



Programación no lineal convexa

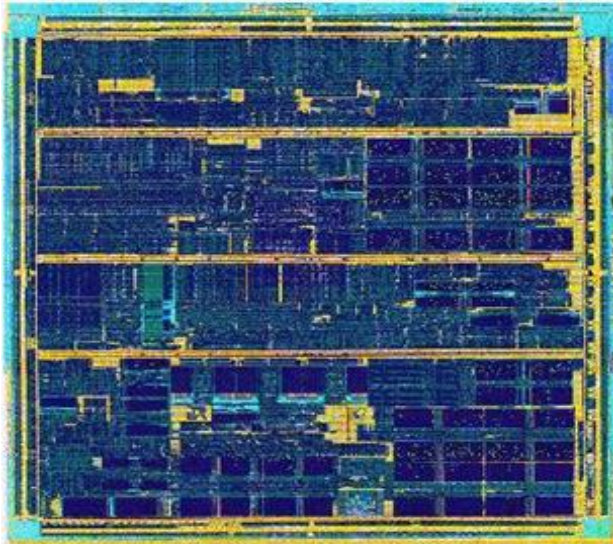
Rodrigo Maranzana

Programación convexa

Floorplanning

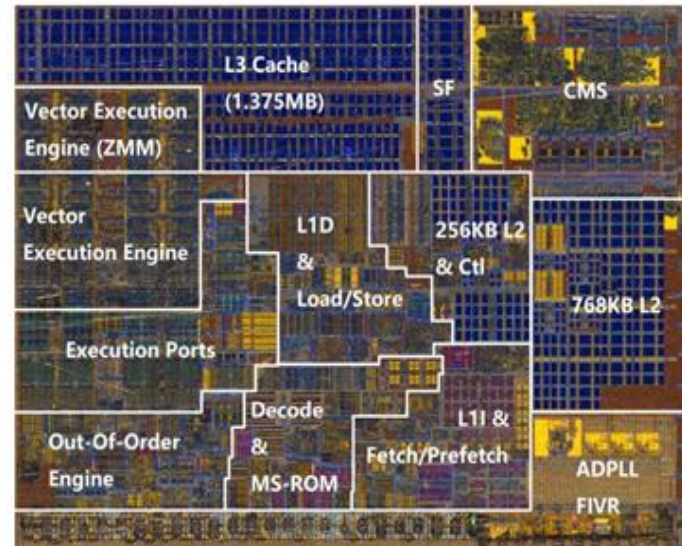
Floorplanning es un proceso de diseño de optimización de particiones relacionado con microelectrónica.

Layout de un chip



Adya y Markov (2004), "Fixed-outline floorplanning: Enabling Hierarchical design"

Intel i7 Skylake floorplan



<https://limsk.ece.gatech.edu/course/ece6133/slides/floorplanning.pdf>

Floorplanning y distribución de plantas industriales

En el campo de distribución de plantas industriales, floorplanning implica optimizar el diseño de un layout de planta sujeto a restricciones.

Algunos ejemplos de restricciones:

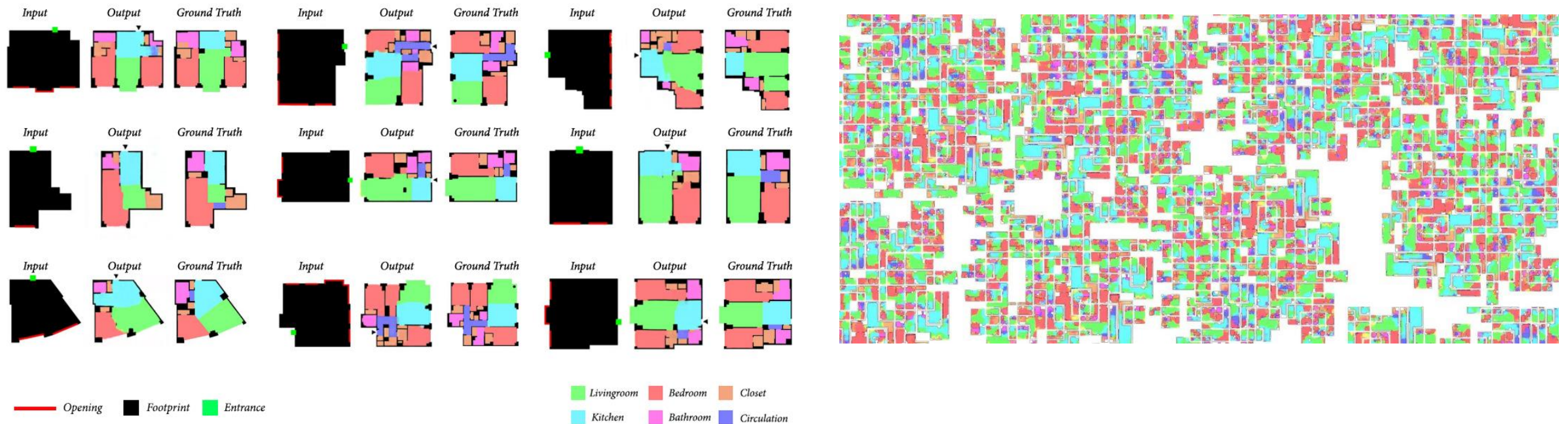
- Flujo de recursos entre áreas o procesos.
- Procesos que deben cumplir requerimientos espaciales. Ej: ancho de zonas de circulación para elevadores.
- Procesos con restricciones de vecindad. Ej: calderas y almacenes con alta carga de fuego.

Nuevas tendencias en Floorplanning

Nuevas tendencias en optimización de floorplan con Generative AI:

- Modelo se entrena con datasets de distribuciones de planta conocidas.
- Infiere sobre casos no vistos anteriormente.

Nvidia (2019) "ArchiGAN: a Generative Stack for Apartment Building Design"



Fuente: <https://developer.nvidia.com/blog/archigan-generative-stack-apartment-building-design/>

Floorplanning como modelo convexo

Si bien el modelo general de **floorplanning es no lineal, no convexo**, cumpliendo condiciones como:

- Las áreas no pueden solaparse y se consideran rectangulares.
- Se optimiza por área o perímetro.

Es un problema combinatorio muy complejo.

Se puede llegar a una **solución convexa**, específicamente de **programación cuadrática**, imponiendo posiciones relativas entre áreas.

Ejemplo: modelo de floorplanning convexo

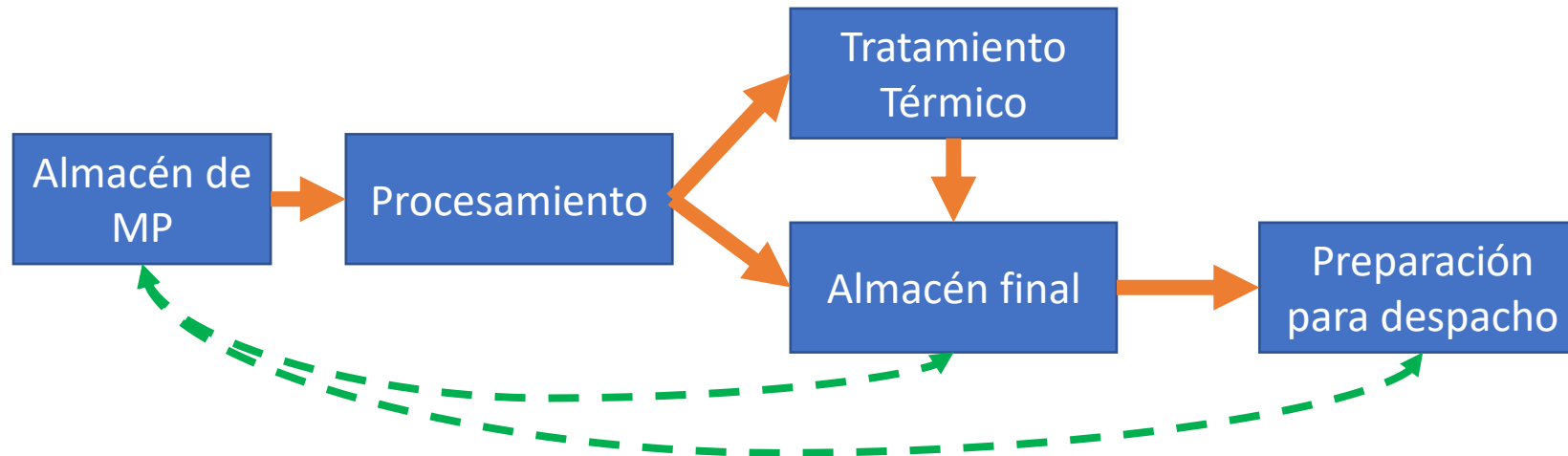
Una empresa metalúrgica planea la construcción de una nueva planta. La misma debería contar con 5 áreas de manufactura que necesitan un mínimo de superficie para poder contener al personal y maquinaria de forma segura y productiva:

