

T.C.

GEBZE YÜKSEK TEKNOLOJİ ENSTİTÜSÜ
MÜHENDİSLİK VE FEN BİLİMLERİ ENSTİTÜSÜ

BİLİMSEL ARAŞTIRMALARDA KARAR
VEREBİLEN UYGULAMALI BİR SİSTEM
TASARIMI VE GERÇEKLEŞTİRİMİ

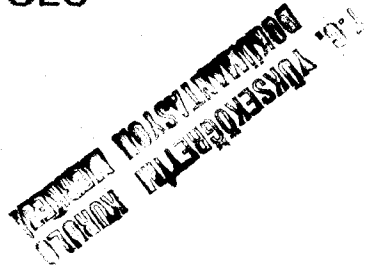
134366

134366

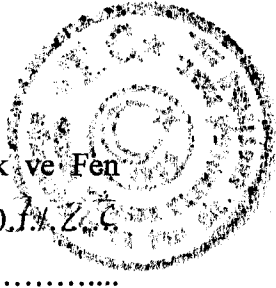
Mehmet POTUR
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

TEZ DANIŞMANI
Doç.Dr.Natali DİMOĞLU

GEBZE
2003



.....Mehmet POTUR.....'ın tez çalışması, GYTE Mühendislik ve Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 16.1.7...2003. tarih ve 2003/1/24 sayılı kararıyla oluşturulan jüri tarafından Bilgisayar Müh...... Anabilim Dalı'nda Yüksek Lisans Tezi olarak kabul edilmiştir.



JÜRI

ÜYE : Doç. Dr. Natali DİMOĞLU
(Tez Danışmanı)

[Signature]

ÜYE : Yrd. Doç. Dr. Mehmet GÖKTÜRK

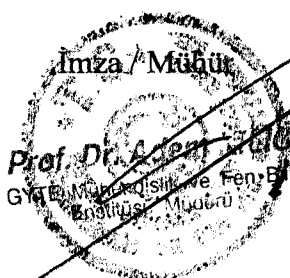
[Signature]

ÜYE : Yrd. Doç. Dr. İlyas KANDİR

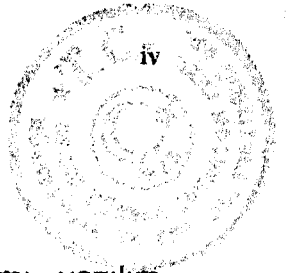
[Signature]

ONAY

GYTE Mühendislik ve Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 21.1.10/2003. tarih ve 2003/1/14..... sayılı kararı.



ÖZET



Farklı bilimsel araştırmalar için, uygulamalı sistem tasarımı, yazılım mühendisliğinin önemli bir yönüdür. Bilgisayar döneminde, insan gücünü büyük oranda azaltmak için tek yol, her branş için yapılan bilimsel araştırmanın hızlı bir şekilde toplanmasıyla elde edilen bilimsel veriyi büyük oranda işlemektir.

Bu çalışma, uzun yıllardır kimyagerlerin ilgisini çeken, bilimsel bir probleme ilişkindir. Bu problem, biyolojik nitelikler üzerinde etkisi olan kimyasal bileşenlerin yapısal özelliklerinin tanınmasıdır. Bu, yapı – aktivite ilişkisi (YAI) (structure activity relationship (SAR)) problemi olarak tanımlanabilir. Bu problemi çözmek için kullanılan tüm olası yaklaşımlar, QSAR ismi ile tanımlanırlar (nicel / nitel SAR metotları (NYAI)). Moleküler modelleme (MM), tüm bu yaklaşımlar için, SAR çalışmasının temeli olarak fakat farklı bir şekilde kullanılır.

Bu araştırmada, öncelikle problemin farklı düzeylerini dikkate almak açısından, tüm olası QSAR yaklaşımları incelenmiştir. Ardından herhangi bir uygulamalı sistemin temeli olan MM hakkında temel görüşler ortaya konmuştur. ETM (graflar ve sıralamalar) talebi için kullanılan alternatif yaklaşımlar tartışılmış ve karşılaştırılmıştır. Sonuç olarak Web için sistem kullanımına ilişkin bazı sorulara yönelik kısa bir tartışma oluşturulmuştur.

SUMMARY

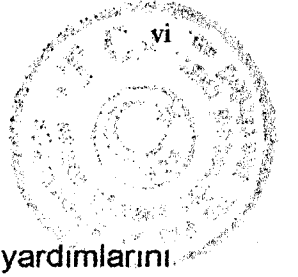


The applied systems design for different scientific researches is an important aspect of software engineering. In the era of computers this is the only way to reduce greatly the amount of human labor needed to process huge volumes of scientific data that are being instantly accumulated in every branch of scientific researches.

In this work, we are concerned with a scientific problem that attracts attention of chemists for many years. It is the problem of recognizing those structural features of chemical compounds that influence their biological properties. It is known as the structure-activity relationships (SAR) problem. All possible approaches used to solve the problem bare the name of QSAR (i.e. quantitative/qualitative SAR methods). Molecular modeling (MM) is being used in all these approaches as a basis for the SAR study but quite differently.

In the thesis, first of all in order to take into account of the following aspects of the problem are studied; a short description of all possible QSAR approaches is given. Next we discuss main aspects of the MM being the basis for any applied system that is intended for the solution of the SAR problem (hypothesis, model). Alternative approaches that are used to supply the ETM (graphs and sequences) are discussed and compared. In conclusion a short discussion of some questions concerning the system use for the Web is given.

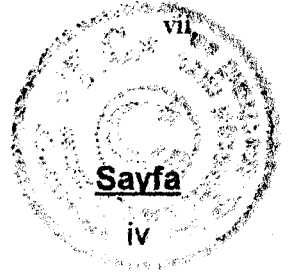
TEŞEKKÜR



Bu çalışmanın hazırlanması sırasında benden yardımlarını esirgemeyen tez danışmanım Doç. Dr. Nataly SHVETS'e, bana gösterdikleri anlayış ve yardımlar için eşim M. Ayla AYYILDIZ POTUR'a, doğumu ile bana çalışma azmi veren kızım Ada POTUR'a, TREDAS Genel Müdürlüğü, TÜBİTAK MAM BTAE ve Araştırma Merkezi Komutanlığı'ndaki çalışma arkadaşlarıma teşekkür ederim.



İÇİNDEKİLER DİZİNİ

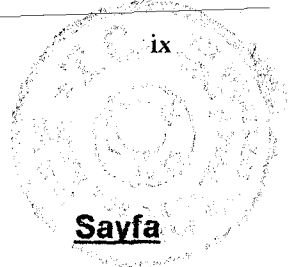


ÖZET	iv
SUMMARY	v
TEŞEKKÜR	vi
İÇİNDEKİLER DİZİNİ	vii
SİMGELER VE KISALTMALAR DİZİNİ	viii
ŞEKİLLER DİZİNİ	ix
TABLolar DİZİNİ	ix
1. GİRİŞ	1
2. NYAI VE MOLEKÜLER MODELLEME:ANA ÖZELLİKLER	3
2.1 NYAI'nin Kısa Açıklaması	3
2.1.1. Nicel YAI Yöntemleri	6
2.1.2. Nitel YAI yöntemleri	7
2.1.3. Yapısal Yöntemler	8
2.2. Moleküler Modellemenin (MM) ETM Açısından İncelenmesi	11
3. ELEKTRO TOPOLOJİK YAKLAŞIMIN TEMELLERİ (ETY)	14
3.1 ETM Modeli ve ETM-TGL dili	14
3.2 Parçaların seçilmesi için ETM Yön temi	17
4. ETM VE İLİŞKİLİ SERİLER (İS) TEKNİKLERİ	21
4.1 İS'yi tanıtmanın amacı	21
4.2. İS'de Desen Tanıma	24
4.3 Kullanıcı Sorgulamaları	26
5 SONUÇ VE TARTIŞMALAR	31
KAYNAKLAR	33
ÖZGEÇMİŞ	36
EK -1:Aktif Bileşikler	
EK- 2: Kuantum Kimyasal Hesaplamaların Çıkış Dosyası	
EK- 3: Aktif Moleküller	
EK- 4: Giriş Dosyası (AKTİF)	
EK- 5: Kaynak Kodlar	
Ek- 6: Çıkış Dosyaları	
EK -7 Yüklele Birleştirilmiş Sıralı Atomik Uzaklık Tablosu	

SİMGELER VE KISALTMALAR DİZİNİ



YAI	Yapı Aktivite İlişkisi
NYAI	Nitel/Nicel Yapı Aktivite İlişkisi
BYAD	Bileşik Yapı Açıklama Dili
ETM	Elektro Topolojik Metot
SA	Sinir Ağları
YSA	Yapay Sinir Ağları
AM	Açıklayıcı Merkez
ETMC	Elektro Topolojik Matrislerin Bileşimi
DT	Desen Tanıma
3B	3 Boyutlu
4B	4 Boyutlu
ETM-GTD	Elektro Topolojik Metot – Graf Teorisi Dili
ETM-İSD	Elektro Topolojik Metot – İlişkili Seriler Dili
MM	Moleküler Metot
İS	İlişkili Seriler
AÖ	Aktiflik Özelliği
İÖ	İnaktiflik Özelliği



ŞEKİLLER DİZİNİ

Sekil

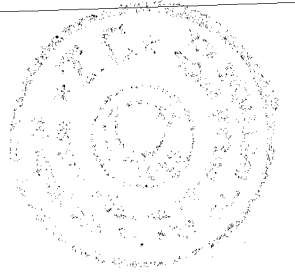
2.1. NYAİ Metotlarının Yapısal ilişkileri	5
3.1. ETM Çevresi	15
3.2. Bir 3B Moleküler Grafik ve Buna Karşılık Olan ETMB	15
3.3. Çok Kalmanlı Elektron-Topolojik Birleşim Matrisi (ETMB)	16
3.4. Yapısal NYAİ ve ETM'de Yaygın Parçaların Aranması	17
3.5. Süper Graf Oluşturulması (b) Ortak Tam Alt Graftlar	
(a) moleküler Graf Karşılaştırma Süreci	18
4.1. Aktivlik ve Pasiflik Özelliklerinin Dağılımı	28
4.2. Esneklik Aralıkları İçin En Uygun Değer Araştırması Örneği	29
4.3. Atom Çiftleri arası Bağlar Bağlamında İlişkili Yükler Toplamı Grafığı	30

TABLULAR DİZİNİ

Tablo

Sayfa

4.1. İki Alternatif ETM Yaklaşımının Özelliklerinin Karşılaştırılması	23
---	----



1. GİRİŞ

Günümüzde bilim adamlarının çalışmalarına hizmet eden uygulamalı sistemlerin tasarımı yazılım mühendisliğinin önemli bir yönüdür. Bilgisayar çağında bu, bilimsel çalışmaların her dalında hızla biriken yüksek miktardaki bilimsel verilerin işlenmesinde duyulan iş gücü ihtiyacını azaltmanın tek yoludur.

Bahsi geçen uygulamalı sistemler oldukça farklı problemleri çözerler, hemen hepsinin esas amacı veri ayıklama, sınıflandırma ve süreç sonuçlarından anlam çıkarmaktır. Anlamlı hale getirilmiş veri bilimsel araştırmaların bir çok farklı alanında karar vermede temel teşkil edebilir.

Doğru karar verme çok önemli bir problemdir; normal olarak bir çok durumda hemen çözülemez. Kural olarak uygun bir modelin tasarımı bu karar verme yolunda ilk adımdır. Bu ilk adım gerçekleştirilecek ve kullanılacak uygulamalı sistemin tasarımında diğer bütün basamakları etkiler.

Bu çalışmada yıllardır kimyagerlerin dikkatini çeken bilimsel bir problemle ilgilendik. Bu, kimyasal bileşiklerin biyolojik özelliklerini etkileyen yapısal özelliklerinin tanınması problemidir. Bu problemi çözmek için kullanılan bütün olası yaklaşımlar NYAI (Nicel / Nitel Yapı Aktivite ilişkisi) ismini alır.

Moleküler Modelleme bütün bu yaklaşımlarda YAI çalışmaları için oldukça farklı bir temel olarak kullanılmıştır.

Tezde problemin şu yönleri üzerinde çalışılmıştır.

Bölüm 2.1'de olası bütün NYAI yaklaşımlarının, problem ile ilgili farklı seviyelerde bilgileri ele alan kısa açıklamaları verilmiştir.

Bölüm 2.2'de MM'nin ana yöntemleri (hipotez, model, modeli iyileştirme yöntemleri) ve bunların YAI problemi çözümündeki rollerine adanmıştır.

Bölüm 3'te altyapı tanıma yönelik NYAI yaklaşımlarında kullanılan desen tanıma tekniklerinin bazı karakteristikleriyle ilgilenilmiştir. Seçilen modelin desen tanıma prosedürünü nasıl etkilediğini gösteren orijinal bir ETM (graf ve seriler) yaklaşımı tartışılmış ve karşılaştırılmıştır.

Bölüm 4'te yaklaşımın gerçekleşmesindeki ET modeli yapısındaki ilişkili seriler temeline dayanan ayrıntılar sunulmuştur.

Sonuç bölümünde sisteminde farklı amaçlar için kullanımıyla ilgili bazı sorulara yönelik kısa bir tartışma yapılmıştır.



2. NYAI VE MOLEKÜLER MODELLEME:ANA ÖZELLİKLER

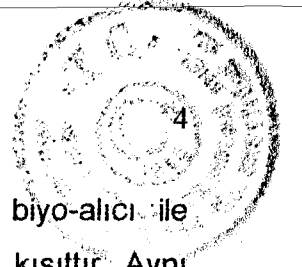
2.1. NYAI'nin Kısa Açıklaması

Yapısal özellikler problemi 19. yüzyıl sonunda kimyada dalında formüle edilmiştir. Problem dünyadaki bütün olası ilişkilerin, birbirleri ile ilişki kurabilen parçalar ile açıklanabileceğini ön kabul olarak alır. Bu yüzden bu problem, günümüz bilimi için hala bir sıcak alandır. Çoğunlukla bu problemler üzerine çalışmalar kimya, biyoloji, tıp, malzeme bilimi, matematik, bilişim teknolojileri gibi birçok bilim dalına yönelik olarak yapılmaktadır.[1]

Bu nedenle teorik kimyada kimyasal bileşiklerin biyolojik aktivitelerini öngörmeyi ve istenilen özelliklere sahip yeni bileşiklerin tasarımını amaçlayan moleküler mühendislik adında yeni bir dal ortaya çıkmıştır.

Prensip olarak, önceden verilmiş özelliklerdeki moleküllerin bilgisayar yardımlı tasarımı aktif bileşik ve onun alt parçaları arasındaki mikro mekanizma ilişki bilgisine -ki bu çoğu durumda önceden bilinmez, dayanmalıdır. Bunun yanı sıra moleküler yapıların aktifliklerine yönelik dolaylı yöntemler geliştirmeleri açısından bir çok çalışma yapılmıştır. Böyle metotlar literatürde nicel yapı aktivite ilişkileri çalışmaları olarak adlandırılır (NYAI) ve bileşik yapısı tanımında yoğun matematik teknikler ve metot gerçeklenmesinde sıkça kullanılan programlama araçları kullanılır. Bu ikisi de NYAI metotları olarak bilinir.

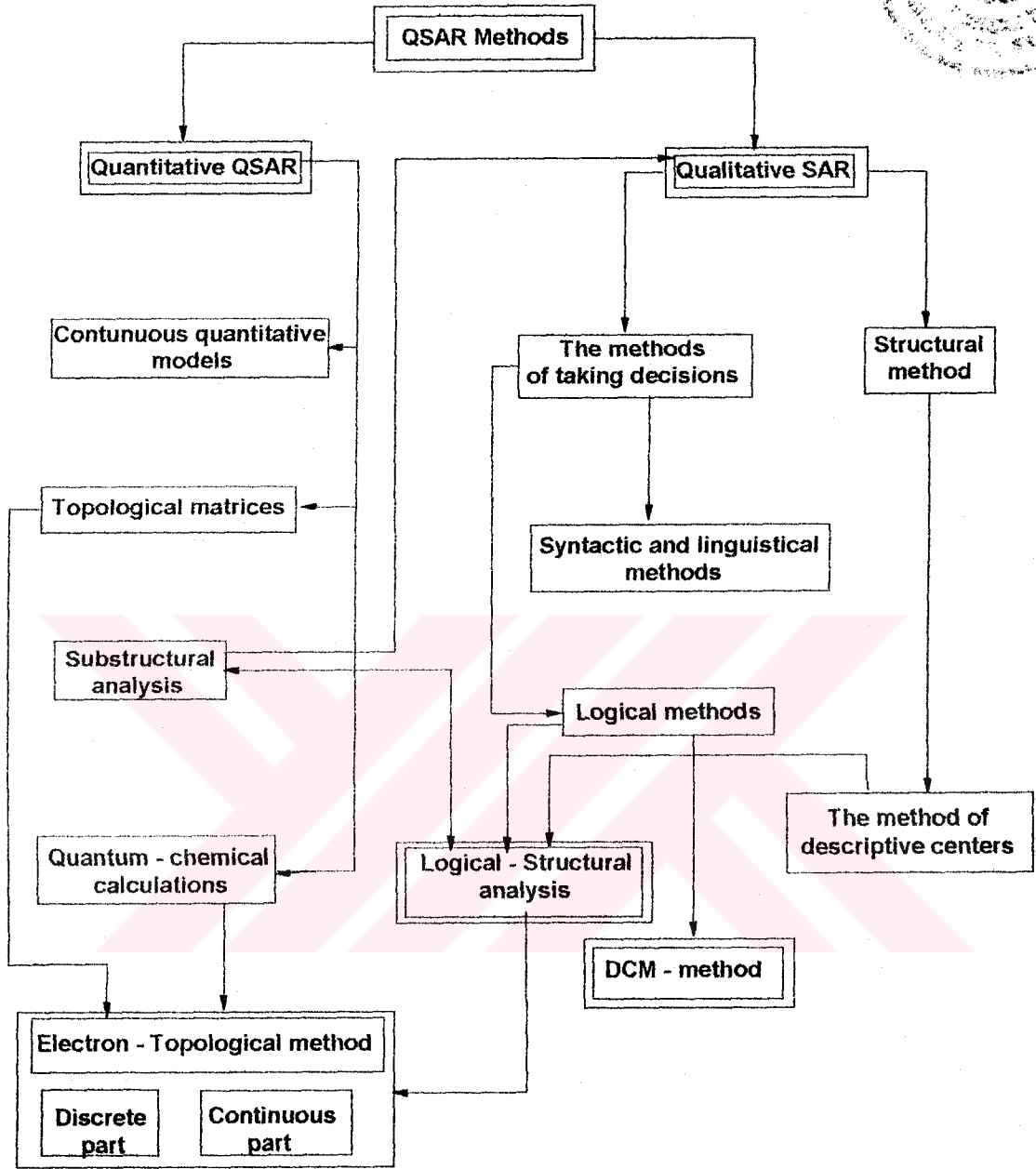
Bir bio-alıcının yapısı bilinirse bir aktif molekülle etkileşimi hemen modellenebilir; fakat bu yöntem çok yüksek miktarda bilgisayar hafızası ve zamanı gerektirir.



Kural olarak bir seferde sadece bir aktif molekülün bir biyo-alıcı ile etkileşim kabiliyeti açısından kontrol edilebilir olması bir başka kısıttır. Aynı yapıdaki bir başka molekülün inaktif olduğunun nedenlerini bulmak için bir yol yoktur. Böyle bir modellemenin olumlu yanı etkileşimi görselleştirme olanağıdır. Bu üç ve dört boyutlu (3B, 4B) olarak adlandırılan YAI tasarımıdır.

Karşıt durumda (algılayıcının yapısı kara kutu olduğu durumda),NYAI metodunun amacı bileşik yapıları (S_i) ve aktivite seviyeleri (A_i) arasındaki bağıntıyı yeniden kurmaktır: $A_i = F(S_i)$, her $i \in I$ için. aktiflik tanımı problemi için iki yol vardır. Birincisi sürekli modelleme paradigmasını kullanır ve amacı altyapıların (S_i) moleküler karakteristiklerinin toplamını temsil eden bazı ara parametrelerden F fonksiyonu için bir analitik ifade bulmaktır. Diğer yol (yapısal modelleme) sadece aktif bileşiklerde bulunan bazı moleküler parçaları (aktiflik kesitleri olarak adlandırılır), bulmaya çalışır. İkinci yol prosedür F 'de bir bileşiğin aktif olup olmadığının ($A=1/0$) belirlenmesinde kullanılır. Burada istatistiksel tahminler baş rolü oynar. Her iki paradigma da her biri sağlam matematiksel temellere dayanan yüzlerce metot ve tekniği ifade eder. Yapısal yaklaşımlar çoğunlukla desen tanıma (DT) teorisi ve değişik biçimlerini kullanır.

NYAI için birçok farklı matematiksel araçlar kullanılmıştır. Bunlardan en çok kullanılanları yapay sinir ağları (SA), bulanık kümeler(BK), graf teorisi (GT) olarak sayılabilir. Günümüzde bunlardan ilk ikisi yaygın olarak sürekli modellemede kullanılmaktadır. GT ise hem sürekli hem yapısal modellemede, örneğin daha sonra açıklanacak Elektro-Topolojik Yöntem (ETY) olarak adlandırılan bir yapısal yaklaşımda olduğu gibi kullanılır.



Şekil.2.1. NYAI Metotlarının Yapısal İlişkileri.

Yapay Sinir Ağları (YSA yada SA) NYAI çalışmalarında sıkça kullanılmaya başlayan bir grup yöntemi ifade eder [1-3]. Bu yöntemler bazı yapı-aktivite ilişkilerini açıklarlar ve bu ilişkinin herhangi bir doğrusal olmayan karakterini hesaba katarlar.

BK'ler, SA yaklaşımlarıyla birlikte uygulandıklarında daha iyi sonuçlar verebilirler. Fakat farklı aktiviteleri tahmin etmedeki potansiyelleri veri seçimi ve gösterimine veya daha iyi ifade etmek gerekirse giriş veri kaynağı olarak hizmet eden modellere oldukça bağlıdır.

2.1.1. Nicel YAI Yöntemleri

Problemi formülize etmek için Y bileşik, A_Y olası aktivitelerinden birisi ve S_Y d yapısı olsun. S_Y genellikle bazı yapısal parametrelerin bir altkütmesi olarak verilir $S_Y = \{S_Y^i\}$, $i=1,2,...,n$, öyle ki $S=(S^1, S^2, \dots S^m)$ tüm bileşikler için ortak moleküler parçalar olsun. $(Y_1, Y_2, \dots, Y_n)$ sadece $X_1, X_2, \dots$ ile değişsin Böylece yukarıda söylenenlere göre:

1. Bir bileşiğin yapısıyla veya yapısının bir bölümüyle biyolojik aktivitesini ilişkilendiren bir $F(S_Y) = A_Y$ fonksiyonu vardır.
2. Fonksiyonun katsayıları normal olarak karşılık gelen S^k parçalarının çalışılan aktiviteye toplamını ifade eder.
3. Eğer $F(S_Y)$ fonksiyonun analitik ifadesi bulunursa bu ifade benzer yapıdaki (aynı S kümesinde) başka bileşik serilerine $(Z_1, Z_2, \dots)$ tahmin edilebilir.

$F_{Z_1}(S_{Z_1}), F_{Z_2}(S_{Z_2}), \dots$ ifadeleri hesaplanarak serideki verilen bir aktivite dizisi $A_{X_1} > A_{X_2} > \dots$ sıralanabilir. S ve aktiflik yoğunluğu arasındaki bağıntı $F(S, A)=0$ denklemi ile de ifade edilebilir.

Bütün nicel yöntemlerin ortak dezavantajları aşağıdadır.

