Solving Knapsack Problems in a Sticker based model

Computación Bioinspirada

Fernando Sancho Caparrini & Mario de Jesús Pérez Jiménez

Adam Lei Yi Chen Abolacio

January 19, 2025

Table of Contents

Modelo Sticker de Roweis

- Representación de la información mediante cadenas de memoria (n, p, m).
- Hebras simples de ADN.
- Se trabaja con tubos que contienen multiconjuntos finitos de complejos de memoria.
- Bibliotecas (k, l): conjuntos de complejos de memoria cuyas primeras l regiones contienen todas las combinaciones.

Modelo Sticker de Roweis

Operaciones

- $Merge(T_1, T_2) = T_1 \cup T_2$
- $Separate(T, i) : +(T, i), -(T, i)$
- $Set(T, i)$
- $Unset(T, i)$
- $Read(T)$

Table of Contents

Problema de la suma de subconjuntos (SSP)

Entrada:

- Un conjunto de numeros enteros $A = \{1, \ldots, p\}$.
- Una función $w : A \to \mathbb{N}$.
  Notación: $w(A) = \sum_{i \in A} w(i)$
- Un número entero (target) $k \in \mathbb{N}$.

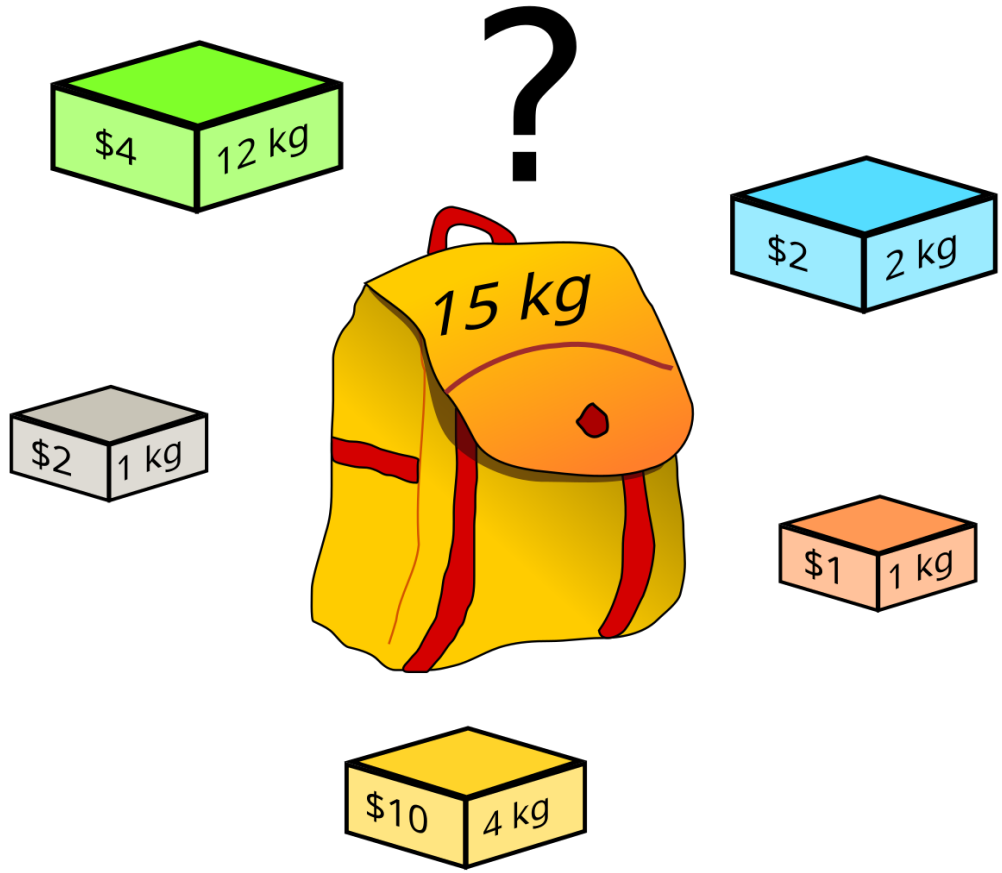
Objetivo: decidir si existe un subconjunto $B \subseteq A$ cuya suma de los pesos de B sea exactamente k. $w(B) = k$.

