Taller 2 — Extensiones del Job Shop Scheduling Problem en MiniZinc

Mantenimiento programado y Tardanza ponderada

John Freddy Belalcázar
Samuel Galindo Cuevas
Nicolás Herrera Marulanda

2 de noviembre de 2025

Índice

Repositorio del proyecto

Código fuente, instancias, scripts y PDF están disponibles en:
`https://github.com/Herreran903/taller-2-restricciones`

1. Jobshop Mantenimiento

Planificación en un taller con trabajos, cada uno como una secuencia de operaciones que deben ejecutarse en máquinas específicas; cada máquina procesa a lo sumo una operación a la vez y las operaciones de un mismo trabajo respetan su orden. Se entregan intervalos de *mantenimiento* por máquina durante los cuales no están disponibles, y el objetivo es calendarizar todas las operaciones evitando solapes en la misma máquina y cumpliendo precedencias y ventanas de mantenimiento, de modo que se minimice el *makespan*.

1.1. Modelo

Parámetros

P1 — `JOBS`: Cantidad de trabajos.

P2 — `TASKS`**:** Cantidad de máquinas.

P3 — `PROC_TIME`**:** Matriz de duraciones $p_{i,m}$ de tamaño `JOBS` $\times$ `TASKS`: `PROC_TIME`$[i, m] = p_{i,m}$.

P4 — `MAX_MAINT_WINDOWS`**:** Tope global de ventanas de mantenimiento por máquina.

P5 — `MAINT_START`, `MAINT_END`**:** Inicios $a_{m,k}$ y fines $b_{m,k}$ de cada ventana k en máquina m.

P6 — `MAINT_ACTIVE`**:** Indicadores booleanos `MAINT_ACTIVE`$[m, k]$ que activan la ventana $[a_{m,k}, b_{m,k})$.

Constantes derivadas

D1 — H**:** Horizonte superior *seguro*, derivado como

$$H \;=\; \sum_{i=1}^{\texttt{JOBS}} \sum_{m=1}^{\texttt{TASKS}} p_{i,m} \;+\; \sum_{m=1}^{\texttt{TASKS}} \sum_{k=1}^{\texttt{MAX\_MAINT\_WINDOWS}} (b_{m,k} - a_{m,k})\, \mathbf{1}[\texttt{MAINT\_ACTIVE}[m, k]]\,.$$

D2 — J**:** Conjunto de trabajos, $J = \{1, \ldots, \texttt{JOBS}\}$.

D3 — M**:** Conjunto de máquinas, $M = \{1, \ldots, \texttt{TASKS}\}$.

Variables

V1 — $s_{i,m}$**:** Inicio de la operación del trabajo i en la máquina m, con $s_{i,m} \in [0, H]$.

V2 — `END`**:** *Makespan* del programa, `END` $\in [0, H]$.

Restricciones principales

R1 — **Precedencias dentro del trabajo:** Las operaciones de cada trabajo siguen su orden dado.

$$\forall i \in J,\ \forall m \in \{1, \ldots, |M| - 1\} : \quad s_{i,m} + p_{i,m} \;\le\; s_{i,m+1}, \qquad s_{i,|M|} + p_{i,|M|} \;\le\; \texttt{END}.$$

R2 — **No solape por máquina:** En cada máquina, las operaciones se procesan de a una (restricción disyuntiva).

$$\forall m \in M,\ \forall i, k \in J,\ i < k : \quad (s_{i,m} + p_{i,m} \le s_{k,m}) \;\vee\; (s_{k,m} + p_{k,m} \le s_{i,m}).$$

R3 — **Bloqueos por mantenimiento:** Ninguna operación se ejecuta durante una ventana activa de mantenimiento.

$$\forall m \in M,\ \forall k \in \{1, \ldots, \texttt{MAX\_MAINT\_WINDOWS}\} \text{ con } \texttt{MAINT\_ACTIVE}[m, k] = \texttt{true},\ \forall i \in J : \\ (s_{i,m} + p_{i,m} \le a_{m,k}) \;\vee\; (b_{m,k} \le s_{i,m})\,,$$

donde $a_{m,k} = \texttt{MAINT\_START}[m, k]$ y $b_{m,k} = \texttt{MAINT\_END}[m, k]$, con $0 \le a_{m,k} < b_{m,k} \le H$.

