T.C.
SÜLEYMAN DEMİREL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

λ-ALGORİTMASINDA KONGRUANSLARIN TÜRETİMİ

AYTEKİN ERYILMAZ

YÜKSEK LİSANS TEZİ
MATEMATİK ANABİLİM DALI

Bu tez .... 199.. tarihinde aşağıdaki jüri tarafından 100 tam puan üzerinden .... puan not takdir edilerek kabul edilmiştir.

Prof. Dr. Ali GÖRGÜLÜ	Doç. Dr. Adil KILIÇ	Yrd. Doç. Dr. LÜTFULLAH ALBAYRAK (Danışman)

ÖZET

λ-Algoritması yazılım yapısı, türetim teknikleri bakımından Church [13], Kleene[15], Barendregt[10], Krivine[16], Levy[17], Hindley and Seldin [14], Ünlü[21], Mirasyedioğlu[18,19], Albayrak[2,3,4,5,6] vb. tarafından çalışıldı. Albayrak[2] λ-Algoritmasında Cebirsel Yapılar üzerinde çalıştı.

Bu tezde, (i) EBK ve λ-terimleri aracılığında önceden verilen λ-kültürü, β-redeksinin indirgemesi kavramlarına dayalı olarak, λ-kongruansının tanımı verildi. (ii) Albayrak [2, 3, 4, 5, 6] 'da verilen T_i $(i = 1, 2,..., 17)$ operatörlerine dayalı T_{18} operatörü türetildi. (iii) λ-Algoritması kuralları içinde T_i $(i=1,2,...,18)$ operatörleri ile λ-kongruansları türetildi. (iv) Lineer λ-kongruans-larının tanımı ve türetimi yapıldı.

Sonuçta, geliştirilen bilgilerin uygulamada kullanılmasına ilişkin bazı öneriler verildi.

ABSTRACT

The software structure of λ-Algorithm has been studied from the point of derivational technigues by Church[13], Kleene[15], Barendregt [10], Krivine[16], Levy[17], Hindley and Seldin[14], Ünlü[21], Mirasyedioğlu[18, 19], Albayrak[2, 3, 4, 5, 6], etc. Albayrak[2], while studying λ-Algorithm, expanded upon algebraic structures.

In this thesis, (i) the definition of λ-congruence was given and on the basis of λ-culture, the reduction of β-redex which were given by EBK and λ-terms. (ii) T_{18} operator was derivated on the basis of T_i ($i = 1, 2, \ldots, 17$) operators given in Albayrak [2,3, 4,5,6]. (iii) λ- congruences were derivated by T_i ($i = 1,2,\ldots,18$) operators in the rules of λ-Algorithm. (iv) The definition and derivation of Lineer λ- congruences were made.

In conclusion, some suggestions have been made regarding the application of developed knowledge.

ÖNSÖZ

Bu tez çalışması, SDÜ Fen Bilimleri Enstitüsü Matematik Anabilimi Uygulamalı Matematik Dalında sayın hocam Yrd. Doç. Dr. Lütfullah ALBAYRAK yönetiminde yapılmıştır.

Yüksek lisans programına kabulümden itibaren tezimin sonuçlanmasına kadar her zaman bana bilimsel destek sağlayan ve yardımlarını esirgemeyen sayın tez yöneticime teşekkürü borç bilirim. Ayrıca emeği geçen S.D.Ü. Fen-Edebiyat Fakültesi Matematik Bölümü öğretim elemanlarına teşekkür ederim.

Temmuz-1996 Aytekin ERYILMAZ

İÇİNDEKİLER

1. GİRİŞ

Geçen elli yıl içinde gelişmiş ve gelişmekte olan toplumlar, bilgisayar desteğinde bilgi işleme becerisini gösterebildikleri için baş döndürücü gelişmeler kaydetmişlerdir. Bu gelişmeleri değişik yönleri ile incelemek için bilgisayar bilimleri adı altında, alt yapı olarak değişik bilim dallarına dayalı, yeni bir bilim dalı doğmuş ve çoğu üniversitelerin öğretim programlarına konulmuştur (Albayrak [2]).

Bilgisayar bilimlerinin gelişmesiyle birlikte Lambda Calculus'e de büyük ilgi vardır. Lambda Calculus ile bilgisayar bilimleri arasındaki alaka çok açıktır. Örneğin, LISP programlama dilinin planlanması λ-algoritmasından etkilenmiştir (Revesz [20]). Ayrıca Auto Cad dilindeki formüllerin kodlanması λ-Algoritmasındaki gibidir (Albayrak[4]).

Lambda Calculus 1930'larda mantık bilimcisi Alanzo Church tarafından teorik temelleri kurulan, bir fonksiyon notasyonuna bağlı, çeşitli sistemlerin bir kümesidir (Hindley and Seldin[14]).

Bilgisayar Bilimlerinde Lambda Calculus'un açık ve sistematik kullanımı Peter Landin, Christopher Strachey, ve Lambda Calculus üzerine kurulmuş proğramlama dillerinin teorik anlatımını geliştiren diğer bilim adamları tarafından başlatılmıştır (Revesz[20]).

Lambda Calculus'un amacı fonksiyonların en genel özelliklerini incelemektir. Bu, matematiğin çeşitli bölümlerinde kullanılan fonksiyon çeşitleri ile bütünleştirilmiş bir fonksiyonlar teorisi geliştirilmek istendiği anlamına gelir. Bunu yapma yollarından biri, temel olarak küme teorisini kullanmak ve bunun üzerine bir fonksiyon fikrini inşa etmektir. Bu fikre göre fonksiyon, sıralı ikililerin (genellikle sonsuz) kümesidir.

Lambda Calculus'de fonksiyonlar sıralı ikililerin kümesiyle değil, lambda bağıntıları denilen sembolik notasyonlarla gösterilir ve fonksiyonların bu sembolik gösterimini kullanarak fonksiyonlar teorisi kurulabilir.

Lambda bağıntıları üzerinde bazı işlemler tanımlanarak ve bu işlemler altında bunların özellikleri incelenecektir. Lambda Calculus, programlama dillerinin matematiksel özelliklerini incelemek için en önemli araçlardan birisidir. Bu matematiksel mantık dilini kullanarak benzer amaçlar için geliştirilmiş fakat bilgisayar bilimlerindeki en son gelişmelerle daha da geliştirilmiştir. (Revesz [20]).

Bu tezin amacı;

i) EBK ve λ-bilgisi ile λ-kültürünü tanımlamak,

ii) Önceden bilinen T_i ($i=1, \ldots, 17$) operatörlerini λ-kongruansların türetiminde ve indirgenmesinde kullanmak,

iii) T_{18} operatörünü türetmek,

iv) Bir bilinmeyenli lineer λ-kongruanslarını tanımlamak ve türetmek,

v)Geliştirilen bilgilerin uygulamada kullanımına ilişkin bazı önerilerde bulunmaktadır.

Tezin anlatımı üç bölüm içinde, değişik tanımlara dayalı 7 yardımcı teorem, ·17 teorem ile verilerek, bunlardan 16 tanesinin ispatı yapılmıştır.

Bu çalışmanın birinci bölümü giriş, ikinci bölümü konu ile ilgili ön bilgileri kapsamaktadır. Üçüncü bölüm tezin özü olan λ-kongruansları ve bir bilinmeyenli lineer λ-kongruansların tanım ve teoremini içermektedir. Üçüncü bölüm üç alt bölümden oluşmuştur. Bunlardan birinci alt bölümde, kongruanslar için temel tanımlar ve ön iki teoremin ispatını, ikinci alt bölümde λ-kongruansların türetimi ile ilgili tanımlar ve teoremler, üçüncü alt bölümde de bir değişkenli λ-kongruanslarının tanımı ve teoremi verilmiştir. Dördüncü bölüm sonuç ve önerileri kapsamaktadır. Bu bölümde geliştirilen bilgilerin uygulamada kullanımına ilişkin bazı öneriler sunulmuştur.

Ayrıca kaynaklardan baz olarak alınan kavramlara kolayca erişimi sağlamak için sonda ek-i (i=1,2) diye adlandırılan iki ek konulmuştur. Ek-1, λ-operatörleri; Ek-2 de gösterim ve notasyonlar sunulmuştur.

Kaynaklar listesi yirmibir girdiden oluşmuştur. Tezin amacı için gerekli olan bilgiler bu kaynaklarda vardır.

2. ÖN BİLGİLER

2.1. λ- Bilgisi

λ - Algoritmasının alfabesi, yazılım yapısı Church, Kleene, Curry, Turing, Hermes, Scott, Egli, Wadsworth, Barendregt, Korfhage, Robinson, Welch, vb. tarafından ortaya konuldu. Daha sonraları Ünlü, Mirasyedioğlu, Albayrak tarafından çalışıldı. Bu bilgiler için [2], [3], [4], [5], [6], [7], [8], [10], [13], [14], [15], [16], [17], [18], [19], [20], [21] kaynaklarına bakılabilir.

Bu bölümde konu ile ilgili önemli bilgiler kısaca özetlenerek, gösterimlerde kullanılacak olan beş tane yardımcı teorem sunulacaktır.