SSP es un problema NP-Completo.

Table of Contents

Problema de la mochila (KP)

- Objetos
- Peso máximo (mochila)
- Valor (objeto)
- Peso (objeto)
- Multiplicidad (objeto)



El problema de la mochila es uno de los **21 problemas NP-Completos de Karp** (1972).
KP es un problema de optimización.

Problema de la mochila (KP)

En este trabajo veremos la versión 0/1, donde cada objeto tiene multiplicidad 1.

Entrada:

- Un conjunto $A = \{1, \dots p\}$ que representan los índices de cada objeto.
- Una función $w : A \to \mathbb{N}$ que asocia un peso a cada objeto.
- Una función $\rho : A \to \mathbb{N}$ que asocia un valor a cada objeto.
- Un valor $k \in \mathbb{N}$ que indica el peso máximo de la mochila.

Problema de la mochila (KP)

Versión 0/1 unbounded: determinar si existe un subconjunto $B \subseteq A$ tal que $\rho(B) = max\{\rho(C) : C \subseteq A \wedge w(C) \leq k\}$.

Versión 0/1 bounded: dado un valor adicional de entrada k' (valor mínimo) determinar si existe un subconjunto $B \subseteq A$ tal que $w(B) \leq k$ y $\rho(B) \geq k'$.

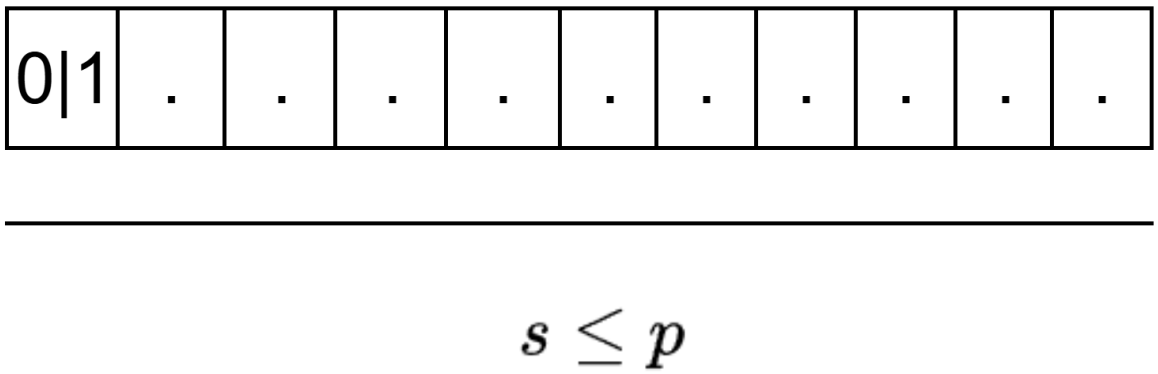
Table of Contents

Programa para ordenar por cardinalidad

Sea: $A = \{1, \ldots, p\}, B = \{b_1, \ldots, b_s\} \subseteq A, \mathcal{F} = \{D_1, \ldots, D_t\} \subseteq \mathcal{P}(A)$.
Ordenar los conjuntos de $\mathcal{F}$ según la cardinalidad relativa a B.

