5.57 calendar

DESCRIPTION LINKS

Origin [26]

Constraint calendar(INSTANTS, MACHINES)

Type UNAVAILABILITIES : collection(low−int, up−int)

Arguments

$$\begin{array}{ll} \texttt{INSTANTS} & : \texttt{collection}\left(\begin{array}{l}\texttt{machine}-\texttt{dvar},\\ \texttt{virtual}-\texttt{dvar},\\ \texttt{ireal}-\texttt{dvar},\\ \texttt{flagend}-\texttt{int}\end{array}\right)\\ \texttt{MACHINES} & : \texttt{collection}(\texttt{id}-\texttt{int}, \texttt{cal}-\texttt{UNAVAILABILITIES}) \end{array}$$

Restrictions

required(UNAVAILABILITIES, [low, up])
UNAVAILABILITIES.low ≤ UNAVAILABILITIES.up
required(INSTANTS, [machine, virtual, ireal, flagend])
in_attr(INSTANTS, machine, MACHINES, id)
INSTANTS.flagend ≥ 0
INSTANTS.flagend ≤ 1
|MACHINES| > 0
required(MACHINES, [id, cal])
distinct(MACHINES, id)

Purpose Makes the link between an universal calendar and resource dependent calendars. Given a collection of machines MACHINES where each machine is defined by its identifier and its unavailability periods the calendar constraint maps items of real and virtual dates depending on the machine assignment as well as of the fact that we consider start (flagend = 0) or end (flagend = 1) times. Virtual dates on a given machine m do not consider the unavailability periods on m, while real dates consider all time points.

Example

$$\left(\begin{array}{l}
\left\langle\begin{array}{llll}
\texttt{machine}-1 & \texttt{virtual}-2 & \texttt{ireal}-3 & \texttt{flagend}-0,\\
\texttt{machine}-1 & \texttt{virtual}-5 & \texttt{ireal}-6 & \texttt{flagend}-1,\\
\texttt{machine}-2 & \texttt{virtual}-4 & \texttt{ireal}-5 & \texttt{flagend}-0,\\
\texttt{machine}-2 & \texttt{virtual}-6 & \texttt{ireal}-9 & \texttt{flagend}-1,\\
\texttt{machine}-3 & \texttt{virtual}-2 & \texttt{ireal}-2 & \texttt{flagend}-0,\\
\texttt{machine}-3 & \texttt{virtual}-5 & \texttt{ireal}-5 & \texttt{flagend}-1,\\
\texttt{machine}-4 & \texttt{virtual}-2 & \texttt{ireal}-2 & \texttt{flagend}-0,\\
\texttt{machine}-4 & \texttt{virtual}-7 & \texttt{ireal}-9 & \texttt{flagend}-1
\end{array}\right\rangle,\\
\left\langle\begin{array}{l}
\texttt{id}-1 \quad \texttt{cal}-\langle \texttt{low}-2\ \texttt{up}-2, \texttt{low}-6\ \texttt{up}-7\rangle,\\
\texttt{id}-2 \quad \texttt{cal}-\langle \texttt{low}-2\ \texttt{up}-2, \texttt{low}-6\ \texttt{up}-7\rangle,\\
\texttt{id}-3 \quad \texttt{cal}-[],\\
\texttt{id}-4 \quad \texttt{cal}-\langle \texttt{low}-3\ \texttt{up}-4\rangle
\end{array}\right\rangle
\end{array}\right)$$

Figure 5.129 illustrates the example. It present four machines with their respective

unavailability periods (in grey) as well as four tasks (in blue and pink). Each item of the INSTANTS collection corresponds to the start or to the end of one of the previous four tasks. The calendar constraint holds since:

- The real date 3 (INSTANTS[1].ireal = 3) associated with the start (INSTANTS[1].flagend = 0) of task (a) in the universal time corresponds to the virtual date 2 (INSTANTS[1].virtual = 2) on machine 1 (INSTANTS[1].machine = 1).
- The real date 6 (INSTANTS[2].ireal = 6) associated with the end (INSTANTS[2].flagend = 1) of task (a) in the universal time corresponds to the virtual date 5 (INSTANTS[2].virtual = 5) on machine 1 (INSTANTS[2].machine = 1).
- The real date 5 (INSTANTS[3].ireal = 5) associated with the start (INSTANTS[3].flagend = 0) of task (b) in the universal time corresponds to the virtual date 4 (INSTANTS[3].virtual = 4) on machine 2 (INSTANTS[3].machine = 2).
- The real date 9 (INSTANTS[4].ireal = 9) associated with the end (INSTANTS[4].flagend = 1) of task (b) in the universal time corresponds to the virtual date 6 (INSTANTS[4].virtual = 6) on machine 2 (INSTANTS[4].machine = 2).
- The real date 2 (INSTANTS[5].ireal = 2) associated with the start (INSTANTS[5].flagend = 0) of task (c) in the universal time corresponds to the virtual date 2 (INSTANTS[5].virtual = 2) on machine 3 (INSTANTS[5].machine = 3).
- The real date 5 (INSTANTS[6].ireal = 5) associated with the end (INSTANTS[6].flagend = 1) of task (c) in the universal time corresponds to the virtual date 5 (INSTANTS[6].virtual = 5) on machine 3 (INSTANTS[6].machine = 3).
- The real date 2 (INSTANTS[7].ireal = 2) associated with the start (INSTANTS[7].flagend = 0) of task (d) in the universal time corresponds to the virtual date 2 (INSTANTS[7].virtual = 2) on machine 4 (INSTANTS[7].machine = 4).
- The real date 9 (INSTANTS[8].ireal = 9) associated with the end (INSTANTS[8].flagend = 1) of task (d) in the universal time corresponds to the virtual date 7 (INSTANTS[8].virtual = 7) on machine 4 (INSTANTS[8].machine = 4).

Typical

$|\texttt{INSTANTS}| > 1$
$|\texttt{MACHINES}| > 1$

Symmetries

- Items of INSTANTS are permutable.
- Items of MACHINES are permutable.

Arg. properties

Contractible wrt. INSTANTS.

Usage

The calendar constraint is used as a channelling constraint in resource scheduling problems where resources have unavailability periods that can preempt the execution of a task. In this context two time coordinates systems are used:

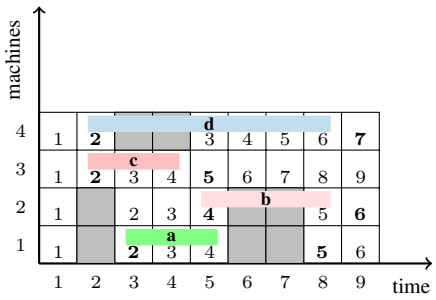


Figure 5.129: Four machines with their unavailability periods as well as four tasks assigned to these machines (virtual dates mentioned in the **Example** slot use a bold font)

- A first coordinate system, so called the *virtual coordinate system*, ignores all unavailability periods on the different resources. All resource constraints are stated within this virtual coordinate system.
- A second coordinate system, so called the *real coordinate system*, corresponds to the real time. All temporal constraints (e.g., precedence constraints) are stated within this real coordinate system.

In this context, each task has a *virtual origin*, a *virtual duration*, a *virtual end*, a *real origin*, a *real duration*, a *real end* and the `calendar` constraint links together the virtual origin and the real origin as well as the virtual end and the real end. The virtual duration (i.e., the real duration plus the sum of the unavailability periods crossed by the task) is linked to the virtual end and the virtual origin through an equality constraint on the difference between the virtual end and the virtual origin. The real duration is linked in a similar way to the real end and the real origin. The keyword scheduling with machine choice, calendars and preemption provides a concrete example of resource scheduling problem using the `calendar` constraint.

Reformulation

The `calendar` constraint can be reformulated into two generalised `case` constraints (i.e., two `case` constraints augmented with linear constraints). Part (A) (respectively Part (B)) of Figure 5.130 provides the directed acyclic graph that allows mapping the virtual start and real start (respectively the virtual end and real end) of a task. This directed acyclic graph can be computed directly from the arguments of the `calendar` constraint:

1. We create an initial root node labelled by m and we partition the set of machines into classes of consecutive machines that all share exactly the same unavailability periods. For each such class we create an arc from the root node to a new node vs labelled by the corresponding interval of consecutive machines identifiers. In Part (A) this corresponds to node m and its three outgoing arcs respectively labelled by intervals $[1, 2]$, $[3, 3]$ and $[4, 4]$.
2. For each class of consecutive machines found previously, we label in increasing order each timepoint that is not part of an unavailability period. We create an arc from the corresponding node vs for each maximum interval of available timepoints to a new node labelled by rs. In Part (A) this translate to:

- For the class corresponding to machines 1 and 2 we create three outgoing arcs labelled by the time intervals $[1, 1]$, $[2, 4]$ and $[5, 6]$.
- For the class corresponding to machine 3 we create the outgoing arc labelled by time interval $[1, 9]$.
- For the class corresponding to machine 4 we create the two outgoing arcs labelled by the time intervals $[1, 2]$ and $[3, 7]$.

