Universidad de los Andes Colombia

Solving Hard Shortest Path Problems with the Pulse Framework

Lecture 2: Intuition and extensions

Andrés Medaglia, Ph.D.

(amedagli@uniandes.edu.co)

Joint work with:

L. Lozano, Ph.D. (U. of Cincinnati); D. Duque, Ph.D.(c) (Northwestern U.);
N. Cabrera, M.Sc. & D. Yamín (U. de los Andes); M. Bolívar, M.Sc. (Glovo)

Departamento de Ingeniería Industrial
Centro para la Optimización y Probabilidad Aplicada
Universidad de los Andes (Colombia)

Universidad de la República; Montevideo (Uruguay), Marzo 9-13

Agenda

- Part I: fundamentals
- Part II: intuition
- Part III: extensions
- Part IV: applications
- Part V: perspectives

Agenda

- Part I: fundamentals

- Part II: intuition
  - **The pulse algorithm**
  - Constrained Shortest Path Problem (CSP)

- Part III: extensions

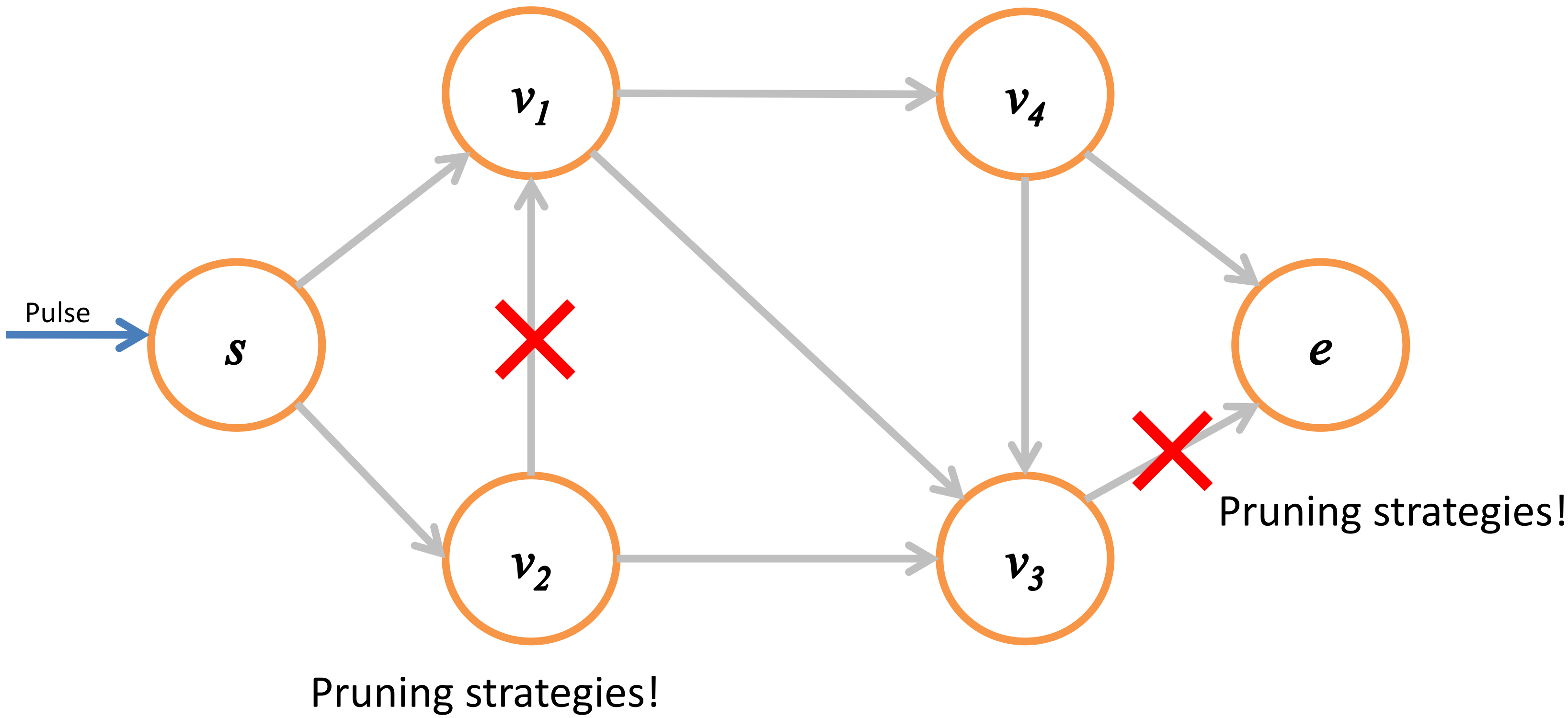
- Part IV: applications

- Part V: perspectives

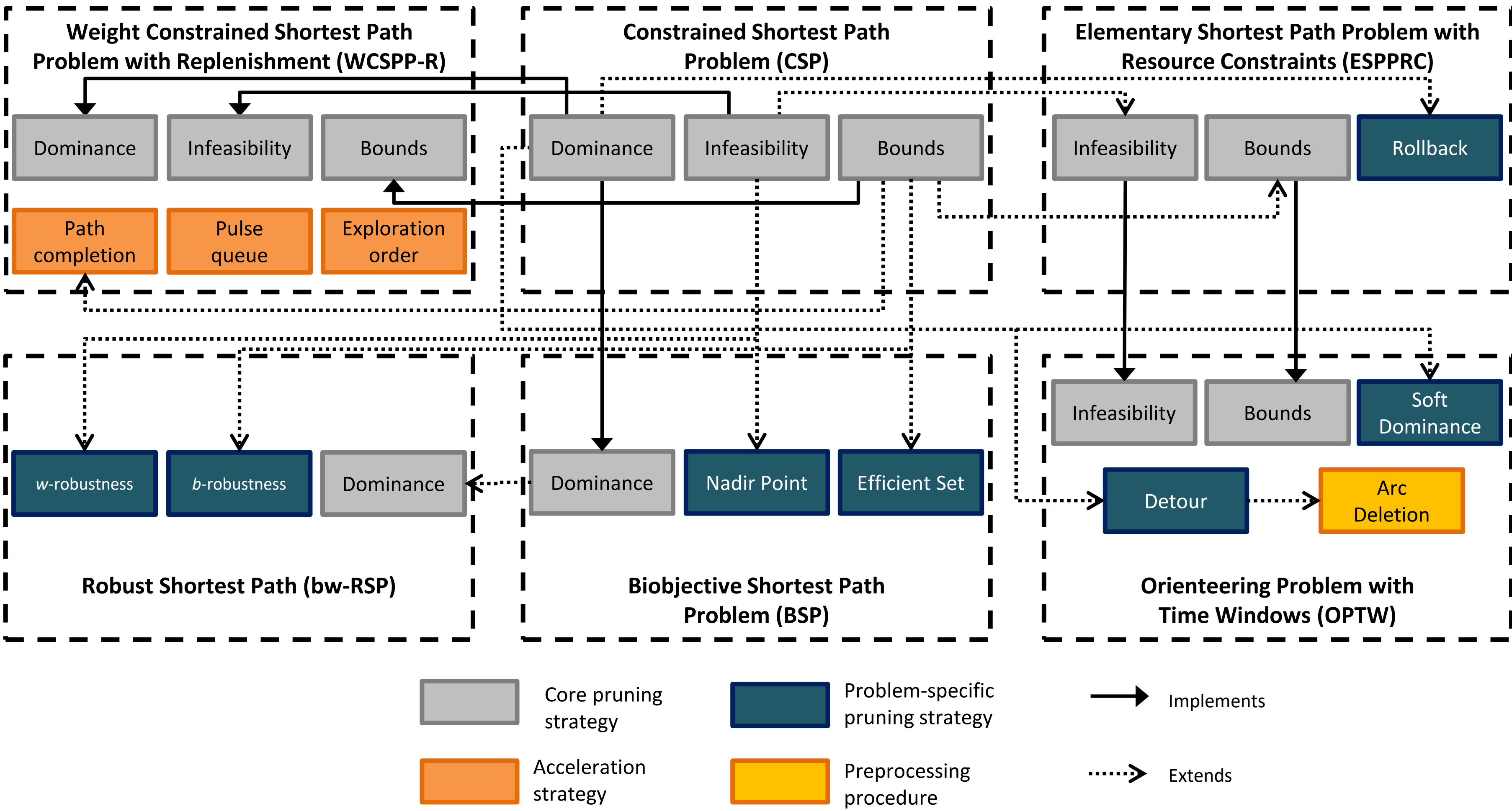
The pulse framework

Algorithm overview

save: $\{s, v_1, v_3, e\}$



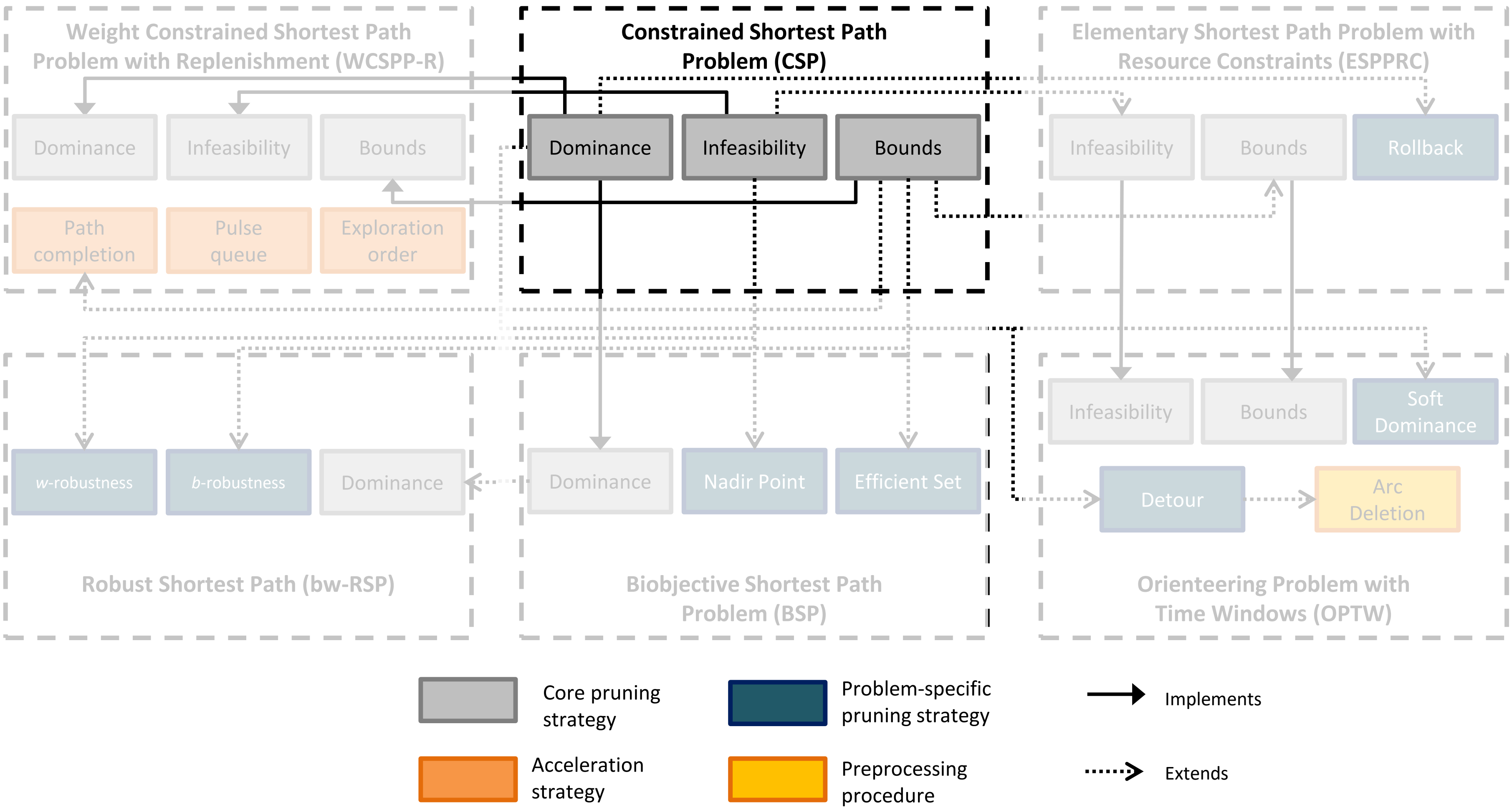
Pulse Algorithm for Hard Shortest Path Problems



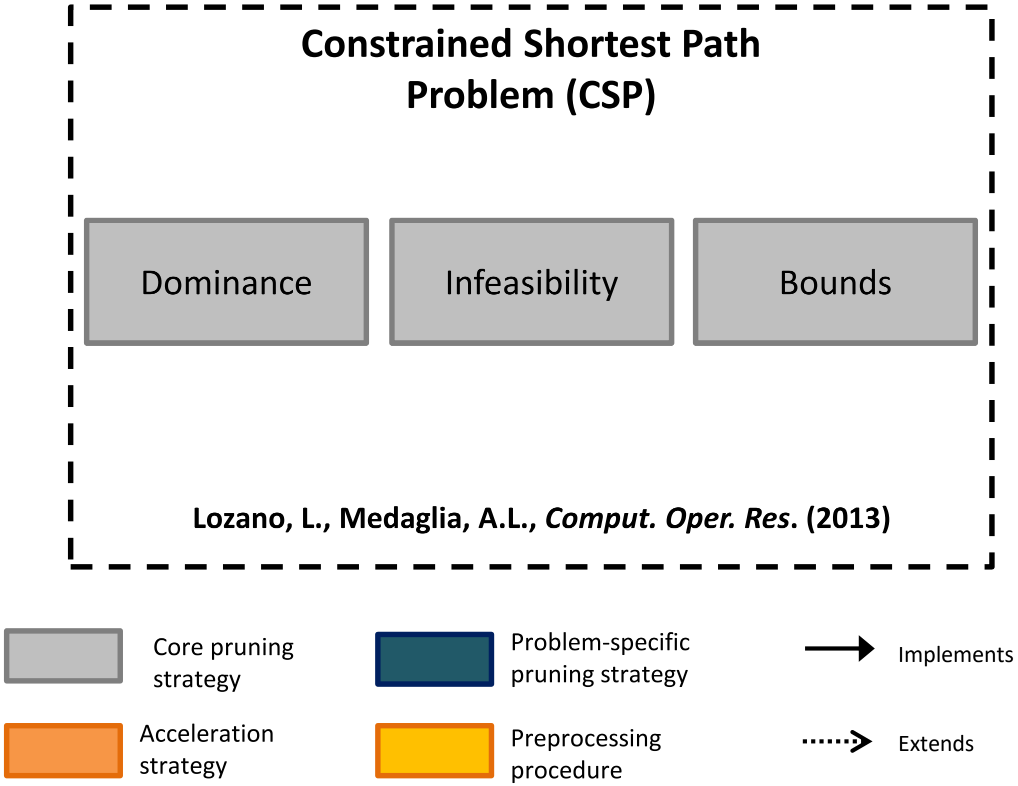
Agenda

- Part I: fundamentals
- Part II: intuition
    - The pulse algorithm
    - **Constrained Shortest Path Problem (CSP)**
- Part III: extensions
- Part IV: applications
- Part V: perspectives

Pulse Algorithm for Hard Shortest Path Problems



Constrained Shortest Path Problem (CSP)



- Dumitrescu & Boland (2003)

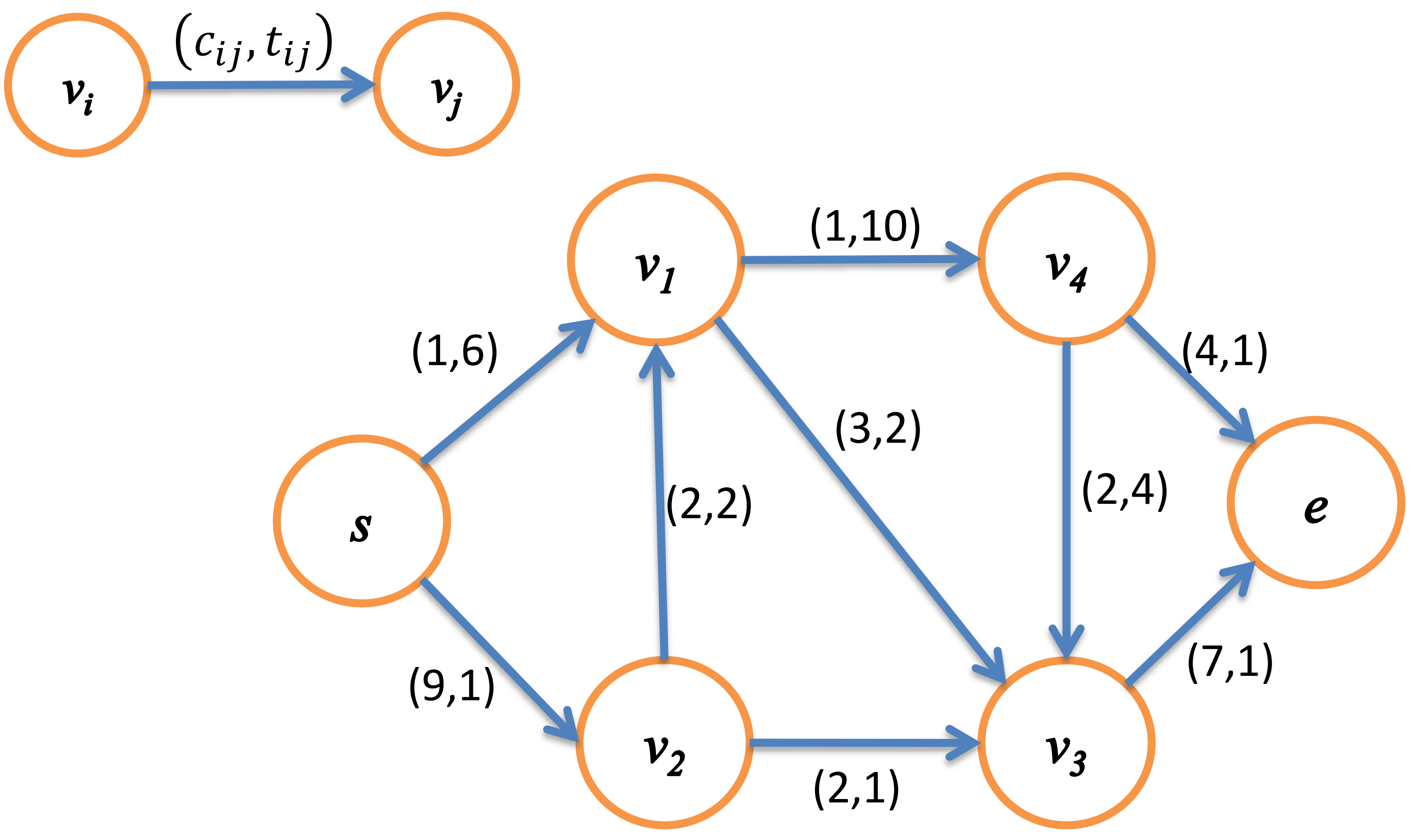
Constrained Shortest Path Problem (CSP)

Problem statement

- The CSP is defined by:
  - Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$
    - $\mathcal{N} = \{v_1, \dots, v_i, \dots, v_n\}$
    - $\mathcal{A} = \{(i,j) | v_i \in \mathcal{N}, v_j \in \mathcal{N},\ i \neq j\}$
  - Find a minimum cost path starting at node v_s and ending at node v_e
  - Nonnegative weights c_{ij} and t_{ij} are the cost and travel time of traversing arc $(i,j) \in \mathcal{A}$
  - Time constraint T

Constrained Shortest Path Problem (CSP)

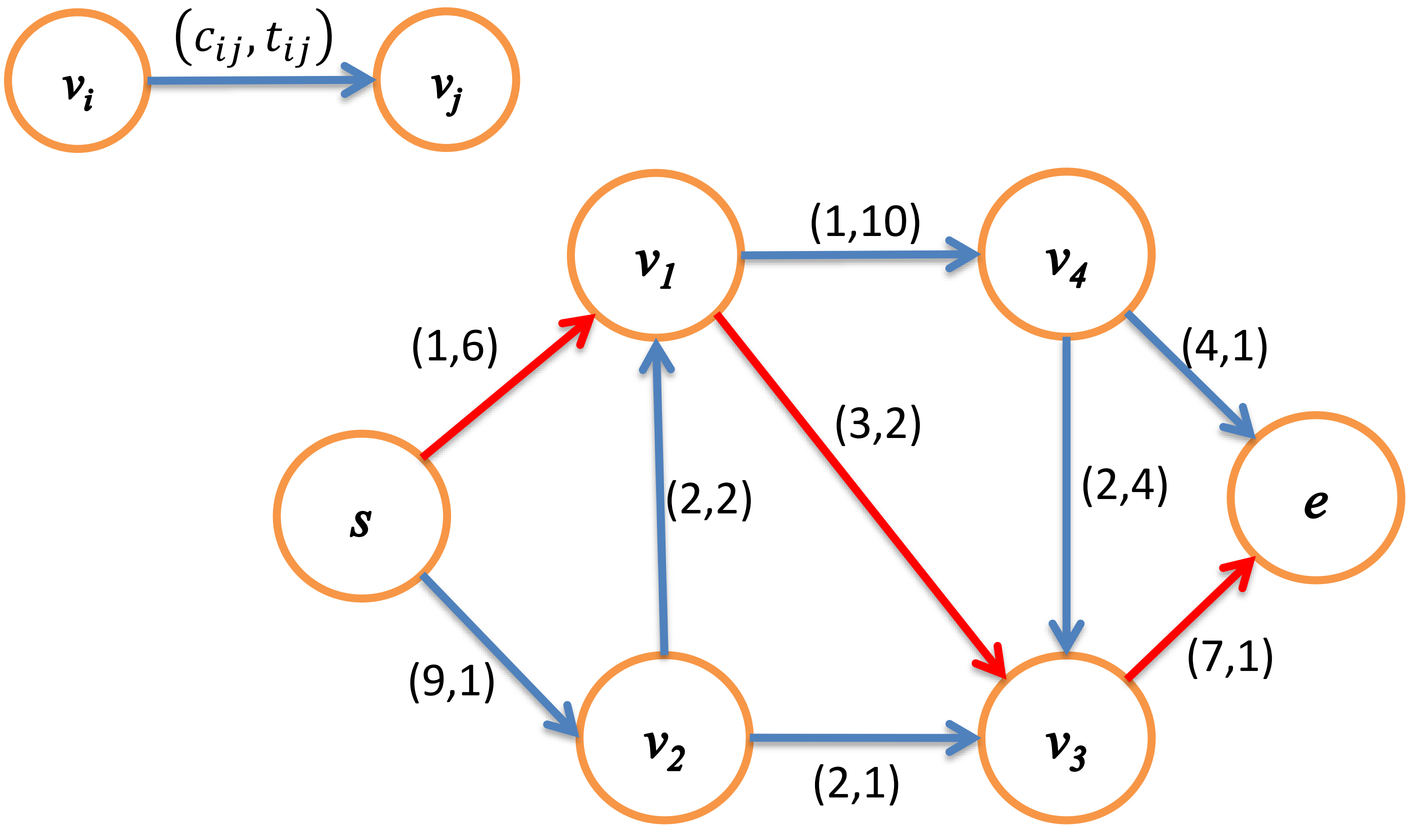
Problem statement



$T = 14$

Constrained Shortest Path Problem (CSP)

Problem statement

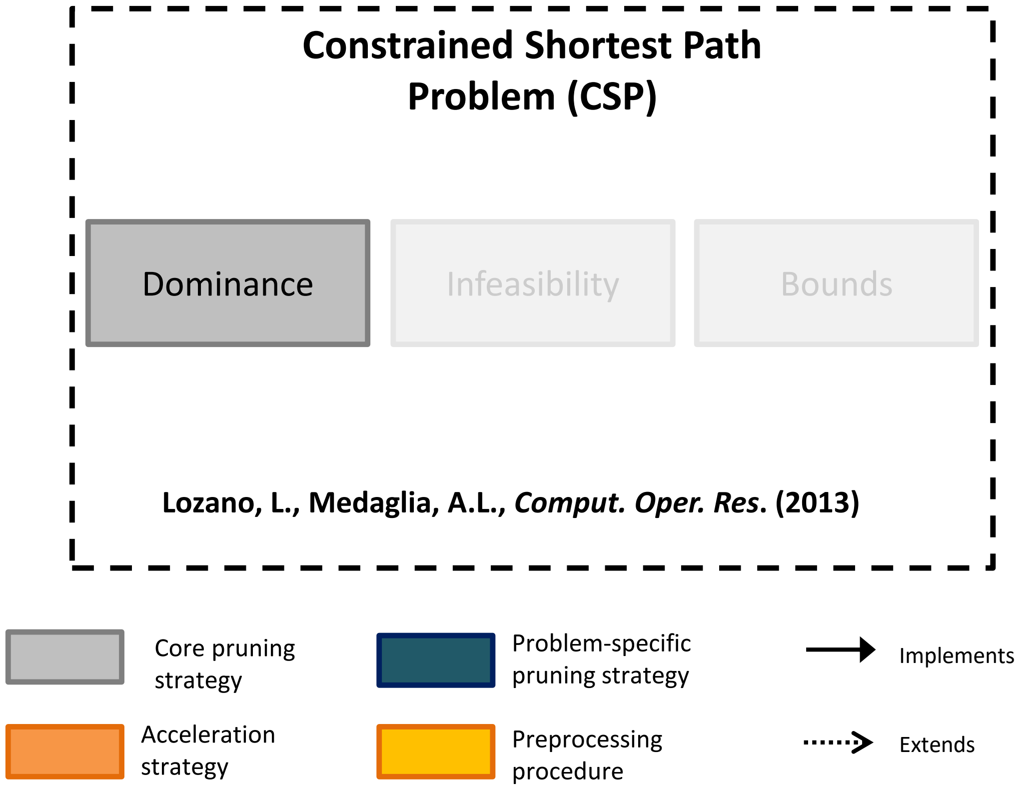


$T = 14$

$\mathcal{P} \leftarrow \{s, v_1, v_3, e\}$
$c(\mathcal{P}) = 11$
$t(\mathcal{P}) = 9$

Constrained Shortest Path Problem (CSP)

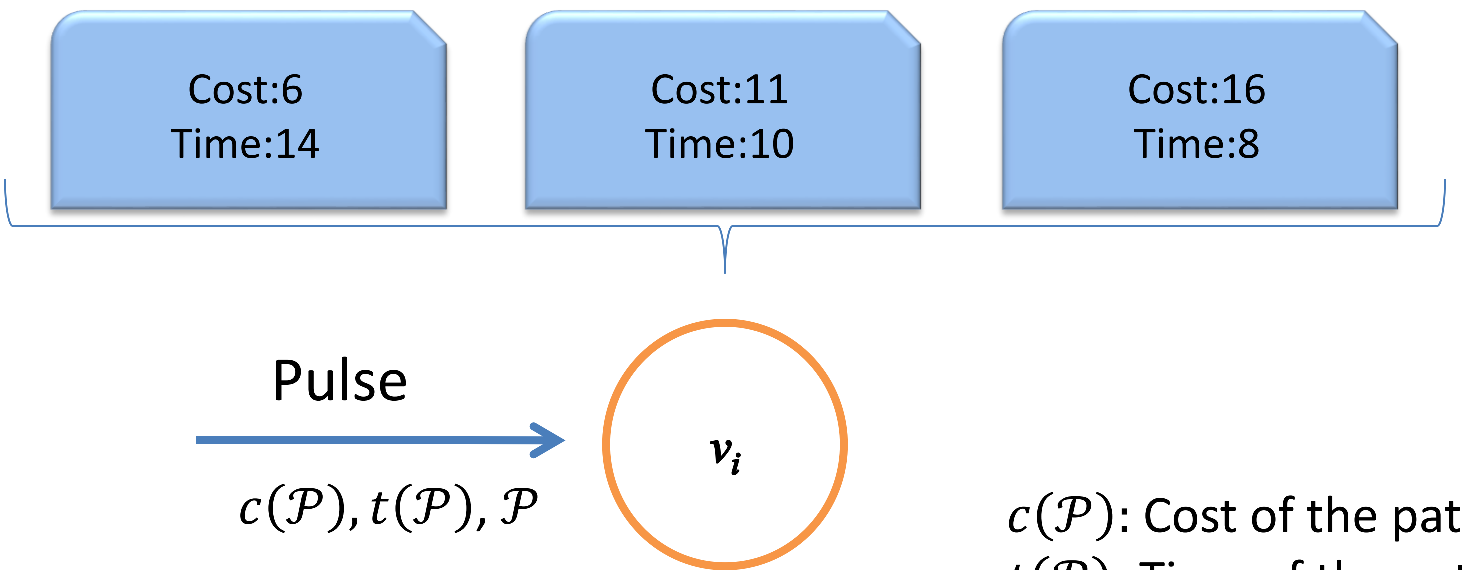
Dominance pruning



Constrained Shortest Path Problem (CSP)

Dominance pruning

- Dominance relationships can be defined over partial paths as in DP approaches

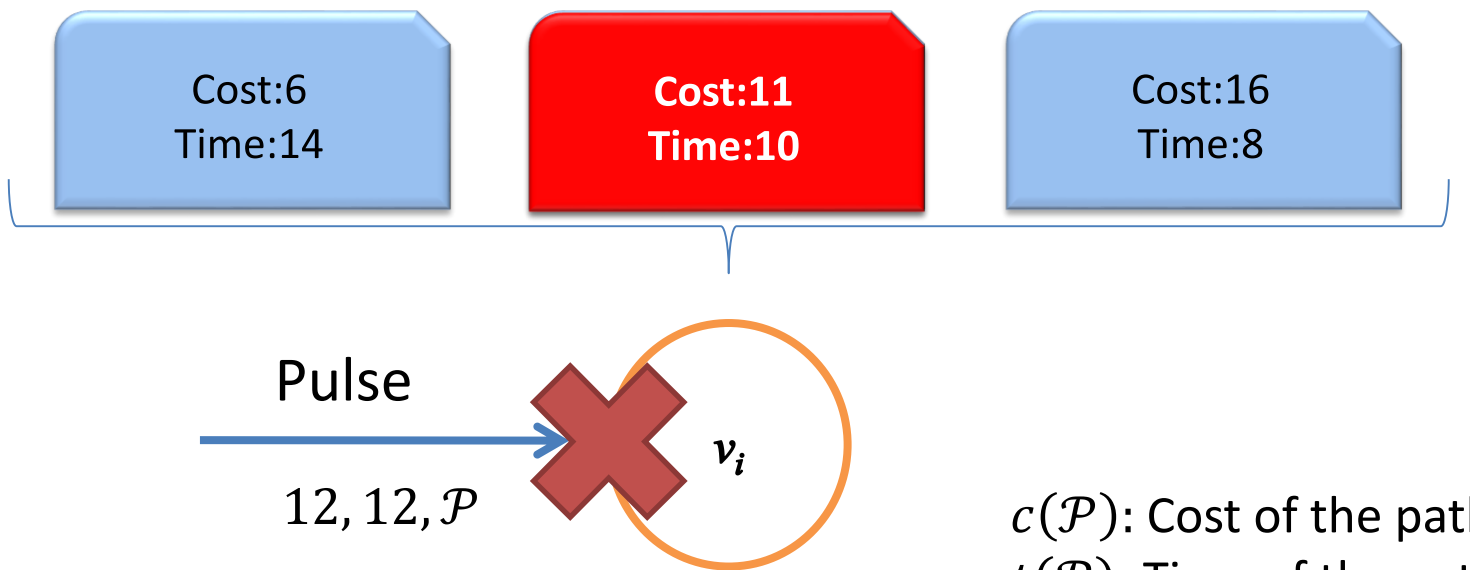


$c(\mathcal{P})$: Cost of the path $\mathcal{P}$
$t(\mathcal{P})$: Time of the path $\mathcal{P}$
$\mathcal{P}$: Partial path to v_i

Constrained Shortest Path Problem (CSP)

Dominance pruning

- Dominance relationships can be defined over partial paths as in DP approaches



Pulse is discarded by dominance pruning!

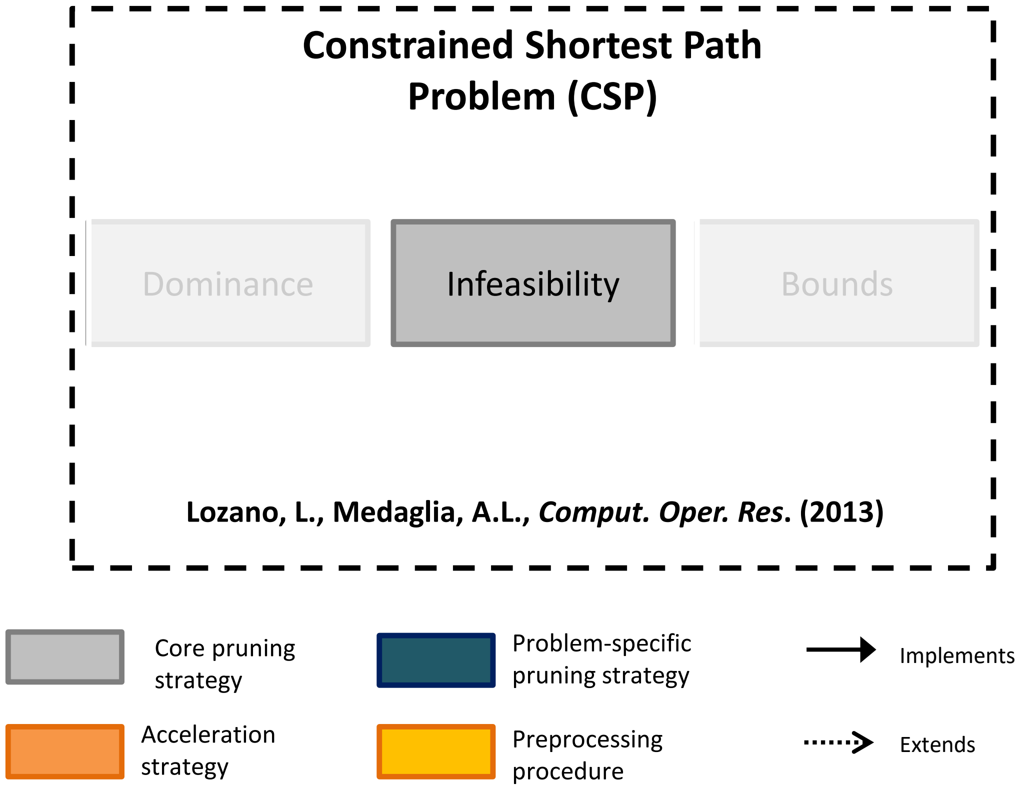
$c(\mathcal{P})$: Cost of the path $\mathcal{P}$

$t(\mathcal{P})$: Time of the path $\mathcal{P}$

$\mathcal{P}$: Partial path to v_i

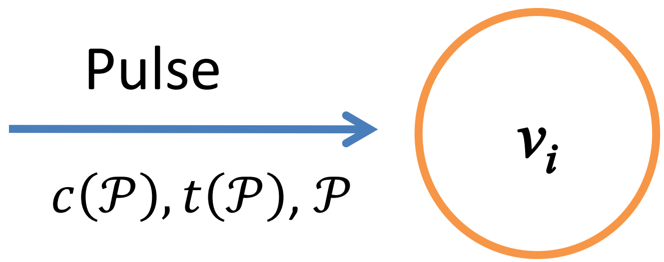
Constrained Shortest Path Problem (CSP)

Infeasibility pruning



Constrained Shortest Path Problem (CSP)

Infeasibility pruning

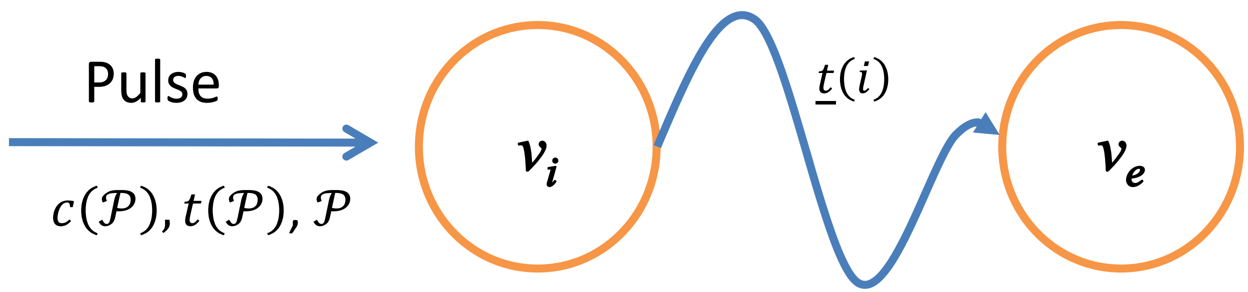


An incoming pulse to v_i is pruned if:

$$t(\mathcal{P}) > T$$

Constrained Shortest Path Problem (CSP)

Infeasibility pruning



An incoming pulse to v_i is pruned if:

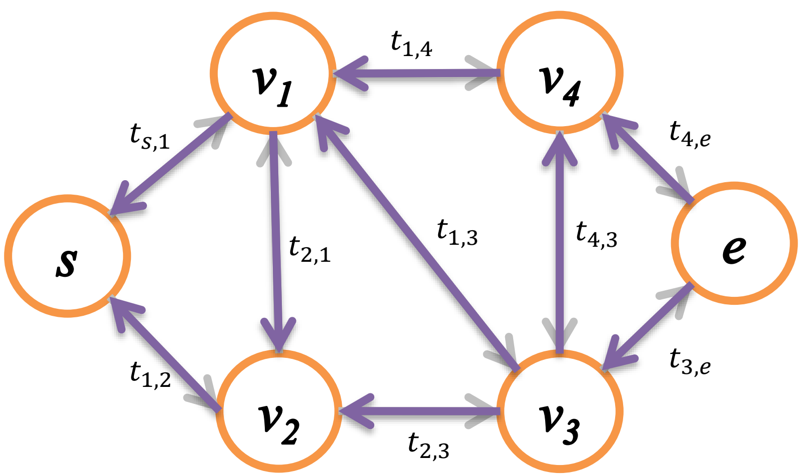
$$t(\mathcal{P}) + \boxed{\underline{t}(i)} > T$$

Lower bound on the resource consumption from v_i to v_e

Constrained Shortest Path Problem (CSP)

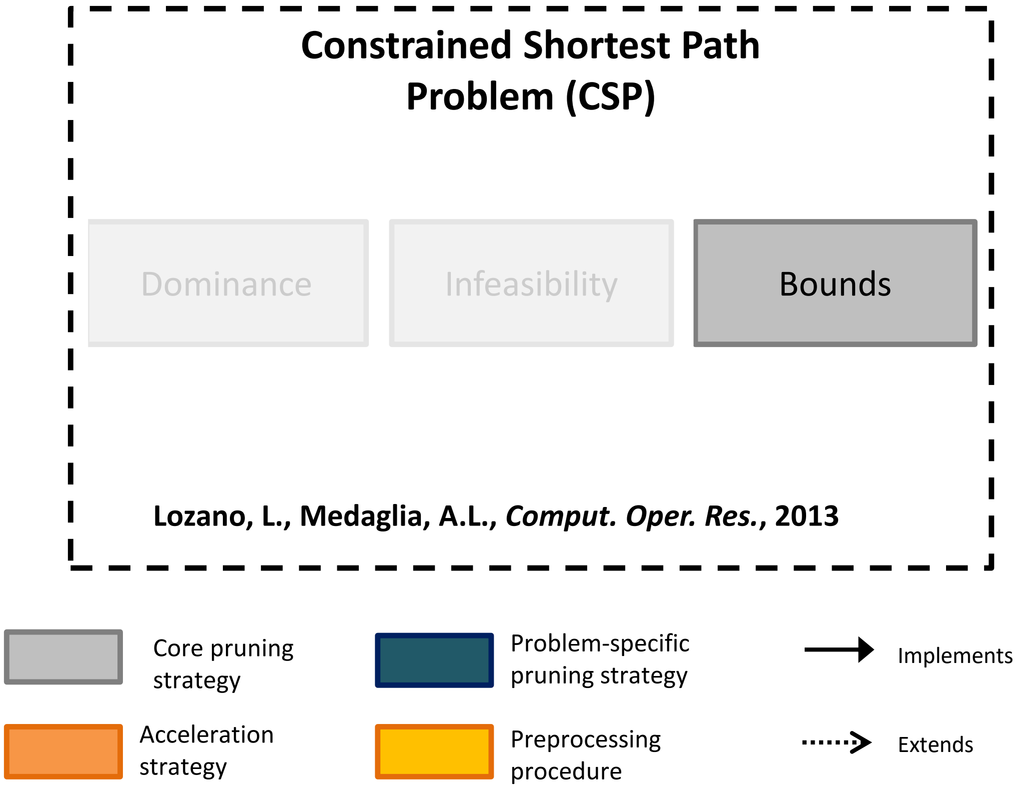
Infeasibility pruning

- Calculate a bound on the minimum resource required $\underline{t}(i)$ from node v_i to the final node
    - Reverse the network
    - Use a shortest path algorithm over the reversed network for the travel time attribute



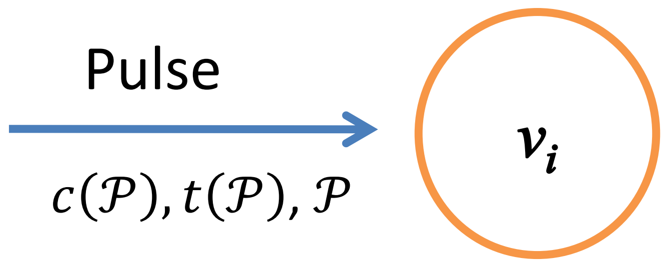
Constrained Shortest Path Problem (CSP)

Bounds pruning



Constrained Shortest Path Problem (CSP)

Bounds pruning



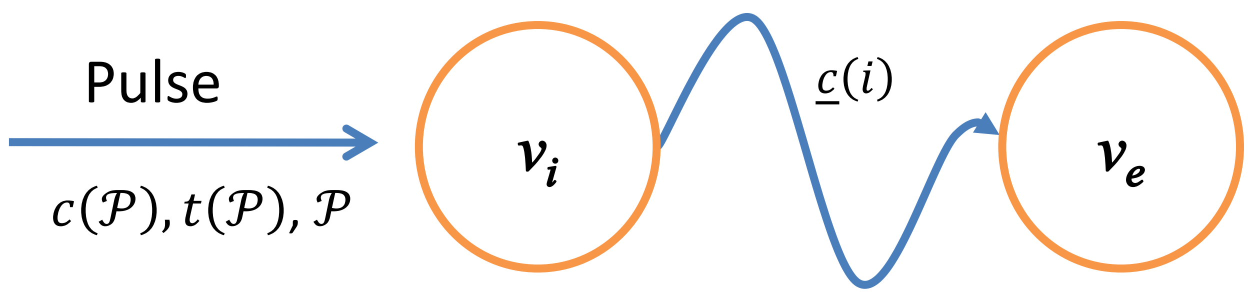
An incoming pulse to v_i is pruned if:

$$c(\mathcal{P}) \geq \boxed{\bar{c}} \longrightarrow \boxed{\text{Primal bound}}$$

Constrained Shortest Path Problem (CSP)

Bounds pruning

Universidad de los Andes Colombia



An incoming pulse to v_i is pruned if:

$$c(\mathcal{P}) + \boxed{\underline{c}(i)} \geq \boxed{\overline{c}}$$

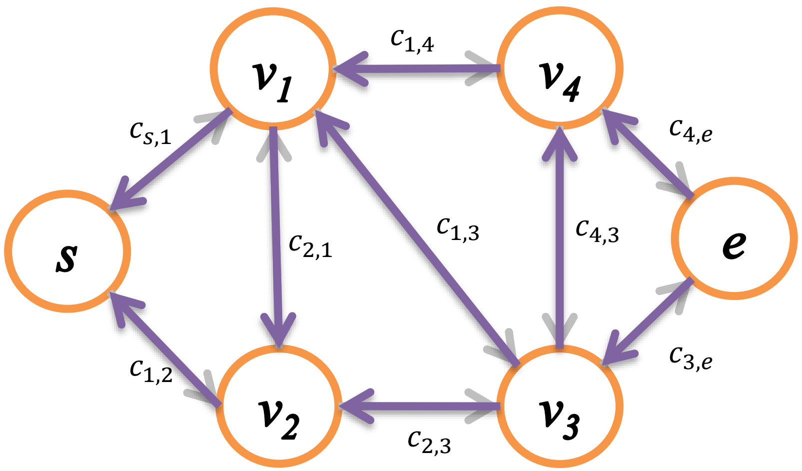
Primal bound

Lower bound on the cost from v_i to v_e

Constrained Shortest Path Problem (CSP)

Bounds pruning

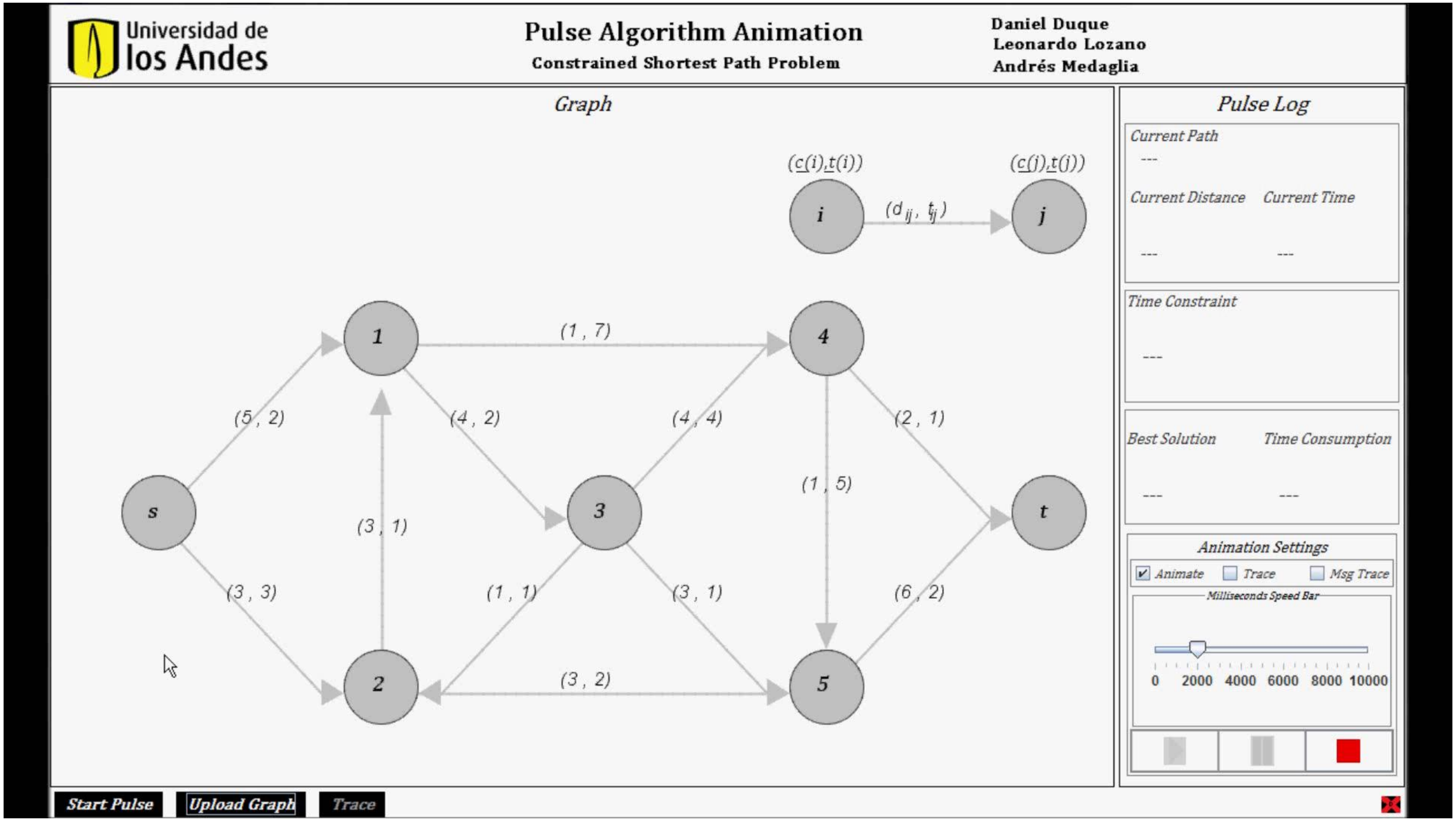
- Calculate a bound on the minimum cost $\underline{c}(i)$ from node v_i to the final node
  - Reverse the network
  - Use a shortest path algorithm over the reversed network for the cost attribute



Constrained Shortest Path Problem (CSP)

Numerical example

https://github.com/dukduque/jPulseBase



Constrained Shortest Path Problem (CSP)

Computational experiments

Setup:

- Benchmark algorithm proposed by Santos et al. (2007)
- Pulse algorithm coded in Java and compiled in Eclipse SDK 3.4.2
- CPU: Intel mobile core 2 Duo @ 2.4GHz 512MB of RAM for the JVM
- The amount of labels is set to 3

Constrained Shortest Path Problem (CSP)

Computational experiments

Table 1
Computational results for the Santos et al. [29] instances.

