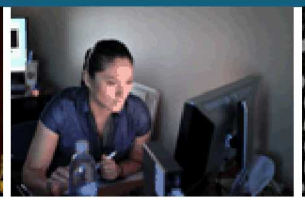
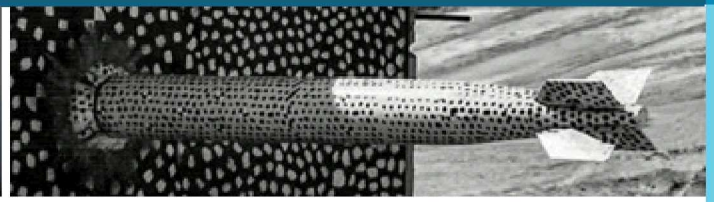
SAND2019-7037C

Mixed Integer Programming Formulations for the Unit Commitment Problem





PRESENTED BY

Bernard Knueven

Co-authors: James Ostrowski (UTK)

Jean-Paul Watson (Sandia)

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

The Unit Commitment Problem

The Unit Commitment Problem (UC) is a large-scale mixed-integer nonlinear program for finding a low-cost operating schedule for power generators.

These problems typically have quadratic objective functions and non-linear, non-convex transmission constraints.

- Typically both of these are linearized

Starting in 2005 with PJM, market operators in the United States have transitioned from using Lagrangian relaxation to solve UC to using mixed-integer programming (MIP) and a commercial solver such as CPLEX, Gurobi, or Xpress.

- MIPs usually have many equivalent formulations, and UC is no exception.

The day-ahead problem has an hourly time horizon which is solved for 36 to 48 hours ahead to prevent end-of-horizon effects, and has hundreds to thousands of generators and up to tens of thousands of buses.

In practice, it is desirable to have a UC solution in 10 to 15 minutes.

Contributions

We catalog existing formulations for the UC problem as originally described by Carrion and Arroyo (2006).

- Improvements to this formulation have been the subject of several subsequent papers, including Ostrowski et al. (2012), Morales-Espana et al. (2013), Damci-Kurt et al. (2016), Pan et al. (2016), K. et al. (2018), K. et al. (2018), Atakan et al. (2018).

We preform computational experiments on 41 different UC formulations, some novel, and some from the literature, on 68 UC instances.

- Largest instance has 900+ generators over 48 hours with hourly time horizon
- This took approximately two weeks of wall-clock time!

We make publically available on GitHub reference implementations for all the formulations examined in the Pyomo modeling language in the EGRET library. We will also make publically available the UC instances considered.

We contribute some additional results on both valid variable upper bound inequalities and piecewise linear formulations for production costs, driven in part by our prior computational experience.

We introduce two novel UC formulations, one of which is a new combination of existing components, and the other draws on on new components as well as existing formulations. The later formulation significantly improves on the performance of any previously reported UC formulation, establishing a new state-of-the-art.

The Unit Commitment Problem

$$\min \sum_{g \in \mathcal{G}} \sum_{t \in \mathcal{T}} c_g(t)$$

$$\text{s.t.} \sum_{g \in \mathcal{G}} A_g(p_g, \overline{p}_g, u_g) + N(s) = L$$

$$(p_g, \overline{p}_g, u_g, c_g) \in \Pi_g \qquad \forall g \in \mathcal{G}.$$

UC is that of minimizing system operating costs subject to the system constraints and the technical constraints of the generators.

Generator technical constraints

- Convex (piecewise linear) production costs
- Minimum and maximum output levels
- Ramping constraints
- Minimum up/down time
- Downtime dependent startup costs

Polyhedral Results for Generator Scheduling

1-binary variable model (1-bin)

- We can write the feasible region of a generator using two variables per time period.
- $p(t)$ is the continuous variable representing the power output at time t.
- $u(t)$ is the binary variable representing if the generator is on/off.
- There is a known convex hull description for this polyhedron with simple bounds on output and minimum up/down times, but it is vary large (exponential).
- But, a polynomial time cutting-plane method exists (Lee et al. 2004).

3-binary variable model (3-bin)

- Add to the 1-bin model two additional variables:
- $v(t)$ is the binary variable representing a *turn on* at time t,
- $w(t)$ is the binary variable representing a *turn off* at time t.
- The two additional variables are redundant. But, they allow us to write tight descriptions of the generator polytope with minimum up/down times (Rajan and Takriti 2005), start-up and shutdown power constraints (Gentile et al. 2017), and convex piecewise production costs (K. et al. 2018) (with additional variables for each piecewise segment) with a linear number of constraints and variables.