Tanım 2.1.1:

λ- terimleri " (,) , λ " sembolleri ve x, y, z,... değişkenlerinden oluşan sonlu dizilerdir. Bu diziler λ-teriminin sonlu bir sayıda uygulanmasından elde edilir.

Ayrık sembollerin sonsuz bir dizisinin **değişkenler**, ayrık sembollerin sonlu, sonsuz veya boş bir dizisinin de **sabitler** olduğu kabul edilirse,

λ-bağıntılarının kümesi de denilen λ-terimleri tekrarlı olarak aşağıdaki gibi tanımlanır.

a) Bütün değişkenler ve sabitler λ-terimleridir. Sabitlerin ve değişkenlerin ikisine birden **atomlar** denir.

b) M ve N iki λ-terimi ise, (MN) ' de bir λ-terimidir. (MN) 'ye **application** veya **komut** da denir.

Bir λ-bağıntısının başka bir λ-bağıntısına uygulanmasına **application** denir. (MN) bağıntısında M 'ye **operatör** (işleyen), N 'ye **operand** (işlenen) denir.

Bu demektir ki her λ-bağıntısı hiç bir kısıtlama olmaksızın hem operatör hem de operand olarak kullanılabilir.

c) Eğer M bir λ-terimi ve x de bir değişken ise, (λx.M) de bir λ-bağıntısıdır. (λx.M) 'ye **abstraction** veya M gövde bağıntılı **λ-bağıntısı** da denir.

Abstraction'ın amacı, verilen bir λ-bağıntısından daha farklı bir λ-bağıntısı yapmaktır.

Atomlar en basit λ-bağıntılarıdır. Daha karmaşık λ-bağıntıları application ve abstractiondan oluşan iki bağıntıyı kullanarak elde edilir.

Açıklaması yapılmak şartıyla atomlar ve onlardan oluşan daha üst λ-bağıntıları (örneğin, application, abstraction vb.) türetilebilir ve bunlar iletişimde kullanılan latin alfabesi karakterleri ile gösterilir.

λx.M terimi, M 'de x yerine N değerini koyarak hesaplanan operatörün değerini gösterir (Hindley and Seldin [14]).

Örnek 2.1.1:

(i) ((λx.y)N) = y

(ii) ((λx.x(xy))N) = (N(Ny))

Tanım 2.1.2:

λ-Algoritmasındaki λ-terimlerinin bulunduğu kümeye **λ-terim uzayı** denir ve IB ile gösterilecektir.

Tanım 2.1.3:

λ- Algoritması kuralları ile IB 'de türetilen her λ-bağıntısına λ-terimi denir. Bundan sonra, bir "P λ-terimi" yerine yazımda kolaylık için "P terimi" kullanılacaktır.

Tanım 2.1.4:

P terimi Q teriminin içinde bulunur ($P \subset Q$) bağıntısı Q ' da tekrarlı (recursive) olarak şöyle tanımlanır.

a) P, P 'de bulunur.

b) Eğer P terimi M ' de veya N ' de bulunursa P terimi (MN) ' de bulunur.

c) Eğer P terimi, M 'de bulunur veya $P \equiv x$ ise P terimi ($\lambda x.M$) de bulunur.

"P terimi Q teriminde bulunur"un anlamı:

Örneğin, " $((xy)\ (\lambda x.(xy)))$ içinde iki defa (xy) bulunur, üç defa x bulunur ve iki defa y bulunur" demektir.

Tanım 2.1.5.

M 'nin bulunuşu ve M içindeki atomların, abstractionların, applicationların λ-bilgisinin belli kısıtlamaları altında λ-türetim kuralları ile bulunduğu alandır. Daha kesin bir anlamda şöyle tanımlanabilir.

Bir P bağıntısında $\lambda x.M$ bağıntısı için, M ' nin bulunuşuna λ'nın **faaliyet alanı (scope)** denir.

Örnek 2.1.2:

$P = (\lambda y.yx\ (\lambda x.y\ (\lambda y.z)\ x))$ olsun.

En soldaki λ 'nın faaliyet alanı $yx\ (\lambda x,y(\lambda y,z)x))$ dir. İkinci λ'nın faaliyet alanı $y(\lambda y.z)x$ ve üçüncü λ 'nın faaliyet alanı z 'dir.

Tanım 2.1.6

a) $\lambda x.M$ 'deki λ 'ya yönetici, x 'e bağlayıcı değişken denir.

b) Eğer x değişkeni $P=\lambda x.M$ biçiminde bir bağıntının içinde bulunursa, bu durumda x ' e **bağımlı değişken** denir. Aksi takdirde x serbest değişkendir. P terimindeki serbest değişkenlerin kümesi **FV(P)** ile gösterilir.

c) Serbest değişkeni olmayan λ-terimlerine kapalı terim (closed term) denir.

Örnek 2.1.3:

P= (λx.(z)x) λy.(y)x bağıntısında ilk x bağlayıcı değişken, ikinci x bağımlı değişken, üçüncü x de serbest değişkendir. Birinci y bağlayıcı değişken, ikinci y bağımlı değişkendir. z serbest değişkendir. P deki serbest değişkenlerin kümesi,

FV(P) = {x,z}

dir.

Buradan şu anlaşılıyor ki, bir değişken bir bağıntıda hem serbest değişken hemde bağımlı değişken olarak bulunabilir (Barendregt[6], Revesz [13]).

2.1.5. Örnek:

P∈IB olmak üzere,

(i) P = λy. (λx. (x)y) λy. (y)x

olsun. λy 'den sonra gelen her iki y 'de bağımlı değişkendir.

(ii) P = x (λy. xy)

olsun x değişkeni iki defa serbest olarak bulunmaktadır, y değişkeni bağımlı değişkendir.

Tanım 2.1.7:

Her M, N, x için [N/x]M ' in anlamı bir M bağıntısı içindeki x serbest değişkeninin yerine N nin yazılmasıdır ve aşağıdaki gibi tanımlıdır.

a) [N/x] x =N;

b) [N/x] a = a; a ≠ x olan atomlar için,

c) $[N/x](PQ) = ([N/x]P\ [N/x]Q)$;

d) $[N/x](\lambda x.P) = \lambda x.P$;

e) $[N/x](\lambda y.P) = \lambda y.\ [N/x]P$ eğer $y \neq x$ ve $y \in FV(N)$ veya $x \in FV(P)$;

f) $[N/x](\lambda y.P) = \lambda z.\ [N/x][z/y]P$, eğer $y \neq x$ ve $y \in FV(N)$ ve $x \in FV(P)$.

2.1.8. Tanım:

$\lambda y.x$ bağıntısı, değeri her zaman x olan bir sabit fonksiyonu gösterir. Dolayısıyla,

$[W/x](\lambda y.x) = \lambda y.w$

bağıntısı değeri her zaman w olan sabit fonksiyonu gösterir.

$[w/x](\lambda w.x) = \lambda w.w$

bağıntısı ise λ-birim fonksiyonu gösterir.

Örnek 2.1.6.

(i) $[(\lambda y.xy)/x](\lambda y.x(\lambda x.x)) = (\lambda y.xy)(\lambda x.x)$

(ii) $[(\lambda y.vy)/x](y(\lambda v.xv)) = (y(\lambda v.(\lambda y.vy)v))$

Yardımcı Teorem 2.1.1:

a) $[x/x]M = M$;

b) $x \notin FV(M) \Rightarrow [N/x]M = M$

c) $x \in FV(M) \Rightarrow FV([N/x] = FV(N) \cup (FV(M) - \{x\})$;

İspat: Bakınız Hindley and Seldin [14].

Yardımcı Teorem 2.1.2:

x,y,v farklı değişkenler olsun.

a) $v \notin FV(M) \Rightarrow [P/v][v/x]M \equiv [p/x]M$;

b) $v \notin FV(M) \Rightarrow [x/v][v/x]M \equiv M$;

c) $y \notin FV(P) \Rightarrow [P/x][Q/y]M \equiv [([P/x]Q)/y][P/x]M$;

d) $y \notin FV(P)$, $x \notin FV(Q) \Rightarrow [P/x][Q/y]M \equiv [Q/y][P/x]M$;

e) $[P/x][Q/x]M \equiv [([P/x]Q)/x]M$.

İspat: Bakınız Hindley and Seldin [14].

Tanım 2.1.9:

$P \in IB$, $\lambda x.\ M$ bağıntısını içinde bulunduran bir bağıntı, ve $y \notin FV(M)$ olsun.

$\lambda y.[y/x]M$

ifadesine P ’deki bağımlı değişkenlerin **yeniden adlandırılması** denir.

2.1.8. Örnek:

$\lambda xy{:}\lambda(xy) \equiv \lambda x.(\lambda y.x(xy))$

$\lambda x.(\lambda y.[y/v]x(xy)) \equiv \lambda x(\lambda v.x(xv))$

$\lambda x.[x/u](\lambda v.x(xv)) \equiv \lambda u.(\lambda v.u(uv)) \equiv \lambda uv.u(uv)$.

Yeniden adlandırma işleminde "$P \equiv Q$" için $P \equiv_\alpha Q$ kullanılabilir.

Yardımcı Teorem 2.1.3:

(a) Eğer $P \equiv_\alpha Q$ ise $FV(P) = FV(Q)$

(b) Herhangi bir P ve $x_1, \ldots, x_n$ için P ’ deki $x_1, \ldots, x_n$’in hiç biri bağımlı ve $P \equiv_\alpha P'$ olacak şekilde P' vardır.

