Department of Computer Science
George Mason University   Technical Reports

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/   703-993-1530

A Decision Guidance System for Optimal Operation ofHybrid Power Desalination Service Network

Bedor Alyahya
balyahya@gmu.edu

Alexander Brodsky
brodsky@gmu.edu

Technical Report GMU-CS-TR-2020-3

Abstract

While modeling and optimization of desalination systems' operation have been extensively studied, currentapproaches are hard-wired to specific designs and performance metrics, without the flexibility to reuse orextend these models. Bridging this gap, reported in this paper is the development of a formal analytic modeland a decision guidance system for desalination service networks that can be applied to a broad range ofdesalination designs and architectures. The model and the system are based on an extensible repository ofatomic component models, initially including models for pumps, renewable energy sources, water and powerstorage, and reverse osmosis units. An experimental study is conducted to demonstrate the flexibility of themodel and system, and its scalability to support realistic size problems.

1 Introduction

The world is facing a great challenge in satisfying increasing water demand. To deal with the challenge, the United Nations has formed a plan in its 2030 Agenda to ensure availability and sustainable management of water for all (UN, 2015). In an attempt to align with the plan requirements, many solutions have been proposed to meet the increasing demand for fresh water while minimizing its negative environmental impacts.

The current trend toward solving the shortage in freshwater is by combining renewable energy sources with desalination systems. However, renewable sources of energy are intermittent in their supply of electric power. The uncertainty in both demand and, especially, supply of power creates a new challenge in operating interconnected system components to maximize financial, social and environmental benefits.

There has been work on optimizing desalination systems, from a desalination unit (Ahmed et al., 2019) through hybrid desalination system (Mokheimer et al., 2013) to fully integrated water and power supply chains (Al-Nory, 2019; Abdelshafy et al., 2018). Also, several studies have examined strategical challenges in combining renewable energy with desalination systems. In (Gençoğlu and Merzi, 2016), the authors focus on strategical decisions in water desalination supply chain by using mathematical modeling to optimize investment terms. Al-Nory and El-Beltagy (2015) focuses on minimizing the interruption of the power supply by selecting optimal pumped hydro storage as well as power storage (e.g., batteries.) Azhar et al. (2017) propose an efficient desalination system design combined it with renewable sources to produce water and energy. Marques et al. (2014) optimize the investment in water infrastructures by adapting a flexible water system plan using decision trees. However, these models are either (1) limited in their focus on specific units or technologies rather than optimizing the desalination system as a whole, (2) hard-wired to specific designs and objectives which limit its usability in solving other designs or optimizing other performance metrics (Mokheimer et al., 2013; Al-Nory, 2019; Al-Nory and Brodsky, 2014), or (3) focused on long-term investment in desalination system infrastructures while making simplifying assumptions on, rather than accurately modeling the operation of, the underlying infrastructures over the time horizon (Al Nory and Graves, 2013). We believe, however, that accurately modeling the resource flows, such as of power and water, within and across the system components over the given operational period can reveal unseen saving in both operation and investment.

To bridge the gap, we propose an approach that helps decision makers optimize desalination system operation over a given operational window (e.g., 72 hours) without the need to hard-wire the model to a specific desalination design, architecture or performance metrics. More specifically, the contributions of this paper are threefold. First, we develop a formal *Analytic Model* (AM) for desalination *Service Networks* (SN) that can be applied to a broad range of desalination designs and architectures.

Second, we develop an extensible repository of analytic models for desalination system components, which initially includes pumps, renewable energy sources, water and energy storage and Reverse Osmosis (RO) plants. Unlike hard-wired models, our frameworks allows extending model repository with additional components, and instantiating an arbitrary desalination service network, without any other model modifications.

Third, based on the developed model repository, we develop a Decision Guidance System (DGS) that (1) allows desalination system engineers to instantiate their specific desalination design/architecture, (2) performs iterative optimization over operational time windows for the selected objective, such as minimizing CO2-adjusted operational cost, while satisfying the system and demand constraints; and, (3) makes actionable recommendations to desalination system operators/control system on precise controls of each desalination system component for every time interval. The system is based on Decision Guidance Analytics Language (DGAL) (Brodsky and Luo, 2015) and the Decision Guidance Management System architecture (Unity DGMS) proposed in (Nachawati et al., 2017).

Finally, we conduct a preliminary experimental study to demonstrate the flexibility of the system applied to four examples of desalination design, and its scalability to handle realistic size optimization problems.

The paper is organized as follows. Section 2 illustrates the concept of desalination service network using an example. Section 3 overviews a high-level architecture of the developed Desalination Decision Guidance system and methodology. Section 4 formalizes the desalination service network model. Section 5 discusses the results of the experimental study. Finally, Section 6 presents concluding remarks and briefly outlines directions for future work.

2 Desalination Service Network by example

The purpose of the Service Network Desalination Model (SNDM) is to solve a scheduling optimization problem for different desalination system designs and performance metrics. We use a generic structure called a *service network* (SN) to allow the model to handle different designs. The SN, as described in (Brodsky et al., 2017), is a hierarchy of services that are connected together to capture the flow of commodity over the network. By using a Service Network, we can create different desalination system designs, and through the use of the desalination Analytical Model AM, we can optimize the performance metrics of these designs.

To illustrate this concept, consider an example of the service network for a hybrid energy system with Reverse Osmosis (RO) desalination plant system depicted in Fig 1. The root of the service hierarchy is the Water Desalination System. Within it there are sub-services for a Pump Station, Low Reservoir, High Reservoir, Power Sources, Power Storage. These services are connected together to produce fresh water. In the figure, each arrow indicates the flow of some resource (such as sea water, water under pressure, fresh water and power) between these services. A composite service, such as Power Sources, contains other sub-services, such as Renewable Sources and Power Grid, which are *atomic* services (i.e., do not have sub-services.) Both atomic and composite services are optional. Now, we can create many designs by extending the hierarchy with other sub-services that mimic the system we intend to represent.

