1. GİRİŞ

1.1 Amaçlarımız

Tezimizin amacı başlangıçta şu anki şeklinde olmasa bile zaman içinde son halini almıştır.

Tezimiz Trakya Üniversitesi Bilgisayar Mühendisliği Bölümü'nde dilbilim alanında yapılması planlanan bir dizi çalışmanın tamamlanan ilk adımı olarak düşünülebilir. Çalışmamız anlambilim alanında yoğunlaşmıştır. Dolayısıyla dilbilim alanında ve özellikle de anlambilim kısmında kullanılan teorilerin bir incelemesi olarak düşünülebilir. Tezimizin yazımında da bu durum gözönüne alınmış ve anlatımda elden geldiğince (genel de olsa) bir bilgi verme amacı güdülmüştür. Belirttiğimiz durum tezimizin kullandığı dile de yansımıştır ve bu sebeple anlambilim alanındaki bizce en önemli iki yaklaşım incelenmiş ve anlatılmaya çalışılmıştır. Anlattığımız durum çok açık olmamakla birlikte tezimizin bir amacıdır diyebiliriz.

Tezimizin birincil amacı ise incelediğimiz teorilerden özellikle Montague yaklaşımı ile Türkçe için emir kipinde cümleleri kullanarak anlamsal bir analiz yapılması ya da daha doğru bir ifade ile en azından biçimsel gösterimlerinin elde edilmesidir. Tezimiz için yazdığımız programımız bu işi gerçekleştirmektedir.

Kullandığımız yaklaşımdan öte, benim kendi kişisel ilgim olarak da değerlendirilebilecek bir durum, "kapalı" ya da "kendine has" dünyalar kavramı ile ilgilenebilmektir. Bu durum özellikle Olası Dünyalar kavramı ve model kullanımı ile gerçeklenebilecek bir yaklaşımdır. Daha da açık hali ile, yine kişisel kanaatim olarak, bir makinanın akıllı kabul edilebilmesi için bir insan kadar yetenekli olmasına gerek yoktur. Yapacağı işe göre özelleşmiş bir dünya (ve anlayışı) ile yine yapacağı işe göre biçimlenmiş davranışlar bir makina için yeterlidir. Bizim programımızın da dolaylı olarak bu yaklaşımı kullandığı görülecektir. Ortada varolmayan bir nesnenin ekrandaki nesneleri doğru biçimde hareket ettirebilmesi için girilen cümlelerin biçimsel bir hale getirilmesi sözkonusudur. Bunu yaparken de fiillerimizin yüklemlerinden isimlerin yorumlanmasına kadar her şey

yine yapılacak işe göre şekillenmiştir. Diyebiliriz ki bu amacımız programımız vasıtası ile gerçeklenmiştir. Programımız gelişmiş program olmamakla birlikte bir başlangıç çalışması olarak sahip olduğumuz bazı düşüncelerimizin daha rafine hale gelmesi konusunda yardımcı olmaya yetmiştir. Bu durum da tezimizin teorik amacı/katkısı olarak düşünülebilir.

1.2 Tezimizin Organizasyonu

Belirttiğimiz amaçlarımızı gerçekleştirebilmek için öncelikle nasıl bir program yazmalıyız diye düşünmeye başladık ve Yapay Zeka ve dilbilim alanında adeta bir kırılma noktası olarak kabul edilen SHRDLU'yu incelemeye karar verdik. Benzer biçimde ELIZA hakkında da bilgi toplama yoluna gittik. Ancak ilk hali ile ELIZA'nın bizim düşüncemize çok da uymadığını farkedip SHRDLU kadar gelişmiş olmamasına karşın (SHRDLU'nun MIT'de bir doktora çalışması sırasında geliştirildiği düşünülecek olursa normal bir durumdur) onun temel yaklaşımını kullanan ve adeta görünmeyen bir varlık için anadilimizdeki cümleleri biçimsel hale getiren bir program yazmaya karar verdik. O sebeple tezimizin ikinci bölümünde ilham kaynağımız olan SHRDLU'ya yer verdik.

Daha sonra anlambilim alanındaki yaklaşımları incelemeye başladık. İlk karşımıza çıkan yaklaşım anlambilim alanında çok önemli ve biçimsel olarak ilk çalışma diyebileceğimiz Montague yaklaşımı oldu. Bu sebeple tezimizin üçüncü bölümünü kökleri ile birlikte Montague yaklaşımını incelemeye ayırdık.

Montague yaklaşımına göre çok daha yeni bir yaklaşım olan Durum Teorisi ve Anlambilimi de bizim için önemli bir konu olduğu için onu da fazla matematiksel ve biçimsel tanımlamalara girmeksizin tezimizin dördüncü bölümünde anlattık. Bu şekilde iki teori ve dolayısıyla anlambilim alanındaki yaklaşımların beznerlikleri ve eksikleri konusunda tartışabileceğimizi düşündük.

Mevcut teorileri değerlendirdikten sonra fazla gelişmiş olmasa da Montague yaklaşımını kullanarak tezimizin açıkladığımız son iki amacına yönelik olarak bir program yazma işine giriştik. O sebeple ilk işimiz programımız için veri toplama ve genel yapının kurulması olmuştur. Programımızın genel yapısını beşinci bölümde anlatmaya çalıştık.

Programımızın genel yapısı belirlendikten sonra bunun sözdizimsel ve anlambilimsel yapısının program içine oturtulması gerekiyordu. Bu işlemlerin nasıl yapıldığı ise sırası ile altıncı ve yedinci bölümlerde anlatılmıştır.

Son olarak yaptığımız programın ve tezimizin sonuçlarının değerlendirilmesinin yer aldığı kısım sekizinci bölüm olarak tezimizde yer almaktadır. Ayrıca şunu da belirtmek gerekir ki tezimiz bitmiş bir çalışma olarak düşünülmemelidir. Bu durum programımızın yapısından da rahatlıkla anlaşılabilir. Çeşitli zaman sıkıntıları vb. ile tezimizin bu hali ile tamamlanması gerekmiştir. Ayrıca daha önce de belirttiğimiz gibi tezimizde incelediğimiz yaklaşımlardan sadece biri kullanılmıştır. Dolayısıyla yapılabilecek ilerletme ve çalışmalar fazlası ile mevcuttur. Bu konu da yine sekizinci bölümümüzde Gelecekte Yapılabilecek Çalışmalar başlığı altında toplanmıştır.

Ayrıca tezimizin dili konusunda daha önce söylediklerimize ek olarak şunu da belirtebiliriz: Anlaşılabilirlik ve bilgi verme amaçlı olarak standart bir anlatımdan ve yazımdan yer yer uzaklaşılmış olabilir. Ancak, bir kez daha vurgulamak gerekirse, bunun sebebi anlaşılabilirliğin ve yapılan incelemeler konusunda bilgi aktarımının ön planda tutulmasıdır. Bu durumun tezimizin okuyucuları tarafından (belirttiğimiz amaçlar göz önüne alınarak) mazur görüleceğini umuyoruz.

2. ÖRNEK BİR PROGRAM OLARAK SHRDLU

Bu kısımda SHRDLU'nun program olarak yapısından bahsedilmiş fakat çok fazla detaya girilmemiştir. Bizim programımız için örnek teşkil eden bir program olan SHRDLU'nun yapısı için bu genel bakış açısının yeterli olacağı kanaatindeyiz. Yazımızda Malta Üniversitesinin Doğal Dil İşleme derslerinde faydalanılan Profesor Ben-Avi'ye ait notlardan faydalanılmıştır. Belirttiğimiz ders notları http://www.cs.um.edu.mt/~mros/cs305/shrdludetails.html adresinden bulunabilmektedir.

2.1 Giriş

SHRDLU Terry Winograd tarafından 1968-1970 yılları arasında MIT Yapay Zeka Laboratuvarında yazılmış, doğal dil anlamaya yönelik bir programdır. Bloklardan oluşan bir dünyada bir robot kolunun simule edilmesi işlemini yerine getirmiştir. Simule edilmesi işlemi diye nitelendirilmesinin sebebi, masa üzeri dünyası olarak da nitelendirilebilecek olan bu dünyadaki işlemlerin aslında programın içsel veri yapıları üzerinde gerçekleştirilmiş olması, ancak bu yapılan değişikliklerin de paralel olarak görsel biçimde ekrana yansıtılmış olmasıdır. Yani içsel olarak herhangi bir gösterim sözkonusu olmayıp, kullanılan veri yapılarındaki değişikliklere denk görüntü işlemleri yapılarak ekrana yansıtılmaktadır. İlerleyen bölümlerde programın yapısı/tasarımı, özellikleri ve yapay zeka alanına getirdiği yeniliklerden bahsedilecektir.

2.2 Programın Tasarımı

2.2.1 Hiyerarşik olmayan düşünce yapısı

SHRDLU aslında bir tek program değil üç farklı programın bir arada çalışmasıyla oluşan bir programdır. Bu şekilde bir programın entegrasyonu için uygulanabilecek yöntemler seri, hiyerarşik ya da hiyerarşik olmayan olarak sınıflandırılabilir. (Burada hiyerarşik olmayan ile İngilizce "heterarchical" kelimesi karşılanmak istenmiştir. Yani seri düşünce yapısı bu gruba dahil değildir.)

Seri birleştirme işleminde bir programın çıktısı diğer bir program tarafından girdi olarak kullanılmaktadır.

Hiyerarşik birleşirme işleminde ise bir program genel kontrolü elinde bulundurmaktadır. Bu program ihtiyaca göre alt programlar olarak nitelendirilebilecek programları çağırarak işlemlerin yerine getirilmesini sağlar. Alt programlar da kendi içlerinde alt programlara ayrılabilir. Bu yöntemde yetkilendirme daima daha yukarı bir seviyeden gelmektedir. Daha da ötesinde ise yapıyı bir ağaç gibi düşünecek olursak kardeş düğümlerin birbirleri ile haberleşmesi diye bir kavram yoktur. Yani yatayda haberleşme yoktur. Hiyerarşik kontrol alt programlara belli ya da değişen sıralarla aktarılabilir. İkinci yöntem daha esnektir. Ancak her durumda performansı etkileyen bütün kararlar daha yukarı seviye(ler)de alınmaktadır.

Hiyerarşik olmayan organizasyonda ise kontrol sorumluluğu bütün sisteme eşit biçimde dağıtılabilir. Hiyerarşik olmayan olarak birbiri ile ilişkilendirilmiş programlar birbirleriyle yukarıdan aşağı, aşağıdan yukarı ya da yatayda haberleşebilirler, birbirlerinin çağırabilirler.

Terry Winograd'ın programını oluşturan üç program gramer, anlam ve çıkarım olarak adlandırılabilir. Bunlardan ilki İngilizce cümleleri parçalamak/ayrıştırmak ve cümlenin sözdizim yapısını belirlemekte kullanılmaktadır. İkincisi anlam ile ilgilenen programdır ve yapıya özel anlamlandırma parçalarından oluşmaktadır. Bu yapıya özel kısımlar isim, sıfat öbekleri gibi yapıların anlamlandırılmasında kullanılmaktadır. Üçüncü program ise çıkarımda bulunmak amacı ile kullanılmaktadır. Bu programlardan hiçbiri sistem hakkında genel bir görüşe sahip değildir. Hatta monitör program olarak isimlendirilen program bile böylesi bir görevi yerine getirmemektedir.

SHRDLU fonksiyonlarını hiyerarşik olmayan bir biçimde yerine getirmektedir. Her ne kadar gramer programı genel koordinatör olarak tanımlanmış olsa da çıkarım sistemi her aşamada kullanılmaktadır. Hem sözdizim programı hem de anlam programı çıkarım programını kendi işlemlerinin bazı aşamalarında kullanabilmektedir. Örneğin sözdizim kuralları ile karmaşık bir cümlenin anlamının çözümlenmesi esnasında anlam programı istenebilir. Bunun dışında zamir gibi yapıların referans çözümlemeleri esnasında hem

anlam hem de çıkarım programının yardımına başvurulabilmektedir. Yani SHRDLU'nun anlama kabiliyeti ağırlıklı olarak çıkarım sistemine dayanmaktadır. Örneğin İngilizce'de karmaşık bir cümle olarak kabul edilebilecek bir cümle olan "Put the blue pyramid on the block in the box" biçiminde bir cümlenin iki biçimde yorumlanması mümkündür ve ayrıştırıcı için bu güç bir durumdur. Bu iki farklı yorumlama aşağıdaki gibi olabilmektedir:

1. Put the blue pyramid (on the block in the box)
2. Put (the blue pyramid on the block) in the box

Bu durumda cümlenin çözümlemesi şu adımlarda gerçekleştirilmektedir. Öncelikle fiil (put) emir kipinde olarak belirlenmekte ve bu durumda bir isim öbeği beklenmektedir. Ancak yukarıdaki maddelerden de anlaşıldığı gibi sonrasında gelen isim öbeğinin iki farklı biçimde yorumlanması mümkündür. Ayrıştırma işleminde bir bölüm tamamlandığında anlam programına başvurularak bu işlemin geçerli olup olmadığı kontrol edilmektedir. Örneğin burada "pyramid" hareket ettirilebilecek bir nesne olarak belirlenmektedir. Ancak "on" kelimesi yüzünden ortaya çıkabilecek karışıklık "on" kelimesinin çözümlemesinde kullanılan anlamsal program parçası tarafından tespit edilmektedir. Burada "on" kelimesi konulacak yer belirten bir anlamda kullanılabileceği gibi (1 no'lu maddedeki durum) piramid'i özel olarak bir yerde bulunan piramit diye belirlemek için de kullanılmış olabilir (2 no'lu maddedeki durum). Bu durumda çıkarım programına başvurularak piramidin bir bloğun üstünde durup durmadığı sorulur. Eğer çıkarım programının cevabı olumsuz olursa ilk yorumlama kullanılır ve ayrıştırma işlemi bir sonraki öğe ile devam eder. Eğer cevap olumlu olursa ikinci yorumlama geçerlidir. Her iki durumda da ayrıştırma işlemine nasıl devam edileceği değişmektedir ve ayrıştırma işlemi yapılan belirlemeye uygun olarak devam etmektedir. Burada çevresel durum, yani şekillerin o an için birbirlerine göre durumları bilgisi, cümlenin nasıl yorumlanacağı üzerinde etkili olmuştur.

Bu örnekle hiyerarşik olmayan program birleştirme işleminin sağladığı esneklik anlaşılmaktadır. Bizim programımızın yapısı hakkındaki bilgiler Programımızın Genel Yapısı ve Prensipleri bölümü ile takip eden bölümlerde bulunabilir.

2.3 SHRDLU'nun Sahip Olduğu Bilgi

Genel olarak SHRDLU'nun sahip olduğu bilgiyi üç ana gruba ayırmak mümkündür. Bunlardan ilki programın genel yapısı içinde yerleşik olarak bulunan genel problem çözme bilgisidir. Programın çıkarım bölümü bu kategoridedir. Çıkarım programının kullanılan programlama dili ve yapılan bazı tasarım tercihleri sebebi ile bir takım dezavantajları bulunmaktadır. Örneğin İngilizce olarak "every" ya da "all" olarak ifade edilen ve Türkçe'ye "her" ya da "hepsi" gibi çevirebileceğimiz evrensel niteleyici ile ilgili bazı problemler buna örnek olarak gösterilmektedir. Ancak bu konuda daha fazla detaya girmeyi gereksiz görüyoruz. İkinci grupta ise gramer ve anlam programlarının içine yerleştirilmiş olan dilbilimsel bilgi bulunmaktadır. Bu bilgiye örnek olarak cümlecik, isim öbeği gibi yapılar için özel olarak hazırlanmış, bu yapıların sözdizimsel olarak incelenmesi ve anlamsal değerlerinin verilmesi işlemini yürüten program parçaları verilebilir. Yani sözdizimsel program yapıya özel fonksiyonlar barındırmaktadır. Üçüncü kategoride ise anlamsal bilgi bulunmaktadır. Bu grup kendi içinde genel anlamsal bilgi ve SHRDLU'nun blok dünyasına ilişkin özelleşmiş bilgi olarak iki farklı bölüme ayrılmaktadır. Blok dünyasına ilişkin bilgi elemanların boyutları, şekli, rengi, o andaki ve önceki konumları gibi bilgileri içermektedir. Bu bilgiler sayesinde neyin, nasıl yerine getirileceği gibi belirlemeler yapılabilmektedir. Örneğin "pick up" gibi bir fiilin cansız bir varlığı özne olarak alması mümkün değildir ve bu durum çevresel bilgi olarak kodlanmıştır. Buradan da anlaşıldığı gibi SHRDLU'nun davranışları ve yapacağı değişiklikler programın sahip olduğu içsel dünya modeline son derece bağlıdır. Küçük, kapalı ya da kendine has dünyalar konusundaki düşüncelerimiz ve bunun programımızda kullanılması konusundaki düşüncelerimiz için Programımızın Genel Yapısı ve Prensipleri bölümü ile program yapısının anlatıldığı takip eden bölümlere bakılabilir.

2.4 Diğer Özellikler

Bu bölümde SHRDLU'nun sahip olduğu özellikler belli konu başlıkları altında toplanmadan anlatılacaktır.

SHRDLU'nun yazımında LISP ve LISP tabanlı MICRO-PLANNER dilleri kullanılmıştır. Çıkarım programı ve blok dünya bilgisi PLANNER ile kodlanmıştır. Ayrıca SHRDLU'nun ayrıştırıcısı LISP'e gömülü olarak kullanılabilen ve Terry Winograd'ın yazdığı PROGRAMMAR ile kodlanmıştır.

PROGRAMMAR sistematik olarak tanımlanabilen sözdizimsel yapıları oluşturmak üzere temel fonksiyonları sağlamaktadır. Dilin arkasındaki düşünce ise prosedürlerin, iterasyon ve özyinelemenin bilişsel işlemler için temel olduğu biçimindedir. PROGRAMMAR ayrıştırma işlemi için seçim yapması gereken noktada anlam prosedürlerinden de yardım almaktadır. Bu sayede ayrıştırma işlemini belli bir noktaya getirdiğinde içinde bulunulan durum açısından mantıklı bir ifade olup olmadığını kontrol edebilmektedir. (Bu durumun somut örneği hiyerarşik olmayan düşünce yapısı bölümünün sonunda verilmiştir.) Sistemin başarılı olmasının bir sebebi olarak da PLANNER'ın muhakeme prosedürleri, anlam analizi ve PROGRAMMAR'ın başarılı bir biçimde etkileşimi gösterilebilir. İlk bölümde anlatıldığı gibi bu yapılar arasında hiyerarşik olmayan bir bağ vardır ve gerekli noktalarda ihtiyaç duyulan başka bir programdan yardım alınabilmektedir. Her üç eleman da gelen girdiyi inceleyip ayrıştırma işlemine yardım edebilmektedir. Bu durum SHRDLU'nun özellikle zamirler ve refarans çözümlemesinde başarılı olmasının bir sebebi olarak da gösterilebilir.

Programın yazımında, anlamın prosedürel yapılar olarak ifade edilebileceği ve dilin dinleyen kişide bazı prosedürleri aktive edeceği ilkelerinden yola çıkılmıştır. Bu sebeple SHRDLU'nun bilgi tabanı düz gerçekler ve kurallar olarak ifade edilmemiş bunun yerine prosedürel gösterim kullanılmıştır. Bu durumda kullanıcıdan gelen veri bazı prosedürlerin harekete geçirilmesi için kullanılmakta ve bu şekilde istenen iş yerine getirilebilmektedir. Burada önemli olan nokta bilginin prosedürel olarak ifade edilmiş olmasıdır.

SHRDLU'nın dünya modeli ve muhakeme kısmı MICRO-PLANNER dili ile kodlanmıştır. Dünyanın durumu hakkındaki bilgi MICRO-PLANNER yüklemleri[1], işleme

[1] Burada kullanılan "yüklem" kelimesi yerine İngilizce "assertion" kelimesi kullanılmaktadır. Ancak Türkçe'ye tam karşılığı ile çevirmek yerine bu şekilde karşılanması bizce daha uygun olduğundan bu kullanım tercih edilmiştir.

yönelik ve muhakemeye ilişkin bilgiler ise MICRO-PLANNER programları olarak gösterilmiştir.

Örnek konuşmalara bakıldığı zaman SHRDLU'nun diğer bazı özellikleri de anlaşılabilmektedir. Örneğin ilk anda kendisine "steeple" ile ilgili bir soru sorulduğunda

- Sorry, I don't know the word "steeple"
  derken,
- A "steeple" is a stack which contains two green cubes and a pyramid
  gibi bir cümleye
- I understand

şeklinde cevap vererek bunu kendi bilgileri arasına ekleyebilmektedir. Daha sonra bu kelimeye ilişkin sorulara da cevap vermeye başlamaktadır. Bunun yanında sahip olduğu nesnelerin komut veren kişi tarafından isimlendirilmesine ve daha sonra bu isimlerin kullanılmasına izin vermektedir. Kısacası belli miktarda öğrenme yeteneğine sahip diyebiliriz.

Geçmişte olayları hatırlayabilmektedir. Eğer birden fazla yorumlama mümkün olan bir durumla karşılaşırsa mümkün olan yorumları kullanıcıya gösterip hangisini tercih ettiğini öğrenerek işlemlerine bu şekilde devam edebilmektedir. Örneğin;

- How many things are on top of green cubes?
  gibi bir soruya
- I'm not sure what you mean by "on top of" in the phrase "on top of green cubes"

  Do you mean

  1 – Directly on the surface

  2 – Anywhere on top of

biçiminde cevap vermekte ve kullanıcının 1 ya da 2 biçiminde cevabına göre işlemine devam ederek gerçek cevabı vermektedir. Buradaki karışıklığın üzerinde, arasında gibi Türkçe kelimeler içinde geçerli olduğu unutulmamalıdır. Çünkü Türkçe'de de "arasında" dendiğinde hakikaten iki şeklin arasındaki doğru parçasının orta noktasının mı

yoksa o doğrunun üzerinde olmamakla birlikte iki şeklin arasındaki herhangi bir noktanın mı anlatılmak istendiğinin belirlenmesi gerekebilir.

Ayrıca “it” gibi bir zamir ile karşılaştığında onun neye referansta bulunmak için kullanıldığını varsayıyorsa durumu kullanıcıya bildirip öylece işlemlerine devam etmektedir. Örneğin önceden sorulmuş soru ile bağlı olarak sorulan şöyle bir soruda;

- What color is it?
  önce
- By “it”, I assume you mean the shortest thing the tallest pyramid’s support supports
  dedikten sonra
- Red
  biçiminde cevap vermektedir.

Kullanıcısı kadar büyük bir yeteneğe sahip olmasa da yukarıdaki anlatımdan da anlaşılabildiği gibi belli miktarda cevap verme yeteneğine de sahiptir. Örneğin bazı kalıplara ya da deyimlere de uygun cevap girişleri mevcuttur. Örneğin “Thank you” – “You’re welcome!” kalıbı gibi.

SHRDLU’nun yapısı hakkındaki olumsuzluklardan birisi kapalı dünya modeli kullanması yani sahip olduğu aksiyomların kendisinin bilmesi gereken tüm gerçekleri verdiğini düşünmesidir. Bununla beraber küçük dünyalar olarak nitelendirebileceğimiz kendine has dünyalar kavramı için SHRDLU son derece iyi bir örnek oluşturmaktadır. Bu noktaya kadar anlatılanlar ve bunların bizim programımızın tasarımı üzerindeki etkileri, kapalı ve küçük dünyalar, bu dünyaların modellenmesi hakkındaki görüşlerimiz daha önce de belirttiğimiz gibi Programımızın Genel Yapısı ve Prensipleri bölümünde ve takip eden bölümlerde belirtilmiştir. Ayrıca sahip olunan aksiyomların kapalı ya da açık bir dünya hakkında bilinmesi gerekenler için yeterli olup olmadığı konuları Montague Anlambilimi ve Durum Teorisi bölümlerinde ele alınmıştır.

SHRDLU doğal dil işleme alanında büyük bir adım olarak görülmektedir. Çünkü daha önceki yapay zeka tabanlı dil programları dilbilimsel olarak basit bir yapıya sahiptiler, anahtar kelime ya da örnek yönelimli gramerler kullanıyorlardı. Dilbilimciler tarafından kullanılan daha güçlü gramer modelleri dahi çıkarım ve anlamsal bilgiyi cümle analizinde

az kullanıyorlardı. Bu iki yapının birden SHRDLU'da kullanılması gerçekten başarılı sonuçlar vermiştir.

3. MONTAGUE ANLAMBİLİMİNE GENEL BİR BAKIŞ

3.1 Giriş

Bu bölümde İngilizce'nin belli bir parçası için biçimsel bir anlam analizi yapmış olan Richard Montague'nin anısına Montague Anlambilimi olarak adlandırılan yaklaşımdan bahsedilecektir. Özellikle Durum Teorisi konusunun getirdiği bakış açısının anlaşılabilmesi ve bizim programımızda da faydalandığımız kendine has dünya kavramı için bu bölümde anlatılacak olanlar önem taşımaktadır. Anlatımızda faydalanacağımız temel kaynak (Dowty vd. 1992)dir. Diğer kaynaklardan yapılan alıntılar açık biçimde belirtilecektir ve özel olarak kaynak belirtmediğimiz durumlarda dilbilim alanında Montague Anlambilimi konusunda temel kaynak olarak kabul edilen (Dowty vd. 1992)'den faydalandığımız varsayılmalıdır.

Montague matematiksel mantık alanında geliştirilen teknikleri doğal dilde anlam analizinde uygulamaya çalışmış ve İngilizce'nin bir bölümünü kapsayan bir yapı hazırlamıştır. Montague'nin çalışması dilbilimciler ve filozoflar arasında artan bir ilgiye sebep olmuştur. Çünkü üretici dönüşümsel yaklaşımların sözdizim alanına getirdikleri biçimsel açıklık ve belirliliğin benzerinin anlambilim alanına getirilebileceği fikrini uyandırmıştır. Ayrıca dilbilimsel teoriler için yeni bir düşünme ve sorgulama biçimine de yol açmıştır.

Anlambilim filozoflar ve mantıkçılar tarafından ilgilenilmesi zor olmayan bir alan olarak görülse de biçimsel mantık için geliştirilmiş sistemlerin doğal diller alanında kullanılması pek de kolay olmayan bir yaklaşım olarak kabul edilmiştir. Anlambilim sözdizim ya da fonolojiye göre daha soyut bazı özelliklere sahiptir. Çünkü uğraşılması çok kolay olmayan "anlam" ile ilgilenmektedir. Bu sebeple çoğu zaman dilbilimin diğer alanlarının kullandıklarından farklı yaklaşımların kullanılması gereken bir alan olarak düşünülmüştür. Fakat Montague ile birlikte anlambilim alanında teori kurma ve test etme işlemlerinin, en azından prensip olarak, diğer dilbilim alanlarından çok farklı olmadığı düşünülmeye başlamıştır.

Ancak Montague Anlambilimi'ni açıklamaya başlamadan önce dayandığı diğer çalışmalar hakkında genel olarak bilgi verilmesinin konuyu anlamak açısından avantaj sağlayacağını düşünüyoruz. O sebeple sonraki bölümde Frege ile başlayıp Montague'ye uzanan çalışmalardan bahsedilecektir.

3.2 Montague Anlambilimine Temel Oluşturan Çalışmalar

3.2.1 Frege ve anlamsal değer

Gottlob Frege, (Harris, 1997) de belirtildiği gibi, bir matematikçi idi ve aynı kaynakta aşağıda belirtildiği gibi yaptığı çalışmaları ile dilbilim ve özellikle de anlambilim alanında etkili olmuştur.

"... Frege'nin matematiksel ilgisi kendisini dilin insan düşüncesi konusundaki rolünü düşünmeye götürmüştür ve anlam ve doğruluk alanındaki sonuçları oldukça etkileyici olmuştur. ... 1879 yılında yayınlanan Begriffsschift (Conceptul Notation)'ın açılış paragrafı Frege'nin dikkatinin anlambilim alanındaki problemlere odaklandığını göstermektedir. "

Benzer biçimde Frege'nin ortaya attığı kavramların tartışıldığı (Miller 2003)te bir cümle için anlamsal değerin Frege açısından tanımı aşağıdaki gibi verilmektedir.

"... Frege'ye göre bir cümlenin anlamsal değeri doğruluk değerlerinden biridir, doğru ya da yanlış."

Yine aynı kaynakta Frege'nin anlam analizi çalışmalarına ilişkin şu açıklama yer almaktadır:

"Genel olarak karmasık bir ifadenin anlamsal değeri parçalarının anlamsal değerleri ve biraraya geliş biçimleri ile belirlenmektedir. ..., bu iki tezi Frege'nin anlambilimsel teorisi içinde düşünebiliriz."

Karmaşık ifadelerin anlamsal değerlerinin parçalarından elde edilmesine ise Frege'nin Birleştirilebilme İlkesi[2] denmektedir ve sonraki bölümde ayrıntılar ile ele alınacaktır.

