1+L2+4,!.
len(mlt(dvd(K,F),dvd(M,N)),L):-
```

```
                len(dvd(K,F),L1),
                len(dvd(M,N),L2),
                L=L1+L2+1,!.
len(mlt(A,min(M,N)),L):-
                len(A,L1),
                len(min(M,N),L2),
                L=L1+L2+2,!.
len(mlt(A,pls(M,N)),L):-
                len(A,L1),
                len(pls(M,N),L2),
                L=L1+L2+2,!.
len(mlt(min(M,N),B),L):-
                len(min(M,N),L1),
                len(B,L2),
                L=L1+L2+2,!.
len(mlt(pls(M,N),B),L):-
                len(pls(M,N),L1),
                len(B,L2),
                L=L1+L2+2,!.
len(mlt(A,B),L):-
                len(A,L1),
                len(B,L2),
                L=L1+L2,!.
len(pls(A,min(int(0),B)),L):-
                len(A,L1),
                len(B,L2),
                finddvd(A,F1),
                finddvd(B,F2),
                lenpls(L1,L2,F1,F2,L),!.
len(pls(A,B),L):-
                len(A,L1),
                len(B,L2),
                finddvd(A,F1),
                finddvd(B,F2),
                lenpls(L1,L2,F1,F2,L),!.
len(min(int(0),B),L):-
                len(B,L2),
                L2=1,
                L=L2+1,!.
len(min(int(0),B),L):-
                len(B,L2),
                L=L2+3,!.
len(min(A,min(int(0),B)),L):-
                len(A,L1),
                len(B,L2),
                finddvd(A,F1),
                finddvd(B,F2),
                lenpls(L1,L2,F1,F2,L),!.
len(min(A,min(B,C)),L):-
                len(A,L1),
                len(min(B,C),L2),
                finddvd(A,F1),
                finddvd(min(B,C),F2),
                lenpls(L1,L2,F1,F2,Ls),
                L=Ls+2,!.
len(min(A,pls(B,C)),L):-
                len(A,L1),
                len(pls(B,C),L2),
                finddvd(A,F1),
                finddvd(pls(B,C),F2),
                lenpls(L1,L2,F1,F2,Ls),
                L=Ls+2,!.
len(min(A,B),L):-
                len(A,L1),
                len(B,L2),
```

```
                finddvd(A,F1),
                finddvd(B,F2),
                lenpls(L1,L2,F1,F2,L),!.
len(dvd(A,B),L):-
                len(A,L1),
                len(B,L2),
                maxlen(L1,L2,L),!.
len(pts(A,B),L):-
                len(A,L1),
                len(B,L2),
                L=L1+L2,!.
len(exp(U),L):-
                len(U,L1),
                L=L1+5,!.
len(i(U),L):-
                len(U,L1),
                L=L1+6,!.
len(var(_),1).
len(int(N),2):--9<=N,N<=0,!.
len(int(N),3):--99<=N,N<=-10,!.
len(int(N),4):-999<=N,N<=-100,!.
len(int(N),5):-9999<=N,N<=-1000,!.
len(int(_),6):-10000<=-10000,!.
len(int(N),1):-0<=N,N<=9,!.
len(int(N),2):-10<=N,N<=99,!.
len(int(N),3):-100<=N,N<=999,!.
len(int(N),4):-1000<=N,N<=9999,!.
len(int(N),5):-10000<=N,!.

/************************ Line of the division ******************/

dvdline(L,X,Y):-
                X1=X+L,
                line(X,Y,X1,Y),!.

/************************ Numerator and Denumerator Positions ***/

uxvx(L1,L2,X,Y,X,VX):-L1>=L2,
                      L11=L1*8,
                      L22=L2*8,
                      D=L11-L22,
                      R=round(D/2),
                      VX=X+R,
                      dvdline(L11,X,Y),!.
uxvx(L1,L2,X,Y,UX,X):-L1<L2,
                      L11=L1*8,
                      L22=L2*8,
                      D=L22-L11,
                      R=round(D/2),
                      UX=X+R,
                      dvdline(L22,X,Y),!.

/************************ Find a division ***********************/

finddvd(pls(U,V),F):-
                finddvd(U,F1),
                finddvd(V,F2),
                F=F1+F2,!.
finddvd(min(U,V),F):-
                finddvd(U,F1),
                finddvd(V,F2),
                F=F1+F2,!.
finddvd(mlt(U,V),F):-
                finddvd(U,F1),
                finddvd(V,F2),
```

```
                F=F1+F2,!.
finddvd(pts(U,V),F):-
                finddvd(U,F1),
                finddvd(V,F2),
                F=F1+F2,!.
finddvd(exp(U),F):-
                finddvd(U,F),!.
finddvd(i(U),F):-
                finddvd(U,F),!.
finddvd(dvd(_,_),1):-!.
finddvd(_,0).