3. For each class of consecutive machines and for each maximum interval $[i, j]$ of available timepoints previously computed, we find out the number of unavailable timepoints b_i on the same class of machines that are located before the virtual date i. We create an outgoing arc from the corresponding node rs to a new node labelled by true (there is a single true node for the full directed acyclic graph). This arc is labelled by the interval $[i + b_i, j + b_i]$ and by the linear constraint $r_s = v_s + b_i$. In Part (A) this translate to:
   - For the class corresponding to machines 1 and 2 and for each r_s node associated with the time intervals $[1, 1]$, $[2, 4]$ and $[5, 6]$ we respectively create an outgoing arc labelled by intervals $[1, 1]$, $[3, 5]$ and $[8, 9]$. To each of these arcs we also respectively associate the linear constraints $r_s = v_s + 0$ ($+0$ since on machines 1 and 2 there is no unavailability period before the virtual date 1), $r_s = v_s + 1$ ($+1$ since on machines 1 and 2 there is a single unavailable timepoint before the virtual date 2) and $r_s = v_s + 3$ ($+3$ since on machines 1 and 2 there is three unavailable timepoints before the virtual date 5).
   - For the class corresponding to machine 3 and for the r_s node associated with the time interval $[1, 9]$ we create the outgoing arc labelled by time interval $[1, 9]$ and by $r_s = v_s + 0$ (i.e., since their is no unavailability period at all on machine 3).
   - For the class corresponding to machine 4 and for each r_s node associated with the time intervals $[1, 2]$ and $[3, 7]$ we respectively create an outgoing arc labelled by $[1, 2]$ and $[5, 9]$. To each of these arcs we also respectively associate the linear constraints $r_s = v_s + 0$ ($+0$ since on machine 4 there is no unavailability period before the virtual date 1) and $r_s = v_s + 2$ ($+2$ since on machine 4 there is two unavailable timepoints before the virtual date 3).

The `calendar` constraint can also be reformulated into a conjunction of reified constraints. This is done by generating, for each pair of items $(\mathcal{I}, \mathcal{M})$ of the `INSTANTS` and `MACHINES` collections, a set of reified constraints expressing:

- The link between the real and the virtual dates under the hypothesis that the `machine` attribute of item $\mathcal{I}$ is assigned to the value of the `id` attribute of item $\mathcal{M}$. More precisely, we generate one reified constraint for each available time interval on machine `id`.
- The fact that a real date should not be located within an unavailability period of its corresponding machine.

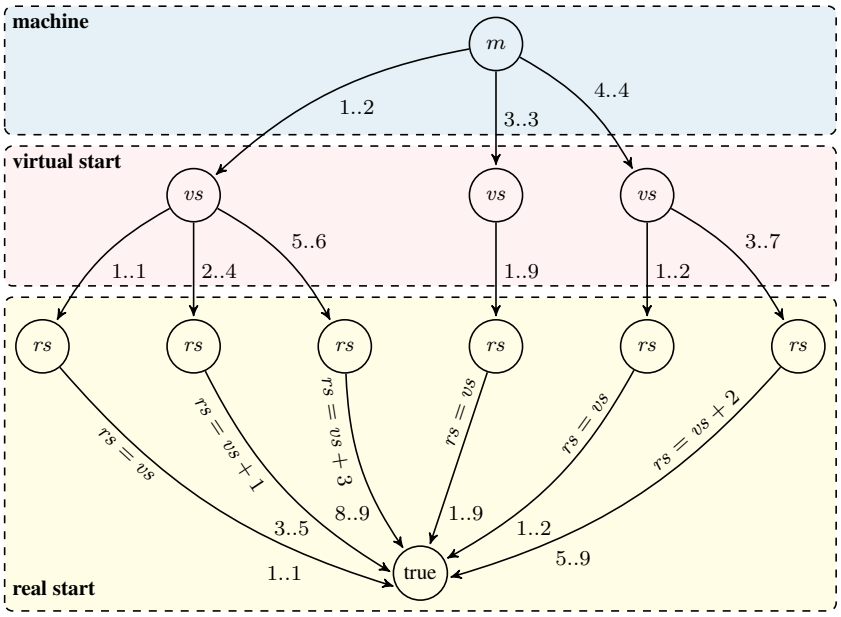
Operationally, this leads to the following cases:

1. When machine `id` has *no unavailability at all* we state an equality constraint between the real and virtual dates.
2. When the real date is located *before the first unavailability period* we also state an equality constraint between the real and virtual dates.

