İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

KİNETİK MONTE CARLO YÖNTEMİ İLE POLİMER SİMÜLASYONLARI

YÜKSEK LİSANS TEZİ
Müh. Kemal Batuhan KISACIKOĞLU
(509001159)

Tezin Enstitüye Verildiği Tarih : 5 Mayıs 2003
Tezin Savunulduğu Tarih : 28 Mayıs 2003

Tez Danışmanı: Prof. Dr. Önder PEKCAN

Diğer Jüri Üyeleri: Doç. Dr. Yaşar YILMAZ

Prof. Dr. Türkan HALİLOĞLU (B.Ü.)

MAYIS 2003

ÖNSÖZ

Bu çalışma daha önce incelenmemiş bir fiziksel olayı aydınlatmaya yönelik bir araştırma değildir. Daha çok araştırmacıların alet çantalarına, yük getirmeden fazladan anahtarlar koymayı hedef alan bir çalışmadır. Ayrıca bu çalışmanın diğer önemli özelliği eğitimde kullanılabilir olmasıdır. Bu yönüyle de çok faydalı olacağına inanıyorum.

Bu çalışmayı yaparken bana konu serbestliğini sağlayan ve katkılarını esprili yaklaşımı ile veren değerli hocam Prof. Dr. Önder Pekcan'a içtenlikle teşekkür ederim.

Polimer simülasyonları konusunda çok faydalandığım sayın hocam Prof. Dr. Ahmet Giz'e de teşekkür ederim.

Ayrıca bu tezi hazırlarken, bana verdikleri hiç bitmeyen sevgilerinden ve sağladıkları moral desteklerinden dolayı aileme de teşekkür ederim.

Mayıs, 2003 Kemal Batuhan Kısacıkoğlu

İÇİNDEKİLER

KİNETİK MONTE CARLO YÖNTEMİ İLE POLİMER SİMÜLASYONLARI

ÖZET

Bilgisayarların icadıyla ilk simülasyonlar yapılmış ve bilgisayar teknolojisindeki hızlı gelişme daha fazla işlem gücü isteyen simülasyonların yapılmasına olanak sağlamıştır. Ancak doğanın inanılmaz hesap gücüne ulaşmamız şu an için imkansız gibi görünmektedir.

Bu çalışmada Monte Carlo yöntemini kullanan polimer simülasyonları için temel bir iskelet oluşturulması hedeflenmiştir. Bu hedef doğrultusunda polimer fiziğinin yapı taşlarının bilgisayar ortamında yaratılmasını sağlayan bir kütüphane hazırlanmıştır. Polimer zincirlerinin ve monomerlerin oluşturulması tamamen kütüphane tarafından yapılır. Yazılan kütüphane molekül hareketlerini kinetik Monte Carlo yöntemi ile hesaplamaktadır. Kütüphanenin sağladığı veri yapıları oldukça esnektir ve kafes yapılar oluşturmak gibi değişik amaçlar için de kullanılabilir. Simülasyon kütüphanesi doğrudan enerji transferi metoduyla, donör ve akseptörlerle işaretlenmiş polimer zincirlerinin difüze ettiği bir sistemde, donör bozunma profillerini de oluşturabilmektedir. Bu profiller sayesinde donör ve akseptörlerle işaretli polimer zincirlerinin ne kadar karıştıkları ölçülebilmektedir. Çalışmanın sağladığı en büyük yarar çeşitli simülasyonların çok hızlı bir şekilde yazılabilmesini sağlamasıdır. Bu doğrultuda kütüphanenin kullanımı ve dayandığı fiziksel ve kimyasal temeller örnek simülasyonlarla anlatılmıştır.

POLYMER SIMULATIONS WITH KINETIC MONTE CARLO METHOD

SUMMARY

First simulations were made just after the invention of computers. The rapid development in computer technology makes more complex simulations possible. But it seems that we will never reach the enormous computation power of nature.

The aim of this work is to develop a framework for polymer simulations that use kinetic Monte Carlo method. In this perspective a library that creates the fundamental elements of polymer physics is produced. This library does the creation of polymer chains and monomers automatically. Library calculates the movements of polymers by kinetic Monte Carlo method. The data structures of library are very flexible and can be used for different purposes such as lattice creation. Time-resolved fluorescence measurements in combination with the direct energy transfer method are used to evaluate the extension of diffusion of dye-labeled reptating polymer chains. The main benefit of library is making the coding process of a polymer simulation faster. The usage of the library, physical and chemical backgrounds is explained with sample simulations.

1. GİRİŞ

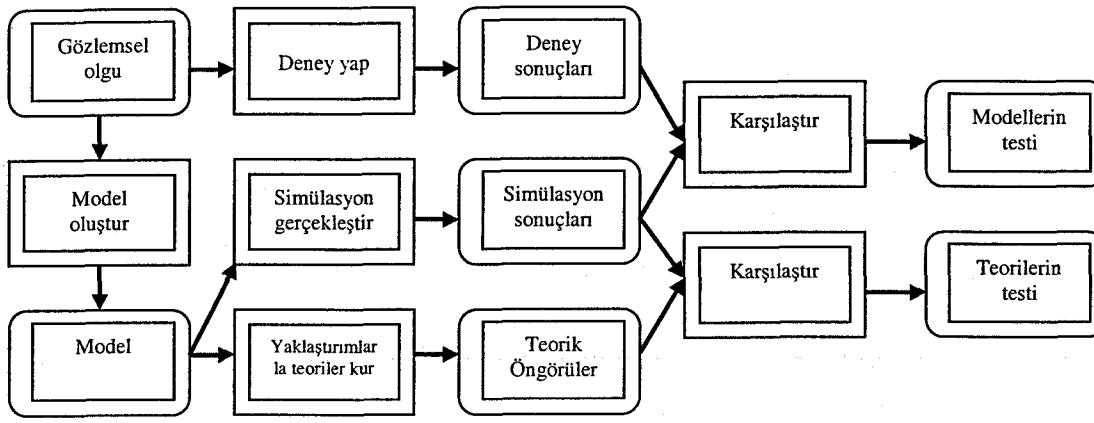
Günümüzde bilgisayarlar bilimsel araştırmaların ayrılmaz parçası haline gelmiştir. Hızlı işlem kapasitesine sahip bu aletler hem deneysel hem de teorik araştırmalarda değişik amaçlar için kullanılmaktadır. Bu kullanım amaçları içinde en önemlisi ise simülasyonlardır. Simülasyonlar teorisyenlerin ve deneycilerin çantalarındaki önemli bir araç haline gelmiştir. Bilgisayar teknolojisinin hızlı gelişimi, bu aracı giderek daha da işlevsel hale getirmektedir. Çok yakın tarihin süper bilgisayarları evlerimize girmekte hatta dizlerimizin üzerinde durmaktadır. Moore yasası[↓] halen geçerliliğini korumakta ve bu hızlı gelişim sürmektedir.

Bilgisayarların gelişimini sadece "işlem hızlarının artması" olarak yorumlamak çok yanlıştır. Bilgisayarların "bağlanabilme" ve "grafik" özellikleri geçtiğimiz 10 yılda tahminlerin ötesinde gelişmiştir. Bağlanabilme değince ilk akla gelen İnternetin gelişimidir. Bunun dışında bilgisayarların diğer çevre birimleriyle bağlantı olanakları da artmıştır. Ancak İnternet çok değişik pencereler açmış, diğer bütün olumlu kullanım amaçlarının dışında bir de bilgisayar simülasyonları için çalışma ortamı haline gelmiştir. "Seti@Home" gibi büyük çaplı projeler ortaya konmuş ağır nümerik hesaplamalar gönüllü İnternet kullanıcılarının bilgisayarlarında dağıtılmış olarak yapılmıştır. Grafik alanındaki gelişme ise büyük çapta eğlence sektörünün ihtiyaçları üzerinde yürümüştür. Çok değil bundan 5 yıl öncesine kadar grafik iş istasyonu olarak adlandırılan çok yüksek fiyatlı bilgisayarlardan daha gelişmiş grafik yeteneklerine sahip bilgisayarlar evlerde çocukların oyun makinelerine dönüşmüştür. Bu kapasitenin bilimsel çalışmalar, özellikle de bilimsel eğitimin görselleştirilmesi için kullanılabileceği aşikardır.

Simülasyonlar temel olarak iki amaca hizmet ederler: Modellerin test edilmesi ve teorilerin test edilmesi. Şekil 1.1'de bu iki durum özetlenmiştir. Simülasyon sonuçları deneysel sonuçlar ile teorik öngörüler arasında yer alır. Deneysel sonuçlardan farkı sizin programlamadığınız bir olguyu size gösterememesidir.

[↓] Intel şirketinin kurucularından Gordon Moore'un 1965'te ortaya koyduğu her sene bilgisayar entegre devrelerindeki transistor sayısının ikiye katlanacağına dair tahmini gözlem.

Ancak, modelinizde sizin öngöremediğiniz bazı durumları ortaya çıkarabilir; bu yönüyle sonuçları beklenmedik olabilir. Teorik öngörülerden farkı ise yaklaştırımlar içermemesidir. Yaklaştırımların yerini bilgisayarın hesap gücü almıştır. Simülasyon sonuçlarının, deneysel sonuçların değerlendirilmesi için kullanılan istatistiksel yöntemlerle incelenmesi gereklidir [1].



Şekil 1.1: Simülasyonların kullanım şeması [2]

Moleküler seviyede simülasyon modeli olarak ilk akla gelenler Moleküler Dinamik ve Monte Carlo teknikleridir. Diğer modellere örnek olarak iz integrali gibi kuantum mekaniksel teknikleri ve hücresel otomata gibi ayrık ve komşu ilişkisine dayalı teknikleri sayabiliriz [1].

2. POLİMERLER

Polimerlerin diğer bir adı makromoleküllerdir. Bu özel moleküller iki ismin çağrıştırdığı iki ayrı özelliğe sahiptirler. Yunanca'da "poli" çok, "mer" ise parça anlamına gelir [3]. Polimerler birçok tekrarlayan parçanın (molekülün, monomerin) birleşmesinden oluşmuşlardır. Tekrarlayan parça monomer olarak adlandırılır. Makromolekül ismi ise bildiğimiz molekül ölçeğinin çok üzerinde büyüklüklere çıkmalarından dolayı verilmiştir. Polimerler genelde organik karbon tabanlı moleküllerdir. Organik polimerlerde çoğunlukla bulunan diğer elementler hidrojen, oksijen, azot, sülfür ve silikondur. Karbon elementi çoğu polimer zincirinin halkaları gibidir; polimerin omurgasını oluşturur.

Bir polimerin molekül ağırlığı aşikardır ki onu oluşturan monomerin molekül ağırlığıyla polimeri oluşturan monomer sayısının çarpımına eşittir. Ancak pratikte bu tanım tek başına pek yaralı değildir. Tek boyda polimer üretecek bir polimerizasyon metodu yoktur. Dolayısıyla polimer örnekleri için ortalama moleküler ağırlık tanımlamak daha kullanışlıdır. Bu çalışmaya daha uygun olacağı için molekül ağırlık ortalamaları, mol kesirleri ve molekül ağırlıkları yerine polimer uzunlukları ve bu uzunluktaki polimer sayıları cinsinden verilmiştir.

$$\bar{M}_n = \frac{\sum_i i \cdot n_i}{\sum_i n_i} \tag{2.1}$$

Denklem (2.1)'de i tane tekrarlayan monomere sahip polimer sayısı n_i olarak tanımlanmıştır. Başka bir değişle monomerlerin molekül ağırlığı "1" alınmıştır. Simülasyon çalışması için böyle bir kabul kolaylık getirecektir. Bu ortalama molekül ağırlığı "sayı ortalaması" olarak adlandırılır. Bir diğer molekül ağırlığı ortalaması "ağırlık" veya "kütle" ortalaması olarak adlandırılır ve tanımı aşağıdaki formülle verilir.

$$\bar{M}_w = \frac{\sum_i i^2 \cdot n_i}{\sum_i i \cdot n_i} \qquad (2.2)$$

Bu iki ortalama molekül ağırlığının birbirine oranı polimer örneğindeki molekül ağırlıklarının dağılımını karakterize eder ve PDI (polydispersity index) olarak adlandırılır.

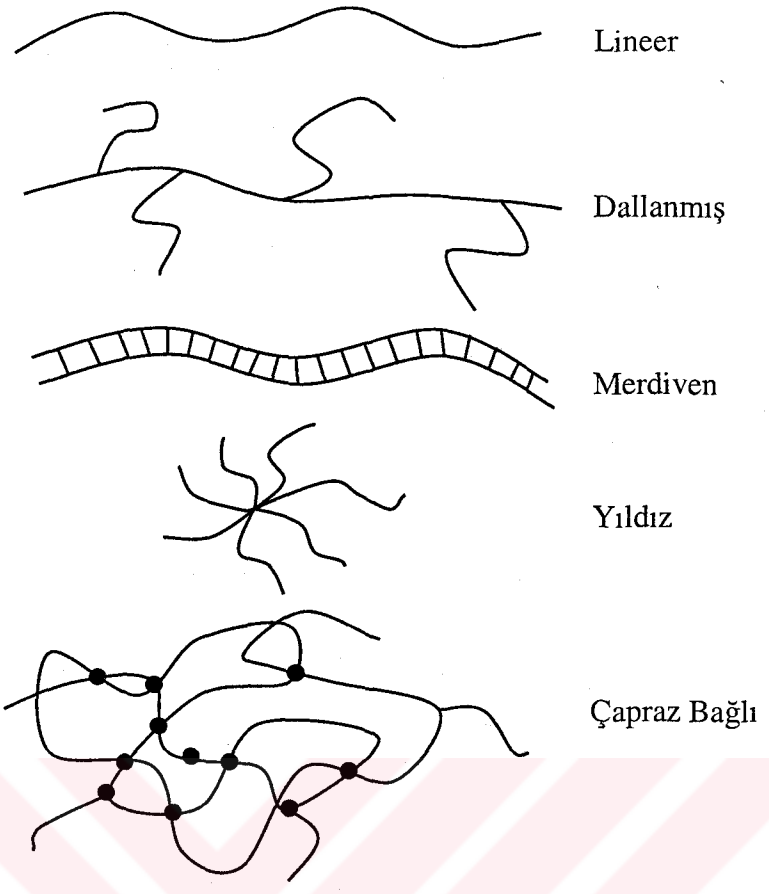
$$PDI = \frac{\bar{M}_w}{\bar{M}_n} \qquad (2.3)$$

2.1. Zincir Mimarisi

Polimerler üç boyutlu yapılardır. Çok değişik geometrik yapılarda polimerler mevcuttur. Bunun dışında aynı yapı taşlarından oluşmuş iki polimer de şekil olarak tamamen farklı olabilir. Bu durumlar aşağıdaki başlıklarda daha derinlemesine irdelenecektir.

2.1.1. Zincir organizasyonu

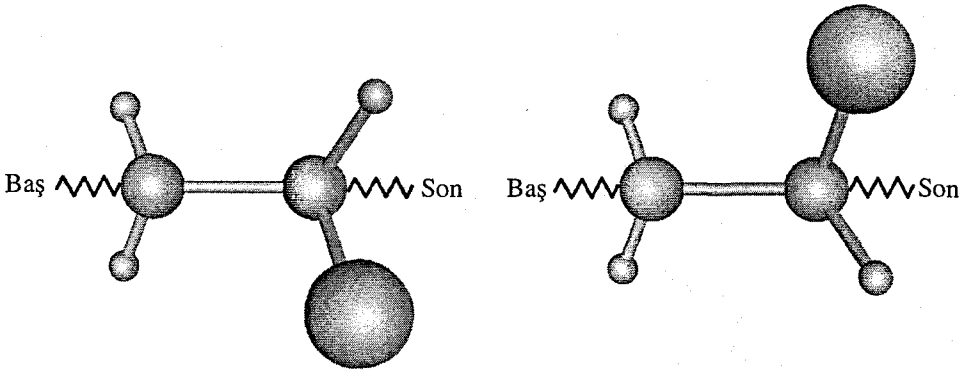
Zincir organizasyonu zincirin bütününün şeklinin nasıl olduğunu tarif eder. Şekil 2.1'de değişik zincir organizasyonları listelenmiştir. Polimerin şekli özelliklerini belirlemesi açısından önemlidir. Örnek olarak merdiven şeklindeki polimerler yüksek dayanıklılığa sahipken çapraz bağlı polimerler kolay erimiş duruma geçmezler.



Şekil 2.1: Polimerlerin bazı zincir organizasyonları

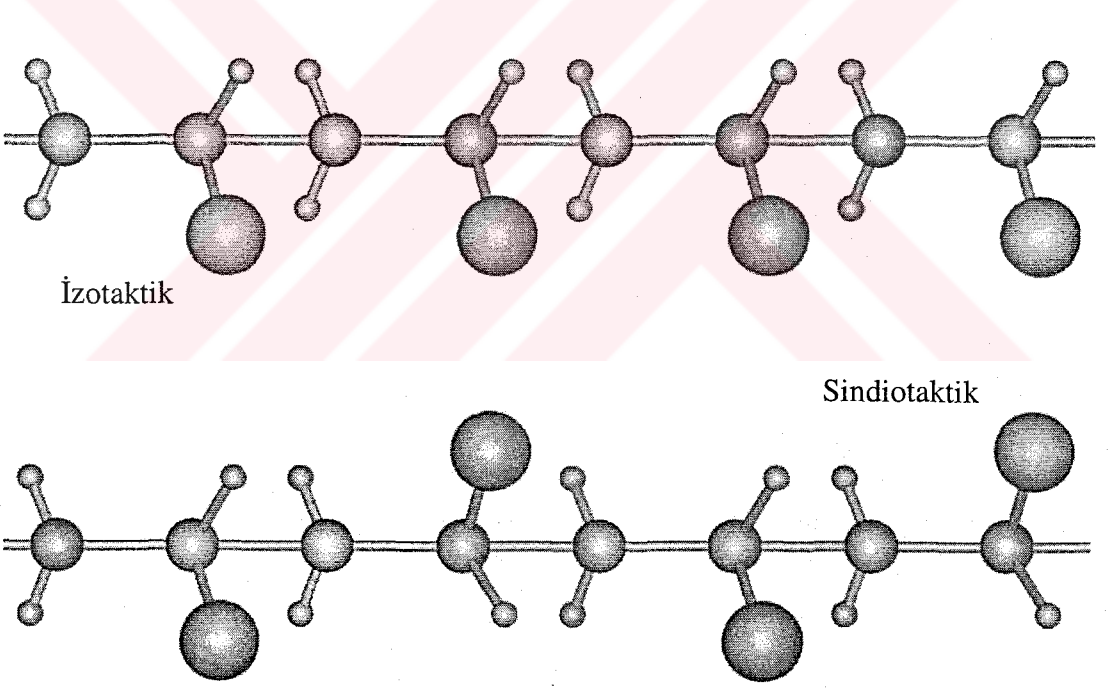
2.1.2. Zincir konfigürasyonu

Zincir konfigürasyonu zincirin değişmez bir özelliğidir ve polimer zincirinin 3 boyutlu yapısından kaynaklanır. Eğer iki polimerdeki monomerlerden bazıları birbirleriyle aynı diğerleri ise birbirlerinin ayna görüntüsü ise iki polimeri döndürerek birbirlerine dönüştürmek imkansızdır. Tabi monomerlerin asimetrik olması gereklidir, aksi takdirde ayna görüntüleri kendileri ile aynı olacaktır.



