Extended Formulations, Column Generation, and stabilization:
synergies in the benefit of large scale applications

François Vanderbeck

University of Bordeaux        INRIA Bordeaux-Sud-Ouest

Co-authors: A. Pessoa, R. Sadykov, E. Uchoa, L.A. Wolsey

EURO/INFORMS, Rome, July 2013


**An approach based on an extended formulation**

- An **EASY WAY** to bring-in combinatorial structure.
- Its size can be coped with by **combining** ideas of
  - Restriction / Relaxation,
  - Benders projection, and
  - Dantzig-Wolfe dynamic generation.

- **With dynamic generation**, a small % of variables and constraints are needed; hence it **scales up** to real-life applications.

- Is well suited for **efficiency enhancement** features: **cuts** on lifted variables, **Dynamic Progr. state-space-relax.**, **red.-cost-fixing**.
1 Extented Formulations
   • Definitions
   • Interests
   • Coping with its large size

2 Dynamic Row-and-Column Generation
   • Methodology
   • Practical issues

3 Large-scale application
   • Freight transport by rail in Russia
1 Extented Formulations
   - Definitions
     - Interests
     - Coping with its large size

2 Dynamic Row-and-Column Generation
   - Methodology
   - Practical issues

3 Large-scale application
   - Freight transport by rail in Russia
(CO) ≡ \min\{c(s) : s \in S\}
where S is the “discrete” set of feasible solutions.
Combinatorial Optimization Problem

\[(CO) \equiv \min \{c(s) : s \in S\}\]

where \(S\) is the “discrete” set of feasible solutions.

**Formulation**

A polyhedron \(P = \{x \in \mathbb{R}^n : Ax \geq a\}\) is a formulation for \((CO)\) iff

\[\min \{c(s) : s \in S\} \equiv \min \{cx : x \in P_I = P \cap \mathbb{N}^n\}.\]
A formulation is typically not unique

$P$ and $P'$ can be alternative formulations for $(CO)$ if

$$(CO) \equiv \min\{cx : x \in P \cap \mathbb{N}^n\} \equiv \min\{c'x' : x' \in P' \cap \mathbb{N}^{n'}\}$$

warning: can expressed in different variable-spaces.
Stronger formulation (in the same space)

Formulation $P' \subseteq \mathbb{R}^n$ is a **stronger** than $P \subseteq \mathbb{R}^n$ if $P' \subset P$. Then,

$$\min\{cx' : x' \in P'\} \geq \min\{cx : x \in P\}$$
The Convex hull of an IP set, $P_I$

$\text{conv}(P_I)$ is the smallest closed convex set containing $P_I$.

$\text{conv}(P_I)$ is an ideal polyhedron / formulation

If $P_I$ is defined by rational data, $\text{conv}(P_I)$ is a polyhedron.
Given an initial **compact formulation**: 

\[
\begin{align*}
    x_1 & = 1, 2, 3, 4 \\
    x_2 & = 5
\end{align*}
\]
Extended Formulations

François Vanderbeck

Extended Formulations & Column Generation: Synergies
The Projection

of \( Q = \{(x, w) \in \mathbb{R}^{n+e} : Gx + Hw \geq d\} \) on the \( x \)-space is:

\[
\text{proj}_x(Q) := \{x \in \mathbb{R}^n : \exists w \in \mathbb{R}^e \text{ such that } (x, w) \in Q\}.
\]
The Projection

of \( Q = \{(x, w) \in \mathbb{R}^{n+e} : Gx + Hw \geq d\} \) on the \( x \)-space is:

\[
\text{proj}_x(Q) := \{x \in \mathbb{R}^n : \exists w \in \mathbb{R}^e \text{ such that } (x, w) \in Q\}.
\]

Farka's Lemma

Given \( \tilde{x} \),

\[
\{w \in \mathbb{R}^n_+ : Hw \geq (d - G \tilde{x})\} \neq \emptyset
\]

if and only if

\[
\forall v \in \mathbb{R}^m_+ : vH \leq 0, \quad v(d - G \tilde{x}) \leq 0.
\]

Hence, a polyhedral description of the projection in the \( x \)-space is:

\[
\text{proj}_x(Q) = \{x \in \mathbb{R}^n : v^j (d - Gx) \leq 0 \quad j \in J\}
\]

\( \{v^j\}_{j \in J} \), exteme rays. of \( \{v \in \mathbb{R}^m_+ : vH \leq 0\} \).
An extended formulation for an IP set $P_I \subseteq \mathbb{N}^n$ is a polyhedron $Q = \{(x, w) \in \mathbb{R}^{n+e} : Gx + Hw \geq d\}$ such that $P_I = \text{proj}_x(Q) \cap \mathbb{N}^n$. 
An extended formulation for an IP set $P_I \subseteq \mathbb{N}^n$ is a polyhedron $Q = \{(x, w) \in \mathbb{R}^{n+e} : Gx + Hw \geq d\}$ such that $P_I = \text{proj}_x(Q) \cap \mathbb{N}^n$. 
A **tight** extended formulation for an IP set $P_I \subseteq \mathbb{N}^n$ is a polyhedron $Q = \{(x, w) \in \mathbb{R}^{n+e} : Gx + Hw \geq d\}$ such that $\text{conv}(P_I) = \text{proj}_x(Q)$. 

François Vanderbeck

Extended Formulations & Column Generation: Synergies 16/70
An extended IP-formulation for an IP set $P_I \subseteq \mathbb{N}^n$
is an IP-set $Q_I = \{(x, w) \in \mathbb{R}^n \times \mathbb{N}^e : Gx + Hw \geq b\}$ s.t.

$$P_I = \text{proj}_x Q_I.$$
Change of variables: $x = T w$
An extended formulation based on a **change of variables**: $x = Tw$.

$$Q = \{(x, w) \in \mathbb{R}^{n+e} : Tw = x, \quad Hw \geq h\}.$$ 

Then,

$$\text{proj}_x(Q) = T(W) := \{x = Tw \in \mathbb{R}^n : Hw \geq h, w \in \mathbb{R}^e \}.$$ 

A reformulation for an IP-set $P_I \subseteq \mathbb{N}^n$ is a polyhedron $W$ along a linear transformation, $x = Tw$, s.t.

$$P_I = T(W) \cap \mathbb{N}^n$$

A **IP-reformulation** for an IP-set $P_I \subseteq \mathbb{N}^n$ is an IP-set $W_I = W \cap \mathbb{N}^e$ along a linear transformation, $x = Tw$, s.t.,

$$P_I = T(W_I)$$
Polyhedron $\text{conv}(P_I)$ can be defined by its extreme points and rays:

$$Q = \{(x, \lambda, \mu) \in \mathbb{R}^n \times \mathbb{R}_+^{|G|} \times \mathbb{R}_+^{|R|} : x = \sum_{g \in G} x_g \lambda_g + \sum_{r \in R} v_r \mu_r, \sum_{g \in G} \lambda_g = 1\}$$

change of variables: $x = X \lambda + V \mu$. 
Example: Steiner Tree
Example: Steiner Tree