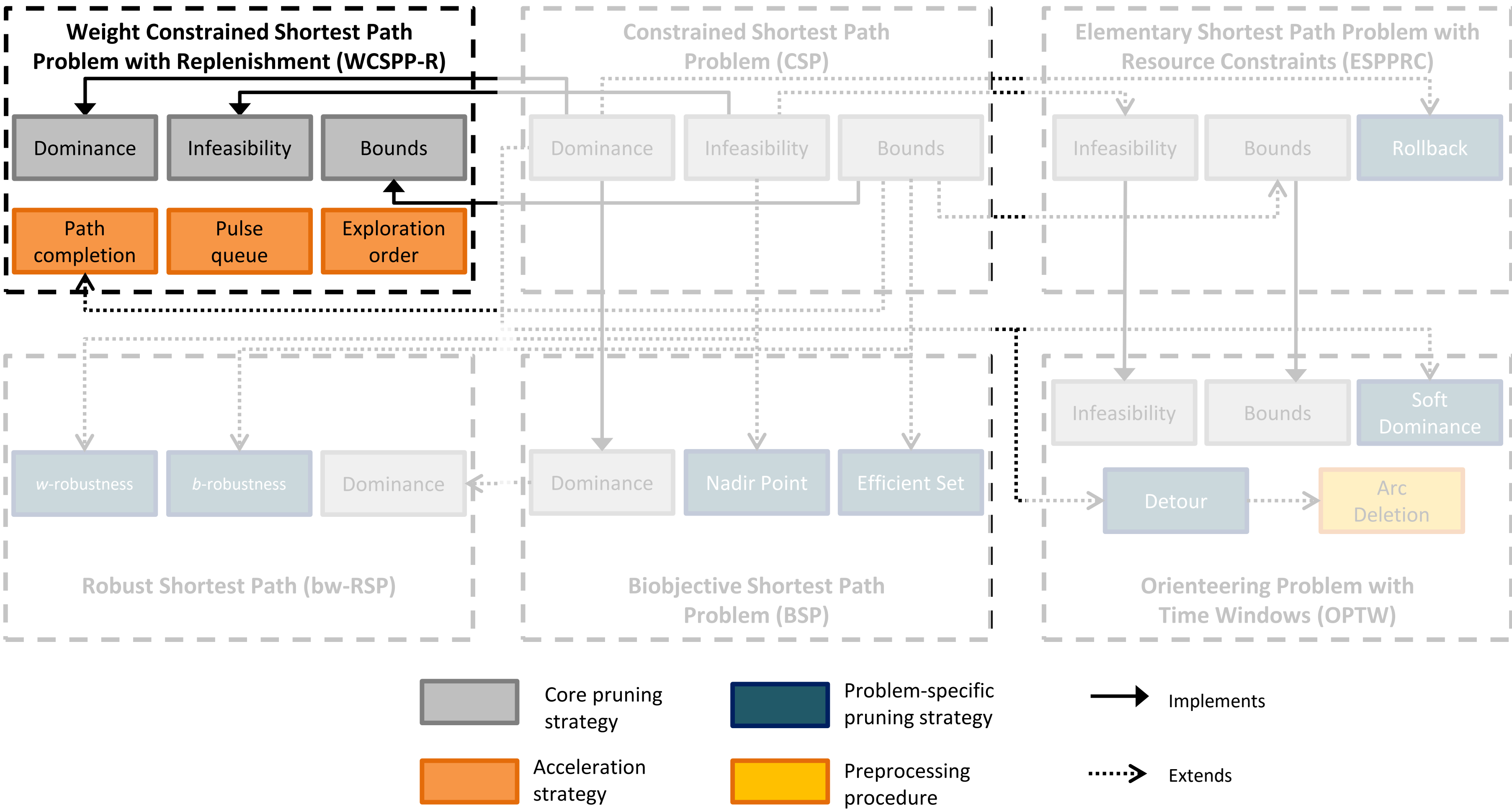
Nodes	Arcs	$p=0.1$			$p=0.2$			$p=0.4$			$p=0.6$			$p=0.8$		
		Pulse	**LRA**	**Speedup**	**Pulse**	**LRA**	**Speedup**	**Pulse**	**LRA**	**Speedup**	**Pulse**	**LRA**	**Speedup**	**Pulse**	**LRA**	**Speedup**
40,000	60,000	0.05	2.40	48.00	0.05	2.40	48.00	0.05	2.40	48.00	0.05	2.40	48.00	0.05	2.40	48.00
40,000	100,000	0.08	3.00	37.50	0.08	3.00	37.50	0.08	3.00	37.50	0.08	3.10	38.75	0.08	3.10	38.75
40,000	200,000	0.12	4.90	40.83	0.12	4.90	40.83	0.12	4.90	40.83	0.12	5.00	41.67	0.12	5.10	42.50
40,000	400,000	0.21	7.70	36.67	0.22	7.70	35.00	0.26	7.80	30.00	0.22	8.00	36.36	0.21	8.30	39.52
40,000	600,000	0.29	10.70	36.90	0.34	10.70	31.47	0.31	10.90	35.16	0.31	11.20	36.13	0.31	11.70	37.74
40,000	800,000	0.39	13.00	33.33	0.47	13.10	27.87	0.52	13.40	25.77	0.55	13.80	25.09	0.43	14.40	33.49
Geometric mean				40.32			38.41			37.26			41.22			41.10

- Other benchmark algorithm: Zhu & Wilhelm (2012)
- See Lozano & Medaglia (2013)
- Other algorithmis:
  - Sedeño-Noda & Alonso-Rodriguez (2015) - k-SP
  - Thomas, Calogiuri & Hewitt (2018) – RC-BDA*

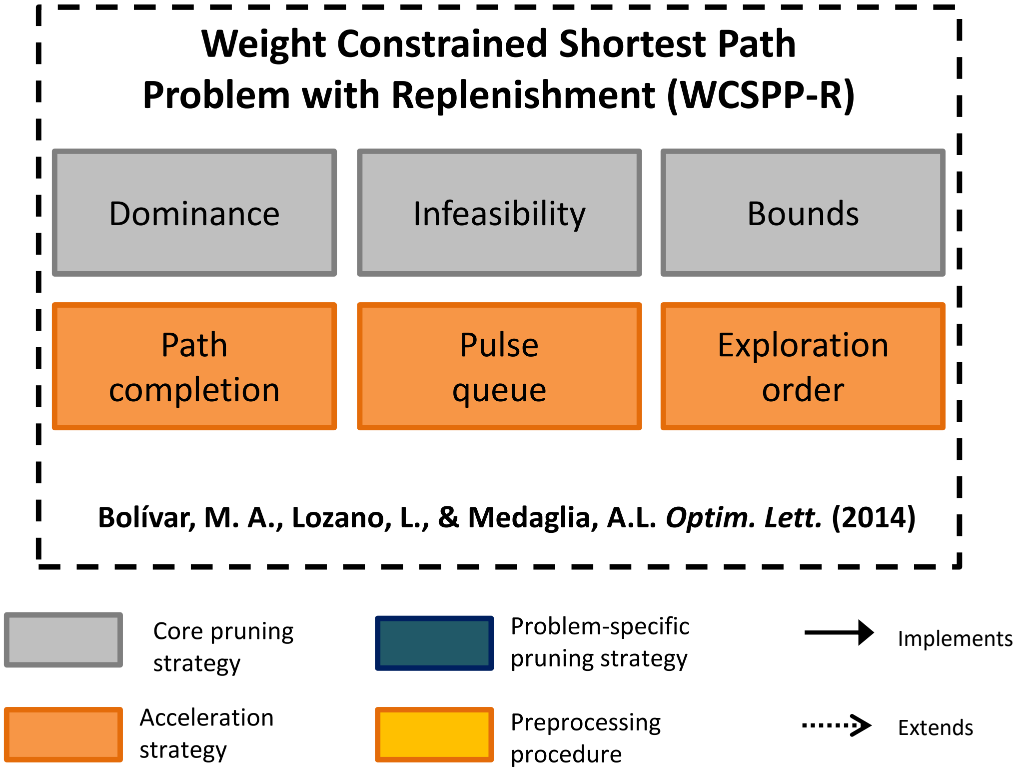
Agenda

- Part I: fundamentals
- Part II: intuition
- Part III: extensions
  - **Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)**
  - Biobjective Shortest Path Problem (BSP)
  - Elementary Shortest Path Problem with Resource Constraints (ESPPRC)
  - Orienteering Problem with Time Windows (OPTW)
  - Robust Shortest Path (bw-RSP)
- Part IV: applications
- Part V: perspectives

Pulse Algorithm for Hard Shortest Path Problems



Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)



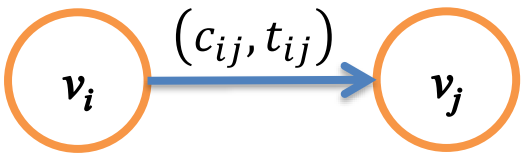
- Smith, Boland & Waterer (2012)
- Lozano & Medaglia (2013)

Universidad de los Andes Colombia

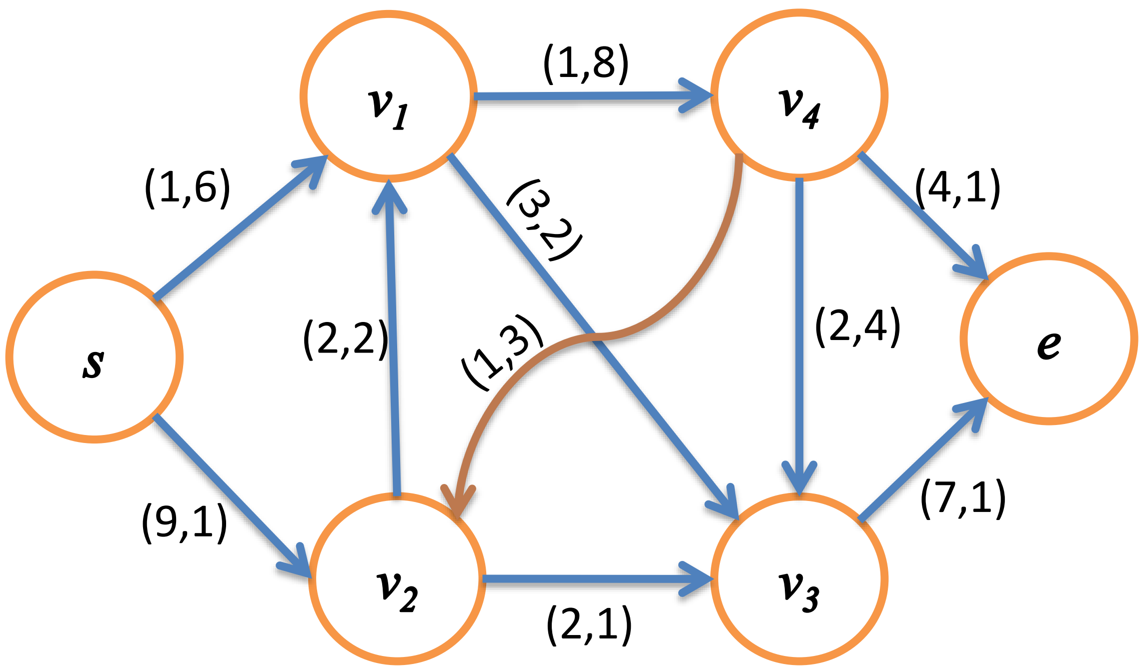
- The WCSPP-R is defined by:
  - Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$
    - $\mathcal{N} = \{v_1, \dots, v_i, \dots, v_n\}$
    - $\mathcal{A} = \{(i,j) | v_i \in \mathcal{N}, v_j \in \mathcal{N},\ i \neq j\}$
  - Find a minimum cost path starting at node v_s and ending at node v_e
  - Nonnegative weights c_{ij} and t_{ij} are the cost and travel time of traversing arc $(i,j) \in \mathcal{A}$
  - Time constraint T
  - $\boxed{\text{Replenishment arcs}}$

Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Problem statement

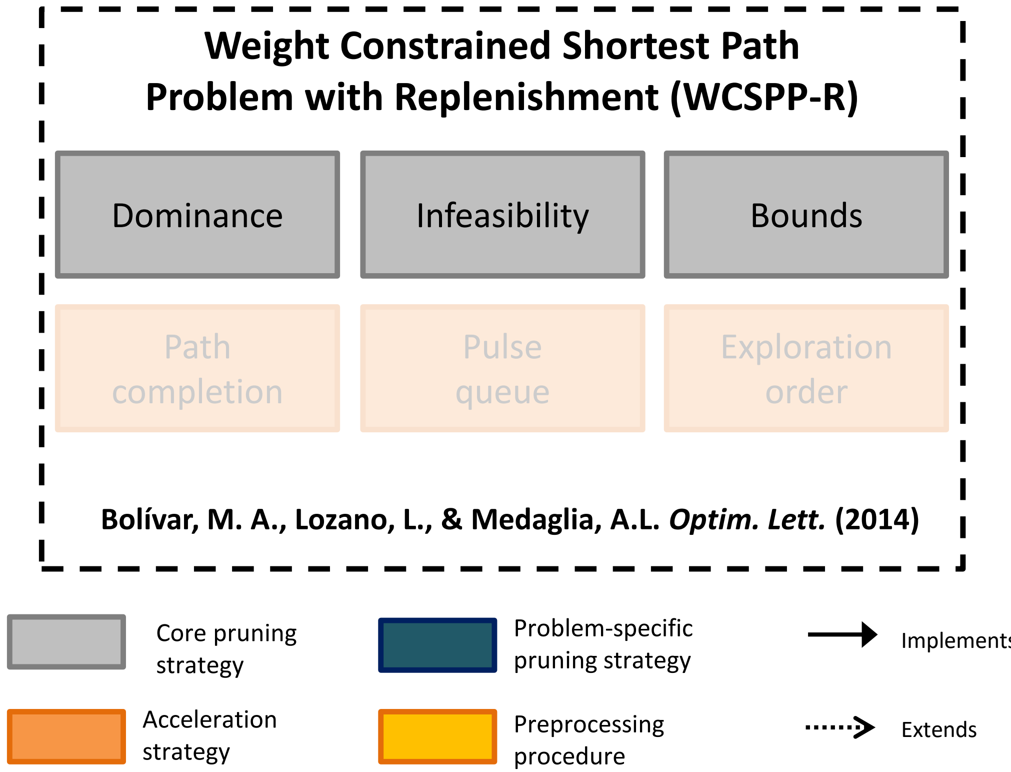


$T = 14$



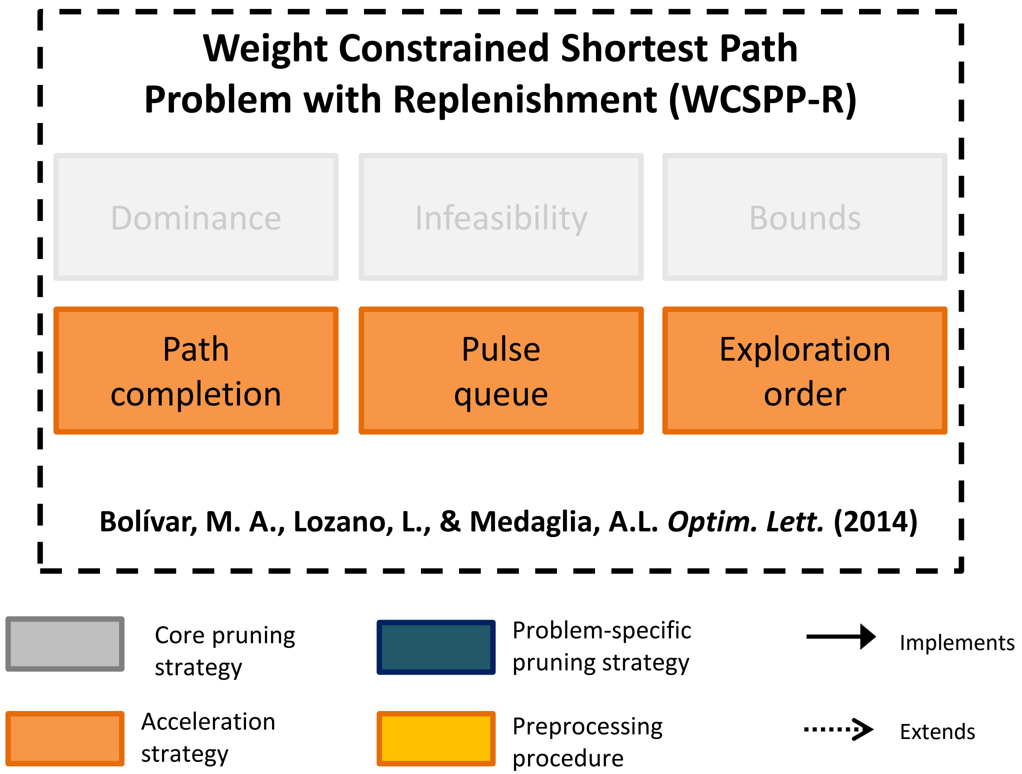
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pruning strategies



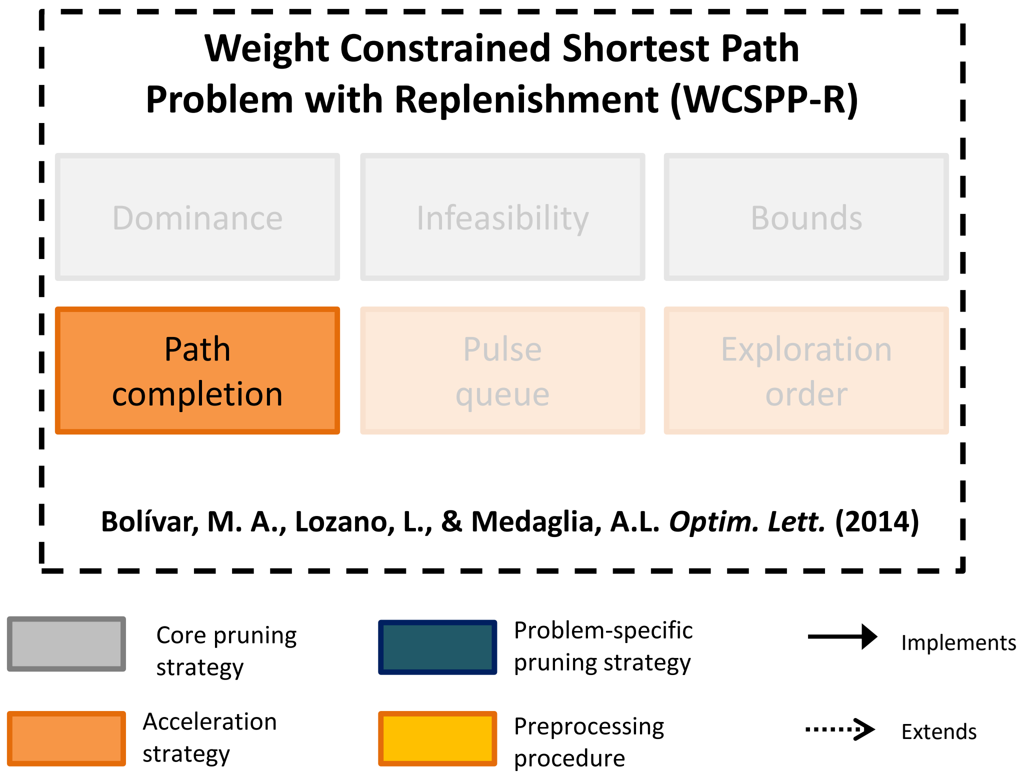
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Acceleration strategies

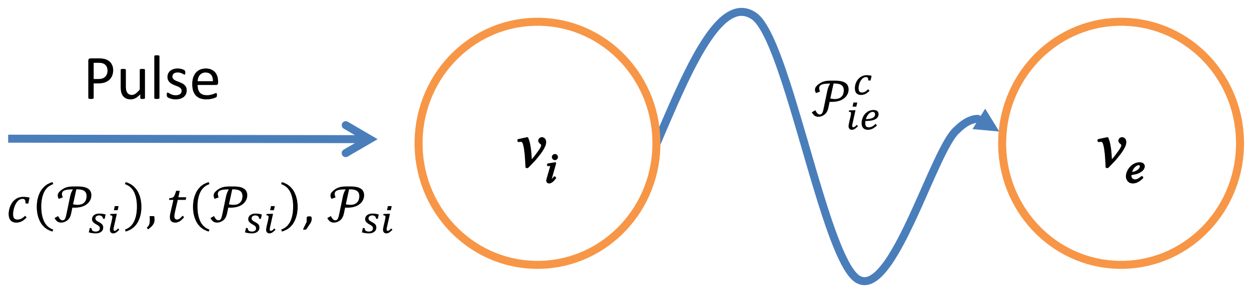


Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Acceleration strategies



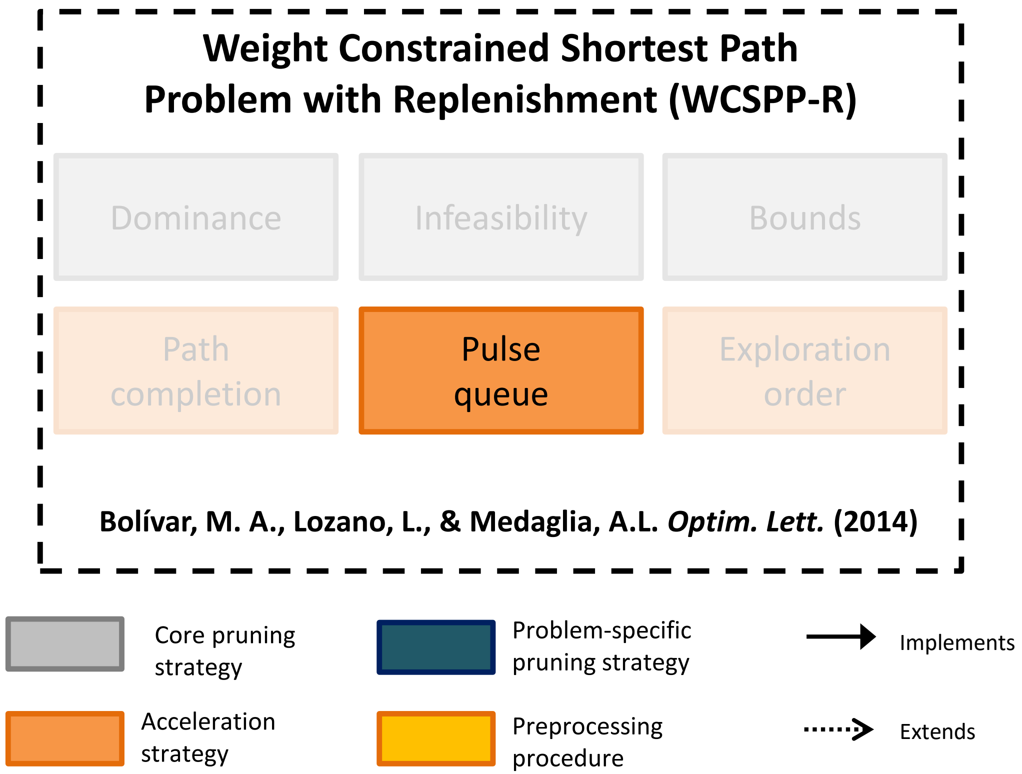
Path completion



$$t(\mathcal{P}_{si}) + t(\mathcal{P}_{ie}^c) \leq T \wedge \texttt{feasible}(\mathcal{P}_{ie}^c)$$

Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

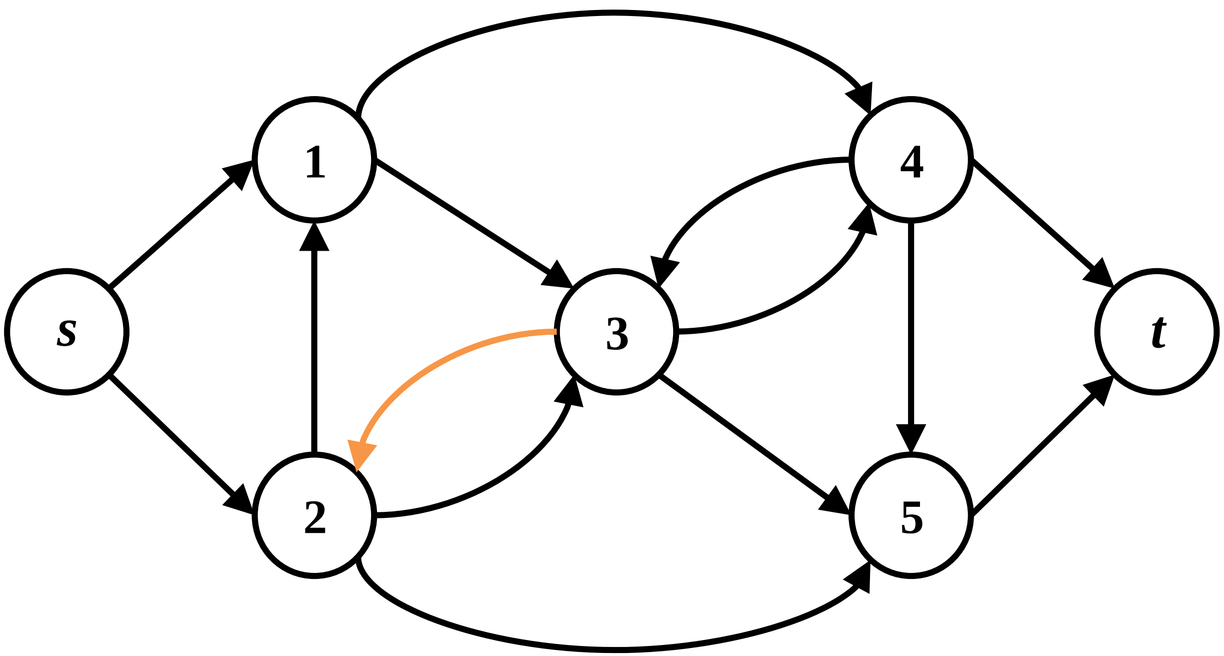
Acceleration strategies



Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2

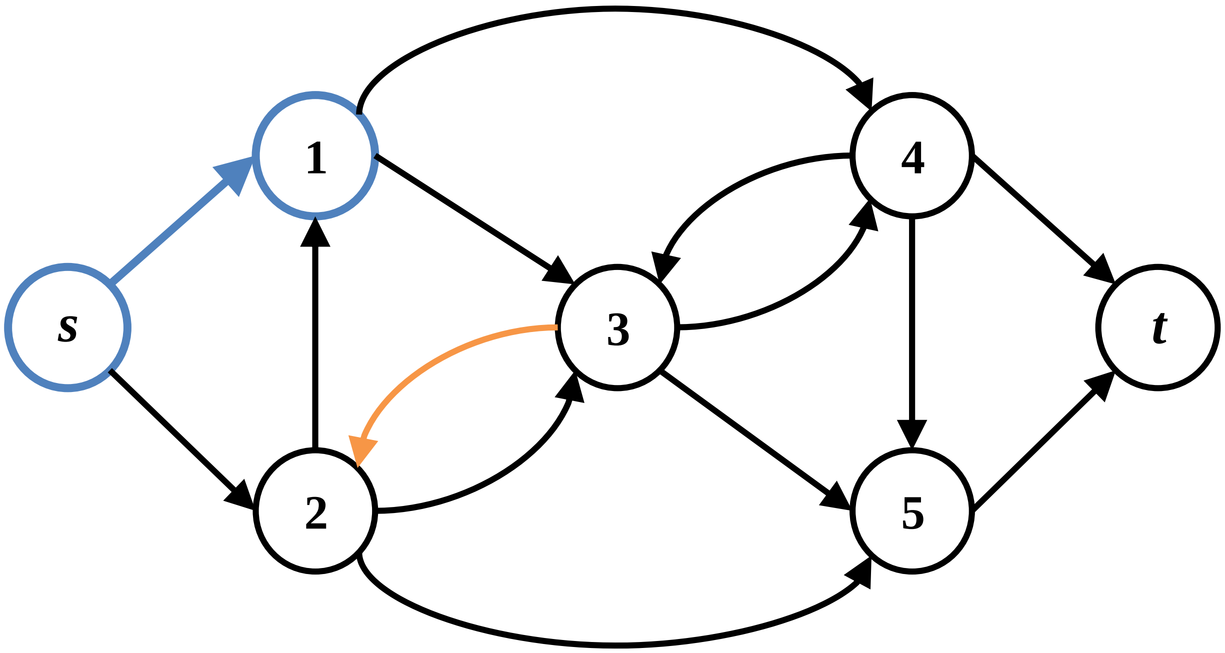


Pulse queue

Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2

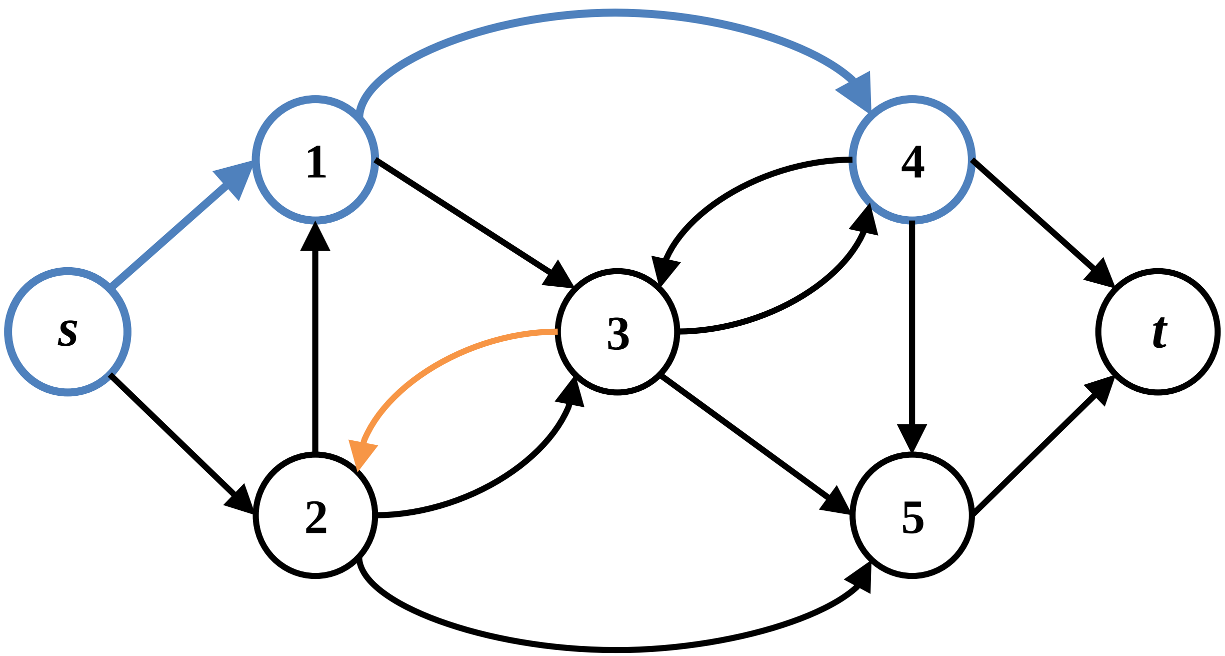


Pulse queue

Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

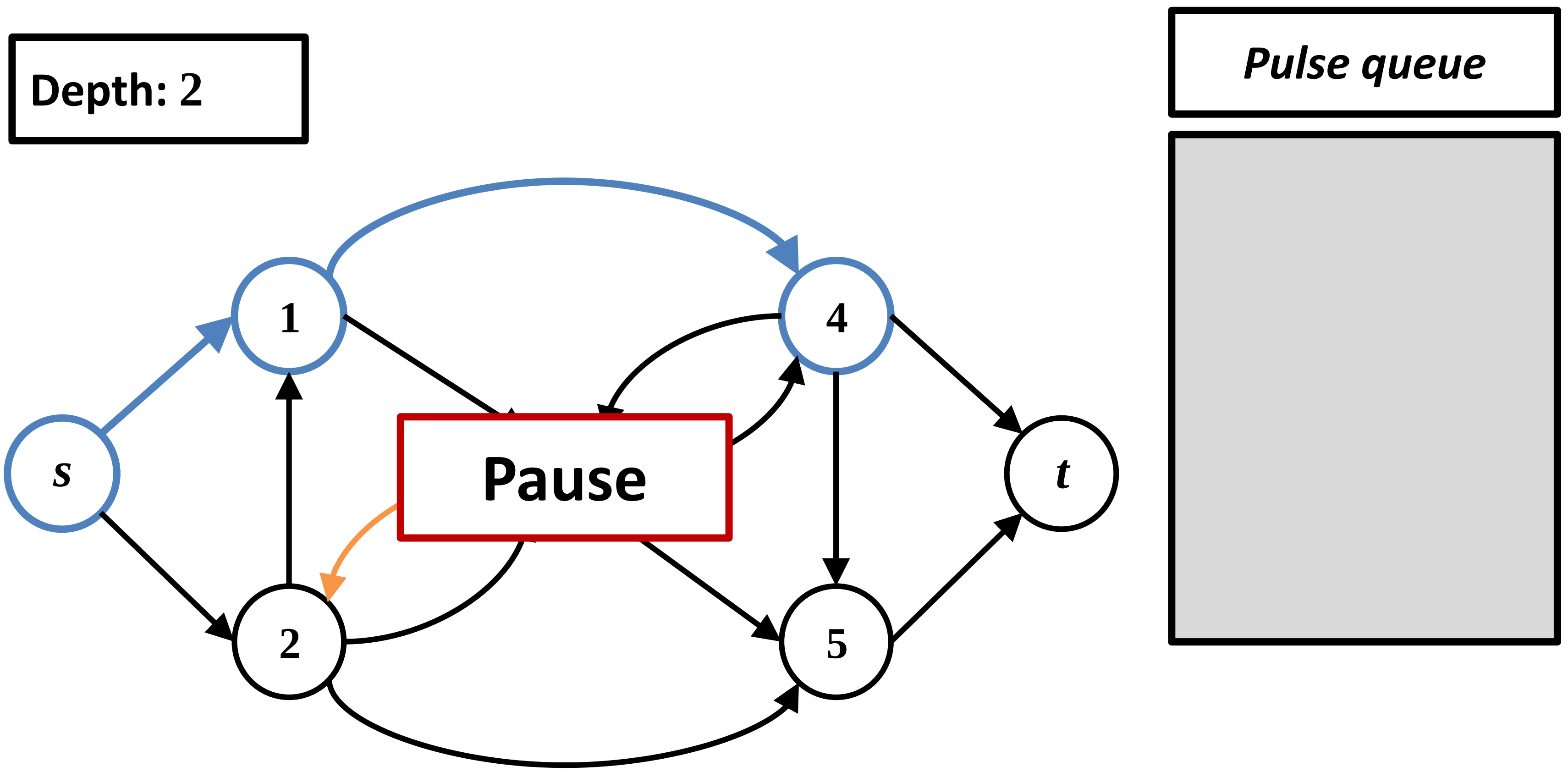
Depth: 2



Pulse queue

Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

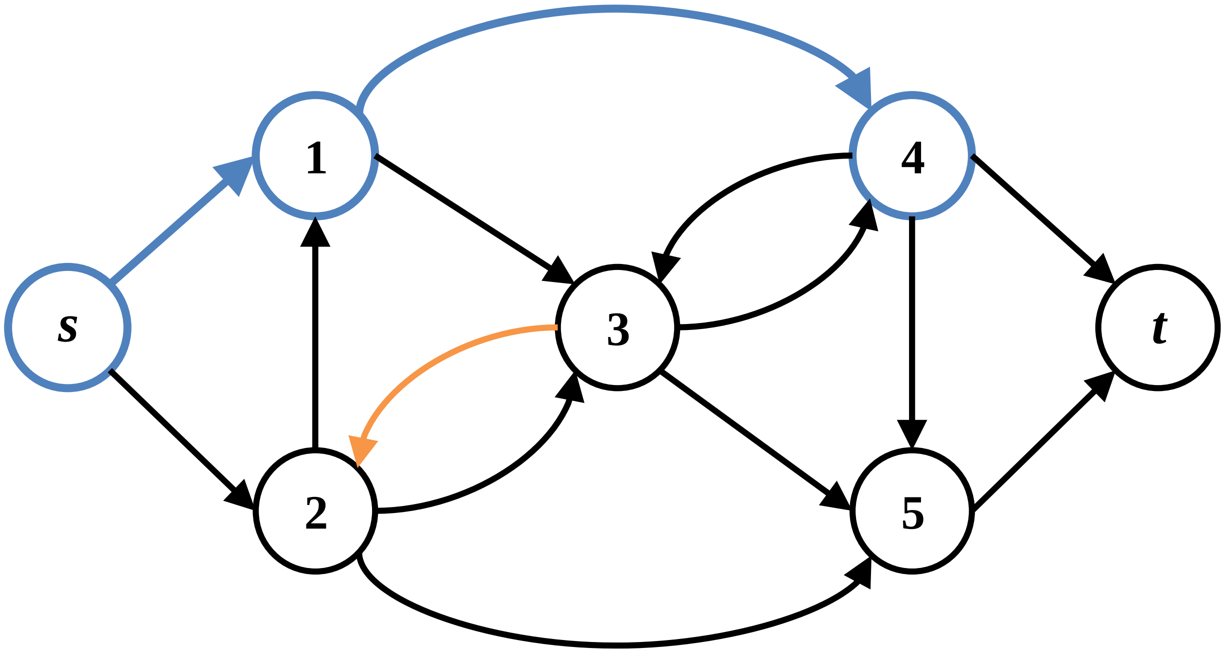
Pulse queue



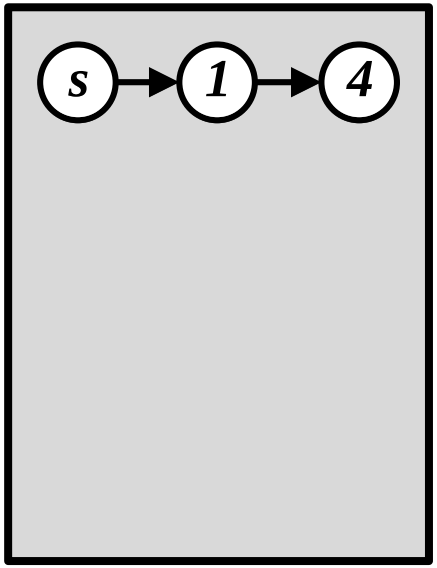
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



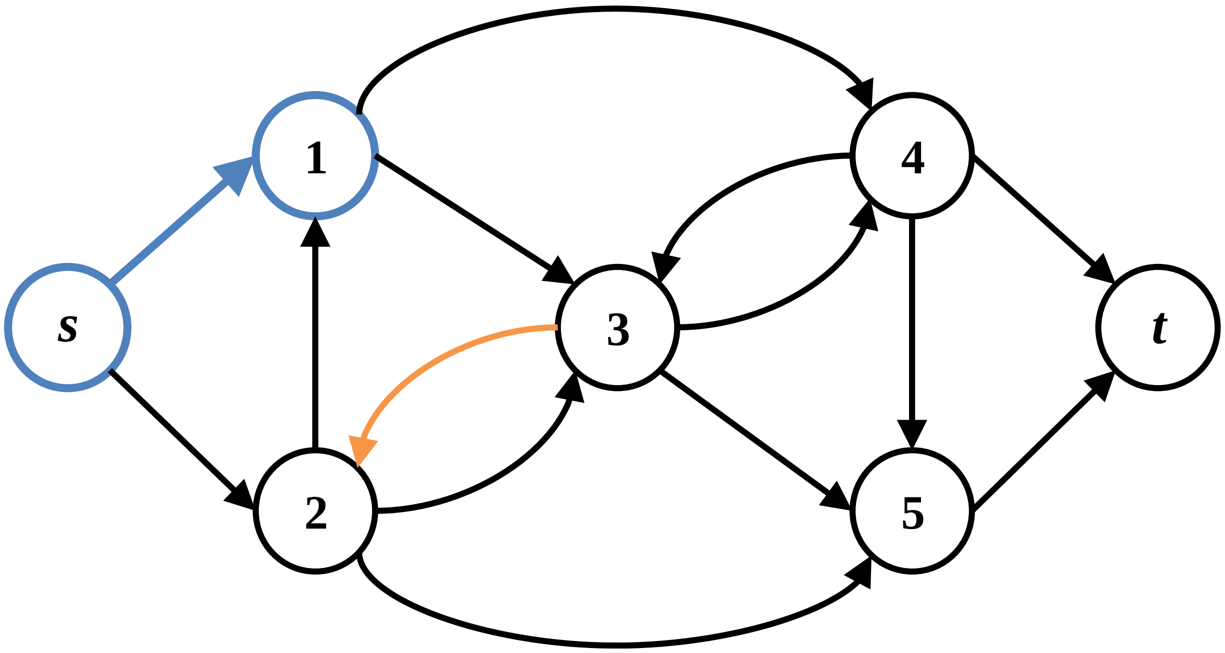
Pulse queue



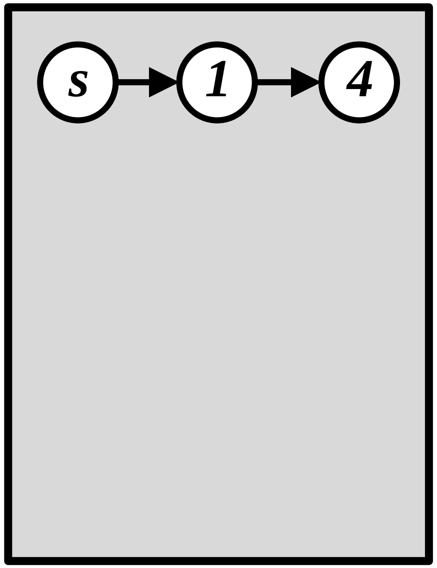
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2