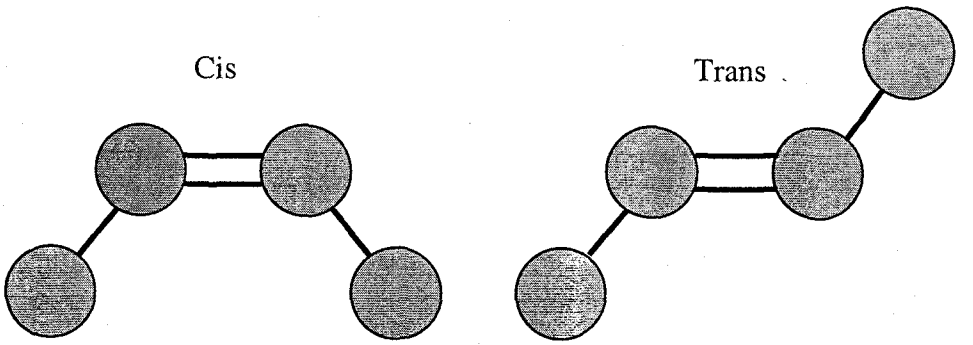
Şekil 2.2: İki farklı konfigürasyonel durumdaki polimer [3]

Şekil 2.2'de görüldüğü üzere monomerlerden birinin farklı konfigürasyonda olması polimeri farklı bir konfigürasyonel duruma sokar. Şekil 2.2'de "baş" ve "son" kısımları aynıdır ve polimerlerden biri diğerine eksensel çevirmelerle veya bağ çevresindeki dönmelerle dönüştüremeyiz. Ancak dikkat edilmesi gereken bir husus vardır. Eğer bir polimerin monomerlerinin hepsi bir konfigürasyonda diğer bir polimerin monomerlerinin hepsi diğer konfigürasyonda ise bu iki polimer başta ve sondaki ufak uyumsuzluk dışında kendi çevrelerinde bütün bir molekül olarak döndürülürse birbirlerine dönüştürülebilirler. İşte böyle bütün monomerleri tek bir konfigürasyonda olan polimerlere "izotaktik" polimerler denir. Monomerlerinin konfigürasyonu arka arkaya sıralı olarak değişen polimerler "sindiotaktik" olarak isimlendirilirler. Son durum değişik konfigürasyondaki monomerlerin polimeri rasgele sırada oluşturmasıdır. Bu tip polimerler ise "ataktik" polimerler olarak adlandırılır. Şekil 2.3 bu durumları özetlemektedir.



Şekil 2.3: İzotaktik ve sindiotaktik polimerler [3]

Bir polimerin iki ayrı konfigürasyonda olmasının bir diğer yolu polimerin çift bağa sahip olmasıdır. İki element arasındaki çift bağ polimerin geri kalanının bu bağ etrafında dönmesini engeller. Şekil 2.4'te polimerin iki ayrı konfigürasyonel şekli görülmektedir. Bu iki farklı durum "cis" ve "trans" olarak adlandırılır. "cis" ve "trans" durumları birbirlerine dönüştürülemez.

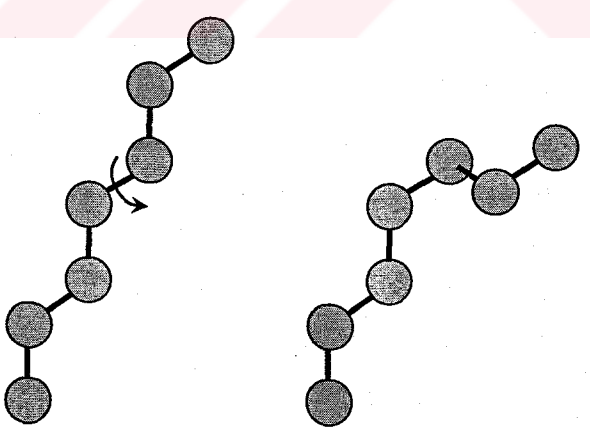


Şekil 2.4: Çift bağ ile oluşan "cis" ve "trans" durumları

Konfigürasyonel durum, polimerin düşük sıcaklıklarda nasıl davranacağını belirler. Düzenli olmayan bir konfigürasyona sahip (ataktik gibi) bir polimer düşük sıcaklıklarda kristalize olmaz ve camsı bir yapıya dönüşür. Ancak düzenli bir yapıya sahip polimer (sindiotaktik gibi) düşük sıcaklıklarda kristalize olup yarı-kristal bir yapıya dönüşebilir.

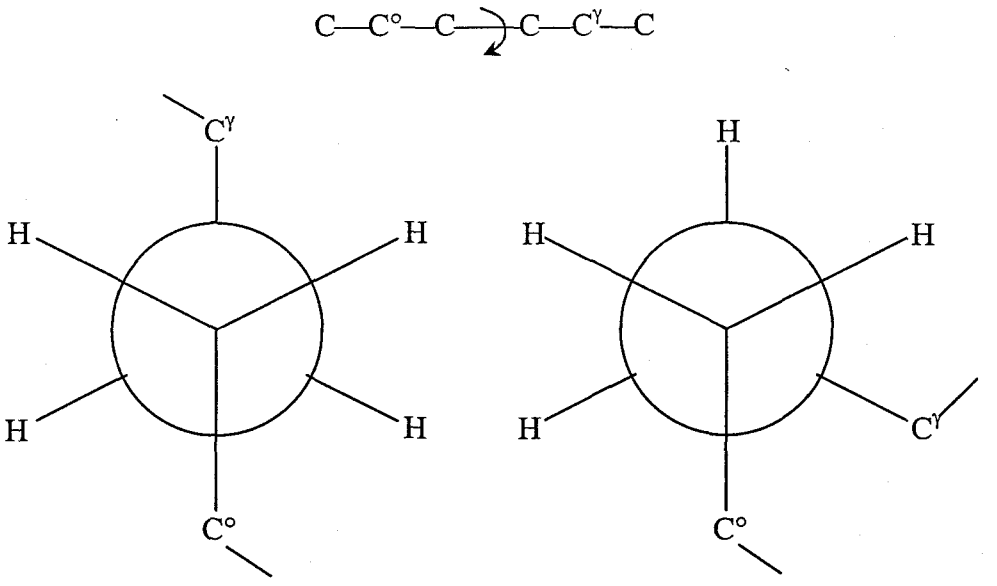
2.1.3. Zincir konformasyonu

Polimerin değişik zincir konformasyonları, zincirin omurgasındaki tek bağların etrafında geri kalan kısmının döndürülmesi ile elde edilir. Şekil 2.5'te iki farklı zincir konformasyonu gösterilmektedir. Bu iki durum birbirine dönüştürülebilir. Bu aradaki bağın tek bağ olmasından kaynaklanır. Çift ve üçlü bağlar bu tür dönmelere izin vermezler.



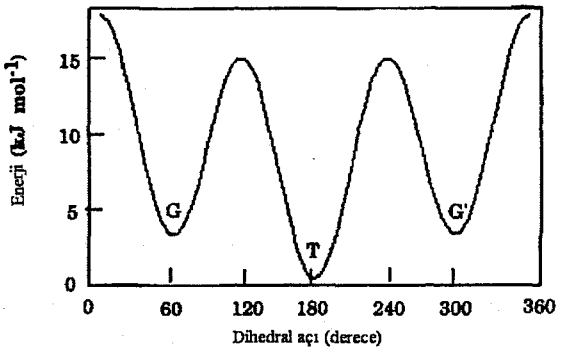
Şekil 2.5: Polimer zincirinde konformasyonel değişim

Tek bağlar etrafında dönme her açıda olmaz. Bazı açılar polimerin tercih ettiği minimum enerji durumlarına karşılık gelir. Newman projeksiyonları bu durumu daha anlaşılır kılmaktadır. Şekil 2.6'da n-alkan zincirinin Newman projeksiyonu görülmektedir [4].



Şekil 2.6: n-alkan zincirinin Newman projeksiyonu ilk durumda $\phi = 180°$ ikinci durumda $\phi = 300°$'dir.

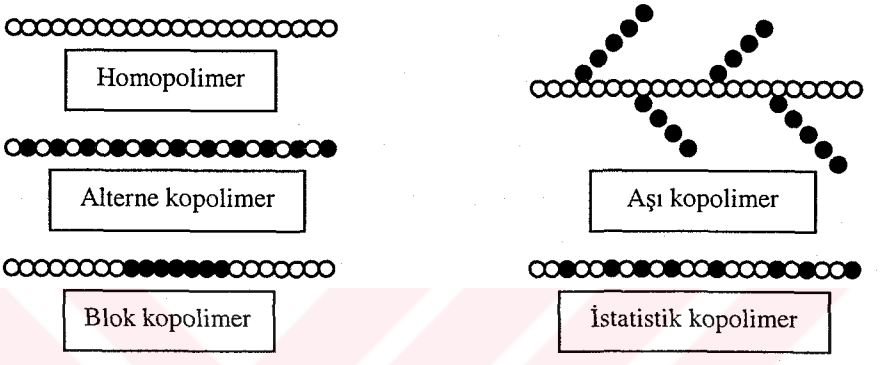
Şekil 2.6'te görüleceği üzere C° ve C^{γ} üst üste geldiğinde $\phi = 0°$ değerini alır. Şekilden görülebileceği gibi iki karbon atomu birbirine en yakın konumdadır ve durum kararsızdır. Üç kararlı durum ϕ açısının 60°, 180° ve 300° değerlerini aldığı hallerdir. Bu üç durum sırasıyla G (gauche), T (trans) ve G′(gauche) olarak adlandırılır. Bazı kaynaklar T durumunda ϕ açısını 0° olarak almaktadır bu durumda iki G durumu G^- ve G^+ olarak adlandırılır, ϕ açısı ise -180° ile +180° arasında değer alır. Şekil 2.7'de n-bütanın ϕ açısına karşı konformasyon enerjisinin grafiği verilmiştir [3].



Şekil 2.7: n-bütanın dihedral açısına (ϕ) karşı konformasyon enerjisi grafiği. Ulf Gedde'nin Polymer Physics [3] kitabından alınmıştır.

2.1.4. Homopolimerler ve kopolimerler

Homopolimer sadece bir çeşit monomerden müteşekkil polimerlere verilen isimdir. Kopolimerler ise birden fazla çeşitte monomerden oluşur. Bu farklı monomerler değişik şekillerde sıralanabilirler. Polimerler, eğer değişik monomerleri, bloklar halinde bulunuyorsa, blok kopolimer, ardışık olarak değişiyorsa, alterne kopolimer, rasgele dağılmışlarsa, istatistik kopolimer, olarak adlandırılırlar. Aşı kopolimer de ise bir monomer dizisinden diğer bir monomer dizisi dallanmıştır. Şekil 2.8 kopolimer çeşitlerini özetlemektedir.



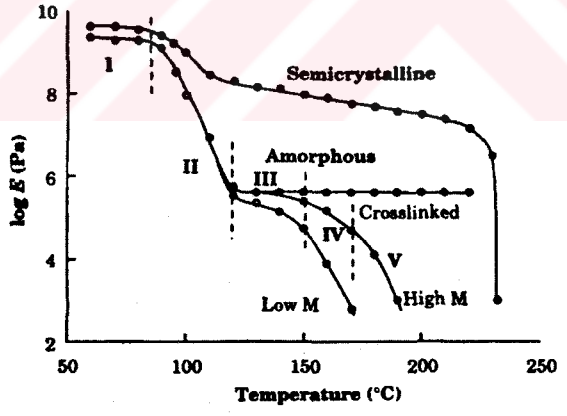
Şekil 2.8: Kopolimer çeşitleri

2.2. Polimerlerin Fazları

Genel olarak polimerler iki gruba ayrılabilir: tamamen amorf ve yarı kristal. Daha önceki bölümlerde de bahsedildiği üzere konfigürasyonel olarak düzenli olmayan polimerler düşük sıcaklıklarda kristalize olmazlar. Bunun yerine amorf cam bir yapıya dönüşürler. Amorf cam fazdaki bir polimer rijit ve kırılgan yapıdadır. Bunun nedeni konformasyonel değişimleri gerçekleştirmesi için gerekli enerjiye sahip olmamasıdır. Camsı polimer ısıtılırsa yavaş yavaş lastik benzeri bir faza geçiş yapar. Bu fazdaki polimerler, üzerilerine sürekli bir kuvvet uygulanırsa (yerçekimi gibi) tersinmez bir şekilde akarlar. T_g olarak tanımlanan cam geçiş sıcaklığı bu geçişin kritik noktasıdır. Bu sıcaklığın biraz üzerinde polimer camsı iken altında sert ama derimsidir. Burada dikkat edilmesi gereken husus T_g kritik sıcaklığının hem altında hem de üstünde polimer zincirlerinin düzenli olmayışıdır. Bu sebepten dolayı T_g kritik sıcaklığı, katı-sıvı faz geçişi gibi karakteristik bir faz geçişini işaret etmez. Polimer, tamamen lastiğimsi hale geçtikten sonra ısıtılmaya devam edilirse, belirli bir sıcaklığa kadar lastiğimsi özelliklerini korur sonra tamamen sıvı faza geçer. Ancak çapraz bağlı polimerler sıvı faza geçmezler. Bunun sebebi çapraz bağların

zincirlerin birbirleri üzerinde kaymasını engellemesidir. Çapraz bağlı polimerlerin üstüne sabit bir kuvvet uygulanırsa, polimer esner; kuvvet çekildiği zaman polimer eski şekline geri döner.

Yarı kristal durumdaki polimerler ısıtıldıkça daha farklı davranış gösterirler. İlk önce yarı kristal durumdan bahsetmek yararlı olacaktır. Kristalleşebilen polimerler konfigürasyonel yapıları düzgün olan polimerlerdir. Kristalleşme tabakalar halinde olur ancak bu tabakalar arasında yine düzensiz yapılar oluşur. Kristal tabakalar polimere sağlamlık katarlar. Bu özellikleri dolayısıyla ticari kullanımları (Kevlar ve Spectra fiberleri gibi) oldukça yaygındır. Isıtıldıkça ara tabakalardaki amorf yapılar konformasyonlarını değiştirebilme özelliği kazansa da, kristal kısımlar çapa etkisi göstererek polimerin akışkan hale geçmesini önlerler. Bu yüzden T_g kritik sıcaklığının üstünde de amorf polimerlerden farklı olarak plastik deformasyona uğramazlar. Kristalize polimerler için daha karakteristik bir faz geçişi T_m sıcaklığında olur. Bu sıcaklık erime noktası olarak adlandırılır. Bu sıcaklıkta düzenli polimer yapısı tamamen düzensiz ve akışkan yapıya dönüşür. Değişik erime sıcaklıklarına sahip polimerler değişik ticari amaçlar için kullanılır. Şekil 2.9'deki grafik anlatılanları özetlemektedir.



Şekil 2.9: Elastisite modülünün logaritmasının sıcaklığa karşı grafiği. Ulf Gedde'nin Polymer Physics kitabından alınmıştır [3].

Her ne kadar değişik bir fazı ifade etmese de polimerler çözeltilerine de değinmek gerekir. Polimerler çeşitli çözücüler içerisinde çözelti durumunda olabilirler. İyi bir çözücü polimerin açılmasına yol açarken kötü bir çözücü polimerin büzülmesine yol açacaktır. Bu durum polimerdeki moleküllerin çözücü moleküllere yakın olmayı veya kendi zincirine yakın olmayı tercih etmesinden kaynaklanır. Eğer çözücü

moleküllere yakın olmayı tercih ediyorsa bağlar dönerek daha düzenli bir hale geçerler ve polimerin açılmasına yol açarlar. Diğer durumda bağlar düzensiz konformasyonel şekiller alırlar bu da polimerin büzülmesine yol açacaktır. İki durumunda tercih edilmediği özel çözeltiler mevcuttur. Böyle çözeltileri oluşturan kimyasallara "teta" çözücüleri denir. Bu tür çözeltide bulunan polimere ise "teta" durumunda denir.

2.3. Zincir Baş - Son Uzaklığı

Zincir baş – son uzaklığı r polimerin şeklini karakterize etmekte oldukça kullanışlı bir parametredir. Ancak r çevresel etkilere ve sıcaklığa oldukça sıkı bir şekilde bağlıdır. Daha önceki bölümde bahsedildiği üzere kötü bir çözücü içindeki polimer katlanarak ufalır, buna mukabil iyi bir çözücü içindeki polimer ise açılacaktır. Bu aynı polimerin değişik r değerleri alması demektir. Zincir baş-son uzaklığının ifadesinin yazılabilmesi için çeşitli kabullerin yapılması gereklidir. Bunlardan ilki polimerin daha evvelki bölümde bahsedildiği üzere "teta" çözücüleri kullanılarak hazırlanan çözeltide bulunmasıdır. İkinci kabul zincirin rasgele bir konformasyonda bulunduğudur yani "trans" ve "gauche" durumları rasgele dağılmışlardır. Bu tür zincirlere "rasgele zincir" veya "Gaussien zincir" denir. Üçüncü kabul ise sadece kısa erişimli etkilerin göz önüne alındığıdır. Bu kabul, zincirin dönerek kendini kesebileceği yani zincirin hayalet bir zincirmiş gibi davranacağı manasına gelir. Bu şartlara uyan tüm polimerler için $\langle r^2 \rangle$ aşağıdaki ifade ile verilir.