Ayrıca Frege'nin anlamsal değer kavramının uygulanarak bir cümleye değer atanabilmesi durumu beraberinde **doğruluk şartları kavramı**nı da getirmektedir. Bu durum (Miller 2003)te Frege'nin Birleştirilebilme İlkesi'nin basit bir dile uygulanması örneği ile anlatılmakta ve konu ile ilgili olarak

"... Ek olarak, cümlelerin doğruluk şartlarının cümlenin parçalarına anlamsal değerlerinin atanması yolu ile nasıl türetildiğini de göstermek için kullanabiliriz. Bir cümle için doğruluk şartı, T (doğru yerine kullanılmış bir gösterim) ile gösterilen doğruluk değeri değil, bunun yerine cümlenin doğru olabilmesi için dünyada karşılanmış olması gereken bir şarttır"

biçiminde bir açıklama yer almaktadır. Doğruluk şartları ile ilgili geniş açıklama Montague Anlambiliminin Özellikleri isimli bölümümüzde ayrıca ele alınarak yapılacaktır.

Anlaşıldığı gibi bir matematikçi olarak Frege anlam analizi konusu ile ilgilenmiş ve ortaya bazı tezler atmıştır. Aynı kaynakta Frege'nin ortaya attığı kavramlar ve tezler adım adım anlatılmakta bunun yanında da gelen bazı itirazlar, Frege'nin teorisindeki bir kısım problemler dile getirilmektedir. Bizim anlatımımız açısından ortaya atılan iki kavramın daha adını açıklamanın yeterli olduğunu düşünüyoruz. Bunlar **sense**[3] ve **reference** kavramlarıdır. Sense yerine İngilizce kaynaklarda **intension**, reference yerine de **extension** kavramları da kullanılabilmektedir. Bu kavramların açıklamaları ise Montague Anlambiliminin Özellikleri bölümümüzde Olası Dünyalar kavramından sonra verilecektir. Sense ile intension ve reference ile de extension kavramları ilerleyen bölümlerde değişmeli olarak kullanılacaktır.

---

[2] Bu ilkeye Frege'nin Birleştirilebilme İlkesi denmesine, Frege'nin kendisinin de bu işlemi açıkca uygulamasına karşın açıkca bu şekilde adlandırmadığı ve sonradan bu biçimde isimlendirildiği (Löbner 2002)de anlatılmaktadır.

[3] "sense - intension", "reference - extension" terimleri için Türkçe karşılıklarını kullanma yoluna gitmememyi uygun gördük. İngilizce "reference" kelimesi Türkçe'ye kolaylıkla "gösterdiği" gibi çevirilebilecek olmasına karşın "sense - intension" kavramlarının tanımları o kadar kolay olmadığından hepsini İngilizce olarak bırakmayı uygun gördük.

Burada önemli olan nokta anlamsal değer, doğruluk şartları, intension, extension kavramlarının ve Birleştirilebilme İlkesinin Frege'ye kadar uzanmasıdır.

3.2.2 Tarski ve model teori

Model teori (Hodges 1998)de "biçimsel bir dilin cümleleri ile onları doğru ya da yanlış yapan yorumlamaları ya da yapıları arasındaki ilişkiler üzerinde çalışır" şeklinde açıklanmaktadır. Aynı biçimde model teorinin doğruluk, mantıksal doğruluk ve sonuçları, anlam gibi kavramlar konusunda açık tanımlar önerdiği dile getirilmektedir. Durum aşağıdaki gibi açıklanmaktadır;

"1954 yılında Alfred Tarski yeni bir matematik dalının ortaya çıktığını ilan etmiştir. O bunu, biçimsel teroilere ait cümleler ile bu cümlelerin içinde yer aldığı matematiksel yapılar arasındaki karşılıklı ilişkiler üzerinde çalışan "model teori" olarak adlandırmıştır. ... Örneğin, İngilizce için model teoretik bir yaklaşım İngilizceye biçimsel bir dil olarak davranmalı ve İngilizce yapılara anlamları matematiksel yapılar olarak ifade edilebilen olası dünyalar aracılığı ile atamalıdır. ... Tarski "biçimsel bir teorinin cümlesi" kavramı ile anlamlı kelimeler ve [bazen **şematik değişkenler** olarak adlandırılan] anlamsız sembollerden oluşan ve sembollere anlam verildiği zaman anlamlı hale gelen, mantıkçıların **cümle şeması** dedikleri kavramı kastetmektedir. Örneğin;

(1) Everything which is a P is a Q [4]

bir cümle şemasıdır. ... Bir matematiksel yapı ise şemadaki sembolleri yorumlamanın bir biçimidir. Bu da şematik sembollerin anlamlarını açıklayan ... nesneler topluluğunu, ... ilişkileri, fonksiyonları içermektedir. ... Model teorinin merkezi aracı **doğruluk ilişkisi**dir. Varsayalım ki φ bir cümle şeması ve M, φ'yı yorumlayan yapı olsun. Yorumlandığında ise φ ya doğrudur ya da yanlış. Eğer doğru ise φ, M'de doğrudur deriz ...

[4] Burada İnglizce cümleyi Türkçe'ye çevirmeye çalışmadık. P ve Q dışındaki kelimelerin anlamlarının olduğu ve adı geçen harflerin de sembollere karşılık geldiği yeterince açık diye düşündük.

Tarski her M yapısının kümesel bir nesne ve φ ile gösterilen şemaların da geleneksel biçimsel mantık cümleleri olduğunu varsaymıştır. ”

Tüm bunlardan da anlaşıldığı gibi yaklaşımın temelinde biçimselleştirilmiş cümleler ve bu cümlelerin doğruluklarının kontrol edilebileceği bir model ile doğruluklarının kontrolü bulunmaktadır. Ayrıca yine dikkat edilmesi gereken bir diğer nokta Tarski'nin yaklaşımında da [biçimsel] cümleler için doğru ya da yanlış değerlerinin kullanıldığıdır. Bu durum Frege'nin cümlenin anlamsal değeri olarak aynı şekilde doğru ya da yanlış tanımlamalarını kullanmış olması ile örtüşmektedir ve (Hodges 2002)de her ifadenin bir değerinin olduğu ve bu değerlerin Frege'nin verdiği değerler ile aynı olduğu İngilizce "Fregean values" terimi kullanılarak açıklanmıştır.

Bu noktadan sonra Tarski'nin yaklaşımı ile Olası Dünyalar kavramının birlikte kullanımı gibi konulara girmeye gerek duymuyoruz. Ancak son bir anlatım olarak aynı kaynakta geçen

"**Montague** [**Tarski'nin bir öğrencisi**] bu tip araçlar kullanarak İngilizcenin bir bölümü için bir doğruluk tanımlaması önermiştir."

şeklinde bir ifade ile Montague Anlambilimi'ne uzanan yolu açıklamak mümkündür.

3.3 Montague Anlambiliminin Özellikleri

Önceki bölümde anlatılanlar ile artık Montague Anlambilimi'nin kökleri anlaşıldığı gibi özelliklerinin anlaşılmasının da kolaylaşacağını düşünmekteyiz.

Montague'nin yaklaşımının en temel özellikleri şunlardır;

Doğruluk şartlarına bağlıdır.

Model kullanımını gerektirir.

Olası dünyalar kavramının kullanılmasını sağlar.

Bu bölümdeki anlatımda (Dowty vd. 1992)den faydalanılmıştır. O sebeple bu bölümdeki anlatımımızda da her yerde kaynak göstermek yerine onun dışındaki kaynakları belirtme yoluna gideceğiz.

3.3.1 Doğruluk şartlarına bağlılık

Anlambilimde doğruluk şartlarına bağlılık ile kastedilen, bildirim özelliğine sahip (declarative) bir cümlenin anlamını bilmenin o cümlenin doğru olabilmesi için dünyanın nasıl bir durumda olması gerektiğini bilmektir. Yani bir cümlenin doğruluğunu belirlemek, onun için gerekli olan doğruluk şartlarını belirlemektir. Daha da açık hali ile cümlenin doğru olabilmesi için gerekli ve yeterli şartların sağlanmasıdır. (Doğruluk şartları kavramının ne olduğu bir önceki bölümde "Frege ve anlamsal değer" konu başlığı altında açıklandığı için tekrarlamaya gerek duymuyoruz). Burada **doğru** olma ile kastedilen, cümle ile belirtilen durumun mevcut dünya için geçerli olması halidir ve bir önceki bölümde İngilizce olarak "Fregean value" dediğimiz kavrama denk gelmektedir. Anlam konusunda daha farklı yaklaşımlar da mevcuttur. Örneğin bir cümlenin anlamının onu anlayan kişinin zihnindeki bir imge veya fikir olması bunlardan biridir. Ayrıca (Löbner 2002)de anlatıldığı gibi Saussure'e kadar giden Yapısal Yaklaşım ile temel kelimeler ve aralarındaki ilişkiler ile ilgilenerek anlam konusu ile ilgilenenler ve buna bağlı yaklaşımlar da vardır. Bu konuda daha fazla bilgi için adı geçen kaynağa başvurulabilir. Ancak Montague'nin yaklaşımında ilk olarak söylediğimiz tanıma uygun bir yapı mevcuttur. Doğruluk şartları konusunun temel düşüncelerinden biri cümle ile mevcut dünya arasında varolan ilişkidir. Çünkü doğal dilin temel özelliklerinden biri de insanlar arasında dünya üzerindeki şeyler hakkında iletişimde bulunmakta kullanılmasıdır. Burada dünya kavramı cümlenin hakkında bilgi verme amacı güttüğü karmaşık nesne ve durumlardan oluşan bir bütün olarak düşünülmelidir. Bir cümle ile ifade edilenler dünyanın tamamına ilişkin değil bir parçasına ya da bir duruma ilişkindir. Ayrıca doğruluk şartlarına bağlı anlambilimin üzerine kurulduğu temel varsayım, dilin dünya ile nasıl bir bağlantı kurduğunun belirlenmesi çalışmasıdır.

Bu konuyu bir örnek ile açmak gerekirse

Karadeniz Türkiye'nin kuzeyindedir.

gibi bir cümlenin doğru olabilmesi için "Karadeniz" olarak adlandırılan varlığın "Türkiye" olarak adlandırılan varlıkla, ona göre kuzeyde bulunma ilişkisi içinde bulunması gerekir. Eğer mevcut dünya için bu ilişki var ise cümle doğrudur denir. Bizim de cümleyi anlayabilmek için bu durumu/ilişkiyi biliyor olmamız gerekir. Buradan da anlaşılacağı gibi "... cümlesinin anlamı nedir?" gibi bir soruya, cümlenin doğru olabilmesi için dünyanın o cümle ile ilgili parçası üzerindeki varlıkların ya da nesnelerin nasıl bir ilişki içinde bulunmaları gerektiğinin açıklanmasıdır biçiminde de cevap verilebilir.

Örneğimizle birlikte yeni bir kavram daha ortaya çıkmaktadır. Bu da **ilişkilerin durumu** diye Türkçe'ye çevirebileceğimiz kavramdır. İlişkilerin durumu ve dünya kavramı arasındaki temel fark Montague yaklaşımının üçüncü özelliği olan Olası Dünyalar başlığı altında açıklanmıştır.

Burada şöyle paradoksal bir yapı olduğu düşünülebilir; Türkçe bir cümlenin anlamını belirlemek için yine Türkçe cümleler ile ifade edilen bir ilişkiden bahsettik. Bu noktada ise **amaç dil** ve **üst**[5] **dil** ayrımı önemlidir. Kısaca açıklamak gerekirse amaç dil, üzerinde araştırma yaptığımız dil, bizim için Türkçe, üst dil ise amaç dil için ilişkilerin durumunu açıklamada kullandığımız dildir ve burada bizim için yine Türkçe'dir. Aslında üst dil olarak şekilsel gösterim vb. yapılar da kullanılabilir. Ancak yazım için en kolayı yine yazı dilini ve en iyi bildiğimiz dili kullanmaktır. Örneğimizi İngilizce bir cümle ve ilişkilerin durumunu açıklamak için Türkçe'yi kullanmış olsaydık bu iki dil birbirinden farklılaşmış ve sorun gibi görünen bu durum kısmen ortadan kaybolmuş olurdu.

Doğal dillerde teorik olarak doğru biçimde kurulabilecek cümle sayısının bir sınırı yoktur. Bu sebeple anlam analizi için kullanılacak olan Montague yaklaşımında da ilişkilerin durumlarının sınırsız biçimde üretilmesi gerekmektedir. Çünkü en başta da belirttiğimiz gibi cümlelerin anlamları ancak bu yolla belirlenebilmektedir. Dolayısıyla sözdizim alanında özyineleme ile elde edilen sınırsızlığın anlambilim alanında da yakalanması gerekmektedir. Burada ortaya çıkan iki soru "Özyinelemeli olarak ilişkilerin

[5] Burada İngilizce'de "meta" olarak belirtilen kelimeye Türkçe'ye üst diye çevirmeyi uygun gördük. Dolayısıyla "meta-language" terimi de Türkçe'ye "üst dil" olarak çevrilmiş oldu.

durumu belirlemesinin nasıl yapılacağı" ve "Mevcut dünyada nelerin varolduğu" biçimindedir. Dünyada bağımsız varlıklar[6] olarak nitelendirebileceğimiz nesneler/varlıklar, onların sahip oldukları özellikler ve birbirlerine göre ilişkileri bulunmaktadır. Farklı durumlarda/zamanlarda bu varlıklar/nesneler birbirlerine göre farklı özelliklere sahip olabilir ya da farklı ilişkiler içinde bulunabilirler.

Yukarıdaki açıklama ne yazık ki sorulan iki soru için de yeterli bir yanıt teşkil etmemektedir. Hala sınırsız sayıda ilişkilerin durumu belirlemesinin özyinelemeli olarak nasıl belirlenebileceği sorunu mevcuttur. Bunu aşmanın yolu ise sözdizimde cümle üretimi için kullanılan kuralların anlambilim alanına da yansıtılabileceği fikridir. Bu şekilde, sözdizim kurallarına uygun olarak sınırsız sayıda ilişkiler durumu amaç dil cümlelerine uygun olarak üretilebilir. Burada amaç dil cümleleri ile uyum içinde ve onlara karşılık gelecek biçimde ilişkiler durumu üretilebilmesine dikkat edilmesi gerekir. Çünkü bu yaklaşıma göre bir cümleye uygun ilişkiler durumu belirlenemiyorsa o cümlenin doğruluğu ya da anlamı konusunda birşey söylenemez. Bu çözüm yolu ile varılan nokta ise Frege'nin Birleştirilebilme İlkesi olarak bilinen ilkedir.

Önceki bölümde basit bir tanımını vermekle yetindiğimiz bu ilkeyi burada (Dowty vd. 1992)den faydalanıp daha da açarak ve örnekleyerek anlatacağız. Frege'nin Birleştirilebilme İlkesi'nin en temel tanımı "Bütünün anlamı parçalarının anlamlarının ve durumlarının bir fonksiyonu olarak belirlenir" biçimindedir. Bu durumda "Sözdizimsel yapıların anlamsal değerleri ne olacak?" biçiminde bir soru akla gelebilir. Bu durumda örneğin "Ece" gibi bir özel isim ile kastedilen gerçek dünyada o kişiye denk gelen insandır. Burada örnek olarak da kullandığımız özel isimler konusunda Frege'nin yaklaşımı ve gelen itirazlar için (Miller 2003)e bakılabilir. Ayrıca Montague'nin özel isimler için kullandığı gösterim konusunda da (Dowty vd. 1992)ye başvurulabilir. Bu noktada fazla ayrıntıya girmeye gerek duymuyoruz. Ayrıca Diğer Bazı Özellikler konu başlığı altında diğer kelime türlerinin analizi konusundan kısaca bahsedilmiştir.

[6] Burada İngilizce'deki "entity" kavramını Türkçe'ye "bağımsız varlık" olarak çevirmeyi uygun gördüğümüz için böyle bir anlatıma başvurulmuştur.

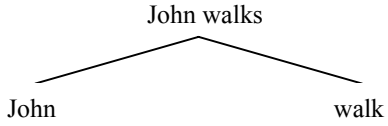
Bu noktada Birleştirme İlkesinin anlam analizinde kullanılması konusunda açıklama şu şekilde yapılabilir. Her cümle temel ifadeler diyebileceğimiz daha alt birimlerden/bileşenlerden oluşmaktadır. Bu bileşenler sözdizim kurallarına göre biraraya gelerek cümleleri oluşturmaktadırlar. Eğer bu bileşenlerin herbirinin mevcut dünyaya ilişkin (varlık, özellik, ilişki vb. biçiminde) bir şeyi belirttiğini düşünürsek sözdizim ve anlambilim için temel özyinelemeyi elde etmiş oluruz. Bu durumda a, b, c, ..., n gibi bileşenleri girdi olarak alan R gibi bir sözdizim kuralımız varsa buna denk gelen ve yine a, b, c, ..., n'i girdi olarak alan R′ gibi bir anlambilim kuralı tanımlanabilir. Eğer bütün sözdizim kurallarına karşılık gelecek bu biçimde anlambilim kuralları tanımlanabilirse gerçekten bir dil için anlam analizi yapmak mümkün olabilir.

Bu konuda (Kılıçaslan, Y., ve Tüysüz, M.A.A. 2002) çalışmasında yer alan şu örnek kullanılabilir;

ÖZNE-YÜKLEM SÖZDİZİM KURALI: *(İngilizce için)*

Üçüncü tekil şahıs kodlayan isimler geçişsiz bir eylemin (fiilin) üçüncü tekil şahsa ait geniş zamanlı formuyla birleşebilirler.

Bu kuralın uygulanması neticesinde sözkonusu cümle için oluşacak analiz ağacı şudur:



Özne-Yüklem kuralı ile ilişkilendirilmiş çeviri kuralı ise aşağıdakine benzer bir ifadeyle tanımlanmıştır:

ÖZNE-YÜKLEM ÇEVİRİ KURALI: *(İngilizce için)*

Eğer α bir isim ya da isim öbeği, δ bir geçişsiz eylem ve α ve δ için semantik ifadeler sırasıyla α' ve δ' ise $\alpha'(\delta')$, α ve δ'nın birleşiminden oluşan cümlenin semantik ifadesidir.

Elde edilen anlamsal ifadenin dünya modeline başvurmak vb. yollarla değerlendirilerek doğru ya da yanlış biçiminde doğruluk değeri alabilir.

Ayrıca burada elde edilen anlamsal ifadenin matematiksel mantığa ait bir ifade olduğunu (fazla gereği olmasa da) belirtmek istiyoruz. Çünkü ancak bu yolla doğal dile ait cümleler biçimsel bir hale getirilebilmekte ve (Yıldırım 2000)de de örneklerle açıklandığı gibi doğal dile ait muğlak anlatımdan kurtulabilmektedir. Tezimizde bu durumu tekrardan vurgulamanın gereğini duymuyoruz.

3.3.2 Model gerektirme durumu

Montague Anlambilimi'nin ikinci temel özelliği ise model kullanımını gerektirmesidir. Bunun anlamı, amaç dildeki ifadelerin anlamsal değerlerinin verilebilmesi için dünyada bulunan varlıkların/nesnelerin matematiksel soyut bir modelle ifade edilmesinin gereğidir. Model teorisi doğruluk şartlarına bağlı anlambilim alanında kullanılan en başarılı metottur.

Model teori için belirleyici olan durum, karmaşık ifadelerin anlamlarının kendilerini oluşturan daha basit ifadelerin anlamlarından elde edilmesidir. Anlam ise dünyada bulunan varlıkların/nesnelerin veya durumlarının bilinmesi demektir. Önceki bölümdeki örneğimiz için, o dünya modelinde "Ece" ile gösterilen bir şahsın varlığının belirtilmesi gerekmektedir ya da daha önceki örneğimiz için Karadeniz'in Türkiye'nin kuzeyinde kalma durumunun bu varlıkların dünya üzerinde bulundukları bilgisi ile birlikte dünya modelinde bulunması gerekmektedir. Bu noktada ilk akla gelen dünya modelinin **Kümeler Teorisi'**nden faydalanarak küme gösterimi ile yapılmasıdır. Önceki bölümde Tarski'nin yaklaşımında da kümesel bir yapının kullanımının akla geldiğini dile getirmiştik.

Bir modelin kurulması öncelikle dünyada nelerin bulunduğunun belirlenmesi ile başlar. Daha sonra amaç dil için uygun bir şekilde biçimsel olarak farklılaştırılmasının nasıl yapılacağı belirlenir. Burada yorumlama ile kastedilen ifade ve onun sahip olacağı anlamsal değer çiftidir.

Sonra yorumlamaya göre değişmeyecek olan "ve", "veya", "ise" gibi yapıların belirlenmesi ve yorumlaması için bilgilerin girişi yapılır. Bu yapılar genellikle modelden modele değişiklik göstermezler.

Verilen örnekte de anlatıldığı gibi sözdizim ve anlambilim kurallarından faydalanılarak verilen doğal dile ait cümle öncelikle biçimsel bir forma dönüştürülmekte sonra modele başvurarak yorumlaması yapılmaktadır. Bu durum önceki bölümde de model teoretik yaklaşımın İngilizce için uygulanmasında nasıl ele alınması gerektiği ile açıklanmıştır. Montague'nün en önemli yayınlarından birinin adı da "English as a Formal Language" (1970) biçimindedir ve en başta da söylediğimiz gibi İngilizce'ye biçimsel bir dil gibi yaklaşmıştır. Bu durum aynı zamanda Tarski'ye dayanan model kullanımının ve yorumlamanın bir gereği olarak da düşünülebilir.

Bu noktadan sonra fazla detaya girilmeyecektir. Bizim programımızın yapısı hakkında gerekli bilgiler Programımızın Genel Yapısı ve Prensipleri bölümü ile takip eden bölümlerde verilmiştir.

3.3.3 Olası dünyalar

Doğruluk Şartları başlığı altında açıklandığı gibi cümlenin doğruluğu dünyada o anki ilişkiler durumuna göre belirlenmektedir. Yani dünyaya bir başvuruda bulunulmaktadır. Bu işlemden, Örnek Bir Program Olarak SHRDLU bölümünde anlatıldığı gibi ayrıştırma işlemi için de faydalanılabilmektedir. Ancak bir cümlenin anlamı sadece o anda mevcut olan dünya modeli ile belirlenemeyebilir. Örneğin "Zengin olsaydım" gibi bir yapı ile başlayan bir cümleyi değerlendirebilmek için mevcut olan gerçek dünya modeli yeterli olmayacaktır.

Model kullanımında olası dünyalar arasından bir tanesi seçilir ve bu dünya modelinin cümleleri değerlendirebilmek için gerekli olan tüm gerçeklik şartlarını içerdiği varsayılır. Yani bir cümlenin anlatmak istediği herşeyin bu dünyaya ilişkin olduğu varsayılır. Burada ilişkilerin durumu ve dünya kavramı arasındaki farkı şu şekilde belirtebiliriz; ilişkilerin durumu bir dünyaya göre daha küçük bir parçadır ve dünyanın yalnızca bir yanını içerdiği düşünülebilir. Zaten cümlenin de dünyanın ancak bir parçasına ilişkin bilgi taşıdığı düşünülürse doğruluk değerinin belirlenmesi işleminin ilişkilerin durumuna göre yapılmasının nedeni anlaşılacaktır.

Bunun dışında (Löbner 2002) de anlatıldığı gibi anlam için farklı bakış açıları / basamaklar kullanmak mümkündür. Bu durum belirtilen kaynakta "Levels of Meaning" konu başlığı ile yeterince derin biçimde anlatılmaktadır. Ancak burada bizim için önemli olan bir durumu açıklamak için aynı kaynaktan faydalanarak konuya kısaca değinmekte fayda vardır.

Bir cümlenin kullanıldığı ortam ya da bağlamdan soyutlanarak, bileşenlerinin sadece sözlük manaları gözönüne alınıp değerlendirilmesi ile elde edilen anlama ifadesel anlam denmektedir. Ancak bunun dışında kullanıldığı/söylendiği bir bağlamda değerlendirilmesi ile cümlenin anlamı değişebilmektedir. Çünkü örneğin "bisiklet" kelimesi sözlük anlamında kullanıldığı gibi bir oyun kartını gösterecek biçimde anlam kaymasına uğrayabilir. Bu durum için şöyle iki senaryo[7] düşünülebilir.

SENARYO 1:

İki arkadaştan biri yürüme için uzak bir mesafeye gitmek için diğer arkadaşının bisikletini istemiş olabilir. Ancak diğer arkadaşının bazı tavırlarına sinirlenerek aralarında geçen bir konuşmanın sonunda "Senin bisikletine ihtiyacım yok!" diyebilir.

SENARYO 2:

İki arkadaş ellerinde çeşitli araçların resimleri bulunan oyun kartları ile oyun oynuyor olabilirler. Bir tanesi diğerine elindeki bisiklet resimli kartı isteyip istemediğini sorduğunda "Senin bisikletine ihtiyacım yok!" cevabını alabilir.

Her iki durumda da kullanılan kelime bisiklet olmasına rağmen gösterdikleri nesneler değişmektedir. İşte cümlenin söylendiği bağlamda değerlendirilmesi ile söylemsel anlam elde edilmektedir. Bunların dışındaki anlam seviyeleri için adı geçen kaynağa başvurulabilir.

[7] Burada verilen senaryolar (Löbner 2002)den faydalanılarak fakat değişikliğe uğratılarak hazırlanmıştır.

Burada bizim için önemli olan nokta kullanıldığı bağlama göre kelimelerin anlamının ve dolayısıyla da cümlenin anlamsal değerinin değişebilmesidir. Bu durumun Montague Yaklaşımı ile modellenebilmesi ise (Löbner 2002)de de belirtildiği gibi Olası Dünyalar kavramı ile olabilmektedir. Bu durum adı geçen kaynakta aşağıdaki gibi açıklanmaktadır;

"Bizim bu zamana kadar söylemin yapıldığı bağlam/ortam olarak adlandırdığımız şey biçimsel anlambilimde Olası Dünyalar olarak adlandırılmaktadır. Olası bir dünya bir cümlenin doğruluk değerinin bağlı olduğu bütün gerçeklerin ya da şartların toplamıdır. Olası bir dünya konuşmanın zamanını ve yerini, kimin konuşmacı kimin hitap edilen kişi olduğu gibi durumları, şahıslar topluluğunu ve ilgili gerçekleri sabitler. Bizim gerçek dünya olarak düşündüğümüz dünyaya karşılık gelen olası dünyalar olduğu gibi başka biçimde düzenlenmiş dünya modelleri de olabilir. ... Alternatif olarak gerçek olmayan dünyalar tamamen farklı olmayacak fakat bazı detaylarda değişiklik olacaktır."

Bu durum aynı zamanda önceki bölümde sözünü edip tanımlamasını yapmadığımız Frege'nin **sense – intension** ve **reference – extension** kavramlarının da açıklanmasını gerektirmektedir. Bu kavramların tarifinin o kadar kolay olmaması sebebi ile birkaç kaynaktan alıntı yaparak açıklama yoluna gideceğiz. İlk olarak (Perry 1998)de intension bir özellik ve extension ise bir küme olarak tanımlanıp sonra da örneksel bir anlatımla şahıslar için durum düşünüldüğünde ilki bir şart (veya şahıs kavramı) ikincisi ise intension'ın göterdiği şahıs olarak ifade edilmektedir. (Löbner 2002)de ise Olası Dünyalara kavramı ve matematiksel Modele Bağlı Yaklaşımı da kullanarak intension, herhangi bir ifade için olası dünyalardan birinde uygun tipte bir atama işlemi yapan fonksiyon, atamanın yapıldığı dünya modeli içinde karşılık gelen atanan değere de extension denilmektedir. Burada atanan değer ifadenin gösterdiği şahıs vb. olabilir. (Önceki bölümdeki "Ece" örneğimiz ve onun gerçek dünyada karşılık geldiği şahıs durumu gibi.) Belirtilen kaynakta anlatım daha matematiksel olup burada daha sözel bir hale dönüştürülmüştür. Ayrıca sense kavramının sahip olduğu özellikler ve konu ile ilgili tartışmalar için (Dowty vd. 1992), (Miller 2003) kaynaklarına bakılabilir.

Bu noktada (Löbner 2002)de yapılan şu tanımlamaların da faydalı olacağı kanaatindeyiz.

“Biçimsel dillere uygulandığında olası dünya bir modele karşılık gelir: referansları ve doğruluk değerlerini sabitler. ... herhangi bir ifadenin belli bir modeldeki değeri ise uygun olan olası dünyadaki extension’ıdır. Bu tip modelleri extensional modeller olarak adlandıracağız. Biçimsel bir dilin intensional modelinde ise temel ifadeler yorumlamaları olarak intension’larını alırlar.”

Bu noktadan sonra daha fazla ayrıntıya girerek biçimsel dildeki temel ifadelerin intension ve extension olarak nelere karşılık geldikleri konularına girmiyoruz. Adı geçen kaynak(lar)da bu konular geniş biçimde açıklanmıştır.

Olası dünyalar kavramının bizim programımız ve tezimiz açısından ayrı bir önemi de mevcuttur. Bu durum program tasarımımızın anlatıldığı Programımızın Genel Yapısı ve Prensipleri bölümü ve takip eden bölümlerde anlatılmaktadır.

3.3.4 Olası dünyalar kavramının getirdikleri

Olası dünyalar kavramı ve beraberinde getirdiği anlambilim yaklaşımının (bu yaklaşıma Olası Dünyalar Anlambilimi – Possible Worlds Semantics de denilmektedir) potansiyeli (Löbner 2002)de aşağıdaki gibi anlatılmıştır.