1. Tümleşik bileşik karakteristikleri olarak alınan ve deneysel yollarla bulunan sabitler açık ortak iskeletli bileşik serileri için tutarlı olmayabilir.
2. Sıkça kullanılan farklı alt parçaların aktiviteye göre toplamı bağımsız ve toplanma karakterli olduğu hipotezi genellikle çığnenir. Hipoteze rağmen, ihtiyaç duyulan ilişki karmaşık doğrusal olmayan biçimde davranır.

3. Daha iyi korelasyonlar elde etmek için yapılan hesaplamaların boyutu çok büyük olabilir.
4. farklı fizyolojik çevrelerdeki aynı bileşiklerin davranışları tam hacimde dikkate alınamaz.
5. Bazı zorluklar (bileşiklerin alıcılara ulaşım rotasında gerçekleşen karmaşık kimyasal değişimler nedeniyle: değişimler denklemde kullanılan fiziksel-kimyasal parametreleri dikkate değer ölçüde değiştirebilir) nedeniyle bileşiğin alıcı ile etkileşim mekanizması yeterince açıklanamaz.

Sonuç olarak, eğer bir bileşiğin yapısı **S** bir bütün olarak yani ortak bir iskelet olarak (bir altyapı bile değiştirilirse farklı özelliklerde farklı bir molekül elde edilir)

Düşünülürse, nicel NYAI çok kısıtlıdır. Yapısal yöntemler listelenen dezavantajların üstesinden gelme amacıyla önerilmiştir. İstatistiksel metotların rolü çok daha güçlü metotların belirlediği günümüzde bile küçümsenemez. Şimdilerde yaygınca bilinmeyen YAI'yi kabul ettirmekten ziyade aktiviteden sorumlu tüm moleküler karakteristikleri açığa çıkarmakta kullanılır.

2.1.2. Nitel YAI yöntemleri

Bir grup yapıya dayalı NYAI yöntemi en bilgilendirici yapısal parametre seçimi ve verilen bileşiklerin birkaç aktiviteye göre sınıflandırılmasını (en azından aktif ve aktif olmayan) amaçlarken desen tanıma teorisinin fikirlerini kullanır ve bileşik yapı açıklama dili (BYAD) için bazı özel dillere ihtiyaç duyar.

Bu yüzden gruba ait metotlar arasındaki birincil fark sadece tanımlama dilinin seçimidir. Molekülün aktivitesini etkileyen yapısal parametreler cinsinden tanımlanması daha iyidir ve desen tanıma kullanılması ve moleküllerin aktif ve inaktif olarak ayrılması daha etkindir. Böylece açıklayıcı merkez (AM) fikrinden önce bir gelişme kaydedilmiş olundu.

Genellikle biyo-alıcıların fonksiyonel gruplarıyla etkileştiği varsayılan moleküllerin potansiyel olarak aktif atomları ve/veya alt parçaları AM'ler olarak seçilir. Alt parçalar heteroatomları (N - azot, O - oksijen, S - kükürt, P - fosfor, halojenler, metaller, her biri farklı çevrelerde farklı olarak algılanmalıdır), çevrimsel ve koku alt parçaları, doymamış bağlar, vb. içerir. Her molekül bir AM kümesiyle tanımlanır. Komşu atomları hesaba katmak için yöntemin bazı versiyonlarında aynı molekülün farklı **AM**'leri kenarları ya AM'ler arasındaki uzaklıklar veya bağlarının parametreleri olan bir graf oluşturarak birbirine bağlantılıdır. Bu tanımlar bilgisayarlarla göreceli olarak daha kolay idare edilebilir.

Fakat matematiksel yapılar aynı olduğunda bile elemanlarının yorumu büyük ölçüde değişebilir ve çok çeşitli algoritmalar ortaya çıkabilir. Bunun yanında yazılım sisteminin yapısı, güvenilirlik derecesi, ve algoritmalarıyla elde edilen sonuçların kullanılabilirliği hesaba katılan çok fazla mantıksal faktörlere bağlıdır. Bu faktörler arasında yazılım sisteminin düzeltme eğilimi, kendi kendine öğrenme yeteneği, güvenilirmez yada eksik veriyi iyileştirme yeteneği, uzman mantığı benzetimi vb. sayılabilir.

Başlıca YAI araştırmalarında kullanılan kuantum-kimyasal yöntemler daha sonraları yapısal metotlar için giriş veri kaynağı oldu. Yöntemler ayrık atomların, bağların farklı karakteristiklerini (teorik veya gerçekçi) ve moleküllerin topolojik karakteristiklerini (uzaklık, açı) hesaplar. Karakteristikler kural olarak gerçeğe daha yakın olan deneylerden (yarı gerçekçi karakteristikler), de alınabilir. Nitel YAI, yeni bileşikler çalışma için ortaya konduğunda aktiflik tahmini için bir sisteme sebep olur. Nicel YAI ile kıyaslandığında biraz daha öndedir.

2.1.3. Yapısal Yöntemler

Nitel NYAI'de bir alt grup olan yapısal yaklaşımlar bileşiklerin yapı tanımı ve graf yaklaşımı (veya vektörler ve matrisler) nedeniyle ortaya çıkmıştır. Hepsisi

belirli alt parçaların aktif bileşiklerin moleküler yapılarındaki varlığı söz konusu aktiviteye sebep olur diye tanımlanan alt-parçasal analiz hipotezine uyarlar.

Bunların ortak özellikleri aşağıdaki gibidir.

1. Bileşiklerin kimyasal yapıları için bir tanımlama dili seçin.
2. Karar kuralları sınıfını belirleyin.
3. Bileşikleri aktiflik sınıflarına parçalayın. (seçilen karar kuralına göre)
4. Aktiflik sınıflarına karşı gelmesi için seçilen özellikleri istatistiksel olarak tahmin edin ve seçin.
5. Yeterli sonuçlar varsa ileriki basamaklara geçin, yeterli sonuçlar yoksa karar kurallarını ayrıntılı hale getirin veya değiştirin. (basamak 2)
6. Elde edilen özellikler doğrultusunda biyolojik aktivite tahmini için (yeni moleküler yapıları için) bir sistem kurun.

Taslaktan görüleceği üzere bütün yapısal yöntemler şu hususlarda farklılık gösterebilirler:

- Bileşiklerin yapısal tanımlama dillerinde
- Bileşiklerin aktivite sınıflarına ayrılmasını amaçlayan desen tanıma yöntemlerinde..
- kimyasal yapı tanımlama diline bağlı olan, bulunan aktivite özellikleri formunda

AM fikrinden çıkan iki grup yapısal tanım vardır. Bunların ikisi de vektörler (veya doğrusal tanımlar) ve matrisler algoritmaları tarafından sağlanır. Birinci grup ayrık yapısal parametre örneklerini (yapısal parçalar, fiziksel-kimyasal parametreler) kullanılır. İkincisi moleküllerin topolojisini yani atomik grupların ve atomların uzaysal düzenini tanımlar . (bazı BYAD'ler daha sonraki bölümlerde ayrıntılı olarak tartışılacaktır).

Aktiflik tanıma algoritmaları DT teorisinin farklı yaklaşımları tabanlı olabilir. Bunlardan üçü (sentetik, dilsel ve mantıksal) uzman sistemlerde yaygınca

kullanılmıştır. genellikle yapısal metotlar tarafından kullanılan geometrik ve tahmini yaklaşımlar ek iki yaklaşım örneği olarak verilebilir.

1. Desen Tanımaya geometrik yaklaşım bir C_k (bileşik) nesnesinin bir doğrusal tanımı tarafından bileşenler vektörü S_{ik} , $i = 1, 2, 3, \dots$ olarak kullanılır. (yapısal özellikleri bir çeşit parametrelerle temsil edilir) :

$$C_k = (S_{1k}, S_{2k}, \dots, S_{nk}) \quad (1)$$

Bu n-tuple n boyutlu bir uzayda bir nokta olarak da düşünülebilir. Öklid uzaklığı $D(C_k, C_m) = \sqrt{\sum (s_{ik} - s_{im})^2}$ herhangi iki noktanın yakınlığı için bir ölçü olarak alınabilir. Parametrelerin boşluğu doğru seçilirse aynı aktivite sınıfına ait noktalar birbirine diğer sınıfın noktalarından daha yakın olacaktır. Parametre setlerini değiştirerek ve süreksiz olanları silerek bileşiklerin uygun bir yapısal gösterimi seçilebilir. Fakat aramanın başarılı bir özellik belirleyebileceğinin garantisi yoktur.

Böyle bir yorumda küme fikri (bir net sınıf veya denklik sınıfı) öğrenmenin en basitidir. Kümeleme için bir çok öğrenme algoritması mevcuttur (hiper-düzlemler metodu, araştırmacı katılımlı bazı detaylı arama teknikleri gibi)

Desen Tanımının olasılıksal yöntemi, tanımayı olasılık teorisi ve matematiksel istatistik ile ele almamıza olanak sağlar. Bu sayede bir A_k sınıfının olasılığı $P(A_k)$, ve, sınıfı tanımlayan parametrelerin toplamı S_i için şartlı olasılık değerleri $P(S_{ik}/A_k)$ üzerinde işlem yapılacaktır.

Bildiğimiz gibi desen tanıma algoritmaları için geçerlilik problemi her zaman mevcuttur. Bazı mantıksal çıkarım kuralları belirlenmeli veya parçaları arayan her prosedürde en azından özelliklerin istatistiksel değerlendirilmesi yapılmalıdır. (özelliğin yeteneğini tahmin etmek için).

Geometrik yorumda noktalar arasındaki uzaklıkların toplamını aynı küme için maksimum farklı kümeler için minimum ise, parametre uzayını kümelere ayırmanın en uygun olduğu düşünülebilir. Olasılıksal yorumda sınıflama kalitesi için bir kriter formülize etmemiz gerekir (örneğin yanlış kararlar nedeniyle kayıpların ortalamasını karakterize eden ortalama risk kriteri)

Aktivite arama özellikleri için ortak taslağın tanımındaki basamak 2'den 'bazı durumlarda doğru karar kurallarının aranması alt parça analiz için bir prosedür geliştirilmesi ile paralel olarak yapılabilir' anlamı çıkarılabilir.

Bu çalışmada DT'yı diğer yapısal yaklaşımlara göre avantajları olan bir yaklaşımda kullanıyoruz. Bu yaklaşımın adı Elektro-Topolojik Metodu (ETM)'tur. Bu yaklaşımın taslağı ileride uygun BYAD tasarımı için önemi olan moleküler modellemenin ortak yönlerine giriş yapıldıktan sonra yapılacaktır.

2.2. Moleküler Modellemenin (MM) ETM Açısından İncelenmesi

MM formları diğer bazı NYAI yaklaşımları üzerine temeller oluşturur. DT tabanlı NYAI yazılımının gelişimi için öncelikli olarak ele alınması gereken NYAI 'a ilişkin çözümler ve DT'nin MM'ye ilişkin problemlerin çözümü için bazı seçimler gerekir.

- Bir ya da daha fazla MM hakkında çalışma hipotezi
- Uygun bir moleküler model
- Bileşik Yapı Açıklama Dili (BYAD) için bir dil ve
- DT için uygun matematiksel araçlar

Görünen odur ki, açık ya da kapalı olsun tüm NYAI metotları yukarıda bahsi geçen bu 4 özelliğe dayanır. Başvurulan DT tekniklerinin verimliliği temel

olarak, seçilen modelin uygunluğu ile belirlenir, modelin kendisi, tüm çalışma hipotezlerini sağlamalı, ve sonunda, modelde, dil ve matematiksel araçlar NYAI yaklaşımlarına dayanmalı ve yakından ilişkili olmalıdır.

Böylelikle, ETM yi de kapsayan oldukça az sayıda QSAR metodun PR bağlamındaki çerçeveleri key lock hipotezine dayanır. Bu, etkileşimli bileşenlerin karşıt yapılara sahip olduğunu gösterir.

Bununla beraber, herhangi bir ilişki için, atomik düzeydeki elektronik özelliklerin, moleküler düzeyde olan fiziksel – kimyasal özelliklerden daha önemli rol oynadığı bilinmektedir. ETM için kullanılan moleküler model, verimli olarak oldukça doğal yollarla (moleküller– kuantum kimyasal graflar olarak görülebilir.) elektronik özellikler içerir, çünkü, üç boyutlu moleküler karakteristikler içerir. Kuantum kimyasal grafta her atom, ağırlıklı uçtur ve her kimyasal bağ, ağırlıklı köşedir. Eğer hiçbir köşe, atom çiftlerinin arasında oluşmazsa, bu bilgi açığı atomlar arası mesafenin alınması ile doldurulur.

Moleküllerin esnekliğini de, hesaba katmak için, son zamanlarda bir hipotez daha ortaya atılmıştır. Bu hipotez, temelde 3B ve 4B NYAI'ye dayandırılmıştır. Bu hipoteze göre, çalışma esnasında moleküllerin topolojik olarak değişimi koşulu nedeniyle, çeşitli elektronik karakteristiklerin hesaplamaları, sürekli olarak yenilenmelidir. Bu değişiklikler, moleküller algılayıcılara çok fazla yakınlaştığında oluşur. 2 boyutlu model kullanan (çoğunlukla, kimyasal formüllerden başka bölümler içermezler) pek çok DT Tabanlı NYAI yaklaşımın aksine ETM yöntemde moleküllerin esnekliği de hesaba katılır, fakat, farklı bir yolla, büyük hacimli hesaplamalar gerektirmez. ETM de esnekliği hesaba katmak için, özel bağlantılar, 3 boyutlu grafikler ve onların uç ve bağları ile tanıtılır.

Matematiksel yapıda bahsi geçen 3 boyutlu grafikleri göstermek için, daha önce kullanılan dil Elektro Topolojik Matrislerin Bileşimi (ETMB) dilidir. LCSD



nin en önemli özelliği, 3 boyutlu grafiklere dayanmasıdır ki, bunlar katı biçimsel modellere sahiptir. Bu dil, kısaca ETM – GTD (Graf Teorisi Dili) olarak tanımlanır.

Burada (ilişkili serilerin dili olarak ifade edilen (ETM – İSL) moleküller hakkında daha fazla bilgiyi hesaba katmak için, bileşenleri aktif, inaktif maddeler olarak sınıflandırarak bileşenlerine ayırmanın etkili yollarını bulmak için ve etkinlik tahmini için kullanılmaktadır. Bu iki dil arasındaki farklılıkları göstermek için, daha detaylı bir açıklama verilmelidir.





3. ELEKTRO TOPOLOJİK YAKLAŞIMIN TEMELLERİ (ETY)

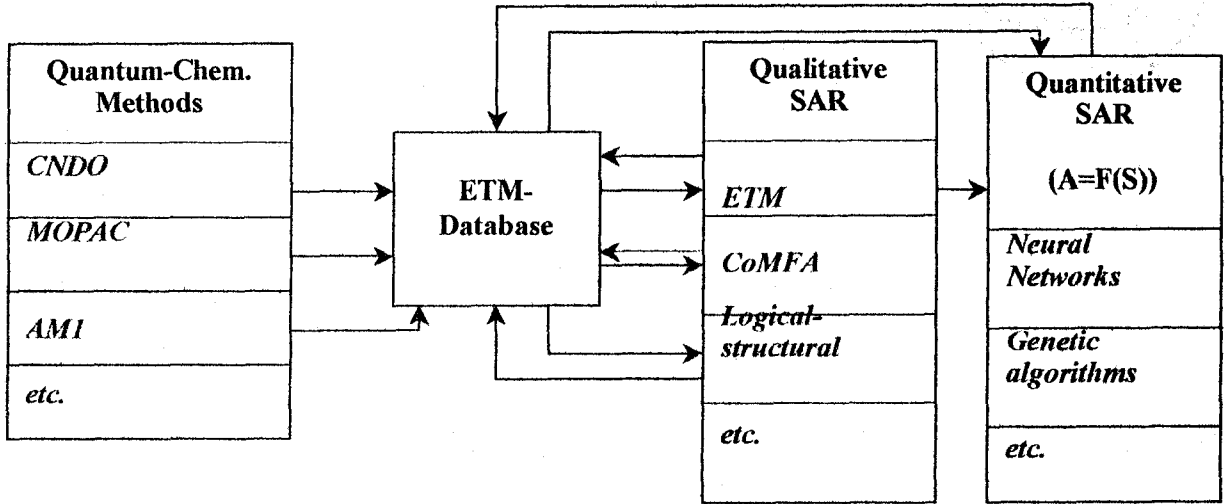
3.1 ETM Modeli ve ETM-TGL dili

Daha önceden söylenmiş olduğu gibi, YAI çalışmalarına yönelik klasik tabanlı yaklaşımlar sıradan kimyasal formüllerine kıyasla kimyasal yapı temsilleri için olan 2B modellerine göre daha çok güvenirlir. Bu yüzden, önemli farmakofor araştırmalarında olasılıkları oldukça sınırlıdır. "Fit-on-fly" model bazlı 3B ve 4B NYAI yeterince verimlidir, ancak bunlar bir oturumda sadece tek bir molekül yapısı için uygulanabilirler. Buna ek olarak hesaplamalar çok uzundur ve tek bir molekül için bile saatler hatta günler sürebilir.

Yani, 2B modellerle temsil edilen molekül serileri üzerinde yapılan az verimli hesaplamalarla verimli ama büyük hacimli 3B, 4B NYAI arasında bir hız boşluğu vardır. Bu boşluğu kapatmaya çalışan iyi bilinen bir sistem ise CoMFA'dır [4], ancak bu yöntem bileşiklerin ortak parçalarını molekülüleri 3 boyutlu uzayda süperimpozlayarak bulmaya çalışırken sadece bileşiklerin topolojilerindeki benzerlikleriyle ilgilenir.

Bu kısıtlamaları aşabilmek için, spesifik bir elektron-topolojik (ET) yaklaşım önerilmiştir [5-7]. Moleküler dinamik ve kuantum kimyası programları topolojik ve elektronik verileri ETM'ye giriş dataları olarak sunarlar. Bu veriler yerel veritabanında saklandıktan sonra, değişik DT yaklaşımlarında kullanılabilirler.

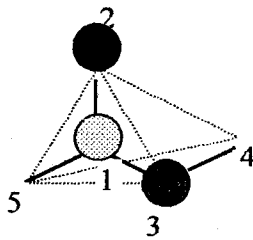
ETM haberleşmesi kendi çevresi ile birlikte aşağıda gösterilmektedir:



Şekil.3.1. ETM çevresi

ETM-3D modeline göre, her bir molekül kenar ve köşelerine (atomlara ve bağlara) bir veya birden fazla elektronik özellikler atanmış tam bir grafikdir. Bir çift atom için hiç bir bağın bulunmadığı durumlarda, eksik olan bilgi 3D uzaklıkların kalıntılarından hesabına telafi edilir.

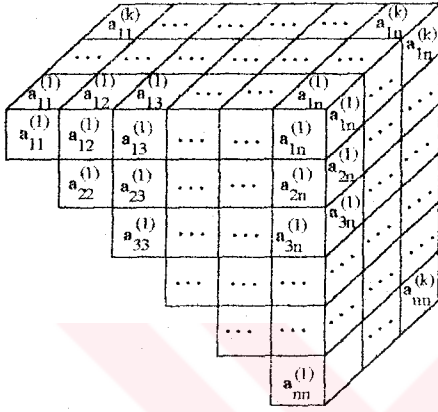
3D grafikleri temsil eden ET bitişiklik matrisleri moleküler yapı tanımları [8] için bir dil olarak ele alınırlar. (Bknz. Şekil.3.1. bu şekil C,O,H atom çeşitleridir, köşegen elementler seçilmiş olan atomik karakterlerin değerleridir, mavi renkler bağ karakterlerine aittir, diğerleri ise mesafelerdir.) Bu küçük molekülün atomları ve bağları için bir özellik seçilmiştir.



1(C)	2(O)	3(O)	4(H)	5(H)
$a_{11}(0.43)$	$a_{12}(1.23)$	$a_{13}(1.32)$	$a_{14}(1.96)$	$a_{15}(1.10)$
	$a_{22}(-0.38)$	$a_{23}(1.95)$	$a_{24}(2.40)$	$a_{25}(1.90)$
		$a_{33}(-0.29)$	$a_{34}(0.95)$	$a_{35}(1.93)$
			$a_{44}(0.15)$	$a_{45}(3.21)$
				$a_{55}(0.09)$

Şekil.3.2 Bir 3B moleküler grafik ve buna karşılık olan ETMB.

Vektörlerin ağılıklar olarak kullanıldığı durumda ETMB matrisi 3 boyutlu olabilir (Şekil. 3.3). Her bir katman bir atomik özelliğinin ve bağ özelliğinin(dış-köşegen elemanlar) tek bir sabit bileşiminin değerlerini taşıyan normal bir matristir. Bir bağ eksik olduğunda karşılık gelen mesafe bütün katmanlardan alınır.



Şekil.3.3. Çok katmanlı Elektron- Topolojik Birleşim Matrisi (ETMB)

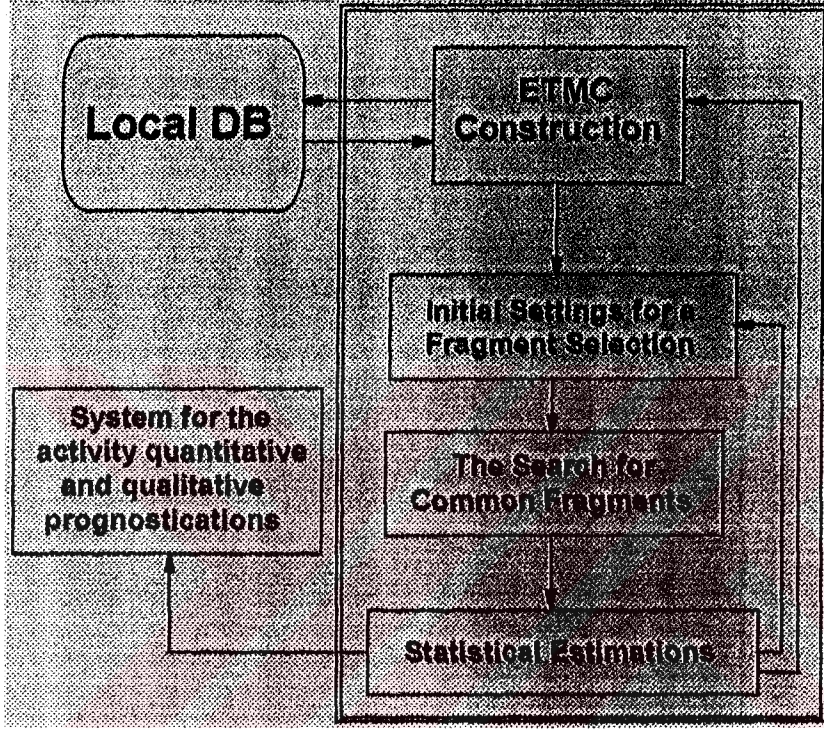
ET modeli genelde kullanılan 3B moleküler modeline çok yakındır; daha önce bahsedildiği gibi, bu model elektronik özelliklerin ve moleküllerin geometrilerin moleküllerin başlıca özelliklerini değişik aktiviteler olarak belirlediği olgusuna dayanır. çeşitli algoritmalarıyla birlikte grafik teorisi bu model için her şeyin başında PR'a uygulanacak doğal bir araçtır.

Ancak tabii ki bütün bu algoritmalar 3B moleküler grafikleri için tanımlanan yeni kimlik ve benzerlik kavramları da hesaba katılarak yeniden türetilmelidir.

Aşağıdaki bölümde, ETM-iş yönteminden moleküler yapıların içerisindeki ortak parçaların aranması için önceden kullanılmış olan ETM yönteminin farkını göstermek için ETMB'leri uygun dil olarak kullanan ETM yöntemleri anlatılmaktadır.

3.2. Parçaların seçilmesi için ETM Yöntemi

Daha önceden, DT'nın adımlarını bütün yapısal yöntemler için listelenmişti. Bu adımlar şematik olarak Şekil. 3.4.'deki gibi gösterilebilir.