Extended Formulations

D
2 3
4 5
9 10 11 12 13
6
8
7
1
A
B
C

Special cases:
- $T = \{ i \}$: shortest path from $r$ to $i$
- $T = V \{ r \}$: minimum cost spanning tree

François Vanderbeck
Extended Formulations & Column Generation: Synergies 22/70
Example: Steiner Tree

Special cases:

- \( T = \{i\} \): shortest path from \( r \) to \( i \)
- \( T = V \setminus \{r\} \): minimum cost spanning tree
**Steiner Tree: Arc flow formulation**

**Variables**
- \( x_{ij} \in \{0, 1\} \) — arc \((i, j)\) is used or not
- \( y_{ij} \in \mathbb{N} \) — number of connections going through \((i, j)\)

\[
\begin{align*}
\min & \sum_{(i,j) \in A} c_{ij} x_{ij} \\
\text{subject to} & \sum_{j \in V^+(r)} y_{rj} = |T| \\
\sum_{j \in V^-(i)} y_{ji} - \sum_{j \in V^+(i)} y_{ij} &= 1 \quad i \in T \\
\sum_{j \in V^-(i)} y_{ji} - \sum_{j \in V^+(i)} y_{ij} &= 0 \quad i \in V \setminus (T \cup \{r\}) \\
y_{ij} &\leq |T| x_{ij} \quad (i, j) \in A \\
y &\in \mathbb{R}_+^{|A|} \\
x &\in \{0, 1\}^{|A|}
\end{align*}
\]
Steiner Tree: Multi commodity flow formulation

Variable splitting

- \( w_{ij}^t \in \{0, 1\} \) — arc \((i, j)\) is used to connect terminal \( t \)
- \( y_{ij} = \sum_k w_{ij}^t \) — defines a linear transformation

\[
\begin{align*}
\min & \sum_{(i,j) \in A} c_{ij} x_{ij} \\
\sum_{j \in V^+(r)} w_{rj}^t &= 1 \quad t \in T \\
\sum_{j \in V^-(i)} w_{ji}^t - \sum_{j \in V^+(i)} w_{ij}^t &= 1 \quad i = t \in T \\
\sum_{j \in V^-(i)} w_{ji}^t - \sum_{j \in V^+(i)} w_{ij}^t &= 0 \quad i \in V \setminus \{r, k\}, \ t \in T \\
\end{align*}
\]

- \( w_{ij}^t \leq x_{ij} \ (i, j) \in A, \ t \in T \)
- \( w \in \mathbb{R}^{K \times |A|}_+ \)
- \( x \in \{0, 1\}^{|A|} \)
Example: Steiner Tree

François Vanderbeck
Extended Formulations & Column Generation: Synergies
projection in the $x$-space

\[
\begin{align*}
\min \quad & \sum_{(i,j) \in A} c_{ij} x_{ij} \\
\sum_{(i,j) \in \delta^+(S)} x_{ij} & \geq 1 \quad S \ni r, T \setminus S \neq \emptyset \\
x & \in \{0, 1\}^{|A|},
\end{align*}
\]
Steiner Tree: Network design formulation

projection in the $x$-space

\[
\begin{align*}
\min & \sum_{(i,j) \in A} c_{ij}x_{ij} \\
\sum_{(i,j) \in \delta^+(S)} x_{ij} & \geq 1 \ S \ni r, T \setminus S \neq \emptyset \\
x & \in \{0, 1\}^{|A|},
\end{align*}
\]

Note: This projection onto the $x$ space
- has the **same LP value** than the multi-commodity flow formulation
- is **better than** the initial **compact** aggregate flow formulation.
Extended Formulations

Ways to obtain extended formulations

- **Variable Splitting**
  - Multi-Commodity Flow: \( x_{ij} = \sum_k x_{ij}^k \)
  - Unary expansion: \( x = \sum_{q=0}^u q w_q, \sum_{q=0}^u w_q = 1, w \in \{0, 1\}^{u+1} \)
  - Binary expansion: \( x = \sum_{p=0}^{\log|u|} w_p, , w \in \{0, 1\}^{\log|u|} \)

- **Dynamic Programming** Solver \( \rightarrow \) Network Flow LP [Martin et al]

- **Separation** is easy \( \rightarrow \) Separation LP [Martin et al]

- **Reduced coefficient / basis** reformulations [Aardal et al]

- **Union of Polyhedra** [Balas]

- \( \ldots \)
Single machine scheduling problem (with integer data):

\[ S_j \geq S_i + p_i \text{ or } S_i \geq S_j + p_j \quad \forall \ i, j \]

requires big M formulation:

\[ S_j \geq S_i + p_i - M(1 - x_{ij}). \]
Unary expansion: Time-Indexed Formulation

Single machine scheduling problem (with integer data):

\[
S_j \geq S_i + p_i \text{ or } S_i \geq S_j + p_j \quad \forall \; i, j
\]

Change of variables:

\[
S_j = \sum_t t \; w_{jt}
\]

with \( w_{jt} = 1 \) iff job \( j \) starts at the beginning of \([t, t + 1]\).

\[
\sum_{j \in J} w_{j0} = 1
\]

\[
\sum_{j \in J} w_{jt} - \sum_{j \in J} w_{j,t-p_j} = 0 \quad \forall \; t \geq 1
\]
Ways to obtain extended formulations

- **Variable Splitting**
  - Multi-Commodity Flow: \( x_{ij} = \sum_k x_{ij}^k \)
  - Unary expansion: \( x = \sum_{q=0}^{u} q w_q, \sum_{q=0}^{u} w_q = 1, w \in \{0, 1\}^{u+1} \)
  - Binary expansion: \( x = \sum_{p=0}^{\log u} w_p, w \in \{0, 1\}^{\log u} \)

- Dynamic Programming Solver \( \rightarrow \) Network Flow LP [Martin et al]
- Separation is easy \( \rightarrow \) Separation LP [Martin et al]
- Reduced coefficient / basis reformulations [Aardal et al]
- Union of Polyhedra [Balas]
- ...
DP based reformulation: the knapsack example

\[
\max \left\{ \sum p_i x_i : \sum a_i x_i \leq b, x_i \in \mathbb{N} \right\}
\]

- **DP Recursion:** \( V(c) = \max_{i=1,\ldots,n : c \geq a_i} \{ V(c - a_i) + p_i \} \)