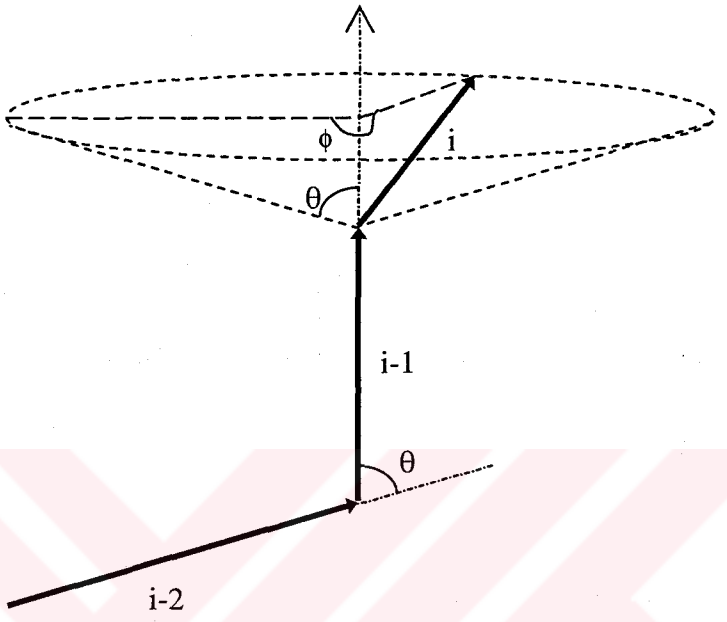
$$\langle r^2 \rangle_o = Cnl^2 \tag{2.4}$$

Denklem (2.4)'te alt indis "teta" durumunu ifade etmektedir. C polimerin cinsine bağlı bir parametredir; n monomer sayısı, l ise bir bağ uzunluğudur. Flory erimiş durumdaki polimerlerin de yukarıdaki eşitliğe uyduğunu önermiştir ve yıllar sonra bu öneri deneysel verilerle ispatlanmıştır [3]. Uzun erimli etkileşimler (zincirin dönerek kendini kesmesi gibi) zincirin biraz daha açılmasına yol açar. Bu durumda düzeltilmiş $\langle r^2 \rangle$ aşağıdaki gibi verilir.

$$\langle r^2 \rangle = \alpha^2 \langle r^2 \rangle_o \tag{2.5}$$

α doğrusal genişleme faktörü olarak adlandırılır. Bu faktörün aynı zamanda sıcaklığa ve çözücü tipine de bağlı olduğu hem deneysel hem de teorik olarak ispatlanmıştır [3].

Denklem (2.4)'teki C parametresi çeşitli zincir modelleri için hesaplanmıştır. Şekil 2.10 üzerinden bu değişik modellerin tasviri yapılabilir.



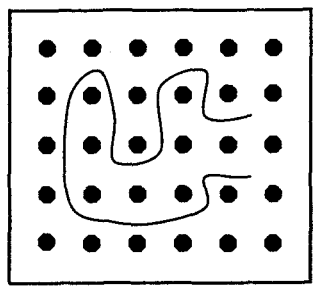
Şekil 2.10: Zincir üzerinde θ ve ϕ açılarının gösterimi

İlk model serbest bağlı zincirdir. Bu model için $C = 1$ değerini alır. Bu modelde ardışık bağlar için θ ve ϕ açıları tamamen rasgeledir. İkinci model serbest dönebilen zincir olarak adlandırılır. Bu model için θ açısı tüm zincir için sabit iken ϕ açısı rasgele değerler alır. Serbest dönebilen zincir için hesaplanan C değeri 2'ye eşittir. Son modelde ϕ açısı sadece belirli değerler alır. Bu model için $C > 2$ olarak hesaplanır.

2.4. de Gennes Sürüngen Modeli

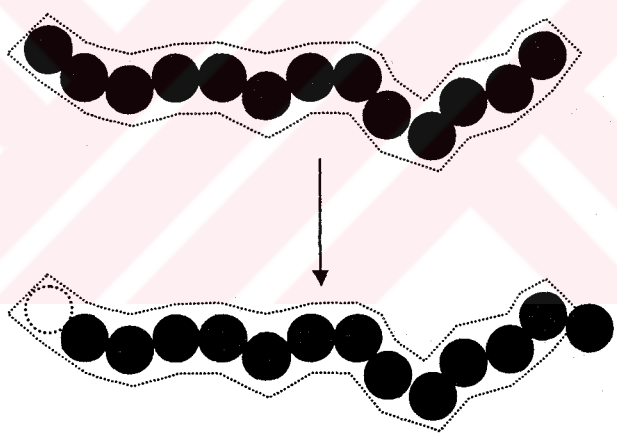
Erimiş ve çözelti içinde çok yoğun halde bulunan polimerlerin hareketleri için bazı modeller geliştirilmiştir. Bu modeller içinde en başarılısı de Gennes sürüngen modelidir [5]. de Gennes sürüngen modeli iki boyutta şu şekilde tasvir edilebilir. Bir yüzey üzerine bir çok çivi çakılı olsun (Şekil 2.11). Bu yüzey üzerine bir ip attığımızda ve ipi bir yönden çektiğimizde ipin yaptığı hareket de Gennes ortaya

koyduğu modele uyar. Bu aslında, hareketi diğer zincirler tarafından kısıtlanmış bir zincirdir. Yüzeydeki çiviler düzlemi delen diğer zincirler gibi düşünülebilir.



Şekil 2.11: Düzlemde sürüngen modeli

Daha genel olarak hareket şöyle tanımlanabilir. Zincir ilk önce bir tüp içinde düşünülür. Zinciri tüpün bir ucundan çektiğimiz zaman diğer uçta tüpün bir kısmı boşalır. Bu boşalan tüpü siler ve zincirin dışarı çıkan kısmını yeni bir tüp içinde olduğunu düşünürsek başlangıçtaki duruma döneriz. Bu şekilde tekrar ederek zinciri hareket ettirebiliriz (Şekil 2.12).



Şekil 2.12: de Gennes sürüngen modeli

Hareket yeterince devam ettirilirse başlangıçta oluşturulan tüpün bütün parçaları yok olur ve zincir yeni bir şekil alır. Tüpün tamamen kaybolması için geçen süre "tüp yenilenme zamanı" (T_r) olarak tanımlanır. Sürüngen modelinin önerdiği hareket, flüoresans boyalarla işaretlenmiş çok yüksek molekül ağırlığındaki DNA moleküllerinin, yüksek zincir konsantrasyonuna sahip çözeltide hareketinin takibi ile açıkça görülmüştür [6].

2.5. Polimerizasyon

Polimerizasyon polimerleri üretme sürecine verilen isimdir. Simülasyonlar için polimerizasyon yürüyüş mekanizması önemlidir. Bu yüzden polimerizasyon sürecindeki kimyasal reaksiyonlardan ziyade, genel mekanizma üzerinde durulacaktır. Genel olarak iki polimerizasyon şekli vardır (basamaklı ve zincir katılma) ve simülasyon kütüphanesi de bu iki polimerizasyon şekli doğrultusunda dizayn edilmiştir. Genel olarak polimerizasyon yürüyüş şekillerini aşağıda listelenmiştir.

- Basamaklı Polimerizasyon
- Radikal zincir (katılma) polimerizasyonu
- İyonik zincir (katılma) polimerizasyonu
- Halka açılması polimerizasyonu
- Sterospesifik polimerizasyon
- Kopolimerizasyon

2.5.1. Basamaklı polimerizasyon

Bu tür polimerizasyon için iki tür molekülün olması gerekmektedir. Bir örnekle açıklanırsa; birinci tür molekülümüz $X-R-X$ olsun diğer molekülümüz ise $Y-R'-Y$ olsun. Burada R ve R' reaksiyonlar boyunca değişmeyen molekülleri temsil etmektedir. $-X$ ve $Y-$ ise birleşerek $-Z-$ bağı oluşturabilen organik fonksiyonel grupları temsil etmektedir. $X-R-X$ ve $Y-R'-Y$ molekülleri birleşirse ortaya $X-R-Z-R'-Y$ yapısı çıkacaktır. Reaksiyon sonucunda baş ve sondaki fonksiyonel gruplar korunduğu için reaksiyon aynı şekilde adım adım devam edecektir. Reaksiyon boyunca polimerlerin ağırlığı sürekli artacaktır. Reaksiyon zamanının artırılmasıyla daha büyük moleküller elde edilebilir. Şu anda simülasyon kütüphanesi $X-R-X+X-R'-X \rightarrow X-R-Z-R'-X$ şeklindeki basamaklı polimerizasyonu desteklemektedir. Ancak bu kısıtlama kütüphanenin yapısından kaynaklanmamaktadır. Değişik fonksiyonel grup implementasyonu kütüphaneye yakın zamanda eklenecektir.

2.5.2. Radikal zincir katılma polimerizasyonu

Bu polimerizasyon şeklinde basamaklı polimerizasyondan farklı olarak reaksiyon zincire tekrarlanan birimin katılması ile olur. Reaksiyonun kinetiği bir başlatıcının iki adet serbest radikale dönüşmesi ile başlar.

$$I \xrightarrow{k_d} 2R^* \tag{2.6}$$

İkinci adım serbest radikalin bir monomerle reaksiyona girip zincir başlatıcı bir eleman oluşturmasıdır.

$$R^* + M \xrightarrow{k_i} M_1^* \tag{2.7}$$

Daha sonra bu monomere diğer monomerler eklenerek polimer zinciri oluşmaya başlar.

$$M_n^* + M \xrightarrow{k_p} M_{n+1}^* \tag{2.8}$$

Zincir oluşumu iki serbest uca sahip zincirin reaksiyona girmesi ile sona erer.

$$\begin{aligned} M_n^* + M_m^* &\xrightarrow{k_{tc}} M_{n+m} \\ M_n^* + M_m^* &\xrightarrow{k_{td}} M_n + M_m \end{aligned} \tag{2.9}$$

Eğer reaksiyon ortamına inhibitör katılmışsa zincir oluşumunu sona erdiren şu reaksiyonlarda oluşur. Dikkat edilirse ilk reaksiyonda henüz polimer oluşmamıştır. İkinci reaksiyonda ise polimer inhibitörle reaksiyona girdiği sırada çok kısa olabilir. İnhibitörler çok aktiftirler reaksiyon başında etkileri görülür daha sonra tükendikleri için etkileri gözlenmez.

$$\begin{aligned} R^* + X &\xrightarrow{k_x} RX \\ M_n^* + X &\xrightarrow{k_x'} M_n \end{aligned} \tag{2.10}$$

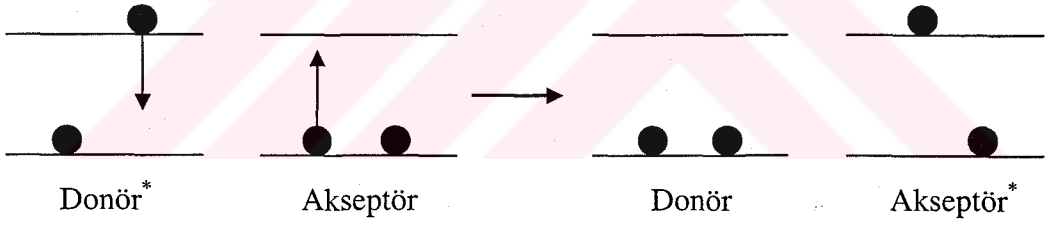
Zincir polimerizasyonunda monomer konsantrasyonu giderek azalır. Yüksek molekül ağırlıklı polimerler oluşturmak bu yöntemle daha kolaydır. Reaksiyon boyunca polimer molekül ağırlıkları çok az değişir. Reaksiyon zamanının uzatılması, verimi artırır fakat daha yüksek molekül ağırlığına sahip polimerler oluşmasını sağlamaz. Reaksiyon karışımı monomer, yüksek molekül ağırlıklı polimer ve çok az miktarda

büyüyen zincir içerir. Simülasyon kütüphanesi sayesinde bu tür bir polimerizasyon mekanizması kolayca oluşturulabilir.

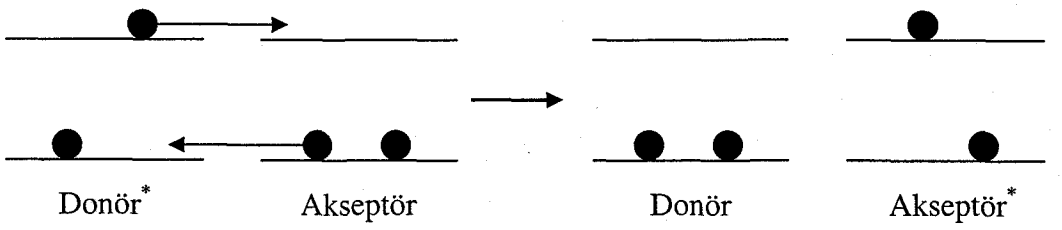
3. DOĞRUDAN ENERJİ TRANSFERİ

3.1. Doğrudan Enerji Transferinin Genel Mekanizması

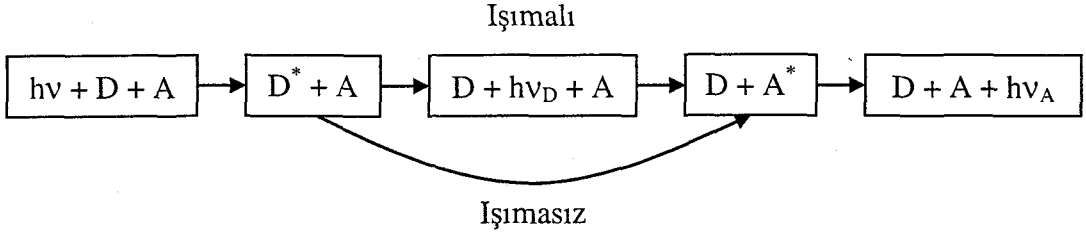
Işımasız enerji transferi, uyarılmış durum enerjisinin donörden akseptöre transferi olarak tanımlanabilir. Bu transfer dipol-dipol etkileşimi ile veya elektron transferi ile olabilir. Bu çalışmada dipol-dipol etkileşmesi ile enerji transferi incelenmiştir. Enerji transferinin bir diğer yolu ise ışımalı enerji transferidir. Bu tip enerji transferinde uyarılmış durumdaki donör bir foton yayımlayarak taban durumuna döner. Açığa çıkan foton akseptörü uyararak onu uyarılmış duruma geçirir. Bu tür transfer, kullanılan kabın boyutu, uyarma ve yayımlama dalga boylarındaki optik yoğunluk, yayım ve uyarma akslarının geometrik oryantasyonu gibi ortamın optik özelliklerine bağlıdır. Halbuki ışımasız enerji transferi donör ve akseptör çiftlerinin moleküler özellikleri hakkında önemli bilgiler verir. Aşağıdaki şekillerde enerji transferi mekanizmaları şematik olarak gösterilmiştir.



Şekil 3.1: Dipol-dipol etkileşmesi ile enerji transferi.

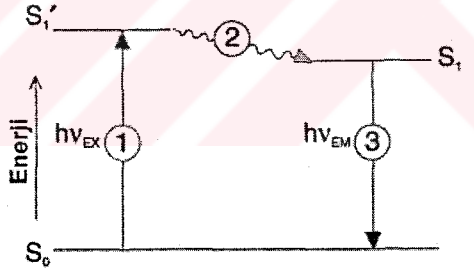


Şekil 3.2: Elektron transferi ile enerji transferi



Şekil 3.3 Işımalı ve ışımasız enerji transferinin şematik gösterimi

Dipol-dipol etkleşimi ile olan ışımasız enerji transferi 10-100 Å mesafede meydana gelir. Bu etkileşme iki diyapazonun etkileşmesine benzetilebilir. Titreşen bir diyapazon oldukça uzak mesafedeki başka bir diyapazonda titreşim başlatabilir. Bu olay titreşen diyapazonun harmoniklerinden birinin diğer diyapazonun doğal frekansını tutması ile meydana gelir. Bu durumda, enerji havada basınç dalgaları ile transfer edilmektedir. Buna benzer olarak dipol-dipol etkileşimi ile enerji transferinde enerji Coulombik rezonans etkileşimi sonucu elektromanyetik alan ile taşınmaktadır. Elektron transferi ise kuantum mekaniksel bir olaydır ve donör ile akseptörün çarpışmaya yakın durumlarında meydana gelir. Bu olay etkileşen donör ve akseptörün dalga fonksiyonlarının üstüste gelmesi olarak özetlenebilir.



Şekil 3.4: Stoke's kayması

Şekil 3.3'ten görülebileceği gibi spektroskopik yöntemle ν_A frekanslı ışığın ν_D frekanslı ışığa oranı ölçülerek donör ve akseptörlerin birbirlerine ne kadar karıştıkları izlenebilir. $\lambda = c / \nu$ formülünü kullanarak her freakansa bir dalga boyu karşılık getirirsek bu dalga boylarının sıralanışı şöyle olacaktır: $\lambda < \lambda_D < \lambda_A$. Bu giderek daha az enerjinin transfer edildiği anlamına gelmektedir. Bu olaya "Stoke's kayması" adı verilir. Şekil 3.4'den de takip edilebileceği gibi molekül ilk önce S_0 taban durumundadır. Daha sonra ν_{ex} frekanslı ışık ile uyarılarak titreşimsel olarak uyarılmış S_1' durumuna geçer ve çok kısa bir zaman zarfında ısı enerjisi yayarak S_1

uyarılmış durumuna iner. Daha sonra ν_{em} frekanslı ışık yayarak taban duruma geri döner. İşte bu fiziksel olay "Flüoresans Rezonans Enerji Transfer Spektroskopisi"ni (FRET) mümkün kılar.

Transfer edilen enerjinin miktarı donörün emisyon spektrumu ile akseptörün absorbsiyon spektrumlarının üst üste gelme oranına, donör ve akseptör geçiş dipollerinin birbirine göre yönelimine ve aralarındaki mesafeye bağlıdır. Enerji transfer miktarının mesafeye bağlı oluşu donör ve akseptörlerin arasındaki mesafenin ölçülmesini sağlar.

Bir donörden bir akseptöre iletilen enerji oranı (k_T)

$$k_T = \frac{1}{\tau_d}\left(\frac{R_0}{r}\right)^6 \tag{3.1}$$

olarak verilir. Burada τ_d akseptör olmayan bir sistemde donörün ömrü, r donör ve akseptör arasındaki mesafe, R_0 ise Förster mesafesi [7] olarak adlandırılan karakteristik bir mesafedir. Förster mesafesi, enerji transferinin verimliliğinin %50'ye düştüğü uzunluk olarak tanımlanır.

Birbirinden r mesafe uzaklıkta bir donör ve akseptör göz önüne alalım. Donörden akseptöre transfer edilen enerji miktarı [8]

$$k_T = \frac{9000(\ln 10)\kappa^2\phi_d}{128\pi^5 n^4 N r^6 \tau_d}\int \frac{F_d(\bar{\nu})\varepsilon_a(\bar{\nu})}{\bar{\nu}^4}d\bar{\nu} \tag{3.2}$$

$$k_T = (r^{-6}J\kappa^2 n^{-4}\lambda_d)\times 8.71\times 10^{23}\ s^{-1} \tag{3.3}$$

formülleriyle verilir. Burada ϕ_d akseptör olmayan bir sistemde donörün kuantum verimidir ve yayımlanan fotonların absorblanan fotonlara oranı olarak tanımlanabilir. N Avogadro sayısını, r donör ve akseptörler arasındaki uzaklığı ve τ_d'de daha evvel belirtildiği gibi akseptör olmayan bir ortamda donörün ömrünü temsil etmektedir. Denklem (3.2)'deki ϕ_d/τ_d oranı denklem (3.3)'de λ_d olarak yazılmıştır ve donörün emisyon oranı olarak adlandırılır. Denklem (3.2)'deki integral donör emisyon ile akseptör absorbsiyon spektrumlarının üst üste gelme derecesini belirler.

