Mathematical Programming Computation (2020) 12:225–248
https://doi.org/10.1007/s12532-020-00176-5

FULL LENGTH PAPER

A novel matching formulation for startup costs in unit commitment

Bernard Knueven[1] · James Ostrowski[2] · Jean-Paul Watson[3]

Received: 19 July 2018 / Accepted: 21 November 2019 / Published online: 24 February 2020

Abstract

We present a novel formulation for startup cost computation in the unit commitment problem (UC). Both our proposed formulation and existing formulations in the literature are placed in a formal, theoretical dominance hierarchy based on their respective linear programming relaxations. Our proposed formulation is tested empirically against existing formulations on large-scale UC instances drawn from real-world data. While requiring more variables than the current state-of-the-art formulation, our proposed formulation requires fewer constraints, and is empirically demonstrated to be as tight as a perfect formulation for startup costs. This tightening can reduce the computational burden in comparison to existing formulations, especially for UC instances with large reserve margins and high penetration levels of renewables.

Keywords Mixed-integer linear programming (MILP) · Unit commitment (UC) · Optimization

B. Knueven and J.-P. Watson were supported by the U.S. Department of Energy (DOE), Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program under contract number KJ0401000 through the project "Multifaceted Mathematics for Complex Energy Systems", and the Grid Modernization Initiative of the U.S. DOE, under project 1.4.26, as part of the Grid Modernization Laboratory Consortium, a strategic partnership between DOE and the national laboratories to bring together leading experts, technologies, and resources to collaborate on the goal of modernizing the nation's grid. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and 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. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. J. Ostrowski was supported by NSF award number 1332662 and by DOE (ASCR) award DE-SC0018175.

Extended author information available on the last page of the article

List of symbols

Indices and sets

$g \in \mathcal{G}$	Thermal generators
$l \in \mathcal{L}_g$	Piecewise production cost intervals for generator g: $1, \ldots, L_g$
$s \in \mathcal{S}_g$	Startup categories for generator g, from hottest (1) to coldest (S_g)
$t \in \mathcal{T}$	Hourly time steps: $1, \ldots, T$

Parameters

c_g^l	Cost coefficient for piecewise segment l for generator g (\$/MWh)
c_g^s	Startup cost in category s for generator g (\$)
c_g^u	Cost of generator g running and operating at minimum production $\underline{P}_g$ (\$/h)
$D(t)$	Load (demand) at time t (MW)
DT_g	Minimum down time for generator g (h)
$\overline{P}_g$	Maximum power output for generator g (MW)
$\overline{P}_g^l$	Maximum power for piecewise segment l for generator g (MW)
$\underline{P}_g$	Minimum power output for generator g (MW)
$R(t)$	Spinning reserve at time t (MW)
RD_g	Ramp-down rate for generator g (MW/h)
RU_g	Ramp-up rate for generator g (MW/h)
SD_g	Shutdown rate for generator g (MW/h)
SU_g	Startup rate for generator g (MW/h)
TC_g	Time down after which generator g goes cold, i.e., enters state S_g
$\underline{T}_g^s$	Time offline after which the startup category s is available ($\underline{T}_g^1 = DT_g$, $\underline{T}_g^{S_g} = TC_g$)
$\overline{T}_g^s$	Time offline after which the startup category s is no longer available ($= \underline{T}_g^{s+1}$, $\overline{T}_g^{S_g} = +\infty$)
UT_g	Minimum up time for generator g (h)
$W(t)$	Aggregate renewable generation available at time t (MW)

Variables

$p_g(t)$	Power above minimum for generator g at time t (MW), ≥ 0
$p_W(t)$	Aggregate renewable generation used at time t (MW), ≥ 0
$p_g^l(t)$	Power from piecewise interval l for generator g at time t (MW), ≥ 0
$r_g(t)$	Spinning reserves provided by generator g at time t (MW), ≥ 0
$u_g(t)$	Commitment status of generator g at time t, $\in \{0, 1\}$
$v_g(t)$	Startup status of generator g at time t, $\in \{0, 1\}$
$w_g(t)$	Shutdown status of generator g at time t, $\in \{0, 1\}$
$c_g^{SU}(t)$	Startup cost for generator g at time t (\$), ≥ 0
$\delta_g^s(t)$	Startup in category s for generator g at time t, $\in \{0, 1\}$
$x_g(t, t')$	Indicator arc for shutdown at time t, startup at time t' uncommitted for $i \in [t, t')$, for generator g, $\in \{0, 1\}$
$y_g(t, t')$	Indicator arc for startup at time t, shutdown at time t' committed for $i \in [t, t')$, for generator g, $\in \{0, 1\}$

1 Introduction

The unit commitment problem (UC) concerns the scheduling of thermal generators to meet projected demand for electricity while minimizing system operations cost [34]. Here, we propose a new formulation for representing thermal generator startup costs, which leads to a tightening of the linear programming (LP) relaxation of the mixed-integer linear programming (MILP) UC problem. We then empirically demonstrate that the tighter LP relaxation can translate into reduced run-times to solve the MILP UC using commercial branch-and-cut solvers.

MILP formulations for UC have been of interest since Garver's original formulation [8]. These are extremely difficult problems to solve in practice, e.g., at the scale of the Midcontinent Independent System Operator (MISO) in the United States. Many practical problems involve hundreds to thousands of generators and a time horizon of at least 48 h. Further, solutions must be computed in at most tens of minutes. As a consequence, system operators often have to use substantially suboptimal solutions to comply with the time limit, i.e., with optimality gaps that are sometimes tens of percents [3].

There are a few approaches for reducing run-times to an optimal UC solution. One approach is via decomposition. The intuition is that loosely-connected parts of the UC problem can be decomposed into easier subproblems, and a solution to the original can be developed through an iterative process. One way to decompose UC is by generators—splitting the generator set $\mathcal{G}$ into subsets (by location or other criteria). Classical decomposition methods (such as ADMM) can then be used to force convergence between subproblems [6,28]. Another possible decomposition is on the time horizon—the principle here being that after a sufficiently long period decisions made previously do not significantly impact decisions made now. Such an approach is explored in [12].

An alternative approach for reducing run-times is stronger formulations for UC, and this research has found its way into practice. Most of this work has focused on tightening the polyhedral description of a single generator's dispatch. In [16] an exponential convex hull description for minimum up and down times in terms of a generator's status variables is given; [27] uses the startup and shutdown status variables to describe the same set using only a linear number of inequalities. This result is extended in [10] to generators with startup and shutdown power constraints. Inequalities to tighten the formulation of the ramping process are considered in [5,13,21,23].

A formulation for time-dependent startup costs based on generator commitment variables appears in [20]; [2] considers the same formulation in the context of a MILP approach to UC. Startup cost categories together with associated indicator variables are introduced in [19]. Morales-España et al. [17] improves on the indicator formulation from [30] and demonstrates empirically that the use of startup category indicators results in a tighter formulation than those described in [2,20]. Morales-España et al. [18] uses this same approach to model generator startup and shutdown energy production. Brandenberg et al. [1] shows that the epigraph for concave non-decreasing startup costs modeled using generator status variables has an exponential number of facets. However, [1] provides a linear-time separation algorithm for computing these

facets. Finally, a restrictive temperature-based model for startup cost is presented in [29].

In this paper we introduce a novel matching formulation for time-dependent startup costs in UC. We theoretically analyze the strength of our formulation relative to existing formulations in the literature, and introduce an additional formulation as an intermediary to ease the comparison between existing formulations. We then empirically analyze the impact of our new formulation, both in an absolute sense and relative to other formulations, on the ability of commercial branch-and-cut software packages to solve industrial-scale UC problems.

The remainder of this paper is organized as follows. We begin in Sect. 2 with a discussion of the base UC formulation, without startup cost components. Section 3 then details both existing and two novel startup cost formulations for UC. In Sect. 4, we establish a provable dominance hierarchy concerning the relative tightness of LP relaxations for the different startup cost formulations. We empirically compare the performance of the various startup cost formulations in Sect. 5, using large-scale UC instances based on industrial data. We discuss the implications of our results in Sect. 6. Finally, we conclude with a summary of our contributions in Sect. 7.

