# An Optimization Framework for Solving Integrated Planning and Scheduling Problems for Dense Energy Carriers

**R. Cory Allen** [*,**] **Stefanos G. Baratsas** [*,**]
**Rahul Kakodkar** [*,**] **Styliani Avraamidou** [**]
**Joseph B. Powell** [***] **Clara F. Heuberger** [****]
**C. Doga Demirhan** [***] **Efstratios N. Pistikopoulos** [*,**]

[*] *Artie McFerrin Department of Chemical Engineering, Texas A&M University, College Station, TX, USA*
[**] *Texas A&M Energy Institute, Texas A&M University, College Station, TX, USA*
[***] *Shell Technology Center, Royal Dutch Shell, Houston, TX, USA*
[****] *Shell Technology Center, Royal Dutch Shell, Amsterdam, NLD*
*Email address: stratos@tamu.edu (Efstratios N. Pistikopoulos)*

**Abstract:** Due to rapid population growth, the global demand of energy and water resources are accelerating. Simultaneously, there is a global trend to transition to renewable energy systems. Recently, researchers have begun to investigate how green dense energy carriers (DECs) can be apart of this transition. In this work, we present an optimization framework for solving large-scale supply chain problems and we apply it to explore the economic and environmental impacts of DECs. Specifically, we look at utilizing DECs to transport renewable energy produced in areas with high solar and wind potentials to regions with low renewable potential. To reduce the computational burden of the large-scale optimization problem, we have developed a greedy randomized adaptive search procedure (GRASP). The GRASP leverages the linear programming (LP) relaxation of the problem to generate feasible solutions. We have found that the GRASP is able to reduce the computational time by approximately two orders of magnitude as compared to a commercial grade mixed-integer linear programming (MILP) solver ran out-of-the-box on large-scale instances.

*Keywords:* Scheduling and Optimization; Energy Processes; Industrial Applications

## 1. INTRODUCTION

Energy and water are vital for sustained life on earth. The demands for these two resources have amplified due to rapid population growth, urbanization, and improvements in standards of living. According to the International Energy Agency and the United Nations, the demand for energy and water will increase 30% by 2050 and 50% by 2070 IRENA (2018); UN-Water and UNESCO (2020). Tangentially, 90% of global power generation is currently categorized as water intensive. Furthermore, a large amount of energy is required to treat and transport water to consumers. To meet the growing demands of both energy and water, future infrastructure planning decisions should explicitly consider implicit intrinsic nexus connections Allen et al. (2019).

Currently, fossil fuels make up the lion's share of feedstocks for energy generating systems. However, due to their harmful effects on the environment, there has been a push towards green energy generators such as solar and wind farms. Unfortunately, the dispatchability of solar and wind farms is subject to geographical factors, temporal fluctuations, and stochastic intermittencies. While electric power storage, namely, rechargeable batteries, underground compressed air energy storage, and pumped-storage hydro-power can address some of these challenges, their performance is geographically dependent and/or still capital intensive. An alternative approach would be to utilize water and energy produced from solar and/or wind farms to manufacture green dense energy carriers (DECs). The DECs can then be stored in pressure vessels and/or storage tanks so as to be converted into green electrical power at a later time Demirhan et al. (2020, 2021); Palys and Daoutidis (2020).

In this work, we propose a mixed-integer linear programming (MILP) formulation for solving large-scale industrial supply chain problems with DECs. The mathematical modeling formulation is modeled as an integrated infrastructure planning and operational scheduling problem. To address the known computational difficulties that come with solving these types of problems, see Pochet and Wolsey (2006), we have developed a greedy randomized adaptive search procedure (GRASP) that we integrate into a commercial grade MILP solver Feo and Resende (1995); Oliveira et al. (2004). We demonstrate the effectiveness of the framework wherein we examine producing DECs in a region with high solar and wind potentials to partially meet the energy demand of a region with low potentials.

We utilize an augmented pareto front to highlight the intrinsic energy-water nexus (EW-N) connections that exists within green DECS.

The outline of this article is as follows: in Section 2 we state the problem; in Section 3 we present the mathematical programming formulation and the GRASP; in Section 4 we highlight and discuss the results; and in Section 5 we provide some closing remarks.

## 2. PROBLEM STATEMENT

Consider a central planner who is trying to develop a supply chain for renewable energy systems from the ground up. The central planner would like to generate power via solar and wind farms due to their low greenhouse gas emissions; however, the central planner realizes both generators suffer from seasonal variations, hourly fluctuations, asymmetric intermittencies, and geographical variabilities. Consequently, the central planner would like to incorporate energy storage devices to ensure that the systems energy demands are met on an hourly basis, while simultaneously maintaining economic viability of the supply chain.