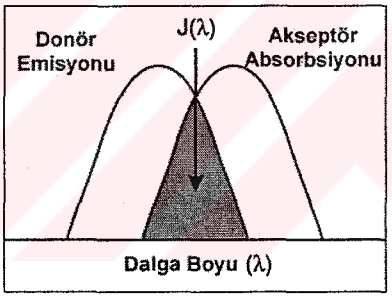
Denklem (3.3)'de bu integral J olarak gösterilmiştir. İntegral dalga sayısı yerine dalga boyu ile yazılmak istenirse

$$J = \int_0^\infty F_d(\lambda)\varepsilon_a(\lambda)\lambda^4 d\lambda \tag{3.4}$$

şekline dönüşür. $\varepsilon_a(\lambda)$ akseptörün λ dalga boylu ışığı söndürme katsayısıdır. Burada $F_d(\lambda)$ aşağıdaki gibi tanımlanır.

$$F_d(\lambda) = \frac{f_d(\lambda)}{\int_0^\infty f_d(\lambda)d\lambda} \tag{3.5}$$

$f_d(\lambda)$ bir birim dalga boyu aralığındaki donör flüoresansıdır. $F_d(\lambda)$ ise donör flüoresans spektrumunun dalga boyu skalasında normalize halidir. Şekil 3.5'de J integralinin temsili gösterilmiştir.



Şekil 3.5: J integralinin temsili

Denklem (3.1) ve denklem (3.2) bir arada kullanılarak Förster mesafesinin (R_0) ifadesi elde edilebilir.

$$R_0^6 = \frac{9000(\ln 10)\kappa^2\phi_d}{128\pi^5 n^4 N}\int \frac{F_d(\bar{\nu})\varepsilon_a(\bar{\nu})}{\bar{\nu}^4}d\bar{\nu} \tag{3.6}$$

Dikkat edilirse $r = R_0$ için k_T, τ_d^{-1} eşit olmaktadır. τ_d^{-1}'e donörün bozunma oranı denir. Denklemdeki sabitler R_0'ı cm biriminden vermek üzere ayarlanmıştır. Eğer sabitler hesaplanır ve A° birimine çevrilirse R_0'ın değeri için aşağıdaki ifade elde edilir.

$$R_0 = 9.79\times10^3(\kappa^2 n^{-4}\phi_d J)^{1/6} \quad (A^\circ) \tag{3.7}$$

Bir önemli parametrede (E) enerji transferinin verimidir. Bu donörün absorbe ettiği fotonların akseptöre geçenlere oranıdır.

$$E = \frac{k_T}{\tau_d^{-1} + k_T} \tag{3.8}$$

Transfer verimini bir başka hesaplama yöntemi de akseptörün sistemde olduğu durumda (F_{da}) ve olmadığı durumdaki (F_d) göreli flüoresans veirmlerini veya ömürleri (sırasıyla τ_{da} ve τ_d) oranlamaktır.

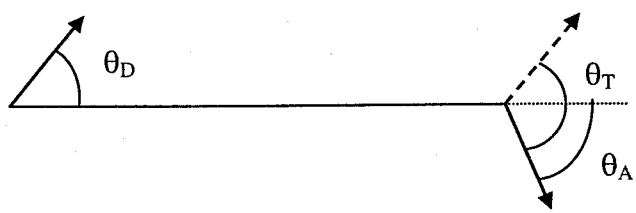
$$E = 1 - \frac{\tau_{da}}{\tau_d} \tag{3.9}$$

$$E = 1 - \frac{F_{da}}{F_d} \tag{3.10}$$

Denklem (3.9), denklem (3.8)'den $\tau_{da} = (\tau_d^{-1} + k_T)^{-1}$ veya $k_T = \tau_{da}^{-1} - \tau_d^{-1}$ dönüşümleri ile elde edilebilir. Denklem (3.10)'da $F_{da}/F_d = \tau_d^{-1}/(\tau_d^{-1} + k_T)$ dönüşümü ile elde edilebilir.

Denklem (3.1) ve denklem (3.9) kullanılarak transfer verimi için yeni bir ifade türetilebilir.

$$E = \frac{R_0^6}{R_0^6 + r^6} \tag{3.11}$$



Şekil 3.6: κ^2'nin hesabı için donör – akseptör dipol açıları [8]

Bir diğer önemli parametre κ^2'dir. Bu parametre 0 ile 4 arasında değerler alır. Birbirine paralel dipoller için 4, paralel fakat ters yöne bakan dipoller için 1 ve

birbirine dik dipoller için 0 değerini almaktadır. κ^2'nin 1 ile 4 arasındaki değişimi donör akseptör arası mesafenin (r) hesaplanmasında %26 hataya sebep olur ancak $\kappa^2 = 0$ durumu hesaplarda önemli hataya sebebiyet verebilir. κ^2'nin ifadesi şöyledir:

$$\kappa^2 = (\cos\theta_T - 3\cos\theta_d \cos\theta_a)^2 \tag{3.12}$$

Denklem (3.12)'de kullanılan açılar şekil 3.6'da gösterilmiştir. κ^2 donör ve akseptörlerin rasgele yöneldiği durumlar için 2/3 kabul edilir. Aynı zamanda literatürde verilen Förster mesafeleri, ayrıca belirtilmemişse, $\kappa^2 = 2/3$ alınarak hesaplanmıştır. Buna alternatif olarak, uyarılmış durum boyunca statik bir donör akseptör oryantasyonu olduğu kabul edilirse $\kappa^2 = 0.476$ alınabilir [8].

3.2. Zamana Bağlı Flüoresans

Doğrudan enerji transferi kullanılan, zamana bağlı flüoresans deneylerinde, lateks parçacıklarının uygun flüoresan boyalarla etiketlenmesi gerekir. Karışımın bir kısmı donör ile, diğer kısmıda akseptör ile işaretlenir. Örnek ilk oluşturulduğunda flüoresans bozunma profili tek üslüdür. Difüziyon başladıktan sonra flüoresans bozunma profili iki terime sahip olur. İlk terim (B_1) birbiriyle etkileşen kısmın enerji transferinden, diğer terimse (B_2) donör olarak işaretlenmiş fakat akseptörlerle karışamamış zincirlerden gelir. Donörlerin bozunma profili

$$\frac{I(t)}{I(0)} = B_1 \exp(-\frac{t}{\tau_d} - C(\frac{t}{\tau_d})^{1/2}) + B_2 \exp(-\frac{t}{\tau_d}) \tag{3.13}$$

ifadesiyle [9] verilir. Bu ifadede C akseptörlerin konsantrasyonu ile orantılı bir parametre ve τ_d donörlerin ömrüdür.

Donörler ışıkla ani olarak uyarıldıklarında tekrar taban durumlarına, flüoresans bir foton yayınlayarak yada ışıma yapmayan bir mekanizma ile dönerler. İdeale yakın bir sistemde donörler, ani bir ışık uyarılmasına maruz bırakıldıklarında flüoresans şiddeti üstel olarak azalır. Fakat uyarılmış donörün yakınlarında bir akseptör varsa uyarılmış donörden taban durumundaki akseptöre doğrudan enerji transferi olma

ihtimali vardır. Doğrudan enerji transferinin klasik probleminde, r_i noktasindaki bir akseptörü gözönünde bulundurarak, r_k noktasındaki bir donörün bozunma olasılığı

$$P_k(t) = \exp(\frac{-t}{\tau_d} - w_{ik}t) \tag{3.14}$$

olarak verilir. Burada w_{ik} Förster [7] tarafından verilen enerji transfer oranıdır.

$$w_{ik} = \frac{3}{2}\kappa^2 \frac{1}{\tau_d}(\frac{R_0}{r_{ik}})^6 \tag{3.15}$$

Burada R_0 kritik Förster mesafesidir. κ birbirleriyle etkileşen dipollerin geometrisiyle ilgili birimsiz bir parametredir. Eğer sistem N_D donör ve N_A akseptör içeriyorsa donör flüoresans şiddetinin bozunması denklem (3.14)'den türetilebilir [10].

$$\frac{I(t)}{I(0)} = \exp(\frac{-t}{\tau_d})\frac{1}{N_D}\int n_D(r_k)dr_k \prod_{i=1}^{N_A}\frac{1}{N_A}\int n_A(r_i)dr_i \exp(-w_{ik}) \tag{3.16}$$

Burada n_D ve n_A donör ve akseptörlerin dağılım fonksiyonlarıdır. Termodinamik limitte denklem (3.16)

$$\frac{I(t)}{I(0)} = \exp(\frac{-t}{\tau_d})\frac{1}{N_D}\int n_D(r_k)dr_k \times \exp(-\int n_A(r_i)dr_i(1-\exp(w_{ik}t))) \tag{3.17}$$

şekline dönüşür. Bu denklem Monte-Carlo tekniği kullanılarak donör bozunma profilleri oluşturulmasında kullanılabilir. Eğer denklem (3.16)'da $r_{ik} = r_i - r_k$ koordinat dönüşümü yapılırsa

$$\frac{I(t)}{I(0)} = \exp(\frac{-t}{\tau_d})\frac{1}{N_D}\int n_D(r_k)dr_k \times \prod_{i=1}^{N_A}\int_{r_k}^{R_g - r_k} n_A(r_{ik} - r_k)dr_{ik} \exp(-w_{ik}) \tag{3.18}$$

elde edilir. Burada R_g keyfi bir üst limittir. Bir donör orijine yerleştirilir ve karışan ve karışmayan kısımların, difüziyon sırasında oluştuğu kabul edilirse denklem (3.18) aşağıdaki şekile dönüşür.

$$\frac{I(t)}{I(0)} = f_1 \exp(\frac{-t}{\tau_d}) \prod_{i=1}^{N_A} \frac{1}{N_A} \int_0^{R_g} n_A(r_{ik}) dr_{ik} \exp(-w_{ik} t) + f_2 \exp(\frac{-t}{\tau_d}) \tag{3.19}$$

$$f_{1,2} = \frac{1}{N_D} \int_{1,2} n_D(r_k) dr_k \tag{3.20}$$

Burada f_1 donörlerin karışmış bölümünü f_2 ise karışmamış bölümünü temsil etmektedir. Denklem (3.19)'daki integral förster tipi bir fonksiyon oluşturur [9].

$$\prod_{i=1}^{N_A} \frac{1}{N_A} \int_0^{R_g} n_A(r_{ik}) dr_{ik} \exp(-w_{ik} t) = \exp(-C(\frac{t}{\tau_d})^{1/2}) \tag{3.21}$$

Burada C akseptör konsantrasyonu ile orantılıdır. Şimdi denklem (3.19) ile denklem (3.13) aynı forma gelmiştir.

$$\frac{I(t)}{I(0)} = f_1 \exp(\frac{-t}{\tau_d} - C(\frac{t}{\tau_d})^{1/2}) + f_2 \exp(\frac{-t}{\tau_d}) \tag{3.22}$$

Buradaki f_1 ve f_2 denklem (3.13)'deki B_1 ve B_2'ye karşılık gelmektedir. Buradan karışım miktarı K belirlenebilir.

$$K = \frac{B_1}{B_1 + B_2} = \frac{f_1}{f_1 + f_2} \tag{3.23}$$

4. KİNETİK MONTE CARLO

Eğer doğayı işleten tüm kuvvetlerin malumatına, tüm parçacıkların konumlarının, hızlarının ve kuvvetler için gerekli özelliklerinin bilgisine ve tabi ki bütün bu bilgiyi analiz edecek kadar işlem gücüne sahip olsaydık geçmiş ve gelecek şu an gibi gözlerimizin önünde apaçık olacaktı diyor Laplace. Bu düşünce temel olarak doğru olsa da kuantum mekaniğinin gelişimi ile bu bilgiye hiçbir zaman ulaşılamayacağı ortaya çıkmıştır. Simülasyon yöntemi olarak ilk akla gelen Moleküler Dinamik bir bakıma Laplace'ın bu düşüncesinden türemiştir. Özellikle N cisim probleminin analitik çözümünün olmayışı Moleküler Dinamiğin ortaya çıkmasında önemli paya sahiptir [1]. Ancak Moleküler Dinamik yüksek sayıda matematiksel hesaplama içeren bir yöntemdir ve bazı problemlerin çözümü Moleküler Dinamikle mümkün olsa da cevabı alma süresi insan ömründen bile uzun olabilir. Ayrıca Moleküler Dinamik, simüle edilen sistemlerdeki tüm kimyasal özelliklerin bilinmesini gerektirir. Bu da Moleküler Dinamiği gerçek sistemlerin simüle edilmesi için ideal bir yöntem yapar. Ancak simüle edilebilen zaman dilimi saniyenin kesirlerini geçmez. Monte Carlo tekniği problemlere değişik bir bakış açısı getirir.

Monte Carlo tekniğinde simülasyonun yürümesi için rasgele üretilmiş sayılardan yararlanılır. Faz uzayında sadece konum bilgisi vardır momentum bilgisi ise yoktur. Hareketlerin yönü rasgele sayılar sayesinde saptanır ve hareket modele göre belirlenen kurallar çerçevesinde kabul veya reddedilir. Sistemdeki tüm kimyasal özellikler tek bir potansiyelde birleştirilmiştir. Monte Carlo tekniği bu sayede daha uzun zaman dilimlerinin simülasyonu için kullanılabilir. Monte Carlo tekniğinin moleküler olayların simülasyonu için kullanılması Moleküler Monte Carlo olarak adlandırılır. Moleküler Monte Carlo kendi içinde gruplara ayrılır. Bunlar:

- Klasik Monte Carlo: Örnekler dağılım fonksiyonlarından (çoğunlukla klasik Boltzmann Dağılımı) çıkarılır. Kullanım alanları termodinamik özelliklerin tespiti, minimum enerji durumlarının çıkarılması olarak sıralanabilir.
- Kuantum Monte Carlo: Rasgele adımlar kuantum mekaniksel enerjileri ve dalga fonksiyonlarının hesabı için kullanılır.

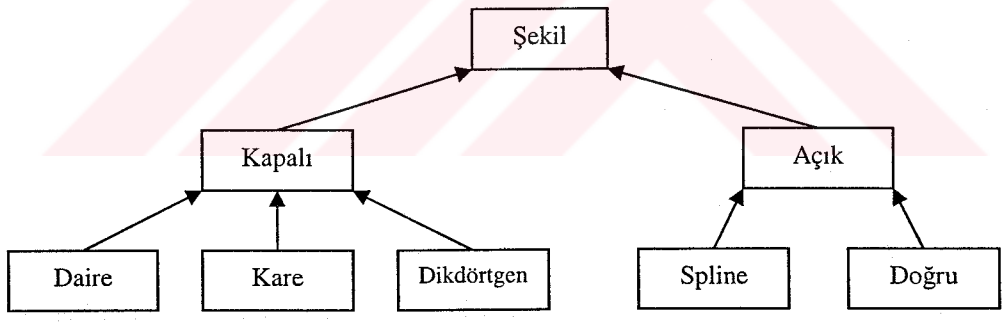
- İz integrali Kuantum Monte Carlo: İstatistik kuantum mekaniksel integrallerin hesabı, termodinamik özelliklerin tespiti için, Feynman iz integralleri kullanılarak yapılır.
- Hacimsel (Volumetric) Monte Carlo: Rasgele sayılar moleküler hacimler, ve moleküler uzay faz yüzeyleri oluşturmak için kullanılır.
- Simülasyon Monte Carlo: Moleküler dinamiğe stokastik etkiler dahil edilerek molekül dinamiği simüle edilir. Kinetik Monte Carlo yöntemi en çok kullanılan Simülasyon Monte Carlo Metodudur.

Bu çalışmada bir Simülasyon Monte Carlo Metodu olan Kinetik Monte Carlo kullanılmıştır. Zincirlerin hareketi rasgele üretilen sayıların yardımıyla yapılmakla beraber zincir elemanları aynı hacmi paylaşamazlar.

Polimerler endüstri için çok önemli malzemelerdir. Günlük hayatımızda kullandığımız bir çok cihaz polimer malzemeler içerir. Bununla beraber yaşamın temeli olan proteinler, genetik bilgi taşıyıcısı DNA ve RNA, dünya üzerindeki birçok canlının enerji kaynağı polisakaritler, doğada bulunan polimerlerdir. Böyle önemli malzemelerin özellikleri doğal olarak bir çok araştırmacıyı cezp etmektedir. Bu da polimerlerin kimyasal ve fiziksel özellikleri üzerine birçok çalışma yapılmasına sebebiyet vermektedir. Bu çalışmalarda simülasyonlar önemli yer tutar. Polimerler çok büyük boyutlu moleküllerdir. Bunların karışım halinde bulunduğu bir sistemin incelenmesi için çeşitli modeller geliştirilmiştir. Bu modellerin test edilmesi büyük ölçüde simülasyonlarla yapılır. Bu çalışmada da polimerlerin çeşitli kimyasal ve fiziksel özelliklerinin simüle edilebilmesi için kinetik Monte Carlo yöntemini kullanan kütüphane yazılım hazırlanmıştır.

5. SİMÜLASYON KÜTÜPHANESİ

Bu çalışmada amaç genel kullanım amaçlı bir kütüphane oluşturulmasıdır. Bu sebepten oluşturulacak veri tiplerinin olabildiğince esnek yapıya sahip olması gerekmektedir. Kütüphane C++ programlama dili ile nesne yönelimli olarak yazılmıştır. Programlamada nesne yönelimini tanıtmakta fayda vardır. Nesne yöneliminden önce nesne tabanlı programlama gelişmiştir. Nesne tabanlı programlama nesne yönelimli programlamanın alt kümesidir. Nesne tabanlı programlarda yapılacak işe göre özel veri tipleri ve bu veri tiplerine ait özel fonksiyonlar üretilir. Sonuçta program bu veri tipleri kullanılarak yazılır. Nesne yönelimli programlar nesne tabanlı programlardan miras (inheritance) özelliğini kullanmaları ile ayrılırlar [11]. Miras özelliğini bir veri tipinden yeni bir veri tipi türetilmesine izin verir. Yeni veri tipi aynı zamanda türetildiği veri tipinin özelliklerini taşır. Şekil 5.1'de örnek bir veri tipi şeması verilmiştir.