2 Unit commitment formulation

We present a MILP UC formulation based on [17] that we will use as the baseline for our comparison between startup cost formulations. We assume that the production cost is piecewise linear convex in $p_g(t)$, where L_g is the number of piecewise intervals and $\overline{P}_g^0 = \underline{P}_g$ is the start of the first interval. Let $\mathcal{G}^1$ be the subset of generators that have $UT^g = 1$ and $\mathcal{G}^{>1}$ be the subset of generators with $UT^g > 1$. We then formulate the UC problem as follows:

$$\min \sum_{g\in\mathcal{G}} \sum_{t\in\mathcal{T}} \left(\sum_{l\in\mathcal{L}_g} (c_g^l \; p_g^l(t)) + c_g^u \, u_g(t) + c_g^{SU}(t) \right) \tag{1a}$$

subject to:

$$\sum_{g\in\mathcal{G}} \Big(p_g(t) + \underline{P}_g u_g(t) \Big) + p_W(t) = D(t) \qquad \forall t \in \mathcal{T} \tag{1b}$$

$$\sum_{g\in\mathcal{G}} r_g(t) \geq R(t) \qquad \forall t \in \mathcal{T} \tag{1c}$$

$$\begin{aligned} p_g(t) + r_g(t) &\leq (\overline{P}_g - \underline{P}_g) u_g(t) \\ &\quad - (\overline{P}_g - SU_g) v_g(t) \qquad \forall t \in \mathcal{T},\ \forall g \in \mathcal{G}^1 \end{aligned} \tag{1d}$$

$$\begin{aligned} p_g(t) + r_g(t) &\leq (\overline{P}_g - \underline{P}_g) u_g(t) \\ &\quad - (\overline{P}_g - SD_g) w_g(t+1) \qquad \forall t \in \mathcal{T},\ \forall g \in \mathcal{G}^1 \end{aligned} \tag{1e}$$

$$p_g(t) + r_g(t) \leq (\overline{P}_g - \underline{P}_g) u_g(t)$$

$$
\begin{aligned}
&- (\overline{P}_g - SU_g)v_g(t) \\
&- (\overline{P}_g - SD_g)w_g(t+1) && \forall t \in \mathcal{T},\ \forall g \in \mathcal{G}^{>1} && \text{(1f)}\\
&p_g(t) + r_g(t) - p_g(t-1) \le RU_g && \forall t \in \mathcal{T},\ \forall g \in \mathcal{G} && \text{(1g)}\\
&p_g(t-1) - p_g(t) \le RD_g && \forall t \in \mathcal{T},\ \forall g \in \mathcal{G} && \text{(1h)}\\
&p_g(t) = \sum_{l \in \mathcal{L}_g} p_g^l(t) && \forall t \in \mathcal{T},\ \forall g \in \mathcal{G} && \text{(1i)}\\
&p_g^l(t) \le (\overline{P}_g^l - \overline{P}_g^{l-1})u_g(t) && \forall t \in \mathcal{T},\ \forall l \in \mathcal{L}_g,\ \forall g \in \mathcal{G} && \text{(1j)}\\
&u_g(t) - u_g(t-1) = v_g(t) - w_g(t) && \forall t \in \mathcal{T},\ \forall g \in \mathcal{G} && \text{(1k)}\\
&\sum_{i=t-UT_g+1}^{t} v_g(i) \le u_g(t) && \forall t \in [UT_g, T],\ \forall g \in \mathcal{G} && \text{(1l)}\\
&\sum_{i=t-DT_g+1}^{t} w_g(i) \le 1 - u_g(t) && \forall t \in [DT_g, T],\ \forall g \in \mathcal{G} && \text{(1m)}\\
&p_W(t) \le W(t) && \forall t \in \mathcal{T} && \text{(1n)}\\
&p_g^l(t) \in \mathbb{R}_+ && \forall t \in \mathcal{T},\ \forall l \in \mathcal{L}_g,\ \forall g \in \mathcal{G} && \text{(1o)}\\
&p_g(t), r_g(t) \in \mathbb{R}_+ && \forall t \in \mathcal{T},\ \forall g \in \mathcal{G} && \text{(1p)}\\
&p_W(t) \in \mathbb{R}_+ && \forall t \in \mathcal{T} && \text{(1q)}\\
&u_g(t), v_g(t), w_g(t) \in \{0,1\} && \forall t \in \mathcal{T},\ \forall g \in \mathcal{G}. && \text{(1r)}
\end{aligned}
$$

The objective function (1a) minimizes the cost to operate the system, with respect to generators' piecewise production costs, minimum running costs, and startup costs. Constraints (1b–1r) are standard in UC formulations without time-varying startup costs [13,17]. Constraint (1b) balances supply and demand, while Constraint (1c) ensure sufficient (spinning) reserves are available. The remaining constraints describe the technical capabilities of each generator. Constraints (1d–1f) ensure each generator does not exceed its maximum capacity $\overline{P}$ when on ($u_g(t) = 1$), its startup capacity SU_g when turning on ($v_g(t) = 1$), nor its shutdown capacity SD_g when turning off ($w_g(t+1) = 1$). Constraints (1g) and (1h) ensures each generator does not exceed its ramp-up and ramp-down capability, respectively, between consecutive time periods. Constraints (1i) and (1j) model the convex piecewise production cost for each generator. Constraints (1k – 1m) describe the relationship between the on ($u_g(t)$), turn-on ($v_g(t)$), and turn-off ($w_g(t)$) variables; constraints (1l) and (1m) additionally model the minimum up time and down time constraints, respectively. Finally, Constraint (1n) sets an upper bound on the aggregate renewable generation scheduled at each time t, while the remaining constraints describe the non-negativity and binary restrictions on the appropriate variables.

We will take the above formulation as given, and for the remainder of the paper we will focus on the formulation of the startup costs $c_g^{SU}(t)$.

3 Startup cost formulations

In this section, we introduce the formulations for startup cost $c_g^{SU}(t)$ examined in this paper. For notational ease, since in all cases we are referencing a single generator, we will drop the subscript g on all variables and parameters in this section and in the following section.

3.1 Formulations from the literature

3.1.1 One binary formulation (1-bin)

The typical formulation for startup costs using only the status variable u is [2,20]

$$c^{SU}(t) \geq c^s \left(u(t) - \sum_{i=1}^{\underline{T}^s} u(t-i) \right) \qquad \forall s \in \mathcal{S},\ \forall t \in \mathcal{T} \tag{2a}$$

$$c^{SU}(t) \geq 0 \qquad \forall t \in \mathcal{T}. \tag{2b}$$

This formulation has the advantage of only needing as many constraints as startup types, and no additional variables.

3.1.2 Strengthened one binary formulation (1-bin*)

As pointed out in [29], the 1-bin formulation above can be strengthened by increasing the coefficients on the $u(t-i)$ variables as follows:

$$c^{SU}(t) \geq c^s \left(u(t) - \sum_{i=1}^{DT} u(t-i) \right) - \sum_{k=1}^{s-1} \left((c^s - c^k) \sum_{i=\underline{T}^k+1}^{\overline{T}^k} u(t-i) \right) \qquad \forall s \in \mathcal{S},\ \forall t \in \mathcal{T} \tag{3a}$$

$$c^{SU}(t) \geq 0 \qquad \forall t \in \mathcal{T}. \tag{3b}$$

3.1.3 Startup type indicator formulation (STI)

The formulation proposed in [17] introduces binary indicator variables for each startup type. Specifically, for each startup type s, we have $\delta^s(t)$, $\forall t \in \mathcal{T}$, which is 1 if the generator has a type s startup in time t and 0 otherwise. The corresponding constraints are

$$\delta^s(t) \leq \sum_{i=\underline{T}^s}^{\overline{T}^s-1} w(t-i) \qquad \forall s \in \mathcal{S}\backslash\{S\},\ \forall t \in \mathcal{T} \tag{4a}$$

$$v(t) = \sum_{s=1}^{S} \delta^s(t) \qquad \forall t \in \mathcal{T}. \tag{4b}$$

We can replace the objective function variables $c^{SU}(t)$ using the substitution

$$c^{SU}(t) = \sum_{s=1}^{S} c^s \delta^s(t) \qquad \forall t \in \mathcal{T}. \tag{4c}$$

