

1 Machine Scheduling

- Machine scheduling with unary resource
- Scheduling with cumulative resource
- Job-shop problem

2 Timetabling

3 Employees Scheduling

4 Exercises

- Operator Scheduling
- Meeting Scheduling
- Scheduling of Computational Jobs
- Room Assignment

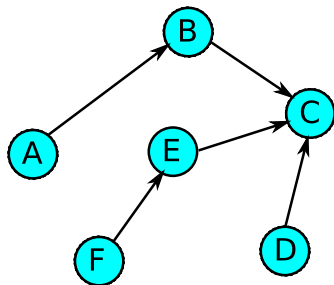
Machine scheduling with unary resource: problem & example

Problem: Create a schedule for several tasks with

- earliest (est) and latest completion time (lct)
- processing time (P)
- precedence constraints given by the graph

on machine of unit capacity such that the makespan is minimized

Task T	est(T)	lct(T)	PT
A	0	10	2
B	0	15	3
C	5	25	4
D	0	20	1
E	10	25	5
F	0	5	3



Machine scheduling with unary resource: variables

- Start time variables $StartT$ for each task T
- $lct(T) = lct(T) - PT$
 $StartT$ in $est(T)..lct(T)$
- Example:

Task T	$est(T)$	$lct(T)$	PT
A	0	10	2
B	0	15	3
C	5	25	4
D	0	20	1
E	10	25	5
F	0	5	3

$StartA$ in $0..8$, $StartB$ in $0..12$, $StartC$ in $5..21$,
 $StartD$ in $0..19$, $StartE$ in $10..20$, $StartF$ in $0..2$

Machine scheduling with unary resource: constraints

Precedence constraints for each tasks $T1 \ll T2$

- $StartT1 + PT1 \#=< StartT2$
- $StartA+2 \#=< StartB$, $StartB+3 \#=< StartC$,
 $StartF+3 \#=< StartE$, $StartE+5 \#=< StartC$, $StartD+1 \#=< StartC$

Unary resource for all tasks T given by

- start time variables $StartT$
- end time variables $EndT \# = StartT + PT$
 $cumulative([task(StartA,2,EndA,1,A), \dots, task(StartF,3,EndF,1,F)],Options)$

Machine scheduling with unary resource: constraints

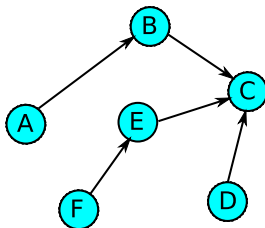
Precedence constraints for each tasks $T1 \ll T2$

- $StartT1 + PT1 \# = < StartT2$
- $StartA + 2 \# = < StartB$, $StartB + 3 \# = < StartC$,
 $StartF + 3 \# = < StartE$, $StartE + 5 \# = < StartC$, $StartD + 1 \# = < StartC$

Unary resource for all tasks T given by

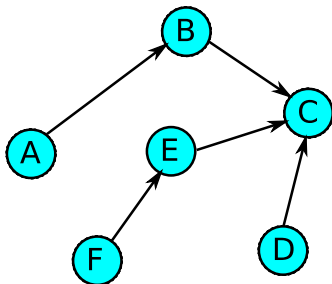
- start time variables $StartT$
 - end time variables $EndT \# = StartT + PT$
- `cumulative([task(StartA,2,EndA,1,A), .., task(StartF,3,EndF,1,F)],Options)`

Task T	est(T)	lct(T)	PT
A	0	10	2
B	0	15	3
C	5	25	4
D	0	20	1
E	10	25	5
F	0	5	3



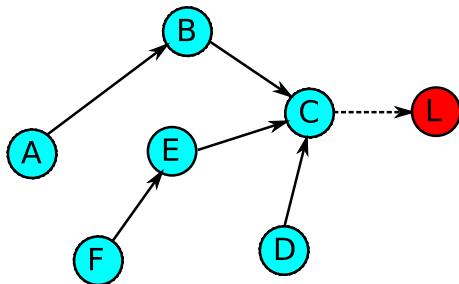
Scheduling with unary resource: optimization

- New task L with $PL=0$ added
- Precedence constraints
 between L and tasks with no successor added
- Example: $StartC+4 \neq \leq StartL$



