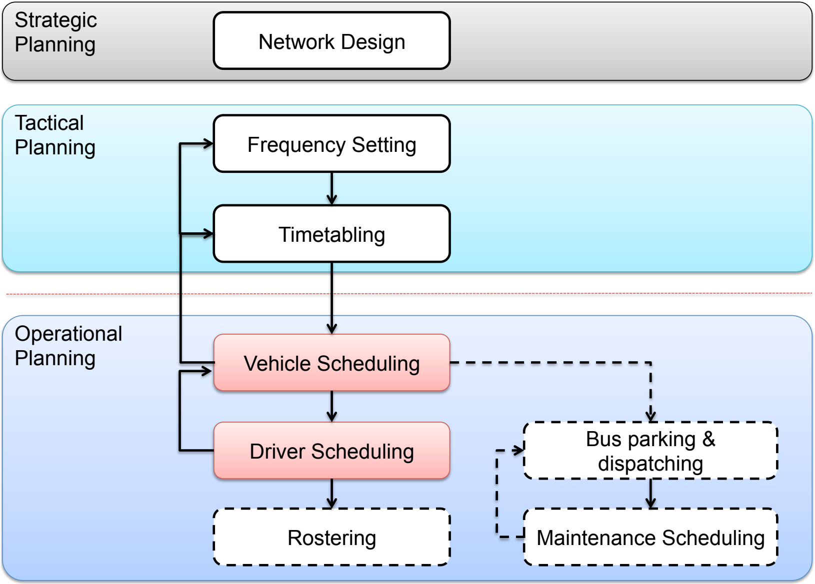
Crew Scheduling: Models and Algorithms

Stefano Gualandi
Università di Pavia, Dipartimento di Matematica

email: stefano.gualandi@unipv.it
twitter: @famo2spaghi
blog: http://stegua.github.com

Overview of Planning Activities (Desaulniers&Hickman2007)



Crew Scheduling

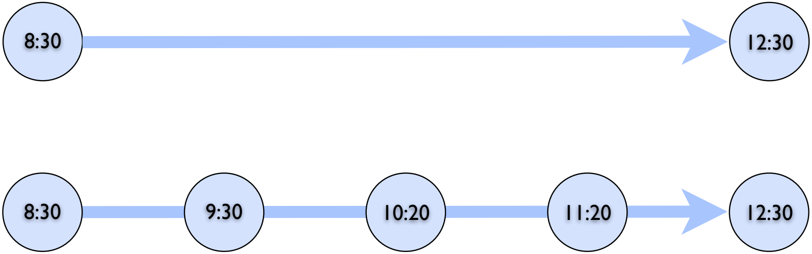
Definition (Relief times)

Each **vehicle duty** (herein called **block**) has a set of **relief times** where a driver substitution may occur.

Crew Scheduling

Definition (Relief times)

Each **vehicle duty** (herein called **block**) has a set of **relief times** where a driver substitution may occur.



Crew Scheduling

Definition (Piece of Work (PoW))

A piece of work p is a continuous driving period from $s(p)$ to $e(p)$.
A piece of work is feasible for a block k if both $s(p)$ and $e(p)$ are relief times of k.

Example: Given

- a block that starts at 8:30 and ends 12:30
- relief times at $\{8{:}30, 9{:}30, 10{:}20, 11{:}20, 12{:}30\}$
- constraint: a PoW lasts at least 01:00 and at most 02:00

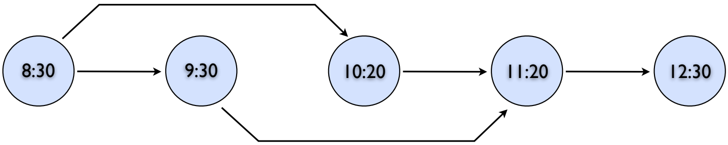
Crew Scheduling

Definition (Piece of Work (PoW))

A piece of work p is a continuous driving period from $s(p)$ to $e(p)$. A piece of work is feasible for a block k if both $s(p)$ and $e(p)$ are relief times of k.