In order for a system to satisfy fresh water demand for a given operational window, it should produce fresh water by setting the right amount of flows throughout the system while minimizing the production cost and the carbon emissions for a given operational window. Optimization is based, as described in Section 4, on an analytic model (AM) which computes metrics such as cost and CO2 emissions, as well as feasibility constraints, for a given operational window (e.g., of 72 hours), as a function of operational controls for each component and time interval (e.g., of 1 hour.) In the model computation, the flows and feasibility have to be aggregated bottom-up, for all intervals, by recursively calling each *composite* service its' sub-services starting with the root service until reaching the *atomic* services at the bottom of the hierarchy, as in figure 2 (step $1{:}1_a,1_c$). At that point, the AM uses the type of the *atomic* service to refer to the corresponding *atomic* AM in the library of *atomic* analytical Models (AMs) which then calculates the flows and feasibility for that *atomic* service as in figure 2 (step 1_b). Then, we repeat the same process given the output from the first step but this time we aggregate and calculate the metrics for the whole periods as well as some additional constraints (as in step 2).

3 Desalination decision Guidance System

The developed Desalination Decision Guidance System (DGS) (1) allows desalination system engineers to instantiate their specific desalination design/architecture, (2) performs iterative optimization over operational time windows for the selected objective, such as minimizing CO2-adjusted operational cost, while satisfying the system and demand constraints; and, (3) makes actionable recommendations to desalination system operators/control system on precise controls of each desalination system component for every time interval. The system is based on Decision Guidance Analytics Language (DGAL) (Brodsky and Luo, 2015) and the Decision Guidance Management System (Unity DGMS) proposed in (Nachawati et al., 2017), which in turn is based on the concept proposed in (Brodsky and Wang, 2008).

Figure 3 depicts the high-level architecture of Desalination DGs. The middle layer represent the decision guidance management system that contains a repository of reusable, modular, and composable models. Through the Graphical User Interface (GUI), the user can construct many desalination designs. Through the analytical engine, the DGMS hides from the user the complexity in dealing with external tools to perform different analytical tasks (such as optimization, learning and prediction). For example, to perform optimization for desalination service network, the analytics engines machine generates a mixed-integer linear programming (MILP) model from the simulation-like analytic model (formalized in Section 4), which was written in Python. The input required for DG optimization includes (1) an analytic model, (2) an