The central planner wants to leverage the high solar and wind potentials in one region of the supply chain to partially fulfill the energy demands of another region in supply chain with low renewable potentials by utilizing DECs. Specifically, the central planner would like to produce hydrogen and ammonia solely in Amarillo, TX, (Texas) via electrolysis and the Haber-Bosch process respectively, both of which are powered by solar and/or wind farms. Once the DECs are produced, they are stored in Texas until they are transported to New York City, NY (New York). The DECs can depart Texas once per day and they arrive in New York 48 hours later. In New York, renewable power can be produced locally by solar and/or wind farms and can be stored in lithium-ion batteries. Once the DECs arrive in New York, they can be stored or utilized immediately for power generation, in conjunction with locally generated renewable power, to partially meet the areas energy demands.

The parameters, cost functions, material availabilities, material demands, etc., utilized in this problem are taken from a case study in the literature, see Demirhan et al. (2020), except for the cost parameters for solar farms, wind farms, and lithium-ion batteries, which have been updated to more recent values NREL (2020).

## 3. SOLUTION FRAMEWORK

In this section, the framework for solving large-scale infrastructure planning and scheduling problems with DECs is given and includes: (i) a generalized MILP formulation; (ii) a problem specific GRASP; and (iii) the procedure to integrate the GRASP into a MILP optimization solver.

### *3.1 Mathematical Programming Model*

The generic MILP of the framework is given by Eq. (1).

$$\begin{aligned} \min \quad & J_1 + J_2 + J_3 + J_4 + J_5 \\ \text{s.t.} \quad & \text{Eqs. } (2) - (11) \end{aligned} \tag{1}$$

The constraints are given by Eqs. (2) – (11), and the objective function terms are given by Eqs. (12) – (16).

*Sets* The sets and subsets utilized are as follows:

| | |
|---|---|
| $\mathcal{L}$ | locations, $\{l_1, l_2, \ldots, l_{\lvert\mathcal{L}\rvert}\}$ |
| $\mathcal{M}$ | materials, $\{m_1, m_2, \ldots, m_{\lvert\mathcal{M}\rvert}\}$ |
| $\mathcal{O}$ | operating modes, $\{o_1, o_2, \ldots, o_{\lvert\mathcal{O}\rvert}\}$ |
| $\mathcal{P}$ | processes, $\{p_1, p_2, \ldots, p_{\lvert\mathcal{P}\rvert}\}$ |
| $\mathcal{S}$ | material storage units, $\{s_1, s_2, \ldots, s_{\lvert\mathcal{S}\rvert}\}$ |
| $\mathcal{T}$ | time horizon, $\{t_1, t_2, \ldots, t_{\lvert\mathcal{T}\rvert}\}$ |
| $\mathcal{W}$ | representative weeks, $\{w_1, w_2, \ldots, w_{\lvert\mathcal{W}\rvert}\}$ |
| $\mathcal{E}(\cdot)$ | tuples that represent from which locations material, $m \in \mathcal{M}$, is transported from and to as well as the modes that a process and storage unit, $q \in \mathcal{P} \cup \mathcal{S}$, can transfer from and to |
| $\mathcal{L}(l,m)^-$ | locations where material, $\bar{m} \in \mathcal{M}$, can be routed from, $\{\bar{l} \in \mathcal{L} \mid (\bar{l}, l) \in \mathcal{E}(m)\}$ |
| $\mathcal{L}(l,m)^+$ | locations where material, $\bar{m} \in \mathcal{M}$, can be routed to, $\{\bar{l} \in \mathcal{L} \mid (l, \bar{l}) \in \mathcal{E}(m)\}$ |
| $\mathcal{O}(q)$ | operating modes, $o \in \mathcal{O}$, that a component, $q \in \mathcal{P} \cup \mathcal{S}$, can operate in |
| $\mathcal{O}(o,q)^-$ | operating modes, $\bar{o} \in \mathcal{O}$, that a component, $q \in \mathcal{P} \cup \mathcal{S}$, can previously operate in, $\{\bar{o} \in \mathcal{O} \mid (\bar{o}, o) \in \mathcal{E}(q)\}$ |
| $\mathcal{O}(o,q)^+$ | operating modes, $\bar{o} \in \mathcal{O}$, that a component, $q \in \mathcal{P} \cup \mathcal{S}$, can subsequently operate in, $\{\bar{o} \in \mathcal{O} \mid (o, \bar{o}) \in \mathcal{E}(q)\}$ |
| $\mathcal{P}(l)$ | processing units, $p \in \mathcal{P}$, that are located in a location, $l \in \mathcal{L}$ |
| $\mathcal{S}(l,m)$ | storage units, $s \in \mathcal{S}$, that are located in a location, $l \in \mathcal{L}$, that store a particular material, $m \in \mathcal{M}$ |
| $\mathcal{T}(w)$ | time periods, $t \in \mathcal{T}$, corresponding to a representative week, $w \in \mathcal{D}$ |

