RÉPUBLIQUE
FRANÇAISE
Liberté
Égalité
Fraternité

ONERA
THE FRENCH AEROSPACE LAB

Orbit Slot Allocation in Earth Observation Constellations

Sara Maqrot Stéphanie Roussel ***Gauthier Picard*** Cédric Pralet
ONERA/DTIS, Université de Toulouse, France

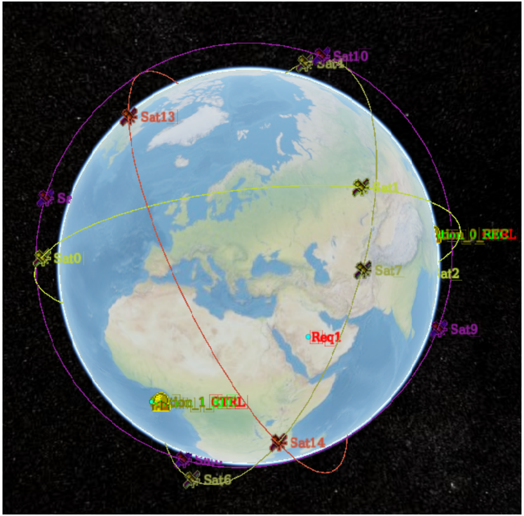
11th International Conference on Prestigious Applications of Intelligent Systems, 25/07/2022

Introduction
Applicative Context and Motivation

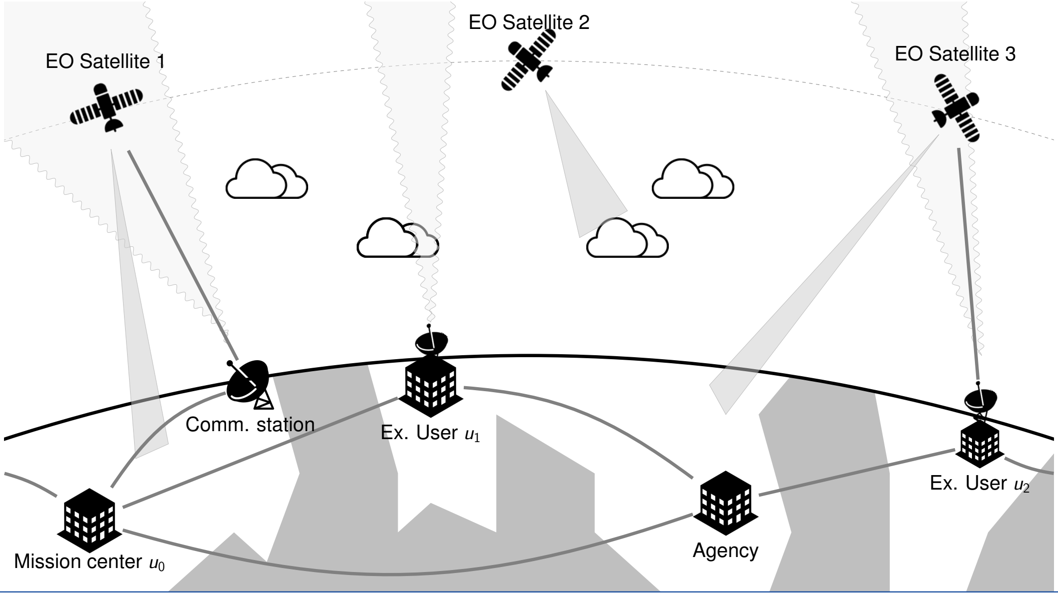
LiChIE projet for a new constellation for observing Earth (ADS, INRIA, ONERA, IXBLUE, EREMS)

- **Huge number of image requests** to (a single) mission center
- More and more **complex requests** (periodic, systematic, etc.)
- **Overloading** and **over-constraining**
- **No guarantee** for end-users

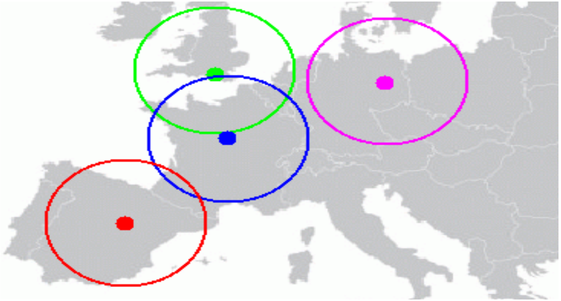
⇒ New paradigm: **orbit slot ownership**



Introduction
Applicative Context and Motivation



Introduction
Illustrative Example

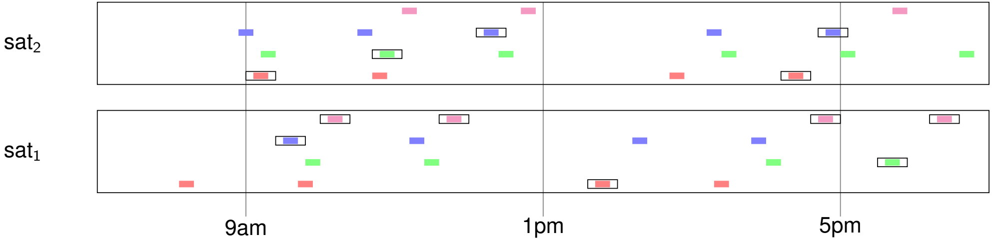


r_4 (global-30min)

r_3 (periodic-4h)

r_2 (periodic-4h)

r_1 (periodic-4h)



Introduction
Objective

- **Problem :** exploitation of the same constellation by several stakeholders

Offline reservation orbit slots —— Exclusivity periods → **Online planning** Image acquisition

- **Current allocation scheme:** first come, first served

- **Objective**

Requests (POI, dates tolerance, ...) → **Candidate slots** (Objects to allocate) → **Utilitarian** or **fair allocation**

Today's Menu

Today's Menu

General Definitions

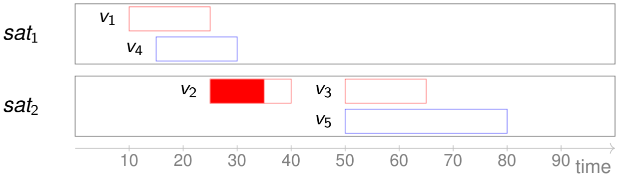
Satellite reservation window

A *satellite reservation window* w is defined by

- a satellite sat_w
- a time window $[start_w, end_w]$
- an individual score ω_w

Allocated orbit slot

An *allocated orbit slot* o within satellite reservation window w corresponds to a time window $[start_o, end_o]$ included in $[start_w, end_w]$