Şekil 5.1: Veri yapılarında miras özelliğine örnek. Bu tür sınıf şemalarında okların taban sınıfa doğru çizilmesi standartlaşmıştır.

Şekil 5.1'deki veri yapısını biraz açarsak "Daire" tipinde bir veri hem "Kapalı" şekiller için üretilen fonksiyonları hem de "Şekil" tipi için üretilen fonksiyonları kullanabileceği gibi türetildiği bu sınıfların üye elemanlarına da erişebilir. "Kapalı" veri tipinin alan hesaplayan bir fonksiyonu ve alan bilgisini tutan bir üyesi olduğunu farz edelim. Dairenin, karenin ve dikdörtgenin alanları farklı şekillerde hesaplanır. Bu durumda "Kapalı" veri tipi için alan hesaplama fonksiyonu çağırılırsa program çalışma esnasında, kapalı veri tipi daire ise "Daire" sınıfının, kare ise "Kare" sınıfının, dikdörtgen ise "Dikdörtgen" sınıfının alan hesaplama rutinini çağıracaktır.

Burada ki vurgulanması gereken nokta bu seçimin program çalışırken yapılıyor olmasıdır. Bu tür programlara şekil değiştirebilen manasında, polimorfik programlar adı verilir. "Açık" veri tipi için alan fonksiyonu ve üyesi anlamsızdır. Bu tiplerin hepsi "Şekil" veri tipinden türetilmiştir. "Şekil" veri tipinde tüm şekiller için ortak üyeler bulunur (örneğin uzunluk).

Bu kütüphaneyi mümkün kılan bir diğer unsur işaretçilerdir. İşaretçiler hafızada bir verinin yerini gösteren veri tipleri olarak tanımlanabilir. Her ne kadar kütüphane fonksiyonları yoğun olarak işaretçi kullanıyorlarsa da kütüphane, kullanıcının işaretçi kullanmadan birçok karmaşık simülasyon yapmasını sağlayacak şekilde tasarlanmıştır.

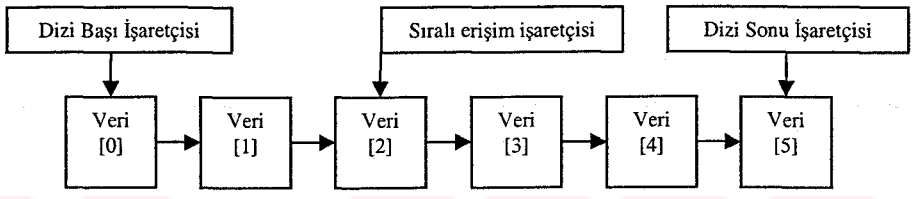
5.1. Veri Yapıları

Veri yapılarının dizaynı kütüphanenin en önemli noktasını oluşturmaktadır. Dizayn ne kadar iyi olursa kütüphanenin genişletilebilirliğinin o kadar iyi olacağı aşikardır. İlk adım olarak bir taşıyıcı sınıf oluşturulmuştur. Bu yapı istenilen veri tipinde dizi oluşturmaktadır. Dizi bağlı liste şeklinde oluşturulduğundan eleman ekleme, eleman silme, iki dizinin birleştirilmesi gibi işlemler çok hızlı bir şekilde yapılabilmektedir. Bu taşıyıcı sınıf "smartarray" olarak adlandırılmıştır. Aşağıdaki program parçasında smartarray ile yapılabilecek bazı fonksiyonlar gösterilmiştir.

```
/*1*/ smartarray<double> Dizi1(5),Dizi2;
/*2*/ double Yenieleman=3.14;
/*3*/ Dizi2.Allocate(10);
/*4*/ Dizi1.Init(0);
/*5*/ Dizi1.AddEl2Head(YeniEleman);
/*6*/ Dizi2.ShrinkFromTail(5);
/*7*/ Dizi1.Merge(Dizi2);
/*8*/ Dizi1[0]=6.28;
```

Yapılanlar kısaca özetlenecek olursa ilk satırda "double" tipinde iki adet dizi oluşturulmuştur. "Dizi1" için beş elemanlık hafıza ayrılmış "Dizi2" için ise hafıza ayrılmamıştır. Üçüncü satırda "Dizi2"ye 10 elemanlık hafıza ayrılmıştır. Dördüncü

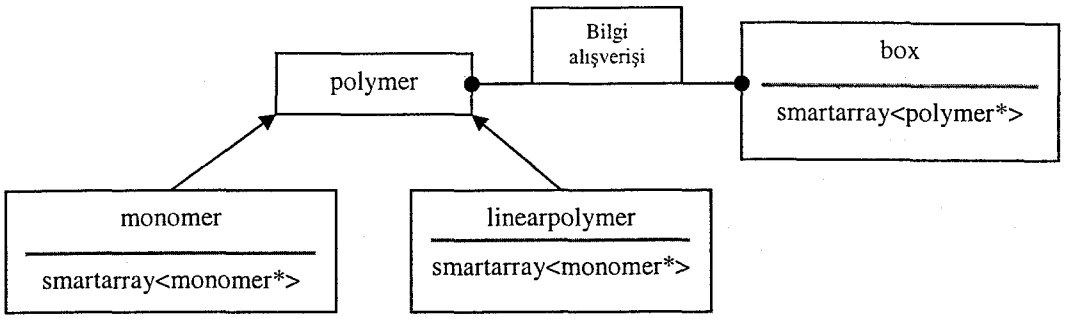
satırda "Dizi1"in elemanları sıfıra eşitlenmiştir. Beşinci satırda "Dizi1"in başına bir eleman daha eklenmiştir ("Dizi1" 6 elemanlı olmuştur.). Altıncı satırda "Dizi2"nin sondan 5 elemanı diziden ayrılmıştır ("Dizi2" 5 elemanlı olmuştur.). Yedinci satırda iki dizi "Dizi1" adı altında birleştirilmiştir. Son satırda ise "Dizi1"in ilk elemanına (Değeri, değiştirilmeden önce 3.14 olan) 6.28 değeri atanmıştır. "smartarray" veri taşıyıcısının burada gösterilmeyen birçok fonksiyonu daha vardır. Oluşturulan veri yapıları bu taşıyıcı kullanılarak inşa edilmiştir. Dolayısıyla "smartarray"de fonksiyonların isimleri korunarak yapılacak optimizasyonlar doğrudan simülasyon kütüphanesinin performansını etkileyecektir. Bu programın nesne tabanlı olmasının getirdiği bir avantajdır. Şekil 5.2'de "smartarray"in veri taşıma yapısı resmedilmiştir.



Şekil 5.2: "smartarray"in veri taşıma yapısı

"smartarray" Şekil 5.2'den de görüleceği üzere elemanları hafızada bağlı liste şeklinde tutmaktadır. Bu tür yapılarda eleman ekleme ve çıkartma işlemleri çok hızlı olmakla beraber rasgele erişim yavaştır. Bir elemana erişmek için bütün elemanları iterasyonla kat etmek gereklidir. Sıralı erişim özel bir işaretçi sayesinde optimize edilmiştir. Bu özel işaretçi en son eriştiğiniz elemanın hafıza adresini gösterir. Bir sonraki elemana erişmek istenildiğinde son erişilen elemandan itibaren iterasyon yapılır. Erişmek istenilen eleman, en son erişilen elemandan hemen sonra ise iterasyona gerek kalmadan istenilen elemana ulaşılabilir.

Diğer veri yapılarının hepsi daha evvel belirtildiği gibi "smartarray" yapısı kullanılarak oluşturulmuştur. Temel yapının adı "polymer"dir. Bu veri tipinden "monomer" ve "linearpolymer" sınıfları türetilmiştir. Bunlardan ayrı olarak "box" adında polimerlerin simüle edileceği kutuyu oluşturan bir yapı daha mevcuttur. Şekil 5.3 simülasyondaki yapıları göstermektedir. Yaratılan her yapıya bir kimlik numarası (ID) atanır. Bu numara bazı algoritmaların çalışması için önemli rol oynar. Hiç bir yapının kimlik numarasının aynı olma ihtimali yoktur.



Şekil 5.3: Simülasyondaki veri tiplerinin sınıf şeması

Bu yapılar gerçek nesnelere karşılık geldiğinden çalışmanın geri kalanında bu yapıların isimlerinin Türkçe karşılıkları kullanılacaktır (monomer → monomer, polymer → polimer, linearpolymer → lineer polimer, box → kutu).

Monomerler bir "monomer işaretçileri" dizisine sahiptirler. Bu sayede monomerler kendi aralarında bağlanabilir. Bağlanan monomerler birbirlerini gösteren işaretçilere sahip olurlar. Lineer polimerlerde "monomer işaretçileri" dizisine sahiptir. Kendilerine eklenen monomerlerin bilgisini bu dizide tuttukları gibi kendi monomerlerini de oluşturabilirler. Kutu (box) ise kendisine bağlı bulunan tüm tiplere bağlantısını sağlayan bir "polimer işaretçileri" dizisine sahiptir. Bütün diğer yapılar (şu an için monomer ve lineer polimer) polimer tipinden türetildiği için bu işaretçi her iki veri tipini de gösterebilmektedir. Daha sonra eklenecek yapılar (dallanmış polimer gibi) yine polimer tipinden türetileceği için kutu yapısında değişikliğe gitmeye gerek kalmayacaktır.

Monomerler kütüphanenin temel taşlarıdır. İster lineer polimerler olsun ister daha sonra türetilecek yapılar olsun monomerlerden oluşurlar. Kullanıcı isterse sadece monomer yapıları ile de çalışabilir. Monomerler birbirlerine bağlanabilirler veya bağlı monomerler birbirlerinden ayrılabilirler. Bir monomer diğerine bağlandığında iki monomer birbirinden haberdar hale gelir. Bağlantı yapma kapasiteleri teorik olarak sınırsızdır ancak pratikte programlama dilinin ve donanımın getirdiği sınırlara sahiptir. Şöyle ki 32 bit bir programda işaretsiz tamsayıların alabileceği maksimum değer 4,294,967,295'dir. Bir monomere bu sayının üzerinde bağlantı yapılamaz. Zaten bu limit yapılacak simülasyonda kullanılacak monomer sayısının çok üzeride kalmaktadır ve bu miktarda veriden çok daha azı bile hafızayı taşıracaktır. Monomerlerin bağlantı kapasiteleri sınırlandırılabilir; bu sayede monomerler bağlantı kapasiteleri dolduğunda daha fazla bağlantı kabul etmezler. Bununla beraber bağlantı kapasitesi dinamik olarak tekrar artırılabilir. Monomerler arasında bağlantı

yapıldıktan sonra herhangi iki monomerin birbirine bağlı olup olmadıkları kontrol edilebilir. Burada kastedilen bağlılık, doğrudan komşu bağlılığı değildir. Bağlılık iki monomerin başka monomerler üzerinden bir yolla bağlı olup olmadığını ifade eder ve Bölüm 5.4'te anlatılacak algoritma ile bulunur. Algoritmanın çalışması bağlantı şemasında bir kısıtlamaya gidilmesine gerek olmayacak şekilde tasarlanmıştır. İstenirse (fiziksel karşılıkları olmasa bile) bir monomer, kendine ve başka monomere, birden fazla bağ ile bağlanabilir. Kapalı çevrimler oluşturulabilir. Monomerlerin karşılıklı bağlanma yetenekleri dışında bir de tek taraflı bağlantı yetenekleri vardır. Bir monomer başka bir monomere tek taraflı bağlandığında bağlanılan monomer kendisine başka bir monomerin bağladığı bilgisine sahip olmaz. Bu bir monomerin yakınındaki elemanların listesini tutmak amaçlı kullanılabilir.

Monomerler taşıdıkları bağlantı bilgisi yanında koordinat ve çap bilgisi taşırlar. Bu bilgi "data3d" olarak adlandırılan ufak bir yapı içinde saklanır. İstenildiği zaman üye fonksiyonlar ile bu bilgi değiştirilebilir veya bu bilgiye ulaşılabilir.

Monomerlerin koordinat ve çap dışında başka bilgiler taşıması da nesne tabanlı yapı sayesinde az bir programcılık ile sağlanabilir. Bundan daha az gelişmiş aynı temele sahip bir yapı daha önceki bir çalışmada yapay sinir ağları oluşturulması için kullanılmıştır. Aslında monomerleri fiziksel nesneler haline getiren çap ve koordinat bilgisidir. Bunun yanında monomerler hangi kutu içinde bulundukları bilgisini taşırlar. Monomer tipi içine programcıların serbestçe kullanabileceği bir bayrak değişken de yerleştirilmiştir. Monomer yapısının kullanımı aşağıda ufak bir program parçasında gösterilmiştir.

```
monomer A,B,C,D,E;      //Monomer yapıları oluşturuldu
                        //Başlangıç bağlantı kapasitesi 2'dir
data3d koordinat;       //Koordinat tanımı
koordinat.x=1.0;koordinat.y=1.0;koordinat.z=1.0;
koordinat.Radius=2.0;
A.ChangeFreeCLCap(1);   //A'nın bağlantı kapasitesi 1 yapıldı
A.Connect(B);           //A ile B bağlandı.
if(!(A.Connect(C))){    //A ile C bağlanmaya çalışıldı. Ancak A'nın
                        //bağlantı kapasitesi dolduğundan "false"
                        //geri döndü "if"in içi "true" olduğundan
```

```
                        //"if" bloğuna girildi.
  A.ChangeFreeCLCap(1);//A'nın bağlantı kapasitesi bir artırıldı
  A.Connect(C);         //A ile C bağlandı
}
A.FreeCLConnect(D);     //A ile D bağlandı. Bu özel bağlantı A'nın
                        //bağlantı kapasitesi dolu olsa bile önce
                        //kapasiteyi artırarak bağlanmayı mümkün
                        //kılar.
A.Disconnect(C);        //A ile C ayrıldı
D.Connect(E);           //D ile E bağlandı
A.IsConnected(D);//A ile D bağlılık kontrolü "true" geri dönecektir
A.IsConnected(E);//A ile E bağlılık kontrolü "true" geri dönecektir
A.IsConnected(C);//A ile C bağlılık kontrolü "false" geri dönecektir
A.ChangeCoor(koordinat);//A'nın koordinat bilgisinin değiştirilmesi
```

Lineer polimer veri tipi, isminden anlaşılabileceği gibi uzun bir programlama zamanından tasarruf edilerek, kolayca lineer polimer zincirleri oluşturmak için tasarlanmıştır. Lineer polimerler kendi aralarında, zincir başı veya sonundan bağlanabilirler. Bunun yanında lineer polimerlere monomer eklenebilir. Yani monomer veri tipi ile de bağlantı kurabilirler. Monomerler ve lineer polimerler daha sonra açıklanacak olan kutu veri tipinin içine yerleştirilmişlerse, birbirleriyle belirli bir mesafeye geldiklerinde otomatik olarak bağlanmaları sağlanabilir. Bu mesafe kullanıcı tarafından tanımlanır. Her bir lineer polimeri için değişik bağ açısı (θ) belirlenebilir. Bunun yanında dihedral açıların (ϕ) oluşturulması için, kullanıcı tarafından yazılan herhangi bir dağılım fonksiyonu kullanılabilir. Yazılacak bu dağılım fonksiyonu, 0° ile 360° arasında açı değeri alıp 0 ile 1 arasında bu açıya ait olasılığı vermesi gerekmektedir. Dağılım fonksiyonu verilmezse zincir serbest dönenilen zincir olarak oluşturulur. Aşağıdaki örnekte bir lineer polimer oluşturulmaktadır.

```
#include "polymer.h"
double dagilimfonk(double aci){
```

```
  if(aci>=179.0||aci<=181.0)return 1.0;
  else return 0.0;
}
int main(){
  box kutu1(100,100,100);
  linearpolymer A;
  monomer B;
  A.CreatePolymer(20,1,90,kutu1,dagilimfonk);
  A.AddMonomer(B,HEAD,15.0)
}
```

Yukarıdaki kısa programda, 100 birim kenar uzunluğuna sahip bir küpün içerisinde, çapları 1 birim olan 20 monomerden bir lineer polimer oluşturulmaktadır. Bu polimerin bağ açısı 90°, dihedral açısı ise 179° ile 181° arasında değişmektedir. Şekli merdiven gibi olacaktır. Bu tamamen "trans" durumunda bir polimerdir ve kristalize olmuş bir polimer olarak düşünülebilir. Programın son satırında ise lineer polimerin başına 15° dihedral açı ile bir monomer eklenmektedir. Eğer polimerin başı yerine sonuna monomer eklenmek isteniyorsa "HEAD" yerine "TAIL" kullanılması gerekmektedir. Eklenen monomer programın bu noktasından itibaren polimerin parçası haline gelir. Monomerin özelliklerinin (koordinat, bağlantı kapasitesi gibi) değiştirilmesine izin verilmez.

Son olarak bahsedilecek yapı "box" yani kutu yapısıdır. Kutu simülasyonun yürütüleceği bir kap olarak görev yapar. Kutu içine konulan polimerler ve monomerlerin üst üste gelmemesinin kontrolü kutu yapısı sayesinde olur. Ancak istenirse üst üste gelme kontrolü iptal edilebilir. Kutular birbirleriyle üç asal eksen doğrultusunda birleştirilebilir. Ancak kutuların birleştikleri yüzeylerin enleri ve boyları, birbirleri ile aynı olması gereklidir. Bu bağlamda kutular ancak dikdörtgenler prizması veya küp şeklinde olur. Kutu duvarları iticidir. Periyodik bağ koşulları tam olarak bitirilememiştir, bu yüzden şu an için kütüphane tarafından desteklenmemektedir. Periyodik bağ koşulları ile ilgili sorun açı hesaplarından

kaynaklanmaktadır[↓]. Aşağıdaki programda iki kutunun birleştirilmesinin örneği verilmiştir.

```
box A(100,50,50),B(50,50,50);
A.MergeBox(B,XAXIS);
```

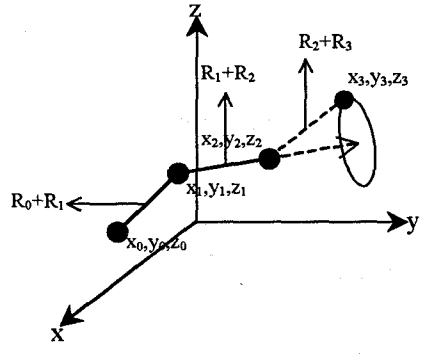
Birleşmenin ardından B kutusuna ait tüm polimerler ve monomerler A kutusuna ait olur ve A kutusunun boyutu genişler.