(c) $\equiv_\alpha$ geçişme, yansıma ve simetri özelliklerine sahiptir.

İspat: Bakınız Hindley and Seldin [14]

Yardımcı Teorem 2.1.4:

Yardımcı Teorem 2.1.2'de $\equiv$ yerine $\equiv_\alpha$ yazılırsa Yardımcı Teoremin doğruluğu görülür.

İspat: Bakınız Hindley and Seldin [14].

Yardımcı Teorem 2.1.5:

$M \equiv_{\alpha} M'$, $N \equiv_{\alpha} N' \Rightarrow [N/x]M \equiv_{\alpha} [N'/x]M'$.

İspat: Bakınız Hindley and Seldin [14].

2.2. β-Reduction (β-İndirgemesi)

Bu alt bölümde, β-redex ve β-redexin büzülmesi ile ilgili tanım, yardımcı teoremler ve Church-Rosser Teoremi verilecektir.

Tanım 2.2.1

((λx.M)N) biçimindeki bir λ-bağıntısı bir N operandının bir λx.M operatörüne uygulanmasını gösterir.

((λx.M)N) 'nin anlamı M içindeki x 'lerin yerine N 'nin yazılmasıdır.

((λx.M)N) biçimindeki bir λ-bağıntısına β-redex, [N/x]M terimine de **β-redexin büzülmesi** (β-contraction) denir ve P 'nin P' 'ye β-redex büzülmesi

$P \geq_{1\beta} P'$

ile, P 'nin Q ' ya β-indirgemesi

$P \geq_{1\beta} Q$

ile gösterilir.

Eğer herhangi bir karşılık söz konusu değilse 1β kaldırılabilir.

Örnek 2.2.1:

(i) $((\lambda x.\ x(xy))N) \geq_{1\beta} (N(Ny))$

(ii) $(\lambda x.y)N \geq_{1\beta} y$

(iii) $\lambda x.(\lambda y.yx)z)v \geq_{1\beta} [V/x]((\lambda y.yx)z) \equiv (\lambda y.yv)z$

$\geq_{1\beta} [z/y](yv) \equiv zv$

P 'den Q 'ya sonlu sayıdaki büzülmeler serisine **β-indirgemesi** denir. Eğer P λ-bağıntısı hiç indirgeme yapılamıyorsa büzülme sayısı sıfır, indirgemesi kendisidir.

Tanım 2.2.2.

Bir Q bağıntısının hiç bir β-redexi yoksa, Q 'ya bir **β-normal form** denir. Bütün β-normal formların kümesine **β-nf** veya **λβ-nf** denir. Eğer bir P terimi Q 'ya β-nf'da β-indirgemesi yapılırsa, Q 'ya P 'nin bir **β-normal formu** denir. Eğer karışıklık söz konusu değilse β kaldırılabilir.

Örnek 2.2.2:

Örnek 2.2.1: (iii)'de, zv (λx.(λy.yx)z)v 'nin bir β-normal formudur.

Yardımcı Teorem 2.2.1

$P \equiv_\alpha P'$, $Q \equiv_\alpha Q'$ ve $P \geq_\beta Q$ ise, $P' \geq_\beta Q'$ dur.

İspat: Bakınız Hindley and Seldin [14].

Yardımcı Teorem 2.2.2: (β-indirgemesi için Yerine Koyma)

(a) $P \geq_\beta Q \Rightarrow (v \notin FV(P) \Rightarrow v \notin FV(Q)$;

(b) $P \geq_\beta Q \Rightarrow [P/x]M \geq_\beta [Q/x]M$;

(c) $P \geq_\beta Q \Rightarrow [N/x]P \geq_\beta [N/x]Q$.

İspat: Bakınız Hindley and Seldin [14].

Teorem 2.2.1: β-indirgemesi için Church-Rosser Teoremi)

Eğer $P \geq_\beta M$ ve $P \geq_\beta N$ ise,

$M \geq_\beta T$ ve $N \geq_\beta T$

olacak şekilde bir T vardır.

$$
\begin{tikzcd}
 & P \arrow[dl] \arrow[dr] & \\
M \arrow[dr, dotted, no head] & & N \arrow[dl, dotted, no head] \\
 & \exists T &
\end{tikzcd}
$$

İspat: Bakınız Hindley and Seldin [14].

Sonuç 2.2.1: Eğer P 'nin M ve N gibi iki tane β-normal formu varsa M, N 'ye denktir.

3. KONGRUANS TÜRETİMLERİ

3.1 Temel Tanımlar

Bu alt bölümde, Albayrak [2]'de verildiği şekilde IB'de tanımlı EBK, kapsam, kardinal, kartezyen çarpımı, ikili Bağıntı, Erişimli Bilgi Fonksiyonu, Erişimli bilgilerin Denklik Bağıntısı bilindiği kabul edilecek, m modlu bilgiler (m-leme yöntemi ile elde edilmiş bilgi), λ-kültürü tanımları ve λ-Algoritmasında ${}^{m}IB$ 'nin elemanlarını sıralayan bir teorem verilecektir.

Tanım 3.1.1

${}^{k}IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid ... \mid {}^{k-1}Ib$, $(k=0,1,..., L\ (\infty))$, EBK verildiğinde:

a) ${}^{2}IB \leftarrow {}^{0}b \mid {}^{1}Ib$,

${}^{3}IB \leftarrow {}^{0}Ib\ I\ {}^{1}Ib \mid {}^{2}Ib$,

.

.

.

${}^{m}IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid ... \mid {}^{m-1}\ Ib$, $(m= 2,3,...,L)$,

gösterimlerine IB, EBK' lığında 2-leme, 3-leme,..., m-leme yöntemi ile elde edilmiş EBK'lıkları denir.

b) Bir IB, EBK'lığından m-leme yöntemiyle elde olunan (veya türetilen) her bilgi kümesine m-modundadır denir.

c) Bir m-modu bilgisi ${}^{i}Ib$, $(i=0,1,2,..., m-1)$ 'in özel olarak bulunduğu mod belirtilmek istenirse $Mod({}^{i}Ib) = m$ biçiminde gösterilir (Albayrak [2]).

Teorem 3.1.1:

Bir IB EBK'lığından elde edilen ${}^{m-1}IB$ bilgisi ${}^{m}IB$ bilgisince örtülür.

ispat:

${}^{m-1}IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid \ldots \mid {}^{m-2}Ib$ ve

${}^{m}IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid \ldots \mid {}^{m-1}Ib$ olduğundan

${}^{m}IB = {}^{m-1}IB \cup {}^{m}IB \Leftrightarrow {}^{m-1}IB \subset {}^{m}IB$ dir.

Sonuç: 3.1.1:

a) $\forall\, {}^{i}Ib \in {}^{m-1}IB \Rightarrow {}^{i}Ib \in {}^{m}IB$

b) $\forall$ Mod $({}^{i}Ib) \leq m-1$ özelliğine sahip ${}^{i}Ib$ bilgisi $\in {}^{m}IB$ dir.

Uygulama 3.1.1:

Yapısal olarak; L=4 olmak üzere, $IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid {}^{3}Ib \mid {}^{4}Ib$ EBK'lığı verilmiş olsun.

a) (i) ${}^{2}IB \leftarrow {}^{0}Ib \mid {}^{1}Ib$, 2-leme:

(ii) ${}^{3}IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid {}^{2}Ib$, 3-leme:

(iii) ${}^{4}IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid {}^{2}Ib \mid {}^{3}Ib \mid$, 4-leme:

yöntemi ile elde edilmiş EBK'larıdır.

b) IB EBK'lığından elde olunan

(i) ${}^{2}IB$ 2-leme yöntemi ile elde edilmiş bir bilgi öbeği olduğundan 2-modundadır.

(ii) ${}^{3}IB$ 3-leme yöntemi ile elde edilmiş bir bilgi öbeği olduğundan 3-modundadır.

(iii) ${}^{4}IB$ 4-leme yöntemi ile elde edilmiş bir bilgi öbeği olduğundan 4-modundadır.

c) ${}^{1}Ib$ bilgisi gözönüne alındığında, eğer

(i) ${}^{1}Ib \in {}^{2}IB$ ise Mod $({}^{1}Ib) = 2$,

(ii) ${}^{1}Ib \in {}^{3}IB$ ise Mod $({}^{1}Ib) = 3$,

(iii) ${}^{1}Ib \in {}^{4}IB$ ise Mod $({}^{1}Ib) = 4$ dür.

Teorem 3.1.1. gereğince:

$$\left.\begin{array}{l} {}^{4}IB = {}^{3}IB \cup {}^{3}Ib \Leftrightarrow {}^{3}IB \subset^{4} IB \\ {}^{3}IB = {}^{2}IB \cup {}^{2}Ib \Leftrightarrow {}^{2}IB \subset^{3} IB \end{array}\right\} \Rightarrow^{2} IB \subset^{3} IB \subset^{4} IB$$

dir (Albayrak[2]).

Tanım 3.1.2:

Bir t anında, elemanları λ-bilgisi ile tanımlanmış her m-modu Bilgisi (λ-b, ${}^{m}IB$, t) üçlüsü ile gösterilir ve **λ-kültürü** diye adlandırılır (Albayrak [2]).