*Parameters* The parameters utilized are as follows:

| | |
|---|---|
| $\beta^t_{m,l}$ | nominal demand of material, $m \in \mathcal{M}$, that must be met at a location, $l \in \mathcal{L}$, during a time period, $t \in \mathcal{T}$ |
| $\gamma_{m,l,\bar{l}}$ | maximum amount of material, $m \in \mathcal{M}$, that can be transported from one location, $l \in \mathcal{L}$, in the supply chain to another, $\bar{l} \in \mathcal{L}(m,l)^+$ |
| $\dot{\gamma}_{m,l,\bar{l}}$ | fractional amount of material, $m \in \mathcal{M}$, lost in transport between locations, $l$, and, $\bar{l}$, where $(l, \bar{l}) \in \mathcal{E}(m)$ |
| $\zeta^-_{q,o}$ | minimum percentage of the nameplate capacity that a process or storage unit, $q \in \mathcal{P} \cup \mathcal{S}$, can operate at while in a given mode, $o \in \mathcal{O}(q)$ |
| $\zeta^+_{q,o}$ | maximum percentage of the nameplate capacity that a process or storage unit, $q \in \mathcal{P} \cup \mathcal{S}$, can operate at while in a given mode, $o \in \mathcal{O}(q)$ |
| $\eta^t_{p,m}$ | conversion factor of a material, $m \in \mathcal{M}$, for a process, $p \in \mathcal{P}$, in a time period, $t \in \mathcal{T}$ |
| $\rho^t_{m,l}$ | maximum amount of material, $m \in \mathcal{M}$, that can be purchased at a location, $l \in \mathcal{L}$, during a time period, $t \in \mathcal{T}$ |

$\dot{\sigma}_s$ fractional amount of material lost during a time period for a storage unit, $s \in \mathcal{S}$

$\tau_{m,l,\bar{l}}$ time periods required to transport material, $m \in \mathcal{M}$, between locations, $l$, and, $\bar{l}$, where $(l, \bar{l}) \in \mathcal{E}(m)$

$\tau_t^-$ time period prior to a time period, $t \in \mathcal{T}$

$\tau_t^+$ time period subsequent to a time period, $t \in \mathcal{T}$

$\omega_t$ weight assigned to a time period, $t \in \mathcal{T}$

*Cost Functions* The cost functions utilized are as follows:

$\Gamma(m, l, \bar{l}, \cdot)$ piecewise linear function that approximates the cost to transport material, $m \in \mathcal{M}$, from one location, $l \in \mathcal{L}$, to another, $\bar{l} \in \mathcal{L}(l, m)^+$

$Z(q, o, \bar{o}, \cdot)$ linear function that approximates the cost to transition a component, $q \in \mathcal{P} \cup \mathcal{S}$, operating in mode, $o \in \mathcal{O}(q)$, to operating mode, $\bar{o} \in \mathcal{O}(o, q)^+$

$N(q, \cdot)$ piecewise linear function that approximates the cost of constructing a component, $q \in \mathcal{P} \cup \mathcal{S}$

$P(m, l, \cdot)$ piecewise linear function that approximates the cost to purchase material, $m \in \mathcal{M}$, at a location, $l \in \mathcal{L}$

$Y(q, o, \cdot)$ piecewise linear function that approximates the cost to operate a component, $q \in \mathcal{P} \cup \mathcal{S}$, operating in a mode, $o \in \mathcal{O}(q)$

*Binary Variables* The binary variables utilized are as follows:

$\bar{y}_{q,o}^t$ 1 if a component, $q \in \mathcal{P} \cup \mathcal{S}$, is operating in mode, $o \in \mathcal{O}$, during a time period, $t \in \mathcal{T}$; otherwise, 0

$\bar{z}_{q,o,\bar{o}}^t$ 1 if a component, $q \in \mathcal{P} \cup \mathcal{S}$, is operating in mode, $\bar{o} \in \mathcal{O}$, during a time period, $t \in \mathcal{T}$ and in mode, $o \in \mathcal{O}$, during the previous time period, where $(o, \bar{o}) \in \mathcal{E}(p)$; otherwise, 0

*Continuous Variables* The continuous variables utilized are as follows:

$g_{m,l,\bar{l}}^t$ amount of material, $m \in \mathcal{M}$, that is transported between locations, $(l, \bar{l}) \in \mathcal{E}(m)$, in time period, $t \in \mathcal{T}$

$p_{m,l}^t$ amount of material, $m \in \mathcal{M}$, purchased at a location, $l \in \mathcal{L}$, in a time period, $t \in \mathcal{T}$

$s_s^t$ storage capacity of a storage unit, $s \in \mathcal{S}$, in time period, $t \in \mathcal{T}$

$v_q$ nameplate capacity of a component, $q \in \mathcal{P} \cup \mathcal{S}$

$x_p^t$ production rate of a process unit, $p \in \mathcal{P}$, during a time period, $t \in \mathcal{T}$