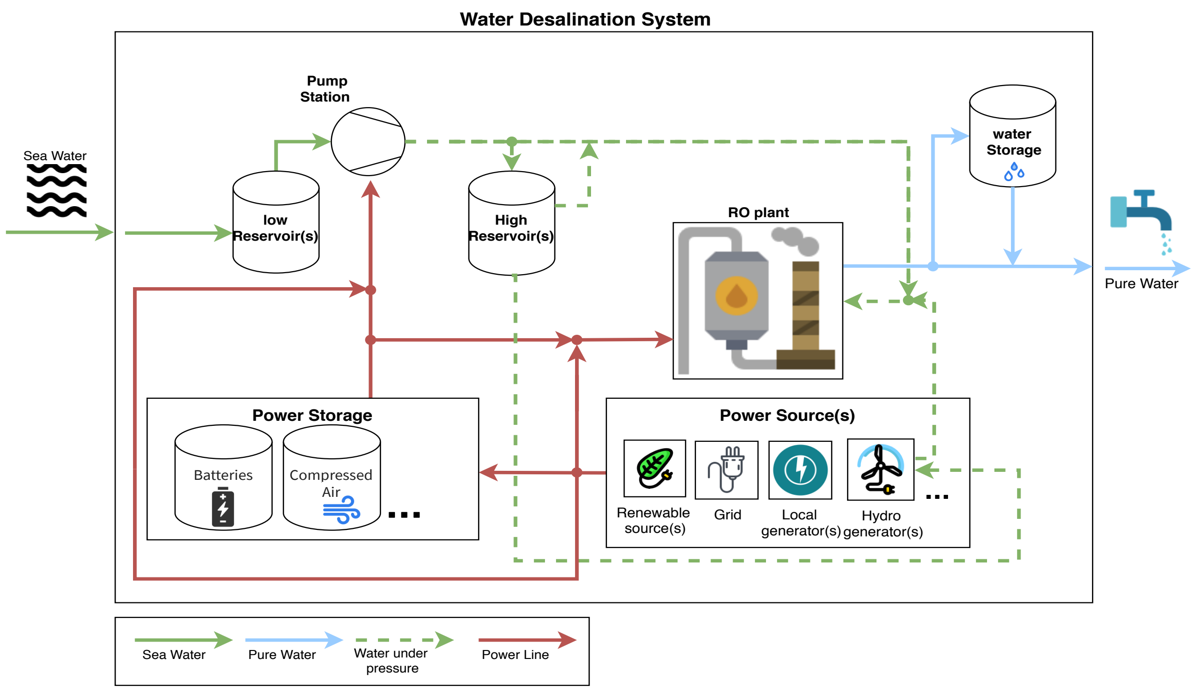

Figure 1: Desalination Service Network

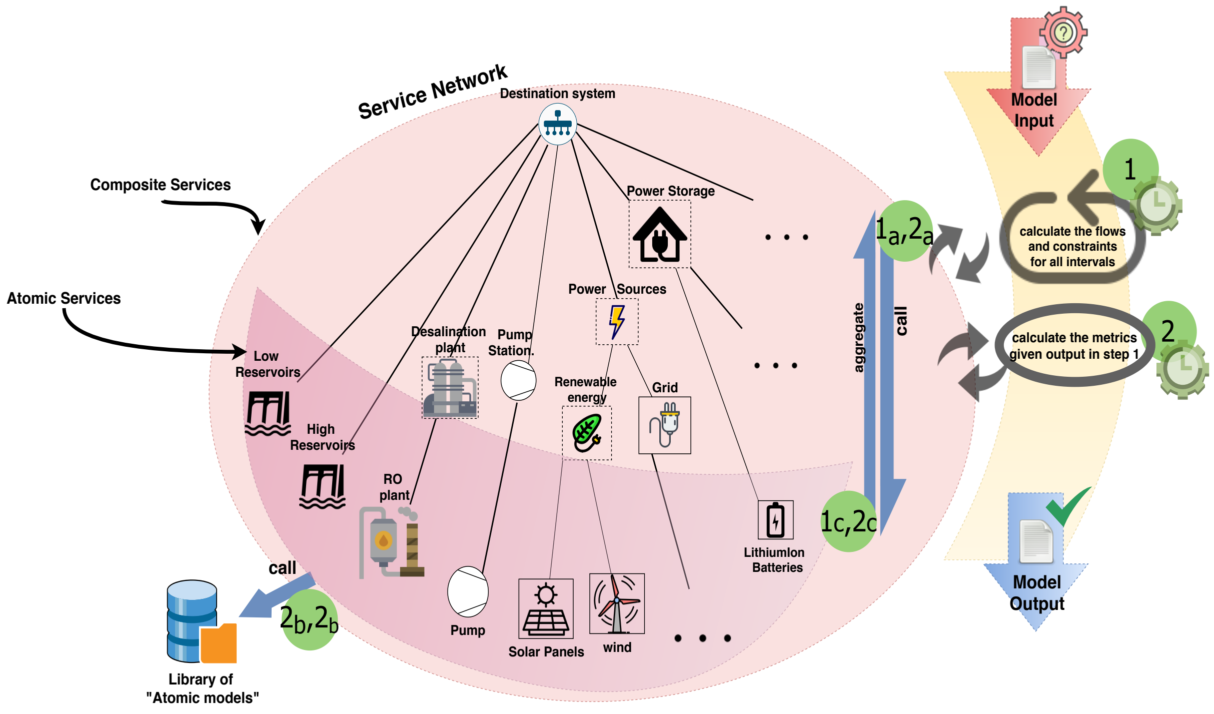

Figure 2: How the Desalination model works

analytic model input annotated to indicate which input values serve as decision variables, and (3) indication of which of the computed model metrics serves as optimization objective and constraints.

For desalination plant users, we envision the following workflow for using Desalination DGS:

1. A desalination plant engineer interacts with the DGS to create an instance of the plant's design and architecture.
2. An desalination plant engineer interacts with the DGS to instantiate additional system parameters, such as the length of operational window (e.g., 72 hours), and the frequency of re-optimization (e.g., at the top of every hour)
3. A plant operator, on a periodic basis, updates the demand for fresh water. The system performs optimization and makes actionable recommendation on precise controls of every system component, every operational interval. Some of these controls can automatically actuate the underlying system components, whereas some other will be displayed to the plant operator, who can approve and actuate plant controls.

4 Formalization of Service Network Desalination Model

4.1 High-level Optimization Problem

The desalination operational optimization is based on the analytic performance model (AM), which computes performance metrics, such as cost and carbon emissions, as well as feasibility constraints, as a function of fixed and control (decision) parameters of the desalination service network over the operational time window.

More formally, the analytic performance model AM is a function:

$$AM : IN \to OUT \tag{1}$$

where:

IN is a set of all valid inputs

OUT is a set of all valid outputs

The AM forms a valid output instance $out \in OUT$ of performance metrics, such as operational cost or waste, from a valid input $in \in IN$ of fixed and controlled operational parameters.

Then, the desalination optimization problem is:

$$\begin{aligned} \min_{in \in IN} \quad & obj(AM(in)) \\ \text{s.t.} \quad & C(AM(in)) \end{aligned} \tag{2}$$

where

- $Obj : OUT \to \mathbb{R}$ is an objective function, which gives the real objective value in $\mathbb{R}$ given a valid output instance of the AM.
- $C : OUT \to \{T, F\}$ is a constraint function C, which gives *True* or *False* given a valid output instance of the AM.

In contrast of hardwired models, we describe the objective and constraints as a function of analytical model AM output. By doing that we loosening the tightly connected model, so that same AM can be used to formulate multiple desalination optimization problems using different system designs and objective functions.

In the following, we will use the following notation for a set of *key-value* pairs:

$$\begin{aligned} m = \{&\mathbf{key_1} : value_1, \\ &\mathbf{key_2} : value_2, \\ &\dots \\ &\mathbf{key_n} : value_n\} \end{aligned} \tag{3}$$

where the *keys* are unique identifiers. Note that this set represents a mapping

$$m : \{key_1, \dots, key_n\} \longrightarrow \cup_{i=1}^{n} D_i$$

from the set of keys to union of the domains so that $m(key_i) \in D_i$ for all $i = 1, \dots, n$. We will use the notation $keys(m) = \{key_1, \dots, key_n\}$ to denote the set of all keys associated with the set m of key-value pairs.

Now using the above notations we can describe all of the components above; starting with a valid service network desalination model output instance *out* in section 4.2, followed by the input instance *in* in section 4.3, and finally, we describe the analytic model which is a function that computes an output instance from the input instance.

