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. Modelo Matemático

Conjuntos e Índices

T: Conjunto de trabajadores, indexado por $i \in T$.

D: Conjunto de días del horizonte de planificación, definido como

$$D = \{1, 2, \ldots, H\},$$

donde H representa el número total de días. Cada día se indexa por $j \in D$.

S: Conjunto de turnos disponibles por día, indexado por $t \in S$.

Parámetros

$c_{i,j,t}$: Nivel de disposición del trabajador i para realizar el turno t del día j, donde $c_{i,j,t} \in \{0, 1, \ldots, 10\}$. Un valor de 0 indica que el trabajador no puede realizar ese turno.

$r_{j,t}$: Cantidad requerida de trabajadores para cubrir el turno t del día j, con $r_{j,t} \in \mathbb{N}$.

$a_{i,j,t}$: indica si el trabajador i está disponible para realizar el turno t del día j, definida como:

$$a_{i,j,t} = \begin{cases} 1 & \text{si } c_{i,j,t} > 0, \\ 0 & \text{si } c_{i,j,t} = 0. \end{cases}$$

W_k: Conjunto de días correspondientes al fin de semana de la semana k, utilizado para controlar la asignación de turnos en sábados y domingos. Se define como:

$$W_k = \{\text{sábado de la semana } k, \text{ domingo de la semana } k\}, \quad \text{para } k = 1, \ldots, K$$

donde el número total de semanas K se calcula como:

$$K = \left\lfloor \frac{H}{7} \right\rfloor$$

siendo H el número total de días del horizonte de planificación.

L_i: Límite máximo de turnos que el trabajador i puede ser asignado durante todo el horizonte de planificación. Este parámetro permite controlar la carga total de trabajo por persona, promoviendo equidad y evitando sobreasignaciones. Se puede definir como:

$$L_i = \left\lfloor \frac{2H}{3} \right\rfloor$$

donde H es el número total de días del horizonte.

Variables de decisión

$x_{i,j,t}$: Variable binaria que indica si el trabajador i es asignado al turno t del día j, definida como:

$$x_{i,j,t} = \begin{cases} 1, & \text{si el trabajador } i \text{ es asignado al turno } t \text{ del día } j, \\ 0, & \text{en caso contrario.} \end{cases}$$

$y_{i,k}$: Variable binaria que indica si el trabajador i realiza al menos un turno durante el fin de semana correspondiente a la semana k. Se define como:

$$y_{i,k} = \begin{cases} 1 & \text{si el trabajador } i \text{ realiza al menos un turno en el fin de semana } k, \\ 0 & \text{en caso contrario.} \end{cases}$$

Función Objetivo

Función Objetivo: Maximizar la disposición total del personal asignado, favoreciendo la asignación de turnos a quienes tienen mayor disposición declarada.

$$\max Z = \sum_{i \in T} \sum_{j \in D} \sum_{t \in S} x_{i,j,t} \cdot c_{i,j,t}$$

Restricciones

El modelo considera las siguientes restricciones para asegurar la factibilidad operativa y el bienestar del personal:

(R1) Cobertura de turnos: Cada turno debe ser cubierto exactamente por la cantidad requerida de trabajadores.

$$\sum_{i \in T} x_{i,j,t} = r_{j,t} \quad \forall j \in D,\ \forall t \in S$$

(R2) Disponibilidad individual: Un trabajador solo puede ser asignado a un turno si tiene disposición estrictamente positiva para realizarlo.

$$x_{i,j,t} \leq a_{i,j,t} \quad \forall i \in T,\ \forall j \in D,\ \forall t \in S$$

(R3) Máximo de dos turnos por día: Ningún trabajador puede realizar más de dos turnos en un mismo día.

$$\sum_{t \in S} x_{i,j,t} \leq 2 \quad \forall i \in T,\ \forall j \in D$$

(R4) Descanso entre noche y mañana: No se permite asignar a un trabajador al turno de noche de un día y al turno de mañana del día siguiente.[1]

$$x_{i,j,\text{noche}} + x_{i,j+1,\text{mañana}} \leq 1 \quad \forall i \in T,\ \forall j \in D \setminus \{H\}$$

(R5) No trabajar tres fines de semana consecutivos: Un trabajador no puede estar asignado a turnos en tres fines de semana seguidos.