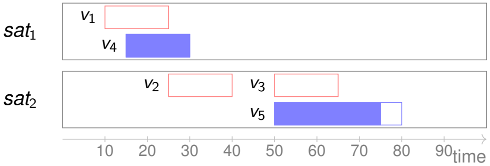


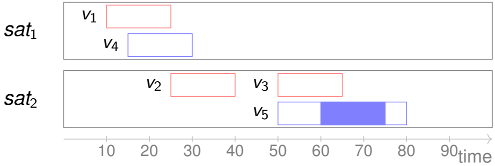
Global Allocation Request

Global allocation request

A (multi-mode) *global allocation request* r is defined by

- a set of satellite reservation windows $\mathcal{V}_r$
- a minimum duration $minSlotDur_r$ for each slot
- a list of allocation modes $\mathcal{M}_r = [\mathcal{M}_{r,1}, \ldots, \mathcal{M}_{r,K}]$, where for all $m \in \mathcal{M}_r$
  - m is an alternative to fulfill r
  - $globalDur_m$ is a global duration required over all orbit slots reserved in $\mathcal{V}_r$



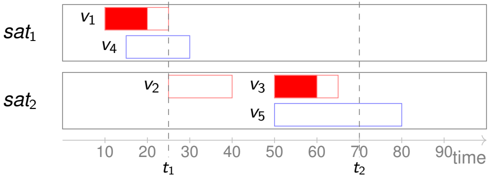


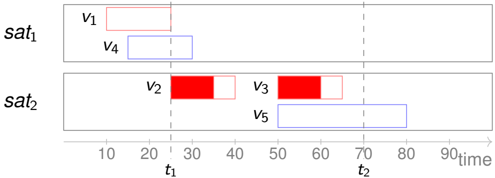
Time-tagged Allocation Request

Time-tagged allocation request

A *time-tagged allocation request* r is defined by

- a set of satellite reservation windows $\mathcal{V}_r$
- a minimum duration $minSlotDur_r$ for each slot
- a set of time references $\mathcal{T}_r$, with for each time reference $t \in \mathcal{T}_r$ a subset $\mathcal{V}^r_t \subseteq \mathcal{V}_r$ that defines the reservation windows associated with t
- a list of allocation modes $\mathcal{M}_r = [\mathcal{M}_{r,1}, \ldots, \mathcal{M}_{r,K}]$, where each allocation mode $m \in \mathcal{M}_r$ is defined by a subset $\mathcal{T}_m \subseteq \mathcal{T}_r$ of time references around which an orbit slot must actually be reserved





Orbit Slot Allocation Problem

Orbit Slot Allocation Problem

An *Orbit Slot Allocation Problem* (OSAP) is defined by

- a set of satellites $\mathcal{S}$
- a set of requests $\mathcal{R} = \mathcal{R}_G \cup \mathcal{R}_T$
  - $\mathcal{R}_G$ a set of global allocation requests
  - $\mathcal{R}_T$ a set of tagged-time allocation requests

Solution for an OSAP

A *solution* $\mathcal{A}$ for an OSAP is defined by one allocation $\mathcal{A}_r$ for each $r \in \mathcal{R}$

A solution is said to be feasible if and only if

- for every request r, allocation $\mathcal{A}_r$ satisfies request r
- for each satellite $s \in \mathcal{S}$, there is no overlapping between the orbit slots booked over s for all allocations in $\{\mathcal{A}_r \mid r \in \mathcal{R}\}$

Orbit Slot Allocation Problem

Orbit Slot Allocation Problem

An *Orbit Slot Allocation Problem* (OSAP) is defined by

- a set of satellites $\mathcal{S}$
- a set of requests $\mathcal{R} = \mathcal{R}_G \cup \mathcal{R}_T$
  - $\mathcal{R}_G$ a set of global allocation requests
  - $\mathcal{R}_T$ a set of tagged-time allocation requests

OSAP is NP-hard

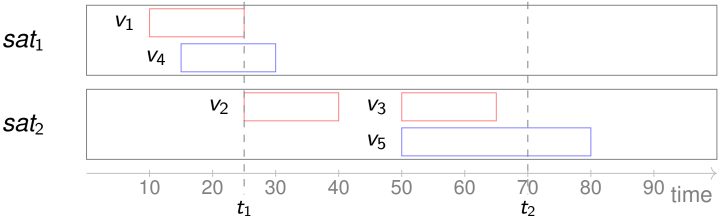
Solution for an OSAP

A *solution* $\mathcal{A}$ for an OSAP is defined by one allocation $\mathcal{A}_r$ for each $r \in \mathcal{R}$

A solution is said to be feasible if and only if

- for every request r, allocation $\mathcal{A}_r$ satisfies request r
- for each satellite $s \in \mathcal{S}$, there is no overlapping between the orbit slots booked over s for all allocations in $\{\mathcal{A}_r \mid r \in \mathcal{R}\}$

Example



- a time-tagged allocation request A (in red)
  - $minSlotDur_A = 10$
  - $\mathcal{V}_A = \{v_1, v_2, v_3\}$ with $v_1 = [10, 25]$, $v_2 = [25, 40]$, $v_3 = [50, 65]$
  - $\mathcal{V}_{r_1}^{t_1} = \{v_1, v_2\}$ and $\mathcal{V}_{r_1}^{t_2} = \{v_3\}$
  - 3 modes a_1, a_2 and a_3, with $\mathcal{T}_{a_1} = \emptyset$, $\mathcal{T}_{a_2} = \{t_1\}$ and $\mathcal{T}_{a_3} = \{t_1, t_2\}$
- a global allocation request B (in blue)
  - $minSlotDur_B = 15$
  - $\mathcal{V}_B = \{v_4, v_5\}$ with $v_4 = [15, 30]$, $v_5 = [50, 80]$
  - 3 modes b_1, b_2 and b_3, with global duration of 0, 15 and 40

How to Assess Allocation Quality?