4.2 Service Network desalination Instance: The Model Output

A valid *SN* desalination output instance *out* is a set of *key:value* pairs:

$$\begin{aligned} \{\mathbf{config} &: \langle \text{ config parameters} \rangle \\ \mathbf{rootServiceID} &: \text{ root service id}, \\ \mathbf{services} &: \langle \text{ set of } services \rangle, \\ \mathbf{constraints} &: \text{"True"}or\text{"False"}, \\ \mathbf{metrics} &: \langle \text{ set of metrics} \rangle\} \end{aligned} \tag{4}$$

where :

- **config** value is a set of:

  $$\begin{aligned} \{&\mathbf{operationalInterval} : \text{value}, \\ &\mathbf{operationalWindow} : w\} \end{aligned}$$

  where **config** value defines the time horizon using the **operationalInterval** string that represents the unit of time in which the system operates (e.g, hour). The **operationalWindow** (w) represents the number of intervals, so that operational decisions can be made over these intervals.
- **rootServiceID** is the id of a service in *services* designated as a root service.

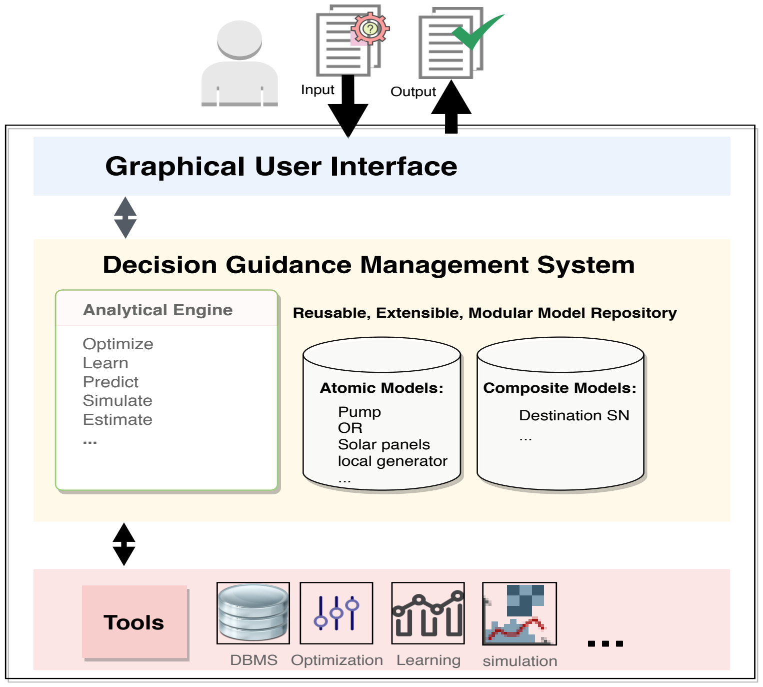

Figure 3: Desalination Guidance System Architecture

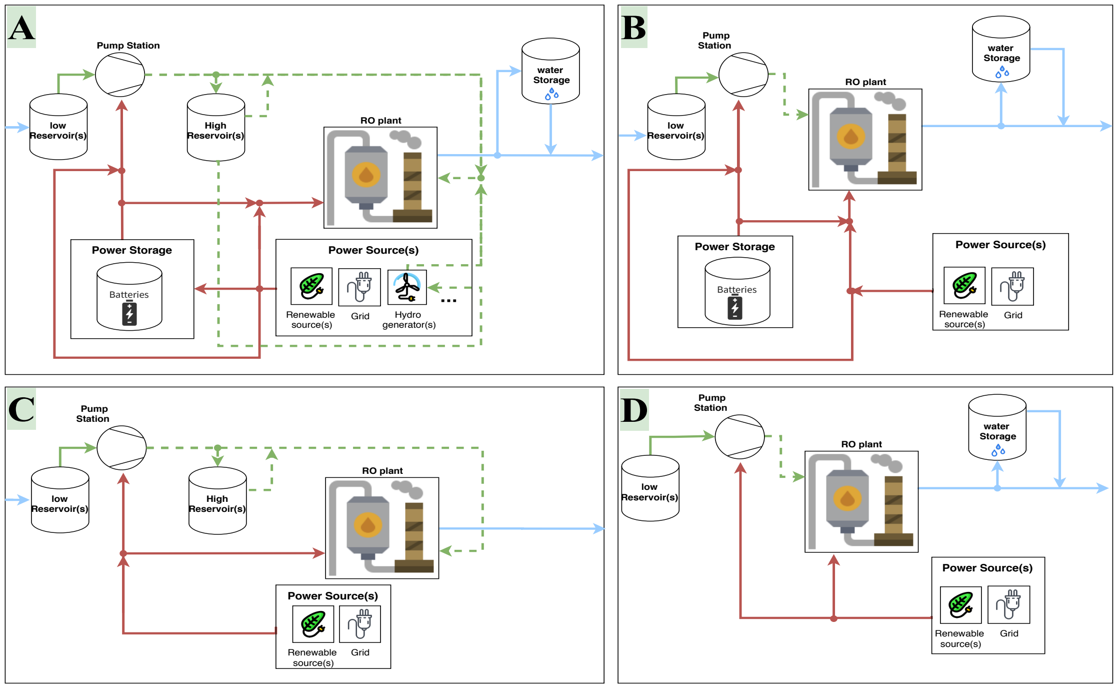

Figure 4: Four Desalination System Designs

- **Services:** is a set optional *key:value* pairs of the form:

$$\begin{aligned}\{\textbf{powerService}:&\{\textbf{energyContracts}: \mathit{service}_1,\\ &\textbf{renewableEnergy}: \mathit{service}_2\},\\ &\textbf{other}: \mathit{service}_3\},\\ \textbf{lowReservoirs}:&\mathit{service}_4,\\ \textbf{powerStorage}:&\mathit{service}_5,\\ \textbf{pumpStation}:&\mathit{service}_6,\\ \textbf{highReservoirs}:&\mathit{service}_7,\\ \textbf{desalinationPlant}:&\mathit{service}_8,\\ \textbf{waterStorage}:&\mathit{service}_9\}\end{aligned} \tag{5}$$

  where each ***key*** represent service *id* which uniquely identifies that service and each $\mathit{service}_{\mathrm{i}} \in \{\mathit{service}_1 \ldots\ \mathit{service}_9\}$ is either a ***composite*** or an ***atomic*** service. As each *composite* service, such as *powerService*, contains at least one subservice, it includes the IDs of these *services* under *subService*. So, each ***composite*** service has the following form:

$$\begin{aligned}\{\textbf{type}:&\text{"composite"}\\ \textbf{inFlow}:&\{\mathit{inF}_1:\langle\text{f-value}\rangle,\\ &\mathit{inF}_2:\langle\text{f-value}\rangle,\ldots\}\\ \textbf{outFlow}:&\{\mathit{OutF}_1:\langle\text{f-value}\rangle,\\ &\mathit{OutF}_2:\langle\text{f-value}\rangle,\ldots\}\\ \textbf{metrics}:&\{\mathit{cost}:\langle\text{m-value}\rangle,\\ &\mathit{CO}_2:\langle\text{m-value}\rangle,\ldots\}\\ \textbf{constraints}:&\text{"True"}\mathit{or}\text{"False"}\\ \textbf{subServices}:&\{\text{set of service ids}\}\}\end{aligned} \tag{6}$$

  where each **inFlow** and **outFlow** contains a set of $\mathrm{f_i}$ and $\langle$ f-value $\rangle$ pairs that represent the flows going in and out each service. The form of each $\langle$f-value$\rangle$ can be express as follow:

$$\begin{aligned}\{\ \textbf{qty:}\ &[\mathit{num}_1 \ldots \mathit{num}_w],\\ \textbf{total:}\ &\mathit{numeric\ value}\}\end{aligned}$$

  where the **qty** is a sequence that shows the quantities of flow at each operational interval while the **total** shows the total flow for the whole operational window.

  In the same manner, the ***metrics*** value contains as set of metrics (such as cost or CO_2) with their corresponding $\langle m - value\rangle$ in the form of:

$$\begin{aligned}\{\textbf{perInt:}\ &[\mathit{num}_1 \ldots \mathit{num}_w],\\ \textbf{total}:\ &\mathit{numeric\ value}\}\}\end{aligned} \tag{7}$$

  where the *perInt* is a sequence that shows the metric per interval while the *total* shows the summation for all intervals.

  The ***constraints*** value indicates whether the service satisfies its constraints.

- The ***metrics*** it shows the metrics for the *rootService*.
- The ***constraints*** indicates whether all the constraints of the *rootservice* are satisfied.

The ***atomic*** service has the similar form except that:

- There is no **subServices** *key:value* pair.
- The *type* refers to one of the *atomic* analytical model in the library.
- Additional set of *key:value* pair:

$$\textbf{\textit{onFlag}}: [\text{Boolean}_1, \ldots, \text{Boolean}_w\ ]$$

  where each boolean value in the list indicate if the service is running "1" or not "0" at each interval of the operational window (w).
- An optional set of key value pair:

  $\{\ldots$
  typeSpecific:$\{$ set of *key:value* pairs $\}$,
  $\ldots\}$

  where **typeSpecific** value represents, using a *key:value* pairs , the parameters that are needed by the *atomic* analytic model **type** to calculate its metrics and constraints.
- An optional set of *key:value* pairs:

$$\begin{aligned}\textbf{state}: \{&st_1: [\mathit{num}_1 \ldots \mathit{num}_w],\\ &st_2: [\mathit{num}_1 \ldots \mathit{num}_w],\\ &\ldots\}\end{aligned}$$

  where the set keys (state) represent the temporal elements inside the service. So, each st_i, i maps to a list of numeric values that capture the state of the service in each interval.

In the next section, we describe a valid input model needed to compute the output instance.

4.3 Service Network desalination Instance: The Model input

The model input (*in*) follow the same structure as the output, but with some modification:

- No *metrics* and *constraints* objects.
- Instead of having a list to describe *qty* for each flow in the *composite* service, we replace it with a list of lower bound (*LB*).
- For an *atomic* service:
  - Instead of having a list to depict the *state*, we replace it with a single value that depicts the *state* at the beginning of the operational window:

$$\begin{aligned}\textbf{state}: \{&st_1: \mathit{num}_{\mathrm{initial}},\\ &st_2: \mathit{num}_{\mathrm{initial}},\\ &\ldots\}\end{aligned}$$

4.4 Analytic Model (AM)

To describe how analytical model forms the valid output (*out*) from a valid input (*in*), we indicate how the differences between the *out* (in form 4) and the *in* are computed:

$$\begin{aligned}
&\textbf{\textit{let}}\\
&\quad x = out(rootServiceID) = in(rootServiceID)\\
&\textbf{\textit{then}}\\
&\quad out(constraints) = out(services)(x)(constraints)\\
&\quad out(metrics) = out(services)(x)(metrics)\\
&\quad out(services) =\\
&\qquad \bigcup_{id \in ID} \boldsymbol{mOut}\Big(\boldsymbol{opOut}\big(in(services)(id)\big)\Big)
\end{aligned}$$

Below, we describe how the **(*opOut*)** and **(*mOut*)** functions construct the *out(services)* form from the input *composite* and *atomic* services *in(services)*. In section 4.4.1, we show how the operationOut **(*opOut*)** calculates out(services) without the metrics. Then, the result is used, by mertricOut **(*mOut*)**, to calculate the metrics in section 4.4.2.