$y_{q,o}^t$ production rate or capacity of a component, $q \in \mathcal{P} \cup \mathcal{S}$, in a mode, $o \in \mathcal{O}(q)$, during a time period, $t \in \mathcal{T}$

*Material Balance Constraints* Equation (2) is a material balance constraint for each location, $l \in \mathcal{L}$, and material, $m \in \mathcal{M}$, in the supply supply chain. It is assumed that the cardinality of $\mathcal{S}(\cdot, \cdot)$ is at most one. In the event that there are multiple storage units that can store the same material in the same location, quasi-axillary materials can be created for each of the additional storage units.

$$\begin{aligned} &\sum_{s \in \mathcal{S}(m,l)} (1 - \dot{\sigma}_s) \cdot s_s^{\tau_t^-} \ \ldots \\ &+ \sum_{\bar{l} \in \mathcal{L}(m,l)^-} (1 - \dot{\gamma}_{m,\bar{l},l}) \cdot g_{m,\bar{l},l}^{t - \tau_{m,\bar{l},l}} + \sum_{p \in \mathcal{P}(l)} \eta_{p,m}^t \cdot x_p^t \ \ldots \\ &+ p_{m,l}^t = \beta_{m,l}^t + \sum_{\bar{l} \in \mathcal{L}(m,l)^+} g_{m,l,\bar{l}}^t + \sum_{s \in \mathcal{S}(m,l)} s_s^t \ \ldots \\ &\forall\, t \in \mathcal{T},\, m \in \mathcal{M},\, l \in \mathcal{L} \end{aligned} \tag{2}$$

*Mode Transition Constraints* Equations (3) – (5) are utilized to model the operating modes, $\mathcal{O}(q)$, the components, $q \in \mathcal{P} \cup \mathcal{S}$, can operate in and transition between. It should be noted that the operating modes, $\mathcal{O}(q)$, implicitly capture the minimum down-times, minimum run-times, and ramping-times for each component, $q \in \mathcal{P} \cup \mathcal{S}$.

$$\begin{aligned} &\sum_{\bar{o} \in \mathcal{O}(o,q)^+} \bar{z}_{q,o,\bar{o}}^{\tau_t^+} - \sum_{\bar{o} \in \mathcal{O}(o,q)^-} \bar{z}_{q,\bar{o},o}^t = 0 \ \ldots \\ &\forall\, t \in \mathcal{T},\, q \in \mathcal{P} \cup \mathcal{S},\, o \in \mathcal{O}(q) \end{aligned} \tag{3}$$

$$\sum_{\bar{o} \in \mathcal{O}(o,q)^-} z_{q,\bar{o},o}^t = \bar{y}_{q,o}^t \ \ \forall\, t \in \mathcal{T},\, q \in \mathcal{P} \cup \mathcal{S},\, o \in \mathcal{O}(q) \tag{4}$$

$$\sum_{o \in \mathcal{O}(q)} \bar{y}_{q,o}^t = 1 \ \ \forall\, t \in \mathcal{T},\, q \in \mathcal{P} \cup \mathcal{S} \tag{5}$$

*Mode Mapping Constraints* Equation (6) and Eq. (7) are utilized to project the mode based variables into the material balance constraint, Eq. (2).

$$s_s^t = \sum_{o \in \mathcal{O}(s)} y_{s,o}^t \ \ \forall\, t \in \mathcal{T},\, s \in \mathcal{S} \tag{6}$$

$$x_p^t = \sum_{o \in \mathcal{O}(p)} y_{p,o}^t \ \ \forall\, t \in \mathcal{T},\, p \in \mathcal{P} \tag{7}$$

*Mode Based Bound Constraints* Equation (8) and Eq. (9) ensure that a component, $q \in \mathcal{P} \cup \mathcal{S}$, can only operate within its lower and upper bound percentages relative to its nameplate capacity when it is operating in the corresponding mode, $o \in \mathcal{O}(q)$.

$$y_{q,o}^t \geq \zeta_{q,o}^- \cdot v_q \cdot \bar{y}_{q,o}^t \ \ \forall\, t \in \mathcal{T},\, p \in \mathcal{P} \cup \mathcal{S},\, o \in \mathcal{O}(q) \tag{8}$$

$$y_{q,o}^t \leq \zeta_{q,o}^+ \cdot v_q \cdot \bar{y}_{q,o}^t \ \ \forall\, t \in \mathcal{T},\, q \in \mathcal{P} \cup \mathcal{S},\, o \in \mathcal{O}(q) \tag{9}$$

It should be noted that Eq. (8) and Eq. (9) are linearized via standard reformulations, see Williams (2013).