Mode reward for global allocation request

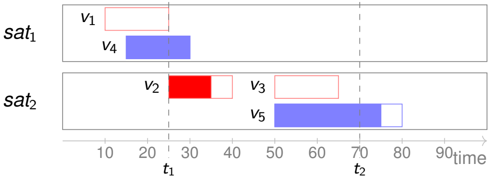
For a global allocation request r and a possible mode $m \in \mathcal{M}_r$, the reward Ω_m associated with m corresponds to quantity $globalDur_m$

Mode reward for time-tagged allocation request

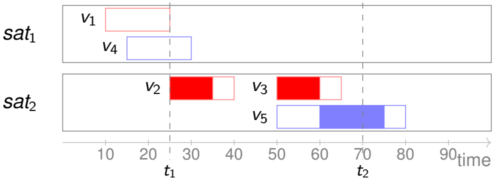
For a time-tagged allocation request r and a possible mode $m \in \mathcal{M}_r$, the reward Ω_m associated with mode m corresponds to quantity $|\mathcal{T}_m| \cdot minSlotDur_r$, that is to the total satellite time required by m over all its relevant time references

- Mode utility
  - *utilitarian allocation*: maximizing $u(\mathcal{A}) = \sum_{r \in \mathcal{R}} \Omega_{m(\mathcal{A}_r)}$
  - *fair (leximin) allocation*: maximizing $\vec{u}(\mathcal{A}) = [\Omega_{m(\mathcal{A}_{r_1})}, \ldots, \Omega_{m(\mathcal{A}_{r_n})}]$
- Window utility : $u^{slot}(\mathcal{A}) = \sum_{r \in \mathcal{R}} \sum_{v \in \mathcal{A}_r} \omega_v$

Example



(a) Utilitarian-optimal allocation $\mathcal{A}_{util}$



(b) Leximin-optimal allocation $\mathcal{A}_{fair}$

- modes a_2 for A and b_3 for B
- $u(\mathcal{A}_{util}) = 10 + 40 = 50$
- $\vec{u}(\mathcal{A}_{util}) = [10, 40]$

- modes a_3 for A and b_2 for B
- $u(\mathcal{A}_{util}) = 20 + 15 = 35$
- $\vec{u}(\mathcal{A}_{util}) = [15, 20]$

Today's Menu

Utilitarian CP Encoding

$$\text{maximize} \quad \sum_{r \in \mathcal{R}} \sum_{m \in \mathcal{M}_r} \Omega_m \cdot \mathrm{x}_m \tag{1}$$

$$\text{s.t.} \quad (2), (3), (4), (5), (6)$$

- itv_v: interval variables with minimun size of $minSlotDur_r$
- x_m: boolean variable for choosing mode m

$$\forall r \in \mathcal{R}, \quad \sum_{m \in \mathcal{M}_r} \mathrm{x}_m = 1 \tag{2}$$

$$\forall s \in S, \quad noOverlap(\{\mathrm{itv}_v | v \in \bigcup_{r \in \mathcal{R}} \mathcal{V}_r \wedge sat_v = s\}) \tag{3}$$

$$\forall r \in \mathcal{R}_T, \forall m \in \mathcal{M}_r, \forall t \in \mathcal{T}_m, \quad \sum_{v \in \mathcal{V}_r^t} presenceOf(\mathrm{itv}_v) \geq \mathrm{x}_m, \tag{4}$$

$$\forall r \in \mathcal{R}_T, \forall t \in \mathcal{T}_r, \quad \sum_{v \in \mathcal{V}_t} presenceOf(\mathrm{itv}_v) \leq 1 \tag{5}$$

$$\forall r \in \mathcal{R}_G, \forall m \in \mathcal{M}_r, \quad \sum_{v \in \mathcal{V}_r} lengthOf(\mathrm{itv}_v) \geq \mathrm{x}_m \cdot globalDur_r \tag{6}$$

Leximin CP Encoding

Intuition

- Solve as many CP optimization problems as requests
- The k-th CP problem allows to compute the k-th component of the sorted leximin vector $\vec{u} = [u_1, \ldots, u_n]$
- The objective is to lexicographically maximize vector $\Lambda = [\Lambda_1, \ldots, \Lambda_n]$ obtained after ordering $[u_1, \ldots, u_n]$ following an increasing order

- Variables
  - $\lambda \in [\Lambda_{K-1}, \max_{r \in \mathcal{R}} Z_r]$ is a real variable representing the utility obtained at level K in Λ
  - y_{rk} is a binary variable equal to 1 if request $r \in \mathcal{R}$ plays the role of the request associated with level $k \in [1..K-1]$ in $[\Lambda_1, \ldots, \Lambda_{K-1}]$, 0 otherwise
  - u_r is a real variable in $[0, Z_r]$ representing the utility of request r

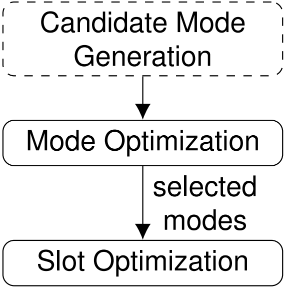
Leximin CP Encoding (cont.)

- Determining the Kth level

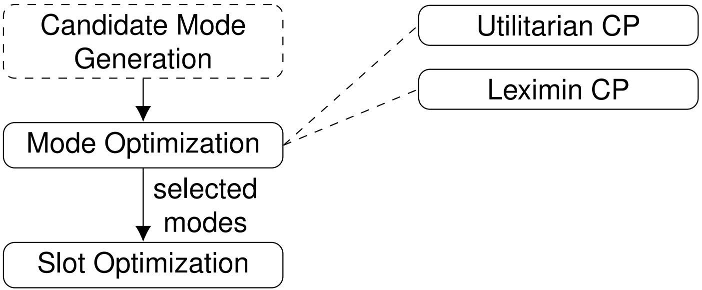
$$
\begin{aligned}
\text{maximize} \quad & \lambda && (7)\\
\text{s.t.} \quad & (2), (3), (4), (5), (6) && \\
\forall r \in \mathcal{R}, \quad & \mathrm{u}_r = \sum_{m \in \mathcal{M}_r} \Omega_m \cdot \mathrm{x}_m && (8)\\
\forall k \in [1..K-1], \quad & \sum_{r \in \mathcal{R}} \mathrm{y}_{rk} = 1 && (9)\\
\forall r \in \mathcal{R}, \quad & \sum_{k \in [1..K-1]} \mathrm{y}_{rk} \leq 1 && (10)\\
\forall r \in \mathcal{R}, \quad & \lambda \leq \mathrm{u}_r + M \sum_{k \in [1..K-1]} \mathrm{y}_{rk} && (11)\\
\forall r \in \mathcal{R}, \quad & \mathrm{u}_r \geq \sum_{k \in [1..K-1]} \Lambda_k \cdot \mathrm{y}_{rk} && (12)
\end{aligned}
$$

Today’s Menu

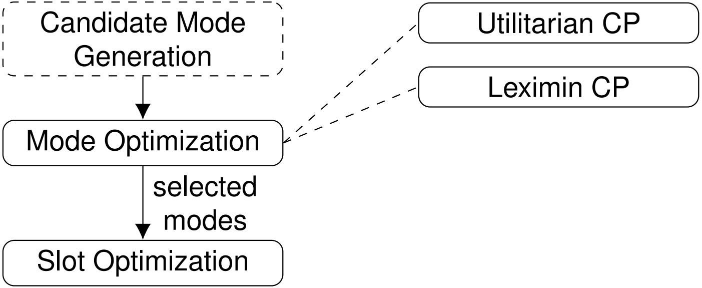
Optimization Architecture



Optimization Architecture

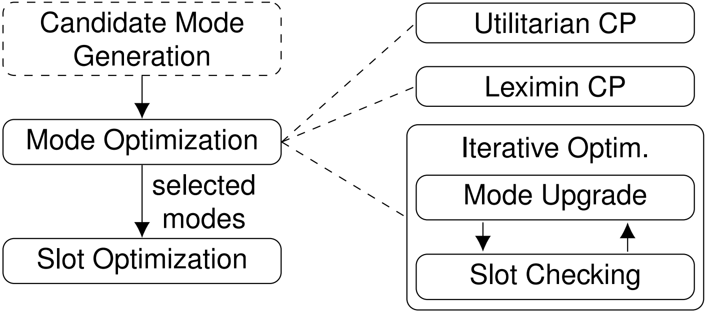


Optimization Architecture



- Major issue: optimal CP approaches won't scale up!