- **in LP form:**

\[
\begin{align*}
\min V(b) \\
V(c) - V(c - a_i) & \geq p_i & i = 1, \ldots, n, c = a_i, \ldots, b \\
V(0) & = 0
\end{align*}
\]

- **its Dual:** “longest path problem”

[Diagram of a network with vertices labeled 0 to 7, showing paths and weights.]
DP based reformulation: the knapsack example

\[
\max \left\{ \sum_i p_i x_i : \sum_i a_i x_i \leq b, x_i \in \mathbb{N} \right\}
\]

- **DP Recursion:** \( V(c) = \max_{i=1,\ldots,n:c\geq a_i} \{ V(c-a_i) + p_i \} \)

- **in LP form:**
  \[
  \begin{align*}
  \min & \quad V(b) \\
  & \quad V(c) - V(c-a_i) \geq p_i \quad i = 1, \ldots, n, \ c = a_i, \ldots, b \\
  & \quad V(0) = 0
  \end{align*}
  \]

- **its Dual:** “longest path problem”

\[
\begin{align*}
\max & \quad \sum_{j=1}^{n} \sum_{r=0}^{b-a_i} c_i w_{ic} \\
\sum_i w_{ic} & = 1 \quad c = 0 \\
\sum_i w_{ic} - \sum_i w_{i,c-a_i} & = 0 \quad c = 1, \ldots, b - 1 \\
\sum_i w_{i,c-a_i} & = 1 \quad c = b \\
w_{ic} & \geq 0 \quad i = 1, \ldots, n; \ c = 0, \ldots, b - a_i
\end{align*}
\]
DP based reformulation: Multi-Echelon Lot-Sizing

Variables

- $x_{e,t}$ — production of intermediate product of echelon $e$ in period $t$
- $s_{e,t}$ — stock of echelon $e$ product at the end of period $t$

\[
x_{e,t} + s_{e,t-1} = x_{e+1,t} + s_{e,t} \quad \text{for } e = 1, \ldots, E - 1
\]
\[
x_{e,t} + s_{e,t-1} = d_t + s_{e,t} \quad \text{for } e = E
\]
Dominance property

\[ \exists \text{ opt solution where } x_{e,t} \cdot s_{e,t-1} = 0 \ \forall e, t, \Rightarrow \text{ production plan is a tree:} \]

\[ \begin{array}{c}
e = 1 \\
e = 2 \\
e = 3 \\
\end{array} \]

\[ t \]
Dominance property

∃ opt solution where $x_{e,t} \cdot s_{e,t-1} = 0 \ \forall e,t, \Rightarrow$ production plan is a tree:

Dynamic programming

State $(e,t,a,b)$ corresponds to accumulating at echelon $e$ in period $t$ a production covering exactly the demand of periods $a, \ldots, b$.

$$V(e,t,a,b) = \min \{ V(e,t+1,a,b), \min_{l=a,\ldots,b} \{ V(e+1,t,a,l) + c_{et}^k D_{al}^k + f_{et}^k + V(e,t+1,l+1,b) \} \}$$
DP Recursion:

\[ V(e, t, a, b) = \min \{ V(e, t + 1, a, b), \]
\[ \min_{l=a,...,b} \{ V(e + 1, t, a, l) + c_{et}^k D_{al}^k + f_{et}^k + V(e, t + 1, l + 1, b) \} \} \]

in LP form:

\[
\max V(1, 1, 1, T) \\
V(e, t, a, b) \leq V(e, t + 1, a, b) \forall e, t, a, b \\
V(e, t, a, b) \leq V(e + 1, t, a, l) + c_{et}^k D_{al}^k + f_{et}^k + V(e, t + 1, l + 1, b) \forall e, t, a, b, l \\
V(E + 1, t, a, b) = 0 \forall t, a, b
\]

its Dual: flow on hyper-arcs

\[ w_{e,t,a,l,b} = 1 \] if at echelon \( e \) in period \( t \) production covers demands from period \( a \) to period \( l \), while the rest of demand up to \( b \), shall be covered in the future.
[Martin et al OR90] When a problem can be solved by dynamic programming,

\[ V(l) = \min_{(J,l) \in A} \left\{ \sum_{j \in J} V(j) + c(J,l) \right\}, \]

an extended formulation consist in modeling a decision tree in an hyper-graph.
Ways to obtain extended formulations

- **Variable Splitting**
  - Multi-Commodity Flow: $x_{ij} = \sum_k x_{ij}^k$
  - Unary expansion: $x = \sum_{q=0}^u q w_q$, $\sum_{q=0}^u w_q = 1$, $w \in \{0, 1\}^{u+1}$
  - Binary expansion: $x = \sum_{p=0}^{\log \lceil u \rceil} w_p$, $w \in \{0, 1\}^\log u$

- **Dynamic Programming Solver** → **Network Flow LP** [Martin et al]

- **Separation is easy** → **Separation LP** [Martin et al]

- **Reduced coefficient / basis reformulations** [Aardal et al]

- **Union of Polyhedra** [Balas]

- …
Outline

1. Extented Formulations
   - Definitions
   - Interests
   - Coping with its large size

2. Dynamic Row-and-Column Generation
   - Methodology
   - Practical issues

3. Large-scale application
   - Freight transport by rail in Russia
**Extended Formulation: Interests**

**Improved formulation** *(better LP bound & rounding heuristic)*

- Extra variables
  - Tighter relations,
  - Linearisation

---

*Vehicle routing:*

\[
x_a = \sum_{l=0}^{C} w_{al} = \begin{cases} 1 & \text{if vehicle on arc } a \text{ with load } l, \\ 0 & \text{otherwise} \end{cases}
\]

\[
\sum_l \sum_a \in \delta^- (i) w_{al} - \sum_l \sum_a \in \delta^+ (i) w_{al} = d_i \rightarrow \text{knapsack cover cuts.}
\]
Extended formulation: Interests

1. **Improved formulation** (better LP bound & rounding heuristic)
2. **Simpler formulation** (captures the combinatorial structure)

- Extra variables
- Fewer constraints
- Structure built into var. definitions

Vehicle routing:

\[
\sum_{l=0}^{C} w_{a_l} = 1 \text{ if vehicle on arc } a \text{ with load } l,
\]

\[
\sum_{l} \sum_{a \in \delta^{-}(i)} w_{a_l} - \sum_{l} \sum_{a \in \delta^{+}(i)} w_{a_l} = d_i \rightarrow \text{knapsack cover cuts.}
\]
Extended formulation: Interests

1. Improved formulation (better LP bound & rounding heuristic)
2. Simpler formulation (captures the combinatorial structure)
3. Direct use of a MIP-Solver (solved by standard tools)
Extended formulation: Interests

1. **Improved formulation** (better LP bound & rounding heuristic)
2. **Simpler formulation** (captures the combinatorial structure)
3. **Direct use of a MIP-Solver** (solved by standard tools)
4. **Rich variable space** (to express cuts or branching)

**Vehicle routing:**

\[
\begin{align*}
x_a &= \sum_{l=0,\ldots,C} c \, w^a_l \\
w^a_q = 1 & \text{ if vehicle on arc } a \text{ with load } l,
\end{align*}
\]

\[
\sum_l \sum_{a \in \delta^{-}(i)} l w^a_l - \sum_l \sum_{a \in \delta^{+}(i)} l w^a_l = d_i
\]

→ knapsack cover cuts.  