3.1.4 Extended formulation (EF)

The authors of [26] propose an extended formulation for startup and shutdown sequences, which provides a perfect formulation for startup costs, in the space of binary variables. We call a formulation *perfect* if the vertices of the polytope described by the formulation are integer. Note that if other variables or constraints are added, a formulation may lose this property. Let $y(t, t') = 1$ if there is a startup in time t and a shutdown in time t' and 0 otherwise, for $t' \geq t + UT$. Similarly let $x(t, t') = 1$ if there is a shutdown in time t and a startup in time t' and 0 otherwise, for $t' \geq t + DT$. The constraints are

$$\sum_{\{t'|t'>t\}} y(t, t') = v(t) \qquad \forall t \in \mathcal{T} \tag{5a}$$

$$\sum_{\{t'|t'<t\}} y(t', t) = w(t) \qquad \forall t \in \mathcal{T} \tag{5b}$$

$$\sum_{\{t'|t'<t\}} x(t', t) = v(t) \qquad \forall t \in \mathcal{T} \tag{5c}$$

$$\sum_{\{t'|t'>t\}} x(t, t') = w(t) \qquad \forall t \in \mathcal{T} \tag{5d}$$

$$\sum_{\{\tau,\tau'|\tau\leq t<\tau'\}} y(\tau, \tau') = u(t) \qquad \forall t \in \mathcal{T}. \tag{5e}$$

Note that with constraints (5a–5e), constraints (1k–1m) become redundant. Hence, the u, v, and w variables may be eliminated along with constraints (1k–1m) while not losing validity or strength. The startup costs are calculated by placing the appropriate coefficient on the x variables via

$$c^{SU}(t) = \sum_{s=1}^{S} c^s \left(\sum_{t'=t-\overline{T}^s+1}^{t-\underline{T}^s} x(t', t) \right) \qquad \forall t \in \mathcal{T}, \tag{5f}$$

where the inside summation is understood to be taken over valid t'.

From integer programming theory [33], we know this formulation to be integral because it is a network flow model, where the vertices are two partite sets, one for

startups and the other for shutdowns, and the arcs y connect startups to feasible shutdowns and the arcs x connect shutdowns to feasible startups. By putting a flow of one unit through the network, we arrive at a feasible generator schedule. Note integrality comes at the cost of needing $\mathcal{O}(|\mathcal{T}|^2)$ additional variables to model startup costs.

3.2 Novel formulations

Here we present two new formulations for startup costs. The first can be seen as a relaxation of EF, and the second as 1-bin* with the inequalities strengthened by using the startup/shutdown indicators v and w.

3.2.1 Matching formulation (match)

Similar to EF, for $t \in \mathcal{T}$ let $x(t', t) = 1$ if there is a shutdown in time t' and a startup in time t and 0 otherwise, for $t' \in \mathcal{T}$ such that $t - TC < t' \leq t - DT$. Note that this only requires $(TC - DT)|\mathcal{T}|$ additional variables. The associated constraints are

$$\sum_{t'=t-TC+1}^{t-DT} x(t', t) \leq v(t) \qquad \forall t \in \mathcal{T} \tag{6a}$$

$$\sum_{t'=t+DT}^{t+TC-1} x(t, t') \leq w(t) \qquad \forall t \in \mathcal{T} \tag{6b}$$

(where again the sums are understood to be taken over valid t'), and the objective function is

$$c^{SU}(t) = c^S v(t) + \sum_{s=1}^{S-1} (c^s - c^S) \left(\sum_{t'=t-\overline{T}^s+1}^{t-\underline{T}^s} x(t', t) \right) \qquad \forall t \in \mathcal{T}. \tag{6c}$$

Note that if $v(t)$ and $w(t)$ are already determined, these equations serve to match shutdowns with startups. That is, if $v(t) = 1$ and $w(t') = 1$, then in any optimal solution $x(t', t) = 1$ since $c^s - c^S < 0$. We arrive at this formulation by eliminating the arcs y from EF and the arcs $x(t', t)$ such that $t - t' \geq TC$.

3.2.2 Three binary formulation (3-bin)

This formulation is similar in spirit to the 1-bin* formulation, only instead of using the status variables u, we use the startup/shutdown variables v and w to keep track of the different types of startups, as follows:

$$c^{SU}(t) \geq c^s v(t) - \sum_{k=1}^{s-1} \left((c^s - c^k) \sum_{i=\underline{T}^k}^{\overline{T}^k-1} w(t-i) \right) \qquad \forall s \in \mathcal{S},\ \forall t \in \mathcal{T} \tag{7a}$$

$$c^{SU}(t) \geq 0 \qquad \forall t \in \mathcal{T}. \qquad (7b)$$

Equation (7a) works analogously to equation (3a). That is, if we did not shutdown in the last $\overline{T}^s$ time periods, (7a) ensures that we pay at least c^s for a startup in time t. Note that when $s = 1$, the second term is an empty sum, and hence is 0. Equation (7b) ensures that the startup cost is never negative. Note that relative to STI, this formulation needs the same number of constraints and has fewer variables (only the additional $|\mathcal{T}|$ variables for $c^{SU}(t)$, which can be eliminated for STI, while not needing the indicator variables δ). This formulation is presented to ease the comparison between STI and the 1-bin formulations.

4 Dominance hierarchy of startup cost formulations

In this section we establish relationships between the six formulations presented in Sect. 3. First, we consider the relationship between the tightness of each formulation. Let z_{EF}, z_{Match}, z_{STI}, z_{3bin}, z_{1bin*}, and z_{1bin} be the linear programming relaxation values for the respective formulations; recall we are always interested in a minimization problem. A basic assumption we need is that startup costs are non-decreasing, that is, for every $g \in \mathcal{G}$, $c_g^s \leq c_g^{s+1}$, $\forall s \in \mathcal{S}_g \backslash \{S_g\}$. The formulations presented in Sect. 3 are invalid without some version of this assumption, except for EF. In practice it is a safe assumption because the heat required to restart a generator is an increasing function of time, and so the total cost to restart a generator will also be increasing in time. We have the following:

Theorem 1 *When startup costs are non-decreasing,*

$$z_{1bin} \leq z_{1bin*} \leq z_{3bin} \leq z_{STI} \leq z_{Match} \leq z_{EF},$$

that is, EF is the tightest formulation, 1-bin is the weakest formulation, with the relationship above amongst the others.

Proof Since the EF formulation is the convex hull description, it is clear that it is the tightest formulation, implying $z_{Match} \leq z_{EF}$. Furthermore, [29] shows that 1bin* is a tighter formulation than 1bin, implying that $z_{1bin} \leq z_{1bin*}$. As a result, we only need to prove the inner three binary relationships. To prove these relationships, i.e., that $z_A \leq z_B$, it is sufficient to show: (1) there is a linear mapping from the polytope associated with B onto the polytope associated with A that preserves objective value, and (2) that through this linear mapping, every constraint in formulation A is implied by constraints in formulation B. This is sufficient to show that $z_A \leq z_B$ as it shows that *all* feasible solutions for B can be mapped to solutions feasible for A with the same objective value.

$\mathbf{z_{STI} \leq z_{Match}}$: We proceed by demonstrating that all the inequalities in STI are implied by the inequalities in Match. First, consider the linear transformation from Match to STI

$$\delta^s(t) = \sum_{t'=t-\overline{T}^s+1}^{t-\underline{T}^s} x(t', t) \qquad \forall s \in \mathcal{S}\backslash\{S\},\ \forall t \in \mathcal{T} \tag{8}$$

$$\delta^S(t) = v(t) - \sum_{t'=t-TC+1}^{t-DT} x(t', t) \qquad \forall t \in \mathcal{T}. \tag{9}$$

The equality constraints (4b) follow directly from the sum of (8) and (9). To see (4a), notice that by (6b),

$$x(i, t) \leq w(i) \qquad \forall t \in \{i + DT, \ldots, i + TC - 1\}, \forall i \in \mathcal{T}. \tag{10}$$

By (8) and (10), we have

$$\delta^s(t) \leq \sum_{i=t-\overline{T}^s+1}^{t-\underline{T}^s} w(i) = \sum_{i=\underline{T}^s}^{\overline{T}^s-1} w(t-i) \qquad \forall s \in \mathcal{S}\backslash S,\ \forall t \in \mathcal{T}, \tag{11}$$

which is just (4a).

