Yapay Zeka

Ders 8

Doç. Dr. Mehmet Dinçer Erbaş
Bolu Abant İzzet Baysal Üniversitesi
Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü

Koşul Tatmin Problemleri

- Şu ana kadar gördüğümüz standart arama problemlerinde
  - Durumlar bir kapalı kutu gibidir.
  - Ardıl fonksiyonu, buluşsal fonksiyonu ve hedef testini destekleyen herhangi bir veri yapısı durumu tanımlayabilir.
- KTP (Koşul tatmin problemleri) farklı özelliklere sahiptir.
  - Durum, X_i değişkenleriyle tanımlanır ve her değişken D_i tanım kümesinden değer alır.
  - Hedef testi, bir koşul kümesinden oluşur ve her koşul değişkenlerin altkümelerinin alabileceği değer kombinasyonlarını tanımlar.
  - Bu tip tanımlar basit bir resmi tanımlama dili örneğidir.
  - Bu tip problemler üzerinde genel kullanıma uygun ve standart arama algoritmalarından daha verimli çalışan yöntemler geliştirilebilir.

Koşul tatmin problemleri

- Örnek: Harita renklendirme
  - Değişkenler: WA, NT, Q, NSW, V, SA, T
  - Tanım kümesi Di = {kırmızı, yeşil, mavi}
  - Koşullar: Komşu bölgeler farklı renge boyanmalı
    - Örnek: WA ≠ NT veya (WA, NT) içinde {(kırmızı, yeşil), (kırmızı, mavi), (yeşil, kırmızı), (yeşil, mavi), (mavi, kırmızı), (mavi,yeşil)}

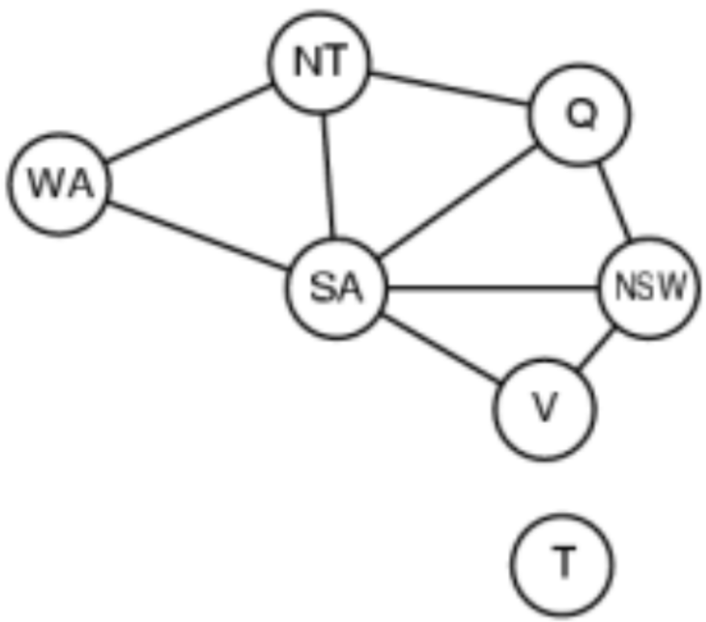
Koşul tatmin problemleri

- Örnek: harita renklendirme

- Çözümler: Bütün ve tutarlı eşleştirmeler.
  - Örneğin: WA = kırmızı, NT = yeşil, Q = kırmızı, NSW = yeşil, V = kırmızı, SA = mavi, T = yeşil