Scheduling with unary resource: optimization

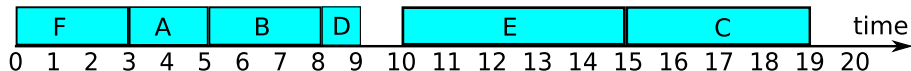
- New task L with $PL=0$ added
- Precedence constraints
 between L and tasks with no successor added
- Example: $StartC+4 \#=< StartL$



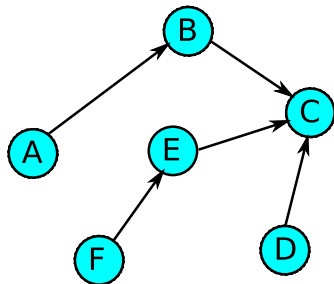
- $minimize(\text{Makespan}) = minimize(\text{StartL})$

Scheduling with unary resource: solution

Solution:



Task T	est(T)	lct(T)	PT
A	0	10	2
B	0	15	3
C	5	25	4
D	0	20	1
E	10	25	5
F	0	5	3



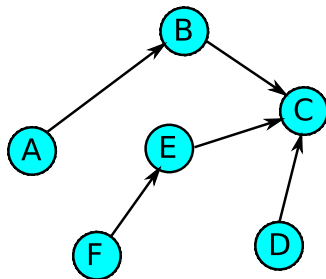
Scheduling with cumulative resource: problem & example

Problem: Create a schedule for several tasks with

- earliest (est) and latest completion time (lct)
- processing time (p)
- requested capacity of the resource (cap)
- precedence constraints given by the graph

on machine of capacity 3 such that the makespan is minimized

Task T	est(T)	lct(T)	PT	CapT
A	0	10	2	1
B	0	15	3	2
C	5	25	4	2
D	0	20	1	3
E	10	25	5	2
F	0	5	3	2



Scheduling with cumulative resource: modeling

Same model as for scheduling with unary resource with

- unary resource replaced by cumulative resource

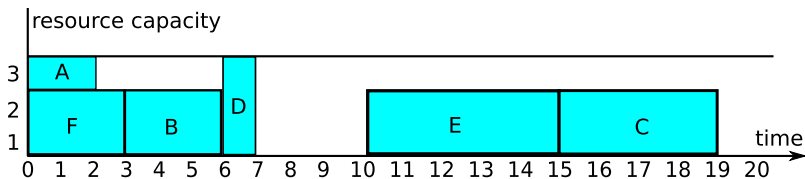
Cumulative resource for all tasks T given by

- start time variables $Start_T$
- duration PT
- requested capacity of the resource
- example: `cumulative([task(StartA,2,EndA,1,A), ..., task(StartF,3,EndF,2,F)], [limit(3)|Options])`

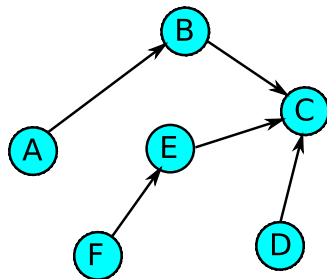
Task T	est(T)	lct(T)	PT	CapT
A	0	10	2	1
B	0	15	3	2
C	5	25	4	2
D	0	20	1	3
E	10	25	5	2
F	0	5	3	2

Scheduling with cumulative resource: solution

Solution:



Task T	est(T)	lct(T)	PT	CapT
A	0	10	2	1
B	0	15	3	2
C	5	25	4	2
D	0	20	1	3
E	10	25	5	2
F	0	5	3	2



Job-shop problem: problem

Create a schedule for several tasks such that

- each task consists of several jobs
- ordering of jobs for each task is fixed
- jobs of each tasks are processed on different dedicated machine
- machines have unit capacity
- makespan is minimized

Job-shop problem: example

- Machines: M1, M2, M3
- Tasks T1, T2 and T3 with jobs noted by (machine,task)

T1: (3,1)«(2,1)«(1,1)

T2: (1,2)«(3,2)