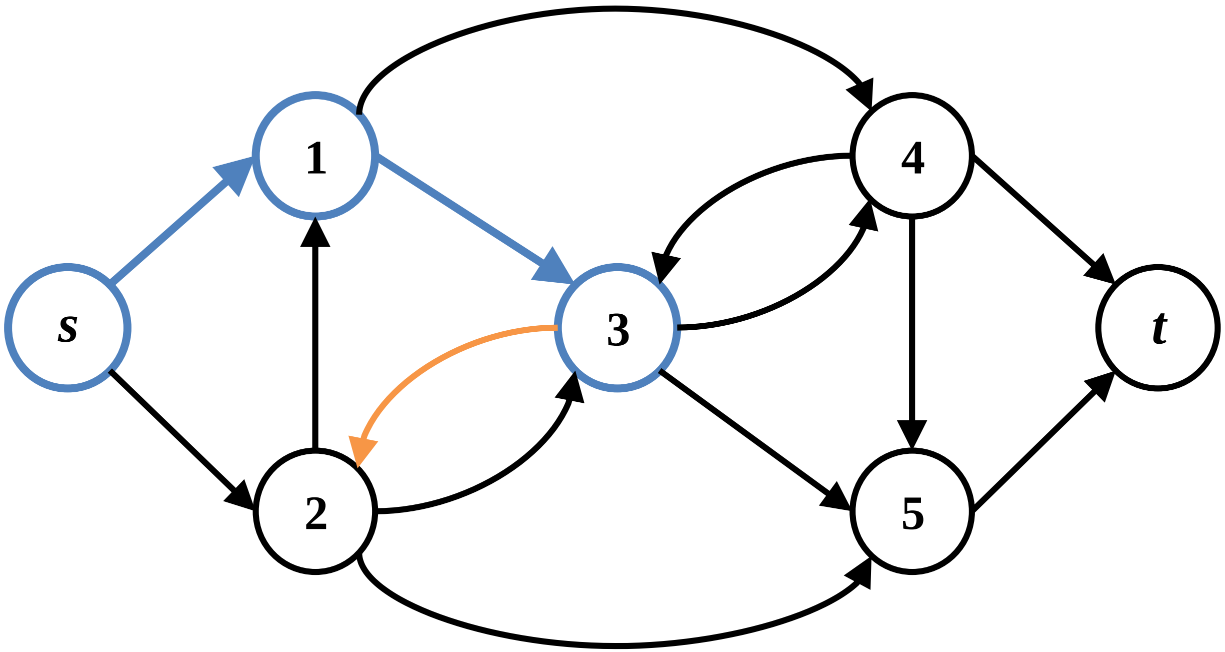


Pulse queue

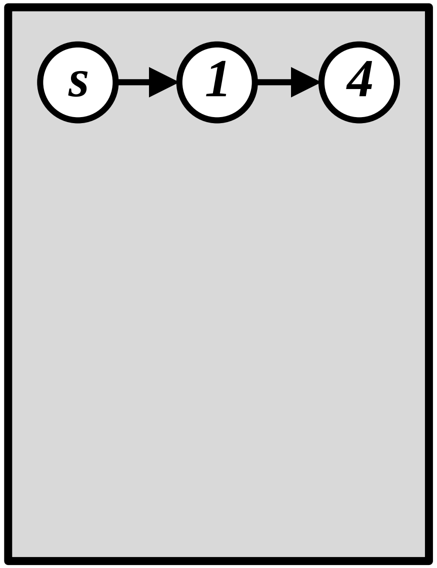


Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)
Pulse queue

Depth: 2

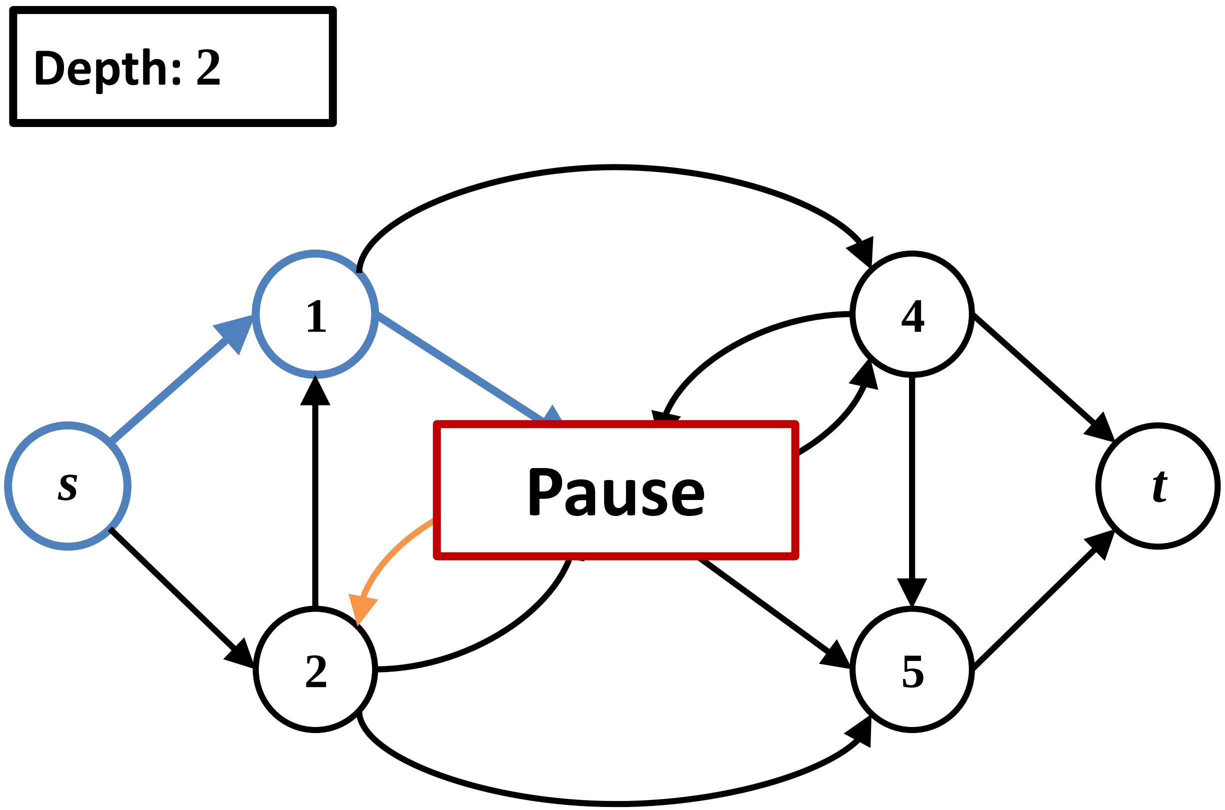


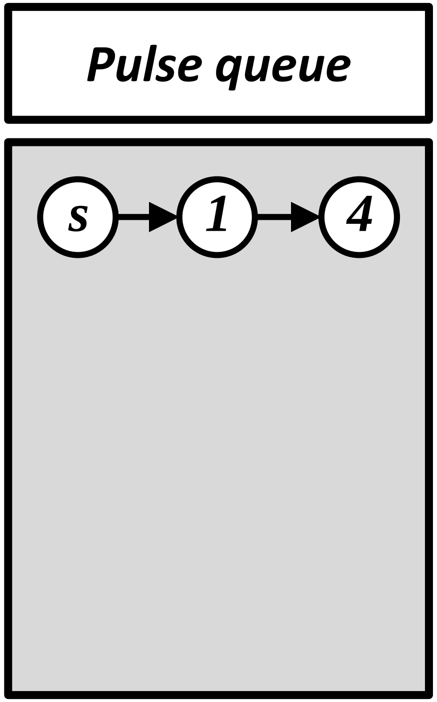
Pulse queue



Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

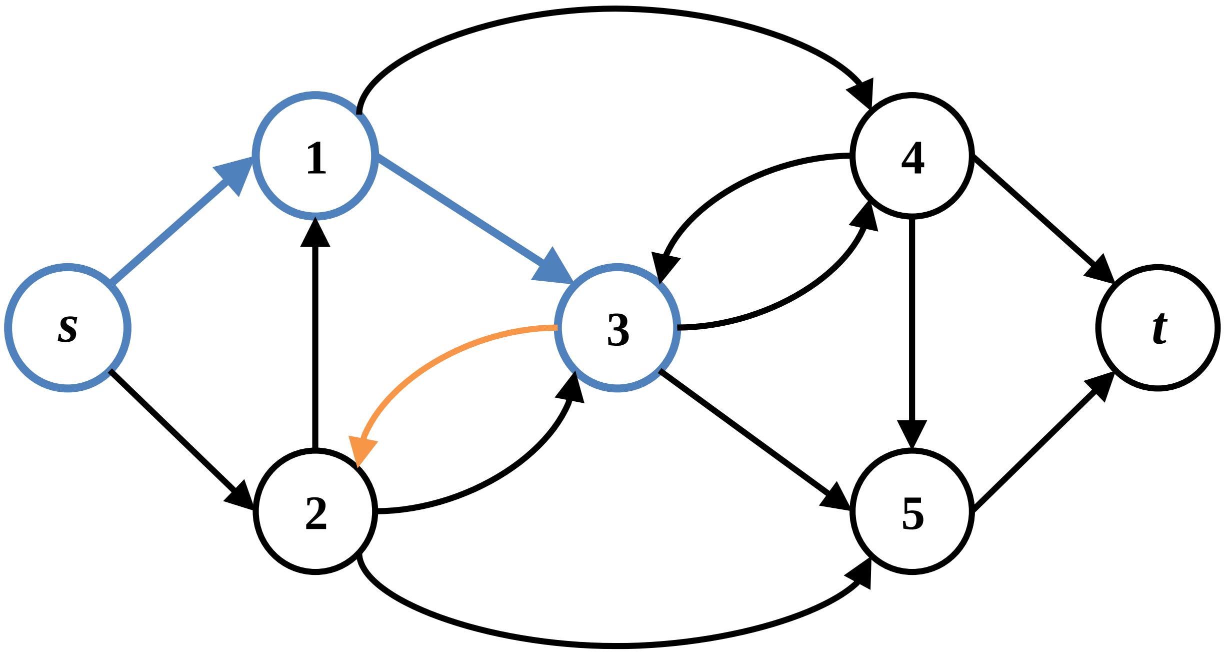




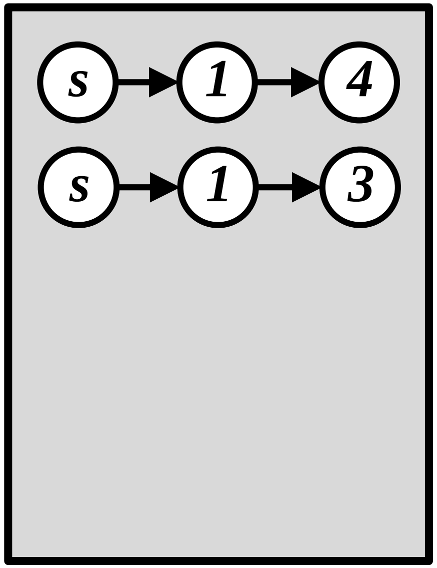
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

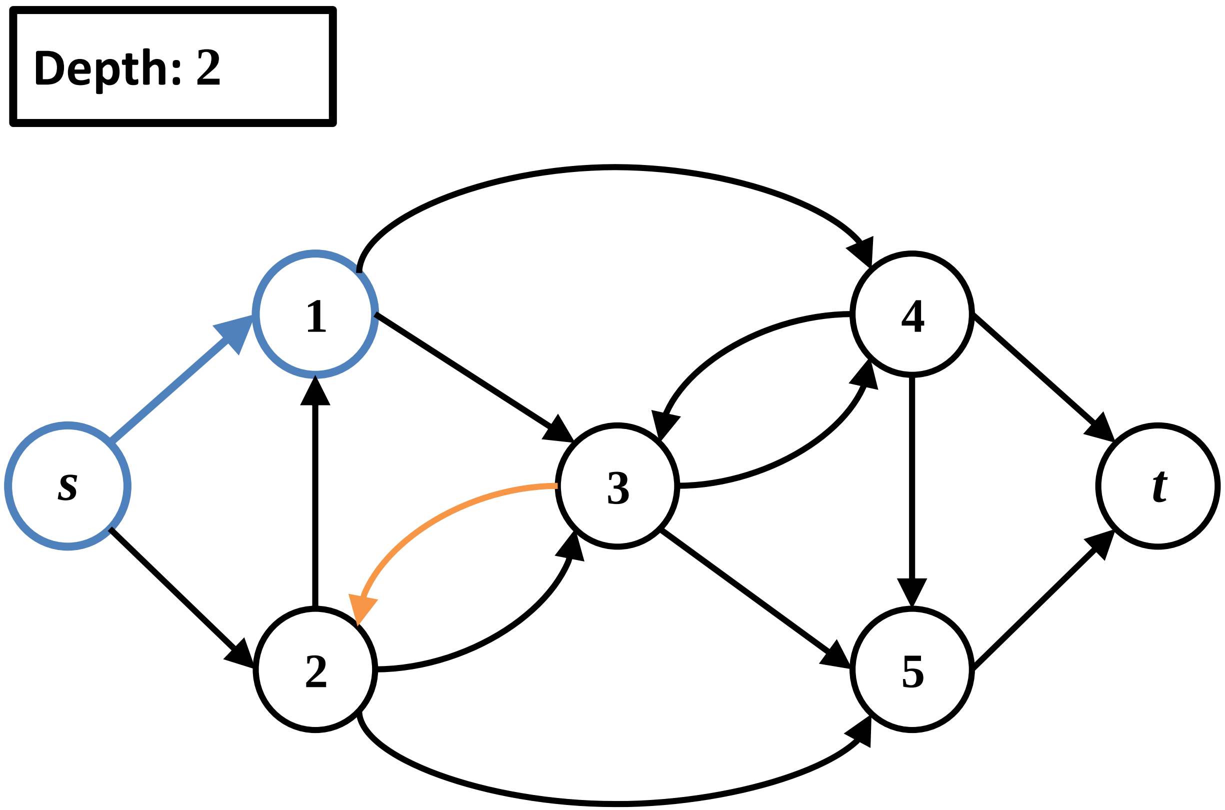
Depth: 2

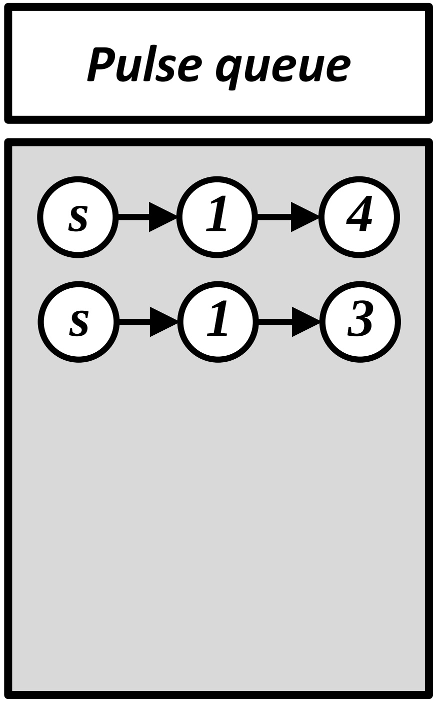


Pulse queue



Universidad de los Andes Colombia

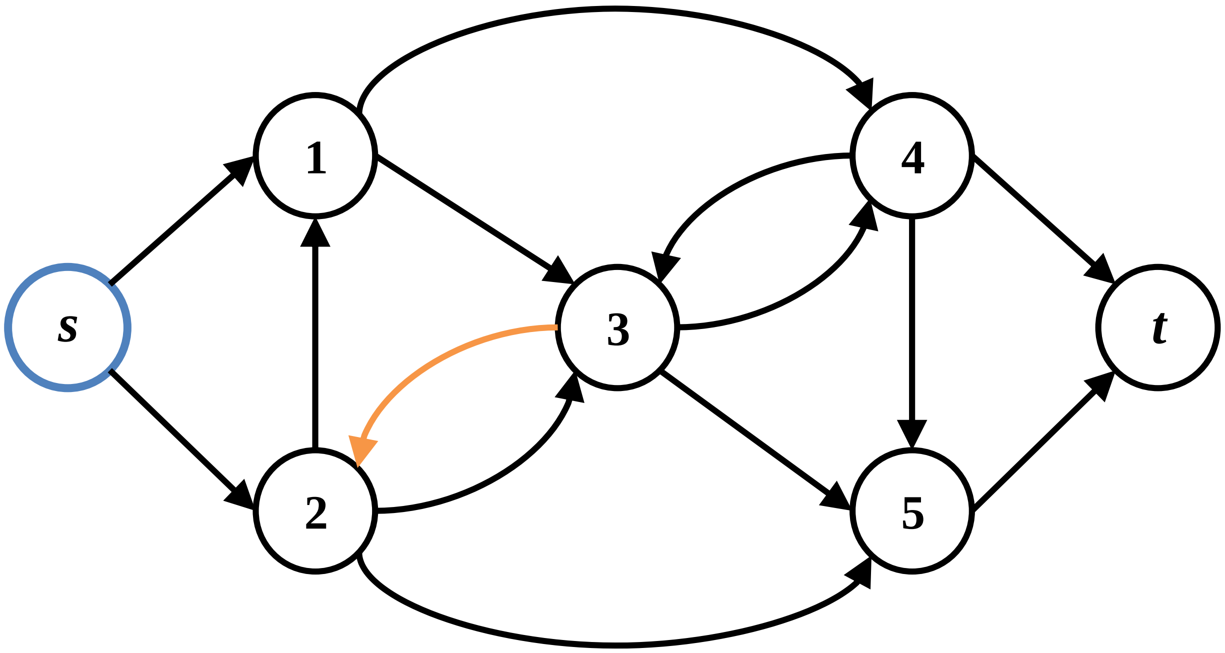




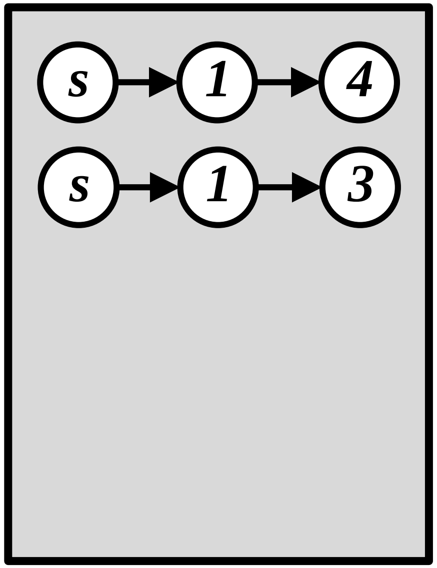
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2

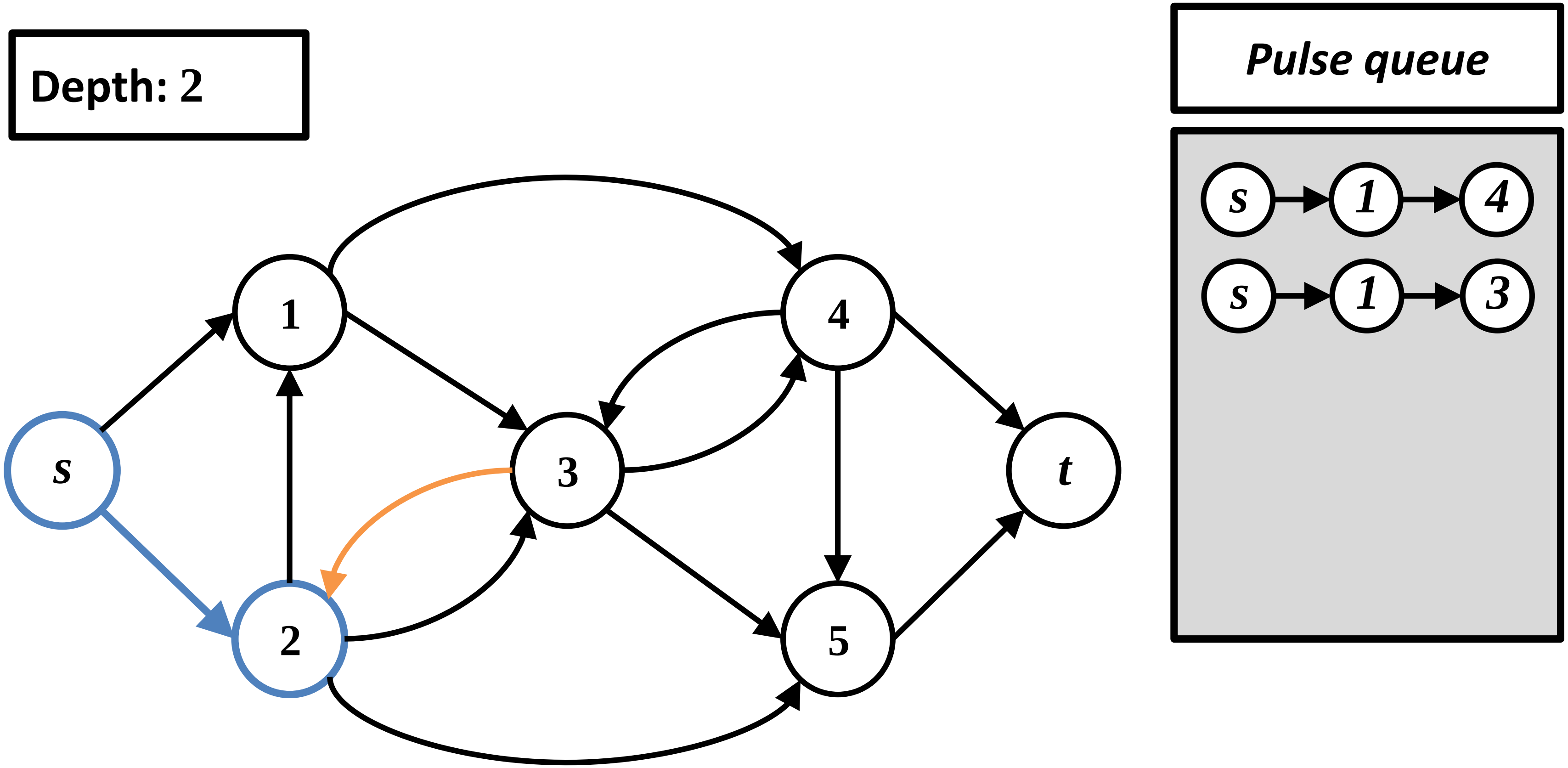


Pulse queue



Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

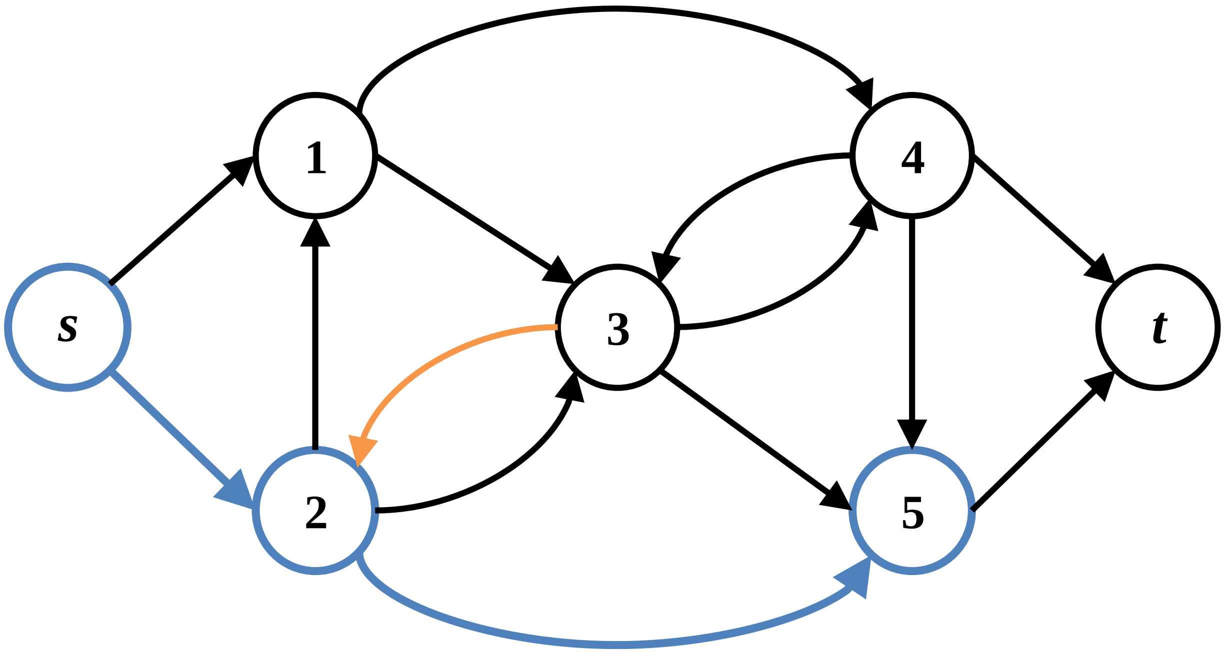
Pulse queue



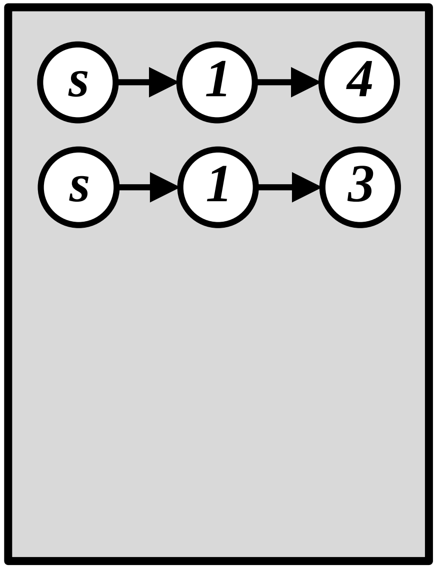
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



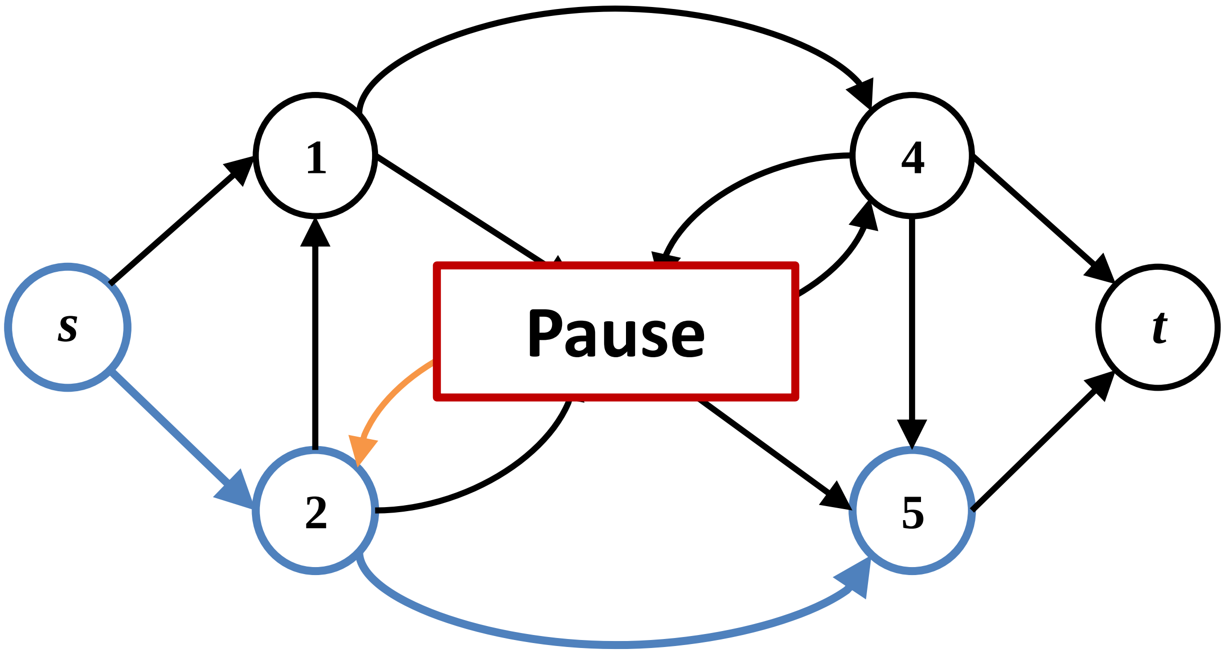
Pulse queue



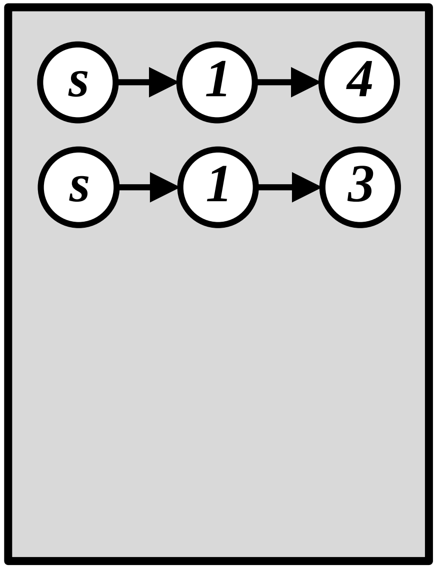
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



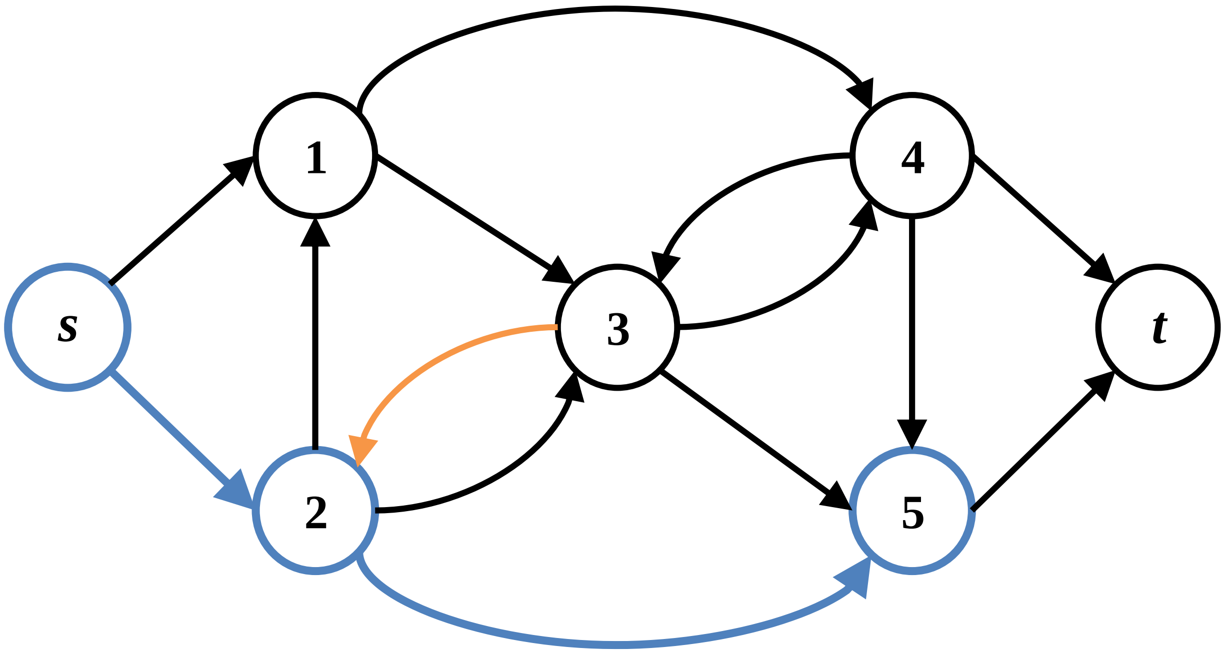
Pulse queue



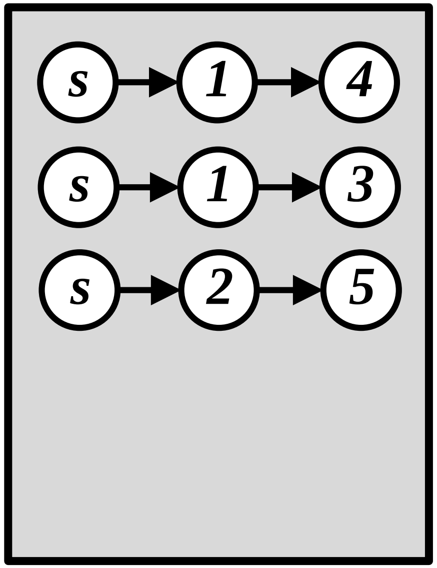
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2

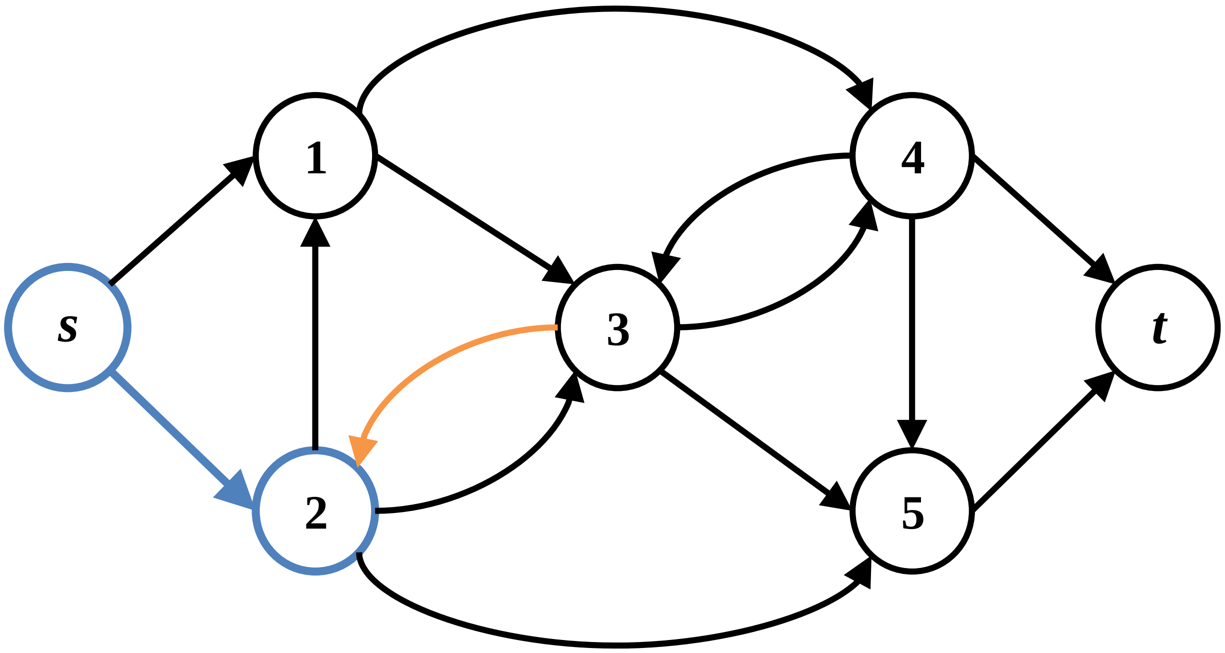


Pulse queue

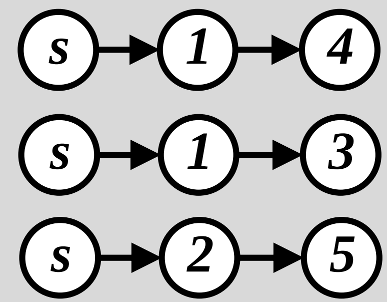


Universidad de los Andes Colombia

Depth: 2

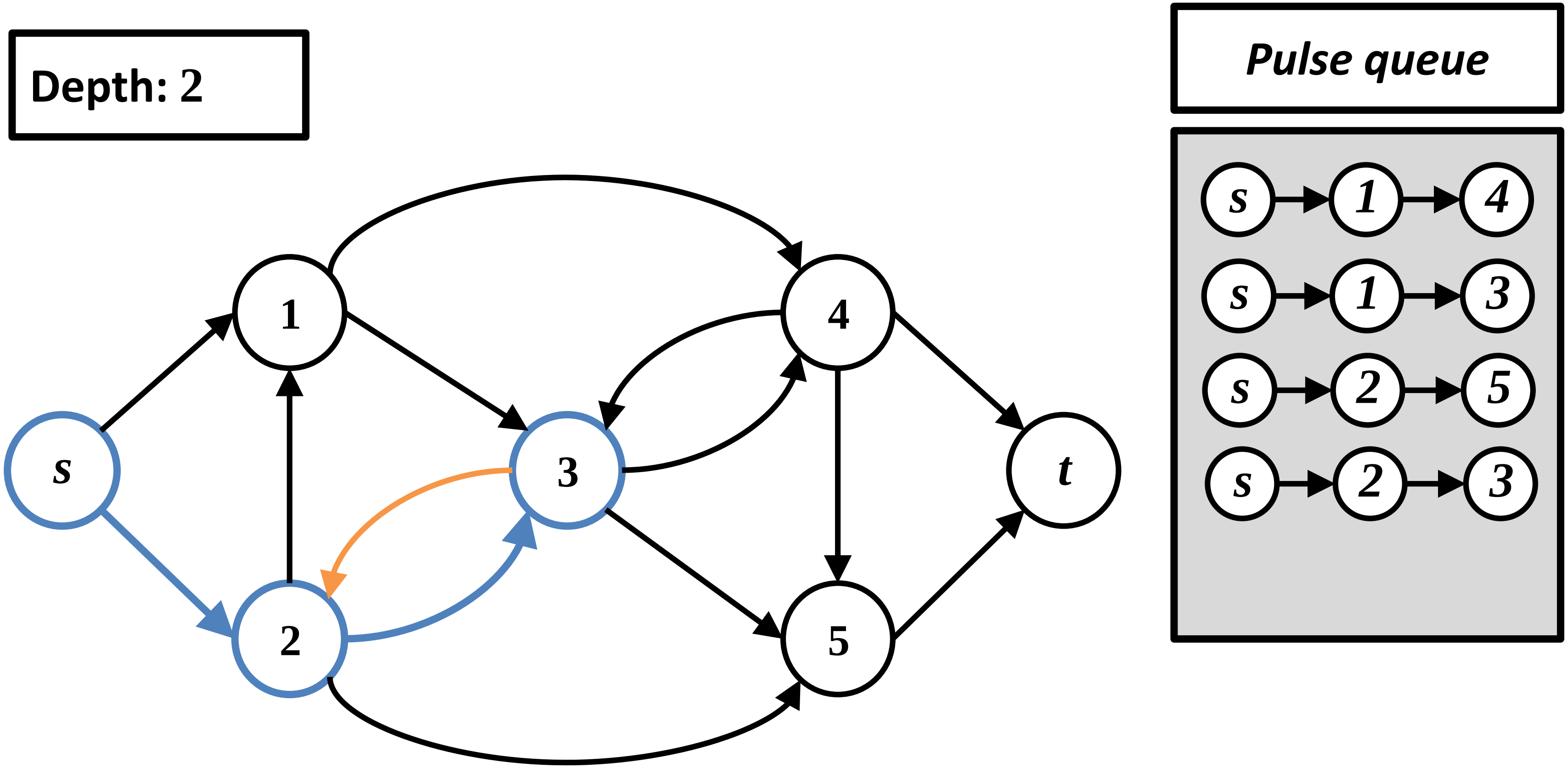


Pulse queue



Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

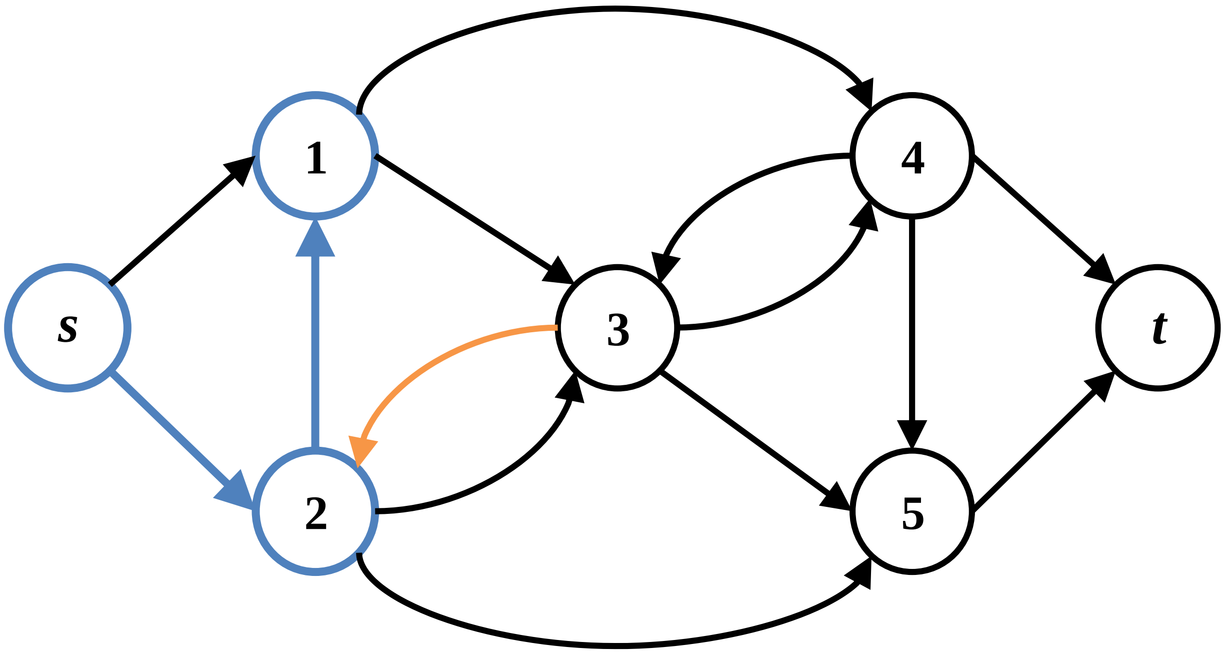
Pulse queue



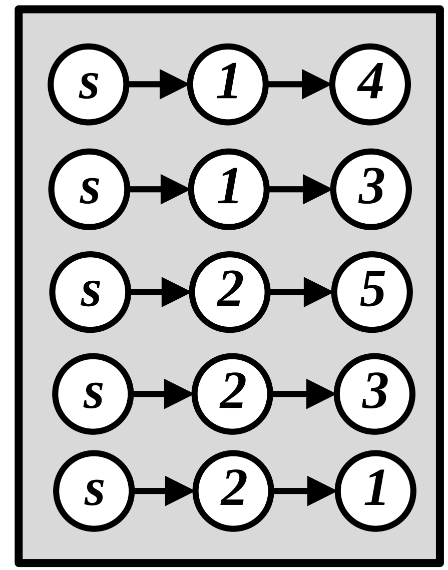
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