$\mathbf{z_{3bin}} \leq \mathbf{z_{STI}}$: This follows by eliminating the indicators δ from the objective function using (4a) and (4b). As $c^k \geq c^s$ for all $k \in \mathcal{S}$ such that $k > s$, we have

$$\begin{aligned} c^{SU}(t) &= \sum_{k=1}^{S} c^k \delta^k(t) \\ &\geq \sum_{k=1}^{s-1} c^k \delta^k(t) + c^s \sum_{k=s}^{S} \delta^k(t) \\ &= c^s \sum_{k=1}^{S} \delta^k(t) - \sum_{k=1}^{s-1} (c^s - c^k)\delta^k(t) \\ &\geq c^s v(t) - \sum_{k=1}^{s-1} \left((c^s - c^k) \sum_{i=\underline{T}^k}^{\overline{T}^k-1} w(t-i) \right) \qquad \forall s \in \mathcal{S},\ \forall t \in \mathcal{T}, \end{aligned} \tag{12}$$

which is (7a). Equation (7b) follows from the non-negativity of c^s and $\delta^s(t)$.

$\mathbf{z_{1bin^*}} \leq \mathbf{z_{3bin}}$: (3b) and (7b) are the same, so we need show that (7a) implies (3a). Consider the inequality (7a), noting $v(t) \geq u(t) - u(t-1)$ by (1k) and $-w(t) \geq -u(t-1)$ by (1k) and (1l)

$$
\begin{aligned}
c^{SU}(t) &\geq c^s v(t) - \sum_{k=1}^{s-1} \left((c^s - c^k) \sum_{i=\underline{T}^k}^{\overline{T}^k - 1} w(t-i) \right) \\
&\geq c^s (u(t) - u(t-1)) - \sum_{k=1}^{s-1} \left((c^s - c^k) \sum_{i=\underline{T}^k}^{\overline{T}^k - 1} u(t-1-i) \right) \\
&\geq c^s (u(t) - u(t-1)) - \sum_{k=1}^{s-1} \left((c^s - c^k) \sum_{i=\underline{T}^k + 1}^{\overline{T}^k} u(t-i) \right) \\
&\geq c^s \left(u(t) - \sum_{i=1}^{DT} u(t-i) \right) - \sum_{k=1}^{s-1} \left((c^s - c^k) \sum_{i=\underline{T}^k + 1}^{\overline{T}^k} u(t-i) \right) \\
&\forall s \in \mathcal{S},\ \forall t \in \mathcal{T}, \qquad (13)
\end{aligned}
$$

which is (3a). □

We note that Theorem 1 does not establish strict dominance. We only guarantee that, for example, EF is no worse than Match in its linear programming relaxation. Some of these relationships hold with equality under non-decreasing startup costs.

Theorem 2 *Suppose startup costs are non-decreasing in time. Then* $z_{3bin} = z_{STI}$ *and* $z_{Match} = z_{EF}$.

We defer proof of Theorem 2 to the online supplement [14]. We additionally show that in the space of binary variables, the Match formulation is integer optimal when startup costs are non-decreasing. This makes Match interesting from both a theoretical and practical perspective—under certain restrictions on the objective function it returns an integer optimal solution, just like the EF. But because Match exploits the objective structure, it is able to use many fewer variables and constraints, which allows it to be practically useful. In particular, Match only grows linearly in time (as compared to quadratic for the EF), so for long time horizons it is much more computationally tractable, while still preserving guarantees on integrality. As we will see in Sect. 5, Match also preserves this edge over EF when embedded in a large UC problem.

Table 1 compares formulation size as a function of problem parameters for each startup formulation. Note we only consider the variables needed in addition to the baseline formulation (1).

5 Computational experiments

The dominance hierarchy for the various startup cost formulations introduced in Sect. 4 formally establishes their relative tightness. We quantify tightness as the optimal objective function value for the LP relaxation of the UC problem with a given startup cost formulation. In the context of a MILP, tighter LP relaxations can lead to more efficient branch-and-cut search, due to increased fathoming opportunities. However, the

Table 1 Size of the formulations

Formulation	# Variables	# Constraints
1-bin	$\mathcal{O}(\|\mathcal{T}\|)$	$\mathcal{O}(\|\mathcal{S}\|\|\mathcal{T}\|)$
1-bin*	$\mathcal{O}(\|\mathcal{T}\|)$	$\mathcal{O}(\|\mathcal{S}\|\|\mathcal{T}\|)$
3-bin	$\mathcal{O}(\|\mathcal{T}\|)$	$\mathcal{O}(\|\mathcal{S}\|\|\mathcal{T}\|)$
STI	$\mathcal{O}(\|\mathcal{S}\|\|\mathcal{T}\|)$	$\mathcal{O}(\|\mathcal{S}\|\|\mathcal{T}\|)$
Match	$\mathcal{O}((TC - DT)\|\mathcal{T}\|)$	$\mathcal{O}(\|\mathcal{T}\|)$
EF	$\mathcal{O}(\|\mathcal{T}\|^2)$	$\mathcal{O}(\|\mathcal{T}\|)$

size and structure of the underlying LP varies across startup cost formulations, and reductions in branch-and-cut search time (measured in terms of number of tree nodes explored) may be offset by the cost of solving the LP relaxations at each node. Further, formulation details interact with heuristics and other features of MILP solvers, often in unpredictable ways.

In this context, we now experimentally compare the performance of the range of startup cost formulations for UC, using two state-of-the-art commercial MILP solvers. We consider two sets of problem instances. The first set of instances are realistic instances derived from publicly available market and regulatory data obtained from the California Independent System Operator (CAISO) in the US. The second set is the FERC generator set [15] (which itself is based on data from the PJM Interconnection in the US), with demand, reserve, and wind scenarios based on publicly available data obtained from PJM for 2015 [24,25]. Both sets of problem instances are available as part of the IEEE PES Power Grid Lib—Unit Commitment benchmark library (https://github.com/power-grid-lib/pglib-uc).

The "CAISO" instances have 610 thermal generators, of which 410 are schedulable, i.e., not forced to run. Generators with quadratic cost curves were approximated using $L_g = 2$. Five 48-h demand scenarios were examined; demands were taken directly from CAISO historical data. Four of the demand scenarios are based on historical information, while "Scenario400" is a hypothetical scenario where wind supply is on average 40% of demand; the wind profile is constructed based on actual CAISO wind data, scaled appropriately. For each instance the reserve level was varied from 0%, 1%, 3%, and 5% of demand, resulting in a total of 20 test instances. We allow for the possibility of curtailment of wind generation by (1b) and (1n). Each generator has only two startup categories, i.e., $S_g = 2$.

The "FERC" instances are based on two generator sets publicly available from the US Federal Energy Regulatory Commission (FERC): a "Summer" set of generators and a "Winter" set of generators [15]. We use the Summer set of generators for dates in April - September and the Winter set for the remaining dates. After (i) excluding generators with missing or negative cost curves, (ii) letting $UT_g = DT_g = 1$ for generators g with missing up/down time data, and (iii) eliminating generators marked as wind (we consider wind power separately), the Summer and Winter sets respectively contain 978 and 934 generators. No data on startup or shutdown power limits was provided by FERC, so we assume $SU_g = SD_g = \underline{P}^g$. Similarly, FERC provided no data for cool-down times, so we set $TC_g = 2DT_g$. All generators had at most two

startup types, i.e., $S_g \leq 2$, and the piecewise production cost curves are based on market bids, such that $1 \leq L_g \leq 10$.