/************************ Find an exponent **********************/

findpts(pls(U,V),F):-
                findpts(U,F1),
                findpts(V,F2),
                F=F1+F2,!.
findpts(min(U,V),F):-
                findpts(U,F1),
                findpts(V,F2),
                F=F1+F2,!.
findpts(mlt(U,V),F):-
                findpts(U,F1),
                findpts(V,F2),
                F=F1+F2,!.
findpts(dvd(U,V),F):-
                indpts(U,F1),
                findpts(V,F2),
                F=F1+F2,!.
findpts(exp(U),F):-
                findpts(U,F),!.
findpts(i(U),F):-
                findpts(U,F),!.
findpts(pts(_,_),1):-!.
findpts(_,0).

/************************ Exponent Cases ************************/

ptscases(U,V,X,Y,1,1,0,0):-
                        mwrite(U,X,Y),
                        X1=X+8,
                        Y1=Y-8,
                        mwrite(V,X1,Y1),!.
ptscases(U,V,X,Y,L1,1,0,0):-
                        outtextxy(X,Y,"("),
                        X1=X+8,
                        mwrite(U,X1,Y),
                        X2=X1+L1*8,
                        outtextxy(X2,Y,")"),
                        Y1=Y-8,
                        X3=X2+8,
                        mwrite(V,X3,Y1),!.
ptscases(U,V,X,Y,L1,1,F1,0):-
                        F1>0,
                        Y1=Y-8,
                        Y2=Y+8,
                        outtextxy(X,Y1,"\218"),
                        outtextxy(X,Y,"\179"),
                        outtextxy(X,Y2,"\192"),
                        X1=X+16,
                        mwrite(U,X1,Y),
                        X2=X1+L1*8,
                        outtextxy(X2,Y1,"\191"),
                        outtextxy(X2,Y,"\179"),
```

```
                    outtextxy(X2,Y2,"\217"),
                    X3=X2+8,
                    mwrite(V,X3,Y1),!.
ptscases(U,V,X,Y,1,L2,0,0):-
                    mwrite(U,X,Y),
                    X1=X+8,
                    Y1=Y-8,
                    outtextxy(X1,Y1,"("),
                    X2=X1+8,
                    mwrite(V,X2,Y1),
                    X3=X2+L2*8,
                    outtextxy(X3,Y1,")"),!.
ptscases(U,V,X,Y,L1,L2,0,0):-
                    outtextxy(X,Y,"("),
                    X1=X+8,
                    mwrite(U,X1,Y),
                    X2=X1+L1*8,
                    outtextxy(X2,Y,")"),
                    Y1=Y-8,
                    X3=X2+8,
                    outtextxy(X3,Y1,"("),
                    X4=X3+8,
                    mwrite(V,X4,Y1),
                    X5=X4+L2*8,
                    outtextxy(X5,Y1,")"),!.
ptscases(U,V,X,Y,L1,L2,F1,0):-
                    F1>0,
                    Y1=Y-8,
                    Y2=Y+8,
                    outtextxy(X,Y1,"\218"),
                    outtextxy(X,Y,"\179"),
                    outtextxy(X,Y2,"\192"),
                    X1=X+8,
                    mwrite(U,X1,Y),
                    X2=X1+L1*8,
                    outtextxy(X2,Y1,"\191"),
                    outtextxy(X2,Y,"\179"),
                    outtextxy(X2,Y2,"\217"),
                    X3=X2+8,
                    outtextxy(X3,Y1,"("),
                    X4=X3+8,
                    mwrite(V,X4,Y1),
                    X5=X4+L2*8,
                    outtextxy(X5,Y1,")"),!.
ptscases(U,V,X,Y,1,L2,0,F2):-
                    F2>0,
                    mwrite(U,X,Y),
                    X1=X+8,
                    Y1=Y-16,
                    Y2=Y1-8,
                    Y3=Y1+8,
                    outtextxy(X1,Y2,"\218"),
                    outtextxy(X1,Y1,"\179"),
                    outtextxy(X1,Y3,"\192"),
                    X2=X1+8,
                    mwrite(V,X2,Y1),
                    X3=X2+L2*8,
                    outtextxy(X3,Y2,"\191"),
                    outtextxy(X3,Y1,"\179"),
                    outtextxy(X3,Y3,"\217"),!.
ptscases(U,V,X,Y,L1,L2,0,F2):-
                    F2>0,
                    outtextxy(X,Y,"("),
                    X1=X+8,
                    mwrite(U,X1,Y),
```