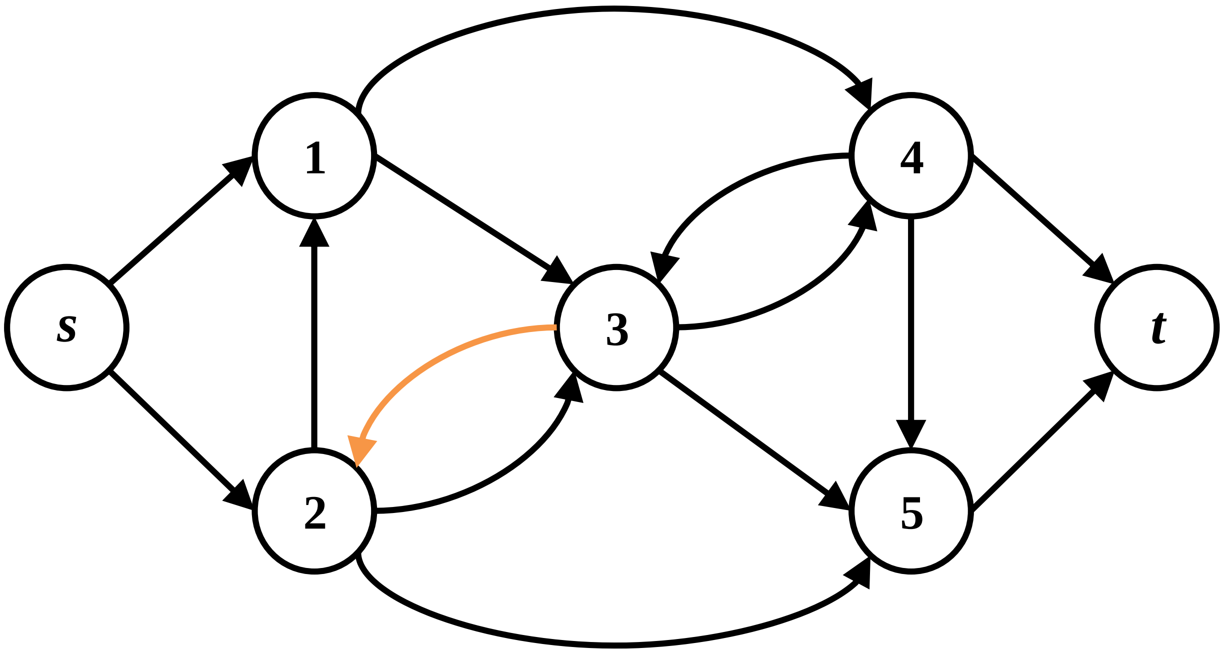
Pulse queue



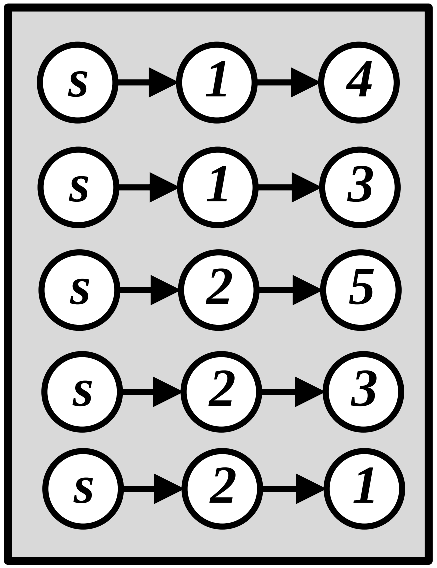
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

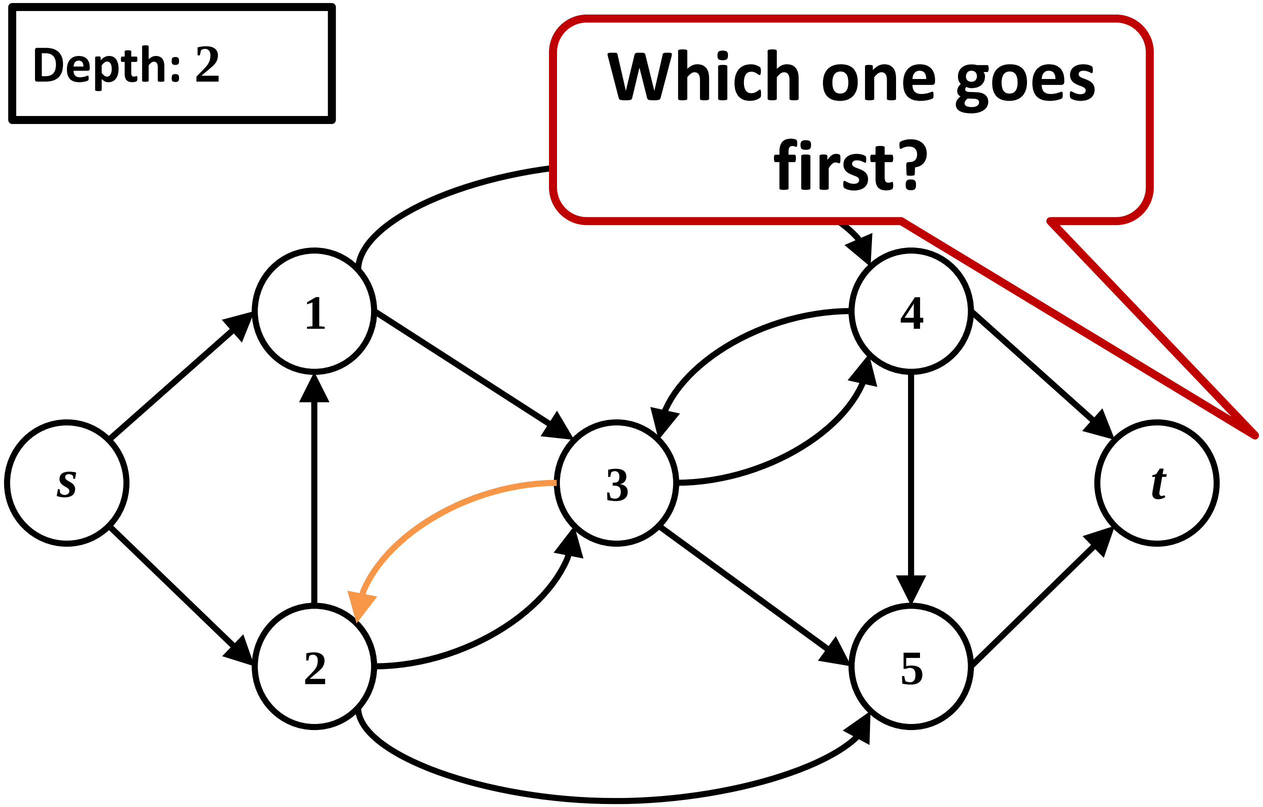
Depth: 2



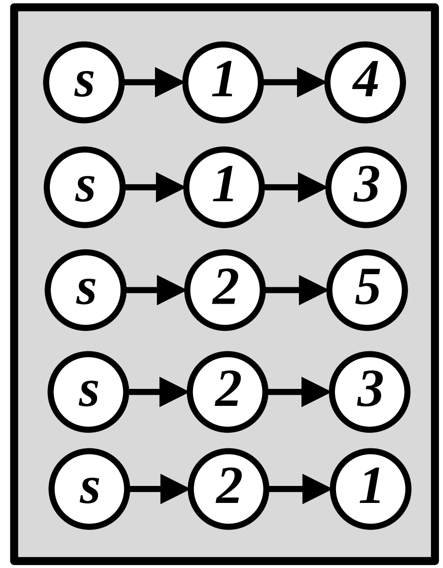
Pulse queue



Depth: 2



Pulse queue



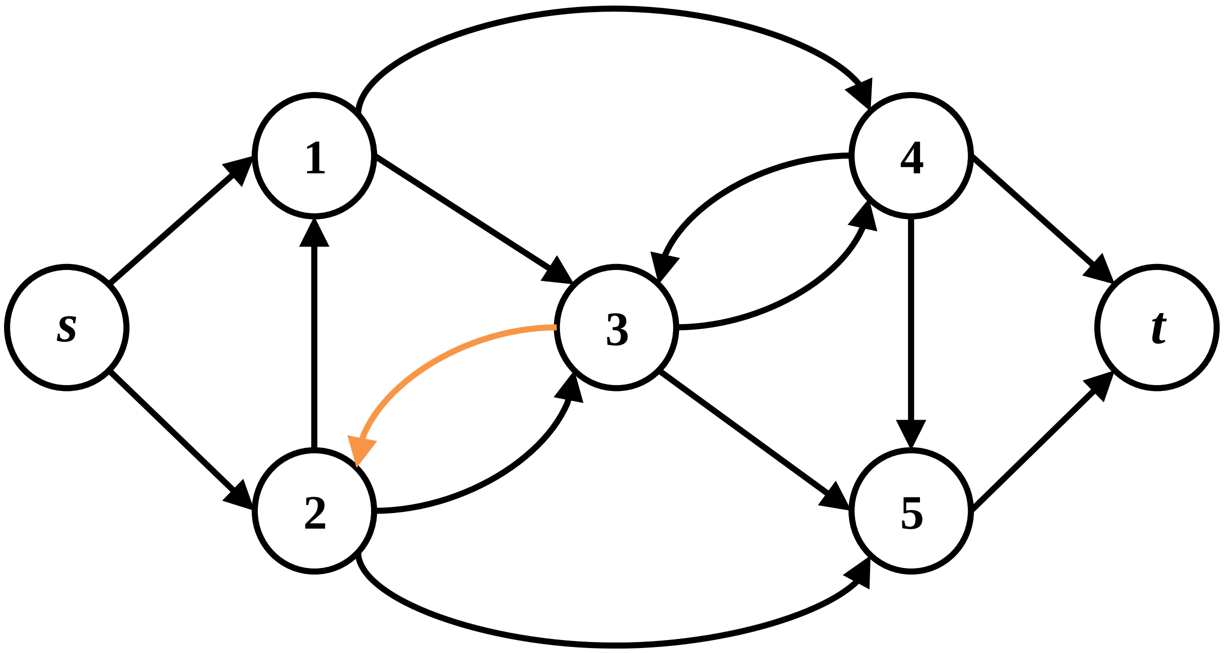
Treatment order:

$$min\, c(\mathcal{P})$$

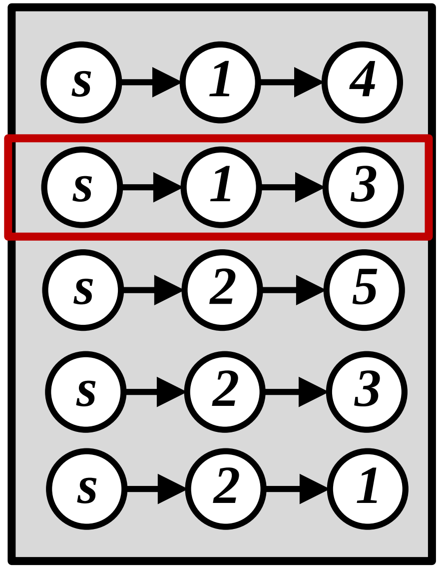
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



Pulse queue



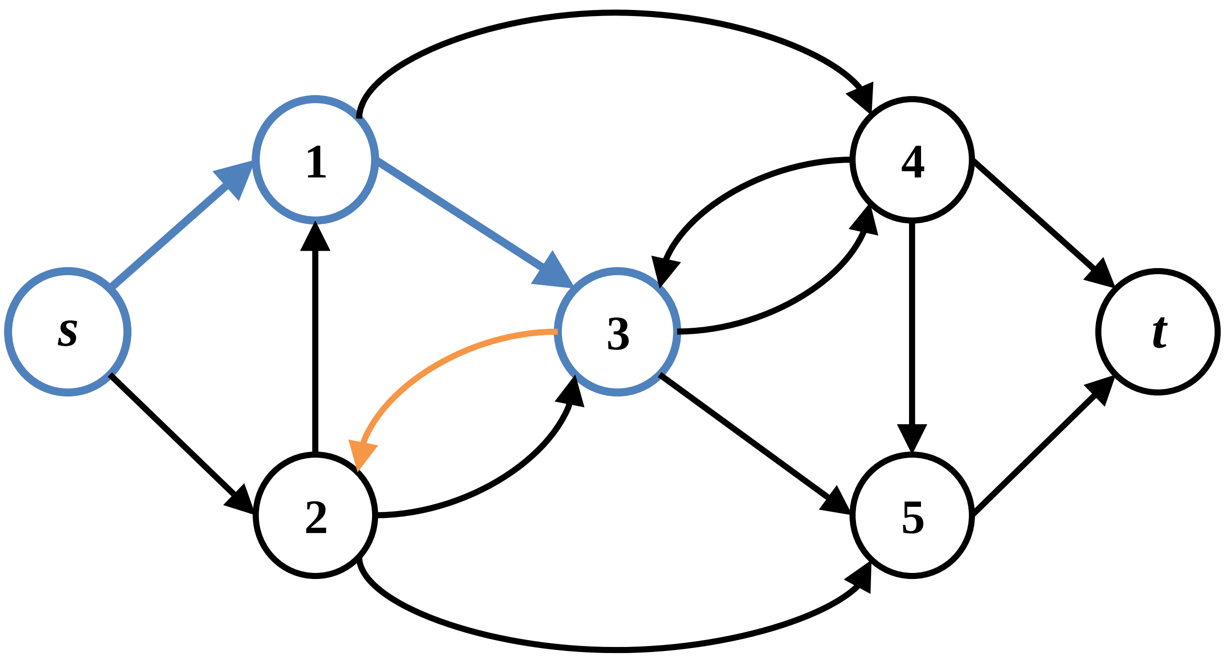
Treatment order:

$$min\, c(\mathcal{P})$$

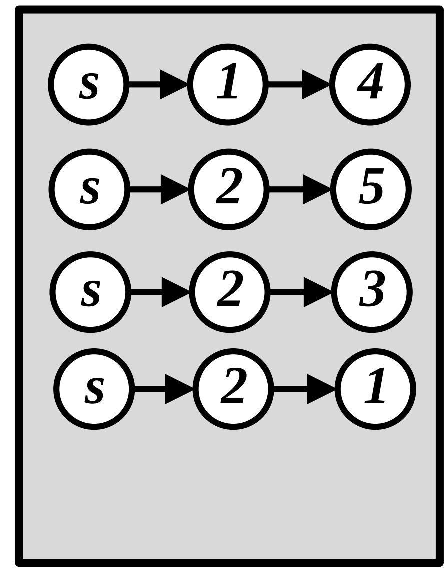
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



Pulse queue

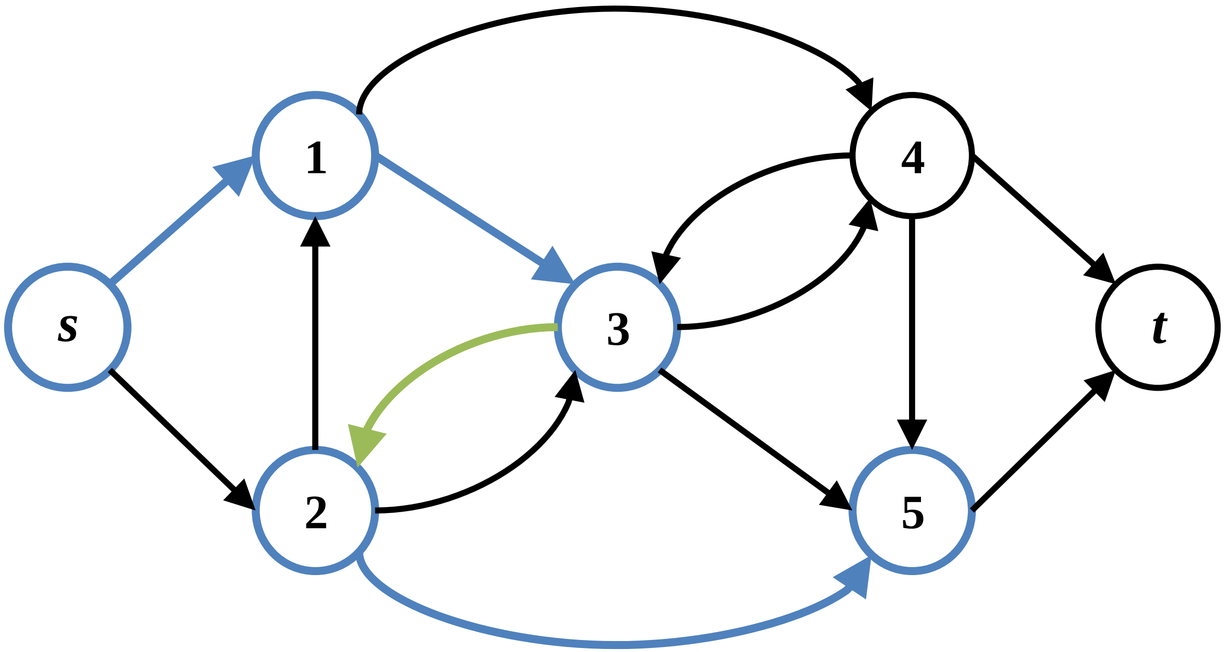


Treatment order:

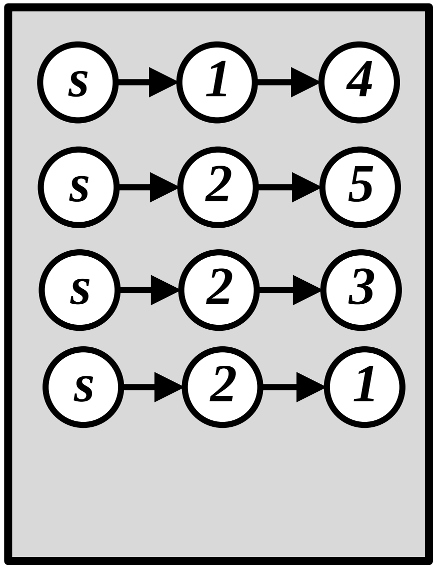
$$min\, c(\mathcal{P})$$

Pulse queue

Depth: 2



Pulse queue

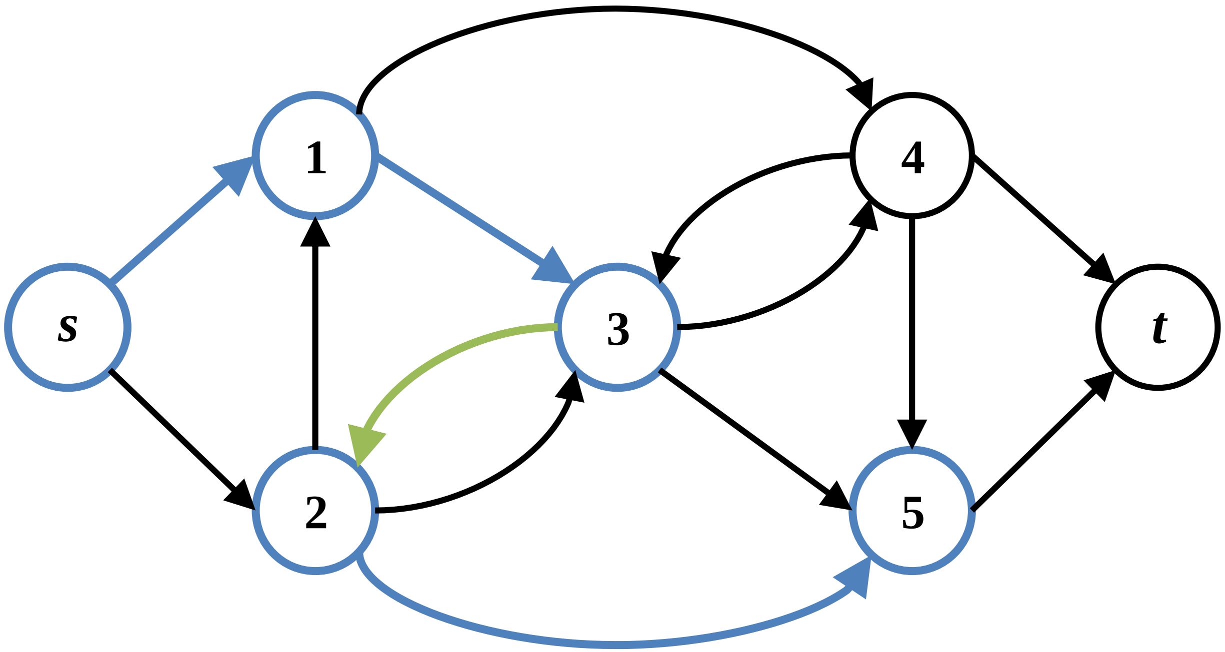


Treatment order:

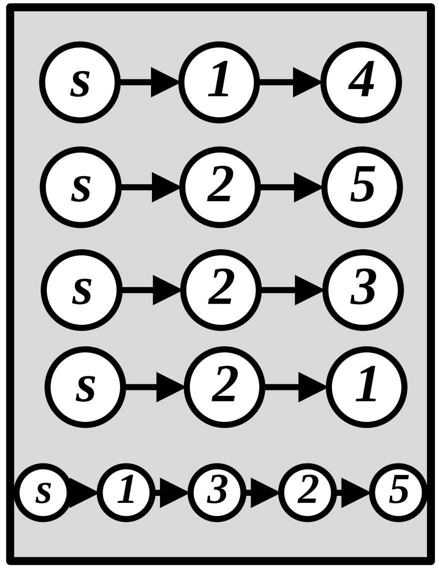
$$min\, c(\mathcal{P})$$

Pulse queue

Depth: 2



Pulse queue



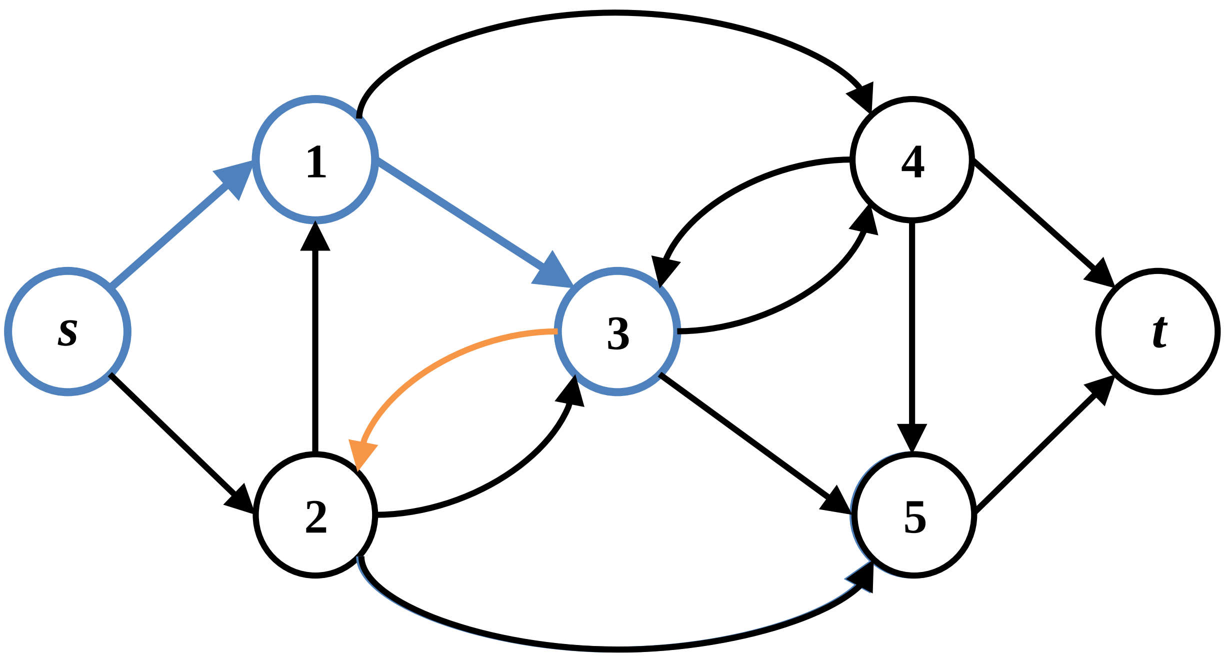
Treatment order:

$$min\, c(\mathcal{P})$$

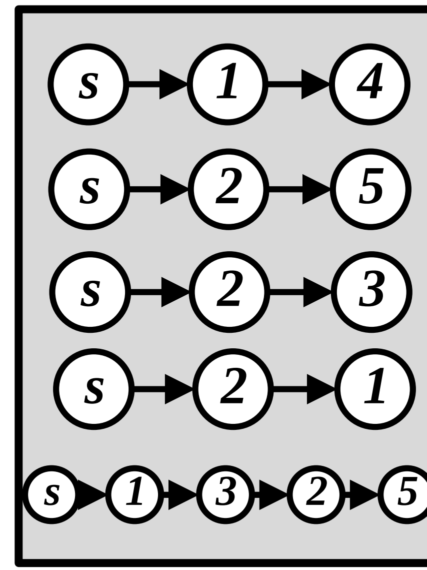
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



Pulse queue



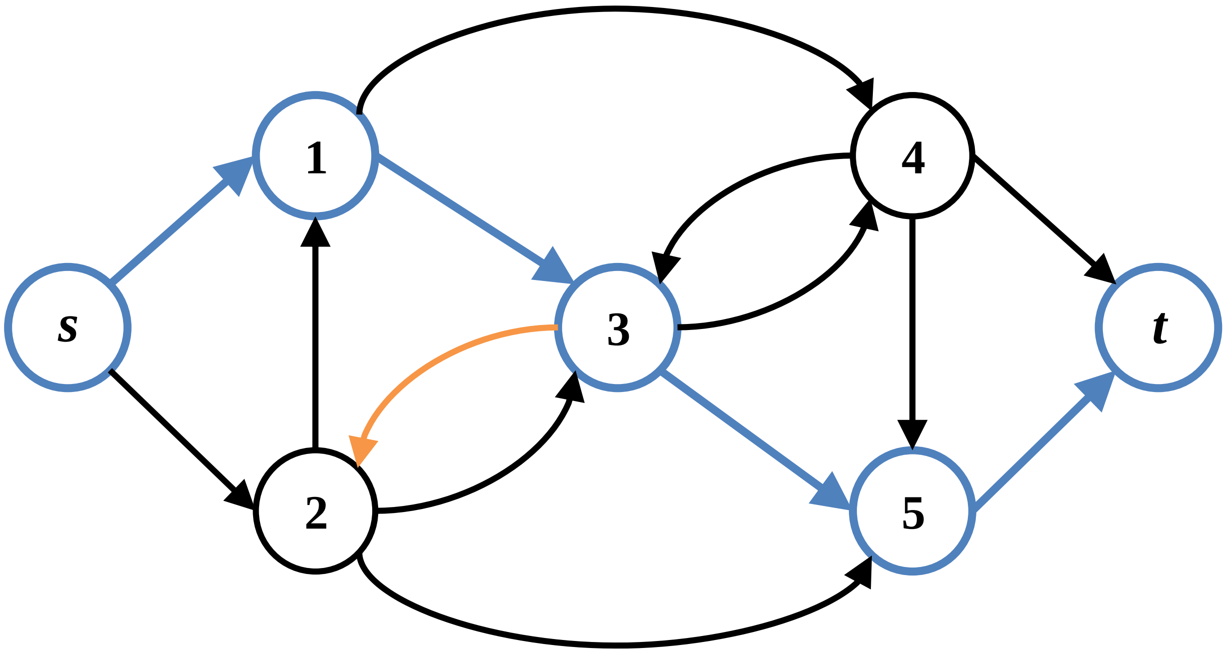
Treatment order:

$$min\, c(\mathcal{P})$$

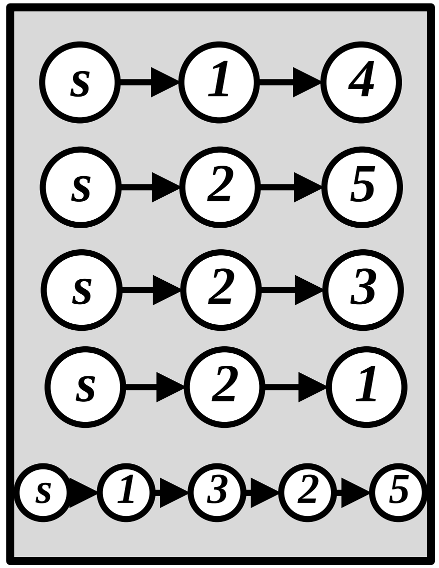
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



Pulse queue



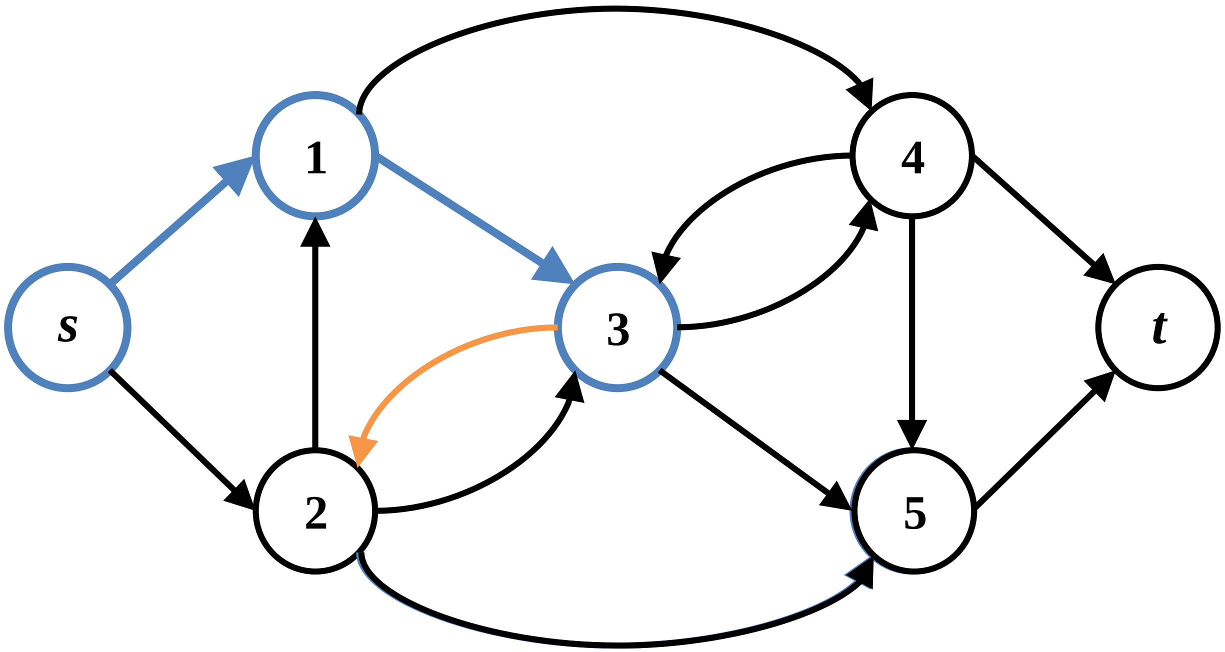
Treatment order:

$$min\, c(\mathcal{P})$$

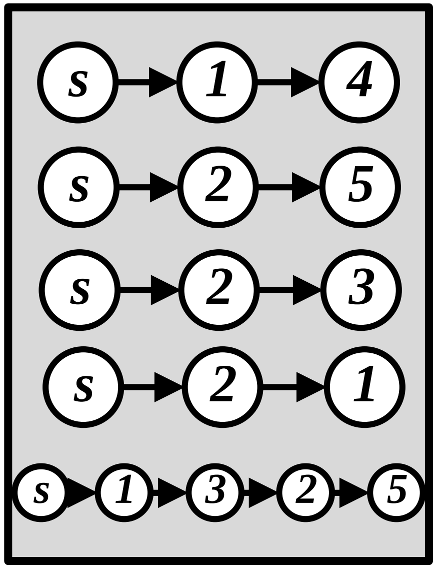
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



Pulse queue



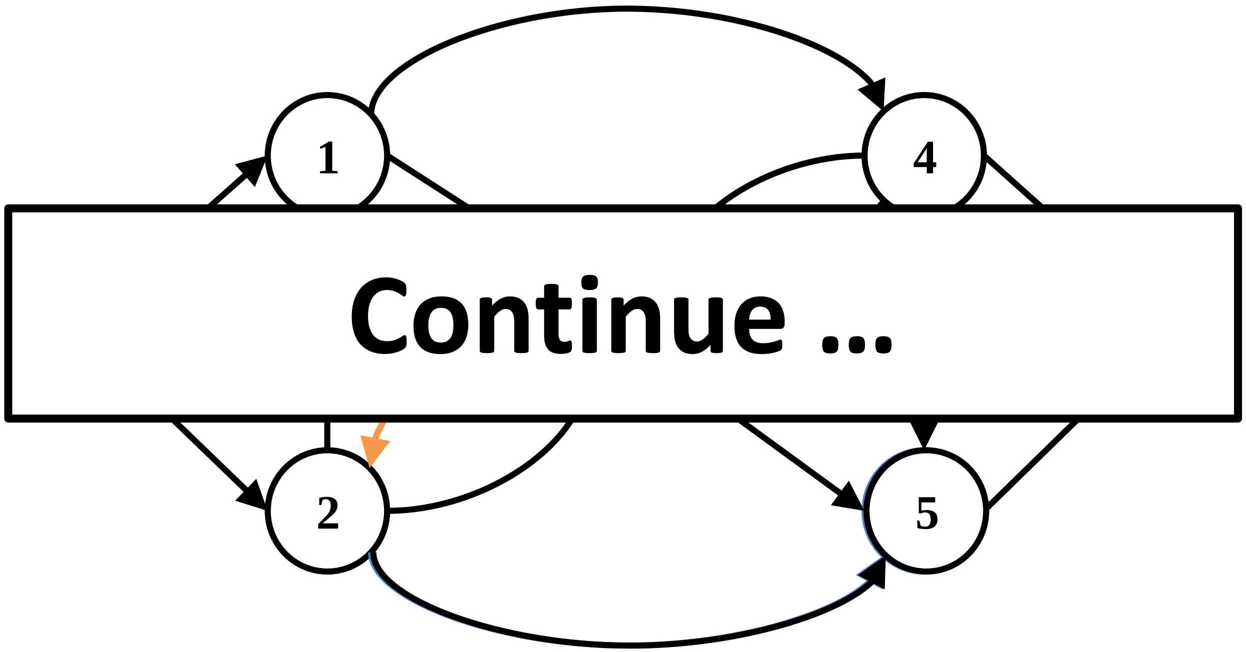
Treatment order:

$$min\, c(\mathcal{P})$$

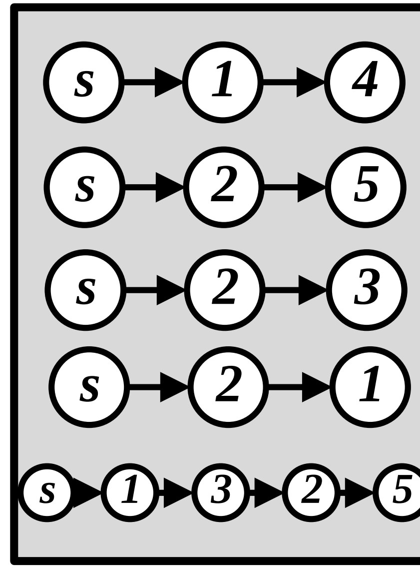
Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Pulse queue

Depth: 2



Pulse queue

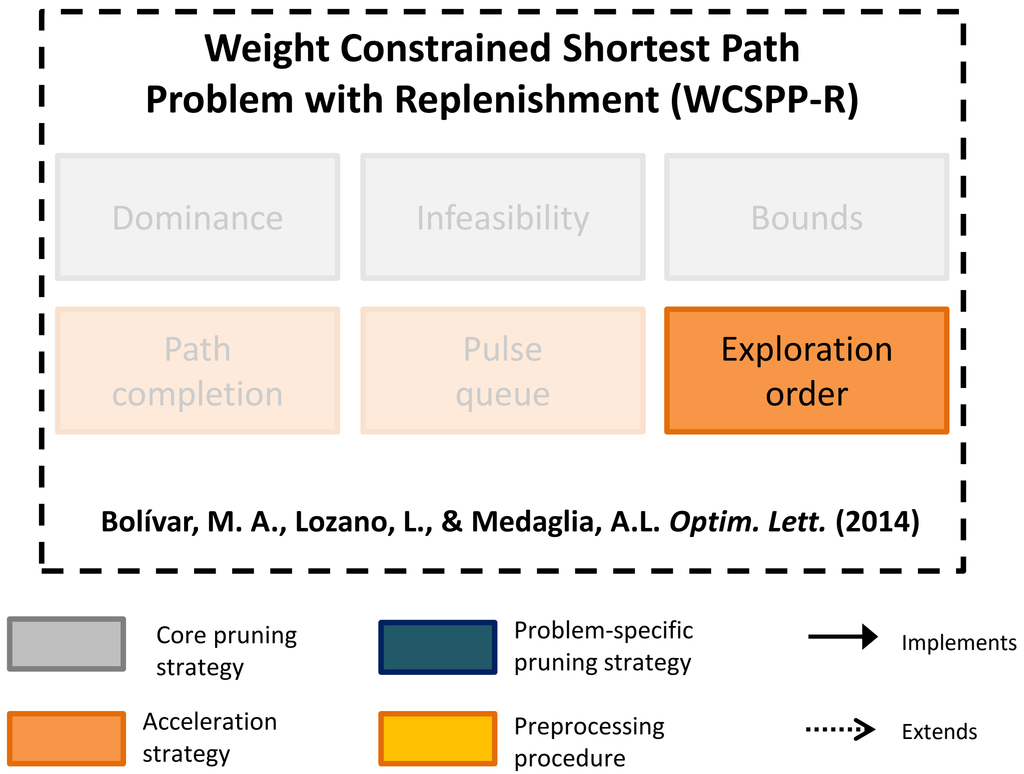


Treatment order:

$$min\, c(\mathcal{P})$$

Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)

Acceleration strategies

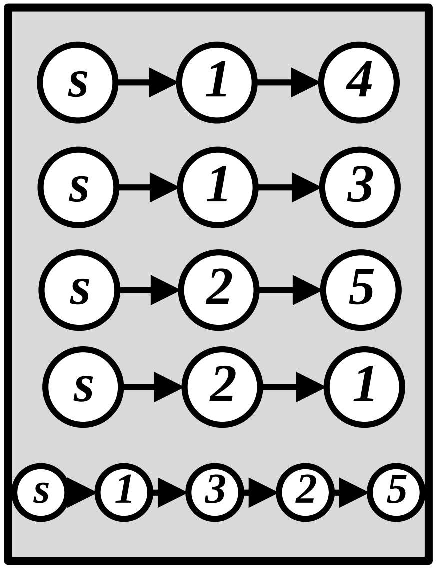


We propose to follow a best-promise exploration order

Promise $\psi(\cdot)$ is defined as:

$$\psi(\mathcal{P}_{si}) = c(\mathcal{P}_{si}) + c(\mathcal{P}_{ie}^{c})$$

Pulse queue



Treatment order:
$\min \psi(\mathcal{P}_{si})$

Universidad de los Andes
Colombia

Setup:

- Benchmark algorithm proposed by Smith et al. (2012)
- Pulse algorithm coded in Java and compiled in Eclipse SDK 3.6.1
- CPU: Intel Xeon X5450 @ 3.00 GHz 6.0GB of RAM (JVM)
- Exploration order: *Best Promise*
- Pulse depth: *2*

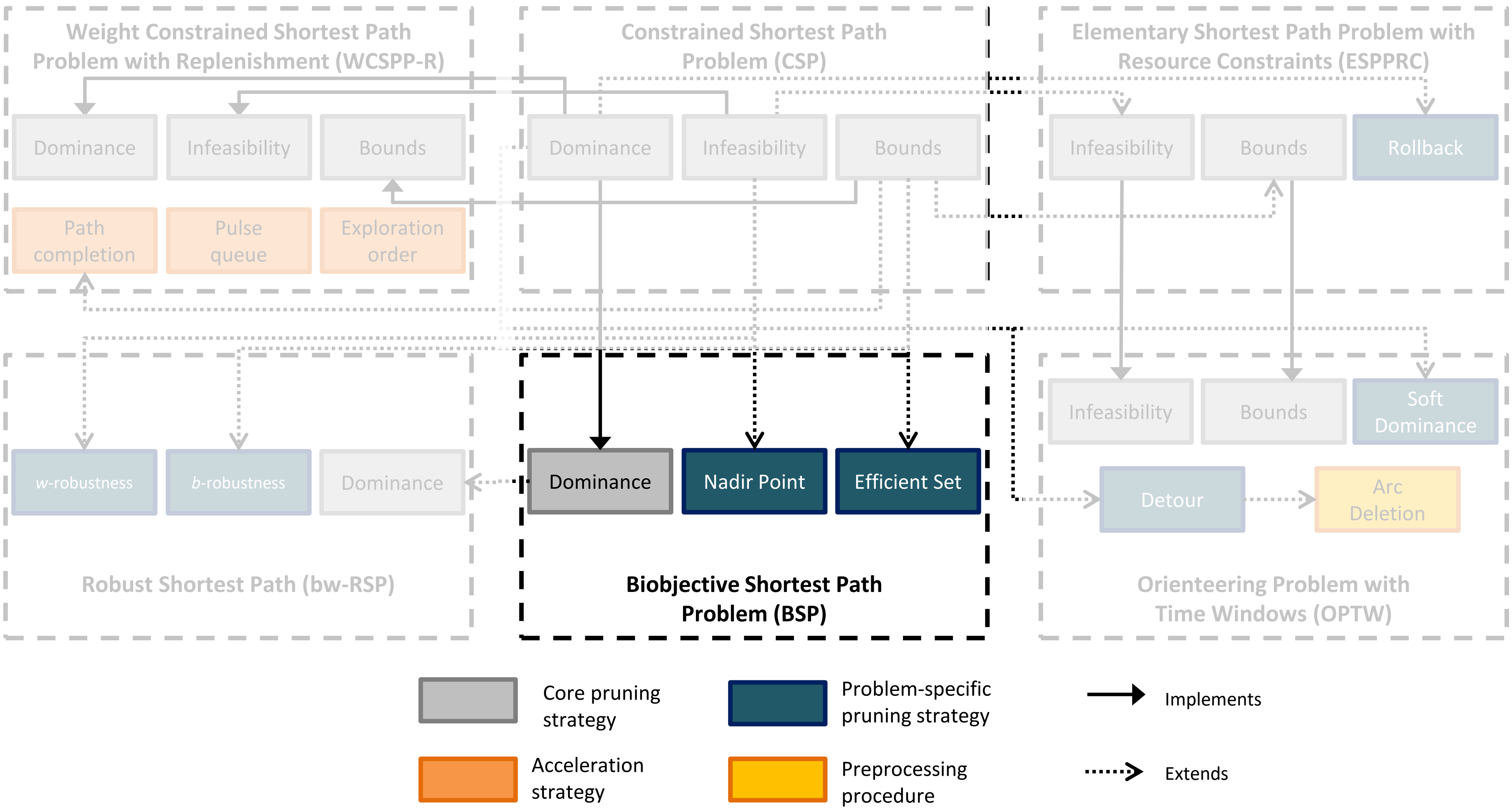
Computational experiments

Network	Nodes	Arcs	α	Time_{pre} (s)	LC/Java (s)	Pulse (s)	Avg. speedup
CAL California and Nevada	1,890,815	4,657,742	0	5.44	3.76	1.39	2.84
			0.1	5.17	2.04	1.10	6.95
			0.5	4.80	1.38	0.04	127.59
			0.9	4.60	0.07	<0.01	53.00
LKS Great Lakes	2,758,119	6,885,658	0	8.43	2.95	3.70	1.55
			0.1	11.04	3.35	3.26	4.38
			0.5	7.86	4.02	0.31	98.57
			0.9	6.71	0.47	0.05	27.83
E Eastern USA	3,598,623	8,778,114	0	12.73	2.69	4.42	1.26
			0.1	16.03	3.63	2.90	4.24
			0.5	13.35	2.32	0.16	55.12
			0.9	14.83	0.29	<0.01	219.64
W Western USA	6,262,104	15,248,146	0	57.68	11.99	11.66	1.59
			0.1	64.25	8.59	2.38	6.69
			0.5	62.96	5.04	0.56	100.81
			0.9	43.38	0.82	0.04	145.46
Overall avg.							40.36

Agenda

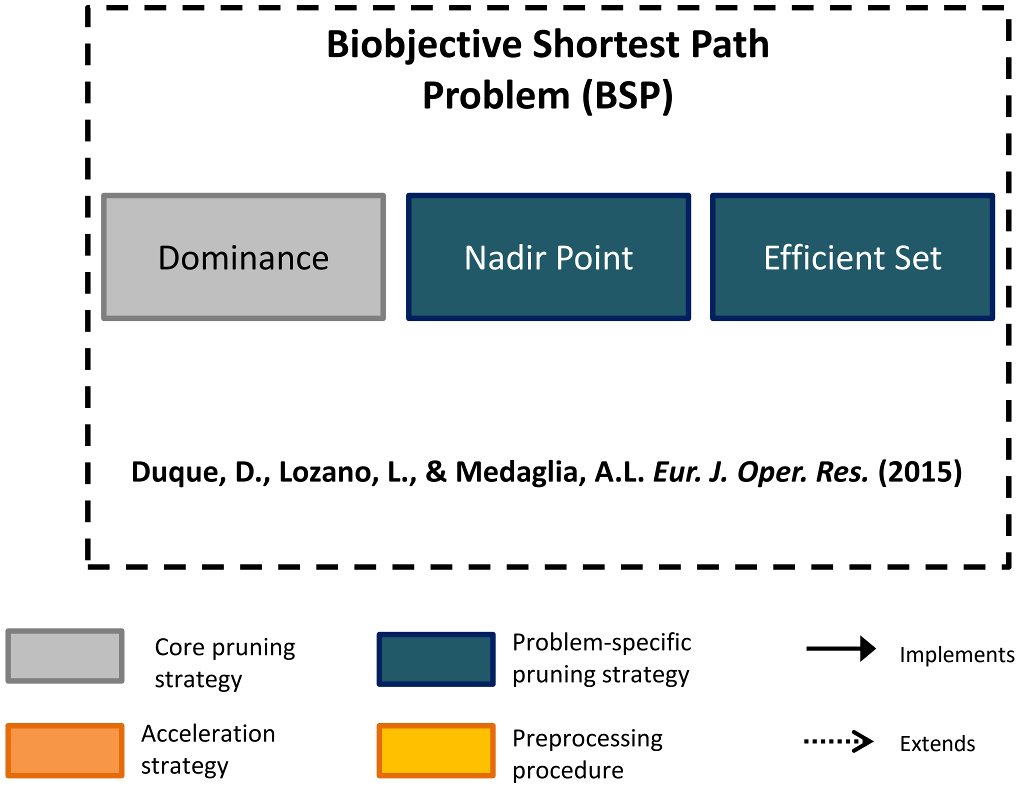
- Part I: fundamentals
- Part I: intuition
- Part II: extensions
  - Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)
  - **Biobjective Shortest Path Problem (BSP)**
  - Elementary Shortest Path Problem with Resource Constraints (ESPPRC)
  - Orienteering Problem with Time Windows (OPTW)
  - Robust Shortest Path (bw-RSP)
- Part III: applications
- Part IV: perspectives

