

# Entrega 1: Formulación Matemática y Generación de Instancias para el Problema de Asignación de Turnos

INF292 – Optimización  
Campus San Joaquín – Grupo 2

## Integrantes:

Fernanda López Saldías	202373638-4
Matías Romo Vargas	202373632-5
Simón Parra Saldías	202373560-4
Vicente Jiménez Sepúlveda	202373523-k

Octubre 2025

## 1. Solucionador y modelo (Entrega 2)

### 1.1. Modelo y notación

Planteamos un modelo de Programación Lineal Entera Binaria cuya función objetivo maximiza la suma de disposiciones asignadas:

$$\begin{aligned} \text{máx} \quad & \sum_{i \in I} \sum_{d \in D} \sum_{t \in T} c_{idt} x_{idt} \\ \text{exts.a.} \quad & \sum_{i \in I} x_{idt} = r_{dt} \end{aligned} \quad \forall d \in D, t \in T \quad (\text{R1})$$

$$x_{idt} = 0 \quad \text{si } c_{idt} = 0 \quad \forall i, d, t \quad (\text{R2})$$

$$\sum_{t \in T} x_{idt} \leq 2 \quad \forall i \in I, d \in D \quad (\text{R3})$$

$$\begin{aligned} x_{i,d,\text{noche}} + x_{i,d+1,\text{mañana}} &\leq 1 & \forall i \in I, d = 1, \dots, |D| - 1 & (\text{R4}) \\ w_{i,k} \in \{0, 1\}, x_{idt} &\in \{0, 1\} & \forall i, d, t & \end{aligned}$$

Donde  $I$  es el conjunto de trabajadores,  $D$  los días y  $T$  los turnos (para instancias *small*:  $T = \{\text{día}, \text{noche}\}$ ; para *medium/large*:  $T = \{\text{mañana}, \text{tarde}, \text{noche}\}$ ). La restricción (R5), “no tres fines de semana consecutivos”, se modela con variables auxiliares  $w_{i,k}$  que indican si el trabajador  $i$  trabaja en el fin de semana  $k$  y con  $w_{i,k} + w_{i,k+1} + w_{i,k+2} \leq 2$ .

**Aclaración sobre R4 en instancias *small*.** En *small* no existe el turno *mañana* como tal, por lo que R4 aplica literalmente solo a *medium/large*. De requerirse, puede activarse una variante “noche→día” en *small* sin alterar el resto del modelo.

## 1.2. Tecnología utilizada

Aunque el enunciado menciona `lpsolve`, en Python 3.12 no hay soporte oficial para `lpsolve55`. Utilizamos **PuLP** con backend **CBC**, equivalente en términos de resolución MILP y totalmente reproducible. El script principal es `Entrega_2/solver/solucionador_de_instancias_lpsolve.py` e implementa:

- Construcción del modelo, lectura de instancias JSON y extracción de solución.
- Lote de ejecución para las 15 instancias y guardado de resultados en JSON.
- Comprobaciones de R1–R5 y extracción de métricas (objetivo, tiempo, estado).

## 1.3. Formato de entrada y salida

`extbfEntrada` (instancias): JSON con campos `trabajadores`, `dias`, `demanda_dias` y `disposicion`.

`extbfSalida`: JSON por instancia con estado (*Optimal/Infeasible*), `valor_objetivo`, `tiempo_resolucion_seg` y el arreglo asignaciones con tuplas (trabajador, día, turno, disposición).

## 1.4. Validez y cobertura

Verificamos automáticamente la cobertura exacta (R1) cruzando demanda esperada y asignaciones para cada día–turno. En *small* también generamos calendarios por trabajador/día que permiten validar visualmente R1 y R3. Las instancias 1–11 y 13–14 son *óptimas*; las 12 y 15 resultan *infactibles* (§2).

# 2. Análisis de resultados

## 2.1. Función objetivo vs tamaño

La Figura 1 resume la relación entre el valor de la función objetivo y el tamaño de instancia. Observamos un crecimiento casi lineal con el número de variables: la correlación con el tamaño agregado es muy alta (*1.000* con tamaño, *0.977* con número de trabajadores y *0.904* con días). La tendencia estimada es  $\hat{y} = 8.78x - 75.43$ .