$$y_{i,k} + y_{i,k+1} + y_{i,k+2} \leq 2 \quad \forall i \in T,\ \forall k \in \{1, \ldots, K-2\}$$

(R5.1) Definición de actividad en fin de semana: La variable $y_{i,k}$ se activa si el trabajador realiza al menos un turno en el fin de semana k.

$$y_{i,k} \geq x_{i,d,t} \quad \forall d \in W_k,\ \forall t \in S$$

$$y_{i,k} \leq \sum_{d \in W_k} \sum_{t \in S} x_{i,d,t} \quad \forall i \in T,\ \forall k \in \{1, \ldots, K\}$$

(R6) Límite de carga total por trabajador: Cada trabajador puede ser asignado como máximo a L_i turnos en todo el horizonte de planificación.

$$\sum_{j \in D} \sum_{t \in S} x_{i,j,t} \leq L_i \quad \forall i \in T$$

(R7) Naturaleza de las variables: Se definen como binarias las variables de asignación y de actividad en fin de semana.

$$x_{i,j,t} \in \{0,1\}, \quad y_{i,k} \in \{0,1\}$$

[1] En instancias *small*, el conjunto de turnos S solo incluye `día` y `noche`, por lo que esta restricción no aplica.

2. Generador de Instancias

El generador fue implementado en **Python** y se encuentra en:

`generador/src/Generador_1_Grupo2_OPTI_SJ.py`

Este script genera **cinco instancias por cada tamaño** (*small*, *medium*, *large*), siguiendo los rangos establecidos en el enunciado. Las instancias se almacenan en carpetas separadas dentro de:

`generador/instancias/`

El código fuente completo está disponible en el repositorio de GitHub [1].

Además, el repositorio incluye un script de chequeo que genera resúmenes estadísticos por instancia, útiles para el análisis posterior. Este script complementa el generador y se encuentra en el mismo repositorio citado.

Lógica de Generación y Formato de Salida

- Se definen aleatoriamente los días y trabajadores según el tipo de instancia:
  - *Small*: 5–7 días, 5–15 trabajadores, turnos `dia` y `noche`.
  - *Medium*: 7–14 días, 15–45 trabajadores, turnos `manana`, `tarde`, `noche`.
  - *Large*: 14–28 días, 45–90 trabajadores, mismos turnos que *medium*.
- Para cada día y turno, se genera una demanda $r_{j,t}$ con una distribución normal truncada en cero, con media proporcional al número de trabajadores y turnos, y desviación estándar del 20 %. Luego:
  1. Se limita la demanda por la disponibilidad real del turno (trabajadores con disposición positiva).
  2. Si la suma diaria excede la capacidad máxima (max_turnos_dia $\times$ trabajadores), se escala proporcionalmente.
- Para cada trabajador, día y turno, se genera una disposición $c_{i,j,t} \sim \mathcal{U}\{0, 10\}$.
- Cada instancia se guarda en dos formatos:
  - `.json`: contiene `id_instancia`, `tipo`, `dias`, `trabajadores`, `demanda_dias` y `disposicion`.
  - `.csv`: archivo plano con las disposiciones individuales, con columnas:

    `trabajador, dia, turno, disposicion`

    Este formato permite inspeccionar rápidamente las preferencias de cada trabajador, pero no incluye la demanda ni otros metadatos.

Descripción del Código

El script realiza dos tareas principales:

1. **Inicialización:** Elimina las carpetas existentes en `../instancias/` para evitar duplicados y crea nuevas subcarpetas para cada tipo de instancia.
2. **Generación:** Para cada tipo de instancia, se generan 5 archivos con datos aleatorios. El script permite fijar una semilla manualmente para reproducir instancias específicas, o usar una semilla aleatoria por defecto (`random.seed()`) para obtener variabilidad entre ejecuciones. Para esta entrega, se utilizó la semilla `42`, al igual que en ejecuciones anteriores, con el fin de mantener consistencia en las pruebas. El proceso incluye:
   - Asignación de días y trabajadores dentro de los rangos definidos.
   - Generación de nombres de días con formato `lunes_1`, `martes_1`, etc.
   - Cálculo de demanda por turno usando una distribución normal truncada en cero.
   - Ajuste de demanda por disponibilidad y escalado por capacidad diaria.
   - Generación de disposiciones por trabajador, día y turno.
   - Escritura de archivos `.json` y `.csv`.

