On the Mathematical Models and Methods for the Hydro Unit Commitment Challenges

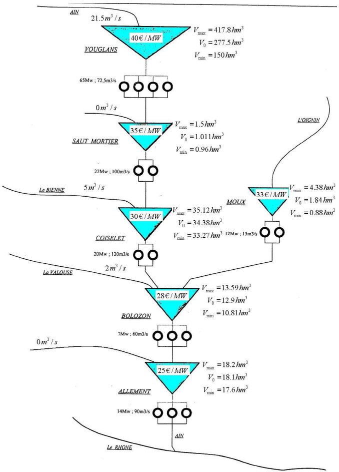
Claudia D'Ambrosio

CNRS & École Polytechnique, France

CWM^{3}EO, Budapest, September 25th, 2014

The Hydro-Plant Unit Commitment & Scheduling Problem

- Find the optimal scheduling of several plants with multi-unit pump-storage hydro power station.
- Short term.
- Assumptions: forecast electricity prices and inflows, price-taker.
- 16,2 % of the total energy produced worldwide.



Variables

Independent variables:

- q_{jt} = water flow in turbine j in period t ($j \in J, t \in T$), with $q_{j0} = Q_{j0}$;
- s_t = spillage in period t ($t \in T$);

Dependent variables:

- v_t = water volume in the basin in period t ($t \in T$), with $v_0 = V_0$;
- p_{jt} = power generated or consumed by turbine j in period t ($j \in J, t \in T$);

where $T = \{1, \dots, \bar{t}\}$:= the set of time periods considered,
$J = \{1, \dots, \overline{n}\}$:= the set of turbine-pump units.

Plus auxiliary variables for modeling discontinuities, etc.

Variables domain

For each period t, we have the three possible cases that can occur relative to turbine-pump unit j:

- if unit j is generating power $\rightarrow q_{jt} \geq 0$ and $p_{jt} \geq 0$;
- if unit j is pumping water $\rightarrow q_{jt} \leq 0$ and $p_{jt} \leq 0$;
- if unit j is not operating $\rightarrow q_{jt} = 0$ and $p_{jt} = 0$.

- $q_{jt} \in [\underline{q}_{jt}^{-}; \overline{q}_{jt}^{-}] \cup \{0\} \cup [\underline{q}_{jt}^{+}; \overline{q}_{jt}^{+}]$ $(j \in J, t \in T)$.

Also potential forbidden zones.

- $s_t \geq 0$ $(t \in T)$;
- $\underline{v} \leq v_t \leq \overline{v}$ $(t \in T)$;

Physical Constraints

Typically hard constraints:

- Water flow balance equations
- Respect allowed operational points: (dis-)continuous, discrete, turbine/pump related
- Forbid of simultaneous pump and turbine mode
- Power production depending on water flow and head effect
- Minimum number of periods to be spent in a status by the unit (minimum starting up/down times)
- spillage
- ...

Strategic Constraints

Typically soft constraints:

- Ramp up/down bound constraint
- Irrigation requirement/Ecological flows/Water rights
- Load balance equations constraints
- Minimum release of water per period
- Final reservoir level

Objective Function(s)

- Minimize the water consumption
- Maximize the profit
- Minimize the number of startups and shutdowns of generating units in the day
- Minimize the cost of power generation loss and generating unit start-up/shut-downs
- Maximize the daily plants global efficiency
- Minimize hydro-logic alteration/damage by inundation/the risk due to overtopping/sum of the reservoir releases through the bottom outlet

The challenges

- Large-scale problem
- Need to solve in short amount of time
- Combinatorial aspects
- Non linearities
- Multiple (conflicting) objectives
- etc!