Tüm veri tiplerine ait burada anlatılmayan üyeler ve üye fonksiyonları kütüphanenin başlık dosyası "polymer.h"dan görülebilir.

5.2. Zincir Mimarisinin Oluşturulması

Zincirlerin yapısı birbirine tek noktadan değen küreler şeklindedir. Zincir mimarisi koordinat dönüşümleri ve eksenler etrafında dönmeler ile oluşturulabilir. Ancak önemli nokta işlemleri mümkün olduğu kadar sadeleştirme gereğidir. Şu an da hesaplanan her yeni koordinat için sadece bir kosinüs ve bir sinüs trigonometrik fonksiyonları kullanılmaktadır.

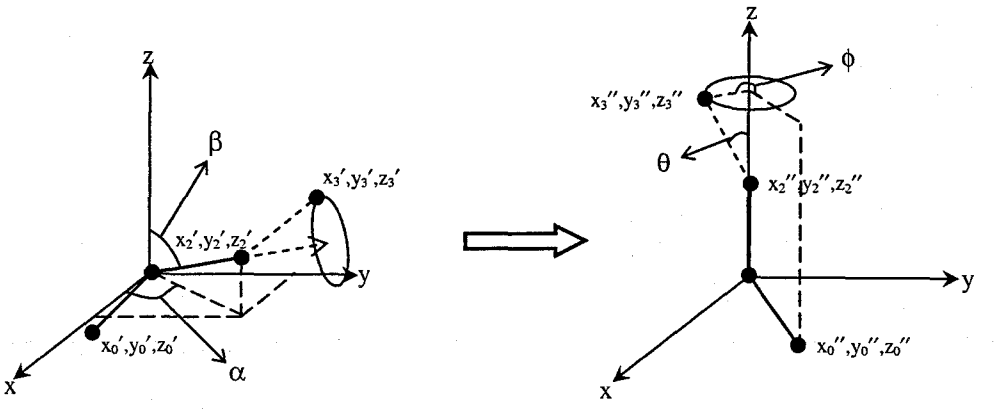
İlk önce problemin tarif edilmesinde fayda vardır. Şekil 5.4'te bilinen noktalar ve hesaplanması gereken nokta gösterilmiştir.



Şekil 5.4: Zincirin koordinat eksenlerine göre gösterimi. (x_3, y_3, z_3) hesaplanacak olan noktadır. R_0, R_1, R_2, R_3 monomerlerin yarıçaplarıdır.

[↓] Eğer lineer polimerler serbest bağlı zincirler olarak tasarlanırsa periyodik bağ koşulları kolayca programlanabilir.

Noktalar arası uzaklıklar karşılıklı iki monomerlerin yarıçaplarının toplamı kadardır. İlk yapılacak koordinat dönüşümü (x_1,y_1,z_1) noktasını orijine taşınmasıdır. Arkasından (x_2',y_2',z_2') z-ekseni ile çakışacak biçimde tüm sistem önce z-ekseni çevresinde $-\alpha$ açısı kadar daha sonrada y ekseni etrafında $-\beta$ açısı kadar döndürülür. Şekil 5.5'te açılar ve işlemler gösterilmiştir.



Şekil 5.5: Koordinat dönüşümlerinin gösterimi. θ belirli olan bağ açısı ϕ ise rasgele belirlenecek olan dihedral açıdır.

Şekil 5.5'ten görülebileceği gibi rasgele üretilecek (x_3'',y_3'',z_3'') noktasının koordinatları artık kolayca bulunabilir. Daha sonra yapılacak işlem üretilen bu noktanın yapılan dönüşümlerin tersi yapılarak yerine taşınmasıdır. (x_3'',y_3'',z_3'') noktasının üretilmesi hariç, yapılan tüm işlem aşağıdaki eşitlikle özetlenebilir.

$$\begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = Z(\alpha)\times Y(\beta)\times \begin{bmatrix} x_3'' \\ y_3'' \\ z_3'' \end{bmatrix} + \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{5.1}$$

$Z(\alpha)$ ve $Y(\beta)$ sırasıyla z ve y eksenleri çevresinde dönme matrisleridir. Bu matrisler aşağıda verilmiştir.

$$Y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$
$$Z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

Dikkat edilirse dönme matrisleri için gerekli olan açıların kosinüs ve sinüsleri bilinen koordinatlar cinsinden yazılabilir. Bu durum Şekil 5.5'te açık şekilde görülmektedir. Aşağıda bilinen koordinatlar cinsinden gerekli trigonometrik fonksiyonların ifadeleri verilmiştir.

$$\cos\beta = \frac{z_2'}{\sqrt{x_2'^2 + y_2'^2 + z_2'^2}}$$
$$\sin\beta = \frac{\sqrt{x_2'^2 + y_2'^2}}{\sqrt{x_2'^2 + y_2'^2 + z_2'^2}} \tag{5.3}$$

$$\cos\alpha = \frac{x_2'}{\sqrt{x_2'^2 + y_2'^2}}$$
$$\sin\alpha = \frac{y_2'}{\sqrt{x_2'^2 + y_2'^2}} \tag{5.4}$$

Bu koordinatlar ise ilk koordinatlar cinsinden aşağıdaki şekilde yazılır.

$$x_2' = x_2 - x_1$$
$$y_2' = y_2 - y_1 \tag{5.5}$$
$$z_2' = z_2 - z_1$$

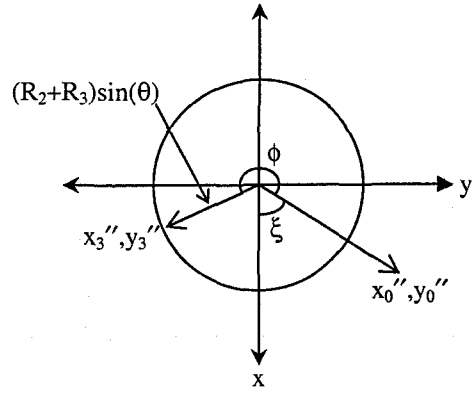
Bu noktada denklem (5.1)'de (x_3'',y_3'',z_3'') noktası hariç bilinmeyen değer kalmamıştır. Bu noktanın da z koordinatı Şekil 5.5 ve Şekil 5.4'ten takip edilebileceği gibi kolayca hesaplanabilir.

$$z_3'' = R_1 + R_2 + (R_2 + R_3)\cos\theta \tag{5.6}$$

(x_3'',y_3'',z_3'') noktasının x ve y koordinatlarının üretilmesi için rasgele (veya kullanıcının verdiği dağılım fonksiyonuna göre) bir ϕ açısı belirlenir. ϕ açısının nereden itibaren ölçüleceğinin bulunması için ise (x_0'',y_0'',z_0'') noktasının koordinatlarının bulunması gereklidir. Serbestçe dönen zincir için bu noktanın hesap edilmesine gerek yoktur. Çünkü konformasyon (x_0,y_0,z_0) noktasındaki monomerden bağımsızdır. (x_0,y_0,z_0) noktasının döndürülmüş hali (x_0'',y_0'',z_0'') denklem (5.2), (5.3) ve (5.4)'te verilen dönüşüm matrisleri ile kolayca bulunabilir. Yalnızca dönmeler ters tarafa doğrudur; $\sin(-\varphi) = -\sin(\varphi)$ ve $\cos(-\varphi) = \cos(\varphi)$ olduğundan aynı eşitlikler sinüsteki eksi farkına dikkat edilerek kullanılabilir.

$$\begin{bmatrix} x_0'' \\ y_0'' \\ z_0'' \end{bmatrix} = Y(-\beta) \times Z(-\alpha) \times \begin{bmatrix} x_0 - x_1 \\ y_0 - y_1 \\ z_0 - z_1 \end{bmatrix} \tag{5.7}$$

Şekil 5.6'da hesaplamada gelinen nokta gösterilmektedir. Tepeden projeksiyon durumu daha anlaşılır kılmaktadır.



Şekil 5.6: Dihedral açının tepeden gösterimi

(x_3'', y_3'', z_3'') noktasının x ve y koordinatlarının üretilmesi için dağılım fonksiyonuna göre belirlenen ϕ açısına, ξ açısının eklenmesi gerekmektedir.

$$\begin{aligned} x_3'' &= \left[(R_2 + R_3)\sin\theta\right]\cos(\phi + \xi) \\ y_3'' &= \left[(R_2 + R_3)\sin\theta\right]\sin(\phi + \xi) \end{aligned} \tag{5.8}$$

Yukarıdaki denklemde hesaplanması gereken sadece ξ açısı kalmıştır. (x_0'', y_0'') noktası bilindiği için ξ ile ϕ açıları toplamının, kosinüs ve sinüsleri de aşağıdaki gibi hesaplanabilir.

$$\begin{aligned} \cos(\phi + \xi) &= \frac{x_0''}{\sqrt{x_0'' + y_0''}}\cos\phi - \frac{y_0''}{\sqrt{x_0'' + y_0''}}\sin\phi \\ \sin(\phi + \xi) &= \frac{y_0''}{\sqrt{x_0'' + y_0''}}\cos\phi + \frac{x_0''}{\sqrt{x_0'' + y_0''}}\sin\phi \end{aligned} \tag{5.9}$$

(x_3'', y_3'', z_3'') noktası başta verilen denklem (5.1)'in içine yerleştirildiğinde bulunmak istenen (x_3, y_3, z_3) noktasının koordinatları hesaplanmış olur.

ϕ açısının dağılım fonksiyonuna göre belirlenmesi ise geri çevirme metodu (rejection method) [12] ile yapılır. Bu yöntem düzgün dağılıma sahip bir rasgele sayı dizisini

istenilen dağılımda rasgele sayı dizisine çevirir. Ancak bir kısım rasgele sayı harcanacaktır. Yöntem kısaca şöyle çalışır. İlk önce rasgele bir ϕ açısı üretilir bu açı kullanıcının sağladığı dağılım fonksiyonuna gönderilir ve bu fonksiyon geriye 0 ile 1 arasında bir sayı döndürür. 0 ile 1 arasında rasgele bir sayı atılır ve atılan bu sayı dağılım fonksiyonunun döndürdüğü sayıdan küçükse açı kabul edilir büyükse yeni bir ϕ açısı üretilir ve işlem tekrar edilir. Daha az rasgele sayı harcanması için kullanıcının dağılım fonksiyonunun global maksimumunu 1 olacak şekilde ayarlaması gereklidir.

5.3. Polimerlerin Hareket Ettirilmesi

Lineer polimerler ve monomerler çeşitli şekillerde hareket ettirilebilirler. Bütün yapılar herhangi bir vektör doğrultusunda ötelenebilirler. Bütün yapılar için öteleme fonksiyonunun adı aynıdır.

```
box kutu1(100,100,100);
monomer A;
linearpolymer B;
A.PutRandomCoorIn(kutu1);
B.CreatePolymer(20,1,60,kutu1);
A.Translation(polymer::RanPointOnSphere(1.0));
B.Translation(polymer::RanPointOnSphere(1.0));
```

Yukarıdaki örnek programda lineer polimer ve monomer kutu içinde rasgele bir yöne doğru ötelenmişlerdir. "RanPointOnSphere(double)" kütüphane tarafından sağlanan, yarıçapını kullanıcının belirlediği ve merkezi orijinde olan küre üzerinde rasgele nokta üreten bir fonksiyondur.

Bunun yanında lineer polimerlere özgü hareketler de vardır. Bunlar de Gennes modeline göre sürüngen hareketi ve dihedral açı değişimleridir.

```
B.Reptation(HEAD);
B.Rotation(5,6,30);
```

Yukarıdaki örnek programın ilk satırında B lineer polimeri, baş tarafına doğru bir bağ uzunluğu kadar sürüngen hareketi yapmıştır. İkinci satırda ise polimerin 6. monomerinden sonraki parçası, 5. ve 6. monomeri bağlayan bağ çevresinde 30° döndürülmüştür.

Kutu içinde konulan nesneler birbirleri ile üst üste gelemezler. Esasen her hareket ettirme rutini sisteme yapılan bir öneri mahiyetindedir; hareket edilmek istenen hacmin dolu veya boş olmasına göre kabul veya ret edilebilir. Hareket rutini bir "if" yapısı içine konursa hareketin yapılıp yapılamadığı kontrol edilebilir.

```
if(!(B.Reptation(TAIL)){
  //Hareket yapılamadıysa program buraya yazılacak kodu çalıştırır
}
```

Sürüngen hareketinde, eğer belirlenen koordinatta başka bir monomer varsa yeni bir koordinat belirlenir. Bu işlem kullanıcının belirlediği bir sayı kadar tekrarlanır. Maksimum tekrar sayısına ulaşılmış ve halen hareket edilebilecek bir koordinat bulunamamışsa hareket ret edilir fonksiyon mantıksal yanlış (false) geri döndürür. Büyük açılarla yapılan dönme hareketleri zincirin kendisini keserek dönmesine karşılık gelebilir buna kullanıcının dikkat etmesi gereklidir. Başlangıç açısında ve son açıda üst üste gelme olmadığı için hareket kabul edilir ancak dönme sürekli yapıldığında (ufak açılarla adım adım döndürerek) toplamda aynı açı kadar dönme mümkün olmayabilir.

Hareket eden polimerler ve monomerler diğer polimer ve monomerlerin işgal ettikleri hacimlere giremezler. Kütüphane dahilindeki algoritma bir koordinatın bir monomer tarafından işgal edilip edilmediğini kontrol etmek için basitçe tüm monomer bilgilerini taramaktadır. Bu sırf işlem gücüne dayalı bir algoritmadır. Daha akıllı algoritmalarla önemli performans artışları sağlanabilir. Alternatif bir yöntem istatistik değerleri kullanarak kesişmesi imkansız olan polimerlerin monomerlerinin kontrolünün atlanması olabilir. Örnek olarak kontrol edilecek koordinatın, polimerin ağırlık merkezinden uzaklığı, polimerin baş-son mesafesi ile karşılaştırılabilir. Eğer koordinat yeterince uzaksa polimerin tek tek monomer koordinatlarına bakılmaz. Burada önemli nokta hareket sırasında ağırlık merkezi bilgisinin sürekli yenilenmesi

gereğidir. Daha değişik bir yaklaşım ağırlık merkezi yerine polimerin baş, son ve orta monomerlerinin kontrol edilecek koordinata uzaklıklarının hesaplanması olabilir. Bu şekilde ağırlık merkezi bilgisi hesabı yapılmasına gerek kalmaz.

Bir kutu için üst üste gelme kontrolü

```
kutu1.DisableOverlapCheck();
```

satırı ile iptal edilebilir.

5.4. Bağlılık Algoritması

Birbirleriyle rasgele bağlanmış monomerlerin bağlılık kontrolü mantıksal yapı olarak kütüphanedeki en karışık algoritmadır. Kendisini çağıran (recursive) fonksiyonlara ihtiyaç duyar. Bağlılık kontrolü sadece monomer yapısıyla çalışan bir kullanıcı için çok gereklidir. Özellikle monomer yapılarıyla perkolasyon tarzında bir çalışma programlanması monomerler arasında bağlılık kontrolünü gerekli kılar.

Bağlılık kontrolünü gerçekleştirmenin iki yolu vardır. İki yöntem arasında seçim karşılaşılan probleme bağlıdır. Yöntemler aşağıda sıralanmıştır:

- Monomerlerin bağlantısı bittikten sonra iki monomerin bağlılığının kontrolü kendini çağıran bir fonksiyon yardımı ile bulunur. Her bağlılık kontrolü için fonksiyonun tekrar çalışması gerekir. Bağlantı işlemi hızlı ancak kontrol işlemi yavaştır
- Monomerler bağlanırken özel bir yapı yardımı ile gruplaşmalar tutulur. Bu yöntemde bağlanma yavaş ancak bağlılık kontrolü çok hızlıdır. Bağlılık kontrolü sadece iki grup numarasının aynı olup olmadığının kontrolünden ibarettir.

Her ne kadar ikinci yöntem için bağlanmanın yavaş olduğu ifade edilmiş olsa da bazı bağlantılar yine çok hızlı yapılabilir. İkinci yöntem oldukça iyi optimize edilmiştir. Ayrıca ikinci yöntem birinci yöntemde kullanılan algoritmayı da içinde barındırır.

Birinci algoritmayı kullanan fonksiyonun akışı aşağıda verilmiştir.

1. A ve B monomerleri bağlılık kontrolü için alındı.

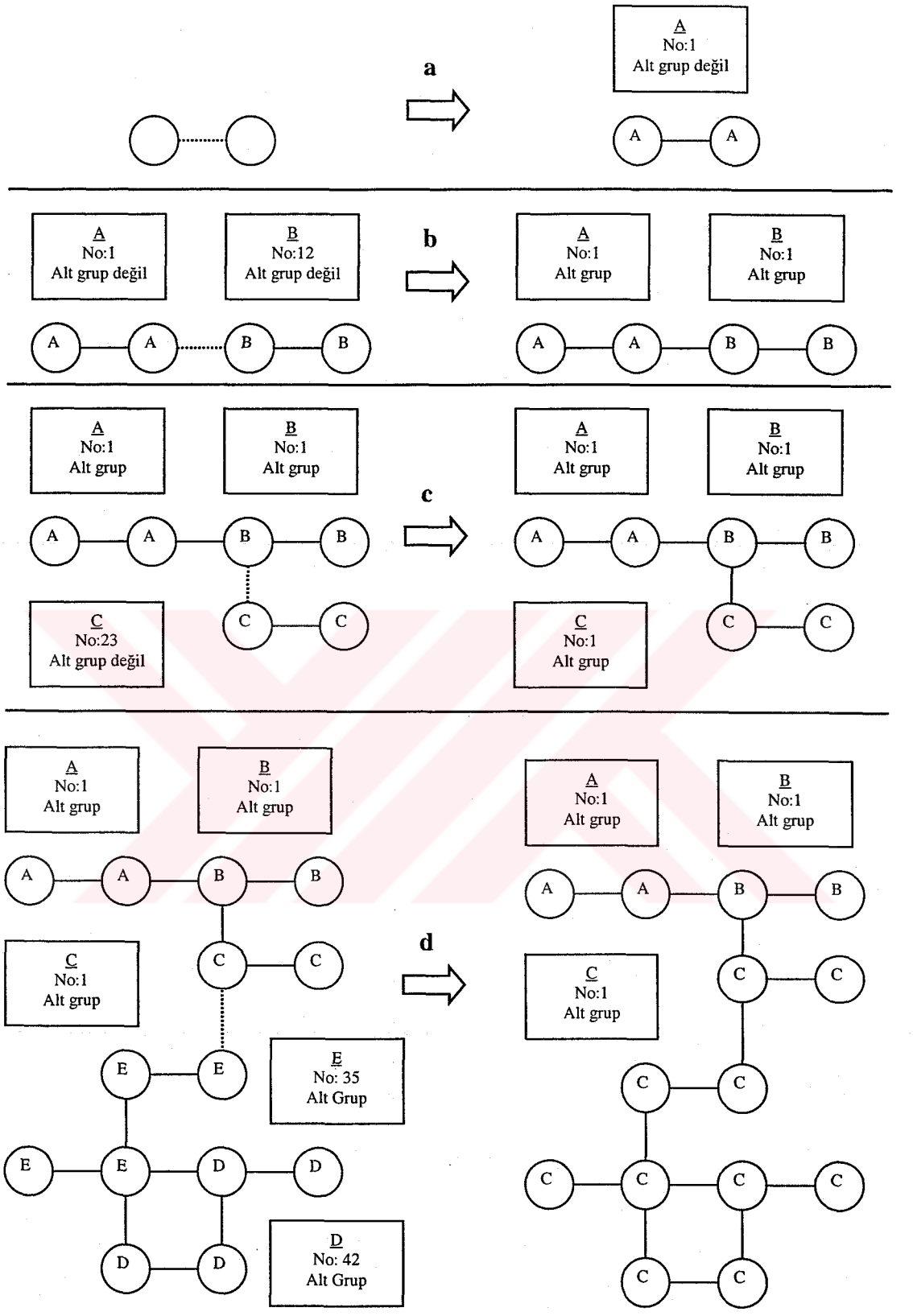
2. A monomerinin komşuları üzerinden döngü başlat.
    a. B monomerinin kimlik numarası ile komşunun kimlik numarasını karşılaştır. Eğer aynı ise mantıksal doğru (return true) geri döndür.
    b. Komşunun kimlik numarasını kontrol edilmişler listesinde ara eğer listede mevcutsa döngüyü kaldığı yerden tekrar döndür. Eğer mevcut değilse komşunun kimlik numarasını kontrol edilmişler listesine kaydet.
    c. Komşu ve B monomeri için fonksiyonu tekrar çağır. Fonksiyon mantıksal doğru döndürüyorsa mantıksal doğru döndür (return true).
3. A monomerinin komşuları üzerinden döngü sonu.
4. Mantıksal yanlış geri döndür (return false).