- Área de almacenamiento de materia prima, 350 m²
- Área de procesamiento: corte, trefilado y punzonado, 300 m²
- Área de tratamiento térmico, 200 m²
- Área de enfriamiento y almacenaje de producto final, 300 m²
- Preparación para despacho, 300m²

Se busca determinar la posición, largo y ancho de cada área minimizando el perímetro total de la planta.

Ejemplo: modelo de floorplanning convexo

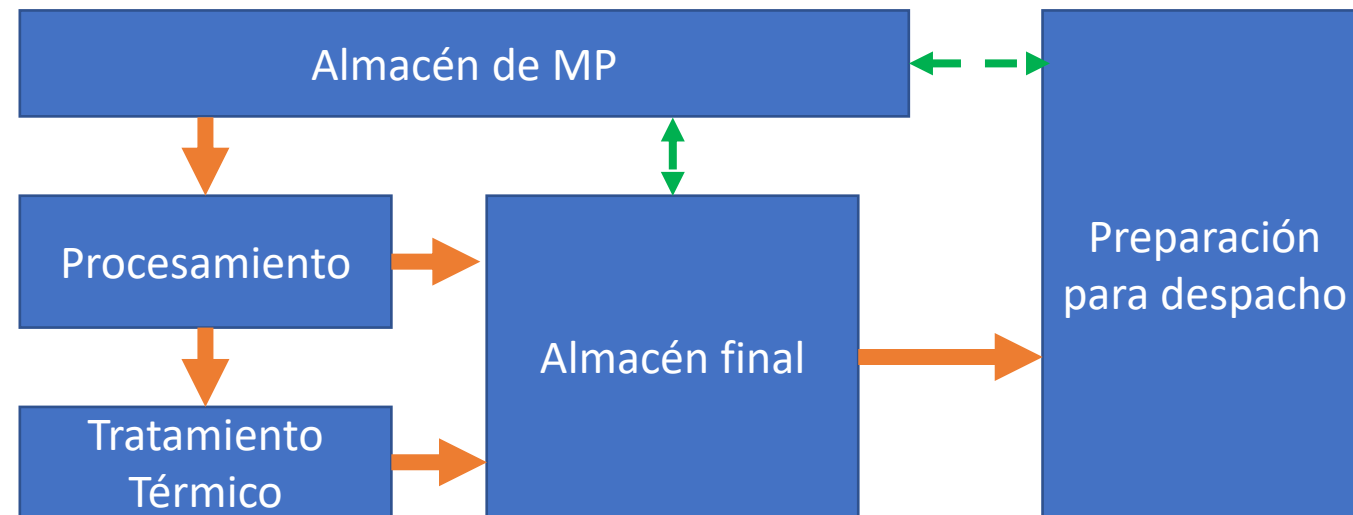
Las áreas no pueden construirse de cualquier forma, ya que algunas deben estar comunicadas. El flujo productivo puede describirse en el grafo en naranja y comunicación directa adicional en verde.



- Como podemos ver, no todos los productos se someten a tratamiento térmico.
- Todas las áreas logísticas están comunicadas.

Ejemplo: modelo de floorplanning convexo

- Para saber la ubicación de cada área, es necesario transformar el grafo anterior, a uno geográfico con arcos verticales y horizontales. Los arcos no pueden cortarse.
- Si bien este es un problema de optimización adicional, escapa a la materia y vamos a resolverlo intuitivamente (pueden buscar sus propias soluciones).
- Conocer o suponer la posición relativa de las áreas, permite que este problema combinatorio pueda resolverse con un Modelo de Optimización Convexa.