4.4.1 operationOut (opOut)

In this section, we show how the operationOut **(*opOut*)** calculates the *inFLow* and *outFlow* quantities over the operational window as well as some flow constraints.

The *composite* service: As in form 6, for every *composite* service the quantity (***qty***) of every *inFlow* and *outFlow* is expressed recursively as:

$$\begin{aligned}
&\textbf{\textit{let}}\\
&\quad SUB = cs(subService)\\
&\quad qtyIn(id,x,y) =\\
&\qquad out(services)(id)(inFlow)(f_x)(qty)[y]\\
&\quad qtyOut(id,x,y) =\\
&\qquad out(services)(id)(outFlow)(f_x)(qty)[y]\\
&\textbf{\textit{then}}\\
&\quad \forall cs \in CS, i \in keys(inFlow),\\
&\quad j \in keys(outFlow), k \in \{1,\ldots,w\}\\
&\quad cs(inFlow)(f_i)(qty)[k] =\\
&\qquad \sum_{sub \in SUB} qtyIn(sub,i,k) - qtyOut(sub,j,k)\\
&\quad cs(outFlow)(f_j)(qty)[k] =\\
&\qquad \sum_{sub \in SUB} qtyOut(sub,j,k) - qtyIn(sub,i,k)
\end{aligned}$$

Therefore, the (***total***) for every *inFlow* and *outFlow* are:

$$\begin{aligned}
&cs(inFlow)(f_i)(total) =\\
&\quad \sum_{k=1}^{w} services(cs)(inFlow)(f_i)(qty)[k]\\
&cs(outFLow)(f_j)(total) =\\
&\quad \sum_{k=1}^{w} services(cs)(outFlow)(f_j)(qty)[k]
\end{aligned}$$

Also, for every composite service $cs \in CS$, the (***constraints***) expressed as a conjunction of demandConstraint(cs), boundConstraint(cs), and subServiceConstraints(cs). Each constraint is expressed recursively as follows:

$$\begin{aligned}
\textbf{\textit{let}}\;\; & SUB = cs(subService)\\
& inKeys(id) = keys(id(inFlow))\\
& outKeys(id) = keys(id(outFlow))
\end{aligned}$$

- domandConstraint(cs) $\equiv$

$$\begin{aligned}
&\forall sub \in SUB, k \in \{1,\ldots,w\}\\
&\quad \forall i \in \Big\{[outKeys(sub) \cup inKeys(sub)]\\
&\qquad - [inKeys(cs) \cup outKeys(cs)]\Big\}\\
&\qquad \boldsymbol{qtyIn}(sub,i,k) \geq \boldsymbol{qtyOut}(sub,i,k)
\end{aligned}$$

- boundConstraint(cs) $\equiv$

$$\begin{aligned}
&\forall i \in \{outKeys(sub) \cup inKeys(sub)\}, k \in \{1,\ldots,w\}\\
&in(cs)(services)(inFlow)(i)(LB)[k] \geq \boldsymbol{qtyIn}(cs,i,k)
\end{aligned}$$

- subServiceConstraints(cs) $\equiv$

$$\begin{aligned}
&\forall sub \in SUB\\
&\quad sub(constraints)
\end{aligned}$$

The *atomic* service (as): For every atomic service the quantity (***qty***) of every *inFlow* and *outFlow* is calculated by calling the analytical model of its *type* (see appendix). Additionally, every atomic ***constraints*** is expressed as a conjunction of bound Constraint, on Flag Constraint. The boundConstraint(as) is expressed as in the composite service boundConstraint(cs), while onFlagConstraint(as) is expressed as follows:
boundConstraint(as) $\equiv$

$$\begin{aligned}
&\forall i \in \{outKeys(as) \cup inKeys(as)\}, k \in \{1,\ldots,w\}\\
&\quad \big(as(onFlog) = 0\big) \rightarrow qtyOut(as,i,k)
\end{aligned}$$

The *state(as,i)* for $\forall s \in AS$ and $\forall k \in \{1,\ldots,w\}$ is expressed as:

$$state(as,i) = \begin{cases} newState(as,k,state(as,k-1)) & \text{if } k \geqslant 1\\ as(state) & \text{if } k = 1 \end{cases}$$

where *newState* is a function that returns the new state from a given *state* for a service with $id \in AS$, and interval $i \in \{1,\ldots,w\}$. Further, the atomic analytical model updates the *serviceSpecific key:value* pairs by adding some information that are needed later on to calculate the metrics which need larger intervals. For example, the desalination system operate per hour, while calculating some metrics like the cost for *energyContract* need the knowledge of average consumption for the last two months.

Table 1: The optimization time and objective value for the four desalination designs

Design	High Reservoir	Hydro generator	Power Storage	Water Storage	CPU time (s)	Objective Value	Peak demand bound (k)	Average power consumption (KW)
A	Yes	Yes	Yes	Yes	3.4	426.54	44.5	9.43
B	No	No	Yes	Yes	1.52	472.51	41.71	9.43
C	Yes	No	No	No	0.46	436.42	21.00	11.23
D	No	No	No	Yes	0.84	472.51	41.71	9.43

4.4.2 metricOut (mOut)

In this section, we show how the metricOut (***mOut***) calculates the *metrics* over the operational window as in form 7.

