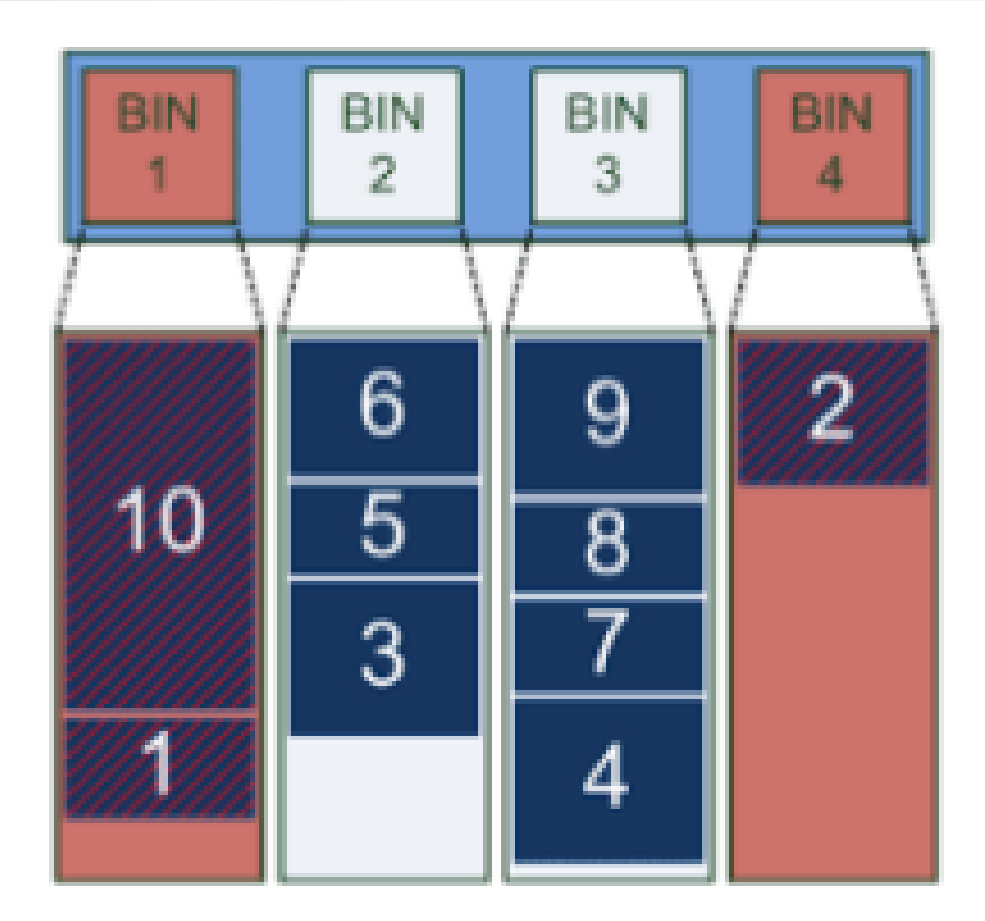
Bin Packing

Pablo Álvarez Cabreba
Johanna Capote Robayna
Cristina de la Torre Villaverde
Guillermo Galindo Ortuño
Adolfo Soto Werner

El problema



Datos:
-Número de objetos a embalar
-Tamaño de cada objeto
-Volumen de cada embalaje

¿Cómo insertamos los objetos para minimizar el número de embalajes necesarios?

Diseño de la solución

S=Conjunto de contenedores de tamaño V.
C=Conjunto de candidatos
S=1
Mientras C != 0 hacer:
  x=mayor elemento de C
  C=C\{x}
  Si {x} cabe en Si
    Si=Si U {x}
  Si no
    S aumenta su tamaño en 1
    Sn={x}
Fin-mientras
Devuelve S

n=índice del último vector de S
i=índice del vector de S

Componentes Greedy

- **Lista de candidatos:** Elementos que tenemos que empaquetar.
- **Lista de candidatos utilizados:** Los elementos que ya han sido empaquetados.
- **Función solución:** Comprueba si la lista de candidatos está vacía (se han empaquetado todos los elementos)
- **Función de selección:** Se selecciona el objeto de mayor volumen.
- **Criterio de factibilidad:** El volumen de los objetos contenidos en una caja es menor o igual que el volumen de la caja (si no es así se crea otra)
- **Función objetivo:** Devolver la forma de empaquetar los candidatos de forma que se utilicen el menor numero posible de paquetes.