“Herşeyin ötesinde, Olası Dünyalar Anlambilimi anlamın temiz/açık bir teorisidir: Olası Dünyalar Anlambiliminde sözlükten alınan elemanlar ya da sözlük dışı elemanlar için anlam (intensional bir modelde) onun intension’ıdır. Cümleler için intension, cümlenin söylendiği bağlamı gösteren olası dünyalardan herbiri için doğruluk değeri ataması yapan fonksiyonlardır. Olası Dünyalar Anlambiliminde bir cümlenin anlamsal ifadesi onun doğruluk şartlarının bir ifadesidir. Her türlü ifade için bu yaklaşım sadece anlam (intension) kavramını değil aynı zamanda olası bir dünyada verilen bir ifadenin o dünyadaki gerçek veya potansiyel olarak gösterdiği nesne/varlık olan reference kavramını da sağlamaktadır.”

Ayrıca durum şu şekilde özetlenmektedir;

- “Sözlükten alınan veya sözdizimsel olarak karmaşık ifadenin anlamı onun intesion’ıdır
- Konuşmanın geçtiği bağlam olası bir dünyadır
- Bir ifadenin (potansiyel) olarak gerçek dünyadaki karşılığı verilen bir dünyadaki extension’ıdır.
- Bir cümlenin doğruluk şartları onun intension’ının açıklanmasıdır

Olası Dünyalar Anlambilimi şunları da sunmaktadır;

- Biçimsel bir dilde anlam gösterimi
- Cümleler ve karmaşık ifadeler için genel birleştirilebilme kurallarının ifadesi
- Gerektirme gibi mantıksal ilişkiler ve özelliklerin ifade edilmesi”

3.4 Lambda Soyutlaması ve Prolog Gösterimi

Bildirimsel cümlelerimizi yüklem mantığı ile göstermek istediğimizde fiillerimiz yükleme diğer bileşenler ise yüklemin parametrelerine dönüşmektedir. Bu durumda “Ali uyudu” ve “Ali elma yedi” cümleleri için mantıksal gösterimler aşağıdaki gibi olacaktır.

uyudu (Ali)

yedi (Ali, elma)

Ancak fiiller ilk akla geldikleri zaman gösterimimizdeki gibi özel olarak değerler almış biçimde değil ne tür parametrelere ihtiyaç duydukları ile değerlendirilirler. Yani genel bir haldedirler. Yukarıdaki örneklere bakıldığı zaman ise genel değil özel halde oldukları görülür. Yani fiilin tek ya da iki parametreli olacağı bilindiği gibi bu parametrelerin neler olacağı da sabit olarak belirlenmiştir. Örneğin “uyumak” eylemi bir kişi/varlık tarafından gerçekleştirilebilir. “yeme” eylemi ise yine bir kişi/varlık tarafından gerçekleştirildiği gibi bir de yenen varlığın ne olduğu bilgisine ihtiyaç duymaktadır. (Bu aşamada İngilizce Thematic roller olarak adlandırılan konuya girmeye gerek duymuyoruz. Konuyla ilgili başlangıç seviyesinde bilgi “Semantic Roles” başlığı ile (Tallerman 1998)de bulunabilir.)

Montague bu durumu ortadan kaldırmak için Lambda Soyutlaması adı verilen bir yönteme başvurmuştur. Lambda Soyutlaması ya da Hesabı ilk olarak Alonza Church tarafından 1940'lı yıllarda tasarlanmıştır. Lambda Soyutlaması konusunda (Penrose 2000)de şöyle bir anlatım mevcutur. "Church'ün yönteminin ana fikri özünde gerçekten soyuttur. Church'ün soyutlama olarak tanımladığı bir matematik işlemidir. ... Church'ün fonksiyonlarından birini temsil eden bir harf, örneğin x harfini ("sahte değişken" adını verdiğimiz harfi) koyuyoruz. 'x' değişkeninin yer aldığı her kare parantez, ifadenin tümünü izleyen herhangi bir simgenin yerini tutabilen bir 'boşluk' olarak kabul edilebilir. Buna göre

λx [fx]

yazdığımız zaman fonksiyonun, diyelim a'ya uygulandığında fa sonucunu verdiğini anlatırız. Başka deyişle

(λx [fx]) a = fa

Bir başka deyişle λx [fx], 'f' fonksiyonundan ibarettir. Yani:

λx [fx] = f"

Daha biçimsel bir anlatım için (Barendregt 2001)e bakılabilir. Adı geçen kaynakta son derece biçimsel biçimde Lambda Soyutlaması, sözdizim ve anlambilimsel özellikleri konusunda durulmaktadır.

Benzer biçimde Montague tarafından λ simgesi kullanılarak yüklemlerin parametreleri soyutlanmış, fiillerin yerini yüklemler almış ve gösterimde köşeli parantezden sonra gelen herhangi bir parametre ile yer değiştirmesi sağlanmıştır. Bu durumda örnek bir gösterim aşağıdaki gibi olacaktır.

λx [uyudu (x)]

Bu gösterim ile x parametresi soyutlanmış ve herhangi bir parametre ile yer değiştirebilir bir hal almıştır. Yani genel hale getirilmiştir. Bundan sonra yer değiştirme işlemi ise a ile "Ali" kastedilmek sureti ile gösterimsel olarak aşağıdaki gibi yerine getirilebilir.

λx [uyudu (x)] . a ≡ uyudu(a)

Ancak burada şöyle bir gösterimsel durumdan bahsetmek gereklidir. Montague İngilizce için yapmış olduğu çalışmalarında kelimelerin İngilizce karşılıklarının üzerine ′ işareti koyarak onları dilden bağımsız hale getirme yoluna gitmiştir. Biz burada anlatım amacı ile Türkçe gösterim kullandık ancak Programımızın Anlamsal Yapısı başlıklı bölümde de açıklanacağı gibi kodlamada Montague'ye sadık kalarak mantıksal gösterim için İngilizce kelimeleri tercih ettik. Yukarıdaki örneğimizin Montague'ye uygun halde yeniden yazılmış hali aşağıdaki gibidir.

λx [sleep′ (x)] . a ≡ sleep′ (a)

Dikkat edilirse soyutlanan parametre herhangi bir ifade ile yer değiştirebilmektedir. Ancak bu her ifade için istenen bir durum değildir. O sebeple kodlamamızda Durum Teorisi'ni ve fiillerimizin subcat listelerini kullanarak soyutlanan parametrelerin nasıl ifadelerle yer değiştirebileceği belirlenmektedir. Bu konu Durum Teorisindeki ksıtlanmış değişkenler ve programımızın yapısının anlatıldığı bölümlerde yeri geldikçe açıklanmıştır. Programımızda Lambda Soyutlamasının nasıl gerçeklendiği/kodlandığı ise yine programımızın yapısının anlatıldığı bölümlerde açıklandığı için burada konuya değinmeye gerek görmüyoruz.

3.5 Diğer Bazı Özellikler

Bu bölümde ana başlıklar altında verilmeyen bazı ayrıntılar hakkında bilgi verilecektir.

İlk olarak önceki anlatımda sözlüğe ilişkin anlambilim, daha açık hali ile kelimelerin anlamları ve diğer kelimelerle ilişkileri gibi konular üzerinde durulmamıştır. Çünkü Montague yaklaşımında herhangi bir ifadenin anlamsal değeri kendisini oluşturan

bileşenlerin anlamsal değerlerinin bir fonksiyonu olarak belirlenmektedir. Bu durumda sözdizimsel olarak analiz edilemeyen ve temel ifadeler olarak geçen yapılar anlamsal olarak da analiz edilemezler. Bu durum Montague tarafından analiz edilemeyen ifadelerin gösteriminde İngilizce karşılıklarının üzerine bir çizgi çekilerek belirtilmiştir.

Ayrıca doğruluk şartlarına bağlı anlambilimde bildirimsel cümleler olarak ifade edilen cümleler dışında cümleler ile ilgilenilmemektedir. Bizim programımızda komut verme amaçlı olarak kullanılan emir cümleleri ile ilgili işlemlerin nasıl yapıldığı program tasarımımızın anlatıldığı Programımızın Genel Yapısı ve Prensipleri bölümü ve takip eden bölümlerde anlatılmaktadır.

Bunların dışında Montague Grameri olarak da adlandırılan yapıda bulunan Tip'e Bağlı Yaklaşım, zaman gibi yapıların işin içine katılması ile ortaya çıkan iki ve daha fazla boyutlu yaklaşımlar ve bu yaklaşımlara bağlı olarak dünya modelinin seçilmesi gibi konular üzerinde durulmayacaktır. Çünkü buraya kadar anlattıklarımızın Montague Anlambilimi için genel bir bakış açısı kazandırmak ve sonraki bölümlerdeki açıklamalarımızı anlayabilmek için yeterli olduğu kanaatindeyiz. Ayrıca Kümeler Teorisi ya da başka yapılar aracılığı ile dünya modelinin kurulması, dünyadaki varlıkların/nesnelerin, ilişkilerin belirlenmesi, sözdizim kuralları, onlara denk düşen anlambilim kurallarının belirlenmesi ve anlamlandırma işleminin yapılışı konusunda örnekler verilerek ayrıntıya girilmemiştir. Bu konuda daha fazla bilgi için başlangıç düzeyinde bir çalışma olarak kabul edilebilecek olan (Kılıçaslan, Y. ve Tüysüz, M.A.A. 2002) çalışmasına bakılabilir. Daha geniş açıklamalar ise en başta da belirttiğimiz gibi temel kaynak kabul edilen (Dowty vd. 1992)de bulunabilir.

4. DURUM TEORİSİ HAKKINDA GENEL BİLGİ

4.1 Giriş

Durum Teorisi (Tın ve Akman 1994)'de anlatıldığı gibi anlamın matematiksel bir teorisidir denilebilir. Benzer biçimde (Restall 1996)da da "Durum teorisi doğal dilin anlambilimi üzerine yoğunlaşan bir disiplin olarak başladı" ifadesi kullanılmıştır. İlk olarak Barwise ve Perry (1983) tarafından *Situations and Attiudes* isimli kitapları ile ortaya atılmıştır. Klasik mantığın sahip olduğu bazı kısıtlamalar sebebiyle anlambilimde kullanılması pek uygun olmamaktadır. Montague bu durumu yüksek seviyeli mantık kullanarak ya da İngilizce karşılığı ile Intensional Logic kullanarak aşmaya çalışmıştır.

Benzer biçimde (Kılıçaslan 1998)'de durum teorisi, durumlar hakkında konuşabilmek için kullanılan soyut bir teori olarak tanımlanmıştır. Ayrıca yine aynı kaynakta durumların, gerçekliğin sınırlı parçaları olarak düşünülmesi gerektiği vurgulanmıştır.

Geleneksel model anlayışından farklı olarak durum kavramı, bir modelin belli bir parçasının durumu olarak düşünülebilir. Fakat bu durumlar başka durumların elemanları olabilirler ve bir başka durum ile ilişki içinde bulunabilirler.

4.2 Durum Teorisi ve Durum Anlambilimi Ayrımı

Durum Teorisi ve Durum Anlambilimi arasındaki ayrım konusunda (Black 1993)te aşağıdaki gibi bir anlatım mevcuttur;

"Durum Anlambilimi ([Barwise & Perry 83]) doğal dil anlambilimi konusunda olası dünyalar anlambilimine bir alternatif önerdiler. Daha sonraki çalışma teorinin matematiksel, mantıksal ve felsefi yönleri olan *durum teorisi* ve doğal dil anlambilimi hakkında teoriler olan *durum anlambilimi* arasındaki ayrımı göstermiştir."

Durum Teorisi gerçekten de Barwise ve Perry tarafından ortaya atılmış Keith Devlin gibi mantıkçılar tarafından da biçimselleştirilmeye çalışılmıştır. Bu açıdan

bakıldığında Durum Teorisini matematiksel bir teori ve Durum Anlambilimi ise mevcut teorinin doğal dil alanına uygulanması olarak düşünülebilir.

Durum anlambilimi konusunda (Perry 1998)de de aşağıdaki şekilde bir anlatım bulunmaktadır;

"Durum anlambilimi özellikle çeşitli problemli yapıların analizi için uygun olarak extensional model teorisi ve olası dünyalar anlambilimine bir alternatif olarak düşünüldü. En baştaki hali ile temel temel fikirleri:

Parçasallık : Durumlar, Her sorunun cevabını ve her önermenin doğruluk değerini belirleyen dünyaların zıddıdır. Bir durum algıladığımız, hakkında fikir yürüttüğümüz ve içinde yaşadığımız gerçekliğin sınırlı bir parçasıdır. Bu durumlarda nelerin olduğu bazı meseleler için cevapları belirleyecektir ancak hepsi için değil. ...

Gerçekçilik : Temel özellikler ve ilişkiler gerçek nesneler olarak alınırlar, ...

Anlamın İlişkisel Teorisi : ϕ gibi bir ifadenin anlamı konuşmanın geçtiği bir durum, bağlantısal bir durum ve açıklanan bir durumun bir ilişkisi olarak düşünülebilir, ..."

Daha önce Montague Anlambilimi'nden bahsettiğimiz bölümümüzde olası dünyalar ile ilgili başlık altında söylediğimiz durum (Perry 1998)de parçasallık başılığı ile ortaya çıkıyor ve bizim belirtilen konu başlığı altında verdiğimiz dünya ve ilişkilerin durumu ayrımı burada durum ve dünya kavramları için veriliyor. İlişkilerin durumu kavramı ile Durum Teorisinin temnel kavramlarından olan infon arasındaki ilişki Temel Kavramlar başlıklı bölümümüzde verilecektir. Burada önemli olan nokta Durum Teorisi'nin dünyanın bütününe ilişkin olmayışıdır.

Bir sonraki bölümde Durum Teorisi ve temel kavramları konusunda daha geniş bilgi verilirken parçasallık gibi bazı noktalar daha fazla açıklığa kavuşacaktır.

4.3 Durum Teorisinin Amacı ve Yaklaşımı

Durum Teorisinin biçimselleştirilmesi için çalışmalar yapan Keith Devlin'in (Cooper vd. 1990)da yer alan "Infons and Types in an Information Based Logic" isimli makalesinde Durum Teorisi hakkında şu şekilde bir açıklama mevcuttur.

"Durum teorisinin amaçlarından biri de (benim, durum teorisine dahil olmama neden olan, kişisel amacım) klasik mantıkta olduğu gibi doğruluk şartları üzerinde kurulu olmak yerine bilgi içeriğine dayalı bir mantığın geliştirilmesidir.

Bu biçimde bilgi içeriği üzerine odaklanan seçim klasik mantık ya da matematik alanında uygulanan klasik mantıktan oldukça farklı bir bakış açısı ile bakmamızı sağlamıştır.

Matematiksel doğrunun 'mutlak', bağlamdan bağımsız yapısı sebebi ile klasik mantık cümlelerin söylemlerine dikkat etmeyi gerektirmeyen matematiksel yapılar üzerinde uygulandılar ve iddia edilebilir ki yüklem mantığının bilgi içeriği ile olan tek alakası, matematiksel bazı yapılar altında incelendiğinde, doğruluk değeridir.

Bununla birlikte matematiksel dünyanın dışında şeyler daha farklı biçimde şekillenmektedir. Kim ne söyler, ne zaman söyler, hangi amaçla söyler, dinleyici kimdir ve konuşmacı ile dinleyen arasında ortak olan bilginin gövdesini oluşturan çeşitli arkaplan özelliklerin hepsi bir söylemde bulunan/taşınan bilgiyi belirlemekte rol oynarlar. Durumsal Mantık bu çeşitli bağlamsal özellikleri gözönüne almayı deneyen her türlü mantık için kullanılabilecek genel bir isimdir. Geliştirmeye çalıştığımız durumsal mantık, konuşmacının söylemin taşımasını istediği bilgi açısından bir söylemin bilgisel içeriği üzerinde yoğunlaşmaktadır. ..."

Burada doğruluk değeri ve matematiksel yapılar aracılığı ile yorumlamadan kastedilen Montague Anlambilimi ile yapılan çalışmanın da ana yaklaşımıdır. Benzer biçimde (Perry 1998)de de belirtildiği gibi bu yaklaşımın bazı problemlerini aşma amaçlı olarak bilgisel içerik üzerine ve parçasal bir yaklaşım getiren Durum Teorisi de anlam analizi açısından yeni bir bakış açısıdır. Tezimizin amaç bölümünde de belirttiğimiz gibi bizim tezimizin gerçekleme amaçlı olarak kullandığı yaklaşım Montague'ye dayanan

yaklaşımdır. Tezimizin bir diğer amacı da anlambilim alanında mevcut olan teorilerin incelenmesi olduğu için Durum Teorisi konusunda da yeterince genel amaçlı ve köklerini de içeren bir bilgi verme yoluna gidilmiştir. Tezimizin Sonuçlar ve Gelecekte Yapılabilecek Çalışmalar başlıklı bölümünde anlattığımız teoriler açısından bazı karşılaştırmalar ve yapılabilecek ilerletmeler konusunda gerekli bilgiler bulunacaktır. Durum Teorisi hakkında bu kadar genel bilgi verdikten sonra teoriye ilişkin temel kavramların verilmesini uygun buluyoruz.

4.4 Temel Kavramlar

Bu bölümde Durum Teorisi ve Anlambilimine ilişkin temel kavramlardan bahsedilecektir. Ancak Türkçe açısından şu durumun açıklığa kavuşturulması gerekmektedir. Durum Teorisi içinde "durum" kelimesi İngilizce "situation" kelimesi yerine kullanılmaktadır. Ancak özellikle çevirilerimizde İngilizce "state" gibi bir kelimeyi de Türkçeye "durum" olarak çevireceğimiz için anlatımda gereksiz biçimde yinelenen "durum" kelimeleri varmış gibi gelecektir. Bu durumun çözümü için referansta bulunduğumuz orijinal aynakların incelenmesi ya da bağlamdan bunun çıkarılmaya çalışılması gibi seçimi okuyucuya kalan bir durum sözkonusudur. Türkçe için mevcut durumun dışında ilerleyen kısımlarda geleceğimiz gibi durumlar soyut matematiksel yapılardır ve İngilizce için de (Barwise vd. 1991)de Keith Devlin'e ait "Situations as Mathematical Abstractions" başlıklı makalenin bir yerinde İngilizce olarak "Situations are just that: situations" biçiminde bir açıklama ile "Durumlar sadece (bildiğimiz anlamıyla) durumlardır" gibi bir anlatım yapılmıştır. Durumları ayrıntılı olarak tarif etmeye çalıştığımız noktada nasıl bir matematiksel soyutlama olduklarından da bahsedilecektir.

Konumuza başlangıç için (Kılıçaslan 1998)den alınan aşağıdaki anlatımın uygun olacağını düşünüyoruz;

"Durum Teorisi durumlar hakkında konuşmak için soyut bir teoridir. Durumlar gerçekliğin sınırlanmış parçaları olarak düşünülebilirler. Gabbay (1993)de belirtildiği gibi, geleneksel model düşüncesi ile karşılaştırıldığı zaman durum, modelin bazı parçasal

durumları olarak düşünülmelidir. ... Durumların önemli bir özelliği de bilgi tabanlı karakterizasyonlarıdır."

Anlatımdan anlaşıldığı ve bir önceki bölümde de anlatıldığı gibi durum teorisi cümlenin taşıdığı bilgi tabanlı bir yaklaşıma sahiptir. Montague yaklaşımında da cümlenin dış dünyaya ait bilgi verme amaçlı olarak kullanıldıkları düşüncesi mevcuttur. Ancak belirtildiği gibi cümlenin serfedildiği bağlam da Durum Teorisi açısından önemlidir ve taşınan bilgi buna göre yorumlanmaktadır. Montague yaklaşımında da Olası Dünyalar ile bağlamı modelleyerek bağlama ilişkin durumun aşılması mümkündür ancak daha önce de belirtildiği gibi dünya kavramı ile durum kavamı birbirinden farklıdır.

Yine (Kılıçaslan 1998)deki anlatım ile devam edecek olursak;

"Gerçekliğin, diğer parçalardan bilgi edinebilmek için algılama yeteneğine sahip olan parçalarına **organizmalar** denir. Her organizma da dünyayı kendine göre algılayarak kişiselleştirme özelliğine sahiptir. Durum teorisinin temel varsayımlarından biri de organizmalar tarafından kişiselleştirilen belli bazı değişmezler/nesneler[8] veya düzenli durumlar mevcut olduğudur. Bir organizmanın bu değişmezler/nesneler biçiminde modellenen dünyayı algılama biçimine ise kişiselleştirme biçimi denir."

Organizmadan organizmaya değişen kişiselleştirme biçimi gerçekliğin parçalarının yani durumların sınıflandırılabilmesini sağlamıştır. Sonuçta da s, s_1, s_2, ... r, r_1, $r_{2, \ldots}$ gibi gösterimler ortaya çıkmıştır. Burada kullanılan s ve r ifadeleri durumları belirtmektedir. Ancak ilerleyen bölümlerde durumların, Açıklanan Durum ve Kaynak Durum biçiminde ayrılması ile s ve r biçiminde farklı ifadelere neden ihtiyaç duyulduğu daha iyi anlaşılacaktır.

Yine aynı kaynaktan anlatıma devam edersek;

---

[8] Keith Devlin makalelerinde İngilizce olarak "objects (or uniformities)" biçiminde bir anltıma başvurmaktadır. Devlin'in "object" olarak kullandığı kelimeyi Türkçeye "nesne" olarak çevirmek mümkündür. "uniformity" kelimesini Türkçeye "düzenlilik, değişmezlik" vb. biçiminde çevirmek bizce uygun olmaktadır. Bu durumda her iki anlamı birarada kullanmayı tercih ediyoruz.

“Durum teorisinin temel değişmezleri ya da bileşenleri şahıslar, yerler, özellikler ve ilişkilerdir. Ayrıca yerler uzayda ve zamanda yer gösteren ifadeler olarak kendi içinde de sınıflandırılmaktadır. Bunun dışında, özellikler ilişkiler altında da yer alabilmektdir.”

Buradaki şahıslar, yerler ve ilişkiler (özellikleri de içererek) kullanılarak daha karmaşık yapılar elde edilebilmektedir. Yani basit yapılardan daha karmaşık yapılara gidilebilmesi mümkündür. Bu şekilde elde edilen karmaşık nesnelerin gösterilebilmesi için Geliştirilmiş Kamp Notasyonu/Gösterimi kullanılabilen notasyonlar arasındadır. Geliştirilmiş Kamp Notasyonu/Gösterimi ile ilgili olarak (Aczel vd. 1993)te yer alan Jon Barwise ve Robin Cooper’a ait “Extended Kamp Notation: A Graphical Notation for Situation Theory” isimli makalede şöyle bir anlatım mevcuttur.

“.... Diğer yandan, durum teorisinin gösterim problemini çözmeyi denedik: Karmaşık durum teorisel nesneleri kolay anlaşılır biçimde nasıl ifade edebiliriz? ... Durum teorisi tipik olarak yapısal, bilgi teorisel nesnelerin evreni olarak kabul edilmiştir. Yıllarca bir çoğumuz teorinin matematiksel tabanını kurmaya çalıştık. ... Varsayıyoruz ki Aczel 1990 ve Aczel ve Lunnon 1991 bakış açısı ile yapısal nesnelerin evreninde çalışıyoruz. Bu nesneler bileşenlere sahip olarak düşünülebilirler. Bu bileşenlere şunlar uygulanabilir;

Bileşenler (aynı türde) başka bileşenler ile yerdeğiştirebilir (Aczel 1990 da belirtildiği gibi)

Bilşenler bize yeni yeni nesneler vermek üzere soyutlanabilirler (Aczel ve Lunnon 1991 de belirtildiği gibi)”

Bu noktadan sonra ise aynı kaynakta, nesnelerin, soyulama işleminin, birleşimin, ve diğer işlemlerin Geliştirilmiş Kamp Notasyonu ile nasıl gösterilebileceği, bu şekilde bir gramer belirlemesi ile sözdizim ve anlambilim işlemlerinin nasıl yapılabileceği geniş biçimde anlatılmaktadır. Ancak bizim tezimiz açısından bu kadarının yeterli olduğunu düşünüyoruz.

Ayrıca (Kılıçaslan 1998)de de kullanılan notasyon/gösterim budur. Biz de anlatılan her iki kaynak sebebi ile tezimizde bu notasyonu tercih edeceğiz. Burada her terim bir

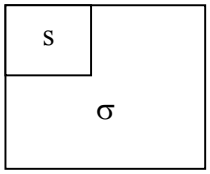
nesne türündedir. Bu kavram ise en iyi durum ve inforn kavramlarının açıklanması ile anlaşılacaktır.

Temel kavramlar konusunda açıklayacağımız son iki kavram ise bilgi parçacığı ve durumdur. Bilgi parçacığı biçiminde Türkçe'ye çevirdiğimiz infon bilgi içeren en temel birimdir. (Devlin 2001)de de mesele "Durum teorisinde, bilgi daima bir duruma ilişkin olarak ve infon adı verilen ayrık birimler biçiminde alınırlar" şeklinde belirtilmiştir. Ayrıca (Restall 1996)da yapılan aşağıdaki anlatım ile de Montague yaklaşımındaki ilişkilerin durumu kavramı ile Durum Teorisi'ndeki infon kavramının aynı şey olduğu görülmektedir.

"Barwise ve Perry'nin Situation and Attitudes'unda (bir infon'u belirten) A ilişkilerin durumu olarak tanımlanır. Daha sonra Barwise ve Etchemendy[5] ve Devlin[11] çalışmalarında, A bir infondur."

Bilgi parçacıklarının Geliştirilmiş Kamp Notasyonu dışında <<Yüklem, Arg_1, ..., Arg_n, Doğruluk>> biçiminde bir gösterimi de mevcuttur. Ayrıntılı bilgi için (Cooper vd. 1990), (Barwise vd. 1991), (Aczel vd. 1993), (Devlin 2001)e bakılabilir.

Durumun kavramı için ise çok kesin bir tanımlama yapılamayabilir ancak bilişsel özelliklere sahip bir birim tarafından kişiselleştirme amaçlı olarak kullanılan ve gerçekliğin yapısal bir parçası biçiminde düşünülmesi mümkündür. Durumlar destekledikleri temel bilgi parçacıkları ile karakterize edilmektdir. Örneğin s bir durum ve σ bir bilgi parçacığını göstermek üzere bir önerme şu şekilde oluşturulabilir.



Şekil 4.1 Durum teorisine göre bir önermenin gösterimi

Şekil 4.1 deki gibi gösterilen yüklemlere ise özel olarak Austinian yüklemleri denilmektedir. Diğer kaynaklarda durum ve bilgi parçacığından faydalanılarak farklı bir gösterim/notasyon ile nasıl yüklem elde edileceği anlatılmaktadır. O notasyona da doğrusal notasyon denilmektedir. Ancak her iki gösterimin de yorumlanması aynıdır ve s ile gösterilen durumun σ ile gösterilen bilgi parçacığını gerçek hale getirdiği biçimindedir.

Ayrıca (Barwise vd. 1991)de Keith Devlin'e ait "Situations as Mathematical Abstractions" isimli makalede temel bilgi parçacığı olarak tanımlanan infon kavramının fizikteki elktron, proton, nötron vb. kavramlara paralel olarak seçildiği ve bu şekilde isimlendirildiği anlatılmaktadır. Bunun yanı sıra matematiğin düşünme ve üzerinde çalışma amaçlı olarak soyut nesneleri kullandığı, benzer biçimde Durum Teorisi için de soyut ve üzerinde çalışma amaçlı olarak durum kavramını kullandığı matematiksel örnekler kullanılarak geniş biçimde anlatılmıştır. Ayrıca durumların teori açısından önemi ve soyut olma meselesi ise aşağıdaki gibi dile getirilmiştir.

"... bilişisel bir varlığın dünyası karşılaşılan, kastedilen, hakkında bilgi alınan vb biçiminde durumların koleksiyonuna ayrılmaktadır. ... Bu sebeple insanların davranışları da karşılaştıkları durumlara göre şekillenmektedir: hepsi oldukça farklı tepkilere yol açan tehdit arzeden, korku verici, mutluluk veren, ilgi çekici ... durumlar. ... Belirtilenler durum hakkında bilgi veriyor ancak durumun tanımının ne olduğunu vermiyor. Durumlar sadece durumlardır. Bağlam, arkapalan vb. meseleleri işleyebilmek/meseleleri ile ilgilenebilmek için kullandığımız soyut nesnelerdir."

Burada adı geçen arkaplan durum ise bizim için önemli bir kavramdır. Kısaca tanımlamak gerekirse dünya hakkında sahip olduğumuz genel bilgiye dayanarak olayların yorumlanmasıdır denilebilir. Bu durum aynı kaynakta bir yumurtanın düşmesi örneği ile açıklanmıştır.

"Örneğin, bildik bir kısıtlama ile eğer bir yumurta düşerse kırılacaktır. İlk bakışta bu durum yeterince mantıklı görünebilir. Fakat yeterince iyi bir analiz ile bunun evrensel olarak elde edilemeyeceğini görürüz. Bu tamamıyla arkaplan varsayımlarımızın bir bütünüdür. Başlangıç olarak dünyanın çekimsel durumu için geçerlidir. Ayrıca bir minimum yüksekliğin üzerindeki mesafeden düşen yumurtalar için geçerlidir. Ayrıca

pişirilmemiş yumurtalar için geçerli bir durumdur. ... Ayrıca yumurtanın düştüğü yüzey de gözönüne alınmalıdır. ... ”

Bu noktadan kısıtlamalar kavramına geçiş yapmak mümkündür. Durum teorisine göre bilgi akışı bu kısıtlamalar üzerinden yapılmaktadır. Ancak bizim anlatımımız açısından bu kadarının yeterli olduğunu düşünüyoruz. Daha fazla bilgi için adı geçen kaynağa başvurulabilir. Ancak arkaplan bilgi ile bizim tezimizde göze görülmeyen bir varlığın şekilleri hareket ettirmek ve bulundukları dünya konusunda gerekli bilgilere sahip olması örtüşmektedir. Montague yaklaşımı ve Durum Teorisi ile ilgili değerlendirme için tezimizin Sonuçlar ve Gelecekte Yapılabilecek Çalışmalar başlığına bakılabilir. Son olarak (Restall 1996)da belirtilen “Durumu bir dünyanın bir parçası olarak düşünebiliriz (bu Barwise ve Perry’nin Situations and Attitudes’teki ... yaklaşımıdır)” ifadesi ile durum ve Olası Dünyalar kavramının da içiçe olabileceğini göstermek mümkündür.