The *composite* service: For every composite service every ***metric*** (such as cost and CO_2) are expressed recursively as:

$$\begin{aligned}
&\boldsymbol{let}\\
&\quad \boldsymbol{mIn}(id) = opOut(in(services)(id))\\
&\quad mPerInt(id,x) =\\
&\qquad \boldsymbol{mOut}(mIn(id)(metrics)(cost)(perInt)[x])\\
&\quad mTotal(id) =\\
&\qquad \boldsymbol{mOut}(mIn(id)(metrics)(cost)(total))\\
&\boldsymbol{then}\\
&\qquad \forall cs \in CS, k \in \{1,\ldots,w\}\\
&\qquad mPerIn(cs,k) = \sum_{sub \in SUB} mPerIn(sub,k)\\
&\qquad mTotal(cs) = \sum_{sub \in SUB} mTotal(sub)
\end{aligned}$$

The *atomic* service (as): The metrics and constraints for the atomic services are calculated by calling the analytical model of its *type* (see appendix).

5 Experimentation

In this section, we show how the proposed desalination AM can support diverse desalination systems. We run an experimentation that aim to asses the capability of the model in solving realistic problems using a machine with a 1.8 GHz Intel Core i5 processor and 8 GB of DDR3 memory executed at 1600 MHz. We used CPLEX 12 as an optimization tool.

We implement the system using the architecture proposed in 3. Then, we create a design that includes low reservoir, pump, RO plant and power source which in turn includes renewable source (solar panels) and the grid. We assume a time horizon of 24 hours and generate a random supply of renewable energy source (solar panels), that follow the normal bell shape curve during the first 12 hours and 0 otherwise. We also use a power contract agreement which charges extra fee over its fixed fee using the following formula:

$$Extra = (k - Avg) * r \tag{8}$$

where:
Avg:the average of power consumption over the time horizon
k: The peak demand bound
r: is the rate of extra KWH.

Figure 4 shows the four desalination architectures we design. In architecture A, we add the high reservoir connected with hydro generator and power and water storage to produce the variable demand. In architecture B, we add power and water storage. while in architecture C, we only add the high reservoir and in architecture D we did not add any extra component. We then specify for each component we added its *type specific* parameter (like the maximum capacity for the water storage). Then, we optimize the the flows of power and water as well as the peak demand bound against the following objective function:
*Total cost of operation + 0.2 * Total carbon emission*

Table 1 shows the objective values for the four designs. These architectures use different storage technologies to store excess supplies during intervals of low demand to satisfy the fluctuations in demand. We can see that architecture A and C achieve better results than C and D under the same input assumption. These comparison can lead to useful insight in knowing which component can contribute to the system the most.

For the purposes of evaluating the model in solving realistic size problems, we generate four different instances from architecture A by varying two dimensions:(1) The number of *atomic* services (AS) and (2) The number of intervals in the time horizon (*w*). Table 2 shows the number of atomic services and the number of intervals used in each instance. Figure 5 show the progress of the solver in solving the four instances. We can see that when we set the time horizon (w) to 24 hour, we reach the optimal solution in 16 seconds and 3 minutes when the number of atomic services are 78 and 306, respectively. Whereas, when we use 168 intervals we converge to near optimal solution within 0.43% gap in 25 seconds and 1.29% gap in 45 seconds when the number of atomic services are 42 and 78, respectively. As an initial step, the solution time to optimality is practical to operate on an interval (e.g., 1 hour).

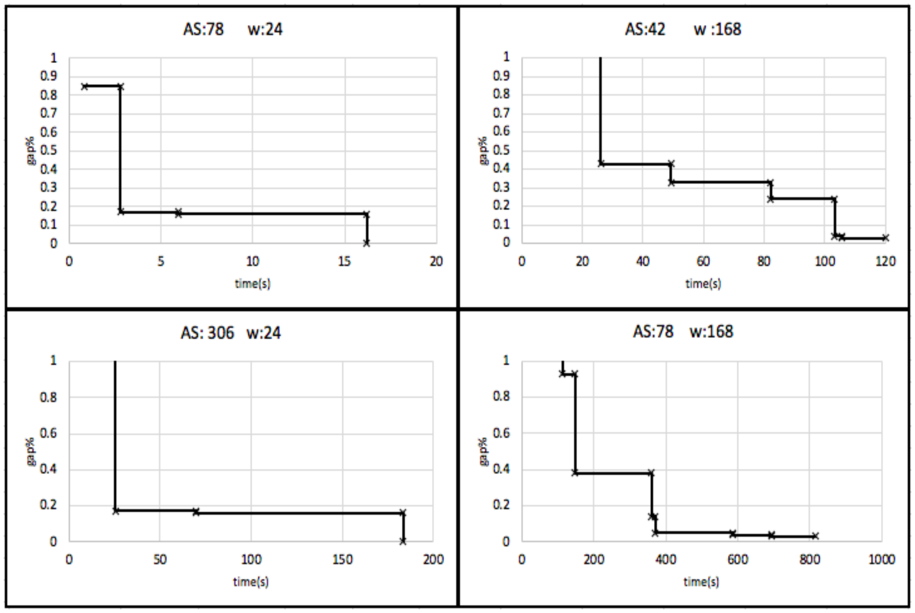

Figure 5: CPLEX solution progress

Table 2: shows different variation of problem sizes

Intervals (w)	Atomic Services (AS)	Gap (%)	Time (s)
24	78	0	16
24	306	0	183
168	42	0.03	105
168	78	0.03	693

6 Conclusions and Future Work

We reported on the development of a formal analytic model and a decision guidance system for desalination service networks that can be applied to a broad range of desalination designs and architectures. The model and the system are based on an extensible repository of atomic component models, initially including models for pumps, renewable energy sources, water and power storage, and reverse osmosis units. We conducted an experimental study to demonstrate the applicability of the model and the system to a range of desalination designs, using four examples, and the scalability of the solution to realistic size problem.