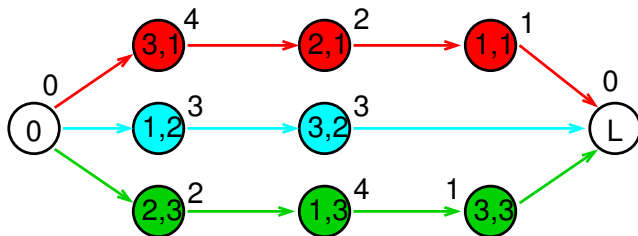
T3: (2,3)«(1,3)«(3,3)

- Processing times:

$P_{31}=4, P_{21}=2, P_{11}=1$

$P_{12}=3, P_{32}=3$

$P_{23}=2, P_{13}=4, P_{33}=1$



Additional first and last activities O and L with $P_0 = P_L = 0$

Job-shop problem: modeling

Variables and domains

- Start_{IJ} start time variables for J task running on machine I

Constraints

- ordering of jobs modeled through precedence constraints
Start₃₁+P₃₁ #=< Start₂₁, Start₂₁+P₂₁ #=< Start₁₁, ...
- unary resource constraint for each machine I
with jobs (I,J) for all tasks J
1st machine: cumulative([task(Start₁₁,1,End₁₁,1,11),
task(Start₁₂,3,End₁₂,1,12), task((Start₁₃,4,End₁₃,1,13))],Options),
2nd machine: cumulative(...), 3d machine: cumulative(...)

Job-shop problem: modeling

Variables and domains

- Start_{IJ} start time variables for J task running on machine I

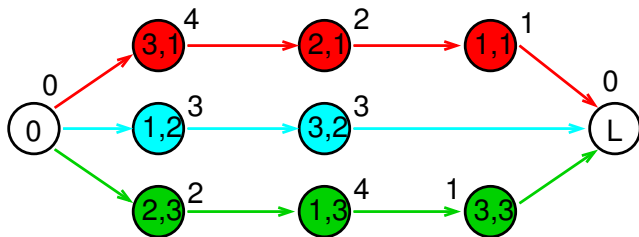
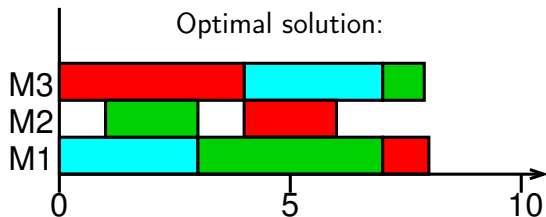
Constraints

- ordering of jobs modeled through precedence constraints
Start₃₁+P₃₁ #=< Start₂₁, Start₂₁+P₂₁ #=< Start₁₁, ...
- unary resource constraint for each machine I
with jobs (I,J) for all tasks J
1st machine: cumulative([task(Start₁₁,1,End₁₁,1,11),
task(Start₁₂,3,End₁₂,1,12), task((Start₁₃,4,End₁₃,1,13))],Options),
2nd machine: cumulative(...), 3d machine: cumulative(...)

Optimization

- precedence constraints: Start₁₁+1 #< Start_L, Start₃₂+3 #< Start_L,
Start₃₃+1 #< Start_L
- minimize(Makespan) = minimize(Start_L)

Job-shop problem: solution



Class Timetabling: problem

Create schedule for N periods for classes with

- given duration
- given teacher
- given number of students
- prohibited time periods

Several classrooms (M) with specified number of seats are given.

There are sets of classes creating a curriculum

- no overlaps within curricula allowed

Class Timetabling: variables and domains

- Class represents activity with given duration
- Start time variables for each class **StartA**
 - StartA in $0..(N-1)$ (N number of periods)
 - for each prohibited time period i of the class A:
StartA \neq ProhibitedAi

Class Timetabling: variables and domains

- Class represents activity with given duration
- Start time variables for each class **StartA**
 - StartA in $0..(N-1)$ (N number of periods)
 - for each prohibited time period i of the class A:
StartA \neq ProhibitedAi
- Classroom represents resource
- Classrooms are ordered by the number of seats
 - smallest classroom = 0
 - largest classroom = $M-1$ (M number of rooms)
- Classroom variable for each class **ResourceA**
 - ResourceA in $K..(M-1)$ such that K is the smallest classroom where the class fits by the number of students
 - example
 - 4 classrooms with sizes 20, 20, 40, 80 corresponding to 0,1,2,3
 - class A wants a room of the size 20: ResourceA in 0..3
 - class B wants a room of the size 40: ResourceB in 2..3
 - class C wants a room of the size 80: ResourceC = 3