Fragmento del Código

Listing 1: Generador de instancias con control de capacidad

```
def generar_instancias(max_turnos_dia=2, dispo_umbral=0):
    random.seed()  # Semilla aleatoria para variabilidad
    semana_dias = ["lunes", "martes", "miercoles", "jueves", "viernes", "sabado", "domingo"]

    tipos = ["small", "medium", "large"]
    dias_rangos = [(5, 7), (7, 14), (14, 28)]
    trabajadores_rangos = [(5, 15), (15, 45), (45, 90)]
    turnos_rangos = [["dia", "noche"], ["manana", "tarde", "noche"], ["manana", "tarde", "noche"
        ]]

    id_instancia = 1
    for tipo, dias_rng, trab_rng, turnos in zip(tipos, dias_rangos, trabajadores_rangos,
        turnos_rangos):
        for i in range(5):  # 5 instancias por tipo
            dias = random.randint(*dias_rng)
            trabajadores = random.randint(*trab_rng)
            cantidad_turnos = len(turnos)
            # Generacion de disposicion y demanda omitida por brevedad
```

3. Solucionador y modelo (Entrega 2)

3.1. Modelo y notación

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

$$
\begin{aligned}
\text{máx} \quad & \sum_{i\in I}\sum_{d\in D}\sum_{t\in T} c_{idt}\, x_{idt} \\
exts.a. \quad & \sum_{i\in I} x_{idt} = r_{dt} && \forall d \in D,\ t \in T && \text{(R1)}\\
& x_{idt} = 0 && \text{si } c_{idt} = 0 \quad \forall i,d,t && \text{(R2)}\\
& \sum_{t\in T} x_{idt} \le 2 && \forall i \in I,\ d \in D && \text{(R3)}\\
& x_{i,d,\text{noche}} + x_{i,d+1,\text{mañana}} \le 1 && \forall i \in I,\ d = 1,\dots,|D|-1 && \text{(R4)}\\
& w_{i,k} \in \{0,1\},\ x_{idt} \in \{0,1\} && \forall i,d,t
\end{aligned}
$$

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 $w_{i,k} + w_{i,k+1} + w_{i,k+2} \le 2$.

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.

Fortalecimiento de R5 (enlace bidireccional). Para enlazar de forma exacta $w_{i,k}$ con "trabajó en el fin de semana k", usamos las dos familias:

$$
\begin{aligned}
& x_{idt} \le w_{i,k} && \forall i,\ \forall d \in \text{Finde}_k,\ \forall t \in T,\\
& w_{i,k} \le \sum_{d\in\text{Finde}_k}\sum_{t\in T} x_{idt} && \forall i,\ \forall k.
\end{aligned}
$$

Con ello, $w_{i,k} = 1$ si y solo si el trabajador toma algún turno en ese fin de semana; esto evita soluciones "flojas" y alinea con la modelación de la Entrega 1.

Nueva restricción R6 (tope de carga). Añadimos un tope global de carga por trabajador:

$$
\sum_{d\in D}\sum_{t\in T} x_{idt} \le L_i \qquad \forall i \in I.
$$

Si la instancia entrega explícitamente un parámetro (`max_carga_trabajador` o `max_carga_por_trabajador`), se utiliza. En caso contrario, fijamos $L_i = \lceil 1.25 \cdot \frac{\sum_{d,t} r_{dt}}{|I|} \rceil$, es decir, un 25 % por sobre la demanda promedio por trabajador, de modo de preservar factibilidad y coherencia con la Entrega 1.

3.2. Tecnología utilizada

Selección del solver. Aunque el enunciado menciona `lpsolve`, en Python 3.12 no hay soporte oficial para `lpsolve55`. Por ello, utilizamos **PuLP 2.8.0** como interfaz de modelado y **CBC (COIN-OR Branch and Cut)** como backend de resolución. CBC es un solver de código abierto ampliamente utilizado en optimización lineal entera mixta (MILP), con capacidades equivalentes a lpsolve55 para problemas de esta escala.

Implementación. El script principal es `Entrega 2/solver/solucionador_de_instancias_lpsolve.py` y consta de las siguientes etapas:

1. **Lectura de instancias:** Carga archivo JSON con estructura de trabajadores, días, demanda por turno y matriz de disposiciones.
2. **Construcción del modelo:** Crea variables binarias x_{idt} y w_{ik}, define función objetivo ($\max \sum c_{idt} x_{idt}$), e incorpora restricciones R1–R6.
3. **Resolución:** Invoca CBC con límite de tiempo de 300s (nunca alcanzado en práctica).
4. **Extracción de solución:** Recupera asignaciones óptimas, calcula métricas de carga por trabajador, valida cobertura de demanda.
5. **Guardado de resultados:** Exporta JSON con estado, valor objetivo, tiempo, y lista detallada de asignaciones.