*Capacity Bound Constraints* Equations (10) – (11) are utilized to ensure that the physical limits are not violated.

$$g_{m,l,\bar{l}}^t \leq \gamma_{m,l,\bar{l}} \ \ \forall\, t \in \mathcal{T},\, m \in \mathcal{M}, (l, \bar{l}) \in \mathcal{E}(m) \tag{10}$$

$$p_{m,l}^t \leq \rho_{m,l}^t \ \ \forall\, t \in \mathcal{T},\, m \in \mathcal{M},\, l \in \mathcal{L} \tag{11}$$

*Objective Functions* Piecewise linear functions are utilized to the capture non-convex cost functions in the problem and are given by Eqs. (12) – (16): Eq. (12) captures

the transportation costs; Eq. (13) captures the mode based costs; Eq. (14) captures the cost to construct processes and storage units; Eq. (15) captures the cost to purchase material; and Eq. (16) captures the cost to operate processes and storage units. For the sake of compactness and brevity we do not incorporate the associated binary variables that we utilize in the linearization process.

$$J_1 \triangleq \sum_{t\in\mathcal{T}} \sum_{m\in\mathcal{M}} \sum_{(l,\bar{l})\in\mathcal{E}(m)} \omega_t \cdot \Gamma\big(m,l,\bar{l},g^t_{m,l,\bar{l}}\big) \tag{12}$$

$$J_2 \triangleq \sum_{t\in\mathcal{T}} \sum_{q\in\mathcal{P}\cup\mathcal{S}} \sum_{(o,\bar{o})\in\mathcal{E}(q)} \omega_t \cdot Z\big(q,o,\bar{o},z^t_{p,o,\bar{o}}\big) \tag{13}$$

$$J_3 \triangleq \sum_{q\in\mathcal{P}\cup\mathcal{S}} N\big(q,v_q\big) \tag{14}$$

$$J_4 \triangleq \sum_{t\in\mathcal{T}} \sum_{m\in\mathcal{M}} \sum_{l\in\mathcal{L}} \omega_t \cdot P\big(m,l,p^t_{m,l}\big) \tag{15}$$

$$J_5 \triangleq \sum_{t\in\mathcal{T}} \sum_{q\in\mathcal{P}\cup\mathcal{S}} \sum_{o\in\mathcal{O}} \omega_t \cdot Y\big(m,l,y^t_{q,o}\big) \tag{16}$$

*3.2 GRASP*

The GRASP for generating feasible solutions to the MILP is given by Alg. 3.1 Feo and Resende (1995). Initially the GRASP creates an empty set, *solutionPool*, to store feasible solutions. The GRASP then repeats the following procedures: (i) first it generates a solution to a subset of the binary variables in the MILP via a greedy randomized construction heuristic, `Construction(·,·)`; (ii) then it ensures that the remaining binary variables in the MILP are feasible via a repairing function, `RepairSolution(·)`; (iii) then it improves upon the feasible solution via a local search function, `LocalSearch(·)`; and (iv) finally it appends the feasible solution to the solution pool and indexes the *seed* by one. The GRASP repeats these procedures until it has reached the maximum number of iterations, *maxIterations*, and then determines its best solution, via `GetBestSolution(·)`, and returns it.

**Algorithm 3.1:** GRASP

**input :** *maxIterations* – maximum iterations
*seed* – initial seed for the psuedo-random number generator
*LPsolution* – solution to the LP relaxation
**output:** *bestSolution* – best solution found
**Function** `GRASP(`*maxIterations, seed, LPsolution*`)`:
  *solutionPool* ← ∅
  **repeat**
    *solution* ← `Construction(`*seed, LPsolution*`)`
    *solution* ← `RepairSolution(`*solution*`)`
    *solution* ← `LocalSearch(`*solution*`)`
    *solutionPool* ← *solutionPool* ∪ {*solution*}
    *seed* ← *seed* + 1
  **until** *maxIterations reached*
  *bestSolution* ← `GetBestSolution(`*solutionPool*`)`
  **return** *bestSolution*
**End Function**

*Greedy Randomized Construction* The greedy randomized construction heuristic, given by Alg. 3.2, builds a solution to a subset of the binary variables in the MILP, $\{\bar{y}^t_{q,o}\}^{t\in\mathcal{T}}_{q\in\mathcal{P}\cup\mathcal{S},\, o\in\mathcal{O}(q)}$, by exploiting the LP relaxation of the problem. Initially, the heuristic creates an empty set, *solution*. Then the heuristic repeats the following procedures until a solution is constructed: (i) first it builds a restricted candidate list, *RCL*, of operating modes, via `BuildRCL(·,·)`; and (ii) then it selects an operating mode at random based upon the *seed* from the restricted candidate list and constructs its corresponding solution, via `BuildRandomSolution(·,·)`; and (iii) finally it appends the new partial solution from the restricted candidate list to the set of solutions, *solution*.