Class Timetabling: resource constraints

Teacher represents a unary resource

- classes of one teacher cannot overlap
- all classes of each teacher are constrained by unary resource constraint
- classes are represented with their **StartA** and **PA** variables
- $\text{EndA} \neq \text{StartA} + \text{PA}$
- for each teacher I and all his classes I_1, \dots :
cumulative([task(Start I_1 , PI I_1 , End I_1 , 1, I_1), ...], Options)

Class Timetabling: resource constraints

Teacher represents a unary resource

- classes of one teacher cannot overlap
- all classes of each teacher are constrained by unary resource constraint
- classes are represented with their **StartA and PA** variables
- $EndA \neq StartA + PA$
- for each teacher I and all his classes I_1, \dots :
cumulative([task(Start I_1 , PI $_1$, End I_1 , 1, I_1), ...], Options)

Curriculum represents a unary resource

- classes of one curriculum cannot overlap
- classes of one curriculum define one unary resource constraint
- classes are represented with their **StartA and PA** variables
- $EndA \neq StartA + PA$
- for each curriculum J and all its classes J_1, \dots :
cumulative([task(Start J_1 , PI $_1$, End J_1 , 1, J_1), ...], Options)

Class Timetabling: time and classrooms

Constraint:

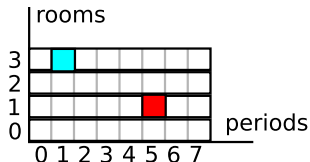
- Each time at most one course must be taught at any classroom

All classrooms together represents one unary resource

- all classes request this resource
- each class is encoded by activity with the starting time

$\text{StartResourceA} \neq \text{StartA} + \text{ResourceA} * N$

and duration PA



Class Timetabling: time and classrooms

Constraint:

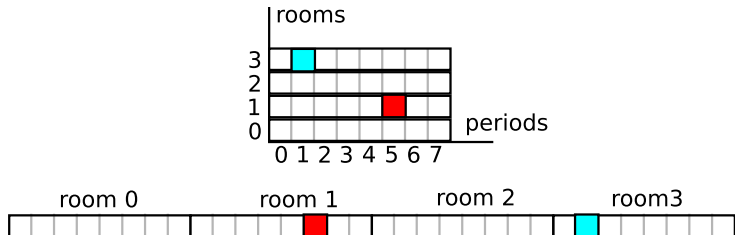
- Each time at most one course must be taught at any classroom

All classrooms together represents one unary resource

- all classes request this resource
- each class is encoded by activity with the starting time

$$\text{StartResourceA} \neq \text{StartA} + \text{ResourceA} * N$$

and duration PA



Class Timetabling: time and classrooms

Constraint:

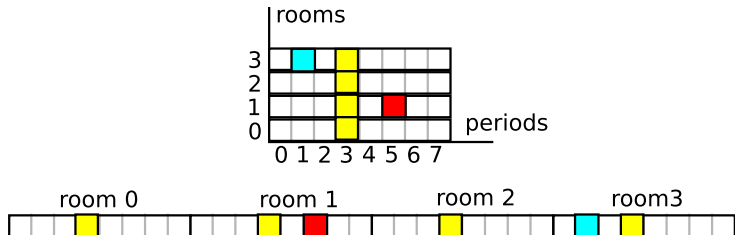
- Each time at most one course must be taught at any classroom

All classrooms together represents one unary resource

- all classes request this resource
- each class is encoded by activity with the starting time

$$\text{StartResourceA} \neq \text{StartA} + \text{ResourceA} * N$$

and duration PA



For all classes A_1, \dots :

$\text{cumulative}([\text{task}(\text{StartResourceA}_1, \text{PA}_1, (\text{StartResourceA}_1 + \text{PA}_1), 1, A_1), \dots], \text{Options})$