For the FERC instances, we consider twelve 48-h demand, reserve, and wind scenarios from 2015, one for each month. In 2015, wind generation accounted for 2% of the electricity supplied in PJM, so we created twelve additional "high-wind" scenarios by multiplying the wind data for 2015 by a constant factor of 15 to increase mean wind energy supply for the year to 30% of load. A recent study conducted for PJM suggests that in less than a decade, renewables could achieve 30% penetration rates in the interconnection [9]. Like the CAISO instances, we allow for the curtailment of wind generation.

The two test instance sets represent vastly different systems. The CAISO instances consist of mostly small, flexible generators. Of the 410 schedulable generators, only 20 have irredundant ramping constraints (i.e., $RU_g \geq (\overline{P}_g - \underline{P}_g)$ and $RD_g \geq (\overline{P}_g - \underline{P}_g)$). Therefore, for 390 of the generators (95% of the total), EF, together with the equations from (1), is a convex hull description of each generator's dispatch. These flexible generators account for 75% of schedulable capacity. For both the Summer and Winter FERC generator sets, such flexible generators only account for 50% of the fleet, and approximately 30% of schedulable capacity.

Computational experiments were conducted on a Dell PowerEdge T620 with two Intel Xeon E5-2670 processors, for a total of 16 cores and 32 threads, 256GB of RAM, running the Ubuntu 14.04.5 Linux operating system. The Gurobi 7.0.1 MILP solver was used for the experiments labeled "Gurobi", while the CPLEX 12.7.1.0 MILP solver was used for the experiments labeled "CPLEX". Both solvers were allowed to use all 32 threads in each experimental trial. Here we present summaries of the computational experiments; the full results are available in the online supplement [14].

5.1 CAISO instances

We first consider the experimental results for the CAISO instances. For both Gurobi and CPLEX, we impose a wall-clock time limit of 600 s; all other settings were left at their defaults. To communicate the scale of the problems, Table 2 reports the mean size of each UC formulation in terms of the number of rows, columns, and non-zeros in the constraint matrix. Note that these are mostly the same, though the hypothetical "Scenario400" instances have 48 more variables than the others for the aggregate renewables production. We see that all the variants, except for EF, have approximately 1.2 million non-zeros in the constraint matrices. The EF constraint matrices are an

Table 2 Summary of computational experiments for CAISO instances: problem size

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Rows	473,591	472,201	471,255	440,670	440,670	440,670
Columns	1,700,551	316,171	335,291	276,731	276,731	276,731
Non-zeros	15,719,918	1,247,464	1,254,913	1,162,656	1,226,079	1,226,079

We report the mean for each category across the 20 instances

Table 3 Summary of computational experiments for CAISO instances: problem size after first presolve using Gurobi

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Rows	131,101	116,298	115,980	132,516	133,125	133,125
Columns	1,048,028	106,851	98,130	115,076	115,685	115,685
Non-zeros	9,363,964	385,237	369,223	420,672	483,085	483,085

We report the mean for each category across the 20 instances

Table 4 Summary of computational experiments for CAISO instances: problem size after first presolve using CPLEX

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Rows	152,937	138,181	102,151	134,808	135,417	135,417
Columns	1,067,614	126,482	101,048	115,109	115,718	115,718
Non-zeros	10,187,193	446,590	388,940	421,540	465,931	465,889

We report the mean for each category across the 20 instances

Table 5 Summary of computational experiments for CAISO instances using Gurobi

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Time (s)	370.5	**43.12**	52.84	91.43	600	600
# of times best	0	**11**	8	1	0	0
# of times 2nd	0	8	**11**	1	0	0
Max. time (s)	600	**130**	243	600	600	600
# of time outs	7	**0**	**0**	1	20	20
# B&C nodes	1.510	**13.50**	20.22	39.09	5,914	5,827

For time (s) and number of branch-and-cut (B&C) nodes we report the geometric mean across the 20 instances, including those which reach the wall-clock limit of 600 s

order of magnitude larger with approximately 15.7 million non-zeros. We report the same problem size statistics after the (first) presolve for both Gurobi and CPLEX in Tables 3 and 4.

We summarize the results for these instances in Tables 5 and 6. For each UC formulation we report the geometric mean time to an optimal solution (Time (s)), the number of instances for which that method did best (# of times best), the number of instances for which that method did second best (# of times 2nd), the longest run-time across the 20 instances (Max. time (s)), and the number of instances for which that method hit the 600 s time limit (# of time outs). When a solver times out for an instance, we substitute the time limit in the calculation for the geometric mean, leading to an underestimation when an instance fails to solve for a given UC formulation. In the last row, we report the shifted geometric mean number of branch-and-cut tree nodes explored by the solver, substituting the number of nodes explored when the solver hits the time limit. To compute the shifted geometric mean, we add 1 to each node count, so as to avoid multiplying by 0 when the solver identifies a solution at the root node.

Table 6 Summary of computational experiments for CAISO instances using CPLEX

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Time (s)	261	48.0	**40.8**	62.9	600	600
# of times best	0	5	**11**	4	0	0
# of times 2nd	0	6	**8**	6	0	0
Max. time (s)	600	**110**	223	423	600	600
# of time outs	2	**0**	**0**	**0**	20	20
# of B&C nodes	3.60	**2.83**	4.75	32.6	15,442	15,210

For time (s) and number of branch-and-cut (B&C) nodes we report the geometric mean across the 20 instances, including those which hit the wall-clock limit of 600 s

A bold-faced entry in a row denotes the startup cost variant that performed best for the given measure.

We immediately see that both of the 1-bin variants are not competitive, and in no case identify an optimal solution within the time limit, even after exploring a considerable number of branch-and-cut nodes. This is consistent with results reported recently in the UC literature. Gurobi identifies optimal solutions to the EF variant in less than half the cases, and CPLEX identifies optimal solutions in all but two cases. However, for both solvers, the EF variant exhibits significantly larger run-times—presumably due to the size of the LP formulation—than those observed for the Match, STI, or 3-bin variants.

Overall, the 3-bin variant is not competitive with the Match and STI variants, and Gurobi times out for one instance. Using CPLEX, the 3-bin variant is often the best or second-best, but when it performs poorly 3-bin often takes much longer than the Match and STI variants.

Comparing the Match and STI variants, we can see that overall Gurobi performs better using the Match variant while CPLEX performs better using the STI variant. However, for both solvers, the Match variant has a significantly (approximately 50%) lower maximum time across the CAISO test instances, suggesting it may be a more robust UC formulation in practice. That said, on average Gurobi seems to perform better on the Match variant and CPLEX on the STI variant. This may be partially explained by differences in the default presolve operations. Returning to Tables 3 and 4, we see that while the presolved constraint matrix non-zero counts are similar for Match and STI with Gurobi, with CPLEX the STI variant's non-zero count is considerably smaller than that of Match after presolve. Additionally, for both the Match and STI variants, solution times generally grow with increases in reserve level; we refer to the detailed results in [14]. The latter observation has significant potential impact on stochastic unit commitment solvers, as we discuss further below in Sect. 6.

Turning to the number of branch-and-cut nodes explored, in the case of the 1-bin variants, the large number of nodes explored is consistent with the inability of the solver to identify optimal solutions within the specified time limit. Interestingly, Gurobi and CPLEX typically did not leave the root node processing phase within the 600 s time limit when considering the EF variant. Further, we note that the size of the EF formulation makes cut generation (and heuristics) at the root node more

Table 7 Computational results for CAISO instances: relative integrality gap (%), geometric mean across 20 instances

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Gap (%)	0.008	0.008	0.033	0.033	1.525	1.569

Table 8 Computational results for CAISO instances: root relaxation relative integrality gap (%) using Gurobi, geometric mean across 20 instances

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Gap (%)	0.008	0.008	0.031	0.031	1.516	1.516

Table 9 Computational results for CAISO instances: root relaxation relative integrality gap (%) using CPLEX, geometric mean across 20 instances

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Gap (%)	0.008	0.008	0.033	0.033	1.524	1.567

difficult. In the case of the Match and STI variants, both Gurobi and CPLEX identify an optimal solution at the root node for instances with relatively low reserve levels, with CPLEX finding a root node solution more often. However, as reserve levels increase, the number of nodes explored increases. Finally, we observe that the relatively few number of tree nodes explored with the tighter Match and STI variants indicates relatively few opportunities for parallelism, at least in terms of accelerating the tree search process.