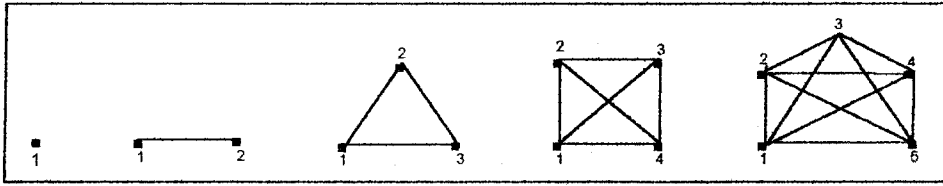
Şekil. 3.4. Yapısal NYAI ve ETM'de yaygın parçaların aranması.

İlk adım (ETMB Oluşumu) ETM modelinin tanımında açık olarak vardır.

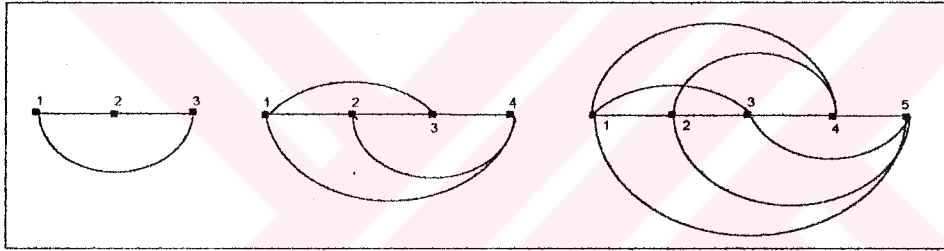
Adım 2 (başlangıç ayarları) aşağıdaki seçimlerle ilgilenir:

- Esneklik aralıkları $\Delta 1$, $\Delta 2$ (sırasıyla ETMB'nin köşegensel ve dış-köşegensel elemanlarının karşılaştırılması için aralıklar, bir şablonla birlikte),
- Şablon bileşiğinin kendisi ve
- Bulunan ortak parçalar için tercih edilen istatistiksel sezim düzeyi.

Moleküler 3B grafiklerin karşılaştırılması yöntemiyle ortak yapısal parçaların aranması yöntemi kısaca aşağıdaki iki resimle açıklanabilir. (Şekil.3.5.a,b) İlk resim değişik boyutluluğun (bu grafiklerdeki köşelerin sayısına bağlı olarak) tam grafiğini göstermektedir. İkinci şekil, ortak alt-grafiklerin daha yüksek boyutluluktaki tam moleküler süper-grafiklerin oluşturulması amacıyla birleştirilmesini (verilen bir çift moleküle ait tüm olası ortak ikili grafların setinden bir tam ikili graf eklenmesi sayesinde) göstermektedir.



(a) (k)-grafikleri ($k=1, 2, \dots, 5$)



(b) $(2) + (2) \rightarrow (3)$

$(3) + (2) \rightarrow (4)$

$(4) + (2) \rightarrow (5)$

Şekil.3.5 Süper graf oluşturulması (b) ortak tam alt graflar (a) moleküler graf karşılaştırma süreci

Herhangi iki molekülün karşılaştırılmasıyla, grafiklerin özdeşliği kavramı grafiklerin denkliği kavramının yerine kullanılmaktadır (her iki kavram da [10] tanıtılmıştır). Her ikisi de seçilen Δi limitlerinde elektronik ve topolojik parametrelerin benzerliğini hesaba katar, ancak aralarındaki fark, özdeşlik özelliğinin atom adlarının (oksijen, nitrojen, hidrojen) da özdeşliğine ihtiyaç duymasıdır. Bu olgu (özdeşlik grafiklerin düzeltilmesi) ETM'yi kimyasal formüllere diğer NYAI metotlarına nazaran daha az bağımlı yapar ve onların verimli DT'da sınırlı kullanılmalarına sebep olur. [11]

Ortak özelliklerin seçilmesini takip eden son adım özelliklerin istatistiksel sezimidir. ETM diğer birçok yapısal yaklaşımda ispat edilmiş olan formülleri kullanmıştır.

P_{α} olasılığı bütün bileşiklerdeki aktivite özelliklerini (aktiflik özelliği ($A\ddot{O}$)) veya bulunmuş aktivite parçalarının varlığını hesaplarken a_{α} olasılığı da parçaların kalitelerini hesaplar:

$$P_{\alpha} = (n_1 + 1) / (n_1 + n_3 + 2) \quad (\text{her zaman } 1' \text{ den küçük}) \quad (2)$$

$$a_{\alpha} = (n_1 * n_4 - n_2 * n_3) / (N_1 * N_2 * N_3 * N_4)^{1/2} \quad (\text{asla } 0.5' \text{ i aşmaz}) \quad (3)$$

Burada n_1 ve n_2 sahip olan, karşılık olarak da, aktif bileşikler sınıfında bulunmuş olan $A\ddot{O}$ 'ye sahip olmayan moleküllerin sayısını, n_3 / n_4 'ün inaktif bileşikler sınıfında analog bir anlamı vardır; N_1 / N_2 ilgili sınıflardaki aktif/inaktif moleküllerin sayısıdır, ve $N_3 = n_1 + n_3$ ve $N_4 = n_2 + n_4$ olarak hesaplanmış olan sayılar da tüm serileri ifade etmektedirler.

Sıklıkla, sadece P_{α} sezimi tüm bileşikleri aktif ve inaktif sınıflarına ayırmak ve anlamlı $A\ddot{O}$ 'ler bulmak için yeterlidir. Ve Etm açıklamasını sonlandırırken, tanımlanmış olan bu adımların hesaplanan olasılık istenen olasılıkla eşit oluncaya kadar değişik başlangıç değerleriyle (Δ 'den biri, şablon bileşik, olasılık seviyesi veya hepsi birden) tekrar edildiğine dikkat çekmek gerekir (Bknz. Adım 2).

Bilinen algoritmalar NYAI problemine ETM ile uyarlanırken, çeşitli ETM uygulamalarında değişik yöntemler uygulanmıştır:

- Bütün eksiksiz alt-grafiklerin geniş araması,
- Azami eksiksiz alt-grafiklerin etkin aranması,
- Grafiklerin tüm anlamlı kliklerinin aynı istatistiksel sezimle aranması.

Bunları daha verimli kılabilmek için, algoritmalar uygulanırken deęişik deneysel ve yarı-deneysel düşünceler hesaba katılmıştır.

Aynı yöntem sadece inaktif bileşiklerde yaygın olan inaktivite özellikleri (İÖ) veya "aktivite aralıkları"nı elde etmek için uygulanabilir.



4. ETM VE İLİŞKİLİ SERİLER (İS) TEKNİKLERİ

4.1 İS'yi tanıtmamanın amacı

Grafik tabanlı ETM diğer yapısal NYAI yaklaşımlarıyla karşılaştırıldığında bir kaç sebepten ötürü daha gelişmiş bir yaklaşımdır.

1. ETM çok çeşitlendirilmiş yapısal iskeletlerle çalışabilir.
2. Aktiviteyi oldukça yüksek olasılıklarla tahmin edebilir.
3. ETM'nin daha belirgin olan özelliği ise; ETM hesaplamaları sonucunda bulunan AÖ ve İÖ sistemi sentezlenmiş bileşiklerde bulunması gereken bu moleküler parçaları göstererek yeni bileşiklerin tasarımı için bir sistem oluşturur (aktivite daha önceden verilmişti).
4. Ayrıca, daha bir pozitif moment de AÖ'leri arama (iki molekülü karşılaştırarak ortak alt-grafikler arama) için kullanılan yöntemlerin ve görüş tahminindeki aktivitenin (yeni bir bileşik yapısında ortak alt-grafik varlığı arama) neredeyse benzer olmasıdır (yazılım bütünlüğü).

Ancak, başlangıç bileşenlerinin %100 kullanımdan bahsedilirse ETM-hesaplamalarında eksiler de vardır. Çünkü, NYAI üzerine yazılmış kitapların çoğunda belirtildiği gibi, bugünlerde sorun bileşikler hakkında herhangi bir veri almak değil, verilen problem için hiç bir önemi olmayan verilerden nasıl kurtulmak gerektiğidir.

Bu hesaplamalar için BYAD olarak kullanılan matrisler,

- Köşegen dışı elemanların karşılaştırılmaları için sadece bir delta kullanırız (daha efektif algoritmalar elde etmek için);
- Ortak alt-grafikler bulmak için veriyi moleküllerin topolojisi üzerinde kısmen kullanırız.
- İstenilen olasılığı biz kendimiz belirleriz (AÖ'leri arama zamanı artar)

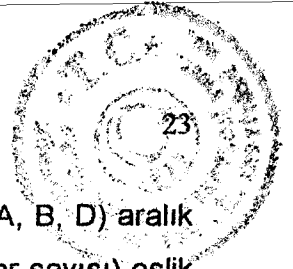
- Bir tam seri bileşik (aktif ve inaktif) içerisinde AÖ'leri arandığında bulunan parçalar hedef bir bileşiğe aittirler (diğer bileşiklerle karşılaştırma için kullanılan bir koordinat sistemine benzer); şablonu değiştirerek koordinat sistemini değiştiririz ancak bu olgu bir miktar gürültüye de neden olur.
- TG algoritmaları kullanılması sayesinde veri süzme ayrı bir işlem haline gelir (ortak alt-grafikleri ararken çok fazla gereksiz bilgi işlenmektedir ve çok fazla gürültülü parça ortaya çıkar)

Sadece değerli olan parçaların aranması bölümünde araştıracının işinin daha fazla otomasyonu işlem zamanının artmasına ve algoritmalarının uzamasına neden olur.YAİ için DT'a daha fazla bilgi girmek ve ETM'ye bazı kullanılmayan olasılıkları eklemek için ilgili diziler olarak bilinen İS'nin kullanılması fikri ortaya çıkmıştır.

Her şeyden önce,

Grafik karşılaştırmanın üç bağımsız aralığında faydalanmak ve hedef bileşiği referans noktası olmaktan kurtarmak için, ET matrisleri 3 RS'e dizileri - elemanları arasındaki ilişki ile ilgili bilgiyi kaybetmeden- ayrı ayrı karşılaştırmak amacıyla bölünmüştür:

- İlgili CSDL'nin köşegeni olarak alınan atomik özelliğin değerleri (ISA ve karşılık gelen Δ_A);
- Sabit bir bağ özelliği için olan üçgensel matrisin lineer gösterimi, mesela 0'lı bir ETMC'nin köşegen dışı elemanları, hiç bağ olmadığı durumda (ISB ve karşılık gelen Δ_B);
- Uzaklıkların üçgensel matrisinin lineer gösterimi (ISD ve karşılık gelen Δ_D).



Dizide her bir değere 3B grafikte modülü ve uzunluğun Δ_i ($i=A, B, D$) aralık sayısına eşit bir tam sayıyı temsil eden indeks (bir tepe veya kenar sayısı) eşlik eder. İlk indeks dizi üzerindeki tüm değişikliklerin kaydedilmesini sağlar.

Üçlülerin seti (R_A, R_B, R_D) bileşiklerin yapısal tanımları için alternatif bir dil olarak kabul edilebilir, veya kısaca ETM-İSD. Karşılık gelen üçlü ($\Delta_A, \Delta_B, \Delta_D$) moleküllerin esnekliğinin ele alınmasında yine hizmet eder. Bu sefer bizim görevimiz sadece bütün bileşiklerin serilerini iki ayrı sınıfa bölmek değil, aynı zamanda en uygun aktivite ve inaktivite (AÖ'ler ve İÖ'ler) parçalarının efektif araması için kriter bulmaktır.

Özetlemek gerekirse, aşağıdaki tablo ETM-TG ve ETM-İS yaklaşımları arasındaki farklılıkları göstermektedir.

Tablo.4.1. İki alternatif ETM yaklaşımının özelliklerinin karşılaştırılması

Metotun özelliği	ETM-TG	ETM-İS
1. 3D moleküler model	+	+
2. Moleküllerin esnekliğinin hesaba katılması	+	+
3. Karar mekanizmalarının önceden seçilmiş olması	+	-
4. Kontrollü veri süzgeçlemesi	-	+
5. Yöntemin geçerliliğinin gerekliliği	+	+
6. Çok katmanlı ETMB'nin eşzamanlı kullanımı	-	+
7. AF aramasında grafiklerin kullanımının gerekliliği	-	+
8. Algoritmaların basitliği	-	+
9. Hesaplama işlemlerinde etkileşimli diyalog	-	+
10. Sonuçların geçerliliğinin gerekliliği	+	-
11. Kullanılmış olan bağlar ve uzaklıklar için ayrı aralıklar	-	+
12. Birleşmiş referans çerçevesi (şablon eksikliği)	-	+



Burada 3 madde ile ilgili açıklama yapmak gereklidir. Kullanıcının program ile diyalogu boyunca AÖ'lerin aranması işlemi devam ettiği sürece, karar alma işlemi her adımda gerçekleşir ve gereksiz verilerin atılmasıyla ilgilidir. Ancak, karar verme kurallarının formülasyonu AÖ'lerin aranmasını mümkün olduğunca otomatik yapabilmek için ETM-İS sisteminin çok sayıda göreve (multiple tasks) uygulanmasının bir sonucu olabilir. Bizim programımız sorgu-yönelik AÖ araması için planlanmıştır.

4.2. İS'de Desen Tanıma

Biz moleküllerin çalışma koşullarında aktif ve inaktif bileşen gruplarına efektif biçimde ayrılma özelliğine sahip olan AÖ ve İÖ'nün yapısal özellikleriyle ilgilenmekteyiz. Bunu başarmak için, öncelikle, (ΔA , ΔB , ΔC) den oluşan ve bu bölünmeyi mümkün kılan bir üçlüye ihtiyaç vardır. Başka bir deyişle, bazı önemsiz değerleri hesaba katmamak gerekir ve sadece aktif parçaları hesaba katan bir aralıklandırma sistem bulmak zorunluluğu vardır.

Değerlerin aralıklara dağıtımı aktif ve inaktif bileşenler için paralel olarak üretilmiştir. Böylesi bir dağıtımın doğru olma durumu, üst üste geçen aralıkların (diğer atomlar, bağlar ve mesafeler için) yokluğudur. Buna rağmen, bu ideal bir durumdur, çünkü her zaman bir aktif parçanın bir parçası olmaya aday birkaç atom vardır. Böylelikle, bizim ana amacımız, tüm olası adaylar (atomlar, bağlar, mesafeler) arasındaki en gözde ilişkileri bulmak ve gereksiz olanlardan arınarak, uygun olanını seçmek olmalıdır.

Çalışmada ifade edilen verinin araştırılması için bir ortak şema aşağıdaki gibi tanımlanmıştır.

- 1.Çözümlenecek görevin özelliklerini belirlemek (bileşenlerin sınıfı, 3 esnek aralık için başlangıç değerleri, bileşenlerin sınıfı, programda data girdisinden sağlanan kimyasal hesaplamalar, vb.,)

2. Dosya girdilerini okuma (aktif ve inaktif bileşenler için) ve her molekül için 3 İS (RS_A , RS_B , RS_D) ile gösterilen moleküler tanımlamaları oluşturma
3. Tüm moleküller için İS'leri sıralama, her karakteristik (A, B, D) için varyasyonların aralığını hesaplama
4. Farklı kullanıcıların sorgulamalarını karşılamak için, çevirim ve sonuçlandırıcı graphları sunma (okuma başvurularını mümkün kılmak ve daha güçlü grafik arayüzler oluşturmak için, veri aynı zamanda, çıktı dosyaları içinde biçimlendirilmiş olarak saklanmalıdır. Bunlar ya tablolardır ya da tüm moleküller için aynı ölçülere sahip olan vektör kümeleridir, ki bu durum özellikle, bahsi geçen grafik programlar için önemlidir.)

Görüldüğü üzere, İS'de, graflar üzerinde, DT için kullanılan tüm bilgiyi koruyabiliriz ve, buna ek olarak, ET Matrislerinin taşıdığı tüm topolojik bilgiyi de içermesi mümkündür. (ancak, tam bir graf elde etmek için tüm üç ilişkili seriye ihtiyacımız vardır; RS_A , RS_B , RS_D .)

Çıktı dosyalarının biçimi aynı zamanda, önemlidir. Çünkü, moleküllerin ETMB 'ye göre tanımlanması doğal olarak güçlü grafik arayüzlerinin kullanımına olanak vermez: moleküller, ET Matrislerine karşılık gelen sıralanmaları belirleyen çeşitli sayılarda atom içerir.

2. aşamada, varyasyon için, (Δ_A , Δ_B , Δ_D) en uygun aralıklar ana algoritmanın modifikasyonları için aynı zamanda ilerleme sağlayabilir. Bunların sınırları görüntünün özelliği ile belirlenir ve genellikle bilinir. Örneğin, eğer, yükler atomlar için (diyagonal elementler) alındıysa, ve Wiberg indisleri bağlar için alındıysa, ardından, Δ_A için, aralık, 0.03-0,15 iken, Δ_B için, bu, 0.05-0.2 ise, (Δ_3) varyasyonu için aralık her zaman 0.05-0.7A olmalıdır.

Kullanıcının sorgulamaları, veri işleme sürecinde yapılan denemeler sonucunda belirlenir. Bu nedenle, sistem gelişmelere açıktır. Bunun anlamı, veri

filtreleme için, yeniden biçimlendirme ve graphlar ya da dosyalar olarak gösterme için yeni prosedürler zamanla eklenebilir.

Şimdi, biz konu üzerinde çalışılan süreçte bulunan bazı uygulanan kullanıcı sorgulamalarını tanımlamalıyız.

4.3. Kullanıcı Sorgulamaları

Yapısal bilgi işleme sürecinde aşağıdaki davranışlar ve bağımlılıklar dikkate alınmalıdır.

- 1.ETM – İS için kullanılan dosya girdilerini oluşturmadan önce, matris sıralamalarını azaltmak için önemsiz hidrojen atomlarını silmek seçeneğini kullanabiliriz.
- 2.Başlangıç datasının değerinin indirgenmesi (üçgen matrislerin bağlar için az yoğun matrisler olduğu gerçeğini kullanabiliriz. Ve bu elementle, beraber bir elementin indeksini korumak için tüm sıfırlar silinmelidir.
- 3.Kullanıcının isteği üzerine, İS'nin bazı elementleri, eğer kullanıcı tarafından tanımlanan koşula uymuyorsa, silinebilir ya da sıfırlanabilir. (örneğin, eğer atomik karakterler yüklü ise, 0,1 den az olan tüm elementler silinmek istenebilir.)
- 4.Kullanıcının isteğiyle bir çeşit graphın gösterilmesi. Graphlar, birkaç tip fonksiyon sunar, örneğin,

A. Bir moleküle karşılık gelen RS_A , RS_B , RS_D sıralarından herhangi birinin eleman dağılımı kullanıcı tarafından seçilmektedir. Bu histogramlara karşılık gelen fonksiyon (ya da aralık dağılımları) şöyledir (karşılık gelen Δ_i sabit ve aralıklar h_k , $k=1, \dots, n_{\Delta i}$ şeklinde numaralandırılmıştır):

$$a. f_{\Delta i} = h_1 * m_1 + h_2 * m_2 + \dots + h_{n_{\Delta i}} * m_{n_{\Delta i}}, \quad (4)$$

m_k s_i sıralamasının h_k de olan karşılıklar toplamıdır. m_k değerleri ya mutlak toplamlardır (örn. h_k aralığındaki elemanlar toplamı) ya da bağıl toplamlardır (NN'de olduğu gibi $m_k / \text{sizeof}(s_i)$). Burada minimal ve maksimal değerler sıralama nedeniyle oluşmaktadır, ve bizim durumumuzda, iki sıralama da pozitif değerler içerebilir.

Tanıtılan fonksiyon basit ama çok önemli özelliklere sahiptir.

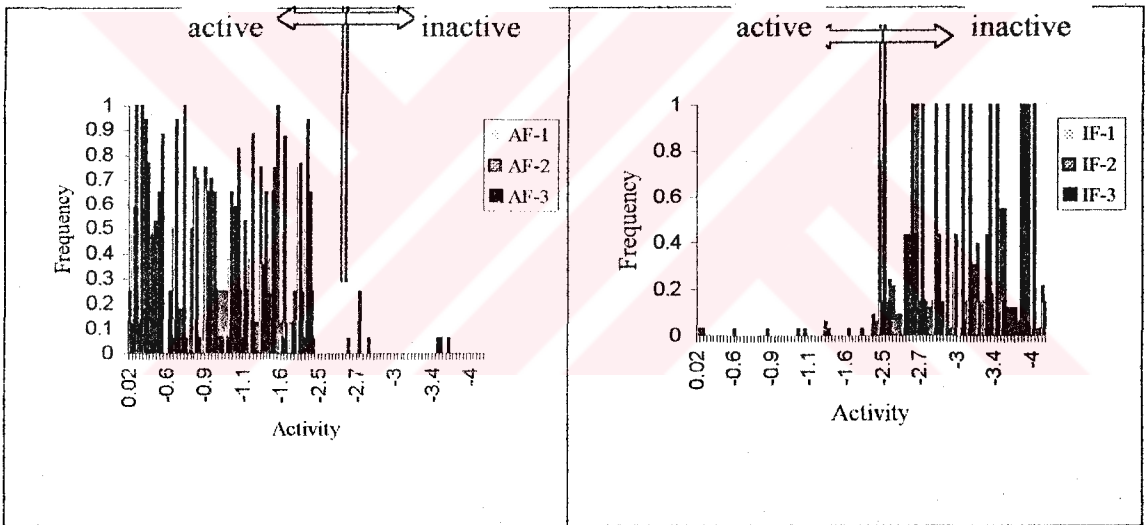
- Eğer h_k bütün k değerleri için 1 ise, ve m_k mutlak toplamsa, $f_{\Delta_i} = \text{sizeof}(RS_i)$ olur (bütün Δ_i değerleri için).
- Çok küçük Δ_i değerleri için bir h_k aralığı birden fazla eleman içerebilir. Ancak Δ_i değeri belli bir aralığın dışındaysa, fonksiyondan anlamlı bir sonuç alınamayacaktır.
- Aynı durum için, ancak bağıl toplamlarla, $f_{\Delta_i} = 1$ olur (tüm Δ_i değerleri için).

B. Birden çok serinin aynı anda işlendiği durumlarda $f_{\Delta_i}^s$ fonksiyonu aşağıdaki yolla hesaplanabilir;

Her bir molekül için, her $m_k > 0$ ise 1 olarak kabul edilir ve böylece bu her bir seri için karakteristik fonksiyon haline gelir. Bir molekül serisi için (örneğin tüm aktif bileşikler için) tüm moleküllerin toplamı hesaplanır ve bu da seriyi belirten h_k aralıklarının tahminini ifade eder. Diğer bir deyişle, bu sonuç katsayılar kesin tahmin yöntemiyle belli aralıkta değeri bulunan molekül sayılarıdır. Bağıl tahmin yöntemi ise bu katsayıların seriler içinde bulunan toplam molekül sayısına oranlanması ile ifade edilir. Sonuçlar ayrıca uygun bir görsel yöntem ile de desteklenebilir (Örneğin aktif ve inaktif serilerde değerlerin aralıklara göre dağılımı grafiği).

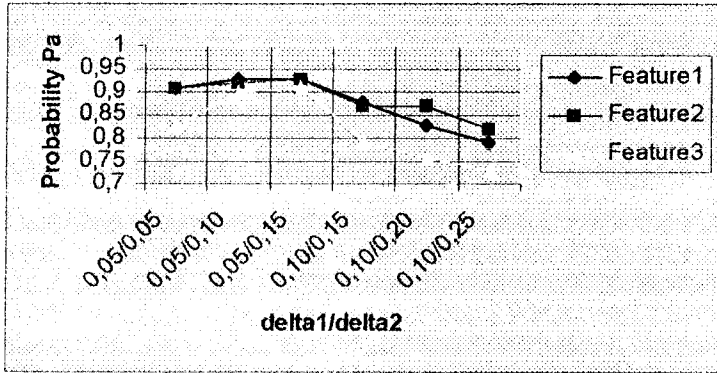
Bu arada bu üç İS'nin hepsinin aynı yöntemle incelenmesi gerektiği unutulmamalıdır. $f_{\Delta_i}^s$ fonksiyonunun bir seri kümesine uygulanması (i sabit, $i=A, B, D$ için) bize anlamlı aralıklar serisini verir.