Pulse Algorithm for Hard Shortest Path Problems



Biobjective Shortest Path Problem (BSP)

Pruning strategies



- Tung & Chew (1992)
- Raith & Ehrgott (2009)
- Raith (2010)
- Demeyer et al. (2013)

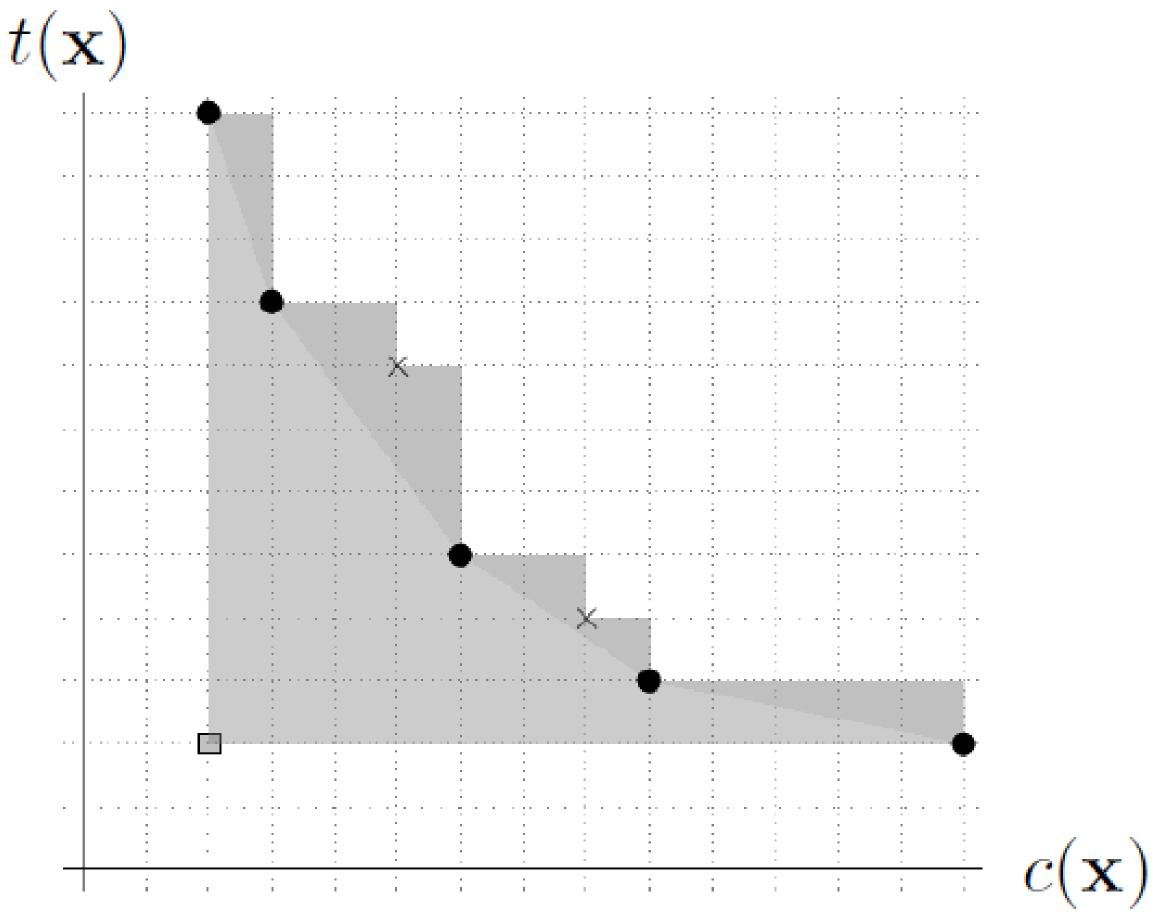
Biobjective Shortest Path Problem (BSP)

Problem statement

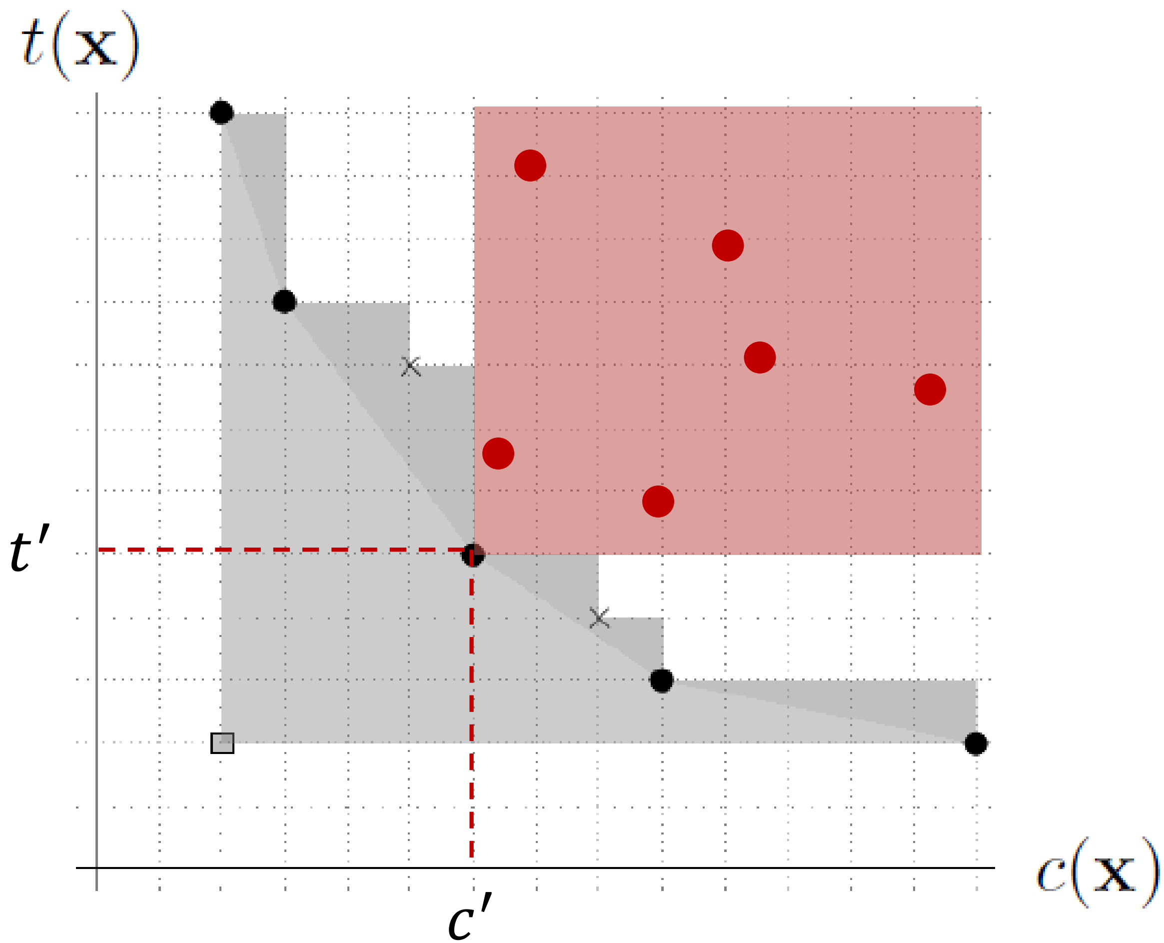
- The BSP is defined by:
  - Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$
    - $\mathcal{N} = \{v_1, \dots, v_i, \dots, v_n\}$
    - $\mathcal{A} = \{(i,j) | v_i \in \mathcal{N}, v_j \in \mathcal{N},\ i \neq j\}$
  - Find a set of efficient solutions (paths) starting at node v_s and ending at node v_e
  - Nonnegative weights c_{ij} and t_{ij} are the cost and travel time of traversing arc $(i, j) \in \mathcal{A}$
  - Two objectives $c(\mathbf{x})$ and $t(\mathbf{x})$

Biobjective Shortest Path Problem (BSP)

Problem statement

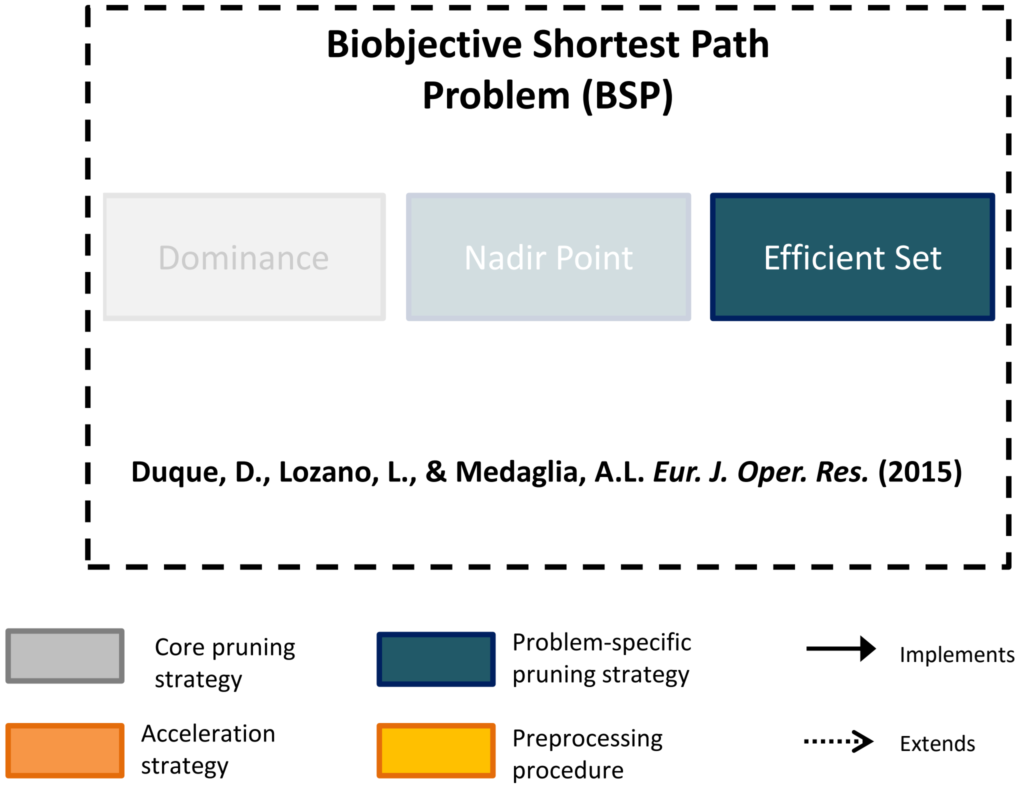


Biobjective Shortest Path Problem (BSP)
Problem statement



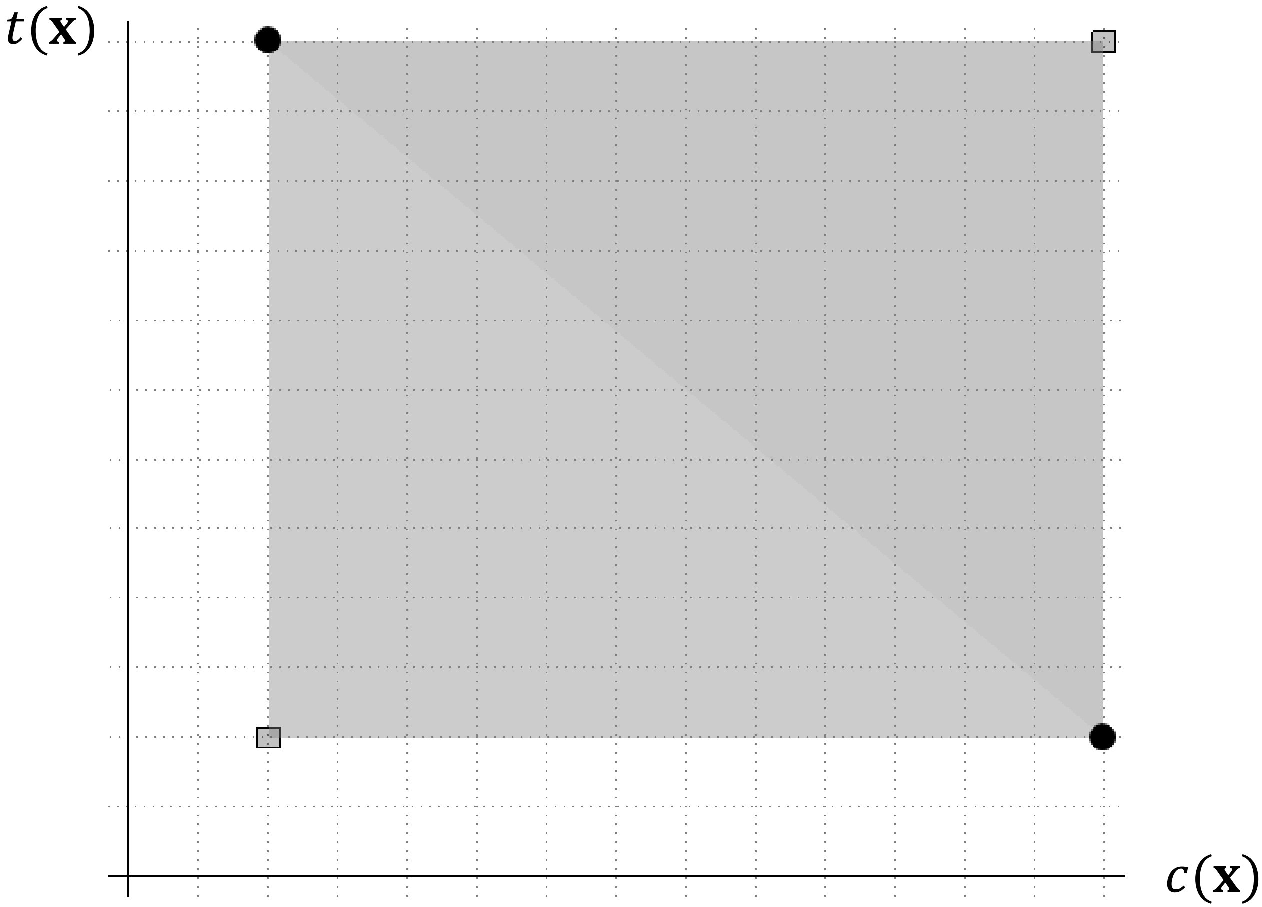
Biobjective Shortest Path Problem (BSP)

Pruning strategies



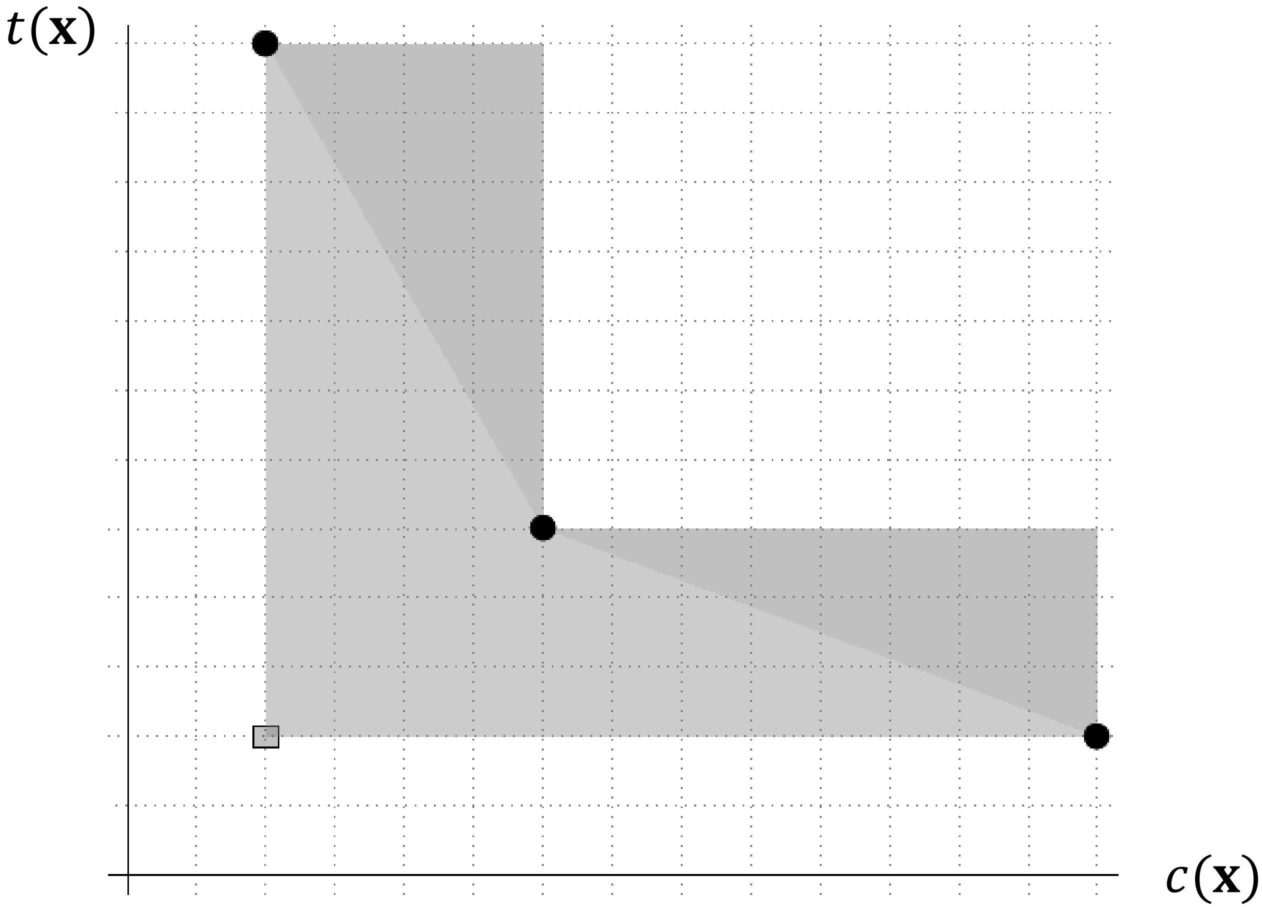
Biobjective Shortest Path Problem (BSP)

Efficient set pruning



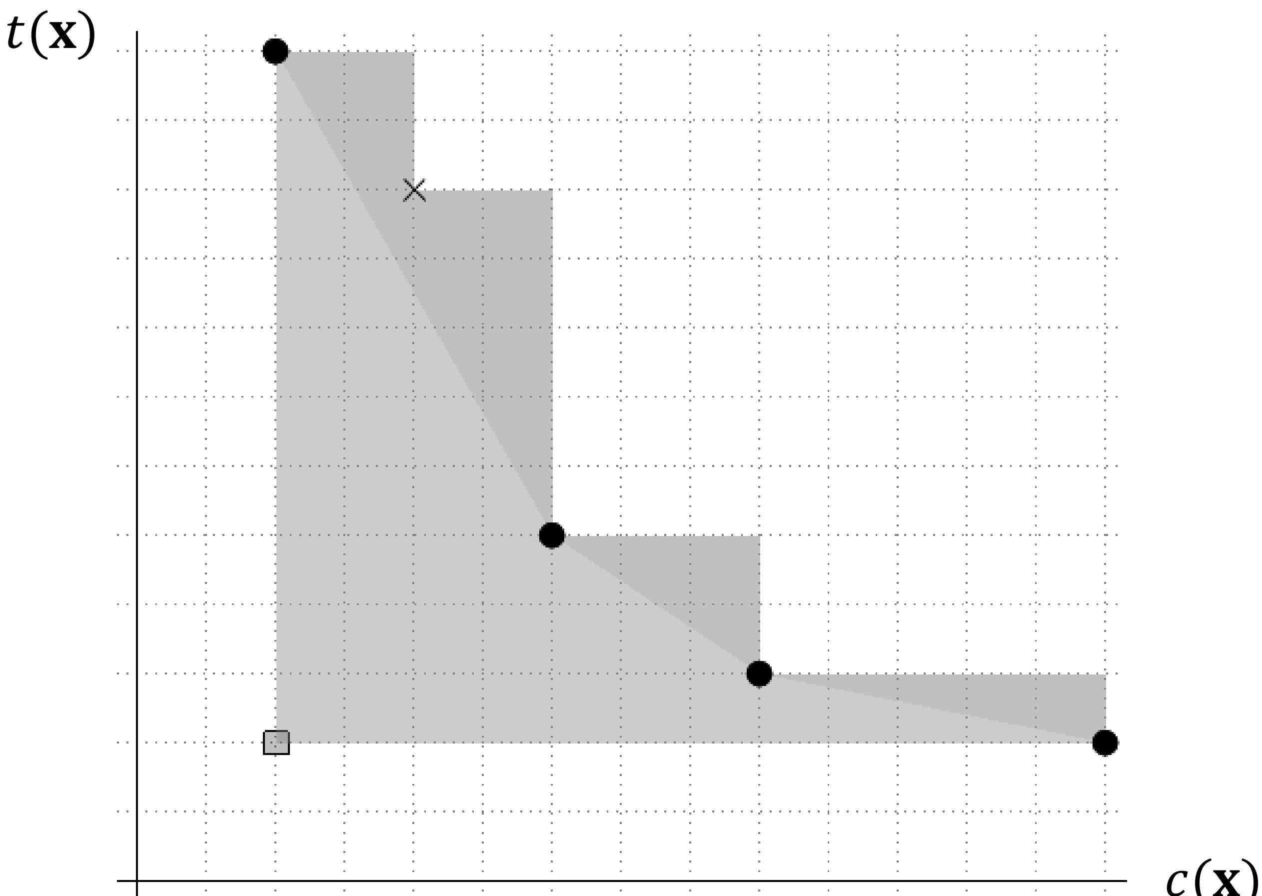
Biobjective Shortest Path Problem (BSP)

Efficient set pruning



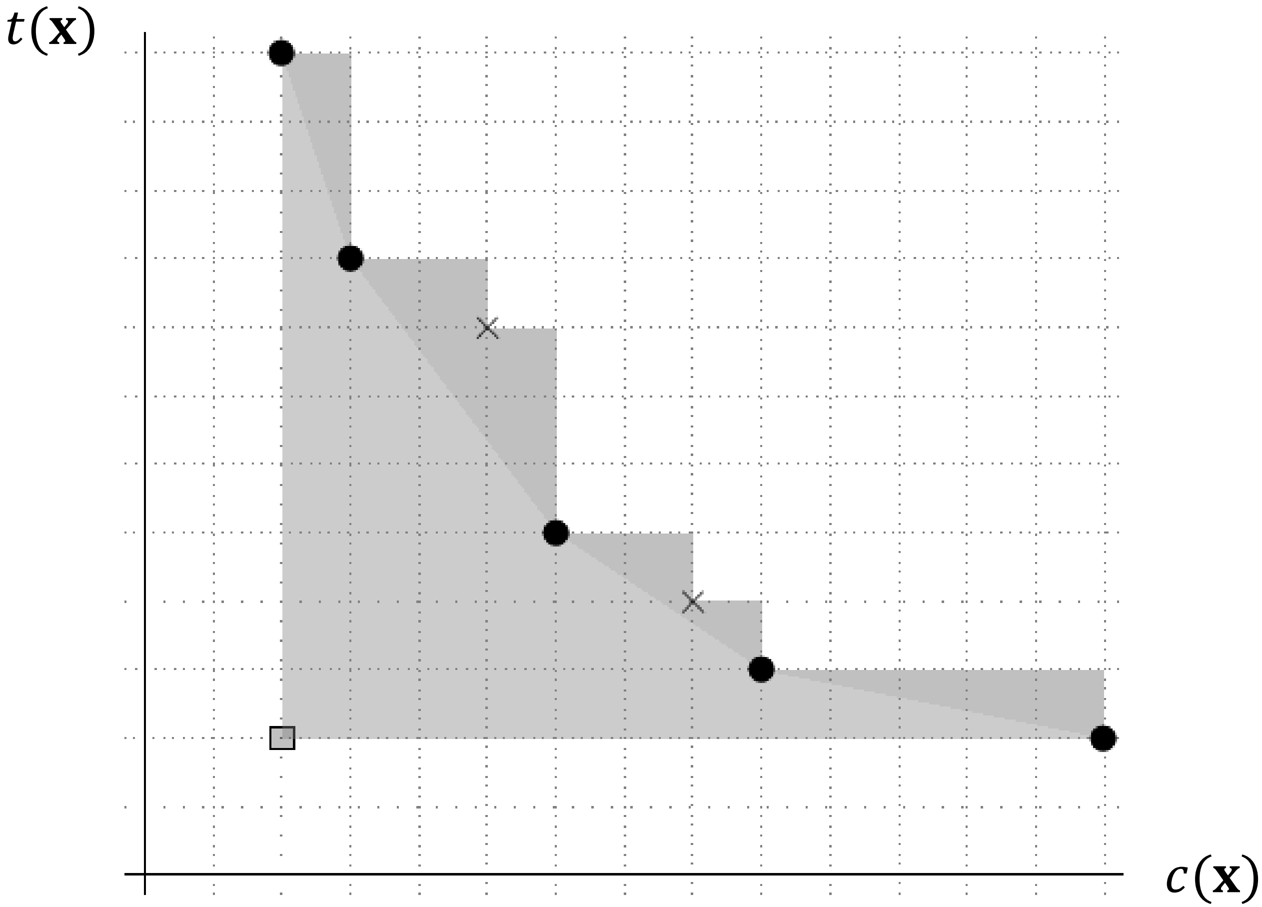
Biobjective Shortest Path Problem (BSP)

Efficient set pruning



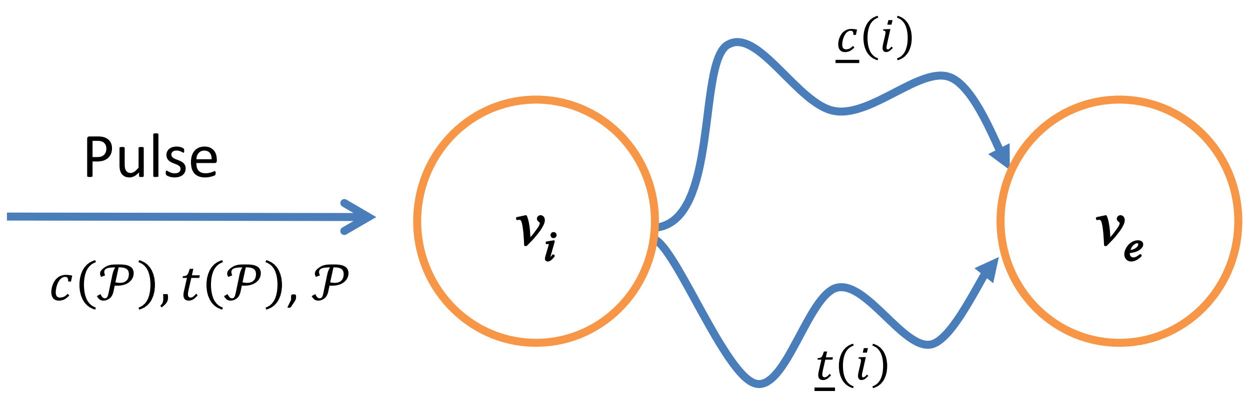
Biobjective Shortest Path Problem (BSP)

Efficient set pruning



Biobjective Shortest Path Problem (BSP)

Efficient set pruning



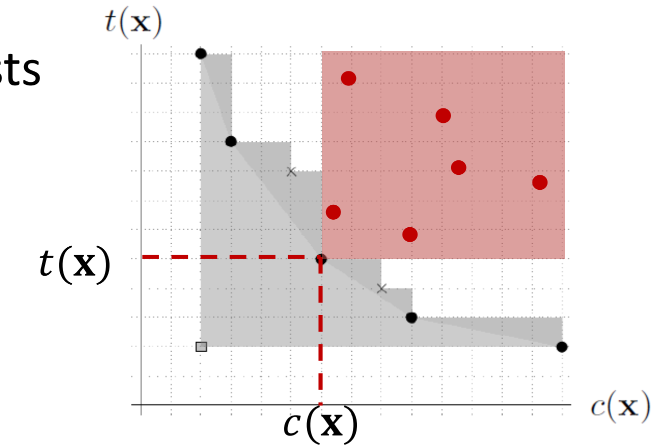
An incoming pulse to v_i is pruned if it exists

$\mathbf{x} \in X_E$ such that:

$$c(\mathcal{P}) + \underline{c}(i) \geq c(\mathbf{x})$$

and

$$t(\mathcal{P}) + \underline{t}(i) \geq t(\mathbf{x})$$



Biobjective Shortest Path Problem (BSP)

Computational experiments

Setup:

- Benchmark algorithm proposed by Raith (2010)
- Pulse and benchmark algorithms are coded in Java and compiled in Eclipse SDK 4.3.0
- CPU: Intel Core i7 (2 cores) Duo @ 1.90GHz 6GB RAM for JVM
- The amount of labels is set to 20
- Arcs are sorted according to the sum of both lower bounds of the arc's head node
- There is a computational time limit of 3,600 seconds

Biobjective Shortest Path Problem (BSP)

Computational experiments

Table 2: Computational results over real road networks from the 9-th DIMACS challenge.

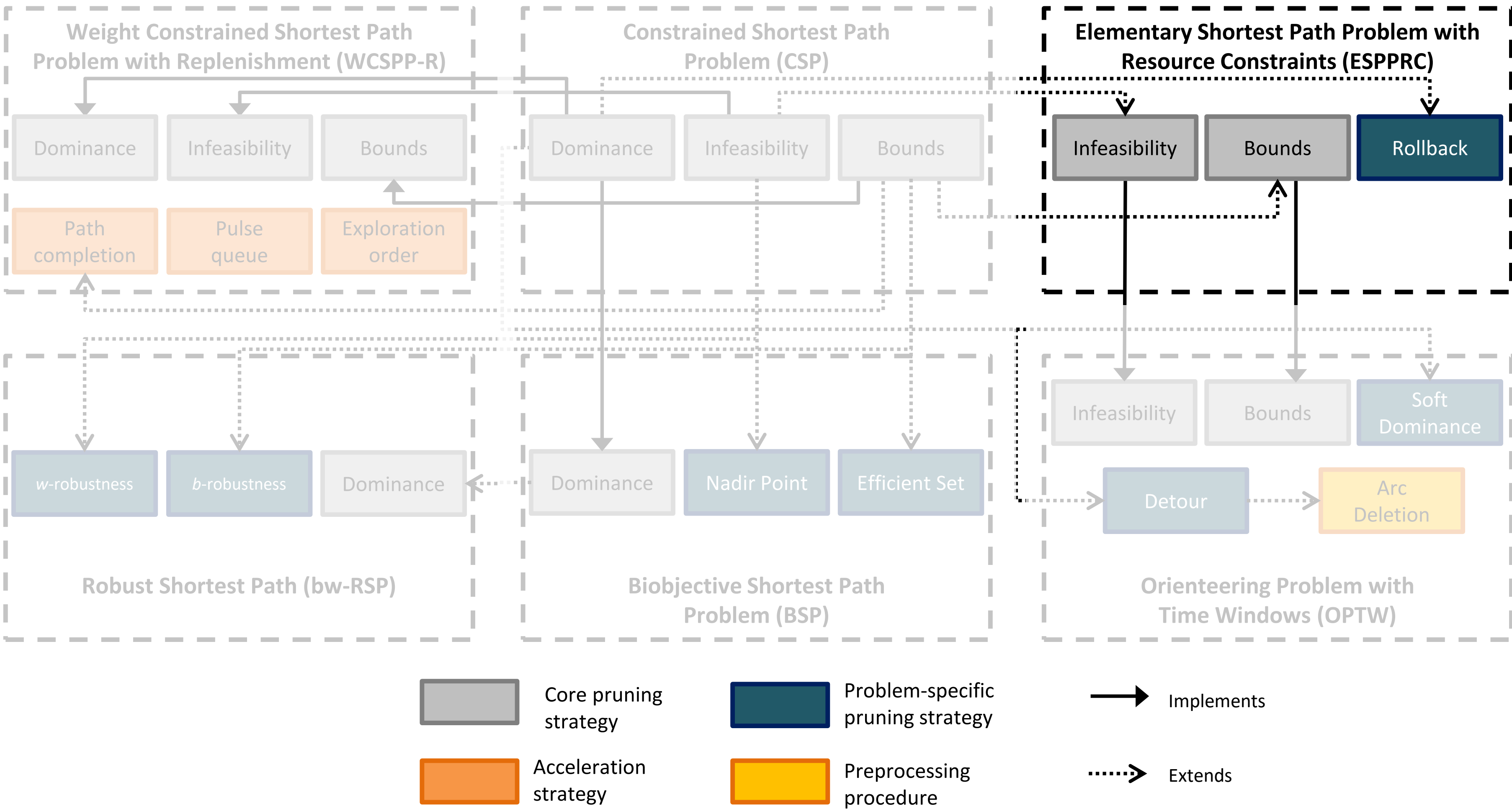
Cluster	n	Nodes	Arcs	Average	bLSET		Pulse		Geometric mean	# instances pulse
				$\lvert\mathcal{Z}_N\rvert$	Average time (s)	Solved	Average time (s)	Solved	of speedups	is faster
NY-G1	10			34.10	62.39	10	**0.32**	10		9
NY-G2	10	264,346	733,846	147.40	301.16	10	**52.32**	10	7.25	10
NY-G3	10			422.70	**881.26**	10	1367.66*	7		4
BAY-G1	10			8.80	6.78	10	**0.16**	10		4
BAY-G2	10	321,270	800,172	49.90	55.24	10	**5.70**	10	4.28	10
BAY-G3	10			171.80	317.43	10	**105.55**	10		8
COL-G1	10			18.20	7.30	10	**0.20**	10		8
COL-G2	10	435,666	1,057,066	87.10	**233.03**	10	381.94*	9	9.61	8
COL-G3	10			328.40	865.76	10	**508.76**	10		7
FLA-G1	10			14.70	330.12	10	**0.35**	10		9
FLA-G2	10	1,070,376	2,712,798	94.10	566.15*	9	**347.91**	10	45.89	10
FLA-G3	10			552.30	2627.43*	4	**888.59***	9		8
NW-G1	10			39.00	260.73	10	**1.99**	10		10
NW-G2	10	1,207,945	2,840,208	124.20	1109.98*	8	**81.96**	10	21.70	9
NW-G3	10			281.60	1443.66*	8	**438.54***	9		9
						139/150		144/150		123/150

[*] Average time is calculated with a computational time of 3,600 seconds for unsolved instances

Agenda

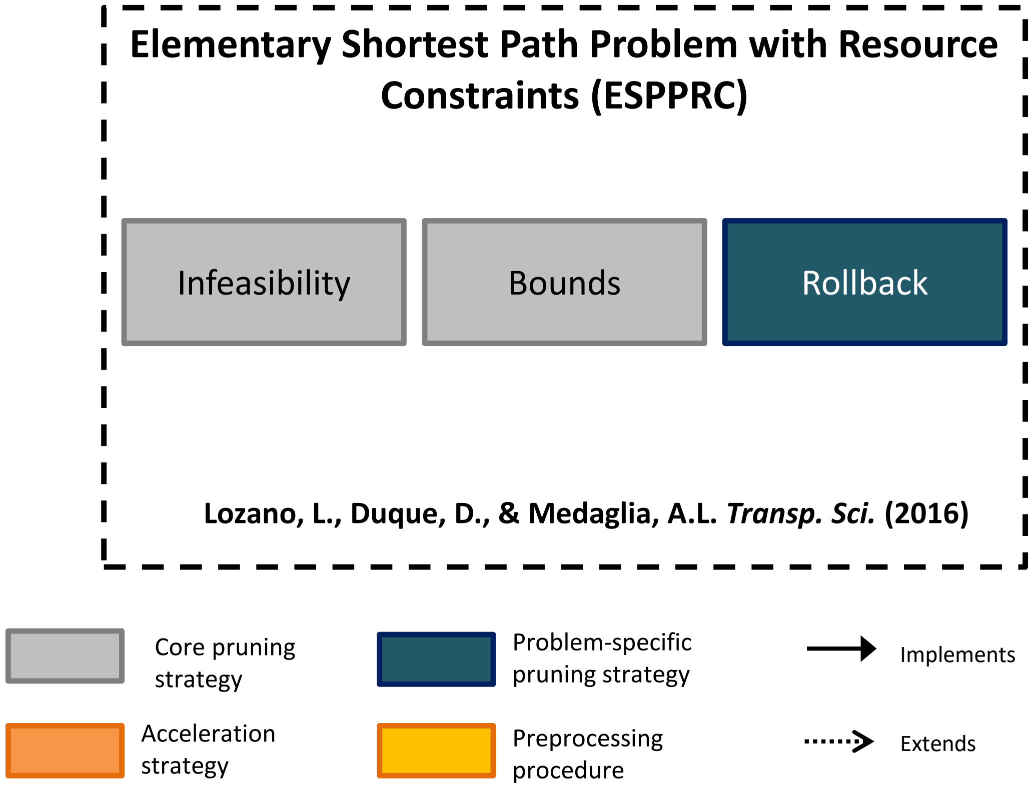
- Part I: fundamentals
- Part II: intuition
- Part III: extensions
  - Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)
  - Biobjective Shortest Path Problem (BSP)
  - **Elementary Shortest Path Problem with Resource Constraints (ESPPRC)**
  - Orienteering Problem with Time Windows (OPTW)
  - Robust Shortest Path (bw-RSP)
- Part IV: applications
- Part V: perspectives

Pulse Algorithm for
Hard Shortest Path Problems



Elementary Shortest Path Problem with Resource Constraints

Pruning strategies



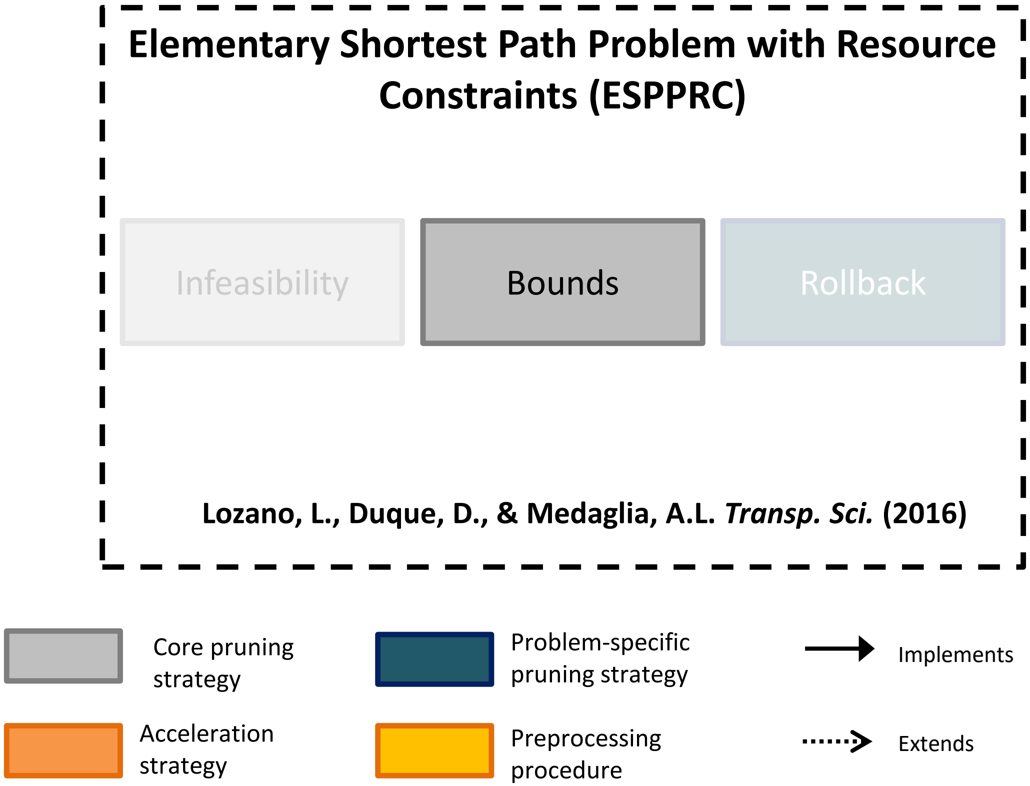
- Righini & Salani (2008)
- Desaulniers, Lessard & Hadjar (2008)
- Baldacci & Mingozzi (2011)
- Contardo, Desaulniers & Lessard (2008)

Elementary Shortest Path Problem with Resource Constraints
Problem statement

- The ESPPRC is defined by:
  - Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$
    - $\mathcal{N} = \{v_1, \dots, v_i, \dots, v_n\}$
    - $\mathcal{A} = \{(i,j) | v_i \in \mathcal{N}, v_j \in \mathcal{N},\ i \neq j\}$
  - Find a minimum cost path starting at node v_s and ending at node v_e
  - Cost c_{ij} for traversing arc $(i,j) \in \mathcal{A}$
  - Nonnegative weight t_{ij} is the travel time between nodes
  - Nonnegative demand d_i for visiting node v_i
  - Resource constraints (e.g., time windows, vehicle capacity, maximum travel time)

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning



Discarding Paths Using Bounds:

- Save a primal bound $\bar{C}$
- Calculate a bound on the minimum cost $\underline{c}(i)$ to the final node
- Compare cumulative cost plus $\underline{c}(i)$ with $\bar{C}$

An incoming pulse to v_i is pruned if:

$$c(\mathcal{P}) + \underline{c}(i) \geq \bar{c}$$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

Discarding Paths Using Bounds:

- Save a primal bo
- Calculate a bou inal node
- Compare cumul

ESPPRC for every node!