Algoritma ifade olarak kısadır yalnız yapılması gereken işlemler özel veri tipleri gerektirir. Öncelikle komşuların kimlik numaralarının tutulması için genişleyebilen bir diziye ihtiyaç vardır. Fonksiyon monomerleri gösteren işaretçileri giriş olarak alır. Fonksiyonun komşu ve B monomeri için kendini tekrar çağırması monomer tipinin sahip olduğu genişleyebilen monomer listesi (bağlı olduğu monomerlerin işaretçileri) sayesinde mümkün olur. Herhangi bir kısıtlamaya sahip değildir.

İkinci yöntem birbirine bağlı monomerlerin bilgisinin hafızada tutulması ve anında erişilebilmesi için oldukça zarif bir yöntem kullanır. Ancak bağlantıların koparılmasını desteklemez. Bağlantıların koparılmasının desteklenmesi yakın zamanda sağlanacaktır. Bu yöntem kullanımı bir anahtara bağlanmıştır. Bunun sebebi grup bilgisinin her zaman tutulmasına ihtiyaç olmayışıdır. Böylelikle hafızadan tasarruf edilmiş olur. Bu yöntemi etkinleştirmek için "polymer.h" dosyasının başına

```
#define ONLYCONNECTANDGROUP
```

tanımlaması yapılmalıdır. Bu tanımlama yapıldığında özel bir grup yapısı kullanıma girer. Aynı zamanda monomer veri tipinin grup işaretçisi ve grup yapısının özelliklerinden faydalanan özel fonksiyonları aktif hale geçerler. Algoritma Şekil 5.7'nin yardımı ile anlatılacaktır.



Şekil 5.7: Gruplama algoritmasının çalışma şeması

"Grup" veri tipiden nesneler, polimer veri tiplerinde olduğu gibi bir kimlik numarasına sahiptirler. Ancak grupların bu kimlik numarası değiştirilerek birbirleriyle aynı yapılabilir. Yalnız kullanıcının bu işlemi yapmaya yetkisi yoktur. Kimlik numarası değişimi bağlantılar kuruldukça kütüphane tarafından otomatik yapılır. Şekil 5.7'de görülen dikdörtgenler grupları temsil etmektedir. Dikdörtgenlerin içindeki altı çizili ilk harf grubun hafızadaki adresini temsil etmektedir. Onun altında grup kimlik numarası ve alt grup olup olmadığını gösteren bayrak değişken vardır. Her monomer bir grubun hafıza adresini gösteren bir işaretçiye sahiptir. Grup bilgisi monomer içinde saklanmaz. Gruplar monomerler tarafından ortak kullanılır. Ortak kullanım fikri bağlantıların hızlı yapılmasını sağlar. Şekil 5.7'de görülen daireler ise monomerlerdir. Monomerlerin içerisindeki harfler grup işaretçisinin değerini gösterir. Şekil 5.7'de karşılaşılacak durumların hepsi gösterilmiştir.

Şekil 5.7a'da hiç bir gruba üye olmayan iki monomer bağlanacaktır. Bu durumda algoritma dinamik olarak hafızda bir grup oluşturur ve grubun hafıza adresini iki monomere de kaydeder. Bir gruba üye olan monomere hiçbir gruba üye olmayan monomer bağlanmak istenirse yapılacak işlem, hiçbir gruba üye olmayan monomerin grup işaretçisine bir gruba üye olan monomerin grup işaretçisini kopyalamaktır. Eğer bağlanmak istenilen iki monomerin grup numarası aynı ise (bu durumda grup kapalı döngü içerecektir) hiç bir şey yapılmadan fonksiyondan çıkılır. Bu durumun önemi kendini çağırma işleminin temel durumu olup algoritmanın sonsuz döngüye girmesini engellemesinden kaynaklanmaktadır. Şekil 5.7b'de ise ikisi de bir gruba üye iki monomer bağlanmak istenmektedir. İkisinin grubu da alt grup olarak işaretlenmemiştir. Alt grup bilgisinin gerekliliği algoritmanın sonraki adımlarında daha iyi anlaşılacaktır. Bu durumda monomerlerden birinin gösterdiği adresteki grubun kimlik numarası diğer grubun kimlik numarası ile değiştirilir. Bu sayede tüm monomerlerin grup bilgisi değişmiş olur. Halbuki grup bilgisi monomerlerin içinde saklansaydı tüm monomerlerin grup bilgisini teker teker değiştirmek gerekecekti. Artık grup iki ayrı adresteki alt gruplara bölündüğü için iki grupta alt grup olarak işaretlenir. Şekil 5.7c'de alt gruba üye monomerle, alt grup olmayan bir grubun üyesi monomerin bağlantısı görülmektedir. Bu durumda alt grup olmayan grubun kimlik numarası değiştirilir ve bu grupta alt grup olarak işaretlenir. Şekil 5.7d ise işlem gücüne ihtiyaç duyulan tek durumu göstermektedir. Alt gruba üye iki monomerin

bağlanması ancak bir tarafa bağlı tüm monomerlerin grup bilgilerinin değiştirilmesi ile mümkündür. Bunun için ilk yöntemde kullanılan kendini çağıran fonksiyonun algoritmasına benzeyen bir yöntem kullanılır. İlk önce bağlanmak istenilen taraf seçilir (kural olarak hep ikinci taraf). Daha sonra bu monomerin grup işaretçisi diğer monomerin grup işaretçisi ile değiştirilir. Monomerin komşuları üzerinden bir döngü başlatılır. Komşunun kimlik numarası eğer monomerin kendi kimlik numarası ile aynı ise döngü bir sonraki komşu ile devam ettirilir. Bu durum sadece monomerin kendisine bağlı olması gibi çok özel bir durumda ortaya çıkmaktadır. Daha sonra tüm anlatılan fonksiyon monomerin kendisi ve komşusu için tekrar çağırılır. Bu şekilde tüm monomerlerin grupları değiştirilmiş olur.

İki monomerin bağlı olup olmadığının kontrolü sadece monomerlerin gruplarının kimlik numaralarının aynı olup olmadığı kontrol edilerek yapılabilir.

Birden fazla monomerin bağlılık kontrolü de mümkündür ancak girilen monomer sayısının bildirilmesi gerekmektedir. Aşağıdaki örnek programda bağlılık kontrollerinin nasıl yapıldığı gösterilmektedir.

```
monomer1.IsConnected(monomer2);    //Birinci yöntemle bağlılık kont.
IsConnected(2,monomer1,monomer2); //İkinci yöntemle bağlılık kont.
IsConnected(5,monomer1,monomer2,monomer3,monomer4,monomer5);
//İkinci yöntemle birden fazla monomerin bağlılık kontrolü
```

Bu algoritma doğru çalışmakla beraber bir teknik sorun vardır. Bu algoritmanın dinamik grup yaratmasından kaynaklanan bir sorundur. Eğer bir program dinamik olarak hafızadan yer ayırıyorsa bu alanı programı terk etmeden önce silmelidir. Eğer boşaltmazsa "hafıza akıntısı" denilen sorun meydana gelir. Her ne kadar hafıza yönetimleri daha sıkı olan Windows NT ve Windows 2000 gibi sistemler bu ayrılmış hafızayı program bitiminde boşaltabilseler de Windows 9x ve devamı olan Windows ME sistemlerinde bu alan ayrılmış olarak kalabilir. Bu da arka arkaya programın çalıştırılması durumunda sistem kaynaklarının tükenmesine neden olur. Şekil 5.7d'den görülebileceği gibi bazı gruplarla bağlantı kopmaktadır ancak bu gruplar hafızadan silinmezler. Grup yapısı, tüm oluşturulan grupların hafızadan silinmesi için

özel bir fonksiyon içerir. Bu fonksiyon program bitiminde çağrılmalı ve hafıza boşaltılmalıdır. Fonksiyon "`group::FlushGroups()`" şeklinde çağrılır.

6. SİMÜLASYON SONUÇLARI

6.1. Basamaklı Polimerizasyon

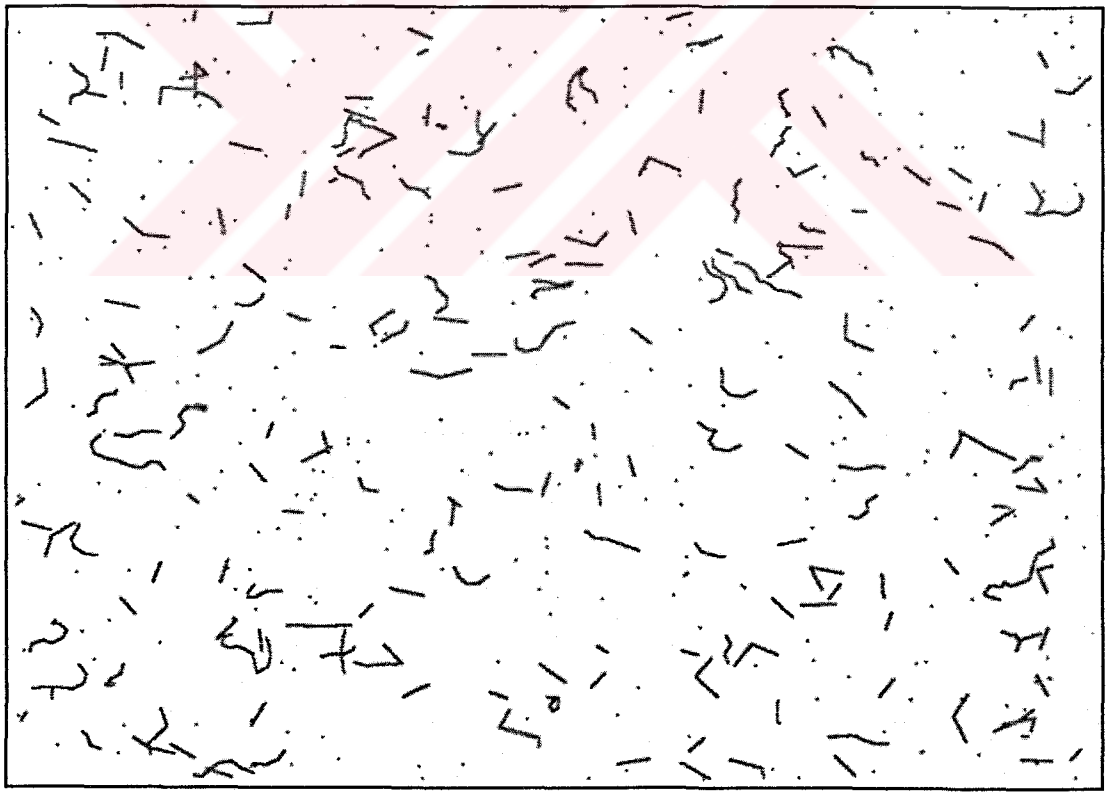
Basamaklı polimerizasyon simülasyonu için bir kutu içinde polimerlerin rasgele bağlantı yapmaları gerekmektedir. Tüm parçacıklar tek monomere sahip lineer polimerler olarak oluşturulur bu sayede hepsi otomatik olarak kendilerine başka polimerleri (bağlanılan polimer de tek monomerden oluşuyor olabilir) bağlama yeteneğine sahip olurlar.

Aşağıdaki program basamaklı polimerizasyon simülasyonudur.

```
int i,j;
int polymernum=1000;
linearpolymer *polymers;
box box1(150,150,150);
polymers=new linearpolymer[polymernum];
for(j=0;j<polymernum;j++)polymers[j].CreatePolymer(1,1,45,box1);
for(i=0;i<MAXITER;i++){//Simülasyon ana döngüsü
  for(j=0;j<polymernum;j++){
    polymers[j].Translation(polymer::RanPointPointOnSphere(1.0));
    polymers[j].Reptate(RANDOM);
    polymers[j].AutoConnect();
  }
}
```

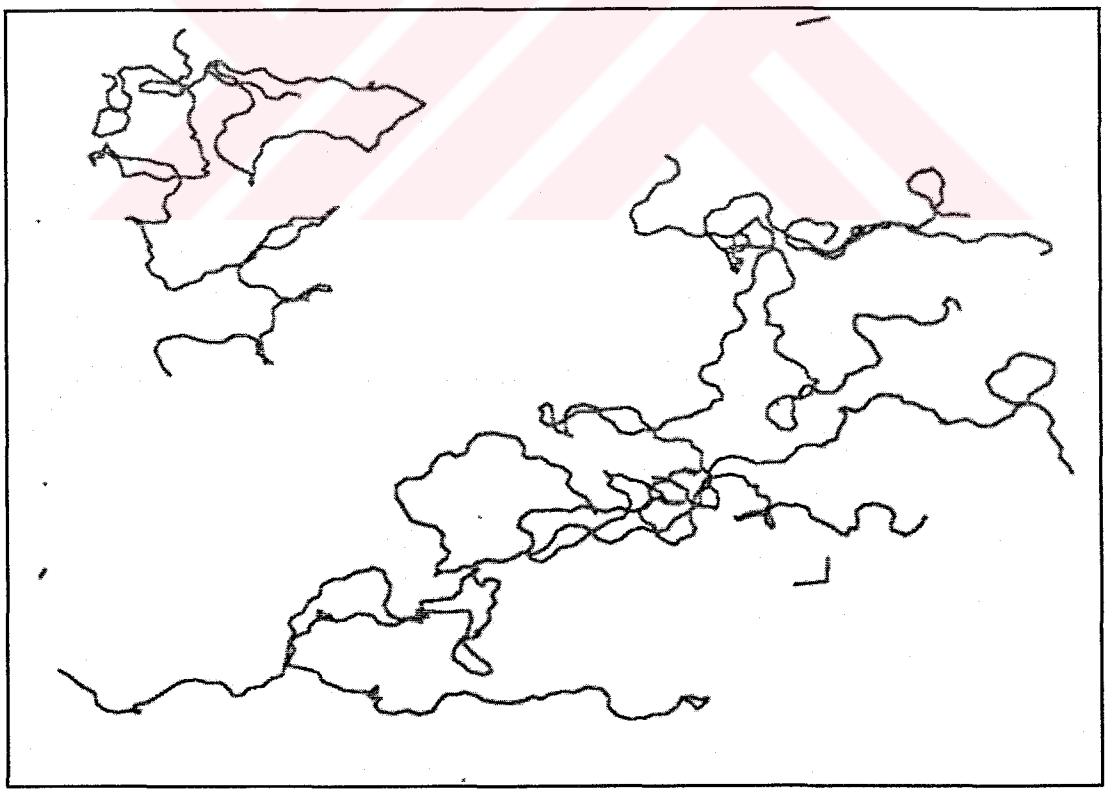
Yukarıdaki simülasyonda 150 birim ayrıt uzunluğuna sahip küpün içerisinde 1000 adet tek monomerden oluşan polimer oluşturulmaktadır. Daha sonra bunlar hareket ettirilmektedir. Birbirine bağ yapma mesafesi kadar yaklaşanlar otomatik olarak bağ kurmaktadır. Her iterasyonda polimerler rasgele bir yönde 1 birim ötelenmekte ve sürüngen modeli için uygun uzunluğa gelmiş (4'ün üzerinde monomere sahip) polimerler bu modele göre hareket ettirilmektedir. Hareketlerin hepsi daha evvel

belirtildiği gibi öneri mahiyetindedir. Hareket edecek yer yoksa veya hareket o an için tanımlı değilse yapılmamaktadır. Bu kullanıcının ekstra kontroller yapmadan simülasyonu kolayca yazmasını sağlar. Yukarıdaki simülasyon basit olması için bazı fiziksel gerçekleri göz ardı etmiştir. Örneğin zincirler uzadıkça çevresindeki moleküllerle daha fazla sürtünmeye başladığı için hareket etme olasılığı düşer. Bunu kullanıcı yine Monte Carlo metodu ile simüle edebilir. Zincirlerin boyuna bağlı olarak 0 ile 1 arasında sayı üreten (fiziksel olarak zincir uzadıkça 0'a daha yakın sayılar üretmesi gerekir) bir fonksiyon yazılır. 0 ile 1 arasında rasgele bir sayı üretilir. Eğer üretilen sayı fonksiyonun değerinden küçükse polimer hareket ettirilir. Bu öneri simülasyonu karmaşıklaştırmak için yapılabileceklerden bir tanesidir. Görüldüğü gibi kütüphane kullanıcının zincirleri ve hareketleri oluşturmak için zaman kaybetmesini önleyerek asıl yapmak istenilen işe daha fazla vakit ayrılmasını sağlamaktadır. Şekil 6.1'de yukarıdaki programla yapılan simülasyonun gerçek zamanlı gösteriminden alınan resimler sıralanmıştır. Simülasyon Pentium III tabanlı Celeron 1200 MHz bir bilgisayarda 20 dakika sürmüştür.