Example: Given

- a block that starts at 8:30 and ends 12:30
- relief times at {8:30,9:30,10:20,11:20,12:30}
- constraint: a PoW lasts at least 01:00 and at most 02:00



- (each of these arcs is a valid piece of work)

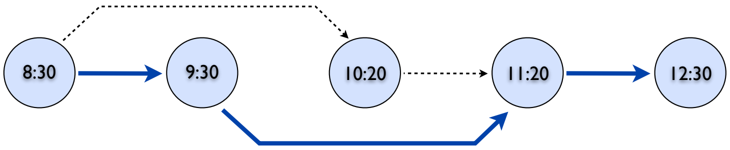
Crew Scheduling

Definition (Piece of Work (PoW))

A piece of work p is a continuous driving period from $s(p)$ to $e(p)$. A piece of work is feasible for a block k if both $s(p)$ and $e(p)$ are relief times of k.

Example: Given

- a block that starts at 8:30 and ends 12:30
- relief times at $\{8{:}30, 9{:}30, 10{:}20, 11{:}20, 12{:}30\}$
- constraint: a PoW lasts at least 01:00 and at most 02:00



- (each of these arcs is a valid piece of work)

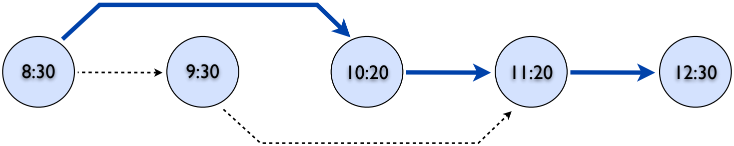
Crew Scheduling

Definition (Piece of Work (PoW))

A piece of work p is a continuous driving period from $s(p)$ to $e(p)$. A piece of work is feasible for a block k if both $s(p)$ and $e(p)$ are relief times of k.

Example: Given

- a block that starts at 8:30 and ends 12:30
- relief times at {8:30,9:30,10:20,11:20,12:30}
- constraint: a PoW lasts at least 01:00 and at most 02:00



- (each of these arcs is a valid piece of work)

Crew Scheduling

Definition (Crew duty)

A crew duty consists of a set of pairs (p, k) where p is a piece of work associated to block k.

Definition (Crew Scheduling)

Given a Vehicle Schedule (i.e. a collection of vehicle duties), the **Crew Scheduling** problem consists of finding a set of crew duties to be assigned to drivers in order to guarantee the daily service.

Crew Scheduling: Urban and Regional

Crew Scheduling

- $\{1, \ldots, r\}$ vehicle duties (blocks) indexed by k
- $T_k = \{t_1^k, \ldots, t_{u_k}^k\}$ is the set of relief times for block k
- t_1^k and $t_{u_k}^k$ are the starting and ending time of the block k
- P_k set of pieces of work feasible for block k
- $\mathcal{D} = \{d_1, \ldots, d_{|\mathcal{D}|}\}$ set of all feasible crew duties

Partition of blocks into pieces of work

For each block, we define the network $G_k = (N_k, A_k)$ where

- $N_k = T_k$ one node for each relief time
- $A_k = \{(s(p), e(p)) \mid p \in P_k\}$ an arc for each piece of work

The problem of finding a partition of a block into pieces of work is:

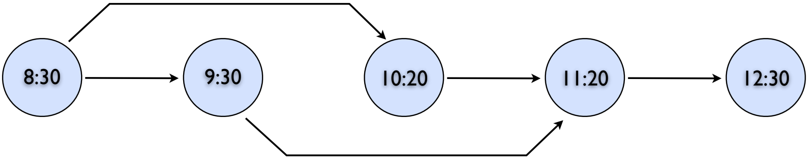
$$\sum_{p \in P_K | s(p)=i} y_p^k - \sum_{p \in P_k | e(p)=i} y_p^k = \begin{cases} 1 & \text{if } i = t_1^k \\ 0 & \text{if } i = t_j^k, j = 2, \ldots, u_k - 1 \\ -1 & \text{if } i = t_{u_k}^k \end{cases}$$

$$y_p^k \in \{0, 1\} \qquad \forall p \in P_k$$

We can write in compact form:

$$E^k y^k = b^k, \qquad y^k \in \{0, 1\}$$

Partition of blocks into pieces of work



The problem of finding a partition of a block into pieces of work is:

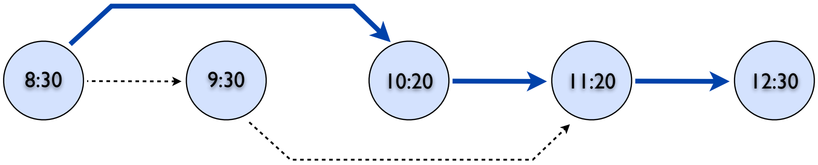
$$\sum_{p \in P_K | s(p)=i} y_p^k - \sum_{p \in P_k | e(p)=i} y_p^k = \begin{cases} 1 & \text{if } i = t_1^k \\ 0 & \text{if } i = t_j^k, j = 2, \dots, u_k - 1 \\ -1 & \text{if } i = t_{u_k}^k \end{cases}$$

$$y_p^k \in \{0, 1\} \qquad \forall p \in P_k$$

We can write in compact form:

$$E^k y^k = b^k, \qquad y^k \in \{0, 1\}$$

Partition of blocks into pieces of work



The problem of finding a partition of a block into pieces of work is:

$$\sum_{p \in P_K | s(p)=i} y_p^k - \sum_{p \in P_k | e(p)=i} y_p^k = \begin{cases} 1 & \text{if } i = t_1^k \\ 0 & \text{if } i = t_j^k, j = 2, \dots, u_k - 1 \\ -1 & \text{if } i = t_{u_k}^k \end{cases}$$

$$y_p^k \in \{0,1\} \qquad \forall p \in P_k$$

We can write in compact form:

$$E^k y^k = b^k, \qquad y^k \in \{0,1\}$$

Crew Scheduling: Basic Model

- Let λ be a $|\mathcal{D}|$-vector of binary variables corresponding to the set of all feasible crew duties
- Let I_{pk} be the subset of all the crew duty indices corresponding in G to arcs incident to (p, k)

$$
\begin{aligned}
\min \quad & \sum_{d \in \mathcal{D}} c_d \lambda_d && && (1)\\
\text{s.t.} \quad & E^k y^k = b^k && \forall k \in 1, \ldots, r && (2)\\
& \sum_{d \in I_{pk}} \lambda_d = y_p^k && \forall p \in P_k, k = 1, \ldots, r && (3)\\
& y^k \in \{0, 1\}^{m_k} && \forall k = 1, \ldots, r && (4)\\
& \lambda \in \{0, 1\}^{|\mathcal{D}|} && && (5)\\
& \lambda \in \mathcal{D}. && && (6)
\end{aligned}
$$

Crew Scheduling and Regional Transit

In Regional Transit, Crew Scheduling is performed before of Vehicle Scheduling, and in practice the set of pieces of work is given (there are very few relief times).

- Let P be the set of piece of work
- Let $\mathcal{D}$ be the set of every possible crew duty
- The cost of a duty j is denoted by c_j
- $b_{ij} = \begin{cases} 1 & \text{if the piece of work } i \text{ appears in duty } j \\ 0 & \text{otherwise} \end{cases}$

Crew Scheduling and Regional Transit

$$\min \sum_{j \in \mathcal{D}} c_j \lambda_j \tag{7}$$

$$\text{s.t.} \sum_{j \in \mathcal{D}} b_{ij} \lambda_j = 1 \qquad \forall i \in P \quad \rightarrow \quad \text{partition of PoW} \tag{8}$$

$$\lambda_j \in \{0, 1\} \qquad \forall j \in \mathcal{D} \quad \rightarrow \quad \text{every possible duty} \tag{9}$$

A set partitioning problem

Crew Scheduling: Set Partitioning Formulation

$$\min \quad \sum_{j \in \mathcal{D}} c_j \lambda_j \tag{10}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{D}} b_{ij} \lambda_j = 1 \qquad \forall i \in P \quad \rightarrow \quad \text{partition of PoW} \tag{11}$$

$$\lambda_j \geq 0 \qquad \forall j \in \mathcal{D} \quad \rightarrow \quad \text{every possible duty} \tag{12}$$

First step: to solve the continuous relaxation

QUESTION: Is it easy to solve the LP?

ISSUE: the size of $\mathcal{D}$ is exponential in $|P|$!

Column Generation

$$(LP) \quad \min \{cx \mid Ax \geq b, x \in \mathbb{R}^n\}$$

- Column Generation is efficient for solving **very large linear programs as (LP-MP)**
- Since most of the variables will be **non-basic** and assume a value of zero in the optimal solution, only a subset of variables need to be considered
- Column generation leverages this idea to generate only the variables which have the potential to improve the objective function, that is, to find variables with negative reduced cost

Dealing with Finitely Many Columns

The main idea is to start with a subset of columns $\bar{\mathcal{D}} \subset \mathcal{D}$ such that a feasible solution to the following problem exists:

$$z_{RMP} = \min \sum_{j \in \bar{\mathcal{D}}} c_j \lambda_j \tag{13}$$

$$\text{s.t.} \sum_{j \in \bar{\mathcal{D}}} b_{ij} \lambda_j \geq 1 \qquad \forall i \in P \tag{14}$$

$$\lambda_j \geq 0 \qquad \forall j \in \bar{\mathcal{D}} \tag{15}$$

Using the Duality Theory of Linear Programming we can generate as set of improving columns. . .