Burada dikkat edilmesi gereken ve (Devlin 2001)de belirtilen husus şudur; bilgi parçacıklarının kendisinin doğruluk ya da yanlışlık gibi değerleri mevcut değildir. Bir durum hakkında bilgi veren bilgi parçacıkları için doğruluktan ya da yanlışlıktan bahsedilebilir. Aynı durum (Kılıçaslan 1998)de şu şekilde anlatılmıştır;

“Barwise ve Etchemendy (1990) bilgi parçası ve önerme arasındaki ayrımın akılda tutulması gerektiğine dikkati çekmişlerdir. ... Bilgi parçalarının kendileri doğru ya da yanlış olabilen şeyler değillerdir. Bu değerlere bir duruma bağlı olarak sahip olabilirler ya da olamazlar. Fakat diğer yandan önermeler ise kendileri doğru ya da yanlış olabilirler”

Bunlardan da anlaşıldığı gibi bir bilgi parçacığı bir durum tarafından desteklenmelidir. Ancak bu şekilde doğruluk değeri alabilir. Çünkü bir bilgi parçacığının bir durum tarafından desteklenmesi ile ortaya çıkan yapıya önerme denilmektedir ve önermeler doğruluk değeri alabilen yapılardır.

4.4.1 Parametreler, parametrik nesneler ve değer atayıcılar[9]

Belirsiz değerler için X, Y, Z gibi büyük harflerle gösterilen parametreler kullanılabilmektedir. Bu şekilde bir nesnenin bileşenleri daha genel hale getirilebilmektedir. Daha doğrusu değişken hale getirilebilmektedir. Bu şekilde bileşenleri parametre ile değiştirilmiş nesnelere parametrik nesneler denir. Bu yapılara soyutlanmış yapılar da denilebilir. Elde edilen parameterik nesne farklı değerler alabilmektedir. Bu durumda parametreye değer atayacak olan bir yapıya da ihtiyaç vardır. Ancak bu sayede parametre uygun bir değer ile yer değiştirebilir. Bu işi yapan mekanizmaya da değer atayıcı ya da İngilizce anchor denilmektedir. Daha resmi bir tanımlama ile değer atayıcılar, bir parametre kümesinden parametre ile yer değiştirmesi mümkün olan bir nesneler alt kümesine eşleşme yapan fonksiyondur.

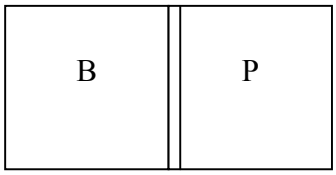
4.4.2 Sınırlandırılmış parametreler

Eğer elimizde herhangi bir parametre varsa bu parametre için sınırlı sayıda bilgi parçacığı bileşimimiz mevcut ise bu parametreye sınırlandırılmış parametre denir. Sınırlandırılmış parametrelerin bizim çalışmamız için ayrı bir önemi mevcuttur. Çünkü örneğin "Kırmızı kutuyu mavi kutunun üzerine koy" gibi bir cümlede kutu nesnelerinin özellikleri verilmiştir ve bu özellikleri sağlayan kutular üzerinde işlem yapılması gerekecektir. Her ne kadar bizim programımızda aynı renkli birden fazla kutu olması düşünülmese de yapı bu şekilde bazı sınırlayıcı şartlar ile kurularak işlemlerin yapılması düşünülmektedir. Benzer durum "kırmızının sol üst köşesine" gibi bir yapı için de geçerlidir. Yeri belirleyebilmek için cümle ile belirtilen sınırlamaların hepsinin sağlanması gerekmektedir.

Sınırlandırılmış nesnelerin gösterimi için (Kılıçaslan 1998)de de kullanılan Geliştirilmiş Kamp Gösterimi uygulanabilir;

---

[9] İngilizce'de anchor olarak kullanılan kelimeyi kullanıldığı anlam sebebi ile değer atayıcı olarak çevirmeyi tercih ettik. Bu durum konu başlığı asltındaki açıklama ile daha iyi anlaşılacaktır.



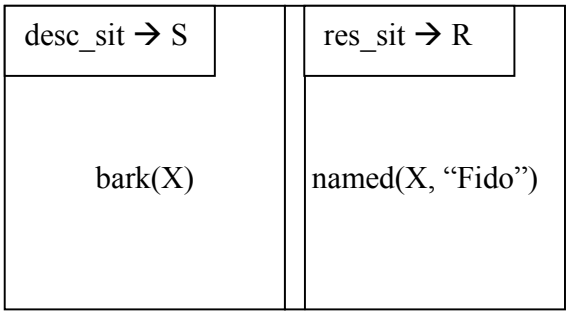
Şekil 4.2 Sınırlandırılmış nesne gösterimi

Burada B bir nesne ve P ise bir önermedir.

4.4.3 Açıklanan durum ve kaynak durum

Durum Anlambiliminde Açıklanan Durum ile Kaynak Durum arasında bir ayrım yapılmaktadır. Açıklanan Durum, bildirim cümlelerin seslendirilmesiyle hakkında konuşulan ya da başvurulan gerçeklik parçası olarak tanımlanabilir. Bu halde Açıklanan Durum cümle ile anlatılmak istenen asıl mesele olmaktadır. Kaynak Durumlar ise kendilerinden başka durumları açıklamak amacıyla faydalanılan durumlardır.

Durumu açıklamak için (Kılıçaslan 1998)de yer alan aşağıdaki örnekten faydalanılabilir. İngilizce olarak "Fido barks" biçimindeki gibi bir bildirim cümlesinin Durum Teorisine uygun olarak ve Açıklanan Durum, Kaynak Durum ayrımı yapılarak gösterimi şöyle verilmiştir.



Şekil 4.3 Açıklanan durum ve Kaynak durumun gösterimi

Daha sonra şu açıklamaya yer verilmiştir; "Şekilden de anlaşılabileceği gibi S Açıklanan Durumdur ve cümle ile öncelikli olarak kastedilen de bu durumdur. R de yine anlaşılabileceği gibi Kaynak Durum olarak görev yapmaktadır ve özel isimle belirtilen varlığı göstermektedir." Dikkat edilmesi gereken bir husus da yüklemin parametrik bir yapıya sahip olması ve kaynak durmun da parametreyi açık bir hale getirmesidir.

Ayrıca (Aczel vd. 1993)de yer alan Robin Cooper'a it "Generalized Quantifiers and Resource Situations" isimli makalede kaynak durumlar için aşağıdaki gibi bir anlatım da mevcuttur.

"Barwise ve Perry (1983) kaynak durum düşüncesini belirtili tanımlamalar (İngilizce de 'the' ile yapılan anlatımlar/tanımlamalar) ile ilgilenebilmek için duyurdular. ... her ne kadar 'the dog ran away' gibi bir cümle için evrende bir tek köpeğin olması gerekmese de açık tanımlamaların eşsizlik gereksinimini korumayı istediler."

Dolayısıyla geriye yönelik olarak hakkında bilgi sahibi olunan ve bir ilişkide alt sınır şartına bağlı olarak bulunması gereken parametreyi bu şekilde belirlemiş olduklarını söyleyebiliriz. Konu ile ilgili olarak (Cooper vd. 1990)da yer alan Keith Devline' ait makalede şöyle bir anlatım mevcuttur;

"Bizim ilişkilere bakışımız oldukça gelişmiştir, ... biz ilişkileri açık ve bazen de karışık yapıları olan soyut nesneler olarak düşünüyoruz. Özellikle, verilen bir ilişki için, ne tür varlıkların çeşitli argüman rollerini alabileceği ya da alamayacağı üzerine (uygunluk şartları olarak adlandırılan) çok açık sınırlamalarımız ve bir infon elde edebilmek için hangi argüman rollerinin doldurulması gerektiğini dayatan belirli alt sınır[10] şartlarımız vardır.

Örneğin yeme ilişkisi yiyen, yenilen, yemenin zamanı ve yeri için argüman rollerine sahiptir. Yiyen rolünü doldurabilecek şeyler sadece canlı varlıklar, yenilen şey için yenilebilen nesneler, zaman rolü için uygun bir zaman bileşeni ve yer için de yine ugun bir yer bileşeni gerekmektedir. Alt sınır şartı, bir infon elde edebilmek için yiyen ve yenilen şey rollerinin doldurulmasını gerektirir."

[10] Burada alt sınır şartları ile kastedilen İngilizce "minimality conditions" kavramıdır.

Buradan da anlaşıldığı gibi yüklem mantığında yüklemin parametrelerinin belirlenmesinde tematik rollerin kullanılması gibi bir durum burada da vardır. Bizim programımız açısından sözlüğün belirlenmesinde ve fiillerin yüklemlere dönüştürülmesinde de benzer bir yaklaşım kullanılmıştır. Ayrıca yukarıdaki anlatım ile de verdiğimiz örnekte havlama işini yapması gereken varlığın kaynak durumda belirlendiği düşünülebilir.

Bunların dışında konuyla alakalı olarak (Devlin 2001)e bakıldığı zaman Seslendirme Durumu, Kaynak Durumu ve Odak Durum diye üç farklı durumdan bahsedilmektedir. Ancak bizim tezimiz için (Kılıçaslan 1998)de faydalanılan Açıklanan Durum ve Kaynak Durum ayrımı yeterlidir. Yine aynı kaynakta yer alan İngilizce bir örnek için durum şu şekildedir. (Serbest çeviri ile Türkçeleştirirsek) "Dün koşarken gördüğüm adam kapıda" gibi bir cümle için daha önce şahit olunan bir durumun kullanımını içermektedir. Bu durum da bizim için kaynak durum olarak değerlendirilebilir.

Kaynak Durumdan faydalanılabilmesi için ortaya çıkması konusunda (Devlin 2001)de aşağıdaki maddelere yer verilmiştir.

Konuşmacı tarafından algılanmak suretiyle

Dünya hakkında ortak olarak bilinen bir bilgi nesneleri aracılığıyla,

Dünyanın içinde bulunduğu durum sebebiyle ve

Daha önceki bir konuşma vb. sebebiyle.

4.5 Ek Bilgiler ve Sonuç

Buraya kadar yapmış olduğumuz açıklamaların bundan sonra programımızın tasarımı ve çalışma ilkelerinin anlatıldığı bölüm ve sonuçların tartışıldığı bölümde vereceğimiz bilgiler için yeterli olduğunu düşünüyoruz. O sebeple tip soyutlaması (type-abstraction) gibi konulara girmiyoruz.

Bunun dışında bilgi parçacıklarının birleşimi, farkları ve daha birçok işleme değinilmemiştir. Benzer biçimde matematiksel tanımlamalardan kaçınılmış böylece genel bir bilgi kazandırılarak biçimsel kısımdan biraz uzak kalınmıştır. Ancak Durum Teorisi'nin matematiksel tabanı da bugün oturmuş durumdadır.

Bu konular ve daha fazlası hakkında ayrıntılı bilgi adını verdiğimiz kaynaklardan edinilebilir.

5 PROGRAMIMIZIN GENEL YAPISI VE PRENSİPLERİ

5.1 Giriş

Bundan sonraki bölümlerde sırası ile programımızın sözdizimsel ve anlamsal yapıları açıklanacaktır. Ancak bu açıklamalardan önce genel yapı ve programımızı hangi düşünceler çerçevesinde hazırladığımızın açıklaması bu bölümde yapılacaktır.

Öncelikle daha önceki bölümlerde de ara ara bahsettiğimiz "küçük", "kapalı" ya da "kendine has" dünya kavramının bizim için ne ifade ettiği ve programımızda bundan nasıl faydalandığımızı açıklayacak sonra kod yazımında kullandığımız tekniklerden bahsedeceğiz. Programımızın tam açıklaması ise bölümümüzü takip eden sözdizimsel ve anlamsal yapısının anlatıldığı bölümlerde verilecektir.

5.2 Programımızın Dayandığı Temel Prensipler

Daha önce de Örnek Bir Program Olarak SHRDLU bölümünde ele alındığı gibi esin kaynaklarımızdan biri olan SHRDLU kapalı/küçük bir dünya modeline sahiptir. Yani biz insanlar kadar gelişmiş bir dünya görüşü yoktur. Bildiği/tanıdığı cisimler program tarafından belirlenmiş ve tüm dünyanın kendi bildiği ile sınırlı olduğu varsayılmıştır. Bu durumda programın yapısı da belirttiğimiz modele göre oturtulmuştur. Sonuçta bir robot kolu simüle edildiği için de belirtilen yapı yeterli olmuş program yapay zeka alanında başarılı ve örnek bir çalışma olarak değerlendirilmiştir.

Bizim de programımızın sözlüğü, sözdizim amacıyla tanıyacağı/analiz edebileceği cümle çeşitleri yapacağı işe göre şekillenmiştir. O sebeple Türkçe'nin ancak çok küçük bir kısmını kullanarak gerekli işlemleri yerine getirmektedir. Ancak burada savunmakta olduğumuz temel ilke şudur; Bir programın akıllı ya da görevini iyi yerine getirdiği kabul edilen bir durumda olması için biz insanlar gibi gelişmiş bir yapıya sahip olmasına gerek yoktur. Tüm yapısı yapacağı işe göre şekillendirilebilir. Bunu söylerken temel varsayımımız yine belirttiğimiz gibi akıllı olarak nitlendirilebilmesinin yapması gereken işi ne kadar doğru biçimde yaptığı ile belirlenmesidir.

Bilişsel bilimlerin en yakın bilim dallarından olan Psikoloji'de insanların da dünyayı olduğu gibi algılamadıkları ve içsel bir dünya modeline dolayısıyla da kendilerine has bir bakış açısına sahip oldukları vurgulanmaktadır. Dolayısıyla programların, robotların ya da makinaların yapacakları işe göre özelleşmiş dünya anlayışlarının olmasının hiçir sakıncası olmadığı gibi kendi işleri açısından en az maliyetle en iyi performansın elde edilmesini de sağlayacağı kanaatindeyiz.

Bizim programımızda ise belirtilen durum sıralayacağımız aşamalardan geçilerek elde edilmiştir. İstanbul Bilgi Üniversitesi 2002 – 2003 dönemi ikinci ve üçüncü sınıf öğrencileri ile yapılan anketlerde öğrencilere şekilleri hareket ettirmek için kullanılabilecekleri cümleleri yazmaları istenmiştir. Bunun sonucu olarak elde edilen cümlelerden öncelikle kelimeler ayrıştırılarak programımızın sözlüğü elde edilmiştir. Elde edilen sözlükte kelimeler yalın bir sıra ile değil sınıflandırılarak (isim, sıfat vb.) bulunmaktadır. Bu ayrıştırma işleminden sonra kullanılan cümle yapılarının analiz edilmesi ile sözdizimsel olarak elde edilen kelime sınıflarından başka ne tür sözdizimsel öbek oluşturma kurallarına gereksinim olduğu ortaya çıkarılmıştır. Böyle bir simülasyon programı için ilk akla gelen cümle yapısının emir kipinde olduğu belirlenmiş ve esas olarak bu cümleler ile ilgilenilmiştir. Dolayısıyla programımız emir cümlelerini işleyecek yapıda hazırlanmıştır.

Dediğimiz gibi girilebilecek cümle yapıları analiz edilerek uygun sözcük öbeği oluşturma kuralları çıkarılmış ve sözdizim ağaçları oluşturulmuştur. Burada esinlenmiş olduğumuz bir diğer çalışma da İngilizce "chatbot" Türkçe olarak da "karşılıklı sohbet motoru" diyebileceğimizi programlar arasında en başarılılarından biri olarak kabul edilen ALICE'dir. Bu alanda Turing Testi'ni geçmeye en çok yaklaşan programlara yani insana en yakın cevapları vererek sohbet eden programlara yönelik Loebner Prize isimli bir yarışma bulunmaktadır ve ALICE bu yarışmayı 2000 ile 2001 yıllarında kazanmış olan programdır. Konuyla ilgili daha ayrıntılı bilgi http://www.loebner.net/Prizef/loebner-prize.html adresinde bulunabilmektedir. ALICE'in yapısında AIML isimli bir yapay zeka dili kullanılmış olmakla birlikte esas dayanılan temel sıradan / gündelik sohbetler esnasında kullanılan cümlelerin yapısının değerlendirilmesi ve bu yapıya uygun tasarımın

kullanılmasıdır. Örneğin sıradan konuşmalarda geriye yönelik olarak fazla soru sorulmadığı düşünülerek gelişmiş bir belleğe gereksinim duyulmadığı düşünülmüş o sebeple de konu ile ilgili geçmişte söylenen bir cümlenin yeterli olacağı düşünülerek tek seferlik hafıza gibi bir yapı AIML'de kullanılmıştır. Bunun dışında ALICE İngilizce için yapıldığından İngilizce'de hangi kelimelerden sonra hangi kelimelerin gelebileceğine dair bir istatistik çıkarılarak programın yapısı kurulmuştur. ALICE, AIML ve diğer konularda http://www.alicebot.org/ adresinden gerekli bilgi online olarak alınabilmektedir. Benzer biçimde bizim programımızda yapılan anketler ile elde edilen sonuçlar ışığında şekillenmiş bir yapıya ya da kapalı dünya modeline sahiptir dememizin bizce bir sakıncası yoktur.

5.3 Programımızın Kodlaması Hakkında

Kodlamada yer yer İngilizce sözcükler ve kısaltmaların kullanılması da yine kolaylıkla farkedilecek bir durumdur. Bu durumun başlıca iki sebebi bulunmaktadır. Bunlardan ilki faydalanılan teorilere sadık kalmak ve onların gösterimlerini tercih etmektir. Örneğin Özellik Değer Matrislerindeki özellikler İngilizce sözcüklerin kısaltması biçiminde tanımlanmıştır. Yine aynı biçimde fiillerin anlamsal gösterimleri için İngilizce kelimeler seçilmiştir. Bu noktada da Montague'nin kendi gösterimine sadık kalınmıştır. Bir diğer nedense doğru Türkçe karşılıkların bulunamaması ve kazanılmış olan kodlama alışkanlıklarıdır. Örneğin İngilizce "V-bar" olarak adlandırılan yapı Türkçe'ye çevirilememiş ve programımız içinde aynen kullanılmıştır. Ayrıca kodlama standartları ve isimlendirmeler konusunda genellikle yabancı makaleler ile standartlar benimsendiği için program içinde de benzer kullanımlar mevcuttur. Örneğin "nesne" yerine "object"in zaman zaman kulanılmış olması gibi. İlerleyen bölümlerde bu tip kullanımlara rastlanılacağı için önceden bilgi vermek ve durum dolayısıyla af dilemenin faydalı olacağı kanaatindeyiz.

5.4 Prolog Hakkında

Prolog programlamaya ilişkin olarak gerektiğini düşündüğümüz açıklamalar gereken bölümlerde ve gereken yerlerde yapılmıştır. Çünkü programı değerlendirecek

kişilerin açıklamalarımızın yanında temel de olsa Prolog bilgilerinin olması gerektiğine inanıyoruz.

6. PROGRAMIMIZIN SÖZDİZİMSEL YAPISI

6.1 Giriş

Tezimizin bu bölümünde programımızın sözdizimsel yapısı ayrıntıları ile açıklanacak ve yaptıklarımızın gerekçeleri anlatılacaktır. Programımız iki aşamada geliştirilmiştir. İlk olarak sadece sözdizimsel analiz yapması sağlanmış, sonra yapılan sözdizimsel analize anlambilimsel eklemeler yapılmıştır. Bu bölümde programımızın sadece sözdizimsel özellikleri ile ilgilenilecek anlamsal kısmı bir sonraki bölümde ele alınacaktır. O sebeple bölüm içinde verdiğimiz örneklerimizde sem(antics) özelliğinin de boş olmadığı görülecektir. Ancak anlambilimsel anlatım bu bölümde yapılmayacaktır.

Bu bölümde sırasıyla programımızın genel olarak parçaları, sözlüğündeki kelimelerin çeşitleri ve sahip oldukları sözdizimsel özellikler, sözcük öbeği oluşturma kurallarımız, ayrıştırıcımız ve çalışma şekli, son olarak da cümle biçimindeki girişlerin nasıl yapılabileceği anlatılacaktır. Bölümümüz ve sonrasındaki Programımızın Anlamsal Yapısı isimli bölüm bir bütünlük oluşturmaktadır. Bölümümüz altında anlatılan özellikler anlamsal yapının anlatıldığı bölümde tekrarlanmayacak sadece eklenen yeni özellikler anlatılacaktır. O sebeple programımızın anlaşılabilmesi için iki bölümün birlikte değerlendirilmesi gerekecektir. Çünkü özellikle anlam analizi kısmının anlaşılabilmesi için gerekli bazı kısımlar bu bölüm altında bulunacaktır.

6.2 Kodumuzun Genel Yapısı

Programımızda öncelikle sözlüğümüz sonra sözcük öbeği oluşturma kurallarımız onun sonrasında ayrıştırıcımız ve en son olarak da kullanıcının daha kolay cümle girişi yapabilmesi için düz cümle girişini sağlayan bir parça bulunmaktadır. Sonraki başlıklarda bu bölümler ayrıntıları ile anlatılacaktır.

6.3 Sözlüğümüzdeki Kelimeler İçin Genel Yapı

Bizim programımız açısından kelimelerin değerlendirilmesinde daha doğrusu sözlüğe girişlerinde kullanılan yöntem Özellik Değer Matrisi olarak adlandırılmaktadır. Bu yöntemde adındanda anlaşılabilceği gibi kelimelerin sahip oldukları özellikler ve bunlara karşılık verilen değerler bulunmaktadır. Özellik Değer Matrisleri konusu ile ilgili giriş düzeyinde bilgi HPSG çerçevesinde hazırlanmış ve sağlık sistemleri için düşünülmüş olan Hermes sisteminin anlatıldığı (Rivera ve Cercone 1998)de bulunabilir. Bizim sözlüğümüzdeki kelimelerde kaynakta anlatıldığı kadar geniş bir özellik kümesi kullanılmamış programımızın yapacağı işe göre şekillenmiş bir yapı tercih edilmiştir. Ancak genel yapı olarak bizim sözlüğümüz de Özellik Değer Matrisini kullanıyor demek yanlış olmaz. Zaten tezimizin temel varsayımlarından biri olan kendine özgü dünyalar ya da SHRDLU'yu anlattığımız bölümdeki gibi küçük dünyalar kavramı bize bunu yapmak konusunda izin veriyor diye düşünüyoruz. Bizim kendine özgü dünyalar konusundaki düşüncelerimiz için Programımızın Genel Yapısı bölümünde anlatılmaktadır.

Tezimizde sözlük kelimesi ile kastettiğimiz İngilizce "lexicon" kelimesidir, "dictionary" kelimesi değil. Dolayısıyla sözcük olarak kasttetiğimiz de İngilizce "lexeme" kelimesidir. En kısa hali ile kastettiğimiz kelimelerin dilbilimsel olarak teknik kavramlar olduklarını ve bizim günlük hayatta kullandığımız sözlük ve sözcük kavramından farklı olduklarını söyleyebiliriz. Konu ile ilgili olarak (Löbner 2002)deki anlatıma başvurulabilir. Dolayısıyla sözcüklerimizin program içindeki girişlerinde de "lexeme" terimi kullanılmıştır.

Bizim sözlüğümüzdeki kelimelerin sahip oldukları özellikler ise şunlardır;

1. phon
2. syn
    a. cat
    b. case
    c. subcat
3. sem
    a. par
    b. pred
    c. spec

Saydığımız elemanların sırası ile açıklaması şöyledir; phon(ology) kelimenin yazılışını yani kelimenin ne olduğunu belirlemektedir.

syn(tax) kelimenin sözdizimsel özelliklerinin ne olduğunu belirlemektedir. Syn kısmının sahip olduğu alt özelliklerden cat(egory) kelimenin cinsini (isim, sıfat, fiil vb. olarak) belirlemektedir. Case bilgisi kelimenin hangi halde bulunduğunu belirtmektedir. Programımızda kelimelerin morfolojik özellikleri ile ilgilenilmemektedir. O sebeple takı analizi gibi bir durum söz konusu değildir ve kelimeler aldıkları ekleri ile birlikte sözlüğe girilmiştir. Türkçe için Morfolojik analiz açısından (Çetinoğlu 2001) çalışması ve orada belirtilen referanslara başvurulabilir. Subcat(egorization) ise sözcüğün doğru kullanımı açısından birlikte kullanılması gereken kelimenin sözdizimsel özelliklerini belirlemektedir. İçeriği syn ve sem(antic) özelliğinin birleşimidir.

Sem(antic) özelliği kelimenin yüklem olarak taşıdığı/taşıyacağı anlamı gösterme amaçlı olarak düşünülmüştür. Yukarıda par(ameters) ve pred(icate) biçiminde adı geçen alt özellikler anlambilimsel kısımda açıklanacaktır. Ancak spec(ifies) özelliği bizim için ayrıca önemlidir. Çünkü kelimenin neyi kastettiğini ya da gösterdiğini belirtmektedir. Bu durum şöyle açıklanabilir "mavi" gibi bir kelime ilk bakışta sıfat olarak değerlendirilebilmektedir. Ancak "Mavi sağa kaysın" gibi bir cümlede bir nesneyi göstermektedir. O sebeple gösterdiği şey (bizim programımız için) bir nesne olmaktadır. Bu kullanımın sözcük öbeği oluşturma kuralları açısından sağladığı faydalar ise kuralların anlatıldığı bölümde açıklanacaktır.

6.4 Sözlüğümüzdeki Kelimelerin Çeşitleri

Sözlüğümüzdeki kelimlerimiz öncelikle ana başlıklara ayrılmış sonra gerekirse bu başlıklarda kendileri altında yeni alt başlıklara ayrılmışlardır. Ana başlıklarımız şunlardır;

1. Bağlaçlar
2. İsimler
3. Sıfatlar
4. Fiiller
5. Yön belirten kelimeler

6. Konum belirten kelimeler
7. Yer belirten kelimeler

İlk bakışta konum ve yer belirten kelimelerin ayrı ayrı belirtilmiş olması bir hata gibi görünse de şu anda özellikle sözcük öbeği kurallarımız ve fiillerimizin yapısı sebebiyle böyle bir ayrıma ihtiyaç duyduğumuzu belirtmekle yetineceğiz. Saydığımız konu başlıkları altında da ayrıntılı açıklamalar bulunmaktadır.

Şimdi sırasıyla sözlüğümüzdeki bu kelime çeşitlerini inceleyecek, kendilerine neden ihtiyaç duyduğumuzu, yapılarının nasıl olduğunu ve bu yapıya neden ihtiyaç duyduğumuzu açıklayacağız.

Ancak şu unutulmamalıdır ki kodumuzda herşey programımızın yapısına göre özelleştirilmiştir. Bu durum kelimelerin girişlerinden sözcük öbeği kurallarına kadar her kısımda geçerlidir.

Bizim programımızın da sözlüğü (Çetinoğlu 2001)de olduğu gibi kısıtlıdır. Ancak bizim programımızın amacı hedeflenen işlemin anlam analizinin yapılabileceğini göstermek ve genişletilebilir bir özelliğe sahip olarak örnek teşkil etmektir. Hem sözlüğümüz hem de ayrıştırıcımız gerekli sözcük öbeği kurallarının da eklenmesi koşuluyla geliştirilebilir ya da ileride yapılacak çalışmalara örnek teşkil edebilir bir durumdadır. Çünkü bizim şiekillerimizin büyüklük vb. özelliklerinin aynı olduğu varsayılmıştır.

6.4.1 Bağlaçlar

Bu kısımda “ile” ve “ve” kelimeleri bulunmaktadır. “X ile Yyi yerdeğiştir”, “X ve Yyi birleştir” gibi cümleler açısından düşünülmüşlerdir. İki isim arasında kullanılırlar ancak kodda subcat listelerine bakıldığı zaman bir tane isim istedikleri görülmektedir. Bunun nedeni kendilerinden önce gelen isim ile birleşerek bir isim öbeği oluşturmalarıdır. İkinci isim ile de fiilin birleşerek bir fiil öbeği oluşturması biçiminde kurallar kullanılmıştır.

6.4.2 İsimler

Kodumuzun bu kısmında “mavi”, “kırmızı”, “yeşil” kelimeleri sıfat olmalarının dışında “Yeşil yokolsun” gibi bir cümlede isim olarak da kullanılabildikleri için isim biçiminde belirlenmişlerdir. Ayrıca bu kelimelerin –i, -e, -nin biçiminde ek almış halleri de isim olarak girilmişlerdir. Bu kelimelerin hepsi isim olarak değerlendirilmiştir ancak anlam açısından aldıkları önermeler özellikle –nin biçiminde sahiplik eki almış olanlar için farklı olacaktır. Bunların dışında “pencerenin”, “sistemin”, “ekranın” biçiminde girişler mevcuttur. Ayrıca yine belirleyici bir sıfat ile kullanıldıkları zaman bir nesne gösterme özelliğine sahip olan “olan”, “kutu”, “kutucuk” kelimeleri de –i, -e, -nin eki almış halleriyle birlikte bulunmaktadır. Bu kelimelerin subcat listelerinde ise nesne belirten (spec özelliği object olan) bir sıfat istedikleri belirtilmiştir. Örneğin “kırmızı olan”, “mavi kutucuk” gibi yapılarda bizim programımız için ayırdedici özellik olan renk bilgisi ile kullanıldıkları zaman ancak doğru biçimde yorumlanabilmektedirler. Bu sebeple de subcat listeleri belirttiğimiz biçimde ayarlanmıştır.