```
                        X2=X1+L1*8,
                        outtextxy(X2,Y,")"),
                        Y1=Y-16,
                        X3=X2+8,
                        Y2=Y1-8,
                        Y3=Y1+8,
                        outtextxy(X3,Y2,"\218"),
                        outtextxy(X3,Y1,"\179"),
                        outtextxy(X3,Y3,"\192"),
                        X4=X3+8,
                        mwrite(V,X4,Y1),
                        X5=X4+L2*8,
                        outtextxy(X5,Y2,"\191"),
                        outtextxy(X5,Y1,"\179"),
                        outtextxy(X5,Y3,"\217"),!.
ptscases(U,V,X,Y,L1,L2,_,_):-
                        Y1=Y-8,
                        Y2=Y+8,
                        outtextxy(X,Y1,"\218"),
                        outtextxy(X,Y,"\179"),
                        outtextxy(X,Y2,"\192"),
                        X1=X+8,
                        mwrite(U,X1,Y),
                        X2=X1+L1*8,
                        outtextxy(X2,Y1,"\191"),
                        outtextxy(X2,Y,"\179"),
                        outtextxy(X2,Y2,"\217"),
                        X3=X2+8,
                        Y3=Y-16,
                        Y4=Y3-8,
                        Y5=Y3+8,
                        outtextxy(X3,Y4,"\218"),
                        outtextxy(X3,Y3,"\179"),
                        outtextxy(X3,Y5,"\192"),
                        X4=X3+8,
                        mwrite(V,X4,Y3),
                        X5=X4+L2*8,
                        outtextxy(X5,Y4,"\191"),
                        outtextxy(X5,Y3,"\179"),
                        outtextxy(X5,Y5,"\217"),!.

/************************ Addition and Subtraction Cases ********/

plmncases(int(0),V,X,Y,_,_,_,Op):-
                                len(V,L2),
                                L2=1,
                                outtextxy(X,Y,Op),
                                X1=X+8,
                                mwrite(V,X1,Y),!.
plmncases(int(0),V,X,Y,_,_,_,Op):-
                                outtextxy(X,Y,Op),
                                X1=X+8,
                                outtextxy(X1,Y,"("),
                                X2=X1+8,
                                mwrite(V,X2,Y),
                                len(V,L2),
                                X3=X2+L2*8,
                                outtextxy(X3,Y,")"),!.
plmncases(U,min(int(0),V),X,Y,L1,_,_,"+"):-
                                mwrite(U,X,Y),
                                X1=X+8*L1,
                                outtextxy(X1,Y,"-"),
                                X2=X1+8,
                                mwrite(V,X2,Y),!.
plmncases(U,min(int(0),V),X,Y,L1,_,_,"-"):-
```

```
                                    mwrite(U,X,Y),
                                    X1=X+8*L1,
                                    outtextxy(X1,Y,"+"),
                                    X2=X1+8,
                                    mwrite(V,X2,Y),!.
plmncases(U,min(V,T),X,Y,L1,_,0,"-"):-
                                    mwrite(U,X,Y),
                                    X1=X+8*L1,
                                    outtextxy(X1,Y,"-"),
                                    X2=X1+8,
                                    outtextxy(X2,Y,"("),
                                    X3=X2+8,
                                    mwrite(min(V,T),X3,Y),
                                    len(min(V,T),L2),
                                    X4=X3+8*L2,
                                    outtextxy(X4,Y,")"),!.
plmncases(U,pls(V,T),X,Y,L1,_,0,"-"):-
                                    mwrite(U,X,Y),
                                    X1=X+8*L1,
                                    outtextxy(X1,Y,"-"),
                                    X2=X1+8,
                                    outtextxy(X2,Y,"("),
                                    X3=X2+8,
                                    mwrite(pls(V,T),X3,Y),
                                    len(pls(V,T),L2),
                                    X4=X3+8*L2,
                                    outtextxy(X4,Y,")"),!.
plmncases(U,min(V,T),X,Y,L1,_,F2,"-"):-
                                    F2>0,
                                    mwrite(U,X,Y),/*2.5*/
                                    X1=X+8*L1,
                                    outtextxy(X1,Y,"-"),
                                    X2=X1+8,
                                    Y1=Y-8,
                                    Y2=Y+8,
                                    outtextxy(X2,Y1,"\218"),
                                    outtextxy(X2,Y,"\179"),
                                    outtextxy(X2,Y2,"\192"),
                                    X3=X2+8,
                                    mwrite(min(V,T),X3,Y),
                                    len(min(V,T),L2),
                                    X4=X3+8*L2,
                                    outtextxy(X4,Y1,"\191"),
                                    outtextxy(X4,Y,"\179"),
                                    outtextxy(X4,Y2,"\217"),!.
plmncases(U,pls(V,T),X,Y,L1,_,F2,"-"):-
                                    F2>0,
                                    mwrite(U,X,Y),
                                    X1=X+8*L1,
                                    outtextxy(X1,Y,"-"),
                                    X2=X1+8,
                                    Y1=Y-8,
                                    Y2=Y+8,
                                    outtextxy(X2,Y1,"\218"),
                                    outtextxy(X2,Y,"\179"),
                                    outtextxy(X2,Y2,"\192"),
                                    X3=X2+8,
                                    mwrite(pls(V,T),X3,Y),
                                    len(pls(V,T),L2),
                                    X4=X3+8*L2,
                                    outtextxy(X4,Y1,"\191"),
                                    outtextxy(X4,Y,"\179"),
                                    outtextxy(X4,Y2,"\217"),!.
plmncases(U,V,X,Y,L1,0,0,Op):-
                                    mwrite(U,X,Y),
```

