UNIVERSIDAD
DE GRANADA

MODELOS DE MARKOV OCULTOS Y APLICACIONES A LA BIOLOGÍA

XUSHENG ZHENG

Trabajo Fin de Grado

Doble Grado en Ingeniería Informática y Matemáticas

Tutores

Lidia Fernández Rodríguez

FACULTAD DE CIENCIAS

E.T.S. INGENIERÍAS INFORMÁTICA Y DE TELECOMUNICACIÓN

Granada, a 2 de noviembre de 2022

ÍNDICE GENERAL

RESUMEN

Nos basamos en el trabajo desarrollado en [Tur36].

Occaecati expedita cumque est. Aut odit vel nobis praesentium dolorem sed eligendi. Inventore molestiae delectus voluptatibus consequatur. Et cumque quia recusandae fugiat earum repellat porro. Earum et tempora vel voluptas. At sed animi qui hic eaque velit.

Saepe deleniti aut voluptatem libero dolores illum iusto iusto. Explicabo dolor quia id enim molestiae praesentium sit. Odit enim doloribus aut assumenda recusandae. Eligendi officia nihil itaque. Quas fugiat aliquid qui est.

Quis amet sint enim. Voluptatem optio quia voluptatem. Perspiciatis molestiae ut laboriosam repudiandae nihil.

Parte I

PARTE DE MATEMÁTICAS

Esta es una descripción de la parte de matemáticas. Nótese que debe escribirse antes del título

1

INTRODUCCIÓN A LAS CADENAS DE MARKOV

En este capítulo vamos a inicializar la teoría de cadenas de Markov, avanzado progresivamente hacia las cadenas de Markov ocultas. En primer lugar, vamos a presentar el concepto de procesos de Markov:

1.1 PROPIEDAD DE MARKOV

Sea $\mathbb{N}$ un conjunto finito de forma $\{x_1, ..., x_n\}$, definimos un proceso estocástico sobre $\mathbb{N}$ como una secuencia de variables aleatorias $\{\mathcal{X}_0, \mathcal{X}_1, \mathcal{X}_2, ...\}$ o $\{\mathcal{X}_t\}_{t=0}^{\infty}$ para acortar, donde cada $\mathcal{X}_t$ es una variable aleatoria que toma valores en $\mathbb{N}$.

A pesar de que el índice t puede representar cualquiera magnitud, lo más común es que represente el tiempo. Si tomamos el índice t como el tiempo, obtenemos la noción de "pasado" y "futuro", esto es, si $t < t'$, entonces $\mathcal{X}_t$ es una variable "pasada" para $\mathcal{X}_{t'}$, mientras que $\mathcal{X}_{t'}$ es una variable "futura" para $\mathcal{X}_t$. Sin embargo, esto no es siempre así, por ejemplo, si el proceso estocástico corresponde al de las secuencias de genomas de un organismo, el conjunto $\mathbb{N}$ estará formado por los cuatro símbolos para las subunidades de nucleótidos $\{A, C, G, T\}$ y las secuenciaciones tienen un significado más espacial que temporal.

Definición 1.1. Un proceso $\{\mathcal{X}_t\}_{t=0}^{\infty}$ se dice que posee **la propiedad de Markov**, o es un **proceso de Markov**, si para toda $t \geq 1$ y $(u_0, ..., u_{t-1}, u_t) \in \mathbb{N}^{t+1}$ se tiene que:

$$P[\mathcal{X}_t = u_t | \mathcal{X}_0 = u_0, ..., \mathcal{X}_{t-1} = u_{t-1}] = P[\mathcal{X}_t = u_t | \mathcal{X}_{t-1} = u_{t-1}] \tag{1.1}$$

Parte II

PARTE DE INFORMÁTICA

2

SECCIÓN TERCERA

El siguiente código es un ejemplo de coloreado de sintaxis e inclusión directa de código fuente en el texto usando `minted`.

```
-- From the GHC.Base library.
class  Functor f  where
    fmap        :: (a -> b) -> f a -> f b

    -- | Replace all locations in the input with the same value.
    -- The default definition is @'fmap' . 'const'@, but this may be
    -- overridden with a more efficient version.
    (<$)        :: a -> f b -> f a
    (<$)        =  fmap . const

-- | A variant of '<*>' with the arguments reversed.
(<**>) :: Applicative f => f a -> f (a -> b) -> f b
(<**>) = liftA2 (\a f -> f a)
-- Don't use \$ here, see the note at the top of the page

-- | Lift a function to actions.
-- This function may be used as a value for `fmap` in a `Functor` instance.
liftA :: Applicative f => (a -> b) -> f a -> f b
liftA f a = pure f <*> a
-- Caution: since this may be used for `fmap`, we can't use the obvious
-- definition of liftA = fmap.

-- | Lift a ternary function to actions.
liftA3 :: Applicative f => (a -> b -> c -> d) -> f a -> f b -> f c -> f d
liftA3 f a b c = liftA2 f a b <*> c


{-# INLINABLE liftA #-}
{-# SPECIALISE liftA :: (a1->r) -> IO a1 -> IO r #-}
{-# SPECIALISE liftA :: (a1->r) -> Maybe a1 -> Maybe r #-}
{-# INLINABLE liftA3 #-}
```

```
{-# SPECIALISE liftA3 :: (a1->a2->a3->r) -> IO a1 -> IO a2 -> IO a3 -> IO r #-}
{-# SPECIALISE liftA3 :: (a1->a2->a3->r) ->
                                Maybe a1 -> Maybe a2 -> Maybe a3 -> Maybe r #-}

-- | The 'join' function is the conventional monad join operator. It
-- is used to remove one level of monadic structure, projecting its
-- bound argument into the outer level.
join              :: (Monad m) => m (m a) -> m a
join x            =  x >>= id
```

Vivamus fringilla egestas nulla ac lobortis. Etiam viverra est risus, in fermentum nibh euismod quis. Vivamus suscipit arcu sed quam dictum suscipit. Maecenas pulvinar massa pulvinar fermentum pellentesque. Morbi eleifend nec velit ut suscipit. Nam vitae vestibulum dui, vel mollis dolor. Integer quis nibh sapien.

BIBLIOGRAFÍA

[Tur36] Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.