Column Generation: A Dual Persepective

Consider the LP relaxation of the "master" problem and its dual:

$$(P)\ \min \sum_{j\in\bar{\mathcal{D}}} c_j\lambda_j$$
$$\text{s.t.} \sum_{j\in\bar{\mathcal{D}}} b_{ij}\lambda_j \geq 1, \quad \forall i \in P,$$
$$\lambda_j \geq 0, \qquad \forall j \in \bar{\mathcal{D}}.$$

$$(D)\ \max \sum_{i\in P} \pi_i$$
$$\text{s.t.} \sum_{i\in P} b_{ij}\pi_i \leq c_j, \quad \forall j \in \bar{\mathcal{D}},$$
$$\pi_i \geq 0, \qquad \forall i \in P.$$

Using the Duality Theory of Linear Programming we can generate a set of improving columns. . . by separating inequalities on the dual of the master problem!

Pricing Subproblem (Separation on the Master Dual)

The question is:

Does a column (duty) in $\mathcal{D} \setminus \bar{\mathcal{D}}$ that could improve the current optimal solution of the linear relaxation exist?

Does a column (row of the dual) exist such that ...?

$$\exists j \in \mathcal{D} \setminus \bar{\mathcal{D}} : \quad \sum_{i \in P} b_{ij} \pi_i > c_j$$

Pricing Subproblem (Separation on the Master Dual)

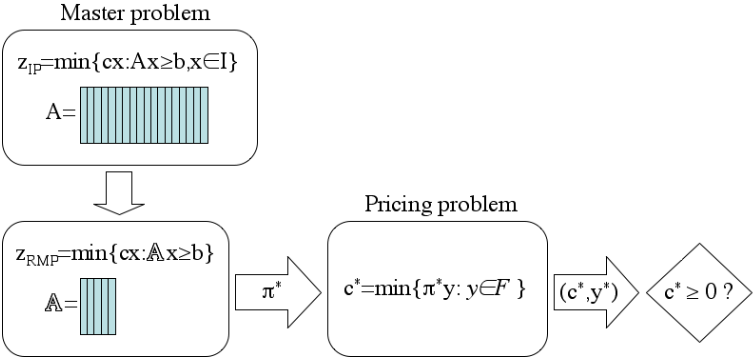
Given the vector of optimal dual multipliers $\bar{\pi}$ for (RMP), we look for a column (duty) such that:

$$\begin{aligned} c^* = \min \quad & c_j - \sum_{i \in P} \bar{\pi}_i y_i \\ \text{s.t.} \quad & y \in F \\ & y_i \in \{0, 1\}. \end{aligned}$$

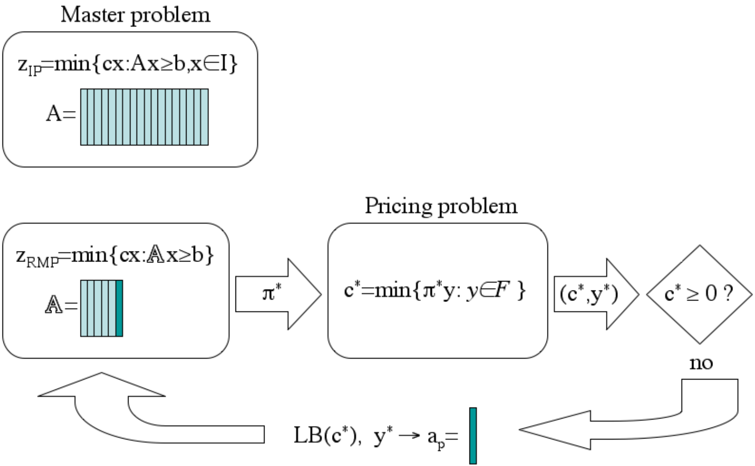
If $c^* < 0$, the vector of variables y is the incidence vector of an *"improving"* column. It corresponds to a variable with **negative reduced cost** in the (restricted) master problem.

What is F in Crew Scheduling problems?