```
                                X1=X+8*L1,
                                outtextxy(X1,Y,Op),
                                X2=X1+8,
                                mwrite(V,X2,Y),!.
plmncases(U,V,X,Y,L1,_,_,Op):-
                                mwrite(U,X,Y),
                                X1=X+8*L1+8,
                                outtextxy(X1,Y,Op),
                                X2=X1+16,
                                mwrite(V,X2,Y),!.

/************************ Multiplication Cases ******************/

mltcases(U,V,X,Y,1,1,_,_):-
                                mwrite(U,X,Y),
                                X1=X+8,
                                mwrite(V,X1,Y),!.
mltcases(pls(A,B),min(C,D),X,Y,L1,L2,0,0):-
                                outtextxy(X,Y,"("),
                                X1=X+8,
                                mwrite(pls(A,B),X1,Y),
                                X2=X1+L1*8,
                                outtextxy(X2,Y,")"),
                                X3=X2+8,
                                outtextxy(X3,Y,"("),
                                X4=X3+8,
                                mwrite(min(C,D),X4,Y),
                                X5=X4+L2*8,
                                outtextxy(X5,Y,")"),!.
mltcases(min(A,B),pls(C,D),X,Y,L1,L2,0,0):-
                                outtextxy(X,Y,"("),
                                X1=X+8,
                                mwrite(min(A,B),X1,Y),
                                X2=X1+L1*8,
                                outtextxy(X2,Y,")"),
                                X3=X2+8,
                                outtextxy(X3,Y,"("),
                                X4=X3+8,
                                mwrite(pls(C,D),X4,Y),
                                X5=X4+L2*8,
                                outtextxy(X5,Y,")"),!.
mltcases(pls(A,B),pls(C,D),X,Y,L1,L2,0,0):-
                                outtextxy(X,Y,"("),
                                X1=X+8,
                                mwrite(pls(A,B),X1,Y),
                                X2=X1+L1*8,
                                outtextxy(X2,Y,")"),
                                X3=X2+8,
                                outtextxy(X3,Y,"("),
                                X4=X3+8,
                                mwrite(pls(C,D),X4,Y),
                                X5=X4+L2*8,
                                outtextxy(X5,Y,")"),!.
mltcases(min(A,B),min(C,D),X,Y,L1,L2,0,0):-
                                outtextxy(X,Y,"("),
                                X1=X+8,
                                mwrite(min(A,B),X1,Y),
                                X2=X1+L1*8,
                                outtextxy(X2,Y,")"),
                                X3=X2+8,
                                outtextxy(X3,Y,"("),
                                X4=X3+8,
                                mwrite(min(C,D),X4,Y),
                                X5=X4+L2*8,
                                outtextxy(X5,Y,")"),!.
```