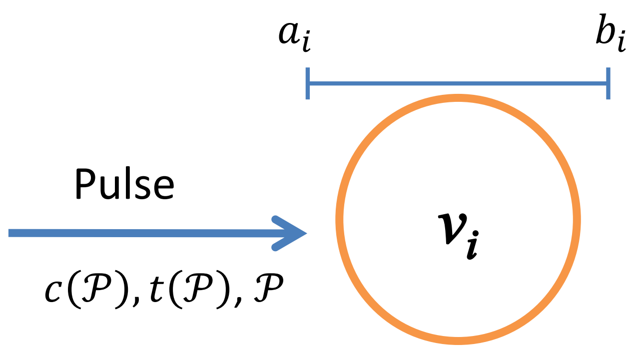
An incoming pulse to v_i is pruned if:

$$c(\mathcal{P}) + \underline{c}(i) \geq \bar{c}$$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

$T = 100$

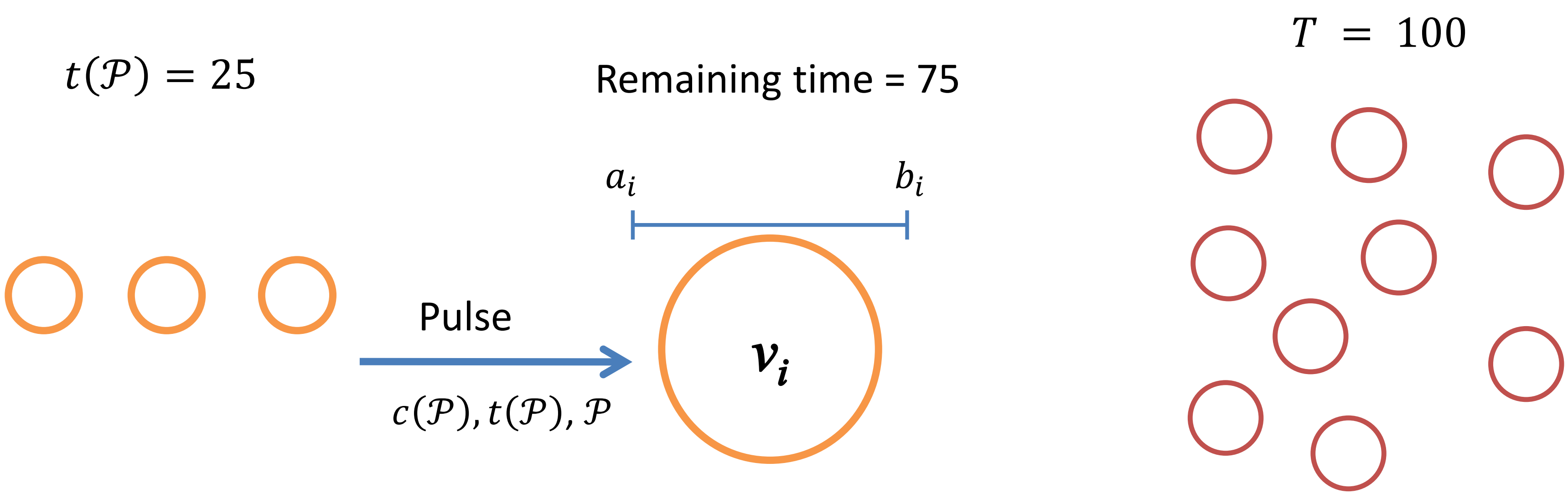


An incoming pulse to v_i is pruned if:

$$c(\mathcal{P}) + \underline{c}(i) \geq \bar{c}$$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

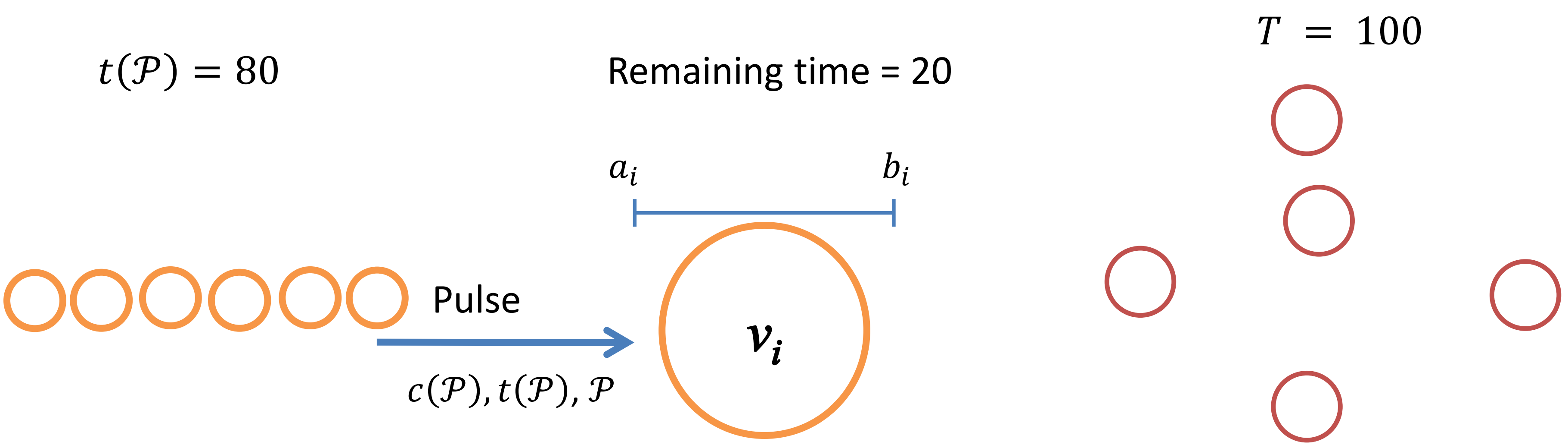


An incoming pulse to v_i is pruned if:

$$c(\mathcal{P}) + \underline{c}(i) \geq \bar{c}$$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

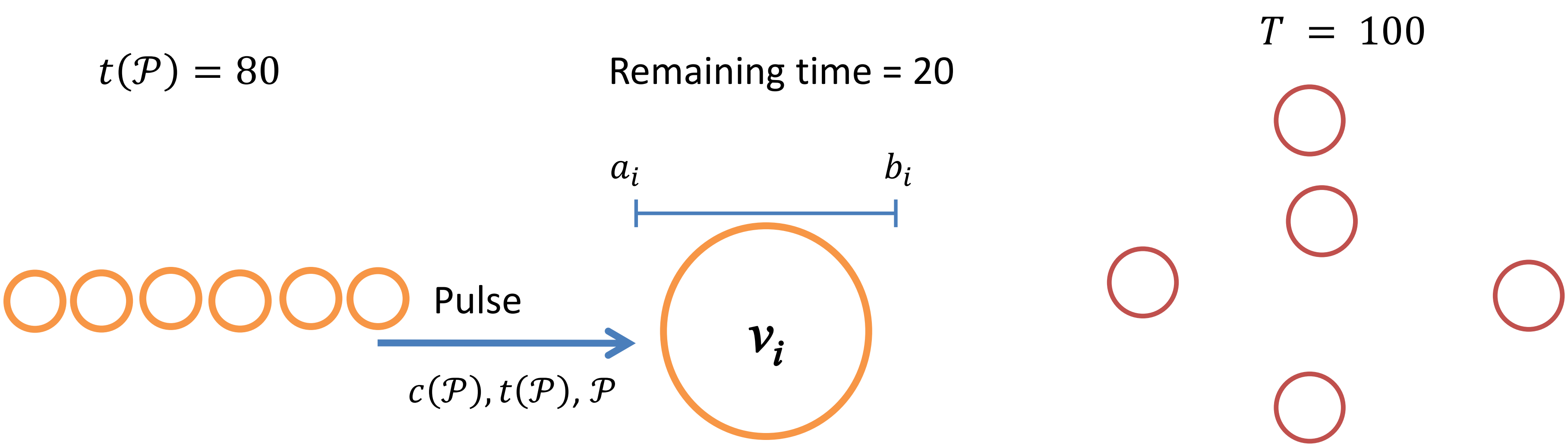


An incoming pulse to v_i is pruned if:

$$c(\mathcal{P}) + \underline{c}(i) \geq \bar{c}$$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning



An incoming pulse to v_i is pruned if:

$$c(\mathcal{P}) + \boxed{\underline{c}(i, t(\mathcal{P}))} \geq \bar{c}$$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

Find ESPPRC for every node given a consumed resource τ

Lower bounds $\underline{c}(i,\tau)$

$$\boxed{\tau = \bar{t}}$$

$\bar{t} \triangleq T$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

Find ESPPRC for every node given a consumed resource τ

↓

Lower bounds $\underline{c}(i,\tau)$

$$\boxed{\tau = \bar{t} - \delta} \leftarrow \boxed{\tau = \bar{t}}$$

$\bar{t} \triangleq T$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

Find ESPPRC for every node given a consumed resource τ

Lower bounds $\underline{c}(i,\tau)$

$\tau = \bar{t} - 2\delta$ ← $\tau = \bar{t} - \delta$ ← $\tau = \bar{t}$

$\bar{t} \triangleq T$

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

Find ESPPRC for every node given a consumed resource τ

Lower bounds $\underline{c}(i,\tau)$

$\tau = \bar{t} - 3\delta$ ← $\tau = \bar{t} - 2\delta$ ← $\tau = \bar{t} - \delta$ ← $\tau = \bar{t}$

$\bar{t} \triangleq T$

Elementary Shortest Path Problem with Resource Constraints
Bounds pruning

Universidad de los Andes Colombia

$\underline{c}(i, \tau)$

Nodes

Time Consumed (τ)

	1	2	3	4	5	6	7	8	9	10
$\tau = 100$	-	-	-	-	-	-	-	-	-	-

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

$\underline{c}(i,\tau)$

Nodes

Time Consumed (τ)

	1	2	3	4	5	6	7	8	9	10
$\tau = 100$	-	-	-	-	-	-	-	-	-	-
$\tau = 90$	-	-	-1	-	-3	-2	-	-	2	1

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

$\underline{c}(i, \tau)$

Nodes

Time Consumed (τ)

	1	2	3	4	5	6	7	8	9	10
$\tau = 100$	-	-	-	-	-	-	-	-	-	-
$\tau = 90$	-	-	-1	-	-3	-2	-	-	2	1
$\tau = 80$	-5	-3	-3	2	-7	-4	-2	0	-2	-5

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

$\underline{c}(i,\tau)$

Nodes

Time Consumed (τ)

	1	2	3	4	5	6	7	8	9	10
$\tau=100$	-	-	-	-	-	-	-	-	-	-
$\tau=90$	-	-	-1	-	-3	-2	-	-	2	1
$\tau=80$	-5	-3	-3	2	-7	-4	-2	0	-2	-5
$\tau=70$	-12	-4	-4	-4	-11	-6	-5	-2	-4	-7

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

$\underline{c}(i, \tau)$

Nodes

Time Consumed (τ)

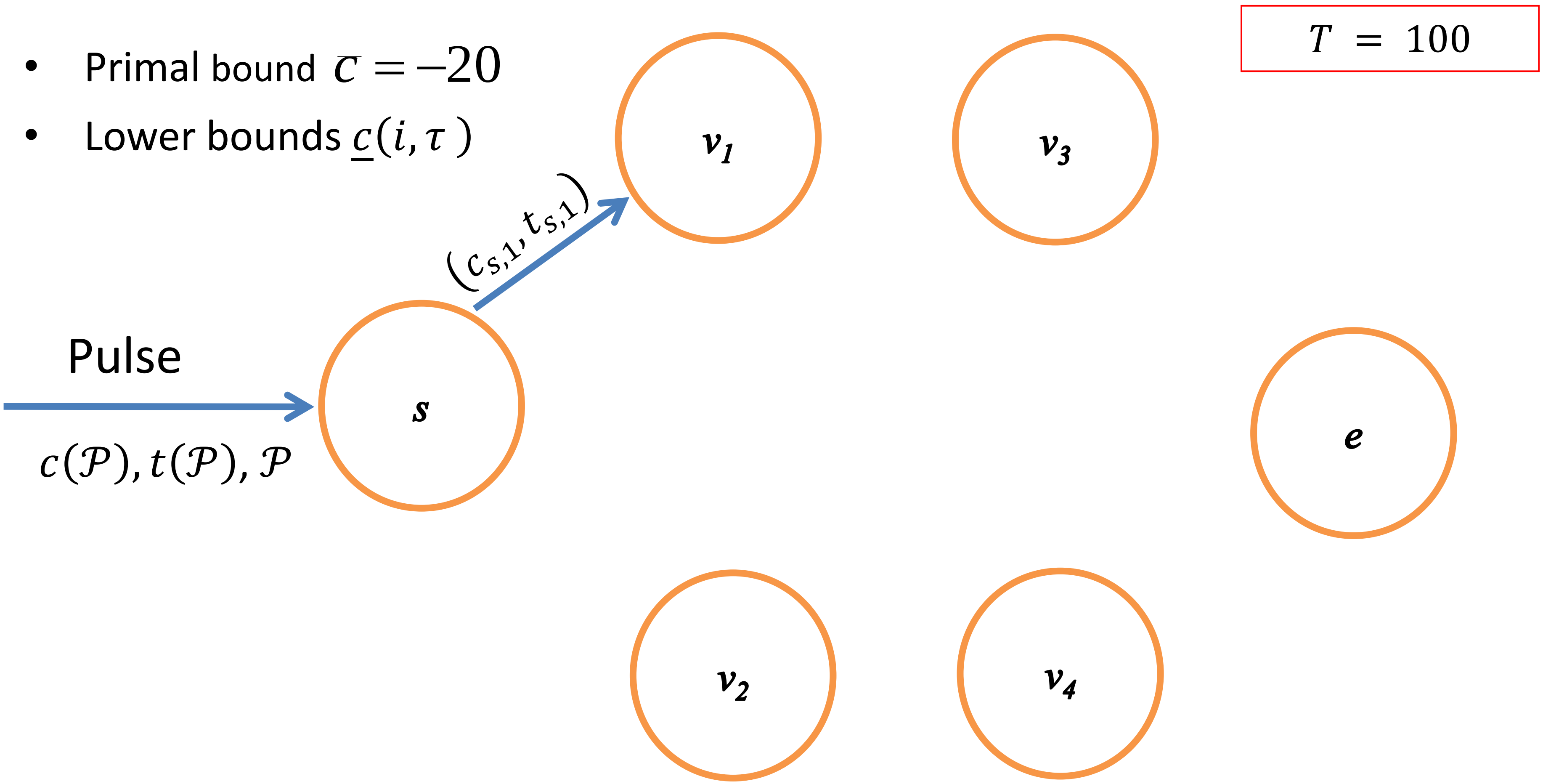
	1	2	3	4	5	6	7	8	9	10
$\tau = 100$	-	-	-	-	-	-	-	-	-	-
$\tau = 90$	-	-	-1	-	-3	-2	-	-	2	1
$\tau = 80$	-5	-3	-3	2	-7	-4	-2	0	-2	-5
$\tau = 70$	$\boxed{-12}$	-4	-4	-4	-11	-6	-5	-2	-4	-7

Minimum cost that can be achieved **starting at node 1** with **70 units of time already consumed**

Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

- Primal bound $\bar{c} = -20$
- Lower bounds $\underline{c}(i, \tau)$

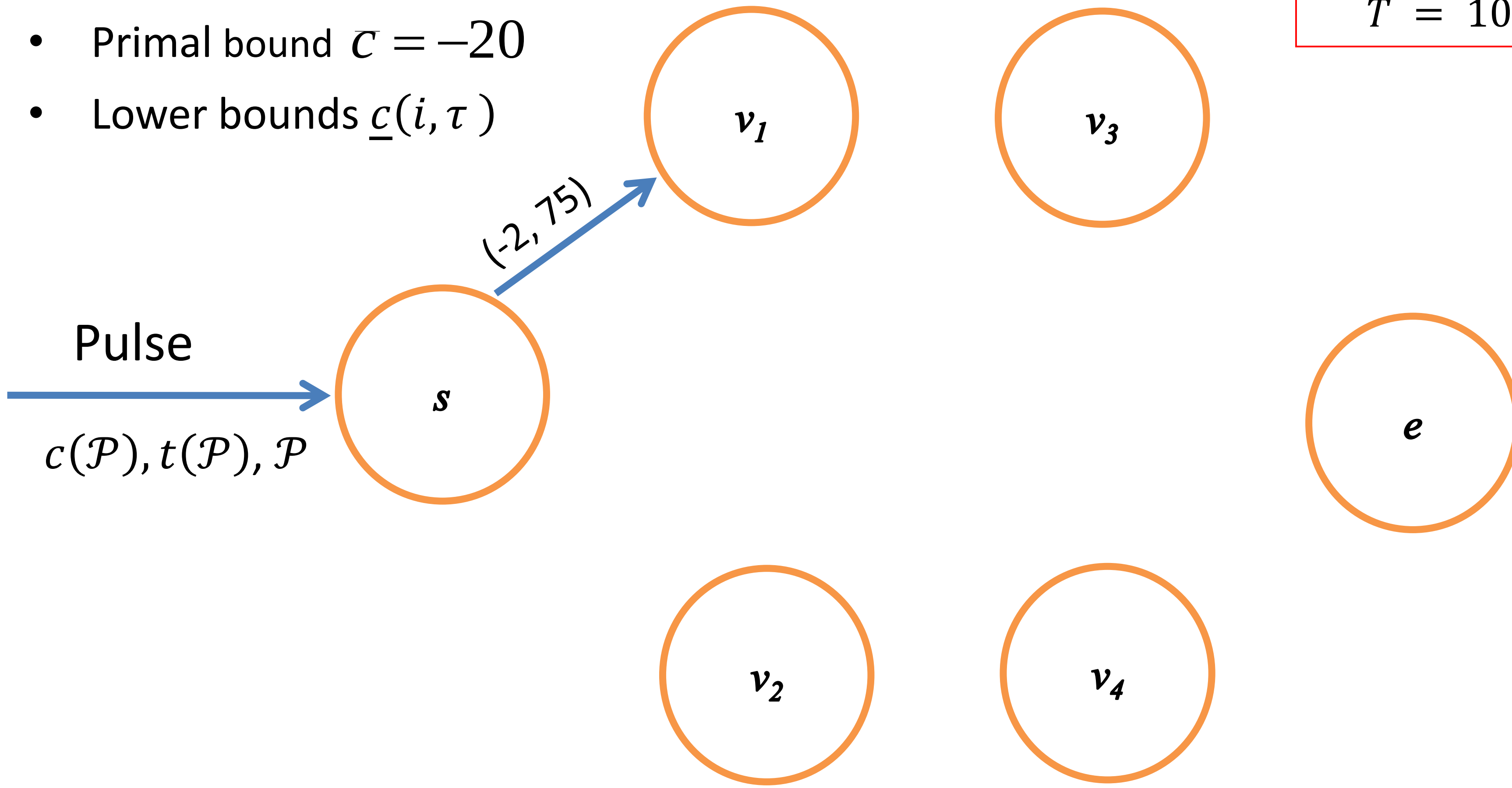


Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

- Primal bound $\overline{c} = -20$
- Lower bounds $\underline{c}(i, \tau)$

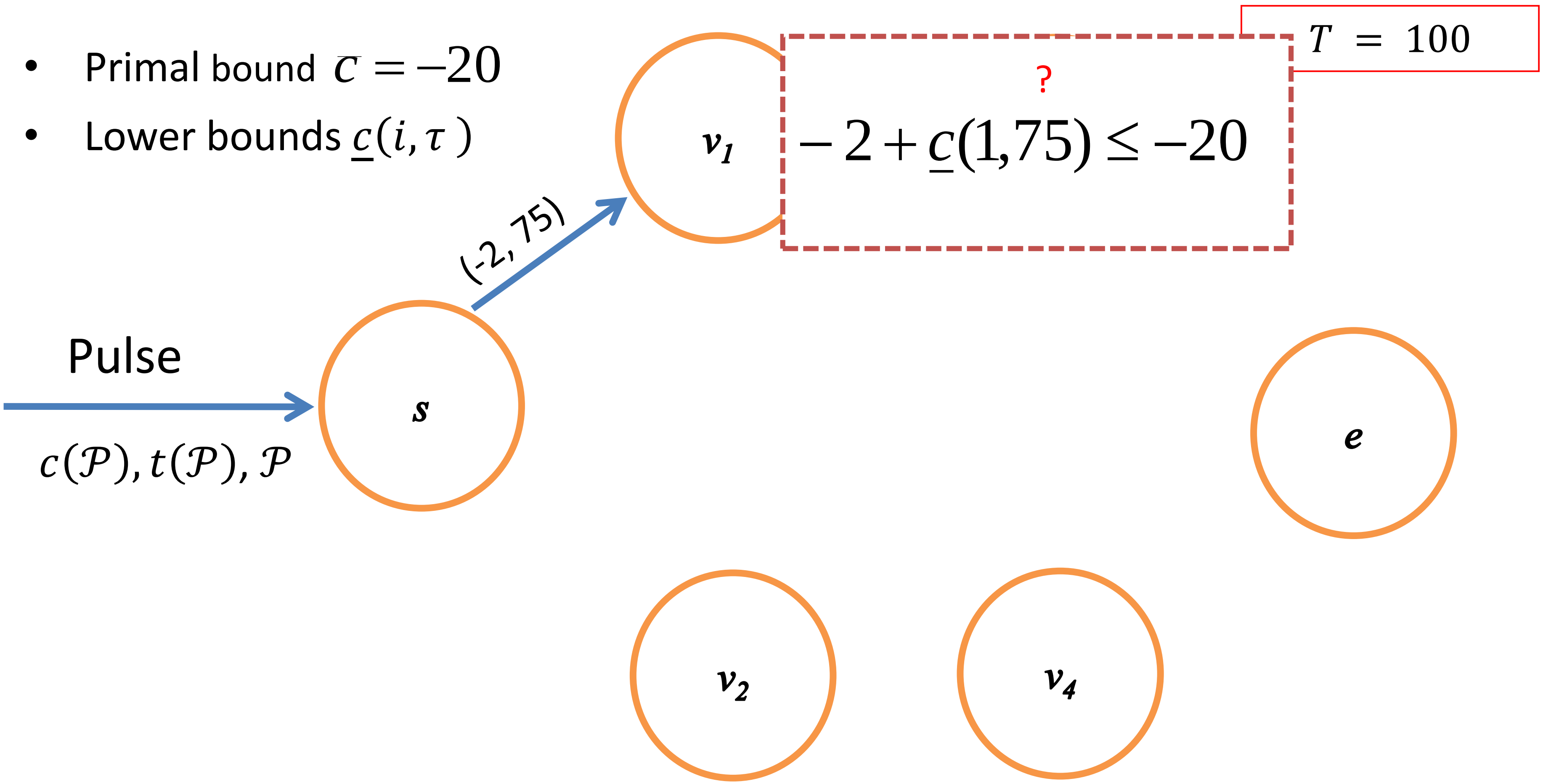
$T = 100$



Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

- Primal bound $\overline{c} = -20$
- Lower bounds $\underline{c}(i, \tau)$



Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

- Primal bound $\bar{c} = -20$
- Lower bounds $\underline{c}(i,\tau)$

$T = 100$

(-2, 75)

?

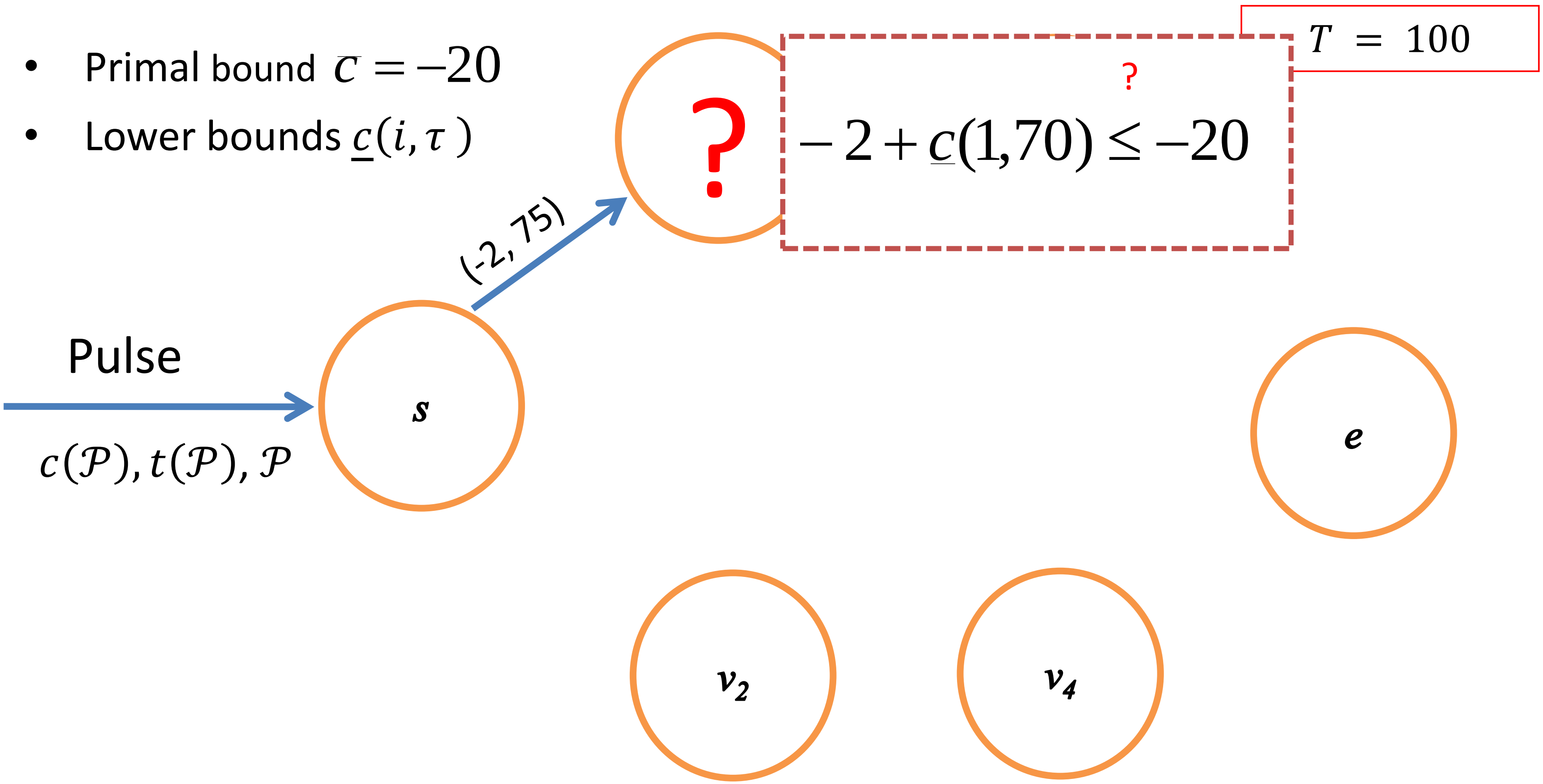
$$-2 + \underline{c}(1{,}75) \overset{?}{\leq} -20$$

	1	2	3	4	5	6	7	8	9	10
$\tau = 100$	-	-	-	-	-	-	-	-	-	-
$\tau = 90$	-	-	-1	-	-3	-2	-	-	2	1
$\tau = 80$	-5	-3	-3	2	-7	-4	-2	0	-2	-5
$\tau = 70$	-12	-4	-4	-4	-11	-6	-5	-2	-4	-7

Elementary Shortest Path Problem with Resource Constraints

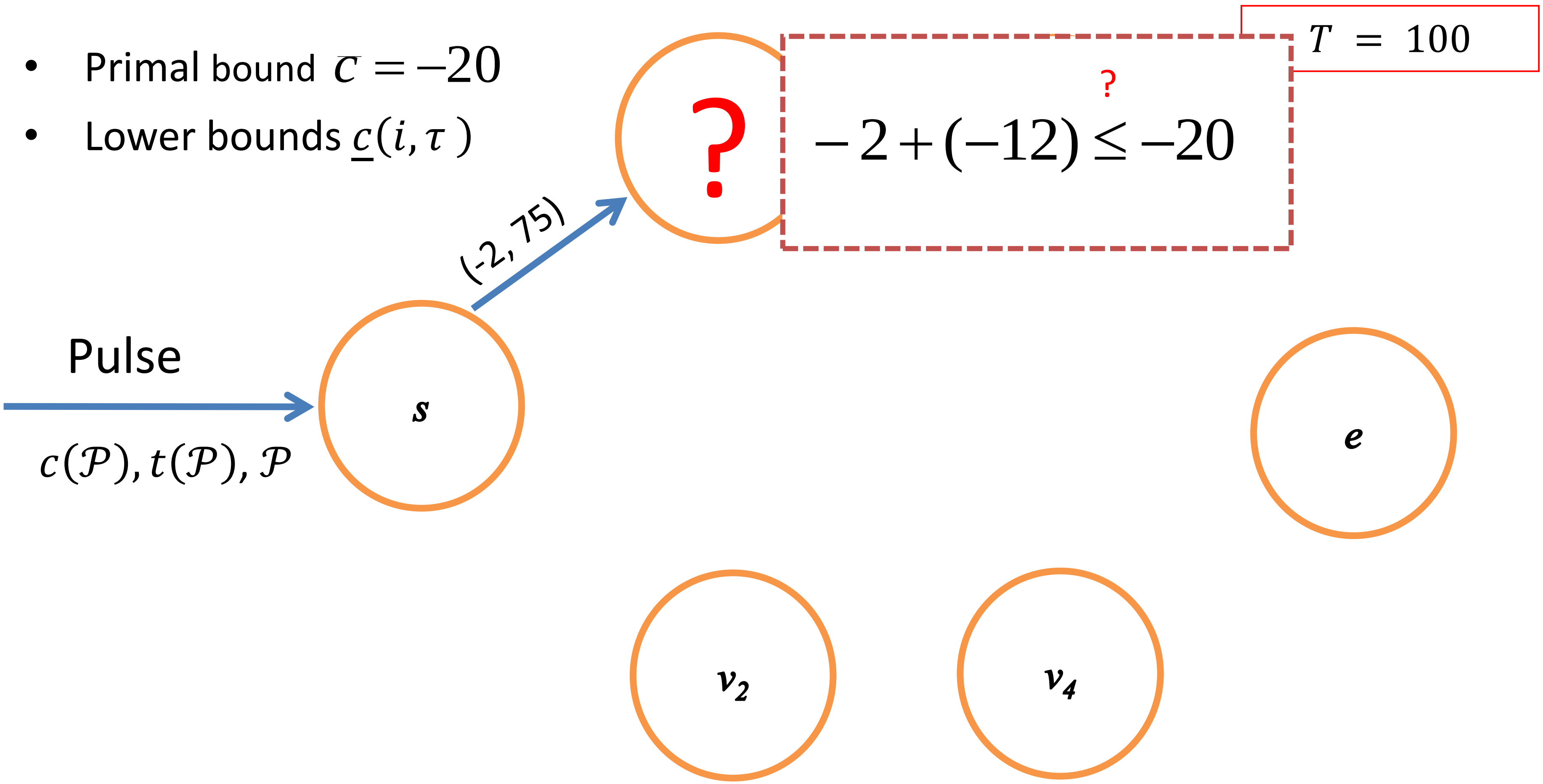
Bounds pruning

- Primal bound $\overline{c} = -20$
- Lower bounds $\underline{c}(i, \tau)$



Bounds pruning

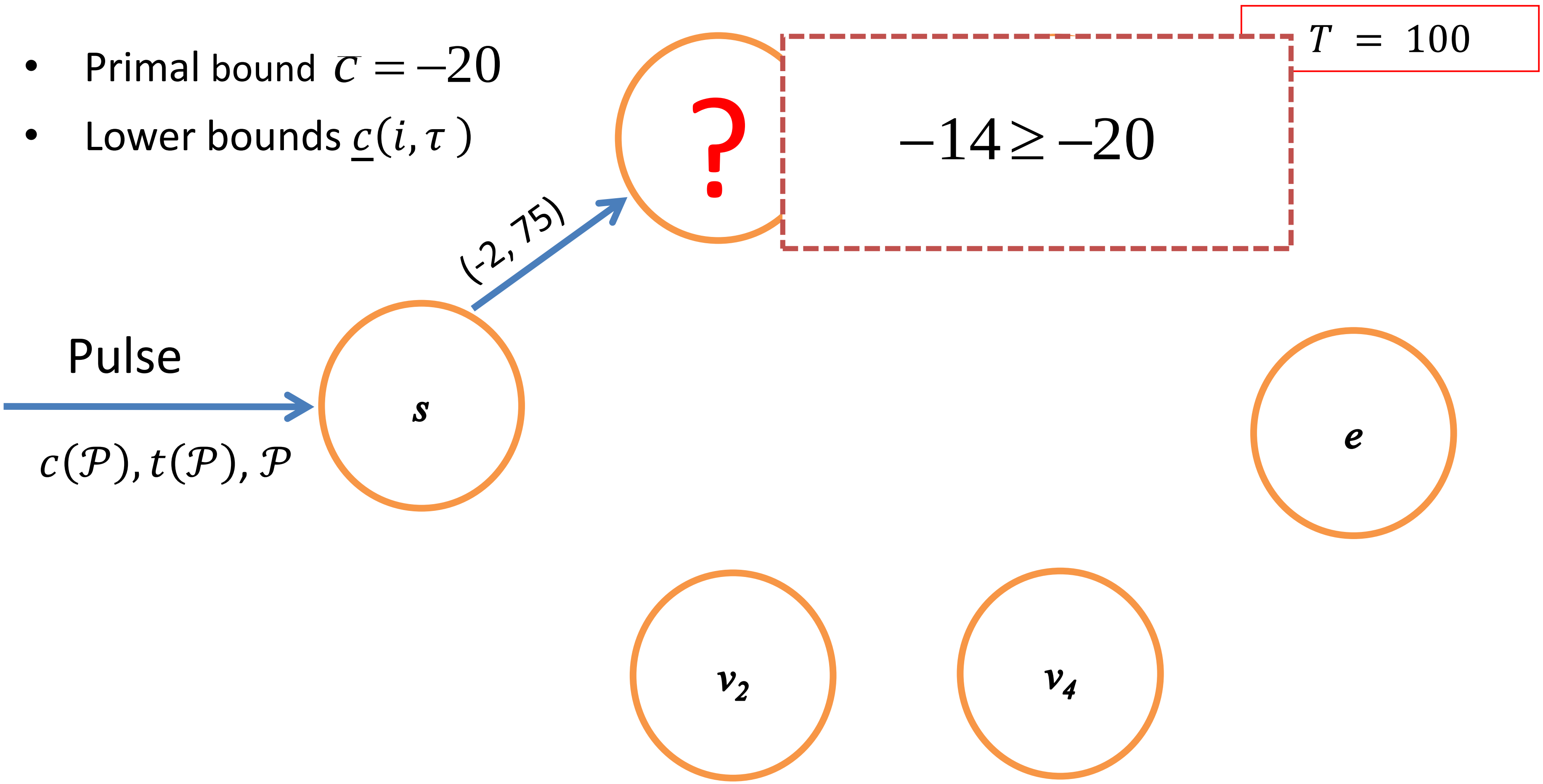
- Primal bound $\overline{c} = -20$
- Lower bounds $\underline{c}(i, \tau)$



Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

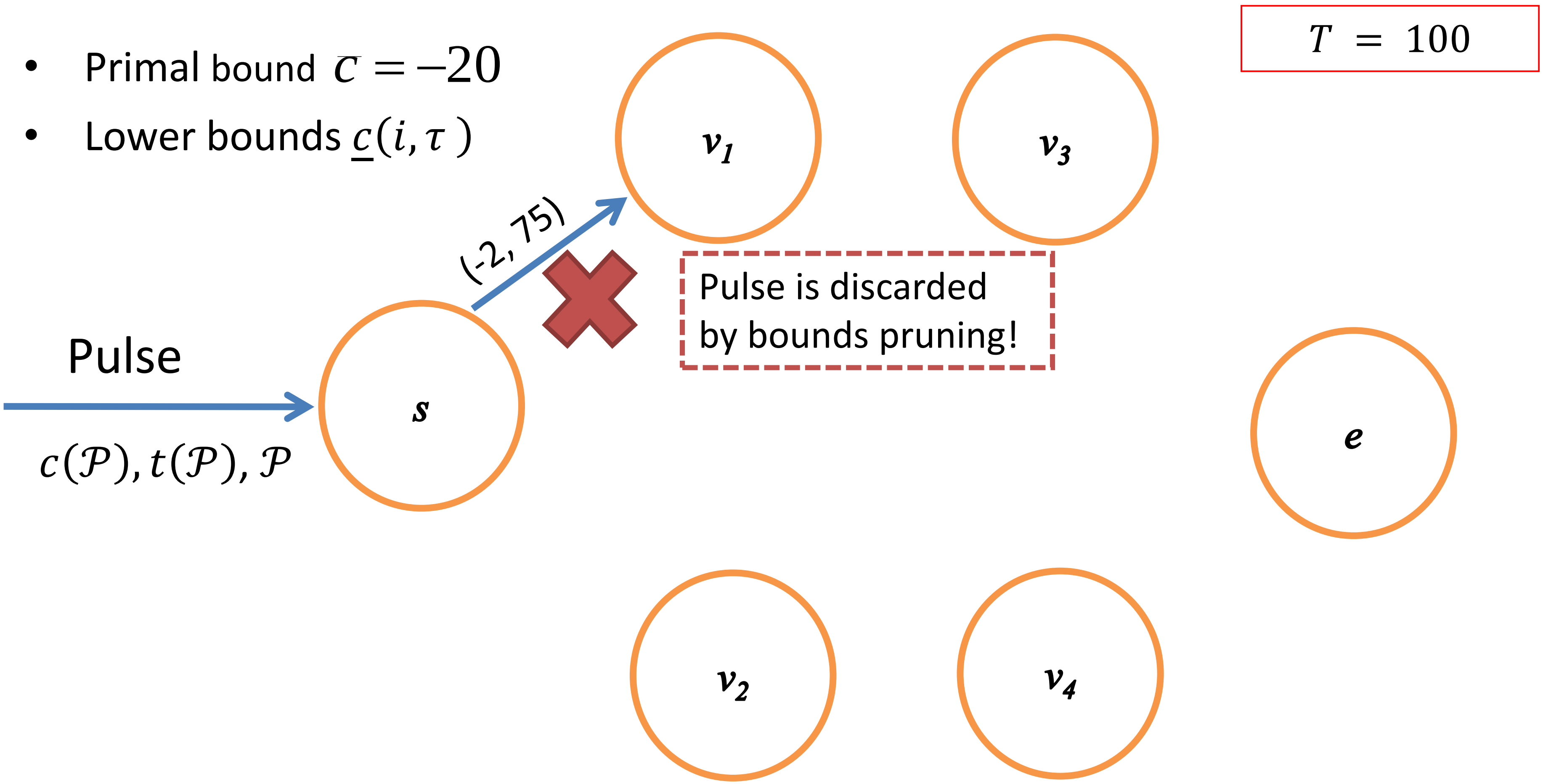
- Primal bound $\overline{c} = -20$
- Lower bounds $\underline{c}(i, \tau)$



Elementary Shortest Path Problem with Resource Constraints

Bounds pruning

- Primal bound $\overline{c} = -20$
- Lower bounds $\underline{c}(i, \tau)$



Setup:

- ESPPRC is embedded in a CG procedure to solve the VRPTW root node
- Benchmark algorithms by Baldacci et al. (2011) and Desaulniers et al. (2008)
- Pulse algorithm coded in Java and compiled in Eclipse SDK 4.3.0
- CPU: Intel Core i7 (2 cores) Duo @ 2.00GHz 512MB of RAM for JVM
- Tabu search heuristic (Desaulniers et al., 2008) is used for the first iterations of the CG procedure
- Master problem is solved using Gurobi 5.0.1

Elementary Shortest Path Problem with Resource Constraints

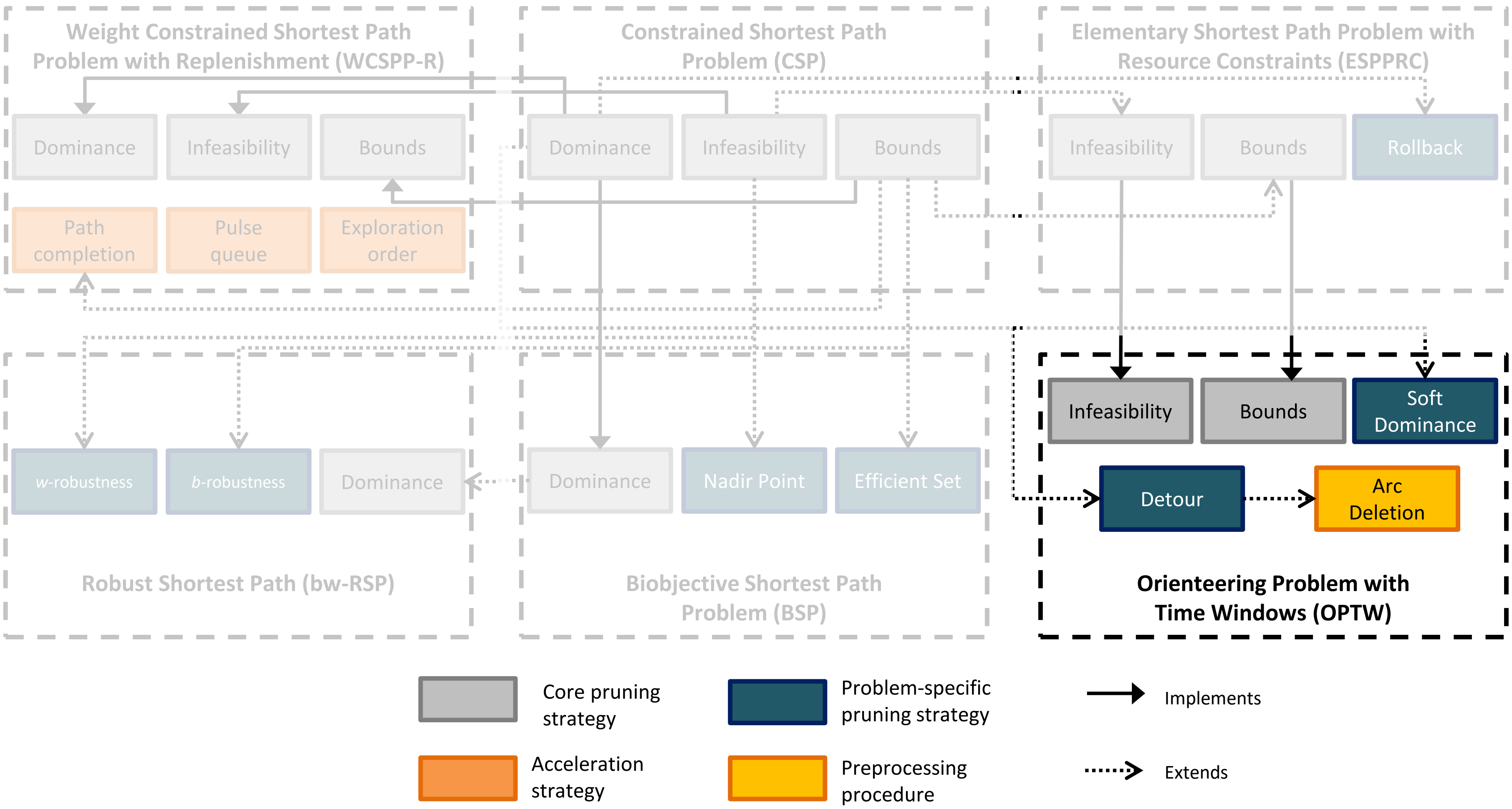
Computational experiments

	Baldacci et al. (2011a)		Col. Generation with pulse and TS			
Instance	Time (s)	LB_3	Time (s)	Scaled time (s)	Lower bound	Bound improvement (%)
r201.100	17.0	1140.3	**9.3**	**13.0**	1140.3	0.00
r202.100	22.0	1021.2	18.0	25.2	1022.2	**0.10**
r203.100	165.0	865.8	**72.4**	**101.4**	866.9	**0.13**
r204.100	194.0	724.2	**133.2**	**186.4**	724.9	**0.10**
r205.100	31.0	938.0	25.3	35.4	938.9	**0.10**
r206.100	76.0	866.3	**44.7**	**62.6**	866.9	**0.07**
r207.100	368.0	789.9	**127.5**	**178.6**	790.7	**0.10**
r208.100	2405.0	690.3	**266.3**	**372.9**	692.0	**0.24**
r209.100	59.0	840.6	42.3	59.2	841.4	**0.10**
r210.100	57.0	888.2	**34.4**	**48.2**	889.4	**0.14**
r211.100	219.0	734.1	**76.8**	**107.6**	734.7	**0.08**
rc201.100	12.0	1255.4	**6.2**	**8.6**	1255.9	**0.04**
rc202.100	13.0	1086.2	**7.1**	**9.9**	1088.1	**0.17**
rc203.100	22.0	919.5	34.7	48.5	922.5	**0.33**
rc204.100	455.0	778.4	**322.7**	**451.7**	779.7	**0.17**
rc205.100	14.0	1145.8	**7.7**	**10.8**	1147.6	**0.16**
rc206.100	16.0	1037.7	19.2	26.9	1038.6	**0.09**
rc207.100	62.0	945.8	**42.8**	**59.9**	947.3	**0.16**
rc208.100	168.0	765.8	441.9	618.6	766.7	**0.12**
c201.100	n/a	n/a	2.4	3.4	589.1	n/a
c202.100	n/a	n/a	165.7	231.9	589.1	n/a
c203.100	n/a	n/a	173.6	243.0	588.7	n/a
c204.100	182.0	588.1	323.3	452.6	588.1	0.00
c205.100	n/a	n/a	4.8	6.7	586.4	n/a
c206.100	n/a	n/a	4.6	6.5	586.0	n/a
c207.100	n/a	n/a	8.2	11.4	585.8	n/a
c208.100	n/a	n/a	7.8	10.9	585.8	n/a

Agenda

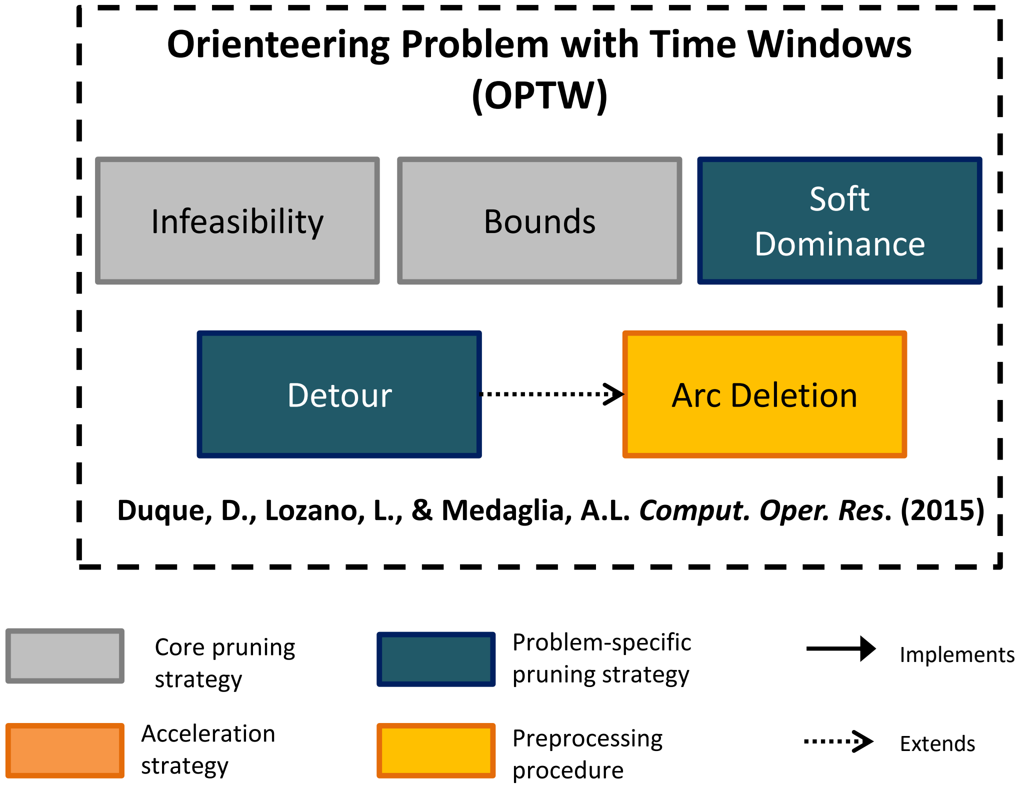
- Part I: fundamentals
- Part II: intuition
- Part III: extensions
  - Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)
  - Biobjective Shortest Path Problem (BSP)
  - Elementary Shortest Path Problem with Resource Constraints (ESPPRC)
  - **Orienteering Problem with Time Windows (OPTW)**
  - Robust Shortest Path (bw-RSP)
- Part IV: applications
- Part V: perspectives