Koşul tatmin problemleri

- Örnek: Harita renklendirme
  - Koşul grafı



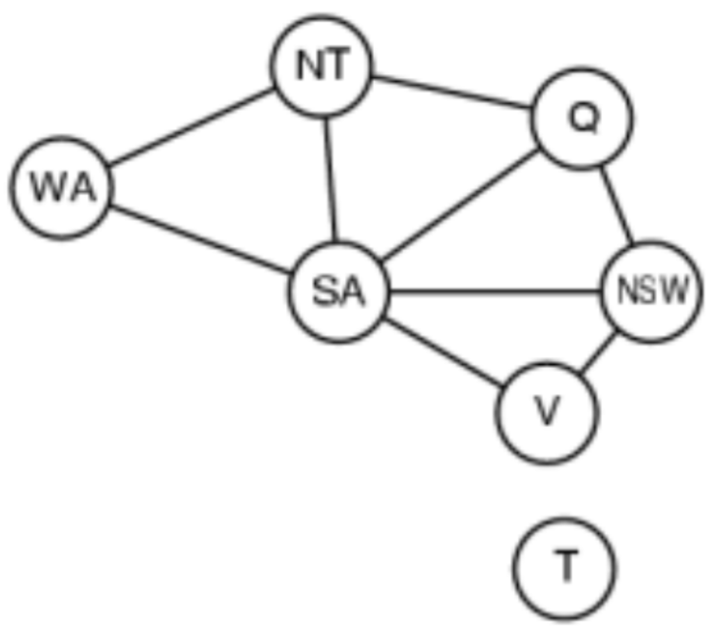
X={WA, NT, Q, SA, NSW, V, T}

C={SA≠WA, SA≠NT, SA≠Q, SA≠NSW, SA≠V, WA≠NT, NT≠Q, Q≠NSW, NSW≠V }

D = {kırmızı, mavi, yeşil}

Koşul tatmin problemleri

- Örnek: Harita renklendirme
  - İkili KTP: Her koşul iki değişkeni ilgilendiriyor.
  - Koşul grafı: düğümler değişkenleri, yaylar koşulları temsil eder.



Koşul tatmin problemleri

- Farklı KTP türleri
  - Kesikli değişkenler
    - Sınırlı tanım kümesi
      - N değişkenimiz var ve tanım kümesi d büyüklüğünde ise $O(d^n)$ tam eşleşme bulunur.
    - Sınırsız tanım kümesi
      - Bu tür problemlerde mümkün olan tüm değer kombinasyonlarını sıralandırarak koşulları tanımlamak mümkün değildir.
      - Örnek: İş planlama, değişkenler her işin başlangıç ve bitiş günü
      - Bu tür problemler için koşul dili tanımlanmalıdır
        - Örneğin $\text{Başlangıçİş}_1 + 5 \leq \text{Başlangıçİş}_2$
  - Devamlı değişkenler
    - Örnek: Hubble uzay teleskobunun gözlem başlangıç/bitiş zamanları
      - Deneyler için kesin tanımlı başlangıç/bitiş zamanı tanımlanmalı
      - Başlangıç/bitiş zamanları devamlı değişkenlerdir ve birçok astronomik, öncelik ve güç kullanımı koşuluna uymalıdır.

Koşul tatmin problemleri

- Farklı KTP türleri
  - Global koşul
    - Birçok farklı sayıda değişkeni etkileyen koşullara global koşul denir.
    - En çok karşılaşılan global koşula örnek HepsiFarkli koşuludur.
      - Bu koşul mevcut ise her bir değişken farklı değer almalıdır.
      - Örnek: Sudoku probleminde her satır ve her sütun için HepsiFarkli kuralı geçerlidir.
- Farklı koşul tipleri
  - Tekli koşullar sadece bir değişkeni etkiler
    - Örnek: SA ≠ yeşil
  - İkili koşul değişken çiftlerini etkiler
    - Örnek: SA ≠ WA
  - Yüksek düzey koşullar 3 veya daha fazla sayıda değişkeni etkiler.
    - Örnek: Y değişkeninin değeri X ile Z değişkenlerinin değerleri arasında olmalıdır.

Koşul tatmin problemleri

- Gerçek hayattan KTP örnekleri
  - Şu ana kadar gördüğümüz koşul örnekleri kesin koşullardır.
    - Bu koşulların tatmin edilmediği potansiyel bir çözüm devre dışı kalır.
  - Bir çok gerçek dünyadan KTP ise tercih koşulu içerir.
    - Bu koşullar tercih edilen çözümleri belirtir.
  - Örnek: Üniversite sınıf-planlaması problemi, “aynı hoca aynı anda iki derse giremez” kesin koşulunu içerebilir.
  - Ancak ayrıca, “Prof. T sabahları derse girmeyi tercih eder”, “Prof E öğleden sonraları derse girmeyi tercih eder” gibi tercih koşulu içerebilir.
  - Prof T’nin öğleden sonra ders vereceği bir ders planlaması kabul edilebilir ancak tercih edilmeyecektir.
  - Tercih koşulları genellikle tekil değişken atamaları için maliyet oluşturur şekilde tanımlanabilir.
    - Örnek: Prof T’nin öğleden sonra ders verdiği bir planlama, hedef fonksiyonu için ekstradan 2 maliyet puanına neden olurken, Prof T’nin sabah ders verdiği bir planlama 1 maliyete neden olabilir.
  - Bu tür problemlere ayrıca koşul optimizasyon problemi adı verilir.