Polyhedral Results for Generator Scheduling

Shortest path formulation

- Add additional variables $y(t_1, t_2)$ to represent at start-up at time t_1 and shutdown at time t_2, on continuously in between, and additional variables $z(t_1, t_2)$ to represent at shutdown at time t_1 and start-up at time t_2, off continuously in between.
- These start-up/shutdown sequences can be linked using a full shortest-path formulation (Pochet and Wolsey, 2006), where the path is from "turn on" nodes to "turn off" and from "turn off" nodes to "turn on" nodes.
- This formulation has $O(T^2)$ edges (variables), but provides a convex hull description for downtime dependent start-up costs. There is a clear link to the u, v, w variables, so the prior results on start-up/shutdown power and piecewise production costs carry through (note the edges themselves can enforce minimum up/down time).
- If we are okay with integer optimal, we can use the "matching" formulation from K. et al. (2018), which keeps the 3-bin variables and only those z's which represent a hot/warm start-up. The benefit is far fewer variables (on order $(TC - DT) \cdot T$).

Polyhedral Results for Generator Scheduling

Extended formulation

- To the shortest path formulation, add additional variables $p(t, t_1, t_2)$ for the output of the generator at time t given a start-up and time t_1 and a shutdown at time t_2, on continuously in between.
- Requires on order T^3 variables and constraints, but gives a convex hull representation for ramping constraints, and every other technical constraint mentioned (Frangioni and Gentile (2015), K. et al. (2018), Guan et al. (2018).
- Very large but still polynomial. K. et al. (2018) uses it for cut-generation.

Choosing a Formulation

To summarize:

- 1-bin formulation (smallest): Not a convex hull for any interesting phenomenon
- 3-bin formulation (small): Convex hull for minimum up/down times, convex piecewise production, and start-up/shutdown power
- Shortest path (large): Convex hull for above plus downtime-dependent start-up costs, smaller if integer optimal is sufficient
- Extended formulation (very large): Convex hull description for everything above plus ramping constraints, and more!

Computational experience to date indicates

- 1-bin: too weak
- EF: too large
- 3-bin: just right

Since the convex hull representation for a single generator is too large, it is important to consider which classes of (perhaps imperfect) constraints we are going to include in a practical formulation.

In this case, engineering is more important than math.

Paper

On Mixed Integer Programming Formulations for the Unit Commitment Problem

Bernard Knueven
Discrete Math & Optimization, Sandia National Laboratories, Albuquerque, NM 87185, bknueve@sandia.gov

James Ostrowski
Industrial and Systems Engineering, University of Tennessee, Knoxville, TN 37996, jostrows@utk.edu

Jean-Paul Watson
Data Science & Cyber Analytics, Sandia National Laboratories, Livermore, CA 94551, jwatson@sandia.gov

We provide a comprehensive overview of mixed integer programming formulations for the unit commitment problem (UC). UC formulations have been an especially active area of research over the past twelve years, due to their practical importance in power grid operations, and this paper serves as a capstone for this line of work. We additionally provide publicly available reference implementations of all formulations examined. We computationally test existing and novel UC formulations on a suite of instances drawn from both academic and real-world data sources. Driven by our computational experience from this and previous work, we contribute some additional formulations for both production upper bound and piecewise linear production costs. By composing new UC formulations using existing components found in the literature and new components introduced in this paper, we demonstrate that performance can be significantly improved – and in the process, we identify a new state-of-the-art UC formulation.

Key words: Unit commitment, mixed integer programming, mathematical programming formulations

Available on *Optimization Online*: http://www.optimization-online.org/DB_HTML/2018/11/6930.html

EGRET: Electrical Grid Research and Engineering Tools

EGRET is a Python-based package for electrical grid optimization based on the Pyomo optimization modeling language. EGRET is designed to be friendly for performing high-level analysis (e.g., as an engine for solving different optimization formulations), while also providing flexibility for researchers to rapidly explore new optimization formulations.