In Table 7 we report the geometric mean relative integrality gap for each startup cost formulation. For each instance, we compute the relative integrality gap by taking the best integer solution objective value found across all six formulations and both solvers, denoted z^*_{IP}, and the objective value of the LP relaxation for each instance (as computed by Gurobi after relaxing the binary variables), denoted z^*_{LP}; we then report $(z^*_{IP} - z^*_{LP})/z^*_{IP}$ as a percentage. First, we observe that the results in Table 7 are consistent with and empirically verify the correctness of Theorem 1. The 1-bin variants are significantly weaker than the other variants, with the relative integrality gap typically exceeding 1%. We also note that in all instances, the relative integrality gap (and hence LP relaxation) for the EF and Match variants is identical; an analogous situation is observed for the STI and 3-bin variants. Tables 8 and 9 report the same geometric mean relative integrality gap using the MILP root relaxation in place of the LP relaxation, for both Gurobi and CPLEX, respectively. We report these values because MILP presolve routines can often tighten a formulation before any cut generation is done. However, for both Gurobi and CPLEX, we observe that neither presolve routine can add much additional tightening beyond the LP relaxation for these instances. Lastly, we note that the Match formulation typically closes 50–90% of the integrality gap relative to STI (74% in geometric mean), which explains its computational benefit despite the additional variables required.

Table 10 Summary of computational experiments for FERC instances: problem size

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Rows	761,587	745,175	708,006	658,000	658,000	658,000
Columns	2, 072,172	513,131	524,617	466,489	466,489	466,489
Non-zeros	20,760,555	2,501, 087	2,435,228	2,325,603	2,492,992	2,492,992

We report the mean for each category across the 24 instances

Table 11 Summary of computational experiments for FERC instances: problem size after first presolve using Gurobi

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Rows	370,213	333,375	329,731	331,263	346,927	346,927
Columns	1,595,099	289,096	259,611	260,628	275,678	275,678
Non-zeros	12,834,297	1,980,389	1,951,896	1,964,124	2,373,371	2,373,371

We report the mean for each category across the 24 instances

5.2 FERC instances

Because the FERC instances are larger and therefore likely more difficult than the CAISO instances, we increased the wall-clock time limit to 900 s. Further, for Gurobi, we set the `Method` parameter to 3 so Gurobi would use the non-deterministic concurrent optimizer to solve the root LP relaxations. The non-deterministic concurrent optimizer solves LPs by running primal and dual simplex on one thread each and a barrier plus crossover method on the remaining 14 threads, returning an optimal LP basis from whichever method finishes first. All other settings for Gurobi were left at their defaults. CPLEX settings were preserved at their defaults. When describing the computational results below, we separate the instances into two categories: the results considering the 2% wind penetration levels observed in 2015 and hypothetical 30% wind penetration levels based on the same data.

Tables 10, 11 and 12 report the mean problem sizes in terms of the number of rows, columns, and non-zeros in constraint matrices, both before and after the first presolve. There is some variation in the size of FERC instances due to the different number of "Winter" versus "Summer" generators, but all problem sizes are of the same order of magnitude. The FERC instances are approximately twice as large a the CAISO instances as measured by the number of constraint matrix non-zeros: all the variants except EF have 2.3–2.5 million non-zeros in the constraint matrix before presolve, and again the EF variant is an order of magnitude larger with 20 million constraint matrix non-zeros.

We now summarize the computational experiments for both Gurobi and CPLEX for the FERC instances. Tables 13 and 14 report the same statistics for the FERC instances as Tables 5 and 6 did for the CAISO instances. First, we consider the 2% wind penetration instances, which are reported in part (a) of both tables. We observe that the 1-bin variants and EF are not competitive with the Match and STI variants. As was the case with the CAISO instances, Match performs best with Gurobi, whereas

Table 12 Summary of computational experiments for FERC instances: problem size after first presolve using CPLEX

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
Rows	374,996	338,503	326,172	327,366	343,208	343,208
Columns	1,600,419	295,723	264,914	258,299	273,277	273,276
Non-zeros	17,389,320	1,427,958	1,347,588	1,351,194	1,529,739	1,523,974

We report the mean for each category across the 24 instances

Table 13 Summary of computational experiments for FERC instances using Gurobi

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
(a) 2% Wind Penetration						
Time (s)	702	**154**	218	267	712	739
# of times best	0	**6**	4	2	0	0
# of times 2nd	0	**6**	5	1	0	0
Max. time (s)	900	**411**	491	841	900	900
# of time outs	4	**0**	**0**	**0**	7	7
# of B&C nodes	1.00	**1.38**	5.91	9.03	67.5	50.8
(b) 30% Wind Penetration						
Time (s)	808	**215**	391	401	799	804
# of times best	0	**8**	2	2	0	0
# of times 2nd	2	1	**6**	3	0	0
Max. time (s)	900	**648**	900	900	900	900
# of time outs	6	**0**	2	3	10	10
# of B&C nodes	1.00	**4.66**	51.7	78.2	142	130

For time (s) and number of branch-and-cut (B&C) nodes we report the geometric mean across the 12 instances, including those which hit the wall-clock limit of 900 s

STI performs better with CPLEX. Returning to Tables 11 and 12, we see for these instances the CPLEX presolve routine is able to reduce the number of constraint matrix non-zeros for the STI variant significantly more than for the Match variant. As a result, CPLEX does not find these instances difficult, solving all 12 problems using the Match and STI variants at the root node. The 3-bin variant is occasionally the fastest method for CPLEX for a given instance, but mirroring the CAISO instances it has a significantly inferior worse-case solve time than either Match or STI.

We now consider the FERC instances with 30% wind penetration levels. Here, we see that only the Match variant able to solve all 12 instances within the time limit on both solvers. Examining the geometric mean solve time, we observe that the Match variant reduces the solve time relative to the STI variant by 45% for Gurobi, with a more moderate reduction for CPLEX. Overall, the 30% wind penetration level instances are noticeably more difficult than the 2% wind penetration instances. However, for Gurobi, the Match variant requires only 40% more computational time on average to solve the former, while the STI variant requires more than 80% additional computational time.

Table 14 Summary of computational experiments for FERC instances using CPLEX

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
(a) 2% Wind Penetration						
Time (s)	478	136	**114**	162	499	538
# of times best	0	2	**5**	**5**	0	0
# of times 2nd	0	4	**6**	2	0	0
Max. time (s)	900	222	**150**	737	900	900
# of time outs	1	**0**	**0**	**0**	4	5
# of B&C nodes	1.23	**1.00**	**1.00**	7.52	356	414
(b) 30% Wind Penetration						
Time (s)	604	**185**	211	298	784	798
# of times best	0	**5**	2	**5**	0	0
# of times 2nd	1	3	**8**	0	0	0
Max. time (s)	900	**269**	900	900	900	900
# of time outs	2	**0**	1	3	10	10
# of B&C nodes	1.95	**1.94**	5.91	25.3	1,171	1.153

For time (s) and number of branch-and-cut (B&C) nodes we report the geometric mean across the 12 instances, including those which hit the wall-clock limit of 900 s

The situation is similar for CPLEX, where the Match variant only needs 36% more computational time on average for 30% wind instances, whereas STI variant needs 85% more computational time.

Next, we consider the number of branch-and-cut tree nodes explored when solving each instance. Looking at Table 13, we observe that for the Match variant, Gurobi typically locates an optimal solution at the root node, or at least very early in the tree search process. Overall, the number of tree nodes explored under the Match variant is significantly less than that under the STI variant; the latter in turn dominates, as expected, the 3-bin and 1-bin variants. Mirroring the results for CAISO instances, Gurobi does not exit root node processing on the EF variant—in all cases the root relaxation is solved, but the time limit is exhausted applying cuts and heuristics. For both 1-bin variants, Gurobi spends a significant amount of time during root node processing generating cuts, which is why for some instances a small number of nodes are explored before the time limit expires. Consistent with the increase in relative instance difficulty, Gurobi requires more nodes to identify an optimal solution in the case of 30% wind instances, but the increase is much less pronounced than for the STI or 3-bin variants.