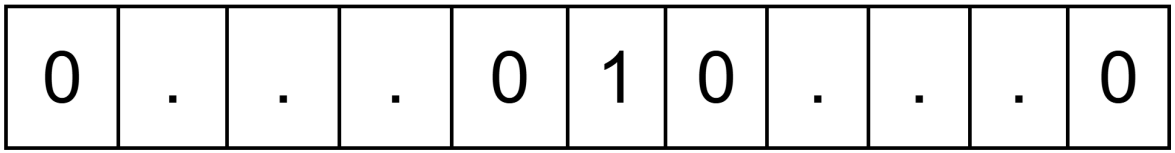
Entrada: (T_0, B)
T_0: tubo de entrada, contiene complejos de memoria σ que representan los elementos seleccionados de B.

Codificación de $\sigma \in T_0$: Si la región b_i está activa, entonces el elemento b_i pertenece al conjunto representado en σ.



Programa para ordenar por cardinalidad

Ejemplo de codificación de $\sigma \in T_0$: un conjunto D que solo contiene a b_1



b_i

Programa para ordenar por cardinalidad

Idea intuitiva del programa: un bucle principal FOR de s pasos, en cada paso i, generar $i+1$ tubos: $T_0, T_1, \ldots T_i$.
Cada tubo generado verifica:

$$\forall \sigma (\sigma \in T_j \rightarrow |\sigma \cap \{b_1, \ldots b_i\}| = j)$$

Para cumplir la propiedad anterior se generan los tubos de la siguiente manera:

$$\begin{cases} T_0 = & -(T_0, b_{i+1}) \\ T_j = & +(T_{j-1}, b_{i+1}) \cup -(T_j, b_{i+1}) \quad (1 \leq j \leq i) \\ T_{i+1} = & +(T_i, b_{i+1}) \end{cases}$$

Programa para ordenar por cardinalidad

Cardinal_sort(T_0, B), Cardinal_sort(T_0), Cardinal_sort(T_0, l, k)

Input: (T_0, B)

for $i = 1$ **to** s **do**

$(T_0, T_1') \leftarrow$ separate(T_0, b_i)

for $j = 0$ **to** $i - 1$ **do**

$(T_j'', T_{j+1}') \leftarrow$ separate(T_j, b_i)

$T_j \leftarrow T_j' \cup T_j''$

end for

$T_i \leftarrow T_i'$

end for

Output: $T_0, \ldots, T_s$

- $2s$ tubos auxiliares.
- $\frac{s(s+3)}{2}$ operaciones moleculares (*separate*).

Programa para ordenar por cardinalidad

Verificación Formal

Input: T_0

$T_{0,0} \leftarrow T_0$; $T_{0,-1} \leftarrow \emptyset$; $T_{0,1} \leftarrow \emptyset$

for $i = 1$ **to** s **do**

$T_{i,-1} \leftarrow \emptyset$; $T_{i,i+1} \leftarrow \emptyset$

for $j = 0$ **to** i **do**

$T_{i,j} \leftarrow +(T_{i-1,j-1}, b_i) \cup -(T_{i-1,j}, b_i)$

end for

end for

Output: $T_{s,0}, \ldots, T_{s,s}$

Programa para ordenar por cardinalidad

Verificación Formal

Notación: $B_j = \{b_1, \ldots, b_j\}$ y $B_0 = \emptyset$.

Proposición 1

$$\forall i\,(1 \leq i \leq s \rightarrow \forall j \leq i\, \forall \sigma(\sigma \in T_{i,j} \rightarrow |\sigma \cap B_i| = j))$$

Inducción en i:

- $i = 1 \rightarrow \forall j \leq 1\, \forall \sigma(\sigma \in T_{1,j} \rightarrow |\sigma \cap B_1| = j)$
  - Si $\sigma \in T_{1,0} = +(T_{0,-1}, b_1) \cup -(T_{0,0}, b_1)$, $j = 0$ $T_{0,-1} = \emptyset$ y $T_{0,0} = T_0$, entonces $T_{1,0} = -(T_0, b_1)$ y se cumple que $|\sigma \cap B_1| = 0$ ya que $b_1 \notin \sigma$
  - Si $\sigma \in T_{1,1} = +(T_{0,0}, b_1) \cup -(T_{0,1}, b_1)$, $j = 1$ $T_{0,0} = T_0$, $T_{0,1} = \emptyset$, entonces $T_{1,1} = +(T_0, b_1)$ y $b_1 \in \sigma$ por lo que también se cumple $|\sigma \cap B_1| = 1$

Paso inductivo en i, suponemos que se verifica la **Proposición 1** para i:

$$\forall j \leq i \, \forall \sigma \in T_{i,j} \, (|\sigma \cap B_i = j|)$$

verifiquemos que también se cumple para $i+1$.