[Uchoa]
1. Extented Formulations
   - Definitions
   - Interests
   - Coping with its large size

2. Dynamic Row-and-Column Generation
   - Methodology
   - Practical issues

3. Large-scale application
   - Freight transport by rail in Russia
Coping with size: Related work on Multi-Route-VRP

[Macedo, Alves, Valerio de Carvalho, Clautiaux, Hanafi. EJOR2011]

Variables

- \( w_{st}^r \) — nb of vehicles using route \( r \) that starts in \( s \) and ends in \( t \)

\[
\begin{align*}
\min \sum_{rst} c_{st}^r w_{st}^r \\
\sum_{r \in i, s, t} w_{st}^r &= 1 \quad \forall \text{order } i \\
\sum_{rt} w_{0t}^r &= V \\
\sum_{r,t} w_{rt}^r - \sum_{r,s} w_{s\tau}^r &= 0 \quad \forall \tau > 1 \\
w_{st}^r &\in \{0, 1\} \quad r, s, t
\end{align*}
\]
Extented Formulations

Coping with size: Related work on Multi-Route-VRP

[Macedo, Alves, Valerio de Carvalho, Clautiaux, Hanafi. EJOR2011]

Relaxation

- round-up start time: $S = \{s : \lceil s \rceil = S\}$
- round-down termination time: $T = \{t : \lfloor t \rfloor = T\}$
- define relaxed route arcs: $w_{S,T}^r = \sum_{s \in S, t \in T} w_{s,r}^r$.

**Automatic Desaggragation Algorithm:**

1. Solve problem over aggregate time periods.
2. Try to build a desaggregate feasible solution.
3. If it fails, desaggregate the time period of conflict.
Mastering the size extended formulations

1. Use of a **relaxation** [Van Vyve & Wolsey MP06]
   - Drop some of the constraints
   - Aggregate commodities/nodes (down-rounding of durations)
   - Partial reformulation

   → static outer approximation of the extended formulation
1. **Use of a relaxation** [Van Vyve & Wolsey MP06]
   - Drop some of the constraints
   - Aggregate commodities/nodes (down-rounding of durations)
   - Partial reformulation
   \[\rightarrow \text{static outer approximation of the extended formulation}\]

2. **Use of a restriction**
   - define only some transitions in a dynamic program
   - up-rounding of durations
   \[\rightarrow \text{static inner approximation of the extended formulation}\]
1. Use of a relaxation [Van Vyve & Wolsey MP06]
   - Drop some of the constraints
   - Aggregate commodities/nodes (down-rounding of durations)
   - Partial reformulation
   \[\rightarrow\text{static outer approximation of the extended formulation}\]

2. Use of a restriction
   - define only some transitions in a dynamic program
   - up-rounding of durations
   \[\rightarrow\text{static inner approximation of the extended formulation}\]

3. Projection: Benders’ cuts (applying Farkas Lemma)
   \[\rightarrow\text{dynamic outer approximation of the extended formulation}\]
Mastering the size extended formulations

1. Use of a relaxation [Van Vyve & Wolsey MP06]
   - Drop some of the constraints
   - Aggregate commodities/nodes (down-rounding of durations)
   - Partial reformulation
   \[\text{\rightarrow static outer approximation of the extended formulation}\]

2. Use of a restriction
   - define only some transitions in a dynamic program
   - up-rounding of durations
   \[\text{\rightarrow static inner approximation of the extended formulation}\]

3. Projection: Benders’ cuts (applying Farkas Lemma)
   \[\text{\rightarrow dynamic outer approximation of the extended formulation}\]

   \[\text{\rightarrow dynamic inner approximation of the extended formulation}\]
Outline

1. Extented Formulations
   - Definitions
   - Interests
   - Coping with its large size

2. Dynamic Row-and-Column Generation
   - Methodology
   - Practical issues

3. Large-scale application
   - Freight transport by rail in Russia
Extended formulation based on a subproblem

**Original formulation**

\[
[F] \equiv \min \left\{ c x \right\} \\
A x \geq a \\
B x \geq b \\
x \in \mathbb{N}^n
\]

**Subproblem**

\[
P \equiv \left\{ B x \geq b \right\} \\
x \in \mathbb{R}_+^n
\]

\[P_1 = P \cap \mathbb{N}^n\]

Decomposition + SP Reformulation
Dynamic Row-and-Column Generation

Extended formulation based on a subproblem

<table>
<thead>
<tr>
<th>Original formulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( [F] \equiv \min \left{ cx \right} )</td>
</tr>
<tr>
<td>( Ax \geq a )</td>
</tr>
<tr>
<td>( Bx \geq b )</td>
</tr>
<tr>
<td>( x \in \mathbb{N}^n )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subproblem</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \equiv \left{ \begin{array}{c} Bx \geq b \ x \in \mathbb{R}_+^n \end{array} \right} )</td>
</tr>
<tr>
<td>( P_I = P \cap \mathbb{N}^n )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Assumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subproblem ( P_I ) admits an <strong>IP-reformulation</strong> ( W_I ): ( \exists ) polyhedron</td>
</tr>
<tr>
<td>( W = \left{ Hw \geq h, w \in \mathbb{R}_+^e \right} )</td>
</tr>
<tr>
<td>and a <strong>linear transformation</strong> ( T ), such that</td>
</tr>
<tr>
<td>( P_I = \text{proj}_x(W_I) = T(W_I) = \left{ x = Tw : Hw \geq h, w \in \mathbb{N}^e \right} )</td>
</tr>
</tbody>
</table>
### Extended formulation based on a subproblem

<table>
<thead>
<tr>
<th>Original formulation</th>
<th>Extended reformulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>([F] \equiv \min \left{ c x \right} )</td>
<td>([R] \equiv \min \left{ c T w \right} )</td>
</tr>
<tr>
<td>(A x \geq a)</td>
<td>(A T w \geq a)</td>
</tr>
<tr>
<td>(B x \geq b)</td>
<td>(H w \geq h)</td>
</tr>
<tr>
<td>(x \in \mathbb{N}^n)</td>
<td>(w \in \mathbb{N}^e)</td>
</tr>
</tbody>
</table>