Major features:

- Expression and solution of unit commitment problems, including full ancillary service stack
- Expression and solution of economic dispatch (optimal power flow) problems (e.g, DCOPF, ACOPF)
- Library of different problem formulations and approximations
- Generic handling of data across model formulations
- Declarative model representation to support formulation development

EGRET is available under the BSD License at https://github.com/grid-parity-exchange/Egret

A modular framework for UC formulations in EGRET

```
from egret.model_library.unit_commitment.uc_model_generator \
                import UCFormulation, generate_model

## get the formulation from Carrion and Arroyo (2006)
formulation = UCFormulation(status_vars = 'CA_1bin_vars',
                            power_vars = 'basic_power_vars',
                            reserve_vars = 'CA_power_avail_vars',
                            generation_limits = 'CA_generation_limits',
                            ramping_limits = 'CA_ramping_limits',
                            production_cost = 'CA_production_costs',
                            uptime_downtime = 'CA_UT_DT',
                            startup_costs = 'CA_startup_costs',
                            )

## construct the model based on the data md
model = generate_model(md, formulation)
```

Over 100,000 formulations

This instantiates a Pyomo `ConcreteModel` (`model`) based on the data provided in the object `md`, which can be used as part of a script.

The eights components of `UCFormulation` can be changed as easily as modifying a string in this file. Runtime checks to ensure incompatible components are not combined.

Number of implemented formulations per component:

- `status_vars`: 5
- `power_vars`: 3
- `reserve_vars`: 4
- `generation_limits`: 9
- `ramping_limits`: 8
- `production_costs`: 12
- `uptime_downtime`: 5
- `startup_costs`: 9

Some Theoretical Results

We give a convex hull formulation for a generator with convex piecewise production costs, start-up and shutdown ramp rates, generation limits, and minimum up/down times, which is $O(T \cdot L)$, where T is the number of time periods and L is the number of piecewise segments.

- Proof technique: an extension of the proof in Gentile et al. (2017) for a generator with generation limit, start-up and shutdown ramp rates, and minimum up/down times.

We also show that adding the start-up cost formulation from K. et al. (2018) to this results in a tight formulation when start-up costs are increasing.

- Proof: this is a corollary of the above result and a result on the tightness of the start-up cost formulation from K. et al. (2018).
- $O((TC - DT) \cdot T \cdot L)$, where TC is the number of time periods after which the generator goes cold and DT is the minimum downtime of the generator.

Ramping still makes things difficult! Recent papers (Damci-Kurt et al. 2016, Pan and Guan 2017, 2018) suggest there is no linear convex hull formulation when ramping constraints are added.

Still, we attempt to tighten the ramping constraints by introducing new variable upper bound inequalities which do have a linear description.

- Experimental results from Damci-Kurt et al. (2016) and K. et al (2018) suggest variable upper bounds are often important in practice.

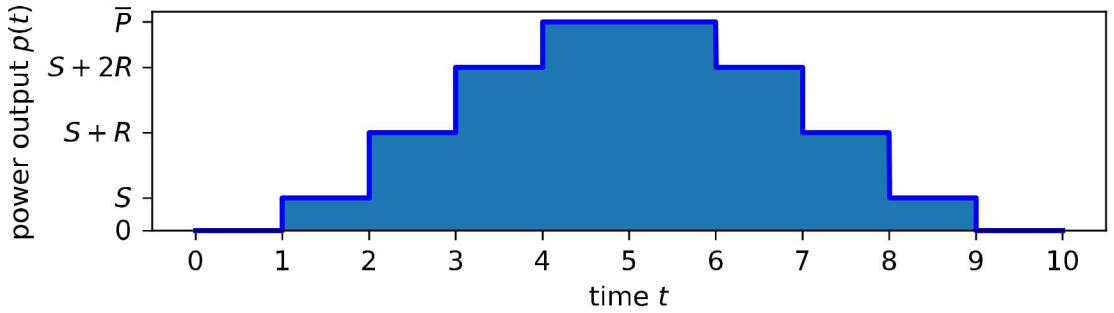
Variable Upper Bound Inequalities

Let $T^{RU} = \left\lfloor \frac{\overline{P}-SU}{RU} \right\rfloor$ and $T^{RD} = \left\lfloor \frac{\overline{P}-SU}{RD} \right\rfloor$, so T^{RU} is the number of time periods the generator needs to ramp up from off to maximum power and T^{RD} is the number of periods the generator needs to ramp down from maximum power to off.

If $UT \geq T^{RU} + T^{RD} + 2$, then the following is a generalized upper bound inequality:

$$p'(t) \leq (\overline{P} - \underline{P})u(t) - \sum_{i=0}^{T^{RU}} (\overline{P} - (SU + iRU))v(t-i) - \sum_{i=0}^{T^{RD}} (\overline{P} - (SD + iRD))w(t+1+i)$$

This limits the ramping trajectory when the generator is starting-up or shutting-down. Notice the condition $UT \geq T^{RU} + T^{RD} + 2$ ensures that one and only one of the start-up and shutdown indicators v and w are 1 and that if a v or w are 1 then $u(t)$ is 1.