Koşul tatmin problemleri

- Gerçek hayattan KTP örnekleri
  - Eşleme problemleri
    - Örnek: Hangi hoca hangi dersi verir.
  - Zaman planlama problemleri
    - Örnek: Hangi ders nerede ve ne zaman verilir?
  - Nakliye planlama
  - Fabrika üretim planlama

- Görüleceği üzere birçok gerçek hayattan problemde gerçek-zamanlı değere sahip değişkenler bulunmaktadır.

Koşul tatmin problemleri

- KTP nasıl çözülür?
  - Standart arama formulasyonu
    - Öncelikle direk çözüm arayan metotları inceleyeceğiz.
      - Daha sonra bu metotları geliştirmek için neler yapılabilir sorusunu cevaplayacağız.
  - Durumlar, şu ana kadar eşlenmiş değerler ile tanımlanır.
    - Başlangıç durumu: Eşleştirmelere boş küme { }
    - Ardıl fonksiyonu: henüz değer eşleştirilmemiş bir değişkene şu ana kadar yapılan eşleştirmelerle çakışmayan bir değer eşleştir.
      - Olası eşleştirme kalmamışsa çözümsüz demektir.
    - Hedef testi: Yapılan eşleştirme tamam mı?

Koşul tatmin problemleri

- KTP çözüm bulma
  - Normal durum uzayı aramalarında tek yapabilileceğimiz aramaktır.
  - KTP'de ise iki farklı işlem yapabiliriz.
    - Arama yapabiliriz: Yeni bir değişkene değer eşleme
    - Koşul yayılması (İng: Constraint propagation): koşulları kullanarak diğer değişkenlere eşlenebilecek değer sayısını azaltabiliriz.
  - Koşul yayılması, arama ile birlikte yapılabilir veya arama öncesi bir işlem olarak uygulanabilir.
  - Bu bölümde koşul yayılması yaparken kullanılan farklı metotları göreceğiz.

Koşul tatmin problemleri

- Düğüm tutarlılığı
  - Bir değişkenin tanım kümesindeki her değer, değişkenin tekli koşullarını tatmin ediyorsa, bu değişkene düğüm-tutarlı diyoruz.
  - Örnek: Harita renklendirme örneğimizde, SA bölgesi yeşil renk istemiyor olsun.
  - SA değişkeninin tanım kümesi başlangıçta {kırmızı, yeşil, mavi} olur.
  - Bu değişkeni düğüm-tutarlı yaparsak tanım kümesi {kırmızı, mavi} olacaktır.
  - Bir ağdaki bütün düğümler düğüm-tutarlı ise ağ düğüm-tutarlıdır denir.

Koşul tatmin problemleri

- Yay tutarlılığı
  - Bir değişkenin tanım kümesinde bütün değerler, bu değişkenin ikili koşullarını tatmin ediyorsa, değişken yay-tutarlıdır.
  - Daha resmi tanımlarsak; X_i ve X_j değişkenlerini ele aldığımızda, X_i değişkeninin şu anki tanım kümesi D_i içerisindeki her değer için X_j değişkeninin D_j tanım kümesinde (X_i, X_j) yayının tanımladığı koşulu sağlayan değerler var ise X_i değişkeni yay-tutarlıdır.
  - Örnek: $Y = X^2$ koşulunu düşünelim. X ve Y'nin tanım kümesi rakamlar kümesi olsun.
  - Bu koşulu açık bir şekilde $<(X,Y),\{(0,0),(1,1),(2,4),(3,9)\}>$ şeklinde yazabiliriz.
  - X'i, Y'ye göre yay-tutarlı yapabilmek için, X'in tanım kümesini {0,1,2,3} olacak şekilde azalttık.
  - Y'yi X'e göre yay-tutarlı yapmak istersek Y'nin tanım kümesini {0,1,4,9} olarak azaltmamız gerekir.
  - Böylece KTP'nin tamamı yay-tutarlı olur.