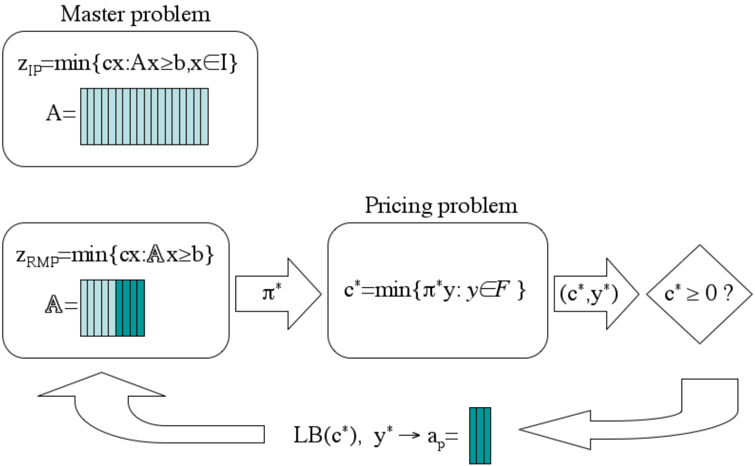
Column Generation: Algorithmic Persepective



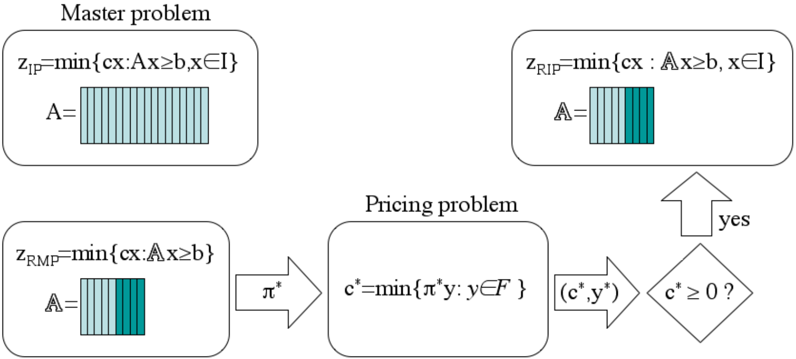
Column Generation: Algorithmic Persepective



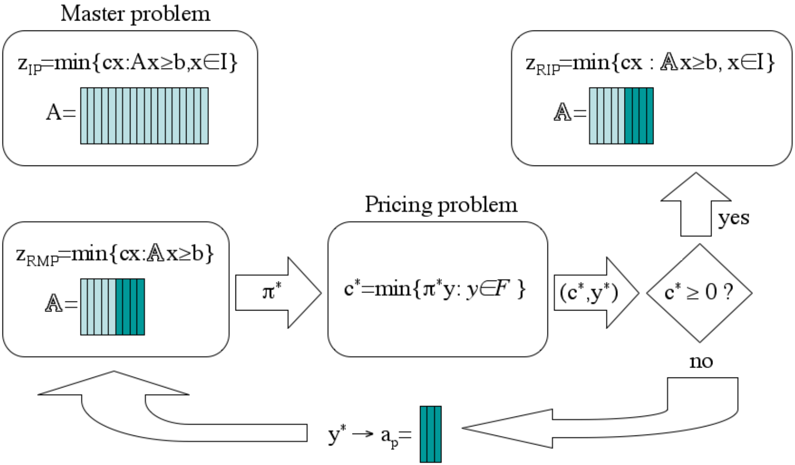
Column Generation: Algorithmic Persepective



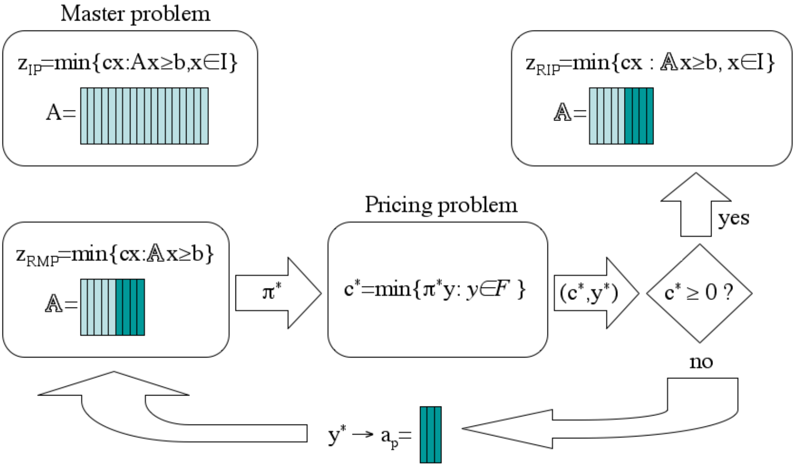
Column Generation: Algorithmic Persepective



Column Generation: Algorithmic Persepective



Column Generation: Algorithmic Persepective



What is F in Crew Scheduling problems?