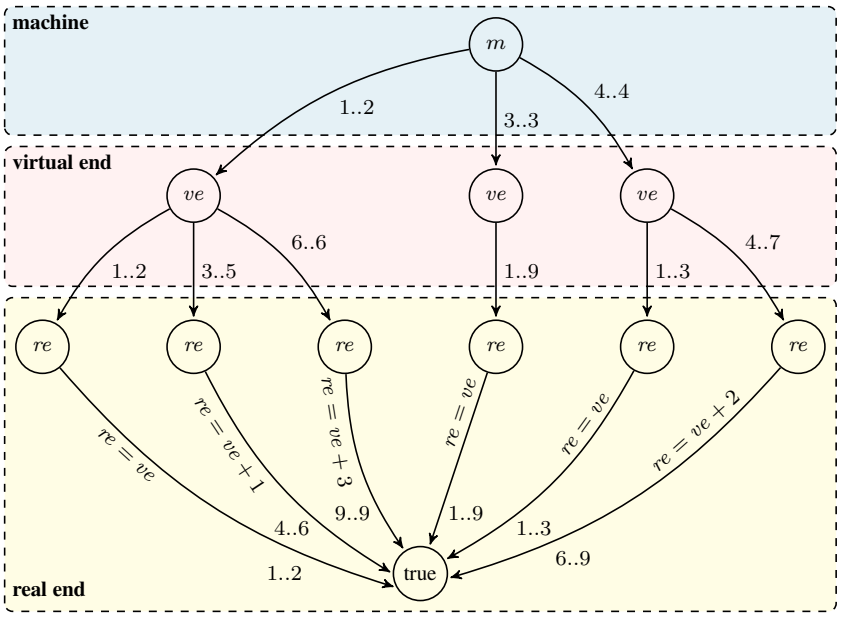
3. When the real date is located *between two consecutive unavailability periods* we state:
   - An equality constraint between the real date and the virtual date plus the sum of all unavailabilities located before the real date.
   - An implication between the fact that the real date belongs to the first unavailability period (among the two consecutive unavailability periods) and the fact that the real date is not assigned to the machine that contains the unavailability period.
4. When the real date is located *after the last unavailability period* we state:
   - An equality constraint between the real date and the virtual date plus the sum of all unavailabilities.
   - An implication between the fact that the real date belongs to the last unavailability period and the fact that the real date is not assigned to the machine that contains the unavailability period.

As an example consider again consider the instance given in the **Example** slot. For the *start of task* a (i.e., the first item $\langle \texttt{machine} - 1\ \texttt{virtual} - 2\ \texttt{ireal} - 2\ \texttt{flagend} - 0 \rangle$ of collection `INSTANTS`), we generate the following reified constraints, where equivalences of the form $\mathrm{true} \Leftrightarrow \mathrm{true}$ are shown in bold:

- (*if task* a *is assigned on machine 1*)
  - ⋆ before $[2, 2]$: $1 = 1 \wedge 3 < 2 \Leftrightarrow 1 = 1 \wedge 3 = 2$
  - ⋆ between $[2, 2]$ and $[6, 7]$: $\mathbf{1 = 1 \wedge 3 > 2 \wedge 3 < 6 \Leftrightarrow 1 = 1 \wedge 3 = 2 + 1}$
  - ⋆ after $[6, 7]$: $1 = 1 \wedge 3 > 7 \Leftrightarrow 1 = 1 \wedge 3 = 2 + 3$
  - ⋆ do not cross $[2, 2]$, $[6, 7]$: $3 \in [2, 2] \Rightarrow 1 \neq 1, 3 \in [6, 7] \Leftrightarrow 1 \neq 1$
- (*if task* a *is assigned on machine 2*)
  - ⋆ before $[2, 2]$: $1 = 2 \wedge 3 < 2 \Leftrightarrow 1 = 2 \wedge 3 = 2$
  - ⋆ between $[2, 2]$ and $[6, 7]$: $1 = 2 \wedge 3 > 2 \wedge 3 < 6 \Leftrightarrow 1 = 2 \wedge 3 = 2 + 1$
  - ⋆ after $[6, 7]$: $1 = 2 \wedge 3 > 7 \Leftrightarrow 1 = 2 \wedge 3 = 2 + 3$
  - ⋆ do not cross $[2, 2]$, $[6, 7]$: $3 \in [2, 2] \Rightarrow 1 \neq 2, 3 \in [6, 7] \Leftrightarrow 1 \neq 2$
- (*if task* a *is assigned on machine 3*)
  - ⋆ no unavailability: $1 = 3 \Leftrightarrow 1 = 3 \wedge 3 = 2$
- (*if task* a *is assigned on machine 4*)
  - ⋆ before $[3, 4]$: $1 = 4 \wedge 3 < 3 \Leftrightarrow 1 = 4 \wedge 3 = 2$
  - ⋆ after $[3, 4]$: $1 = 4 \wedge 3 > 4 \Leftrightarrow 1 = 4 \wedge 3 = 2 + 2$
  - ⋆ do not cross $[3, 4]$: $3 \in [3, 4] \Rightarrow 1 \neq 4$