Uygulama 3.1.2:

$IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid {}^{2}Ib \mid {}^{3}Ib$ EBK'lığı verilmiş olsun. Bu EBK'lığından 3-leme yöntemi ile elde edilmiş olan ${}^{3}IB \leftarrow {}^{0}Ib \mid {}^{1}Ib \mid {}^{2}Ib$ bilgi kümesini gözönüne alalım. ${}^{3}IB$ bir 3-modu bilgisidir. Eğer 3-modu bilgisinin öğeleri bir t anında aşağıda verilen λ-b 'lerden oluşmuşsa

${}^{0}Ib \xleftarrow{t} \lambda x_0 \, {}^{0}G$

${}^{1}Ib \xleftarrow{t} \lambda x_1 \, {}^{1}G$

${}^{2}Ib \xleftarrow{t} \lambda x_2 \, {}^{2}G$

${}^{3}Ib \xleftarrow{t} \lambda x_3 \, {}^{3}G \xleftarrow{t} {}^{0}Ib \mid {}^{1}Ib \mid {}^{2}Ib$

o zaman (λ-b, ${}^{3}IB$, t) bir λ-kültürüdür. Burada t herhangi bir zaman birimi olarak seçilebilir (Albayrak [2]).

Tanım 3.1.3:

Bir $\underline{B}$ = (λ-b, ${}^{m}IB$, t) λ-kültürü verildiğinde, $\underline{B}$ içinde ${}^{m}IB$ 'in elemanları herhangi bir λ-algoritması ile sıraya konulabiliyorsa, $\underline{B}$ 'ye sıralama özelliğine sahip λ-kültürü denir.

Teorem 3.1.2:

λ-algoritmasında en azından λ-b (λ bilgisi) kuralları ile tanımlı,

^{m}IB 'nin elemanlarını sıralayan λ-algoritma yöntemi vardır.

İspat:

Albayrak [2] ve Ünlü [21] verilen S_i karakter dizilerini gözönüne alalım:

a) λ-Algoritmasından rastgele alınan bilgilerin değer olarak bağlantıları $S_0 \mid S_1 \mid S_2 \mid ... \mid Sn$ simgeleri verildiğinde x_1 ve x_2 birbirlerinden farklı iki mantık değişkeni ve S'de bu karakter dizilerinin en son elemanı ise,

$S_i \leftarrow \lambda x_1 ((x_1 S_i)S)$, $\quad$ $i=0,1,..., L \langle\infty$;

$^{0}Ö \leftarrow S_0 \leftarrow \lambda x_1 ((x_1 S_0)S)$,

$^{1}Ö \leftarrow S_0 S_1 \leftarrow \lambda x_1 ((x_1 S_0) \lambda x_1 ((x_1 S_1) S))$,

$^{2}Ö \leftarrow S_0 S_1 S_2 \leftarrow \lambda x_1 ((x_1 S_0) \lambda x_1 ((x_1 S_1) \lambda x_1((x_1 S_2)S))$,

... ...

$^{n}Ö \leftarrow S_0 S_1 S_2 ...S_n \leftarrow \lambda x_1((x_1 S_0) \lambda x_1 ((x_1 S_1) \lambda x_1 ((x_1 S_2)...)S)...)$

değer bağlamaları λ-algoritmasında vardır (Ünlü [21]).

b) $^{m}|K = \{^{i}A \mid i = 0,1,2,...,m-1 \leq L\} = \{^{0}A \leftarrow 0_1\ ^{1}A \leftarrow 1,$

$^{2}A \leftarrow 2,..., \ ^{m-1}A \leftarrow m-1\}$

$0 \leftarrow {}^{0}A \leftarrow \underline{Y}\ {}^{0}\underline{D} = \lambda x_1 \lambda x_2 x_1$

$1 \leftarrow {}^{1}A \leftarrow \underline{Y}\ {}^{1}\underline{D} = \lambda x_1 \lambda x_2 (x_2\ \underline{Y}^{0}\underline{D})$,

.

.

.

$m-1 \leftarrow {}^{m-1}A \leftarrow \underline{Y}^{m-1}\ \underline{D} = \lambda x_1 \lambda x_2 (x_2\ \underline{Y}^{m-2}\ \underline{D})$,

$m \leftarrow {}^{m}\underline{A} \leftarrow \underline{Y}\ {}^{m}\underline{D} = \lambda x_1 \lambda x_2 (x_2\ \underline{Y}\ {}^{m}\underline{D})$,

değer bağlamaları gözönüne alınırsa $({}^{i}Ö\ {}^{i}A) = S_i$. $(i=0,1,2,..., L \langle\infty)$ olduğu[21] de verilmiştir. Bundan dolayı,

$^{i}\text{Ib} \xleftarrow{t} {}^{i}\text{A}$, $(i = 0,1,2,\ldots, m-1)$ değer bağlamaları yapılırsa, $\forall$ ^{m}IB, $^{m-1}\text{Ö}$ aracılığında λ algoritmasındaki sayılar sıralanabilir. (Bu sıralamada, $^{i}\text{Ib} \in {}^{m}\text{IB}$'ye $(i=0,1,2,\ldots,L)$, t anında değer olarak bağlanan λ-bilgisindeki karakter uzunluğu bir sıralama ölçüsü gibi yorumlanacaktır (Albayrak [2]).

3.2. λ- Kongruanslarının Türetimi

Bu alt bölümde birinci, ikinci ve 3.1 bölümlerinde verilen bilgi ve IB, ^{m}IB özel kümelerine dayalı λ-algoritmasında λ-kongruansı ile ilgili tanımlar, teoremler, ((ÜST a)k) komutu ile a^{k}'nın tanımı verilecektir.

Bu tanımlarda geçen, ayrıca ek [1] de belirtilen operatörler Albayrak [2], Mirasyedioğlu [18], Ünlü [21] da bilindiği gibi kabul edilmiştir. Mod'lu bilgilerdeki komut yapısı ve türetim tekniği Albayrak[2,3,5]'e dayanmaktadır.

Tanım 3.2.1:

m ← $\underline{Y}$ $^{m}\underline{D}$, a ← $\underline{Y}^{a}\underline{D}$, b ←$\underline{Y}$ $^{b}\underline{D}$ ve a,b, m ∈IB olsun.

Eğer (a-b) ← ((çıkar)a)b) sayısı m ile tam bölünebiliyorsa a ve b sayılarına modül m 'ye göre **λ-kongruenti** denir.

a ≡ b (mod m) ← (((~a) b)m)

şeklinde gösterilir. Bu tanıma göre,

m | (a-b) ⇔ a ≡ b (mod m)

olacaktır. Yani λ-Algoritmasında,

m | ((çıkar a)b) ⇔ (((~ a)b)m)

olur.

Eğer ((çıkar a)b) sayısı m sayısına bölünmüyorsa a sayısı modül m ye göre b sayısına λ-kongruenti değildir denir ve

(((≁a)b)m)

şeklinde gösterilir.

Örnek 3.2.1:

a←32, b←4 ve a,b,m ∈ IB olmak üzere, λ-Algoritması kuralları ile m | ((çıkar 32)4) ⇔ (((~ 32)4)m) veya m | 28 dir. Buna göre m, 28'in pozitif bölenleri olacaktır. Şu halde, m ∈ {1, 2, 4, 7, 14, 28} olur.

Teorem 3.2.1:

m, a, b ∈ IB λ-Algoritmasında sayılar olsun. (((~a)b)m) λ-kongruenti olmak üzere,

(((~a)b)⇔a = ((topla ((çarp q)m))b)

sağlanacak şekilde λ-Algoritmasında bir q∈IB sayısı vardır.

İspat:

m, a, b ∈ IB λ-Algoritmasında sayılar ve (((~a)b)m) λ kongruent olduğundan,

m | ((çıkar a) b) veya ((çıkar a)b) = ((çarp m)q) olacak şekilde bir q∈IB sayısı vardır. Bu ise teoremde ispatı istenen,

a = ((topla((çarp q)m))b) olacak şekilde bir q∈IB sayısının varlığını gerektirir.

Teorem 3.2.2

a, b, c, m ∈IB λ-Algoritmasında sayılar olsun. (((~a)b)m) λ-kongruenti ise,

(i) (((~((topla a)c)) ((topla b)c))m)

(ii) (((~((çarp a)c)) ((çarp b)c))m)

dir.

İspat:

(((~a)b)m) λ kongruenti olduğundan,

(i) a = ((topla ((çarp q)m))b) olacak şekilde bir q∈IB sayısının olduğu Teorem 3.2.1'den biliniyor. Bu eşitliğin her iki tarafına c sayısı λ-Algoritması kuralları ile eklenirse,

(((~((topla a) c)) ((topla b) c))m)

olur. Bu durumda,

((çıkar((topla a)c)) ((topla b)c)) = ((çarp q)m)

olur. Bu ise ispatı istenen (i) ifadesidir.

(ii) a = ((topla ((çarp q)m))b) olacak şekilde bir q∈IB sayısının olduğu Teorem 3.2.1'den biliniyor. Bu eşitliğin her iki tarafını c sayısıyla λ-Algoritması kuralları ile çarpılırsa,

((çarp a)c) = ((topla ((çarp b)c)) ((çarp ((çarp q)c))m))

olur.