6.4.3 Sıfatlar

Programımızda şekillerin ayırdedici özellikleri olarak renklerinin kullanılması düşünüldüğü için sadece “kırmızı”, “yeşil”, “mavi” biçiminde üç renk için sıfat girişleri yapılmıştır. Programımızdaki şekillerin boyut vb. özelliklerinin aynı özellikte olduğu varsayılmış olması sebebiyle bu sıfatlar bizim için yeterlidir.

6.4.4 Fiiller

Programımız açısından yapılacak işi ve bu iş için gerekli elemanları belirleyen en önemli sözcük grubudur. Kendi içinde tek, çift ve üç parametreli olarak sınıflara ayrılmaktadır. Bu sınıflamada fiillerin önerme biçiminde gösterildiğinde kaç parametre aldıkları göz önüne alınmıştır. Örneğin “gizlensin” gibi bir fiil düşünüldüğünde akla “Kim gizlensin?” gibi bir soru gelmektedir. Benzer biçimde “götür” gibi bir fiil için “Kimi nereye götüreyim?” gibi bir soru akla gelmektedir.

En başta da belirttiğimiz gibi programımız açısından yapılacak işi belirlemede en önemli kelime grubu fiillerdir. Çünkü istedikleri bileşenlerin hangi hallerde (case parametresi ile belirlenen yapı) olmaları gerektiğini de yine fiiller belirlemektedir. Zaten bir fiil akla geldiğinde istediği bileşenler düşünülürken bu durum da kendiliğinden ortaya çıkmaktadır. İkinci örneğimiz için düşünecek olursak "götür" gibi bir fiil için "Kimİ nereyE götüreyim?" diye bir soru akla gelmekte ve bu sorudaki ifadeye bakılacak olursa götürülecek nesne bilgisini –i halinde götürülecek yer bilgisini ise –e halinde istediği ortaya çıkar. Somut cümleler ile düşünülecek olursa "Maviyi yeşilin sağına götür" gibi bir cümlede nesne –i halinde yer bilgisi ise –e halindedir. Burada yer bilgisi en basit hali ile "sağa" ya da "sola" biçiminde olabileceği gibi "mavinin sol alt köşesine" biçiminde daha karmaşık bir biçimde de olabilir. Zaten programımızda yön, konum ve yer şeklinde ayrı sözcük sınıflarının ve bunlara ilişkin sözcük öbeği oluşturma kurallarının bulunmasının sebebi de fiilin yer ifadesi gereksinmesi ve bu ifadelerin basitten zora doğru giden yapılarının olmasıdır. Bu kısım Sözcük Öbeği Oluşturma Kuralları bölümünde daha net biçimde açıklanacaktır.

Ayrıca bazı fiillerin birden fazla biçimi bulunması halinde uygun başlıklar altında tekrar uygun subcat listeleri ile girişleri yapılmıştır. Aşağıda fiiller için sınıflandırma anlatılırken verilen örneklerde de farklı sınıflarda aynı fiili görmek mümkün olacaktır ve bu durum fiilin önerme olarak aldığı parametre sayısının ya da parametre olarak istediği ifadelerin hallerinin (case) değişmesi sebebiyle ortaya çıkmıştır.

Yukarıdaki anlatımdan faydalanarak fiillerimizin sözlükteki sınıflandırması şu şekilde anlatılabilir;

1. KİM biçiminde yalın halde ve nesne belirten bir ifadeye gereksinimi olan fiiller (Örneğin; gizlensin, saklansın, kaybolsun)

2. KİMİ biçiminde –i halinde ve nesne belirten bir ifadeye gereksinimi olan fiiller (Örneğin; büyüt, küçült, yoket)

3. KİMİ, NEREYE biçiminde –i halinde, nesne belirten ifadeye ve –e halinde yer belirten bir ifadeye gereksinimi olan fiiller. Bu grup sözlüğümüzün en geniş kısmını oluşturmaktadır. (Örneğin; koy, daya, yasla, yaklaştır, gönder, bırak, getir, götür, kaydir, sürükle, yerleştir, yolla, al, konumlandır, it, yapıştır.)

4. KİM, NEREYE biçiminde yalın halde nesne belirten bir ifadeye ve –e halinde yer belirten bir ifadeye gereksinim duyan fiiller. (Örneğin; kaysın, gitsin)

5. KİMİ, KİME biçiminde –i halinde nesne belirten bir ifadeye ve –e halinde nesne belirten ikinci bir ifadeye ihtiyaç duyan fiiller. (Örneğin; yapıştır, yasla, yaklaştır, daya.)

6. KİM bağlaç KİMİ biçiminde ilki yalın halde nesne belirten bir ifadeye ve bağlaçtan sonra –i halinde nesne belirten bir ifadeye ihtiyaç duyan fiiller. Burada bağlaç olrak kullanılabilecek kelimelerimiz "ile" ve "ve" biçimindedir. Sözlüğümüzde yer alan fillerimiz ise "birleştir" ve "yerdeğiştir"dir.

7. KİM ile KİM biçiminde ikisi de yalın halde birer nesne gösteren ve "ile" bağlacı ile bağlanmış iki nesne ifadesine ihtiyaç duyan fiiller. Sözlüğümüzde bu biçimde bir kullanıma sahip tek fiil "birleşsin"dir.

8. KİM, KİME DOĞRU biçiminde yalın halde nesne gösteren bir ifadeye ve –e halinde nesne bildiren bir ifadeye ihtiyaç duyan fiiler. Burada "doğru" gibi bir ifadenin kullanımı gereklidir. Yön bildiren kelimeler anlatılırken durum açıklanacaktır. Sözlüğümüzde bulunan belirtilen yapıdaki fiiller "ilerlesin, "kaysın, "gitsin"dir.

9. KİMİ, KİME DOĞRU biçiminde –i halinde nesne gösteren bir ifadeye ve –e halinde nesne bildiren bir ifadeye ihtiyaç duyan fiiler. Burada "doğru" gibi bir

ifadenin kullanımına izin verilmiştir. Sözlüğümüzde bulunan belirtilen yapıdaki fiiller “kaydır”, “getir”, “götür”, “sürükle”, “yolla”, “yaklaştır”, “gönder”dir.

10. KİM, KİME biçiminde yalın halde nesne belirten bir ifadeye ve –e halinde yine bir nesne belirten ifadeye gerek duyan fiiler. Sözlüğümüzde bulunan belirtilen yapıdaki fiiller “değsin” ve “dokunsun”dur.

11. KİM, KİMİN, NERESİNE biçiminde üç parametreli gibi görünen ancak gerçekte iki parametreye ihtiyaç duyan fiiler. Burada KİMİN kısmı bizim rogramımızdaki yer ifadeleri tarafından NERESİNE kısmı ile birleştirilerek bir parametre haline getirilmekte ve bu şekilde kullanılmaktadır. Bu durumun neden varolduğu gerekli sözcük öbeği oluşturma kuralı açıklanırken anlatılacaktır. Bu yapıdaki fiilerimiz yalın halde nesne belirten bir ifadeye ve –e halinde yer gösteren bir ifadeye gereksinim duymaktadırlar. Sözlüğümüzdeki fiillerimiz daha önceden de farklı girişleri yapılmış olan “değsin” ve “dokunsun”dur.

12. KİM, KİME, NEREDEN biçiminde üç parametreye ihtiyaç duyan fillerimiz. Bu fiillerimiz önce yalın halde nesne gösteren bir ifadeye, -e halinde yine nesne gösteren bir ifadeye ve –den halinde yer gösteren bir ifadeye gereksinim duymaktadırlar. Sözlüğümüzde bu şekilde kulanılabilecek fiiller olarak giriş yapılanlar yine “değsin”, “dokunsun”dur.

13. KİM ile KİMİ, NEREDEN biçiminde girşimiz yine “ile” için özelleşmiş bir yapıya sahiptir. “ile” ilgili açıklamalarımız sözcük öbeği oluşturma kısmında bulunabilir. Bu durumdaki tek fiilimiz “birleştir”dir. İstediği nesne belirten ifadelerden ilki yalın halde ikincisi –i halindedir ve son olarak da –den halinde bir yer ifadesine ihtiyaç duymaktadır.

14. KİM ile KİM, NEREDEN biçimindeki fiilimiz de bir tanedir ve “birleşsin”dir. Bir önceki maddeden farklı olarak ihtiyaç duyduğu ikinci nesne belirten ifadenin de yalın olmasını istemektedir.

Fiillerimiz açıklarken kullanığımız nesne belirten ifade ve yer ifadesi tanımları özellikle sözcük öbeği kuralları ile daha iyi anlaşılacaktır. Ancak nesne belirten ifadeler için İsimler konu başlığıunda anlatılan “olan”, “kutu”, “kutucuk” gibi kelimeler ile nesnelerimizin tek ayırdedici özellikleri olan renk bilgisinin birleşimi ile oluşan “mavi kutu”, “yeşil olanı”, “kırmızı kutucuğa” gibi yapılar akla gelmelidir. İfadelerin halleri ise ikinci sözcüğün hali olarak belirlenmiştir. Bu durum Sözcük Öbeği Oluşturma Kuralları başlıklı bölümde tekrar ele alınacaktır.

Yer ifadeleri ile kastettiğimiz ise daha önce de açıkladığımız gibi sözcük öbeği oluştrma kuralları vasıtasıyla elde edilebilen basit ya da karmaşık yer bildiren ifadelerdir. Bu durum da yine örneklerle birlikte Sözcük Öbeği Oluşturm Kuralları başlığı altında anlatılacaktır.

6.4.5 Yön belirten kelimeler

Diğer bir kelime grubumuz ise yön belirten ifadeleri içermektedir. Bunlar “sağ”, “sol”, “alt” ve “üst” biçimindedir. Ayrıca “doğru” kelimesi de yön belirten bir kelime olarak girilmiştir. Bu kelimenin bizim için ayrı bir anlamı bulunmaktadır. Çünkü “sol alt köşesine doğru” biçiminde kullanılabileceği gibi “yeşile doğru” biçiminde de kullanılabilmektedir. Bu sebeple “doğru” kelimesi için iki ayrı giriş bulunmaktadır. Birincisinin subcat listesinde ilk örneğimizde olduğu gibi bir yer ifadesi beklediği ikincisinin subcat listesinde ise ikinci örneğimizde olduğu gibi nesne belirten bir ifade beklediği belirtilmiştir. Her iki durumda da subcat listesinde istediği ifadenin –e halinde olması gerekmektedir. (Çetinoğlu 2001)de de bazı kelimeler aldıkları eklere göre morfolojik olarak özel bir hal alırsa sözlüğe ikinci bir giriş yapılması yoluna gidilmiştir ve biz de tezimizde benzer bir yolu seçtik. Bu durum yalnızca “doğru” kelimesi için değil daha önceden de belirttiğimiz gibi fiillerimiz ve diğer kelime çeşitlerimiz için de geçerlidir.

Burada sözlüğümüze direk olarak girişini yaptığımız yön ifadeleri basit olarak yön belirtmektedir. Bunun dışında "sol alt" ya da "sağ üst" gibi biraraya gelerek daha karmaşık yapılar kurmaları ile ilgili bir sözcük öbeği oluşturma kuralı mevcuttur.

6.4.6 Konum belirten kelimeler

Sözlüğümüzün bu kısmında iki kelime ve bu iki kelimenin iki farklı hali yer almaktadır. Bunlar sırası ile "köşesine", "yanına", "köşesinden" ve "yanından" biçimindedir. Hepsinin subcat listelerinde yön belirten ifadelere ihtiyaç duydukları belirtilmiştir. Bu şekilde sözcük öbeği kuralları yardımıyla "sağ köşesine" ya da "sol alt yanına" gibi yapıların oluşturulabilmesi sağlanmıştır. Zaten kelimelerin aldıkları eklere bakıldığı zaman da yine başka ifadelere ihtiyaç duydukları görülmektedir. Bunun dışında tek başlarına bir yer belirtmeleri durumuna karşı da Yer Belirten Kelimeler başlığı altında da subcat listeleri dğiştirilerek bildirilmişlerdir. Çünkü bu halleri ile "mavinin köşesine" gibi bir kullanıma izin vermemektedirler. Belirtilen problem bir sözcük öbeği oluşturma kuralı ile halledilmiştir.

6.4.7 Yer belirten kelimeler

Yer belirten ifadeler çeşitli öbek kuralları kullanılarak oluşturulmuştur ancak bunların dışında doğrudan sözlüğe giriş yapma yöntemiyle de bir kısmı tanımlanmıştır. Konum olarak belirtilen dört sözcük aynı zamanda yer belirten sözcükler olarak da sözlüğe girilmiştir. Ancak yer belirten bu kelimeler de kendi içinde referans nesneye ihtiyaç duyanlar ve duymayanlar olarak sınıflandırılmıştır. –e halinde bulunan "köşesine", "yanına", "sağına" ve "soluna" biçimindeki kelimelerin –nin ekine sahip nesne gösteren ifadelere ihtiyaç duydukları subcat listelerinde belirtilmiştir. Bu şekilde "mavinin sağına" gibi öbekler oluşturulmaktadır. "köşesinden", "yanından", "sağından" ve "solundan" biçimindeki kelimelerin subcat listeleri boştur. Çünkü referans nesne gerektirme durumları ayrı bir öbek oluşturma kuralı ile belirlenmiştir. Ayrıca "sağa" ve "sola" biçiminde kelimeler de mevcuttur. Ancak bu kelimeler "Mavi sağa gitsin" gibi referans nesneye ihtiyaç duymayan durumlar için uygundurlar ve subcat listeleri boştur.

6.5 Sözcük Öbeği Oluşturma Kuralları

Programımızda basitten zora doğru yapıları çözümlemek ve ayrıştırıcının kendilerinden faydalanarak cümleyi gramer olarak analiz etmesini sağlayan bir takım sözcük öbeği oluşturma kurallarımız vardır. Biz bu kuralların açıklamasında önce genel olarak bir sınıflandırma yapıp, bu sınıfların içindeki kuralları genel halleri ile vermeyi anlaşılabilirlik açısından uygun gördük. Bazı durumlarda kuralların sıralamasına güvenilerek koda yerleştirilmişlerdir. Bu da (Bratko 2001)de belirtildiği gibi Prolog programlamada programın çalışma zamanındaki doğruluğunu ve performansını etkileyen bir özelliktir. Ancak bazı durumlarda daha az iş yapmak için sırlamaya güvenilebilir ve bizim programımızda da fiiller için böylesi bir durum söz konusudur. Ayrıca sözlük girişlerinin açıklandığı bölümde kısaca değinilen işlemler burada açıklanacaktır. Açıklamalarımızda programımızdaki sırayı takip etmek yerine birbiri ile ilgili kuralları aynı başlık altında toplayarak vermeyi tercih edeceğiz.

Kod parçalarımızı da vererek açıklama yoluna gideceğimiz için Prolog'a ilişkin şu açıklamayı yapmayı da gerekli görüyoruz. Kurallarımız da ==> operatörüne kadar olan kısım kuralın kendisini daha doğrusu yeni oluşturulan öbeğin kendisini ifade etmektedir. Buna Prolog terminolojisinde İngilizce olarak "rule head" yani kuralın ilk kısmı denilmektedir. Sonra gelen iki eleman liste biçiminde verilmiş ve virgül ile ayrılmıştır. Bu elemanlar da öbek oluşturmak için gerekli elemanları göstermektedir. Yine Prolog terminolojisinde kuralın bu kısmına (ker iki elemanı da içine alarak) İngilizce "rule body" yani kuralın gövde kısmı denilmektedir. Ayrıca daha önce belirttiğimiz Özellik Değer Matrisi özelliklerini birbirinden ayırmak için & operatörü ve özelliklerin değerleri verilirken de :: oparatörü kullanılmıştır. Normalde Prolog'da ==>, & ve :: operatörleri içsel olarak tanımlı değillerdir. Ancak bizim programımızda en başta iki parametreli ve araya gelecek biçimde operatör olarak tanımlanmış, programımız boyunca da kullanılmışlardır. Tanımlamaları aşağıdaki gibidir.

```
:- op(500, xfy, ==>).
:- op(400, xfy, ::).
:- op(450, xfy, &).
```

6.5.1 Yön, konum ve yer belirten kelimeler ile daha karmaşık ifadelerin oluşturulması

Daha önce sözlüğümüzdeki kelime türlerini anlatırken de dediğimiz gibi kelimelerin bir kısmı sözlüğe girilerek, basit ifadelerin doğrudan sözlükten alınması sağlanmıştır. Ancak daha karmaşık ifadeler için sözcük öbeği oluşturma kuralları kullanılmıştır. Bunlara en güzel örnek yön, konum ve yer belirten kelimelerin hem sözlüğe girilmiş hem de öbek kurallarının kodlanmış olmasıdır. Burada öncelikle mantıksal gösterim ile faydalandığımız kurallar sonra da herbirine örnekler verilecektir.

Gösterimlerimizde büyük harfler ile özyinelemeli biçimde tekrar açılan ifadeler, küçük harfler ile de sözlükte girişi yapılmış olan, belirtilen sınıfa ait kelimeler kastedilmektedir. Yön ifadelerimiz için durum şu şekildedir;

YÖN → yön
YÖN → yön, YÖN

Burada ilk kural kelimenin sözlükten doğrudan alınarak kullanılabileceğini belirtmektedir. Yani "sağ", "üst" vb. biçiminde. İkinci kural ise sözlükteki kelimelerin yanyana gelerek "sağ üst", "sol alt" gibi daha karmaşık ifadeler oluşturabileceğini belirtmektedir. Burada şu durumu belitmenin faydası vardır. İkinci kural sebebi ile "sağ sağ" ya da "üst üst" gibi yapıların da ayrıştırıcı tarafından kabul edilmesi mevcut kural sebebiyle mümkündür. Bu durumun çözümü için yönlere indeks değeri gibi değerler atanıp daha doğrusu özellik olarak özellik değer matrislerine girilip bu sorun halledilebilir. Örneğin birbiriyle aynı yönler için indeks aynı olacağından bir kural ile kelimelerin indeks değerleri aynı değilse geçerli olsun vb. biçiminde bir yapı kurmak mümkündür.

Yukarıda belirttiğimiz ikinci kural programımızda aşağıdaki gibi belirtilmiştir.

```
(syn::(cat::noun & case::nom & subcat::[]) &
 sem::(par::X & pred::(PRED1 & PRED2) & spec::yon)) ==>

      [(syn::(cat::noun & case::nom & subcat::[]) &
        sem::(par::X & pred::(PRED1) & spec::yon)),

       (syn::(cat::noun & case::nom & subcat::[]) &
        sem::(par::X & pred::(PRED2) & spec::yon))].
```

Her ikisi de isim biçiminde girilmiş ancak yön belirten (spec özellikleri yön olan) yalın haldeki kelimeler yanyana gelerek yine bir yön ifadesi oluşturabilmektedirler.

Bundan sonra programımızda yer ifadeleri oluşturmak için iki tane öbek oluşturma kuralımız vardır. Bunlardan ilki

YER → YON, konum

şeklindedir. Burada yine yukarıda anlatıldığı gibi basit ya da daha karmaşık yön ifadeleri ile konum belirten kelimelerin biraraya gelmesi ile bir yer ifadesi elde edilmesi kastedilmektedir. Örneğin "sağ üst köşesine" böyle bir yapıdır.

Ayrıca açıklamamız gereken bir durum da öbek oluşturma kuralları ile oluşturulan yer ifadelerimizin tamamının referans bir nesne ile kullanılmalarının zorunluluğudur. Bu durum öbek oluşturma kuralında yer ifadesinin subcat listesinin referans nesne özellikleri ile doldurulması yoluyla sağlanmıştır. Bunun açıklamasını ikinci kuralımızı da verdikten sonra yapmayı uygun buluyoruz. İlk yer ifadesi oluşturma kuralımız

```
(syn::(cat::noun & case::CASE &
 subcat::[syn::(cat::noun & case::gen & subcat::[]) &
 sem::(_ & spec::object)]) & sem::(par::X & pred::(PRED_KONUM & PRED_YON)
       & spec::yer)) ==>

            [(syn::SYN_YON &
              sem::(par::X & pred::PRED_YON & spec::SPEC_YON)),

             (syn::(cat::noun & case::CASE &
              subcat::[syn::SYN_YON & sem::(_ & spec::SPEC_YON)]) &
              sem::(par::X & pred::PRED_KONUM & spec::konum))].
```

biçimindedir. Anlattığımız gibi ilk elemanının yön gösteren bir ifade olması gerekmektedir. Bu durum ise sözlükte bütün konum bildiren kelimelerin subcat listelerine yalın halde, yön bildiren (spec değerleri yön olan), isim türünde kelimelere ihtiyaç duydukları belirtilerek sağlanmıştır. Sözlüğümüzdeki örnek bir konum sınıfı kelime girişimiz

```
lexeme(   phon::kosesine &
       syn::(cat::noun & case::dat &
       subcat::[syn::(cat::noun & case::nom & subcat::[]) & sem::(_ &
                spec::yon)]) &
       sem::(par::[X] & pred::(corner(X)) & spec::konum)).
```

şeklinde olup subcat listesi bizim belirttiğimiz gibidir. Kuralımızda da yön belirten ifade için sözdizimsel özellikler zaten konum bildiren kelimenin subcat listesinden alınmaktadır. (SYN_YON ile gösterilen değişken konum bildiren kelimenin subcat listesinde belirtilen yön ifadesi için gerekli sözdizimsel özellikleri göstermektedir.)

Burada yine önemli bir nokta kural sonucu elde edilen yer ifadesinin subcat listesinin bir referans nesneye ihtiyaç duyacak biçimde doldurulmuş olmasıdır. İkinci kuralımızı da açıkladıktan sonra bu konuya geri döneceğiz.

İkinci yer ifadesi oluşturma kuralımız ise

YER → Referans Nesne, YER

biçimindedir. Buradan da anlaşılabileceği gibi aslında ilk ifade ile elde edilen yer bilgisi istediği/ihtiyaç duyduğu referans nesne ile bu kural sayesinde birleştirilerek geçerli bir sözcük öbeği oluşturmaktadır. Programımızda kodlanmış olan kuralımız aşağıdaki gibidir.

```
(syn::(cat::noun & case::CASE & subcat::[]) &
 sem::(par::X & pred::(PRED_REF & PRED_YER) & spec::yer)) ==>

      [(syn::(SYN_REF_OBJ) &
        sem::(par::X & pred::(PRED_REF) & spec::SPEC_REF_OBJ)),

       (syn::(cat::noun & case::CASE &
        subcat::[syn::(SYN_REF_OBJ) & sem::(_ & spec::SPEC_REF_OBJ)]) &
        sem::(par::X & pred::(PRED_YER) & spec::yer))].
```

Bu kuralımızda da referans nesnenin sözdizimsel özellikleri ilk kuralımız ile oluşturulmuş olan YER ifadesinden alınmaktadır. Yani burada iki kuralın birbiriyle tamamlayıcı nitelikte çalışması sözkonusudur. İlk kural yön ifadeleri ile konum bildiren ifadeleri birleştirerek yer bildiren ifadeler elde etmektedir. Ancak bu ifadeler “sol alt

köşesine" gibi bir yapı göstermektedir. Bu durumda ilk akla gelen soru "Kimin sol alt köşesine?" biçiminde olmaktadır. Buradan da anlaşılacağı gibi bizim referans nesne dediğimiz isim ya da isim öbeği –nin eki almış ve nesne gösteren ifadeler olup bu sorunu ortadan kaldırmaktadırlar. Örneğin "kırmızının", "yeşil olanın", "mavi kutucuğun" gibi yapılar bu işe yaramaktadır. Bu şekilde referans nesne kullanımının zorunlu kılınması ile anlatımda ortaya çıkabilecek olan belirsizliklerin büyük bir kısmı ortadan kaldırılmış olacaktır. Kuralımıza dikkat edilirse referans nesnenin (syn özelliği ile kodlanan) sözdizimsel özellikleri SYN_REF_OBJ değişkeni ile yer ifadesinin subcat listesinden alınmaktadır.

Sözlüğümüzde doğrudan yer ifadesi olarak girilmiş kelimelerimiz de referans nesne isteyenler ve istemeyenler olarak iki gruba ayrılmışlardır. İlkinin subcat listeleri uygun biçimde doldurulmuş iken ikinci grubun subcat listeleri boş bırakılmıştır. Sözlükteki durum Yer belirten kelimeler başlığı altında açıklanmıştır.

Belirtilmesi gereken bir nokta da şudur; "Yeşil mavinin köşesine gitsin" gibi bir ifade geçerlidir. Çünkü dediğimiz gibi sözlüğümüzde "köşesine" kelimesi direk yer belirten kelime olarak da girilmiştir. İkinci kuralımız gereği "mavinin köşesine" biçimine getirilmesi doğaldır. Ancak sözlükteki bu giriş sebebi ile "hangi köşesi olduğu belli olmayan" belirsiz bir anlam ortaya çıkmaktadır. Programımızda bu sorun bilerek giderilmemiştir. Gerekli geliştirme işlemleri yapılıp programımız kullanıcı ile cümle girişlerinin ötesinde etkileşimli bir hale getirilebilirse böylesi bir durumda "Hangi köşesine?" diye sorarak bilgiyi kullanıcıdan alma yoluna gidebilir diye düşünülmüştür.

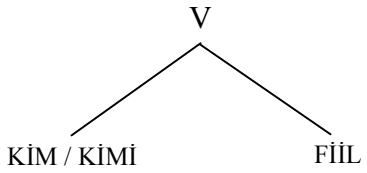
Kısaca özetlemek gerekirse ilk yer ifadesi kuralımız yön ve konum bilgisini birleştirip subcat listesini de referans nesne isteyecek biçimde doldurmaktadır. Bu şekilde nesne istemeyip direk sözlükten alınan yer belirten kelimeler için de durum yukarıda ve Yer belirten ifadeler başlığı altında belirtildiği gibidir. İkinci kuralımız da ister sözlükten gelsin isterse ilk kural ile oluşturulmuş olsun yer ifadelerini gereksinim duydukları referans nesne ifadesi ile birleştirmektedir.

6.5.2 Fiiller ile ilgili kurallar

Fiillerimiz ile ilgili olarak altı tane kuralımız bulunmaktadır. Bunlardan iki tanesi özellikle üç parametreli fiiler ile ilgilenebilmek için eklenmiştir. Sırası geldikçe ağaç yapıları ile birlikte durum açıklanacaktır.

Genel olarak fiillere ilişkin öbek oluşturma kurallarını, İngilizce verb phrase (VP) olarak geçen fiil öbeği oluşturma, yine özne içermeyen İngilizce V-bar oluşturma ve son olarak da bu elemanlar ile bir nesne ifadesinin birleşmesi ile oluşan cümle yapısı olarak sınıflandırmak mümkündür. Biz son durumda cümleye karşılık gelen kısma da cat(egory) değeri olarak "verb" dedik. Daha önce de belirttiğimiz gibi programımız açısından en önemli sözcük grubu fiildir ve yapılacak işi belirlemektedir.

En basit fiillerimiz tek parametreli olan "yoket", "gizlensin" gibi fiillerdir. Fiilleri düşününce de akla gelen ilk sorular "Kimi?" ya da "Kim" biçimindedir. Bu tür fiiler için sözcük öbeği oluşturma kurallarımızın en başında bir tane kural tanımlanmıştır. Burada Prolog programlamada sıralamanın program üzerindeki etkisinden faydalanılmıştır. Çünkü aşağıda şekilsel olarak verilen ifade fiil öbeği ile aynı yapıya sahiptir. Dolayısıyla cümle olarak girilen bir ifadenin fiil öbeği olarak değerlendirilmemesi için bu kuraldan önce tanımlanması gerekmektedir.



Şekil 6.1 Tek parametreli fiiller için programımızın kullandığı sözdizim ağacı

Şeklimiz ile daha aşağıda fiil öbeği (VP) için verilen Şekil 6.2'de gösterilen ağaç karşılaştırıldığında yapılarının aynı olduğu görülecektir. Bu problem girilen bir kural ve tek parametreli fiiller için girilen bu kuralın öne alınması ile giderilmiştir.

Kodumuzda bulunan kural ise aşağıdaki gibidir.