**Algorithm 3.2:** Greedy Randomized Construction

**input :** *seed* – seed for the psuedo-random number generator
*LPsolution* – solution to the LP relaxation
**output:** *solution* – feasible solution
**Function** `Construction(`*seed, LPsolution*`)`:
  *solution* ← ∅
  **repeat**
    *RCL* ← `BuildRCL(`*solution, LPsolution*`)`
    *s* ← `BuildRandomSolution(`*seed, RCL*`)`
    *solution* ← *solution* ∪ {*s*}
  **until** *solution is constructed*
  **return** *solution*
**End Function**

The function, `BuildRCL(·,·)`, builds a set of operating modes for the restricted candidate list via a three-stage process. In the first stage, a set covering problem is solved to find a minimum quasi-feasible operating mode, $\texttt{SetCover}(t,q) \triangleq \arg\min\{\sum_{\bar{o}\in\mathcal{O}(q)} y^{t*}_{q,\bar{o}} - \delta^-_{q,o}\cdot v^*_q \mid \sum_{\bar{o}\in\mathcal{O}(q)} y^{t*}_{q,\bar{o}} - \delta^-_{q,o}\cdot v^*_q \geq 0,\, v^*_q > 0,\, o\in\mathcal{O}(q)\}$ – where $v^*_q$ and $y^{t*}_{q,\bar{o}}$ indicate the solution of the linear programming relaxation to $v_q$ and $y^t_{q,\bar{o}}$ respectively. On the off chance, $\texttt{SetCover}(t,q)$, is infeasible, the function returns the lowest operating mode of the component, $q\in\mathcal{P}\cup\mathcal{S}$. In the second stage, the function creates an order set, $\mathcal{O}(t,q) \triangleq \{\texttt{SetCover}(t,q),\cdots,\arg\max\{\delta^+_{q,o} \mid o\in\mathcal{O}(q)\}\} \subseteq \mathcal{O}(q)$, for the components, $q\in\mathcal{P}\cup\mathcal{S}$, ranging from the lowest to the highest operating modes. In the final stage, the function returns the first $\lceil\alpha\cdot|\mathcal{O}(t,q)|\rceil$ elements of the order set, where $\alpha\in(0,1]$. The parameter, $\alpha$, is utilized to control the "greediness" and "randomness" of the procedure. For instance if the parameter, $\alpha$, is equal to one the procedure returns a random solution, while on the other hand if the parameter, $\alpha$, is equal to $|\mathcal{O}(t,q)|^{-1}$ the procedure returns a greedy solution.

*Solution Repair* The function, `RepairSolution(·)`, ensures that the remaining binary variables in the MILP, $\{\bar{z}^t_{q,o,\bar{o}}\}^{t\in\mathcal{T}}_{q\in\mathcal{P}\cup\mathcal{S},\, o\in\mathcal{O}(q),\bar{o}\in\mathcal{O}(o,q)^+}$, are feasible. Initially, this function checks if Eqs. (3) – (5) are violated. If the constraints are not violated the solution is feasible. If one or more constraints are violated the operating mode for the corresponding component, $q\in\mathcal{P}\cup\mathcal{S}$, and time period, $t\in\mathcal{T}$, is increased by one iteratively until feasibility is achieved for the problem.

*Local Search* Once a feasible solution to the binary variables in the MILP is found, the solution is passed to a local search function, `LocalSearch(·)`. In the local search

function, neighboring solutions are explored by solving a integer program that is bounded by Eqs. (3) – (5) and the operating modes, $o \in \mathcal{O}(t,q)$, found in the construction heuristic. After the neighborhood is fully explored, the MILP is resolved with all the binary variables fixed to their locally optimal solutions to determine the locally optimal values of the remaining variables.

### *3.3 Integration of the GRASP and the MILP Solver*

The unification of the GRASP and MILP solver is accomplished by utilizing a three stage procedure. In the first stage, the LP relaxation of the relevant variables are accessed from the MILP solver. In the second stage, their respective solutions are fed to the GRASP and the GRASP is executed. In the final stage, the solution produced by the GRASP is fed back to MILP solver and utilized to generate a upper bound to the problem.

## 4. RESULTS AND DISCUSSION

The mathematical programming model, Eq. (1), was implemented in Python and solved utilizing Gurobi V9.0 Gurobi Optimization, LLC (2020). The GRASP was written in Python and conjoined with mathematical programming model via Gurobi's callback features. The computational experiments were performed on a machine with a Intel Xeon W-10885M Processor and 64 GB of RAM.

### *4.1 Part I: Minimize the LCOE*

The infrastructure required to minimize the levelized cost of electricity (LCOE) for system given the nominal process parameters, cost functions, material availabilities, material demands, etc. is given in Table 1. As aforementioned, DECs cannot be produced in New York; therefore, their name plate capacities are given by "NaN".