### Assumption

Subproblem \(P_I\) admits an **IP-reformulation**, \(W_I\): \(\exists\) polyhedron

\[ W = \{ H w \geq h, w \in \mathbb{R}_+^e \} \]

and **a linear transformation**, \(T\), s.t.

\[ P_I = \text{proj}_x(W_I) = T(W_I) = \left\{ x = Tw : H w \geq h, w \in \mathbb{N}^e \right\} \]
Extended formulation based on a subproblem

**Original formulation**

\[ \mathcal{F} \equiv \min \left\{ c \, x \right\} \]
\[ A \, x \geq a \]
\[ B \, x \geq b \]
\[ x \in \mathbb{N}^n \}

**Extended reformulation**

\[ \mathcal{R} \equiv \min \left\{ c \, T \, w \right\} \]
\[ A \, T \, w \geq a \]
\[ H \, w \geq h \]
\[ w \in \mathbb{N}^e \}

**Special case: Dantzig-Wolfe Reformulation**

\[ \mathcal{M} \equiv \min \left\{ \sum_{g \in G} c \, x^g \, \lambda_g \right\} \]
\[ \sum_{g \in G} A \, x^g \, \lambda_g \geq a \]
\[ \sum_{g \in G} \lambda_g = 1 \]
\[ \lambda \in \{0, 1\}^{|G|} \}

Applying Minkowski

\[ x = \sum_{g \in G} x^g \, \lambda_g \]
Dynamic Row-and-Column Generation

Extended formulation based on a subproblem

Original formulation

\[ [F] \equiv \min \left\{ c x \right\} \]
\[ A x \geq a \]
\[ B x \geq b \]
\[ x \in \mathbb{N}^n \]

Extended reformulation

\[ [R] \equiv \min \left\{ c^T w \right\} \]
\[ A^T w \geq a \]
\[ H w \geq h \]
\[ w \in \mathbb{N}^e \]

Column-and-row generation

- Dynamic generation of the variables of [R] by bunch, solving the column generation subproblem of [M] over \( W_I \).
- Adding rows that become active.
Restricted reformulations

\(\bar{S} = \{w^s\}_{s \in \bar{S}}\): a subset of integer solutions to \(W_I\).

\(\bar{w}\) = restriction of \(w\) to the non-zero components in \(\bar{S}\).

\(\bar{G} = \{g \in G : x^g = T w^s, s \in \bar{S}\}\)

\[
\begin{align*}
[\bar{R}_{LP}] & \equiv \min \left\{ c \bar{T} \bar{w} \right\} \\
A \bar{T} \bar{w} & \geq a \\
\bar{H} \bar{w} & \geq \bar{h} \\
\bar{w} & \in \mathbb{R}^e_+ \}
\end{align*}
\]

\[
[\bar{M}_{LP}] \equiv \min \left\{ \sum_{g \in \bar{G}} c x^g \lambda_g \right\} \\
\sum_{g \in \bar{G}} A x^g \lambda_g & \geq a \\
\sum_{g \in \bar{G}} \lambda_g & = 1 \\
\lambda & \in \mathbb{R}^{||\bar{G}||}_+ \}
\]
Restricted reformulations

\( \overline{S} = \{ w^s \}_{s \in \overline{S}} \): a subset of integer solutions to \( W_I \).
\( \overline{w} = \) restriction of \( w \) to the non-zero components in \( \overline{S} \).
\( \overline{G} = \{ g \in G : x^g = T w^s, s \in \overline{S} \} \)

\[
[\overline{R}_{LP}] \equiv \min \left\{ c \overline{T} \overline{w} \right\} \quad \text{subject to} \quad \begin{align*}
A \overline{T} \overline{w} & \geq \quad a \\
\overline{H} \overline{w} & \geq \quad \overline{h} \\
\overline{w} & \in \quad \mathbb{R}_+^e
\end{align*}
\]

\[
[\overline{M}_{LP}] \equiv \min \left\{ \sum_{g \in \overline{G}} c x^g \lambda_g \right\} \quad \text{subject to} \quad \begin{align*}
\sum_{g \in \overline{G}} A x^g \lambda_g & \geq \quad a \\
\sum_{g \in \overline{G}} \lambda_g & = \quad 1 \\
\lambda & \in \quad \mathbb{R}_{+}^{\overline{G}}
\end{align*}
\]

**Proposition 1**

\( v[\overline{M}_{LP}] =_* v[\overline{R}_{LP}] \leq v[\overline{R}_{LP}] \leq v[\overline{M}_{LP}] \quad (\ast) \text{ if tight reformulation} \)
Column-and-row generation procedure

Step 1: Solve $[\overline{R}_{LP}]$ and collect the dual solution $\overline{\pi}$ associated to constraints $A \overline{T} \overline{z} \geq a$, only.

Step 2: Obtain a solution $w^*$ of the pricing problem:

$$\min \{(c - \overline{\pi}A)^T w : w \in W_I\}$$

Step 3: Compute the Lagrangian dual bound:

$$L(\overline{\pi}) \leftarrow \overline{\pi} a + (c - \overline{\pi}A)^T w^*, \beta \leftarrow \max\{\beta, L(\overline{\pi})\}.$$ 

If $v^{[\overline{R}_{LP}]} \leq \beta$, STOP.

Step 4: Update $\overline{S}$ by adding solution $w^*$ and iterate

Proposition 2

Either $v^{R}_{LP} \leq \beta$ (stopping condition),
or some of the components of $w^*$ have negative reduced cost in $[\overline{R}_{LP}]$. 
Example: parallel machine scheduling

\[ [R] \equiv \min \left\{ \sum_{jt} c_{jt} w_{jt} \right\} \]

\[ \sum_{t=0}^{T-p_j} w_{jt} = 1 \quad \forall j \in J \]

\[ \sum_{j \in J} w_{j0} = m \]

\[ \sum_{j \in J} w_{jt} - \sum_{j \in J} w_{j,t-p_j} = 0 \quad \forall t \geq 1 \]

\[ w_{jt} \in \{0, 1\} \quad \forall j, t \]

\[ [M] \equiv \min \left\{ \sum_{g \in G} c^g \lambda_g \right\} \]

\[ \sum_{g \in G} \sum_{t=0}^{T-p_j} w_{jt}^g \lambda_g = 1 \quad \forall j \in J \]

\[ \sum_{g \in G} \lambda_g = m \]

\[ \lambda_g \in \{0, 1\} \quad \forall g \in G \]
Solve the pricing subproblem (obtain a pseudo schedule)
1. Solve the pricing subproblem (obtain a pseudo schedule)

2. Disaggregate the subproblem solution in arc variables $w$. 
1. Solve the pricing subproblem (obtain a pseudo schedule)