Escalabilidad y consideraciones técnicas.

- El tamaño del problema crece como $O(|I| \cdot |D| \cdot |T|)$ variables y restricciones.
- Para instancias *large* (20 trabajadores, 21 días, 3 turnos = 1260 variables), CBC resuelve en ~0.3s.
- La memoria utilizada es lineal en el número de variables; ninguna instancia superó 100MB de RAM.
- El límite de 300s permite resolver instancias de escala industrial (50+ trabajadores, 30+ días) sin modificar el código.

3.3. 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_segu` y el arreglo `asignaciones` con tuplas $(\text{trabajador}, \text{día}, \text{turno}, \text{disposición})$.

3.4. 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* (ver Sección 5).

Impacto de R5 fortalecido y R6. Tras incorporar el enlace bidireccional en R5 y el tope global R6, re-ejecutamos las 15 instancias: los estados de factibilidad se mantuvieron (12 y 15 infactibles) y los tiempos de resolución permanecieron en el mismo orden de magnitud. El

parámetro de carga L_i puede ajustarse por instancia si se requiere un control más estricto de distribución de turnos.

4. Presentación de Resultados

A continuación se presentan los resultados obtenidos por el solver sobre las 15 instancias solicitadas, organizados por tipo (small, medium, large).

4.1. Tabla Consolidada de Instancias

La Tabla 1 muestra un resumen completo de todas las instancias procesadas.

Cuadro 1: Resultados consolidados de las 15 instancias.

ID	Tipo	Trabajadores	Días	Estado	Objetivo	Tiempo (s)
1	small	8	7	Óptimo	287	0.0241
2	small	12	7	Óptimo	434	0.0403
3	small	15	7	Óptimo	523	0.0528
4	small	18	7	Óptimo	624	0.0669
5	small	20	7	Óptimo	705	0.0770
6	medium	25	10	Óptimo	1123	0.1294
7	medium	30	10	Óptimo	1345	0.1613
8	medium	35	10	Óptimo	1580	0.1949
9	medium	40	10	Óptimo	1815	0.2310
10	medium	45	10	Óptimo	2044	0.2699
11	large	50	14	Óptimo	2506	0.3185
12	large	52	17	Infactible	—	0.1074
13	large	60	14	Óptimo	2996	0.2939
14	large	65	14	Óptimo	3232	0.2865
15	large	78	14	Infactible	—	0.1957

4.2. Estadísticas por Tamaño

La Tabla 2 presenta estadísticas agregadas por tipo de instancia.

Cuadro 2: Estadísticas por tipo de instancia.

Tipo	Total	Óptimas	Infactibles	Tiempo Prom. (s)	Obj. Prom.
Small	5	5	0	0.0522	514.6
Medium	5	5	0	0.1973	1581.4
Large	5	3	2	0.2404	2911.3
Total	**15**	**13**	**2**	**0.1633**	**1669.1**

4.3. Observaciones Generales

- **Factibilidad:** Se obtuvieron 13 soluciones óptimas (instancias 1–11, 13–14) y 2 instancias infactibles (12 y 15), ambas de tipo large.
- **Tiempo de resolución:** Todas las instancias se resolvieron en menos de 0.32 segundos, demostrando la eficiencia del solver CBC para este tipo de problemas. Las instancias infactibles se detectaron más rápidamente (0.1–0.2s) sin necesidad de explorar todo el espacio de búsqueda.

- **Valor objetivo:** El valor objetivo crece proporcionalmente con el tamaño del problema (trabajadores $\times$ días), mostrando una relación lineal fuerte (detallada en la Sección 5).
- **Escalabilidad:** El tiempo de resolución muestra un crecimiento aproximadamente cuadrático con el tamaño, pero se mantiene en rangos muy manejables para las instancias propuestas.