Inducción en j: $\forall j \leq i+1 \, \forall \sigma \in T_{i+1,j} \, (|\sigma \cap B_{i+1} = j|)$

- $j = 0$
  $\sigma \in T_{i+1,0} = +(T_{i,-1}, b_{i+1}) \cup -(T_{i,0}, b_{i+1})$
  $T_{i,-1} = \emptyset$, entonces $\sigma \in T_{i,0}$ y $b_{i+1} \notin \sigma$. Por hipótesis de inducción $|\sigma \cap B_i| = 0$, como $b_{i+1} \notin \sigma$ se cumple que $|\sigma \cap B_{i+1}| = 0$.

Inducción en j:

$$\forall j \leq i+1 \, \forall \sigma \in T_{i+1,j} \, (|\sigma \cap B_{i+1} = j|)$$

- $j > 0$
  $\sigma \in T_{i+1,j} = +(T_{i,j-1}, b_{i+1}) \cup -(T_{i,j}, b_{i+1})$
  - Si $\sigma \in T_{i,j-1}$, entonces $b_{i+1} \in \sigma$. Por hipótesis de inducción $|\sigma \cap B_i| = j-1$, además como $b_{i+1} \in \sigma$ tenemos que $|\sigma \cap B_{i+1}| = j-1+1 = j$
  - Si $\sigma \in T_{i,j}$, entonces $b_{i+1} \notin \sigma$. Por hipótesis de inducción $|\sigma \cap B_i| = j$, como $b_{i+1} \notin \sigma$, $|\sigma \cap B_{i+1}| = j$.

Hemos verificado que se cumple la **Proposición 1** para cualquier valor de i, j.

Proposición 2

$$\forall \sigma \in T_0 \, \forall i \, (0 \leq i \leq s \rightarrow \sigma \in T_{i,|\sigma \cap B_i|})$$

Inducción en i:

- $i = 0$
  El resultado es trivial, ya que $B_0 = \emptyset$, entonces $|\sigma \cap B_0| = 0$, por lo que $\forall \sigma \in T_0 \rightarrow \sigma \in T_{0,0}$ esto se cumple ya que $T_{0,0} = T_0$.

Suponemos que se verifica para cualquier i.

- $i + 1$
  - Si $b_{i+1} \in \sigma$ tenemos que $|\sigma \cap B_{i+1}| = 1 + |\sigma \cap B_i|$. Por hipótesis de inducción $\sigma \in T_{i,|\sigma \cap B_i|}$, por lo que $\sigma \in +(T_{i,|\sigma \cap B_i|}, b_{i+1}) \subseteq T_{i+1,|\sigma \cap B_i|+1}$
  - Si $b_{i+1} \notin \sigma$ tenemos que $|\sigma \cap B_{i+1}| = |\sigma \cap B_i|$. Por hipótesis de inducción $\sigma \in T_{i,|\sigma \cap B_i|}$, por lo que $\sigma \in -(T_{i,|\sigma \cap B_i|}, b_{i+1}) \subseteq T_{i+1,|\sigma \cap B_i|} = T_{i+1,|\sigma \cap B_{i+1}|}$

Programa para ordenar por cardinalidad

Verificación Formal

Corolario 1. (Solución correcta)

$$\forall j \, \forall \sigma \, (0 \leq j \leq s \wedge \sigma \in T_{s,j} \rightarrow |\sigma \cap B| = j)$$

Corolario 2. (Solución completa)

$$\text{If } \sigma \in T_0 \text{ and } |\sigma \cap B| = j, \text{ then } \sigma \in T_s, j$$

Table of Contents

Subrutina de codificación

Notación: $f(B) = \sum_{i \in B} f(i)$

Sea:

- $A = \{1, \ldots, p\}$
- $r \in \mathbb{N}$
- $f : A \to \mathbb{N}$: función que asigna un valor a cada elemento del conjunto A.