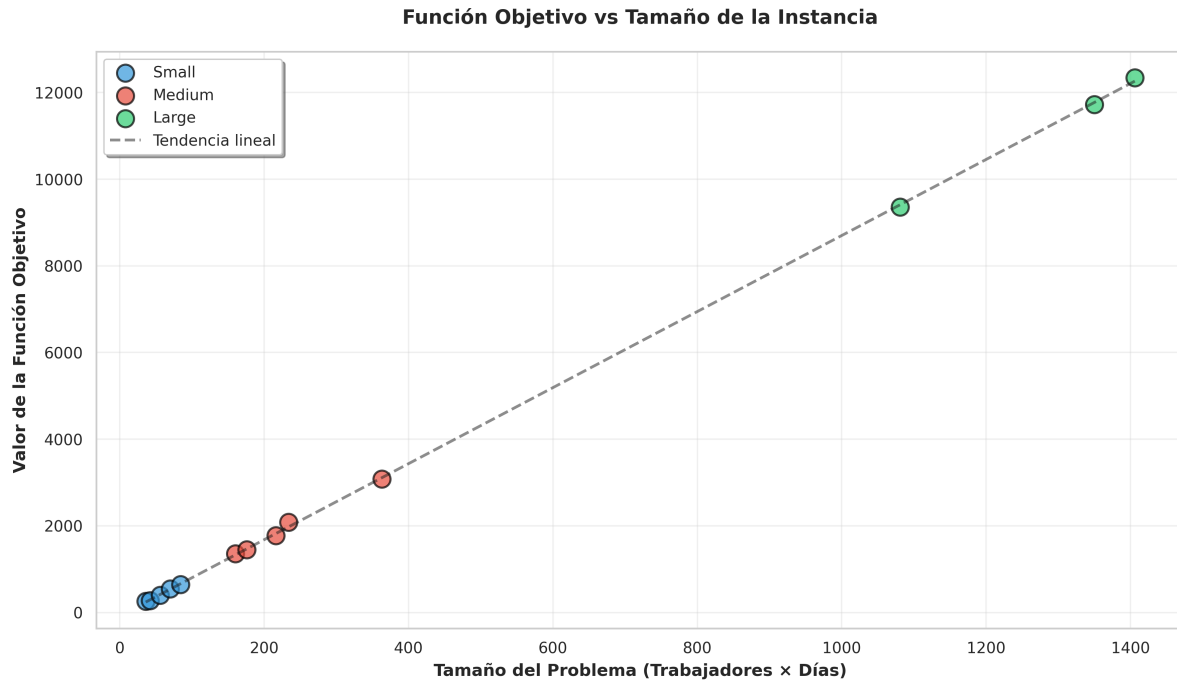


Figura 1: Función objetivo vs tamaño de instancia (small/medium/large).

Complementamos con vistas alternativas: descomposición por trabajadores y días, promedio por tipo y matriz de correlación (Figuras 2 y 3).

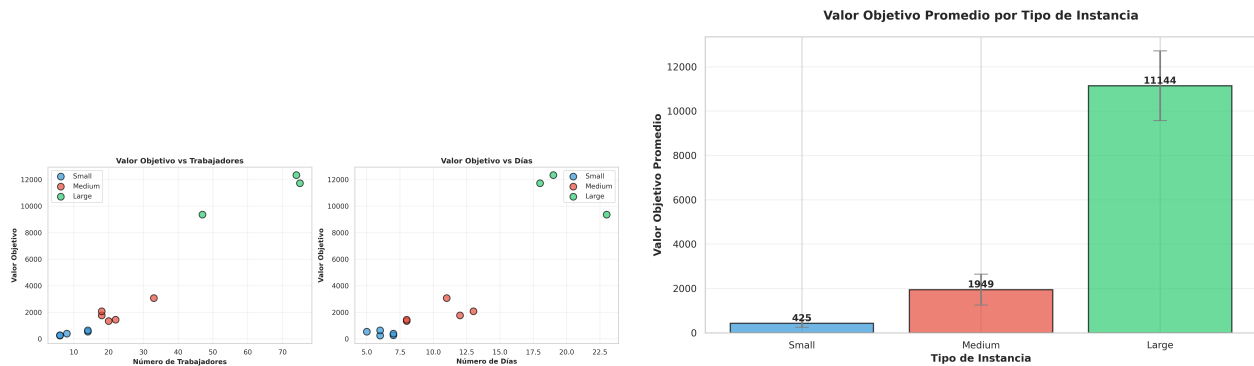


Figura 2: Descomposición de la función objetivo por trabajadores/días y promedio por tipo.