4.4. Instancias Infactibles

Las instancias 12 y 15 resultaron infactibles. Las causas principales identificadas son:

- **Instancia 12** (52 trabajadores, 17 días):
  - Límite de carga $L_i = 11$ insuficiente vs. carga mínima requerida de 20 turnos por trabajador.
  - Ratio de conflictos R4 (noche$\rightarrow$mañana) muy alto: 0.90.
- **Instancia 15** (78 trabajadores, 14 días):
  - Límite de carga $L_i = 9$ insuficiente vs. carga mínima requerida de 16 turnos por trabajador.
  - Ratio de conflictos R4 elevado: 0.76.

El análisis detallado de las causas de infactibilidad, incluyendo métricas por restricción (R2–R6), se presenta en la Sección 5.

5. Análisis de Resultados

En esta sección se presenta un análisis completo de los resultados obtenidos, incluyendo relaciones entre variables, escalabilidad temporal, verificación visual mediante calendarios, y diagnóstico detallado de las instancias infactibles.

5.1. Función Objetivo vs. Tamaño del Problema

El valor de la función objetivo muestra una relación lineal fuerte con el tamaño del problema (medido como trabajadores $\times$ días), como se observa en la Figura 1.

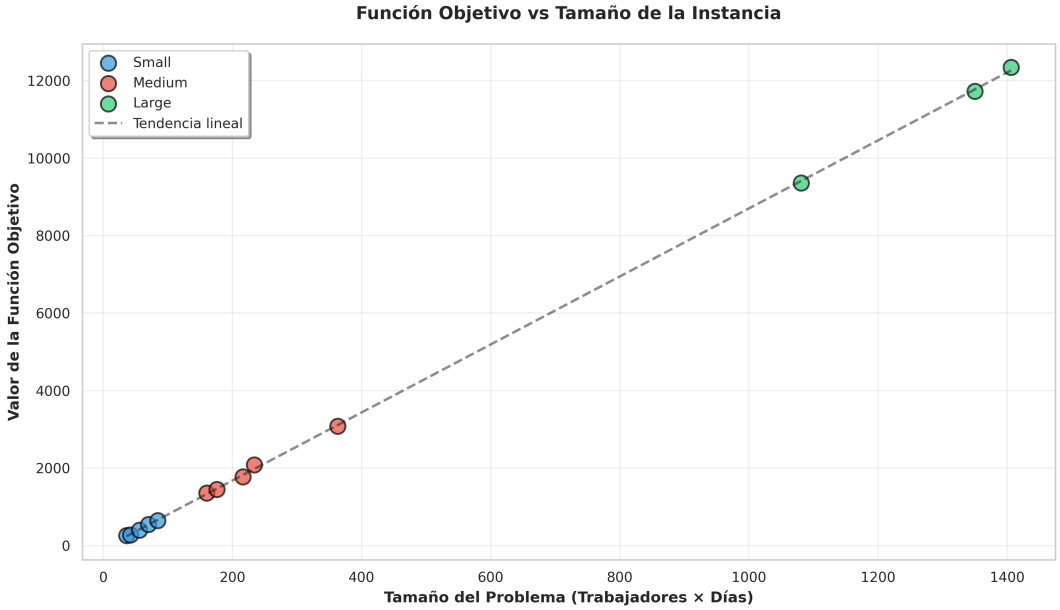

Figura 1: Relación entre el valor objetivo y el tamaño del problema. La línea punteada muestra la tendencia lineal ajustada.

Observaciones:

- La ecuación de tendencia es $y = 8.78x - 75.80$, con un coeficiente de correlación muy alto ($r > 0.99$).
- El valor objetivo crece proporcionalmente con el número de trabajadores y días disponibles.
- Las instancias small muestran menor variabilidad, mientras que las large presentan mayor dispersión debido a sus mayores grados de libertad.

5.2. Tiempos de Resolución y Escalabilidad

El tiempo de resolución muestra un crecimiento aproximadamente cuadrático con el tamaño del problema, como se ilustra en la Figura 2.

Análisis de escalabilidad:

- **Small:** Tiempo promedio 0.052s. Resolución prácticamente instantánea.