Pulse Algorithm for Hard Shortest Path Problems



Orienteering Problem with Time Windows (OPTW)

Pruning and acceleration strategies



- Righini & Salani (2009)
- Vansteenwegen et al. (2011)
- Gambardella, Montemanni & Weyland (2012)
- Archetti, Bianchessi & Speranza (2013)

Orienteering Problem with Time Windows (OPTW)

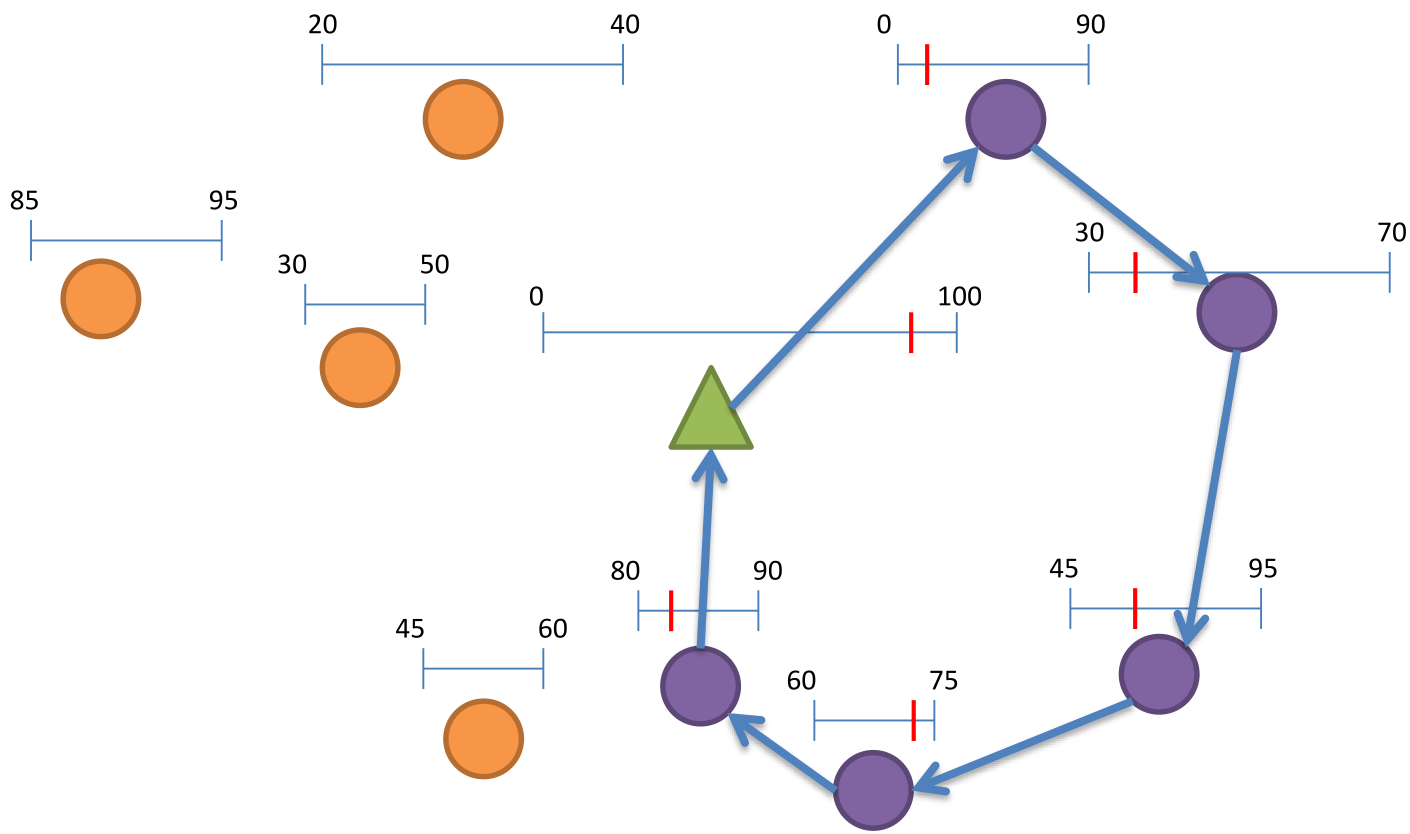
Problem statement

- The OPTW is defined by:
  - Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$
    - $\mathcal{N} = \{v_1, \dots, v_i, \dots, v_n\}$
    - $\mathcal{A} = \{(i,j) | v_i \in \mathcal{N}, v_j \in \mathcal{N},\ i \neq j\}$
  - Find a maximum score path starting at node v_s and ending at node v_e
  - Nonnegative score s_i for visiting the node v_i
  - Nonnegative weight t_{ij} is the travel time between nodes
  - Maximum travel time T
  - Time window $[a_i, b_i]$ at each node v_i

Orienteering Problem with Time Windows (OPTW)

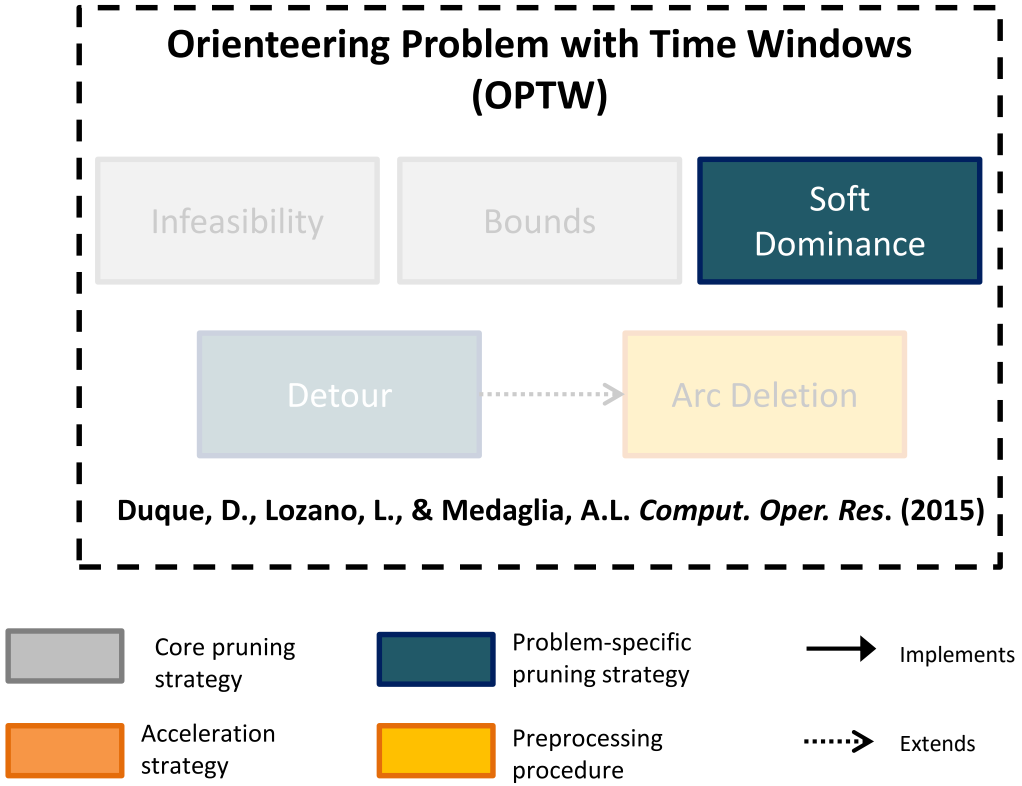
Problem statement

Which nodes to visit?
How to route those nodes within the time constraints?



Orienteering Problem with Time Windows (OPTW)

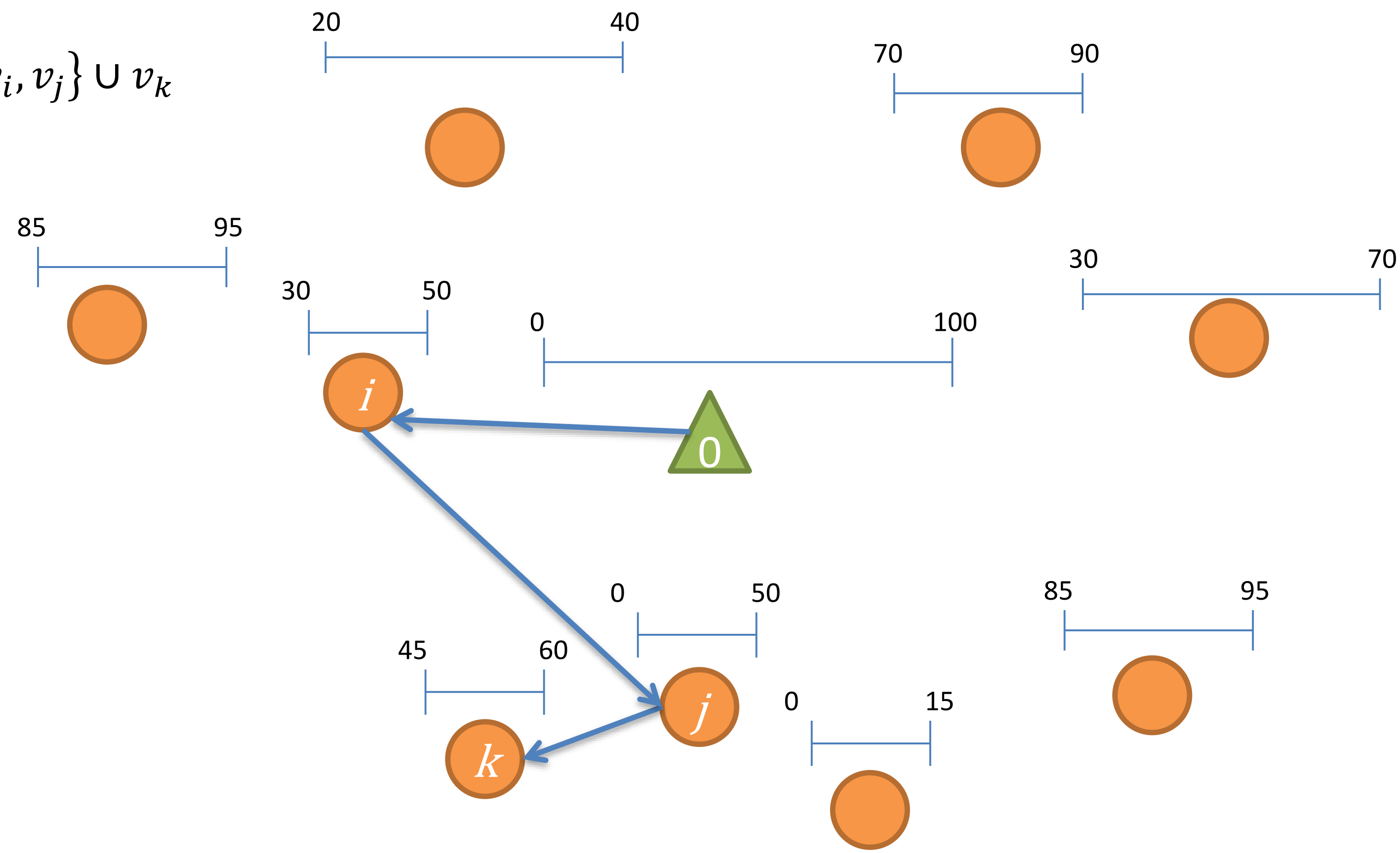
Soft-dominance pruning



Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

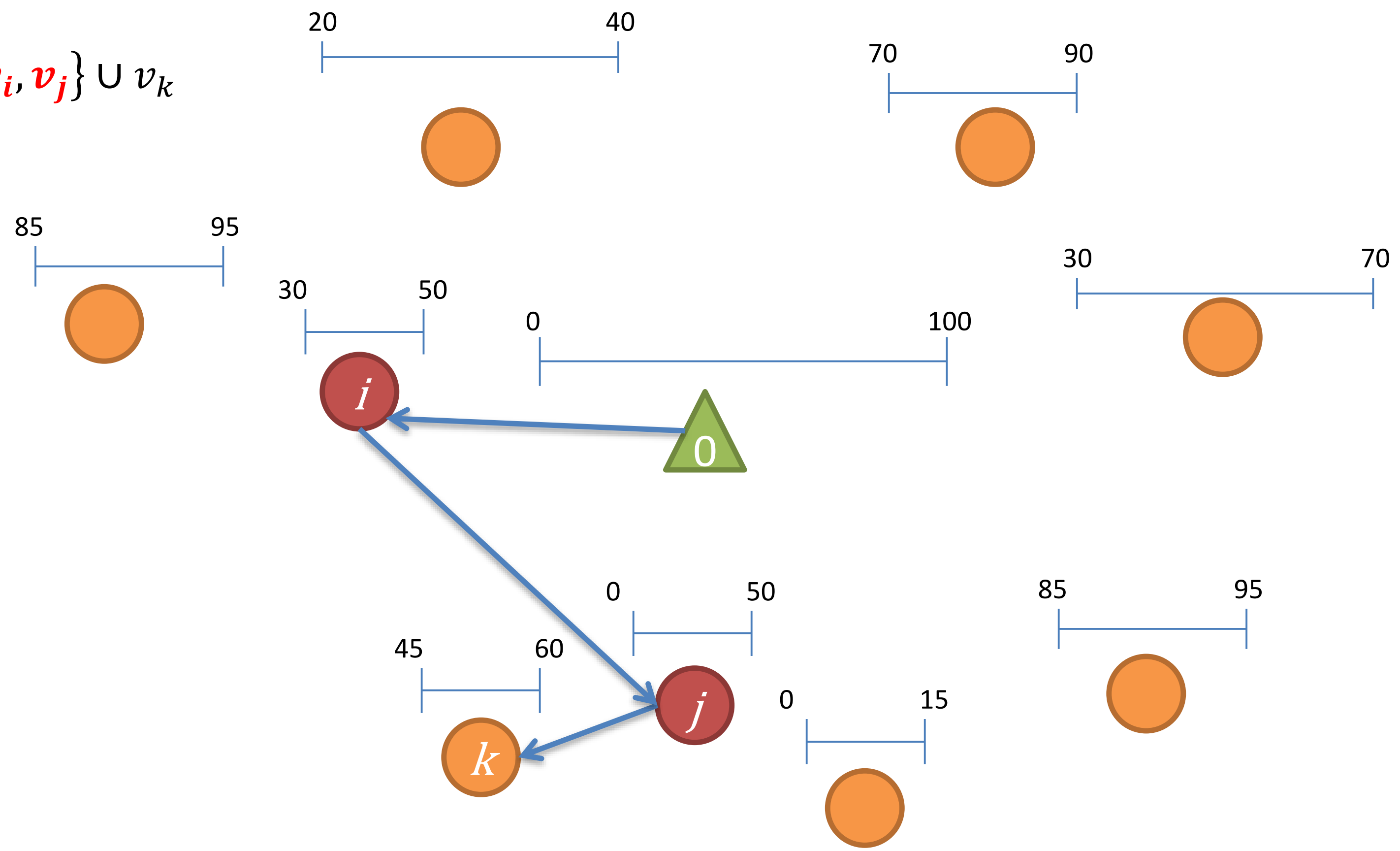
Pulse log:
$S(\mathcal{P}) = 35$
$t(\mathcal{P}) = 58$
$\mathcal{P} = \{v_0, v_i, v_j\} \cup v_k$



Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

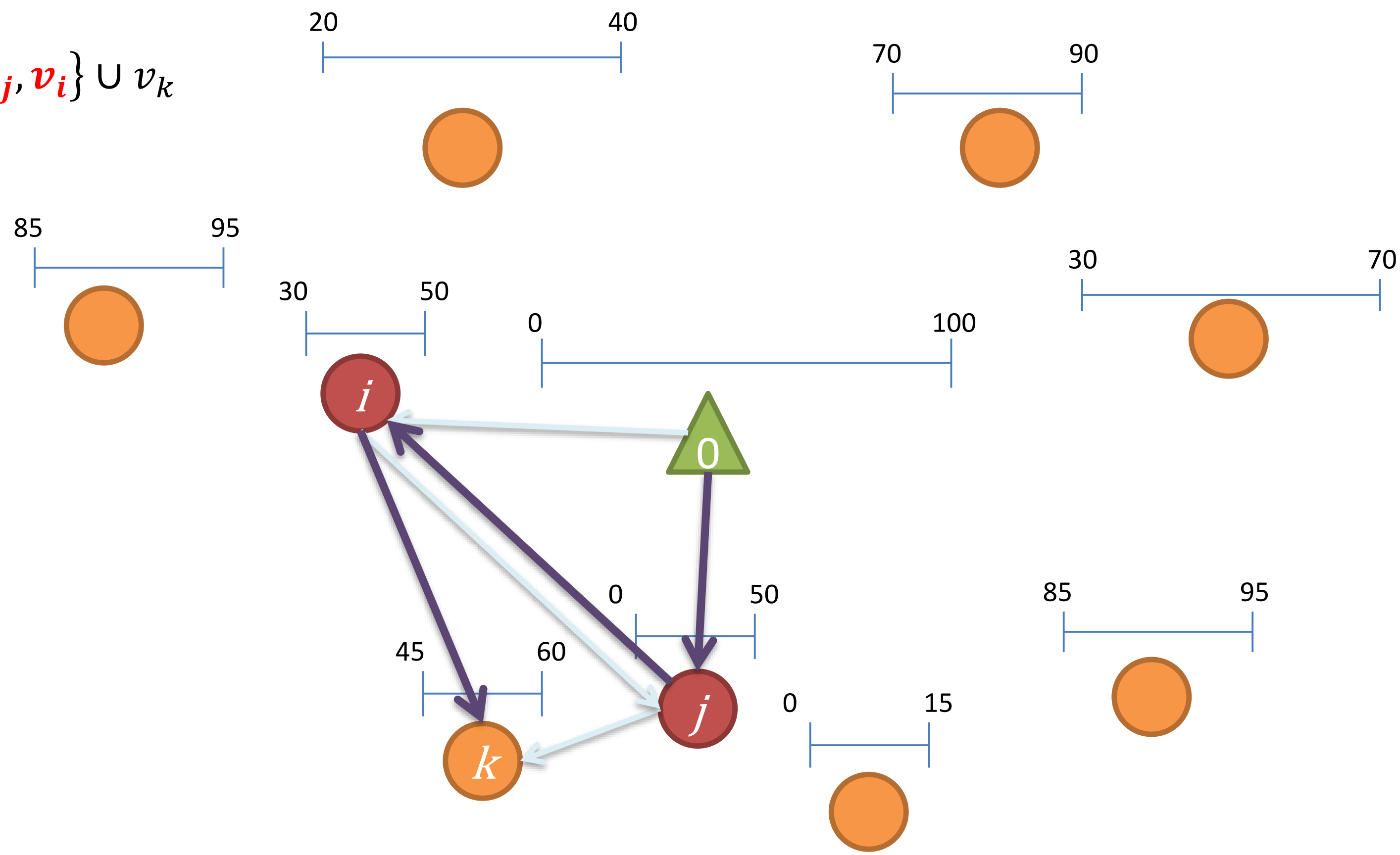
Pulse log:
$S(\mathcal{P}) = 35$
$t(\mathcal{P}) = \boxed{58}$
$\mathcal{P} = \{v_0, \boldsymbol{v_i}, \boldsymbol{v_j}\} \cup v_k$



Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

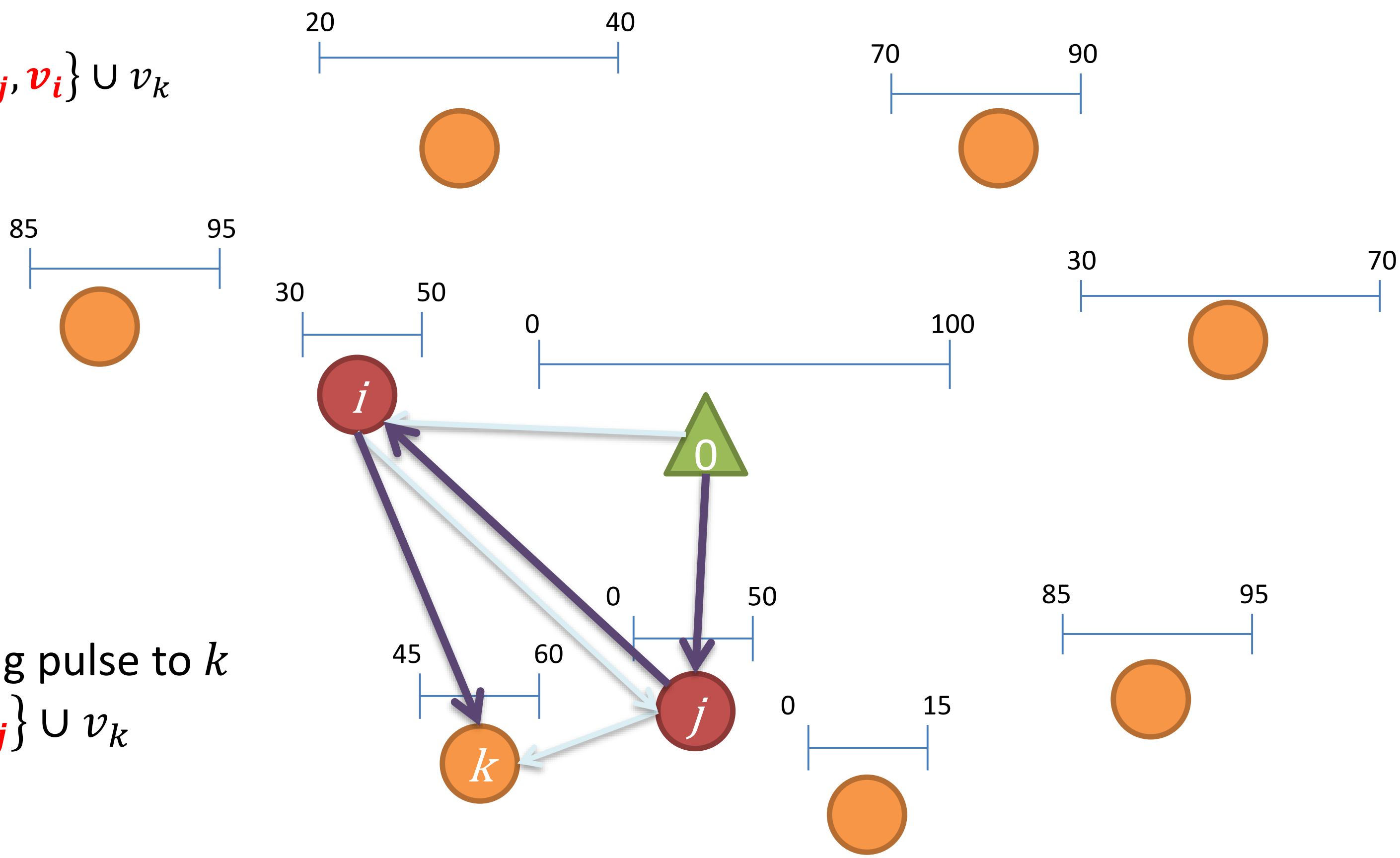
Pulse log:
$S(\mathcal{P}) = 35$
$t(\mathcal{P}) = \boxed{40}$
$\mathcal{P} = \{v_0, \boldsymbol{v_j}, \boldsymbol{v_i}\} \cup v_k$



Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

Pulse log:
$S(\mathcal{P}) = 35$
$t(\mathcal{P}) = 40$
$\mathcal{P} = \{v_0, \boldsymbol{v_j}, \boldsymbol{v_i}\} \cup v_k$



Prune incoming pulse to k
$\mathcal{P} = \{v_0, \boldsymbol{v_i}, \boldsymbol{v_j}\} \cup v_k$

Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

$$\mathcal{P}_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, v_{(3)}, v_{(4)}, v_{(5)}\} \cup \{v_i\}$$

Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

$$\mathcal{P}_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, v_{(3)}, v_{(4)}, v_{(5)}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{\boldsymbol{v_{(5)}}, v_{(2)}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(1)}}\} \cup \{v_i\}$$

Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

$$\mathcal{P}_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, v_{(3)}, v_{(4)}, v_{(5)}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{\boldsymbol{v_{(5)}}, v_{(2)}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(1)}}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, \boldsymbol{v_{(5)}}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(2)}}\} \cup \{v_i\}$$

Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

$$\mathcal{P}_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, v_{(3)}, v_{(4)}, v_{(5)}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{\boldsymbol{v_{(5)}}, v_{(2)}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(1)}}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, \boldsymbol{v_{(5)}}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(2)}}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, \boldsymbol{v_{(5)}}, v_{(4)}, \boldsymbol{v_{(3)}}\} \cup \{v_i\}$$

Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

$$\mathcal{P}_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, v_{(3)}, v_{(4)}, v_{(5)}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{\boldsymbol{v_{(5)}}, v_{(2)}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(1)}}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, \boldsymbol{v_{(5)}}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(2)}}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, \boldsymbol{v_{(5)}}, v_{(4)}, \boldsymbol{v_{(3)}}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, v_{(3)}, \boldsymbol{v_{(5)}}, \boldsymbol{v_{(4)}}\} \cup \{v_i\}$$

Orienteering Problem with Time Windows (OPTW)

Soft-dominance pruning

$$\mathcal{P}_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, v_{(3)}, v_{(4)}, v_{(5)}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{\boldsymbol{v_{(5)}}, v_{(2)}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(1)}}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, \boldsymbol{v_{(5)}}, v_{(3)}, v_{(4)}, \boldsymbol{v_{(2)}}\} \cup \{v_i\}$$

$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, \boldsymbol{v_{(5)}}, v_{(4)}, \boldsymbol{v_{(3)}}\} \cup \{v_i\}$$

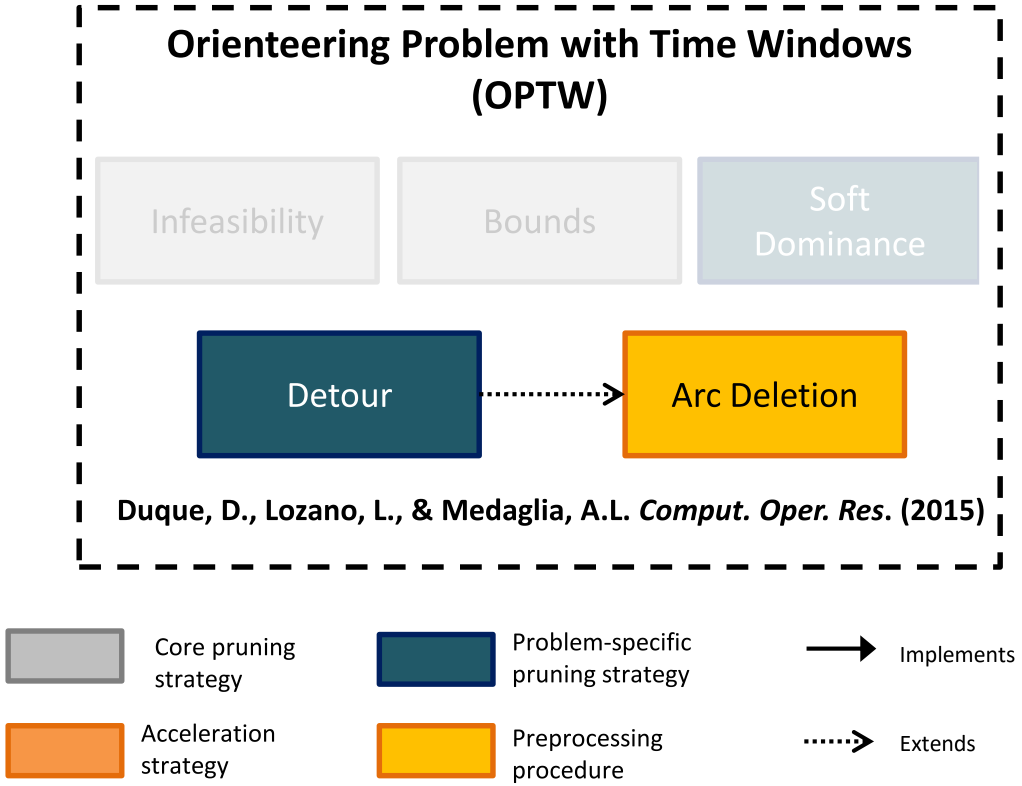
$$\mathcal{P}'_{s,i} \leftarrow \{v_s\} \cup \{v_{(1)}, v_{(2)}, v_{(3)}, \boldsymbol{v_{(5)}}, \boldsymbol{v_{(4)}}\} \cup \{v_i\}$$

An incoming pulse to v_i is pruned if:

$$\texttt{feasible}\left(\mathcal{P}'_{s,i}\right) \text{ and } t\left(\mathcal{P}'_{s,i}\right) \leq t\left(\mathcal{P}_{s,i}\right)$$

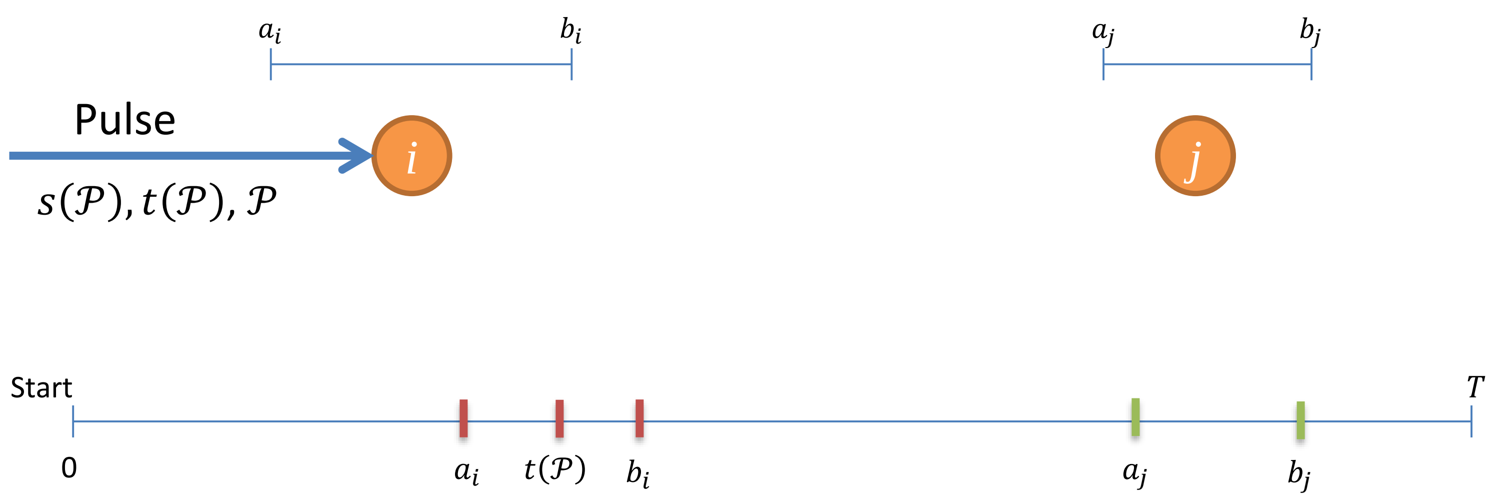
Orienteering Problem with Time Windows (OPTW)

Detour pruning



Orienteering Problem with Time Windows (OPTW)

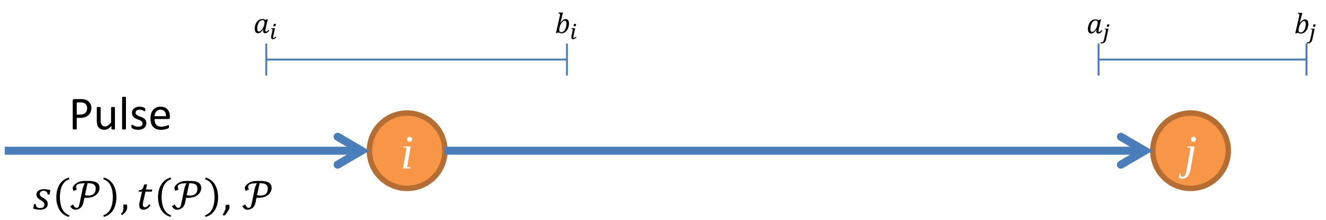
Detour pruning



Orienteering Problem with Time Windows (OPTW)

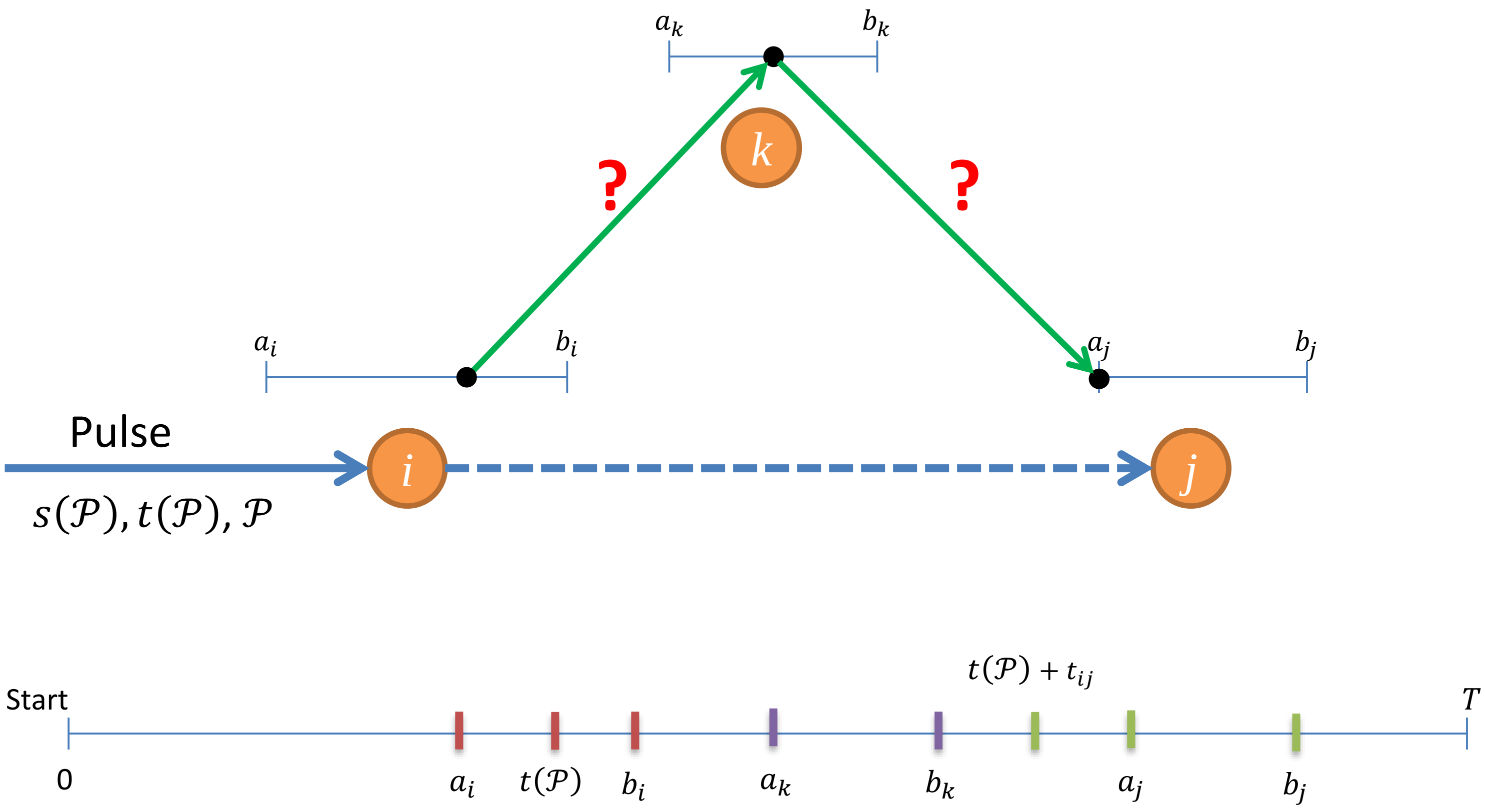
Detour pruning

$$t(\mathcal{P}) + t_{ij} \leq a_j$$



Orienteering Problem with Time Windows (OPTW)

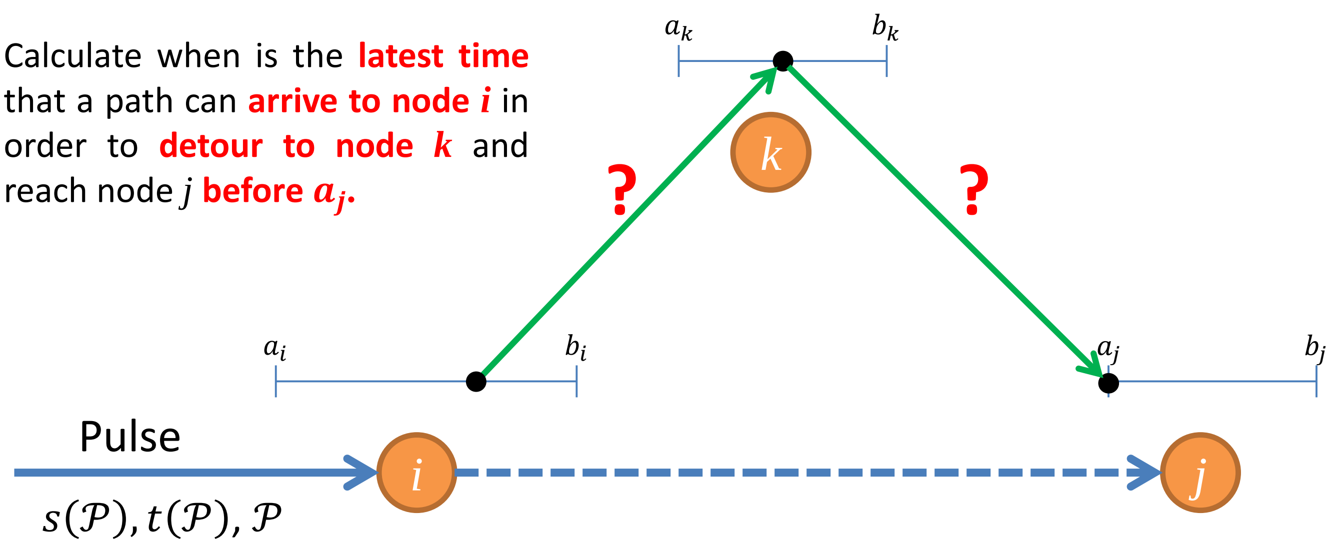
Detour pruning

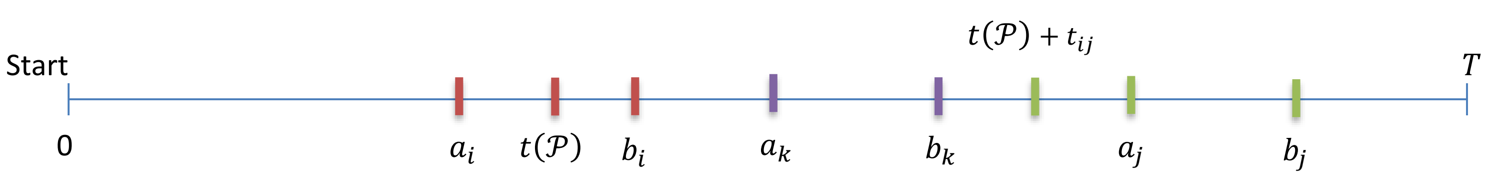


Orienteering Problem with Time Windows (OPTW)

Detour pruning

Calculate when is the **latest time** that a path can **arrive to node *i*** in order to **detour to node *k*** and reach node j **before a_j.**

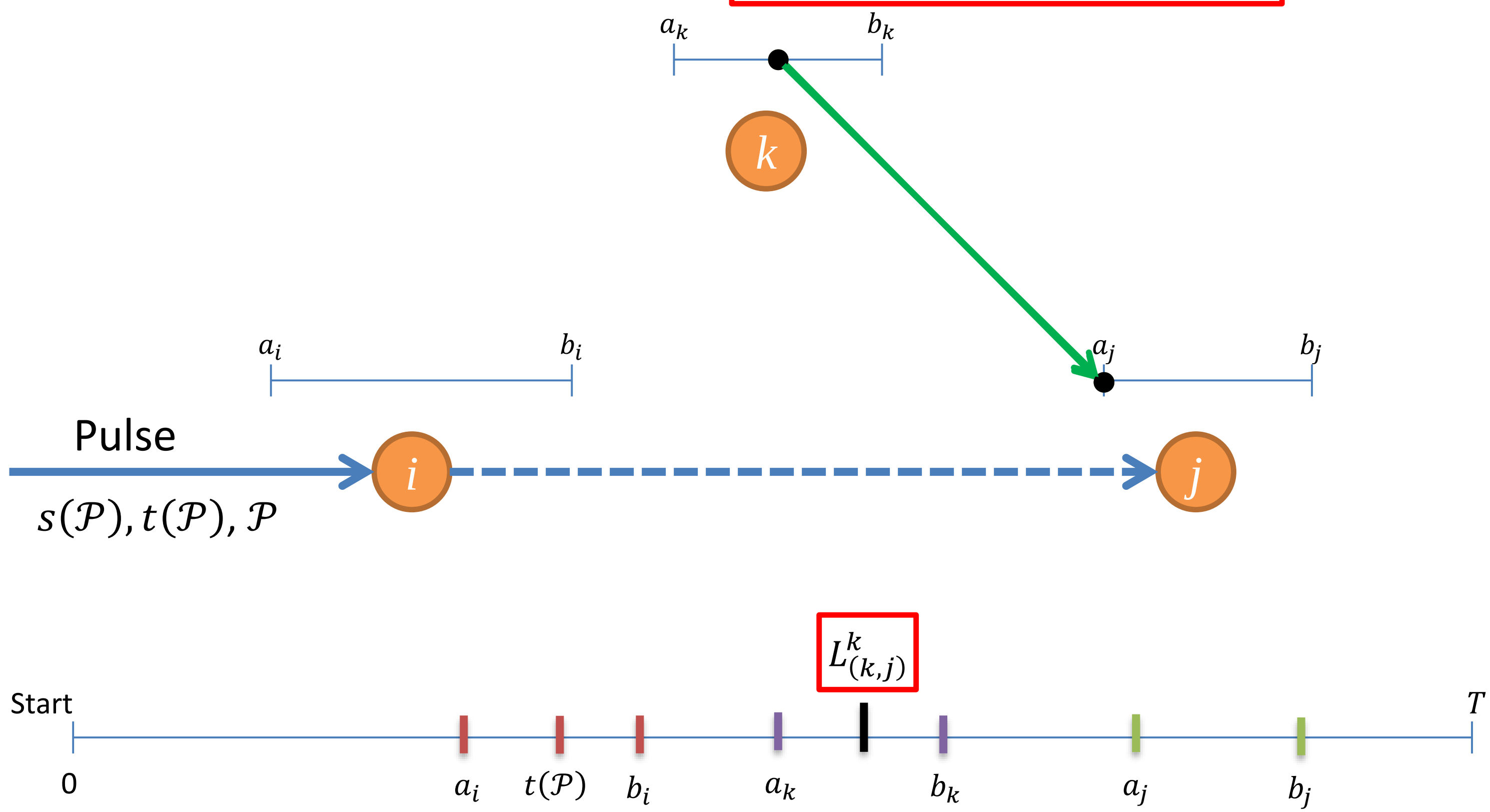




Orienteering Problem with Time Windows (OPTW)

