CP Tutorial
Lagrangian relaxation for domain filtering

Hadrien Cambazard

G-SCOP - Université de Grenoble

Goals of the tutorial

Goals:

- Share some understanding of the basic ideas underlying LR
- Explain/review the use of LR by the CP community

Goals of the tutorial

Goals:

- Share some understanding of the basic ideas underlying LR
- Explain/review the use of LR by the CP community

- Not an expert in Lagrangian relaxation (LR)
- Experience in implementing LR in a constraint solver (mainly in applications) for
  - Bounding and filtering:
    - beyond polynomial subproblems
    - directed by the objective function
- Pragmatic point of view of a user of OR techniques

Outline

1. Principles of Lagrangian relaxation (LR)
   - Lagrangian subproblem
   - Lagrangian dual
2. Solving the Lagrangian dual
   - Kelley's algorithm, Sub-gradients, Golden section
   - Filtering based on the subproblem
3. Domain filtering for NValue using LR
   - LR to compute a lower bound
   - Linear Programming (LP) and LR filtering
4. Use of LR by the CP community
   - LR for domain's filtering
   - LR for filtering global constraints
   - Lagrangian decomposition for domain's filtering
   - Applications

Outline

1. Principles of Lagrangian relaxation (LR)
   - Lagrangian subproblem
   - Lagrangian dual
2. Solving the Lagrangian dual
   - Kelley's algorithm, Sub-gradients, Golden section
   - Filtering based on the subproblem
3. Domain filtering for NValue using LR
   - LR to compute a lower bound
   - Linear Programming (LP) and LR filtering
4. Use of LR by the CP community
   - LR for domain's filtering
   - LR for filtering global constraints
   - Lagrangian decomposition for domain's filtering
   - Applications

Illustrative of:

- **Relaxing a single constraint**
- **Filtering of multi-cost regular**

- **Comparison with LP reduced cost filtering**
- **Filtering of Nvalue**

- **Two CP papers using LR this year**

1- Principles of LR

Some of the key historical papers about Lagrangian relaxation:

[Held, M., Karp, R.M. **The traveling salesman problem and minimum spanning trees,** Operations Research, 1970]

1- Principles of LR

Some of the key historical papers about Lagrangian relaxation:

[Held, M., Karp, R.M. **The traveling salesman problem and minimum spanning trees,** Operations Research, 1970]

[Geoffrion, A.M., **Lagrangean relaxation for integer programming**, Mathematical Programming Study 2, 1974]

[Fisher, M.L. **The Lagrangian Relaxation Method for solving integer programming problems**, Management Science, 1981]

1- Principles of LR

Some of the key historical papers about Lagrangian relaxation:

[Held, M., Karp, R.M. **The traveling salesman problem and minimum spanning trees,** Operations Research, 1970]

[Geoffrion, A.M., **Lagrangean relaxation for integer programming**, Mathematical Programming Study 2, 1974]

[Fisher, M.L. **The Lagrangian Relaxation Method for solving integer programming problems**, Management Science, 1981]

[Fisher, M.L., **An applications oriented guide to Lagrangian Relaxation**, Interfaces, 1985.]

"How to do" paper

1- Principles of LR

Some of the key historical papers about Lagrangian relaxation:

[Held, M., Karp, R.M. **The traveling salesman problem and minimum spanning trees,** Operations Research, 1970]

[Geoffrion, A.M., **Lagrangean relaxation for integer programming**, Mathematical Programming Study 2, 1974]

[Fisher, M.L. **The Lagrangian Relaxation Method for solving integer programming problems**, Management Science, 1981]

[Fisher, M.L., **An applications oriented guide to Lagrangian Relaxation**, Interfaces, 1985.]

[Lemaréchal, C.: **Lagrangian relaxation**. In: Jnger, M., Naddef, D. (eds.). Computational Combinatorial Optimization, 2001]

1- Principles of LR

Some of the key historical papers about Lagrangian relaxation:

[Held, M., Karp, R.M. **The traveling salesman problem and minimum spanning trees,** Operations Research, 1970]

[Geoffrion, A.M., **Lagrangean relaxation for integer programming**, Mathematical Programming Study 2, 1974]

[Fisher, M.L. **The Lagrangian Relaxation Method for solving integer programming problems**, Management Science, 1981]

[Fisher, M.L., **An applications oriented guide to Lagrangian Relaxation**, Interfaces, 1985.]

[Lemaréchal, C.: **Lagrangian relaxation**. In: Jnger, M., Naddef, D. (eds.). Computational Combinatorial Optimization, 2001]

Material for this talk from text books:

Network flows of Ahuja, Magnanti, Orlin

Integer programming of Wolsey,

1- Principles of Lagrangian relaxation

Lagrangian subproblem

Lagrangian Dual

1- Principles of LR

Shortest path with resource constraints

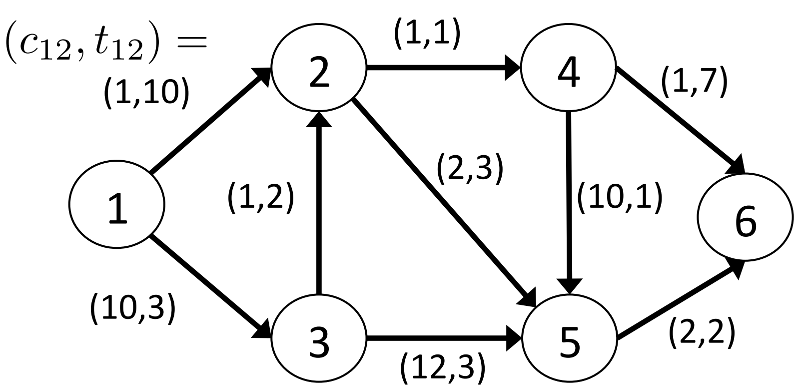
$$\text{Min z} = \sum c_{ij} x_{ij}$$

$$\text{path conservation} \quad (1)$$

$$\sum t_{ij} x_{ij} \leq T \quad (2)$$

$$x_{ij} \in \{0,1\}$$

P



Simplified example taken from *Network flows* of Ahuja, Magnanti, Orlin

1- Principles of LR

Shortest path with resource constraints

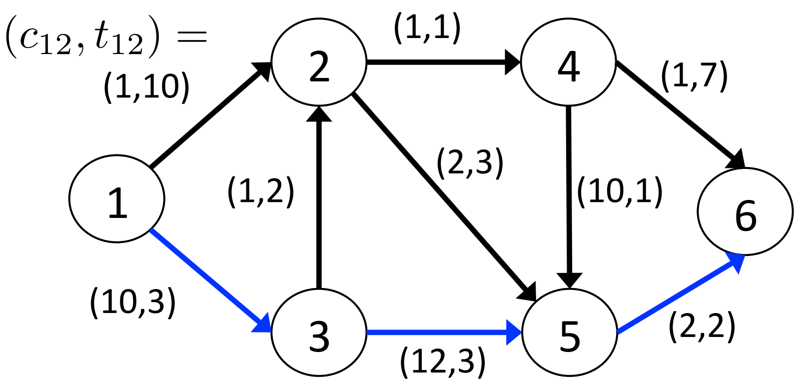
$$\text{Min z} = \sum c_{ij} x_{ij}$$

$$\text{path conservation} \quad (1)$$

$$\sum t_{ij} x_{ij} \leq T \quad (2)$$

$$x_{ij} \in \{0,1\}$$

P



T = 10

A solution:

$$x_{13} = 1, x_{35} = 1, x_{56} = 1$$
$$z = 10 + 12 + 2 = 24$$
$$time = 3 + 3 + 2 \leq 10$$

1- Principles of LR

Shortest path with resource constraints

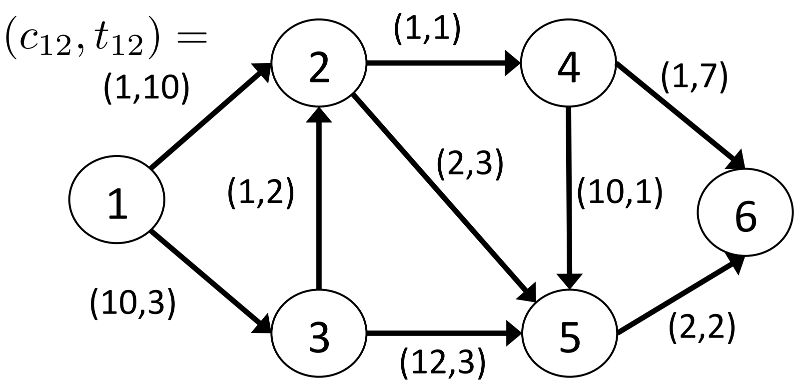
$$\text{Min z} = \sum c_{ij} x_{ij}$$

$$\text{path conservation} \quad (1)$$

$$\sum t_{ij} x_{ij} \leq T \quad (2) \quad (\lambda)$$

$$x_{ij} \in \{0, 1\}$$

P



1- Principles of LR

Shortest path with resource constraints

$$\text{Min } z = \sum c_{ij} x_{ij}$$

path conservation (1)

$$\sum t_{ij} x_{ij} \leq T \qquad (2) \; (\lambda)$$

$$x_{ij} \in \{0, 1\}$$

P

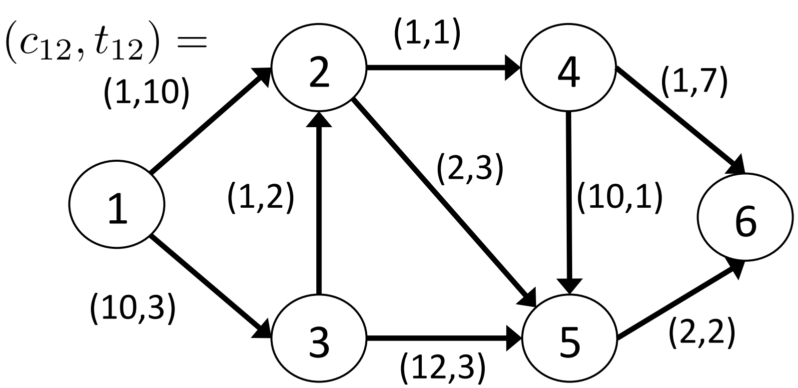
For all $\lambda \geq 0$: **Shortest path**

$$\text{Min } w(\lambda) = \sum c_{ij} x_{ij} - \lambda (T - \sum t_{ij} x_{ij})$$

path conservation (1)

$$x_{ij} \in \{0, 1\}$$

$L(\lambda)$



1- Principles of LR

Shortest path with resource constraints

$$\text{Min z} = \sum c_{ij}x_{ij}$$

path conservation (1)

$$\sum t_{ij}x_{ij} \leq T \qquad (2) \quad (\lambda)$$

$$x_{ij} \in \{0,1\}$$

P

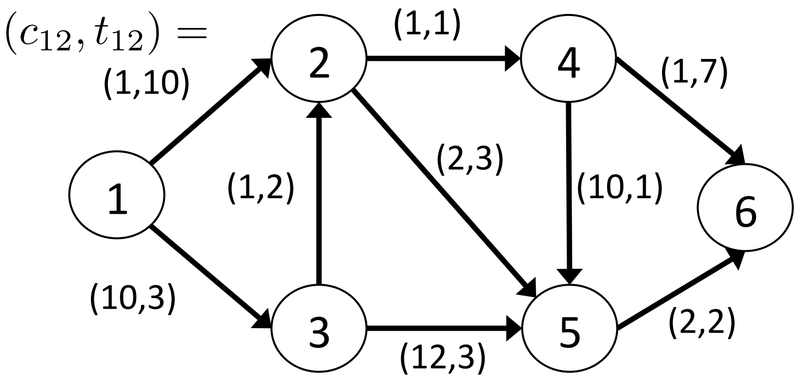
For all $\lambda \geq 0$: **Shortest path**

$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij})$$
$$= \sum (c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

path conservation (1)

$$x_{ij} \in \{0,1\}$$

$L(\lambda)$



1- Principles of LR

Shortest path with resource constraints

$$\text{Min z} = \sum c_{ij} x_{ij}$$

path conservation (1)

$$\sum t_{ij} x_{ij} \leq T \quad (2)$$

$$x_{ij} \in \{0,1\}$$

P

For all $\lambda \geq 0$: **Shortest path**

$$\text{Min } w(\lambda) = \sum c_{ij} x_{ij} - \underbrace{\lambda}_{\geq 0}(\underbrace{T - \sum t_{ij} x_{ij}}_{\geq 0})$$

path conservation

$$x_{ij} \in \{0,1\}$$

$L(\lambda)$

For any $\lambda \geq 0$:

Any feasible solution $\overline{x}$ of P is also feasible for $L(\lambda)$ and $\overline{z} \geq \overline{w}(\lambda)$

1- Principles of LR

Shortest path with resource constraints

$$\text{Min z} = \sum c_{ij}x_{ij}$$

path conservation (1)

$$\sum t_{ij}x_{ij} \leq T \qquad (2)$$

$$x_{ij} \in \{0,1\}$$

P

For all $\lambda \geq 0$: **Shortest path**

$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij})$$

$\lambda \geq 0$, $(T - \sum t_{ij}x_{ij}) \geq 0$

path conservation

$$x_{ij} \in \{0,1\}$$

$L(\lambda)$

For any $\lambda \geq 0$:

Any feasible solution $\overline{x}$ of P is also feasible for $L(\lambda)$ and $\overline{z} \geq \overline{w}(\lambda)$

So we have : $z^* \geq w^*(\lambda)$

1- Principles of LR

Shortest path with resource constraints

$$\text{Min z} = \sum c_{ij} x_{ij}$$

path conservation (1)

$$\sum t_{ij} x_{ij} \leq T \quad (2)$$

$$x_{ij} \in \{0,1\}$$

P

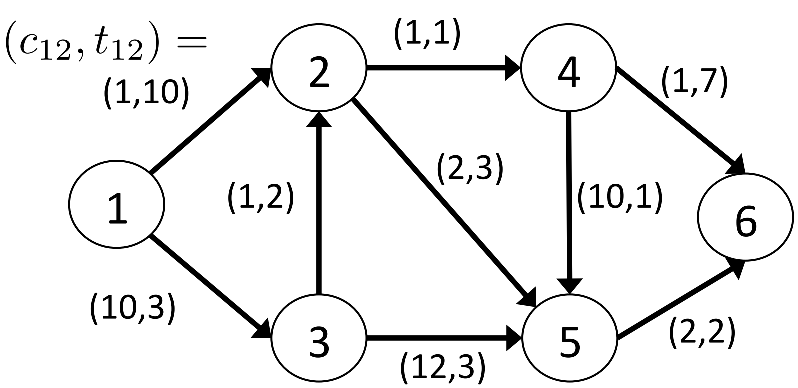
For all $\lambda \geq 0$: **Shortest path**

$$\text{Min } w(\lambda) = \sum c_{ij} x_{ij} - \lambda(T - \sum t_{ij} x_{ij})$$
$$= \sum (c_{ij} + \lambda t_{ij}) x_{ij} - \lambda T$$

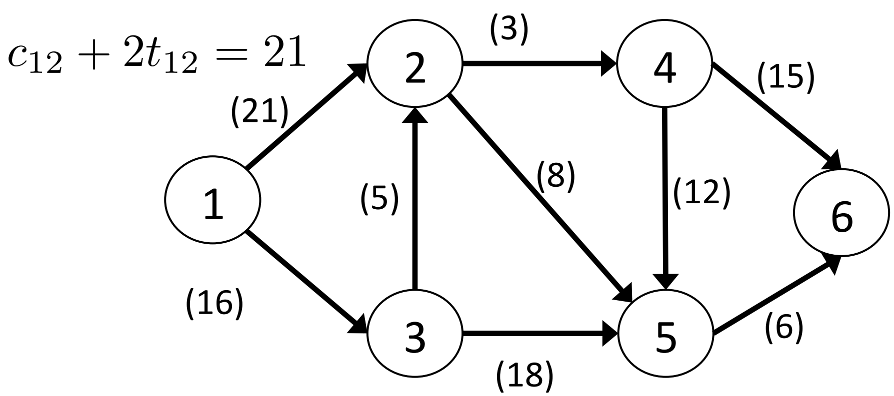
path conservation (1)

$$x_{ij} \in \{0,1\}$$

$L(\lambda)$



Lagrangian sub-problem for $\lambda = 2$



1- Principles of LR

$$\text{Min } z = \sum c_{ij}x_{ij}$$

path conservation (1)

$$\sum t_{ij}x_{ij} \leq T \qquad (2)$$

$$x_{ij} \in \{0,1\}$$

P

T = 10

For all $\lambda \geq 0$:

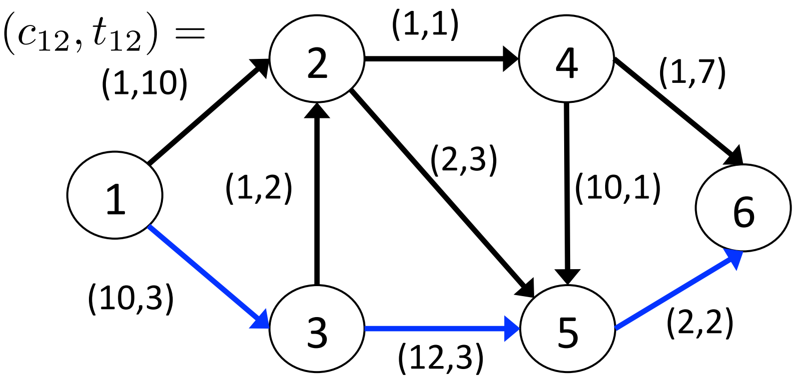
$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij})$$
$$= \sum(c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

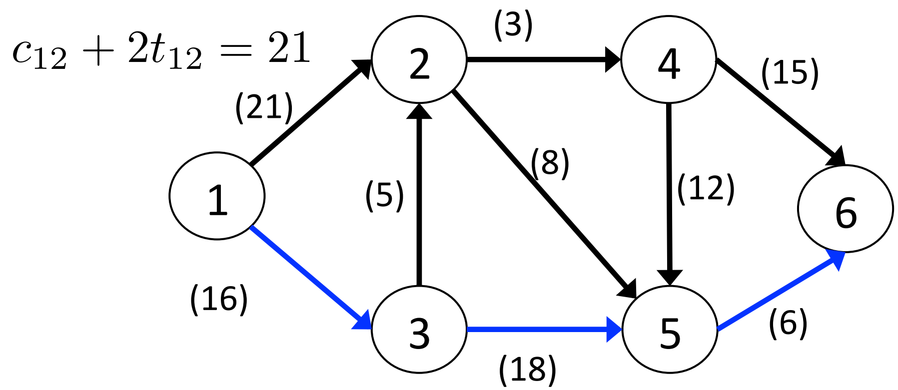
path conservation (1)

$$x_{ij} \in \{0,1\}$$

$L(\lambda)$

$\lambda = 2$





$$\overline{z} = 10 + 12 + 2 = 24$$

$$\overline{w} = 16 + 18 + 6 - 20 = 20$$

For any $\lambda \geq 0$:

Any feasible solution $\overline{x}$ of P is also feasible for $L(\lambda)$ and $\overline{z} \geq \overline{w}(\lambda)$

So we have : $z^* \geq w^*(\lambda)$

1- Principles of LR

$$\text{Min } z = \sum c_{ij}x_{ij}$$

path conservation (1)

$$\sum t_{ij}x_{ij} \leq T \qquad (2)$$

$$x_{ij} \in \{0,1\}$$

P

T = 10

For all $\lambda \geq 0$:

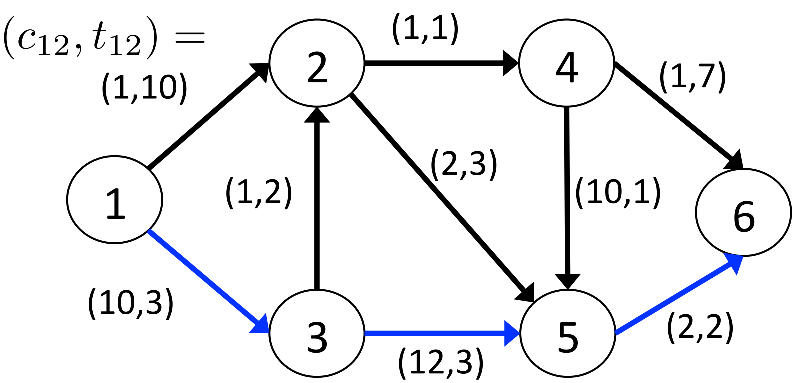
$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij}) = \sum(c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

path conservation (1)

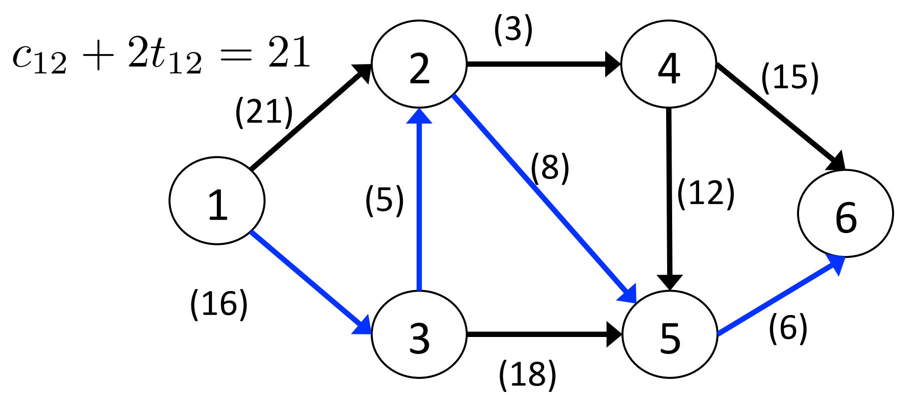
$$x_{ij} \in \{0,1\}$$

$L(\lambda)$

$\lambda = 2$



$$\overline{z} = 10 + 12 + 2 = 24$$



$$w^*(2) = 35 - 20 = 15$$

For any $\lambda \geq 0$:

Any feasible solution $\overline{x}$ of P is also feasible for $L(\lambda)$ and $\overline{z} \geq \overline{w}(\lambda)$

So we have : $z^* \geq w^*(\lambda)$

1- Principles of LR

Min $z = \sum c_{ij}x_{ij}$

path conservation (1)

$\sum t_{ij}x_{ij} \leq T$ (2)

$x_{ij} \in \{0,1\}$

P

T = 10

For all $\lambda \geq 0$:

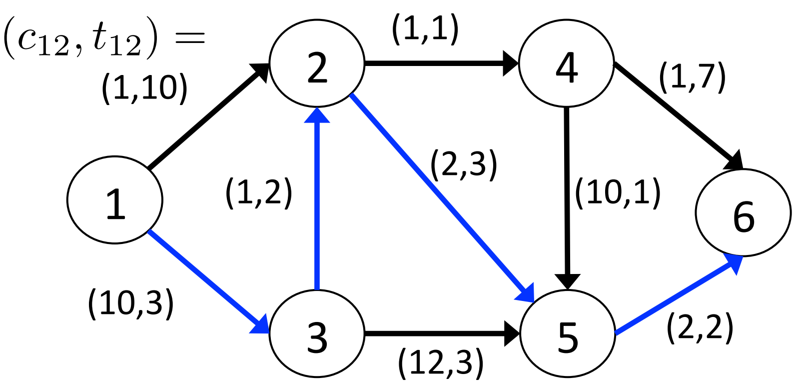
$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij})$$
$$= \sum (c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

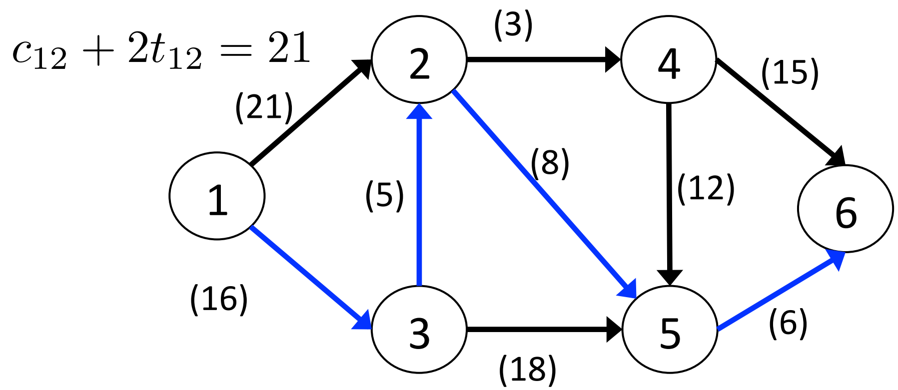
path conservation (1)