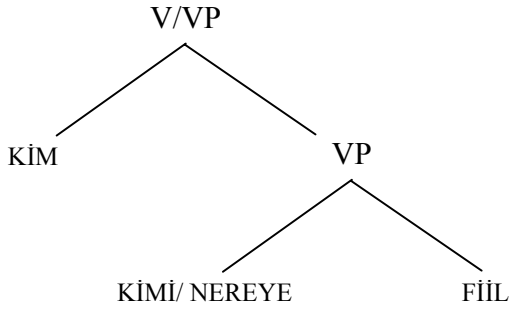
```
(syn::(cat::verb & case::nom & subcat::[]) &
 sem::(par::X & pred::(PRED_VERB & PRED_NOUN) & spec::action)) ==>

   [(syn::(cat::noun & case::CASE & subcat::[]) &
     sem::(par::X & pred::PRED_NOUN & spec::object)),


    (syn::(cat::verb & case::nom &
     subcat::[(syn::(cat::noun & case::CASE & subcat::[]) &
     sem::(_ & spec::object))]) & sem::(par::X & pred::PRED_VERB &
            spec::action))].
```

Kuralımızdan da anlaşıldığı gibi fiil istediği bileşen için hali / takıyı belirlemekte ve CASE değişkeni ile durumun gereken bileşende olması sağlanmaktadır. Ayrıca kuralın KİMİ ve KİM türünde parametre isteyen fiiler için ortak olarak kullanılması da bu şekilde sağlanmıştır.

Tezimizin bundan sonrasında kullanacağımız şekillerimizde "bölü gösterimi"ni kullanmayı tercih edeceğiz. Adı geçen notasyon (Gazdar ve Mellish 1989)da şu şekilde ortaya çıkmaktadır;

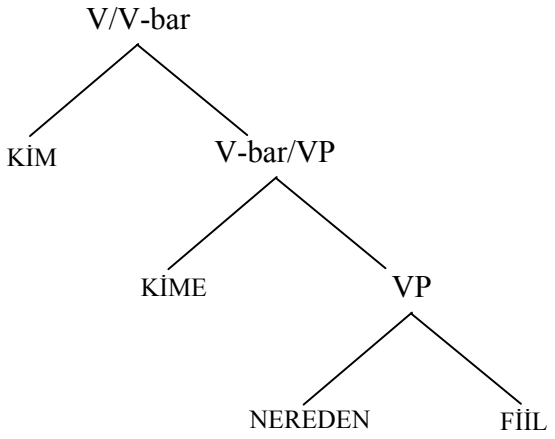
"… kategorisi X olan ve bölüsü Y olan X/Y biçiminde yazılan gösterimsel kabulu bölü diye adlandıracağız … X/Y biçimindeki bir kategori, Y kategorisindeki parçası bulunmayan X kategorisini temsil etmektedir. Böylece S/NP (S bölü NP biçiminde okunur) henüz bulunmayan NP'si olan / NP'ye ihtiyaç duyan bir cümledir (veya cümleciktir.)"

İki ve üç parametreli fiiller için özellikle doğru biçimde anlam analizi yapılabilmesi için iki ayrı fiil öbeği (VP) kuralı konmuştur. Durumun şekil ile ifadesi aşağıdaki gibidir.



Şekil 6.2 İki parametreli fiiller için programımızın kullandığı sözdizim ağacı

Şekilden de anlaşıldığı üzere sözdizim ağacımız fiilin istemiş olduğu bileşenlere göre şekillenmiştir. Bu bileşenler fiillerin subcat listelerinde bulunacakları hal / takı parametreleri de dahil olmak üzere belirtilmiştir. Ancak burada önemli olan nokta VP olarak gösterdiğimiz fiil öbeğinin hal bilgisini sol alt ağacından aldığıdır. Yani NEREYE biçiminde bir eleman varsa (örneğin "sol üst köşesine") fiil öbeği için belirlenen hal bilgisi –e hali, KİMİ biçiminde bir eleman varsa (örneğin "yeşil ile maviyi yerdeğiştir" derken "maviyi") fiil öbeği için belirlenen hal bilgisi –i hali olacaktır. Durum ilerleyen kısımlarda üç parametreli fiiller için hazırlanan Şekil 6.4'deki ağaç ile daha iyi anlaşılacaktır.



Şekil 6.3 Üç parametreli fiiller için programımızın kullandığı sözdizim ağacı

İki parametreli fiillerde olduğu gibi üç parametreli olanlarda da fiil öbeği (VP) hal bilgisini sol alt ağacından almaktadır. Ancak dikkat edilirse sol alt ağacı her zaman –den

halinde olmaktadır. Bunun üzerinde durmamızın sebebi doğru anlam analizi yapılabilmesi için iki ayrı VP kuralının girilmiş olmasının gereğidir. Aslında sadece sözdizimsel analiz yapılmak istense idi tek bir kuralla da bunu başarmak mümkündü. Çünkü dediğimiz gibi fiil öbekleri hal bilgilerini sol alt ağaçtan almaktadır. Mevcut durum için girilmiş olan fiil öbeği kurallarımız şunlardır.

```
(syn::(cat::vp & case::abl & subcat::Rest) &
 sem::(par::[First,Second] & pred::(PRED_VERB & PRED_YER) & spec::_)) ==>

   [(syn::(cat::CAT & case::abl & subcat::SUB) &
     sem::(par::RestPar & pred::(PRED_YER) & spec::SPEC)),

    (syn::(cat::verb & case::nom &
     subcat::[syn::(cat::CAT & case::abl & subcat::SUB) & sem::(_ &
                                              spec::SPEC)|Rest]) &
     sem::(par::[First,Second|RestPar] &
           pred::(PRED_VERB) & spec::action))].



(syn::(cat::vp & case::CASE & subcat::Rest) &
 sem::(par::[First] & pred::(PRED_VERB & PRED_YER) & spec::_)) ==>

      [(syn::(cat::CAT & case::CASE & subcat::SUB) &
        sem::(par::Second & pred::(PRED_YER) & spec::SPEC)),

       (syn::(cat::verb & case::nom &
        subcat::[(syn::(cat::CAT & case::CASE & subcat::SUB) &
        sem::(_ & spec::SPEC))|Rest]) & sem::(par::[First,Second] &
                                  pred::(PRED_VERB) & spec::action))].
```

Kurallara bakıldığı zaman ikinci kuralın birinci kuralı kapsadığı daha doğrusu birinci kuralın ikincinin özel bir hali olduğu görülür. Sadece sözdizimsel analiz için hakikaten sadece ikinci kuralın olması yeterlidir. Çünkü kuralımız hal bilgisini fiilin subcat listesinin ilk elemanından almakta, fiilin istediği türde bir eleman ile bir araya gelerek fiil öbeği oluşturmasını sağlamakta ve elde etmiş olduğu hal bilgisinin hem fiilin istemiş olduğu elemanın hem de elde edilen fiil öbeğinin hal bilgisi olmasını sağlamaktadır. CASE ile gösterilen değişken bu dediğimiz işlemi gerçekleştirmektedir.

Neden böyle iki kurala gereksinim olduğu sorusunun cevabı ise fiilerin parametre sayılarında yatmaktadır. Şekil 6.2 ve Şekil 6.3'de görüldüğü gibi fiil öbeğinin (VP) sol alt ağacı iki parametreli fiillerde ikinci parametreye, üç parametreli fiilerde ise üçüncü

parametreye denk gelmektedir. Bu sebeple parametre uyuşumunun sağlanabilmesi için böyle bir özel halin ayrıca girilmesine gerek duyulmuştur. Konunun anlam analizi açısından daha geniş biçimde ele alındığı Programımızın Anlamsal Yapısı bölümünde ayrıntılı açıklamayı bulabilirsiniz.

Burada belirtilmesi gereken bir diğer husus da yine Prolog programlamada sıralamanın program üzerindeki etkisinden yararlanılmış olmasıdır. Doğru biçimde anlam analizi yapılabilmesi için diğer programlama dillerinde hata işleme durumunda olduğu gibi özel halin üstte olması gerekmektedir.

Fiil öbekleri oluşturulduktan sonra üç parametreli fiillerde V-bar elemanının oluştrulması işlemi gerçekleştirilmektedir. Bu durum da Şekil 6.3'den görülmektedir. Programımızda bu konu için girilmiş olan kural aşağıdaki gibidir.

```
(syn::(cat::vbar & case::dat & subcat::Rest) &
 sem::(par::[FirstP] & pred::(PRED_VERB & PRED_KIME & PRED_YER) &
       spec::_)) ==>

      [(syn::(First) &
        sem::(par::SecondP & pred::(PRED_KIME) & spec::SpecFirst)),

       (syn::(cat::vp & case::abl &
        subcat::[syn::(First) & sem::(_ & spec::SpecFirst)|Rest]) &
        sem::(par::[FirstP,SecondP] & pred::(PRED_VERB & PRED_YER) &
              spec::_))].
```

Bu noktada dikkat edilmesi gerekn husus hem fiil öbeğinin hem de V-bar biçiminde tanımlanan öbeğin sol alt ağaçlarının ne olacağının fiilin subcat listesi tarafından belirlenmiş olmasıdır. Fiiller başlığı altında da açıkladığımız gibi programımız açısından en önemli kelime grubu fiillerdir. Çünkü cümle içinde olması gereken bileşenleri belirleyen ana grup fiillerdir. Bu durum da subcat listelerinin uygun biçimde düzenlenmesi ile sağlanmıştır. Ayrıca en başta da söylediğimiz gibi bizim bütün girişlerimiz "küçük" ya da "kapalı" dünya modeline göre yani programımızın yapacağı işe göre şekillenmiştir. Örnek Bir Program Olarak SHRDLU bölümünde de belirtildiği gibi SHRDLU da biz insanlar gibi açık olmayan bir dünya modeline sahip olması nedeniyle eleştirilmektedir. Ancak bizim

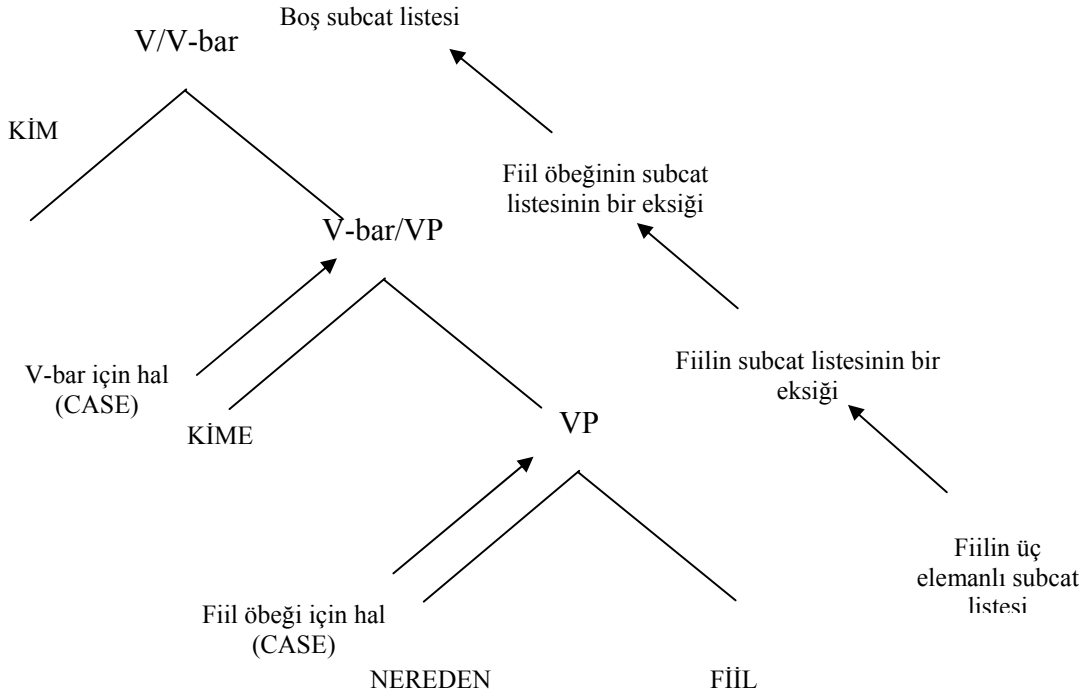
görüşümüze göre bir makina ya da programın yapacağı işe göre şekillenmiş kendine has, kapalı ya da küçük bir dünyası olmasının bir sakıncası yoktur, gereği de yoktur. Yapacağı işi tam olarak yerine getirmesini sağlayacak biçimde şekillenmiş bir dünya modeli program ya da makinalar için yeterlidir diye düşünmekteyiz. Zaten bu konudaki görüşlerimiz Programımızın Genel Yapısı başlıklı bölümde aynı biçimde dile getirilmişti.

Son olarak da fiil öbeği veya V-bar ile nesne gösteren bir ifadenin biraraya gelmesi ile cümlenin değerlendirmesi işlemi tamamlanmaktadır. Bunun için de iki tane kuralımız bulunmaktadır. Bir tanesi fiil öbeği ile diğeri de V-bar ile gerekli ifadeyi birleştirmektedir. Kurallarımız aşağıdaki gibidir.

```
(syn::(cat::verb & case::nom & subcat::[]) & sem::(par::[First|Rest] &
 pred::(PRED_VERB & PRED_KIM & PRED_KIME & PRED_YER) & spec::action)) ==>

[(syn::(SYN_OBJECT) &
  sem::(par::First & pred::(PRED_KIM) & spec::SPEC_OBJECT)),

 (syn::(cat::vbar & case::dat &
  subcat::[syn::(SYN_OBJECT) & sem::(_ & spec::SPEC_OBJECT)])) &
  sem::(par::[First|Rest] & pred::(PRED_VERB & PRED_KIME & PRED_YER) &
        spec::_)].


(syn::(cat::verb & case::nom & subcat::[]) &
 sem::(par::[First|Rest] &    pred::(PRED_VERB & PRED_REF & PRED_YER) &
       spec::action)) ==>

      [(syn::(SYN_OBJECT) &
        sem::(par::First & pred::PRED_REF & spec::SPEC)),

       (syn::(cat::vp & case::_ &
        subcat::[syn::(SYN_OBJECT) & sem::(_ & spec::SPEC)])) &
        sem::(par::[First|Rest] & pred::(PRED_VERB & PRED_YER) &
              spec::_)].
```

Yine dikkat edilirse son bileşene ilişkin sözdizimsel yapı fiil öbeği ya da V-bar'ın subcat listelerinden alınmaktadır. Burada şöyle bir açıklama yapmayı uygun buluyoruz. Cümlenin en önemli öğesi olarak nitelediğimiz fiilin subcat listesi sanki aşağıdan yukarıya doğru eksilerek gelmekte ve son bileşen de işlem gördükten sonra boş liste haline geelmektedir. Fiili öbeği oluşturabilmek için subcat listesinin ilk elemanına uygun cümle bileşeni aranmakta eğer bulunursa fiilin subcat listesinin kalan elemanları fiil öbeğinin

subcat listesi olarak belirlenmektedir. Fiil öbeği oluşturma kurallarına bakıldığında bu durumun SUB ve Rest değişkenleri aracılığı ile sağlandığı görülmektedir. Aynı şekilde fiil öbeği elemanları da miras almış oldukları bu subcat listesine uygun elemanlar ile birleşerek ya cümleyi ya da V-bar elemanını oluşturmakta ve subcat listelerinin kalan elemanlarını bir üst seviyeye vermektedirler. Benzer biçimde daha önce de söylediğimiz gibi fiil öbeği ve V-bar elemanları sol alt ağaçlarının hali / takısı ne ise o hali / takı CASE değişkeni aracılığı ile almaktadırlar. Buraya kadar anlatılanların şekilsel olarak ifadesi ise aşağıdaki gibidir.



Şekil 6.4 Üç parametreli fiiller için özelliklerin alındığı yerler

İki parametreli fiiller için de durum benzerdir. Tek fark fiil öbeğinden sonra arada V-bar katmanının bulunmayışı ve fiilin subcat listesinin iki elemanlı oluşudur.

6.5.3 İsim öbeği oluşturma kuralı

Programımızda şekilleri ifade etmek için doğrudan "kırmızıyı" şeklinde bir yapı kullanılabileceği gibi "kırmızı olanı", "mavi kutuyu", "yeşil kutucuğun" gibi bazı kelimeler de sıfatlar ile birlikte kullanılabilirler. Daha önce de söylediğimiz gibi programımızdaki tek ayırdedici özellik renkler olduğu için bu şekildeki ifadelerin de nesne gösteren ifade olarak değerlendirilmesi gerekmektedir. Bu sebeple bir sıfat ile birleştiğinde nesne belirten ifade oluşturan yapılar için bir öbek oluşturma kuralımız mevcuttur. Sözlüğümüzde "olan", "kutu" ve "kutucuk" şeklindeki kelimeler yalın, -i ve –nin eki almış halde bulunmaktadırlar ve nesne gösteren birer isim olarak değerlendirilmektedirler. Yani cat(egory) elemanları isim, spec(ifies) elemanları nesne biçiminde girilmiştir. Ayrıca bir sıfat ile birlikte kullanılmaları gerektiği subcat listelerinde belirtilmiştir. Kuralımızda öbeğin takısı bu kelimelerin takısı olarak belirlenmekte ve öbek oluşturmak için gerekli kelimenin sözdizimsel yapısı yine bu kelimelerin subcat listesinden alınmaktadır.

```
(syn::(cat::noun & case::NOUN_CASE & subcat::[]) &
 sem::(par::X & pred::(PRED_OBJ & PRED2) & spec::object)) ==>

      [(syn::SYN_SUB &
        sem::(par::X & pred::(PRED_OBJ) & spec::SPEC_SUB)),

       (syn::(cat::noun & case::NOUN_CASE &
        subcat::[syn::SYN_SUB & sem::(_ & spec::SPEC_SUB)]) &
        sem::(par::X & pred::(PRED2) & spec::object))].
```

6.5.4 Bağlaçlara ilişkin öbek oluşturma kuralı

Programımızda sınırlı kullanımı olan iki tane bağlaç mevcuttur. Bunlar sırasıyla "ile" ve "ve"dir. Kullanımlarının kısıtlı olmasından kastımız "Mavi ile yeşili birleştir" ya da "Mavi ve yeşil yerdeğiştirsin" gibi birkaç fiilin subcat listelerinde bağlaçlı kullanıma izin verecek biçimde tanımlanmış olmasıdır. Programımızda çoğul isimler ile ilgilenilmemektedir. O sebeple "ile" ya da "ve" gibi bağlaçlarla elde edilecek çoğul yapılar

gibi durumlara yönelik öbek oluşturma kurallarımız mevcut değildir. Programımızla ilgili olarak yapılabilecek ilerletme çalışmaları ve eksikler Gelecekte Yapılabilecek İlerletmeler ve Öneriler başlığı altında bulunabilir.

Bağlaç olarak sözlüğe girişi yapılmış olan kelimelerimizden "ile" birlikteliği, "ve" ise her ikisi birden belirlemelerini yapmaktadır. Bu sebeple spec(ifies) elemanları buna uygun olarak girilmiştir. O sebeple eğer bir fiil bu iki kelimeyi de içeren yapı kullanımına izin vermek istiyorsa ikisi için de ayrı ayrı girişlere sahip olmalı ya da değeri önemsiz değişken kullanmalıdır. Biz örneğin "birleştir" fiili için (Çetinoğlu 2001)deki gibi subcat listeleri iki kelimenin de kullanımı için farklılaşmış iki giriş yapma yoluna gittik.

Bağlaçlara ilişkin kural spec(ifies) özelliğini hangi bağlaca rastlamışsa ondan alacak biçimdedir. Bu sayede tek kural bizim için yeterli olmuştur ve aşağıdaki gibidir.

```
(syn::(cat::noun & case::nom & subcat::[]) &
 sem::(par::X & pred::PRED & spec::_)) ==>

      [(syn::(SYN_OBJ) & sem::(par::X & pred::PRED & spec::SPEC)),

       (syn::(cat::conj & case::nom &
        subcat::[(syn::(SYN_OBJ) & sem::(_ & spec::SPEC))]) &
        sem::_)].
```

6.6 Ayrıştırıcı

Programımızda kullandığımız ve ayrıştırıcı olarak görev yapan kod parçacığı yapısı itibarıyla basit olarak nitlenedirilebilir ve üç tane kuraldan oluşmaktadır. Bunlardan ilki Prolog programlamanın özyinelemeli yapısı sebebiyle (Weiss 1999)da özyinelemeli fonksiyonların yazımı konusunda durma şartı olarak belirtilen kuraldır ve ilerleme işlemini sonlandırır. Eğer sonuçta geçerli bir cümle elde edilebilmiş ise bu kural ile işleme son verilir. Ayrıştırıcımız cümlenin bileşenlerinden başlayarak yukarı doğru temel bileşenlerden öbekleri ve öbeklerden de cümleyi oluşturacak/geçerliliğini denetleyecek bir yapıda olduğu için tabandan yukarı ya da İngilizcesi ile bottom-up olarak çalışmaktadır. İkinci kuralımız son giren ilk çıkar (İngilizce Last In First Out ya da LIFO) felsefesi ile çalışan yığında en az iki eleman varsa bunları birleştirerek bir öbek oluşturup

oluşturamayacağına bakar. Bunu yaparken de önceki bölümde anlattığımız sözcük öbeği oluşturma kurallarından faydalanır. Eğer bir öbek oluşturabilirse bu yeni oluşturduğu yapıyı yığına ekler ve işleme devam eder. Burada önemli bir nokta ayrıştırıcımızın yığından her defasında iki elemanı alarak işlem yapmasıdır. Bu durum da bizim öbek oluşturma kurallarımızın ikililer biçiminde tanımlanmasına sebep olmuştur. Tek parametreli fiiller için ayrı bir kural girilmesinin en temel nedeni de bu kısıtlamadır. Daha gelişmiş ayrıştırıcı modelleri mevcuttur. Konuyla ilgili olarak (Gazdar ve Mellish 1989) ve (Allen 1995)'e bakılabilir. Son ayrıştırıcı kuralımız ise geçerli bir cümle elde edilemedi ise ve yığında ikiden az eleman varsa veya varolan elemanlardan bir öbek oluşturulamadıysa çalışmaktadır. Kullanıcı tarafından girilen kelimenin sözlükte olup olmadığına bakar. Sözcüğü bulduktan sonra yazımı (programımızda phon özelliği olarak belirtilmiştir) hariç diğer bütün özelliklerini yığına atar. İkinci ve üçüncü kuralarımızın satırları özyinelemeli biçimde ayrıştırıcının yoluna devam etmesini sağlamaktadır. Kod parçası aşağıdaki gibidir.

```
parser([syn::(cat::verb & case::nom & subcat::[]) & sem::(par::_ &
                        pred::SEM & spec::action)],[],SEM).


parser([First,Second|Rest], String, Result) :-
       Mother ==> [Second,First],
       parser([Mother|Rest], String, Result),!.

parser(Stack, [First|Rest], Result) :-
       lexeme(phon::First & syn::SYN_WORD & sem::SEM_WORD),
       parser([syn::SYN_WORD & sem::SEM_WORD | Stack], Rest, Result).
```

6.7 Yardımcı Program Parçaları

Programımıza kullanıcı tarafından cümlelerin kolay olarak girilebilmesi için (Bratko 2001)de mevcut olan bir kod aynı şekli ile aktarılmıştır. getsentence isimli kural / prosedür sayesinde kullanıcının her kelime arasında boşluk bırakmak, noktalama işaretleri kullanmamak ve son kelime ile arasında bir boşluk bırakarak nokta ile bitirmek koşuluyla düz cümle girmesi sağlanmaktadır. Kodun mantığı şu şekilde ifade edilebilir; Cümleler kelimelerden kelimeler de harflerden oluşmaktadır. Bu şekilde yapılacak işi parçalara

bölerek kullanıcının girdiği cümleyi almaktadır. Burada en önemli olan Prolog'un yerleşik elemanı olan name'in kullanılmasıdır. Kodun ve yerleşik elemanın açıklaması ismi geçen kaynakta mevcuttur. Ancak ana ilke bizim de belirttiğimiz gibi harflerin alınıp kelimenin, kelimelerinde alınıp cümlenin oluşturulması biçimindedir. Kod parçacığı aşağıdaki gibidir.

```
getsentence(Wordlist) :-
                      get0(Char),
                      getrest(Char, Wordlist).

getrest(46, []) :- !.             % End of sentence: 46 = ASCII for '.'

getrest(32, Wordlist) :- !,          % 32 = ASCII for Blank
            getsentence(Wordlist).  % Skip the blank



getrest(Letter, [Word|Wordlist]) :-
                getletters(Letter, Letters, Nextchar),
                name(Word, Letters),
                getrest(Nextchar, Wordlist).



getletters(46, [], 46) :- !.      % End of word. 46 = .

getletters(32, [], 32) :- !.      % End of word. 32 = blank

getletters(Let, [Let|Letters], Nextchar) :-
                get0(Char),
                getletters(Char, Letters, Nextchar).
```

Bir de programımızın kolay kullanımı için run adında bir prosedürümüz bulunmaktadır. Bu prosedür yukarıda açıkladığımız getsentence prosedürü ile cümleyi normal formda kullanıcıdan almakta, girilen cümleyi ayrıştırıcıya göndererk sonucu almakta ve ekrana görüntülemektedir. Amacı program etkileşimini kolaylaştırmak olan bu prosedürün kodu aşağıdaki gibidir.

```
run :-
            getsentence(WordList),
            parser([],WordList,Res),
            write(Res).
```

Komut satırından örnek bir kullanım ise aşağıdaki gibidir.

```
?- run.
|: maviyi yesilin soluna gotur .

SEM = ...

Yes
```

Burada SEM ile elde edilen değer cümlenin mantıksal gösterimi olacaktır ve Prolog'da her zamanki gibi "yes" görünmesi ile cümlenin geçerli olduğunu anlıyoruz.

7. PROGRAMIMIZIN ANLAMBİLİMSEL KISMI

7.1 Genel

Programımız sözdizimsel ayrıştırma işleminden faydalanarak anlam analizi işlemini de yerine getirmektedir. Bu konuda Montague Anlambilimi bölümünde anlattığımız Frege'nin Birleştirilebilme İlkesi'nden faydalanılmaktadır. İlgili bölümde de anlatıldığı gibi Birleştirilebilme İlkesi temelde bir bütünün anlamının parçalarının anlamlarının bir fonksiyonu olması ilkesine dayanmaktadır. Belirtilen durumun gerçeklenmesi ise her sözdizim kuralına karşılık bir anlambilim kuralının bulunması ve cümlenin sözdizimsel bir analizden geçirilip dilin yapısına uygun olup olmadığı belirlenirken aynı zamanda bu parçaların anlamsal kısımlarının da belirlenmesi ile olmaktadır. Sonuçta sözdizim kuralları ile birlikte anlambilim kuralları da çalışmakta, aşağıdan yukarı bir ayrıştırıcı ile sözdizimsel olarak cümleye, anlambilimsel olarak da bütünün yani cümlenin anlamına ulaşılmaktadır.

Ayrıştırıcı ve sözcük öbeği oluşturma kuralları aracılığı ile bu şekilde hem sözdizimsel hem de anlamsal biçim elde edildikten sonra gerektiği takdirde cümlenin doğruluğunun değerlendirilebilmesi için programın sahip olduğu dünya modeline ve özellikle de içinde bulunduğu duruma bakılarak cümlenin doğruluğu değerlendirilebilir. Ancak bizim programımızda şu anda sadece emir formunda cümleler değerlendirildiği için çok fazla dünya modeline başvurmaya gerek duyulmamaktadır. Ancak sorgu cümleleri gibi yapılar ile program geliştirilmek istenirse dünyanın o an içinde bulunduğu duruma başvurmak kaçınılmaz bir hal olacaktır.

Ayrıca Montague Anlambilimi bölümünde örnek bir çalışma olarak gösterilen (Kılıçaslan ve Tüysüz 2002)de olduğu gibi ayrı ayrı, sözdizim kuralı şu şekildedir ve bu sözdizim kuralına karşılık gelen anlambilim kuralı da bu şekildedir gibi bir gösterim yerine programımızda anlamsal özellikler sözdizim kuralları ile birleştirilmiştir. İlerleyen konu başlıklarında bu kurallar teker teker incelenecek ve nasıl bir birleştirme ile cümleye kadar ilerlendiği açıklanacaktır.

Yaptığımız anlam analizinin girilen cümleyi doğal dilin anlam kargaşası ya da bulanıklığı gibi durumlardan kurtarmak için biçimsel bir ara forma dönüştürdüğü de unutulmamalıdır. Ancak bu ara forma dönüştürme işlemi yapıldıktan sonra program, elde edilen formu yorumlayarak kendisinden yapması istenen işi anlayabilecektir. Programımızın mevcut halinde ise belirtilen yapı mevcut değildir. Gelecekte yapılabilecek bir ilerletme çalışması olarak değerlendirilebilir.

Son bir nokta ise Montague Anlambilimi bölümünde Olası Dünyalar, Örnek Bir Program Olarak SHRDLU bölümünde ise "kapalı" ya da "küçük" dünya olarak nitelendirilen kavramın bizim progamımızdaki yeridir. Programımızdaki sözcüklerin anlamlarından sözcük öbeği oluşturma kurallarına kadar her şey programımızın yapacağı işe göre şekillenmiştir. O sebeple bazı yapıların biz insanların algıladığından ya da anladığından çok daha kısıtlı yapıya sahip olmasının sebebi budur.

Bundan sonra sırası ile kelimelerin sözlüğümüzdeki girişlerine anlamsal değerlerin nasıl katıldığı ve sözcük öbeği oluşturma kurallarının bu durumdan nasıl faydalandığı anlatılacak, en son olarak da mevcut durum değerlendirilecektir. Bu bölümde sözdizim bölümümüzden farklı olarak hem kelime sınıflarının sıralaması sözlükteki sıralamaya göre yapılmayacak hem de sözcük öbeği oluşturma kuralları ayrı bir başlık altında incelenmek yerine gerekli kategori açıklanırken o kategoriye ilişkin öbek oluşturma kuralının nasıl işlediği anlatılacaktır.