Bu ise ispatı istenen (ii) ifadesidir.

Tanım 3.2.2

Elemanları λ-kültüründe, λ-Algoritması kuralları ile türetilen her kümeye **λ-Algoritması kümesi** denir.

Tanım 3.2.3

IB 'de tanımlanan elemanlarla yansıma, simetri ve geçişme özellikleri olan bir bağıntıya λ-denklik bağıntısı denir ve ifadeleri aşağıdaki gibi yazılır.

(i) (((~a) a)m) (Yansıma özelliği)

(ii) (((~a)b)m) ise (((~b)a)m) (Simetri Özelliği)

(iii) (((~a)b)m) ve (((~b)c)m) ise (((~a)c)m) dir. (Geçişme özelliği)

Teorem 3.2.3

a, b, c, m ∈IB λ-Algoritmasında sayılar olsun. Pozitif sayılar kümesinde tanımlanan λ-kongruentlik bağıntısı bir λ-denklik bağıntısıdır.

İspat:

λ-Algoritmasında negatif sayılar tanımlanmadığından a, b $\in$IB ve a > b iken ((çıkar b)a) < 0 sayısı mutlak değer olarak düşünülecek ve c = b-a sayısının mutlak değeri λ-Algoritmasında $\lambda[C]_{MD}$ şeklinde kabul edilecektir.

Teoremin ispatında, Tanım 3.2.3'de verilen yansıma, simetri ve geçişme özellikleri λ-Algoritması kuralları içerisinde gösterilecektir.

(i) Yansıma özelliği:

m | ((çıkar a) a) = 0 olduğundan (((~a)a)m) dir. Yani, yansıma özelliği sağlanır.

(ii) Simetri özelliği:

a > b ve ((çıkar b)a) = c olsun.

(((~a)b)m) olduğundan m | ((çıkar a)b) ve ((çıkar a) b) = ((çarp q) m) yazılabilir. Yani q$\in$IB sayısı vardır.

((çıkar b)a) = ((çarp λ $[C]_{MD}$)m) olduğundan, m | ((çıkar b)a) veya m | c olur. Bunun anlamı ise λ-kongruetinin tanımı olan (((~b)a)m)'dir. Bu sonuç ise simetri özelliğinin varlığını gösterir.

(iii) Geçişme Özelliği:

Tanım 3.2.3'den (((~a)b)m) ve (((~b)a)m) iken (((~a)c)m) olduğunu göstermeliyiz.

(((~a)b)m) λ-kongruenti iken, a = ((topla((çarp q)m))b) ....(1) olacak şekilde Bir q$\in$IB sayısı vardır.

(((~b)c)m) λ-kongruenti iken, b = ((topla (çarp k)m))c)..................(2) olacak şekilde bir p$\in$IB sayısı vardır. Şimdi (2) ifadesi (1)'de yerine yazılırsa,

a = ((topla c) ((çarp k)m))) ((çarp q)m))

veya

a = ((topla c) ((çarp ((topla k) q)) m))

olur. Bunun anlamı ise,

((çıkar a)c) = ((çarp ((topla k) q))m)

dir. Bu ise,

m | ((çıkar a)c) yani ((~a)c)m)

dir.

Bu Teoremin ispatı ile λ-kongruentlik bağıntısının λ-denklik bağıntısı olduğu gösterildi. Her denklik bağıntısı, üzerinde tanımlanan λ-Algoritması kümesini denklik sınıflarına ayırır. Ayrıca bu denklik sınıfları ikişer ikişer ayrıktır. Örneğin, modül 2 deki λ-kongruanslarının sınıfları, elemanlarının her biri IB'de λ-algoritması kuralları ile türetilen,

K_1 = {0, 2, 4, 6,...} = {((çarp 2) k), k∈IB}

K_2 = {1, 3, 5, 7,...} = {((topla ((çarp 2)k))1), k∈IB}

kümeleridir.

Teorem 3.2.4:

a, b, c, d, m ∈IB λ-Algoritmasında sayılar olsun. (((~a)b)m) ve (((~c)d)m) λ-kongruentleri ise,

(((~((topla a)c)) ((topla b) d))m)

dir.

İspat:

(((~a)b)m) ise a = ((topla b) ((çarp k)m))............(1)

ve

(((~c) d)m) ise c = ((topla c) ((çarp q)m))........ (2)

olacak şekilde k∈IB ve q∈IB sayıları vardır. (1) ve (2)taraf tarafa toplanırken,

p = ((topla k) q) ∈IB gruplaması yapılırsa,

((topla a)c) = ((topla ((topla b) d)) ((çarp p)m))

olur. Bu ise ispatı istenen,

(((~((topla a) c)) ((topla b) d))m)

dir.

Teorem 3.2.5:

a, b, c, d, m $\in$IB λ-Algoritmasında sayılar olsun.

(((~a)b)m) ve (((~c)d)m) λ-kongruentleri ise

(((~((çıkar a) c) ((çıkar b)d))m)

dir.

İspat:

(((~a)b)m) ise a = ((topla b) ((çarp k)m))........(1)

ve

(((~c)d)m) ise c = ((topla d) ((çarp q)m))........(2)

olacak şekilde k$\in$IB ve q$\in$IB sayıları vardır. (1) ve (2) taraf tarafa çıkarılırken,

p = ((çıkar k)q) $\in$IB gruplaması yapılırsa,

((çıkar a)c) = ((topla ((çıkar b) d)) ((çarp p)m))

olur. Bu ise ispatı istenen,

(((~((çıkar a)c)) ((çıkar b) d))m)

dir.

Teorem 3.2.6:

a, b, c, d, m $\in$IB λ-Algoritmasında sayılar olsun.

(((~a)b)m) ve (((~c) d)m) λ-kongruentleri ise,

(((~((çarp a) c)) ((çarp b) d))m)

dir.

İspat:

(((~a)b)m) ise a = ((topla b) ((çarp k)m)).......(1)

ve

(((~c)d)m) ise c = ((topla c) ((çarp q) m))......(2)

olacak şekilde $k \in IB$ ve $q \in IB$ sayıları vardır. (1) ve (2) taraf tarafa çarpılıp,

p = ((topla ((topla ((çarp b) q)) ((çarpk) d))) ((çarp ((çarp k) q))m))

gruplaması yapılırsa,

((çarp a) c) = ((topla ((çarp b)d)) ((çarp m) p))

olur. Bunun anlamı ise

(((~((çarp a)c)) ((çarp b)d)) m)

dır.

3.2.2 , 3.2.4 , 3.2.6 Teoremlerinden şu sonuç ifade edilebilir.

Sonuç 3.2.1:

a, b, c, d, x, y, $m \in IB$ λ-Algoritmasında sayılar olsun.

(((~a)b)m) ve (((~c)d)m) λ-kongruentleri ise,

(((~((topla ((çarp a) x)) ((çarp c) y))) ((topla ((çarp b) x)) ((çarp d) y)))m)

dir.

Tanım 3.2.4:

a, $k \in IB$ için,

$a^k \leftarrow$((ÜST a)k) = ((çarp ((çarp ((...((çarp a)a))a))...))a)

dir.

Teorem 3.2.7:

a, b, k, $m \in IB$ λ-Algoritmasında sayılar olsun. (((~a)b)m) ise

(((~((ÜST a)k)) ((ÜST b) k))m)

dir.

İspat:

(((~a) b)m) $\Leftrightarrow$ m (((çıkar a)b)

dir.

p= ((topla ((topla...((topla ((ÜST a)((çıkar k)1)))((çarp((ÜST a) ((çıkar k)2))b))...)) ((çarp a) ((ÜST b) ((çıkar k) 2))))) ((ÜST b)((çıkar k)1))) gruplaması yapılırsa,

((çıkar ((ÜST a) k)) ((çıkar b)k)) = ((çarp ((çıkar a)b))p)

olur. Dolayısıyle,

m | ((çıkar ((ÜST a) k)) ((ÜST b)k))

olur. Bu ise ispatı istenen,

(((~ ((ÜST a) k)) ((ÜST b)k))m)

dir.

Teorem 3.2.2'de bir λ-kongruansının her iki tarafını aynı c sayısı ile çarpmanın λ-kongruansını bozmadığı görülmüştü. Şimdi verilecek olan teorem sadeleştirmenin nasıl yapılacağını gösterecektir.

Teorem 3.2.8:

a, b, c, d, m $\in$IB λ-algoritmasında sayılar ve (c,m) = d olsun.

(((~((çarp c)a)) ((çarp c)b))m) $\Leftrightarrow$ (((~a)b) ((böl m)d))

dir.

İspat: ($\Rightarrow$)

(((~((çarp c)) ((çarp c)b))m) olsun.

((çıkar ((çıkar c)a)) ((çarp c)b)) = ((çarp q)m)

olacak şekilde Bir q $\in$IB sayısı vardır. Eşitliğin her iki tarafı c $\in$IB sayısı ile bölünürse,

((böl ((çarp c) ((çıkar a)b)))d) = ((böl((çarp q)m))d)

ise

((böl m)d) | ((böl ((çarp c) ((çıkar a)b)))d)

bulunur. (((böl m)d), ((böl c)d)) =1 olduğundan,

((böl m)d) | ((çıkara)b)

dir. Bu ise teoremde ispatı istenen,

(((~a)b) ((böl m)d))

dir.