$x_{ij} \in \{0,1\}$

$L(\lambda)$

$\lambda = 2$





$\overline{z} = 15 = z^*$

$w^*(2) = 35 - 20 = 15$

For any $\lambda \geq 0$:

Any feasible solution $\overline{x}$ of P is also feasible for $L(\lambda)$ and $\overline{z} \geq \overline{w}(\lambda)$

So we have : $z^* \geq w^*(\lambda)$

1- Principles of LR

$$\begin{aligned} &\text{Min z} = \sum c_{ij}x_{ij} \\ &\text{path conservation (1)} \\ &\quad \sum t_{ij}x_{ij} \leq T \qquad (2) \\ &\quad x_{ij} \in \{0,1\} \end{aligned} \qquad P$$

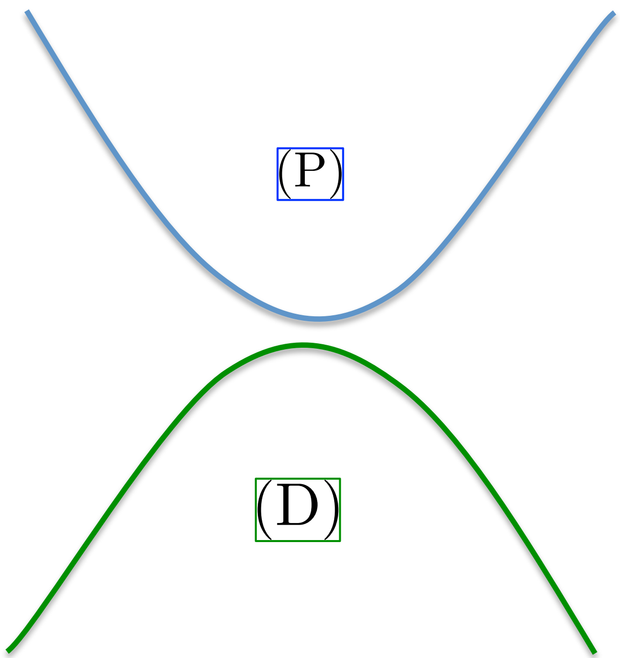
For all $\lambda \geq 0$:

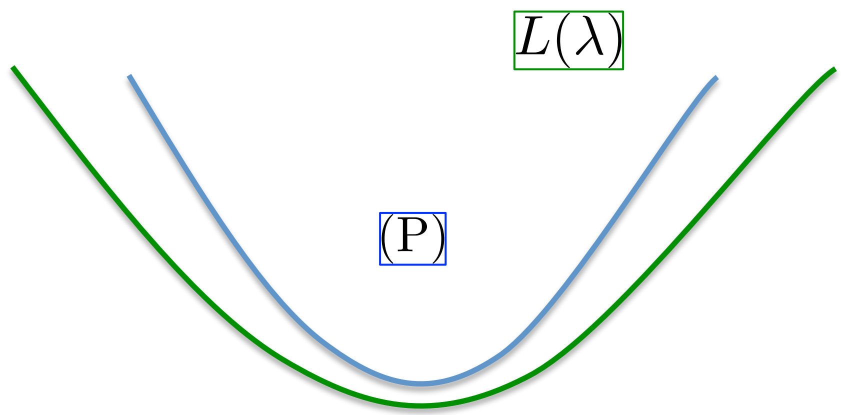
$$\begin{aligned} \text{Min } w(\lambda) &= \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij}) \\ &= \sum (c_{ij} + \lambda t_{ij})x_{ij} - \lambda T \\ &\text{path conservation (1)} \\ &x_{ij} \in \{0,1\} \end{aligned} \qquad L(\lambda)$$

For any $\lambda \geq 0$:

Any feasible solution $\overline{x}$ of P is also feasible for $L(\lambda)$ and $\overline{z} \geq \overline{w}(\lambda)$

So we have : $z^* \geq w^*(\lambda)$





1- Principles of LR

$$\begin{aligned} &\text{Min } z = \sum c_{ij} x_{ij} \\ &\text{path conservation } (1) \\ &\sum t_{ij} x_{ij} \leq T \quad (2) \\ &x_{ij} \in \{0,1\} \end{aligned} \qquad P$$

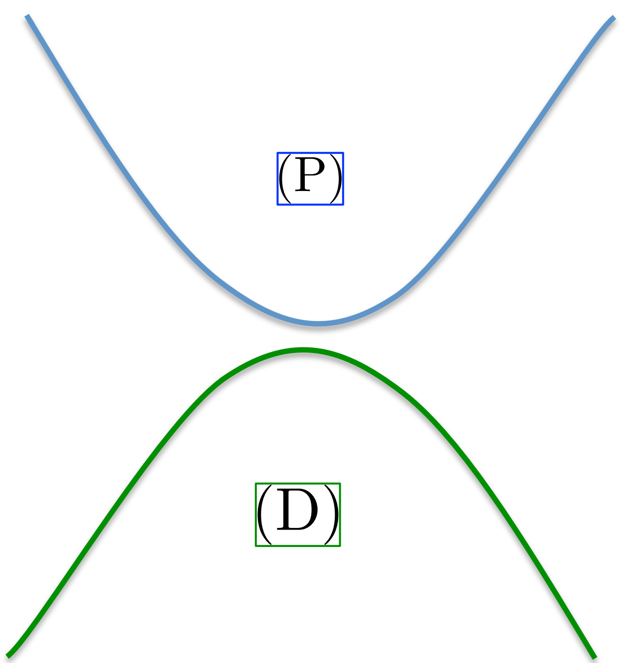
For all $\lambda \geq 0$:

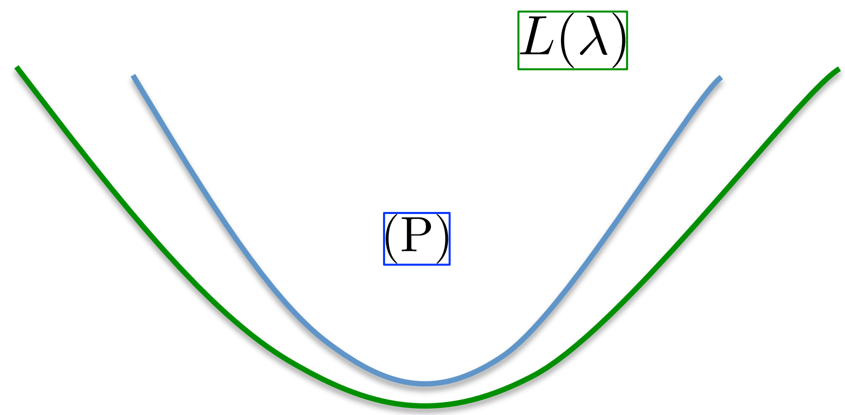
$$\begin{aligned} \text{Min } w(\lambda) &= \sum c_{ij} x_{ij} - \lambda(T - \sum t_{ij} x_{ij}) \\ &= \sum (c_{ij} + \lambda t_{ij}) x_{ij} - \lambda T \\ &\text{path conservation } (1) \\ &x_{ij} \in \{0,1\} \end{aligned} \qquad L(\lambda)$$

For any $\lambda \geq 0$:

Any feasible solution $\overline{x}$ of P is also feasible for $L(\lambda)$ and $\overline{z} \geq \overline{w}(\lambda)$

So we have : $z^* \geq w^*(\lambda)$





Lagrangian Dual:

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$

1- Principles of LR

For all $\lambda \geq 0$:

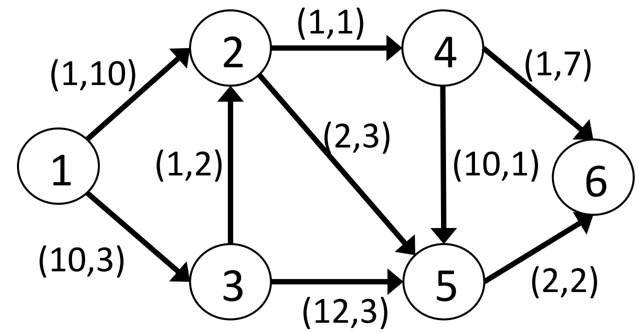
$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij})$$
$$= \sum (c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

path conservation (1)

$$x_{ij} \in \{0,1\}$$

$L(\lambda)$

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$



- *Note that:*
  - Changing λ does not affect the set of feasible solutions of $L(\lambda)$
  - So the cost of given solution of $L(\lambda)$ can be seen as a linear function of λ

1- Principles of LR

For all $\lambda \geq 0$:

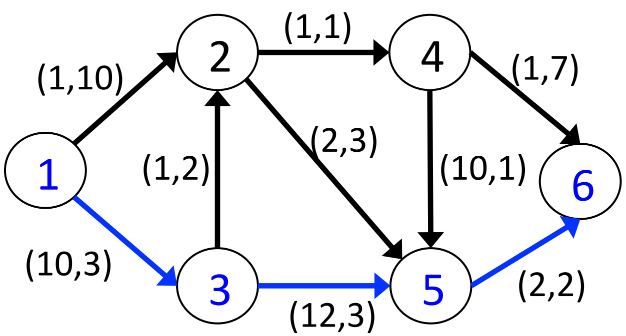
$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij}) = \sum (c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

path conservation (1)

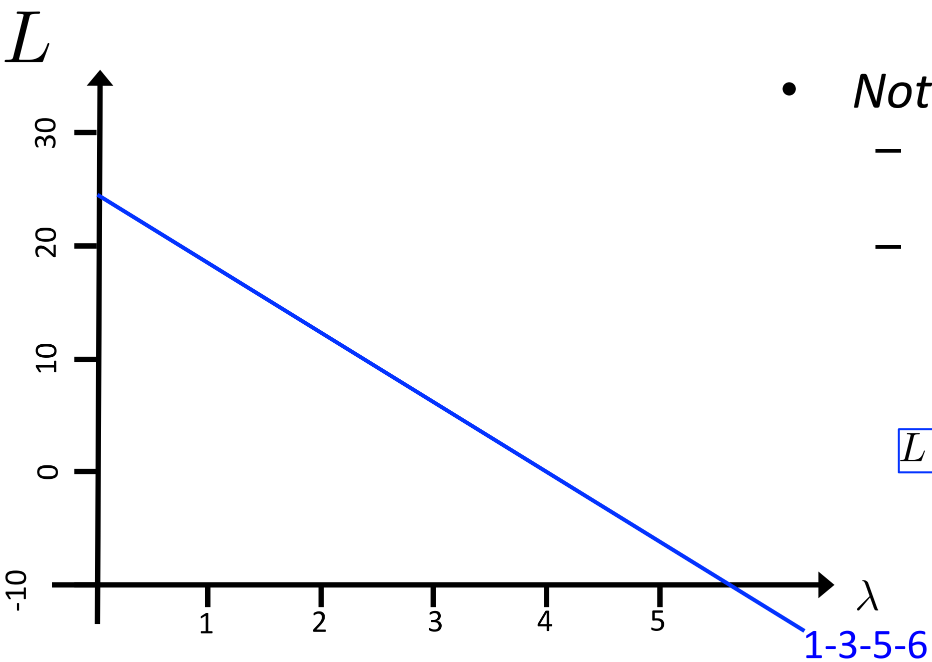
$x_{ij} \in \{0, 1\}$

$L(\lambda)$

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$



T = 14



- *Note:*
  - Changing λ does not affect the set of feasible solutions of $L(\lambda)$
  - So the cost of given solution of $L(\lambda)$ can be seen as a linear function of λ

$$L \leq (10 + 3\lambda) + (12 + 3\lambda) + (2 + 2\lambda) - 14\lambda$$
$$= 24 - 6\lambda \quad \text{(1-3-5-6)}$$

1- Principles of LR

For all $\lambda \geq 0$:

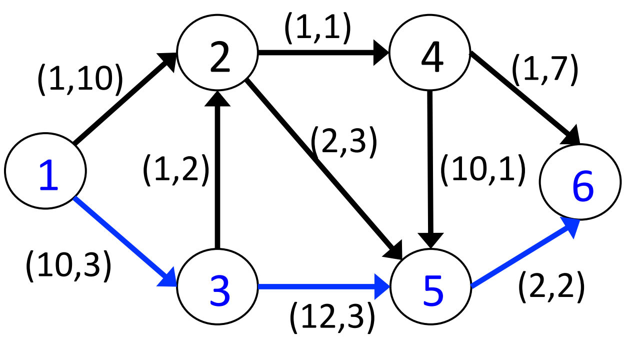
$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij}) = \sum (c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

path conservation (1)

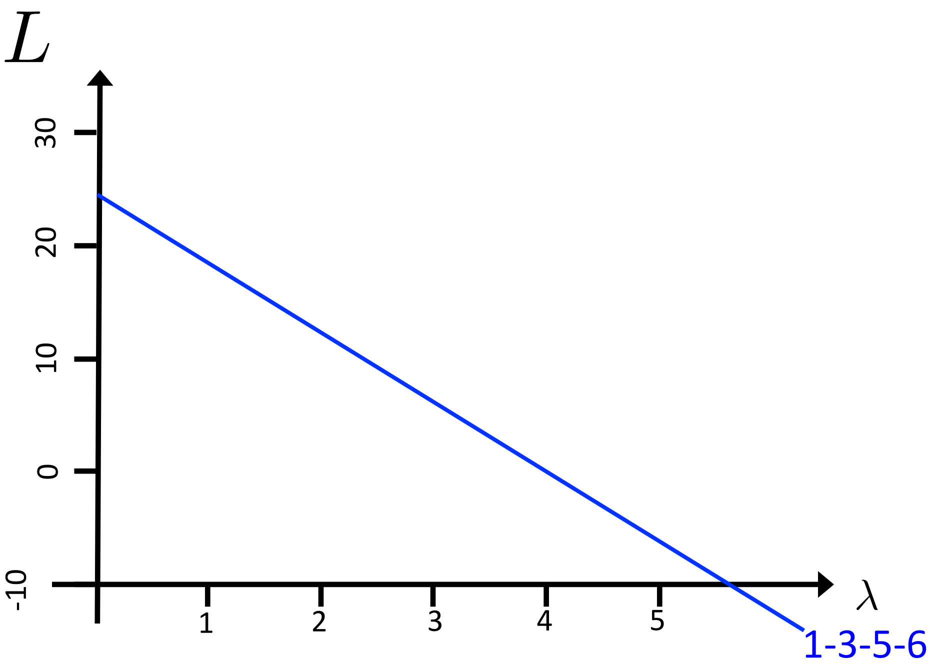
$x_{ij} \in \{0,1\}$

$L(\lambda)$

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$



T = 14



Max L

$L \leq (10 + 3\lambda) + (12 + 3\lambda) + (2 + 2\lambda) - 14\lambda$

$= 24 - 6\lambda$ (1-3-5-6)

1- Principles of LR

For all $\lambda \geq 0$:

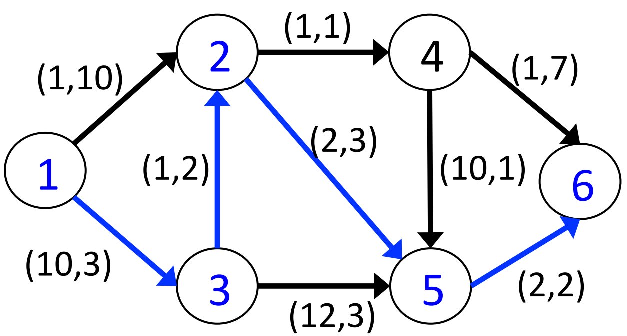
$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij}) = \sum(c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

path conservation (1)

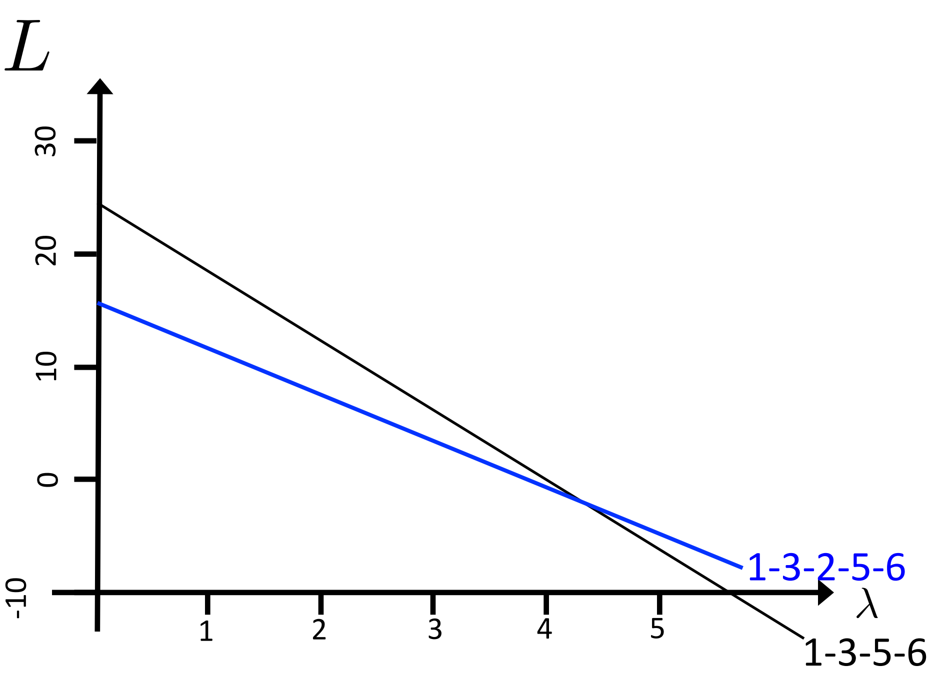
$x_{ij} \in \{0,1\}$

$L(\lambda)$

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$



T = 14



$$\text{Max } L$$

$$L \leq (10 + 3\lambda) + (12 + 3\lambda) + (2 + 2\lambda) - 14\lambda = 24 - 6\lambda$$ (1-3-5-6)

$L \leq 15 - 4\lambda$ (1-3-2-5-6)

1- Principles of LR

For all $\lambda \geq 0$:

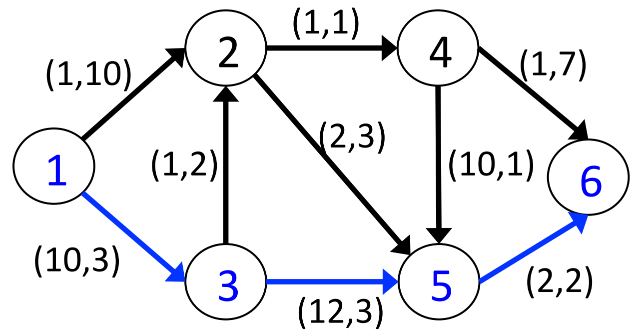
$$\begin{aligned}\text{Min } w(\lambda) &= \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij}) \\ &= \sum (c_{ij} + \lambda t_{ij})x_{ij} - \lambda T\end{aligned}$$

path conservation (1)

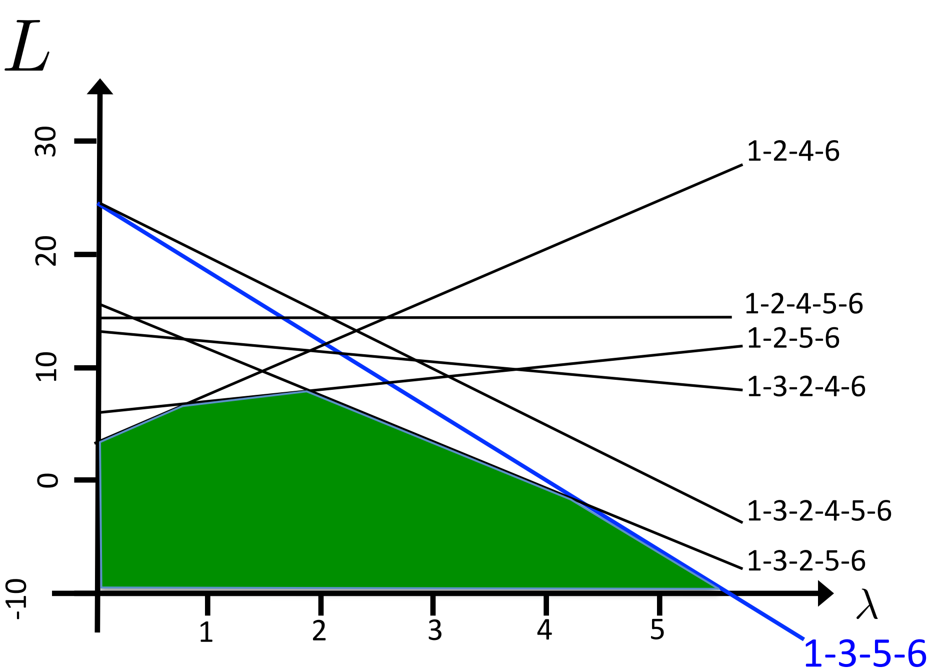
$x_{ij} \in \{0, 1\}$

$L(\lambda)$

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$



T = 14



Max L

$L \leq 3 + 4\lambda$ (1-2-4-6)

$L \leq 14$ (1-2-4-5-6)

$L \leq 5 + \lambda$ (1-2-5-6)

$L \leq 13 - \lambda$ (1-3-2-4-6)

$L \leq 24 - 5\lambda$ (1-3-2-4-5-6)

$L \leq 15 - 4\lambda$ (1-3-2-5-6)

$L \leq (10 + 3\lambda) + (12 + 3\lambda) + (2 + 2\lambda) - 14\lambda$

$= 24 - 6\lambda$ (1-3-5-6)

1- Principles of LR

For all $\lambda \geq 0$:

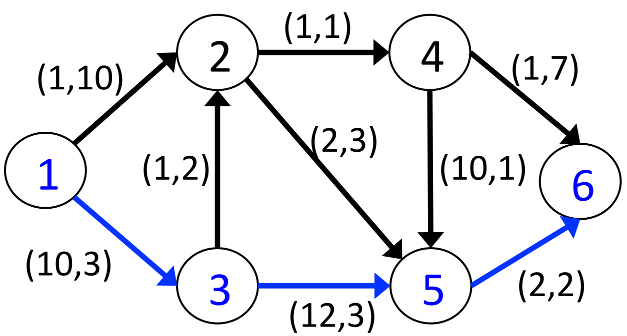
$$\text{Min } w(\lambda) = \sum c_{ij}x_{ij} - \lambda(T - \sum t_{ij}x_{ij}) = \sum (c_{ij} + \lambda t_{ij})x_{ij} - \lambda T$$

path conservation (1)

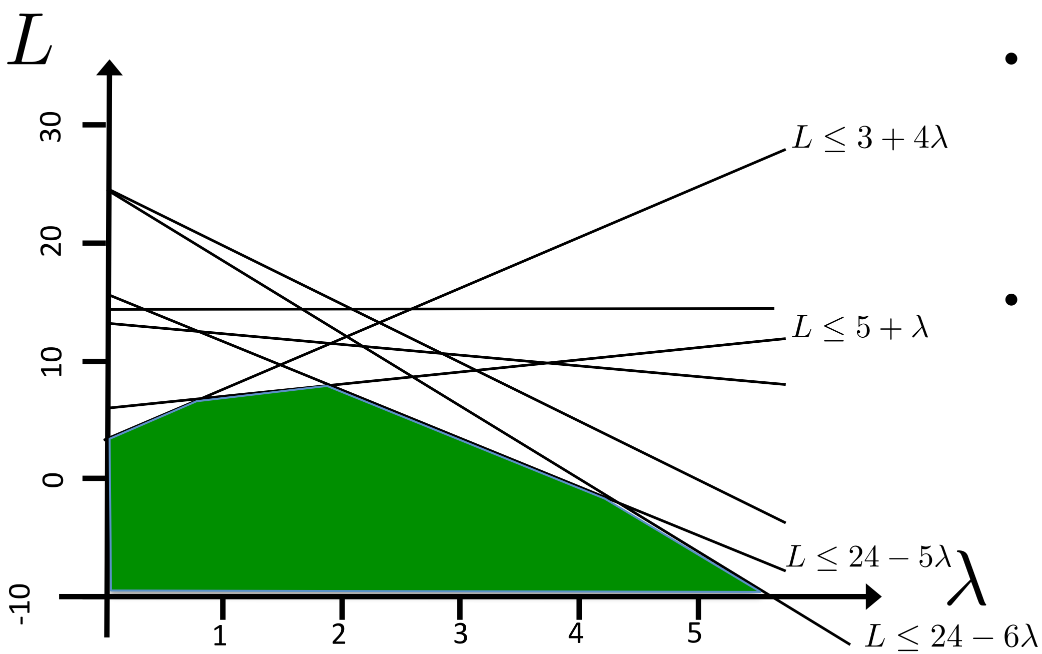
$x_{ij} \in \{0,1\}$

$L(\lambda)$

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$



T = 14



- The Lagrangian dual is a big LP
  (number of constraints = number of feasible solutions of the subproblem)

- Concave non-differentiable function
  - Golden search (one-dimension)
  - Kelley-Cheney-Goldstein algorithm
  - Sub-gradients
  - Bundle Methods (Claude Lemaréchal)

1- Principles of LR

- Well known theorem: $z^*_{LP} \leq L^* \leq z^*_{IP}$
- When the subproblem has the integrality property $Z^*_{LP} = L^*$

[Geoffrion, A. M. **Lagrangean relaxation for integer programming**. 1974]

[Fisher, M.L., **An applications oriented guide to Lagrangian Relaxation**, Interfaces (1985) 15, 10.]

