Ejemplo Prototipo: Asignación de Empleados a Proyectos

CONTEXTO

Una gerente de operaciones debe asignar a sus 4 empleados (Ana, Bruno, Carlos y Diana) entre 4 proyectos disponibles (A, B, C y D). Cada empleado tiene habilidades, experiencia y velocidad de ejecución distintas, por lo que un mismo proyecto requiere diferente cantidad de horas según quién lo haga. Cada empleado debe recibir exactamente un proyecto, y cada proyecto debe ser ejecutado por exactamente un empleado. **Objetivo: minimizar el tiempo total de completación.**

MATRIZ DE TIEMPOS (horas por proyecto)

	Proyecto A	Proyecto B	Proyecto C	Proyecto D
Ana	15	10	18	12
Bruno	20	12	15	14
Carlos	18	16	12	10
Diana	14	11	17	13

Cómo leer la matriz:
Si Ana hace el Proyecto A, le toma 15 horas. Si Carlos hace el mismo Proyecto A, le toma 18 horas. *El tiempo depende de quién ejecuta qué.*

PREGUNTA CENTRAL

¿Cómo asignar empleados a proyectos para minimizar el tiempo total? Existen 4! = 24 asignaciones posibles distintas.

¿Por qué no resolverlo por fuerza bruta?

Probar todas las combinaciones posibles funciona para 4 empleados, pero el número de asignaciones crece factorialmente. Muy pronto rebasa cualquier cómputo razonable.

n = 4

$$4! = 24$$

24 combinaciones.
Triviales en una hoja.

n = 10

$$10! = 3{,}628{,}800$$

Más de 3.6 millones.
Minutos de cómputo.

n = 20

$$20! \approx 2.4 \times 10^{18}$$

Más segundos que la edad
del universo.

La consecuencia práctica

Para problemas reales con 20, 50 o 100 agentes, la fuerza bruta es físicamente imposible. Necesitamos un modelo matemático que permita resolver el problema en tiempo polinomial. Esa es exactamente la promesa del **modelo de asignación**: una formulación de programación lineal entera que se resuelve eficientemente.

Asignación: caso especial del problema de transporte

El problema de asignación es estructuralmente un caso particular del problema de transporte que vimos la semana pasada, con tres restricciones especiales que lo hacen aún más fácil de resolver.

TRANSPORTE GENERAL

Cada origen tiene una oferta s_i

Cada destino tiene una demanda d_j

Las cantidades enviadas son **divisibles**

Ejemplo: una planta envía 100 unidades repartidas como 30 a una tienda y 70 a otra. Las cantidades pueden ser fraccionarias.

ASIGNACIÓN

Cada agente ofrece **exactamente 1 unidad**

Cada tarea demanda **exactamente 1 unidad**

Las asignaciones son **todo o nada (0 o 1)**

Consecuencia: *no podemos partir un empleado a la mitad. O hace todo el proyecto, o no lo hace.*

Esta estructura especial garantiza que la solución óptima siempre sea entera, sin necesidad de programación entera explícita.

Formulación matemática del modelo

La traducción de nuestro problema verbal a lenguaje matemático tiene tres componentes: variables de decisión, función objetivo y restricciones.

1. VARIABLES DE DECISIÓN

$x_{ij} = 1$ si el empleado *i* se asigna al proyecto *j*

$x_{ij} = 0$ en caso contrario

2. FUNCIÓN OBJETIVO

$$\text{Min } Z = \Sigma_i \Sigma_j c_{ij} \cdot x_{ij}$$

donde c_{ij} es el tiempo (horas) que tarda el empleado i haciendo el proyecto j.

3. RESTRICCIONES

$\Sigma_j x_{ij} = 1$ para cada empleado i — *Cada empleado se asigna a exactamente un proyecto.*

$\Sigma_i x_{ij} = 1$ para cada proyecto j — *Cada proyecto recibe exactamente un empleado.*

$x_{ij} \in \{0, 1\}$ para todo i, j — *Las variables son binarias.*

En nuestro caso: 16 variables, 8 restricciones de igualdad. Un modelo trivial para Solver.

La propiedad clave: unimodularidad total

Por qué Simplex LP nos da automáticamente soluciones enteras, sin necesidad de programación entera.

La paradoja aparente: definimos las variables como binarias (0 o 1), pero al resolver el modelo en Excel Solver lo configuramos como Simplex LP (programación lineal continua). ***¿Cómo es posible que un método continuo nos devuelva siempre soluciones binarias?***

1. La estructura de la matriz

Las restricciones del problema de asignación tienen una matriz de coeficientes A donde cada entrada es 0 o 1, y cada columna tiene exactamente dos unos: uno por la restricción del empleado y uno por la del proyecto.