Şekil 6.1a: Simülasyon başladıktan kısa bir süre sonra sistemin resmi.

Şekil 6.1b: Monomerlerin tamamına yakınının tükendiği durumda sistemin resmi.



Şekil 6.1c: Sistem tek bir zincir olana dek simülasyon sürdürülebilir. Yukarıdaki son resim simülasyon durdurulmadan hemen önce alınmıştır. Resimler sistemin 2 boyuta izdüşümüdür.

6.2. Zincir Katılma Polimerizasyonu

Bir başka yapılabilecek polimerleşme simülasyonu zincir katılma polimerizasyonudur. Zincir katılma polimerizasyonu için büyümeye başlayacak monomerin başlatıcı ile reaksiyona girmesi gereklidir. Dolayısıyla başlatıcı sayısından daha fazla polimer oluşması imkansızdır. Kimyasal olarak her başlatıcı iki polimerin oluşmasını sağlayabilir. Ancak simülasyondaki başlatıcı tanımı biraz değişiktir. Başlatıcı olarak yine lineer polimer kullanılır. Yalnız lineer polimerler özel bir fonksiyonla başlatıcı olarak tanımlanırlar ve aktif edilene kadar hiç bir monomer veya polimerle birleşemezler. Aşağıdaki program zincir katılma polimerizasyonunu simüle etmektedir. Reaksiyona inhibitörler de katılmıştır. İnhibitörler tek bağlantı kapasitesine sahip monomerler olarak yaratılabilirler. Simülasyonda 20 inhibitör, 20 başlatıcı (kimyasal olarak 10 başlatıcıya karşılık gelir), 1000 monomer kullanılmıştır.

```
int i,j,k;
int initnum=20;
int monomernum=1000;
int inhibnum=20;
box box1(150,150,150);
linearpolymer *polymers=new linearpolymer[initnum];
monomer *monomers=new monomer[monomernum];
monomer *inhibitors=new monomer[inhibnum];

for(j=0;j<initnum;j++)polymers[j].MakeInitiator(1,45,box1);
for(j=0;j<monomernum;j++)monomers[j].PutRandomCoorIn(box1);
for(j=0;j<inhibnum;j++){
  inhibitors[j].ChangeFreeCLCap(1);
  inhibitors[j].PutRandomCoorIn(box1);
}

k=0;
for(i=0;i<MAXITER;i++){//Simülasyon ana döngüsü
```

```
  if((k<initnum)&&(i%1000==0)){
    polymers[k].ActivateInitiator();
    k++;
  }
  for(j=0;j<monomernum;j++)
    monomers[j].Translation(polymer::RanPointOnSphere(1.0));
  for(j=0;j<inhibnum;j++)
    inhibitors[j].Translation(polymer::RanPointOnSphere(1.0));
  for(j=0;j<k;j++)polymers[j].AutoConnect();
  //j<k yerine j<initnum yazılabilir ama aktif olmayan polimer
  //zaten bağlantı kuramayacağı için otomatik bağlanma isteği
  //ret edilir. j<k yazılması sadece performansı artırır
  //bir gereklilik değildir
}
```

Simülasyonda her 1000 adımda, bir başlatıcı aktif hale gelmektedir. Bu simülasyonda oluşan zincirler hareket ettirilmemiştir. Ancak basamaklı polimerizasyondan da hatırlanabileceği gibi zincirlerin hareket ettirilmesi sadece bir satırlık kodla yapılabilir. Program kodu büyük ölçüde kendini anlatmaktadır. Bu da programlamayı kolaylaştırmaktadır. Bunlara ek olarak istenirse simülasyon ana döngüsü içerisinde, belirli bir iterasyondan sonra veya kullanıcının belirlediği bir şart meydana geldiğinde, kutuya (box) ekstra monomer, lineer polimer, inhibitör ve başlatıcı dinamik olarak katılabilir. Yapılacak tek şey bunları yaratmak ve kutu içine yerleştirilmeleri için komut vermektir. Başka bir yöntem, eklenecekleri başka bir kutuya yerleştirmek ve kutuyu ana kutuyla birleştirmektir. Bu yolla kutunun hacmi de değiştirilebilir.

6.3. Zincir Difüziyonu

Polimer fiziğinde polimer moleküllerinin bir ara yüzeyden difüze etmesi önemli bir konudur. Lateks film oluşumu sırasında polimer moleküllerinin difüziyonu son yıllarda çeşitli teknikler kullanılarak geniş bir biçimde incelenmektedir. Lateks filmler başlangıçta ufak polimer küreciklerden oluşur. Bu kürecikler, genellikle su içinde pelteli bir şekilde yayılırlar. Aşağıdaki simülasyonda iki ayrı kutuda zincirler oluşturulmakta daha sonra bu kutular birleştirilerek bir ara yüzeyle birbirinden

ayrılmış iki polimer karışımı elde edilmektedir. Daha sonra bu zincirlerin sürüngen hareketiyle ara yüzeyden difüziyonu sağlanmaktadır [13].

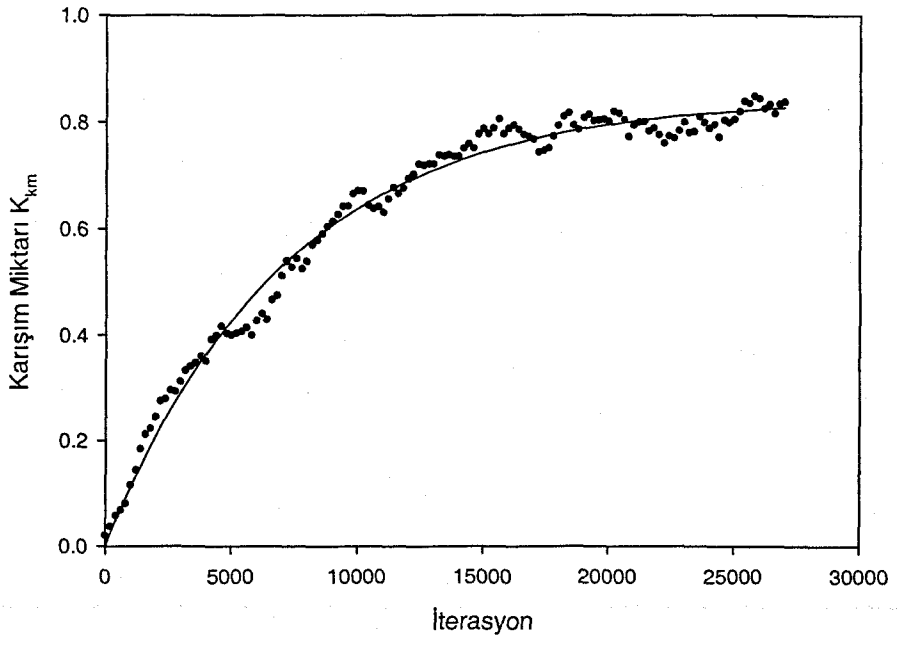
```
const int polymernum=50;
int monomernum=50;
box anakutu(40,80,80),sagkutu(40,80,80);
linearpolymer polimersol[polymernum],polimersag[polymernum];
for(i=0;i<polymernum;i++){
  polimersol[i].CreatePolymer(monomernum,1.0,45.0,anakutu);
  polimersag[i].CreatePolymer(monomernum,1.0,45.0,sagkutu);
}
anakutu.MergeBox(sagkutu,XAXIS);
for(j=0;j<MAXITER;j++){//Simülasyon ana döngüsü
  for(i=0;i<polymernum;i++){
    polimersol[i].Reptate(RANDOM);
    polimersag[i].Reptate(RANDOM);
  }
}
```

Şekil 6.2'de, yukarıdaki programla yapılmış 3 simülasyondan elde edilen verilerin ortalaması ile oluşturulmuş karışım miktarının iterasyonla değişimi görülmektedir. İterasyon simülasyon ana döngüsündeki "j" değişkeninin aldığı değerdir. Karışım miktarı ise iki grup polimerin kütle merkezlerinin birbirlerine olan uzaklığı ile hesaplanmıştır ve aşağıdaki denklemle ifade edilir.

$$K_{km} = 1 - \frac{\left| \overline{r_{km}^{sağ}} - \overline{r_{km}^{sol}} \right|}{l_{kutu}/2} \tag{6.1}$$

Elde edilen sonuç üstel doyum karakteri göstermektedir. Simülasyon sonucu denklem (6.2)'ye uydurulmuştur.

$$y = a(1 - e^{-bx}) \tag{6.2}$$



Şekil 6.2: Bir ara yüzeyle birbirinden ayrılmış iki zincir grubunun karışımı. İnce çizgi simülasyon sonuçlarına uydurulan eğriyi göstermektedir. Uyum istatistikleri: $R= 0.99286926$, $R^2= 0.98578936$

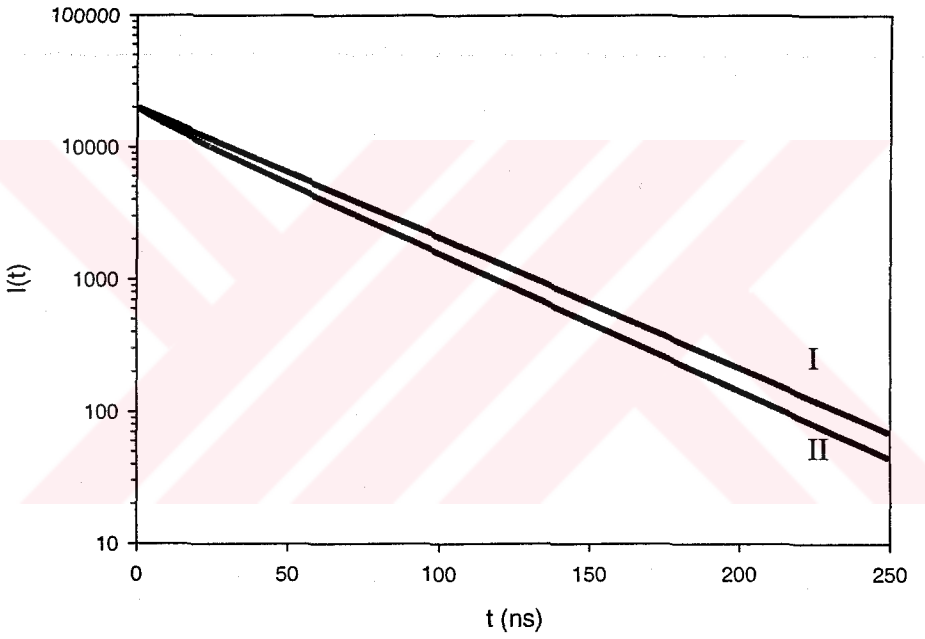
6.4. Donör Bozunma Profilleri

Zincir difüziyonu ağırlık merkezlerinin birbirine yaklaşması ile gözlenebileceği gibi enerji transferi mekanizması ile de gözlenebilir [14]. Bunun için ara yüzeyin bir tarafındaki polimerler akseptör diğer tarafındaki polimerler ise donörlerle işaretlenir ve kutu için donör bozunma profilleri hesaplanır [13]. Profiller denklem (3.13)'e uydurulup denklem (3.23)'e göre karışım miktarı hesaplanabilir. Aşağıda bir polimerin donör ve akseptörle nasıl işaretleneceği ve bir kutu için donör bozunma profilinin nasıl hesaplanabileceği gösterilmiştir.

```
// Bir polimerin (polimerA) N sayıda donörle etiketlenmesi
det::LabelwithDonor(polimerA,N);
// Bir polimerin (polimerB) N sayıda akseptörle etiketlenmesi
det::LabelwithAcceptor(polimerB,N);
// Etiketli polimerler içeren bir kutunda(kutu1) donör bozunma
// profili hesabı
smartarray< smartarray<double> > profil;
```

```
profil=det::CalculateDecay(kutu1)
```

Aşağıdaki grafik karışmış ve karışmamış zincirler için donör bozunma profilleri arasındaki farkı göstermektedir. Aşağıdaki grafiğin oluşturulması için 500 birim ayrıt uzunluğuna sahip bir küp içinde 150 monomerden oluşan 400 zincir kullanılmıştır. Zincirlerin yarısı iki adet akseptör diğer yarısı ise iki adet donör molekülü ile işaretlenmiştir. Förster mesafesi 26 birim, $\kappa^2 = 0.476$ ve $\tau_0 = 44$ ns alınmış $I(0)$ ise 2.0×10^4 olarak seçilmiştir. Donör bozunma profili 250 ns için birer nanosaniyelik kanallar halinde hesaplanmıştır.



Şekil 6.3: Donör bozunma profilleri. (I) karışmamış (II) ise karışmış duruma karşılık gelmektedir.

6.5. Sızma (Perkolasyon)

Her ne kadar kütüphane sızma simülasyonları yapmak için geliştirilmediyse de monomerlerin bağlanabilir yapısı kafes üzerinde sızma ve benzeri çalışmaların yapılmasına olanak sağlar. Özellikle bağlılık algoritması bu tür çalışmaların yapılmasında anahtar rol oynar.

Çok karmaşık yapılar oluşturulabileceği gibi 3 boyutun üstünde çalışmalar da kolayca programlanabilir. Karmaşık yapıdan kasıt her kafes noktasının istenilen

sayıda bağ yapma kapasitesine sahip olmasıdır. Aynı zamanda kafes üzerindeki noktaların her birinin bağlantı yapma kapasiteleri değişik olabilir. Bir başka değişle aynı sistem içerisinde değişik bağ kapasitesine sahip çapraz bağlayıcılar kullanılabilir. Bir kafes noktası, kafes üzerinde herhangi bir noktayla (örneğin iki yan komşusu) bağ kurabilir. Bu durumun fiziksel bir karşılığı olmasa da yapılabileceklerin serbestliğini ortaya koyması bakımından önemlidir. Kullanıcının yapması gereken istenilen boyutta bir monomer dizisi açmak, monomerlerin bağlantı kapasitelerini dilediği şekilde değiştirmek ve monomerleri rasgele veya belirli bir model çerçevesinde birbiriyle bağlamaktır. Bağlama işlemi bitiğinde iki monomerin bağlı olup olmadığını kontrol edebilir; bağları sayabilir. Bağlanma işlemi devam ederken de bağlılık kontrolü yapmak mümkündür. Bunun yanında bir monomere bağlı tüm monomerler listelenebilir. Birden fazla monomerin bağlılığı kontrol edilebilir (Aslında bu kontrol iki monomerin bağlılık kontrolünün arka arkaya tekrarlanmasından ibarettir). Aşağıdaki program parçalarında sızma gibi kafes üzerinde çalışmaları kolaylaştıracak kütüphane fonksiyonlarının kullanımları gösterilmektedir.

```
// 4 boyutlu hiper kübik kafes yapı oluşturulması
const int lb=5;
monomer latis[lb][lb][lb][lb];

// Önemli not: Yukıdaki dizi lokal bir değişken olduğundan
// "stack"te açılır. "Stack" alanı genelde sınırlıdır (1MB).
// Dolayısıyla dizinin boyu büyüdüğünde "stack overflow" mesajı
// ile program çalışmayabilir. Büyük diziler için "heap"ten alan
// açan new ve malloc kullanılmalıdır. Win32 sistemler için bir
// programın kullanabileceği maksimum hafıza miktarı 2 GB'tır.

// İki kafes noktasının birleştirilmesi i,j,k,l kafes üzerindeki
// koordinatı verir.
latis[i][j][k][l].Connect(latis[i+1][j][k][l]);

// Bir noktanın bağlantı yapma kapasitesinin değiştirilmesi
latis[i][j][k][l].ChangeFreeCLCap(4);
```

```
// Bağlantı kontrolü
latis[0][j][k][l].IsConnected(latis[lb][j][k][l]);

// Bir noktaya bağlı tüm noktaların kimlik numaralarının
// listelenmesi
smartarray<unsigned int> liste;
liste=latis[i][j][k][l].ListAllConnectedID();
```

KAYNAKLAR

[1] **Rapaport, D. C.,** 1995. The Art of Molecular Dynamics Simulation, Cambridge University Press, Cambridge.

[2] **Allen, M. P. and Tildesley, D. J.,** 1992. Computer Simulation of Liquids, Oxford Science Publications, Oxford.

[3] **Gedde, U. W.,** 1992. Polymer Physics, Kluwer Academic Publishers.

[4] **Tonelli, A. E. and Srinivasarao, M.,** 2001. Polymers from the Inside Out, John Wiley & Sons, Inc., Publications.

[5] **de Gennes, P. G.,** 1979. Scaling Concepts in Polymer Physics, Cornell Univ Press.

[6] **Strobl, G.,** 1997. The Physics of Polymers, Springer.

[7] **Förster, T.,** 1948. Intermolecular energy transfer and fluorescence, *Ann. Physik,* **2,** 55-75.

[8] **Lakowicz, J. R.,** 1983. Principles of Fluorescence Spectroscopy, Plenum Press.

[9] **Winnik, M. A., Pekcan, Ö. and Cruocher, M. D.,** 1990. Fluorescence Techniques in the Study of Polymer Colloids, Eds. Ottowill R. H. and Candau F., Kluwer Academic Publisher.

[10] **Tachiya, M. and Mozumder, A.,** 1974. Decay of trapped electrons by tunnelling to scavenger molecules in low-temperature glasses, *Chemical Physics Letters*, **28,** 87-89.

[11] **Astrachan, O. L.,** 1999. A Computer Science Tapestry: Exploring Computer Science with C++, McGraw-Hill.

[12] **Teukolsky, S. A., Vetterling, W. T. and Flannery B. P.,** 1992. Numerical Recipes in C, Cambridge University Press.

[13] **Tüzel, E., Kısacıkoğlu, K. B. and Pekcan Ö.,** 2002. Monitoring diffusion of reptating polymer chains by a direct energy transfer method: a Monte

Carlo simulation, *Macromolecular Theory and Simulation,* **11,** 678-686.

[14] Tüzel, E., Kısacıkoğlu, K. B. and Pekcan Ö., 2000. Simulation of interdiffusion in between compartments having heterogenously distributed donors and acceptors, *Polymer,* **41,** 7539.

ÖZGEÇMİŞ

10 Nisan 1977 tarihinde İstanbul'da doğdu. İlkokul öğrenimini Zühtü Paşa İlkokulu'nda orta ve lise öğrenimini ise Hüseyin Avni Sözen Anadolu Lisesi'nde tamamladı. 1995 yılında İstanbul Teknik Üniversitesi, Fen Edebiyat Fakültesi, Fizik Mühendisliği programında öğrenimine başladı. 2000 yılında mezun olup halen bu bölümün Yüksek Lisans programında eğitimine devam etmektedir.