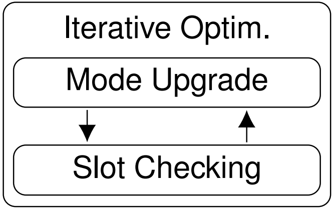
Optimization Architecture



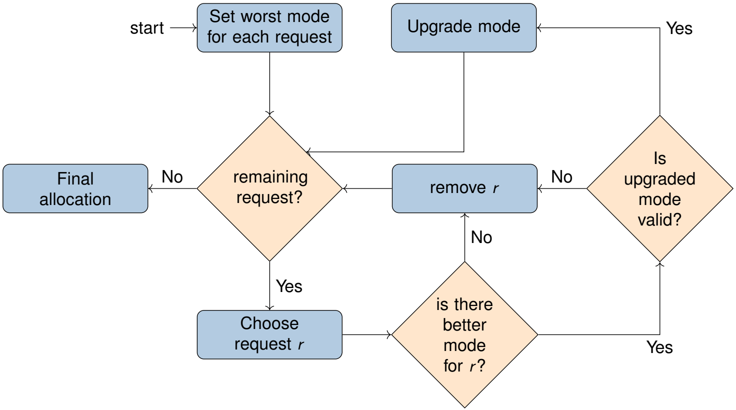
- Major issue: optimal CP approaches won't scale up!
- Proposal: iterative heuristic approach based on mode quality upgrade

Iterative Optimization

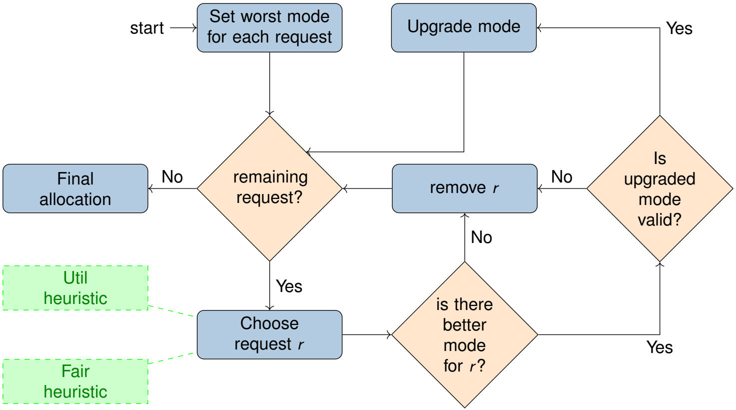
- **Purposes**: balance between utilitarianism and fairness, and scale up
- **Mode Upgrade** produces allocations so that
  - Constraint (2) is satisfied
  - Criteria (1) and (7) are optimized
- **Slot Checking** layer checks Constraints (2)– (6)



Iterative Optimization (cont.)



Iterative Optimization (cont.)



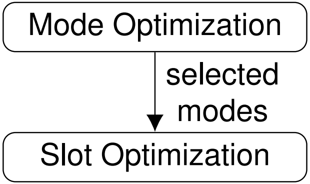
- h^{util} selects the request whose next mode increases the most the global utility of the allocation
- h^{fair} selects the request with the smallest utility

Slot Optimization

- **Purpose:** Optimizing the **slots** for the modes that have been selected at the Mode Optimization step
- **Maximizing the window utility**

$$\begin{aligned} \text{maximize} \quad & \sum_{r \in \mathcal{R}} \sum_{v \in \mathcal{V}_r} \omega_v \cdot \textit{presenceOf}(\mathrm{itv}_v) \quad (13) \\ \text{s.t.} \quad & (2), (3), (4), (5), (6) \end{aligned}$$

with $x_m = 1$ iff m is selected by the Mode Optimization module



Today's Menu

Experimental Setup

Constellation

- Low-Earth Orbit (500km altitude)
- 8 mono-satellite orbital planes with a 60 degrees inclination

Requests

1. randomly selecting a subset of national capitals $\mathcal{C}$
2. randomly parameterize requests
   - Global requests
     - most preferred global duration in $[2h, 4h]$
     - less preferred modes remove 30min down to 0 ($\mathcal{M}_{r,1}$)
     - minimum slot duration in [2min, 4min]
   - Time-tagged requests
     - 2 time references patterns: [8am, 12pm, 4pm, 8pm] and [9am, 1pm, 5pm]
     - 1-hour tolerance: $[t - 1\text{hour}, t + 1\text{hour}]$
     - less preferred mode $\mathcal{M}_{r,1}$ has an empty set of time references
     - more preferred modes add one random time reference
3. randomly picking a national capital in $\mathcal{C}$ as a ground station for each request

Experimental Setup (cont.)

Orderbooks

- 5 different order book instances per configuration (defined by $|\mathcal{R}_G|$ and $|\mathcal{R}_T|$)
- For slot optimization, the reward is linear in [0,1]
- The number of requests we consider is larger than current realistic data

Computing Environment

- Solvers are coded in Java 1.8
- Executon on 20-core Intel(R) Xeon(R) CPU E5-2660 v3 @ 2.60GHz, 62GB RAM, Ubuntu 18.04.5 LTS
- CP Optimizer included in IBM ILOG CPLEX Studio 20.1 is used by the solvers through the Java API, with timeouts

Method	**Mode opt.**	**Slot opt.**	**Slot check**
upgrade – util	n/a	300s	120s
upgrade – fair	n/a	300s	300s
cp – util / cp – fair	300s$\times|\mathcal{R}|$	300s	300s