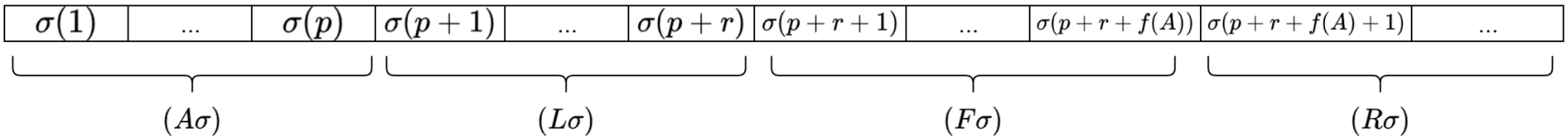
La entrada de la rutina es un multiconjunto de complejos de memoria (n, k, m) que denominamos como T_0, $k \geq p + r + f(A)$.

Subrutina de codificación

Si $\sigma \in T_0$, dividiremos el complejo de memoria cuatro zonas:

$$(A\sigma) = \sigma(1)\dots\sigma(p) \qquad (F\sigma) = \sigma(p+r+1)\dots\sigma(p+r+f(A))$$
$$(L\sigma) = \sigma(p+1)\dots\sigma(p+r) \qquad (R\sigma) = \sigma(p+r+f(A)+1)\dots$$

La salida de la rutina codifica en la zona $(F\sigma)$ el valor total de los elementos codificados en $(A\sigma)$ para cualquier función f.



$$\sum_{i=1}^{p} \sigma(i) f(i) = \sum_{j=p+r+1}^{p+r+f(A)} \sigma(j)$$

Subrutina de codificación

Parallel_fill(T_0, f, p, r)

Input: (T_0, f, p, r)

for $i = 1$ **to** p **do**

$(T^+, T^-) \leftarrow separate(T_0, i)$

for $j = 1$ **to** $f(i)$ **do**

$T^+ \leftarrow set(T^+, p + r + f(A_{i-1}) + j)$

end for

$T_0 \leftarrow merge(T^+, T^-)$

end for

Output: T_0

Subrutina de codificación

Verificación Formal

Input: (T_0, f, p, r)

for $i = 1$ **to** p **do**

$(T_{i,0}^{+}, T_i^{-}) \leftarrow separate(T_{i-1}, i)$

for $j = 1$ **to** $f(i)$ **do**

$T_{i,j}^{+} \leftarrow set(T_{i,j-1}^{+}, p + r + f(A_{i-1}) + j)$

end for

$T_i \leftarrow merge(T_{i,f(i)}^{+}, T_i^{-})$

end for

Output: T_p

Para cada valor i $(1 \leq i \leq p)$ consideramos las regiones:

$$R_i = \{p + r + f(A_{i-1} + 1, \ldots, p + r + f(A_i)\}$$

Definición σ^k

Para cualquier $\sigma \in T_0$ y cada k $(1 \leq k \leq p)$, σ^k es la molécula obtenida de σ en el paso de ejecución k-ésimo del bucle principal.