Examining the node count summaries for CPLEX in Table 14, we observe that using the Match and STI variants the 2% wind instances are easy, never leaving the root node. Similar to our experience with Gurobi, for 30% wind instances there is only a modest increase in node count for the Match variant (only one instance does not solve at the root node), and larger but still modest increases for STI and 3-bin variants. When it solves, the EF variant does so at the root node, and the 1-bin variants need more enumeration—and usually hit the time limit while still exploring the tree.

Table 15 Computational results for FERC instances: relative integrality gap (%), geometric mean across each of the 12 instances

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
(a) 2% Wind Penetration						
Gap (%)	0.068	0.068	0.075	0.075	0.911	0.911
(b) 30% Wind Penetration						
Gap (%)	0.206	0.206	0.314	0.314	3.003	3.003

Table 16 Computational results for FERC instances: root relaxation relative integrality gap (%) using Gurobi, geometric mean across each of the 12 instances

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
(a) 2% Wind Penetration						
Gap (%)	0.045	0.026	0.033	0.033	0.544	0.544
(b) 30% Wind Penetration						
Gap (%)	0.167	0.076	0.136	0.138	1.910	1.910

Table 17 Computational results for FERC instances: root relaxation relative integrality gap (%) using CPLEX, geometric mean across each of the 12 instances

Formulation	EF	Match	STI	3-bin	1-bin*	1-bin
(a) 2% Wind Penetration						
Gap (%)	0.047	0.047	0.054	0.054	0.714	0.714
30% Wind Penetration						
Gap (%)	0.175	0.175	0.270	0.270	2.475	2.475

Finally, we report the relative integrality gap for each combination of instance and variant in Table 15, calculated in the same manner as those reported in Table 7. We also report the relative integrality gap from the root relaxation value for Gurobi and CPLEX in Tables 16 and 17, respectively. Mirroring the results for the CAISO instances, we observe empirical verification of Theorem 1. Further, the EF and Match variants have identical integrality gaps, as do the STI and 3-bin variants. Unlike as was observed for the CAISO instances, the 1-bin* variant is not tighter than the 1-bin variant, and the Match variant typically only closes 0%–40% of the root gap over STI. This result is partially explained by the fact that approximately half of the generators are ramp-constrained—and even in the EF case, we are not using an ideal formulation for ramp-constrained generators. However, for the January and February 30% wind penetration level instances, the difference is significant (Match closes 95% of the root gap for January and 67% of the root gap for February over STI, see [14]). This is consistent with the result that only the Match and EF variants were able to solve these instances within the time limits on both solvers. For the 2% wind penetration instances the Match variant only closes 8% of the relative integrality gap on average versus STI. The difference in performance between Gurobi and CPLEX can be partially attributed

to the way presolve tightens the relaxation: for CPLEX, the STI variant is only slightly less tight than the Match variant, but has fewer non-zeros in the constraint matrix (see Table 12). On the converse, with Gurobi the difference between the number of constraint matrix non-zeros in STI and Match variants is considerably smaller (see Table 11), and at the root node the Match variant is about 25% tighter—nearly to the 0.01% optimality gap on average.

5.3 Statistical analysis

A statistical analysis of the computational results was performed using the Wilcoxon signed-rank test [32] for both Gurobi and CPLEX. On Gurobi, across the entire test set (both CAISO and FERC), Match is superior to all the other formulations examined at the $\alpha = 0.01$ level. On the other hand, using CPLEX, though Match was faster than STI and 3-bin in mean solve time, these differences were not significant at the $\alpha = 0.05$ level. Further, on the "Low Wind" instances (2% Wind Penetration instances from FERC and the instances corresponding to historical dates from CAISO), the STI variant is able to outperform Match at the $\alpha = 0.01$ level, though the difference in magnitude is only 16.8 s. Finally, we note that for the "High Wind" instances (Scenario400 instances from CAISO and 30% Wind Penetration instances from FERC) Match is 60.4 s faster in mean solve time, but this difference was not significant at the $\alpha = 0.05$ level. This is likely due to this test being underpowered at $n = 16$. The full results of the statistical analysis are available in the online supplement [14].

6 Discussion

We now discuss the implications of the computational experiments described above. First, we note that both the CAISO and FERC test instances have $S_g \leq 2$ for all generators g, due to the data available. In real-world instances, a non-trivial number of generators may have $S_g > 2$. Our proposed Match formulation for startup costs in UC can model more startup categories —up to TC_g—by simply changing the objective coefficients. In contrast, with the exception of EF, all other startup cost formulations require additional variables and/or constraints.

As the experiments on the CAISO instances demonstrate, reserve requirements do have a significant impact on the difficulty of solving UC. For instances with a 0% or 1% reserve requirement, Gurobi is able to solve all instances using the Match formulation in under a minute. This is an interesting observation in the context of stochastic unit commitment [31], in which reserve levels for individual scenarios are minimal, as the scenarios themselves are intended to capture the range of uncertainties that may be encountered. Further, we note that effective decomposition techniques for solving stochastic UC problems – including progressive hedging [4]—repeatedly solve individual (and thus deterministic) scenario problems. Thus, we expect our Match formulation to significantly accelerate the solution of stochastic UCs.

Our experiments also demonstrate that with a modern MILP solver, commodity workstation hardware, and tight formulations, we can quickly solve industrial-scale

UC problems to very small ($< 0.01\%$) optimality gaps. In fact, the results for the CAISO instances suggest they could be solved to even tighter gaps than the Gurobi and CPLEX defaults, within the imposed time limit. Reduced optimality gaps are important to guarantee market fairness, i.e., to ensure that a cheaper generator is scheduled in place of a more expensive one. The ability to run to very small optimality gaps is also important in the context of scenario-based decomposition approaches to stochastic UC. Deterministic UC scenarios often have feasible solutions which are far away from optimal. Thus, imposition of a tighter optimality gap can significantly improve convergence of algorithms such as progressive hedging, by providing strong initial solutions of individual scenarios—which are used to guide subsequent iterations of the algorithm. Further, the ability of progressive hedging to generate high-quality lower bounds for stochastic UC is dependent on how tightly the scenario UC problems are solved [4,7].

In the context of stochastic UC—where renewable energy supply is the main driver of uncertainty—it is interesting to note that for both systems the high-wind scenarios ("Scenario400" for CAISO and the 30% wind penetration instances for FERC) are significantly more difficult, independent of startup formulation. However, these instances are also where our proposed Match formulation shows the most improvement over STI. Across all 16 high-wind scenarios, when using Gurobi the Match variant exhibited a $>44\%$ improvement in geometric mean solve time, with a more modest $>15\%$ improvement using CPLEX, and a 57% improvement in geometric mean relative integrality gap. In comparison, on the other 28 instances, when using Gurobi the Match variant only showed a 20% improvement in geometric mean solve time, with a 20% degradation using CPLEX, and a 50% geometric mean relative integrality gap closure over STI. This is not surprising, given that more variability in net-load implies there will be more switches in generator status.

Finally, we comment on the "synthetic" UC instances from [2], which are extended via replication in [21] and again in [17]. These originate from a now dated genetic algorithm UC paper [11], which has no indication that these were drawn from real-world data. Compared to the generator sets gathered from CAISO and FERC, these instances have much less flexible capacity (less than 10% in all cases), which implies that the ramping process is a much bigger factor in adjusting to changes in demand than generator switching. Additionally, the replication of the same 8 or 10 generators induces artificial symmetry into the problem, which can confound the branch-and-cut process. Though modern commercial MILP solvers have sophisticated symmetry detection algorithms, they do not capture all the symmetry in UC [22]. These factors together imply that the synthetic instances are less likely to be impacted by improvements in startup cost formulations. Based on the instances in [21], we created twenty 48-h UC instances, and tested the six startup cost formulations on the platform described in Sect. 5 using Gurobi. After 1800 s of wall-clock time, the Match, STI, and 3-bin variants were able to solve only 6 of the 20 instances, the EF variant was able to solve 2 of the 20 instances, and the 1-bin variants only 1 of the 20 instances. In geometric mean Match was only able to close 5% of the relative integrality gap over STI. The confounding symmetry and inflexibility in these instances makes it difficult to draw a distinction between the Match, STI, and 3-bin variants, though they all out-perform the 1-bin, 1-bin*, and EF variants.