Koşul tatmin problemleri

- AC-3 algoritması yay tutarlılığı kullanır.

fonksiyon AC-3 (*ktp*) **dönüş** tutarsızlık bulunursa *false* aksi takdirde *true*
input: *ktp*, *(X, D, C)* komponentleri ile bir ikili KTP
yerel değişkenler: *sira*, yaylar sırası, başlangıçta ktp'nin bütün yayları
while *sira* boş değil ise **do**
(X_i, X_j) <== İLK-SİL(*sira*)
if KONTROLET(*ktp*, X_i, X_j) **then**
if sizeof D_i = 0 **then return** *false*
for each X_k **in** X_i.KOMSULAR – {X_j} **do**
EKLE(X_k, X_i) to *sira*
return *true*

function KONTROLET(*ktp*, X_i, X_j) **dönüş** *true* ancak ve ancak X_i'in tanım kümesini kontrol ettiysek
kontroledildi <== false
for each *x* **in** D_i **do**
if D_i içerisindeki hiçbir y değeri *(x,y)* çiftinin X_i ile X_j arasındaki koşulu tatmin etmiyorsa **then**
x değerini D_i'den sil
kontroledildi <== true
return *kontroledildi*

Koşul tatmin problemleri

- Yol tutarlılığı
  - Yay tutarlılığı tanım kümelerindeki olası elemanları (tekli koşul) yayları kullanarak (ikili koşul) azaltabilmemizi sağlar.
    - Avustralya haritasının renklendirme problemimizin çözümüne bir katkı sağlamaz.
    - Bu tür problemlerde ilerleme kaydedebilmemiz için daha kuvvetli bir koşul yaklaşımına ihtiyacımız var.
  - Yol tutarlılığı üç değişkeni etkileyen koşulları kullanarak ikili koşulları sıkılaştırır.
  - İki değişkenli $\{X_i, X_j\}$ kümesi üçüncü bir değişken olan X_m'e göre yol tutarlıdır; eğer $\{X_i, X_j\}$ koşuluna uygun her $\{X_i = a, X_j = b\}$ $\{X_i, X_m\}$ ve $\{X_m, X_j\}$ koşullarını tatmin eden bir X_m eşleştirmesi var ise.
    - Bu durum yol tutarlılığı olarak adlandırılır. Çünkü yapılan X_m'in ortada yer aldığı X_i'den X_j'ye bir yol bulunmasına benzer.

Koşul tatmin problemleri

- Sadece iki renk kullanarak (sadece kırmızı ve mavi) yandaki haritamızı renklendirebilir miyiz?
- Bu soruyu cevaplayabilmek için yol tutarlılığı kontrolü yapalım
- {WA, SA} yolunun NT'ye göre yol tutarlılığını kontrol edeceğiz.
- İki rengmiz olduğuna göre {WA, SA} için yapabileceğimiz eşleştirmeler {WA = kırmızı, SA = mavi} ve {WA = mavi, SA = kırmızı} olur.
- Her iki eşleştirme için de NT'ye eşleştirebileceğimiz bir renk kalmıyor.
- Öyleyse NT için geçerli bir eşleştirmemiz yok.
- Sonuç olarak diyebiliriz ki iki renk ile yandaki haritanın renklendirilmesi mümkün değildir.

Koşul tatmin problemleri

- KTP’ler için geri-dönüşlü arama
  - Bazı problemler koşullar üzerinde çıkarım yaparak çözülebilir.
  - Ancak bazı problemler için gördüğümüz metotlar ile çıkarım yapmak yeterli değildir.
  - KTP’leri çözmek için standart derinlik limitli arama kullanılabilir
    - Bir durum tamamlanmamış bir eşleştirmedir.
    - Aksiyonlar, eşleştirme yapılmamış bir değişkene var = değer şeklinde yeni bir eşleştirme yapmadır.
  - n tane değişkeni olan ve d dallanma faktörüne sahip bir KTP için en üst seviyede d*n tane farklı eşleştirme yapılabilir.
  - Bir sonraki seviyede (n-1)*d farklı eşleştirme yapılabilir.
  - Bu şekilde ağacın tamamını oluşturursak toplam yaprak sayısı $n!*d^n$ olacaktır.
    - Halbuki aslında toplam d^n yapılabilecek eşleştirme vardır.
  - Yukarıdaki formülasyonumuz KTP’lerin ortak bir özelliğini görmezden geliyor:
    - Sıra bağımsız olması (İng: Commutativity)