Restricciones redundantes

R4 — **Cota por trabajo:** El *makespan* no puede ser menor que la suma de duraciones de cada trabajo.

$$\forall i \in J : \quad \texttt{END} \;\ge\; \sum_{m \in M} p_{i,m}.$$

R5 — **Carga por máquina:** El *makespan* acota inferiormente la carga total de cada máquina.

$$\forall m \in M : \quad \texttt{END} \;\ge\; \sum_{i \in J} p_{i,m}.$$

R6 — Cota por horizonte: Las fechas de inicio y el *makespan* se restringen al horizonte H.

$$\forall i \in J,\ \forall m \in M:\ 0 \le s_{i,m} \le H, \qquad 0 \le \texttt{END} \le H.$$

Restricciones de simetría

R7 — Trabajos idénticos: Para evitar permutaciones equivalentes, si $p_{i,*} = p_{k,*}$ y $i < k$ se impone orden léxico en los inicios:

$$(s_{i,1}, \ldots, s_{i,|M|}) \le_{\text{lex}} (s_{k,1}, \ldots, s_{k,|M|}).$$

1.2. Implementación

Modelo

El modelo captura de forma correcta la estructura del problema mediante los parámetros definidos en la sección anterior y las restricciones principales $R1$–$R3$. Las variables $s_{i,m}$ y `END` permiten representar explícitamente el instante de inicio y finalización de cada operación, de modo que cualquier configuración factible de estas variables corresponde a un cronograma real. La restricción $R1$ asegura la correcta secuencia de operaciones dentro de cada trabajo, preservando el orden tecnológico sin permitir solapamientos entre tareas consecutivas del mismo job. La restricción $R2$ implementa la capacidad unitaria de cada máquina, garantizando que solo una operación se ejecute a la vez en ella; esto se logra mediante la disyunción de no solape, lo que define implícitamente un orden válido entre operaciones que comparten recurso. Finalmente, $R3$ extiende el modelo clásico incorporando las ventanas de mantenimiento: los intervalos definidos por `MAINT_START`, `MAINT_END` y activados por `MAINT_ACTIVE` se tratan como periodos de *no disponibilidad* de la máquina, impidiendo que cualquier operación se solape con ellos.

Restricciones redundantes

Las restricciones $R4$–$R6$ son **lógicamente implicadas** por las restricciones principales del modelo, es decir, no añaden información nueva sobre el conjunto de soluciones factibles. Su propósito es **reforzar la propagación** de cotas durante la búsqueda, ayudando al solver a podar ramas del árbol de exploración más temprano. $R4$ (cota por trabajo) obliga a `END` a ser al menos la suma de duraciones de cada trabajo individual, descartando de inmediato valores imposibles del objetivo. $R5$ (carga por máquina) exige que `END` no sea menor que la carga total de trabajo acumulada en cada máquina. $R6$ acota todas las variables al horizonte seguro H; en la implementación esta cota se aplica de forma implícita mediante los dominios $[0, H]$ definidos para las variables $s_{i,m}$ y `END`.

Restricciones de simetría

Cuando existen trabajos con la misma secuencia de duraciones ($\texttt{PROC\_TIME}[i,*] = \texttt{PROC\_TIME}[k,*]$), el problema admite soluciones equivalentes donde solo se intercambian las etiquetas de estos trabajos idénticos. La restricción $R7$ **rompe esta simetría** imponiendo un orden léxico sobre los vectores de tiempos de inicio $(s_{i,1}, \ldots, s_{i,|M|})$ para cada par de trabajos idénticos $i < k$. De este modo, se conserva un único representante canónico por cada clase de permutación equivalente. Dado que los trabajos idénticos son intercambiables sin afectar la optimalidad del makespan, esta restricción **elimina soluciones estructuralmente idénticas sin perder la solución óptima**, por lo que no es necesario un proceso posterior para recuperar soluciones.

Estrategias de búsqueda exploradas

Se experimentó con distintas **estrategias de búsqueda**, variando tanto la heurística de selección de variables como la de asignación de valores. Las estrategias consideradas fueron:

- `first_fail + indomain_min`: selecciona la variable con el dominio más pequeño y le asigna el valor mínimo posible.
- `dom_w_deg + indomain_split`: prioriza variables con alto peso de restricciones insatisfechas (mayor *grado ponderado*) y divide su dominio, aprovechando información sobre conflictos acumulados.
- `input_order + indomain_min`: asigna variables en el orden en que fueron declaradas, con el valor mínimo.