Create a one week schedule for employees working on shifts with

- several shift types
- minimal and maximal number of employees per shift
- minimal and maximal number of shift types per employee
- minimal and maximal number of working shifts per employee
- cost for each shift type to be paid to employee working on it
- minimal cost

Employees scheduling: example

- Employees Peter, Paul, Mary, Jane, Keith, Alex, Anne
- One week schedule, i.e. 7 days
- Shift types: M morning, A afternoon, N night
- Each shift – M:3 employees, A:2-3 employees, N:1-2 employees
- Each employee – M: 2-3 shifts, A:1-3 shifts, N: 1-2 shifts
- Working shifts: 4-6
- Cost – $\text{CostM}=10$, $\text{CostA}=11$, $\text{CostN}=13$

Schedule	Mo	Tue	Wed	Thu	...
Peter	M	M	N	-	
Paul	A	A	-	M	
Mary	N	-	M	A	
...					

Matrix of variables corresponding to shifts of employees:

- PeterMo, PeterTue, . . . , PeterSun,
PaulMo, PaulTue, . . . , PaulSun,
MaryMo, MaryTue, . . . , MarySun,
. . .

Domains:

- new shift type F (free) added to record free time shifts
- M, A, N, F corresponds to 1,2,3,4
- domain of variables corresponds to 1..4

Global cardinality constraint (revision)

Global constraint `global_cardinality(List, KeyCounts)`

- List: list of domain variables
- KeyCounts: list of Key-Count tuples with
 - Key: unique integer in the list of keys
 - Count: domain variable (or natural number)
- Each Key is contained in List with cardinality Count

Minimal and maximal number constraints I.

Schedule	Mo	Tue	Wed	Thu	...
Peter	M	M	N	F	...
Paul	A	A	F	M	
Mary	N	F	M	A	
...	...				

Minimal and maximal number of employees per shift

- new domain variables representing these numbers
M1 in $\text{MinM1}..\text{MaxM1}$, A1 in $\text{MinA1}..\text{MaxA1}$, N1 in $\text{MinN1}..\text{MaxN1}$
- global cardinality constraint for each day
`global_cardinality([PeterMo,PaulMo,MaryMo,...], [1-M1,2-A1,3-N1])`

Minimal and maximal number constraints I.

Schedule	Mo	Tue	Wed	Thu	...
Peter	M	M	N	F	...
Paul	A	A	F	M	
Mary	N	F	M	A	
...	...				

Minimal and maximal number of employees per shift

- new domain variables representing these numbers
M1 in $\text{MinM1}..\text{MaxM1}$, A1 in $\text{MinA1}..\text{MaxM1}$, N1 in $\text{MinN1}..\text{MaxN2}$
- global cardinality constraint for each day
`global_cardinality([PeterMo,PaulMo,MaryMo,...], [1-M1,2-A1,3-N1])`

Minimal and maximal number of shift types per employee

- new domain variables representing these numbers
M2 in $\text{MinM2}..\text{MaxM2}$, A2 in $\text{MinA2}..\text{MaxM2}$, N2 in $\text{MinN2}..\text{MaxN2}$
- global cardinality constraint for each employee
`global_cardinality([PeterMo,PeterTue...,PeterSun], [1-M2,2-A2,3-N2])`

Minimal and maximal number constraints II.

Minimal (MinW) and maximal number (MaxW)
of working shifts per employee

- $\text{MinF} = \text{Days} - \text{MaxW}$... minimal number of free time shifts
- $\text{MaxF} = \text{Days} - \text{MinW}$... maximal number of free time shifts
- example
 - $\text{Days}=7, \text{MaxW}=6 \Rightarrow \text{MinF}=1$
 - $\text{Days}=7, \text{MinW}=4 \Rightarrow \text{MaxF}=3$
- new domain variable F in $\text{MinF}.. \text{MaxF}$
- `global_cardinality([PeterMo,PeterTue,...,PeterSun], [4-F])`

Minimal and maximal number constraints II.