Koşul tatmin problemleri

- KTP'ler için geri-dönüşlü arama
  - Verilen aksiyon kümesindeki aksiyonların sıralaması problemin çözümü açısından bir farklılık göstermiyor ise bu tür problemlere sıra bağımsız problem diyoruz.
  - KTP'ler sıra bağımsızdır, çünkü eşleştirmelerin sıralamasından bağımsız olarak aynı eşleştirmeler ile aynı kısmi eşleştirme elde edilir.
    - Örnek: [WA = kırmızı sonra NT = yeşil] ile [NT = yeşil sonra WA = red] aynı sonucu verir.
  - Sıra bağımsızlık özelliğini kullanırsak, her düğümde sadece tek bir değişken için değer eşleştirmeyi değerlendirmeliyiz.
    - $b = d$ ve toplam b^d yaprak olur.
  - KTP'ler üzerinde sadece tek değişken eşleştirerek ilerleyen derinlik öncelikli arama geri-dönüşlü arama (ing: backtracking search).
  - Geri-dönüşlü arama KTP'ler üzerinde kullanılan bilgisiz arama metodudur.

Koşul tatmin problemleri

```
function GERI-DONUSLU-ARAMA (ktp) dönüş bir çözüm veya başarız
   return GERI-DONUS({ }, ktp)

function GERI-DONUS (eşleme, ktp) dönüş bir çözüm veya başarısız
   if eşleme tamamlanmış ise then return eşleme
   var <== EŞLEME-YAPILMAMIŞ-DEĞİŞKEN-SEÇ(ktp)
   for each değer in TANIM-KÜMESİNDEKİ-DEĞERLERİ-SIRALA(var,eşleme,ktp) do
      if değer eşleme ile tutarlı ise then
         ekle {var = değer} to eşleme
         çıkarımlar <== ÇIKARIM(ktp,var,değer)
         if çıkarımlar ≠ başarısız then
            ekle çıkarımlar to eşleme
            sonuç <==GERI-DONUS(eşleme,ktp)
            if sonuç ≠ başarısız then
               return sonuç
      sil eşleme'den {var = value} ve çıkarımlar
   return başarısız
```

Koşul tatmin problemleri

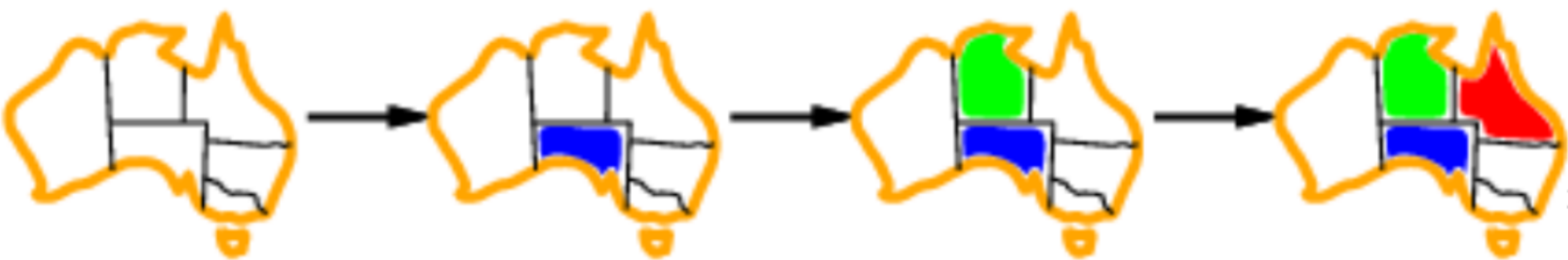
- Önceki bölümde bilgisiz arama metotlarını geliştirebilmek için problem ile ilgili bilgileri kullanmıştık.
- Ktp’lerde ise problem alakalı bilgiye ihtiyaç duymadan çözüme daha hızlı ulaşabiliriz.
- Bunun için aşağıdaji sorulara cevap verebilmemiz gerekir
  - Bir sonraki aşamada hangi değişkeni seçmeliyiz (EŞLEME-YAPILMAMIŞ-DEĞİŞKEN-SEÇ) ve değerleri hangi sıra ile denemeliyiz (TANIM-KÜMESİNDEKİ-DEĞERLERİ-SIRALA)?
  - Her adımda hangi tür çıkarımlar yapılmalıdır (ÇIKARIM)?
  - Arama sırasında bir koşulu ihlal eden eşleştirme yapıldığında, bu hatanın tekrarlanması arama tarafından engellenebilir mi?