1.3. Pruebas

Con el fin de evaluar el desempeño del modelo propuesto, se llevaron a cabo pruebas sobre 10 instancias distintas del problema, numeradas como `test_01` a `test_10`, cada una con configuraciones variadas de trabajos, máquinas y ventanas de mantenimiento. Para cada instancia, se analizaron múltiples combinaciones de solver y estrategias de búsqueda. Las ejecuciones se realizaron tanto con como sin restricciones redundantes y de simetría. Se impuso un límite de tiempo de 60 segundos por ejecución, por lo que en algunas instancias el solver alcanzó dicho umbral sin concluir la búsqueda completa.

Tabla 1: Resultados de pruebas **con** restricciones redundantes.

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth
test_01	gecode	first_fail	indomain_min	7.862	68	20	13
test_02	gecode	first_fail	indomain_min	508.427	264886	132435	28
test_03	gecode	first_fail	indomain_min	73.898	38172	19054	25
test_04	gecode	first_fail	indomain_min	21869.202	12299388	6149673	39
test_05	gecode	first_fail	indomain_min	17700.848	8583304	4291610	40
test_06	gecode	first_fail	indomain_min	20.877	9098	4520	26
test_07	gecode	first_fail	indomain_min	22.030	9125	4554	23
test_08	gecode	first_fail	indomain_min	43979.030	12705667	6352774	39
test_09	gecode	first_fail	indomain_min	3900.950	1514760	757354	41
test_10	gecode	first_fail	indomain_min	59937.474	20139317	10069609	52
test_01	gecode	dom_w_deg	indomain_split	0.960	113	34	34
test_02	gecode	dom_w_deg	indomain_split	9.232	3150	1549	71
test_03	gecode	dom_w_deg	indomain_split	3.973	959	451	69
test_04	gecode	dom_w_deg	indomain_split	28.810	7536	3729	88
test_05	gecode	dom_w_deg	indomain_split	11.892	2338	1114	119
test_06	gecode	dom_w_deg	indomain_split	4.120	994	455	90
test_07	gecode	dom_w_deg	indomain_split	4.192	1283	628	68
test_08	gecode	dom_w_deg	indomain_split	11.540	2173	980	128
test_09	gecode	dom_w_deg	indomain_split	36.976	2702	1311	71
test_10	gecode	dom_w_deg	indomain_split	370.060	41970	20857	180
test_01	gecode	input_order	indomain_min	0.821	200	94	13
test_02	gecode	input_order	indomain_min	10240.779	8250996	4125491	25
test_03	gecode	input_order	indomain_min	281.118	210561	105269	21
test_04	gecode	input_order	indomain_min	58975.739	33415666	16707816	32
test_05	gecode	input_order	indomain_min	59935.589	35953071	17976510	39
test_06	gecode	input_order	indomain_min	856.992	535610	267795	26
test_07	gecode	input_order	indomain_min	173.459	114108	57052	23
test_08	gecode	input_order	indomain_min	59939.545	37098935	18549449	36
test_09	gecode	input_order	indomain_min	10316.886	4784339	2392147	35
test_10	gecode	input_order	indomain_min	59940.290	31019998	15509979	50
test_01	chuffed	first_fail	indomain_min	2.000	68	12	14
test_02	chuffed	first_fail	indomain_min	8.000	537	389	22
test_03	chuffed	first_fail	indomain_min	6.000	675	373	22
test_04	chuffed	first_fail	indomain_min	37.000	2863	2008	26
test_05	chuffed	first_fail	indomain_min	45.000	811	409	37
test_06	chuffed	first_fail	indomain_min	7.000	454	261	27
test_07	chuffed	first_fail	indomain_min	7.000	636	494	19