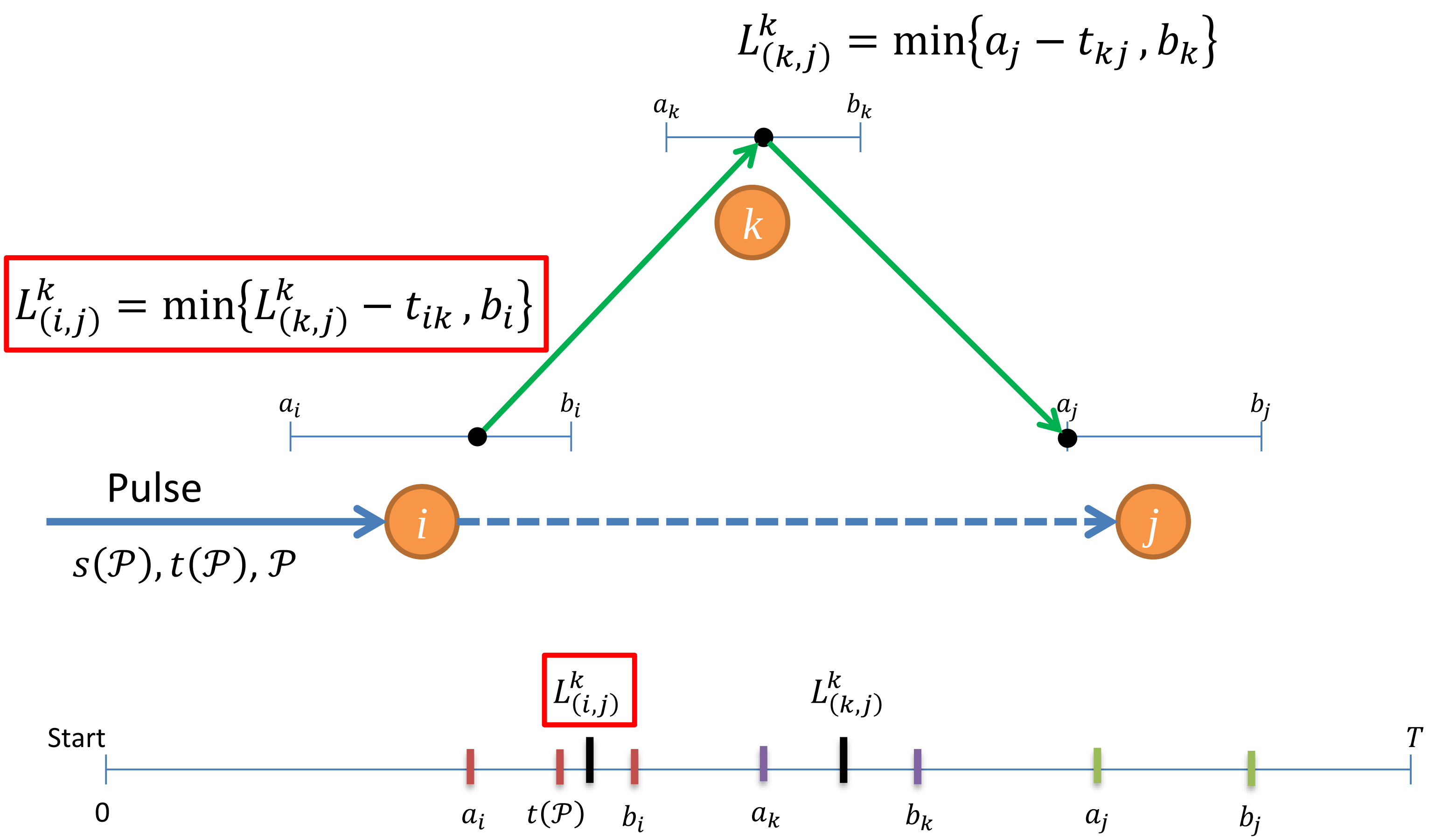
Detour pruning

$$\boxed{L^{k}_{(k,j)} = \min\{a_j - t_{kj}\,,b_k\}}$$



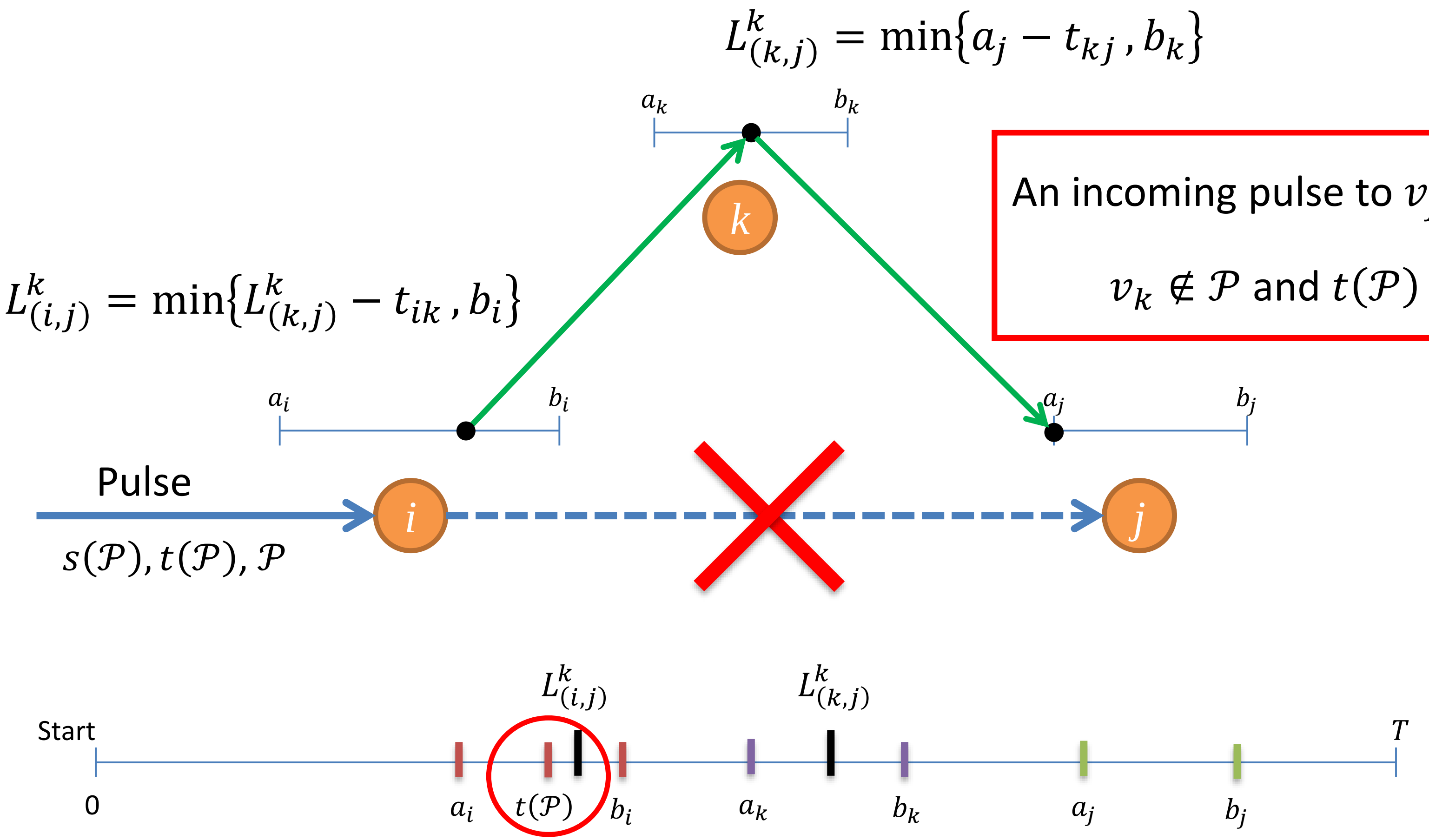
Orienteering Problem with Time Windows (OPTW)

Detour pruning



Orienteering Problem with Time Windows (OPTW)

Detour pruning



An incoming pulse to v_j is pruned if:

$v_k \notin \mathcal{P}$ and $t(\mathcal{P}) \leq L^k_{(i,j)}$

Orienteering Problem with Time Windows (OPTW)

Computational experiments

Setup:

- Benchmark algorithm by Righini & Salani (2009)
- Pulse algorithm coded in Java and compiled with Eclipse SDK 4.2.1
- CPU: Intel Core i7 (2 cores) Duo @ 1.90GHz 2.0 GB RAM for JVM
- The number of threads triggered is fixed to four

Orienteering Problem with Time Windows (OPTW)

Computational experiments

Instance	Optimal score	Righini & Salani DSSR (s)	Pulse (s)	Speedup
C101_100	320	0.06	**0.00**	13.85
C102_100	360	3.81	**0.54**	7.03
C103_100	400	1081.04	**8.94**	$\boxed{120.87}$
C104_100	420	1856.39	**8.94**	$\boxed{207.56}$
C105_100	340	0.12	**0.07**	1.85
C106_100	340	0.14	**0.07**	2.02
C107_100	370	0.20	**0.07**	2.88
C108_100	370	1.43	**0.14**	10.31
C109_100	380	10.57	**0.34**	31.27
R101_100	198	**0.03**	0.07	0.43
R102_100	286	233.20	**7.51**	31.04
R103_100	293	5498.81	**31.29**	$\boxed{175.76}$
R104_100	303	>7200	**80.04**	$\boxed{>89.95}$
R105_100	247	0.23	**0.07**	3.54
R106_100	293	334.49	**12.66**	26.42
R107_100	299	2979.94	**35.49**	$\boxed{83.98}$
R108_100	308	>7200	**75.64**	$\boxed{>95.18}$
R109_100	277	3.09	**0.20**	15.17
R110_100	284	30.83	**0.75**	41.36
R111_100	297	1408.80	**14.56**	$\boxed{96.79}$
R112_100	298	2508.17	**9.41**	$\boxed{266.49}$
RC101_100	219	0.23	**0.07**	3.54
RC102_100	266	6.11	**0.13**	45.48
RC103_100	266	88.12	**0.47**	$\boxed{186.56}$
RC104_100	301	264.84	**1.56**	$\boxed{170.24}$
RC105_100	244	2.86	**0.07**	44.00
RC106_100	252	2.08	**0.13**	15.48
RC107_100	277	49.19	**0.41**	120.76
RC108_100	298	68.95	**0.81**	85.09
Arithmetic mean				$\boxed{68.79}$
Geometric mean				27.83

Orienteering Problem with Time Windows (OPTW)

Computational experiments

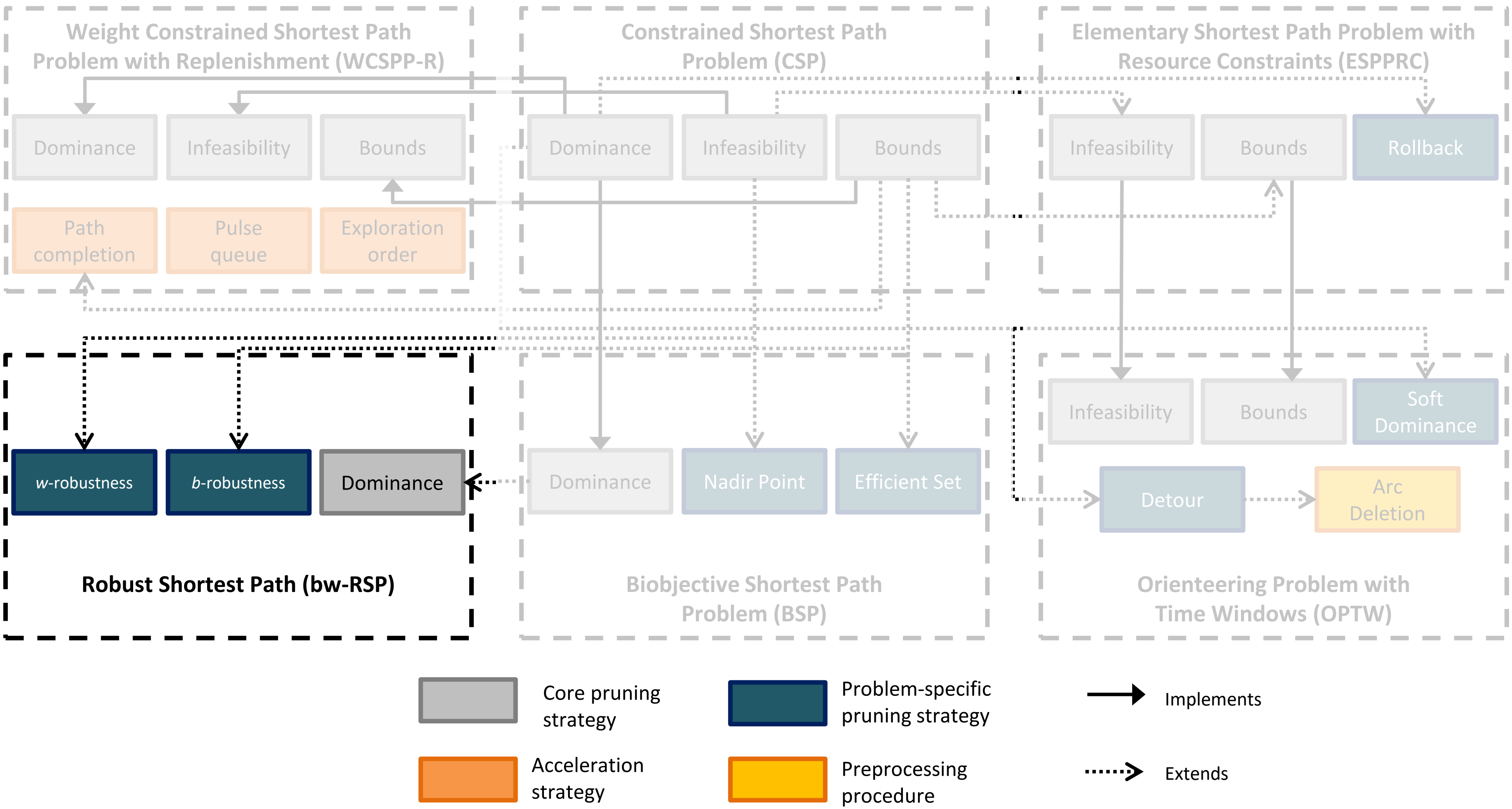
Instance	Optimal score	Righini & Salani DSSR (s)	Pulse (s)	Speedup
pr01_48	308	1.19	**0.14**	8.58
pr02_96	404	37.52	**1.22**	30.81
pr03_144	394	151.73	**2.30**	65.94
pr04_192	489	648.82	**4.06**	159.62
pr05_240	595	6815.82	**36.57**	186.36
pr06_288	591	>7200	**78.23**	>92.04
pr07_72	298	3.65	**0.27**	13.59
pr08_144	463	90.71	**1.69**	53.67
pr09_216	493	3270.88	**81.01**	40.38
pr10_288	594	>7200	**64.26**	>112.04
Arithmetic mean				76.30
Geometric mean				52.45

Agenda

- Part I: fundamentals
- Part II: intuition
- Part III: extensions
  - Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R)
  - Biobjective Shortest Path Problem (BSP)
  - Elementary Shortest Path Problem with Resource Constraints (ESPPRC)
  - Orienteering Problem with Time Windows (OPTW)
  - **Robust Shortest Path (bw-RSP)**
- Part IV: applications
- Part V: perspectives

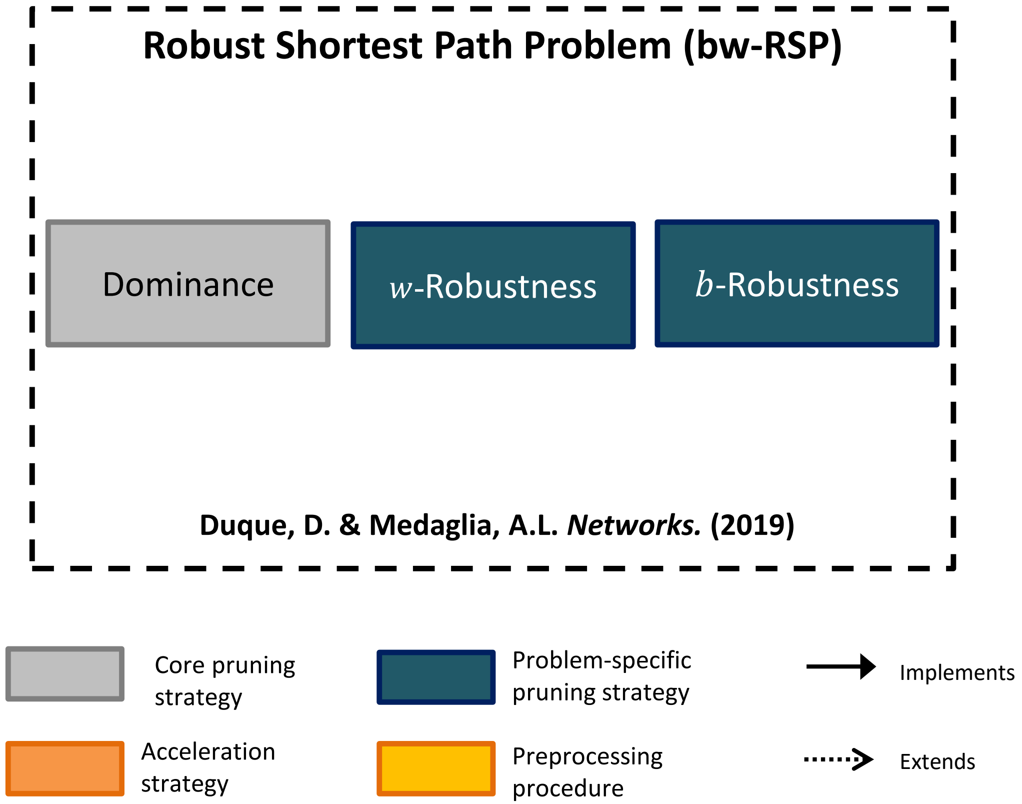
Pulse Algorithm for Hard Shortest Path Problems

Universidad de los Andes Colombia



bw-Robust Shortest Path (*bw*-RSP)

Pruning strategies



- Roy (2010)
- Gabrel, Murat & Wu (2013)

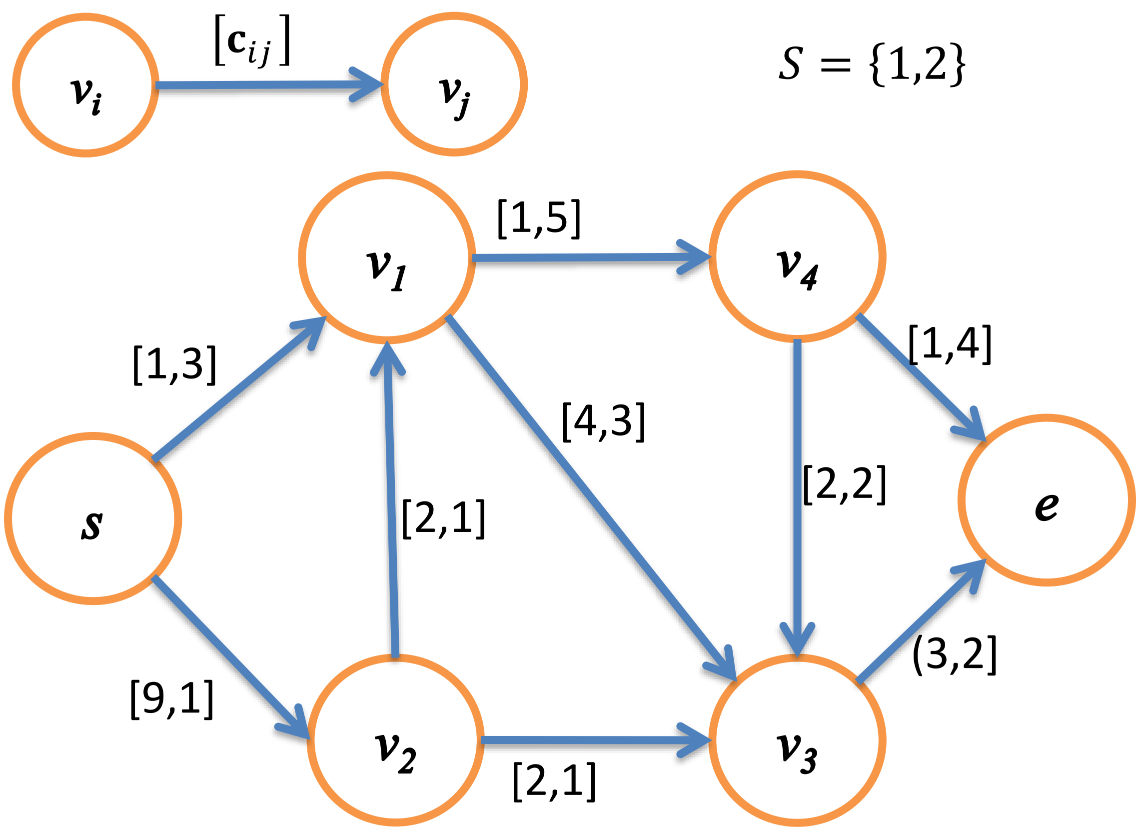
bw-Robust Shortest Path (*bw*-RSP)

Problem statement

- The bw-RSP is defined by:
  - Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$
    - $\mathcal{N} = \{v_1, \dots, v_i, \dots, v_n\}$
    - $\mathcal{A} = \{(i,j) | v_i \in \mathcal{N}, v_j \in \mathcal{N},\ i \neq j\}$
  - Set of scenarios $S = \{1, \dots, r\}$
  - Nonnegative weight c_{ij}^k that is the cost of traversing arc $(i,j) \in \mathcal{A}$ in scenario $k \in S$
  - Find a "robust" path starting at node v_s and ending at node v_e

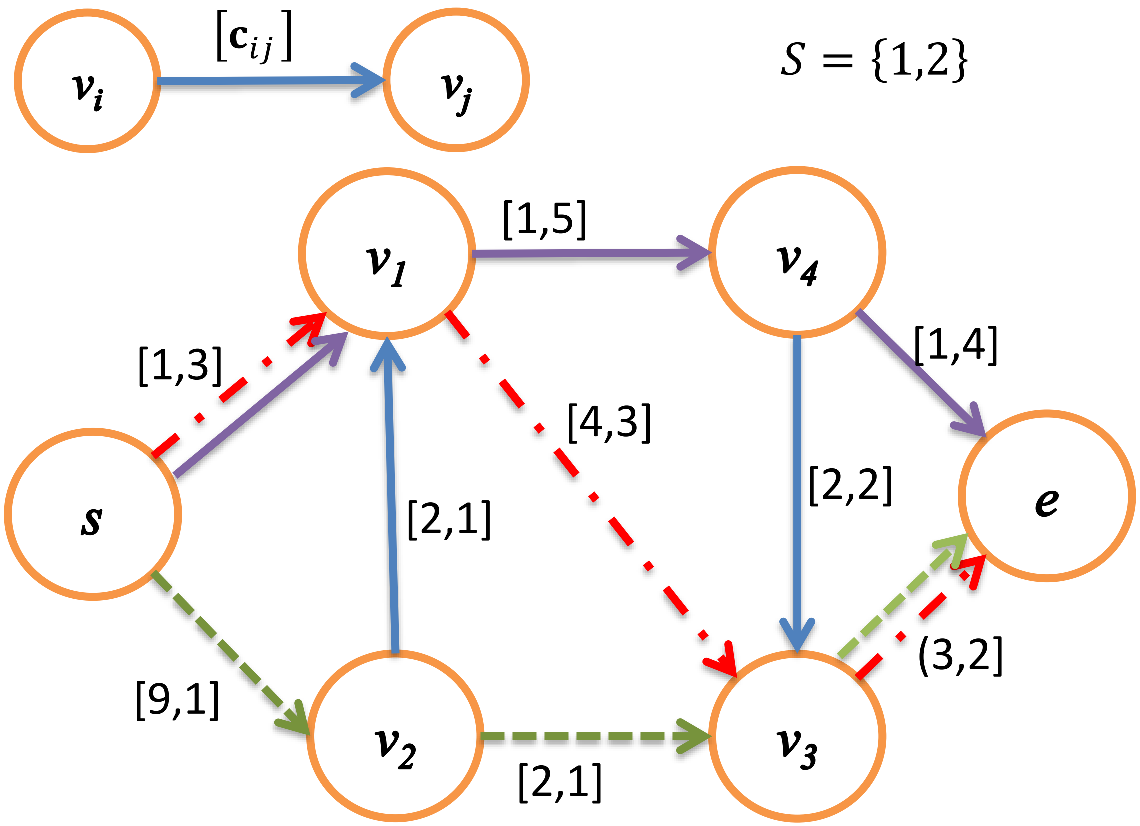
bw-Robust Shortest Path (*bw*-RSP)

Problem statement



bw-Robust Shortest Path (*bw*-RSP)

Problem statement



$S = \{1,2\}$

$\mathcal{P} \leftarrow \{s, v_1, v_4, e\}$
$c^1(\mathcal{P}) = 3$
$c^2(\mathcal{P}) = 12$

$\mathcal{P} \leftarrow \{s, v_2, v_3, e\}$
$c^1(\mathcal{P}) = 14$
$c^2(\mathcal{P}) = 4$

$\mathcal{P} \leftarrow \{s, v_1, v_3, e\}$
$c^1(\mathcal{P}) = 8$
$c^2(\mathcal{P}) = 8$

bw-Robust Shortest Path (*bw*-RSP)

Problem statement

Parameters (b < w):
b: *desirable* (target) bound that the decision maker wants for most scenarios
w: *strict* cost upper bound that the decision maker is not willing to exceed in any scenario

$$\max \sum_{k \in S} y_k$$

Maximizes the number of scenarios with a cost of the path $\leq\ b$

s.t.

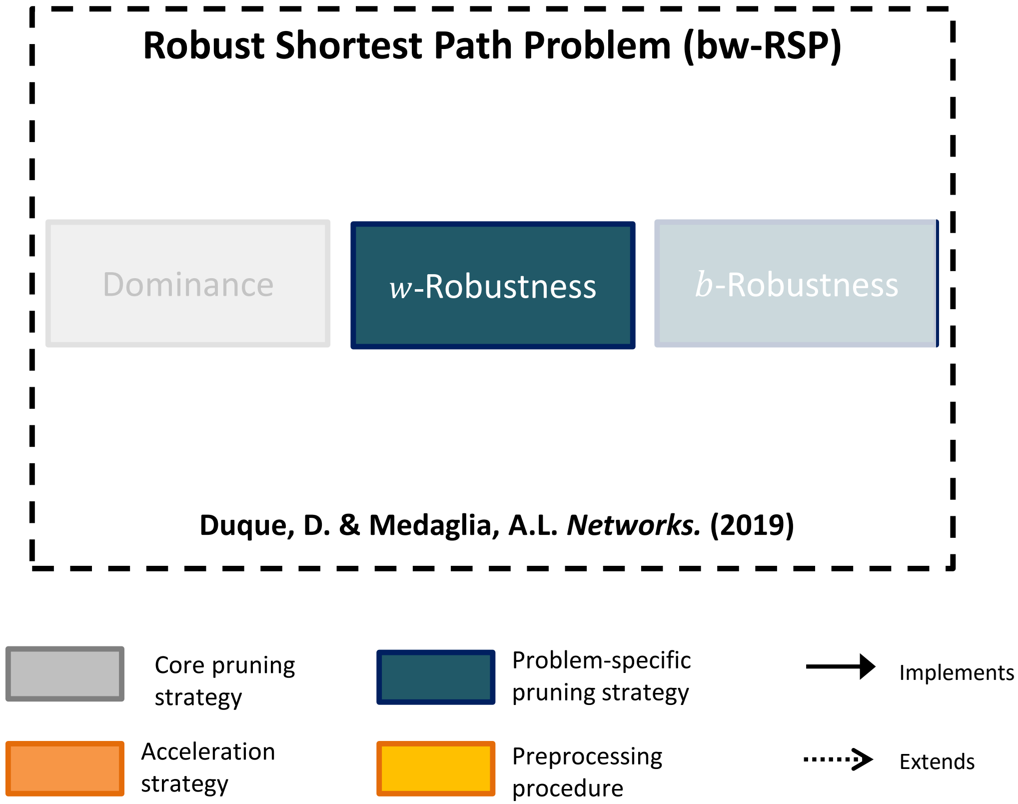
$$\sum_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij} \leq w(1 - y_k) + b y_k \qquad \forall k \in S$$

$$\sum_{\{j|(i,j)\in\mathcal{A}\}} x_{ij} - \sum_{\{j|(j,i)\in\mathcal{A}\}} x_{ji} = \begin{cases} 1, i = s \\ 0, i \neq s, e \\ -1, i = e \end{cases} \quad \forall i \in \mathcal{N}$$

$$y_k \in \{0,1\}, \forall k \in S$$
$$x_{ij} \in \{0,1\}\ \forall (i,j) \in \mathcal{A}$$

bw-Robust Shortest Path (*bw*-RSP)

Pruning by *w*-Robustness (infeasibility)



bw-Robust Shortest Path (*bw*-RSP)

Pruning by *w*-Robustness (infeasibility)

$$\sum_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij} \leq w(1-y_k) + by_k \qquad \forall k \in S$$

$$\boxed{b \leq w}$$

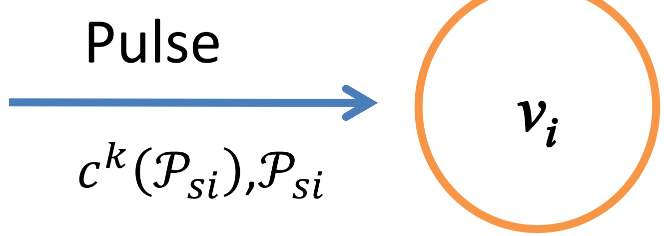
$$\Downarrow$$

$$\sum_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij} \leq w \qquad \forall k \in S$$

bw-Robust Shortest Path (*bw*-RSP)

Pruning by *w*-Robustness (infeasibility)

$S = \{1,2,3\}$



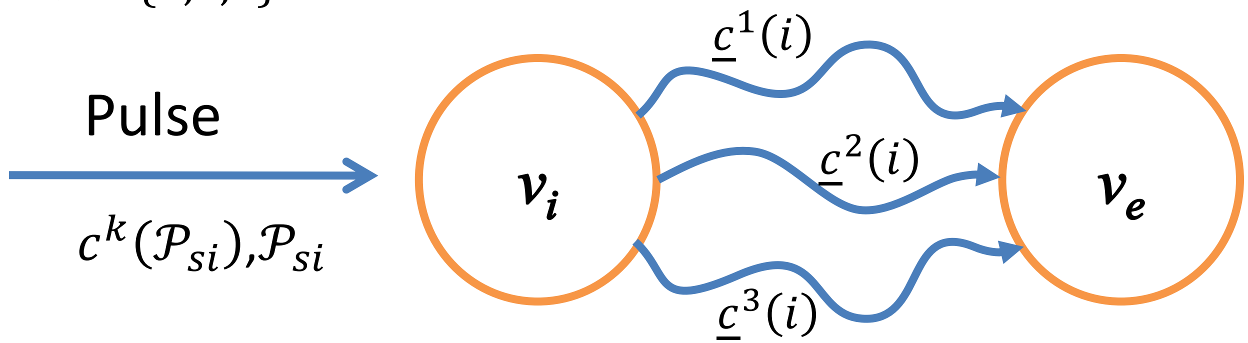
$c^k(\mathcal{P}_{si})$: cost of the path $\mathcal{P}_{si}$ under scenario $k \in S$

$\mathcal{P}_{si}$: partial path to v_i

bw-Robust Shortest Path (bw-RSP)

Pruning by w-Robustness (infeasibility)

$S = \{1,2,3\}$



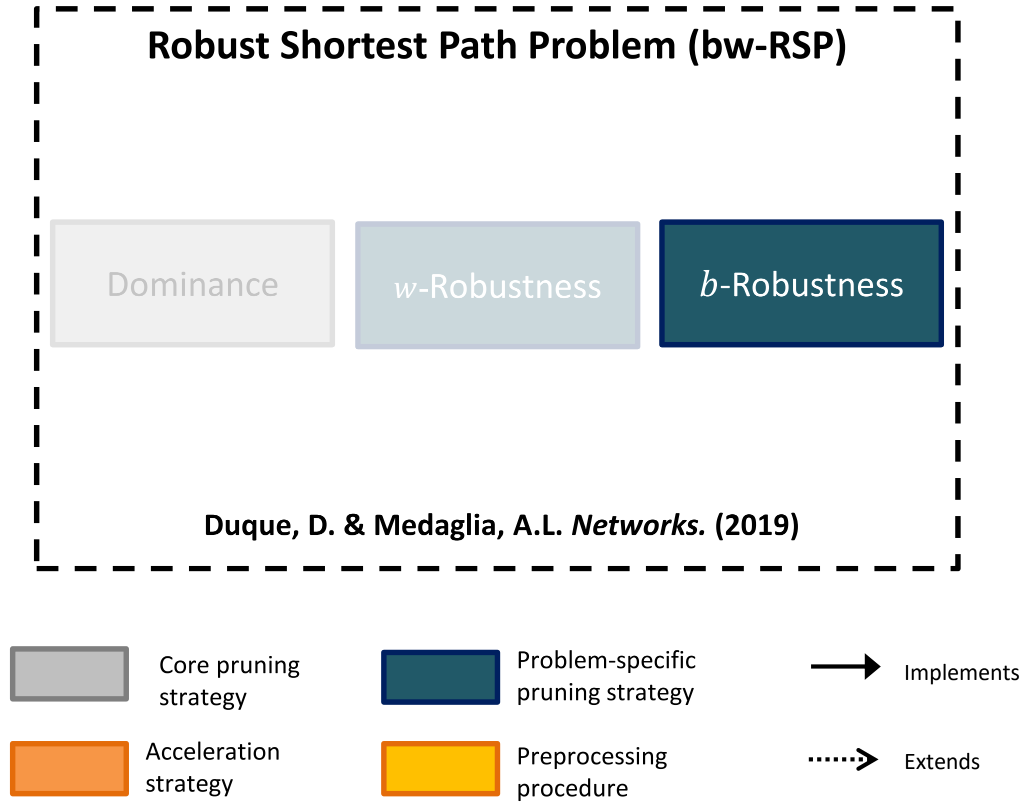
An incoming pulse to v_i is pruned if for any $k \in S$:

$$c^k(\mathcal{P}_{si}) + \boxed{\underline{c}^k(i)} > w$$

Lower bound on the cost under scenario $k \in S$ from v_i to v_e

bw-Robust Shortest Path (*bw*-RSP)

Pruning by *b*-Robustness (bounds)



bw-Robust Shortest Path (bw-RSP)

Pruning by b-Robustness (bounds)

$$\max \sum_{k \in S} y_k$$

$$\sum_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij} \le w(1 - y_k) + b y_k \qquad \forall k \in S$$

$$y_k = 1 \iff \sum_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij} \le b$$

bw-Robust Shortest Path (bw-RSP)

Pruning by b-Robustness (bounds)

- Let $\underline{y}$ be a **lower bound** of the **objective function**.
  - If a feasible path exists, $0 \leq \underline{y} \leq |S|$

- Let $\bar{y}(\mathcal{P}_{si})$ be an **upper bound** for any partial path $\mathcal{P}_{si}$ to node v_i.

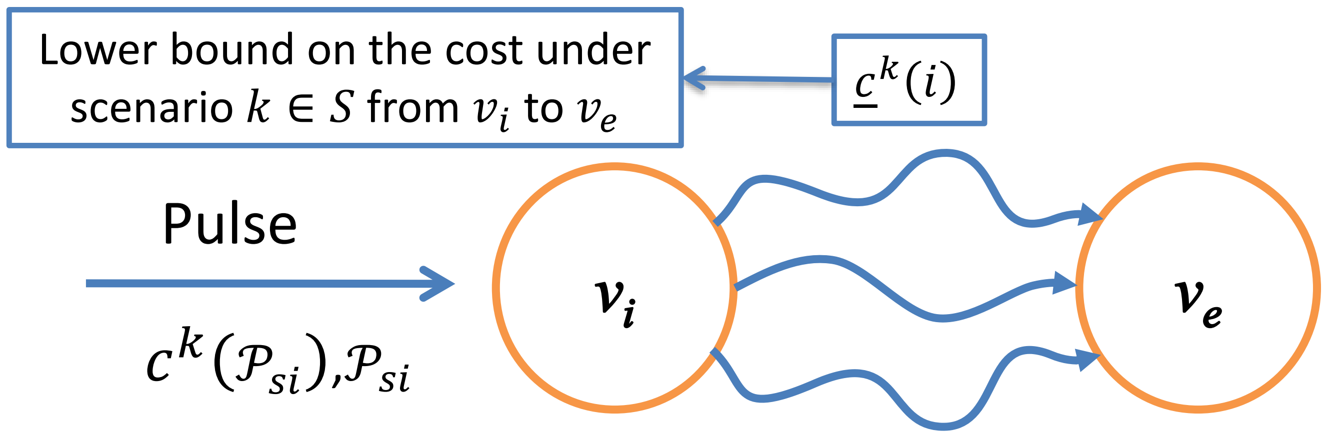
$$\bar{y}(\mathcal{P}_{si}) = \sum_{k \in S} 1_{B^k}(\mathcal{P}_{si})$$

$$1_{B^k}(\mathcal{P}_{si}) = \begin{cases} 1, if\ \mathcal{P}_{si} \in B^k \\ 0, if\ \mathcal{P}_{si} \notin B^k \end{cases}$$

$$\boxed{B^k = \{\mathcal{P}_{si} | c^k(\mathcal{P}_{si}) + \underline{c}^k(v_i) \leq b\}}$$

bw-Robust Shortest Path (*bw*-RSP)

Pruning by *b*-Robustness (bounds)



An incoming pulse to v_i is pruned if:

$$\bar{y}(\mathcal{P}_{si}) \leq \underline{y}$$

$$\bar{y}(\mathcal{P}_{si}) = \sum_{k \in S} 1_{B^k}(\mathcal{P}_{si})$$

bw-Robust Shortest Path (*bw*-RSP)

Computational experiments

- Setup
  - Pulse algorithm is coded in Java and compiled with eclipse SDK 4.4.0.
  - Core Xeon E5-2695 2.42GHz (10 cores) with 32GB of RAM allocated to JVM.
  - IP model solved using Gurobi 6.0
  - Six networks derived from the CSP (Beasley and Christofides, 1989)
    - Nodes: 100 – 500
    - Arcs: 990 – 4,868
  - Scenarios
    - From 10 to 10,000
  - Costs: randomly generated following a gamma distribution $\Gamma(\alpha, \theta)$
    - $\alpha = \{1,2,3\}$
    - $\theta = \mu_{ij}/\alpha$
  - Six experiments per instance varying parameters w, b.
    - Three values for w
    - Two values for b

bw-Robust Shortest Path (*bw*-RSP)

Computational experiments

Network	Nodes	Arcs	Scenarios	IP		
				Solved	Avg. Time (s)	Max. Time
rcsp5	100	990	10	6/6	0.099	0.125
			100	6/6	0.666	0.687
			1,000	6/6	11.484	11.576
			10,000	6/6	246.843	258.838
rcsp7	100	999	10	6/6	0.197	0.374
			100	6/6	2.707	8.783
			1,000	4/6	237.542*	>600
			10,000	3/6	482.828*	>600
rcsp13	200	2,080	10	6/6	0.255	0.297
			100	6/6	6.045	6.818
			1,000	6/6	151.257	253.782
			10,000	0/6	600.000*	>600
rcsp15	200	1,960	10	6/6	0.858	1.248
			100	6/6	12.878	35.693
			1,000	3/6	402.022*	>600
			10,000	0/6	600.000*	>600
rcsp21	500	4,847	10	6/6	2.714	4.789
			100	6/6	19.999	32.479
			1,000	4/6	450.240*	>600
			10,000	0/6	600.000*	>600
rcsp23	500	4,868	10	6/6	1.485	1.544
			100	4/6	242.439*	>600
			1,000	2/6	495.769*	>600
			10,000	0/6	600.000*	>600

* Average time is calculated with a computational time of 600 seconds for unsolved instances

bw-Robust Shortest Path (*bw*-RSP)

Computational experiments

Network	Nodes	Arcs	Scenarios	IP			Pulse			Speedup
				Solved	Avg. Time (s)	Max. Time	Solved	Avg. Time (s)	Max. Time	
rcsp5	100	990	10	6/6	0.099	0.125	6/6	0.005	0.005	19
			100	6/6	0.666	0.687	6/6	0.042	0.042	16
			1,000	6/6	11.484	11.576	6/6	0.178	0.178	65
			10,000	6/6	246.843	258.838	6/6	2.080	2.081	119
rcsp7	100	999	10	6/6	0.197	0.374	6/6	0.005	0.005	42
			100	6/6	2.707	8.783	6/6	0.021	0.023	126
			1,000	4/6	237.542*	>600	6/6	0.183	0.191	>1298
			10,000	3/6	482.828*	>600	6/6	1.894	1.959	>255
rcsp13	200	2,080	10	6/6	0.255	0.297	6/6	0.003	0.003	97
			100	6/6	6.045	6.818	6/6	0.018	0.018	336
			1,000	6/6	151.257	253.782	6/6	0.189	0.190	800
			10,000	0/6	600.000*	>600	6/6	1.864	1.870	>322
rcsp15	200	1,960	10	6/6	0.858	1.248	6/6	0.002	0.002	366
			100	6/6	12.878	35.693	6/6	0.021	0.027	613
			1,000	3/6	402.022*	>600	6/6	0.199	0.237	>2017
			10,000	0/6	600.000*	>600	6/6	2.097	2.185	>286
rcsp21	500	4,847	10	6/6	2.714	4.789	6/6	0.003	0.003	898
			100	6/6	19.999	32.479	6/6	0.029	0.030	689
			1,000	4/6	450.240*	>600	6/6	0.264	0.273	>1708
			10,000	0/6	600.000*	>600	6/6	3.453	3.660	>174
rcsp23	500	4,868	10	6/6	1.485	1.544	6/6	0.003	0.003	469
			100	4/6	242.439*	>600	6/6	0.035	0.044	>6958
			1,000	2/6	495.769*	>600	6/6	0.305	0.391	>1627
			10,000	0/6	600.000*	>600	6/6	3.582	4.405	>168

* Average time is calculated with a computational time of 600 seconds for unsolved instances

Lecciones aprendidas

Intuición y extensiones

- El pulso es un algoritmo de búsqueda profunda originalmente diseñado para resolver de forma exacta (y efectiva) el CSP

- El pulso usa estrategias de poda generales como lo son:
  - Dominancia
  - Cotas
  - Infactibilidad

- El pulso ha probado ser efectivo y es considerado como uno de los mejores algoritmos exactos para resolver el CSP
  - BD-A* - Thomas, Calogiuri & Hewitt (2019)
  - k-SP - Sedeño-Noda & Alonso-Rodríguez (2015)

Lecciones aprendidas

Intuición y extensiones

- La extensión al WCSPP-R aporta al pulso:
  - Path completion
  - Pulse queue
  - Best-promise order (dequeueing)
- La extensión a ESPPRC aporta al pulso:
  - Rollback
  - Bounding matrix
- La extensión a BSP aporta al pulso:
  - Efficient set
  - Nadir point
- La extensión a OPTW aporta al pulso:
  - Soft dominance
  - Detour
- La extensión a bw-Robustness aporta al pulso:
  - b-Robustness
  - w-Robustness

Referencias

- Bolívar, M. A., Lozano, L. & Medaglia, A. L. (2014). Acceleration Strategies for the Weight Constrained Shortest Path Problem with Replenishment. Optimization Letters. 8(8): 2155-2172. DOI:10.1007/s11590-014-0742-x [Pulse extension: WCSPP-R]
- Duque, D., Lozano, L. & Medaglia, A. L. (2015a). An exact method for the biobjective shortest path problem for large-scale road networks. European Journal of Operational Research. 242:788-797. DOI:10.1016/j.ejor.2014.11.003 [Pulse extension: BSP]
- Duque, D., Lozano, L. & Medaglia, A. L. (2015b). Solving the Orienteering Problem with Time Windows via the Pulse Framework. Computers & Operations Research. 54:168-176. DOI: 10.1016/j.cor.2014.08.019 [Pulse extension: OPTW]

Referencias

- Lozano, L. and Medaglia, A. L. (2013). On an exact method for the constrained shortest path problem. Computers & Operations Research. 40 (1):378-384. DOI:10.1016/j.cor.2012.07.008 [Pulse core paper: CSP]
- Duque, D. & Medaglia, A. L. (2019). An exact method for a class of robust shortest path problems with scenarios. Networks. DOI: 10.1002/net.21909. Available at: https://doi.org/10.1002/net.21909 [Pulse extension: bw-Robust SP]
- Lozano, L., Duque, D. & Medaglia, A. L. (2016). An exact algorithm for the elementary shortest path problem with resource constraints. Transportation Science. 50(1):348–357. DOI:10.1287/trsc.2014.0582 [Pulse extension: ESPPRC]
- Medaglia, A. L., Lozano, L., & Duque, D. (2018). Solving hard shortest path problems with the pulse framework. In IFORS News. Tutorial Section. 12(2). Available at: http://ifors.org/newsletter/ifors-news-june-2018.pdf , ISSN: 2223-4373 [Pulse core paper]