Koşul tatmin problemleri

- Değişken ve değer sıralama
  - Geri-dönüş algoritması aşağıdaki satıra sahiptir
    - *var* <== EŞLEME-YAPILMAMIŞ-DEĞİŞKEN-SEÇ(*ktp*)
  - En basit seçim metodu eşleme yapılmamış bir sonraki değişkeni seçmektir.
  - Ancak bu statik değişken sıralama metodu çok seyrek en verimli arama sağlar.
  - Haritamıza geri dönelim
  - WA = kırmızı, NT = yeşil
  - SA için tek seçenek kaldı.
  - SA seçildikten sonra gerisi zaten belli
  - En az seçenek şansı kalmış değişkeni seçmeye minimum geri-kalan değerler metodu (ing: minimum remaining values) denir.
    - Bu metot ayrıca en kısıtlı değişken (ing: most constrained variable) olarak bilinir.

Koşul tatmin problemleri

- Değişken ve değer sıralama
  - Minimum geri-kalan değerler (mgd) metodu ilk eşleme yapılacak değişkenin seçilmesi sırasında bir katkı sağlamaz.
  - İlk değişken seçilirken derece buluşsalı (ing degree heuristic) işe yarar.
  - Derece buluşsalı kullanılarak diğer değişkenler üzerinde en fazla sayıda koşula sahip olan değişken seçilir.
    - Böylece dallanma faktörü azaltılmaya çalışılır.



Koşul tatmin problemleri

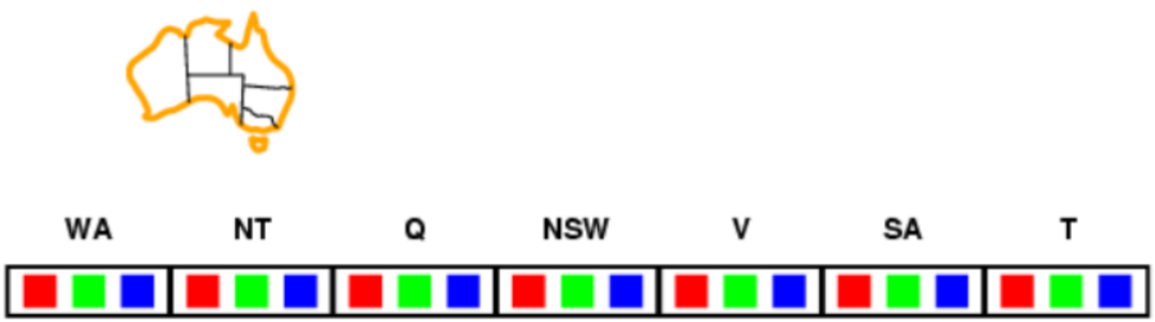
- Değişken ve değer sıralama
  - Değişken seçildikten sonra verilecek değer seçilmelidir.
  - Bu amaçla en-az kısıtlayıcı değer kullanılabilir.
    - Bir değişken için, geri kalan değişkenler için en az sayıda seçeneği eleyen değer seçilir.