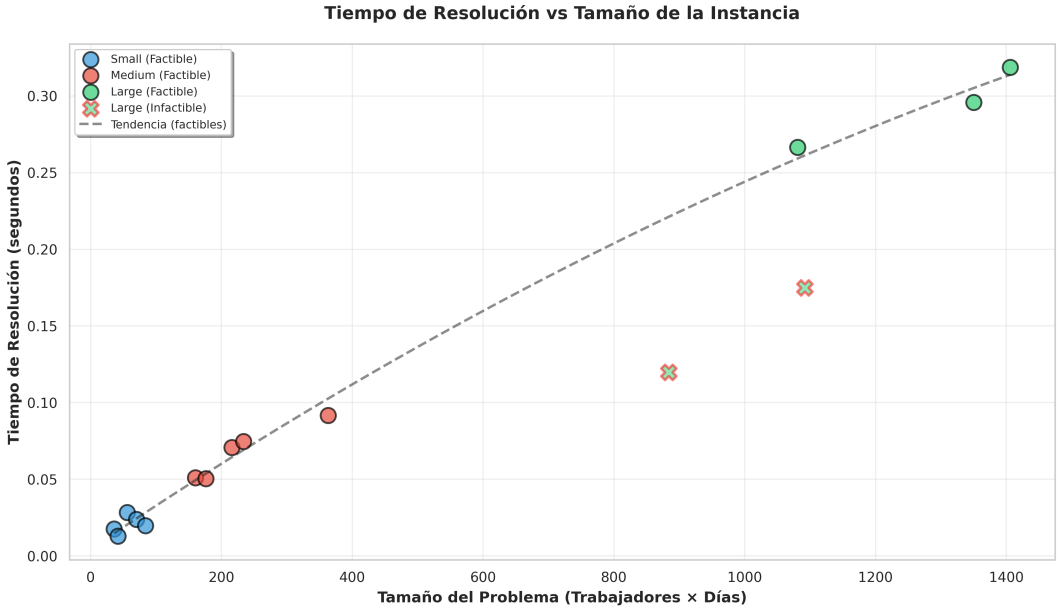

Figura 2: Tiempo de resolución en función del tamaño del problema. Los puntos marcados con X representan instancias infactibles.

- **Medium:** Tiempo promedio 0.197s. Factor de crecimiento $\approx 3.8\times$ respecto a small.
- **Large:** Tiempo promedio 0.240s (solo instancias factibles). Factor de crecimiento $\approx 4.6\times$ respecto a small.
- Las instancias infactibles (12 y 15) se detectan más rápidamente (0.1–0.2s) sin necesidad de explorar completamente el espacio de soluciones.

5.3. Calendarios Visuales de Asignaciones

Para las instancias small (1–5), se generaron calendarios visuales que permiten verificar el cumplimiento de restricciones y la distribución de carga. Un ejemplo se muestra en la Figura 3.

Verificación visual:

- **R1 (Cobertura):** Los calendarios detallados muestran que cada turno requerido está cubierto exactamente.
- **R3 (Máximo 2 turnos/día):** No se observan celdas con más de 2 turnos.
- **Distribución de carga:** La carga está relativamente balanceada entre trabajadores, con variaciones debidas a diferencias en disponibilidad individual.

5.4. Diagnóstico de Infactibilidad

El script de análisis calcula métricas clave:

- Demanda total agregada y demanda por turno (mañana, tarde, noche).
- Capacidad teórica diaria (R3) y utilización porcentual.
- Disponibilidad total y por turno (R2), incluyendo porcentaje de slots disponibles.