($\Leftarrow$)

(((~a)b) ((böl m) d) olsun.

((böl m)d) | ((çıkara)b) ise m | ((çarp ((çıkar a)b))d)

dir. Bu ise, m | ((çıkar ((çarp a)c)) ((çarp b)c)) olduğunu gerektirir.Bunun anlamı ise,

(((~((çarp a)c)) ((çarp b)c))m)

dır.

Sonuç 3.2.2:

Teorem 3.2.8'de eğer d = 1 ve (((~((çarp c)a)) ((çarp c)b))m) ise (((~a)b)m) sonucu elde edilir.

Teorem 3.2.9:

a, b, m ∈IB λ-Algoritmasında sayılar sayılar olmak üzere (((~a)b)m) ise (a, m) = (b, m) dir.

İspat:

(((~a)b)m) olduğundan ((çıkar a)b) = ((çarp k)m) olacak şekilde bir k ∈IB sayısı vardır, ve (a, m) | a ve (a, m) | m olduğundan,

(a, m) | ((çıkar a) ((çarp k) m)) ise (a, m) | b olur. O halde (a, m)| (b, m) dir.

Teorem 3.2.10

i = 1,2,...,r için, (((~a)b)m_i) ise (((~a) b) [m_1,m_2..., m_r]) dir.

İspat:

$i = 1,2,\ldots, r$ için, $(((\sim a)b)m_i)$ ise $m \mid ((\text{çıkar } a)b)$ dir. Dolayısıyla, $((\text{çıkar } a)b)$, $m_1, m_2,\ldots, m_r$ sayılarının ortak bir katıdır. En küçük ortak kat tüm ortak katları böldüğünden $[m_1, m_2,\ldots, m_r] \mid ((\text{çıkar } a)b)$ dır. Bu ise,

$$(((\sim a)b)\ [m_1, m_2,\ldots, m_r]\ )$$

olduğunu gösterir.

Tanım 3.2.5

$P(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0$ polinomu λ-Algoritmasında,

$$P(x) = ((\text{topla}((\text{topla}((\ldots((\text{topla}((\text{çarp } a_n)((\text{ÜST } x)n)))((\text{çarp } a_{n-1})$$
$$((\text{ÜST } x)((\text{çıkar } n)1))))\ldots((\text{çarp } a_1)x))a_0)$$

şeklinde gösterilir ve buna **λ-terimli polinom** veya **λ-Algoritmasında polinom** denir. Buradaki katsayıların, üstlerin ve x'in alacağı değerlerin pozitif ve λ-algoritmasında tanımlı sayıları gözden kaçırılmamalıdır.

Teorem 3.2.11:

$a, b, m \in IB$ λ-Algoritmasında sayılar olsun. $(((\sim a)b)m)$ λ-kongruenti ve $P(x)$ bir polinom ise,

$$(((\sim P(a))\ P(b))m)$$

dir.

İspat:

$$P(x) = ((\text{topla}\ ((\text{topla}((\ldots((\text{topla}((\text{çarp } a_n)\ ((\text{ÜST } x)n)))\ ((\text{çarp } a_{n-1})$$
$$((\text{ÜST } x)\ ((\text{çıkar } n)1))))\ldots((\text{çarp } a_1)x))a_0)$$

olsun. Teorem 3.2.7 'den,

$$(((\sim((\text{ÜST } a)\ 2))\ ((\text{ÜST } b)2))m),\ (((\sim((\text{ÜST } a)3))\ ((\text{ÜST } b)\ 3))\ m)$$
$$\ldots\ (((\sim((\text{ÜST } a)n))\ ((\text{ÜST } b)\ n))m)$$

ve Sonuç 3.2.1'den,

$$(((\sim((\text{topla }((\text{topla}...((\text{topla }((\text{çarp } a_n)\ ((\text{ÜST } a)n)))\ ((\text{çarp } a_{n-1})\ ((\text{ÜST } a)\ ((\text{çıkar } n)1))))...((\text{çarp } a_1)a))a_0))\ ((\text{topla }((\text{topla}...((\text{çarp } a_n)\ ((\text{ÜST } b)n)))\ ((\text{çarp } a_{n-1})\ ((\text{ÜST } b)\ ((\text{çıkar } n)\ 1))))...((\text{çarp } a_1)b))a_0))m)$$

sonucu çıkar.

Tanım 3.2.6:

(a, m) = 1 olmak üzere (((~((çarp a)b)) 1)m) bağıntısını sağlayan b sayısına a 'nın modül m 'ye göre **λ-Algoritmasında tersi** denir ve $\overline{a}$ ile gösterilir.

Örnek 3.2.1

(((~7)3)4) ve P(x) = ((topla ((çıkar ((çarp 2) ((ÜST x) 2)))x))3) olsun. a = 7 ve b =3 olduğundan P(a) = 94 ve P(b) = 18 olup (((~94)18)4) dür.

Tanım 3.2.7:

a, b, m ∈IB ve a ≥ b olmak üzere, ((~a)b)m) λ-kongrenti ise b sayısına a sayısının modül m'ye göre indirgenmiş rezidü sistemi denir.

Tanım 3.2.8:

Eğer her bir sayı K⊂IB kümesinin bir ve yalnız bir elemanına modül m 'e göre λ-kongruent ise K kümesi modül m 'ye göre **λ-Algoritmasında bir tam rezidü sistemidir** denir.

Tanım 3.2.9:

∅(n), n 'den büyük olmayan ve n ile aralarında asal olan sayıların sayısını göstermek üzere $\varnothing$: IB $\rightarrow$ IB fonksiyonuna **Euler $\varnothing$- fonksiyonu** adı verilir.

Bu tanıma göre $\varnothing(1) = 1$, $\varnothing(2) = 1$ $\varnothing(3) = 2$, $\varnothing(4) = 2$, $\varnothing(5) = 4$, $\varnothing(6) = 2$, $\varnothing(7) = 6$, $\varnothing(8) = 4$, $\varnothing(9) = 6$, $\varnothing(10) = 4$ dür, p bir asal sayı ise

$\varnothing(p) = p-1$

olacağı açıktır.

λ-Algoritmasında bir indirgenmiş rezidü sisteminde $\varnothing(m)$ tane eleman bulunacağından bu sistemi $(r_1, r_2, \ldots, r_{\varnothing(m)})$ şeklinde yazılabilir.

Teorem 3.2.12:

a, m, $r_{\varnothing(m)} \in$IB için (a, m) = 1 olsun. Eğer $\{r_1, r_2, \ldots, r_{\varnothing(m)}\}$ λ-Algoritmasında bir indirgenmiş rezidü sistemi ise,

$\{((\text{çarp a})r_1), ((\text{çarp a})r_2), \ldots, ((\text{çarp a})r_{\varnothing(m)})\}$

de λ-Algoritmasında bir indirgenmiş rezidü sistemidir.

İspat:

$(r_1, m) = 1$ ve (a, m) = 1 olduğundan $(((\text{çarp a})r_1),m) = 1$ dir

$\{ ((\text{çarp a})r_1), ((\text{çarp a})r_2), \ldots, ((\text{çarp a})r_{\varnothing(m)}) \}$ ve $\{r_1, r_2, \ldots, r_{\varnothing(m)}\}$ kümesinin eleman sayıları eşit olduğundan $i \neq j$ için,

$(((\sim((\text{çarp a})r_i))\ ((\text{çarpa})r_j))m)$

olduğunu göstermek yeterlidir. Kabul edelim ki,

$(((\sim((\text{çarp a})r_i))\ ((\text{çarp})r_j))m)$ olsun, (a, m) = 1 olduğundan $(((\sim r_i)r_j)m)$ olur ki bu mümkün değildir. Çünkü indirgenmiş rezidü sisteminin elemanları kongruent olamazlar.

Teorem 3.2.13: (Euler Teoremi)

a,m,r $\in$ IB olsun.

Eğer (a,m)=1 ise $(((\sim((\text{ÜST a})\varnothing(m)))1)m)$ dir.

İspat:

$\{r_1, r_2, \ldots, r_{\varnothing(m)}\}$ bir indirgenmiş rezidü sistemi olsun. (a,m)=1 olduğundan Teorem 3.2.12 gereğince,

$\{((\text{Çarp } a)r_1, ((\text{çarp } a)r_2), \ldots, ((\text{çarp } a)r_{\varnothing(m)})\}$ kümesi

de bir indirgenmiş rezidü sistemidir. ((çarp a)r_i) elemanlarının her biri bir ve yalnız bir r_j elemanına kongruent olacağından,

$$(((\sim((\text{Çarp}..((\text{Çarp } a)r_1))\ldots ((\text{Çarp } a)r_{\varnothing(m)})))((\text{Çarp}\ldots ((\text{Çarp } r_1)r_2)) \ldots r_{\varnothing(m)}))m)$$

ise

$$(((\sim((\text{Çarp}\ldots ((\text{Çarp}((\text{Üst } a)\varnothing(m))r_1))\ldots r_{\varnothing(m)}))((\text{Çarp}\ldots ((\text{Çarp } r_1)\, r_2))\, r_{\varnothing(m)}))m)$$

bulunur. (((Çarp... ((Çar r_1)r_2)...)$r_{\varnothing(m)}$, m)=1 olduğundan her iki taraf

$r_1, r_2, \ldots, r_{\varnothing(m)}$ ile sadeleştirilebilir. Bu duruma,

$$(((\sim((\text{Üst } a)\varnothing(m)))1)m)$$

bulunur.