A few similar inequalities can be derived in the case of reserve-up, and modifications can accommodate a weaker assumption on UT.

Tightening Piecewise Production Costs

Easy observation: in the 3-bin formulation, piecewise production costs can be tightened using the start-up and shutdown ramps just like the production variable.

$$p^l(t) \leq (\overline{P}^l - \overline{P}^{l-1})u(t) - C^v(l)v(t) - C^w(l)w(t+1) \qquad l \in \mathcal{L},$$

where

$$C^v(l) := \begin{cases} 0 & \overline{P}^l \leq SU \\ \overline{P}^l - SU & \overline{P}^{l-1} < SU < \overline{P}^l \\ \overline{P}^l - \overline{P}^{l-1} & \overline{P}^{l-1} \geq SU \end{cases}, \qquad C^w(l) := \begin{cases} 0 & \overline{P}^l \leq SD \\ \overline{P}^l - SD & \overline{P}^{l-1} < SD < \overline{P}^l \\ \overline{P}^l - \overline{P}^{l-1} & \overline{P}^{l-1} \geq SD \end{cases},$$

If $UT > 1$, then this along with the minimum up/down time formulation from Rajan and Takriti (2005) is a perfect formulation for a generator with piecewise production costs, minimum up/down times, and start-up and shutdown ramping rates. Just like the result from Gentile et al. (2017), this can be appropriately modified for $UT = 1$.

If there are irredundant ramping constraints, then we can tighten the bounds on each bin as we did the production variable.

Test Instances

One academic test set and and two test sets based on real-world data. All are an hourly 48-hour day-ahead UC.

- RTS-GMLC: 73 thermal generators, 81 renewable generators, 73 buses, 120 transmission lines. Hourly day-ahead data for load and renewable generation for a year. We selected twelve representative days, considered both with and without transmission for a total of 24 test instances.
- CAISO: 410 schedulable thermal generators, 200 must-run thermal generators. We considered five demand/renewables scenarios under four reserve policies: 0%, 1%, 3%, and 5% of demand, for a total of 20 test instances.
- FERC/PJM: Generation set publically available from FERC, approximately 900 thermal units, with demand, reserve, and wind data publically available from PJM for 2015. We selected twelve representative days, and considered the wind data as-is (low-wind) and also scaled it to achieve 30% wind penetration for the year, for 24 test instances total.

The above makes for a total of 68 UC instances across three set of generators.

Some Formulations Considered

CA: Carrion and Arroyo (2006)

OAV-O: Ostrowski et al. (2012) "Original," similar to Arroyo and Conejo (2000)

OAV-UD: Ostrowski et al. (2012) "Up/Downtime"

OAV: Ostrowski et al. (2012) with 2-period consecutive ramping inequalities from this paper

OAV-T: Ostrowski et al. (2012) with all ramping and generalized upper bound inequalities from this paper

MLR: Morales-Espana et al. (2013)

ALS: Atakan et al. (2018)

KOW: K. et al. (2018)

T: "Tight" New formulation using some ideas from the literature and the generalized upper bounds and production costs introduced in this paper

Co: "Compact" New formulation using ideas from the literature that emphasize compactness

R1: "Random 1" New formulation based on sampling formulations and testing against the RTS-GMLC instances

R2: "Random 2" Another new formulation based on sampling and testing against the RTS-GMLC instances

Computational Platform

Dell PowerEdge T620 (circa 2013)

- Two Intel Xeon E5-2670 processors (16 cores/32 threads)
- 256GB RAM
- Ubuntu 16.04
- Gurobi 8.0.1

No other major jobs were running at the time of the computational experiments, and Gurobi settings were preserved at default except a time limit.

- Hence all UC MIPs we attempt to solve to 0.01% optimality gap.

The time limit was set at 300 seconds for the RTS-GMLC and CAISO instances, and a time limit of 600 seconds was imposed for the larger FERC instances.

Computational Results: RTS-GMLC Instances

Formulation	Time (s)	Opt gap (%)	Time outs	Times best	Times 2nd
CA	300.0	13.806%	24	0	0
OAV-O	172.8	0.3335%	13	0	0
OAV-UD	163.2	0.2541%	13	0	0
OAV	169.7	0.2352%	12	0	0
OAV-T	178.0	0.2498%	12	0	0
MLR	86.19	0.0165%	5	0	0
ALS	58.29	0.0122%	2	1	1
KOW	94.81	0.0165%	4	0	1
T	50.65	0.0121%	2	4	4
Co	43.11	0.0121%	**1**	1	4
R1	**32.37**	**0.0114%**	**1**	**14**	6
R2	36.94	0.0114%	**1**	4	**8**

The **R1** and **R2** formulations do well on this test set.