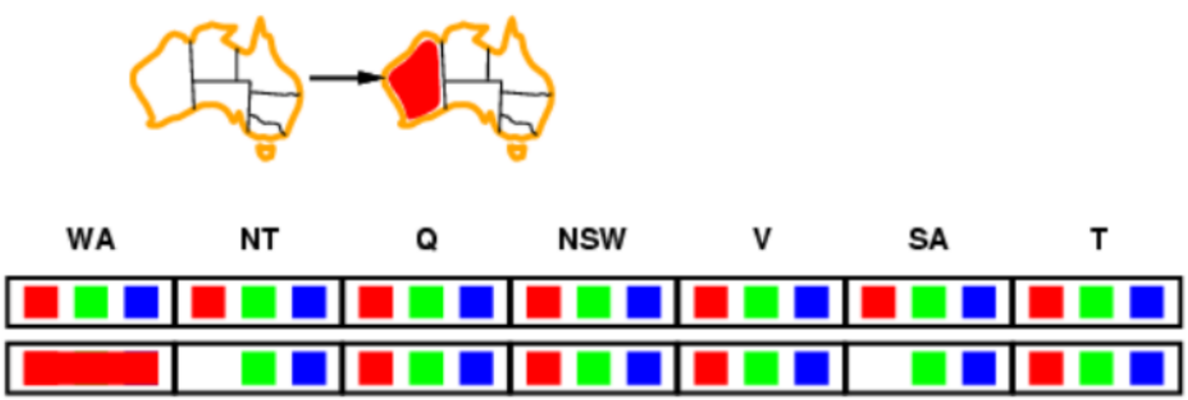
Koşul tatmin problemleri

- Arama ve çıkarımı birlikte yapma
  - Bu bölümde gördüğümüz algoritmalar aramaya başlamadan önce değişkenlerin sahip olabilecekleri değerleri azaltabiliyor.
  - Çıkarım bunun dışında başka yararlar için kullanılabilir.
    - Her değişken için eşleme yaptığımızda, çıkarım yaparak aramanın geri kalanını verimli hale getirebiliriz.
  - Bu yöntemlerden biri ileri kontroldür (ing: forward checking)
  - Bir X değişkenine değer eşlendiğinde, X değişkenine bir koşul ile bağlı olan Y değişkenin değer kümesinden, X değişkenine eşlenen değer ile çelişen değerleri silebiliriz.
  - Bir başka deyişle
    - Eşleşme yapılmamış değişkenler için kalan eşlenebilir değerleri takip et
    - Herhangi bir zamanda bir değişken için eşlenebilir değer kalmaz ise aramayı sonlandır.