Dealing with nonlinearities

Dealing with nonlinearities: Approximations

The power production: a highly non-linear function of the water flow and either the water level or (equivalently) the water volume in the reservoir, of the form:

$$p_{jt} = \varphi(q_{jt}, v_t) \; \forall j \in J, t \in T. \tag{1}$$

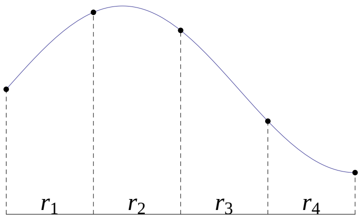
- MILP solvers more efficient than MINLP ones and handle large-scale instances.
- Trying to get rid of the non-linear functions $\rightarrow$ “linearize” and use MILP solvers!!!!
- **Piecewise linear approximation**: Beale & Tomlin, 1970 (*Special Ordered Sets*).

Focus on MINLP with non-linear objective function and linear constraints .

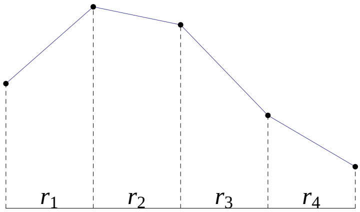
Starting simple: univariate function

Consider a function $f(x)$ and construct its piecewise linear approximation.

- Divide the domain of f in $n-1$ intervals of coordinates $x_1, \ldots, x_n$.
- Sample f at each point x_i with $i = 1, \ldots, n$.
- The piecewise linear approximation of f is given by the convex combination of the samples.



(a)



(b)

Function of 2 variables: Method 1

1. Simply fix the value of one of the 2 variables and obtain a univariate function: $f(x, \tilde{y})$.
2. Apply methods for approximating univariate functions (previous slide).

The quality of the approximation depends on the function at hand.

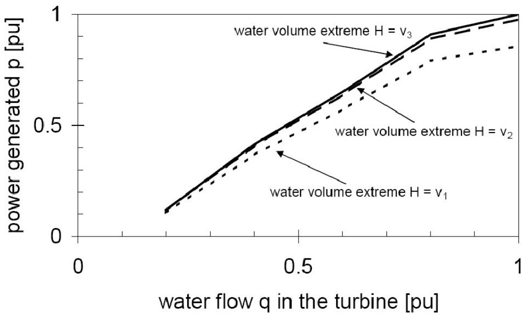
Choose to fix the “less nonlinear” variable.

Function of 2 variables: Method 2

In Conejo et al. (2002) the function $f^a = f(x, y)$ was approximated by considering three prefixed water volumes, say $\widetilde{y}^1$, $\widetilde{y}^2$, $\widetilde{y}^3$ and interpolating, for each $\widetilde{y}^r$, the resulting function

$$f^a = f(x, \widetilde{y}^r)$$

by piecewise linear approximation.

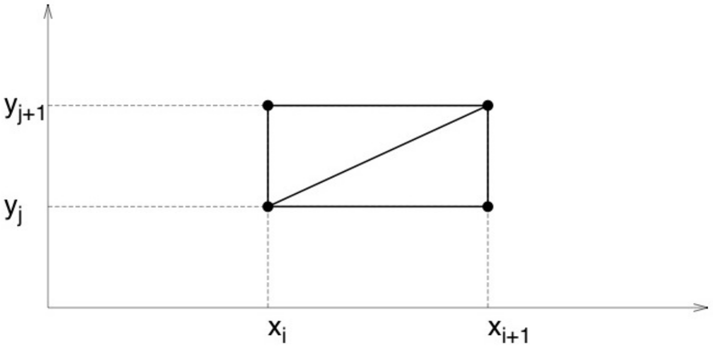


It can be **generalized** by approximating a prefixed number m of values of y.

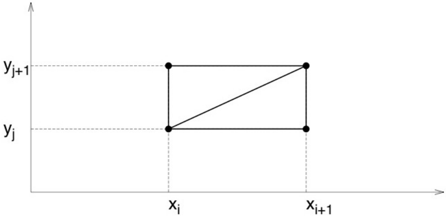
Function of 2 variables: Method 3

Consider a function $f(x, y)$ and construct its piecewise linear approximation.