(A) directed acyclic graph linking the machine attribute m, the virtual start vs and the real start rs of a task



(B) directed acyclic graph linking the machine attribute m, the virtual end ve and the real end re of a task

Figure 5.130: The two generalised `case` constraints for respectively mapping (A) the virtual start and real start of a task corresponding to the **Example** slot as well as (B) the virtual end and real end; the directed acyclic graphs were generated under the hypothesis that the virtual and real dates are located in $[1, 9]$.

For the *end of task* a (i.e., the second item $\langle$`machine`-1 `virtual`-5 `ireal`-6 `flagend`$-1\rangle$ of collection `INSTANTS`), we generate the following reified constraints:

- (*if task* a *is assigned on machine 1*)
  - ⋆ before $[2,2]$: $1 = 1 \wedge 6 < 3 \Leftrightarrow 1 = 1 \wedge 6 = 5$
  - ⋆ between $[2,2]$ and $[6,7]$: $\mathbf{1 = 1 \wedge 6 > 3 \wedge 6 < 7 \Leftrightarrow 1 = 1 \wedge 6 = 5 + 1}$
  - ⋆ after $[6,7]$: $1 = 1 \wedge 6 > 8 \Leftrightarrow 1 = 1 \wedge 6 = 5 + 3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $6 \in [3,3] \Rightarrow 1 \neq 1, 6 \in [7,8] \Rightarrow 1 \neq 1$
- (*if task* a *is assigned on machine 2*)
  - ⋆ before $[2,2]$: $1 = 2 \wedge 6 < 3 \Leftrightarrow 1 = 2 \wedge 6 = 5$
  - ⋆ between $[2,2]$ and $[6,7]$: $1 = 2 \wedge 6 > 3 \wedge 6 < 7 \Leftrightarrow 1 = 2 \wedge 6 = 5 + 1$
  - ⋆ after $[6,7]$: $1 = 2 \wedge 6 > 8 \Leftrightarrow 1 = 2 \wedge 6 = 5 + 3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $6 \in [3,3] \Rightarrow 1 \neq 2, 6 \in [7,8] \Rightarrow 1 \neq 2$
- (*if task* a *is assigned on machine 3*)
  - ⋆ no unavailability: $1 = 3 \Leftrightarrow 1 = 3 \wedge 6 = 5$
- (*if task* a *is assigned on machine 4*)
  - ⋆ before $[3,4]$: $1 = 4 \wedge 6 < 4 \Leftrightarrow 1 = 4 \wedge 6 = 5$
  - ⋆ after $[3,4]$: $1 = 4 \wedge 6 > 5 \Leftrightarrow 1 = 4 \wedge 6 = 5 + 2$
  - ⋆ do not cross $[3,4]$: $6 \in [4,5] \Rightarrow 1 \neq 4$

For the *start of task* b (i.e., the third item $\langle$`machine`-2 `virtual`-4 `ireal`-5 `flagend`$-0\rangle$ of collection `INSTANTS`), we generate the following reified constraints:

- (*if task* b *is assigned on machine 1*)
  - ⋆ before $[2,2]$: $2 = 1 \wedge 5 < 2 \Leftrightarrow 2 = 1 \wedge 5 = 4$
  - ⋆ between $[2,2]$ and $[6,7]$: $2 = 1 \wedge 5 > 2 \wedge 5 < 6 \Leftrightarrow 2 = 1 \wedge 5 = 4 + 1$
  - ⋆ after $[6,7]$: $2 = 1 \wedge 5 > 7 \Leftrightarrow 2 = 1 \wedge 5 = 4 + 3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $5 \in [2,2] \Rightarrow 2 \neq 1, 5 \in [6,7] \Rightarrow 2 \neq 1$
- (*if task* b *is assigned on machine 2*)
  - ⋆ before $[2,2]$: $2 = 2 \wedge 5 < 2 \Leftrightarrow 2 = 2 \wedge 5 = 4$
  - ⋆ between $[2,2]$ and $[6,7]$: $\mathbf{2 = 2 \wedge 5 > 2 \wedge 5 < 6 \Leftrightarrow 2 = 2 \wedge 5 = 4 + 1}$
  - ⋆ after $[6,7]$: $2 = 2 \wedge 5 > 7 \Leftrightarrow 2 = 2 \wedge 5 = 4 + 3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $5 \in [2,2] \Rightarrow 2 \neq 2, 5 \in [6,7] \Rightarrow 2 \neq 2$
- (*if task* b *is assigned on machine 3*)
  - ⋆ no unavailability: $2 = 3 \Leftrightarrow 2 = 3 \wedge 5 = 4$
- (*if task* b *is assigned on machine 4*)
  - ⋆ before $[3,4]$: $2 = 4 \wedge 5 < 3 \Leftrightarrow 2 = 4 \wedge 5 = 4$
  - ⋆ after $[3,4]$: $2 = 4 \wedge 5 > 4 \Leftrightarrow 2 = 4 \wedge 5 = 4 + 2$
  - ⋆ do not cross $[3,4]$: $5 \in [3,4] \Rightarrow 2 \neq 4$

For the *end of task* b (i.e., the fourth item $\langle$`machine`-2 `virtual`-6 `ireal`-9 `flagend`$-1\rangle$ of collection `INSTANTS`), we generate the following reified constraints:

- (*if task* b *is assigned on machine 1*)
  - ⋆ before $[2,2]$: $2 = 1 \wedge 9 < 3 \Leftrightarrow 2 = 1 \wedge 9 = 6$
  - ⋆ between $[2,2]$ and $[6,7]$: $2 = 1 \wedge 9 > 3 \wedge 9 < 7 \Leftrightarrow 2 = 1 \wedge 9 = 6 + 1$
  - ⋆ after $[6,7]$: $2 = 1 \wedge 9 > 8 \Leftrightarrow 2 = 1 \wedge 9 = 6 + 3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $9 \in [3,3] \Rightarrow 2 \neq 1, 9 \in [7,8] \Rightarrow 2 \neq 1$
- (*if task* b *is assigned on machine 2*)
  - ⋆ before $[2,2]$: $2 = 2 \wedge 9 < 3 \Leftrightarrow 2 = 2 \wedge 9 = 6$
  - ⋆ between $[2,2]$ and $[6,7]$: $2 = 2 \wedge 9 > 3 \wedge 9 < 7 \Leftrightarrow 2 = 2 \wedge 9 = 6 + 1$
  - ⋆ after $[6,7]$: $\mathbf{2 = 2 \wedge 9 > 8 \Leftrightarrow 2 = 2 \wedge 9 = 6 + 3}$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $9 \in [3,3] \Rightarrow 2 \neq 2, 9 \in [7,8] \Rightarrow 2 \neq 2$
- (*if task* b *is assigned on machine 3*)
  - ⋆ no unavailability: $2 = 3 \Leftrightarrow 2 = 3 \wedge 9 = 6$
- (*if task* b *is assigned on machine 4*)
  - ⋆ before $[3,4]$: $2 = 4 \wedge 9 < 4 \Leftrightarrow 2 = 4 \wedge 9 = 6$
  - ⋆ after $[3,4]$: $2 = 4 \wedge 9 > 5 \Leftrightarrow 2 = 4 \wedge 9 = 6 + 2$
  - ⋆ do not cross $[3,4]$: $9 \in [4,5] \Rightarrow 2 \neq 4$

For the *start of task* c (i.e., the fifth item $\langle$`machine`-3 `virtual`-2 `ireal`-2 `flagend`$-0\rangle$ of collection `INSTANTS`), we generate the following reified constraints:

- (*if task* c *is assigned on machine 1*)
  - ⋆ before $[2,2]$: $3 = 1 \wedge 2 < 2 \Leftrightarrow 3 = 1 \wedge 2 = 2$
  - ⋆ between $[2,2]$ and $[6,7]$: $3 = 1 \wedge 2 > 2 \wedge 2 < 6 \Leftrightarrow 3 = 1 \wedge 2 = 2 + 1$
  - ⋆ after $[6,7]$: $3 = 1 \wedge 2 > 7 \Leftrightarrow 3 = 1 \wedge 2 = 2 + 3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $2 \in [2,2] \Rightarrow 3 \neq 1, 2 \in [6,7] \Rightarrow 3 \neq 1$
- (*if task* c *is assigned on machine 2*)
  - ⋆ before $[2,2]$: $3 = 2 \wedge 2 < 2 \Leftrightarrow 3 = 2 \wedge 2 = 2$
  - ⋆ between $[2,2]$ and $[6,7]$: $3 = 2 \wedge 2 > 2 \wedge 2 < 6 \Leftrightarrow 3 = 2 \wedge 2 = 2 + 1$
  - ⋆ after $[6,7]$: $3 = 2 \wedge 2 > 7 \Leftrightarrow 3 = 2 \wedge 2 = 2 + 3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $2 \in [2,2] \Rightarrow 3 \neq 2, 2 \in [6,7] \Rightarrow 3 \neq 2$
- (*if task* c *is assigned on machine 3*)
  - ⋆ no unavailability: $\mathbf{3 = 3 \Leftrightarrow 3 = 3 \wedge 2 = 2}$
- (*if task* c *is assigned on machine 4*)
  - ⋆ before $[3,4]$: $3 = 4 \wedge 2 < 3 \Leftrightarrow 3 = 4 \wedge 2 = 2$
  - ⋆ after $[3,4]$: $3 = 4 \wedge 2 > 4 \Leftrightarrow 3 = 4 \wedge 2 = 2 + 2$
  - ⋆ do not cross $[3,4]$: $2 \in [3,4] \Rightarrow 3 \neq 4$

For the *end of task* c (i.e., the sixth item $\langle$`machine`-3 `virtual`-5 `ireal`-5 `flagend`$-1\rangle$ of collection `INSTANTS`), we generate the following reified constraints:

- (*if task* c *is assigned on machine 1*)
  - ⋆ before $[2, 2]$: $3 = 1 \wedge 5 < 3 \Leftrightarrow 3 = 1 \wedge 5 = 5$
  - ⋆ between $[2, 2]$ and $[6, 7]$: $3 = 1 \wedge 5 > 3 \wedge 5 < 7 \Leftrightarrow 3 = 1 \wedge 5 = 5 + 1$
  - ⋆ after $[6, 7]$: $3 = 1 \wedge 5 > 8 \Leftrightarrow 3 = 1 \wedge 5 = 5 + 3$
  - ⋆ do not cross $[2, 2]$, $[6, 7]$: $5 \in [3..3] \Rightarrow 3 \neq 1, 5 \in [7..8] \Rightarrow 3 \neq 1$
- (*if task* c *is assigned on machine 2*)
  - ⋆ before $[2, 2]$: $3 = 2 \wedge 5 < 3 \Leftrightarrow 3 = 2 \wedge 5 = 5$
  - ⋆ between $[2, 2]$ and $[6, 7]$: $3 = 2 \wedge 5 > 3 \wedge 5 < 7 \Leftrightarrow 3 = 2 \wedge 5 = 5 + 1$
  - ⋆ after $[6, 7]$: $3 = 2 \wedge 5 > 8 \Leftrightarrow 3 = 2 \wedge 5 = 5 + 3$
  - ⋆ do not cross $[2, 2]$, $[6, 7]$: $5 \in [3..3] \Rightarrow 3 \neq 2, 5 \in [7..8] \Rightarrow 3 \neq 2$
- (*if task* c *is assigned on machine 3*)
  - ⋆ no unavailability: $\mathbf{3 = 3 \Leftrightarrow 3 = 3 \wedge 5 = 5}$
- (*if task* c *is assigned on machine 4*)
  - ⋆ before $[3, 4]$: $3 = 4 \wedge 5 < 4 \Leftrightarrow 3 = 4 \wedge 5 = 5$
  - ⋆ after $[3, 4]$: $3 = 4 \wedge 5 > 5 \Leftrightarrow 3 = 4 \wedge 5 = 5 + 2$
  - ⋆ do not cross $[3, 4]$: $5 \in [4..5] \Rightarrow 3 \neq 4$

For the *start of task* d (i.e., the seventh item $\langle$`machine` $-$ 4 `virtual` $-$ 2 `ireal` $-$ 2 `flagend` $-$ $0\rangle$ of collection `INSTANTS`), we generate the following reified constraints:

- (*if task* d *is assigned on machine 1*)
  - ⋆ before $[2, 2]$: $4 = 1 \wedge 2 < 2 \Leftrightarrow 4 = 1 \wedge 2 = 2$
  - ⋆ between $[2, 2]$ and $[6, 7]$: $4 = 1 \wedge 2 > 2 \wedge 2 < 6 \Leftrightarrow 4 = 1 \wedge 2 = 2 + 1$
  - ⋆ after $[6, 7]$: $4 = 1 \wedge 2 > 7 \Leftrightarrow 4 = 1 \wedge 2 = 2 + 3$
  - ⋆ do not cross $[2, 2]$, $[6, 7]$: $2 \in [2, 2] \Rightarrow 4 \neq 1, 2 \in [6, 7] \Rightarrow 4 \neq 1$
- (*if task* d *is assigned on machine 2*)
  - ⋆ before $[2, 2]$: $4 = 2 \wedge 2 < 2 \Leftrightarrow 4 = 2 \wedge 2 = 2$
  - ⋆ between $[2, 2]$ and $[6, 7]$: $4 = 2 \wedge 2 > 2 \wedge 2 < 6 \Leftrightarrow 4 = 2 \wedge 2 = 2 + 1$
  - ⋆ after $[6, 7]$: $4 = 2 \wedge 2 > 7 \Leftrightarrow 4 = 2 \wedge 2 = 2 + 3$
  - ⋆ do not cross $[2, 2]$, $[6, 7]$: $2 \in [2, 2] \Rightarrow 4 \neq 2, 2 \in [6, 7] \Rightarrow 4 \neq 2$
- (*if task* d *is assigned on machine 3*)
  - ⋆ no unavailability: $4 = 3 \Leftrightarrow 4 = 3 \wedge 2 = 2$
- (*if task* d *assigned on machine 4*)
  - ⋆ before $[3, 4]$: $\mathbf{4 = 4 \wedge 2 < 3 \Leftrightarrow 4 = 4 \wedge 2 = 2}$
  - ⋆ after $[3, 4]$: $4 = 4 \wedge 2 > 4 \Leftrightarrow 4 = 4 \wedge 2 = 2 + 2$
  - ⋆ do not cross $[3, 4]$: $2 \in [3, 4] \Rightarrow 4 \neq 4$

For the *end of task* d (i.e., the eighth item $\langle$`machine`-4 `virtual`-7 `ireal`-9 `flagend`$-1\rangle$ of collection `INSTANTS`), we generate the following reified constraints:

- (*if task* d *is assigned on machine 1*)
  - ⋆ before $[2,2]$: $4=1 \wedge 9<3 \Leftrightarrow 4=1 \wedge 9=7$
  - ⋆ between $[2,2]$ and $[6,7]$: $4=1 \wedge 9>3 \wedge 9<7 \Leftrightarrow 4=1 \wedge 9=7+1$
  - ⋆ after $[6,7]$: $4=1 \wedge 9>8 \Leftrightarrow 4=1 \wedge 9=7+3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $9 \in [3,3] \Rightarrow 4 \neq 1, 9 \in [7,8] \Rightarrow 4 \neq 1$
- (*if task* d *is assigned on machine 2*)
  - ⋆ before $[2,2]$: $4=2 \wedge 9<3 \Leftrightarrow 4=2 \wedge 9=7$
  - ⋆ between $[2,2]$ and $[6,7]$: $4=2 \wedge 9>3 \wedge 9<7 \Leftrightarrow 4=2 \wedge 9=7+1$
  - ⋆ after $[6,7]$: $4=2 \wedge 9>8 \Leftrightarrow 4=2 \wedge 9=7+3$
  - ⋆ do not cross $[2,2]$, $[6,7]$: $9 \in [3,3] \Rightarrow 4 \neq 2, 9 \in [7,8] \Rightarrow 4 \neq 2$
- (*if task* d *is assigned on machine 3*)
  - ⋆ no unavailability: $4=3 \Leftrightarrow 4=3 \wedge 9=7$
- (*if task* d *is assigned on machine 4*)
  - ⋆ before $[3,4]$: $4=4 \wedge 9<4 \Leftrightarrow 4=4 \wedge 9=7$
  - ⋆ after $[3,4]$: $\mathbf{4=4 \wedge 9>5 \Leftrightarrow 4=4 \wedge 9=7+2}$
  - ⋆ do not cross $[3,4]$: $9 \in [4,5] \Rightarrow 4 \neq 4$

See also **common keyword:** `cumulative` *(scheduling constraint)*, `cumulatives` *(scheduling with machine choice, calendars and preemption)*, `diffn` *(multi-site employee scheduling with calendar constraints, scheduling with machine choice, calendars and preemption)*, `disjunctive` *(scheduling constraint)*, `geost` *(multi-site employee scheduling with calendar constraints, scheduling with machine choice, calendars and preemption)*.

Keywords **constraint type:** predefined constraint, temporal constraint, scheduling constraint.

modelling: channelling constraint, multi-site employee scheduling with calendar constraints, scheduling with machine choice, calendars and preemption, assignment dimension.

modelling exercises: multi-site employee scheduling with calendar constraints, scheduling with machine choice, calendars and preemption.