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth
test_08	chuffed	first_fail	indomain_min	37.000	2139	1523	37
test_09	chuffed	first_fail	indomain_min	25.000	1794	1344	25
test_10	chuffed	first_fail	indomain_min	28609.000	736859	595199	50
test_01	chuffed	dom_w_deg	indomain_split	1.000	114	8	53
test_02	chuffed	dom_w_deg	indomain_split	4.000	945	150	117
test_03	chuffed	dom_w_deg	indomain_split	5.000	974	212	121
test_04	chuffed	dom_w_deg	indomain_split	23.000	3269	1065	145
test_05	chuffed	dom_w_deg	indomain_split	188.000	14992	7081	252
test_06	chuffed	dom_w_deg	indomain_split	16.000	2717	910	152
test_07	chuffed	dom_w_deg	indomain_split	7.000	1267	505	100
test_08	chuffed	dom_w_deg	indomain_split	39.000	5237	1663	247
test_09	chuffed	dom_w_deg	indomain_split	48.000	6423	2433	144
test_10	chuffed	dom_w_deg	indomain_split	12236.000	471285	221818	345
test_01	chuffed	input_order	indomain_min	1.000	53	16	14
test_02	chuffed	input_order	indomain_min	6.000	617	272	22
test_03	chuffed	input_order	indomain_min	5.000	442	248	22
test_04	chuffed	input_order	indomain_min	47.000	3435	2735	26
test_05	chuffed	input_order	indomain_min	273.000	10926	9503	38
test_06	chuffed	input_order	indomain_min	25.000	2088	1657	27
test_07	chuffed	input_order	indomain_min	8.000	737	672	20
test_08	chuffed	input_order	indomain_min	28.000	1610	1189	37
test_09	chuffed	input_order	indomain_min	78.000	4925	4365	26
test_10	chuffed	input_order	indomain_min	16350.000	362590	332479	51

Tabla 2: Resultados de pruebas **sin** restricciones redundantes.

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth
test_01	gecode	first_fail	indomain_min	10.203	68	20	13
test_02	gecode	first_fail	indomain_min	499.978	264817	132389	28
test_03	gecode	first_fail	indomain_min	62.441	24282	12104	25
test_04	gecode	first_fail	indomain_min	17341.491	8810676	4405319	39
test_05	gecode	first_fail	indomain_min	59931.770	50739812	25369861	49
test_06	gecode	first_fail	indomain_min	24.156	11998	5983	26
test_07	gecode	first_fail	indomain_min	50.264	23345	11649	23
test_08	gecode	first_fail	indomain_min	47879.649	14960880	7480383	39
test_09	gecode	first_fail	indomain_min	9620.749	6094237	3047083	41
test_10	gecode	first_fail	indomain_min	59934.424	27947665	13973774	51
test_01	gecode	dom_w_deg	indomain_split	0.912	112	33	36
test_02	gecode	dom_w_deg	indomain_split	5.504	1634	774	71
test_03	gecode	dom_w_deg	indomain_split	1.708	334	133	72
test_04	gecode	dom_w_deg	indomain_split	36.409	9649	4793	77
test_05	gecode	dom_w_deg	indomain_split	10.264	1880	838	141
test_06	gecode	dom_w_deg	indomain_split	5.299	1340	593	93
test_07	gecode	dom_w_deg	indomain_split	11.445	3549	1729	75
test_08	gecode	dom_w_deg	indomain_split	13.378	2937	1287	133
test_09	gecode	dom_w_deg	indomain_split	12.210	3567	1688	96
test_10	gecode	dom_w_deg	indomain_split	304.671	42287	20950	182
test_01	gecode	input_order	indomain_min	0.833	200	94	13
test_02	gecode	input_order	indomain_min	10051.899	8250996	4125491	25
test_03	gecode	input_order	indomain_min	276.237	210561	105269	21
test_04	gecode	input_order	indomain_min	58520.001	33415666	16707816	32
test_05	gecode	input_order	indomain_min	59938.719	36319978	18159963	39
test_06	gecode	input_order	indomain_min	897.716	535610	267795	26
test_07	gecode	input_order	indomain_min	170.944	114108	57052	23
test_08	gecode	input_order	indomain_min	59933.128	37142176	18571070	36
test_09	gecode	input_order	indomain_min	10471.427	4784339	2392147	35