Tanım yönünden (RS_A , RS_B , RS_D) serileri birbiri ile bağıntılıdır (Bunları üçünün bileşimi bir molekülü ifade eder). Böylece aynı istatistiki tahmin seviyesi ile bulunmuş olan anlamlı aralıklardan, bunlara karşı düşen alt graflar tekrar oluşturulabilir ve grafik olarak gösterilebilir. Bu bağlamda bir çift indis yardımıyla atomlar arası tüm ilişkiler tekrar oluşturulabilir. Dönüşümü sağlamak için iki tane fonksiyona ihtiyacımız vardır. Bu fonksiyonlar “geri” diyebileceğimiz ve bir IS elemanın ilgili matris indisinin doğrusal indisten hesaplanmasını sağlayanlardır. Hesaplamanın doğruluk kontrolü tekrar oluşturulan ortak alt grafin alternatif grup içinde olmaması ile yapılabilir.



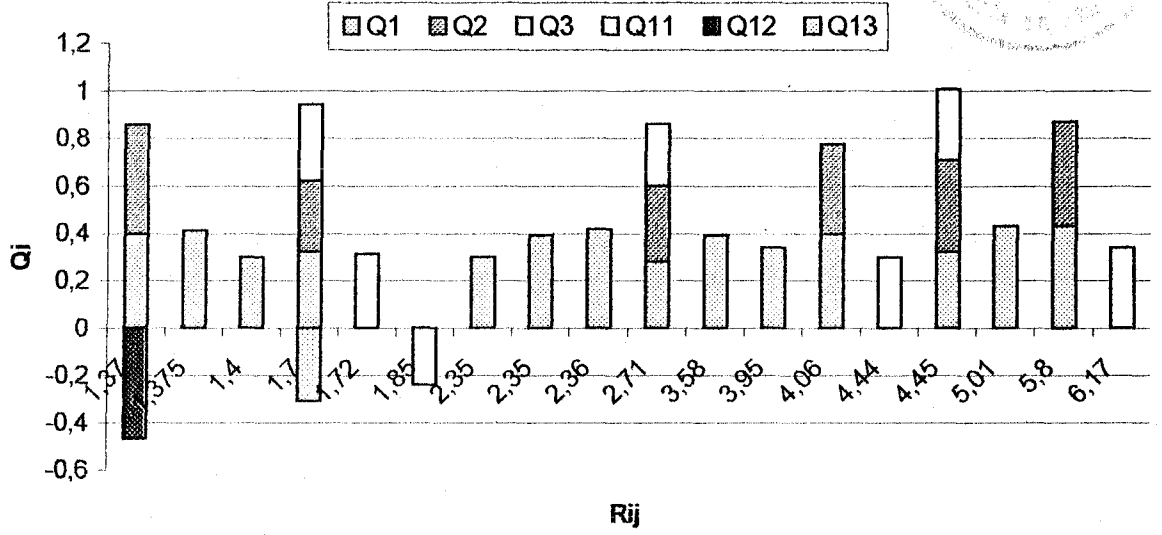
Şekil.4.1 Aktiflik ve pasiflik özelliklerinin dağılımı

D. (Δ_A , Δ_B , Δ_D) Aralıkları için değişik değerler denenirken bir tane önemli karakteristik daha hesaplanabilir. Kullanıcının uygun bir dağılım bulunduğuna karar verdiği anda Δ_A/Δ_B , ve Δ_A/Δ_D arası oranlar hesaplanabilir. Bu iki değerden oluşturulan graflar en uygun değer araştırmasında yararlı olabilir.



Şekil.4.2. Esneklik aralıkları için en uygun değer araştırması örneği.

E. Köşegen dışı önemli elemanların (R_B , veya R_D) ve bunlara karşı düşen atomik özelliklerin (R_A) oluşturduğu grafların araştırması için önemli olduğu ortaya çıkmıştır. Bir atom çifti (i, j) Q_i , Q_j , yüklerine sahip olsun, toplam yük $Q_{ij} = |Q_i| + |Q_j|$ ise, bunun atomik bağ değerlerinin yada atomlar arası mesafelerin karakteristik bağımlılıklarının gösterilmesinde kullanılabileceği ortaya çıkmıştır. Bir seri incelendiğinde, bahsedilen graf bir moleküle pasaport görevi görebilir ve ileride aktiflik tahmininde kullanılabilir. Fakat bu bağımlılık üstünde daha fazla araştırma yapmak gerekmektedir.



Şekil. 4.3. Atom çiftleri arası bağlar bağlamında ilişkili yükler toplamı grafiği (Pozitif bölüm için 3 aktif molekül, negatif bölüm için 3 inaktif molekül kullanılmıştır.)

5. SONUÇ VE TARTIŞMALAR

Bu çalışmada moleküler bilgileri düzenleyen, sıralayan ve aralıklara dağıtıp YAI araştırmasında kullanan, ayrıca kullanıcının taleplerine cevap veren etkileşimli bir sistem gerçekleştirilmiştir.

10 ve 30 aktif/inaktif benzer yapıli bileşik içeren iki seri üstünde çalışmalar yapılmıştır. (Ek1.)

Bileşenler arasındaki benzerlik ve farklılıkları bulabilmek amacıyla en az iki değişik bileşik grubuna ihtiyacımız olduğu için (aktif/inaktif), bileşik aktifliğinin nitel tahmini yöntemi (0/1) kullanıldı. Eğer aktivite nicel olarak ölçülürse, kullanıcıni talebi doğrultusunda, ikiden fazla grup oluşturulabilir.

ETM-İS'de DT Δ_i değişimlerine (Δ_i birbirinden bağımsız değişebildiği için) bağıli olan tüm dağılımların tarihçesini tutan iki yada daha fazla sayıda üç boyutlu tablo üstünde uygulanabilir.

İncelenen kümelerde aralıklar için en iyi dağılım bulunduktan sonra istatistiki karakteristikler ve bunlara ait üç delta (Δ_A , Δ_B , Δ_D) ileride DT için kullanılabilecek bir sistem oluşturur. Aslında değişik ara yüzlerin kullanımı sayesinde bu aşamada oluşturulmuş ve önceden filtre edilmiş sonuç dosyaları ilerideki çalışmalar için kaynak olarak kullanılabilir.

Bahsedilen algoritmalar [16], içinde DT için bir çok değişik yöntemler içeren büyük bir YAI araştırma sisteminin bir parçası olarak gerçekleştirilmiştir.

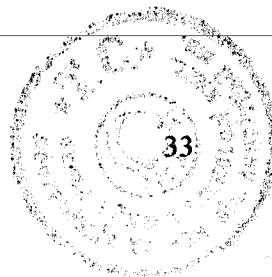
Ayrıca bu, 3 boyutlu verinin filtre edilmesi ve yakınlık kriterine göre sınıflandırılması için kullanılabilecek bir bileşen olarak gerçekleştirilebilir.

Gerçeklenen sistem bir veritabanı sayesinde desteklenebilir (Veritabanı yapısı çalışma sırasında tasarlanmıştır). Veritabanı kullanımı için gerçekleştirilen algoritmaların çıktılarının yönlendirilmesi yeterli olacaktır.

Bu sistem web uyumlu bir uygulama olarak gerçekleştirilebilir. Bu uygulamada tüm hesaplamalar sunucu da gerçekleştirilir ve imciye sadece ilk değerler ve sonuçlar aktarılır.

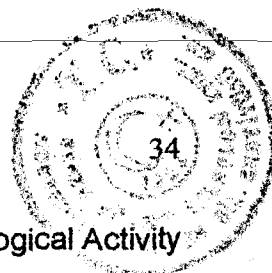
Bu uygulamanın avantajları gelişmelere açık olması, ETM ve SA'lar için kullanılan programlara göre basit ve aynı zamanda çok hızlı olmasıdır.



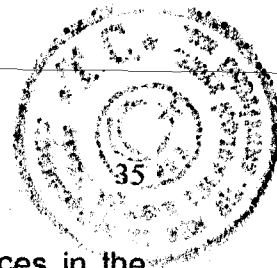


KAYNAKLAR

1. Kovesdi, L.; Dominguez-Rodriguez, M.F.; Orfi, L.; Naray-szabo, G.; Varro, A.; Papp, J. G.; Matyus, P. Application of neural networks in structure-activity relationships. *Med. Res. Rev.* 1999, 19, 249-269.
2. Jalali-Heravi, M.; Parastar, F. Use of artificial neural networks in a QSAR study of anti-HIV activity for a large group of HEPT derivatives. *J. Chem. Inf. Comput. Sci.* 2000, 40, 147-154.
3. Manallack, D.T.; Livingstone, D. J. Neural networks in drug discovery: have they lived up to their promise? *Eur. J. Med. Chem.* 1999, 34, 195-208.
4. Svante Wold, Erik Johansson, Marina Cocchi, PLS – Partial Least-Squares Projection to Latent Structures.
5. H. Kubinyi, 3D QSAR in Drug Design: Theory, Methods and Applications, ESCOM, Leiden, 1993.
6. Dimoglo, Compositional Approach to Electronic Structure Description of Chemical Compounds, Oriented on Computer Analysis of Structure - Activity Relationships, *Chim.-pharm. J.*, 4, 438-444, 1985.
7. N. Shvets, A. Dimoglo, The Electron-Topological Method (ETM): Its Further Development and Use in the Problems of SAR Study. In Book "Molecular Modeling and Prediction of Bioactivity" Edit. By K. Gundertofte and F.S. Jorgensen, Kluwer Academic/Plenum Publishers, New York, 418-429, 1999.
8. G.E. Kellogg, S.F. Semus, HINT, A New Method of Empirical Hydrophobic Field Calculation for CoMFA, *J. Comput. Aided Mol. Des.*, 5, 545-552, 1991.

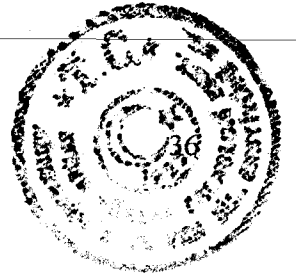


9. N. Shvets, Applied Program System for The Prognosis of Biological Activity of Chemical Compounds: Development and Use, Computer Sciences J. of Moldova, 3, 101-110, 1993.
10. Bersuker, A. Dimoglo, The Electronic-Topological Approach to the QSAR Problem, in Book: "Reviews in Computational Chemistry" (Ed. K.B. Lipkowitz and D.B. Boyd) VCH, New York, Chapter 10, 1991.
11. N. Shvets, A. Terletskaya, A. Dimoglo, Yu. Chumakov, Study of The Electronic and Structural Features Characteristic of 4,5-Dihydro-1-Phenyl-1H-2,4-Benzodi-Azepines Demonstrating Antiarhythmic Activity, TEOCHEM, 463, 105-110, 1999.
12. N. Shvets, The Study of Data and Control Flows and the User Interface Organization in An Applied System Used in Chemistry and Medicine for the Biological Activities Prediction, Comp. Sciences J. of Moldova, 3, 301-311, 1997.
13. N. Shvets, I. Soğukpınar, A. Dimoglo, Pattern Recognition Techniques Application in a Software System for the "Structure-Property" Relationships Study in Chemistry, Biology and Medicine, The 16th Int. Symposium on Computer Science and Information Sciences (ISCIS-16), Nov. 5-7, 2001, Antalya, Turkey, 289-296, 2001.
14. N. Shvets. Program Implementation of the SAR Investigations for the Series of Compounds Oriented on the Interactive Experiment Planning. / Proceedings of the 9-th All-Union Conference "Chemical informatics"-Chernogolovka, Moscow, 11-15 of Jan., 226, 1992.
15. N. Shvets, A. Dimoglo, Structure-Odor Relationships: Results of An Applied Electronic-Topological Approach, Nahrung, 42, 6, 364-370, 1998.



16. N.SHVETS, M.POTUR, Pattern Recognition on Related Sequences in the Frameworks of a System for the Computer-Aided Molecular Design, TAINN, Canakkale, Turkey, 2-4 of July, 2003, p.125.

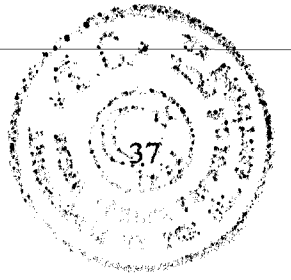




ÖZGEÇMİŞ

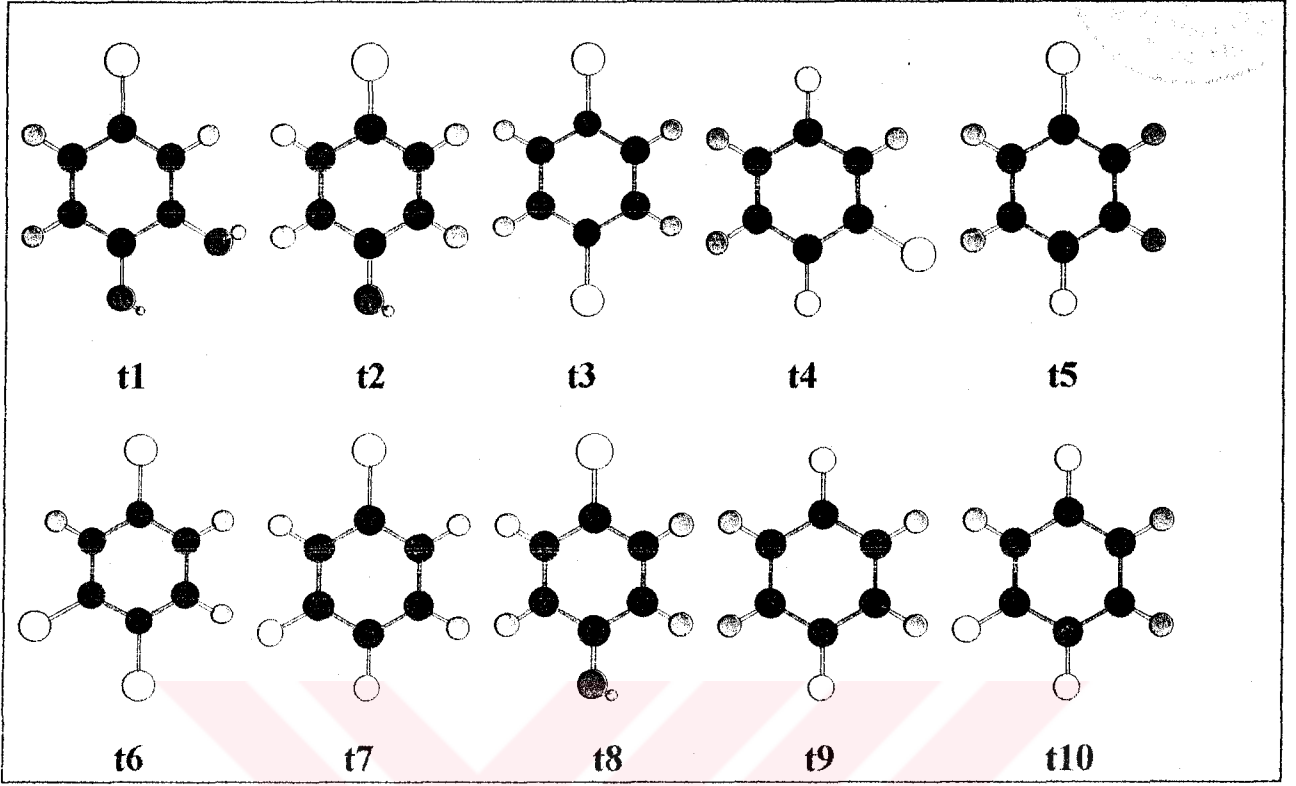
Mehmet POTUR, 1974 yılında Kırklareli’de doğdu. İlk, orta ve lise öğrenimini Kırklareli’de tamamladı. 1999 yılında İTÜ Elektrik-Elektronik Fakültesi, Kontrol ve Bilgisayar Mühendisliği Bölümü’nden mezun oldu. 2000 yılında GYTE Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği yüksek lisans programına başladı. Halen TÜBİTAK Marmara Araştırma Merkezi – Bilişim Teknolojileri Araştırma Enstitüsü – Yazılım Sistemleri Grubu’nda araştırmacı olarak çalışmaktadır.



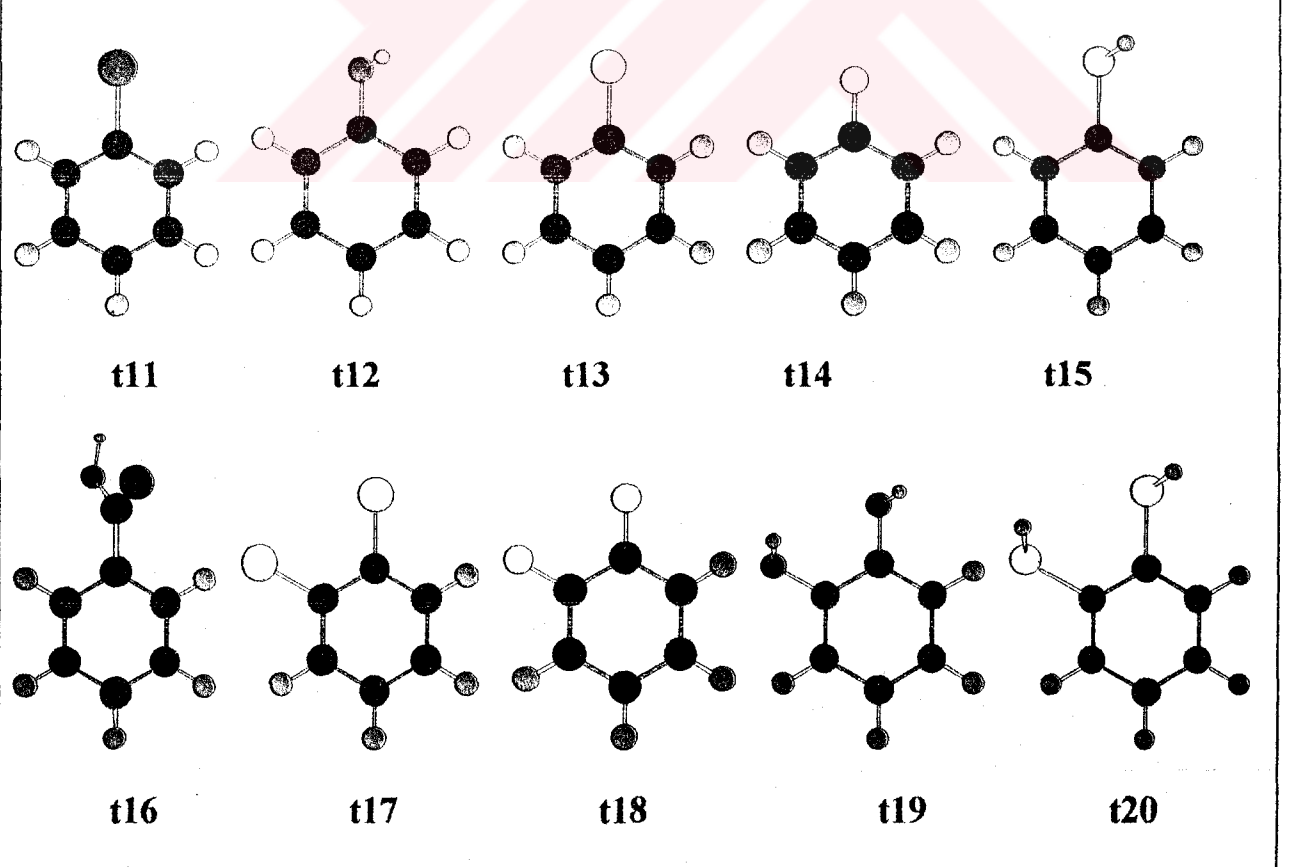


EKLER

EK -1:Aktif Bileşikler



İnaktif Bileşikler:



EK-2: Kuantum Kimyasal Hesaplamaların Çıkış Dosyası



t1

CNDO

RR D

IWRIT1= 1 IWRIT2= 1 IWRIT3= 1 IWRIT4= 1 IWRIT5= 1 IWRIT6= 1

14 ATOMS CHARGE= 0 MULTIPLICITY= 1

ITE= 1

.010000

CARTESIAN COORDINATES

ATOM	AT.NUM.	X	Y	Z
1	6	5.166130	1.406670	-.056180
2	6	4.129440	.555290	-.085230
3	6	4.308210	-.777360	-.063670
4	6	5.562660	-1.270140	-.013040
5	6	6.598270	-.413400	.016710
6	6	6.412480	.914520	-.004470
7	17	4.899170	3.124010	-.084810
8	8	3.232750	-1.604130	-.093480
9	8	5.789870	-2.607530	.008720
10	1	3.099610	.949000	-.127230
11	1	7.626290	-.812720	.058740
12	1	7.285360	1.589800	.020600
13	1	2.341870	-1.220390	-.164040
14	1	6.704370	-2.930820	.079060

MGR= 4 NTB= 0 NDCARB= 0 IEXCITE= 0

P ORBITAL COEFFICIENT 1.00000 D ORBITAL COEFFICIENT 1.00000

ELECTRONIC ENERGY -319.7561000000

ELECTRONIC ENERGY -319.7872000000

ELECTRONIC ENERGY -319.7913000000
ENERGY SATISFIED

TOTAL ENERGY = -99.2769000000 A.U.

BINDING ENERGY= -6.8826590000 A.U.