- Divide the domain of f in a $(n-1) \times (m-1)$ grid of coordinates $x_1, \dots, x_n$, $y_1, \dots, y_m$.
- Divide the rectangles in the (x, y)-space in triangles .
- Sample f at each point (x_i, y_j) with $i = 1, \dots, n$ and $j = 1, \dots, m$.



Function of 2 variables: Method 3 (cont.d)



Any point $(\tilde{x}, \tilde{y})$

- belongs to one of the triangles;
- can be written as a convex combination of its vertices with weights α_{ij}; and
- the value of function f at $(\tilde{x}, \tilde{y})$ is approximated as

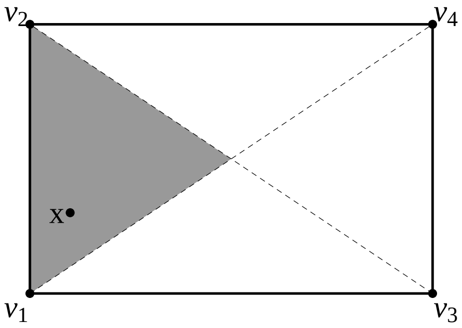
$$f^a = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{ij} f(x_i, y_j).$$

1 triangle $\leftrightarrow$ 1 binary variable $\rightarrow$ $O(n \times m)$ binaries.

Method 3: Standard Triangulation

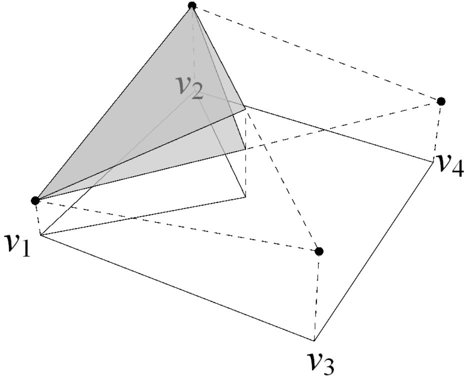
Given a rectangle identified by the four points v_1, v_2, v_3, v_4 we can divide it in 2 triangles in 2 different ways by selecting:

1. diagonal $[v_1, v_4]$; or
2. diagonal $[v_2, v_3]$.



Non-linear $f(x, y) \rightarrow 2$ different f^a for choice 1 and 2 !

Method 3: Standard Triangulation



Diagonal $[v_1, v_4]$:

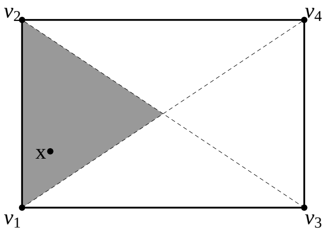
$$\alpha_{v_1} \leq \beta_{[v_1,v_2,v_4]} + \beta_{[v_1,v_3,v_4]}$$
$$\alpha_{v_2} \leq \beta_{[v_1,v_2,v_4]}$$
$$\alpha_{v_3} \leq \beta_{[v_1,v_3,v_4]}$$
$$\alpha_{v_4} \leq \beta_{[v_1,v_2,v_4]} + \beta_{[v_1,v_3,v_4]}$$

Diagonal $[v_2, v_3]$:

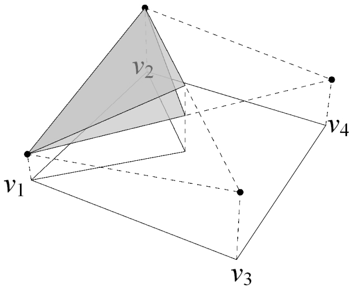
$$\alpha_{v_1} \leq \beta_{[v_1,v_2,v_3]}$$
$$\alpha_{v_2} \leq \beta_{[v_1,v_2,v_3]} + \beta_{[v_2,v_3,v_4]}$$
$$\alpha_{v_3} \leq \beta_{[v_1,v_2,v_3]} + \beta_{[v_2,v_3,v_4]}$$
$$\alpha_{v_4} \leq \beta_{[v_2,v_3,v_4]}$$

Method 4: Optimistic Approximation

with A. Lodi, S. Martello, R. Rovatti



(c)



(d)

Observation is simple:

Why do we need to decide the triangle “offline”?