1- Principles of LR

- Well known theorem: $z^*_{LP} \leq L^* \leq z^*_{IP}$
- When the subproblem has the integrality property $Z^*_{LP} = L^*$

[Geoffrion, A. M. **Lagrangean relaxation for integer programming**. 1974]

[Fisher, M.L., **An applications oriented guide to Lagrangian Relaxation**, Interfaces (1985) 15, 10.]

- So integrality property is often undesired in this context !
- Typical subproblems: *knapsack, shortest path with resource constraints*,…
- Nonetheless, approximating the LP solution by Lagrangian relaxation can be very fast and more efficient than the using an LP solver

In the CP community : [Slusky, M.R. , van Hoeve, W.J. **A Lagrangian relaxation for Golomb rulers**, CPAIOR 2013]

2- Solving the Lagrangian dual

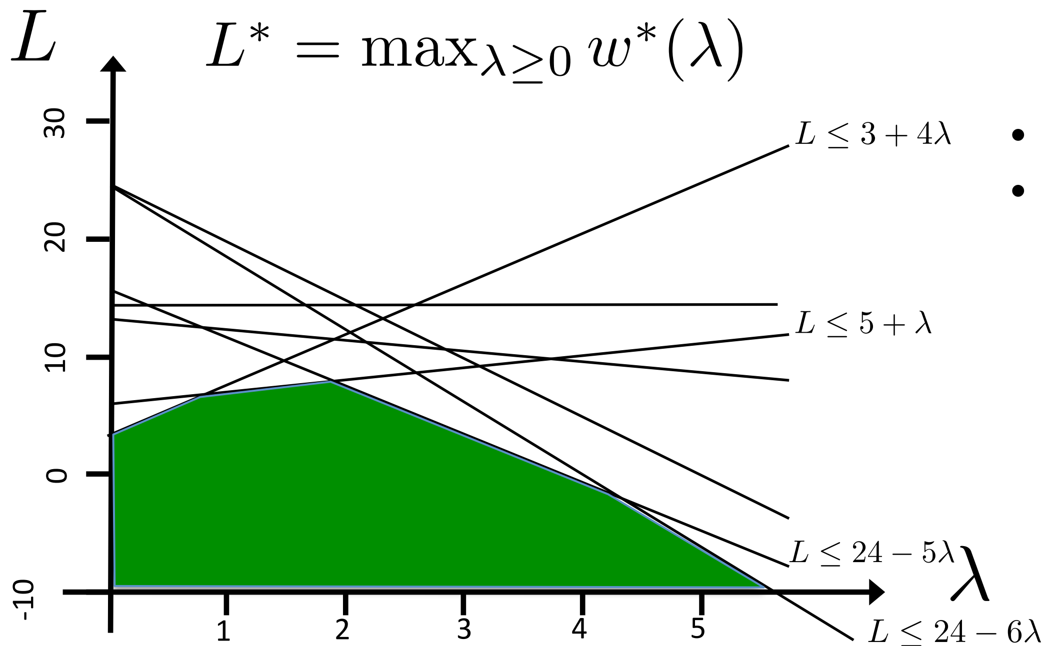
Solving

- Kelley, Cheney-Goldstein algorithm
- Subgradient techniques
- Golden section search for a single multiplier

Filtering from the Lagrangian subproblem

Illustration: multi-cost regular

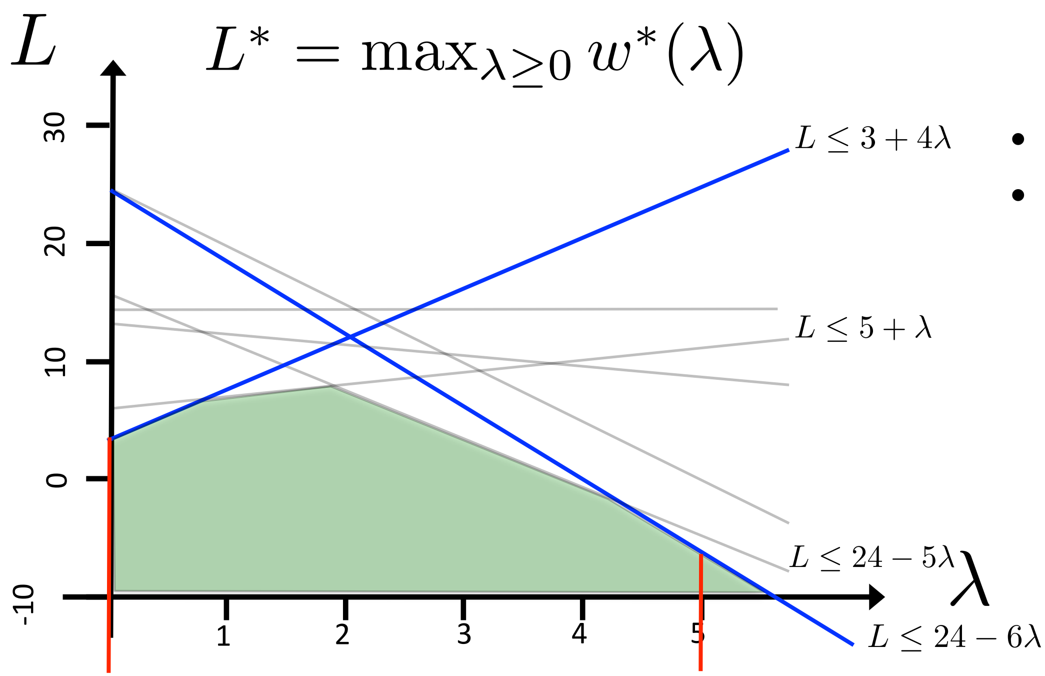
2- Solving the Lagrangian dual



- The Lagrangian dual is just a big LP
- Kelley-Cheney-Goldstein:
  cutting planes

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

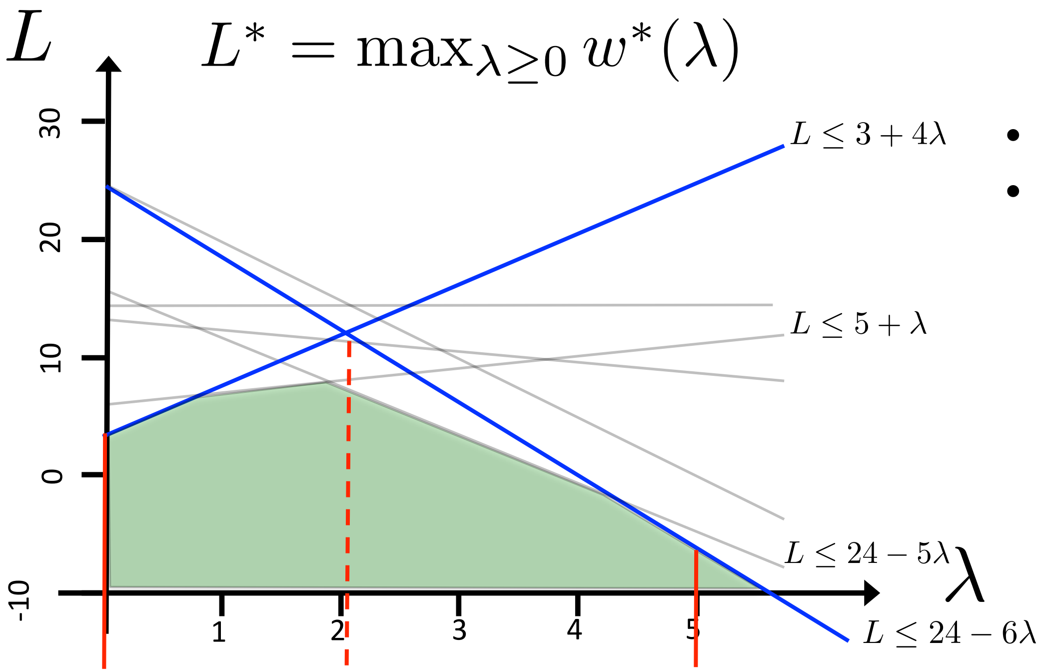
2- Solving the Lagrangian dual



- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs***, *1960*]

2- Solving the Lagrangian dual

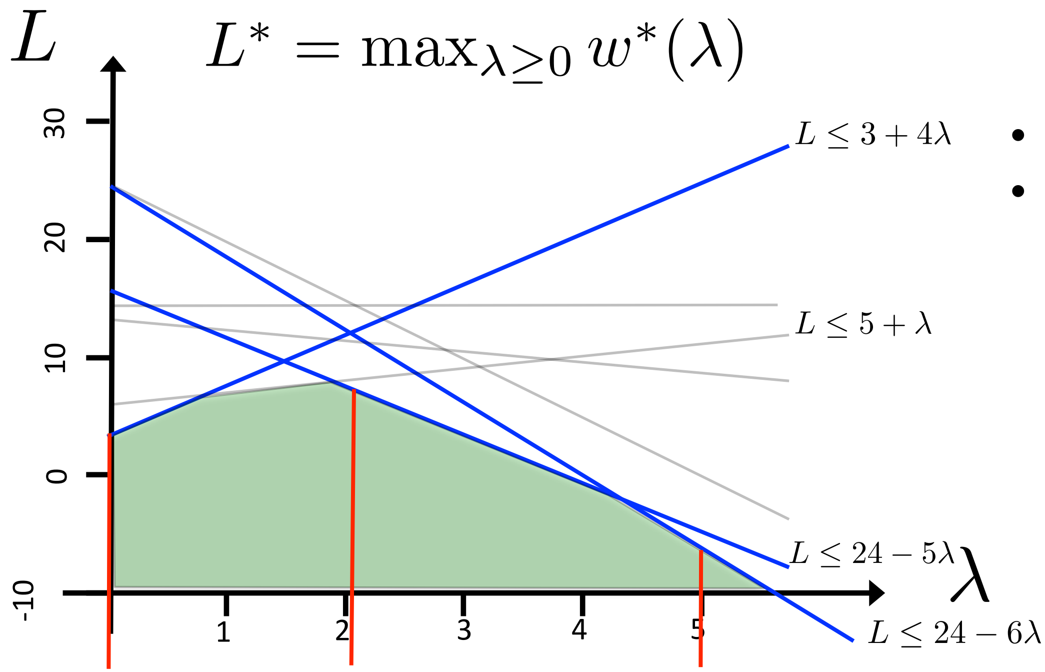


- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP: $\text{Max } L$

$$L \leq 3 + 4\lambda$$
$$L \leq 24 - 6\lambda$$

[*Kelley, J.E.* ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

2- Solving the Lagrangian dual



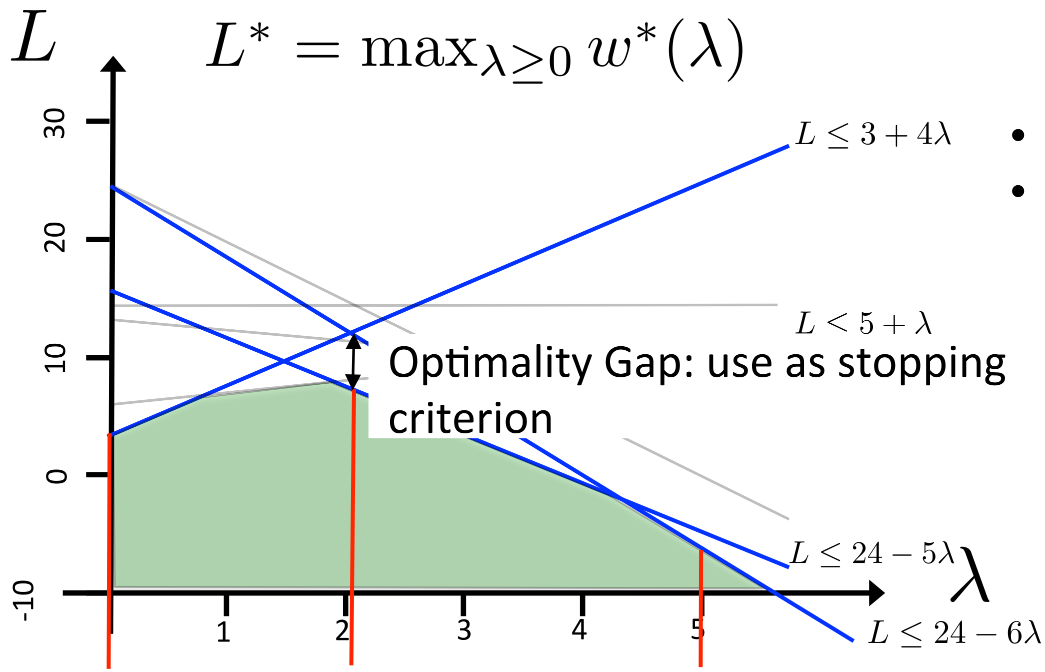
- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP:

$$\begin{aligned}&\text{Max } L\\ &L \leq 3 + 4\lambda\\ &L \leq 24 - 6\lambda\\ &L \leq 24 - 5\lambda\end{aligned}$$

*[Kelley, J.E. **The Cutting Plane Method for Solving Convex Programs**, 1960]*

2- Solving the Lagrangian dual

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$



- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes

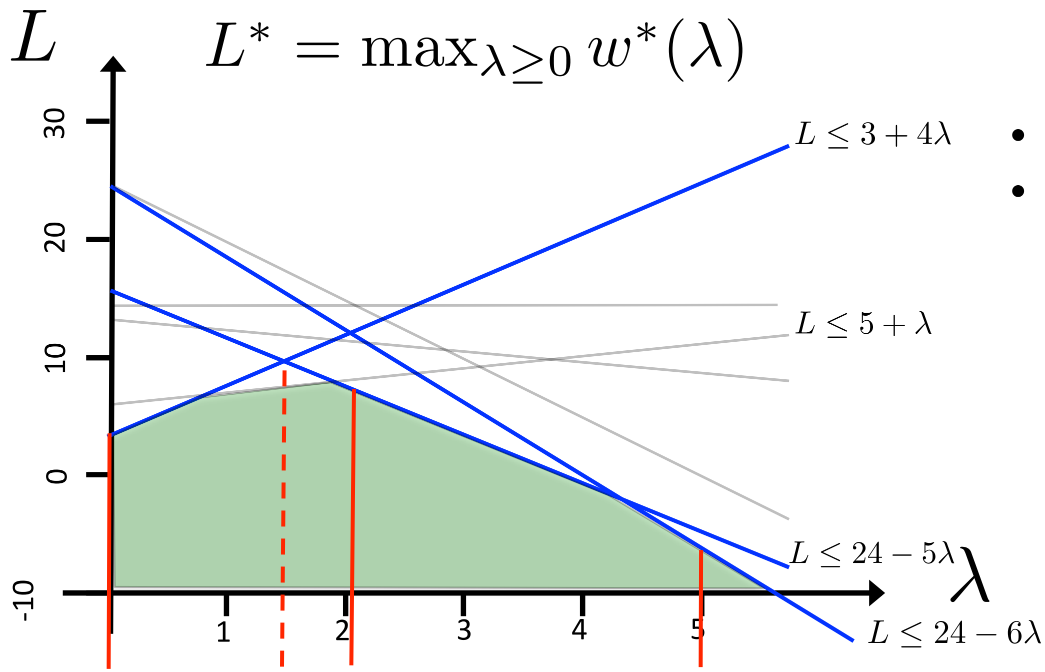
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP: $\text{Max } L$

$$L \leq 3 + 4\lambda$$
$$L \leq 24 - 6\lambda$$
$$L \leq 24 - 5\lambda$$

  - The LP provides an **upper bound**
  - Any subproblem solution is a **lower bound**

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

2- Solving the Lagrangian dual



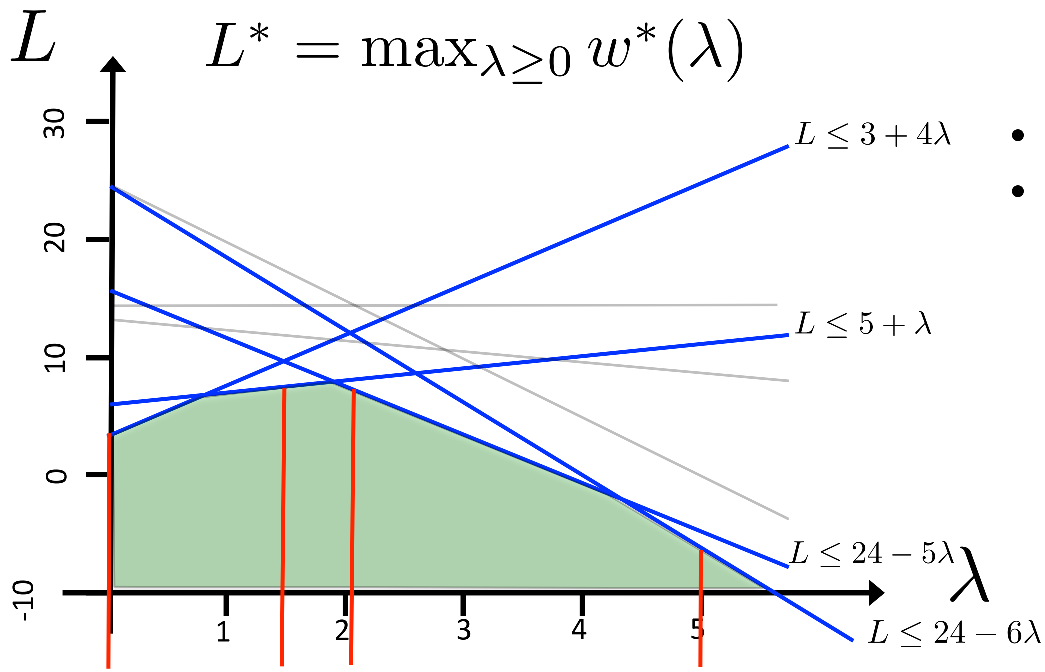
- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP: $\text{Max } L$

$$L \leq 3 + 4\lambda$$
$$L \leq 24 - 6\lambda$$
$$L \leq 24 - 5\lambda$$

  - The LP provides an **upper bound**
  - Any subproblem solution is a **lower bound**

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

2- Solving the Lagrangian dual

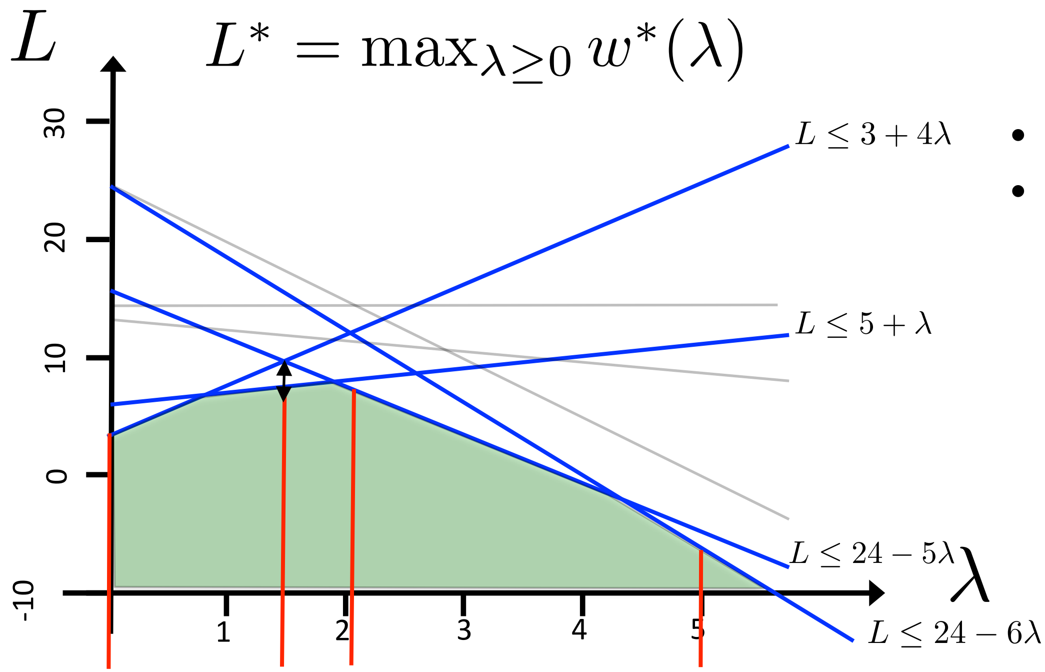


- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP:

$$\text{Max } L$$
$$L \leq 3 + 4\lambda$$
$$L \leq 24 - 6\lambda$$
$$L \leq 24 - 5\lambda$$
$$L \leq 5 + \lambda$$

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

2- Solving the Lagrangian dual

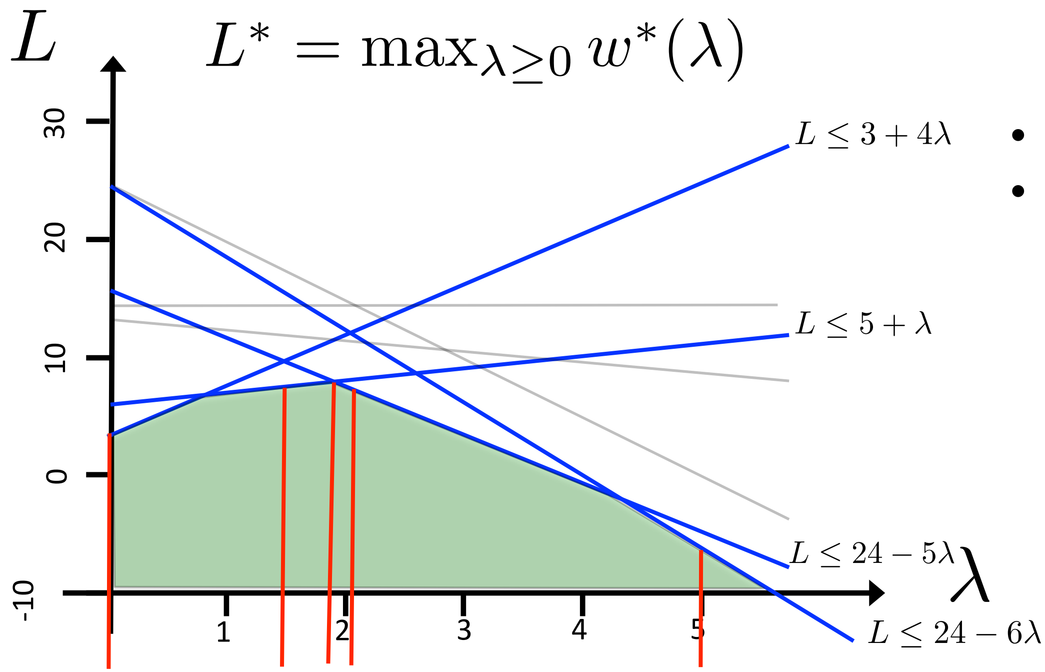


- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP:

$$\begin{aligned} &\text{Max } L \\ &L \leq 3 + 4\lambda \\ &L \leq 24 - 6\lambda \\ &L \leq 24 - 5\lambda \\ &L \leq 5 + \lambda \end{aligned}$$