As future work, we plan to expand the experimentation to study a realistic water supply chain using our proposed model. Additionally, we plan to develop a modular investment model based on the accurate operational model developed in this paper.

References

Abdelshafy, A. M., Hassan, H., and Jurasz, J. (2018). Optimal design of a grid-connected desalination plant powered by renewable energy resources using a hybrid pso–gwo approach. *Energy Conversion and Management*, 173:331–347.

Ahmed, F. E., Hashaikeh, R., Diabat, A., and Hilal, N. (2019). Mathematical and optimization modelling in desalination: State-of-the-art and future direction. *Desalination*, 469:114092.

Al-Nory, M. (2019). Optimal decision guidance for the electricity supply chain integration with renewable energy: Aligning smart cities research with sustainable development goals. *IEEE Access*, PP:1–1.

Al Nory, M. and Graves, S. (2013). Water desalination supply chain modeling and optimization.

Al-Nory, M. T. and Brodsky, A. (2014). Towards optimal decision guidance for smart grids with integrated renewable generation and water desalination. In *2014 IEEE 26th International Conference on Tools with Artificial Intelligence*, pages 512–519.

Al-Nory, M. T. and El-Beltagy, M. (2015). Optimal selection of energy storage systems. In *2015 Saudi Arabia Smart Grid (SASG)*, pages 1–6.

Azhar, M. S., Rizvi, G., and Dincer, I. (2017). Integration of renewable energy based multigeneration system with desalination. *Desalination*, 404:72 – 78.

Brodsky, A., Krishnamoorthy, M., Nachawati, M. O., Bernstein, W. Z., and Menascé, D. A. (2017). Manufacturing and contract service networks: Composition, optimization and tradeoff analysis based on a reusable repository of performance models. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1716–1725.

Brodsky, A. and Luo, J. (2015). Decision guidance analytics language (dgal). In *Proceedings of the 17th International Conference on Enterprise Information Systems - Volume 1*, ICEIS 2015, pages 67–78, Portugal. SCITEPRESS - Science and Technology Publications, Lda.

Brodsky, A. and Wang, X. S. (2008). Decision-guidance management systems (dgms): Seamless integration of data acquisition, learning, prediction and optimization. In *Proceedings of the*

41st annual Hawaii international conference on system sciences (HICSS 2008), pages 71–71. IEEE.

Gençoğlu, G. and Merzi, N. (2016). Trading-off constraints in the pump scheduling optimization of water distribution networks. *Journal of Urban and Environmental Engineering*, 10(1):135–143.

Marques, J., Cunha, M., and Savić, D. (2014). Decision support for optimal design of water distribution networks: A real options approach. *Procedia Engineering*, 70:1074 – 1083. 12th International Conference on Computing and Control for the Water Industry, CCWI2013.

Mokheimer, E. M., Sahin, A. Z., Al-Sharafi, A., and Ali, A. I. (2013). Modeling and optimization of hybrid wind–solar-powered reverse osmosis water desalination system in saudi arabia. *Energy Conversion and Management*, 75:86 – 97.

Nachawati, M. O., Brodsky, A., and Luo, J. (2017). Unity decision guidance management system: Analytics engine and reusable model repository. In *ICEIS (1)*, pages 312–323.

UN (2015). Transforming our world : the 2030 agenda for sustainable development : resolution /. page 35 p. Issued in GAOR, 70th sess., Suppl. no. 49.

APPENDIX

Atomic Models formulation

So far we have created the analytic models (AMs) for water storage, Lithium Ion Batteries, pump, RO desalination unit, solar panels and power contract. Due to page limitation we omit the formalization of the atomic analytic models(AMs). We briefly describe how each atomic services that belong to these AM types produces for every interval its *inFlow* and *outFlow* quantities, performance metrics, and constraints using these atomic AMs.
For all atomic models, we calculate the total cost using operation and maintenance (O&M) cost for each type and the CO_2 emission is calculated by multiplying the emission factor to the amount of power consumed.

WATER STORAGE
High and low reservoir and water storage use the *WATER STORAGE* AM to calculate the total operation cost as well as the constraints that insure:(1) water is balanced between the *inFlow*,*outFlow* and the current state of the water in storage at any given interval. (2) The water in the storage should not exceed a given capacity which is listed under *type specific*.

PUMP The PUMP AM use an equitation to calculate the amount of *outFlow*s given the efficiency of the pump. The constraints insure: (1) the amount of power is sufficient to push the *outFlow* quantities of water given the pump specification (such as height,...) under the *typeSpecific*.

ENERGY CONTRACT The energy contract charges different rates depend on the amount of power consumed (see section 5). The CO_2 emission is calculated using the amount of power consumed. The constraint insure that the *outFlow* power cannot exceed the maximum capacity.

SOLAR PANELS The *outFLow* power from solar is calculated by multiplying the area per panel, number of panels, panel efficiency, and the amount of solar energy given each interval.

REVERSE OSMOSIS (RO) The RO use similar formula as in the PUMP, but the RO require different specifications (such as the number of membrane,...), which can be found in the *typeSpecific*.

LITHIUM ION BATTERIES Calculating the new state of the battery (the battery state) is depending on the efficiency factor which decrees at low rate. The constraint insure that the *outFlow* power is balanced with the power in the batteries and the *outFlow* power at any given interval.