Ejemplo

Volumen de cada embalaje = 7
Número de objetos a insertar = 10
Vector de volúmenes de los objetos = {1,2,2,2,1,2,1,1,2,5}

1	2	2	2	1	2	1	1	2	5

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1
X
X
X
X
X

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1
X
X
X
X
X
X
X

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1	S2
X	
X	
X	
X	
X	
X	X
X	X

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1	S2
X	
X	
X	
X	X
X	X
X	X
X	X

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1	S2
X	
X	X
X	X
X	X
X	X
X	X
X	X

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1	S2	S3
X		
X	X	
X	X	
X	X	
X	X	
X	X	X
X	X	X

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1	S2	S3
X	X	
X	X	
X	X	
X	X	
X	X	
X	X	X
X	X	X

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1	S2	S3
X	X	
X	X	
X	X	
X	X	
X	X	X
X	X	X
X	X	X

Ejemplo

1	2	2	2	1	2	1	1	2	5

S1	S2	S3
X	X	
X	X	
X	X	
X	X	X
X	X	X
X	X	X
X	X	X

Ejemplo

1	2	2	2	1	2	1	1	2	5

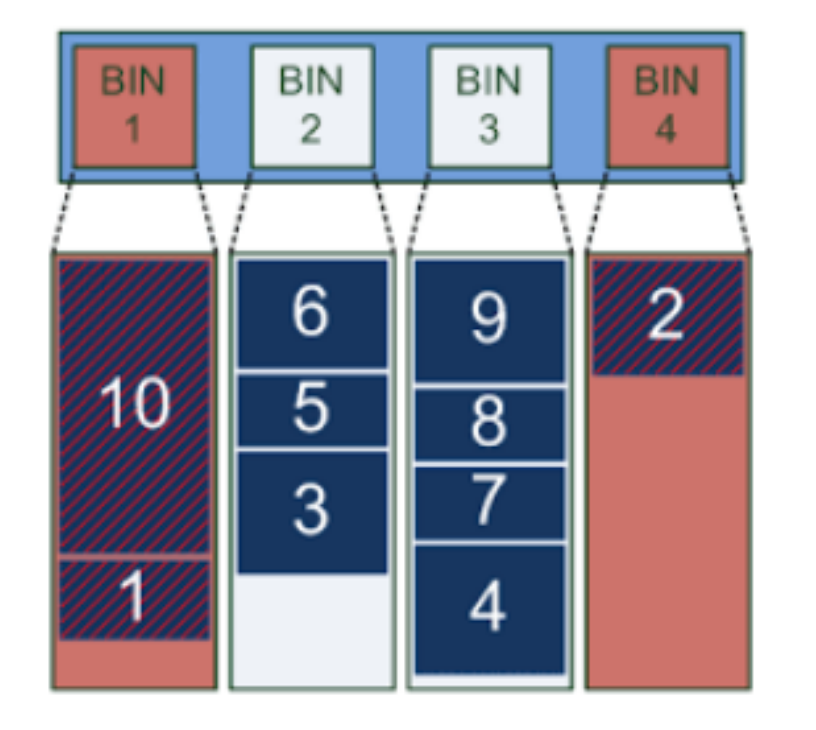
S1	S2	S3
X	X	
X	X	
X	X	X
X	X	X
X	X	X
X	X	X
X	X	X

Ejemplo: Solución final

1	2	2	2	1	2	1	1	2	5

S1	S2	S3
X	X	
X	X	
X	X	X
X	X	X
X	X	X
X	X	X
X	X	X

Ejemplo: otra posible solución



S1	S2	S3	S4
		X	
X		X	
X	X	X	
X	X	X	
X	X	X	
X	X	X	X
X	X	X	X

Comparativa de soluciones

S1	S2	S3
X	X	
X	X	
X	X	X
X	X	X
X	X	X
X	X	X
X	X	X

Solución con el algoritmo implementado:
-3 embalajes
-2 espacios sin utilizar

Solución sin el algoritmo implementado:
-4 embalajes
-8 espacios sin utilizar

S1	S2	S3	S4
		X	
X		X	
X	X	X	
X	X	X	
X	X	X	
X	X	X	X
X	X	X	X

Eficiencia

```
 8 Solucion AlgorimoGreedyBinPacking(Problema p){
 9   Solucion S;
10   vector<pair<int,double> > candidatos(p.getNumObjetos());
11
12   //Inicializamos lista de candidatos
13   for (int i = 0; i < p.getNumObjetos(); i++) {
14     candidatos[i].first = i;
15     candidatos[i].second = p.getVolumen(i);
16   }
17   // Añadimos la primera caja
18   S.addCaja();
```

O(n)

Eficiencia: O(n²)

```
20  //Mientras queden objetos sin incluir en cajas
21  while (!candidatos.empty()) {
22    //Funcion de seleccion (elemento con el máximo volumen)
23    vector<pair<int, double> >::iterator it, sol;
24    pair<int, double> max(-1,0);
25
26    for (it = candidatos.begin(); it != candidatos.end(); it++) {
27      if (max.second < it->second) {
28        max = *it;
29        sol = it;
30      }
31    }
32
33    //Eliminamos el objeto elegido de la lista de candidatos
34    candidatos.erase(sol);
35
36    //Incluimos el objeto elegido en la solucion
37    bool add = false;
38    for (int i = 0; i < S.getNumCajas() && !add; i++) {
39      //Si el objeto cabe en una caja existente, lo incluimos y salimos del bucle
40      if (S.getVolumen(i) + max.second <= p.getVolumen()){
41        S.addObjeto(i,max);
42        add = true;
43      }
44    }
45    //Si no, creamos una caja nueva y añadimos el objeto
46    if(!add){
47      S.addCaja();
48      S.addObjeto(S.getNumCajas()-1, max);
49    }
50  }
```

} O(n²)