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

2- Solving the Lagrangian dual

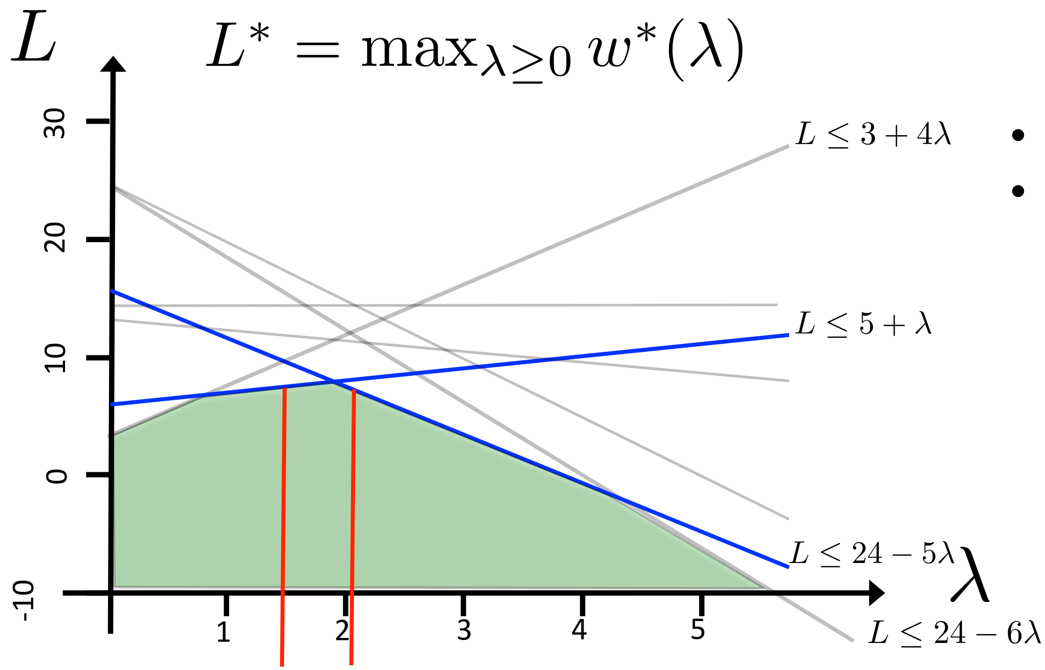


- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP:

$$\text{Max } L$$
$$L \leq 3 + 4\lambda$$
$$L \leq 24 - 6\lambda$$
$$L \leq 24 - 5\lambda$$
$$L \leq 5 + \lambda$$

*[Kelley, J.E. **The Cutting Plane Method for Solving Convex Programs**, 1960]*

2- Solving the Lagrangian dual



- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP: $\text{Max } L$

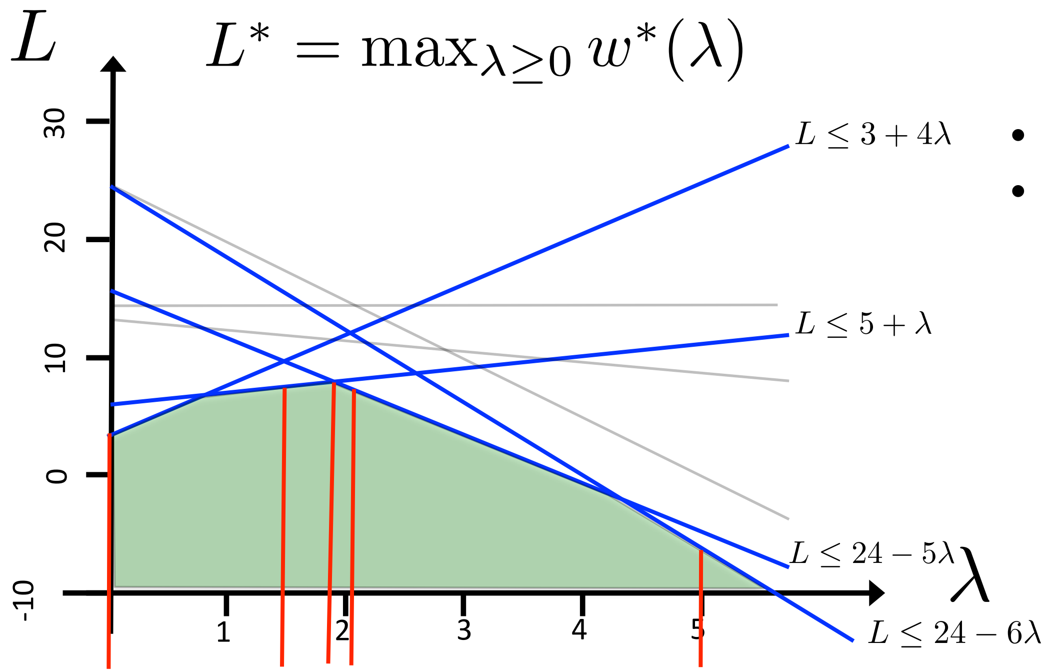
$$L \leq 3 + 4\lambda$$
$$L \leq 24 - 6\lambda$$
$$L \leq 24 - 5\lambda$$
$$L \leq 5 + \lambda$$

  - Not all inequalities are needed, optimal could be found with

$$\text{Max } L$$
$$L \leq 24 - 5\lambda$$
$$L \leq 5 + \lambda$$

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

2- Solving the Lagrangian dual



- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP: $\text{Max } L$

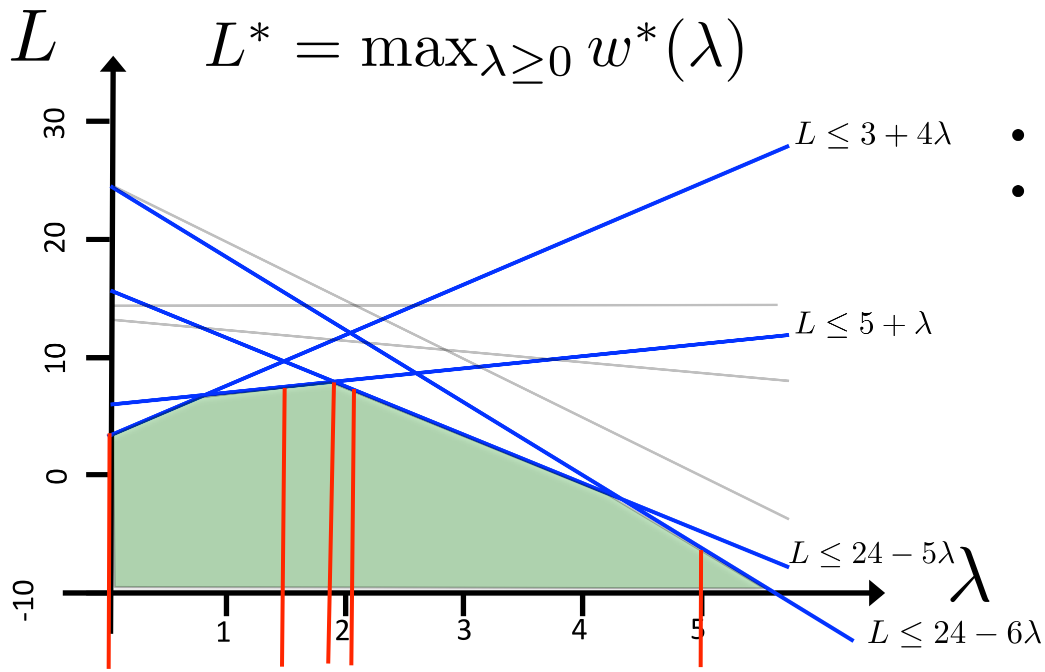
    bundle $\begin{cases} L \leq 3 + 4\lambda \\ L \leq 24 - 6\lambda \\ L \leq 24 - 5\lambda \\ L \leq 5 + \lambda \end{cases}$
  - Not all inequalities are needed, optimal could be found with

$$\text{Max } L$$

bundle $\begin{cases} L \leq 24 - 5\lambda \\ L \leq 5 + \lambda \end{cases}$

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs***, *1960*]

2- Solving the Lagrangian dual



- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP:
    $$\text{Max } L$$
    $$\text{bundle} \begin{cases} L \leq 3 + 4\lambda \\ L \leq 24 - 6\lambda \\ L \leq 24 - 5\lambda \\ L \leq 5 + \lambda \end{cases}$$
  - Not all inequalities are needed, optimal could be found with
    $$\text{Max } L$$
    $$\text{bundle} \begin{cases} L \leq 24 - 5\lambda \\ L \leq 5 + \lambda \end{cases}$$

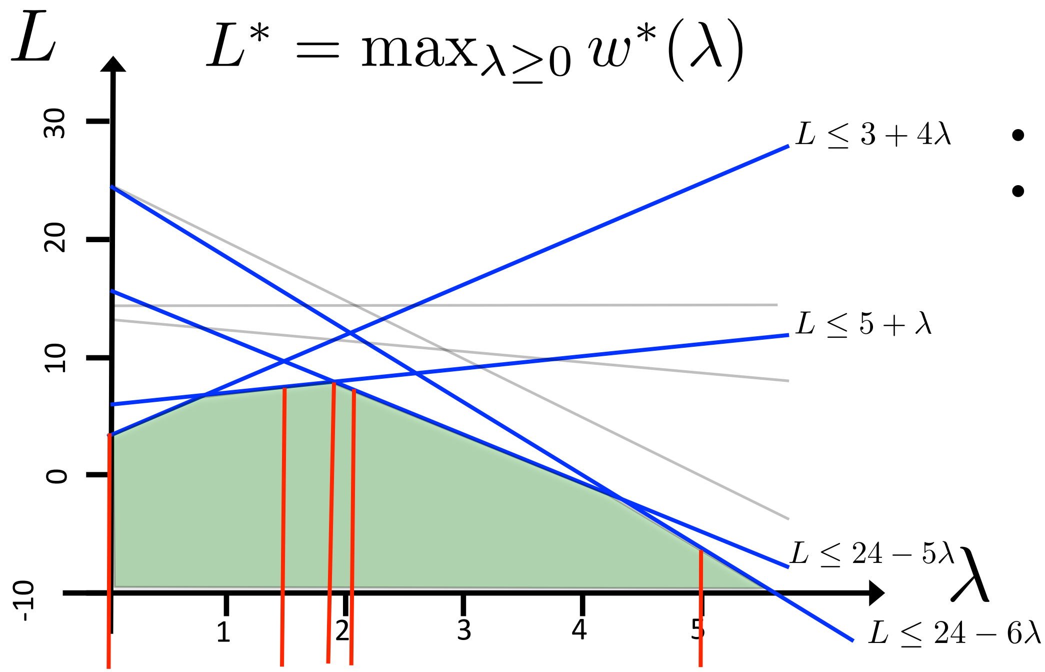
How do we make sure the LP is bounded ?
(think of the first iteration)

- Use a known feasible primal solution

[*Kelley, J.E.* ***The Cutting Plane Method for Solving Convex Programs***, *1960*]

2- Solving the Lagrangian dual

$$L^* = \max_{\lambda \geq 0} w^*(\lambda)$$



- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP:

$$\text{Max } L$$
$$\text{bundle} \begin{cases} L \leq 3 + 4\lambda \\ L \leq 24 - 6\lambda \\ L \leq 24 - 5\lambda \\ L \leq 5 + \lambda \end{cases}$$

  - Not all inequalities are needed, optimal could be found with

$$\text{Max } L$$
$$\text{bundle} \begin{cases} L \leq 24 - 5\lambda \\ L \leq 5 + \lambda \end{cases}$$

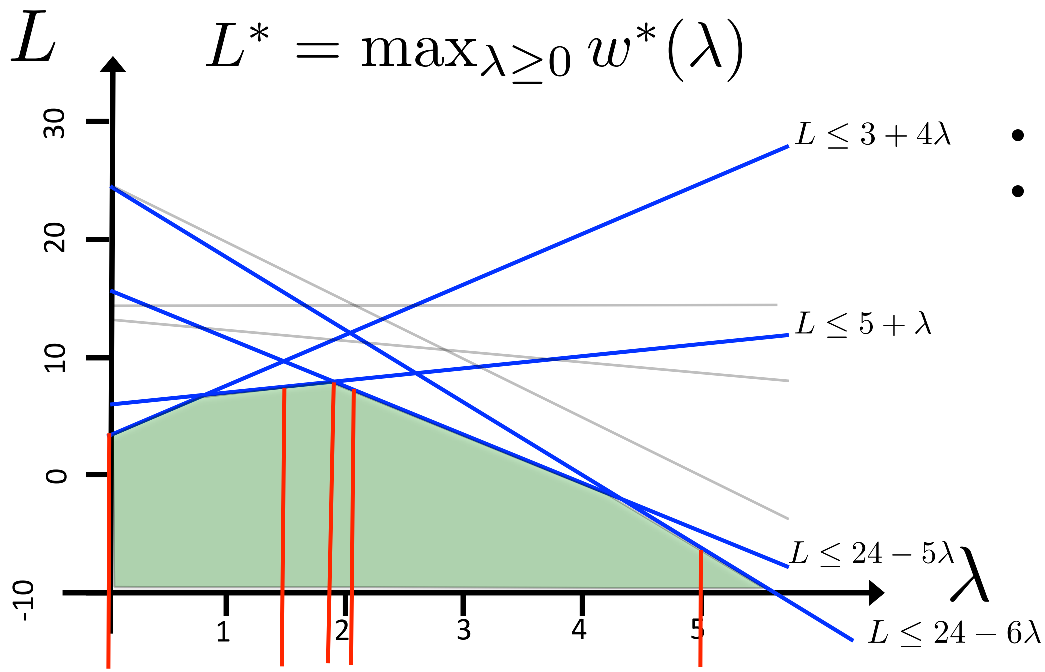
How do we make sure the LP is bounded ? (think of the first iteration)

- Use a known feasible primal solution

How to manage cutting planes over the tree search ?

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

2- Solving the Lagrangian dual



Towards Bundle methods [Lemaréchal, 2001]

- The Lagrangian dual is just a big LP
- Kelley's algorithm: cutting planes
  - Suppose you know two of the inequalities (by solving the subproblem for two values of λ)
  - Current LP: Max L

bundle
$$\begin{cases} L \leq 3 + 4\lambda \\ L \leq 24 - 6\lambda \\ L \leq 24 - 5\lambda \\ L \leq 5 + \lambda \end{cases}$$

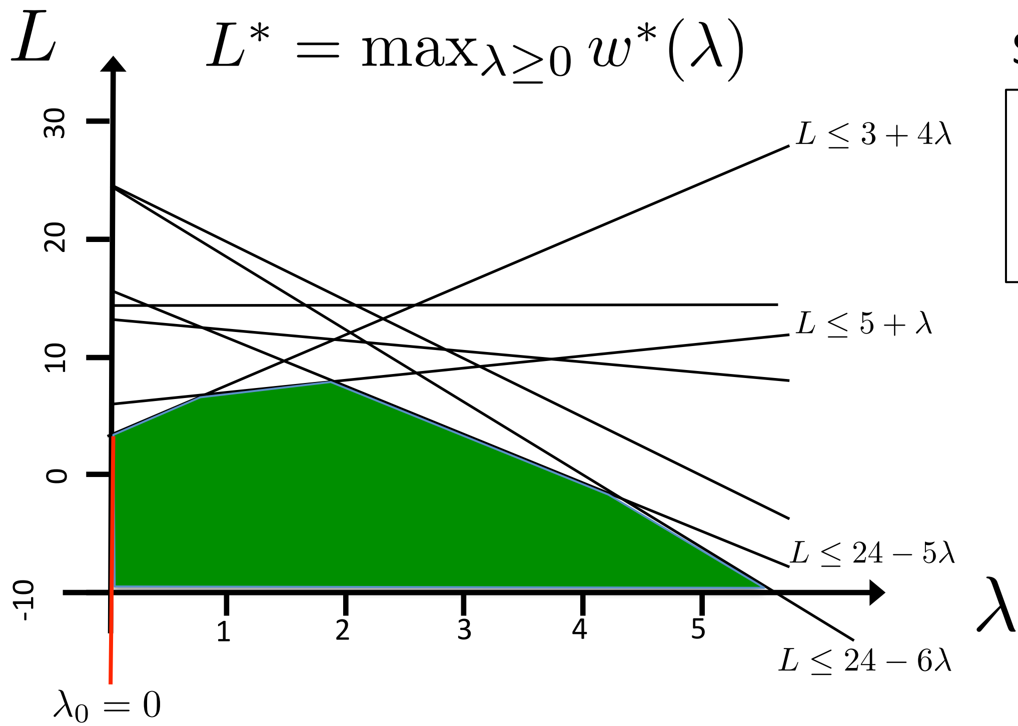
  - Not all inequalities are needed, optimal could be found with

Max L

bundle
$$\begin{cases} L \leq 24 - 5\lambda \\ L \leq 5 + \lambda \end{cases}$$

[Kelley, J.E. ***The Cutting Plane Method for Solving Convex Programs****, 1960*]

2- Solving the Lagrangian dual

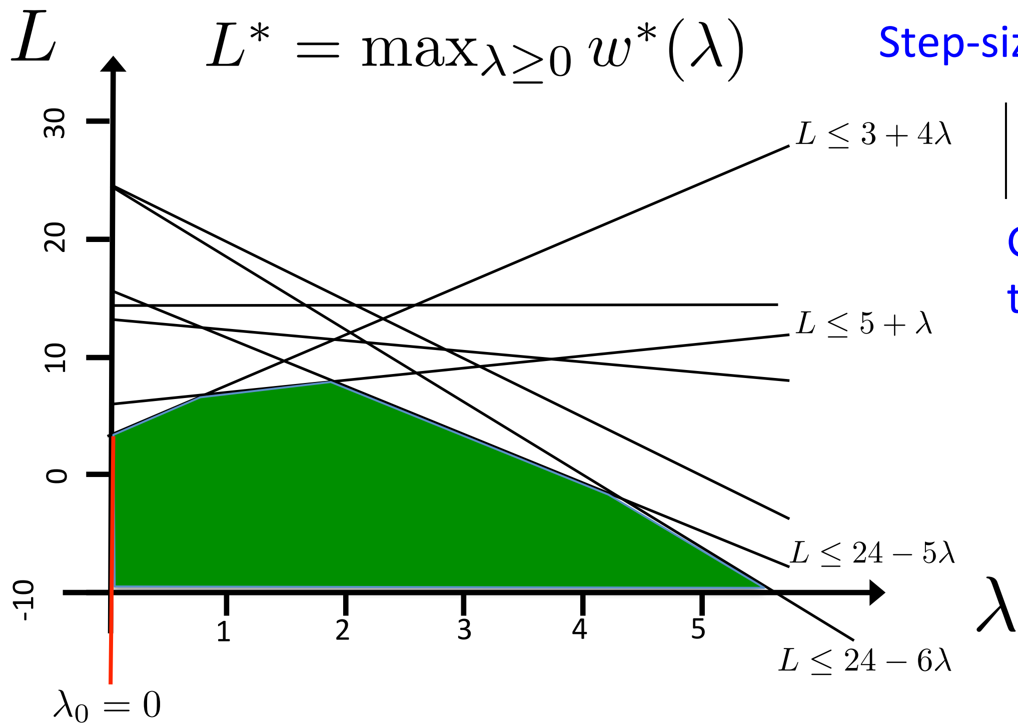


Subgradient algorithm:

$$\lambda_{k+1} \leftarrow \max(0, \lambda_k + \mu_k(\sum t_{ij} x^k - T))$$
$$\mu_{k+1} = \mu_0 (3/5)^k$$

$$\lambda_0 = 0$$
$$\mu_0 = 1$$

2- Solving the Lagrangian dual



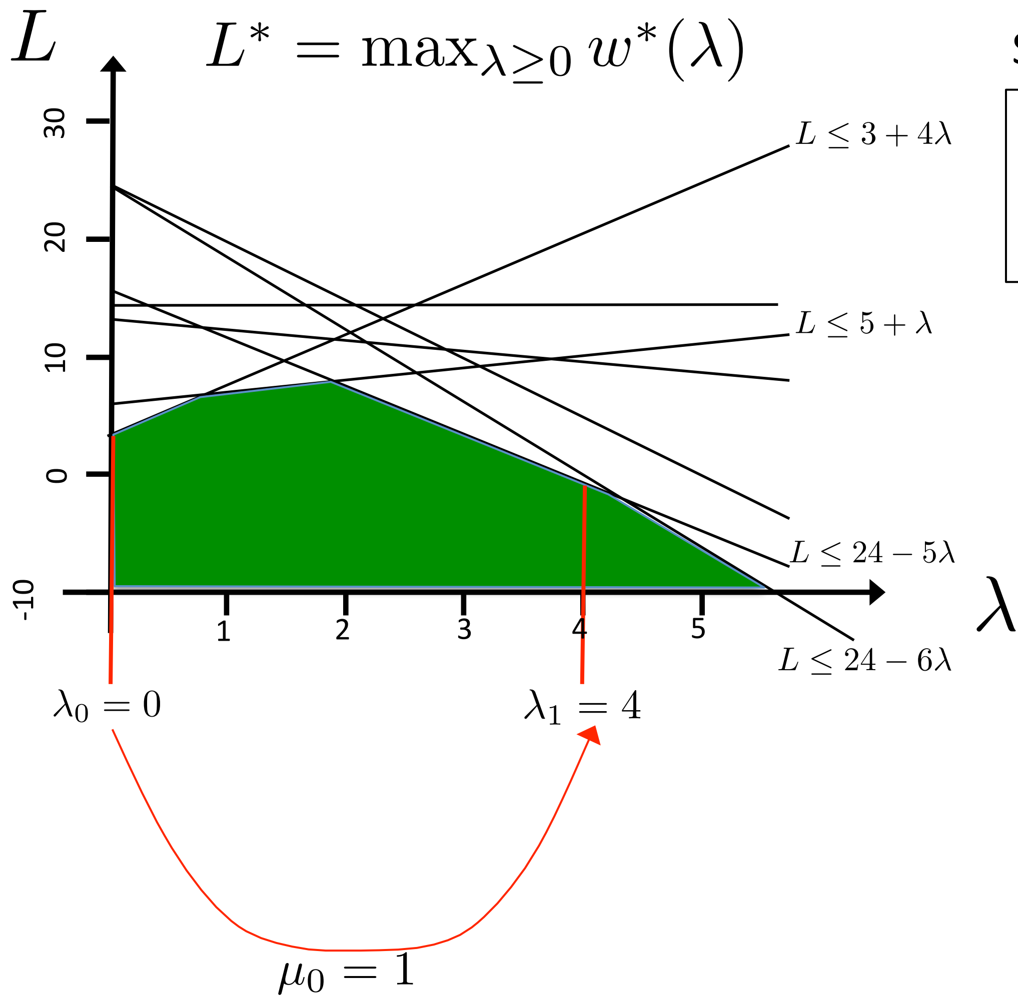
Step-size rule: how far the direction is followed

$$\lambda_{k+1} \leftarrow \max(0, \lambda_k + \mu_k(\sum t_{ij} x^k - T))$$

Gives a direction (decrease or increase) the multiplier

$$\lambda_0 = 0$$
$$\mu_0 = 1$$

2- Solving the Lagrangian dual



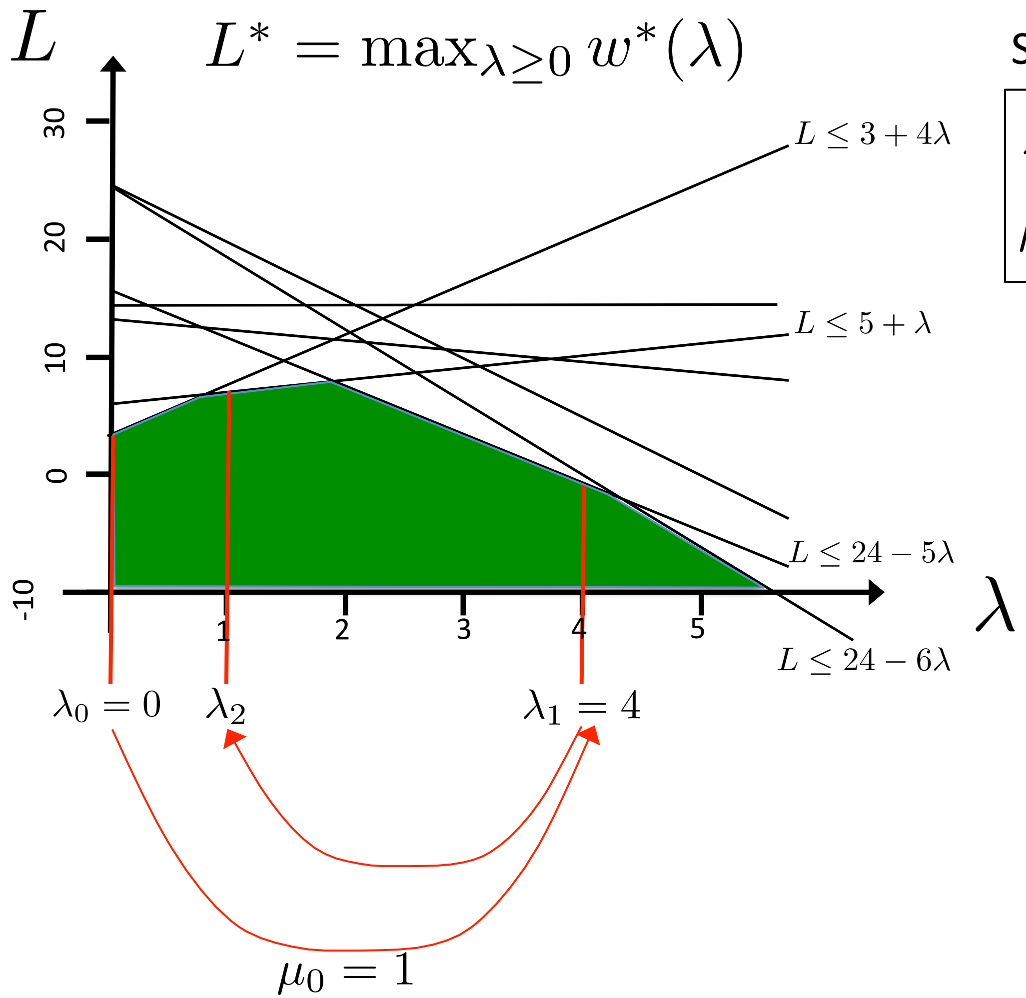
Subgradient algorithm:

$$\lambda_{k+1} \leftarrow \max(0, \lambda_k + \mu_k(\sum t_{ij} x^k - T))$$
$$\mu_{k+1} = \mu_0 (3/5)^k$$

