Apuntes de Algorítmica

Divide y vencerás:

Requisitos para poder aplicar DyV:

- El caso del problema **debe poder dividirse** en uno o más casos equivalentes de tamaño menor, **independientes entre sí**, que puedan **resolverse por separado**.
- Las soluciones a los casos de tamaño menor **deben poder combinarse entre sí** para poder dar lugar a la solución del caso inicial.
- Debe existir, a priori, un **método básico** que resuelva el problema o, en su defecto, **un caso base indivisible** donde el problema este resuelto.

Plantilla de algoritmo DyV:

Función S= DyV(P,n)
    Si "P es suficientemente pequeño", **entonces:**
      Calcular Solucion= BASICO(P,n)
    En otro caso, hacer:
      Dividir P en k subcasos más pequeños P1, P2,..., Pk, con tamaños lo más similares posible n1, n2,..,nk.
      Para cada subcaso generado i=1..k, **hacer:**
          SubSolucion i= DyV(Pi, ni)
      Calcular S= Combinar(SubSolucion1, SubSolucion2,...,SubSolucionk)
    Fin-En otro caso
    Devolver S

Algoritmos Voraces (Greedy)

Características:

- Construir la solución **por etapas**.
- En cada momento **selecciona un movimiento** de acuerdo con un **criterio de selección**.
- **No** vuelve a c**onsiderar** los **movimientos ya seleccionados** ni los modifica en posteriores etapas.
- Se necesita una **función objetivo** o **criterio de optimalidad**.

Pasos y requisitos para aplicar Greedy:

- Diseñar una **lista de candidatos** para formar la solución.
- Identificar una **lista de candidatos ya utilizados**.
- Diseñar una **función solución** para saber cuándo un conjunto de candidatos es solución al problema.
- Diseñar un **criterio de factibilidad** (cuándo un conjunto de candidatos puede ampliarse para formar la solución final).
- Diseñar una **función de selección** del candidato más prometedor para formar parte de la solución.
- Existe una **función objetivo** de minimización/maximización.

Plantilla de algoritmos Greedy:

Función S= Voraz(vector candidatos C)
S=Ø
Mientras (C!= Ø) y (S no es solución) **hacer:**
    x= Selección de candidato de C
    C= C \{x}
    Si es factible (S U {x}) **entonces**
        S= S U {x}
Fin-Mientras
Si S es solución **entonces**
    Devolver S
En otro caso
    Devolver "No hay solución"

Programación dinámica

Características:

- Construir la solución **por etapas**.
- **Dividir** un problema de tamaño **n** en uno o varios problemas de tamaño **n-1** que se **solapen entre sí**, existiendo uno o varios casos base al problema.
- **Mantiene en memoria** la solución a los subproblemas solucionados para **evitar cálculos repetidos**.
- Debe cumplirse el **Principio de Optimalidad de Bellman**.
- Se utilizan para resolver problemas de **optimización** (minimización/maximización).
- Se expresan mediante **ecuaciones recurrentes**.

PRINCIPIO DE OPTIMALIDAD DE BELLMAN:
Si una secuencia de pasos para resolver un problema es óptima, entonces cualquier subsecuencia de estos pasos también es óptima.