ELECTRONIC CHARGES

1	C	3.858729
2	C	4.057711
3	C	3.842673
4	C	3.858440
5	C	4.017061
6	C	4.005111
7	CL	7.179521
8	O	6.250509
9	O	6.251446
10	H	.984719
11	H	.999654
12	H	.994526
13	H	.848935
14	H	.850962

POPULATIONS ON ATOMIC ORBITALS

1.002	1.037	.808	1.012
.998	.982	1.012	1.065
.972	.904	.970	.997
.972	1.018	.859	1.010
.994	.966	1.021	1.036
.989	.982	1.004	1.029
1.904	1.980	1.319	1.977
1.616	1.148	1.554	1.932
1.615	1.350	1.353	1.934
.985	1.000	.995	.849
.851			

DIPOLE MOMENTS

COMPONENTS	X	Y	Z
DENSITIES	.524035	-1.126835	.031211
S.P	.770115	.655633	.021316
P.D	.000000	.000000	.000000
TOTAL	1.294150	-.471202	.052526

DIPOLE MOMENT= 1.378265 DEBYES

CHARGES ON ATOMS

.1412706	-.0577106	.1573274	.1415603	-.0170608
-.0051107	-.1795211	-.2505093	-.2514458	.0152814
.0003458	.0054740	.1510651	.1490380	

VALENT ACTIVITIES OF ORBITALS

	.9998183	.9973067	.9617936	.9998487
.9985075	.9979944	.9992745	.9957350	
.9981303	.9830671	.9920072	.9999720	
.9980735	.9977680	.9770418	.9998900	
.9985343	.9972883	.9990892	.9986962	
.9980728	.9977506	.9986991	.9991331	
.1167835	.0281367	.8219348	.0447732	

.4542923	.9750565	.5263917	.1279799
.4545591	.7487047	.7519943	.1248465
.9997660	1.0000000	.9999701	.9771796
.9777877			

VALENT ACTIVITIES OF ATOMS

	3.9587670	3.9915110	3.9731770	3.9727730
3.9936080	3.9936560	1.0116280	2.0837200	
2.0801040	.9997660	1.0000000	.9999701	
.9771796	.9777877			

ELECTRONIC DENSITY RESIDUE ON ATOM

		-.1000378	.0661993	-.1305041	-.1143334
.0234528	.0114551	6.1678930	4.1667890		
4.1713410	-.0150474	-.0003458	-.0054440		
-.1282448	-.1268257				

WIERBERG S INDEXIES

	1.4432590(1 2)	.0084427(1 3)	.1043617(1 _
4)	.0098496(1 5)	1.4443250(1 6)	
	.9070337(1 7)	.0107801(1 8)	.0133410(1 9)
	.0019962(1 10)	.0130069(1 11)	
	.0022375(1 12)	.0000689(1 13)	.0000648(1 14)
	1.4152710(2 3)	.0054448(2 4)	.1053803(2 _
5)	.0050348(2 6)	.0156587(2 7)	
	.0308032(2 8)	.0105203(2 9)	.9445117(2 10)
	.0009288(2 11)	.0113477(2 12)	
	.0028233(2 13)	.0005282(2 14)	
	1.3617420(3 4)	.0069490(3 5)	.1044054(3 _
6)	.0173354(3 7)	1.0111310(3 8)	
	.0245475(3 9)	.0010895(3 10)	.0104286(3 11)
	.0009031(3 12)	.0004150(3 13)	
	.0105174(3 14)		
	1.4214120(4 5)	.0051921(4 6)	.0053975(4 _
7)	.0247506(4 8)	1.0091170(4 9)	
	.0102500(4 10)	.0015113(4 11)	.0127607(4 12)
	.0104712(4 13)	.0003623(4 14)	
	1.4389280(5 6)	.0178171(5 7)	.0109137(5 _
8)	.0301296(5 9)	.0013487(5 10)	
	.9471423(5 11)	.0004011(5 12)	.0005384(5 13)
	.0027988(5 14)		
	.0157123(6 7)	.0129978(6 8)	.0105429(6 _
9)	.0113464(6 10)	.0004539(6 11)	
	.9446498(6 12)	.0000509(6 13)	.0000164(6 14)
	.0019265(7 8)	.0012831(7 9)	.0138697(7 _
10)	.0025036(7 11)	.0128568(7 12)	
	.0002032(7 13)	.0000306(7 14)	
	.0073607(8 9)	.0101448(8 10)	.0017229(8 _
11)	.0003134(8 12)	.9608333(8 13)	
	.0000429(8 14)		

RECEIVED
JUL 10 1964
U.S. DEPARTMENT OF COMMERCE
BUREAU OF STANDARDS

	.0013895(9 _ 10)	.0092932(9 _ 11)	.0009538(9 _
12)	.0000491(9 _ 13)	.9615762(9 _ 14)	
	.0001336(10 _ 11)	.0021242(10 _ 12)	.0011073(10 _
13)	.0004546(10 _ 14)		
	.0112757(11 _ 12)	.0004578(11 _ 13)	.0011415(11 _
14)			
	.0000268(12 _ 13)	.0001197(12 _ 14)	
	.0001344(13 _ 14)		

EIGNVALES HOMO AND LFMO-(ev) -11.8965 3.8151

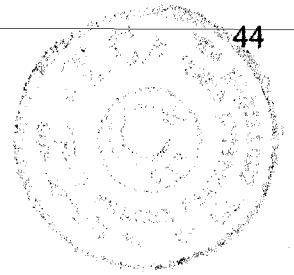
1	1	C	S	.00009	.00017
2	1	C	PX	.01394	.01067
3	1	C	PY	-.00395	-.00298
4	1	C	PZ	-.36679	-.28752
5	2	C	S	.00006	-.00008
6	2	C	PX	.00112	-.02084
7	2	C	PY	-.00068	.00619
8	2	C	PZ	-.03210	.56349
9	3	C	S	.00135	-.00186
10	3	C	PX	-.01182	.01140
11	3	C	PY	.00283	-.00228
12	3	C	PZ	.33365	-.31794
13	4	C	S	-.00128	.00186
14	4	C	PX	-.01538	.00985
15	4	C	PY	.00484	-.00365
16	4	C	PZ	.41614	-.29561
17	5	C	S	-.00021	.00009
18	5	C	PX	-.00502	-.02085
19	5	C	PY	.00196	.00602
20	5	C	PZ	.15486	.56192
21	6	C	S	-.00004	-.00010
22	6	C	PX	.01226	.01037
23	6	C	PY	-.00372	-.00300
24	6	C	PZ	-.33632	-.27775
25	7	CL	S	.00001	-.00007
26	7	CL	PX	-.01645	-.00183
27	7	CL	PY	.00457	.00073
28	7	CL	PZ	.43580	.04843
29	8	O	S	-.00271	.00277
30	8	O	PX	.01579	-.00640
31	8	O	PY	-.00686	.00372
32	8	O	PZ	-.31027	.08810
33	9	O	S	.00313	-.00254
34	9	O	PX	.02173	-.00689
35	9	O	PY	-.00477	.00100
36	9	O	PZ	-.39328	.08194

37	10	H	S	.00011	.00014
38	11	H	S	.00051	-.00014
39	12	H	S	-.00040	.00002
40	13	H	S	.00538	-.00568
41	14	H	S	-.00633	.00523

REACTION CENTERS OF MOLECULE

1	C	.134748
2	C	.001032
3	C	.111474
4	C	.173432
5	C	.024012
6	C	.113272
7	CL	.190214
8	O	.096569
9	O	.155175
10	H	.000000
11	H	.000000
12	H	.000000
13	H	.000029
14	H	.000040
1	C	.082791
2	C	.317999
3	C	.101222
4	C	.087497
5	C	.316228
6	C	.077260
7	CL	.002349
8	O	.007824
9	O	.006770
10	H	.000000
11	H	.000000
12	H	.000000
13	H	.000032
14	H	.000027

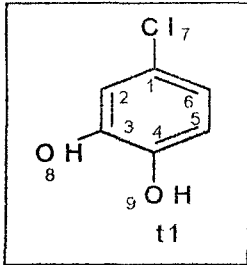
EK- 3: AKTİF MOLEKÜLLER



t1

1.0 9

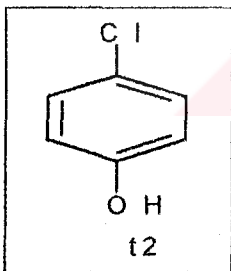
0.141	1.443	2.347	2.706	2.317	1.444	0.907	3.578	4.063	C ₁
-0.058	1.415	2.322	2.654	2.313	2.682	2.338	3.573		C ₂
0.157	1.362	2.320	2.701	3.946	1.011	2.356			C ₃
0.142	1.421	2.344	4.445	2.355	1.009				C ₄
-0.017	1.439	3.926	3.572	2.338					C ₅
-0.005	2.679	4.057	3.577						C ₆
-0.180	5.013	5.801							Cl ₇
-0.251	2.749								O ₈
-0.251									O ₉



t2

1.000 8

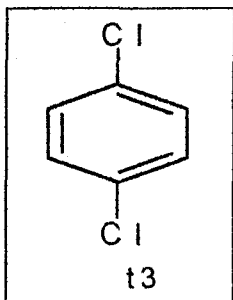
0.116	1.441	2.326	2.717	2.327	1.442	0.906	4.073	
0.014	1.451	2.344	2.671	2.315	2.681	3.587		
-0.025	1.401	2.301	2.670	3.932	2.354			
0.195	1.404	2.343	4.455	1.013				
-0.037	1.451	3.932	2.349					
0.015	2.680	3.583						
-0.181	5.811							
-0.261								



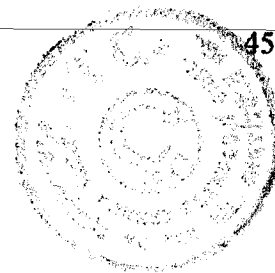
t3

1.000 8

0.133	1.443	2.330	2.698	2.330	1.443	0.913	4.436	
0.006	1.438	2.330	2.680	2.320	2.680	3.933		
0.006	1.443	2.320	2.680	3.933	2.680			
0.133	1.443	2.330	4.436	0.913				
0.006	1.438	3.933	2.680					
0.006	2.680	3.933						
-0.168	6.174							
-0.168								

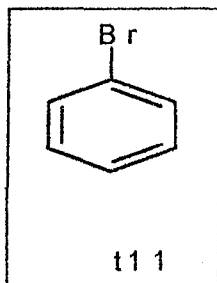


İNAKTİF MOLEKÜLLER



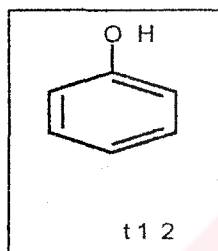
t11

0.000	7						
0.106	1.441	2.329	2.691	2.329	1.441	0.935	
-0.006	1.445	2.325	2.683	2.322	2.822		
	0.031	1.445	2.323	2.683	4.083		
		0.018	1.445	2.326	4.587		
			0.029	1.445	4.083		
				-0.005	2.822		
					-0.131		



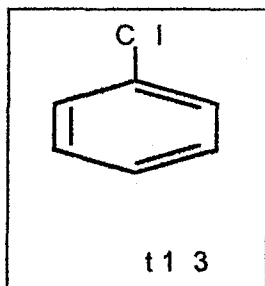
t12

0.000	8						
0.186	1.412	2.342	2.711	2.342	1.412	0.994	2.115
-0.036	1.455	2.323	2.673	2.304	2.350	0.948	
	0.033	1.443	2.318	2.672	3.583	2.116	
		0.004	1.443	2.323	4.065	3.322	
			0.033	1.455	3.583	3.776	
				-0.036	2.350	3.307	
					-0.276	2.588	
						-0.006	



t13

0.000	7						
0.134	1.445	2.329	2.690	2.329	1.445	0.908	
-0.008	1.442	2.325	2.683	2.323	2.679		
	0.032	1.445	2.323	2.683	3.932		
		0.017	1.445	2.325	4.429		
			0.032	1.442	3.932		
				-0.008	2.679		
					-0.177		



EK - 4: GİRİŞ DOSYASI (AKTİF)



t1

14

.1412706 -.0577106 .1573274 .1415603 -.0170608 -.0051107 -.1795211 -.2505093 -.2514458 0152814
.0003458 .0054740 .1510651 .1490380

1.44 0.00 0.00 0.00 1.44 0.91 0.00 0.00 0.00 0.00 0.00 0.00 0.00

1.42 0.00 0.00 0.00 0.00 0.00 0.00 0.94 0.00 0.00 0.00 0.00

1.36 0.00 0.00 0.00 1.01 0.00 0.00 0.00 0.00 0.00 0.00

1.42 0.00 0.00 0.00 1.01 0.00 0.00 0.00 0.00 0.00

1.44 0.00 0.00 0.00 0.00 0.95 0.00 0.00 0.00

0.00 0.00 0.00 0.00 0.00 0.94 0.00 0.00

0.00 0.00 0.00 0.00 0.00 0.00 0.00

0.00 0.00 0.00 0.00 0.96 0.00

0.00 0.00 0.00 0.00 0.96

0.00 0.00 0.00 0.00

0.00 0.00 0.00

0.00 0.00

0.00

1.34 2.35 2.71 2.32 1.34 1.74 3.58 4.06 2.12 3.32 2.13 3.86 4.60

1.34 2.32 2.65 2.31 2.68 2.34 3.57 1.10 3.76 3.32 2.52 4.34

1.35 2.32 2.70 3.95 1.36 2.36 2.11 3.32 3.80 2.02 3.22

1.34 2.34 4.44 2.36 1.36 3.32 2.11 3.34 3.22 2.02

1.34 3.93 3.57 2.34 3.76 1.10 2.12 4.34 2.52

2.68 4.06 3.58 3.32 2.11 1.10 4.60 3.86

5.01 5.80 2.82 4.79 2.84 5.04 6.32

2.75 2.56 4.47 5.16 0.97 3.72

4.46 2.57 4.46 3.72 0.97

4.86 4.24 2.30 5.30

2.43 5.30 2.31

5.69 4.56

4.69

t2

13

.1162853 .0142817 -.0251856 .1948509 -.0366640 .0148520 -.1807132 -.2606478 .1506246 .0068892
.0031888 -.0029012 .0051320

1.44 0.00 0.00 0.00 1.44 0.91 0.00 0.00 0.00 0.00 0.00 0.00

1.45 0.00 0.00 0.00 0.00 0.00 0.00 0.94 0.00 0.00 0.00

1.40 0.00 0.00 0.00 0.00 0.00 0.00 0.95 0.00 0.00

1.40 0.00 0.00 1.01 0.00 0.00 0.00 0.00 0.00

1.45 0.00 0.00 0.00 0.00 0.00 0.95 0.00

0.00 0.00 0.00 0.00 0.00 0.00 0.94

0.00 0.00 0.00 0.00 0.00 0.00

0.96 0.00 0.00 0.00 0.00

0.00 0.00 0.00 0.00

0.00 0.00 0.00

0.00 0.00

0.00

1.34 2.33 2.72 2.33 1.34 1.74 4.07 4.63 2.13 3.32 3.32 2.13

1.34 2.34 2.67 2.32 2.68 3.59 4.36 1.10 2.12 3.77 3.32

1.34 2.30 2.67 3.93 2.35 3.23 2.12 1.10 3.31 3.77

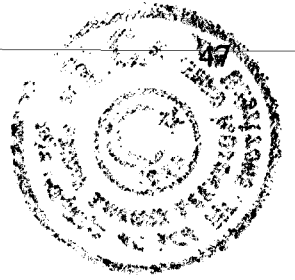
1.34 2.34 4.46 1.36 2.02 3.34 2.11 2.11 3.34

1.34 3.93 2.35 2.55 3.77 3.31 1.10 2.12

2.68 3.58 3.88 3.32 3.77 2.11 1.10

5.81 6.35 2.84 4.80 4.80 2.83

0.97 4.47 2.59 2.59 4.46
5.32 3.56 2.35 4.59
2.43 4.88 4.24
4.22 4.88
2.43



t3

12

.1329350 .0060482 .0060983 .1329386 .0059960 .0060678 -.1679778 -.1679754 .0114714 .0114527
.0114879 .0114658

1.44 0.00 0.00 0.00 1.44 0.91 0.00 0.00 0.00 0.00 0.00
1.44 0.00 0.00 0.00 0.00 0.00 0.94 0.00 0.00 0.00
1.44 0.00 0.00 0.00 0.00 0.00 0.94 0.00 0.00
1.44 0.00 0.00 0.91 0.00 0.00 0.00 0.00
1.44 0.00 0.00 0.00 0.00 0.94 0.00
0.00 0.00 0.00 0.00 0.00 0.94
0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00
0.00 0.00
0.00

1.34 2.33 2.70 2.33 1.34 1.74 4.44 2.13 3.33 3.33 2.13
1.34 2.33 2.68 2.32 2.68 3.93 1.10 2.12 3.78 3.33
1.34 2.32 2.68 3.93 2.68 2.12 1.10 3.33 3.78
1.34 2.33 4.44 1.74 3.33 2.13 2.13 3.33
1.34 3.93 2.68 3.78 3.33 1.10 2.12
2.68 3.93 3.33 3.78 2.12 1.10
6.17 2.83 4.79 4.79 2.83
4.79 2.83 2.83 4.79
2.42 4.89 4.25
4.25 4.89
2.42

GİRİŞ DOSYASI (İNAKTİF)

t11

12

.1061783 -.0058346 .0311604 .0176861 .0292716 -.0053940 -.1311369 -.0021319 -.0122472 -.0137118 -
.0116627 -.0021787

1.44 0.00 0.00 0.00 1.44 0.94 0.00 0.00 0.00 0.00 0.00
1.44 0.00 0.00 0.00 0.00 0.95 0.00 0.00 0.00 0.00
1.44 0.00 0.00 0.00 0.00 0.95 0.00 0.00 0.00
1.44 0.00 0.00 0.00 0.00 0.95 0.00 0.00
1.44 0.00 0.00 0.00 0.00 0.95 0.00
0.00 0.00 0.00 0.00 0.00 0.95
0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00
0.00 0.00
0.00

1.34 2.33 2.69 2.33 1.34 1.90 2.13 3.33 3.80 3.33 2.13
1.34 2.33 2.68 2.32 2.82 1.10 2.12 3.33 3.79 3.33
1.34 2.32 2.68 4.08 2.11 1.10 2.12 3.32 3.79
1.34 2.33 4.59 3.32 2.12 1.10 2.12 3.32
1.34 4.08 3.79 3.32 2.12 1.10 2.11

2.82 3.33 3.79 3.33 2.12 1.10
 2.94 4.94 5.69 4.94 2.94
 2.43 4.23 4.89 4.25
 2.45 4.23 4.89
 2.45 4.23
 2.43

t12

13

.1859319 -.0356832 .0330577 .0044889 .0330608 -.0356712 -.2762218 -.0062103 -.0166680 -.0170766 -
 .0166664 -.0062205 .1538692

1.41 0.00 0.00 0.00 1.41 0.99 0.00 0.00 0.00 0.00 0.00 0.00
 1.46 0.00 0.00 0.00 0.00 0.95 0.00 0.00 0.00 0.00 0.00 0.00
 1.44 0.00 0.00 0.00 0.00 0.95 0.00 0.00 0.00 0.00 0.00
 1.44 0.00 0.00 0.00 0.00 0.95 0.00 0.00 0.00 0.00
 1.46 0.00 0.00 0.00 0.00 0.95 0.00 0.00
 0.00 0.00 0.00 0.00 0.00 0.95 0.00
 0.00 0.00 0.00 0.00 0.00 0.95
 0.00 0.00 0.00 0.00 0.00
 0.00 0.00 0.00 0.00
 0.00 0.00 0.00
 0.00 0.00
 0.00

1.34 2.34 2.71 2.34 1.34 1.35 2.11 3.34 3.81 3.34 2.11 2.02
 1.34 2.32 2.67 2.30 2.35 1.10 2.12 3.33 3.78 3.31 2.90
 1.34 2.32 2.67 3.58 2.12 1.10 2.12 3.32 3.78 4.13
 1.34 2.32 4.07 3.32 2.12 1.10 2.12 3.32 4.62
 1.34 3.58 3.78 3.32 2.12 1.10 2.12 4.13
 2.35 3.31 3.78 3.33 2.12 1.10 2.90
 2.59 4.47 5.17 4.47 2.59 0.97
 2.44 4.23 4.88 4.22 3.01
 2.45 4.23 4.88 4.97
 2.45 4.23 5.71
 2.44 4.97
 3.01

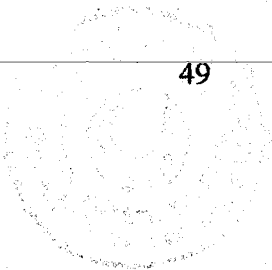
t13

12

.1339092 -.0081921 .0322385 .0165074 .0323465 -.0082474 -.1773043 .0051404 -.0099376 -.0116630 -
 .0099682 .0051610

1.45 0.00 0.00 0.00 1.45 0.91 0.00 0.00 0.00 0.00 0.00
 1.44 0.00 0.00 0.00 0.00 0.94 0.00 0.00 0.00 0.00
 1.45 0.00 0.00 0.00 0.00 0.95 0.00 0.00 0.00
 1.45 0.00 0.00 0.00 0.00 0.95 0.00 0.00
 1.44 0.00 0.00 0.00 0.00 0.95 0.00
 0.00 0.00 0.00 0.00 0.00 0.94
 0.00 0.00 0.00 0.00 0.00
 0.00 0.00 0.00 0.00
 0.00 0.00 0.00
 0.00
 1.34 2.33 2.69 2.33 1.34 1.74 2.13 3.33 3.79 3.33 2.13
 1.34 2.33 2.68 2.32 2.68 1.10 2.12 3.33 3.79 3.33
 1.34 2.32 2.68 3.93 2.11 1.10 2.12 3.33 3.79
 1.34 2.33 4.43 3.32 2.12 1.10 2.12 3.32
 1.34 3.93 3.79 3.33 2.12 1.10 2.11
 2.68 3.33 3.79 3.33 2.12 1.10
 2.83 4.80 5.53 4.80 2.83
 2.44 4.23 4.89 4.25

2.45 4.24 4.89
2.45 4.23
2.44



EK- 5: Kaynak Kodlar

Unit4.h

```
//-----
#ifndef Unit4H
#define Unit4H
//-----
#include <vcl\Classes.hpp>
#include <vcl\Controls.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\ChartFX.hpp>
#include <vcl\OleCtrls.hpp>
//-----
class TForm4 : public TForm
{
__published:      // IDE-managed Components
    TChartFX *ChartFX1;
private:          // User declarations
public:           // User declarations
    __fastcall TForm4(TComponent* Owner);
};
//-----
extern TForm4 *Form4;
//-----
#endif
```

main_calc.h

```
//-----
#ifndef main_calcH
#define main_calcH
//-----
#include <stdio.h>
#include <string.h>
#include <fstream.h>
#include <iostream.h>
#include <stdlib.h>
#include <math.h>
struct Table3{
    int* index;      // interval number
    int* act_val;    // no of active compounds
    int* inact_val;  // no of inactive compounds
};

typedef char string100[101];    //? is 100 enough ?

struct RelSeq{
    int    index;
    float  value;
    int    interval_no;
};

struct Interval{
    int    intrv_index;
    int    intrv_count;
```

```

    bool intrv_chr_func;
};

class MolInfo{
private:
    int mol_size; // atom number of the
compound = N
    int rel_seq_size_ap; // real number of the
relseq ap
    int rel_seq_size_bp; // real number of the
relseq bp
    int rel_seq_size_dist; // real number of the
relseq dist
    void ShowSubMolInfo(RelSeq*, int); // for use of
showmolinfo
    void SortSubRelSqn(RelSeq*, int);
    Interval* CreateIntrvSub (Interval*, RelSeq*, int, int,
float);
public:
    string100 mol_name; // name of the compound
    RelSeq* ap_values; // atomic
properties count is N
    RelSeq* bp_values; // bond
properties count is N(N-1)/2
    RelSeq* dist_values; // distances
count is N(N-1)/2
    Interval* mol_interval_ap; // data structure that
contains interval info
    Interval* mol_interval_bp; // data structure that
contains interval info
    Interval* mol_interval_dist; // data structure that
contains interval info
    float ap_min, ap_max, // atom
bp_min, bp_max, // bond
dist_min, dist_max; // distance
    MolInfo(){} // default
constructor
    MolInfo(const MolInfo &); //
copy constructor
    ~MolInfo(){} // destructor
    void ModifyMemNameSize(string100, int);
    void ModifyRelSeqVal(RelSeq*, int, int, float, int); //
the last one for interval_no ???
    void ShowMolInfo(void);
    void ModifyLastIndex (int, char); //real numbers of relseq
index
    void SortRelSqn(void); //sort relseqs
ap, bp, dist with real numbers
    void CreateIntrv(int, int, int, float, float, float);
// (int number_of_interval_ap, float deltaA) and for bp and distance
    void CreateOutIntrvSub(int, Interval*);
};

class Series{
private:
    string100 act_path_name, inact_path_name; // (paths and
names) for active and inactive files
    int act_num, inact_num; // the number of
active and inactive compounds in the files from CountComps
    MolInfo* active_comps;
// active MolInfo array
    MolInfo* inactive_comps; //
inactive MolInfo array

```

```

        string100  a_prop_name, b_prop_name;
and bond property names
        Interval*  out_intrv_aap;
        // for output file of intervals for all ap's in active mols.
        float      aap_glob_min, aap_glob_max,
atom                                     iap_glob_min, iap_glob_max,
atom                                     abp_glob_min, abp_glob_max,
bond                                     ibp_glob_min, ibp_glob_max,
bond                                     adist_glob_min, adist_glob_max,
                                     idist_glob_min, idist_glob_max;
distance float      deltaA, deltaB, deltaD;
from user int      number_of_interval_aap;
global and fixed number of intervals int      number_of_interval_abp;
for ap's this number is equal to the bigger int      number_of_interval_adist;
side's number (positive or negative) int      number_of_interval_iap;
int      number_of_interval_ibp;
int      number_of_interval_idist;
        int      CountComps (string100 );
        void      ReadComps (string100, bool);
public:
        Series      (string100, string100, float, float, float,
string100, string100); // constructor
        ~Series      () { /*delete active_comps; delete
inactive_comps; */ } // causes error
        void      SortSeries(void);
        void      ShowSeries(void);
        void      ShowMemberData(void);
        void      CreateSeriesIntrv(void);
        void      CreateOutIntrv(void);
};
int*  GetGlobalValues();
Table3  GetTable3Ap();
Table3  GetTable3Bp();
//-----
#endif