$\lambda_0 = 0$
$\mu_0 = 1$

$\lambda_1 = 4$
$\mu_1 = 0.6$

2- Solving the Lagrangian dual



Subgradient algorithm:

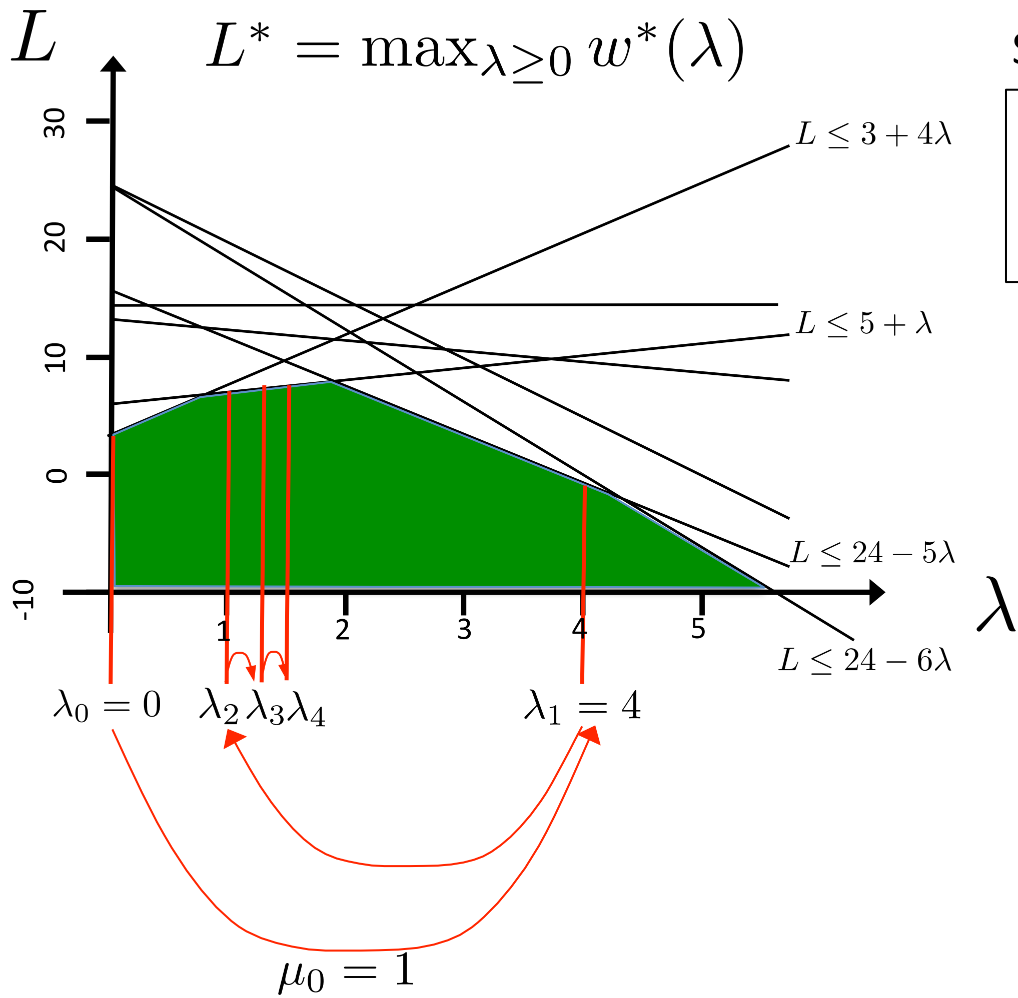
$$\lambda_{k+1} \leftarrow \max(0, \lambda_k + \mu_k(\sum t_{ij} x^k - T))$$
$$\mu_{k+1} = \mu_0 (3/5)^k$$

$\lambda_0 = 0$
$\mu_0 = 1$

$\lambda_1 = 4$
$\mu_1 = 0.6$

$\lambda_2 = 1$
$\mu_2 = 0.6^2 = 0.36$

2- Solving the Lagrangian dual



Subgradient algorithm:

$$\lambda_{k+1} \leftarrow \max(0, \lambda_k + \mu_k(\sum t_{ij} x^k - T))$$
$$\mu_{k+1} = \mu_0 (3/5)^k$$

$\lambda_0 = 0$
$\mu_0 = 1$

$\lambda_1 = 4$
$\mu_1 = 0.6$

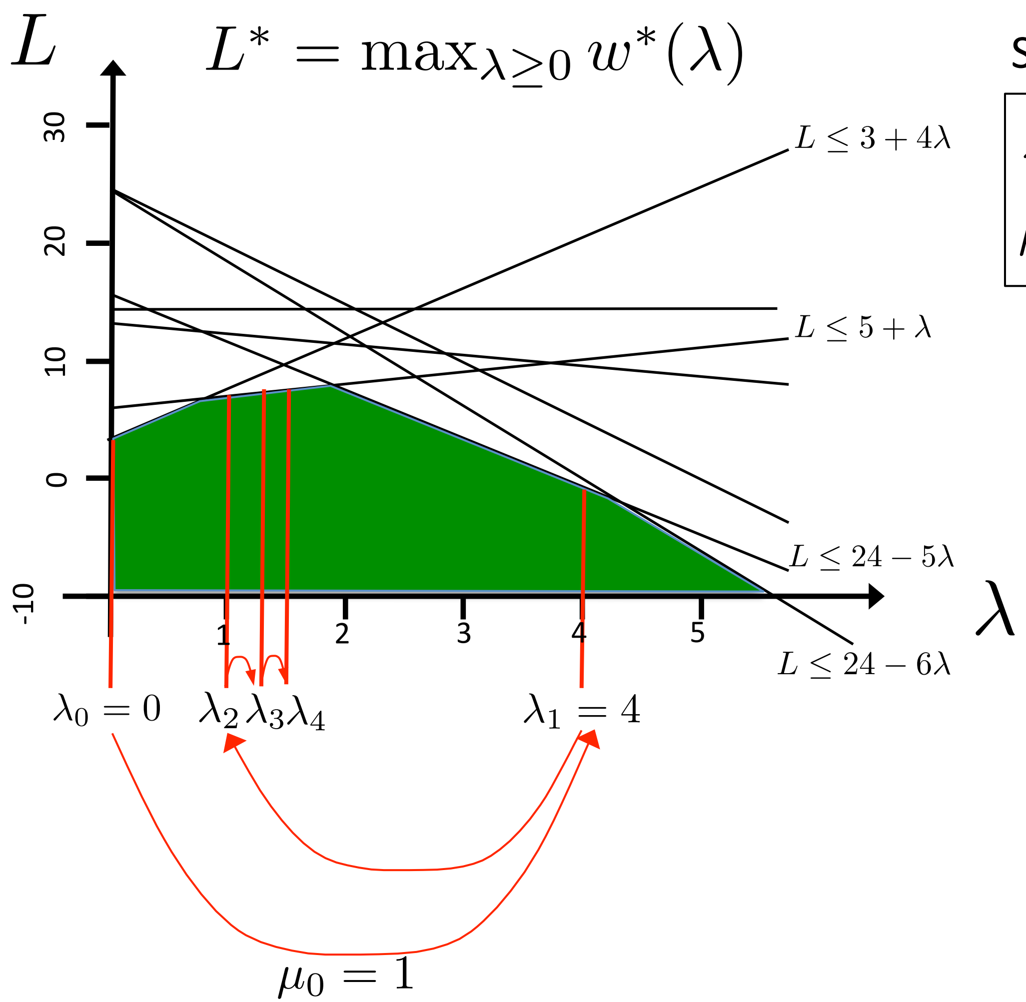
$\lambda_2 = 1$
$\mu_2 = 0.6^2 = 0.36$

$\lambda_3 = 1.36$
$\mu_3 = 0.6^3 = 0.216$

$\lambda_4 = 1.57$

$\cdots$

2- Solving the Lagrangian dual



Subgradient algorithm:

$$\lambda_{k+1} \leftarrow \max(0, \lambda_k + \mu_k(\sum t_{ij} x^k - T))$$
$$\mu_{k+1} = \mu_0 (3/5)^k$$

$\lambda_0 = 0$
$\mu_0 = 1$

$\lambda_1 = 4$
$\mu_1 = 0.6$

$\lambda_2 = 1$
$\mu_2 = 0.6^2 = 0.36$

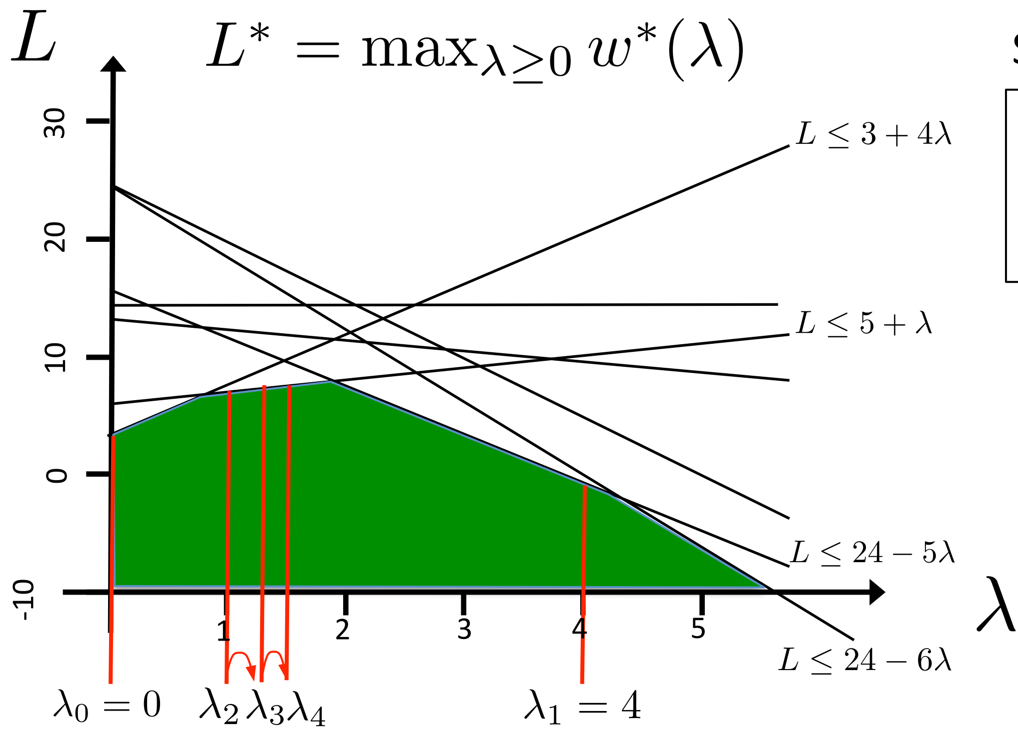
$\lambda_3 = 1.36$
$\mu_3 = 0.6^3 = 0.216$

$\lambda_4 = 1.57$
$\cdots$

To ensure convergence, **we should have**:

$$\mu_k \to 0 \text{ and } \sum_{j=1}^{k} \mu_j \to \infty$$

2- Solving the Lagrangian dual



Subgradient algorithm:

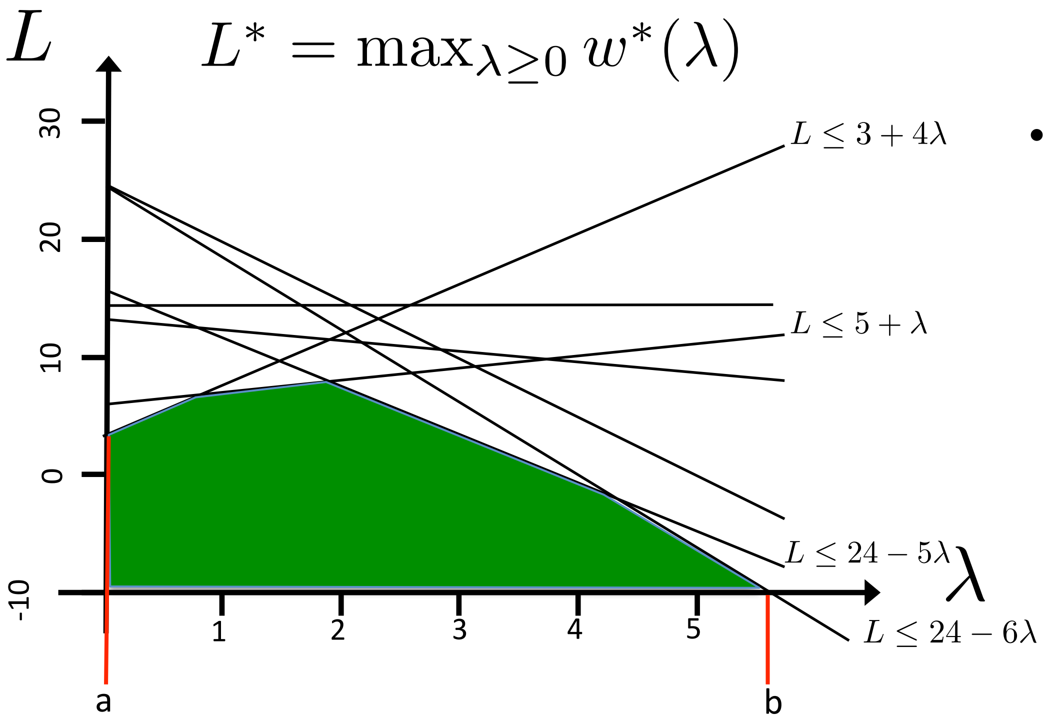
$$\lambda_{k+1} \leftarrow \max(0, \lambda_k + \mu_k(\sum t_{ij} x^k - T))$$
$$\mu_{k+1} = \mu_0 (3/5)^k$$

- Many stepsize rules exist.
  The one used by Held and Karp: $\mu_k = \frac{\beta(\overline{z} - w^*(\lambda_k))}{(\sum t_{ij} x^k - T)^2}$ with $\beta \in]0, 2[$

- History of gradients can be used like in ***deflection***: use as direction a convex combination of the current **gradient** and previous **direction**

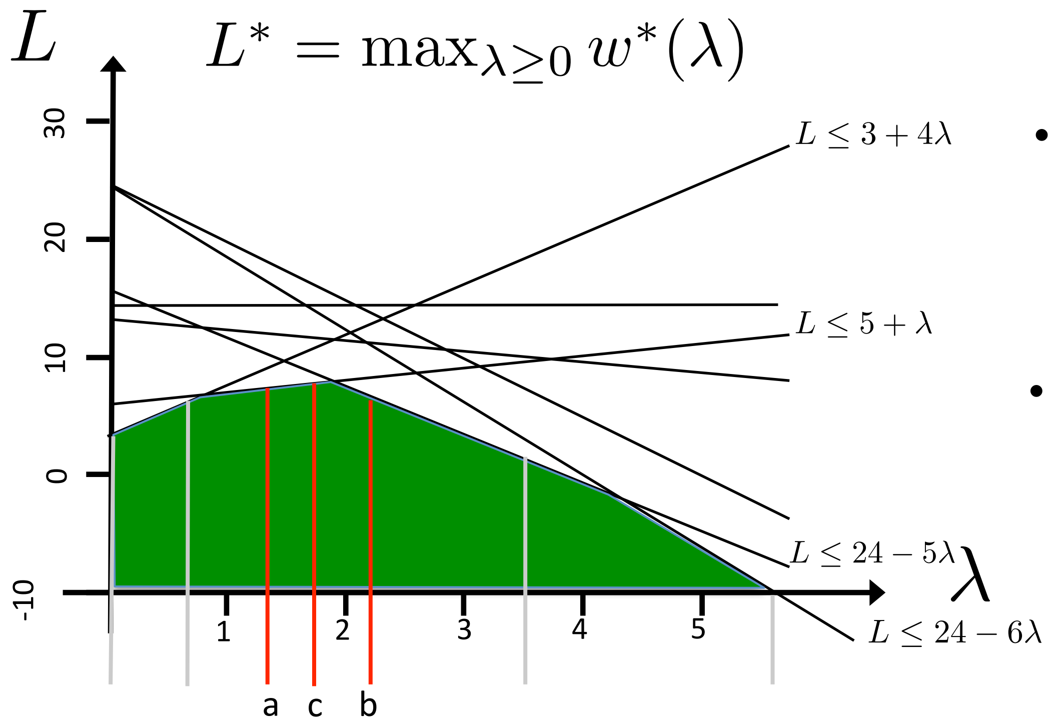
  [Lombardi, M., Gualandi, S.
  ***A New Propagator for Two-Layer Neural Networks in Empirical Model Learning**. CP 2013]*

2- Solving the Lagrangian dual



- We are looking for the maximum of a 1-dimensional concave function in the interval [a,b]

2- Solving the Lagrangian dual



- We are looking for the maximum of a 1-dimensional concave function in the interval [a,b]

- Golden search method:

$$c = a + \frac{1}{\phi}(b - a)$$
$$d = b - \frac{1}{\phi}(b - a)$$

- Ok for one multiplier: propagating two constraints with LR
- Used in *[Sellmann, M., Fahle, T.* ***Constraint Programming based Lagrangian Relaxation for the Automatic Recording Problem****, Annals of OR, 2003]*

2- Solving the Lagrangian dual

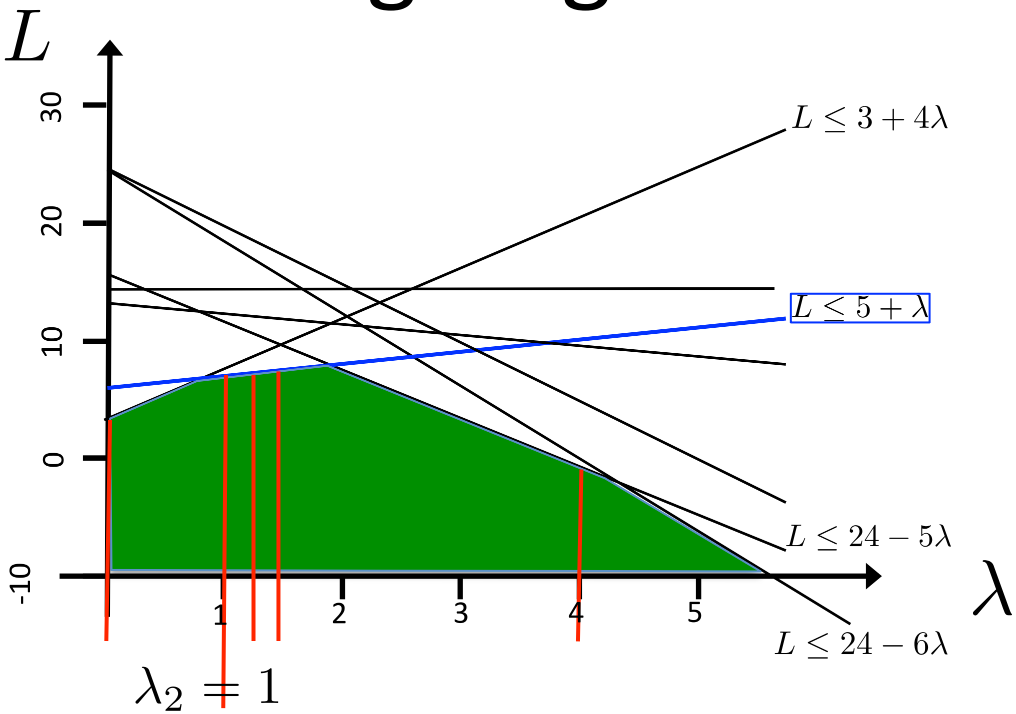
Solving

- Kelley, Cheney-Goldstein algorithm
- Subgradient techniques
- Golden section search for a single multiplier