Şimdi bu teoremden faydalanarak sayılar teorisinde önemli bir yeri olan teoremi ifade ve ispat edelim.

Teorem 3.2.14: (Küçük Fermat Teoremi)

p bir asal sayı ve a, p 'nin katı olmayan bir sayı ise,

$$(((\sim((\text{ÜST } a)((\text{çıkar } p)1)))\ 1)p)$$

dir.

İspat:

a, p 'nin katı olmadığından $p \nmid a$ ve p asal olduğundan $a \nmid p$ dir. Dolayısıyla (a,p)=1 olur.

p asal olduğundan $\varnothing(p)=1$ olacağından Euler teoremimden,

$$(((\sim((\text{ÜST } a)\ ((\text{Çıkar } p)1)))1)m)$$

bulunur.

3.3. Bir Değişkenli Lineer Kongruanslar

Tanım 3.3.1.

a,b ∈ IB olmak üzere, (((~(Çarpa)x))b)m) şeklindeki bir kongruansa, birinci dereceden bir bilinmeyenli **λ-kongruansı** veya m modülüne göre **lineer λ-kongruansı** denir. Eğer x_1, (((~((çarp a)x))b)m) lineer λ-kongruansını sağlayan bir tam sayı ise bu takdirde k∈IB olmak üzere, ((topla x_1)((Çarp k)m)) de lineer λ-kongruansı sağlar.

Bundan başka (((~((topla x_1)((Çapr k)m))x_1)m) yani x_1 denklik sınıfındaki her eleman bir lineer λ-kongruansı sağlar. Bu sebeple böyle kongruanslar için sadece 0,1,2, . . . ,m-1 sayıları arasındaki çözümleri aramak ve onların denklik sınıfındaki her bir sayının çözüm olacağını düşünmek çok tabidir. Örneğin, (((~((Çarp 5)x))2)6) 'nin çözümleri için sadece 0,1,2,3,4 ve 5 'i düşünerek x_1=4' ün bir çözüm olduğunu buluruz. Yani, (((~((çarp 5)4))2)6) ve böylece $\bar{4}$={x:(((~x)4)6)} kalan sınıfındaki her elemanın bir çözüm olacağı anlaşılır.

Her lineer kongruansın çözümü olmayacağı gibi çözümlerinde sadece bir kalan sınıfta bulunmaları gerekmez, örneğin (((~((Çarp 2)x))4) ' ün hiçbir çözümü yokken (((~((çarp 2)x))6)8) lineer λ-kongruansının çözümleri $\bar{3}$={x:(((~x)3)4)} ve $\bar{7}$={x:(((~x)7)8)} kalan sınıflarındaki elemanlardır. (3.3.1) lineer -kongrüansını sağlayan tam sayılara, m modülünün farklı kalan sınıflarına ait iseler "**denk olmayan çözümler**", aynı denklik sınıfına ait iseler "**denk çözümler**" denir. Lineer λ-kongruansının çözüm kümesi, lineer λ-kongruansı sağlayan kalan sınıfların her birinden sadece bir temsilci elemanın alınmasıyla elde edilen küme olarak tanımlanır. O halde bu çözüm kümesi, kalanlar sisteminde olan ve lineer λ-kongruansı sağlayan kalanlar kümesidir. Örneğin, (((~((Çarp 2)x))6))8) 'in çözüm kümesi {3,7} dir. Buna göre çözüm kümesindeki eleman sayısı bir bakıma kalan sınıfların sayısı ile ilgili olduğu gibi m 'ye de bağlıdır.

Bu kısımda daha çok denk olmayan çözümlerin elde ediliş metodlarını ve bu çözümlerin sayısını araştıracağız. (((Çarp 2)x))6)8) şeklindeki bir lineer kongruansın çözümlerini ararken problemi iki grupta toplamak mümkündür. Bunlar,

i) (a,m)=1

ii) (a,m)=d>1

hallerine karşılık gelir.

Teorem 3.3.1.

a,b,m ∈ IB λ-Algoritmasında sayılar olsun. (a,m)=1 ise (((~((Çarp a)x))b)m) λ-kongruansının çözüm kümesi tek elemandan ibarettir.

İspat:

T, m modülünün herhangi bir kalan sistemi olsun. Teorem 3.2.12. gereğince,

{((çarp a)x) :∈T } kümesi de bir kalan sistemidir. Dolayısıyla bir b∈IB verildiğinde (((~((Çarp a)x_0))b)m) olacak şekilde bir tek x_0∈T vardır.

4.SONUÇ VE ÖNERİLER

A^+, IB, $\underline{B}$ ve ^{m}IB kümeleriyle λ-kültüründe λ-Algoritması kurallarına uygun T_i $(i=1,2,\ldots,18)$ operatörleri yardımıyla λ-kongruansları ve lineer λ-kongruansları türetildi.

IB, ^{m}IB kümeleriyle tanımlı $\underline{B}$ λ-kültüründen türetilen doğal sayılar kümesi yardımıyla başlıca operatörler ve cebirde geçerli matematik sistemler ve diğer özel konular λ-Algoritmasında türetilebilir.

KAYNAKLAR

[1] Akkaş, S., Hacısalihoğlu, H.H., Özel, Z., Sabuncuoğlu, A. "Soyut Matematik", Gazi Üniversitesi Yayın No: 43, Ankara, 1984.

[2] Albayrak, L., "λ-Kültüründen Cebirsel Yapı Türetimi", Ege Üniversitesi Fen Fakültesi, İzmir, 1982.

[3] Albayrak, L.,"λ-Algoritmasında λ-Kültürünün Tanımı ve T_{MT} Operatörünün Türetilmesi", A.Ü. Isparta Mühendislik Fakültesi Dergisi, Isparta, 1985.

[4] Albayrak, L., "Matematik Sistemlerin Bilgisayarla Ayrımı" A.Ü. Isparta Mühendislik Fakültesi Dergisi, Isparta, 1986.

[5] Albayrak, L., "λ-Algoritmasında De Morgan Kurallarının Türetimi ve $T_{MÇ}$ Operatörünün Türetilmesi", A.Ü. Isparta Mühendislik Fakültesi Dergisi, Isparta, 1989.

[6] Albayrak, L., "λ-Algoritmasında Yarı Grup ve Monoid Türetimi", Yüzüncü Yıl Üniversitesi III. Ulusal Matematik Sempozyumu, Van, 1990, s 18-29.

[7] Albayrak, L., "λ-Algoritmasında Grup ve Değişmeli Grup Türetimi, "Doğu Akdeniz Üniversitesi VI. Ulusal Matematik Sempozyumu, Magosa, 1993.

[8] Bakker, J.W., "λ-Calculus and Computer Science Theory" Proceedings of Symposium in Rome, pp 27-54, 1975.

[9] Balcı, M.,"İleri Matematik I", Tümay Yayınları , Ankara, 1994.

[10] Barendregt, H.P., " The Lambda Calculus, its Syntax and Semantics", revised ed. , North-Holland, 1984.

[11] Bayraktar, M. , "Soyut cebir ve Sayılar Teorisi" , Atatürk Üniversitesi Yayınları No: 650, Erzurum, 1988.

[12] Byrkıt, D. R, and Petlofrezzo, A. J., "Elements of Number Theory" Prentice-Hall, New Jersey. 1970.

[13] Church, A., "The Calculi of Lambda-Conversion", Princeton University Press, (Reprinted by Kraus Reprint Corporation, New York, 1965.

[14] Hindley, J.R. and Seldin, J.P. , "Introduction to Combinators and λ-Culculus" , Cambridge Unv. Press, England, 1986

[15] Kleene, S.C., "λ-Definability and Recursiveness" , Duke Mathematical Journal. Vol. 2, 1936, pp.340-353.

[16] Krivine, J.L., Lambda Calculus, Types and Models masson, Ellise Harswood, England, 1993.

[17] Levy, J.J, "λ-Calculus and Computer Science Theory", Proceedings of Symposium in Rome, pp 147-163, 1975.

[18] Mirasyedioğlu, Ş., "λ-Boolean Theory ", International Logic Review 35, pp. 13-27, June, 1987.

[19] Mirasyedioğlu, Ş., "λ-Functional Exclusive or and Logical Equivalence", Karadeniz University, Mathematical Journal, Vol. V, No:1 pp.64-76, Trabzon, 1982.

[20] Revesz, G.E., "Lambda Calculus, Combinators and Functional Programing", Cambridge Unv. Press, England, 1988.

[21]Ünlü, F., "Kuramsal λ-Algoritması", Atatürk Üniversitesi, Yayın no:472, Erzurum, 1976.