Let the point $(\tilde{x}, \tilde{y})$ be a convex combination of all the 4 vertices of the rectangle and the MILP solver (optimistically) decide based on the

Method 4: Optimistic Approximation (cont.d)

Let the MILP (optimistically) decide based on the objective function!

In each region:

$$\check{f}(x) = \min \sum_{j=1}^{\nu} \alpha_j f(v_j) \qquad \text{or} \qquad \hat{f}(x) = \max \sum_{j=1}^{\nu} \alpha_j f(v_j)$$

subject to

$$\begin{aligned} \alpha_j &\geq 0 \\ \sum_{j=1}^{\nu} \alpha_j &= 1 \\ \sum_{j=1}^{\nu} \alpha_j x(v_j) &= x \\ \sum_{j=1}^{\nu} \alpha_j y(v_j) &= y \end{aligned}$$

where ν is the number of vertices that characterize the region.

Method 4: Optimistic Approximation Properties

Theorem

The approximations $\check{f}$ and $\hat{f}$ are such that

- $\check{f}$ *(resp. $\hat{f}$) is piecewise convex (resp. concave).*
- $\check{f}$ *and $\hat{f}$ are continuous.*
- *if f is linear then $\check{f} = \hat{f} = f$.*

Method 4: Optimistic Approximation Properties

Theorem

The approximations $\check{f}$ and $\hat{f}$ are such that

- $\Delta_r\left(f,\check{f}\right) \leq D_{\max}(r)$ *and* $\Delta_r\left(f,\hat{f}\right) \leq D_{\max}(r)$ $(\forall r \in \mathcal{R})$.
- *if f is convex (resp. concave) in any $r \in \mathcal{R}$, then $\check{f}$ (resp. $\hat{f}$) is the best possible linear interpolation of the samples $f(v_j)$ in the sense of $\Delta_r(f,\cdot)$.*

where
$\mathcal{R}$ is the collection of rectangles,
$\Delta_r(f,g) = \max_{(x,y)\in r} |f(x,y) - g(x,y)|$, and
$D_{\max}(r)$ is the maximum $\Delta_r\left(f,\tilde{f}\right)$ among all the possible linear interpolations $\tilde{f}$.

Standard vs Optimistic Approach: MILP size

Besides the nice properties, the optimistic approximation provides huge advantages when modeled with a MILP.

- Standard triangulation: 1 binary variable for each triangle $O(n \times m)$.
- Optimistic approximation: 1 binary variable for each rectangle.
- Note: Each axis treated separately, i.e.,
  n binaries for the x axis, and
  m binaries for the y axis. $\rightarrow O(n + m)$.
- For example, 3×3 grid $\rightarrow$ 6 vs 18 binaries
  10×10 grid $\rightarrow$ 20 vs 200 binaries!

$f^a = f(x, y)$: MILP size

	HR$^{\mathrm{Op}}$-std				UJ-std			
m	constr.	0-1 var.	var.	non-zero	constr.	0-1 var.	var.	non-zero
10	5,544	3,528	20,999	130,867	18,816	34,104	51,575	229,483
20	8,904	6,888	74,759	493,747	69,216	134,904	202,775	925,003
30	12,264	10,248	162,119	1,091,827	153,216	302,904	454,775	2,090,923
40	15,624	13,608	283,079	1,925,107	270,816	538,104	807,575	3,727,243
50	18,984	16,968	437,639	2,993,587	422,016	840,504	1,261,175	5,833,963

For $m = 50$:

- Number of binary variables: 16,968 vs 840,504.
- Number of constraints: 18,984 vs 422,016.
- Number of non-zeros: 2,993,587 vs 5,833,963.

$f^a = f(x,y)$: Solving the MILP

Single processor of an Intel Core2 CPU 6600, 2.40 GHz, 1.94 GB of RAM under Linux.