Filtering from the Lagrangian subproblem

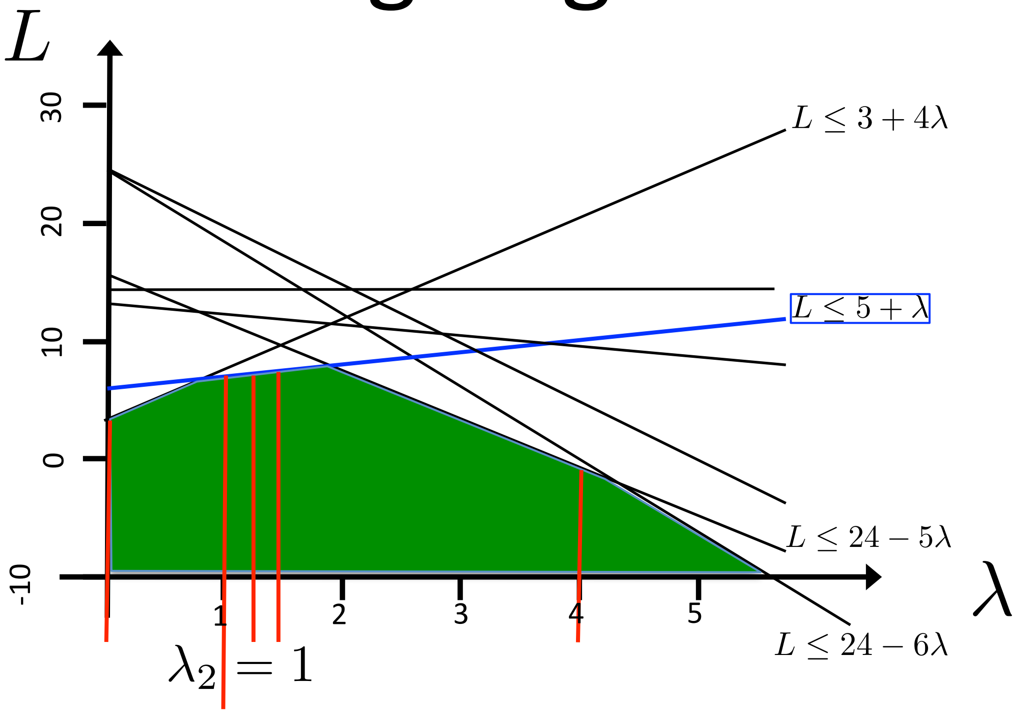
Illustration: multi-cost regular

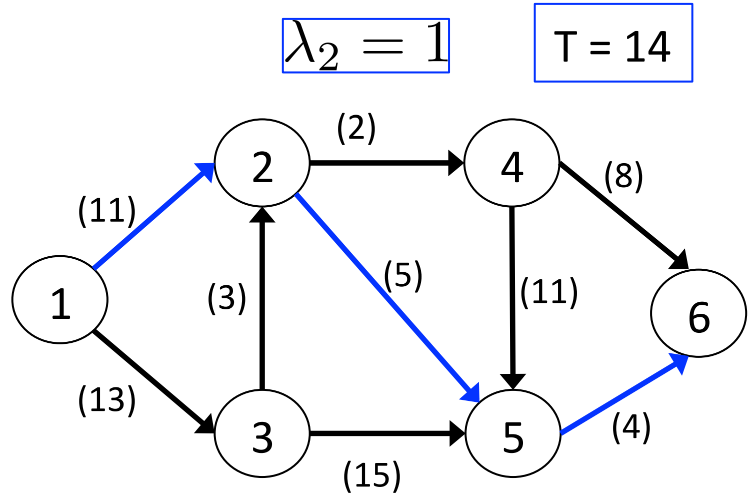
2- Lagrangian relaxation - Filtering



– We can filter from any Lagrangian subproblem and its $w^*(\lambda)$

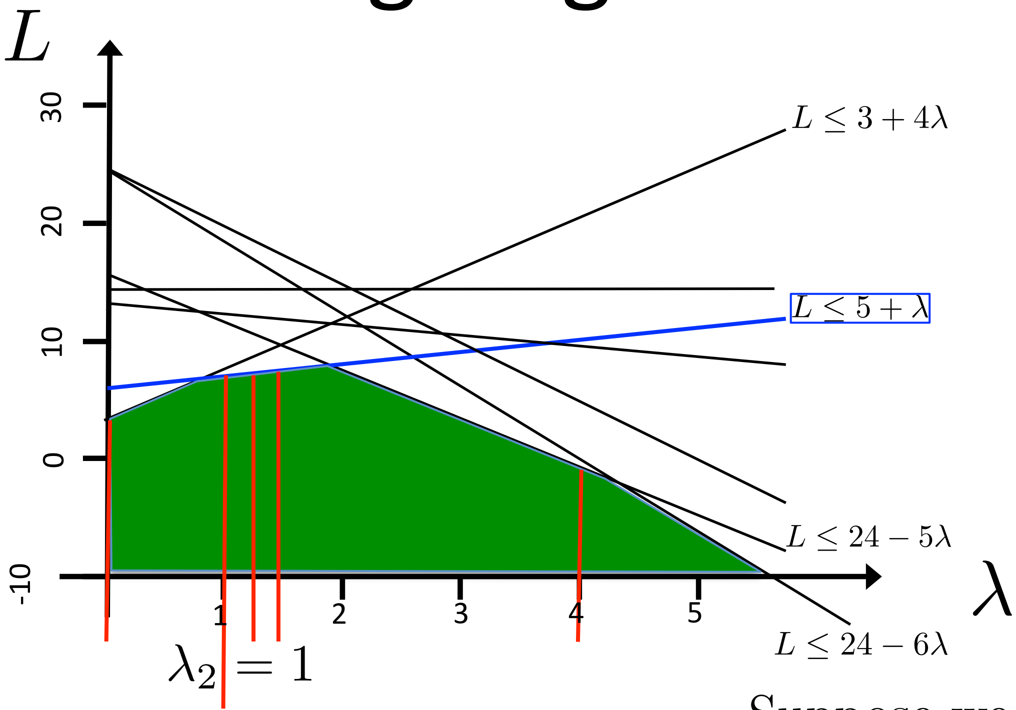
2- Lagrangian relaxation - Filtering

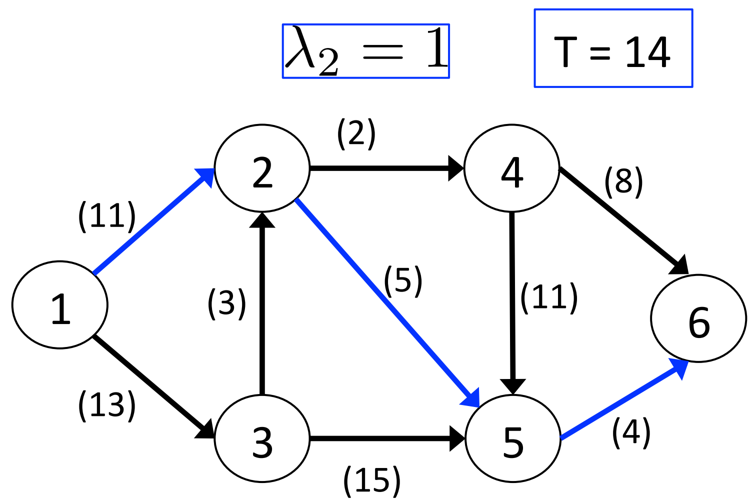




$$w^*(1) = 20 - 14 = 6 \leq z^*$$

2- Lagrangian relaxation - Filtering

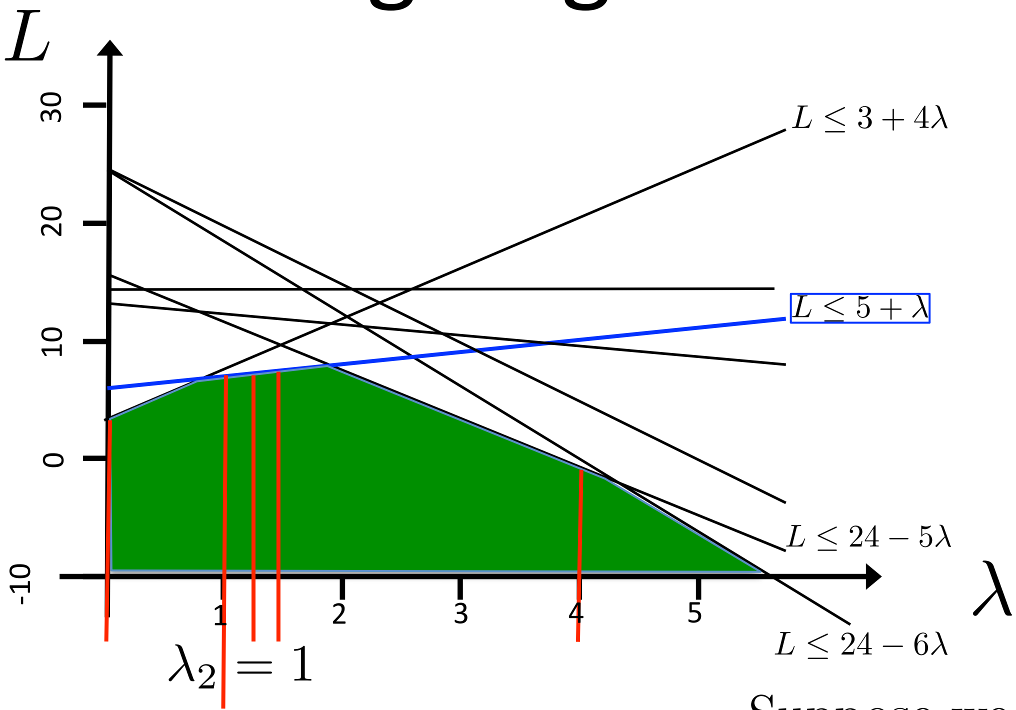




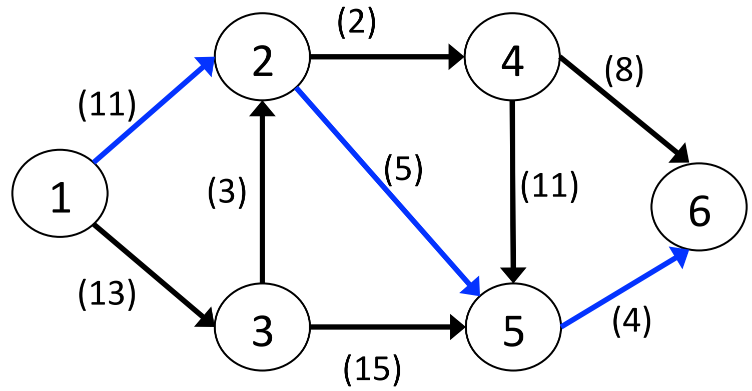
$$w^*(1) = 20 - 14 = 6 \quad \leq z^*$$

Suppose we know an upper bound of $\overline{z} = 15$

2- Lagrangian relaxation - Filtering

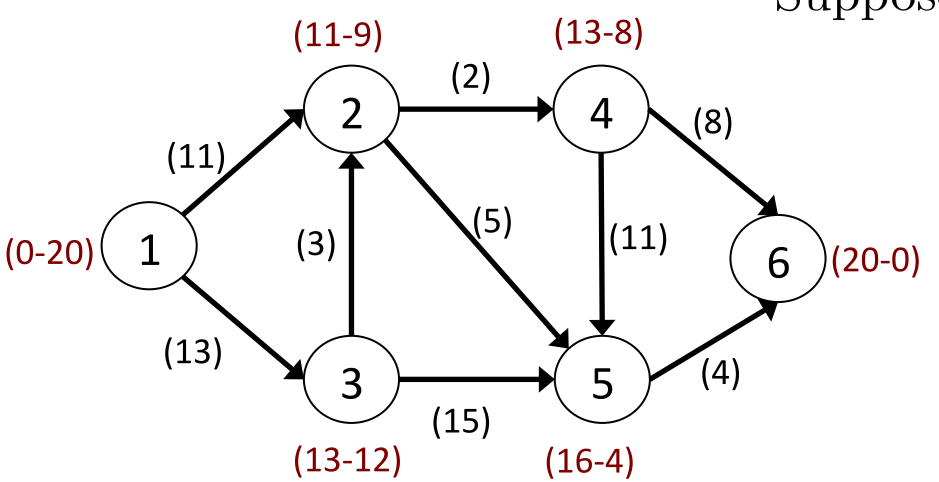


$\lambda_2 = 1$ T = 14



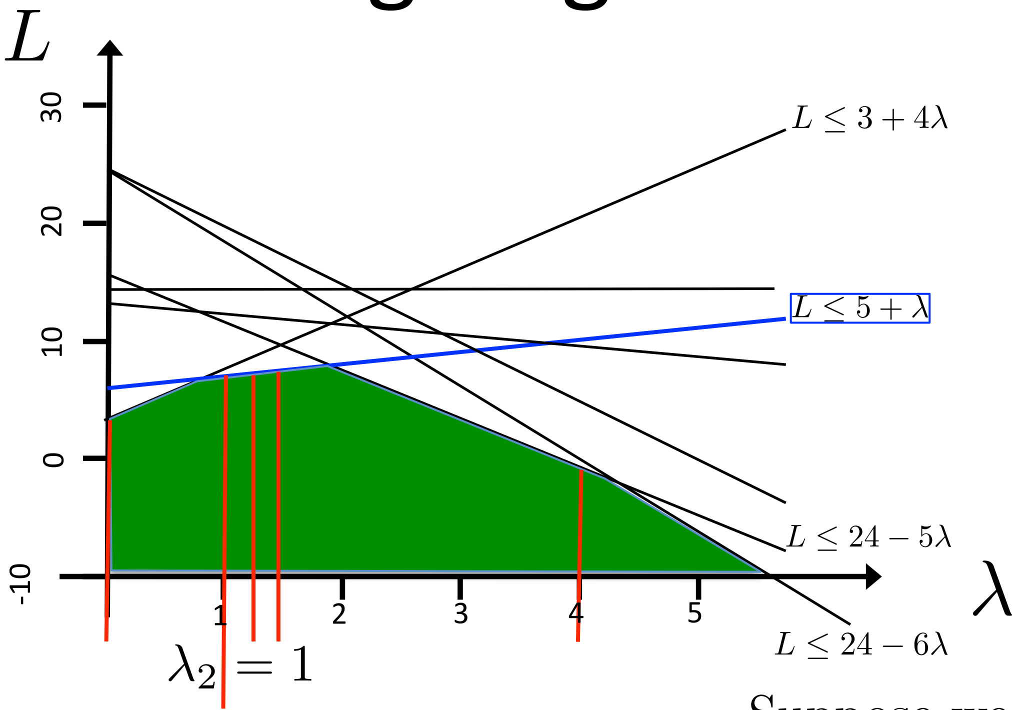
$$w^*(1) = 20 - 14 = 6 \quad \leq z^*$$

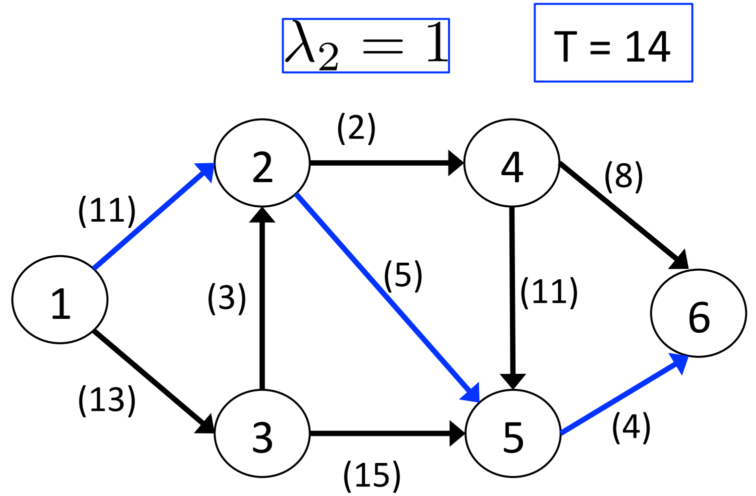
Suppose we know an upper bound of $\overline{z} = 15$



We compute shortest path from source to all other nodes and from all other nodes to sink

2- Lagrangian relaxation - Filtering

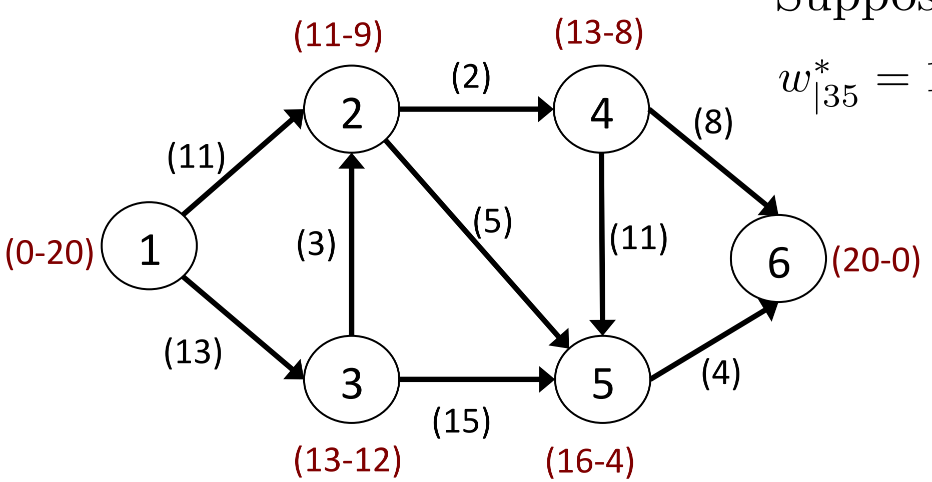




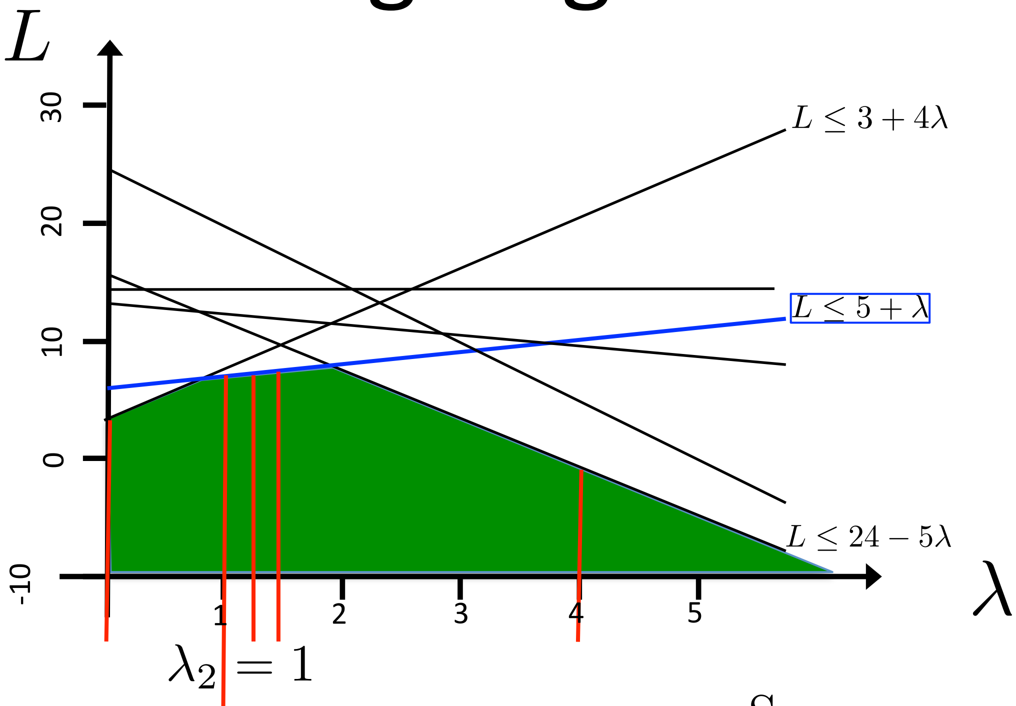
$$w^*(1) = 20 - 14 = 6 \quad \leq z^*$$

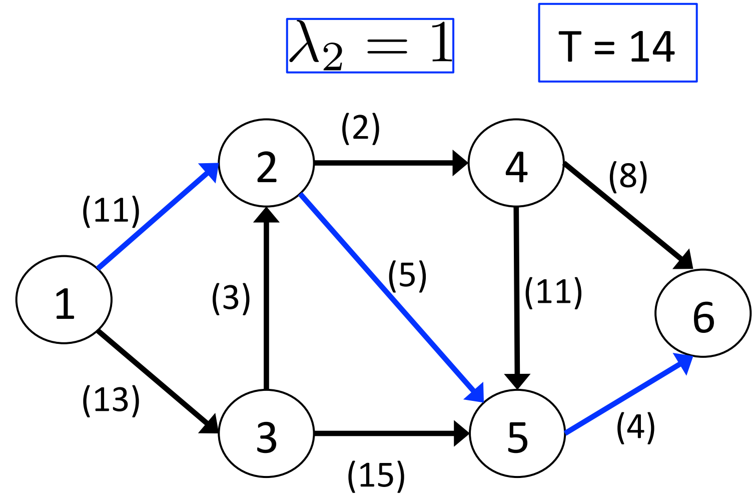
Suppose we know an upper bound of $\overline{z} = 15$

$$w^*_{|35} = 13 + (15) + 4 - 14 = 18 > \overline{z} = 15 \Rightarrow x_{35} = 0$$



2- Lagrangian relaxation - Filtering





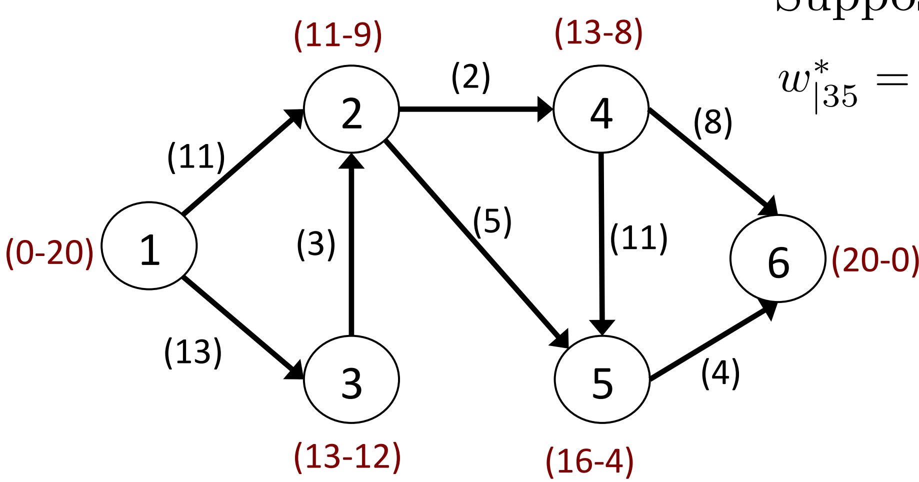
$$w^*(1) = 20 - 14 = 6 \quad \leq z^*$$

Suppose we know an upper bound of $\overline{z} = 15$

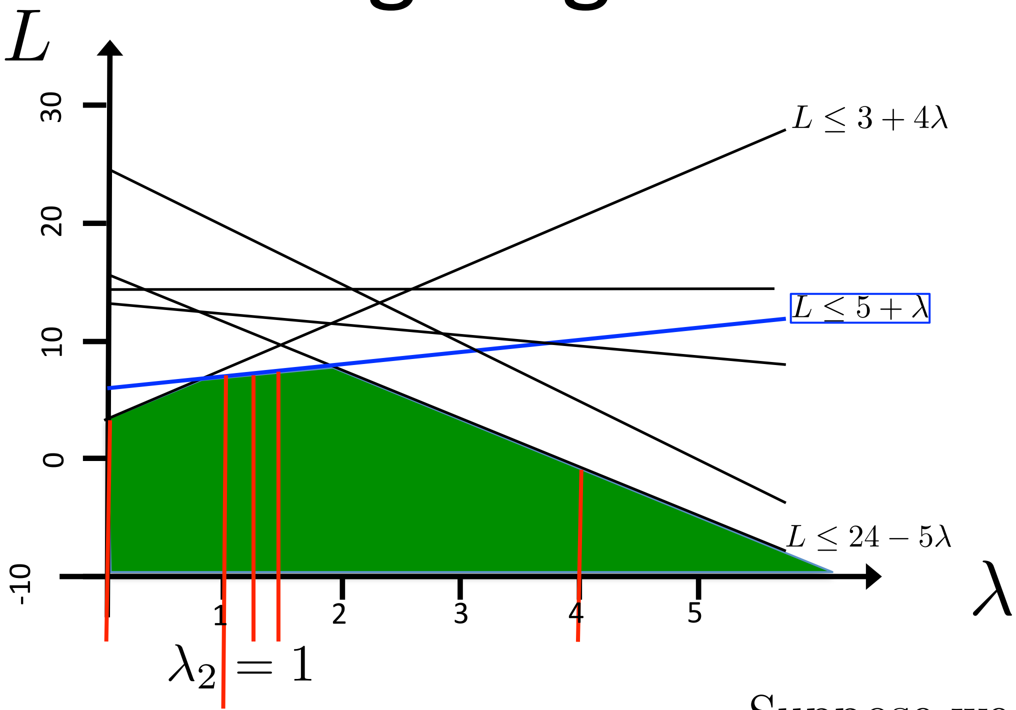
$$w^*_{|35} = 13 + (15) + 4 - 14 = 18 > \overline{z} = 15 \Rightarrow x_{35} = 0$$

[Sellmann, 2004]

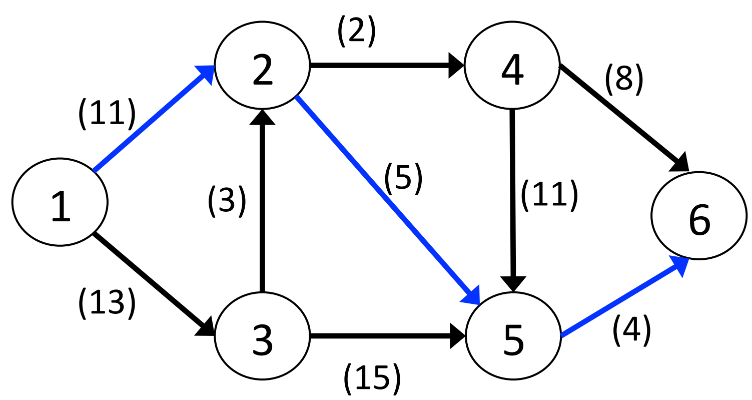
– Lagrangian dual is changed ! does it affect convergence ?