Column or Variable Generation

The problem of putting together a set of pieces of work into a single duty, that is a column or variable of problem (LP-MP), is formalized as a

Resource Constrained Shortest Path Problem

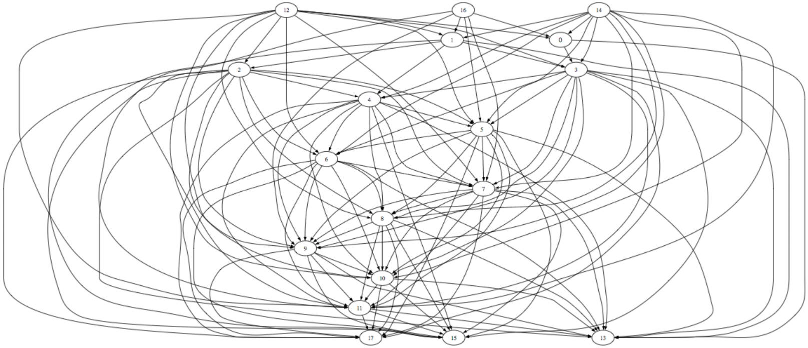
Example 12 pieces of work, 3 depots

ID	Da	A	Inizio	Fine
0	NETTPO	RMANAG	04:30	06:20
1	NETTPO	RMLAUREN	04:40	06:20
2	RMLAUREN	NETTPO	06:20	08:15
3	APRILI	LATINA	07:25	08:05
4	ANZICO	NETTPO	13:00	13:40
5	NETTPO	ANZIO	14:00	14:25
6	ANZIO	NETTPO	14:30	14:50
7	NETTPO	ANZIO	14:50	15:20
8	ANZIO	NETTPO	15:30	16:00
9	NETTPO	ANZIO	16:00	16:20
10	ANZIO	NETTPO	16:30	16:55
11	NETTPO	ANZIO	17:30	18:00

Resource Constraint Shortest Path

Let $G = (N, A)$ be the compatibility graph, weighted, directed, and acyclic:

- $N = P \cup \{\{s^h, t^h\} | h \in D\}$ a node for each PoW, and a pair of nodes for each depot
- A has an arc for each pair (i, j) of compatible PoW, and (s^h, i) (pull-out) and (i, t^h) (pull-in) $\forall h \in D$ and $i \in P$

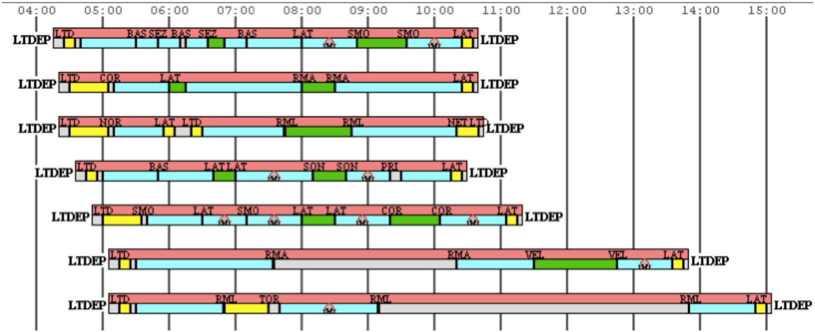


Resource Constraint Shortest Path

- $N = P \cup \{\{s^h, t^h\} | h \in D\}$
- A has an arc for each pair (i, j) of compatible PoW, and (s^h, i) (pull-out) and (i, t^h) (pull-in) $\forall h \in D$ and $i \in P$
- each arc (i, j) has associated a set of resources r_{ij}^k, for each $k \in K$, e.g. **working time**, driving time, and break time (other resources may be used to model working regulation)

	NEDEP	ANZICO	12:35	12:55	VAV
4	ANZICO	NETTPO	13:00	13:40	PG
5	NETTPO	ANZIO	14:00	14:25	PG
6	ANZIO	NETTPO	14:30	14:50	PG
7	NETTPO	ANZIO	14:50	15:20	PG
8	ANZIO	NETTPO	15:30	16:00	PG
9	NETTPO	ANZIO	16:00	16:20	PG
10	ANZIO	NETTPO	16:30	16:55	PG
11	NETTPO	ANZIO	17:30	18:00	PG
	ANZIO	NEDEP	18:00	18:10	VAV
			durata:	5:35	