7.2 Kelimelerin Sözlükteki Anlamsal Girişleri

Daha önce Programımızın Sözdizimsel Yapısı başlıklı bölümde izlediğimiz yöntemi kullanarak sözlüğümüzdeki kelime sınıflarına göre anlamsal girişlerinin nasıl yapıldığını açıklayacağız.

Kelimelerimiz için anlamsal değerler sem(antics) özelliği altında girilmiştir ve başlıca üç maddeden oluşmaktadır. Bunlardan ilki par(ameters) biçiminde kelimeye denk gelen yüklemin ya da önermenin kaç parametresi olduğunu belirten kısımdır. Bu kısım bütün kelimeler için liste halindedir. Anlamsal değerlerin ikinci kısmı ise pred(icate) özelliği olarak kodlanmış olup kelimeye karşılık gelen önerme ya da yüklemi

göstermektedir. İlk özellik olan par(ameters) da mevcut olan değişkenler burada sırası ile kullanılmıştır. Tek ve çok parametreli sözlük girişleri için örnekler sırası ile aşağıda verilmiştir. (Bu noktada, kağıt üzerinde program parçalarımızın istediğimiz gibi görünmediğini o sebeple elden geldiğince anlaşılır parçalar halinde ve biçimlenerek verilmeye çalışıldığını, ancak en doğru biçiminin programımızın kaynak koduna bakılarak anlaşılabileceğini vurgulamak istiyoruz.)

```
lexeme(      phon::mavi      &
           syn::(cat::noun & case::nom & subcat::[]) &
           sem::(par::[X] & pred::(blue(X) & box(X)) & spec::object )).


lexeme(      phon::surukle &
      syn::(cat::verb & case::nom &
      subcat::[(syn::(cat::noun & case::dat & subcat::[]) &
                sem::(_ & spec::yer)),
               (syn::(cat::noun & case::acc & subcat::[]) &
                sem::(_ & spec::object))]) &
      sem::(par::[X,Y] & pred::drag(X,Y) & spec::action)).
```

Ayrıca sözdizimsel analizde de faydalandığımız ancak kullanım amacı olarak sem(antics) özelliğinin bir alt özelliği olarak düşündüğümüz spec(ifies) özelliği ise kelimenin bizim programımızın dünyasında nasıl bir nesneyi/varlığı işaret ettiğini göstermektedir. Sözdizim bölümümüzden de anlaşıldığı gibi bu özellikten sözcük öbeği oluşturma kurallarında faydalanılmıştır. Bu durumun programımızın kendine has dünyası ve ona uygun biçimde işlem yapması düşüncemize uygun olduğunu düşünüyoruz.

Buradaki X ve Y biçiminde verilmiş olan parametrelerin Montague Anlambilimi başlıklı bölümde anlatılan Lambda Soyutlamasına karşılık geldiği düşünülebilir. Çünkü Montague de bir önermede alınabilecek değerleri genişletmek eldeki yapıyı genel bir hale getirmek için bu yönteme başvurmuştur. Biz de kod olarak gerçekleme işleminde yukarıda gösterdiğimiz yaklaşımla Lambda Soyutlaması yapmış oluyoruz. Soyutlama hakkında genel bilgi Montague Anlambilimi başlıklı bölümde Lambda Soyutlaması adı altında anlatıldığı için tekrar ayrıntıya girmiyoruz. Ancak fiillerin subcat listeleri ile de bu değişkenlerin yerini alabilecek değerlerin özelliklerinin neler olduğu belirlenmiştir. Subcat listeleri bizim için Durum Teorisi başlıklı bölümde belirttiğimiz kısıtlayıcı şartları

oluşturmaktadır. Genel olarak ise anlattığımız durum itibarıyla – Lambda Soyutlaması ve subcat listelerinin özelliği açısından - örnek girişlerimizdeki değişkenlere kısıtlanmış değişkenler diyebiliriz.

Montague anlamsal ifadeleri İngilizce karşılıklarının üzerine bir kesme işareti atarak göstermektedir. Bunu yapmaktaki gerekçesi "ağaç" gibi Türkçe bir kelime İngilizce "tree" olarak ifade edilse bile mevcut dünyamızda işaret ettikleri nesne/varlık aynıdır. Dolayısıyla dilden dile gösterimleri/yazılışları değişse bile işaret ettikleri varlık değişmemektedir. O sebeple kelimenin İngilizce karşılığının üzerine bir çizgi çekerek kelimeyi İngilizce olmaktan çıkarıp genel olarak dünyada işaret ettiği nesneyi gösterir hale getirmiştir. Daha doğrusu böyle bir notasyonukullanmıştır. Biz de Montague'nin yaklaşımına sadık kalarak kelimelerin anlamsal değerlerini İngilizce olarak gösterdik. Bizim gösterimimizde üst çizgi bulunmamakla birlikte dayandığımız temel Montague'nin yaklaşımıdır. Bundan sonra kelimelerin anlamsal değerlerinde karşılaşılacak olan İngilizce gösterimlerin sebebi budur.

7.2.1 Bağlaçlar

Mevcut programımızda bağlaçlar için sözlükte özel olarak bir değer girişi yapılmamıştır. Çünkü sözdizimsel bölümde de anlattığımız gibi bağlaç kullanımına ilişkin öbek oluşturma kuralımızda bağlacın ilk parametresi olan isim bizim için önemli idi ve bağlaç sadece bir birleştirme işlemi yerine getirmekte idi. Anlamsal kısımda da aynı şekilde bağlaç ile birarada bulunan ismin sahip olduğu değer bizim için önemlidir ve öbek oluşturma kuralında da anlamsal değer olarak ismin değeri alınmıştır. Kural aşağıdaki gibidir.

```
(syn::(cat::noun & case::nom & subcat::[]) &
 sem::(par::X & pred::PRED & spec::_)) ==>

      [(syn::(SYN_OBJ) & sem::(par::X & pred::PRED & spec::SPEC)),

       (syn::(cat::conj & case::nom &
        subcat::[(syn::(SYN_OBJ) & sem::(_& spec::SPEC))]) &
        sem::_)].
```

Yukarıda da açıkladığımız gibi öbeğin yüklem olarak anlamsal değeri PRED ile birlikte kullanıldığı isimden alınmaktadır.

7.2.2 Sıfatlar

Normalde sözlüğümüzde sıfat kategorisi isim kategorisinden sonra gelmiş olmasına karşın "olan" vb. kelimelerin açıklamaları yapılırken sıfatların açıklamalarına ihtiyaç duyulacağından sıralamayı bu şekilde değiştirmeyi uygun bulduk.

Sözlüğümüzde sadece renk belirten sıfatlar bulunduğu için anlamsal gösterimleri de renk belirtecek biçimde tek parametrelidir. Daha sonra İsimler bölümünde referans olarak nitelenen kelimeler ile birleştikleri zaman daha anlamlı hale gelmektedirler. Sıfatlar için sözlüğümüzdeki örnek bir giriş aşağıdaki gibidir.

```
lexeme(      phon::mavi      &
           syn::(cat::adj & case::nom & subcat::[]) &
           sem::(par::[X] & pred::(blue(X)) & spec::_)).
```

Unutulmaması gereken bir diğer nokta da sıfat olarak girdiğimiz "mavi", "kırmızı" ve "yeşil" kelimelerinin aynı zamanda isim olarak da girişleri mevcuttur ve kodladıkları anlam burada gösterilenden farklıdır.

7.2.3 İsimler

İsimler için girişler yaptıkları göreve göre ayarlanmıştır. Sözdizimsel kısımda da anlatıldığı gibi kelimeler aldıkları takılara göre görevleri değişen birer nesne gibi düşünülebilir. Sözlüğümüzdeki anlamsal girişleri de bu duruma uygun olarak düzenlenmiştir. Ayrıca hep üzerinde durduğumuz gibi programımız için özelleşmiş değerler biçiminde girildikleri de akılda tutulmalıdır. Çünkü "mavi" , "kırmızı" ve "yeşil" kelimeleri sıfat girişlerinin yanında isim olarak girişlere de sahiptirler ve bir nesneyi belirtmek üzere kullanılmaktadırlar. O sebeple anlamsal değerleri de iki parçalı biçimde girilmiştir. Aşağıda mavi için verilen giriş her üç kelime için de aynıdır.

```
lexeme(      phon::mavi      &
           syn::(cat::noun & case::nom & subcat::[]) &
           sem::(par::[X] & pred::(blue(X) & box(X)) & spec::object )).
```

Bir nesneyi rengi ile belirttikleri için pred özellikleri "mavi" ve "nesne" biçiminde girilmiştir. Bir tek nesneye ait özellikler olduğu için her iki önermenin de parametresi

aynıdır. Yani bir nesneyi gösteriyor ve bu nesnenin hem rengini hem de nesne olduğunu belirtiyor diyebiliriz.

Daha sonra –nin eki alarak iyelik belirten isimlerimiz gelmektedir. İyelik belirtme durumu ise bizim programımız için "referans belirtme" olarak algılanmaktadır. Örneğin "sol üst köşesine" gibi yer belirten ifadeler anlatımda belirsizliği ortadan kaldırma amaçlı olarak referans nesne istemektedirler. Çünkü böyle bir ifadeyi gördüğümüz zaman akla gelen ilk soru "Kimin?" ya da "Neyin?" biçiminde olmaktadır. Programımızda yer ifadelerinin referans nesne gerektirmeleri durumu sözdizim bölümünde de anlatıldığı gibi öbek oluşturma kuralları vasıtasıyla zorunlu kılınmaktadır. Belirttiğimiz nedenlerden ötürü sözlüğümüzde –nin ekine sahip kelimelerimiz için yapılan anlamsal girişler yukarıda gösterdiğimiz yalın hallerinden bir yönüyle farklıdır. Örnek bir giriş aşağıdaki gibidir.

```
lexeme(      phon::mavinin   &
           syn::(cat::noun & case::gen & subcat::[]) &
           sem::(par::[X] & pred::(blue(X) & ref_obj(X)) &
                 spec::object)).
```

Görüldüğü üzere fark nesnenin renginden sonra gelen ve nesne olduğunu gösteren bilginin "referans nesne" olarak değiştirilmesi biçimindedir. Aynı durum sözlüğümüzdeki tüm –nin eki almış isimler için geçerlidir. Örneğin "sistemin", "ekranın", "pencerenin", "kırmızının", "yeşilin".

Buraya kadar anlattığımız isimlerin –e ve –i halleri için de girişler ilk olarak verdiğimiz yalın hallerindeki gibidir. Yani bir nesneyi sahip olduğu özellik ile göstermektedirler.

Belirtmemiz gereken önemli bir nokta da şudur; "mavi" gibi isim olarak girilen bir kelime için sadece "blue(X) & box(X)" gibi girişin yapılmama sebebi hem az önce açıklandığı gibi iyelik eki aldıkları zaman referans nesne olma özelliği kazanmaları hem de şimdi anlatacağımız ve zamir olmadıkları için "referans" olarak nitelendirdiğimiz "olan", "kutu", "kutucuk" gibi kelimelerin anlam değerlerinin verilebilmesidir.

Çünkü "olan", "kutu", "kutucuk" gibi kelimeler bir belirleyiciye ihtiyaç duymaktadırlar. Bizim programımızda da tek ayırdedici özelliğin renk olduğu daha önce de

belirtilmişti. Dolayısıyla "olan" gibi bir kelime, özelliği birlikte kullanıldığı kelime tarafından belirlenen bir nesne belirtmektedir. O sebeple anlamsal girişi sadece nesne gösterecek biçimde yapılmıştır. Örnek giriş aşağıdaki gibidir.

```
lexeme(      phon::olan      &
           syn::(cat::noun & case::nom &
                 subcat::[syn::(cat::adj & case::nom & subcat::[]) &
                          sem::(_ & spec::_)]) &
           sem::(par::[X] & pred::(obj(X)) & spec::object)).
```

Görüldüğü gibi anlamı sadece "obj(X)" biçiminde girilmiştir. Ancak böyle bir durumda akla "Hangi nesne?" diye bir soru gelecektir. Bu durum bir öbek oluşturma kuralımız ve referans belirten kelimelerimizin subcat listeleri aracılığı ile çözülmektedir. Kelimelerimizin subcat listeleri ayırdedici bir özellik (renk belirten bir kelime) ile kullanımı zorunlu kılmakta, öbek oluşturma kuralımız da bu şekilde bir isim öbeği oluşturmaktadır. Renk bilgisi ile "olan" kelimesinin birleşimi ile "mavi olan" gibi bir yapı birleştiği zaman da "blue(X) & obj(X)" gibi bir anlamsal gösterim ortaya çıkmaktadır. Referans olarak nitelediğimiz kelimelerin ayırdedici özellik belirten bir sıfat ile kullanılması ve anlamsal gösterimin oluşturulmasını sağlayan öbek oluşturma kuralı aşağıdaki gibidir.

```
(syn::(cat::noun & case::NOUN_CASE & subcat::[]) &
 sem::(par::X & pred::(PRED_OBJ & PRED2) & spec::object)) ==>

      [(syn::SYN_SUB &
        sem::(par::X & pred::(PRED_OBJ) & spec::SPEC_SUB)),

       (syn::(cat::noun & case::NOUN_CASE &
        subcat::[syn::SYN_SUB & sem::(_ & spec::SPEC_SUB)]) &
        sem::(par::X & pred::(PRED2) & spec::object))].
```

Kural ile oluşturulan bir isim öbeğidir. Görüldüğü gibi ilk parametre ile belirtilen sıfatın anlamsal gösterimi (PRED_OBJ) ile referans olarak nitelediğimiz kelimenin anlamı (PRED2) birleştirilerek isim öbeğinin anlamı için bir gösterim elde edilmektedir. Ayrıca takı bilgisini de referans belirten kelimeden almaktadır. "Mavi olan" gibi bir öbek için elde edilen gösterim daha önce de belirttiğimiz gibi "blue(X) & obj(X)" biçiminde, takı bilgisi ise yalın olacaktır.

Burada “mavi”, “mavi kutu/kutucuk” ile “mavi olan” arasındaki ayrımın önemli olduğunu düşünüyoruz. “olan” kelimesi için gösterim olarak “obj(X)” biçiminde bir ifade kullanılmıştır. Dolayısıyla ilk iki örneğimizde elde edilen ifade “blue(X) & box(X)” biçiminde olacakken son örneğimiz için “blue(X) & obj(X)” biçiminde olacaktır. Burada sanki iki ifadenin yorumundan farklı anlamlar çıkıyormuş gibi görünse de durumu bir hiyerarşi ile açıklamak mümkündür. Burada “obj” ile kastedilenin “box” ile kastedilenin genel hali olduğu düşünülebilir. “Her kutu aynı zamanda bir nesnedir” yaklaşımı ile ortaya bir hiyerarşi çıkmaktadır. Bu tür bir hiyerarşiye İngilizce “taxonomy” denilmektedir. Ama özellikle dikkat edilmesi gereken husus İngilizce “is a” Türkçe ise “aynı zamanda” olarak değerlendirilebilecek ilişkinin elemanlar arasında mevcut olduğudur. Konu ile ilgili olarak (Russel ve Norvig 1995)te bulunan “Building a Knowledge Base” isimli bölüme ve (Löbner 2002)de “Lexical Fields” isimli bölümlere bakılabilir. Her iki kaynakta da belirttiğimiz özelliklerde bir hiyerarşi kullanarak girişleri kategorilere ayırmanın ve “aynı zamanda” ilişkisinin kullanarak işlem yapmanın bize kazandırdıkları açık biçimde anlatılmaktadır.

Bizim programımızda şu an için işleyen bir hiyerarşi kodlanmış değildir. Bahsettiğimiz hiyerarşiyi mantıksal bir ilerletme olarak kabul etmek mümkündür.

Referans belirten kelimelerimiz için de –nin takısı almış halleri referans nesne gösterme olarak yorumlanmıştır. Çünkü “mavi olanın” gibi bir yapı düşünüldüğünde iki kullanım akla gelmektedir. Biri “Yeşili mavi olanın yanına koy” diğeri ise “Yeşili mavi olanın sol üst köşesine koy” gibidir. Her iki durumda da isim öbeği ile belirtilen şey bir referans nesne olmaktadır. O sebeple referans belirten kelimeler olarak nitelediğimiz kelimelerin anlamsal değerleri sadece nesne olarak değil “ref_obj(X)” biçiminde referans nesne biçiminde girilmiştir. Böylece “mavi olanın” gibi bir kullanımda “blue(X) & ref_obj(X)” biçiminde bir gösterim elde edilmektedir.

Referans belirten kelimeler için önemli olan “mavi” ile “mavi olan / kutu / kutucuk” gibi bir öbeğin, “mavinin” gibi bir kullanım ile de “mavi olanın / kutunun / kutucuğun” gibi bir öbeğin hiyerarşimiz de düşünülecek olursa aynı anlamsal gösterime sahip olmasıdır. İlki

için örneğin “blue(X) & box(X)” veya “blue(X) & obj(X)”, ikincisi için de örneğin “blue(X) & ref_obj(X)” gibi bir gösterim elde edilecektir.

7.2.4 Yön bildiren kelimeler ve ifadeler

Yön, konum ve yer bildiren kelimelerle bu türden ifadeler oluşturmayı sağlayan öbek oluşturma kurallarını açıkladıktan sonra fiillere ilişkin açıklamalara geçmenin daha uygun olduğunu düşündüğümüz için sıralamada yine bir değişiklik yapmayı uygun bulduk.

Sözlüğümüzde yön belirten kelime olarak altı giriş bulunmaktadır. Bunlardan ikisi “doğru” kelimesi içindir. Çünkü bu kelime bizim için özel bir durum oluşturmaktadır ve özel bir öbek oluşturma kuralı ile işlem görmektedir. Anlamsal gösterim olarak “toward(X)” biçiminde bir değere sahiptir. Onun dışında “sağ”, “sol”, “alt” ve “üst” biçiminde dört kelimemiz daha mevcuttur. Bunlar da sırasıyla “right(X)”, “left(X)”, “under(X)” ve “top(X)” biçimindedir. Yön bildiren kelimelerimiz için asıl önemli olan öbek oluşturma kurallarımızdır.

“doğru” kelimesi ile ilgilenen öbek oluşturma kuralımız aşağıdaki gibidir.

```
(syn::(cat::noun & case::dat & subcat::[]) &
 sem::(par::X & pred::(PRED_YON & PRED_REF) & spec::yon)) ==>

[(SYN_REF_OBJ & sem::(par::X & pred::(PRED_REF) & spec::SPEC_REF_OBJ)),

 (syn::(cat::noun & case::nom &
  subcat::[SYN_REF_OBJ & sem::(_ & spec::SPEC_REF_OBJ)]) &
  sem::(par::X & pred::(PRED_YON) & spec::yon))].
```

Burada kuralın gövde kısmında bulunan ikinci kelime “doğru”dur. Sözlükteki girişinde belirtilen subcat listesine uygun bir eleman ile birleşerek öbek oluşturması sağlanmaktadır. Öbeğin anlamsal gösteriminde ise “doğru”nun önermesi (PRED_YON) gerektirdiği yapının önermesinin (PRED_REF) önüne getirilmektedir. Bu şekilde “kırmızıya doğru” ya da “kırmızının sol üst köşesine doğru” gibi bir kullanımda mantıksal gösterim “toward(X) & ... ” biçiminde olmaktadır. Düşünce olarak ilk akla gelen “Nereye doğru?” gibi bir soru olacaktır ve yönelme belirttiği için “doğru”nun gösterimi öne

alınmıştır. Değişik bir ifade ile bir yönelme vardır ancak bunu bir yöne doğru olması gerekir. Yani yönelmenin olduğu yön kısıtlayıcı şartmış gibi düşünülebilir.

Diğer yön bildiren ifadeler için öbek oluşturma kuralı ise herhangi bir sıralama değiştirmesi yapmaksızın kendisine gönderilen kelimelerin gösterimlerini sırası ile birleştirmektedir. Öbek oluşturma kuralı aşağıdaki gibidir.

```
(syn::(cat::noun & case::nom & subcat::[]) &
 sem::(par::X & pred::(PRED1 & PRED2) & spec::yon)) ==>

      [(syn::(cat::noun & case::nom & subcat::[]) &
        sem::(par::X & pred::(PRED1) & spec::yon)),

       (syn::(cat::noun & case::nom & subcat::[]) &
        sem::(par::X & pred::(PRED2) & spec::yon))].
```

Görüldüğü gibi kuralın gövde kısmındaki elemanların anlamsal gösterimleri sırasıyla PRED1 ve PRED2 olarak alınıp aynı sıra ile birleştirilmektedir. “sol alt”, “sağ üst” gibi yapılar için gösterim bu şekilde elde edilmektedir.

7.2.5 Konum belirten kelimeler

Sözlüğümüzdeki konum bildiren kelimeler basit birer gösterime sahiptir. Zaten sözdizim bölümünde de belirttiğimiz gibi konum bildiren iki farklı kelimemiz ve bunların iki farklı halleri (-e ve –den halleri) bulunmaktadır. Kelimelerimizden “köşesine” ve “köşesinden” için “corner(X)” biçiminde “yanına” ve “yanından” için “beside(X)” biçiminde gösterimler tercih edilmiştir. Konum ifadesi oluşturmak için herhangi bir öbek oluşturma kuralımız mevcut değildir. Konum ifadeleri yer belirten ifadeler oluşturmaya yarayan kurallar ile işlem görmektedirler.

7.2.6 Yer belirten kelimeler ve ifadeler

Yer belirten kelimelerimiz sözdizimsel olarak kendi içlerinde referans nesne isteyenler, istemeyenler gibi ayrıma sahip olsalar da anlamsal değerleri bakımından birbirlerinden farklı değillerdir. O sebeple aldıkları eklere göre subcat listeleri değişmesine karşın anlamsal gösterimleri aynı kalmaktadır ve anlamsal gösterimlerini gruplayarak vermek mümkündür. Sözdizim bölümümüzde anlatılan sebeplerden dolayı iki giriş yapılan

“köşesine” ile “köşesinden” kelimeleri için “corner(X)”, yine aynı biçimde iki giriş yapılan “yanına” ile “yanından” kelimeleri için “beside(X)”, “sağına”, “sağından” ve “sağa” kelimeleri için “right(X)”, “soluna”, “solundan” ve “sola” kelimeleri için “left(X)” biçiminde gösterimler tercih edilmiştir.

Yukarıda belirtilenlerin dışında yer belirten ifade oluşturmak için iki tane öbek oluşturma kuralımız vardır. Bunlardan ilki bir yön bilgisi ile konum ifadesini birleştiren aşağıdaki kuralımızdır.

```
(syn::(cat::noun & case::CASE & subcat::[syn::(cat::noun & case::gen &
 subcat::[]) & sem::(_ & spec::object)]) &
 sem::(par::X & pred::(PRED_KONUM & PRED_YON) & spec::yer)) ==>

            [(syn::SYN_YON &
              sem::(par::X & pred::PRED_YON & spec::SPEC_YON)),

             (syn::(cat::noun & case::CASE &
              subcat::[syn::SYN_YON & sem::(_ & spec::SPEC_YON)]) &
              sem::(par::X & pred::PRED_KONUM & spec::konum))].
```

“sol üst köşesine” gibi yapılar için kullanılan bir kuralımızdır. Dikkat edilirse konum belirten ikinci kelimenin anlamsal kısmı yön belirten kelimenin önüne alınmaktadır. İfadeye bakıldığı zaman “köşe” olduğu öbekte bellidir. Hangi köşe olması gerektiği ise kısıtlayıcı bir şart gibi düşünülebilir. Yani “corner(X) & left(X) & top(X)” şeklinde elde edilen ifade de “Yer olarak ifade edilen köşedir ve şu köşedir” biçiminde bir mantıksal gösterim elde edilmektedir.

İkinci yer belirten ifade oluşturmaya yönelik öbek kuralımız ise yer ifadelerinin gerek duydukları referans nesne ile birleşmelerini sağlamaktadır. Kuralımız aşağıdaki gibi girilmiştir.

```
(syn::(cat::noun & case::CASE & subcat::[]) &
 sem::(par::X & pred::(PRED_REF & PRED_YER) & spec::yer)) ==>

      [(syn::(SYN_REF_OBJ) &
        sem::(par::X & pred::(PRED_REF) & spec::SPEC_REF_OBJ)),

       (syn::(cat::noun & case::CASE &
        subcat::[syn::(SYN_REF_OBJ) & sem::(_ & spec::SPEC_REF_OBJ)]) &
        sem::(par::X & pred::(PRED_YER) & spec::yer))].
```

Burada ifadenin anlamına ilişkin gösterim oluşturulurken sıralamada herhangi bir değişiklik yapılmamış olduğu sıra ile birleştirilmiştir. Örneğin "kırmızının sol üst köşesine" gibi bir yapı için "(red(X) & ref_obj(X)) & corner(X) & left(X) & top(X) gibi bir mantıksal ifade elde edilecektir.

7.2.7 Fiiller ve fiillere ilişkin ifadelerin elde edilmesi

Bu noktaya kadar açıkladığımız kelime türlerimiz için sözlüğümüzde yapılan anlamsal gösterimlerde önermeler tek parametreli idi. Bir bakıma herbiri bir özelliği gösterdiği için bu şekilde idi denebilir. Ancak fiillerimiz için sözlüğümüzde yapılan girişlerimiz tek, iki ve üç parametreli olarak değişmektedir. Bazı fiillerimizin hem iki hem de üç parametreli biçimleri bulunmaktadır. En başta da belirttiğimiz gibi tüm sözlük girişlerimiz için sem(antic) özelliğinin bir alt özelliği olan par(ameters) özelliği liste biçimindedir. Belirtilen parametreler pred(icate) özelliğinde gereken sıraya konularak kullanılmıştır. Sözdizim özelliklerini açıkladığımız kısmında fiillerimizi neye göre sınıflandırdığımızı maddeler halinde vermiştik. O sebeple burada tekrar açıklamaya gerek duymuyoruz.

Fiillerimizin mantıksal gösterimlerinin parametrik yapılarının bizim için şöyle önemli bir yanı vardır. Fiil öbeği, V-bar ya da cümle oluşturmak için kullandığımız öbek oluşturma kurallarımızda istedikleri bileşenlerin parametresi ile fiilin istediği parametreler uyum göstermek zorundadır. Durumu bir örnek ile açıklamak gerekirse KİM, NEREYE türünde bir fiil olan "götür" için gösterimimiz aşağıdaki gibi olacaktır.

```
lexeme(      phon::gotur &
           syn::(cat::verb & case::nom &
           subcat::[(syn::(cat::noun & case::dat & subcat::[]) &
                          sem::(_ & spec::yer)),
                    (syn::(cat::noun & case::acc & subcat::[]) &
                          sem::(_ & spec::object))]) &
           sem::(par::[X,Y] & pred::put(X,Y) & spec::action)).
```

Görüldüğü gibi iki parametrelidir (sem özelliğinin par alt özelliği [X,Y] biçimindedir). Parametrelerin eşleneceği ifade türleri ise subcat listesinde sıra ile belirtilmiştir. En başta Lambda Soyutlaması'ndan bahsederken dediğimiz gibi fiiller için

kısıtlayıcı şartlar olan subcat listeleri hangi değişkenin ne türden bir ifade ile eşleşebileceğini göstermektedir. Subcat listesinin elemanları ile parametreler birbirine paralel olarak düşünülmelidir ancak ters sırada paralel düşünülmelidir. Çünkü fiillerin subcat listeleri kendilerine yakın olan cümle elemanından uzak olan cümle elemanına doğru girilmiştir. Yani cümlenin soldan başladığı düşünülecek olursa ters sıradadır. O sebeple X parametresi KİMİ biçiminde düşünülebilecek olan –i halinde nesne gösteren bir isim ifadesi ile eşleşmelidir. Y parametresi ise yer belirten –e halinde bir ifade ile eşleşmelidir. Burada eşleşmeden kastımız nesne gösteren ifadenin pred özelliğindeki değişkenin (genellikle X biçiminde belirtilmiş olan) fiilin ilk parametresi ile eşleşmesidir. Aynı durum yer ifadesi ve ikinci parametre için de geçerlidir. Prolog'da değişken olarak belirlenen ifadeler eğer bir sabit ile eşleştirilemezlerse Prolog'un çalışma zamanı sırasında değişken tanımlama kurallarına göre tanımladığı değişkenler olarak görünürler. Yani kodda X gibi belirtilen bir değişken çalışma zamanı sırasında _G234 gibi bir değer alabilir. Prolog komut satırından çalıştırıldığında bir değişkeni bir sabit ile yer değiştiremez ise bu şekilde cevaplar geri döndürmektedir. Bizim de programımızda değişkenler hiçbir sabit ile eşleşmemekte ve sonuçta çalışma zamanında Prolog'un tanımladığı değişkenler olarak kalmaktadırlar.