2. Disaggregate the subproblem solution in arc variables $w$.

3. Add them to $\bar{R}$ along with the associated flow conservation constraints.
1. Solve the pricing subproblem (obtain a pseudo schedule)

2. Disaggregate the subproblem solution in arc variables $w$.

3. Add them to $\bar{R}$ along with the associated flow conservation constraints.

4. Solve the restricted extended formulation $\bar{R}$ and update dual prices.
### Machine scheduling: example of convergence

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Subproblem solution</th>
<th>Subproblem solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial solution</td>
<td><img src="image" alt="Initial Solution" /></td>
<td><img src="image" alt="Initial Solution" /></td>
</tr>
<tr>
<td>1</td>
<td><img src="image" alt="Iteration 1" /></td>
<td><img src="image" alt="Iteration 1" /></td>
</tr>
<tr>
<td>2</td>
<td><img src="image" alt="Iteration 2" /></td>
<td><img src="image" alt="Iteration 2" /></td>
</tr>
<tr>
<td>3</td>
<td><img src="image" alt="Iteration 3" /></td>
<td><img src="image" alt="Iteration 3" /></td>
</tr>
<tr>
<td>...</td>
<td><img src="image" alt="Iteration ..." /></td>
<td><img src="image" alt="Iteration ..." /></td>
</tr>
<tr>
<td>10</td>
<td><img src="image" alt="Iteration 10" /></td>
<td><img src="image" alt="Iteration 10" /></td>
</tr>
<tr>
<td>11</td>
<td><img src="image" alt="Iteration 11" /></td>
<td><img src="image" alt="Iteration 11" /></td>
</tr>
<tr>
<td>Final solution</td>
<td><img src="image" alt="Final Solution" /></td>
<td><img src="image" alt="Final Solution" /></td>
</tr>
</tbody>
</table>
Machine scheduling: recombination property

\[ \bar{S} = \{w^1, w^2\} \]
\( \bar{S} = \{ w^1, w^2 \}, \quad \hat{w} \in \bar{W} \setminus \text{conv}(w^1, w^2) \)
Machine Scheduling: numerical results

- Averages on 25 instances (OR-library) with $p_j \in [1, \ldots, 100]$.

<table>
<thead>
<tr>
<th>$m$</th>
<th>$n$</th>
<th>Cplex 12.1 for [R]</th>
<th>Column gen. for [M]</th>
<th>Column-and-row generation for [R]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25</td>
<td>7.1</td>
<td>337</td>
<td>0.9</td>
</tr>
<tr>
<td>1</td>
<td>50</td>
<td>132.6</td>
<td>1274</td>
<td>24.2</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>2332.0</td>
<td>8907</td>
<td>1764.4</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>4.1</td>
<td>207</td>
<td>0.3</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
<td>109.2</td>
<td>645</td>
<td>5.7</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>3564.4</td>
<td>2678</td>
<td>115.5</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>18.7</td>
<td>433</td>
<td>1.5</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>485.7</td>
<td>1347</td>
<td>27.9</td>
</tr>
<tr>
<td>4</td>
<td>200</td>
<td>$&gt;2h$</td>
<td>4315</td>
<td>409.4</td>
</tr>
</tbody>
</table>

- **#it**: number of column generation iterations
- **vars**: percentage of $w$ variables that are generated
- **cpu**: solution time, in seconds
### Machine Scheduling: results with stabilization

<table>
<thead>
<tr>
<th>$m$</th>
<th>$n$</th>
<th>Column gen. for [M]</th>
<th>Column-and-row generation for [R]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>#it</td>
<td>cpu</td>
</tr>
<tr>
<td>1</td>
<td>25</td>
<td>150</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>50</td>
<td>354</td>
<td>3.8</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>781</td>
<td>39.5</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>142</td>
<td>0.2</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
<td>323</td>
<td>1.7</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>715</td>
<td>17.3</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>287</td>
<td>0.6</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>638</td>
<td>8.7</td>
</tr>
<tr>
<td>4</td>
<td>200</td>
<td>1553</td>
<td>87.7</td>
</tr>
</tbody>
</table>
Multi-item Multi-echelon Lot-sizing: extended formulation

- $x_{et}^i, s_{et}^i$ — production/stock for item $i$ at echelon $e$ in period $t$
- $y_{et}^i \in \{0, 1\}$ — setup for item $i$ at echelon $e$ in period $t$

**coupling constraints:**

$$\sum_i y_{et}^i \leq 1 \quad \forall e, t$$

**Subproblems**

- $e = 1$
- $e = 2$
- $e = 3$

**DP based extended formulation as a flow in a hypergraph:**

- $w_{e,t,a,l,b}^i = 1$ if at echelon $e$ in period $t$ production covers demands for item $i$ from period $a$ to period $l$, while the rest of demand up to $b$, shall be covered in the future.
\[ \bar{S} = \{w^1, w^2\}, \quad \hat{w} \in \bar{W} \setminus \text{conv}(w^1, w^2) \]
Multi-echelon lot sizing: results with stabilization

Averages for 10 instances are given

<table>
<thead>
<tr>
<th>$E$</th>
<th>$K$</th>
<th>$T$</th>
<th>Colomn gen. for [M]</th>
<th>Column-and-row generation for [R]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>#it</td>
<td>cpu</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>50</td>
<td>126</td>
<td>1.7</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>50</td>
<td>79</td>
<td>1.8</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>100</td>
<td>332</td>
<td>38.0</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>100</td>
<td>232</td>
<td>31.5</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>50</td>
<td>187</td>
<td>11.8</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>50</td>
<td>112</td>
<td>12.0</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>100</td>
<td>509</td>
<td>454.5</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>100</td>
<td>362</td>
<td>520.4</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>50</td>
<td>296</td>
<td>62.6</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>50</td>
<td>223</td>
<td>66.8</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>100</td>
<td>882</td>
<td>4855.9</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>100</td>
<td>362</td>
<td>4657.8</td>
</tr>
</tbody>
</table>
Outline

1 Extented Formulations
   - Definitions
   - Interests
   - Coping with its large size

2 Dynamic Row-and-Column Generation
   - Methodology
   - Practical issues

3 Large-scale application
   - Freight transport by rail in Russia
Solve the **compact formulation**

**Step 2:** Obtain a solution $x^*$ of the pricing problem:

$$\min\{(c - \pi A)x : x \in P_I\}.$$
Solve the **compact formulation**

Step 2: Obtain a solution $x^*$ of the **pricing problem**:

$$\min\{(c - \pi A)x : x \in P_I\}.$$
Solve the **compact formulation**

**Step 2:** Obtain a solution \( x^* \) of the *pricing problem*:

\[
\min \{ (c - \overline{\pi} A) x : x \in P_I \}.
\]

**Lifting set:**

\[
T^{-1}(x) := \{ w \in \mathbb{N}^e : T w = x, H w \geq h \}
\]