Utility

configurations				cp — fair		cp — util		upgrade — fair		upgrade — util	
$\|\mathcal{R}_G\|$	$\|\mathcal{R}_T\|$	$\|\mathcal{M}\|$	$\|\mathcal{V}\|$	u	u^{slot}	u	u^{slot}	u	u^{slot}	u	u^{slot}
0	5	22.0	107.0	**1980.20**†	**4.43**	**1980.20**∗	**4.43**	**1980.20**	**4.43**	**1980.20**	**4.43**
0	10	44.6	218.2	3925.00†	**8.85**	**3953.40**∗	8.66	3925.00	**8.85**	**3953.40**	8.66
0	15	67.2	326.2	6260.40†	13.15	**6288.80**∗	12.96	6260.40	**13.16**	**6288.80**	12.96
0	20	90.0	439.6	8294.00†	**17.27**	**8322.40**	17.06	8294.00	17.25	**8322.40**	17.03
0	25	112.0	549.8	10313.20	21.09	**10341.60**	20.94	10313.20	**21.16**	10276.60	20.78
5	0	31.4	198.6	39874.00	**4.64**	39911.20	4.50	**42394.00**	4.31	42034.00	4.31
10	0	63.8	405.0	**44646.60**	9.20	42953.60	8.20	44286.60	**9.32**	44286.60	9.27
15	0	96.4	606.2	42109.20	13.29	42730.20	10.76	44291.60	13.51	**44420.00**	**13.90**
20	0	129.6	814.6	27927.20	9.80	40992.60	9.32	43131.20	**14.14**	**43409.00**	13.86
25	0	161.4	1018.2	28864.80	9.80	40489.20	9.30	39645.40	**13.87**	**43117.40**	13.23
5	5	53.8	311.0	39515.60	**8.97**	40998.60	8.43	42395.60	8.30	**44388.00**	7.23
10	10	109.6	627.0	43594.40	15.93	42664.40	15.52	44674.40	**15.98**	**47071.60**	13.98
15	15	165.2	944.0	34171.40	**23.29**	39015.00	20.00	46368.80	21.74	**47244.60**	19.10
20	20	219.4	1258.2	31823.00	**26.31**	41759.80	24.18	45223.60	25.16	**47728.40**	19.45
25	25	274.6	1572.8	29788.40	**28.87**	41641.00	26.05	46824.60	27.16	**47474.80**	21.24

Utility

configurations				cp − fair		cp − util		upgrade − fair		upgrade − util	
$\|\mathcal{R}_G\|$	$\|\mathcal{R}_T\|$	$\|\mathcal{M}\|$	$\|\mathcal{V}\|$	u	u^{slot}	u	u^{slot}	u	u^{slot}	u	u^{slot}
0	5	22.0	107.0	**1980.20†**	**4.43**	**1980.20∗**	**4.43**	**1980.20**	**4.43**	**1980.20**	**4.43**
0	10	44.6	218.2	3925.00†	**8.85**	**3953.40∗**	8.66	3925.00	**8.85**	**3953.40**	8.66
0	15	67.2	326.2	6260.40†	13.15	**6288.80∗**	12.96	6260.40	**13.16**	**6288.80**	12.96
0	20	90.0	439.6	8294.00†	**17.27**	**8322.40**	17.06	8294.00	17.25	**8322.40**	17.03
0	25	112.0	549.8	10313.20	21.09	**10341.60**	20.94	10313.20	**21.16**	10276.60	20.78
5	0	31.4	198.6	39874.00	**4.64**	39911.20	4.50	**42394.00**	4.31	42034.00	4.31
10	0	63.8	405.0	**44646.60**	9.20	42953.60	8.20	44286.60	**9.32**	44286.60	9.27
15	0	96.4	606.2	42109.20	13.29	42730.20	10.76	44291.60	13.51	**44420.00**	**13.90**
20	0	129.6	814.6	27927.20	9.80	40992.60	9.32	43131.20	**14.14**	**43409.00**	13.86
25	0	161.4	1018.2	28864.80	9.80	40489.20	9.30	39645.40	**13.87**	**43117.40**	13.23
5	5	53.8	311.0	39515.60	**8.97**	40998.60	8.43	42395.60	8.30	**44388.00**	7.23
10	10	109.6	627.0	43594.40	15.93	42664.40	15.52	44674.40	**15.98**	**47071.60**	13.98
15	15	165.2	944.0	34171.40	**23.29**	39015.00	20.00	46368.80	21.74	**47244.60**	19.10
20	20	219.4	1258.2	31823.00	**26.31**	41759.80	24.18	45223.60	25.16	**47728.40**	19.45
25	25	274.6	1572.8	29788.40	**28.87**	41641.00	26.05	46824.60	27.16	**47474.80**	21.24

Time-tagged-only allocation requests:

cp − util and upgrade − util provide the best mode utilitarian allocation

Utility

configurations				cp − fair		cp − util		upgrade − fair		upgrade − util	
$\|\mathcal{R}_G\|$	$\|\mathcal{R}_T\|$	$\|\mathcal{M}\|$	$\|\mathcal{V}\|$	u	u^{slot}	u	u^{slot}	u	u^{slot}	u	u^{slot}
0	5	22.0	107.0	**1980.20**†	**4.43**	**1980.20**∗	**4.43**	**1980.20**	**4.43**	**1980.20**	**4.43**
0	10	44.6	218.2	3925.00†	**8.85**	**3953.40**∗	8.66	3925.00	**8.85**	**3953.40**	8.66
0	15	67.2	326.2	6260.40†	13.15	**6288.80**∗	12.96	6260.40	**13.16**	**6288.80**	12.96
0	20	90.0	439.6	8294.00†	**17.27**	**8322.40**	17.06	8294.00	17.25	**8322.40**	17.03
0	25	112.0	549.8	10313.20	21.09	**10341.60**	20.94	10313.20	**21.16**	10276.60	20.78
5	0	31.4	198.6	39874.00	**4.64**	39911.20	4.50	**42394.00**	4.31	42034.00	4.31
10	0	63.8	405.0	**44646.60**	9.20	42953.60	8.20	44286.60	**9.32**	44286.60	9.27
15	0	96.4	606.2	42109.20	13.29	42730.20	10.76	44291.60	13.51	**44420.00**	**13.90**
20	0	129.6	814.6	27927.20	9.80	40992.60	9.32	43131.20	**14.14**	**43409.00**	13.86
25	0	161.4	1018.2	28864.80	9.80	40489.20	9.30	39645.40	**13.87**	**43117.40**	13.23
5	5	53.8	311.0	39515.60	**8.97**	40998.60	8.43	42395.60	8.30	**44388.00**	7.23
10	10	109.6	627.0	43594.40	15.93	42664.40	15.52	44674.40	**15.98**	**47071.60**	13.98
15	15	165.2	944.0	34171.40	**23.29**	39015.00	20.00	46368.80	21.74	**47244.60**	19.10
20	20	219.4	1258.2	31823.00	**26.31**	41759.80	24.18	45223.60	25.16	**47728.40**	19.45
25	25	274.6	1572.8	29788.40	**28.87**	41641.00	26.05	46824.60	27.16	**47474.80**	21.24