Bu durumda "put(X, Y)" biçimindeki bir gösterim komut satırında "put(_G234, _G567)" gibi bir hal alabilmektedir. Benzer biçimde diğer bütün kelimelerin gösterimlerinde de Prolog değişkenleri görünecektir. Bu durumda fiilin değişkenleri ile istediği bileşenlerin değişkenlerinin eşleşmesinden kasıt komut satırında aynı Prolog isimlendirmesi ile görülmesidir. Örneğin "Mavi olanı yeşilin sol üst köşesine götür" gibi bir cümle için aşağıdaki gösterim elde edilmektedir.

```
put(_G352, _G554)& (blue(_G352)&obj(_G352))&
   (green(_G554)&ref_obj(_G554))&corner(_G554)&left(_G554)&top(_G554)
```

Cümleden de anlaşıldığı gibi "mavi olanı" ifadesi fiilimizin istediği ilk parametre, "yeşilin sol üst köşesi" ise yine fiilimizin ikinci parametresidir. Ancak daha önceki bölümlerde de açıkladığımız gibi "yeşilin sol üst köşesine" ifadesi tek adımda oluşturulmamakta ve "sol üst", "sol üst köşesine", "yeşilin sol üst köşesine" biçiminde üç farklı öbek oluşturma kuralı vasıtasıyla oluşturulmaktadır. Tüm bu işlemler esnasında da

içlerindeki parametrelerin bir şekilde fiilden alınması ya da eşleştirilmesi gerekmektedir. Bu durum öbek oluşturma kurallarımız tarafından yapılmaktadır. Çünkü açıklamış olduğumuz durum sadece fiiller için değil öbek oluşturma kurallarımızın tamamı için geçerlidir. Yukarıdaki örneğimizde "yeşilin sol üst köşesine" ifadesi oluşturulurken de aynı parametre ile eşleşmeleri gerekmektedir. Bu durum fiillere ilişkin öbek oluşturma kuralları açıklandıktan sonra şekil ile daha iyi biçimde ifade edilecektir. Ancak örneğimizdeki _G352 ve _G354 olarak ifade edilen değişkenlerin hangi gösterimler için ortak olarak kullanıldığına da dikkat edilmelidir.

Programımızda fiil öbeği oluşturmak için iki adet öbek oluşturma kuralımız bulunmaktadır. İlki ikincisinin özel bir hali olan bu kuraldan neden iki tane olmasına ihtiyaç duyulduğu sözdizim bölümümüzde ayrıntılı olarak açıklanmıştır. Dolayısıyla iki kuralı birarada vererek kısa bir açıklama yapmayı tercih edeceğiz. Kurallarımız aşağıdaki gibidir.

```
(syn::(cat::vp & case::abl & subcat::Rest) &
 sem::(par::[First,Second] & pred::(PRED_VERB & PRED_YER) & spec::_)) ==>

   [(syn::(cat::CAT & case::abl & subcat::SUB) &
     sem::(par::RestPar & pred::(PRED_YER) & spec::SPEC)),

    (syn::(cat::verb & case::nom &
     subcat::[syn::(cat::CAT & case::abl & subcat::SUB) &
     sem::(_ & spec::SPEC)|Rest]) & sem::(par::[First,Second|RestPar] &
           pred::(PRED_VERB) & spec::action))].


(syn::(cat::vp & case::CASE & subcat::Rest) &
 sem::(par::[First] & pred::(PRED_VERB & PRED_YER) & spec::_)) ==>

      [(syn::(cat::CAT & case::CASE & subcat::SUB) &
        sem::(par::Second & pred::(PRED_YER) & spec::SPEC)),

       (syn::(cat::verb & case::nom &
        subcat::[(syn::(cat::CAT & case::CASE & subcat::SUB) & sem::(_ &
                                              spec::SPEC))|Rest]) &
        sem::(par::[First,Second] & pred::(PRED_VERB) & spec::action))].
```

Her iki kuralda da fiilin önermesi ve bileşenin önermesi alınıp fiilinki önde olacak biçimde birleştirilmektedir. Ancak en önemli nokta yukarıda sözünü etmiş olduğumuz değişkenlerin eşleşmesidir. Sözdizim bölümünde de belirtildiği gibi ilk kural üç parametreli

fiiller için girilmişti. Fiilin istediği parametreler ve onların cümlede bulunma sıraları da ters olduğu için ilk kuralda üç parametreli bir fiilin ikinci parametresi, ikinci kuralda ise ikinci parametresi ile gösterilen ifade bulunmaktadır. Dikkat edilirse kurallardan ilkinde fiilin par(ameters) listesi [First,Second | RestPar] biçiminde bölünmüştür. Üç parametreli fiilerde kullanılan ilk kuralda RestPar ile gösterilen parametre istenen bileşenin par özelliği ile eşleştirilmiş, iki parametreli fiillerde ise Second ile gösterilen parametre istenen bileşenin par parametresi olarak belirlenmiştir. Vermiş olduğumuz kurallarımıza dikkatle bakılırsa durum açıkça görülmektedir.

Dikkat edilmesi gereken bir diğer husus ise fiilin par elemanının bir azaltılarak elde edilen öbeğe aktarılmış olmasıdır. Sözdizim kısmında belirtildiği gibi subcat listesi de azaltılarak alınmıştır.

Aynı durum üç parametreli fiiller için kullandığımız V-bar kuralımız için de geçerlidir. Kuralımız aşağıdaki gibidir.

```
(syn::(cat::vbar & case::dat & subcat::Rest) &
 sem::(par::[FirstP] & pred::(PRED_VERB & PRED_KIME & PRED_YER) &
       spec::_)) ==>

      [(syn::(First) &
        sem::(par::SecondP & pred::(PRED_KIME) & spec::SpecFirst)),

      (syn::(cat::vp & case::abl &
       subcat::[syn::(First) & sem::(_ & spec::SpecFirst)|Rest]) &
       sem::(par::[FirstP,SecondP] & pred::(PRED_VERB & PRED_YER) &
             spec::_))].
```

Üç parametreli fiillerimizde V-bar ile ikinci parametre elemanı elde edilmektedir. Böylece geriye hem subcat hem de parametre olarak birer eleman kalmakta ve son adım olarak cümle oluşturma işlemine gönderilmektedir. Ayrıca oluşturulan öbeğin pred elemanına dikkat edilirse fiilin subcat listesinde bulunan elemanların subcat listesine göre ters sırada ama cümlede bulunmalarına göre aynı sırada olduklarına dikkat edilmelidir. Örneğin; “Mavi olan yeşile sol üst köşesinden değsin” gibi bir cümlede V-bar aşamasına gelindiğinde “yeşile” ve “sol üst köşesinden” yapıları sırasıyla fiilin gerektirdiği ikinci ve üçüncü elemanlar olarak belirlenecektir. “sol üst köşesinden” fiil öbeği olarak belirlenecek

ve üçüncü parametre ile, “yeşile” ise V-bar olarak değerlendirilip ikinci parametre ile eşleşecektir.

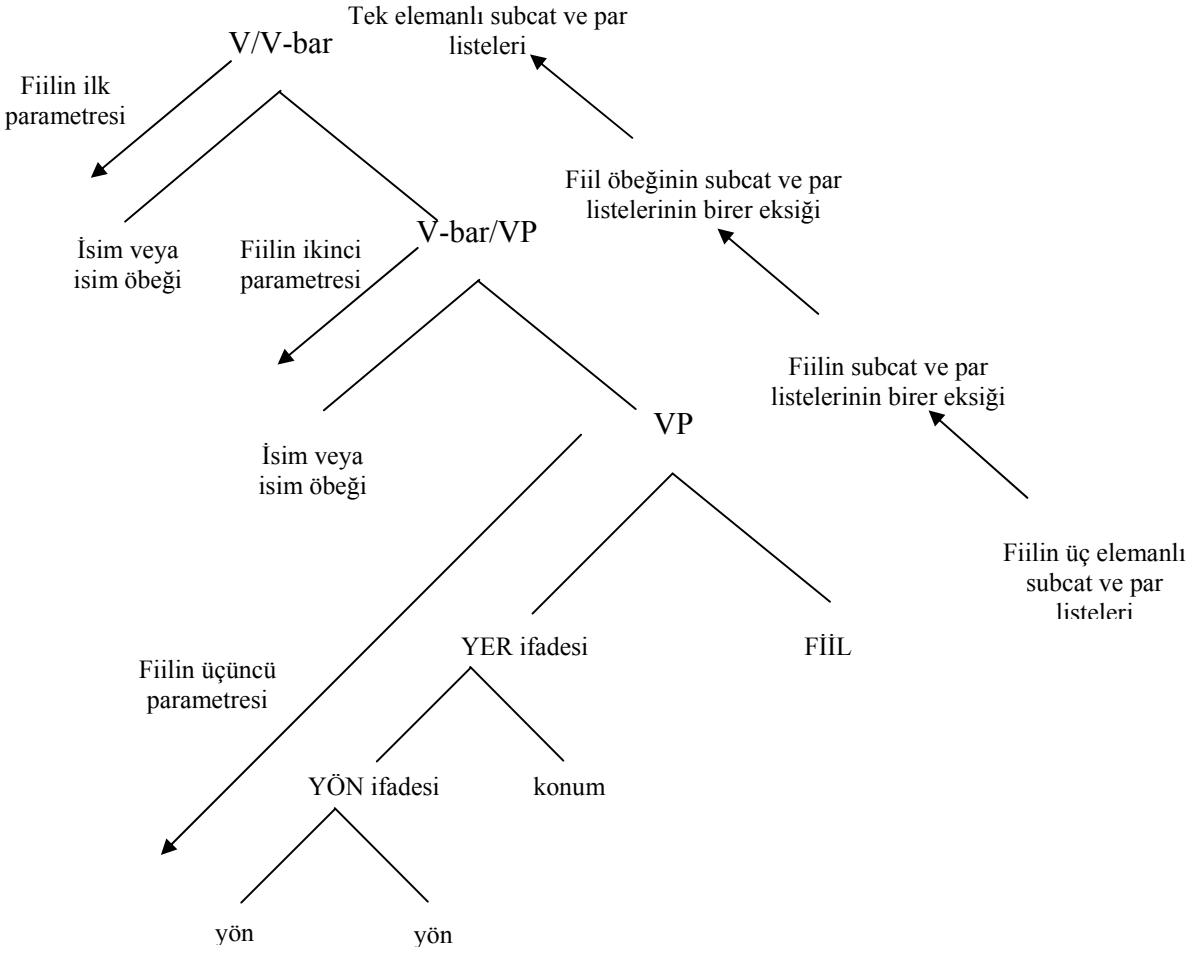
Artık son adımda elde edilen fiile dair bu ifadelerin gereken son eleman ile birleştirilmesi işlemi gerçekleştirilecektir. Daha önce de söylediğimiz gibi programımızdaki tüm yapılar programımızın yapacağı işe göre şekillenmiştir. Gramer kurallarımız da kullanılabilecek cümle yapılarına göre şekillenmiştir. Dolayısıyla Türkçe’de mevcut tüm cümle yapılarının ancak çok küçük bir parçasını işleyebilmekte ve anlam analizi yapabilmektedir. Tüm bu hatırlatmalardan sonra son iki kuralımızın da aşağıdaki gibi olduğunu söyleyebiliriz.

```
(syn::(cat::verb & case::nom & subcat::[]) &
 sem::(par::[First|Rest] &
       pred::(PRED_VERB & PRED_KIM & PRED_KIME & PRED_YER) &
       spec::action)) ==>

[(syn::(SYN_OBJECT) &
  sem::(par::First & pred::(PRED_KIM) & spec::SPEC_OBJECT)),

 (syn::(cat::vbar & case::dat &
  subcat::[syn::(SYN_OBJECT) & sem::(_ & spec::SPEC_OBJECT)])) &
  sem::(par::[First|Rest] & pred::(PRED_VERB & PRED_KIME & PRED_YER) &
        spec::_)].


(syn::(cat::verb & case::nom & subcat::[]) &
 sem::(par::[First|Rest] & pred::(PRED_VERB & PRED_REF & PRED_YER) &
       spec::action)) ==>

      [(syn::(SYN_OBJECT) &
        sem::(par::First & pred::PRED_REF & spec::SPEC)),

       (syn::(cat::vp & case::_ &
        subcat::[syn::(SYN_OBJECT) & sem::(_ & spec::SPEC)])) &
        sem::(par::[First|Rest] & pred::(PRED_VERB & PRED_YER) &
              spec::_)].
```

Kurallarımızdan ilki üç parametreli fiillerimiz için ikincisi ise iki parametreli fiillerimiz için gereken son parametreyi bularak cümleyi oluştururlar. Anlamsal kısımda önemli olan pred parametresi ise yine cümlede bulunma sırasına göre oluşturulmuştur.

Bu son açıklama ile programımızda mevcut olan bütün öbek oluşturma kurallarını açıklamış bulunuyoruz. Aşağıdaki şekil ile hem üç parametreli fiillerde istedikleri

parametrelerin gereken yapılarla eşleştirilmesi açıklanmış hem de bir cümlenin çözümlenmesi esnasında öbek oluşturma kurallarının nasıl kullanıldığı anlatılmak istenmiştir. Yine "Mavi olan yeşilin sol üst köşesine değsin" gibi bir cümlenin çözümlendiğini varsayacağız. Sözdizim kurallarımızda da anlatıldığı gibi yer ifadeleri doğrudan sözlükten de alınabilmektedir. Ayrıca iki parametreli fiilerimiz için tekrardan şekil çizmeye gerek duymuyoruz. Çünkü durum benzer olacaktır. Olabilecek en geniş ağaçlardan biri ile durmu açıklamayı yeterli buluyoruz.



Şekil 7.1 Üç parametreli fiiller için fiilin parametrelerinin belirlenmesi ve eşleşme işlemi

Bu şekil ile hem sözdizim kısmımızda anlatılan bazı değerlerin daha üst kademelere gönderilmesi hem de parametrik eşleşmenin yukarıdan aşağı nasıl yapıldığının daha iyi anlaşılacağını düşünüyoruz. Bu noktaya kadar öbek oluşturma kurallarımız ile yazılı olarak ifade ettiğimiz durumun şekilsel gösterimi Şekil 7.1'deki gibidir.

7.3 Ayrıştırıcımızda Yapılan Değişiklik

Ayrıştırıcımızın üç kuralının olduğu ve bunlardan üçüncüsünün girilen cümledeki kelimeyi phon özelliğine bakarak sözlükte aradığını ve başarılı olursa yığına attığını sözdizim bölümümüzde açıklamıştık. Ancak sözdizim analizi için sadece syn özelliğinin yığına atılması yeterli iken anlam analizi için artık syn(tax)'ın yanında sem(antic) özelliğinin de yığına atılması gerekmektedir. Ayrıştırıcımızın son kuralı bu şekilde düzeltilmiştir. Ayrıca durma şartı olarak nitelediğimiz ilk ayrıştırıcı kuralında da artık cümlenin anlamsal gösterimi elde edilerek durulmaktadır. İkinci kuralımızda ise bir değişikliğe gerek yoktur. Sırasıyla durma şartımız ve son ayrıştırıcı kuralımız aşağıdaki gibidir.

```
parser([syn::(cat::verb & case::nom & subcat::[]) & sem::(par::_ &
                        pred::SEM & spec::action)],[],SEM).


parser([First,Second|Rest], String, Result) :-
       Mother ==> [Second,First],
       parser([Mother|Rest], String, Result),!.

parser(Stack, [First|Rest], Result) :-
       lexeme(phon::First & syn::SYN_WORD & sem::SEM_WORD),
       parser([syn::SYN_WORD & sem::SEM_WORD | Stack], Rest, Result).
```

Ayrıştırıcımız bu şekilde değiştirildiği için – özellikle durma şartımız – sözdizim kısmında anlatıldığı gibi "run" yardımcı prosedürümüz ekranda "ok" biçiminde bir yazıyı değil girilen cümlenin elde edilen anlamsal gösterimini ekrana çıkarmaktadır. Örnek bir kullanım aşağıdaki gibidir.

```
?- run.
|: maviyi yesilin sol kosesine gotur .
put(_G328, _G392)& (blue(_G328)&obj(_G328))&
            (green(_G392)&ref_obj(_G392))&corner(_G392)&left(_G392)

Yes
```

7.4 Programımız Hakkında Ek Bilgi

Buraya kadar yapmış olduğumuz açıklamalarımızdan da anlaşıldığı gibi anlam analizimizde Montague'nin yaklaşımı benimsenmiş ve uygulanmaya çalışılmıştır. Bunun yanında Durum Teorisi'ndeki kısıtlanmış değişkenler kavramına benzer bir kullanıma çalışılmıştır. Lambda Soyutlaması, (Pereira ve Shieber 1987)de anlatıldığı gibi ve Beta Kısaltması kullanılarak yapılmak yerine kelimelerin sem(antic) özelliklerine par(ameters) ve pred(icate) alt özellikleri katılarak yapılmıştır.

Programımızın mevcut halinde dünya modeline başvuru ya da çıkarımda bulunma işlemleri bulunmamaktadır. Özellikle sorgu / soru cümlelerinin de bulunmaması ile bu yapıya yer verilmemiştir. Ancak yapılabilecek ilerletme çalışmalarından biri de programımıza mevcut cümle yapılarının ve açıklandığı gibi gerektirdiği modüllerin katılması olabilir.

Durum Teorisi'ndeki kısıtlayıcı şartlar kavramı burada fiilin parametreleri ile gerektirdiği yapıların değişkenlerinin (sem özelliği altındaki par ile gösterilen alt özellikte verilen elemanlar) eşleştirilmesi ile elde edilmeye çalışılmıştır. Yapılan işlem bölüm içinde ayrıntılı olarak açıklanmıştır.

8. SONUÇLAR VE GELECEKTE YAPILABİLECEK ÇALIŞMALAR

8.1 Sonuçlar ve Değerlendirmeler

Tezimizin birincil amacı olan Montague yaklaşımını kullanarak Türkçe emir cümleleri için biçimsel gösterimlerin elde edilmesi işlemi yazdığımız programımız aracılığı ile gerçeklenmiştir. Programımız girilen cümleleri yüklem mantığına dönüştürülmüş olarak görüntüleyebilmektedir.

Tezimizin teorik bir amacı/katkısı olarak niteleyebileceğimiz durum ise "küçük", "kapalı" ya da "kendine has" dünyalar kavramı üzerinde durmasıdır. Bu durum da programımızın içinde mevcuttur. Özellikle kelimelerimizin anlambilimsel değerlerinden sözcük öbeği oluşturma kurallarımıza kadar her durumda bu göze çarpmaktadır. Bunun en açık örneği olarak sözdizim analizi yapılırken sadece isim öbeği vb. biçiminde belli bir yapının değil aynı zamanda yer ifadesi vb. biçiminde neye karşılık gelen yapının/neyin elde edildiğinin üzerinde durulmasıdır. Yaklaşımımızın sonucu olarak yapılacak işe göre şekillenmiş bir dünya anlayışının hem (daha geniş bir anlayışa sahip dünya modeli ile kıyaslandığında) belli ayrıntılardan bizi uzak tutması ve tasarruf sağlaması hem de istenilen işi yapabilecek yetenekte olması başlangıçta belirttiğimiz bu hedefimize de ulaştığımızı göstermektedir. Çünkü SHRDLU'da olduğu gibi bizim programımız da (aslında burada kastettiğimiz program içinde görünmeyen ve emirleri verdiğimiz varlık) kendine ait dünya anlayışı dışında kalan olaylar hakkında herhangi bir fikre sahip değildir.

Bu noktadan sonra tezimiz ve programımız hakkında şu değerlendirmeler de yapılabilir. Her ne kadar Montague yaklaşımı kullanılmış olsa da Durum Teorisi'ndeki kısıtlanmış değişkenler benzeri yapıların özellikle fiillerimiz için kullanıldığı düşünülebilir. Belirtilen durumu bir açıdan tematik roller bir açıdan da Durum Teorisi'ndeki alt sınır şartları ile açıklamak mümkündür. Bu durum sanki yaklaşımımızdaki netliğin bozulması imiş gibi görülse de Durum Teorisi'nin düşünsel/inançsal bağlam vb. alanlarda Montague yaklaşımının eksiklerini kapatmasının onu tamamen dışlamak olmadığını, arada

benzerlikler bulunduğunu unutmamak gerekir. O sebeple tezimizin yaklaşımının net bir biçimde Montague yaklaşımı olduğunu söyleyebiliriz.

Bunların dışında üniversitemizde bilgisayarlı dilbilim alanında tamamlanan ilk tez olması açısından üzerinde durduğu teoriler bakımından başlangıç için anlaşılır bir ilk bilgi kaynağı olarak da görevini yerine getirebileceği kanaatindeyiz. Çünkü giriş bölümünde de söylediğimiz gibi tezimizi yazarken göz önünde bulundurduğumuz bir durum da budur. Bunun dışında yine tezimizin giriş bölümünde de söylediğimiz tamamlanmış bir çalışma olarak da değerlendirilmemesi, daha sonra yapılacak çalışmalar için bir ilk adım olarak görülmesi gerekmektedir.

8.2 Gelecekte Yapılabilecek Çalışmalar

İlk olarak şunu söyeleyebiliriz ki programımızda emir cümlelerinin kendisine verildiği ve görünmeyen bir varlık varmış gibi düşünülebilir. Yapılabilecek bir ilerletme, cevap verme vb. özellikler eklenerek bu varlığın etkileşimli ve dolayısıyla daha görünür bir varlık olmasını sağlamak olabilir.

İkinci olarak kullandığımız Montague yaklaşımı ve dünya kavramı Durum Teorisi'ne ugun biçimde değiştirilerek herşeyin bilindiği dünya anlayışından gerçekliğin sınırlı parçalarının bilindiği bir yaklaşımla yeniden ele alınabilir.

Bir diğer özellik olarak programımızın sözlüğünün, cümle yapılarının ve kipinin sınırlı olmasıdır. Örneğin "Mavi yeşilin yanında mı?" gibi mevcut durumu sorgulamaya yönlik bir cümle yapısı yoktur. Bu cümle yapısı beraberinde bir dünya modelinin de bulunmasını getirecektir. Programımızın mevcut olan halinde açıkça değil ama kelimelerin yorumlamasına yüklenmiş gizliden bir dünya modeli olduğu düşünülebilir. Öncelikle bu dünya modeli daha somut bir hale getirilip sonra bu modelin sorgulanabilir bir hal alması sağlanabilir. Bunların dışında sözlüğümüze "biraz" vb. miktar niceleyici kelimeler başta olmak üzere pek çok kelime eklenmesi ve buna bağlı olarak da yeni cümle yapılarının eklenmesi sağlanabilir.

Programımız içinde yine açıkça varolmayan "aynı zamanda" mantıklı olarak niteleyebileceğimiz hiyerarşi daha açık hale getirilebilir.

Programımızın grafik bir ara yüzü yoktur. Bu konuda Prolog'a sonradan eklenen ve XPCE denilen grafik arabirim kullanılabilir.

İNGİLİZCE TERİMLER İÇİN KULLANILAN TÜRKÇE KARŞILIKLAR

Abstraction : Soyutlama
Anchor : Değer atayıcılar
Anonymous variable : Anonim değişken
Bottom-up : Tabandan yukarı
Attribute Value Matrice : Özellik Değer Matrisi
Beta Reduction : Beta kısaltması
Case : Hal
Category : Kategori
Compositionality : Birleştirilebilme
Conjunction : Bi(r)leşim
Context : Bağlam
Context of utterance : Söylendiği bağlam
Described situation : Açıklanan durum
Disjunction : Fark
Environmental : Çevresel
Exception handling : Hata işleme
Extended Kamp Notation : Geliştirilmiş Kamp Notasyonu/Gösterimi
Fact : Gerçek
Focal situation : Odak durum
Framework : Çerçeve
Generative : Üretici
Grammar : Gramer
Heterarchical : Hiyerarşik olmayan
Individual : Şahıs
Infix : (Operatörler için) araya gelme durumu
Knowledge base : Bilgi tabanı
Lexeme : (Teknik anlamıyla) kelime
Lexicon : (Teknik anlamıyla) sözlü.

Linear Notation : Doğrusal gösterim

Location : Yer

Manipulative : İşleme yönelik

Model Theoretic : Model gerektiren

Noun Phrase (NP) : İsim öbeği

Object : Nesne

Parameter : Parametre

Parser : Ayrıştırıcı

Pattern : Örnek

Phrase Structure Rule : Sözcük öbeği oluşturma kuralı

Possible worlds : Olası dünyalar

Predicate : Yüklem

Property : Özellik

Proposition : Önerme

Reaoning : Muhakeme / çıkarım

Recursion : Özyineleme

Relation : İlişki

Resource situation : Kaynak durum

Restricted variable : Kısıtlanmış değişken

Rule : Kural

Semantic value : Anlamsal değer

Semantics : Anlambilim / anlamsal

Situation : Durum

KAYNAKLAR

ACZEL P. vd., Situation Theory and Its Applications Vol. 3, CSLI – U.S.A, 1993.

ALLEN J., *Natural Language Understanding*, Benjamin/Cummings Pub. Co., Redwood City, Calif., 1995.

BARWISE J. vd., Situation Theory and Its Applications Vol. 2, CSLI – U.S.A, 1991.

BARENDREGT H.P., The Lambda Calculus Its Syntax and Semantics (Revised Edition – Fifth impression), Elsevier Science B.V., Netherlands, 2001.

BLACK A., *Using Situation Theory in a computational language for natural language processing*, 4th Natural Language Understanding and Logic Programming Conference, Nara, Japan, 1993.

BRATKO I., *Prolog Programming for Artificial Intelligence*, 3rd Edition, Addison Wesley, Dorchester - Dorset, 2001.

COOPER R. vd., Situation Theory and Its Applications Vol.12, CSLI – U.S.A, 1990.

CRIMMINS, MARK[11] (1998). *Semantics*. In E. Craig (Ed.), *Routledge Encyclopedia of Philosophy*. London: Routledge. Retrieved November 11, 2003, from http://www.rep.routledge.com/article/U036SECT1.

ÇETİNOĞLU Ö., A Prolog Based Natural Language Processing Infrastructure for Turkish, Boğaziçi Üniversity, MSc. Thesis, 2001.

DEVLIN K., *Lecture 3: Introduction to Situation Theory*, ESSLLI – Helsinki, Finland, 2001.

DOWTY D.R. vd., *Introduction to Montague Semantics*, Kluwer Academic Publishers, Netherlands, 1992.

[11] Makaleye, indirildiği sanal kütüphanede "How to cite this article" kısmında belirtildiği biçimde referansta bulunulmuştur. Aynı yerden alınan diğer tüm makaleler için de belirtilen durum geçerlidir.

GAZDAR G. ve MELLISH C., *Natural Language Processing in PROLOG*, Addison – Wesley Publishing Company, Chatham – Kent, 1989.

HARRIS R., Landmarks in linguistic thought 1 : the Western tradition from Socrates to Saussure, London ; Routledge, New York, 1997.

HODGES, WILFRID (1998). *Model theory*. In E. Craig (Ed.), Routledge Encyclopedia of Philosophy. London: Routledge. Retrieved November 10, 2003, from http://www.rep.routledge.com/article/Y017.

HODGES W., Truth Definitions and Their Information Content, ESSLLI, Toronto, 2002.

KILIÇASLAN Y., *A form Meaning Interface for Turkish*, The University of Edinburgh, PhD Thesis, 1998.

KILIÇASLAN Y., TÜYSÜZ M.A.A., *Bilgisayar Ortamında Türkçe'nin Semantik Analizi*, Bilgi Teknolojileri Kongresi, 2002.

LÖBNER S., *Understanding Semantics,* Arnold Publishers, Bodmin – Cornwall, 2002.

MILLER A., *Philosophy of Language*, Routledge, Great Britain, 2002.

PERERIRA F. C. N., SHIEBER S. M., Prolog and Natural Language Analysis – Digital Edition, 1987.

PENROSE R., *Bilgisayar ve Zeka – Kralın Yeni Usu – I*, 9.baskı, TÜBİTAK Yayınları – Ankara, 2000.

PERRY, JOHN R. (1998). *Semantics, situation*. In E. Craig (Ed.), *Routledge Encyclopedia of Philosophy*. London: Routledge. Retrieved November 10, 2003, from http://www.rep.routledge.com/article/U041.

PERRY, JOHN R. (1998). *Semantics, possible worlds*. In E. Craig (Ed.), Routledge Encyclopedia of Philosophy. London: Routledge. Retrieved November 10, 2003, from http://www.rep.routledge.com/article/U039.

RESTALL G., *Notes on Situation Theory and Channel Theory*, Macquarie University – Sydney, Technical Report, TR-ARP-08-96, 1996.

RIVERA C. B., CERCONE N., *Hermes: Grammar and Lexicon,* Technical Report CS-98-02, 1998.

RUSSEL S. ve NORVIG P., *Artificial Intelligence: A Modern Approach,* Prentice Hall – New Jersey, 1995.

TALLERMAN M. , *Understanding Syntax,* Arnold Publishers, Bodmin – Cornwall, 1998.

TIN E., AKMAN V., *Computational Situation Theory*, SIGART Bulletin, 1994.

WEISS M. A., *Data Structures and Algorithm Analysis in C++*, Addison Wesley Longnan, USA, 1999.

YILDIRIM C., *Bilim Felsefesi*, 7.basım, Remzi Kitabevi – İstanbul, 2000.

ÖZGEÇMİŞ

Mehmet Ali Aksoy TÜYSÜZ 1979 yılında İstanbul'da doğdu. İlk ve orta öğrenimini İstanbul'da tamamladıktan sonra 1996 yılında girdiği Trakya Üniversitesi Mühendislik – Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü'nden 2000 yılında mezun oldu. Aynı yıl Trakya Üniversitesi Fen Bilimleri Enstitüsü'nde Yüksek Lisansa ve Araştırma Görevlisi olarak İstanbul Bilgi Üniversitesi'nde göreve başladı. Mehmet Ali Aksoy TÜYSÜZ halen İstanbul Bilgi Üniversitesi Bilgisayar Bilimleri Bölümü'nde araştırma görevlisi olarak çalışmaktadır.