7 Conclusions

We have presented a novel matching formulation for time-dependent startup costs in UC, and an additional compact formulation for time-dependent startup costs as an intermediary between the STI and the 1-bin formulations. We have formally placed these two new formulations, in addition to existing alternatives, in a formal dominance hierarchy based on the corresponding LP relaxations. We examined the computational efficacy of the various alternative formulations for time-dependent startup costs on large-scale unit commitment instances based on real-world data from the CAISO and PJM independent system operators in the US using two commercial MILP solvers. We find that the proposed matching formulation is computationally as effective on average than the current state-of-the-art formulation, and is computationally more effective for high-wind penetration scenarios. Additionally, we empirically demonstrated that the proposed matching formulation is as tight as the ideal formulation while being more compact.

References

1. Brandenberg, R., Huber, M., Silbernagl, M.: The summed start-up costs in a unit commitment problem. EURO J. Comput. Optim. **5**, 1–36 (2015)
2. Carrión, M., Arroyo, J.M.: A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem. IEEE Trans. Pow. Syst. **21**(3), 1371–1378 (2006)
3. Chen, Y.: Personal correspondence. Principal advisor at midcontinent independent system operator (2016)
4. Cheung, K., Gade, D., Silva-Monroy, C., Ryan, S.M., Watson, J.P., Wets, R.J.B., Woodruff, D.L.: Toward scalable stochastic unit commitment—part 2: solver configuration and performance assessment. Energy Syst. **6**(3), 417–438 (2015)
5. Damcı-Kurt, P., Küçükyavuz, S., Rajan, D., Atamtürk, A.: A polyhedral study of production ramping. Math. Program. **158**(1–2), 175–205 (2016)
6. Feizollahi, M.J., Costley, M., Ahmed, S., Grijalva, S.: Large-scale decentralized unit commitment. Int. J. Electr. Pow. Energy Syst. **73**, 97–106 (2015)
7. Gade, D., Hackebeil, G., Ryan, S.M., Watson, J.P., Wets, R.J.B., Woodruff, D.L.: Obtaining lower bounds from the progressive hedging algorithm for stochastic mixed-integer programs. Math. Program. **157**(1), 47–67 (2016)
8. Garver, L.L.: Power generation scheduling by integer programming-development of theory. Power apparatus and systems, part III. Trans. Am. Inst. Electr. Eng. **81**(3), 730–734 (1962)
9. GE Energy: PJM renewable integration study. PJM interconnection (2014)
10. Gentile, C., Morales-Espana, G., Ramos, A.: A tight MIP formulation of the unit commitment problem with start-up and shut-down constraints. EURO J. Comput. Optim. **5**, 1–25 (2016)
11. Kazarlis, S.A., Bakirtzis, A., Petridis, V.: A genetic algorithm solution to the unit commitment problem. IEEE Trans. Pow. Syst. **11**(1), 83–92 (1996)
12. Kim, K., Botterud, A., Qiu, F.: Temporal decomposition for improved unit commitment in power system production cost modeling. ANL Technical Report (2017). ANL/MCS-P7073-0717
13. Knueven, B., Ostrowski, J., Wang, J.: The ramping polytope and cut generation for the unit commitment problem. INFORMS J. Comput. **30**(4), 739–749 (2018)
14. Knueven, B., Ostrowski, J., Watson, J.P.: Online supplement for a novel matching formulation for startup costs in unit commitment (2020). https://github.com/bknueven/appendices/raw/master/OnlineSupplementANovelMatchingFormulation.pdf
15. Krall, E., Higgins, M., O'Neill, R.P.: RTO Unit Commitment Test System. Federal Energy Regulatory Commission, Washington, D.C. (2012)
16. Lee, J., Leung, J., Margot, F.: Min-up/min-down polytopes. Discrete Optim **1**(1), 77–85 (2004)

17. Morales-España, G., Latorre, J.M., Ramos, A.: Tight and compact MILP formulation for the thermal unit commitment problem. IEEE Trans. Pow. Syst. **28**(4), 4897–4908 (2013)
18. Morales-España, G., Latorre, J.M., Ramos, A.: Tight and compact MILP formulation of start-up and shut-down ramping in unit commitment. IEEE Trans. Pow. Syst. **28**(2), 1288–1296 (2013)
19. Muckstadt, J.A., Wilson, R.C.: An application of mixed-integer programming duality to scheduling thermal generating systems. IEEE Trans. Pow. Appar. Syst. (12) (1968)
20. Nowak, M.P., Römisch, W.: Stochastic lagrangian relaxation applied to power scheduling in a hydro-thermal system under uncertainty. Ann. Oper. Res. **100**(1–4), 251–272 (2000)
21. Ostrowski, J., Anjos, M.F., Vannelli, A.: Tight mixed integer linear programming formulations for the unit commitment problem. IEEE Trans. Pow. Syst. **27**(1), 39 (2012)
22. Ostrowski, J., Anjos, M.F., Vannelli, A.: Modified orbital branching for structured symmetry with an application to unit commitment. Math. Program. **150**(1), 99–129 (2015)
23. Pan, K., Guan, Y.: A polyhedral study of the integrated minimum-up/-down time and ramping polytope (2016). arXiv:1604.02184
24. PJM: PJM—ancillary services (2016). http://pjm.com/markets-and-operations/ancillary-services.aspx. Accessed 01 July 2016
25. PJM: PJM—system operations (2016). http://www.pjm.com/markets-and-operations/ops-analysis.aspx. Accessed 01 July 2016
26. Pochet, Y., Wolsey, L.A.: Production Planning by Mixed Integer Programming. Springer, Berlin (2006)
27. Rajan, D., Takriti, S.: Minimum up/down polytopes of the unit commitment problem with start-up costs. IBM Research Report (2005). RC23628 (W0506-050)
28. Ramanan, P., Yildirim, M., Chow, E., Gebraeel, N.: Asynchronous decentralized framework for unit commitment in power systems. Procedia Comput. Sci. **108**, 665–674 (2017)
29. Silbernagl, M., Huber, M., Brandenberg, R.: Improving accuracy and efficiency of start-up cost formulations in MIP unit commitment by modeling power plant temperatures. IEEE Trans. Pow. Syst. **31**(4), 2578–2586 (2016)
30. Simoglou, C.K., Biskas, P.N., Bakirtzis, A.G.: Optimal self-scheduling of a thermal producer in short-term electricity markets by MILP. IEEE Trans. Pow. Syst. **25**(4), 1965–1977 (2010)
31. Takriti, S., Birge, J., Long, E.: A stochastic model for the unit commitment problem. IEEE Trans. Pow. Syst. **11**(3), 1497–1508 (1996)
32. Wilcoxon, F.: Individual comparisons by ranking methods. Biom. Bullet. **1**(6), 80–83 (1945)
33. Wolsey, L.A.: Integer Programming. Wiley, Hoboken (1998)
34. Wood, A.J., Wollenberg, B.F., Sheblé, G.B.: Power Generation, Operation and Control. Wiley, Hoboken (2013)

Affiliations

Bernard Knueven[1] · James Ostrowski[2] · Jean-Paul Watson[3]

✉ Bernard Knueven
bknueve@sandia.gov

James Ostrowski
jostrows@utk.edu

Jean-Paul Watson
jeanpaulwatson@llnl.gov

1 Sandia National Laboratories, Albuquerque, NM 87185, USA

2 University of Tennessee, Knoxville, TN 37996, USA

3 Lawrence Livermore National Laboratory, Livermore, CA 94550, USA