```

grafik.h

```

//-----
#ifndef grafikH
#define grafikH
//-----
#include <vcl\Classes.hpp>
#include <vcl\Controls.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\VCFImprs.hpp>
#include <vcl\OleCtrls.hpp>
#include <vcl\ChartFX.hpp>
#include <vcl\Menus.hpp>
#include <vcl\ExtCtrls.hpp>
#include <vcl\Dialogs.hpp>

```

```

#include "main_calc.h"
//-----
class TForm1 : public TForm
{
__published:      // IDE-managed Components
    TButton *Button1;
    TPanel *Panel1;
    TLabel *Label1;
    TPanel *Panel2;
    TPanel *Panel3;
    TPanel *Panel4;
    TPanel *Panel5;
    TPanel *Panel6;
    TLabel *Label2;
    TLabel *Label3;
    TLabel *Label4;
    TLabel *Label5;
    TComboBox *ComboBoxQCM;
    TComboBox *ComboBoxSAR;
    TComboBox *ComboBoxAP;
    TComboBox *ComboBoxBP;
    TLabel *Label6;
    TLabel *Label7;
    TLabel *Label8;
    TLabel *Label9;
    TComboBox *ComboBoxName;
    TComboBox *ComboBoxModel;
    TEdit *EditThreshold;
    TLabel *Label10;
    TLabel *Label11;
    TLabel *Label12;
    TLabel *Label13;
    TEdit *EditDeltaA;
    TEdit *EditDeltaB;
    TEdit *EditDeltaD;
    TLabel *Label14;
    TLabel *Label17;
    TOpenDialog *OpenDialog1;
    TOpenDialog *OpenDialog2;
    TButton *Button2;
    TButton *Button3;
    TEdit *EditAct;
    TEdit *EditInact;
    TComboBox *ComboBoxProMode;
    TLabel *Label15;
    TLabel *Label16;
    TComboBox *ComboBoxGraph;
    void __fastcall Button1Click(TObject *Sender);

private:      // User declarations
public:      // User declarations
    __fastcall TForm1(TComponent* Owner);
};
//-----
extern TForm1 *Form1;
//-----
#endif

```

grafik.cpp

```
//-----
#include <vcl\vcl.h>
#pragma hdrstop

#include "grafik.h"
#include "Unit4.h"

#include <stdio.h>
#include <string.h>
#include <fstream.h>
#include <iostream.h>
#include <stdlib.h>
#include <math.h>
//-----
#pragma resource "*.dfm"

TForm1 *Form1;

//-----
__fastcall TForm1::TForm1(TComponent* Owner)
: TForm(Owner)
{
}
//-----
void DrawGrapA( Table3 table3_ap, int no_of_glob_aap_intrvls,
                int no_of_glob_iap_intrvls){
    char indexn[10];
    int* d1;
    int* d2;
    Form4->ChartFX1->Adm[CSA_MIN]=0;
    Form4->ChartFX1->Adm[CSA_MAX]=10;
    Form4->ChartFX1->Title[CHART_LEFTTIT] = "Number of Molecules";
    Form4->ChartFX1->Title[CHART_BOTTOMTIT] = "Intervals";

    if(no_of_glob_aap_intrvls>no_of_glob_iap_intrvls){
        Form4->ChartFX1->OpenDataEx(COD_VALUES,2,(no_of_glob_aap_intrvls*2));
        d1 = new int[no_of_glob_aap_intrvls*2];
        d2 = new int[no_of_glob_aap_intrvls*2];
        for(int i=0;i<(no_of_glob_aap_intrvls*2);i++){
            itoa((table3_ap.index[i]),indexn,10);
            Form4->ChartFX1->Legend[i] = indexn;
            d1[i]=table3_ap.act_val[i];
            if(((no_of_glob_aap_intrvls-no_of_glob_iap_intrvls)>i)||
                ((no_of_glob_aap_intrvls+no_of_glob_iap_intrvls)<=i)){
                d2[i]=0;
            }
            else{
                d2[i]=table3_ap.inact_val[i-(no_of_glob_aap_intrvls-
no_of_glob_iap_intrvls)];
            }
        }
        Form4->ChartFX1->ThisSerie = 0;
        for (int i=0;i<(no_of_glob_aap_intrvls*2);i++){
            Form4->ChartFX1->Value[i] = d1[i];
        }
        Form4->ChartFX1->ThisSerie = 1;
        for (int i=0;i<(no_of_glob_aap_intrvls*2);i++){
            Form4->ChartFX1->Value[i] = d2[i];
        }
    }
    else{
```



```
Form4->ChartFX1->OpenDataEx(COD_VALUES,2,(no_of_glob_iap_intrvls*2));
    d1 = new int[no_of_glob_iap_intrvls*2];
    d2 = new int[no_of_glob_iap_intrvls*2];
    for(int i=0;i<(no_of_glob_iap_intrvls*2);i++){
        itoa((table3_ap.index[i]),indexn,10);
        Form4->ChartFX1->Legend[i] = indexn;
        d2[i]=table3_ap.inact_val[i];
        if(no_of_glob_aap_intrvls==no_of_glob_iap_intrvls){
            d1[i]=table3_ap.act_val[i];
        }
        if(((no_of_glob_iap_intrvls-no_of_glob_aap_intrvls)>i)||
            ((no_of_glob_aap_intrvls+no_of_glob_iap_intrvls)<=i)){
            d1[i]=0;
        }
        else{
            d1[i]=table3_ap.act_val[i-
(no_of_glob_iap_intrvls-no_of_glob_aap_intrvls)];
        }
    }
    Form4->ChartFX1->ThisSerie = 0;
    for (int i=0;i<(no_of_glob_iap_intrvls*2);i++){
        Form4->ChartFX1->Value[i] = d1[i];
    }
    Form4->ChartFX1->ThisSerie = 1;
    for (int i=0;i<(no_of_glob_iap_intrvls*2);i++){
        Form4->ChartFX1->Value[i] = d2[i];
    }
}

Form4->ChartFX1->SerLeg[0] = "Active Molecules";
Form4->ChartFX1->SerLeg[1] = "Inactive Molecules";
Form4->ChartFX1->CloseData(COD_VALUES);
Form4->Show();
}
//-----
----

void DrawGrapB( Table3 table3_bp, int no_of_glob_abp_intrvls,
                int no_of_glob_ibp_intrvls){
    char indexn[10];
    int* d1;
    int* d2;
    Form4->ChartFX1->Adm[CSA_MIN]=0;
    Form4->ChartFX1->Adm[CSA_MAX]=10;
    Form4->ChartFX1->Title[CHART_LEFTTIT] = "Number of Molecules";
    Form4->ChartFX1->Title[CHART_BOTTOMTIT] = "Intervals";

    if(no_of_glob_abp_intrvls>no_of_glob_ibp_intrvls){
        Form4->ChartFX1->OpenDataEx(COD_VALUES,2,no_of_glob_abp_intrvls);
        d1 = new int[no_of_glob_abp_intrvls];
        d2 = new int[no_of_glob_abp_intrvls];
        for(int i=0;i<no_of_glob_abp_intrvls;i++){
            itoa((table3_bp.index[i]),indexn,10);
            Form4->ChartFX1->Legend[i] = indexn;
            d1[i]=table3_bp.act_val[i];
            if(no_of_glob_ibp_intrvls>i){
                d2[i]=table3_bp.inact_val[i];
            }
            else{
                d2[i]=0;
            }
        }
    }
    Form4->ChartFX1->ThisSerie = 0;
    for (int i=0;i<no_of_glob_abp_intrvls;i++){
```

```

        Form4->ChartFX1->Value[i] = d1[i];
    }
    Form4->ChartFX1->ThisSerie = 1;
    for (int i=0;i<no_of_glob_abp_intrvls;i++){
        Form4->ChartFX1->Value[i] = d2[i];
    }
}
else{
    Form4->ChartFX1->OpenDataEx(COD_VALUES,2,no_of_glob_ibp_intrvls);
    d1 = new int[no_of_glob_ibp_intrvls];
    d2 = new int[no_of_glob_ibp_intrvls];
    for(int i=0;i<no_of_glob_ibp_intrvls;i++){
        itoa((table3_bp.index[i]),indexn,10);
        Form4->ChartFX1->Legend[i] = indexn;
        d2[i]=table3_bp.inact_val[i];
        if(no_of_glob_abp_intrvls==no_of_glob_ibp_intrvls){
            d1[i]=table3_bp.act_val[i];
        }
        if(no_of_glob_abp_intrvls>i){
            d1[i]=table3_bp.act_val[i];
        }
        else{
            d1[i]=0;
        }
    }
    Form4->ChartFX1->ThisSerie = 0;
    for (int i=0;i<no_of_glob_ibp_intrvls;i++){
        Form4->ChartFX1->Value[i] = d1[i];
    }
    Form4->ChartFX1->ThisSerie = 1;
    for (int i=0;i<no_of_glob_ibp_intrvls;i++){
        Form4->ChartFX1->Value[i] = d2[i];
    }
}

Form4->ChartFX1->SerLeg[0] = "Active Molecules";
Form4->ChartFX1->SerLeg[1] = "Inactive Molecules";
Form4->ChartFX1->CloseData(COD_VALUES);
Form4->Show();
}

//-----
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    int    Size = EditDeltaA->GetTextLen();    //Get length of string in
EditDeltaA
    char* temp = new char[++Size];            //Allocate space for
the string
    EditDeltaA->GetTextBuf(temp, Size);        //Puts EditDeltaA->Text
into Buffer
    float delta_a=atof(temp);
    delete    temp;

    Size = EditDeltaB->GetTextLen();    //Get length of string in
EditDeltaB
    temp = new char[++Size];            //Allocate space for the
string
    EditDeltaB->GetTextBuf(temp, Size); //Puts EditDeltaB->Text into Buffer
    float delta_b=atof(temp);
    delete    temp;

    Size = EditDeltaD->GetTextLen();    //Get length of string in
EditDeltaD

```

```

        temp = new char[++Size];                //Allocate space for the
string EditDeltaD->GetTextBuf(temp, Size);        //Puts EditDeltaD->Text
into Buffer
        float delta_d=atof(temp);
        delete temp;

        string100 ap_name = "Atomic charge";    // atomic property name
from outside
        string100 bp_name = "Wieberg's index";  // bond property
name from outside
        string100 act_path_name = "act.txt";    // active file
name from outside
        string100 inact_path_name = "inact.txt"; // inactive file
name from outside

        ifstream actf;
        ifstream inactf;
        actf.open (act_path_name, ios::in );    // opens the file
        inactf.open (inact_path_name, ios::in ); // opens the file
        if (!actf) {cerr<<"Cannot open r-file act!\n";}
        if (!inactf){cerr<<"Cannot open r-file inact!\n";}
        actf.close();                          //
closes the file
        inactf.close();                        //
closes the file

        Series
deneme(act_path_name,inact_path_name,delta_a,delta_b,delta_d,ap_name,bp_name)
;
        deneme.SortSeries();                  // sort the series
        deneme.CreateSeriesIntrv();           //create intervals assign them to relseqs
and show
        deneme.CreateOutIntrv();
        fgetchar();

        int* global_values=GetGlobalValues();
        int no_of_glob_aap_intrvls=global_values[0];
        int no_of_glob_iap_intrvls=global_values[1];
        int no_of_glob_abp_intrvls=global_values[2];
        int no_of_glob_ibp_intrvls=global_values[3];

        Table3 table3_bp=GetTable3Bp();
        Table3 table3_ap=GetTable3Ap();
//      Table3 table3_dist=GetTable3Dist();

        Size = ComboBoxGraph->GetTextLen();    //Get length of
string in ComboBoxGraph
        char *graph_name = new char[++Size];   //Allocate space
for the string
        ComboBoxGraph->GetTextBuf(graph_name, Size);

        if(!strcmp(graph_name,"Table3AtomicProperties")){
                DrawGrapA(table3_ap,no_of_glob_aap_intrvls,no_of_glob_iap_intrvls);
        }
        if(!strcmp(graph_name,"Table3BondProperties")){
                DrawGrapB(table3_bp,no_of_glob_abp_intrvls,no_of_glob_ibp_intrvls);
        }
}
//-----

```

main_calc.cpp

```
#include <main_calc.h>

////////// global files & functions //////////
Table3 table3_ap, table3_bp, table3_dist;
int no_of_glob_aap_intrvls, no_of_glob_iap_intrvls, no_of_glob_abp_intrvls,
    no_of_glob_ibp_intrvls, no_of_glob_adist_intrvls,
no_of_glob_idist_intrvls;
////////// global files & functions //////////

MolInfo::MolInfo(const MolInfo &temp){
    ap_values = new RelSeq[mol_size];
    for (int i= 0; i < mol_size; i++){ap_values[i] = temp.ap_values[i];}
    bp_values = new RelSeq[mol_size*(mol_size-1)/2];
    for (int i= 0; i < mol_size; i++){bp_values[i] = temp.bp_values[i];}
    dist_values = new RelSeq[mol_size*(mol_size-1)/2];
    for (int i= 0; i < mol_size; i++){dist_values[i] =
temp.dist_values[i];}
}

void MolInfo::ModifyMemNameSize(string100 mol_name_out, int mol_size_out){
    strcpy(mol_name, mol_name_out);
    mol_size=mol_size_out;
    int len= (mol_size_out * mol_size_out - mol_size_out)/2;
    ap_values = new RelSeq[mol_size_out];
    bp_values = new RelSeq[len];
    dist_values = new RelSeq[len];
}

void MolInfo::ModifyRelSeqVal(RelSeq* rel_val, int ap_index, int index, float
value, int interval_no){
    rel_val[ap_index].index= index;
    rel_val[ap_index].value= value;
    rel_val[ap_index].interval_no= interval_no;
}

void MolInfo::ShowSubMolInfo(RelSeq* rel_val, int rel_seq_size){
    for(int i=0; i<rel_seq_size; i++){
        cout<<rel_val[i].index<<" ";
        cout<<rel_val[i].value<<" ";
        cout<<rel_val[i].interval_no<<"\n";
    }
}

void MolInfo::ShowMolInfo(void){
    cout<<"mol name="<<mol_name<<" ";
    cout<<"mol size="<<mol_size<<"\n";
    cout<<"atomic properties (index / value / interval_no)="<<"\n";
    ShowSubMolInfo(ap_values, rel_seq_size_ap);
    cout<<"bond properties (index / value / interval_no)="<<"\n";
    ShowSubMolInfo(bp_values, rel_seq_size_bp);
    cout<<"distances (index / value / interval_no)="<<"\n";
    ShowSubMolInfo(dist_values, rel_seq_size_dist);
    fgetchar();
}

void MolInfo::SortSubRelSeqn(RelSeq* rel_val, int rel_seq_size){
    double temp_value;
    int temp_index;
    int temp_inter;
    for (int i=0; i<rel_seq_size-1; i++){ // relseq values
        for (int j=i+1; j<rel_seq_size; j++){
```

```

        if (rel_val[i].value>rel_val[j].value){
            temp_value=rel_val[i].value; rel_val[i].value=rel_val[j].value;
            rel_val[j].value=temp_value; //sort sqn
            temp_index=rel_val[i].index; rel_val[i].index=rel_val[j].index;
            rel_val[j].index=temp_index; //sort its index
            temp_inter=rel_val[i].interval_no;
            rel_val[i].interval_no=rel_val[j].interval_no;
            rel_val[j].interval_no=temp_inter; //sort its interval
        }
    }
}

void MolInfo::SortRelSqn(void){
    SortSubRelSqn(ap_values, rel_seq_size_ap);
    ap_min=ap_values[0].value;
    // local ap minimum
    ap_max=ap_values[rel_seq_size_ap-1].value; // local ap
    maximum
    SortSubRelSqn(bp_values, rel_seq_size_bp);
    bp_min=bp_values[0].value; // local bp
    minimum
    bp_max=bp_values[rel_seq_size_bp-1].value; // local bap maximum
    SortSubRelSqn(dist_values, rel_seq_size_dist);
    dist_min=dist_values[0].value;
    // local dist minimum
    dist_max=dist_values[rel_seq_size_dist-1].value; // local ap
    maximum
}

void MolInfo::ModifyLastIndex(int relseq_index, char selection) {
    //real numbers of relseq index
    if(selection=='A'){rel_seq_size_ap=relseq_index;}
    if(selection=='B'){rel_seq_size_bp=relseq_index;}
    if(selection=='D'){rel_seq_size_dist=relseq_index;}
}

Interval* MolInfo::CreateIntrvSub (Interval* mol_interval, RelSeq* rel_val,
int
number_of_interval, int rel_seq_size, float delta){
    if ((rel_val[0].value)<0){
        mol_interval= new Interval[(number_of_interval*2)];
        int indexl=0;
        for (int i=0;i<(number_of_interval*2+1);i++){
            if((i-number_of_interval)<0){ //
negatif side
                mol_interval[i].intrv_chr_func=false;
                mol_interval[i].intrv_count=0;
                mol_interval[i].intrv_index=((number_of_interval*(-
1))+i);
                while
                (((rel_val[indexl].value)<(delta*((number_of_interval*(-1))+i+1)))
                &&(indexl<rel_seq_size)){
                    mol_interval[i].intrv_count++;

                rel_val[indexl].interval_no=((number_of_interval*(-1))+i); // assigning
the interval no's to relseq
                    indexl++;
                }
                if (mol_interval[i].intrv_count!=0){
                    mol_interval[i].intrv_chr_func=true;
                }
            }
        }
    }
}

```



```

void MolInfo::CreateOutIntrvSub (int number_of_interval, Interval*
mol_interval){

    ofstream table1;
    table1.open("table1.txt", ios::app);
    char *header1 = " intervalno / number of values \n";
    table1.write(header1,strlen(header1));
    for(int j=0;j<number_of_interval;j++){
        char index1[10];
        itoa((mol_interval[j].intrv_index),index1,10);
        table1.write("\t",1);
        table1.write(index1,strlen(index1));
        table1.write("\t\t",2);
        char count1[10];
        itoa((mol_interval[j].intrv_count),count1,10);
        table1.write(count1,strlen(count1));
        table1.write("\n",1);
    }
    table1.write("\n",1);
    table1.close();
}

Series::CountComps(string100 path_name){
    string100 mol_name;
    int index=0;
    int atom_num=0;
    bool end_of_file=false;
    char buffer [11]="\0\0\0\0\0\0\0\0\0\0\0";
    ifstream f;

    f.open (path_name, ios::in ); // opens the file

    while ((f.peek() == ' ')||(f.peek() == '\n'))f.get(); //delete ws and
end of line
    if (f.peek() ==EOF) end_of_file = true; // check the file if it is
empty

    while(!end_of_file){
        while ((f.peek() == ' ')||(f.peek() == '\n')) {f.get();} //skip
ws and endl
        f.get(mol_name,8,'\n');
        while ((f.peek() == ' ')||(f.peek() == '\n')) {f.get();} //skip
ws and endl
        f>>atom_num;
        while ((f.peek() == ' ')||(f.peek() == '\n')) {f.get();} //skip
ws and endl
        //// atomic properties ////
        for (int i=0; i<atom_num; i++){
            f.get(buffer, 10, ' ');
            f.get();
            while ((f.peek() == ' ')||(f.peek() == '\n')) {f.get();}
//skip ws and endl
        }
        while ((f.peek() == ' ')||(f.peek() == '\n')) {f.get();} //skip
ws and endl
        ////init bonds//////////
        int len= (atom_num * atom_num - atom_num)/2;
        for (int i=0; i<len; i++){
            f.get(buffer, 5, ' ');
            while ((f.peek() == ' ')||(f.peek() == '\n')){f.get();}
        }
        ////init distances//////////
        for (int i=0; i<len; i++){
            f.get(buffer, 5, ' ');

```

```

        while ((f.peek() == ' ') || (f.peek() == '\n')) {f.get();}
    }
    index++; // end of a molecule.
    if (f.peek() == EOF) end_of_file = true;
} // end of main while loop
f.close(); // close file
return index; // number of molecules
}

void Series::ReadComps(string100 path_name, bool active){
    string100 mol_name;
    int index=0;
    int atom_num=0;
    bool end_of_file=false;
    double rab;
    MolInfo Temp;
    ifstream f;

    f.open (path_name, ios::in ); // opens the file for input

    while ((f.peek() == ' ') || (f.peek() == '\n')) f.get(); //delete ws and
end of line
    if (f.peek() == EOF) end_of_file = true; // checks the file if it is
empty

    while(!end_of_file){
        while ((f.peek() == ' ') || (f.peek() == '\n')) {f.get();} //skip
ws and endl
        f.get(mol_name,8,'\n');
        while ((f.peek() == ' ') || (f.peek() == '\n')) {f.get();} //skip
ws and endl
        f>>atom_num;
        while ((f.peek() == ' ') || (f.peek() == '\n')) {f.get();} //skip
ws and endl
        Temp.ModifyMemNameSize(mol_name,atom_num);
        ////init atomic properties////////
        int relseq_index=0;rab=0.0; // relseq_index for relseq
arrays
        for (int i=0; i<atom_num; i++){
            char buffer [11]="\0\0\0\0\0\0\0\0\0\0\0";
            f.get(buffer, 10, ' '); //read float string
            rab=atof(buffer);
            if (rab!=0.0){ // only fill non_zero elements

                Temp.ModifyRelSeqVal(Temp.ap_values,relseq_index,relseq_index,rab,0);
                relseq_index++;
            }
            while ((f.peek() == ' ') || (f.peek() == '\n')) {f.get();}
//skip ws and endl
        }
        Temp.ModifyLastIndex(relseq_index,'A'); //real number of ap_values
index
        while ((f.peek() == ' ') || (f.peek() == '\n')) {f.get();} //skip
ws and endl
        int len= (atom_num * atom_num - atom_num)/2;
        ////init bonds////////
        relseq_index=0;rab=0.0; // index for relseq arrays
        for (int i=0; i<len; i++){
            char buffer [11]="\0\0\0\0\0\0\0\0\0\0\0";
            f.get(buffer, 5, ' '); //read float string
            rab=atof(buffer);
            if (rab!=0.0){ // only fill non_zero elements

                Temp.ModifyRelSeqVal(Temp.bp_values,relseq_index,relseq_index,rab,0);