Cplex 10.0.1, time limit of 300', 600', 1 hour. Maximize the profit

		HR$^{\mathrm{Op}}$-std					UJ-std				
m	time limit	solution value	initial %gap	final %gap	CPU time	# nodes	solution value	initial %gap	final %gap	CPU time	# nodes
10	300	31,576.30	1.28	—	9.61	658	31,576.30	1.49	0.22	T.L.	7,684
	600	31,576.30	1.28	—	9.61	658	31,576.30	1.49	—	304.69	13,542
20	300	31,630.90	1.24	—	37.01	631	n/a	n/a	n/a	T.L.	1,121
	600	31,630.90	1.24	—	37.01	631	31,555.10	1.40	0.60	T.L.	3,699
	3,600	31,630.90	1.24	—	37.01	631	31,582.00	1.29	0.41	T.L.	35,382
30	300	31,633.40	1.23	0.02	T.L.	5,057	n/a	n/a	n/a	T.L.	411
	600	31,633.40	1.23	—	320.28	5,356	n/a	n/a	n/a	T.L.	1,285
	3,600	31,633.40	1.23	—	320.28	5,356	31,475.10	1.79	0.84	T.L.	4,310
40	600	31,639.20	1.20	—	265.00	929	n/a	n/a	n/a	T.L.	747
	3,600	31,639.20	1.20	—	265.00	929	n/a	n/a	n/a	T.L.	3,284
50	3,600	31,639.50	1.26	—	697.48	1,473	n/a	n/a	n/a	T.L.	1,697

- Number of solved instances: 5 vs 1.

$f^a = f(x, y)$: Going Logarithmic

Vielma & Nemhauser, 2011 : MILP model for the standard triangulations with a logarithmic number of variables (binary tree structure).Doable also for the Optimistic approximation.

	HR$^{\rm Op}$-std				UJ-log			
m	constr.	0-1 var.	var.	non-zero	constr.	0-1 var.	var.	non-zero
9	5,208	3,192	17,471	107,515	4,368	1,848	16,127	142,963
17	7,896	5,880	55,103	360,187	5,040	2,184	51,407	578,419
33	13,272	11,256	194,879	1,317,115	5,712	2,520	186,143	2,501,683
65	24,024	22,008	732,479	**5,037,307**	6,384	2,856	713,327	**11,056,243**

	HR$^{\rm Op}$-std				UJ-log			
m	solution value	% error	CPU time	# nodes	solution value	% error	CPU time	# nodes
9	31,565.40	-2.34	14.71	1,507	31,538.70	-2.26	18.69	1,723
17	31,577.20	-2.31	755.96	36,507	31,577.20	-2.31	20.84	369
33	31,626.20	-2.35	277.13	2,567	31,624.10	-2.35	231.99	1,531
65	31,640.30	-2.33	2,003.18	2,088	31,640.30	-2.34	530.56	435

$f^a = f(x, y)$: Going Logarithmic (cont.d)

	HR^{Op}-log				UJ-log			
m	constr.	0-1 var.	var.	non-zero	constr.	0-1 var.	var.	non-zero
9	3,864	1,512	15,791	134,395	4,368	1,848	16,127	142,963
17	4,536	1,848	51,071	552,043	5,040	2,184	51,407	578,419
33	5,208	2,184	185,807	2,407,771	5,712	2,520	186,143	2,501,683
65	5,880	2,520	712,991	10,698,571	6,384	2,856	713,327	11,056,243

	HR^{Op}-log					UJ-log				
m	solution value	initial %gap	final %gap	CPU time	# nodes	solution value	initial %gap	final %gap	CPU time	# nodes
9	31,565.40	1.13	—	14.21	1,439	31,538.70	1.14	—	18.69	1,723
17	31,577.20	1.35	—	23.88	653	31,577.20	1.35	—	20.84	369
33	31,626.20	1.24	—	99.90	540	31,624.10	1.25	—	231.99	1,531
65	31,640.30	1.20	—	593.73	599	31,640.30	1.20	—	530.56	435

Why? $\log(nm) = \log(n) + \log(m)$

Advantages of the optimistic approximation: MILP model of limited size (tractable) and easy to implement .