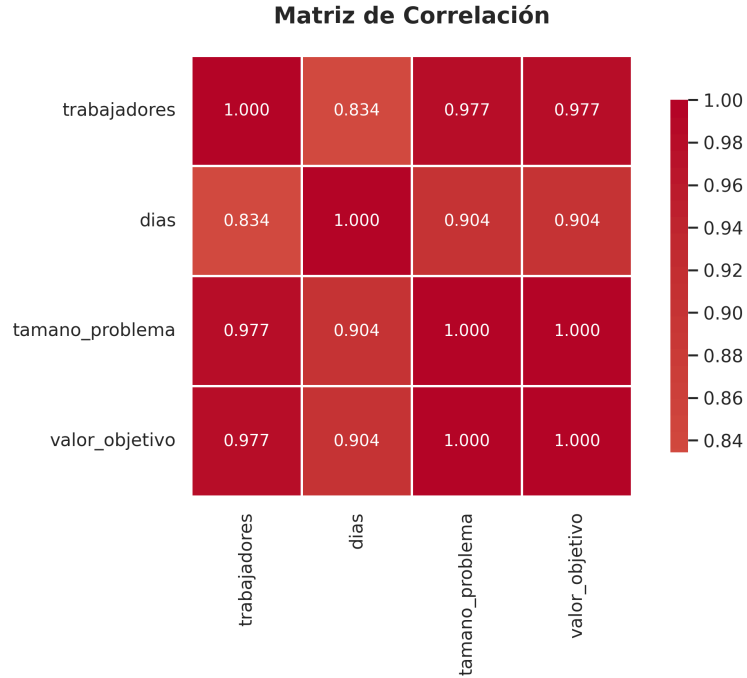


Figura 3: Matriz de correlación entre objetivo y descriptores.

## 2.2. Infactibilidades: instancias 12 y 15

Ambas instancias resultan infactibles. La causa principal es la interacción entre (R1) *cobertura exacta* y (R4) *no noche→mañana*. Aunque la disponibilidad global supera el 90 %, la estructura de turnos y bloqueos secuenciales tras la noche genera cuellos de botella que impiden satisfacer *exactamente* todas las demandas. Se detalla caso a caso en el cuaderno de análisis y en Entrega 2/analisis/factibilidad.md.

## 2.3. Soluciones gráficas (instancias small)

Para las instancias *small* generamos calendarios estilo “trabajador × día” que permiten auditar visualmente (R1) y (R3). La Figura 4 muestra un resumen comparativo; en cada celda se indica el número de turnos asignados (0, 1 o 2). También disponemos de los cinco calendarios individuales.

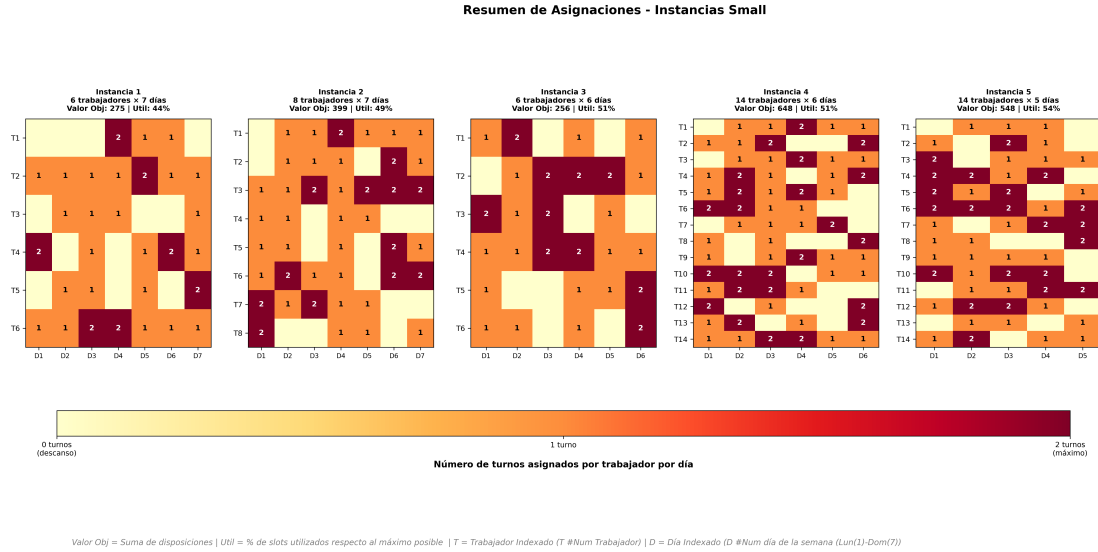


Figura 4: Resumen de asignaciones (instancias small). Cada subgráfico: #trabajadores × #días; color y número indican #turnos por trabajador y día.