```
mltcases(pls(A,B),min(C,D),X,Y,L1,L2,F1,0):-
                                        F1>0,
                                        Y1=Y-8,
                                        Y2=Y+8,
                                        outtextxy(X,Y1,"\218"),
                                        outtextxy(X,Y,"\179"),
                                        outtextxy(X,Y2,"\192"),
                                        X1=X+8,
                                        mwrite(pls(A,B),X1,Y),
                                        X2=X1+L1*8,
                                        outtextxy(X2,Y1,"\191"),
                                        outtextxy(X2,Y,"\179"),
                                        outtextxy(X2,Y2,"\217"),
                                        X3=X2+8,
                                        outtextxy(X3,Y,"("),
                                        X4=X3+8,
                                        mwrite(min(C,D),X4,Y),
                                        X5=X4+L2*8,
                                        outtextxy(X5,Y,")"),!.
mltcases(pls(A,B),pls(C,D),X,Y,L1,L2,F1,0):-
                                        F1>0,
                                        Y1=Y-8,
                                        Y2=Y+8,
                                        outtextxy(X,Y1,"\218"),
                                        outtextxy(X,Y,"\179"),
                                        X1=X+8,
                                        mwrite(pls(A,B),X1,Y),
                                        X2=X1+L1*8,
                                        outtextxy(X2,Y1,"\191"),
                                        outtextxy(X2,Y,"\179"),
                                        outtextxy(X2,Y2,"\217"),
                                        X3=X2+8,
                                        outtextxy(X3,Y,"("),
                                        X4=X3+8,
                                        mwrite(pls(C,D),X4,Y),
                                        X5=X4+L2*8,
                                        outtextxy(X5,Y,")"),!.
mltcases(min(A,B),min(C,D),X,Y,L1,L2,F1,0):-
                                        F1>0,
                                        Y1=Y-8,
                                        Y2=Y+8,
                                        outtextxy(X,Y1,"\218"),
                                        outtextxy(X,Y,"\179"),
                                        outtextxy(X,Y2,"\192"),
                                        X1=X+8,
                                        mwrite(min(A,B),X1,Y),
                                        X2=X1+L1*8,
                                        outtextxy(X2,Y1,"\191"),
                                        outtextxy(X2,Y,"\179"),
                                        outtextxy(X2,Y2,"\217"),
                                        X3=X2+8,
                                        outtextxy(X3,Y,"("),
                                        X4=X3+8,
                                        mwrite(min(C,D),X4,Y),
                                        X5=X4+L2*8,
                                        outtextxy(X5,Y,")"),!.
mltcases(min(A,B),pls(C,D),X,Y,L1,L2,F1,0):-
                                        F1>0,
                                        Y1=Y-8,
                                        Y2=Y+8,
                                        outtextxy(X,Y1,"\218"),
                                        outtextxy(X,Y,"\179"),
                                        outtextxy(X,Y2,"\192"),
                                        X1=X+8,
                                        mwrite(min(A,B),X1,Y),
```

```
                                X2=X1+L1*8,
                                outtextxy(X2,Y1,"\191"),
                                outtextxy(X2,Y,"\179"),
                                outtextxy(X2,Y2,"\217"),
                                X3=X2+8,
                                outtextxy(X3,Y,"("),
                                X4=X3+8,
                                mwrite(pls(C,D),X4,Y),
                                X5=X4+L2*8,
                                outtextxy(X5,Y,")"),!.
mltcases(min(A,B),pls(C,D),X,Y,L1,L2,0,F2):-
                                F2>0,
                                outtextxy(X,Y,"("),
                                X1=X+8,
                                mwrite(min(A,B),X1,Y),
                                X2=X1+L1*8,
                                outtextxy(X2,Y,")"),
                                X3=X2+8,
                                Y1=Y-8,
                                Y2=Y+8,
                                outtextxy(X3,Y1,"\218"),
                                outtextxy(X3,Y,"\179"),
                                outtextxy(X3,Y2,"\192"),
                                X4=X3+8,
                                mwrite(pls(C,D),X4,Y),
                                X5=X4+L2*8,
                                outtextxy(X5,Y1,"\191"),
                                outtextxy(X5,Y,"\179"),
                                outtextxy(X5,Y2,"\217"),!.
mltcases(pls(A,B),pls(C,D),X,Y,L1,L2,0,F2):-
                                F2>0,
                                outtextxy(X,Y,"("),
                                X1=X+8,
                                mwrite(pls(A,B),X1,Y),
                                X2=X1+L1*8,
                                outtextxy(X2,Y,")"),
                                X3=X2+8,
                                Y1=Y-8,
                                Y2=Y+8,
                                outtextxy(X3,Y1,"\218"),
                                outtextxy(X3,Y,"\179"),
                                outtextxy(X3,Y2,"\192"),
                                X4=X3+8,
                                mwrite(pls(C,D),X4,Y),
                                X5=X4+L2*8,
                                outtextxy(X5,Y1,"\191"),
                                outtextxy(X5,Y,"\179"),
                                outtextxy(X5,Y2,"\217"),!.
mltcases(min(A,B),min(C,D),X,Y,L1,L2,0,F2):-
                                F2>0,
                                outtextxy(X,Y,"("),
                                X1=X+8,
                                mwrite(min(A,B),X1,Y),
                                X2=X1+L1*8,
                                outtextxy(X2,Y,")"),
                                X3=X2+8,
                                Y1=Y-8,
                                Y2=Y+8,
                                outtextxy(X3,Y1,"\218"),
                                outtextxy(X3,Y,"\179"),
                                outtextxy(X3,Y2,"\192"),
                                X4=X3+8,
                                mwrite(min(C,D),X4,Y),
                                X5=X4+L2*8,
                                outtextxy(X5,Y1,"\191"),
```