EK-1

λ- OPERATÖRLERİ:

$T_1 \leftarrow \underline{Y} = \lambda x_1 \lambda x_2 x_1$	Yanlış Operatörü
$T_2 \leftarrow \underline{D} = \lambda x_1 \lambda x_2 x_2$	Doğru Operatörü
$T_3 \leftarrow \vee = \lambda\, x_1 \lambda x_2\, ((x_1 \underline{D}) x_2)$	Veya Operatörü
$T_4 \leftarrow \wedge = \lambda\, x_1 \lambda x_2\, ((x_1 x_2) \underline{Y})$	Ve Operatörü
$T_5 \leftarrow \dot{-} = \lambda x_1 ((x_1 Y) \underline{D})$	Değil Operatörü
$T_6 \leftarrow \Rightarrow = \lambda x_1 \lambda x_2 ((x_1 x_2) \underline{D})$	Gerektirir Operatörü
$T_7 \leftarrow \dot{\Rightarrow} = \lambda x_1 \lambda x_2 ((x_2 ((x_1 Y) x_1)) x_1$	Gerektirir Değil Operatörü
$T_8 \leftarrow \dot{\Leftrightarrow} = ((T_8\, a) b) = ((T_4 ((T_6\, a) b))\, ((T_6\, b) a))$:	
$(a \mid b \leftarrow \underline{Y} \mid \underline{D})$,	Çift Gerektirir Operatörü
$T_9 \leftarrow \dot{\Leftrightarrow} = \lambda x_1 \lambda x_2\, ((x_2 ((x_1\, \underline{Y})\, x_2)) x_1)$	Çift Gerektirir Değil Operatörü
$T_{10} \leftarrow TD = \lambda x_1 \lambda x_2\, ((x_1 ((x_1 x_1)\, x_2)) \underline{D})$	Tüm Doğruluk Operatörü
$T_{11} \leftarrow TDD = \lambda\, x_1 \lambda x_2\, ((Y ((x_2 x_1) x_1)) \underline{Y})$	Tüm Doğruluk Değil Operatörü

$T_{12} \leftarrow + \leftarrow \text{topla} = \lambda x \lambda y\, ((yx)\, \lambda z\, (\text{ardı}\, ((\text{topla}\ x) z)))$

$T_{13} \leftarrow - \leftarrow \text{çıkar} = \lambda x \lambda y\, ((yx)\, \lambda z\, (\text{öncü}\, ((\text{çıkar}\ x) z)))$

$T_{14} \leftarrow \bullet \leftarrow \text{çarp} = \lambda x \lambda y\, ((y \underline{D})\, \lambda z\, (\text{topla}\, ((x) z)) x))$

$$T_{15} \leftarrow \div \leftarrow \text{böl} = \begin{cases} \lambda x\, \lambda y\, \underline{Y}\ \underline{D} & \text{eðer } x = y, \\ \lambda x\, \lambda y\, \underline{D} & \text{eðer } y - x \langle 0, \\ \lambda x\, \lambda y\, ((yY\,{}^{''}\underline{D}) \lambda u ((x \underline{D}) \lambda v (\text{ardý}((\text{böl}((\text{ç ýkar}\ x) y)) y)))), & \text{eðer } y \rangle x \end{cases}$$

$T_{16} \leftarrow \oplus \leftarrow \underline{\text{topla}_m}$	m moduna göre toplama
$T_{17} \leftarrow \otimes \leftarrow \underline{\text{çarp}_m}$	m moduna göre çarpma

EK-2

GÖSTERİMLER VE KISALTMALAR

A^+	$A^+=\{X_i,\underline{D},\underline{Y},\lambda,(,),A\}$ λ-Algoritmasının alfabesi
$\underline{B}=(\lambda\text{-B},{}^{m}IB,t)$	λ-kültürü, Bir t anında, elemanları λ-bilgisi ile tanımlamış her m-modu bilgisi
IB	λ-terim uzayı
^{m}IB	λ-Algoritmasında m modlu bilgiler veya terimler kümesi
^{i}Ib	(i=0,1,2,.......,k) bilgi konaklıkları
EBK	Erişimli Bilgi Konaklığı
(MN)	λ-Algoritmasında Komut Türü
T_i	T_i (i=0,1,2,. . . ,18) Operatör gösterimleri
λ-b	λ-bilgisi
$\geq$	indirgeme
$\leftarrow$	değer bağlama
$\sim$	λ-kongruent
$\not\sim$	λ-kongruent değil
$\equiv$	denktir
$\in$	elemanıdır, aittir
$\subset$	tarafından kapsanıyor
$\forall$	her
[N/x]M	Yerine Koyma, M ' deki x 'lerin yerine N 'yi yazma
FV(P)	P 'deki serbest değişkenlerin kümesi

ÖZGEÇMİŞ

1969 yılında Isparta Eğirdir'de doğdu. İlk öğrenimini Yakaafşar İlkokulu'nda, Ortaokulu Konya Ereğli İvriz Öğretmen Lisesi'nde, liseyi Isparta Gönen Öğretmen Lisesi'nde tamamladı. 1986 yılında O.D.T.Ü Eğitim Fakültesi Matematik Öğretmenliği Bölümü'ne girdi ve 1992 Temmuzunda Matematik Öğretmeni ünvanını aldı. 1992 yılında Bingöl Anadolu Lisesi'nde Öğretmenliğe atandı ve halen Isparta Anadolu İmam Hatip Lisesi'nde görevini sürdürmektedir.

ISPARTA KENTİ İÇİN EVSEL VE ENDÜSTRİYEL ATIKLARIN TASFİYESİ TESİSLERİ ÜZERİNE BİR ARAŞTIRMA

Zuhal ÖNCÜ

Makine Mühendisliği, Yüksek Lisans Tezi, 61 s., 1996.

Anahtar Kelimeler: Aktif çamur, Damlatmalı filtreler, Uzun havalandırma, Havalandırmalagünler ve stabilizasyon havuzu, Anaerobik çürütme.

İnceleme konusu olan evsel ve endüstriyel atık suların arıtılması insan sağlığı ile doğrudan ilgilidir. Isparta ilinin evsel atıksuları arıtılmamaktadır ve çok az endüstri atıksuyu arıtılmaktadır. Isparta'nın atıksu kalitesi ölçümlerinin henüz ciddi bir durum olmadığını göstermesine rağmen genel olarak arıtma gerekliliği kenar suların kalitesinde aşamalı bir bozulmaya yol açmaktadır. Sularırnı dereden alan köylüler için potansiyel bir sağlık tehdidinin olduğu söz konusudur. Ayrıca sulama amacıyla Devlet Su İşleri (DSİ) kanalından su alan köylülerde vardır. Tabakaneler başta olmak üzere çok miktarda büyük kirlilik oluşturan ve biyolojik arıtma işlemine zarar veren bazı endüstriler bulunmaktadır. Bu yüzden Isparta kentine bir arıtma tesisi yapma zorunlu bir hale geldiği düşüncesindeyim.

Atıksu Arıtma Tesisi (AAT) için teknik yönden 5 değişik proses alternatifi karşılaştırılmaktadır. Bunlar geleneksel Aktif Çamur, Damlatmalı Filtreler, Uzun Havalandırma, Havalandırmalı Lagünler ve Sıtabilizasyon Havuzudur.

Arıtma tesisinden çıkan çamurun arıtılması için de iki seçenek vardır. Bunlar bir katı atık depolama sahasına uzaklaştırma ve tarımsal araziye uzaklaştırmadır. Çamurun uzaklaştırılmadan önce arıtılmasında en iyi yöntemin anaerobik çürütme olduğu kanaatindeyim.

Jüri: Prof.Dr.Hüseyin ŞALVARLI (Danışman)
Prof.Dr.Z.Kazım TELLİ
Doç.Dr.Mehmet KUNDUZ

AN INVESTIGATION ON SEPARATION PLANTS FOR HOUSE AND INDUSTRIAL WASTE IN ISPARTA

Zuhal ÖNCÜ
Mechanical Engineering, Master Thesis, 61 pg., 1996.

Key Words: Active sludge, dropping filter, long-air-condition, laglins with air-coditioned and stabilization pools.

Prufying domestic and industrial waste water that is the study subject directly related to human health. Domesticwaste water is not purufied. In Isparta, the result of mesauring waste water quality slows that, there was not a serious problem. In spete of thet, the quality of aquifer water slows a gredually pollution. For this reason, there is a neccescity of purifying the waste water.For villagers who are taking water from valley, there is a potential health threat, and also there are villagers who take water from channel of DSİ, in order to irrigate there are some industries (for an example; leather processing place) Which made big pollution to enviroment and demaged biological purifing process. For this reason, in my opinion, it is obligatory to construct the prufinig treatment in Isparta with the tecnical aspects of 5 Different Waste Water Treatment (WWT) are compered. These are ; General Active sludge, dropping filter, long air-condition, Laglins with air-conditioned and stabilization pools.

Active sludge (Compared to long air-condition and Laglins with air-conditioned) needs more little energy. All that reasons, make me think the active sludge best alternative to WWT.

To purify the sludge from WWT, there are two alternatives . There are one; disposel of solid trosh to storgoe land and other ; disposal to agricultural land. I think that , the best way of purifying before sludge disposel , is anaerobic fermantation.

Jüry: Prof.Dr.Hüseyin ŞALVARLI (Supervesior)
Prof.Dr.Z.Kazım TELLİ
Doç.Dr.Mehmet KUNDUZ