Multiple Objectives

Multiple Objectives: general math model

$$
\begin{aligned}
\min f_k(x) & \qquad \forall k \in \{1, \ldots, p\} \\
g_i(x) \leq 0 & \qquad \forall i \in \{1, \ldots, m\} \\
x_j \in \mathbb{Z} & \qquad \forall j \in \{1, \ldots, r\}
\end{aligned}
$$

Multiple Objectives: standard methods

Weighted Sum method:

$$\min \sum_{k=1}^{p} \lambda_k f_k(x)$$

$$g_i(x) \leq 0 \qquad \forall i \in \{1, \ldots, m\}$$

$$x_j \in \mathbb{Z} \qquad \forall j \in \{1, \ldots, r\}$$

with $0 \leq \lambda_k \leq 1 \ \forall k \in \{1, \ldots, p\}$ and $\sum_{k=1}^{p} \lambda_k = 1$.

Multiple Objectives: standard methods

ϵ-constraint method:

$$\begin{aligned} \min f_{\bar{k}}(x) & \\ g_i(x) \leq 0 & \qquad \forall i \in \{1,\ldots,m\} \\ f_k(x) \leq \tilde{f}_k & \qquad \forall ki \in \{1,\ldots,p\}, k \neq \bar{k} \\ x_j \in \mathbb{Z} & \qquad \forall j \in \{1,\ldots,r\} \end{aligned}$$

Multiple Objectives: new approach

With V. Cacchiani (thanks to STSM of COST Action TD1207)

Branch and Bound Algorithm

- branching rule
- dual bounds
- fathoming rules
- refinement procedure

Branch and Bound Algorithm

- Branching rule:
  - At each level j of the decision tree, we generate one child node for each possible fixing of variable x_j to value l, with $l \in \{\mathrm{ub}_j, \ldots, \mathrm{lb}_j\}$
- Dual bounds:
  - The lower bound at the root node is computed by solving p single objective MINLP problems via a general-purpose MINLP solver.
  - At each node of the decision tree, the lower bound is computed by solving p single objective NLP problems obtained by relaxing integrality requirements and by taking into account the branching decisions up to the current node.

Fathoming rules

A node can be fathomed if:

- The corresponding problem is infeasible
- It is an integer feasible leaf node
- Its lower bound is dominated by (at least) one of the solutions, say x^*, of the current Pareto set, i.e., $\text{LB}_k \geq f_k(x^*)\ \forall k \in \{1, \ldots, p\}$
- Each single objective NLP_k problem ($k \in \{1, \ldots, p\}$) is integer feasible and all the p integer solutions coincide

Refinement procedure

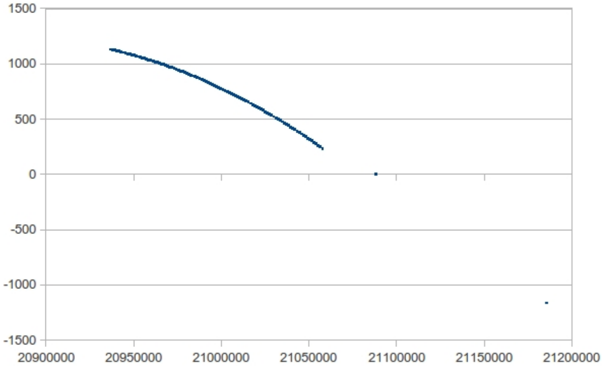
For each solution x^* in the current Pareto set Y^* and for each objective function $f_{\bar{k}}$ ($\bar{k} \in \{1, \ldots, p\}$), we perform the ϵ- constraint method with $\tilde{f}_k$ set to $f_k(x^*)$.