```
                                        outtextxy(X5,Y,"\179"),
                                        outtextxy(X5,Y2,"\217"),!.
mltcases(pls(A,B),min(C,D),X,Y,L1,L2,0,F2):-
                                        F2>0,
                                        outtextxy(X,Y,"("),
                                        X1=X+8,
                                        mwrite(pls(A,B),X1,Y),
                                        X2=X1+L1*8,
                                        outtextxy(X2,Y,")"),
                                        X3=X2+8,
                                        Y1=Y-8,
                                        Y2=Y+8,
                                        outtextxy(X3,Y1,"\218"),
                                        outtextxy(X3,Y,"\179"),
                                        outtextxy(X3,Y2,"\192"),
                                        X4=X3+8,
                                        mwrite(min(C,D),X4,Y),
                                        X5=X4+L2*8,
                                        outtextxy(X5,Y1,"\191"),
                                        outtextxy(X5,Y,"\179"),
                                        outtextxy(X5,Y2,"\217"),!.
mltcases(U,pls(A,B),X,Y,L1,L2,_,0):-
                                        mwrite(U,X,Y),
                                        X1=X+L1*8,
                                        outtextxy(X1,Y,"("),
                                        X2=X1+8,
                                        mwrite(pls(A,B),X2,Y),
                                        X3=X2+L2*8,
                                        outtextxy(X3,Y,")"),!.
mltcases(U,min(A,B),X,Y,L1,L2,_,0):-
                                        mwrite(U,X,Y),
                                        X1=X+L1*8,
                                        outtextxy(X1,Y,"("),
                                        X2=X1+8,
                                        mwrite(min(A,B),X2,Y),
                                        X3=X2+L2*8,
                                        outtextxy(X3,Y,")"),!.
mltcases(pls(A,B),V,X,Y,L1,_,0,_):-
                                        outtextxy(X,Y,"("),
                                        X1=X+8,
                                        mwrite(pls(A,B),X1,Y),
                                        X2=X1+L1*8,
                                        outtextxy(X2,Y,")"),
                                        X3=X2+8,
                                        mwrite(V,X3,Y),!.
mltcases(min(A,B),V,X,Y,L1,_,0,_):-
                                        outtextxy(X,Y,"("),
                                        X1=X+8,
                                        mwrite(min(A,B),X1,Y),
                                        X2=X1+L1*8,
                                        outtextxy(X2,Y,")"),
                                        X3=X2+8,
                                        mwrite(V,X3,Y),!.
mltcases(pls(A,B),V,X,Y,L1,_,F1,_):-
                                        F1>0,
                                        Y1=Y-8,
                                        Y2=Y+8,
                                        outtextxy(X,Y1,"\218"),
                                        outtextxy(X,Y,"\179"),
                                        outtextxy(X,Y2,"\192"),
                                        X1=X+8,
                                        mwrite(pls(A,B),X1,Y),
                                        X2=X1+L1*8,
                                        outtextxy(X2,Y1,"\191"),
                                        outtextxy(X2,Y,"\179"),
```

```
                                outtextxy(X2,Y2,"\217"),
                                X3=X2+8,
                                mwrite(V,X3,Y),!.
mltcases(min(A,B),V,X,Y,L1,_,F1,_):-
                                F1>0,
                                Y1=Y-8,
                                Y2=Y+8,
                                outtextxy(X,Y1,"\218"),
                                outtextxy(X,Y,"\179"),
                                outtextxy(X,Y2,"\192"),
                                X1=X+8,
                                mwrite(min(A,B),X1,Y),
                                X2=X1+L1*8,
                                outtextxy(X2,Y1,"\191"),
                                outtextxy(X2,Y,"\179"),
                                outtextxy(X2,Y2,"\217"),
                                X3=X2+8,
                                mwrite(V,X3,Y),!.
mltcases(U,pls(C,D),X,Y,L1,L2,_,F2):-
                                F2>0,
                                mwrite(U,X,Y),
                                X1=X+L1*8,
                                Y1=Y-8,
                                Y2=Y+8,
                                outtextxy(X1,Y1,"\218"),
                                outtextxy(X1,Y,"\179"),
                                outtextxy(X1,Y2,"\192"),
                                X2=X1+8,
                                mwrite(pls(C,D),X2,Y),
                                X3=X2+L2*8,
                                outtextxy(X3,Y1,"\191"),
                                outtextxy(X3,Y,"\179"),
                                outtextxy(X3,Y2,"\217"),!.
mltcases(U,min(C,D),X,Y,L1,L2,_,F2):-
                                F2>0,
                                mwrite(U,X,Y),
                                X1=X+L1*8,
                                Y1=Y-8,
                                Y2=Y+8,
                                outtextxy(X1,Y1,"\218"),
                                outtextxy(X1,Y,"\179"),
                                outtextxy(X1,Y2,"\192"),
                                X2=X1+8,
                                mwrite(min(C,D),X2,Y),
                                X3=X2+L2*8,
                                outtextxy(X3,Y1,"\191"),
                                outtextxy(X3,Y,"\179"),
                                outtextxy(X3,Y2,"\217"),!.
mltcases(dvd(A,B),dvd(C,D),X,Y,L1,_,_,_):-
                                mwrite(dvd(A,B),X,Y),
                                X1=X+L1*8+8,
                                mwrite(dvd(C,D),X1,Y),!.
mltcases(U,V,X,Y,L1,_,_,_):-
                                mwrite(U,X,Y),
                                X1=X+L1*8,
                                mwrite(V,X1,Y),!.