Example of Crew Schedule (Resources)



Resources:

1. spread time (red)
2. driving time (light blue), corresponds to PoW
3. *out-of-service* time (yellow)
4. long break (grey)
5. breaks (green), very important how they are located

Duty Generation: Pricing Problem

- Duties (or shifts) with max duration between 4h30 (270m) and 6h30 (390m), with a maximum driving time of 5h30 (330m).
- For each interval of 4h30m (270 minutes), inside a duty, there must be at least a break of 15 minutes and at least a break of 30 minutes.
- The cost of each duty is determined by the minutes out of service.

We lay on every arc $(i, j) \in A$ the values:

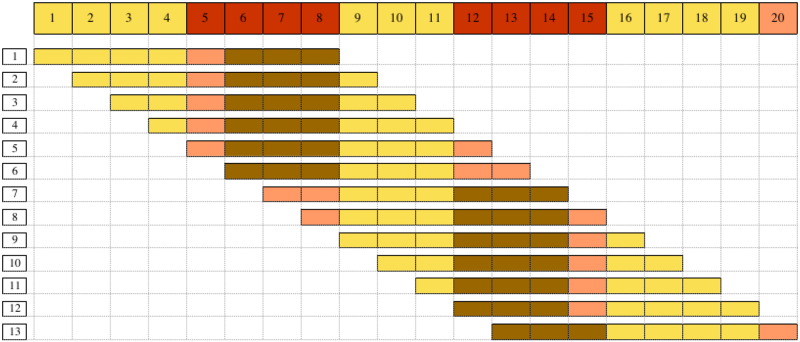
- PG : driving minutes
- FS : minutes of out of service
- PD : minutes of break at the depot
- T1 : number of breaks of type 1 (30 minutes)
- T2 : number of breaks of type 2 (15 minutes)

Pricing Problem MIP Model

$$
\begin{aligned}
\min \quad & \left(1 + \frac{1}{500} \sum_{ij \in A} t_{ij}^{FS} x_{ij}\right) - \sum_{i \in P} \bar{\pi}_i y_i \\
\text{s.t.} \quad & \sum_{ij \in A} x_{ij} = y_i, \sum_{ji \in A} x_{ij} = y_i, \quad \forall i \in N \setminus \{s, t\}, \\
& \sum_{ij \in A} x_{ij} + \sum_{ji \in A} x_{ij} = b_i, \quad \forall i \in \{s, t\}, \\
& \sum_{ij \in A} t_{ij}^{PG} x_{ij} + \sum_{i \in P} t_i^{PG} y_i \leq t^{MAX-PG}, \\
& \sum_{ij \in A} (t_{ij}^{PG} + t_{ij}^{FS} + t_{ij}^{PD}) x_{ij} + \sum_{i \in P} t_i^{PG} y_i \geq t^{MIN}, \\
& \sum_{ij \in A} (t_{ij}^{PG} + t_{ij}^{FS} + t_{ij}^{PD}) x_{ij} + \sum_{i \in P} t_i^{PG} y_i \leq t^{MAX}, \\
& + \text{ Vincolo delle Sequenze}, \\
& x_{ij} \in \{0, 1\}, \ \forall (i, j) \in A, \quad y_i \in \{0, 1\}, \forall i \in P.
\end{aligned}
$$

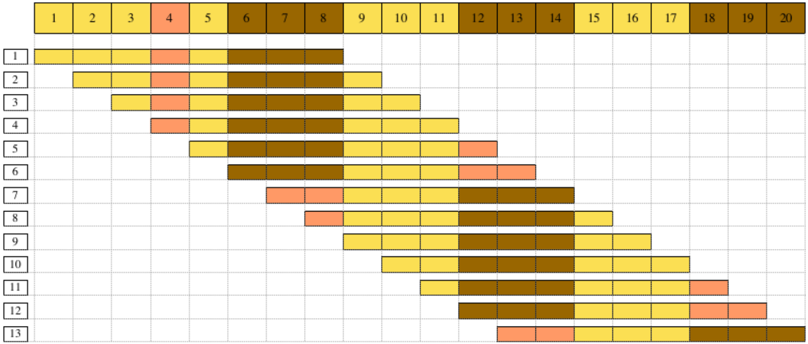
Sequence Constraint: Example

Let's assume to have a duty with 20 units of time, and two types of breaks, one that lasts one unit and one 3 units of time. Every 8 units we want at least one break of each type.



