

BİL-113 Bilgisayar Programlama - I

Ödev - 4

Veriliş Tarihi: 22.11.2010

Teslim Tarihi: 2.12.2010 (Saat : 23:59)

Teslim Şekli: AdSoyadOdev4.rar (veya zip) dosyasına ödevinizi paketleyip, ocsert@etu.edu.tr adresine konusu tam olarak "[bil113] odev4" olacak şekilde e-posta'ya dosyayı ekleyerek gönderiniz.

Kurallar: Geç gönderilen ve teslim şekline uymayan ödevler kabul edilmez. Kopya kesinlikle yasaktır, kopya veren ve alan öğrenciler ödevden 0 ahırlar ve ayrıca üniversite disiplin yönetmeliği kuralları bu öğrencilere uygulanır.

Bu ödevde yapay hayat (artificial life) simülatörlerine bir örnek olan Conway'in hayat oyunu (Conway's Game of Life) adlı simülatörü biraz genişleterek yazacaksınız. Bu simülatör, çok basit kurallardan oldukça kompleks çıktılar oluşabildiğini göstermesi açısından yapay hayat araştırmaları için önemlidir. Detaylı bilgi için wikipedia sayfasına bakabilirsiniz.

Oyun, (yine) iki boyutlu bir grid'de geçiyor. Grid'in (x,y) koordinatları ile belirttiğimiz her bir kısmına bu ödevde hücre diyeceğiz. Her hücrenin çevresinde 8 adet komşu hücre bulunuyor. Grid'in herhangi (x,y) hücresinin t anında iki durumu (state) var: canlı veya ölü. t+1 anında (x,y) hücresinin durumu aşağıdaki kurallarla belirleniyor. (Tür'ün ne olduğunu öğrenmek için okumaya devam edin.)

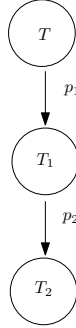
- Herhangi bir canlı hücre, komşularında ikiden az canlı hücre kalmışsa, ölür.
- Herhangi bir canlı hücre, komşularında iki veya üç canlı hücre varsa yaşamaya devam eder.
- Herhangi bir canlı hücre, komşularında üçten fazla canlı hücre varsa, ölür.
- Herhangi bir ölü hücre, komşularında sadece ve tam olarak üç adet aynı türden canlı hücre varsa canlı hale gelir. (Bu kuralın detayları aşağıda).

Bu kurallar oldukça basit gözükse de verilen başlangıç durumunun ne şekilde sonlanacağını tahmin kolay değildir (Bkz. wikipedia sayfası). Buraya kadar anlatılanlar orjinal oyun tanımını dahilindeki tanımlar.

Bu simülasyonu çok basit bir ekosistem simülasyonu şeklinde şu şekilde genişletebiliriz. Yukarıda anlatılan oyunda her bir hücre tek bir tipte. Bu tek tipi temel canlı olarak düşünelim ve T türünde olduğunu varsayalım. Yukarıdaki son kuralda hücrenin canlı hale gelmesi kuralını aşağıdaki şekilde detaylandıralım.

- p_i olasılıkla ortaya çıkan canlı başka bir türde olacak.

T haricinde T_1 ve T_2 adında 2 tane daha tür var. Bunlar arasındaki hiyerarşi aşağıdaki şekilde:



Şekildeki p_1 ve p_2 , sırasıyla T_1 ve T_2 'nin oluşma olasılıklarını gösteriyor ve T_2 T_1 'den, T_1 ise T 'den oluşabiliyor. T_1 ve T_2 için yukarıda belirtilen 4 temel kural da geçerli. T_1 ve T_2 'nin bunlara ek olarak bazı özellikleri var:

T_1 : Her adımda aşağıdaki olasılığa göre T_1 ölür:

$$\frac{1}{\exp(5 - y)}$$

burada y T_1 'in yaşımı gösteriyor. (Dolayısıyla T_1 'in belli bir ömrü var.)

T_2 : Her adımda aşağıdaki olasılığa göre T_2 ölür:

$$\frac{1}{\exp(N_T)}$$

burada N_T , T_2 'nin komşuları arasındaki T türündeki canlı hücre sayısını gösteriyor. (Dolayısıyla T_2 'nin hayatta kalabilmek için T 'ye ihtiyacı var.)

Ekranaya çizmek için gerekli sınıfı yine biz sağlayacağız. Sınıfta, grid oluşturmak için *Grid(int x,int y)* şeklinde bir constructor ve (x,y) koordinatındaki hücreyi canlı hale getirmek için *alive(int x, int y,int tur)* ve ölü hale getirmek için de *dead(int x,int y)* şeklinde iki adet metod bulunuyor. *alive* metodundaki tur parametresi hücrenin hangi türe dönüşeceğini gösteriyor. Her adımdan sonra değişiklikler ekrana yansıtılmalıdır.

Ekranaya çizdiğiniz simülasyonu görünür kılmak için her adımdan sonra tam olarak 0.5 saniye programı durdurunuz. Bunun için *Thread.sleep()* metodunu kullanabilirsiniz. Tam olarak 0.5 saniye durmasına dikkat edin. Bu metodun detayları için Java dokümantasyonuna bakabilirsiniz.

Ödevde yazdığımız kodun sağlaması gerekenler, bütün türler için ayrı birer sınıf olması ve bu sınıfların yukarıdaki hiyerarşiye göre birbirlerinden türetilmiş olması. Bir de ilk çalıştığında kullanıcıdan aşağıdaki gibi bilgileri alması gerekiyor. Sondaki adım sayısı simülasyonun kaç adım süreceğini gösteriyor. Verilen adım sayısı kadar çalıştırılıp program kapatılabilir. En son satırda da başlangıç durumu yer alıyor. Bütün gridin tek satırda olacağını varsayabilirsiniz. Burada hangi koordinatta hangi tipte hücrenin bulunduğu belirtiliyor.

```
Grid büyüklüğü : 20 20
p1 ve p2 : 0.4 0.2
Adım sayısı : 100
Başlangıç durumu : (5,5) t, (6,5) t, (7,5) t, (10,10) t1
```