Table 1. Name plate capacities of the infrastructure in Texas and New York

| | Location | |
|---|---|---|
| Process | New York | Texas |
| Solar Farm [MW] | 1591.7 | 1301.6 |
| Wind Farm [MW] | 705.3 | 542.3 |
| Electrolyzer [MW] | NaN | 873.8 |
| Air Separation Unit [kg/hr] | NaN | 0.0 |
| Haber-Bosch [kg/hr] | NaN | 0.0 |
| Direct Air Capture [kg/hr] | NaN | 0.0 |
| $H_2$ Compression [kg/hr] | 0.0 | 0.0 |
| $H_2$ Liquefaction [kg/hr] | 0.0 | 14,840.4 |
| Lithium-Ion Batteries [MW] | 0.0 | 0.0 |
| $H_2$ Fuel Cell [MW] | 1036.2 | 0.0 |
| $NH_3$ Gas Turbine [MW] | 0.0 | 0.0 |

The unit commitments for the components in New York during one of the representative weeks is given by Fig. 1. It should be noted, that between 24-48 [hr] the storage level of cryogenic $H_2$ does not go to zero. This could be attributed to the fact that the solution has a non-zero mixed integer programming (MIP) gap and/or that the solar and/or wind potentials subside in those corresponding hours in Texas.

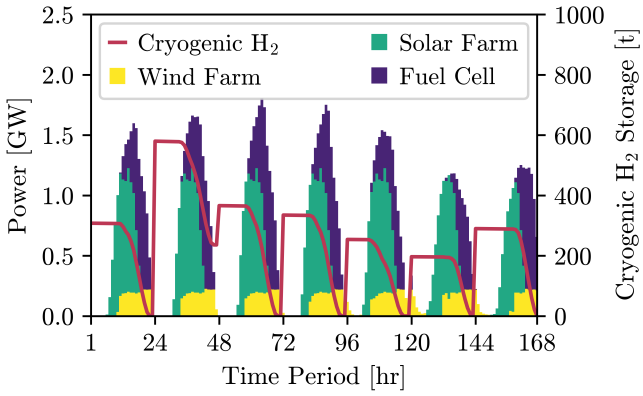


Fig. 1. Energy production and material storage schedule in New York for a representative week

The sensitivity of the LCOE of the system and water consumption in Texas with respect to the penetration of DECs in New York's green energy portfolio is illustrated in an augmented pareto front, Fig. 2. From inspection of Fig. 2, the optimal penetration of DECs in New York's green energy portfolio is approximately 28 [%].

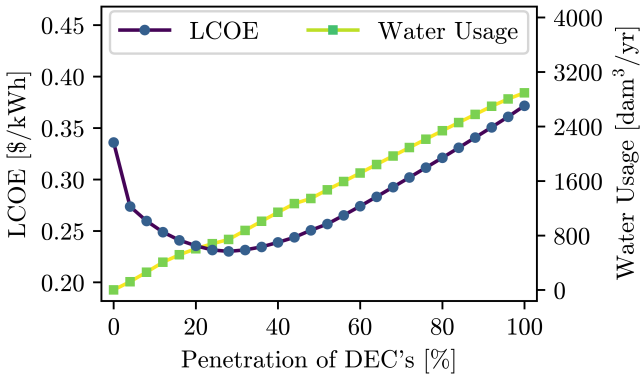


Fig. 2. Relationship between the LCOE of the system and water consumption in Texas given a fixed penetration of DECs in New York

### *4.2 Part II: Computational Experiments*

In this subsection, the proposed optimization framework is pitted against Gurobi's default solver through the use of a set of computational experiments in which the number of representative weeks are varied – in these experiments $\alpha$ is set to $|\mathcal{O}(\cdot,\cdot)|^{-1}$ and `maxIterations` is set to 1. The size of the instances can be seen in Table 2.

Table 2. Size of the MILP

| | Number of | | |
|---|---|---|---|
| Representative Weeks | Binary Variables | Constraints | Continuous Variables |
| 1 | 41,472 | 108,858 | 41,640 |
| 2 | 82,800 | 217,386 | 83,136 |
| 3 | 124,128 | 325,914 | 124,632 |
| 4 | 165,456 | 434,442 | 166,128 |
| 5 | 206,784 | 542,970 | 207,624 |
| 6 | 248,112 | 651,498 | 249,120 |
| 7 | 289,440 | 760,026 | 290,616 |