For small instances:
Utility-optimal and fairness-optimal allocations returned by cp − util and cp − fair

Utility

configurations				cp − fair		cp − util		upgrade − fair		upgrade − util	
$\|\mathcal{R}_G\|$	$\|\mathcal{R}_T\|$	$\|\mathcal{M}\|$	$\|\mathcal{V}\|$	u	u^{slot}	u	u^{slot}	u	u^{slot}	u	u^{slot}
0	5	22.0	107.0	**1980.20**†	**4.43**	**1980.20**∗	**4.43**	**1980.20**	**4.43**	**1980.20**	**4.43**
0	10	44.6	218.2	3925.00†	**8.85**	**3953.40**∗	8.66	3925.00	**8.85**	**3953.40**	8.66
0	15	67.2	326.2	6260.40†	13.15	**6288.80**∗	12.96	6260.40	**13.16**	**6288.80**	12.96
0	20	90.0	439.6	8294.00†	**17.27**	**8322.40**	17.06	8294.00	17.25	**8322.40**	17.03
0	25	112.0	549.8	10313.20	21.09	**10341.60**	20.94	10313.20	**21.16**	10276.60	20.78
5	0	31.4	198.6	39874.00	**4.64**	39911.20	4.50	**42394.00**	4.31	42034.00	4.31
10	0	63.8	405.0	**44646.60**	9.20	42953.60	8.20	44286.60	**9.32**	44286.60	9.27
15	0	96.4	606.2	42109.20	13.29	42730.20	10.76	44291.60	13.51	**44420.00**	**13.90**
20	0	129.6	814.6	27927.20	9.80	40992.60	9.32	43131.20	**14.14**	**43409.00**	13.86
25	0	161.4	1018.2	28864.80	9.80	40489.20	9.30	39645.40	**13.87**	**43117.40**	13.23
5	5	53.8	311.0	39515.60	**8.97**	40998.60	8.43	42395.60	8.30	**44388.00**	7.23
10	10	109.6	627.0	43594.40	15.93	42664.40	15.52	44674.40	**15.98**	**47071.60**	13.98
15	15	165.2	944.0	34171.40	**23.29**	39015.00	20.00	46368.80	21.74	**47244.60**	19.10
20	20	219.4	1258.2	31823.00	**26.31**	41759.80	24.18	45223.60	25.16	**47728.40**	19.45
25	25	274.6	1572.8	29788.40	**28.87**	41641.00	26.05	46824.60	27.16	**47474.80**	21.24

With more than 5 time-tagged requests:
utilitarian and fair approaches converge to different optima

Utility

configurations				cp − fair		cp − util		upgrade − fair		upgrade − util	
$\|\mathcal{R}_G\|$	$\|\mathcal{R}_T\|$	$\|\mathcal{M}\|$	$\|\mathcal{V}\|$	u	u^{slot}	u	u^{slot}	u	u^{slot}	u	u^{slot}
0	5	22.0	107.0	**1980.20**†	**4.43**	**1980.20**∗	**4.43**	**1980.20**	**4.43**	**1980.20**	**4.43**
0	10	44.6	218.2	3925.00†	**8.85**	**3953.40**∗	8.66	3925.00	**8.85**	**3953.40**	8.66
0	15	67.2	326.2	6260.40†	13.15	**6288.80**∗	12.96	6260.40	**13.16**	**6288.80**	12.96
0	20	90.0	439.6	8294.00†	**17.27**	**8322.40**	17.06	8294.00	17.25	**8322.40**	17.03
0	25	112.0	549.8	10313.20	21.09	**10341.60**	20.94	10313.20	**21.16**	10276.60	20.78
5	0	31.4	198.6	39874.00	**4.64**	39911.20	4.50	**42394.00**	4.31	42034.00	4.31
10	0	63.8	405.0	**44646.60**	9.20	42953.60	8.20	44286.60	**9.32**	44286.60	9.27
15	0	96.4	606.2	42109.20	13.29	42730.20	10.76	44291.60	13.51	**44420.00**	**13.90**
20	0	129.6	814.6	27927.20	9.80	40992.60	9.32	43131.20	**14.14**	**43409.00**	13.86
25	0	161.4	1018.2	28864.80	9.80	40489.20	9.30	39645.40	**13.87**	**43117.40**	13.23
5	5	53.8	311.0	39515.60	**8.97**	40998.60	8.43	42395.60	8.30	**44388.00**	7.23
10	10	109.6	627.0	43594.40	15.93	42664.40	15.52	44674.40	**15.98**	**47071.60**	13.98
15	15	165.2	944.0	34171.40	**23.29**	39015.00	20.00	46368.80	21.74	**47244.60**	19.10
20	20	219.4	1258.2	31823.00	**26.31**	41759.80	24.18	45223.60	25.16	**47728.40**	19.45
25	25	274.6	1572.8	29788.40	**28.87**	41641.00	26.05	46824.60	27.16	**47474.80**	21.24

With global allocation requests:
Higher reward but difficult to solve by cp − fair and cp − util

Utility

configurations				cp — fair		cp — util		upgrade — fair		upgrade — util	
$\lvert\mathcal{R}_G\rvert$	$\lvert\mathcal{R}_T\rvert$	$\lvert\mathcal{M}\rvert$	$\lvert\mathcal{V}\rvert$	u	u^{slot}	u	u^{slot}	u	u^{slot}	u	u^{slot}
0	5	22.0	107.0	**1980.20**†	**4.43**	**1980.20**∗	**4.43**	**1980.20**	**4.43**	**1980.20**	**4.43**
0	10	44.6	218.2	3925.00†	**8.85**	**3953.40**∗	8.66	3925.00	**8.85**	**3953.40**	8.66
0	15	67.2	326.2	6260.40†	13.15	**6288.80**∗	12.96	6260.40	**13.16**	**6288.80**	12.96
0	20	90.0	439.6	8294.00†	**17.27**	**8322.40**	17.06	8294.00	17.25	**8322.40**	17.03
0	25	112.0	549.8	10313.20	21.09	**10341.60**	20.94	10313.20	**21.16**	10276.60	20.78
5	0	31.4	198.6	39874.00	**4.64**	39911.20	4.50	**42394.00**	4.31	42034.00	4.31
10	0	63.8	405.0	**44646.60**	9.20	42953.60	8.20	44286.60	**9.32**	44286.60	9.27
15	0	96.4	606.2	42109.20	13.29	42730.20	10.76	44291.60	13.51	**44420.00**	**13.90**
20	0	129.6	814.6	27927.20	9.80	40992.60	9.32	43131.20	**14.14**	**43409.00**	13.86
25	0	161.4	1018.2	28864.80	9.80	40489.20	9.30	39645.40	**13.87**	**43117.40**	13.23
5	5	53.8	311.0	39515.60	**8.97**	40998.60	8.43	42395.60	8.30	**44388.00**	7.23
10	10	109.6	627.0	43594.40	15.93	42664.40	15.52	44674.40	**15.98**	**47071.60**	13.98
15	15	165.2	944.0	34171.40	**23.29**	39015.00	20.00	46368.80	21.74	**47244.60**	19.10
20	20	219.4	1258.2	31823.00	**26.31**	41759.80	24.18	45223.60	25.16	**47728.40**	19.45
25	25	274.6	1572.8	29788.40	**28.87**	41641.00	26.05	46824.60	27.16	**47474.80**	21.24