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth
test_10	gecode	input_order	indomain_min	59936.365	30298504	15149232	50
test_01	chuffed	first_fail	indomain_min	1.000	68	12	14
test_02	chuffed	first_fail	indomain_min	7.000	537	389	22
test_03	chuffed	first_fail	indomain_min	6.000	697	373	22
test_04	chuffed	first_fail	indomain_min	36.000	2889	2008	26
test_05	chuffed	first_fail	indomain_min	16.000	851	409	38
test_06	chuffed	first_fail	indomain_min	6.000	423	261	27
test_07	chuffed	first_fail	indomain_min	7.000	637	494	20
test_08	chuffed	first_fail	indomain_min	42.000	2069	1523	37
test_09	chuffed	first_fail	indomain_min	25.000	1817	1344	25
test_10	chuffed	first_fail	indomain_min	28238.000	736910	595199	51
test_01	chuffed	dom_w_deg	indomain_split	1.000	117	8	56
test_02	chuffed	dom_w_deg	indomain_split	4.000	949	150	121
test_03	chuffed	dom_w_deg	indomain_split	5.000	974	212	121
test_04	chuffed	dom_w_deg	indomain_split	21.000	3272	1065	148
test_05	chuffed	dom_w_deg	indomain_split	190.000	14992	7081	252
test_06	chuffed	dom_w_deg	indomain_split	15.000	2717	910	152
test_07	chuffed	dom_w_deg	indomain_split	7.000	1270	505	103
test_08	chuffed	dom_w_deg	indomain_split	38.000	5237	1663	247
test_09	chuffed	dom_w_deg	indomain_split	47.000	6423	2433	144
test_10	chuffed	dom_w_deg	indomain_split	12393.000	471285	221818	345
test_01	chuffed	input_order	indomain_min	1.000	53	16	14
test_02	chuffed	input_order	indomain_min	5.000	617	272	22
test_03	chuffed	input_order	indomain_min	5.000	442	248	22
test_04	chuffed	input_order	indomain_min	45.000	3435	2735	26
test_05	chuffed	input_order	indomain_min	252.000	10926	9503	38
test_06	chuffed	input_order	indomain_min	25.000	2088	1657	27
test_07	chuffed	input_order	indomain_min	8.000	737	672	20
test_08	chuffed	input_order	indomain_min	29.000	1610	1189	37
test_09	chuffed	input_order	indomain_min	77.000	4925	4365	26
test_10	chuffed	input_order	indomain_min	16677.000	362590	332479	51

Tabla 3: Resultados de pruebas **con** restricciones redundantes y **con** simetría.

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth
test_01	gecode	first_fail	indomain_min	7.862	68	20	13
test_02	gecode	first_fail	indomain_min	508.427	264886	132435	28
test_01	gecode	dom_w_deg	indomain_split	0.960	113	34	34
test_02	gecode	dom_w_deg	indomain_split	9.232	3150	1549	71
test_01	gecode	input_order	indomain_min	0.821	200	94	13
test_02	gecode	input_order	indomain_min	10240.779	8250996	4125491	25
test_01	chuffed	first_fail	indomain_min	2.000	68	12	14
test_02	chuffed	first_fail	indomain_min	8.000	537	389	22
test_01	chuffed	dom_w_deg	indomain_split	1.000	114	8	53
test_02	chuffed	dom_w_deg	indomain_split	4.000	945	150	117
test_01	chuffed	input_order	indomain_min	1.000	53	16	14
test_02	chuffed	input_order	indomain_min	6.000	617	272	22

Tabla 4: Resultados de pruebas **con** restricciones redundantes y **sin** simetría.

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth
test_01	gecode	first_fail	indomain_min	5.143	171	73	13

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth
test_02	gecode	first_fail	indomain_min	1251.807	641482	320733	28
test_01	gecode	dom_w_deg	indomain_split	0.822	153	54	34
test_02	gecode	dom_w_deg	indomain_split	9.131	3911	1934	65
test_01	gecode	input_order	indomain_min	1.239	360	174	13
test_02	gecode	input_order	indomain_min	17282.710	15001952	7500969	26
test_01	chuffed	first_fail	indomain_min	3.000	76	19	14
test_02	chuffed	first_fail	indomain_min	35.000	3546	2824	22
test_01	chuffed	dom_w_deg	indomain_split	1.000	135	17	53
test_02	chuffed	dom_w_deg	indomain_split	10.000	3069	784	117
test_01	chuffed	input_order	indomain_min	1.000	57	20	14
test_02	chuffed	input_order	indomain_min	16.000	2050	1261	22

Tabla 5: Resultados de pruebas **con** redundancia y **con** simetría (todas las soluciones).