**Solving a “preprocessed” feasibility MIP**
Solve the **compact formulation**

**Step 2**: Obtain a solution $x^*$ of the pricing problem:

$$
\min\{(c - \pi A) x : x \in P_I\}.
$$

**Lifting set:**

$$
T^{-1}(x) := \{w \in \mathbb{N}^e : Tw = x; \; Hw \geq h\}
$$

**Solving a “preprocessed” feasibility MIP**

**Example of the Knapsack Problem:**

![Knapsack Problem Diagram](diagram.png)
Coping with the size of the Subproblem

1. Solve the **compact formulation**

   **Step 2:** Obtain a solution $x^*$ of the pricing problem:

   $$\min \{(c - \pi A)x : x \in P_I\}.$$ 

   **Lifting set:**

   $$T^{-1}(x) := \{w \in \mathbb{N}^e : T w = x; H w \geq h\}$$

   **Solving a “preprocessed” feasibility MIP**

   **Example of the Knapsack Problem:**

   ![Diagram of the Knapsack Problem](image-url)
Solve the **compact formulation** (while no master constr. on $w$)

**Step 2:** Obtain a solution $x^*$ of the **pricing problem**:

$$\min \{ (c - \pi A) x : x \in P_I \}.$$

**Lifting set:**

$$T^{-1}(x) := \{ w \in \mathbb{N}^e : T w = x; H w \geq h \}$$

**Solving a “preprocessed” feasibility MIP**

**Lifting operator:**

$$x^* \rightarrow w^* \in T^{-1}(x^*)$$

**Breaking symmetries**
Solve the **compact formulation** (while no master constr. on $w$)

$$\min \{ (c - \pi A) x : x \in X \}.$$ 

$x^* \rightarrow w^* \in T^{-1}(x^*)$

Use a **forward labelling** Dynamic Program

Handling the underlying graph implicitly
1. Solve the **compact formulation** (while no master constr. on \( w \))

\[
\min \{(c - \pi A)x : x \in X\}.
\]

\[x^* \rightarrow w^* \in T^{-1}(x^*)\]

2. Use a **forward labelling** Dynamic Program

Handling the underlying graph implicitly

3. Use **successive approximations**: restrictions or relaxations
Coping with the Subproblem: Related work

[F. Fischer, C. Helmberg, MP2012
Dynamic Graph Generation for Shortest Path in Time Expanded Networks]
Coping with the Subproblem: Related work

[F. Fischer, C. Helmberg, MP2012
Dynamic Graph Generation for Shortest Path in Time Expanded Networks]
Coping with the Subproblem: Related work

[F. Fischer, C. Helmberg, MP2012
Dynamic Graph Generation for Shortest Path in Time Expanded Networks]

Assumption
Capacity as only linking constraints ⇒ reduced cost = \( \bar{c}_a \geq c_a \forall a \in A \).

Proposition
Given a restricted graph \( \bar{G} \subset G \) and its augmentation \( G^+ \):
\[
G^+ = \bar{G} \cup \delta(\bar{G}) \cup \text{SP}(\delta(\bar{G}))
\]
Let \( \hat{c}_a = \bar{c}_a \) for \( a \in \bar{G} \), and \( c_a \) otherwise.
Let \( P^* = \text{argmin}\{\hat{c}(P_{st}) : P_{st} \in G^+\} \).

If \( P^* \in \bar{G} \), then \( P^* = \text{argmin}\{\bar{c}(P_{st}) : P_{st} \in G\} \).
Otherwise, \( \bar{G} \leftarrow \bar{G} \cup P^* \).
Practical issues

1. Coping with the size of the Subproblem
Coping with the size of the Subproblem

Coping with the size of the Master

→ Preprocessing
→ Master cleanup
→ Disaggregate only if it yields recombinations
Practical issues

1. Coping with the size of the Subproblem
2. Coping with the size of the Master
   → Preprocessing
   → Master cleanup
   → Disaggregate only if it yields recombinations

Acceleration of column generation convergence

Stabilization techniques
   → Penalty functions & Smoothing
   → Disaggregations/Recombinations (add waiting arcs)

Strategies for column generation
   → Build a global solution to the master at each iteration
   → Stage-by-stage approach: decreasing restriction/relaxation level

Combination with cut generation
   → Lifting added variables
Practical issues

1. Coping with the size of the Subproblem
2. Coping with the size of the Master
   - Preprocessing
   - Master cleanup
   - Disaggregate only if it yields recombinations
3. Acceleration of column generation convergence
Practical issues

1. Coping with the size of the Subproblem
2. Coping with the size of the Master
   → Preprocessing
   → Master cleanup
   → Disaggregate only if it yields recombinations
3. Acceleration of column generation convergence
   - Stabilization techniques

- Dual oscillations
- Tailing-off effect
- Primal degeneracy
Practical issues

1. Coping with the size of the Subproblem
2. Coping with the size of the Master
   - Preprocessing
   - Master cleanup
   - Disaggregate only if it yields recombinations
3. Acceleration of column generation convergence
   - Stabilization techniques
     - Penalty functions & Smoothing
Practical issues

1. Coping with the **size of the Subproblem**
2. Coping with the **size of the Master**
   - Preprocessing
   - Master cleanup
   - Disaggregate only if it yields recombinations
3. **Acceleration** of column generation convergence
   - Stabilization techniques
     - Penalty functions & Smoothing
     - Disaggregations/Recombinations
Practical issues

1. Coping with the size of the Subproblem
2. Coping with the size of the Master
   → Preprocessing
   → Master cleanup
   → Disaggregate only if it yields recombinations
3. Acceleration of column generation convergence
   - Stabilization techniques
     → Penalty functions & Smoothing
     → Disaggregations/Recombinations (add waiting arcs)
Practical issues

1. Coping with the size of the Subproblem
2. Coping with the size of the Master
   - Preprocessing
   - Master cleanup
   - Disaggregate only if it yields recombinations
3. Acceleration of column generation convergence
   - Stabilization techniques
     - Penalty functions & Smoothing
     - Disaggregations/Recombinations (add waiting arcs)
   - Strategies for column generation
     - Build a global solution to the master at each iteration
Practical issues

1. Coping with the size of the Subproblem
2. Coping with the size of the Master
   - Preprocessing
   - Master cleanup
   - Disaggregate only if it yields recombinations
3. Acceleration of column generation convergence
   - Stabilization techniques
     - Penalty functions & Smoothing
     - Disaggregations/Recombinations (add waiting arcs)
   - Strategies for column generation
     - Build a global solution to the master at each iteration
     - Stage-by-stage approach: decreasing restriction/relaxation level
Practical issues

