

## Práctica 2. Programación dinámica

Antonio Roldan Andrade  
antonio.roldanandrade@alum.uca.es  
Teléfono: +34611404497  
NIF: 49562495W

19 de mayo de 2023

1. Formalice a continuación y describa la función que asigna un determinado valor a cada uno de los tipos de defensas.

$$f(\text{daño}, \text{alcance}, \text{salud}, \text{coste})(\text{daño} + \text{salud} + \text{alcance}) - \text{coste} =$$

La función tomará como criterio para la valoración de las defensas en función a la suma de sus respectivos daño, alcance y salud, a esto se le sustraerá el valor del coste en ases de las mismas. Para la realización de la misma, he visto conveniente crear una estructura para el almacenado de la valoración que se le da y a la defensa en sí, además, he sobrecargado el operador `+` para poder usar la función `sort` de la lista de la STL de C++.

El algoritmo en sí es trivial, simplemente irá recorriendo la lista de defensas y les dará una valoración, la cual se almacena en una lista de elementos de `defensavaloracion` (struct anteriormente mencionada), tras esto se devolverá la lista.

2. Describa la estructura o estructuras necesarias para representar la tabla de subproblemas resueltos.

La estructura necesaria para la representación de la Tabla de subproblemas Resueltos será creada con una estructura llamada TSP. Los elementos que compondrán a la estructura serán: 1) Una matriz representada con una variable de tipo vector (de alto nivel) que a su vez tendrá un vector de enteros dentro, de la siguiente forma, `std::vector<std::vector<int>>>` m. Esto nos permite tener los métodos y funciones de la clase vector y además tener una mayor sencillez a la hora de reservar memoria y demás.

2) Una lista donde están almacenadas una struct `defcost` (que almacena las defensas con sus respectivos costes), aun no siendo necesaria por ser una variable de una defensa, esta nos permite tener una ordenación de menor coste, al cambio de sacrificar un poco de espacio en memoria.

3) Constructores de la struct TSP y struct `defcost`

3. En base a los dos ejercicios anteriores, diseñe un algoritmo que determine el máximo beneficio posible a obtener dada una combinación de defensas y *ases* disponibles. Muestre a continuación el código relevante.

```
// Recibe la struct tsp (que tiene la lista de defensas a colocar etc) y el maximo numero de ases
// Se asume que la TSP ya se ha creado e inicializado con exito
int max_beneficio(TSP& tsp, const int& max_ases) {
    return tsp.matriz_tsp[tsp.def_val_list.size() - 1][max_ases]; // devolvemos el ultimo
    elemento de la matriz que contiene el max valor
}
```

4. Diseñe un algoritmo que recupere la combinación óptima de defensas a partir del contenido de la tabla de subproblemas resueltos. Muestre a continuación el código relevante.

```
std::list<Defense*> recupera_defensas(const std::vector<std::vector<int>>>& tsp,
const std::list<defensa_valoracion>& def_val, const int& filas, const int& cols, std::list<
Defense*> defenses) {
```

```

//Almacenara las defensas que se han usado
std::list<Defense*>sol;

int i = filas - 2;
int j = cols - (*defenses.begin())->cost; //Tenemos que eliminar el coste de la primera
defense (centro de extraccion), que ademas se a ade en cualquier caso

List<Defense*>::iterator it = defenses.end();

it--; //Iniciamos en la posicion anterior a la ultima defensa (seria fin - 1)

//Se Recorrera inversamente ya que partimos del beneficio maximo que se encuentra en
//la ultima posicion de la matriz
while(i > 0)
{
    if(tsp[i][j] != tsp[i-1][j])
    {
        j = j - (*it)->cost;
        sol.push_back(*it);
    }
    i--;
    it--;
}
if(tsp[0][j] != 0) // En caso de que la primera posicion de la fila 0 sea != de 0
significa que la existe una defensa que tiene coste 0 y entrara en la lista
sol.push_back(*it); //Insertamos dicha defensa

return sol; //Devolvemos la lista de defensas
}

```

Todo el material incluido en esta memoria y en los ficheros asociados es de mi autoría o ha sido facilitado por los profesores de la asignatura. Haciendo entrega de este documento confirmo que he leído la normativa de la asignatura, incluido el punto que respecta al uso de material no original.