2- Lagrangian relaxation - Filtering



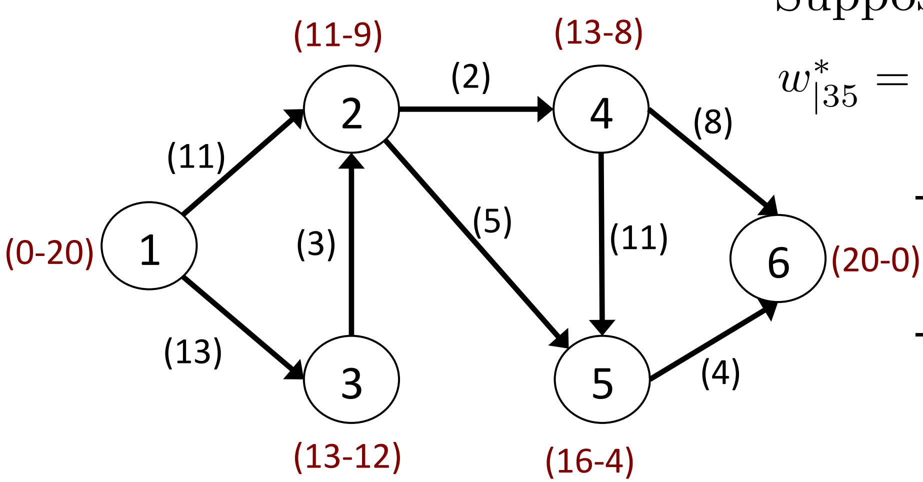
$\lambda_2 = 1$ T = 14



$$w^*(1) = 20 - 14 = 6 \quad \leq z^*$$

Suppose we know an upper bound of $\overline{z} = 15$

$$w^*_{|35} = 13 + (15) + 4 - 14 = 18 > \overline{z} = 15 \Rightarrow x_{35} = 0$$



- Lagrangian dual is changed [Sellmann, 2004]

  does it affect convergence ?
- Filtering takes place near L* most of the time but not necessarily

  What values of λ are good for filtering ?

2- Solving the Lagrangian dual

- And during search ? solve LR
  - *only* at root node [Bergman, 2015] and re-apply filtering only when upper bound is improved
  - *at root node and with a few iterations* of sub-gradients to gather filtering
  - *at each node*

- Reuse information:
  - Restore best multipliers upon backtracking
  - Start solving the dual from the last multipliers found at previous node
  - Restore cutting-planes (subproblem solutions) upon backracking (for Kelley's algorithm)

2- Solving the Lagrangian dual

Solving

- Kelley, Cheney-Goldstein algorithm
- Subgradient techniques
- Golden section search for a single multiplier

Filtering from the Lagrangian subproblem

Illustration: multi-cost regular

2- Lagrangian relaxation - Multi-cost regular

- Regular : $\textsc{Regular}([X_1, \ldots, X_n], A)$ [Pesant, 2004]
  - Propagation based on breath-first-search in the unfolded automaton

2- Lagrangian relaxation - Multi-cost regular

Automaton

- Regular : $\textsc{Regular}([X_1, \ldots, X_n], A)$ [Pesant, 2004]
  - Propagation based on breath-first-search in the unfolded automaton

2- Lagrangian relaxation - Multi-cost regular

- Regular : $\textsc{Regular}([X_1, \ldots, X_n], A)$ [Pesant, 2004]
  - Propagation based on breath-first-search in the unfolded automaton

- Cost regular : $\textsc{Regular}([X_1, \ldots, X_n], A) \wedge \sum_{i=1}^{n} c_{iX_i} = Z$
  - Propagation based on shortest/longest path in the unfolded automaton [Demassey, Pesant, Rousseau, 2004]

2- Lagrangian relaxation - Multi-cost regular

- Regular : $\textsc{Regular}([X_1, \ldots, X_n], A)$ [Pesant, 2004]
  - Propagation based on breath-first-search in the unfolded automaton

- Cost regular : $\textsc{Regular}([X_1, \ldots, X_n], A) \wedge \sum_{i=1}^{n} c_{iX_i} = Z$
  - Propagation based on shortest/longest path in the unfolded automaton [Demassey, Pesant, Rousseau, 2004]

- Multi-cost regular : $\textsc{Multi-cost Regular}([X_1, \ldots, X_n], [Z^1, \ldots, Z^R], A)$

$$\textsc{Regular}([X_1, \ldots, X_n], A) \wedge (\sum_{i=1}^{n} c^r_{iX_i} = Z^r, \forall\, r = 0, \ldots, R)$$

  - Propagation based on **resource constrained shortest/longest path**
  - Sequencing and counting at the same time
    - Personnel scheduling
    - Routing
  - Example: combine Regular and GCC

[Menana, Demassey, 2009]

2- Lagrangian relaxation - Multi-cost regular

- Multi-cost regular :

$$\textsc{Regular}([X_1, \ldots, X_n], A) \; \wedge \; (\sum_{i=1}^{n} c^r_{iX_i} = Z^r, \forall\, r = 0, \ldots, R)$$

  - Propagation based on **resource constrained shortest/longest path**
  - Sequencing and counting at the same time [Menana, Demassey, 2009]
    - Personnel scheduling
    - Routing
  - Example: combine Regular and GCC

- Filtering algorithm based on Lagrangian Relaxation
  - Lagrangian subproblems: shortest path
  - Algorithm for the Lagrangian dual: subgradient

3- Domain filtering for Nvalue using LR

NValue

- Linear relaxation and reduced cost based filtering
- Lagrangian relaxation and cost based filtering

NValue

$$\textsc{NValue}([X_1,\ldots,X_n],N) \Leftrightarrow |\{j \mid \exists X_i \in \mathcal{X}, X_i = j\}| = N$$

Enforce N to be the number of distinct values appearing in the set X of variables

$$\textsc{AtMostNValue}([X_1,\ldots,X_n],N)$$

$$\left(|\{j \mid \exists X_i \in \mathcal{X}, X_i = j\}| \leq N \right)$$

+

$$\textsc{AtLeastNValue}([X_1,\ldots,X_n],N)$$

$$\left(|\{j \mid \exists X_i \in \mathcal{X}, X_i = j\}| \geq N \right)$$

AtMostNValue

$\text{AtMostNValue}([X_1, \ldots, X_6], N)$

$D(X_1)$ = {1, 2, 3, 4, 5, 6}

$D(X_2)$ = {2, 4}

$D(X_3)$ = {1, 2}

$D(X_4)$ = {1, 2, 3}

$D(X_5)$ = {4, 5}

$D(X_6)$ = {4, 5}

$D(N)$ = {1, 2}

A solution:

$\text{AtMostNValue}([2, 2, 2, 2, 4, 4, 2], 2)$

AtMostNValue

$\textsc{AtMostNValue}([X_1, \ldots, X_6], N)$

$D(X_1)$ = {1, 2, 3, 4, 5, 6}
$D(X_2)$ = {2, 4}
$D(X_3)$ = {1, 2}
$D(X_4)$ = {1, 2, 3}
$D(X_5)$ = {4, 5}
$D(X_6)$ = {4, 5}
$D(N)$ = {1, 2}

$D(X_1)$ = {1, 2, ~~3~~, 4, 5, ~~6~~}
$D(X_2)$ = {2, 4}
$D(X_3)$ = {1, 2}
$D(X_4)$ = {1, 2, ~~3~~}
$D(X_5)$ = {4, 5}
$D(X_6)$ = {4, 5}
$D(N)$ = {~~1~~, 2}

A solution:

$\textsc{AtMostNValue}([2, 2, 2, 2, 4, 4, 2], 2)$

AtMostNValue

$\textsc{AtMostNValue}([X_1, \ldots, X_6], N)$

$D(X_1)$ = {1, 2, 3, 4, 5, 6}
$D(X_2)$ = {2, 4}
$D(X_3)$ = {1, 2}
$D(X_4)$ = {1, 2, 3}
$D(X_5)$ = {4, 5}
$D(X_6)$ = {4, 5}
$D(N)$ = {1, 2}

$$D(X_3) \cap D(X_5) = \emptyset$$

$D(X_1)$ = {1, 2, ~~3~~, 4, 5, ~~6~~}
$D(X_2)$ = {2, 4}
$D(X_3)$ = {1, 2}
$D(X_4)$ = {1, 2, ~~3~~}
$D(X_5)$ = {4, 5}
$D(X_6)$ = {4, 5}
$D(N)$ = {~~1~~, 2}

A solution:

$\textsc{AtMostNValue}([2, 2, 2, 2, 4, 4, 2], 2)$

AtMostNValue

$$\textsc{AtMostNValue}([X_1, \ldots, X_6], N)$$

$D(X_1)$ = {1, 2, 3, 4, 5, 6}
$D(X_2)$ = {2, 4}
$D(X_3)$ = {1, 2}
$D(X_4)$ = {1, 2, 3}
$D(X_5)$ = {4, 5}
$D(X_6)$ = {4, 5}
$D(N)$ = {1, 2}

$D(X_1)$ = {1, 2, ~~3~~, 4, 5, ~~6~~}
$D(X_2)$ = {2, 4}
$D(X_3)$ = {1, 2}
$D(X_4)$ = {1, 2, ~~3~~}
$D(X_5)$ = {4, 5}
$D(X_6)$ = {4, 5}
$D(N)$ = {~~1~~, 2}

- Enforcing GAC is NP-Hard
- [Hebrard et al, 2006], [Beldiceanu et al, 2001]:
  - Lower bound of N obtained by a greedy computing an independent set
  - Best lower bound obtained with a linear relaxation

AtMostNValue – Value representation

$\textsc{AtMostNValue}([X_1, \ldots, X_n], [Y_1, \ldots, Y_m], N)$

$Y_j \in \{0, 1\}$: value j occurs at least once

➡ express reasoning on mandatory values

Example:

$$D(X_1) = \{1, 2\}, D(X_2) = \{2, 3\}, D(N) = \{2\}$$
$$D(X_3) = \{2, 4\}$$

➡ propagate $Y_2 = 1$

AtMostNValue – Lagrangian relaxation

Primal

$$\text{Min } z = \sum_j y_j$$

$$\sum_{j \in D(X_i)} y_j \geq 1 \ \forall i$$

$$y_j \in \{0, 1\} \ \forall j$$

P

AtMostNValue – Lagrangian relaxation

Primal

$$\text{Min } z = \sum_j y_j$$

$$\sum_{j \in D(X_i)} y_j \geq 1 \ \forall i \quad (\lambda_i)$$

$$y_j \in \{0, 1\} \ \forall j$$

P

AtMostNValue – Lagrangian relaxation

Primal

$$\text{Min } z = \sum_j y_j$$

$$\sum_{j \in D(X_i)} y_j \geq 1 \;\; \forall i \quad (\lambda_i)$$

$$y_j \in \{0,1\} \;\; \forall j$$

P

For any $\lambda \geq 0$:

Lagrangian subproblem

$$\text{Min } w(\lambda) = \sum_j y_j + \sum_i \lambda_i (1 - \sum_{j \in D(X_i)} y_j)$$

$$y_j \in \{0,1\} \quad \forall j$$

$L(\lambda)$

AtMostNValue – Lagrangian relaxation

Primal

$$\text{Min } z = \sum_j y_j$$

$$\sum_{j \in D(X_i)} y_j \geq 1 \; \forall i \quad (\lambda_i)$$

$$y_j \in \{0,1\} \; \forall j$$

P

For any $\lambda \geq 0$:

Lagrangian subproblem

$$\text{Min } w(\lambda) = \sum_j y_j + \sum_i \underbrace{\lambda_i}_{\geq 0}\underbrace{(1 - \sum_{j \in D(X_i)} y_j)}_{\leq 0}$$

$$y_j \in \{0,1\} \quad \forall j$$

$L(\lambda)$

AtMostNValue – Lagrangian relaxation

Primal

$$\text{Min } z = \sum_j y_j$$
$$\sum_{j \in D(X_i)} y_j \geq 1 \ \forall i$$
$$y_j \in \{0,1\} \ \forall j$$

P

For any $\lambda \geq 0$: Lagrangian subproblem

$$\text{Min } w(\lambda) = \sum_j y_j + \sum_i \lambda_i (1 - \sum_{j \in D(X_i)} y_j)$$
$$= \sum_j \underbrace{(1 - \sum_{i | j \in D(X_i)} \lambda_i)}_{q_j} y_j + \sum_i \lambda_i$$
$$y_j \in \{0,1\} \ \forall j$$

$L(\lambda)$

- Complexity of Filtering is in O(nm) to check the sign of q_j
  - If q_j is negative, set y_j to 1
  - If q_j is positive, set y_j to 0

AtMostNValue – LR filtering

- q_j can be seen as a "Lagrangian reduced-cost":

(the increase/decrease of the objective value for setting y_j *to 1/0)*

- The filtering rules are simply :

$$w^* + (1 - y_j^*)q_j > \overline{N} \implies Y_j \neq 1$$ (forbidden values)

$$w^* - y_j^* q_j > \overline{N} \implies Y_j \neq 0$$ (mandatory values)

- Filtering can be done from any values of λ

AtMostNValue – LR filtering

Algorithm 1: Outline of the ATMOSTWVALUE LR-based propagator

for ($i \in [1, n]$) **do**
    $\lambda_i \leftarrow 0$ // Lagrangian multipliers initialization

$k \leftarrow 0$
while (**not** *subgradient*.`isStopCriterionMet()`) **do**
    `SolveSubProblem()`
    `FilterFromSubProblem()`
    subgradient.`UpdateMultipliers(`k`)`
    $k \leftarrow k + 1$

Solving the Lagrangian dual
And filtering along the way

Back to the Linear relaxation

$$\textsc{AtMostNValue}([X_1, \ldots, X_n], [Y_1, \ldots, Y_m], N)$$

minimize $z = \sum_{j=1}^{m} y_j$

s.t: $\sum_{j \in D(X_i)} y_j \geq 1 \quad \forall i \in [1, n]$

$$y_j \in \{0, 1\} \Rightarrow \begin{cases} y_j \leq 1 & \forall j \in [1, m] \\ y_j \geq 0 & \forall j \in [1, m] \end{cases}$$

- Assume all grounded variables have been removed from the formulation

Linear relaxation

$$\textsc{AtMostNValue}([X_1, \ldots, X_n], [Y_1, \ldots, Y_m], N)$$

minimize $z = \sum_{j=1}^{m} y_j$

s.t:

$$\sum_{j \in D(X_i)} y_j \geq 1 \qquad \forall i \in [1, n]$$

$$\sout{y_j \leq 1 \qquad \forall j \in [1, m]}$$

$$y_j \geq 0 \qquad \forall j \in [1, m]$$

- Assume all grounded variables have been removed from the formulation

Linear relaxation

$$\textsc{AtMostNValue}([X_1, \ldots, X_n], [Y_1, \ldots, Y_m], N)$$

minimize $z = \sum_{j=1}^{m} y_j$

s.t: $\sum_{j \in D(X_i)} y_j \geq 1 \qquad \forall i \in [1, n]$

$y_j \geq 0 \qquad \forall j \in [1, m]$

- Assume all grounded variables have been removed from the formulation

Linear relaxation

LP Primal

Min $z = \sum_{j=1}^{m} y_j$

$$\sum_{j \in D(X_i)} y_j \geq 1 \qquad \forall i$$

$$y_j \geq 0 \qquad \forall j$$

LP Dual

Max $w = \sum_{i=1}^{n} \alpha_i$

$$\sum_{i | j \in D(X_i)} \alpha_i \leq 1 \qquad \forall j$$

$$\alpha_i \geq 1 \qquad \forall i$$

Linear relaxation

LP Primal

Min $z = \sum_{j=1}^{m} y_j$

$$\sum_{j \in D(X_i)} y_j \geq 1 \qquad \forall i$$

$$y_j \geq 0 \qquad \forall j$$

LP Dual

Max $w = \sum_{i=1}^{n} \alpha_i$

$$\sum_{i | j \in D(X_i)} \alpha_i \leq 1 \qquad \forall j$$

$$\alpha_i \geq 1 \qquad \forall i$$

- Linear reduced cost: $r_j^* = 1 - \sum_{i | j \in D(X_i)} \alpha_i^*$
- "Lagrangian reduced-cost": $q_j = 1 - \sum_{i | j \in D(X_i)} \lambda_i$

$$w^* + (1 - y_j^*) q_j > \overline{N} \implies Y_j \neq 1$$ (forbidden values)

$$w^* - y_j^* q_j > \overline{N} \implies Y_j \neq 0$$ (mandatory values)

Linear relaxation

LP Primal

Min $z = \sum_{j=1}^{m} y_j$

$$\sum_{j \in D(X_i)} y_j \geq 1 \quad \forall i$$

$$y_j \geq 0 \quad \forall j$$

LP Dual

Max $w = \sum_{i=1}^{n} \alpha_i$

$$\sum_{i | j \in D(X_i)} \alpha_i \leq 1 \quad \forall j$$

$$\alpha_i \geq 1 \quad \forall i$$

- Linear reduced cost: $r_j^* = 1 - \sum_{i | j \in D(X_i)} \alpha_i^*$

$$w^* + (1 - y_j^*) r_j^* > \overline{N} \implies Y_j \neq 1$$ (forbidden values)

$$w^* - y_j^* r_j^* > \overline{N} \implies Y_j \neq 0$$ (mandatory values)

Linear relaxation

LP Primal

Min $z = \sum_{j=1}^{m} y_j$

$$\sum_{j \in D(X_i)} y_j \geq 1 \qquad \forall i$$

$$y_j \geq 0 \qquad \forall j$$

LP Dual

Max $w = \sum_{i=1}^{n} \alpha_i$

$$\sum_{i | j \in D(X_i)} \alpha_i \leq 1 \qquad \forall j$$

$$\alpha_i \geq 1 \qquad \forall i$$

- Linear reduced cost: $r_j^* = 1 - \sum_{i | j \in D(X_i)} \alpha_i^*$

$$w^* + (1 - y_j^*) r_j^* > \overline{N} \implies Y_j \neq 1$$ (forbidden values)

$$w^* - y_j^* r_j^* > \overline{N} \implies Y_j \neq 0$$ (mandatory values)

Complementary slackness $y_j^* r_j^* = 0$

Linear relaxation

LP Primal

Min $z = \sum_{j=1}^{m} y_j$

$$\sum_{j \in D(X_i)} y_j \geq 1 \quad \forall i$$

$$y_j \geq 0 \quad \forall j$$

LP Dual

Max $w = \sum_{i=1}^{n} \alpha_i$

$$\sum_{i | j \in D(X_i)} \alpha_i \leq 1 \quad \forall j$$

$$\alpha_i \geq 1 \quad \forall i$$

- Linear reduced cost: $r_j^* = 1 - \sum_{i | j \in D(X_i)} \alpha_i^*$

$$w^* + (1 - y_j^*) r_j^* > \overline{N} \implies Y_j \neq 1$$ (forbidden values)

$$w^* - y_j^* r_j^* > \overline{N} \implies Y_j \neq 0$$ (mandatory values)

Complementary slackness $y_j^* r_j^* = 0$

Linear relaxation

LP Primal

Min $z = \sum_{j=1}^{m} y_j$

$$\sum_{j \in D(X_i)} y_j \geq 1 \qquad \forall i$$

$$y_j \geq 0 \qquad \forall j$$

LP Dual

Max $w = \sum_{i=1}^{n} \alpha_i$

$$\sum_{i | j \in D(X_i)} \alpha_i \leq 1 \qquad \forall j$$

$$\alpha_i \geq 1 \qquad \forall i$$

- Linear reduced cost: $r_j^* = 1 - \sum_{i | j \in D(X_i)} \alpha_i^*$

In the LP case, reduced-cost based filtering
is restricted to forbidden values

$$\forall j \in [1, m] \text{ s.t } y_j^* = 0 \qquad z^* + r_j^* > \overline{N} \implies Y_j \neq 1$$

Lagrangian relaxation - Example

$$\begin{aligned}
&w = y_1 + y_2 + y_3 + y_4 + y_5 \\
&y_1 + y_2 \geq 1 \\
&y_2 + y_3 \geq 1 \\
&y_4 + y_5 \geq 1 \\
&y_j \geq 0 \qquad \forall j \in [1,5]
\end{aligned}$$

$$\begin{aligned}
w(\lambda_1, \lambda_2, \lambda_3) = \quad & y_1(1-\lambda_1) + y_2(1-\lambda_1-\lambda_2) \\
& + y_3(1-\lambda_2) + y_4(1-\lambda_3) + y_5(1-\lambda_3) \\
& +(\lambda_1+\lambda_2+\lambda_3) \\
& y_j \in \{0,1\} \qquad \forall j \in [1,5]
\end{aligned}$$

Lagrangian relaxation - Example

$$
\begin{aligned}
& w = y_1 + y_2 + y_3 + y_4 + y_5 \\
& y_1 + y_2 \geq 1 \\
& y_2 + y_3 \geq 1 \\
& y_4 + y_5 \geq 1 \\
& y_j \geq 0 \qquad \forall j \in [1,5]
\end{aligned}
$$

$$
\begin{aligned}
w(\lambda_1, \lambda_2, \lambda_3) = \quad & y_1(1 - \lambda_1) + y_2(1 - \lambda_1 - \lambda_2) \\
& + y_3(1 - \lambda_2) + y_4(1 - \lambda_3) + y_5(1 - \lambda_3) \\
& + (\lambda_1 + \lambda_2 + \lambda_3) \\
& y_j \in \{0, 1\} \qquad \forall j \in [1,5]
\end{aligned}
$$

λ_1	λ_2	λ_3
0.8	0.8	0.8

Lagrangian relaxation - Example

$$
\begin{aligned}
w &= y_1 + y_2 + y_3 + y_4 + y_5 \\
y_1 + y_2 &\geq 1 \\
y_2 + y_3 &\geq 1 \\
y_4 + y_5 &\geq 1 \\
y_j &\geq 0 \qquad \forall j \in [1,5]
\end{aligned}
$$

$$
\begin{aligned}
w(\lambda_1, \lambda_2, \lambda_3) = \quad & y_1(1-\lambda_1) + y_2(1-\lambda_1-\lambda_2) \\
& + y_3(1-\lambda_2) + y_4(1-\lambda_3) + y_5(1-\lambda_3) \\
& +(\lambda_1 + \lambda_2 + \lambda_3) \\
& y_j \in \{0,1\} \qquad \forall j \in [1,5]
\end{aligned}
$$

λ_1	λ_2	λ_3
0.8	0.8	0.8

q_1	q_2	q_3	q_4	q_5
0.2	-0.6	0.2	0.2	0.2

Lagrangian relaxation - Example

$$
\begin{aligned}
& w = y_1 + y_2 + y_3 + y_4 + y_5 \\
& y_1 + y_2 \geq 1 \\
& y_2 + y_3 \geq 1 \\
& y_4 + y_5 \geq 1 \\
& y_j \geq 0 \qquad \forall j \in [1,5]
\end{aligned}
$$

$$
\begin{aligned}
w(\lambda_1, \lambda_2, \lambda_3) = \quad & y_1(1-\lambda_1) + y_2(1-\lambda_1-\lambda_2) \\
& + y_3(1-\lambda_2) + y_4(1-\lambda_3) + y_5(1-\lambda_3) \\
& + (\lambda_1 + \lambda_2 + \lambda_3) \\
& y_j \in \{0,1\} \qquad \forall j \in [1,5]
\end{aligned}
$$