Ejemplo: modelo de floorplanning convexo

Condiciones:

- No existen restricciones fijas de largo y ancho.
- Hay que cumplir un mínimo y máximo de relación de aspecto (largo:ancho); máximo 5:1 y mínimo 1:5
- Alrededor de cada área debe existir un pasillo de circulación de 2,5 metros.

Si bien conocemos la dependencia geográfica de cada área, desconocemos cada ancho, largo y posición exacta.

Este es nuestro problema de Floorplanning.

Ejemplo: modelo de floorplanning convexo

Sets:

i : áreas $\{0, 1, 2, 3, 4\}$ Almacenamiento inicial, procesamiento, tratamiento térmico, almacenamiento final, despacho.

Variables de decisión:

- x_i : Coordenada x en el plano del vértice inferior izquierdo del área i.
- y_i : Coordenada y en el plano del vértice inferior izquierdo del área i.
- w_i : Ancho del área i.
- h_i : Largo del área i.
- W: Ancho total de la planta
- H: Largo del área i.

Parámetros:

- ρ : espacio entre áreas, pasillos de circulación.
- γ_{max} : relación de aspecto máx.
- γ_{min} : relación de aspecto mín.

Ejemplo: modelo de floorplanning convexo

Función objetivo:

$$\textit{Min } W + H$$

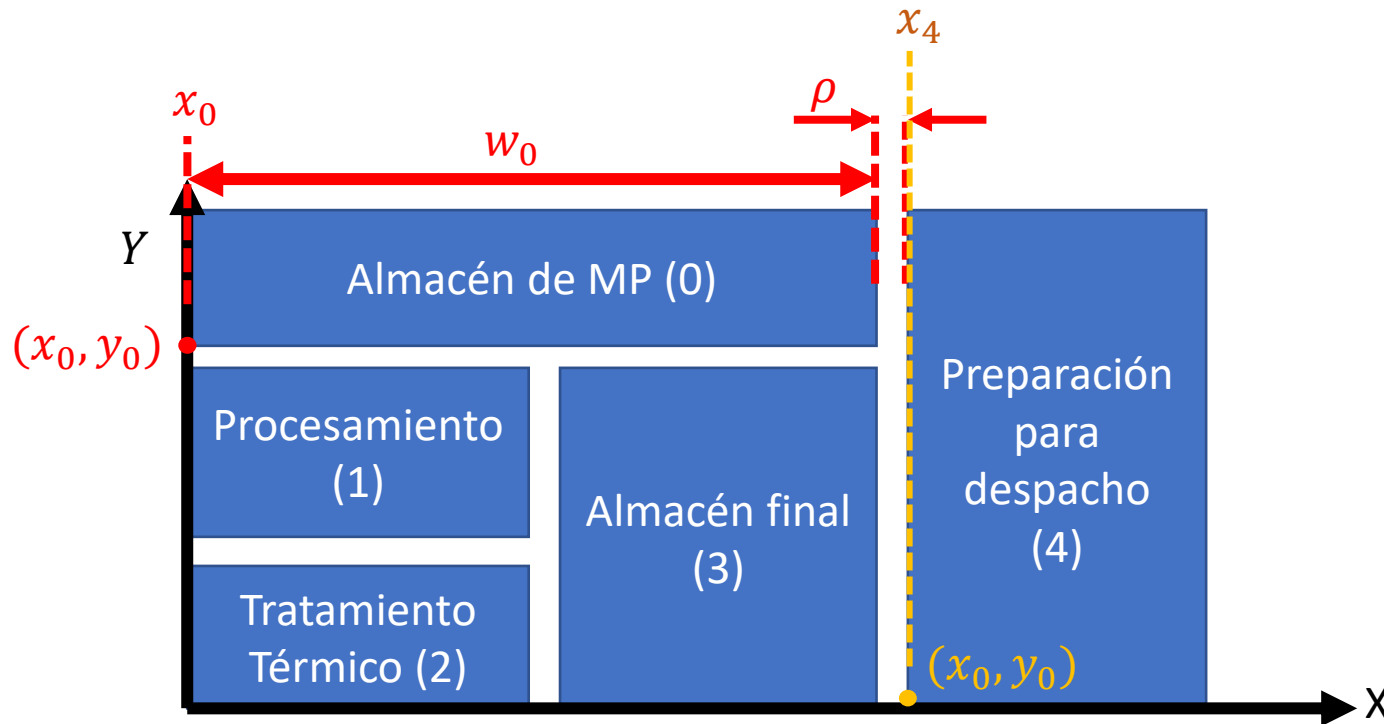
Minimizar perímetro de la planta $2(W + H)$.