Lema 1

La zona inicial $(A\sigma)$ no cambia durante la ejecución del programa:

$$\forall \sigma \in T_0 \, \forall k \, (1 \leq k \leq p \rightarrow (A\sigma) = (A\sigma^k))$$

Subrutina de codificación

Verificación Formal

Lema 2

Las moléculas del paso k-ésimo del bucle principal son almacenados en el tubo T_k:

$$\forall \sigma \in T_0 \, \forall k \, (1 \leq k \leq p \rightarrow \sigma^k \in T_k)$$

Lema 3

Toda molécula de T_k viene de una molécula del tubo inicial:

$$\forall k \, (1 \leq k \leq p \rightarrow \forall \tau \in T_k \, \exists \sigma \in T_0 (\sigma^k = \tau))$$

Lema 4

La ejecución de un paso en el bucle principal no modifica las regiones de los pasos anteriores:

$$\forall \sigma \in T_0 \, \forall i \, \forall k \, (1 \leq i \leq k \leq p \rightarrow \sigma^i_{|R_i} = \sigma^k_{|R_i})$$

Lema 5

Tras la ejecución del paso i-ésimo del bucle principal, la región R_i de σ ha sido modificada para representar el valor de $\sigma(i)$:

$$\forall \sigma \in T_0 \, \forall i \, \forall k \, (1 \leq i \leq k \leq p \rightarrow \sigma^k_{|R_i} \equiv \sigma(i))$$

Lema 6

El programa no modifica las zonas $(L\sigma)$ ni $(R\sigma)$:

$$\forall \sigma \in T_0 \, \forall k \, (1 \leq i \leq k \leq p \rightarrow (L\sigma) = (L\sigma^k) \wedge (R\sigma) = (R\sigma^k))$$

Partiendo de los lemas (1) y (5) podemos verificar la **Proposición 3**.

Proposición 3

Sea $B \subseteq A$ tal que $\sigma_B \in T_0$, para cada valor de k $(1 \leq k \leq p)$ se tiene que:

$$f(B \cap \{1, \ldots, k\}) = \sum_{j=p+r+1}^{p+r+f(A_k)} \sigma_B^k(j)$$

Corolario 3

Para cada $B \subseteq A$ tal que $\sigma_B \in T_0$ existe un $\tau \in T_p$ tal que:

$$f(B) = \sum_{i=p+r+1}^{p+r+f(A)} \tau(i)$$

Table of Contents

Solución molecular al SSP

- $A = \{1, \ldots, p\}$
- $w : A \to \mathbb{N}$: función de pesos
- $k \in \mathbb{N}, (k \leq w(A))$: suma objetivo
- T_0 : biblioteca $(p + w(A), p)$

Determinar si existe un conjunto $B \subseteq A$ tal que $w(B) = k$.

Subset_Sum(p, w, k)

$q_w \leftarrow \sum_{i=1}^{p} w(i)$
$T_0 \leftarrow (p + q_w, p)$-library
$T_1 \leftarrow$ Parallel_Fill$(T_0, w, p, 0)$
$T_{out} \leftarrow$ Cardinal_sort$(T_1, p + 1, p + q_w)[k]$
Read(T_{out})

- $4 + 2q$ tubos usados.
- $2p + q + 1 + \frac{q(q+3)}{2}$ operaciones moleculares.

Correctitud: Si $T_{out} \neq \emptyset$ entonces existe un $B \subseteq A$ tal que $w(B) = k$.

Si tomamos cualquier molécula $\tau \in T_{out}$, por el **Lema 3** sabemos que τ viene de una molécula del tubo de entrada $\sigma \in T_0$. Además, por la **Proposición 3**, la molécula τ codifica correctamente el peso total de B_σ.

Completitud: sea $\sigma \in T_0$ tal que $w(B_\sigma) = k$, entonces $T_{out} \neq \emptyset$

Sea $\sigma \in T_0$ tal que $w(B_\sigma) = k$, tras la ejecución de Parallel_Fill, por el **Corolario 3** ($\forall \sigma_B \in T_0 \exists \tau \in T_p(f(B) = \sum_{i=p+r+1}^{p+r+f(A)} \tau(i))$ sabemos que tenemos una molécula $\tau = \sigma^p \in T_1$ tal que $w(B_\sigma) = \sum_{i=p+1}^{p+q_w} \tau(i)$, por lo que $\tau \in T_{out}$.