```

```

        relseq_index++;
    }
    while ((f.peek() == ' ') || (f.peek() == '\n')) {f.get();}
}
Temp.ModifyLastIndex(relseq_index, 'B'); //real number of bp_values
index
    relseq_index=0; //rab=0.0; // index for relseq arrays
    ///init distances//////////
    for (int i=0; i<len; i++){
        char buffer [11]="\0\0\0\0\0\0\0\0\0\0\0";
        f.get(buffer, 5, ' '); //read float string
        rab=atof(buffer);
    if (rab!=0.0){ // only fill non_zero
elements
        Temp.ModifyRelSeqVal(Temp.dist_values, relseq_index, relseq_index, rab, 0);
        relseq_index++;
    }
    while ((f.peek() == ' ') || (f.peek() == '\n')) {f.get();}
}
Temp.ModifyLastIndex(relseq_index, 'D'); //real number of dist_values
index
    if (active){ //for selecting active or inactive
        active_comps[index]=Temp;
    }
    else{
        inactive_comps[index]=Temp;
    }
    index++;
    if (f.peek() == EOF) end_of_file = true;
} // end of main while loop
f.close();
}

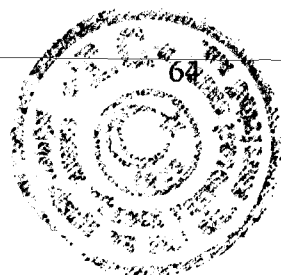
Series::Series( string100 act_path_name_out, string100 inact_path_name_out,
// constructor of Series
                float d1, float d2, float d3,
                string100 a_prop_name_out, string100
b_prop_name_out){

    strcpy(act_path_name, act_path_name_out); // assign the active
file path and name
    strcpy(inact_path_name, inact_path_name_out); // assign the inactive
file path and name
    strcpy(a_prop_name, a_prop_name_out); // assign the
atomic property name
    strcpy(b_prop_name, b_prop_name_out); // assign the bond
property name
    deltaA=d1; deltaB=d2; deltaD=d3; // assign the deltas

    act_num=CountComps(act_path_name); //get the number
of active comp from the specified active file
    cout<<"n_of_active_comp="<<act_num<<"\n";
    inact_num=CountComps(inact_path_name); //get the number of
inactive comp from the specified inactive file
    cout<<"n_of_inactive_comp="<<inact_num<<"\n";

    active_comps = new MolInfo[act_num]; // get memory for
active_comps
    inactive_comps = new MolInfo[inact_num]; // get memory for
inactive_comps
    ReadComps(act_path_name, true); // initialize
active_comps

```



```
ReadComps(inact_path_name, false);           // initialize
inactive_comps
}

void Series::SortSeries(void){
    for(int i=0;i<act_num;i++){
        active_comps[i].SortRelSqn();
    }
    aap_glob_max      =active_comps[0].ap_max;    // finding the global
maxes and mins for active
    abp_glob_max      =active_comps[0].bp_max;
    adist_glob_max     =active_comps[0].dist_max;
    aap_glob_min       =active_comps[0].ap_min;
    abp_glob_min       =active_comps[0].bp_min;
    adist_glob_min     =active_comps[0].dist_min;
    for (int i=1; i<act_num; i++){
        if (aap_glob_max<active_comps[i].ap_max){
            aap_glob_max=active_comps[i].ap_max;
        }
        if (abp_glob_max<active_comps[i].bp_max){
            abp_glob_max=active_comps[i].bp_max;
        }
        if (adist_glob_max<active_comps[i].dist_max){
            adist_glob_max=active_comps[i].dist_max;
        }
        if (aap_glob_min>active_comps[i].ap_min){
            aap_glob_min=active_comps[i].ap_min;
        }
        if (abp_glob_min>active_comps[i].bp_min){
            abp_glob_min=active_comps[i].bp_min;
        }
        if (adist_glob_min>active_comps[i].dist_min){
            adist_glob_min=active_comps[i].dist_min;
        }
    }
    for(int i=0;i<inact_num;i++){
        inactive_comps[i].SortRelSqn();
    }
    iap_glob_max      =inactive_comps[0].ap_max; // finding the global
maxes and mins for inactive
    ibp_glob_max      =inactive_comps[0].bp_max;
    idist_glob_max     =inactive_comps[0].dist_max;
    iap_glob_min       =inactive_comps[0].ap_min;
    ibp_glob_min       =inactive_comps[0].bp_min;
    idist_glob_min     =inactive_comps[0].dist_min;
    for (int i=1; i<inact_num; i++){
        if (iap_glob_max<inactive_comps[i].ap_max){
            iap_glob_max=inactive_comps[i].ap_max;
        }
        if (ibp_glob_max<inactive_comps[i].bp_max){
            ibp_glob_max=inactive_comps[i].bp_max;
        }
        if (idist_glob_max<inactive_comps[i].dist_max){
            idist_glob_max=inactive_comps[i].dist_max;
        }
        if (iap_glob_min>inactive_comps[i].ap_min){
            iap_glob_min=inactive_comps[i].ap_min;
        }
        if (ibp_glob_min>inactive_comps[i].bp_min){
            ibp_glob_min=inactive_comps[i].bp_min;
        }
        if (idist_glob_min>inactive_comps[i].dist_min){
            idist_glob_min=inactive_comps[i].dist_min;
        }
    }
}
```

```

    }
    //double ceil(double x); round up
    // the number of intervals for active aps
    if (aap_glob_max>(aap_glob_min*-
1)) {number_of_interval_aap=int(aap_glob_max/deltaA)+1;}
    else {number_of_interval_aap=int((aap_glob_min/deltaA)*(-1))+1;}
    // the number of intervals for active bps
    number_of_interval_abp =int(abp_glob_max/deltaB)+1;
    // the number of intervals for active dists
    number_of_interval_adist =int(adist_glob_max/deltaD)+1;
    // the number of intervals for inactive aps
    if (iap_glob_max>(iap_glob_min*(-
1))){number_of_interval_iap=int(iap_glob_max/deltaA)+1;}
    else {number_of_interval_iap=int((iap_glob_min/deltaA)*(-1))+1;}
    // the number of intervals for inactive bps
    number_of_interval_ibp =int(ibp_glob_max/deltaB)+1;
    // the number of intervals for inactive dists
    number_of_interval_idist =int(idist_glob_max/deltaD)+1;

    no_of_glob_aap_intrvls=number_of_interval_aap;
    no_of_glob_iap_intrvls=number_of_interval_iap;
    no_of_glob_abp_intrvls=number_of_interval_abp;
    no_of_glob_ibp_intrvls=number_of_interval_ibp;
    no_of_glob_adist_intrvls=number_of_interval_adist;
    no_of_glob_idist_intrvls=number_of_interval_idist;
}

void Series::ShowSeries(void){
    for(int i=0;i<act_num;i++){
        active_comps[i].ShowMolInfo(); //for printout
    }
    for(int i=0;i<inact_num;i++){
        inactive_comps[i].ShowMolInfo(); //for printout
    }
}

void Series::ShowMemberData(void){
    cout<<"a_prop_name="<<a_prop_name<<"\n";
    cout<<"b_prop_name="<<b_prop_name<<"\n";
    cout<<"deltaA="<<deltaA<<" "<<"deltaB="<<deltaB<<"
"<<"deltaD="<<deltaD<<"\n";
    cout<<"active file name="<<act_path_name<<"\n";
    cout<<"number of active molecules="<<act_num<<"\n";
    cout<<"aap_glob_min="<<aap_glob_min<<"
"<<"aap_glob_max="<<aap_glob_max<<"\n";
    cout<<"number_of_interval_aap="<<number_of_interval_aap<<"\n";
    cout<<"abp_glob_min="<<abp_glob_min<<"
"<<"abp_glob_max="<<abp_glob_max<<"\n";
    cout<<"number_of_interval_abp="<<number_of_interval_abp<<"\n";
    cout<<"adist_glob_min="<<adist_glob_min<<"
"<<"adist_glob_max="<<adist_glob_max<<"\n";
    cout<<"number_of_interval_adist="<<number_of_interval_adist<<"\n";
    cout<<"inactive file name="<<inact_path_name<<"\n";
    cout<<"number of inactive molecules="<<inact_num<<"\n";
    cout<<"iap_glob_min="<<iap_glob_min<<"
"<<"iap_glob_max="<<iap_glob_max<<"\n";
    cout<<"number_of_interval_iap="<<number_of_interval_iap<<"\n";
    cout<<"ibp_glob_min="<<ibp_glob_min<<"
"<<"ibp_glob_max="<<ibp_glob_max<<"\n";
    cout<<"number_of_interval_ibp="<<number_of_interval_ibp<<"\n";
    cout<<"idist_glob_min="<<idist_glob_min<<"
"<<"idist_glob_max="<<idist_glob_max<<"\n";
    cout<<"number_of_interval_idist="<<number_of_interval_idist<<"\n";
}

```

```

void Series::CreateSeriesIntrv(void){
    for (int i=0;i<act_num;i++){

        active_comps[i].CreateIntrv(number_of_interval_aap,number_of_interval_a
bp,
        number_of_interval_adist,deltaA,deltaB,deltaD);        //temporary for
intervals
    }
    for (int i=0;i<inact_num;i++){

        inactive_comps[i].CreateIntrv(number_of_interval_iap,number_of_interval
_ibp,
        number_of_interval_idist,deltaA,deltaB,deltaD);        //temporary for
intervals
    }
}

void Series::CreateOutIntrv(void){
    ofstream table1;
    table1.open("table1.txt", ios::out );
    char* header0="\t ACTIVE FILE \n ***** \n";
    table1.write(header0,strlen(header0));
    table1.write("\n",1);
    table1.close();
    for (int i=0;i<act_num;i++){
        table1.open("table1.txt", ios::app );
        char* header1="\t MOLECULE NAME= ";
        table1.write(header1,strlen(header1));

        table1.write((active_comps[i].mol_name),strlen(active_comps[i].mol_name));
        table1.write("\n",1);
        table1.close();
        table1.open("table1.txt", ios::app );
        char* header2="\t ATOMIC PROPERTIES \n";
        table1.write(header2,strlen(header2));
        table1.close();

        active_comps[i].CreateOutIntrvSub(number_of_interval_aap*2,active_comps
[i].mol_interval_ap);
        table1.open("table1.txt", ios::app );
        char* header3="\t BOND PROPERTIES \n";
        table1.write(header3,strlen(header3));
        table1.close();

        active_comps[i].CreateOutIntrvSub(number_of_interval_abp,active_comps[i
].mol_interval_bp);
        table1.open("table1.txt", ios::app );
        char* header4="\t DISTANCES \n";
        table1.write(header4,strlen(header4));
        table1.close();

        active_comps[i].CreateOutIntrvSub(number_of_interval_adist,active_comps
[i].mol_interval_dist);
    }
    table1.open("table1.txt", ios::app );
    char* header5="\t INACTIVE FILE \n ***** \n";
    table1.write(header5,strlen(header5));
    table1.write("\n",1);
    table1.close();
    for (int i=0;i<inact_num;i++){
        table1.open("table1.txt", ios::app );

```

```

        char* header6="\t MOLECULE NAME= ";
        table1.write(header6,strlen(header6));

table1.write((inactive_comps[i].mol_name),strlen(inactive_comps[i].mol_name))
;
    table1.write("\n",1);
    table1.close();
        table1.open("table1.txt", ios::app );
        char* header7="\t ATOMIC PROPERTIES \n";
        table1.write(header7,strlen(header7));
        table1.close();

    inactive_comps[i].CreateOutIntrvSub(number_of_interval_iap*2,inactive_c
omps[i].mol_interval_ap);
        table1.open("table1.txt", ios::app );
        char* header8="\t BOND PROPERTIES \n";
        table1.write(header8,strlen(header8));
        table1.close();

    inactive_comps[i].CreateOutIntrvSub(number_of_interval_ibp,inactive_com
ps[i].mol_interval_bp);
        table1.open("table1.txt", ios::app );
        char* header9="\t DISTANCES \n";
        table1.write(header9,strlen(header9));
        table1.close();

    inactive_comps[i].CreateOutIntrvSub(number_of_interval_idist,inactive_c
omps[i].mol_interval_dist);
    }
/*****table1 to table2*****/
    ofstream table2;
    table2.open("table2.txt", ios::out );
    char* header10="\t ACTIVE FILE ATOMIC PROPERTIES \n
***** \n";
    table2.write(header10,strlen(header10));
    table2.write("\n",1);
    table2.close();
    for (int j=0;j<(number_of_interval_aap*2);j++){
        table2.open("table2.txt", ios::app );
        char index1[10];
        itoa((active_comps[0].mol_interval_ap[j].intrv_index),index1,10);
        table2.write("\t",1);
        table2.write(index1,strlen(index1));
        table2.close();
    }
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    table2.close();
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    for(int i=0;i<act_num;i++){
        for (int j=0;j<(number_of_interval_aap*2);j++){
            char index1[10];
            itoa((active_comps[i].mol_interval_ap[j].intrv_count),index1,10);
            table2.write("\t",1);
            table2.write(index1,strlen(index1));
        }
        table2.write("\t",1);

table2.write((active_comps[i].mol_name),strlen(active_comps[i].mol_name));
        table2.write("\n",1);
    }
    table2.close();
    table2.open("table2.txt", ios::app );

```

```

    table2.write("\n",1);
    char* header11="\t ACTIVE FILE BOND PROPERTIES \n
***** \n";
    table2.write(header11,strlen(header11));
    table2.write("\n",1);
    table2.close();
    for (int j=0;j<number_of_interval_abp;j++){
        table2.open("table2.txt", ios::app );
        char index1[10];
        itoa((active_comps[0].mol_interval_bp[j].intrv_index),index1,10);
        table2.write("\t",1);
        table2.write(index1,strlen(index1));
        table2.close();
    }
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    table2.close();
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    for(int i=0;i<act_num;i++){
        for (int j=0;j<number_of_interval_abp;j++){
            char index1[10];
            itoa((active_comps[i].mol_interval_bp[j].intrv_count),index1,10);
            table2.write("\t",1);
            table2.write(index1,strlen(index1));
        }
        table2.write("\t",1);
    }
    table2.write((active_comps[i].mol_name),strlen(active_comps[i].mol_name));
    table2.write("\n",1);
    }
    table2.close();
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    char* header12="\t ACTIVE FILE DISTANCES \n
***** \n";
    table2.write(header12,strlen(header12));
    table2.write("\n",1);
    table2.close();
    for (int j=0;j<number_of_interval_adist;j++){
        table2.open("table2.txt", ios::app );
        char index1[10];
        itoa((active_comps[0].mol_interval_dist[j].intrv_index),index1,10);
        table2.write("\t",1);
        table2.write(index1,strlen(index1));
        table2.close();
    }
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    table2.close();
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    for(int i=0;i<act_num;i++){
        for (int j=0;j<number_of_interval_adist;j++){
            char index1[10];
            itoa((active_comps[i].mol_interval_dist[j].intrv_count),index1,10);
            table2.write("\t",1);
            table2.write(index1,strlen(index1));
        }
        table2.write("\t",1);
    }
    table2.write((active_comps[i].mol_name),strlen(active_comps[i].mol_name));
    table2.write("\n",1);
    }

```

```

    table2.close();
    //////////**//////////
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    char* header20="\t INACTIVE FILE ATOMIC PROPERTIES \n
*****\n";
    table2.write(header20,strlen(header20));
    table2.write("\n",1);
    table2.close();
    for (int j=0;j<(number_of_interval_iap*2);j++){
        table2.open("table2.txt", ios::app );
        char index1[10];
        itoa((inactive_comps[0].mol_interval_ap[j].intrv_index),index1,10);
        table2.write("\t",1);
        table2.write(index1,strlen(index1));
        table2.close();
    }
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    table2.close();
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    for(int i=0;i<inact_num;i++){
        for (int j=0;j<(number_of_interval_iap*2);j++){
            char index1[10];
            itoa((inactive_comps[i].mol_interval_ap[j].intrv_count),index1,10);
            table2.write("\t",1);
            table2.write(index1,strlen(index1));
        }
        table2.write("\t",1);
    }
    table2.write((inactive_comps[i].mol_name),strlen(inactive_comps[i].mol_name))
;
    table2.write("\n",1);
    }
    table2.close();
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    char* header21="\t INACTIVE FILE BOND PROPERTIES \n
*****\n";
    table2.write(header21,strlen(header21));
    table2.write("\n",1);
    table2.close();
    for (int j=0;j<number_of_interval_ibp;j++){
        table2.open("table2.txt", ios::app );
        char index1[10];
        itoa((inactive_comps[0].mol_interval_bp[j].intrv_index),index1,10);
        table2.write("\t",1);
        table2.write(index1,strlen(index1));
        table2.close();
    }
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    table2.close();
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    for(int i=0;i<inact_num;i++){
        for (int j=0;j<number_of_interval_ibp;j++){
            char index1[10];
            itoa((inactive_comps[i].mol_interval_bp[j].intrv_count),index1,10);
            table2.write("\t",1);
            table2.write(index1,strlen(index1));
        }
        table2.write("\t",1);
    }

```

```

table2.write((inactive_comps[i].mol_name),strlen(inactive_comps[i].mol_name))
;
    table2.write("\n",1);
}
table2.close();
table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    char* header22="\t INACTIVE FILE DISTANCES \n
***** \n";
    table2.write(header22,strlen(header22));
    table2.write("\n",1);
    table2.close();
    for (int j=0;j<number_of_interval_idist;j++){
        table2.open("table2.txt", ios::app );
        char index1[10];
        itoa((inactive_comps[0].mol_interval_dist[j].intrv_index),index1,10);
        table2.write("\t",1);
        table2.write(index1,strlen(index1));
        table2.close();
    }
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    table2.close();
    table2.open("table2.txt", ios::app );
    table2.write("\n",1);
    for(int i=0;i<inact_num;i++){
        for (int j=0;j<number_of_interval_idist;j++){
            char index1[10];
            itoa((inactive_comps[i].mol_interval_dist[j].intrv_count),index1,10);
            table2.write("\t",1);
            table2.write(index1,strlen(index1));
        }
        table2.write("\t",1);

    table2.write((inactive_comps[i].mol_name),strlen(inactive_comps[i].mol_name))
;
        table2.write("\n",1);
    }
    table2.close();
    /**table2 to table3**/**/
    ofstream table3;
    table3.open("table3.txt", ios::out );
    char* header30="\t ACTIVE VERSUS INACTIVE ATOMIC PROPERTIES \n
***** \n";
    table3.write(header30,strlen(header30));
    table3.write("\n",1);
    char* header31="\t ATOMIC PROPERTIES \n";
    table3.write(header31,strlen(header31));
    table3.write("\n",1);
    table3.close();
    Interval* total_chr_func_aap=new Interval[(number_of_interval_aap*2)];
    Interval* total_chr_func_iap=new Interval[(number_of_interval_iap*2)];
    table3.open("table3.txt", ios::app );
    if (number_of_interval_aap>number_of_interval_iap){ //get the bigger to
make same dimension table
        table3_ap.index=new int [(number_of_interval_aap*2)];
        for (int i=0;i<(number_of_interval_aap*2);i++){
            char index1[10];

            itoa((active_comps[0].mol_interval_ap[i].intrv_index),index1,10);
            table3.write("\t",1);
            table3.write(index1,strlen(index1));

```

```

table3_ap.index[i]=active_comps[0].mol_interval_ap[i].intrv_index;
    }
    table3.write("\n",1);
}
else {
    table3_ap.index=new int [(number_of_interval_iap*2)];
    for (int i=0;i<(number_of_interval_iap*2);i++){
        char index1[10];

        itoa((inactive_comps[0].mol_interval_ap[i].intrv_index),index1,10);
        table3.write("\t",1);
        table3.write(index1,strlen(index1));

table3_ap.index[i]=inactive_comps[0].mol_interval_ap[i].intrv_index;
    }
    table3.write("\n",1);
}
table3.close();
    for(int i=0;i<act_num;i++){ // calculate ap chr func
sum for active
        for(int j=0;j<(number_of_interval_aap*2);j++){
            if(active_comps[i].mol_interval_ap[j].intrv_chr_func==1){
                total_chr_func_aap[j].intrv_chr_func++;
            }
        }
        for(int i=0;i<inact_num;i++){ // calculate ap chr func
sum for inactive
            for(int j=0;j<(number_of_interval_iap*2);j++){
                if(inactive_comps[i].mol_interval_ap[j].intrv_chr_func==1){
                    total_chr_func_iap[j].intrv_chr_func++;
                }
            }
            table3.open("table3.txt", ios::app );
            table3_ap.act_val = new int[(number_of_interval_aap*2)];
            table3_ap.inact_val = new int[(number_of_interval_iap*2)];
            for (int i=0;i<(number_of_interval_aap*2);i++){
                char index1[10];
                itoa((total_chr_func_aap[i].intrv_chr_func),index1,10);
                table3.write("\t",1);
                table3.write(index1,strlen(index1));
                table3_ap.act_val[i]=total_chr_func_aap[i].intrv_chr_func;
            }
            char* header32="\t active compounds \n";
            table3.write(header32,strlen(header32));
            for (int i=0;i<(number_of_interval_iap*2);i++){
                char index1[10];
                itoa((total_chr_func_iap[i].intrv_chr_func),index1,10);
                table3.write("\t",1);
                table3.write(index1,strlen(index1));
                table3_ap.inact_val[i]=total_chr_func_iap[i].intrv_chr_func;
            }
            char* header33="\t inactive compounds \n";
            table3.write(header33,strlen(header33));
            table3.write("\n",1);
            table3.close();

            table3.open("table3.txt", ios::app );
            char* header34="\t BOND PROPERTIES \n\n";
            table3.write(header34,strlen(header34));
            Interval* total_chr_func_abp=new Interval[number_of_interval_abp];
            Interval* total_chr_func_ibp=new Interval[number_of_interval_ibp];
            if (number_of_interval_abp>number_of_interval_ibp){ //get the bigger to
make same dimension table

```

72

```

        table3_bp.index=new int [number_of_interval_aap];
        for (int i=0;i<number_of_interval_abp;i++){
            char index1[10];

            itoa((active_comps[0].mol_interval_bp[i].intrv_index),index1,10);
            table3.write("\t",1);
            table3.write(index1,strlen(index1));

        table3_bp.index[i]=active_comps[0].mol_interval_bp[i].intrv_index;
        }
        table3.write("\n",1);
    }
    else
    {
        table3_bp.index=new int [number_of_interval_ibp];
        for (int i=0;i<number_of_interval_ibp;i++){
            char index1[10];

            itoa((inactive_comps[0].mol_interval_bp[i].intrv_index),index1,10);
            table3.write("\t",1);
            table3.write(index1,strlen(index1));

        table3_bp.index[i]=inactive_comps[0].mol_interval_bp[i].intrv_index;
        }
        table3.write("\n",1);
    }
    for(int i=0;i<act_num;i++){
        // calculate bp chr func sum
        for active
        for(int j=0;j<number_of_interval_abp;j++){
            if(active_comps[i].mol_interval_bp[j].intrv_chr_func==1){
                total_chr_func_abp[j].intrv_chr_func++;
            }
        }
        for (int i=0;i<inact_num;i++){
            // calculate bp chr
            func sum for inactive
            for(int j=0;j<number_of_interval_ibp;j++){
                if(inactive_comps[i].mol_interval_bp[j].intrv_chr_func==1){
                    total_chr_func_ibp[j].intrv_chr_func++;
                }
            }
            table3_bp.act_val = new int[number_of_interval_abp];
            table3_bp.inact_val = new int[number_of_interval_ibp];
            for (int i=0;i<number_of_interval_abp;i++){
                char index1[10];
                itoa((total_chr_func_abp[i].intrv_chr_func),index1,10);
                table3.write("\t",1);
                table3.write(index1,strlen(index1));
                table3_bp.act_val[i]=total_chr_func_abp[i].intrv_chr_func;
            }
            char* header35="\t active compounds \n";
            table3.write(header35,strlen(header35));
            for (int i=0;i<number_of_interval_ibp;i++){
                char index1[10];
                itoa((total_chr_func_ibp[i].intrv_chr_func),index1,10);
                table3.write("\t",1);
                table3.write(index1,strlen(index1));
                table3_bp.inact_val[i]=total_chr_func_ibp[i].intrv_chr_func;
            }
            char* header36="\t inactive compounds \n";
            table3.write(header36,strlen(header36));
            table3.write("\n",1);
            table3.close();

            table3.open("table3.txt", ios::app );
            char* header37="\t DISTANCES \n\n";