Minimal (MinW) and maximal number (MaxW)
of working shifts per employee

- $\text{MinF} = \text{Days} - \text{MaxW}$... minimal number of free time shifts
- $\text{MaxF} = \text{Days} - \text{MinW}$... maximal number of free time shifts
- example
 - $\text{Days}=7, \text{MaxW}=6 \Rightarrow \text{MinF}=1$
 - $\text{Days}=7, \text{MinW}=4 \Rightarrow \text{MaxF}=3$
- new domain variable F in $\text{MinF}.. \text{MaxF}$
- `global_cardinality([PeterMo,PeterTue,...,PeterSun], [4-F])`
- better to add to "Minimal and maximal number of shift types per employee" global cardinality constraint
`global_cardinality([PeterMo,PeterTue,...,PeterSun], [1-M2,2-A2,3-N2,4-F])`

Cost minimization

Schedule	Mo	Tue	Wed	Thu	...
Peter	M	M	N	F	...
Paul	A	A	F	M	
Mary	N	F	M	A	
...	...				

- $M1$, $A1$, $N1$ represent number of particular shifts on Monday
- All $M1$ variables for particular days can be summarized into M and same for $A1$ and A , $N1$ and N
 $\text{sum}([M1\text{Mon}, M1\text{Tue}, \dots, M1\text{Sun}], \# = , \text{Cost}M), \dots$
- Total schedule cost corresponds to
 $\text{Cost} \# = M * \text{Cost}M + A * \text{Cost}A + N * \text{Cost}N$

Cost minimization

Schedule	Mo	Tue	Wed	Thu	...
Peter	M	M	N	F	...
Paul	A	A	F	M	
Mary	N	F	M	A	
...	...				

- $M1$, $A1$, $N1$ represent number of particular shifts on Monday
- All $M1$ variables for particular days can be summarized into M and same for $A1$ and A , $N1$ and N
 $\text{sum}([M1\text{Mon}, M1\text{Tue}, \dots, M1\text{Sun}], \# = , \text{CostM}), \dots$
- Total schedule cost corresponds to
 $\text{Cost} \# = M * \text{CostM} + A * \text{CostA} + N * \text{CostN}$

Alternatively

- $M2$, $A2$, $N2$ represent number of particular shifts for Peter
- All $M2$ variables for all employees can be summarized into M
... and same for $A2$ and A , $N2$ and N

Define **domain variables with their domains** and write a **set of constraints** together with possible **optimization criteria** to describe the model of particular constraint satisfaction problems in the following exercises.

There are N tasks and M operators. Tasks should be scheduled within 4 weeks, each week starts on Monday and ends on Friday. Each task I has specified constant number of days D_I to be processed. Find starting day of all tasks with given prerequisites:

- 1 Each day can be processed Limit tasks at most.
- 2 Each operator J has a list of tasks J_1, \dots, J_{K_J} to work on. In addition, each operator can process only one task at any time.
- 3 Each task I has specified week W_I to be completed at the latest.
- 4 Last task N must be processed after completion of three first tasks 1,2,3.

Operator Scheduling: Solution

Variables for starting times:

- $\text{domain}([T_1, \dots, T_N], 1, 20)$

Constraints:

- 1 $\text{cumulative}([\text{task}(T_1, D_1, (T_1 + D_1), 1, 1), \dots, \text{task}(T_N, D_N, (T_N + D_N), 1, N)],$
 $[\text{limit}(\text{Limit}) | \text{Options}])$
- 2 M constraints:
 $\text{cumulative}([\text{task}(T_{J1}, D_{J1}, (T_{J1} + D_{J1}), 1, 1), \dots,$
 $\text{task}(T_{KJ}, D_{KJ}, (T_{KJ} + D_{KJ}), 1, KJ)], \text{Options})$
- 3 N constraints: $T_1 + D_1 \# = < W_1 * 5$
- 4 $T_1 + D_1 \# = < T_N, T_2 + D_2 \# = < T_N, T_3 + D_3 \# = < T_N$

Find time and room for N meetings and K persons. Meetings will run from 8:00 to 17:00 and each meeting J has specified its constant duration DJ .

- 1 Meetings will take place in M rooms. One meeting can place at each time and room at most.
- 2 Each person I will attend 4 meeting $I1, I2, I3, I4$.
- 3 Meeting A must take before meeting B.
- 4 Meeting C must take after meeting D.
- 5 There are no meetings during lunch time from 12:00 to 13:00.