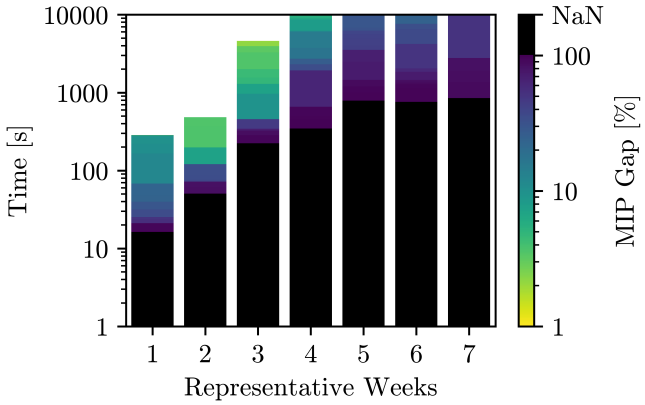


Fig. 3. Performance of Gurobi's default solver

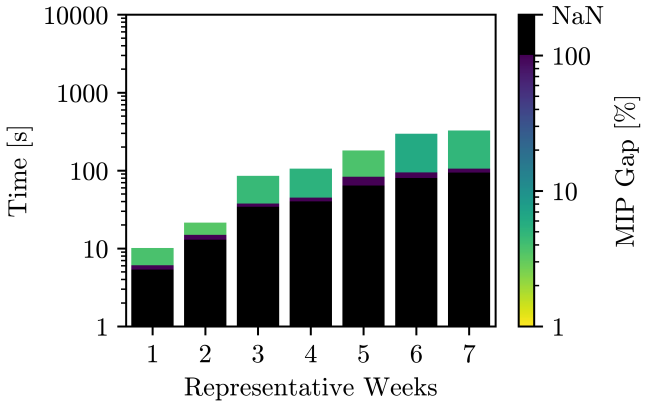


Fig. 4. Performance of the proposed optimization framework utilizing Gurobi's interior point method to solve the root node of the branch-and-bound tree

Figure 3 and Fig. 4 illustrate the MIP gap as function of time and number of representative weeks for Gurobi's default solver and the proposed optimization framework respectively. In these two figures, the bar is colored "black" when the LP relaxation is being computed. It should be highlighted that the in framework: (i) Gurobi's interior point method is utilized to solve the root node; (ii) Gurobi's internal heuristic is turned off; and (iii) Gurobi's MIP Gap tolerance is set to 2 [%]. From inspection, it is apparent that the proposed optimization framework is approximately two orders of magnitude faster than Gurobi's default solver. This is primarily due to the strength of the LP relaxation and the ability of the GRASP to generate feasible solutions explicitly for the binary variables associated with operational decisions and implicitly for the binary variables associated with design decisions. As a result, the user has the capability to conduct a significant more number of scenario and sensitivity analyses within a much shorter period of time (approximately 100 times faster) than they would utilizing Gurobi's default solver.

## 5. CONCLUSION

We have presented a generalized optimization framework that includes a problem specific GRASP for solving large-scale supply chain problems with DECs. We utilize an integrated infrastructure planning and operational scheduling approach for creating the mathematical programming formulation of the framework. We have illustrated the superiority of the optimization framework for solving integrated infrastructure planning and operational scheduling problems with DECs, while simultaneously achieving a substantial reduction in the LCOE for green energy systems. The optimization framework can produce solutions approximately two orders of magnitude faster than Gurobi's default solver; thereby, allowing the user to explore many more scenarios in the same amount of time. The performance of the framework can be theoretically improved by utilizing a Lagrangian decomposition method by decreasing the amount of time required to solve the LP relaxation of the problem.

## 6. ACKNOWLEDGEMENTS

The authors thank Royal Dutch Shell, the National Science Foundation (Grant Numbered 1739977), and the Texas A&M Energy Institute for their financial support.

## REFERENCES

Allen, R.C., Nie, Y., Avraamidou, S., and Pistikopoulos, E.N. (2019). Infrastructure planning and operational scheduling for power generating systems: An energy-water nexus approach. In *Computer Aided Chemical Engineering*, volume 47, 233–238. Elsevier.

Demirhan, C.D., Tso, W.W., Powell, J.B., Heuberger, C.F., and Pistikopoulos, E.N. (2020). A multiscale energy systems engineering approach for renewable power generation and storage optimization. *Industrial & Engineering Chemistry Research*, 59(16), 7706–7721.

Demirhan, C.D., Tso, W.W., Powell, J.B., and Pistikopoulos, E.N. (2021). A multi-scale energy systems engineering approach towards integrated multi-product network optimization. *Applied Energy*, 281.

Feo, T.A. and Resende, M.G. (1995). Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2), 109–133.

IRENA, G.E.T. (2018). A roadmap to 2050. *International Renewable Energy Agency, Abu Dhabi*.

Oliveira, C.A., Pardalos, P.M., and Resende, M.G. (2004). Grasp with path-relinking for the quadratic assignment problem. In *International Workshop on Experimental and Efficient Algorithms*, 356–368. Springer.

Palys, M.J. and Daoutidis, P. (2020). Using hydrogen and ammonia for renewable energy storage: A geographically comprehensive techno-economic study. *Computers & Chemical Engineering*, 106785.

Pochet, Y. and Wolsey, L.A. (2006). *Production planning by mixed integer programming*. Springer Science & Business Media.

Gurobi Optimization, LLC (2020). Gurobi optimizer reference manual. URL `http://www.gurobi.com`.

NREL (2020). 2020 annual technology baseline.

UN-Water and UNESCO (2020). United nations world water development report 2020: Water and climate change.

Williams, H.P. (2013). *Model building in mathematical programming*. John Wiley & Sons.