λ_1	λ_2	λ_3
0.8	0.8	0.8

q_1	q_2	q_3	q_4	q_5
0.2	-0.6	0.2	0.2	0.2

y_1^*	y_2^*	y_3^*	y_4^*	y_5^*
0	1	0	0	0

Lagrangian relaxation - Example

$$
\begin{aligned}
& w = y_1 + y_2 + y_3 + y_4 + y_5 \\
& y_1 + y_2 \geq 1 \\
& y_2 + y_3 \geq 1 \\
& y_4 + y_5 \geq 1 \\
& y_j \geq 0 \qquad \forall j \in [1,5]
\end{aligned}
$$

$$
\begin{aligned}
w(\lambda_1, \lambda_2, \lambda_3) = \quad & y_1(1-\lambda_1) + y_2(1-\lambda_1-\lambda_2) \\
& + y_3(1-\lambda_2) + y_4(1-\lambda_3) + y_5(1-\lambda_3) \\
& +(\lambda_1 + \lambda_2 + \lambda_3) \\
& y_j \in \{0,1\} \qquad \forall j \in [1,5]
\end{aligned}
$$

λ_1	λ_2	λ_3
0.8	0.8	0.8

q_1	q_2	q_3	q_4	q_5
0.2	-0.6	0.2	0.2	0.2

y_1^*	y_2^*	y_3^*	y_4^*	y_5^*
0	1	0	0	0

$w^*(\lambda) = 1.8$

Lagrangian relaxation - Example

$$
\begin{aligned}
& w = y_1 + y_2 + y_3 + y_4 + y_5 \\
& y_1 + y_2 \geq 1 \\
& y_2 + y_3 \geq 1 \\
& y_4 + y_5 \geq 1 \\
& y_j \geq 0 \qquad \forall j \in [1,5]
\end{aligned}
$$

$$
\begin{aligned}
w(\lambda_1, \lambda_2, \lambda_3) = \quad & y_1(1-\lambda_1) + y_2(1-\lambda_1-\lambda_2) \\
& + y_3(1-\lambda_2) + y_4(1-\lambda_3) + y_5(1-\lambda_3) \\
& +(\lambda_1+\lambda_2+\lambda_3) \\
& y_j \in \{0,1\} \qquad \forall j \in [1,5]
\end{aligned}
$$

λ_1	λ_2	λ_3
0.8	0.8	0.8

q_1	q_2	q_3	q_4	q_5
0.2	-0.6	0.2	0.2	0.2

y_1^*	y_2^*	y_3^*	y_4^*	y_5^*
0	1	0	0	0

$$w^*(\lambda) = 1.8$$

$$w^*(\lambda) - q_2 = 2.4 > \overline{N} \qquad \Longrightarrow Y_2 \neq 0$$

Lagrangian relaxation - Example

$$
\begin{aligned}
&w = y_1 + y_2 + y_3 + y_4 + y_5 \\
&y_1 + y_2 \geq 1 \\
&y_2 + y_3 \geq 1 \\
&y_4 + y_5 \geq 1 \\
&y_j \geq 0 \qquad \forall j \in [1,5]
\end{aligned}
$$

$$
\begin{aligned}
w(\lambda_1, \lambda_2, \lambda_3) = \quad & y_1(1-\lambda_1) + y_2(1-\lambda_1-\lambda_2) \\
& + y_3(1-\lambda_2) + y_4(1-\lambda_3) + y_5(1-\lambda_3) \\
& + (\lambda_1 + \lambda_2 + \lambda_3) \\
& y_j \in \{0,1\} \qquad \forall j \in [1,5]
\end{aligned}
$$

λ_1	λ_2	λ_3
0.8	0.8	0.8

q_1	q_2	q_3	q_4	q_5
0.2	-0.6	0.2	0.2	0.2

y_1^*	y_2^*	y_3^*	y_4^*	y_5^*
0	1	0	0	0

$w^*(\lambda) = 1.8$

$$w^*(\lambda) - q_2 = 2.4 > \overline{N} \qquad \Longrightarrow Y_2 \neq 0$$

λ_1	λ_2	λ_3
0.8	0.8	0.9

q_1	q_2	q_3	q_4	q_5
0.2	-0.6	0.2	0.1	0.1

y_1^*	y_2^*	y_3^*	y_4^*	y_5^*
0	1	0	0	0

$w^*(\lambda) = 1.9$

$$w^*(\lambda) + q_1 = 2.1 > \overline{N} \qquad \Longrightarrow Y_1 \neq 1$$

$$w^*(\lambda) + q_3 = 2.1 > \overline{N} \qquad \Longrightarrow Y_3 \neq 1$$

Lagrangian relaxation - Example

$$
\begin{aligned}
w &= y_1 + y_2 + y_3 + y_4 + y_5 \\
& y_1 + y_2 \geq 1 \\
& y_2 + y_3 \geq 1 \\
& y_4 + y_5 \geq 1 \\
& y_j \geq 0 \qquad \forall j \in [1,5]
\end{aligned}
$$

$$
\begin{aligned}
w(\lambda_1, \lambda_2, \lambda_3) = \quad & y_1(1-\lambda_1) + y_2(1-\lambda_1-\lambda_2) \\
& + y_3(1-\lambda_2) + y_4(1-\lambda_3) + y_5(1-\lambda_3) \\
& + (\lambda_1 + \lambda_2 + \lambda_3) \\
& y_j \in \{0,1\} \qquad \forall j \in [1,5]
\end{aligned}
$$

λ_1	λ_2	λ_3
0.8	0.8	0.8

q_1	q_2	q_3	q_4	q_5
0.2	-0.6	0.2	0.2	0.2

y_1^*	y_2^*	y_3^*	y_4^*	y_5^*
0	1	0	0	0

$w^*(\lambda) = 1.8$

$$w^*(\lambda) - q_2 = 2.4 > \overline{N} \qquad \Longrightarrow \quad Y_2 \neq 0$$

λ_1	λ_2	λ_3
0.8	0.8	0.9

q_1	q_2	q_3	q_4	q_5
0.2	-0.6	0.2	0.1	0.1

y_1^*	y_2^*	y_3^*	y_4^*	y_5^*
0	1	0	0	0

$w^*(\lambda) = 1.9$

$$w^*(\lambda) + q_1 = 2.1 > \overline{N} \qquad \Longrightarrow \quad Y_1 \neq 1$$
$$w^*(\lambda) + q_3 = 2.1 > \overline{N} \qquad \Longrightarrow \quad Y_3 \neq 1$$

Optimal LP multipliers

λ_1	λ_2	λ_3
0	1	1

q_1	q_2	q_3	q_4	q_5
1	0	0	0	0

y_1^*	y_2^*	y_3^*	y_4^*	y_5^*
0	0	0	0	0

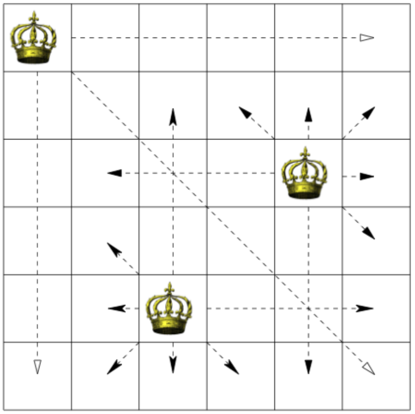
$w^*(\lambda) = 2$

$$w^*(\lambda) + q_1 = 3 > \overline{N} \qquad \Longrightarrow \quad Y_1 \neq 1$$

Subgradient - Example

- Let's consider a simple step-size rule: $\mu^k = 1 \times (0.8)^k$
- The first four iterations of the subgradient:

k	μ^k	λ_1	λ_2	λ_3	$w^*(\lambda_1, \lambda_2, \lambda_3)$	Filtering detected
0	-	0	0	0	0	
1	0.8	0.8	0.8	0.8	1.8	$Y_2 \neq 0$
2	0.64	0.8	0.8	1.44	1.56	
3	0.512	0.8	0.8	0.928	1.9279	$Y_1 \neq 1, Y_3 \neq 1$
4	0.4096	0.8	0.8	1.3376	1.6624	
...	...	...	...	...	...	
		0.8	0.8	1.001	1.999	

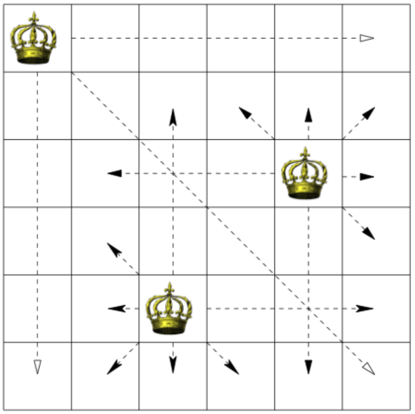


Dominating queens

Slow subgradient
(better convergence)

			CP+NVAL		CP+NVAL(HARM)		
n	v	Feas	Cpu(s)	Fail	Cpu(s)	Fails	Iter (k)
6	3	SAT	0,162	15	0,192	7	25
7	4	SAT	0,136	353	0,504	52	128
8	5	SAT	0,663	2275	1,039	86	210
8	4	UNSAT	155,649	1074789	23,29	1796	2767
9	5	SAT	186,756	920666	14,654	862	1426

- The filtering of LR outperforms the state-of-the art propagator



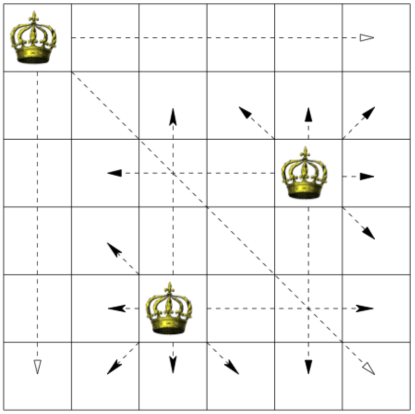
Dominating queens

Slow subgradient (better convergence)

LP + Reduced-costs

n	v	Feas	CP+NVAL Cpu(s)	CP+NVAL Fail	CP+NVAL(HARM) Cpu(s)	CP+NVAL(HARM) Fails	CP+NVAL(HARM) Iter (k)	CP+NVAL(LP) Cpu(s)	CP+NVAL(LP) Fails
6	3	SAT	0,162	15	0,192	7	25	0,513	9
7	4	SAT	0,136	353	0,504	52	128	4,399	93
8	5	SAT	0,663	2275	1,039	86	210	18,825	259
8	4	UNSAT	155,649	1074789	23,29	1796	2767	222,755	2596
9	5	SAT	186,756	920666	14,654	862	1426	153,744	884

- The filtering of LR outperforms the state-of-the art propagator
- LR can be fast (faster than LP)
- LR can filter more than LP



Dominating queens

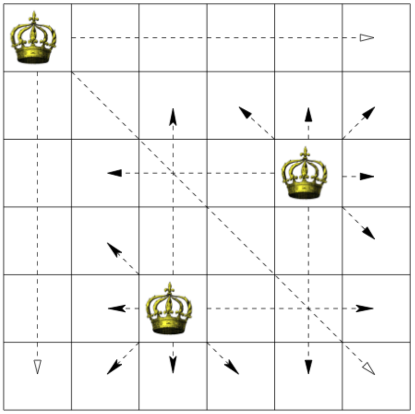
Slow subgradient (better convergence) → CP+NVAL(HARM)

Fast subgradient → CP+NVAL(NEWTON)

LP + Reduced-costs → CP+NVAL(LP)

			CP+NVAL		CP+NVAL(HARM)			CP+NVAL(NEWTON)			CP+NVAL(LP)	
n	v	Feas	Cpu(s)	Fail	Cpu(s)	Fails	Iter (k)	Cpu(s)	Fails	Iter (k)	Cpu(s)	Fails
6	3	SAT	0,162	15	0,192	7	25	0,121	9	3	0,513	9
7	4	SAT	0,136	353	0,504	52	128	0,085	31	9	4,399	93
8	5	SAT	0,663	2275	1,039	86	210	0,236	79	17	18,825	259
8	4	UNSAT	155,649	1074789	23,29	1796	2767	5,073	6260	299	222,755	2596
9	5	SAT	186,756	920666	14,654	862	1426	3,423	1593	157	153,744	884

- The filtering of LR outperforms the state-of-the art propagator
- LR can be fast (faster than LP)
- LR can filter more than LP



Dominating queens

Slow subgradient (better convergence)

Fast subgradient

LP + Reduced-costs

			CP+NVAL		CP+NVAL(HARM)			CP+NVAL(NEWTON)			CP+NVAL(KELLEY)			CP+NVAL(LP)	
n	v	Feas	Cpu(s)	Fail	Cpu(s)	Fails	Iter (k)	Cpu(s)	Fails	Iter (k)	Cpu(s)	Fails	Iter (k)	Cpu(s)	Fails
6	3	SAT	0,162	15	0,192	7	25	0,121	9	3	0,5	6	0,4	0,513	9
7	4	SAT	0,136	353	0,504	52	128	0,085	31	9	1,1	45	3	4,399	93
8	5	SAT	0,663	2275	1,039	86	210	0,236	79	17	2,62	80	6	18,825	259
8	4	UNSAT	155,649	1074789	23,29	1796	2767	5,073	6260	299	181,3	1400	295	222,755	2596
9	5	SAT	186,756	920666	14,654	862	1426	3,423	1593	157	107,1	646	151	153,744	884

- The filtering of LR outperforms the state-of-the art propagator
- LR can be fast (faster than LP)
- LR can filter more than LP

4- Use of LR by the CP community

- LR for domain's filtering
- LR for filtering global constraints
- Lagrangian decomposition (LD) for domain's filtering
- Applications using CP+LR

Lagrangian relaxation for domain filtering

[Focacci, F., Lodi, A., Milano, M.: **Cost-Based Domain Filtering**. CP 1999]

[Focacci, F., Lodi, A., Milano; M.:
Cutting Planes in Constraint Programming: An Hybrid Approach, CP 2000]

[Sellmann. M. **Theoretical foundations of CP-based lagrangian relaxation**. CP 2004.]

[Khemmoudj, M.O., Bennaceur, H. , Nagih, A.:
Combining Arc-Consistency and Dual Lagrangean Relaxation for Filtering CSPs, CP 2005]

...

Lagrangian relaxation for domain filtering

[Focacci, F., Lodi, A., Milano, M.: **Cost-Based Domain Filtering**. CP 1999]

[Focacci, F., Lodi, A., Milano; M.:
Cutting Planes in Constraint Programming: An Hybrid Approach, CP 2000]

[Sellmann. M. **Theoretical foundations of CP-based lagrangian relaxation**. CP 2004.]

[Khemmoudj, M.O., Bennaceur, H. , Nagih, A.:
Combining Arc-Consistency and Dual Lagrangean Relaxation for Filtering CSPs, CP 2005]

…

[Gleixner, A., Weltge, S.:
Learning and Propagating Lagrangian Variable Bounds for Mixed-Integer Nonlinear Programming. CPAIOR 2013]

[Bergman, D., Cire, A.A., van Hoeve, W.J.:
Lagrangian Bounds from Decision Diagrams. Constraints 2015]

Lagrangian based filtering for NP-Hard global constraints

Constraint	Lagrangian Subproblem	Examples of applications	References
Multi-cost-regular	Shortest/Longest Path	Personnel Scheduling	[Menana et al, 2009]
Weighted-circuit	1-Tree (Spanning Tree) Shortest path of n-arcs	Traveling Salesman Problem (TSP) Traveling Purchaser Problem (TPP) Traveling Tournament (TTP) TSP with time-windows	[Caseau et al, 1997] [Benoist et al, 2001] [Benchimol et al, 2012] [Fages et al, 2012] [Ducomman et al, 2015]
AtMostNValue / AtMostWValue	Inspection	Traveling Purchaser Problem Facility location Warehouse location	[Cambazard et al, 2012] [Cambazard et al, 2015]
Bin-Packing with usage costs	Knapsack	Energy optimization in data-centers	
Shortest Path in DAG with resource constraints	Shortest Path	Multileaf collimator sequencing	[Sellmann, 2005] [Cambazard et al, 2010]
Table constraint (extensional CSPs)	Inspection	Extensional CSP	[Khemmoudj, 2005]

Generalizing Lagrangian Relaxation

- Using CP as a framework to generalize Lagrangian relaxation

[Fontaine, D., Michel, L. , Van Hentenryck, P.:
Constraint-Based Lagrangian Relaxation. CP 2014]

Lagrangian decomposition

- Create copies of the decision variables for each constraint set
- Relax the equality constraints between the copies

[Guignard, M.; Kim, S., **Lagrangean decomposition: A model yielding stronger Lagrangean bounds**, Mathematical Programming (1987) 39, 215]

Lagrangian decomposition

- Create copies of the decision variables for each constraint set
- Relax the equality constraints between the copies

[Guignard, M.; Kim, S., **Lagrangean decomposition: A model yielding stronger Lagrangean bounds**, Mathematical Programming (1987) 39, 215]

$$\begin{aligned} &\text{Max } 2x_1 + 3x_2 + 4x_3 \\ &\quad 12x_1 + 19x_2 + 30x_3 \leq 46 \\ &\quad 49x_1 + 40x_2 + 31x_3 \leq 76 \\ &\quad x_1, x_2, x_3 \in \{0, 1\} \end{aligned}$$

Lagrangian decomposition

- Create copies of the decision variables for each constraint set
- Relax the equality constraints between the copies

[Guignard and al, 1987]

$$\text{Max } 2x_1 + 3x_2 + 4x_3$$
$$12x_1 + 19x_2 + 30x_3 \leq 46$$
$$49x_1 + 40x_2 + 31x_3 \leq 76$$
$$x_1, x_2, x_3 \in \{0, 1\}$$

LR1

$$\text{Max } 2x_1 + 3x_2 + 4x_3 + \lambda(76 - 49x_1 - 40x_2 - 31x_3)$$
$$12x_1 + 19x_2 + 30x_3 \leq 46$$
$$x_1, x_2, x_3 \in \{0, 1\} \qquad \lambda \geq 0$$

LR2

$$\text{Max } 2x_1 + 3x_2 + 4x_3 + \lambda(46 - 12x_1 - 19x_2 - 30x_3)$$
$$49x_1 + 40x_2 + 31x_3 \leq 76$$
$$x_1, x_2, x_3 \in \{0, 1\} \qquad \lambda \geq 0$$

Lagrangian decomposition

- Create copies of the decision variables for each constraint set
- Relax the equality constraints between the copies

[Guignard and al, 1987]

$$\begin{aligned} &\text{Max } 2x_1 + 3x_2 + 4x_3 \\ &12x_1 + 19x_2 + 30x_3 \leq 46 \\ &49x_1 + 40x_2 + 31x_3 \leq 76 \\ &x_1, x_2, x_3 \in \{0, 1\} \end{aligned}$$

relax $x = y$

LR1

$$\begin{aligned} &\text{Max } 2x_1 + 3x_2 + 4x_3 \\ &\quad +\lambda(76 - 49x_1 - 40x_2 - 31x_3) \\ &12x_1 + 19x_2 + 30x_3 \leq 46 \\ &x_1, x_2, x_3 \in \{0, 1\} \qquad \lambda \geq 0 \end{aligned}$$

LR2

$$\begin{aligned} &\text{Max } 2x_1 + 3x_2 + 4x_3 \\ &\quad +\lambda(46 - 12x_1 - 19x_2 - 30x_3) \\ &49x_1 + 40x_2 + 31x_3 \leq 76 \\ &x_1, x_2, x_3 \in \{0, 1\} \qquad \lambda \geq 0 \end{aligned}$$

LD

$$\begin{aligned} &\text{Max } 2x_1 + 3x_2 + 4x_3 \\ &\quad +\lambda_1(x_1 - y_1) \\ &\quad +\lambda_2(x_2 - y_2) \\ &\quad +\lambda_3(x_3 - y_3) \\ &12x_1 + 19x_2 + 30x_3 \leq 46 \\ &49y_1 + 40y_2 + 31y_3 \leq 76 \\ &x_1, x_2, x_3 \in \{0, 1\} \\ &y_1, y_2, y_3 \in \{0, 1\} \qquad \lambda \in \mathbb{R} \end{aligned}$$

Lagrangian decomposition

[Guignard and al, 1987]

LR1

$$\text{Max } 2x_1 + 3x_2 + 4x_3 + \lambda(76 - 49x_1 - 40x_2 - 31x_3)$$
$$12x_1 + 19x_2 + 30x_3 \leq 46$$
$$x_1, x_2, x_3 \in \{0,1\}$$

LR2

$$\text{Max } 2x_1 + 3x_2 + 4x_3 + \lambda(46 - 12x_1 - 19x_2 - 30x_3)$$
$$49x_1 + 40x_2 + 31x_3 \leq 76$$
$$x_1, x_2, x_3 \in \{0,1\}$$

LD

$$\text{Max } 2x_1 + 3x_2 + 4x_3 + \lambda_1(x_1 - y_1) + \lambda_2(x_2 - y_2) + \lambda_3(x_3 - y_3)$$
$$12x_1 + 19x_2 + 30x_3 \leq 46$$
$$49y_1 + 40y_2 + 31y_3 \leq 76$$
$$x_1, x_2, x_3 \in \{0,1\}$$
$$y_1, y_2, y_3 \in \{0,1\}$$

LP* = 6.68

LR2* = 6.6

LR1* = 5.84

LD* = 4.5

z* = 4

- Lagrangian decomposition is at least as good as Lagrangian relaxation
- But often involves a lot of multipliers (not well suited for subgradient techniques)

Lagrangian decomposition

- Create copies of the decision variables for each constraint set
- Relax the equality constraints between the copies
- Two CP papers:

[Bergman, D., Cire, A., van Hoeve, W.J.
Improved Constraint Propagation via Lagrangian Decomposition, CP 2015]
[Ha, M.H., Quimper, C.G., Rousseau L.M.
General bounding mechanism for Constraint Programs, CP 2015]

- Improve the lower bound provided by the constraint solver and perform global filtering

- Existing (global) constraints provide efficient solver for the Lagrangian subproblems ?

Applications of CP+LR

- Traveling Salesman Problem [Caseau et al, 1997][Benchimol et al, 2012][Fages et al, 2012]
- Traveling Salesman Problem with time-windows [Focacci et al, CP 2000][Bergman et al, CPAIOR 2015][Ducomman et al, CP 2015]
- Traveling Tournament Problem [Benoist et al, CP 2001]
- Capacitated Network Design [Sellmann et al, 2002]
- Automated Recording Problem [Sellmann et al, 2003]
- Network design [Cronholm et al, CP 2004]
- Resource Constrained Shortest Path Problem [Gellermann et al, CPAIOR 2005]
- Personnel Scheduling [Menana et al, CPAIOR 2009]
- Multileaf sequencing collimator [Cambazard et al, CPAIOR 2009]
- Parallel Machine Scheduling with Additional Resources [Edis et al, CPAIOR 2011]
- Resource Constrained Shortest Path with a Super Additive Objective Function [Gualandi, CP 2012]
- Traveling Purchaser Problem [Cambazard et al, CP 2012]
- Golomb rulers [Slusky et al, CPAIOR 2013]
- Empirical Model Learning [Lombardi et al, CP 2013]
- Net Present Value of Resource-Constrained Projects [Gu et al, CPAIOR 2013]

Conclusion LR and CP

- The roles of LR within CP:

- Allow CP solvers to provide lower bounds
- Provide a simple framework performing cost based filtering for NP-hard global constraints:
  - Bin-Packing
  - Disjunctive
  - …
- Relatively generic approach to allow cooperation between constraints beyond the domains

- CP solvers as a framework to provide “automated” Lagrangian relaxation ?

Conclusion

- Allow CP solvers to provide lower bounds
- Provide a simple framework performing cost based filtering for NP-hard global constraints:
  - Bin-Packing
  - Disjunctive
  - Global Sequencing Constraint

> "For the future, it remains to be seen whether Lagrangian relaxation will continue to exist as a technique that requires a significant ad hoc development effort or whether the essential building blocks of Lagrangian relaxation will find their way into user-friendly mathematical programming codes such as LINDO or IFPS OPTIMUM.
> ...
> It would then be left to the analyst to decide which constraints to dualize and to specify which of the possible Lagrangian problem algorithms to use. (Marshall L. Fisher, 1985)"