Meeting Scheduling: Solution

Variables for starting time and room:

- $\text{domain}([T_1, \dots, T_N], 0, 8)$
0 corresponds to 8:00, 8 corresponds to 16:00
- $\text{domain}([R_1, \dots, R_N], 0, M-1)$

Constraints:

- 1 variable for each meeting l : $\text{TimeRoom}l \# = Rl * 9 + Tl$
 $\text{cumulative}([\text{task}(\text{TimeRoom}1, D1, \text{TimeRoom}1 + D1, 1, 1), \dots,$
 $\text{task}(\text{TimeRoom}N, DN, \text{TimeRoom}N + DN, 1, N)], \text{Options})$
- 2 K constraints:
 $\text{cumulative}([\text{task}(Tl1, Dl1, Tl1 + Dl1, 1, l1), \dots,$
 $\text{task}(Tl4, Dl4, Tl4 + Dl4, 1, l4)], \text{Options})$
- 3 $TA + DA \# = < TB$
- 4 $TD + DD \# = < TC$
- 5 N constraints: $Tl \# \setminus = 4$ (4 corresponds to 12:00)

Scheduling of Computational Jobs

There are N jobs to be processed on M processor cluster. Find starting time of all jobs given duration of the job I equal to D_I and required number of processors by the job R_I .

- 1 Each processor can process one job at any time.
- 2 Each job J requires memory E_J specified in GB. Any time the total allocated memory must be smaller or equal to 128GB available on the cluster.
- 3 Some jobs needs output of other jobs, so that they needs to be processed after completion of given jobs. Specifically, job U uses output of jobs V and job V uses output of W . In addition, job R uses output of S and T .

Also completion time of all jobs must be minimized.

Scheduling of Computational Jobs: Solution

Variables for starting time:

- $\text{sum}([D1, \dots, DN], \# = \text{Sum})$, $\text{domain}([T1, \dots, TN], 0, (\text{Sum}-1))$
starting time of all jobs will be certainly smaller than sum of their durations

Constraints:

- 1 $\text{cumulative}([\text{task}(T1, D1, (T1+D1), R1, 1), \dots, \text{task}(TN, DN, (TN+DN), RN, N)], [\text{limit}(M)|\text{Options}])$
- 2 $\text{cumulative}([\text{task}(T1, D1, (T1+D1), E1, 1), \dots, \text{task}(TN, DN, (TN+DN), EN, N)], [\text{limit}(128)|\text{Options}])$
- 3 $TU \# \geq TV + DV$, $TV \# \geq TW + DW$
 $TR \# \geq TS + DS$, $TR \# \geq TT + DT$

Optimization:

- N constraints: $\text{End} \# \geq T1 + D1$
- $\text{minimize}(\text{End})$

Room Assignment

Find rooms for 6 courses A,B,C,D,E,F taking from 9 am to 4 am. Each course consists of several lectures and each lecture takes one hour. Each lecture has specified its time (starting time of all lectures for each course is given in the table). All lectures of one course must be at the same room. There are three rooms available and one lecture can be at any room at most.

Course/Time period	9 am	10 am	11 am	12 am	1 pm	2 pm	3 pm
A		+	+	+			
B		+	+				
C	+	+	+				
D				+	+	+	
E				+	+		
F					+	+	+

Room Assignment: Solution

Domain variables:

- $\text{domain}([\text{RoomA}, \text{RoomB}, \text{RoomC}, \text{RoomD}, \text{RoomE}, \text{RoomF}], 1, 3)$

Constraints:

- 10 am: $\text{all_different}([\text{RoomA}, \text{RoomB}, \text{RoomC}])$
- 11 am: $\text{all_different}([\text{RoomA}, \text{RoomB}, \text{RoomC}])$
(not necessary, same as for 10 am)
- 12 am: $\text{all_different}([\text{RoomA}, \text{RoomD}, \text{RoomE}])$
- 1 pm: $\text{all_different}([\text{RoomD}, \text{RoomE}, \text{RoomF}])$
- 2 pm: $\text{RoomD} \neq \text{RoomF}$ (not necessary, included in for 1 pm)