1. Coping with the size of the Subproblem
2. Coping with the size of the Master
   - Preprocessing
   - Master cleanup
   - Disaggregate only if it yields recombinations
3. Acceleration of column generation convergence
   - Stabilization techniques
     - Penalty functions & Smoothing
     - Disaggregations/Recombinations (add waiting arcs)
   - Strategies for column generation
     - Build a global solution to the master at each iteration
     - Stage-by-stage approach: decreasing restriction/relaxation level
4. Combination with cut generation
   - Lifting added variables
Outline

1. Extented Formulations
   - Definitions
   - Interests
   - Coping with its large size

2. Dynamic Row-and-Column Generation
   - Methodology
   - Practical issues

3. Large-scale application
   - Freight transport by rail in Russia
The freight car routing problem

[R. Sadykov et al, 2013]
Each **type of railcar** defines a commodity $c \in C$
Multi-commodity flow formulation

**Variables**
- \( x_a \in \mathbb{N} \) — nb of cars using arc \( a \in A_c, \ c \in C \)
- \( y_d \in \{0, 1\} \) — demand \( d \) is accepted or not

\[
\begin{align*}
\max & \quad \sum_{c \in C} \sum_{a \in A_c} p_a x_a \\
\sum_{c \in C} \sum_{a \in A_{cd}} x_a & \geq n_{d_{\min}} y_d \quad \forall d \\
\sum_{c \in C} \sum_{a \in A_{cd}} x_a & \leq n_{d_{\max}} y_d \quad \forall d \\
\sum_{a \in \delta^- (v)} x_a - & \sum_{a \in \delta^+ (v)} x_a = b_v \quad \forall c \in C, v \in V_c \\
x_a & \in \mathbb{N} \quad \forall c \in C, a \in V_c \\
y_d & \in \{0, 1\} \quad \forall d
\end{align*}
\]
LP-Solution approaches

- **Direct**: solving a multi-commodity flow problem using *Clp* (specifically modified)

- **Standard Column Generation**: a column is
  - Option A: A full planning for a type of car (decomposition per commodity)
  - Option B: A in-tree into a sink (decomposition per sink)
  - Option C: A path for origin to destination (decomposition per pair o-d)

- **Column Generation for Extended Formulation**: using option A.
Real-life instances

1'025 stations, up to 6’800 demands, 11 car types, 12’651 cars, and 8’232 sources → 300 thousands nodes and 10 millions arcs.

<table>
<thead>
<tr>
<th>Horizon</th>
<th>Direct</th>
<th>ColGenEF</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>5m24s</td>
<td>1m52s</td>
</tr>
<tr>
<td>90</td>
<td>7m05s</td>
<td>1m47s</td>
</tr>
<tr>
<td>100</td>
<td>9m42s</td>
<td>2m19s</td>
</tr>
<tr>
<td>110</td>
<td>13m38s</td>
<td>3m11s</td>
</tr>
<tr>
<td>120</td>
<td>17m19s</td>
<td>3m57s</td>
</tr>
<tr>
<td>130</td>
<td>25m52s</td>
<td>5m03s</td>
</tr>
<tr>
<td>140</td>
<td>35m08s</td>
<td>5m25s</td>
</tr>
<tr>
<td>150</td>
<td>44m58s</td>
<td>7m02s</td>
</tr>
<tr>
<td>160</td>
<td>57m11s</td>
<td>8m19s</td>
</tr>
<tr>
<td>170</td>
<td>1h13m58s</td>
<td>10m53s</td>
</tr>
<tr>
<td>180</td>
<td>1h26m46s</td>
<td>12m16s</td>
</tr>
</tbody>
</table>

≤ 15 iterations, about 3% of the arc variables have been generated
An approach based on an extended formulation

- An **EASY WAY** to bring-in combinatorial structure.
- Its size can be coped with by **combining** ideas of
  - Restriction / Relaxation
  - Benders projection
  - Dantzig-Wolfe dynamic generation.
- With **dynamic row-and-column generation**, a small % of variables and constraints are needed; hence it **scales up** to real-life applications.
- Is well suited for **efficiency enhancement** features: **cuts** on lifted variables, **Dynamic Progr. state-space-relax.**, **red.-cost-fixing**.
Take away messages

An approach based on an extended formulation

- An **EASY WAY** to bring-in combinatorial structure.
- Its size can be coped with by combining ideas of
  - Restriction / Relaxation
  - Benders projection
  - Dantzig-Wolfe dynamic generation.

- With *dynamic row-and-column generation*, a small % of variables and constraints are needed; hence it scales up to real-life applications.

- Is well suited for efficiency enhancement features: cuts on lifted variables, Dynamic Progr. state-space-relax., red.-cost-fixing.

Perspectives
Take away messages

An approach based on an extended formulation

- An EASY WAY to bring-in combinatorial structure.
- Its size can be coped with by combining generic ideas of
  - Restriction / Relaxation [Soumis et al]
  - Benders projection [Van Vyve & Wolsey, EJCO, 2013]
  - Dantzig-Wolfe dynamic generation.

- With dynamic row-and-column generation, a small % of variables and constraints are needed; hence it scales up to real-life applications.
- Is well suited for efficiency enhancement features: cuts on lifted variables, Dynamic Progr. state-space-relax., red.-cost-fixing.

Perspectives
An approach based on an extended formulation

- An EASY WAY to bring-in combinatorial structure.
- Its size can be coped with by combining ideas of
  - Restriction / Relaxation
  - Benders projection
  - Dantzig-Wolfe dynamic generation.

- With dynamic row-and-column generation [M. Goycoolea et al 2013], a small % of variables and constraints are needed; hence it scales up to real-life applications.

- Is well suited for efficiency enhancement features: cuts on lifted variables, Dynamic Progr. state-space-relax., red.-cost-fixing.

Perspectives
An approach based on an extended formulation

- An **EASY WAY** to bring-in combinatorial structure.
- Its size can be coped with by **combining** ideas of
  - Restriction / Relaxation
  - Benders projection
  - Dantzig-Wolfe dynamic generation.
- With **dynamic row-and-column generation**, a small % of variables and constraints are needed; hence it **scales up** to real-life applications.
- Is well suited for **efficiency enhancement** features: cuts on lifted variables, Dynamic Progr. state-space-relax., red.-cost-fixing.

Perspectives
Take away messages

An approach based on an extended formulation

- An **EASY WAY** to bring-in combinatorial structure.
- Its size can be coped with by **combining** ideas of
  - Restriction / Relaxation
  - Benders projection
  - Dantzig-Wolfe dynamic generation.

- With **dynamic row-and-column generation**, a small % of variables and constraints are needed; hence it scales up to real-life applications.

- Is well suited for **efficiency enhancement** features: cuts on lifted variables, Dynamic Progr. state-space-relax., red.-cost-fixing.

Perspectives