Sequence Constraint: Example

Let's assume to have a duty with 20 units of time, and two types of breaks, one that lasts one unit and one 3 units of time. Every 8 units we want at least one break of each type.



Shortest Path Problem with Resource Constraints

Most CG applications:

- master problem is a (possibly generalized) set partitioning or set covering problem with side constraints (variables are associated with vehicle routes or crew schedules).
- these route and schedule variables are generated by one or several subproblems, each of them corresponding to a shortest path problem with resource constraints (SPPRC) or one of its variants.
- because SPPRC does not possess the integrality property the column generation approach can derive tighter bounds than those obtained from the linear relaxation of arc-based formulations.
- there exist efficient algorithms at least for some important variants of the SPPRC.

Note: different names in the literature:

- shortest path problem with resource constraints (SPPRC)
- resource constrained shortest path problem (RCSPP)
- constrained shortest path problem (CSPP)

With respect to the classical shortest path problem, the SPPRC is complicated by a description of feasible paths:

1. feasibility w.r.t. resources and
2. feasibility w.r.t. path-structural constraints.

Moreover, non-linear cost functions can alone also complicate the classical shortest path problem to the point of not being anymore polynomially solvable.

Consider SPPRC on a simple (no-multiple arcs) digraph $D = (V, A)$:

- An elementary path is a path in which all nodes are pairwise different (as opposed to a cycle)
- the requirement of allowing only elementary paths makes the problem NP-hard
- with no-elementary paths and without other path-structural constraints the problem is solvable in pseudo-polynomial time (Note in acyclic graphs paths are elementary in any case.)

Solution algorithms are labeling algorithms, that is, dynamic programming algorithms with paths encoded by labels (aka, records).

[S. Irnich, G. Desaulniers, 2005]

Dijkstra Algorithm $\quad e(P)$ return the ending vertex of the path P

Procedure LabellingAlgorithm($N = (V, A), s, g$) initialize the open list $\mathcal{Q}$ and closed list $\mathcal{C}$;
initialize $\ell_r = ((), \infty)$;
insert $\ell_s = ((s), 0)$ into $\mathcal{Q}$;
while $\mathcal{Q}$ is not empty **do**
    $\ell \leftarrow$ retrieve and remove the cheapest label from $\mathcal{Q}$;
    if $c(\ell) > c(\ell_r)$ **then break**; $\quad \triangleright$ termination criterion
    if $e(\ell) = t$ and $c(\ell) < c(\ell_r)$ **then** $\quad \triangleright$ $t \equiv$ target node
        $\ell_r \leftarrow \ell$;
        continue;
    foreach node v such that uv in A **do**
        if v is in $\mathcal{C}$ **then continue**;
        $\ell' \leftarrow$ label at v expanded from ℓ;
        if label ℓ'' already exists at v **then**
            if $(c(\ell'') > c(\ell'))$ **then** $\quad \triangleright$ ℓ'' is dominated
                remove ℓ'' from $\mathcal{Q}$;
            else if $c(\ell') > c(\ell'')$ **then** $\quad \triangleright$ ℓ' is dominated
                continue;
        insert ℓ' into $\mathcal{Q}$;
    insert $e(\ell)$ into $\mathcal{C}$;
return $P(\ell_r)$ and $c(\ell_r)$;

Generic Dynamic Programming SPPRC Algorithm

---

set $\mathcal{U} = \{(s)\}$ and $\mathcal{P} = \emptyset$ ▷ Initialize;
while $\mathcal{U} \neq \emptyset$ **do**
  Choose a path $Q \in \mathcal{U}$ ▷ Path extension step ;
  Remove Q from $\mathcal{U}$;
  for arcs $(e(Q), w) \in A$ of the forward star of $e(Q)$ **do**
    if (Q, w) is a feasible path wrt resource vectors **then**
      Add (Q, w) to $\mathcal{U}$;
  Add Q to $\mathcal{P}$;
  if any condition **then**
    Apply dominance algorithm to paths from $\mathcal{U} \cup \mathcal{P}$ ending at some node v;
Filter $\mathcal{P}$, i.e., identify a solution $S \subseteq \mathcal{P}$;