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth	Solutions
test_01	gecode	first_fail	indomain_min	59921.041	3127246	0	25	1563617
test_02	gecode	first_fail	indomain_min	59930.036	2700044	0	32	1350012
test_01	gecode	dom_w_deg	indomain_split	59931.679	3151727	0	45	1575852
test_02	gecode	dom_w_deg	indomain_split	59933.849	2730619	0	82	1365278
test_01	gecode	input_order	indomain_min	59937.490	3173156	0	25	1586572
test_02	gecode	input_order	indomain_min	59937.140	2754654	0	32	1377317
test_01	chuffed	first_fail	indomain_min	59934.000	1517948	1517936	13	1517936
test_02	chuffed	first_fail	indomain_min	59933.000	1294586	1294565	21	1294565
test_01	chuffed	dom_w_deg	indomain_split	59926.000	1905356	1567323	52	1567323
test_02	chuffed	dom_w_deg	indomain_split	59938.000	1547960	1351214	116	1351214
test_01	chuffed	input_order	indomain_min	59934.000	1553352	1553340	13	1553340
test_02	chuffed	input_order	indomain_min	59936.000	1361902	1361881	21	1361881

Tabla 6: Resultados de pruebas **con** redundancia y **sin** simetría (todas las soluciones).

Archivo	Solver	Var heur	Val heur	Time (ms)	Nodes	Failures	Depth	Solutions
test_01	gecode	first_fail	indomain_min	59942.958	3225524	0	25	1612756
test_02	gecode	first_fail	indomain_min	59937.814	2766214	0	32	1383097
test_01	gecode	dom_w_deg	indomain_split	59939.585	3256656	0	45	1628314
test_02	gecode	dom_w_deg	indomain_split	59936.894	2852672	0	76	1426307
test_01	gecode	input_order	indomain_min	59939.783	3251452	0	25	1625720
test_02	gecode	input_order	indomain_min	59933.915	2774174	0	32	1387077
test_01	chuffed	first_fail	indomain_min	59932.000	1566132	1566122	13	1566122
test_02	chuffed	first_fail	indomain_min	59937.000	1332201	1332180	21	1332180
test_01	chuffed	dom_w_deg	indomain_split	59933.000	1931617	1589249	52	1589249
test_02	chuffed	dom_w_deg	indomain_split	59936.000	1592857	1390351	116	1390351
test_01	chuffed	input_order	indomain_min	59936.000	1588141	1588128	13	1588128
test_02	chuffed	input_order	indomain_min	59924.000	1412380	1412359	21	1412359

1.4. Análisis y conclusiones

En el solver `Gecode`, la efectividad de las restricciones redundantes depende fuertemente de la estrategia de búsqueda aplicada. Con la heurística `first_fail` (`ff_min`), su inclusión no garantizó mejoras y en varios

casos fue perjudicial. En ciertas instancias aumentaron el tiempo de ejecución, los nodos y los fallos; es decir, el costo adicional de propagación no siempre se compensó con una poda más efectiva del árbol de búsqueda. En contraste, con la heurística informada `dom_w_deg` (`wdeg_split`) se observaron beneficios claros, en donde en general, los nodos y fallos disminuyeron y los tiempos mejoraron, especialmente en instancias complejas, aunque con excepciones puntuales como `test_10`. Esta estrategia, al ponderar el grado de conflicto de las variables, aprovecha mejor la información adicional de las restricciones redundantes para priorizar ramas más prometedoras. Con la heurística simple `input_order` (`inorder_min`), el impacto fue neutro o negativo: en muchas ejecuciones se alcanzó el límite de tiempo sin mejoras, lo que sugiere que las redundantes añadieron carga computacional sin una poda apreciable.

El solver `Chuffed` mostró un comportamiento distinto y más robusto frente a las restricciones redundantes. Con `ff_min`, estas no afectaron negativamente y en algunas instancias redujeron nodos y fallos, aunque el tiempo fue irregular. Con `wdeg_split`, el efecto fue neutral o levemente positivo, manteniendo o mejorando el rendimiento en las instancias más exigentes gracias a podas tempranas propiciadas por la información redundante. Con `inorder_min`, los resultados fueron mixtos. Algunas instancias mejoraron levemente y otras presentaron un aumento marginal en el tiempo. Esto evidencia que `Chuffed` tolera mejor la sobrecarga de las redundantes que `Gecode`, sufriendo menos degradación en desempeño.