2. La propiedad matemática

Esta matriz es totalmente unimodular: todas sus submatrices cuadradas tienen determinante en el conjunto {-1, 0, 1}. Es una propiedad estructural muy fuerte que muy pocos modelos cumplen.

3. La consecuencia práctica

Cuando A es totalmente unimodular y los lados derechos son enteros, todos los vértices del poliedro factible son enteros. Por lo tanto, Simplex LP siempre se detiene en una solución entera, sin esfuerzo adicional.

Beneficio práctico: el problema se resuelve en tiempo polinomial con Simplex LP, mucho más rápido que cualquier método de programación entera general.

Resolución en Excel Solver: estructura de la hoja

Cinco bloques que deben aparecer en la hoja de cálculo. Una vez configurados, Solver hace el resto en milisegundos.

1

Matriz de costos (datos fijos)

Tabla 4×4 con los tiempos c_{ij}. Es información del problema, no se modifica.

2

Matriz de asignaciones (variables)

Tabla 4×4 paralela donde Solver pondrá los valores de x_{ij}. Inicializar todo en cero.

3

Sumas por fila (restricción de empleados)

Una columna auxiliar con =SUMA(...) por cada fila. Solver forzará que cada una sea = 1.

4

Sumas por columna (restricción de proyectos)

Una fila auxiliar con =SUMA(...) por cada columna. Solver forzará que cada una sea = 1.

5

Celda objetivo (función a minimizar)

Una sola celda con =SUMAPRODUCTO(matriz_costos, matriz_asignaciones).

CONFIGURACIÓN DE SOLVER

Objetivo: minimizar la celda objetivo. **Variables:** matriz de asignaciones. **Restricciones:** sumas de fila = 1, sumas de columna = 1.
Método: **Simplex LP** *(no es necesario marcar 'binarias': la unimodularidad lo garantiza por estructura).*

Resolución y solución óptima del modelo

Una vez armado en Excel, Solver encuentra la asignación óptima en milisegundos. La solución para nuestro problema es la siguiente.

ASIGNACIÓN ÓPTIMA

✓	**Ana**	→ Proyecto B	**10 h**
✓	**Bruno**	→ Proyecto D	**14 h**
✓	**Carlos**	→ Proyecto C	**12 h**
✓	**Diana**	→ Proyecto A	**14 h**

Tiempo total mínimo **50 h**



COMPARACIÓN: ¿qué tan buena es la solución óptima?

Asignación ingenua (orden alfabético): Ana→A, Bruno→B, Carlos→C, Diana→D = 15 + 12 + 12 + 13 = **52 h** *(4 % peor)*
Asignación óptima del Solver: Ana→B, Bruno→D, Carlos→C, Diana→A = 10 + 14 + 12 + 14 = **50 h** *(mínimo absoluto)*

Limitaciones del modelo clásico y dónde extenderlo

Honestidad intelectual: el modelo de asignación es elegante pero descansa en supuestos fuertes. Conviene saber dónde aplica y dónde no.

SUPUESTOS QUE EL MODELO IMPONE

- **Costos conocidos y deterministas**

  asume que sabemos exactamente cuánto tarda cada empleado en cada proyecto.

- **Cada agente puede hacer cualquier tarea**

  no contempla habilidades excluyentes ni proyectos que solo ciertos empleados pueden tomar.

- **Asignación uno-a-uno estricta**

  no permite que un empleado tome dos proyectos pequeños o que un proyecto se reparta entre dos personas.

- **Independencia entre proyectos**

  no captura precedencias, dependencias temporales ni efectos de un proyecto sobre otro.

CUÁNDO Y CÓMO EXTENDER EL MODELO

- → **Asignación generalizada (GAP)**

  permite que un agente tome varias tareas hasta agotar una capacidad.

- → **Asignación con habilidades**

  se modela poniendo costo infinito (M grande) en pares empleado–proyecto incompatibles.

- → **Asignación bajo incertidumbre**

  los costos son variables aleatorias; se resuelve con programación estocástica o simulación.

- → **Programación de proyectos (scheduling)**

  cuando hay precedencias y tiempos, el modelo natural es CPM, PERT o programación entera mixta.

Cierre: el problema de asignación es la base conceptual. Las extensiones se construyen modificándolo, no descartándolo. Quien entiende este modelo, entiende los demás.