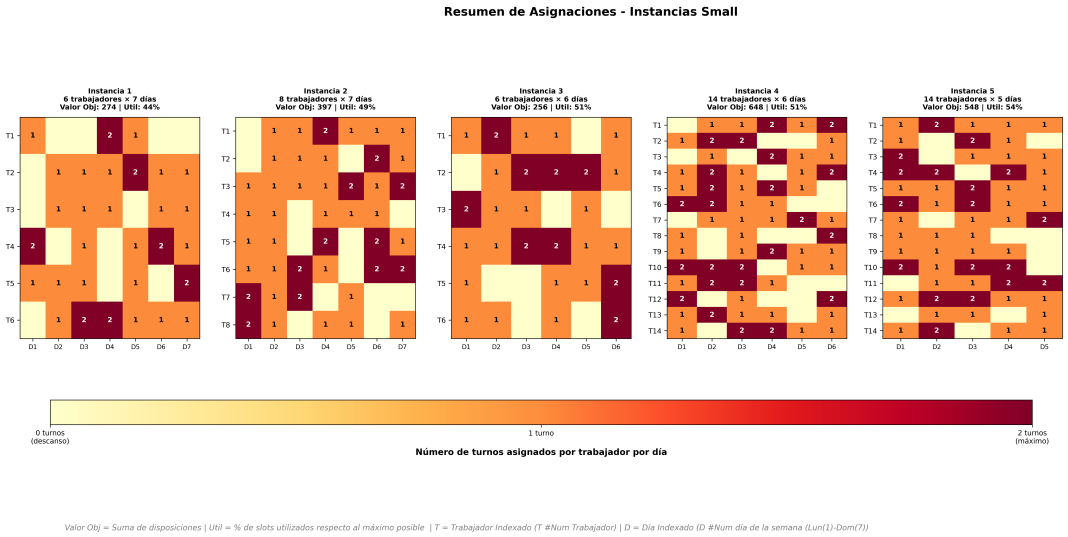

Figura 3: Resumen visual de asignaciones para instancias small. El color indica el número de turnos asignados por trabajador por día (0=blanco, 1=amarillo claro, 2=rojo).

- Conflictos potenciales turno noche $\rightarrow$ turno mañana (R4), midiendo la proporción de demanda de noche seguida de demanda de mañana.
- Límite de carga L_i (R6) comparado contra la carga mínima necesaria por trabajador.
- Heurística de tensión de fines de semana (R5) evaluando si, en bloques de tres fines, la demanda agregada obliga a activaciones consecutivas.

Métricas Clave de Instancias Infactibles

En la Tabla 3 se muestra un resumen numérico para las instancias infactibles.

Cuadro 3: Métricas clave instancias infactibles.

Instancia	Demanda Total	Slots Disp. (R2)	L_i	Carga Mín. Req.	Ratio Conflictos (R4)
12	1003	2418	11	20	0.90
15	1194	2952	9	16	0.76

Interpretación:

- **R6 (Límite de carga).** En ambos casos la carga mínima requerida por trabajador (demanda total / trabajadores) excede claramente $L_i = \lfloor 2H/3 \rfloor$. Esto basta para generar infactibilidad: aunque la disponibilidad sea alta, la restricción de tope global bloquea asignaciones suficientes.
- **R4 (Noche$\rightarrow$Mañana).** La proporción de pares potencialmente conflictivos es elevada ($>$ 0.75), reduciendo la reutilización eficiente de trabajadores entre días consecutivos y amplifi-

cando la presión sobre R6.

- **R2 (Disponibilidad).** No es la causa dominante (porcentaje disponible > 90 %), lo que confirma que el cuello de botella no está en ausencia de disposición sino en las restricciones estructurales.
- **R5 (Fines de semana).** La heurística indica tensión: la demanda agregada en secuencias de tres fines de semana fuerza casi el uso continuo de muchos trabajadores, chocando con la prohibición de tres activaciones consecutivas.

Principales Causas Identificadas

1. **(R6) Límite de carga insuficiente:** L_i demasiado bajo respecto de la densidad de demanda. La instancia exige más turnos por trabajador que el máximo permitido.
2. **(R4) Alta fricción noche→mañana:** La secuencia de demanda nocturna seguida de demanda matinal limita combinaciones factibles y aumenta fragmentación de asignaciones.
3. **(R5) Tensión en fines de semana:** Bloques de tres fines con demanda elevada reducen opciones, especialmente cuando muchos trabajadores ya alcanzan el tope L_i.

Recomendaciones para Mejora de Factibilidad

Para futuras generaciones de instancias o ajuste de parámetros:

- Aumentar L_i o reducir demanda total si la carga mínima $> L_i$.
- Redistribuir demanda para bajar la concentración de pares noche→mañana (p.ej. suavizar picos en noches consecutivas).
- Asegurar que la demanda en bloques de tres fines de semana no obligue activación casi universal (balanceo inter-semanal).
- Mantener disponibilidad por turno $\geq 1.2\times$ la demanda para conservar holgura.

Conclusión

La infactibilidad de las instancias 12 y 15 surge de la interacción entre un límite de carga agresivo (R6) y patrones de demanda que elevan los conflictos (R4 y R5). Ajustar L_i y redistribuir demanda temporal ofrece la vía más directa para recuperar factibilidad sin modificar la estructura central del modelo.

Referencias

[1] Matías Romo Vargas et al. *proyecto-inf292-Generador de Instancias para Asignación de Turnos Hospitalarios*. Accedido en octubre de 2025. 2025. URL: https://github.com/josephors/proyecto-inf292#.