```

```

table3.write(header37,strlen(header37));
Interval* total_chr_func_adist=new Interval[number_of_interval_adist];
Interval* total_chr_func_idist=new Interval[number_of_interval_idist];
if (number_of_interval_adist>number_of_interval_idist){ //get the
bigger to make same dimension table
    for (int i=0;i<number_of_interval_adist;i++){
        char index1[10];

        itoa((active_comps[0].mol_interval_dist[i].intrv_index),index1,10);
        table3.write("\t",1);
        table3.write(index1,strlen(index1));
    }
    table3.write("\n",1);
}
else {
    for (int i=0;i<number_of_interval_idist;i++){
        char index1[10];

        itoa((inactive_comps[0].mol_interval_dist[i].intrv_index),index1,10);
        table3.write("\t",1);
        table3.write(index1,strlen(index1));
    }
    table3.write("\n",1);
}

for(int i=0;i<act_num;i++){ // calculate dist chr
func sum for active
    for(int j=0;j<number_of_interval_adist;j++){
        if(active_comps[i].mol_interval_dist[j].intrv_chr_func==1){
            total_chr_func_adist[j].intrv_chr_func++;
        }
    }
}

for(int i=0;i<inact_num;i++){ // calculate dist chr
func sum for inactive
    for(int j=0;j<number_of_interval_idist;j++){
        if(inactive_comps[i].mol_interval_dist[j].intrv_chr_func==1){
            total_chr_func_idist[j].intrv_chr_func++;
        }
    }
}

for (int i=0;i<number_of_interval_adist;i++){
    char index1[10];
    itoa((total_chr_func_adist[i].intrv_chr_func),index1,10);
    table3.write("\t",1);
    table3.write(index1,strlen(index1));
}

char* header38="\t active compounds \n";
table3.write(header38,strlen(header38));
for (int i=0;i<number_of_interval_idist;i++){
    char index1[10];
    itoa((total_chr_func_idist[i].intrv_chr_func),index1,10);
    table3.write("\t",1);
    table3.write(index1,strlen(index1));
}

char* header39="\t inactive compounds \n";
table3.write(header39,strlen(header39));
table3.write("\n",1);
table3.close();
}

int* GetGlobalValues(){
    int* globals;
    globals= new int[6];
    globals[0]=no_of_glob_aap_intrvls;
    globals[1]=no_of_glob_iap_intrvls;
    globals[2]=no_of_glob_abp_intrvls;
    globals[3]=no_of_glob_ibp_intrvls;

```



```
globals[4]=no_of_glob_adist_intrvls;
globals[5]=no_of_glob_idist_intrvls;
return(globals);
}
Table3 GetTable3Ap(){
return (table3_ap);
}

Table3 GetTable3Bp(){
return (table3_bp);
}

/*Table3 GetTable3Dist(){
return (table3_dist);
} */
```

PROJE1.cpp

```
//-----
#include <vcl\vcl.h>
#pragma hdrstop
//-----
USEFORM("grafik.cpp", Form1);
USERES("projegrafikmak.res");
USEFORM("\Documents and
Settings\mam002\Desktop\tez_10_13\version_6\Unit4.cpp", Form4);
USEUNIT("\Documents and
Settings\mam002\Desktop\tez_10_13\version_6\main_calc.cpp");
USEFILE("\Documents and
Settings\mam002\Desktop\tez_10_13\version_6\main_calc.h");
USEFILE("\Documents and
Settings\mam002\Desktop\tez_10_13\version_6\Unit4.h");
//-----
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
    try
    {
        Application->Initialize();
        Application->CreateForm(__classid(TForm1), &Form1);
        Application->CreateForm(__classid(TForm4), &Form4);
        Application->Run();
    }
    catch (Exception &exception)
    {
        Application->ShowException(&exception);
    }
    return 0;
}
//-----
```

Unit4.cpp

```
//-----
#include <vcl\vcl.h>
#pragma hdrstop

#include "Unit4.h"
//-----
```

```
#pragma resource "*.dfm"
TForm4 *Form4;
//-----
__fastcall TForm4::TForm4(TComponent* Owner)
    : TForm(Owner)
{
}
//-----
```



Ek- 6: Çıkış Dosyaları

TABLE1.TXT

ACTIVE FILE

MOLECULE NAME= 11

ATOMIC PROPERTIES

intervalno / number of values

-3	2
-2	1
-1	3
1	3
2	5
3	0

BOND PROPERTIES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	6
11	2
12	0
13	0
14	1
15	5

DISTANCES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	2
11	0
12	3
13	0
14	8
15	0
16	0
17	0
18	1
19	0
20	0
21	2
22	6
23	1
24	11

25	1
26	4
27	3
28	3
29	2
30	0
31	0
32	0
33	2
34	6
35	0
36	4
37	0
38	5
39	2
40	2
41	2
42	0
43	1
44	2
45	4
46	3
47	1
48	1
49	1
50	0
51	2
52	1
53	0
54	2
55	0
56	0
57	1
58	0
59	1
60	0
61	0
62	0
63	0
64	1

MOLECULE NAME= 12

ATOMIC PROPERTIES

intervalno / number of values

-3	1
-2	1
-1	3
1	5
2	3
3	0

BOND PROPERTIES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	6

11	1
12	0
13	0
14	2
15	4

DISTANCES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	1
11	0
12	4
13	0
14	7
15	0
16	0
17	0
18	1
19	0
20	0
21	1
22	8
23	1
24	8
25	2
26	3
27	4
28	1
29	2
30	0
31	0
32	0
33	1
34	8
35	0
36	3
37	0
38	4
39	1
40	2
41	1
42	0
43	2
44	1
45	3
46	1
47	1
48	0
49	4
50	0
51	0
52	0
53	0
54	1
55	0

56	0
57	0
58	0
59	1
60	0
61	0
62	0
63	0
64	1

MOLECULE NAME= 13

ATOMIC PROPERTIES

intervalno / number of values

-3	0
-2	2
-1	0
1	8
2	2
3	0

BOND PROPERTIES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	6
11	0
12	0
13	0
14	0
15	6

DISTANCES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	4
13	0
14	6
15	0
16	0
17	0
18	2
19	0
20	0
21	0
22	8
23	0

24	6
25	2
26	0
27	6
28	1
29	4
30	0
31	0
32	0
33	0
34	8
35	0
36	0
37	0
38	4
39	0
40	4
41	0
42	0
43	2
44	0
45	2
46	0
47	0
48	4
49	2
50	0
51	0
52	0
53	0
54	0
55	0
56	0
57	0
58	0
59	0
60	0
61	0
62	1
63	0
64	0

INACTIVE FILE

MOLECULE NAME= 111

ATOMIC PROPERTIES

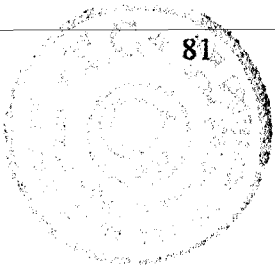
intervalno / number of values

-5	0
-4	0
-3	0
-2	1
-1	7
1	3
2	1
3	0
4	0
5	0

BOND PROPERTIES

intervalno / number of values

1	0
---	---

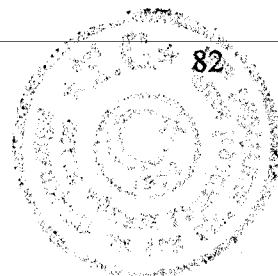


2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	6
11	0
12	0
13	0
14	0
15	6
16	0
17	0
18	0

DISTANCES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	5
13	0
14	6
15	0
16	0
17	0
18	0
19	1
20	0
21	0
22	10
23	0
24	6
25	4
26	0
27	3
28	0
29	2
30	2
31	0
32	0
33	0
34	10
35	0
36	0
37	0
38	5
39	0
40	0
41	2
42	0
43	4



44	0
45	0
46	1
47	0
48	0
49	2
50	2
51	0
52	0
53	0
54	0
55	0
56	0
57	1
58	0
59	0
60	0
61	0
62	0
63	0
64	0
65	0
66	0
67	0
68	0
69	0
70	0

MOLECULE NAME= 112

ATOMIC PROPERTIES

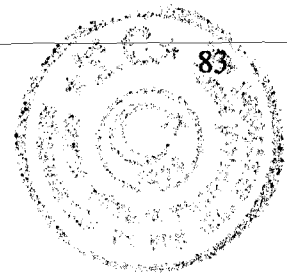
intervalno / number of values

-5	0
-4	0
-3	1
-2	0
-1	7
1	3
2	2
3	0
4	0
5	0

BOND PROPERTIES

intervalno / number of values

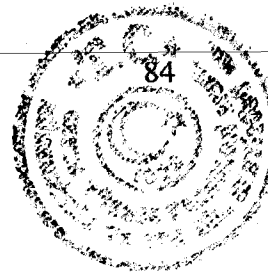
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	7
11	0
12	0
13	0
14	0
15	6
16	0
17	0
18	0



DISTANCES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	1
11	0
12	5
13	0
14	7
15	0
16	0
17	0
18	0
19	0
20	0
21	1
22	10
23	1
24	7
25	4
26	2
27	2
28	1
29	0
30	2
31	2
32	0
33	0
34	10
35	0
36	2
37	0
38	4
39	1
40	0
41	1
42	2
43	4
44	0
45	2
46	0
47	1
48	0
49	2
50	2
51	0
52	1
53	0
54	0
55	0
56	0
57	0
58	1
59	0
60	0
61	0



62	0
63	0
64	0
65	0
66	0
67	0
68	0
69	0
70	0

MOLECULE NAME= t13

ATOMIC PROPERTIES

intervalno / number of values

-5	0
-4	0
-3	0
-2	1
-1	5
1	5
2	1
3	0
4	0
5	0

BOND PROPERTIES

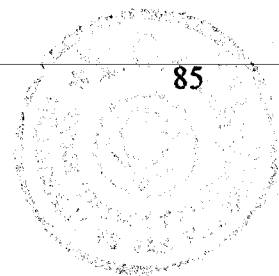
intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	6
11	0
12	0
13	0
14	0
15	6
16	0
17	0
18	0

DISTANCES

intervalno / number of values

1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	5
13	0
14	6
15	0
16	0



17	0
18	1
19	0
20	0
21	0
22	10
23	0
24	6
25	4
26	0
27	5
28	0
29	2
30	0
31	0
32	0
33	0
34	10
35	0
36	0
37	0
38	5
39	0
40	2
41	0
42	0
43	4
44	0
45	1
46	0
47	0
48	0
49	4
50	0
51	0
52	0
53	0
54	0
55	0
56	1
57	0
58	0
59	0
60	0
61	0
62	0
63	0
64	0
65	0
66	0
67	0
68	0
69	0
70	0

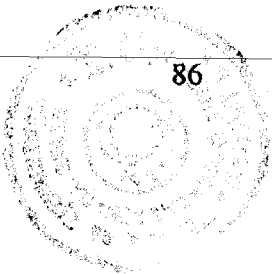


TABLE2.TXT

ACTIVE FILE ATOMIC PROPERTIES

-3	-2	-1	1	2	3	
2	1	3	3	5	0	t1
1	1	3	5	3	0	t2
0	2	0	8	2	0	t3
2	1	3	4	1	1	t4
1	1	2	6	1	1	t5
0	3	1	6	2	0	t6
0	3	2	4	3	0	t7
1	1	2	6	3	0	t8
2	0	4	4	0	2	t9
1	2	2	4	1	2	t10

ACTIVE FILE BOND PROPERTIES

1	2	3	4	5	6	7	8	9	10	11	12
13	14	15									
0	0	0	0	0	0	0	0	0	6	2	0
0	1	5	t1								
0	0	0	0	0	0	0	0	0	6	1	0
0	2	4	t2								
0	0	0	0	0	0	0	0	0	6	0	0
0	0	6	t3								
0	0	0	0	0	0	0	0	0	6	0	0
0	0	6	t4								
0	0	0	0	0	0	0	0	0	6	0	0
0	0	6	t5								
0	0	0	0	0	0	0	0	0	6	0	0
0	0	6	t6								
0	0	0	0	0	0	0	0	0	6	0	0
0	1	5	t7								
0	0	0	0	0	0	0	0	0	7	0	0
0	0	6	t8								
0	0	0	0	0	0	0	0	0	6	0	0
0	0	6	t9								
0	0	0	0	0	0	0	0	0	6	0	0
0	1	5	t10								

ACTIVE FILE DISTANCES

1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64								
0	0	0	0	0	0	0	0	0	2	0	3
0	8	0	0	0	1	0	0	2	6	1	t1
1	4	3	3	2	0	0	0	2	6	0	4
0	5	2	2	2	0	1	2	4	3	1	1
1	0	2	1	0	2	0	0	1	0	1	0
0	0	0	1	t1							

0	0	0	0	0	0	0	0	0	1	0	4
0	7	0	0	0	1	0	0	1	8	1	8
2	3	4	1	2	0	0	0	1	8	0	3
0	4	1	2	1	0	2	1	3	1	1	0
4	0	0	0	0	1	0	0	0	0	1	0
0	0	0	1	t2							
0	0	0	0	0	0	0	0	0	0	0	4
0	6	0	0	0	2	0	0	0	8	0	6
2	0	6	1	4	0	0	0	0	8	0	0
0	4	0	4	0	0	2	0	2	0	0	4
2	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	t3							
0	0	0	0	0	0	0	0	0	0	0	3
0	8	0	0	0	1	0	0	0	6	0	t0
1	3	5	0	1	1	0	0	0	6	0	4
0	3	0	2	2	0	1	0	4	0	0	0
2	1	0	0	0	1	0	1	0	0	0	0
0	0	0	0	t4							
0	0	0	0	0	0	0	0	0	0	0	4
0	7	0	0	0	1	0	0	0	8	0	8
2	2	5	0	2	0	0	0	0	8	0	2
0	4	0	2	1	0	2	0	3	0	0	0
4	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	t5							
0	0	0	0	0	0	0	0	0	0	0	3
0	6	0	0	0	3	0	0	0	6	0	6
1	0	6	5	2	0	0	0	1	6	0	0
0	3	0	6	0	0	1	0	3	0	0	2
3	0	0	0	0	1	0	1	0	0	0	0
0	1	0	0	t6							
0	0	0	0	0	0	0	0	0	0	0	3
0	8	0	0	0	1	0	0	0	6	0	t0
1	2	5	1	2	0	0	0	0	6	0	4
0	3	0	2	2	0	1	0	4	0	0	0
2	1	0	1	0	0	0	0	0	1	0	0
0	0	0	0	t7							
0	0	0	0	0	0	0	0	1	0	0	4
0	7	0	0	0	1	0	0	1	8	1	7
2	2	4	1	2	2	2	0	0	8	0	2
0	4	0	2	1	2	2	0	3	0	1	0
4	2	0	0	0	0	0	0	0	0	1	0
0	0	0	1	t8							
0	0	0	0	0	0	0	0	0	0	0	4
0	8	0	0	0	0	0	0	0	8	0	t0
2	4	3	0	0	0	0	0	0	8	0	4
0	4	0	0	2	0	2	0	4	0	0	0
2	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	t9							
0	0	0	0	0	0	0	0	0	0	0	3
0	9	0	0	0	0	0	0	0	6	0	t2
1	4	3	1	0	0	0	0	0	6	0	6
0	3	0	0	3	0	1	0	4	0	1	0
1	0	0	1	0	1	0	0	0	0	0	0
0	0	0	0	t10							

INACTIVE FILE ATOMIC PROPERTIES

-5	-4	-3	-2	-1	1	2	3	4	5	
0	0	0	1	7	3	1	0	0	0	t11
0	0	1	0	7	3	2	0	0	0	t12
0	0	0	1	5	5	1	0	0	0	t13

t14
t15
t16
t17
t18
t19
t20

林林总总，形形色色，应有尽有，美不胜收。

2	4	0	1	t16					
0	0	0	0	0	0	0	6	0	
0	6	0	0	t17					
0	0	0	0	0	0	0	6	0	
1	5	0	0	t18					
0	0	0	0	0	0	0	7	1	
1	5	0	0	t19					
0	0	0	0	0	0	0	8	0	
0	6	0	0	t20					

ACTIVE FILE DISTANCES

[illegible]

0	0	0	0	0	0	0	0	0	0	0	5
0	7	0	0	0	0	0	0	0	10	0	8
4	2	3	0	0	0	0	0	0	10	0	2
0	5	0	0	1	0	4	0	2	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	t14	
0	0	0	0	0	0	0	0	0	0	0	5
0	7	0	0	0	1	0	0	0	10	0	7
4	0	3	2	2	0	1	2	1	10	0	0
0	5	0	2	0	0	5	1	1	0	0	1
4	0	1	1	0	0	0	1	0	0	1	0
0	0	0	0	0	0	0	0	0	0	t15	
0	0	0	0	0	0	0	0	0	1	0	5
1	7	1	0	0	0	0	1	1	9	1	8
7	1	5	2	1	0	0	0	1	10	1	2
2	6	1	1	0	3	4	1	0	3	0	3
3	2	1	0	1	0	0	1	2	1	1	0
2	0	0	0	1	0	0	0	0	1	t16	
0	0	0	0	0	0	0	0	0	0	0	4
0	6	0	0	0	2	0	0	0	8	0	6
3	0	5	4	0	0	0	0	1	8	0	0
0	4	0	4	0	0	2	0	2	0	0	2
3	0	0	0	0	0	0	2	0	0	0	0
0	0	0	0	0	0	0	0	0	0	t17	
0	0	0	0	0	0	0	0	0	0	0	4
0	8	0	0	0	0	0	0	0	8	0	10
3	2	3	1	0	0	0	0	0	8	0	4
0	4	0	0	2	0	2	0	4	0	0	0
1	0	0	2	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	t18	
0	0	0	0	0	0	0	0	0	2	0	4
0	8	0	0	0	0	0	0	2	8	0	12
3	4	1	3	0	0	0	0	3	8	0	5
1	3	3	0	2	0	2	2	4	3	1	0
1	0	0	2	0	2	0	0	2	0	0	0
0	0	0	0	0	0	0	0	0	0	t19	
0	0	0	0	0	0	0	0	0	0	0	4
0	8	0	0	0	2	0	0	0	8	0	8
3	0	3	5	1	0	2	0	3	10	2	0
0	4	1	4	0	0	4	0	4	0	0	0
7	0	2	0	2	0	0	2	0	0	2	0
0	0	0	0	0	0	0	0	0	0	t20	

TABLE3.TXT

ACTIVE VERSUS INACTIVE ATOMIC PROPERTIES

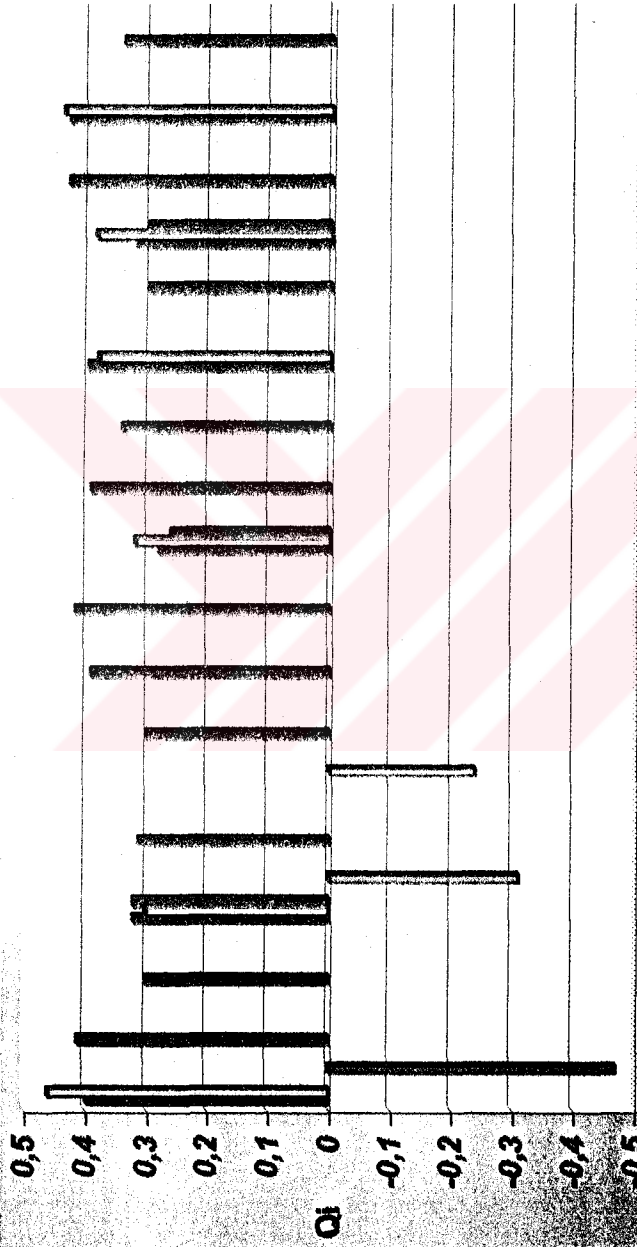
ATOMIC PROPERTIES

-5	-4	-3	-2	-1	1	2	3	4	5		
		7	9	9	10	9	4		active compounds		
0	1	4	6	10	10	7	1	0	1 inactive compounds		

BOND PROPERTIES

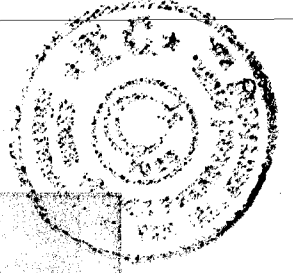
1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18						

EK -7 Yüklele Birleřtirilmiř Sıralı Atomik Uzaklık Tablosu

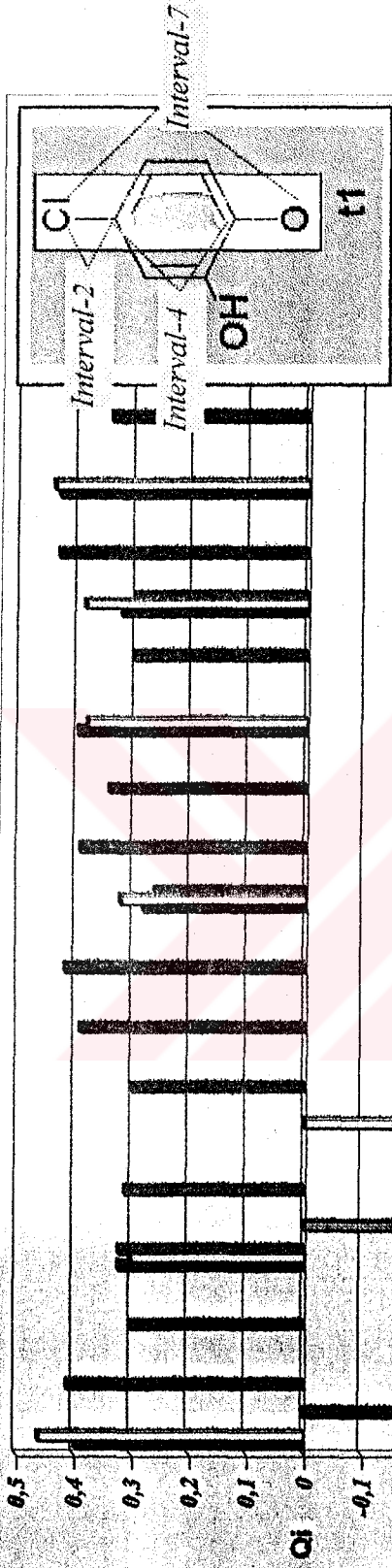


R_{ij}

	1,57	1,58	1,4	1,7	1,72	1,85	2,35	2,35	2,36	2,71	3,58	3,95	4,06	4,44	4,45	5,01	5,8	6,17
11	0,995	0,41	0,3	0,32			0,3	0,59	0,415	0,28	0,39	0,34	0,395		0,32	0,43	0,43	
12	0,46			0,3						0,32			0,38		0,385		0,44	
13				0,32	0,31					0,26				0,3	0,3			0,34
11						-0,24												
12	-0,47																	
13				-0,31														



Yüklerle birleştirilmiş sıralı aralık tablosu



Rij, intervals

	1	2	3	4	5	6	7	8	9	10
1	0,395	0,41	0,3	0,32						
2	0,46		0,3	0,3		0,38	0,385	0,43	0,44	
3		0,32	0,31	0,26			0,3			0,34
4										
5										
6										
7										
8										
9										
10										
11										
12	-0,47									
13		-0,31								