/************************ Division Cases ************************/

dvdcases(U,V,X,Y,L1,L2,F1,0,_,0):-
                                F1>0,
                                NY=Y+4,
                                uxvx(L1,L2,X,NY,UX,VX),
                                Y1=Y-16,
                                mwrite(U,UX,Y1),
```

```
                                    Y2=Y+8,
                                    mwrite(V,VX,Y2),!.
dvdcases(U,V,X,Y,L1,L2,0,F2,_,_):-
                                    F2>0,
                                    NY=Y+4,
                                    uxvx(L1,L2,X,NY,UX,VX),
                                    Y1=Y-8,
                                    mwrite(U,UX,Y1),
                                    Y2=Y+24,
                                    mwrite(V,VX,Y2),!.
dvdcases(U,V,X,Y,L1,L2,F1,F2,_,0):-
                                    F1>0,F2>0,
                                    NY=Y+4,
                                    uxvx(L1,L2,X,NY,UX,VX),
                                    Y1=Y-16,
                                    mwrite(U,UX,Y1),
                                    Y2=Y+16,
                                    mwrite(V,VX,Y2),!.
dvdcases(U,V,X,Y,L1,L2,F1,F2,_,_):-
                                    F1>0,F2>0,
                                    NY=Y+4,
                                    uxvx(L1,L2,X,NY,UX,VX),
                                    Y1=Y-16,
                                    mwrite(U,UX,Y1),
                                    Y2=Y+24,
                                    mwrite(V,VX,Y2),!.
 dvdcases(U,V,X,Y,L1,L2,0,0,_,P2):-
                                    P2>0,
                                    NY=Y+4,
                                    uxvx(L1,L2,X,NY,UX,VX),
                                    Y1=Y-8,
                                    mwrite(U,UX,Y1),
                                    Y2=Y+16,
                                    mwrite(V,VX,Y2),!.
dvdcases(U,V,X,Y,L1,L2,F1,0,_,P2):-
                                    F1>0,P2>0,
                                    NY=Y+4,
                                    uxvx(L1,L2,X,NY,UX,VX),
                                    Y1=Y-16,
                                    mwrite(U,UX,Y1),
                                    Y2=Y+16,
                                    mwrite(V,VX,Y2),!.
dvdcases(U,V,X,Y,L1,L2,_,_,_,_):-
                                    NY=Y+4,
                                    uxvx(L1,L2,X,NY,UX,VX),/*6*/
                                    Y1=Y-8,
                                    mwrite(U,UX,Y1),
                                    Y2=Y+8,
                                    mwrite(V,VX,Y2),!.

/************************* Special Division Cases ****************/

specdvd(U,pts(A,B),X,Y,_,F2,0,0):-
                                    F2>0,
                                    len(U,L1),
                                    len(pts(A,B),L2),
                                    NY=Y+4,
                                    uxvx(L1,L2,X,NY,UX,VX),
                                    Y1=Y-8,
                                    mwrite(U,UX,Y1),
                                    Y2=Y+32,
                                    mwrite(pts(A,B),VX,Y2),!.
specdvd(U,pts(A,B),X,Y,_,_,0,P2):-
                                    P2>0,
                                    len(U,L1),
```

```
                                len(pts(A,B),L2),
                                NY=Y+4,
                                uxvx(L1,L2,X,NY,UX,VX),
                                Y1=Y-8,
                                mwrite(U,UX,Y1),
                                Y2=Y+24,
                                mwrite(pts(A,B),VX,Y2),!.
specdvd(U,dvd(A,B),X,Y,_,_,P1,0):-
                                P1>0,
                                len(U,L1),
                                len(dvd(A,B),L2),
                                NY=Y+4,
                                uxvx(L1,L2,X,NY,UX,VX),
                                Y1=Y-8,
                                mwrite(U,UX,Y1),
                                Y2=Y+32,
                                mwrite(dvd(A,B),VX,Y2),!.
specdvd(U,dvd(A,B),X,Y,_,_,P1,P2):-
                                P1>0,P2>0,
                                len(U,L1),
                                len(dvd(A,B),L2),
                                NY=Y+4,
                                uxvx(L1,L2,X,NY,UX,VX),
                                Y1=Y-8,
                                mwrite(U,UX,Y1),
                                Y2=Y+32,
                                mwrite(dvd(A,B),VX,Y2),!.