Dado que la constante 2 no altera el problema de optimización, la eliminamos.

Ejemplo: modelo de floorplanning convexo

Restricciones de posición relativa de áreas:

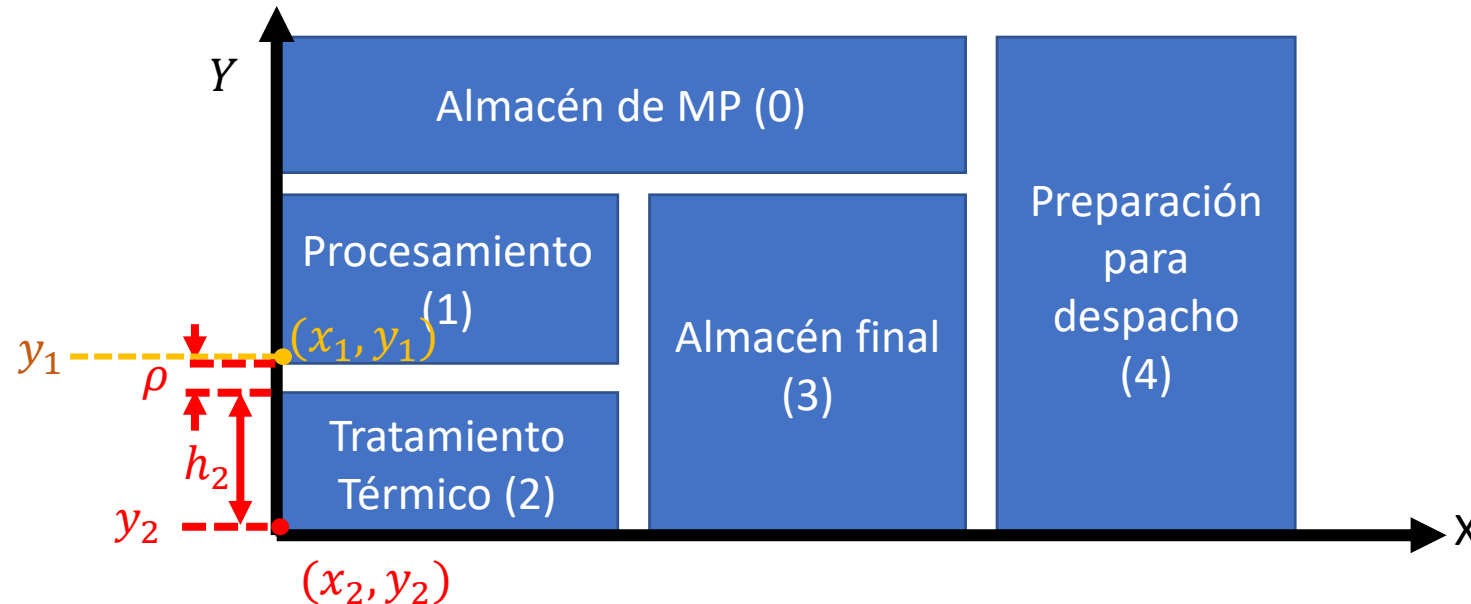
Ejemplo eje x: $x_0 + w_0 + \rho \leq x_4$



Ejemplo: modelo de floorplanning convexo

Restricciones de posición relativa de áreas:

Ejemplo eje y : $y_2 + h_2 + \rho \leq y_1$



Ejemplo: modelo de floorplanning convexo

Restricciones de posición relativa de áreas:

En el eje x:

$$x_0 + w_0 + \rho \leq x_4$$

$$x_1 + w_1 + \rho \leq x_3$$

$$x_2 + w_2 + \rho \leq x_3$$

$$x_3 + w_3 + \rho \leq x_4$$

$$x_4 + w_4 \leq W$$

En el eje y:

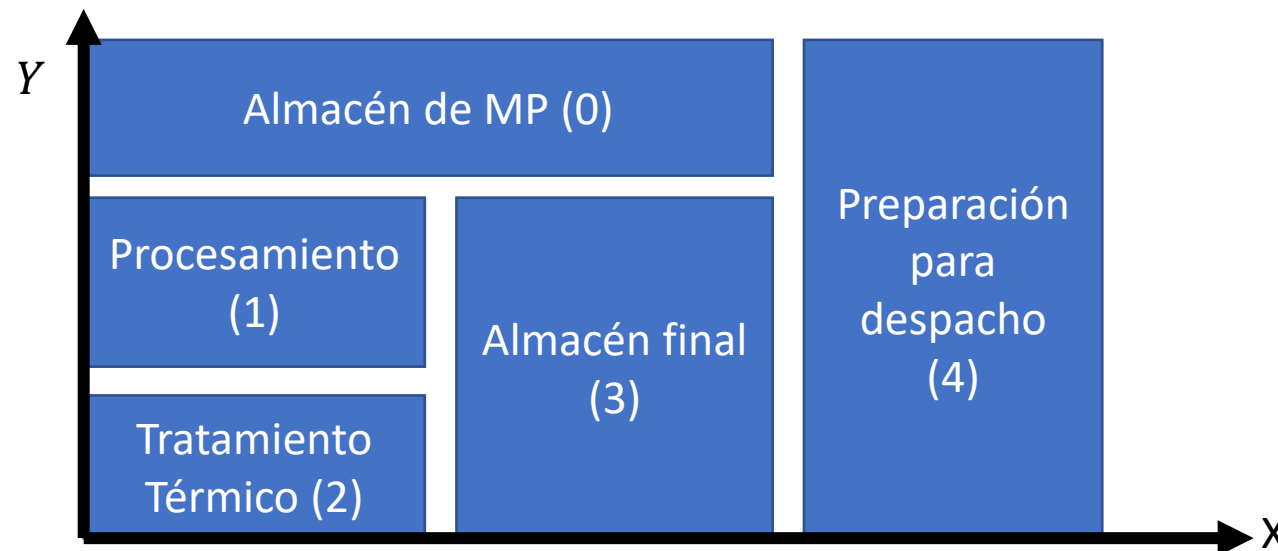
$$y_2 + h_2 + \rho \leq y_1$$

$$y_1 + h_1 + \rho \leq y_0$$

$$y_3 + h_3 + \rho \leq y_0$$

$$y_0 + h_0 \leq H$$

$$y_4 + h_4 \leq H$$



Ejemplo: modelo de floorplanning convexo

Restricciones de área:

Área mínima:

$$Amin_i \leq w_i * h_i \quad \forall i \quad \xrightarrow{\text{convertimos a restricción convexa}} \quad \log(Amin_i) \leq \log(w_i) + \log(h_i) \quad \forall i$$

Límites de relación de aspecto:

$$\begin{aligned} h_i &\leq \gamma_{max} * w_i & \forall i \\ h_i &\geq \gamma_{min} * w_i & \forall i \end{aligned}$$

Ejemplo: modelo de floorplanning convexo

$$\text{Min } W + H$$

s.t.

$$x_0 + w_0 + \rho \leq x_4$$

$$x_1 + w_1 + \rho \leq x_3$$

$$x_2 + w_2 + \rho \leq x_3$$

$$x_3 + w_3 + \rho \leq x_4$$

$$x_4 + w_4 \leq W$$

$$y_2 + h_2 + \rho \leq y_1$$

$$y_1 + h_1 + \rho \leq y_0$$

$$y_3 + h_3 + \rho \leq y_0$$

$$y_0 + h_0 \leq H$$

$$y_4 + h_4 \leq H$$

$$\log(Amin_i) \leq \log(w_i) + \log(h_i) \quad \forall i$$

$$h_i \leq \gamma_{max} * w_i \quad \forall i$$

$$h_i \geq \gamma_{min} * w_i \quad \forall i$$

$$x_i \geq 0, \quad y_i \geq 0, \quad w_i \geq 0, \quad h_i \geq 0 \quad \forall i,$$

$$W \geq 0, \quad H \geq 0$$

Resolución en Python con librería cvxpy

CVXPY es una librería open-source de optimización convexa.