En síntesis, las restricciones redundantes R4 y R5 no son ventajosas de forma global, en donde su impacto depende de la heurística de búsqueda y del solver. Las estrategias informadas, como `wdeg_split`, pueden aprovechar la información extra para mejorar la poda del espacio de búsqueda, mientras que con heurísticas simples la sobrecarga de propagación suele superar los beneficios. Destaca que `Chuffed` logra beneficiarse, o al menos no verse penalizado, por las redundantes en un rango más amplio de configuraciones, especialmente en las de mayor complejidad.

De manera global, la combinación `Chuffed + wdeg_split` resultó la más eficiente. Esta heurística, que prioriza variables con dominios pequeños y alto grado de conflicto, permite una poda temprana del árbol de búsqueda y una reducción sustancial en nodos y fallos. Bajo esta configuración, las restricciones redundantes fueron especialmente útiles, ya que `Chuffed` aprovechó la información adicional sin incurrir en sobrecostos significativos. En contraste, las estrategias más simples (`ff_min` e `inorder_min`) mostraron comportamientos más variables y, en el caso de `Gecode`, incluso contraproducentes, pues el esfuerzo de propagación superó los beneficios en la poda.

Comparando los solvers, `Chuffed` fue más rápido y eficiente que `Gecode` en la mayoría de los escenarios. Esta ventaja fue más evidente con heurísticas menos informadas, donde `Gecode` alcanzó el límite de tiempo con frecuencia. Su arquitectura, basada en *lazy clause generation*, le permite una propagación más inteligente y un mejor aprovechamiento de las redundantes, alcanzando soluciones óptimas con menos esfuerzo. En las instancias más grandes, `Chuffed` resolvió y verificó óptimos con un costo sustancialmente menor.

Respecto a la restricción de simetría (R6), se evaluaron ejecuciones con y sin dicha restricción, manteniendo activas las redundantes, en las instancias `test_01` y `test_02` con trabajos idénticos. En optimización, R6 resultó claramente beneficiosa, especialmente en `test_02`. En `Gecode`, redujo de forma consistente los nodos y fallos, con mejoras notables en tiempo (aproximadamente a la mitad). En `Chuffed`, el efecto fue aún mayor: con `ff_min`, el tiempo bajó de $\sim$35 ms a $\sim$8 ms, y con `wdeg_split` de $\sim$10 ms a $\sim$4 ms, acompañándose de reducciones significativas en nodos y fallos.

Para cuantificar el efecto de R6, se realizaron pruebas en modo `satisfy` para enumerar todas las soluciones, comparando configuraciones con y sin simetría. Aunque la mayoría no completó la enumeración antes del límite de 60 s, en todos los casos el número de soluciones distintas fue ligeramente menor con la simetría activa. La reducción no fue drástica, pero confirma que R6 elimina correctamente soluciones equivalentes por permutación de trabajos idénticos.

En conclusión, la restricción de simetría R6 es una adición útil para instancias con trabajos idénticos, en donde reduce caminos de búsqueda equivalentes, disminuye la redundancia de soluciones y acelera la búsqueda de óptimos. Aunque su impacto en la enumeración completa es moderado, aporta un beneficio sin efectos adversos, por lo que se recomienda incluirla cuando existan trabajos indistinguibles.

2. Jobshop con Tardanza Ponderada

En el *jobshop* con tardanza ponderada, cada trabajo debe ejecutar sus operaciones en una secuencia fija de máquinas, respetando precedencias y evitando solapes (cada máquina procesa a lo sumo una operación a la vez). A cada trabajo i se le asignan un *due date* y un peso w_i; su tardanza se define como $T_i =$

máx$\{0,\ \text{end_job}_i - \text{due_dates}_i\}$. El objetivo es programar los inicios $s_{i,j}$ para minimizar la **suma ponderada de tardanzas** $\sum_i w_i T_i$, priorizando (vía w_i) aquellos trabajos cuya demora resulta más costosa, bajo un horizonte acotado y cumpliendo las restricciones de precedencia y no solape por máquina.

2.1. Modelo

Parámetros

Variables

Restricciones principales

Restricciones redundantes

Restricciones de simetría

2.2. Implementación

Modelo

Restricciones redundantes

Restricciones de simetría

2.3. Pruebas

Tabla 7: Resultados de pruebas **con** restricciones redundantes.

Archivo	Solver	Var heur	Val heur	time	nodes	fail	depth

Tabla 8: Resultados de pruebas **sin** restricciones redundantes.

Archivo	Solver	Var heur	Val heur	time	nodes	fail	depth

2.4. Análisis y conclusiones