Koşul tatmin problemleri

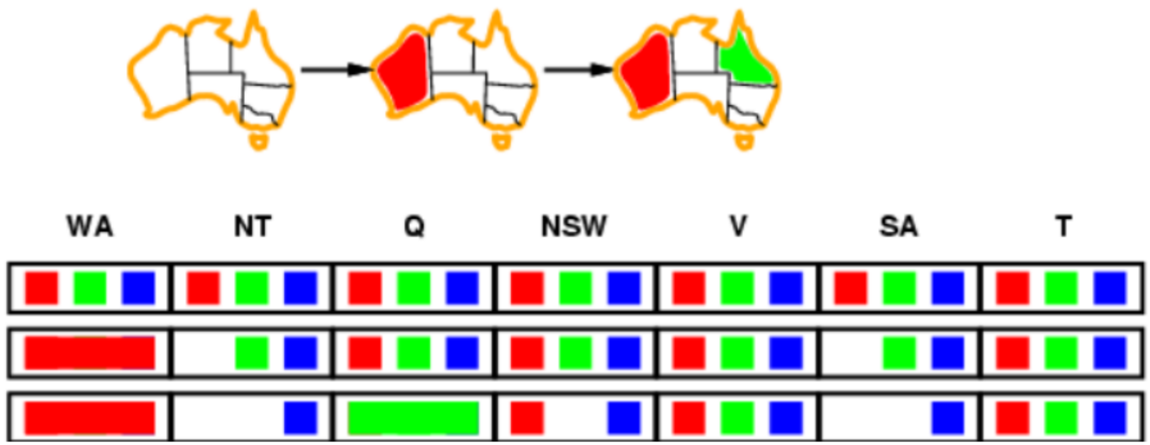
- İleri kontrol

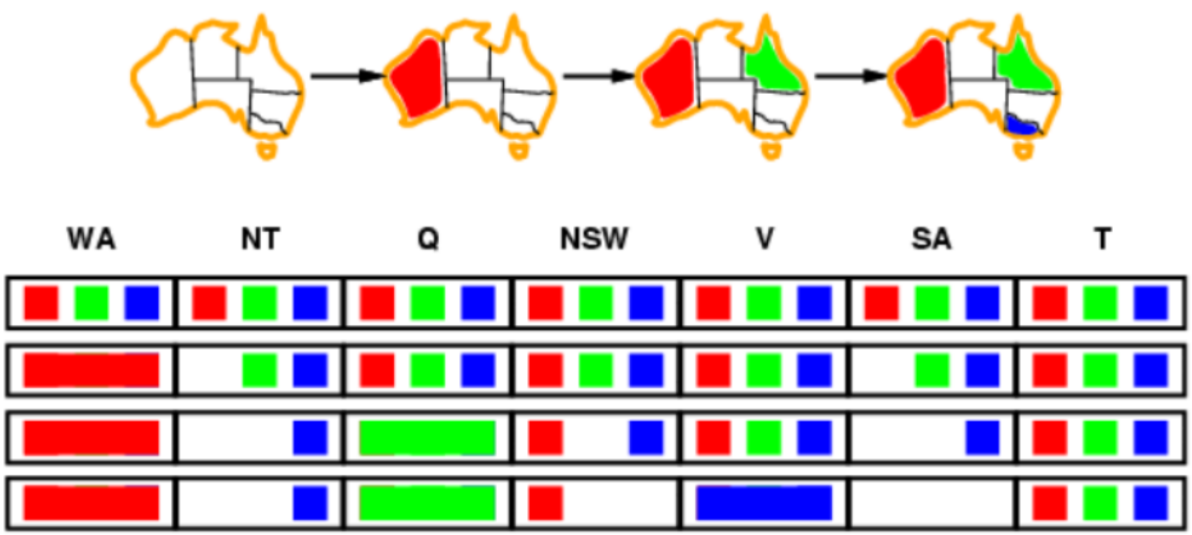




Koşul tatmin problemleri

- İleri kontrol



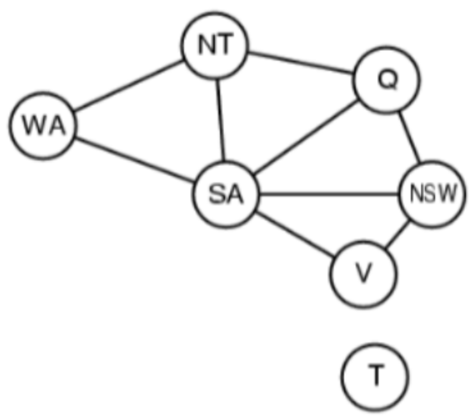


Koşul tatmin problemleri

- Akıllı geri-dönüş
  - Geriye bakış
  - Arama başarısız olduğunda geri dönüş yapılmalıdır.
  - Bu noktada sorulması gereken soru nereye dönülmelidir.
    - Geri dönüş yapılan değişkene farklı bir değer verilecektir.
  - En basit yöntem en son değer eşlenen değişkeni seçmektir.
    - Bu yöntem kronolojik geri-dönüş adı verilir.
  - Bu bölümde başka yöntemleri göreceğiz.

Koşul tatmin problemleri

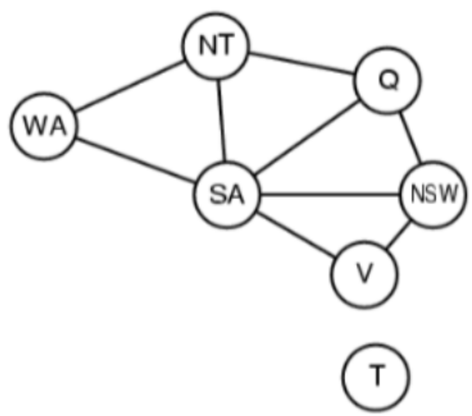
- Akıllı geri-dönüş



  - Belirli bir sıralama ile kronolojik geri dönüş yaptığımız varsayalım.
  - Sıralamamız Q, NSW, V, T, SA, WA, NT olsun.
  - Farzedelim şu kısmi eşleştirmeyi yapalım: Q = kırmızı, NSW = yeşil, V = mavi, T = kırmızı}
    - Bir sonraki değişken olan SA için eşleme yapacağımız değer yok.
    - Bir öncekine geçsek, T düğümüne başka bir değer eşlememiz bir işe yaramayacaktır.

Koşul tatmin problemleri

- Akıllı geri-dönüş



Daha akıllıca yöntem sorunu çözecek olan bir değişkene dönüş yapmaktır.

- SA değişkenine değer eşlenememesine sebep olan değişkenler seçilmeli.
- Örneğimiz için ({Q = kırmızı}, {NSW = yeşil}, {V = mavi}) sorun oluşturan eşleştirmelerdir.
- Bu sebeple T düğümünün üzerinden atlayıp V düğümünün değeri değiştirilerek geri dönüş yapılmalıdır.

Koşul tatmin problemleri

- İleri kontrol
  - İleri kontrol ileriye yeteri kadar bakamadığı için örneğimizdeki tutarsızlık oluşmuştur.
  - Bu durumu çözmek için Yay tutarlılığı sağlama (İng: Maintaining arc consistency) algoritması kullanılabilir.
    - Bir X_i değişkenine değer atandığında ÇIKARIM fonksiyonu AC-3 algoritması, tüm yaylar yerine, sadece (X_i, X_j) şeklinde olan yaylar ile başlatılır.
    - AC-3 algoritması daha önce gösterdiğimiz gibi çalışır ve tanım kümesi boş küme olan bir değişken bulunduğunda geri-dönüş başlatılır.
    - YTS algoritması yinemeli olarak tüm yayları kontrol ettiği için ileri kontrol yönteminden daha etkilidir.