Table of Contents

Solución molecular al KP 0/1 Acotado

- $A = \{1, \ldots, p\}$
- $w : A \to \mathbb{N}$: función de pesos
- $\rho \to \mathbb{N}$: función de valores
- $k \in \mathbb{N}(k \leq w(A))$: peso máximo
- $k' \in \mathbb{N}\ (k' \leq \rho(B))$: valor mínimo

Determinar si existe un conjunto $B \subseteq A$ tal que $w(B) \leq k$ y $\rho(B) \geq k'$.

Solución molecular al KP 0/1 Acotado

Knapsack(p, w, ρ, k, k')

Input: (p, w, ρ, k, k')

$q_w \leftarrow \sum_{i=1}^{p} w(i)$; $q_\rho \leftarrow \sum_{i=1}^{p} p(i)$; $T_0 \leftarrow (p + q_w + q_\rho, p)-$library

$T_0 \leftarrow$ Parallel_fill$(T_0, w, p, 0)$

Cardinal_sort$(T_0, p + 1, p + q_w)$

$T_1 \leftarrow \emptyset$

for $i = 1$ **to** k **do**

$T_1 \leftarrow$ merge$(T_1,$ Cardinal_sort$(T_0, p + 1, p + q_w)[i])$

end for

$T_0 \leftarrow$ Parallel_Fill(T_1, ρ, p, q_w)

Cardinal_sort$(T_0, p + q_w + 1, p + q_w + q_\rho)$

$T_1 \leftarrow \emptyset$

for $i = k'$ **to** q_ρ **do**

$T_1 \leftarrow$ merge$(T_1,$ Cardinal_sort$(T_0, p + q_w + 1, p + q_w + q_\rho)[i])$

end for

Read(T_1)

Table of Contents

Solución molecular al KP 0/1 no acotado

Unbounded_Knapsack(p, w, ρ, k)

Input: (p, w, ρ, k)

$q_w \leftarrow \sum_{i=1}^{p} w(i)$; $q_\rho \leftarrow \sum_{i=1}^{p} p(i)$

$T_0 \leftarrow (p + q_w + q_\rho, p)-$library

$T_0 \leftarrow$ Parallel_fill$(T_0, w, p, 0)$

Cardinal_sort$(T_0, p + 1, p + q_w)$

$T_1 \leftarrow \emptyset$

for $i = 1$ **to** k **do**

$T_1 \leftarrow$ merge$(T_1,$ Cardinal_sort$(T_0, p + 1, p + q_w)[i])$

end for

$T_0 \leftarrow$ Parallel_Fill(T_1, ρ, p, q_w)

$i \leftarrow q_\rho$; $t \leftarrow 0$

Cardinal_sort$(T_0, p + q_w + 1, p + q_w + q_\rho)$

while $i \geq 1 \wedge t = 0$ **do**

$T' \leftarrow$ Cardinal_sort$(T_0, p + q_w + 1, p + q_w + q_\rho)[i]$

if $T' \neq \emptyset$ **then** Read(T'); $t \leftarrow 1$

else $i \leftarrow i - 1$

end if

end while

Solución molecular al KP 0/1 no acotado

- $5 + 2 \cdot max\{q_w, q_\rho\}$ tubos auxiliares
- $4p + k - k' + \frac{q_w(q_w+5)+q_\rho(q_\rho+9)}{2}$ operaciones moleculares

Table of Contents

Conclusiones

- Las soluciones propuestas permiten resolver los problemas SSP y KP 0/1, problemas NP-Completos.
- Número de tubos lineal
- Número de operaciones moleculares cuadrático
- Se requiere un espacio de tamaño exponencial para las bibliotecas
- Solución para valores en $\mathbb{N}$