Starting Pareto set and solving leaf nodes

Since we consider convex problems, the solution of the leaf nodes can generate all Pareto points by varying the weights (Censor 1977).

Hydro UC: A discontinuous Pareto set

Consider just one period and fix each of the 3 configurations (turbine on, pump on, both off): the Pareto set is the union of the three disjoint sets.



Characteristics of the instances

T = number of time periods of one hour considered in the instance

T	# vars	# bin	# constr
1	18	8	19
2	30	14	34
3	42	20	49
4	54	26	64
5	66	32	79
6	78	38	94
7	90	44	109

Computational experiments: setting

- AMPL environment
- Intel Xeon 2.4 GHz with 8 GB Ram running Linux
- SCIP to solve single objective MINLPs
- Ipopt to solve single objective NLPs
- Weighted Sum method to obtain a starting Pareto set (step 0.1)
- Weighted Sum method to solve a leaf node (step 0.1)

Comparison of the three branch-and-bound versions:

- noRF: no refinement
- 1RF: refinement procedure only executed at the end of the resolution
- RF: refinement procedure executed at each update of the Pareto set

Comparison

	N solutions			CPU time		
T	noRF	1RF	RF	noRF	1RF	RF
1	4	4	4	1	1	1
2	11	11	11	3	3	3
3	35	35	30	12	12	15
4	61	61	49	43	43	57
5	108	108	79	150	150	229
6	179	179	120	534	534	891
7	257	257	134	1946	1946	3861

Table: N solutions and CPU time (in s) for the three versions of the branch-and-bound

Fathoming statistics

# T	# nodes	# dom	# leaf
1	12	1	1
2	55	1	5
3	233	4	19
4	862	11	65
5	3056	26	211
6	10415	54	665
7	34185	175	1995

Table: Statistics on the total number of nodes, dominated nodes, and leaf nodes

Pareto sets of the three branch-and-bound versions

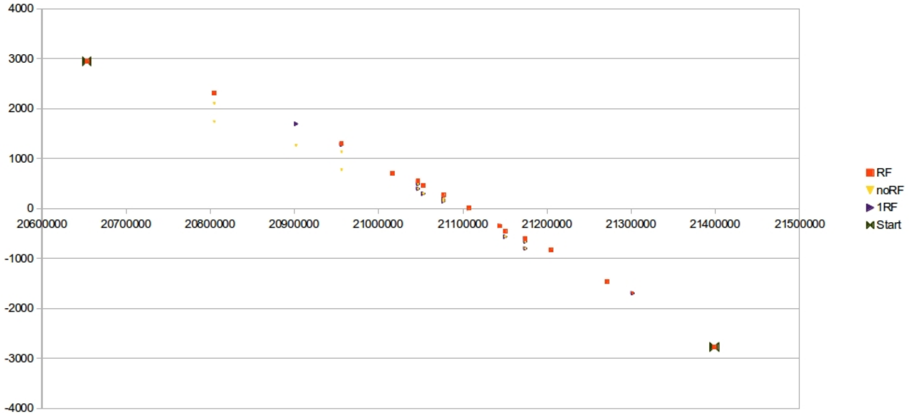


Figure: Pareto sets comparison for $T = 3$.

Comparison with the Weighted Sum method

The Weighted Sum method:

- was executed with a step of 0.001, i.e. executed for 1000 iterations
- ended up in obtaining 27 solutions
- solutions are characterized by a high revenue and a small final reservoir

The branch-and-bound algorithm derives a more diverse Pareto set. The RF solutions are characterized by solutions having revenue and volume in wider ranges.

Conclusions

- Challenging problem from different aspects
- Large-scale problem
- Fast methods needed/online optimization
- Combinatorial aspect
- Nonlinearities
- Multiple (conflicting) objectives

Conclusions

- Challenging problem from different aspects
- Large-scale problem
- Fast methods needed/online optimization
- Combinatorial aspect
- Nonlinearities
- Multiple (conflicting) objectives

Lots of research to be done!