- Not too surprising since they were selected based on their performance on this test set.

Computational Results: CAISO Instances

Formulation	Time (s)	Opt gap (%)	Time outs	Times best	Times 2nd
CA	300.0	1.0987%	20	0	0
OAV-O	300.0	6.7647%	20	0	0
OAV-UD	300.0	6.2269%	20	0	0
OAV	300.0	3.6319%	20	0	0
OAV-T	300.0	6.1679%	20	0	0
MLR	117.1	0.0119%	2	0	1
ALS	260.4	0.0577%	12	0	0
KOW	79.02	0.0102%	2	4	**7**
T	**56.90**	**0.0100%**	**0**	**15**	3
Co	100.6	**0.0100%**	**0**	0	5
R1	147.4	0.0102%	1	0	1
R2	104.1	0.0100%	1	1	3

Here the **T**, **Co**, **R2**, and **KOW** formulations perform well.

Computational Results: FERC Instances

Formulation	Time (s)	Opt gap (%)	Time outs	Times best	Times 2nd
CA	600.0	43.333%	24	0	0
OAV-O	599.7	11.716%	23	0	0
OAV-UD	588.2	1.4575%	20	0	0
OAV	588.4	1.3104%	21	0	0
OAV-T	582.4	1.4389%	22	0	0
MLR	340.5	0.0555%	3	4	1
ALS	394.7	0.0933%	7	2	2
KOW	390.2	0.0117%	2	3	5
T	**268.6**	**0.0104%**	**1**	**8**	4
Co	309.5	0.0596%	3	4	5
R1	308.9	0.0480%	3	3	**6**
R2	373.1	0.0665%	4	0	1

T is clearly the superior performer on this set of instances.

Summary

Formulation **T** performs well across the test set.

- Weaker performance on the RTS-GMLC is mainly due to the network-constrained instances, which require quite a bit of enumeration. We used a B-θ representation for the network, for which Gurobi has a difficult time generating valid cuts.
- Typically operators use a PTDF formation for the network, which will have different behavior.
- Even in the worst case across the 68 instances, with formulation **T** Gurobi terminates at the time limit with only a 0.05% gap, which is the best worst-case performance.

Formulations **R1** and **R2** both performed well on RTS-GMLC, suggesting it may be possible to specialize the formulation used against typical instances.

- Since the generation fleet often does not change, it may be possible to "fit" a formulation for typical or difficult day-ahead instances.

Epilogue: Analysis of formulation T

We attempted to improve on formulation **T** by swapping one of its eight components for another from the literature.

- Also in part to see which components are important to a good formulation.

The swaps that were improving are below.

	RTS-GMLC		CAISO		FERC	
	Time (s)	Opt gap (%)	Time (s)	Opt gap (%)	Time (s)	Opt gap (%)
T	50.65	0.0121%	56.90	0.0100%	268.6	0.0103%
uptime/downtime						
RT 2bin: (4)(7)	48.29	0.0120%	51.30	0.0100%	235.1	0.0100%
generation limits						
MLR: (21)(22)	51.29	0.0122%	52.44	0.0100%	254.8	0.0100%
GMR: (21)(24)	51.37	0.0122%	52.42	0.0100%	254.1	0.0100%
ramping limits						
MLR: (33)(34)	49.28	0.0119%	48.53	0.0100%	242.6	0.0100%
piecewise production						
CA: (45)(46)(47)	48.93	0.0119%	57.87	0.0100%	230.2	0.0100%

Hence there is room for improvement on **T**.

- The most surprising result is the improvement from using the 2-bin formulation for downtime.

Further refinement could be done, though at the risk of over-fitting for these instances.

Conclusions

- Comprehensive literature and computational survey of UC formulations is available
- Along with open-source implementations of all the examined (and many more unexamined) UC formulations
- Literature and computational survey uncovered a new state-of-the-art

- Additional challenges remain
  - Tightening the interaction between the unit commitment and the transmission system (Van den Bergh et al. 2014, Wu 2016)
  - Virtual transactions, which may weaken our ability to tighten system constraints (Chen et al. 2016)
  - More realistic modeling of the transmission system, including the need for reactive power support (Castillo et al. 2016)
  - Better modeling of ancillary service products, which are relatively neglected by the literature and tend to vary by market