Permite modelizar problemas, resolverlos y hacer chequeos de convexidad en las ecuaciones.

Documentación: <https://www.cvxpy.org/>

Repositorio: <https://github.com/cvxpy/cvxpy>

Resolución en Python con librería cvxpy

```
import cvxpy as cp
import numpy as np

# Parametros
rho = 2.5 # Espacio entre cajas
gamma_max = 5 # Aspect ratio superior
gamma_min = 1/5 # Aspect ratio inferior
Amin = np.array([350., 300., 200., 300., 300.])

# Variables
W = cp.Variable(shape=(1)) # Ancho
H = cp.Variable(shape=(1)) # Largo
x = cp.Variable(shape=(5)) # Posición x del vértice inferior izquierdo
y = cp.Variable(shape=(5)) # Posición y del vértice inferior izquierdo
w = cp.Variable(shape=(5)) # Ancho de la caja
h = cp.Variable(shape=(5)) # Largo de la caja
```

Resolución en Python con librería cvxpy

```
# Indicamos la función objetivo:
objective = cp.Minimize(H + W) # Minimizar perímetro total

# Indicamos las restricciones:
## Restricciones de dependencia entre cajas en x:
constraints = [
    x[0] + w[0] + rho ≤ x[4],
    x[1] + w[1] + rho ≤ x[3],
    x[2] + w[2] + rho ≤ x[3],
    x[3] + w[3] + rho ≤ x[4],
    x[4] + w[4] ≤ W
]

## Restricciones de dependencia entre cajas de y:
constraints += [
    y[2] + h[2] + rho ≤ y[1],
    y[1] + h[1] + rho ≤ y[0],
    y[3] + h[3] + rho ≤ y[0],
    y[0] + h[0] ≤ H,
    y[4] + h[4] ≤ H
]
```

$\text{Min } W + H$

$$\begin{aligned}x_0 + w_0 + \rho &\leq x_4 \\x_1 + w_1 + \rho &\leq x_3 \\x_2 + w_2 + \rho &\leq x_3 \\x_3 + w_3 + \rho &\leq x_4 \\x_4 + w_4 &\leq W \\y_2 + h_2 + \rho &\leq y_1 \\y_1 + h_1 + \rho &\leq y_0 \\y_3 + h_3 + \rho &\leq y_0 \\y_0 + h_0 &\leq H \\y_4 + h_4 &\leq H\end{aligned}$$

Resolución en Python con librería cvxpy

```
# Restricción de área mínima para cada caja:
constraints.append(cp.log(Amin) ≤ cp.log(w) + cp.log(h))
# multiplicacion elemento por elemento

# Restricción de aspect ratio:
constraints.append(h ≤ cp.multiply(gamma_max, w))
constraints.append(h ≥ cp.multiply(gamma_min, w))

# Restricciones de positividad:
constraints += [
    W ≥ 0,
    H ≥ 0,
    x ≥ 0,
    y ≥ 0,
    w ≥ 0,
    h ≥ 0,
]
```

$$\log(Amin_i) \leq \log(w_i) + \log(h_i) \quad \forall i$$

$$\begin{aligned} h_i &\leq \gamma_{max} * w_i & \forall i \\ h_i &\geq \gamma_{min} * w_i & \forall i \end{aligned}$$

$$\begin{aligned} x_i \geq 0, \quad y_i \geq 0, \quad w_i \geq 0, \quad h_i \geq 0 & \quad \forall i, \\ W \geq 0, \quad H \geq 0 \end{aligned}$$

Resolución en Python con librería cvxpy

```
# Armamos el problema:  
prob = cp.Problem(objective, constraints)  
  
# Resolvemos  
result = prob.solve()  
  
# Obtenemos los valores optimos de perímetro y área:  
print('Perímetro', 2 * prob.value)  
print('Área', H.value[0] * W.value[0])
```

```
>> Perímetro: 167.96  
>> Área: 1752.59
```