upgrade — util **performs better in terms of utility** in most of the settings

Utility

configurations				cp — fair		cp — util		upgrade — fair		upgrade — util	
$\|\mathcal{R}_G\|$	$\|\mathcal{R}_T\|$	$\|\mathcal{M}\|$	$\|\mathcal{V}\|$	u	u^{slot}	u	u^{slot}	u	u^{slot}	u	u^{slot}
0	5	22.0	107.0	**1980.20†**	**4.43**	**1980.20∗**	**4.43**	**1980.20**	**4.43**	**1980.20**	**4.43**
0	10	44.6	218.2	3925.00†	**8.85**	**3953.40∗**	8.66	3925.00	**8.85**	**3953.40**	8.66
0	15	67.2	326.2	6260.40†	13.15	**6288.80∗**	12.96	6260.40	**13.16**	**6288.80**	12.96
0	20	90.0	439.6	8294.00†	**17.27**	**8322.40**	17.06	8294.00	17.25	**8322.40**	17.03
0	25	112.0	549.8	10313.20	21.09	**10341.60**	20.94	10313.20	**21.16**	10276.60	20.78
5	0	31.4	198.6	39874.00	**4.64**	39911.20	4.50	**42394.00**	4.31	42034.00	4.31
10	0	63.8	405.0	**44646.60**	9.20	42953.60	8.20	44286.60	**9.32**	44286.60	9.27
15	0	96.4	606.2	42109.20	13.29	42730.20	10.76	44291.60	13.51	**44420.00**	**13.90**
20	0	129.6	814.6	27927.20	9.80	40992.60	9.32	43131.20	**14.14**	**43409.00**	13.86
25	0	161.4	1018.2	28864.80	9.80	40489.20	9.30	39645.40	**13.87**	**43117.40**	13.23
5	5	53.8	311.0	39515.60	**8.97**	40998.60	8.43	42395.60	8.30	**44388.00**	7.23
10	10	109.6	627.0	43594.40	15.93	42664.40	15.52	44674.40	**15.98**	**47071.60**	13.98
15	15	165.2	944.0	34171.40	**23.29**	39015.00	20.00	46368.80	21.74	**47244.60**	19.10
20	20	219.4	1258.2	31823.00	**26.31**	41759.80	24.18	45223.60	25.16	**47728.40**	19.45
25	25	274.6	1572.8	29788.40	**28.87**	41641.00	26.05	46824.60	27.16	**47474.80**	21.24

upgrade — fair **outputs quite good utilitarian allocations**

Computation Time (ms)

configurations				cp — fair		cp — util		upgrade — fair		upgrade — util	
$\lvert\mathcal{R}_G\rvert$	$\lvert\mathcal{R}_T\rvert$	$\lvert\mathcal{M}\rvert$	$\lvert\mathcal{V}\rvert$	mode	slot	mode	slot	mode	slot	mode	slot
0	5	22.0	107.0	6.34	2.09	12.96	**1.63**	**4.24**	1.96	4.89	2.32
0	10	44.6	218.2	77.79	3.93	45.96	3.94	**9.73**	4.00	10.45	**3.79**
0	15	67.2	326.2	164.48	242.04	439.87	242.63	**9.41**	243.74	9.63	**241.06**
0	20	90.0	439.6	195.42	300.38	6000.21	300.09	**9.50**	300.13	10.19	300.36
0	25	112.0	549.8	759.76	300.16	7500.19	300.09	**237.50**	300.25	294.00	300.25
5	0	31.4	198.6	1500.57	300.04	1500.14	300.06	390.88	300.03	**339.86**	300.04
10	0	63.8	405.0	3002.12	300.08	3700.13	300.13	**1249.88**	300.06	1279.68	300.06
15	0	96.4	606.2	4502.34	300.06	4500.43	300.12	1849.59	300.11	**1834.56**	300.05
20	0	129.6	814.6	6003.48	300.07	6000.23	300.07	**2496.88**	300.08	2526.24	300.08
25	0	161.4	1018.2	7504.20	300.07	7500.22	300.07	3137.79	300.10	**3074.31**	300.09
5	5	53.8	311.0	1504.79	300.05	3000.13	300.04	**420.16**	300.09	433.25	300.06
10	10	109.6	627.0	3029.96	300.08	6000.21	300.09	**1206.61**	300.08	2026.38	300.06
15	15	165.2	944.0	8926.65	300.09	9000.24	300.14	**2410.81**	300.10	3530.65	300.08
20	20	219.4	1258.2	12011.00	300.15	12000.43	300.12	**3438.31**	300.16	4494.19	300.10
25	25	274.6	1572.8	15014.64	300.19	15000.44	300.14	**5187.83**	300.17	6019.73	300.11

Computation Time (ms)

configurations				cp − fair		cp − util		upgrade − fair		upgrade − util	
$\lvert\mathcal{R}_G\rvert$	$\lvert\mathcal{R}_T\rvert$	$\lvert\mathcal{M}\rvert$	$\lvert\mathcal{V}\rvert$	mode	slot	mode	slot	mode	slot	mode	slot
0	5	22.0	107.0	6.34	2.09	12.96	**1.63**	**4.24**	1.96	4.89	2.32
0	10	44.6	218.2	77.79	3.93	45.96	3.94	**9.73**	4.00	10.45	**3.79**
0	15	67.2	326.2	164.48	242.04	439.87	242.63	**9.41**	243.74	9.63	**241.06**
0	20	90.0	439.6	195.42	300.38	6000.21	300.09	**9.50**	300.13	10.19	300.36
0	25	112.0	549.8	759.76	300.16	7500.19	300.09	**237.50**	300.25	294.00	300.25
5	0	31.4	198.6	1500.57	300.04	1500.14	300.06	390.88	300.03	**339.86**	300.04
10	0	63.8	405.0	3002.12	300.08	3700.13	300.13	**1249.88**	300.06	1279.68	300.06
15	0	96.4	606.2	4502.34	300.06	4500.43	300.12	1849.59	300.11	**1834.56**	300.05
20	0	129.6	814.6	6003.48	300.07	6000.23	300.07	**2496.88**	300.08	2526.24	300.08
25	0	161.4	1018.2	7504.20	300.07	7500.22	300.07	3137.79	300.10	**3074.31**	300.09
5	5	53.8	311.0	1504.79	300.05	3000.13	300.04	**420.16**	300.09	433.25	300.06
10	10	109.6	627.0	3029.96	300.08	6000.21	300.09	**1206.61**	300.08	2026.38	300.06
15	15	165.2	944.0	8926.65	300.09	9000.24	300.14	**2410.81**	300.10	3530.65	300.08
20	20	219.4	1258.2	12011.00	300.15	12000.43	300.12	**3438.31**	300.16	4494.19	300.10
25	25	274.6	1572.8	15014.64	300.19	15000.44	300.14	**5187.83**	300.17	6019.73	300.11