**Nota sobre R4 en small.** En estas instancias solo existen los turnos *día* y *noche*; no hay turno *mañana*. Por ello, R4 (no noche→mañana) aplica literalmente a *medium*/*large*. De requerirse, puede activarse una variante “no noche→día” para *small*.

### 3. Tiempos de resolución y escalamiento

Medimos el tiempo de resolución total y por instancia usando CBC. Todas las instancias se resuelven en menos de 0.3s (máximo observado); el total de las 15 instancias fue  $\approx 1.31$  s. Los resultados muestran un crecimiento aproximadamente cuadrático con el tamaño (Figuras 5 y 6).

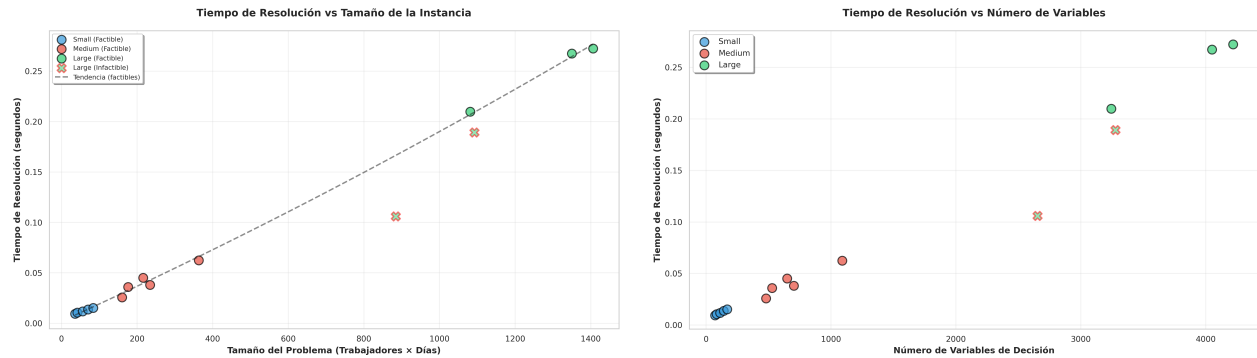


Figura 5: Tiempo de resolución por tamaño y relación con número de variables.

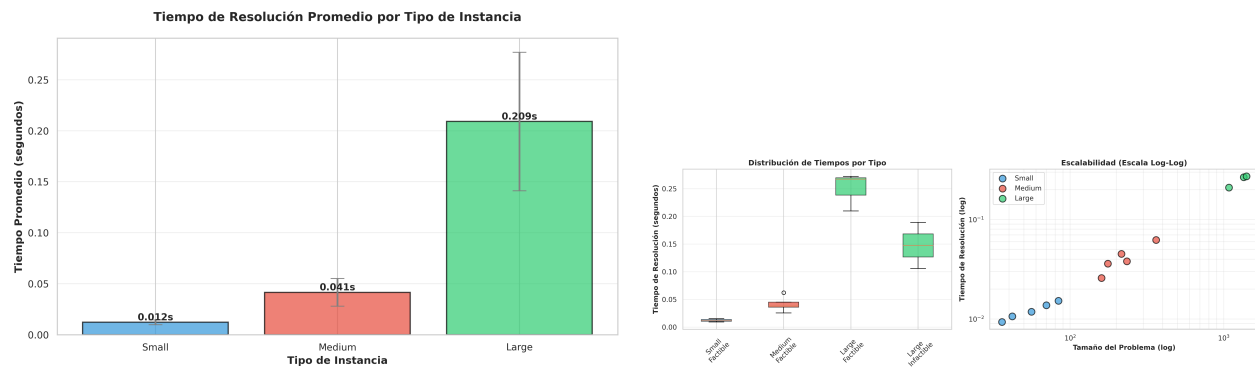


Figura 6: Promedio por tipo y comparación de escalabilidad (log-log).

En términos prácticos, la herramienta es interactiva para el tamaño del problema presentado, y la tendencia sugiere que podemos manejar instancias mayores con tiempos todavía aceptables, especialmente aplicando límites de tiempo o estrategias de arranque si fuese necesario.