Ejercicios Resueltos de Demostración

Técnicas de Diseños de Algoritmos – 1^{er} Cuatri 2025

Ejercicio 1:

Sea `f` la función definida recursivamente de la siguiente manera:

```
def f(n: int, p: list[int]):
  if len(p) == n:
    if is_permutation(p):
      return 1
    else:
      return 0
  res = 0
  for i in [1, 2, ..., n]:
    res += f(n, p + [i])
  return res
```

Donde `is_permutation` es una función que devuelve true si y solo si su parametro de entrada es una permutación.

Demostrar que `f(n, [])` devuelve la cantidad de permutaciones de tamaño n.

Demostración

Primero demostremos que dado un entero n y un arreglo p de longitud menor o igual a n, entonces `f(n, p)` devuelve la cantidad de permutaciones de longitud n que tienen como prefijo p. Procedamos por inducción en la longitud del arreglo p, con caso base con $|p| = n$ y procediendo en orden descendiente.

Para demostrar el caso base, hay que ver que `f(n, p)` devuelve la cantidad de permutaciones de longitud n con prefijo p. Como $|p| = n$ (porque estamos en el caso base), la unica permutacion que puede tener prefijo p es p en sí, entonces la cantidad de permutaciones con prefijo p será 1 si p en sí es una permutación, y 0 sino, que es exactamente lo que computa la función.

Para ver el caso inductivo, tomemos $|p| < n$. Para extender p a una permutación de longitud n hay que agregarle al menos un entero, entonces toda permutación de longitud n con prefijo p tiene como prefijo a $p + [i]$ para algún i entero. Además i estará en el rango $[1, \ldots, n]$ porque todos los elementos de una permutación de longitud n están en ese rango. Luego la cantidad total de permutaciones con prefijo p será igual a la cantidad total de permutaciones con prefijo $p + [i]$, sumados sobre todos los i en el rango $[1, \ldots, n]$.

Por hipótesis inductiva, `f(n, p + [i])` calcula la cantidad de permutaciones de longitud n con prefijo $p+[i]$, pues $|p + [i]| > |p|$ (HI) y $|p + [i]| \leq n$. Entonces el loop final de la función calcula exactamente la suma que queríamos. Fin de la inducción.

Ahora lo que falta ver es que $f(n, [])$ devuelve lo que queremos. Recién demostramos que devuelve la cantidad de permutaciones de longitud n con prefijo `[]`, y como por definición el array vacío es prefijo de todo, esto es igual a la cantidad total de permutaciones de longitud n.

Ejercicio 2:

Sea `A` un arreglo de tamaño $N \times M$ de enteros. Consideremos la siguiente función recursiva:

```
def f(i, j):
  if i <= 0 or j <= 0:
    return inf
  if i == 1 and j == 1:
    return A[i][j]
  return min(A[i][j]+f(i, j-1), A[i][j]+f(i-1, j))
```

Una sucesión de casillas es un camino directo si comienza en $(1,1)$, termina en (N,M) y solo se mueve hacia la derecha o hacia abajo. Demostrar que $f(N,M)$ devuelve la suma de las casillas del camino directo que menor suma tenga.

Demostración

Demostremos por inducción que si $i \leq N$ y $j \leq M$ entonces $f(i,j)$ devuelve la menor suma de las casillas mirando los caminos que comienzan en $(1,1)$ y terminan en (i,j), y solo van hacia la derecha o hacia abajo.

Primero veamos el caso $i \leq 0$ o $j \leq 0$. En estos casos, (i,j) está fuera del tablero, por lo que no existe ningún camino que comience en $(1,1)$ y termine en (i,j), entonces la mínima suma de casillas será ∞, porque ∞ es el elemento neutro del mínimo.

Para el resto de los valores de i y j (es decir, cuando $i \geq 1$ y $j \geq 1$), procedamos por inducción en $i+j$. El caso base es cuando $i+j=2$, que es el minimo valor posible que puede tener esa suma y se alcanza con $i=j=1$. En este caso el único camino posible que empieza y termina en $(1,1)$ es el que contiene solo la casilla $(1,1)$, entonces la suma de sus casillas será `A[i][j]`, que es exactamente lo que devuelve la función.

Ahora veamos el caso recursivo, cuando $i+j>2$. Entonces el camino no será una sola casilla. Sabemos que todos los caminos a considerar tienen como última casilla (i,j). Cuáles son los valores posibles para la anteúltima casilla? Como los caminos siempre van hacia abajo o hacia la derecha la anteúltima casilla solo puede ser $(i,j-1)$ (si el ultimo movimiento fue hacia la derecha) o $(i-1,j)$ (si el último movimiento fue hacia abajo).

Notemos que si la anteúltima casilla de un camino es $(i,j-1)$, entonces el camino que va a (i,j) se compone primero de un camino que va de $(1,1)$ a $(i,j-1)$ más una casilla adicional que es la (i,j). Entonces el óptimo hasta (i,j), si asumimos que pasa por $(i,j-1)$, se va a componer del camino optimo hasta $(i,j-1)$ unión la casilla (i,j). La suma de ese camino óptimo hasta $(i,j-1)$ es, por hipótesis inductiva, $f(i,j-1)$, entonces la suma del camino total es $A[i][j]+f(i,j-1)$.

Análogamente, para el caso donde pasa por $(i-1,j)$ el camino óptimo hasta (i,j) va a ser $A[i][j]+f(i-1,j)$. Como solo hay esas dos posibilidades, el óptimo será $\min(A[i][j]+f(i-1,j), A[i][j]+f(i,j-1))$, que es exactamente lo que devuelve la función. Fin de la demo recursiva.

Entonces poniendo $i=N$ y $j=M$ en lo que demostramos, sabemos que $f(N,M)$ devuelve la suma del camino óptimo hasta (N,M).