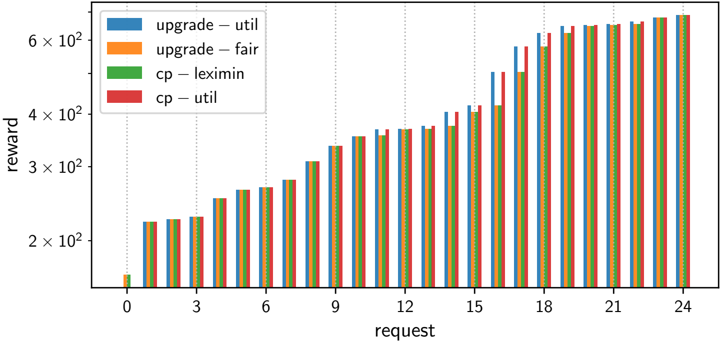
Upgrade-based heuristic methods clearly outperform optimal ones

Computation Time (ms)

configurations				cp − fair		cp − util		upgrade − fair		upgrade − util	
$\lvert\mathcal{R}_G\rvert$	$\lvert\mathcal{R}_T\rvert$	$\lvert\mathcal{M}\rvert$	$\lvert\mathcal{V}\rvert$	mode	slot	mode	slot	mode	slot	mode	slot
0	5	22.0	107.0	6.34	2.09	12.96	**1.63**	**4.24**	1.96	4.89	2.32
0	10	44.6	218.2	77.79	3.93	45.96	3.94	**9.73**	4.00	10.45	**3.79**
0	15	67.2	326.2	164.48	242.04	439.87	242.63	**9.41**	243.74	9.63	**241.06**
0	20	90.0	439.6	195.42	300.38	6000.21	300.09	**9.50**	300.13	10.19	300.36
0	25	112.0	549.8	759.76	300.16	7500.19	300.09	**237.50**	300.25	294.00	300.25
5	0	31.4	198.6	1500.57	300.04	1500.14	300.06	390.88	300.03	**339.86**	300.04
10	0	63.8	405.0	3002.12	300.08	3700.13	300.13	**1249.88**	300.06	1279.68	300.06
15	0	96.4	606.2	4502.34	300.06	4500.43	300.12	1849.59	300.11	**1834.56**	300.05
20	0	129.6	814.6	6003.48	300.07	6000.23	300.07	**2496.88**	300.08	2526.24	300.08
25	0	161.4	1018.2	7504.20	300.07	7500.22	300.07	3137.79	300.10	**3074.31**	300.09
5	5	53.8	311.0	1504.79	300.05	3000.13	300.04	**420.16**	300.09	433.25	300.06
10	10	109.6	627.0	3029.96	300.08	6000.21	300.09	**1206.61**	300.08	2026.38	300.06
15	15	165.2	944.0	8926.65	300.09	9000.24	300.14	**2410.81**	300.10	3530.65	300.08
20	20	219.4	1258.2	12011.00	300.15	12000.43	300.12	**3438.31**	300.16	4494.19	300.10
25	25	274.6	1572.8	15014.64	300.19	15000.44	300.14	**5187.83**	300.17	6019.73	300.11

All methods quickly achieve the slot optimization timeout

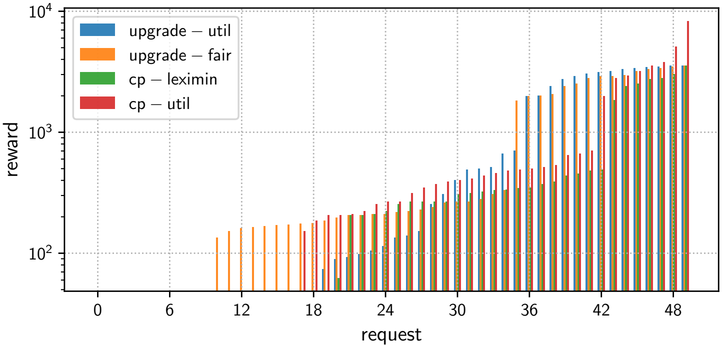
Fairness
Utility profiles an instance with 25 time-tagged allocation requests



- All the methods behave quite **similarly**
- Such problems are **not too constrained**, given the constellation configuration
- The **fair approaches only serve one more request**

Fairness (cont.)

Utility profiles an instance with 25 time-tagged allocation requests and 25 global allocation requests



- The **most rewarding** requests (25 to 49) are the **global allocation** ones
- Some requests **cannot be fulfilled** (even by leximin)
- upgrade — fair **serves more requests** (more than cp — fair, due to time budget)
- upgrade — util **is better than** cp — util on most of the high reward requests
- upgrade — fair and upgrade — util behave very well on the fairness side

Today’s Menu

Conclusions

To Sum Up

- We modeled a **novel problem (OSAP)** for allocating orbit slots
- We considered both **utilitarian and fairness** objectives
- We considered **two types of requests**: time-tagged requests and global requests
- We proposed an iterative **two-level optimization framework**
- We evaluated **four solution methods**
  - **Global allocation requests are the hardest** ones to fulfill
  - $cp - util$ **and** $cp - fair$ **do not scale** on larger instances
  - **Iterative upgrading methods result in good quality solutions** and are **3 times faster** on larger instances

Future Research

- Investigating other types of requests (e.g. areas of interest)
- Investigating other types of mode selection (e.g. searching in the mode space)
- Exploring other iterative schemes (e.g. degrading instead of upgrading)

Acknowledgements

This work has been performed with the support of the French government in the context of the “Programme d’Invertissements d’Avenir”, namely by the BPI PSPC project “LiChIE”

bpifrance

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

ONERA
THE FRENCH AEROSPACE LAB

Thank you for your attention!

www.onera.fr