/************************ Functional Cases ********************/

fcases(U,X,Y,L,F):-
                        F>0,
                        Y1=Y-8,
                        Y2=Y+8,
                        outtextxy(X,Y1,"\218"),
                        outtextxy(X,Y,"\179"),
                        outtextxy(X,Y2,"\192"),
                        X1=X+8,
                        mwrite(U,X1,Y),
                        X2=X1+L*8,
                        outtextxy(X2,Y1,"\191"),
                        outtextxy(X2,Y,"\179"),
                        outtextxy(X2,Y2,"\217"),!.
fcases(U,X,Y,L,_):-
                        outtextxy(X,Y,"("),
                        X1=X+8,
                        mwrite(U,X1,Y),
                        X2=X1+L*8,
                        outtextxy(X2,Y,")"),!.

/************************ Writing *********************************/

mwrite(pls(U,V),X,Y):-
                        finddvd(U,F1),
                        finddvd(V,F2),
                        len(U,L1),
                        plmncases(U,V,X,Y,L1,F1,F2,"+"),!.
mwrite(min(U,V),X,Y):-
                        finddvd(U,F1),
                        finddvd(V,F2),
                        len(U,L1),
                        plmncases(U,V,X,Y,L1,F1,F2,"-"),!.
mwrite(mlt(U,V),X,Y):-
                        len(U,L1),
                        len(V,L2),
```

```
                        finddvd(U,F1),
                        finddvd(V,F2),
                        mltcases(U,V,X,Y,L1,L2,F1,F2),!.
mwrite(dvd(U,pts(A,B)),X,Y):-
                        finddvd(A,F1),
                        finddvd(B,F2),
                        findpts(A,P1),
                        findpts(B,P2),
                        specdvd(U,pts(A,B),X,Y,F1,F2,P1,P2),!.
mwrite(dvd(U,dvd(A,B)),X,Y):-
                        finddvd(A,F1),
                        finddvd(B,F2),
                        findpts(A,P1),
                        findpts(B,P2),
                        specdvd(U,dvd(A,B),X,Y,F1,F2,P1,P2),!.
mwrite(dvd(U,V),X,Y):-
                        len(U,L1),
                        len(V,L2),
                        finddvd(U,F1),
                        finddvd(V,F2),
                        findpts(U,P1),
                        findpts(V,P2),
                        dvdcases(U,V,X,Y,L1,L2,F1,F2,P1,P2),!.
mwrite(pts(U,V),X,Y):-
                        len(U,L1),
                        len(V,L2),
                        finddvd(U,F1),
                        finddvd(V,F2),
                        ptscases(U,V,X,Y,L1,L2,F1,F2),!.
mwrite(exp(U),X,Y):-
                        len(U,L),
                        finddvd(U,F),
                        outtextxy(X,Y,"Exp"),
                        X1=X+24,
                        fcases(U,X1,Y,L,F),!.
mwrite(i(U),X,Y):-
                        len(U,L),
                        finddvd(U,F),
                        Y1=Y-8,
                        Y2=Y+8,
                        outtextxy(X,Y1,"\244"),
                        outtextxy(X,Y,"\179"),
                        outtextxy(X,Y2,"\245"),
                        X1=X+8,
                        X2=X+L*8+24,
                        outtextxy(X2,Y,"dx"),
                        fcases(U,X1,Y,L,F),!.
mwrite(var(V),X,Y):-
                        outtextxy(X,Y,V),!.
mwrite(int(N),X,Y):-
                        str_int(SN,N),
                        outtextxy(X,Y,SN),!.

GOAL

   start.
```

ÖZGEÇMİŞ

Adı Soyadı : _____

Doğum Yeri : _____

Doğum Yılı : _____

Medeni Hali : _____

Eğitim ve Akademik Durumu :

Lise 19../19.. _____

Lisans 19../19.. _____

Yabancı Dil : _____

İş Tecrübesi :

19../19.. _____

19../19.. _____

TÜRKÇE ABSTRAKT (En fazla 250 sözcük)
(TÜBİTAK/TÜRDOK'un Abstrakt Hazırlama Klavuzunu kullanınız.)

Birinci mertebeden doğrusal diferensiyel denklemler, birinci dereceden mantık yapısı içinde incelenerek, sembolik çözümleri dedaktif yaklaşımla araştırılmıştır. Dedaktif yaklaşımda, yönlü graf tekniğine dayalı olarak oluşturulan clause kümesinden, resolution ilkesi ile elde edilen sonuçların doğruluğu, geliştirilen önteorem ve teoremlerle ispatlanmıştır.

İNGİLİZCE ABSTRAKT (En fazla 250 sözcük) :

First order linear differential equations have been studied within the context of first order logic and a deductive approach to their symbolic solutions has been investigated. The correctness of the deductive approach, which obtains solutions by applying the resolution principle to a clause set obtained using directed graph techniques, has been established in a Lemma and Theorem.