Journal of Artificial Intelligence Research 70 (2021) 77-117 Submitted 08/2020; published 01/2021

Integrated Offline and Online Decision Making Under Uncertainty

Allegra De Filippo ALLEGRA.DEFILIPPO@UNIBO.IT
Michele Lombardi MICHELE.LOMBARDI2@UNIBO.IT
Michela Milano MICHELA.MILANO@UNIBO.IT
Department of Computer Science and Engineering
University of Bologna, viale Risorgimento, 2
40136 Bologna, ITALY

Abstract

This paper considers multi-stage optimization problems under uncertainty that involve distinct offline and online phases. In particular it addresses the issue of integrating these phases to show how the two are often interrelated in real-world applications. Our methods are applicable under two (fairly general) conditions: 1) the uncertainty is exogenous; 2) it is possible to define a greedy heuristic for the online phase that can be modeled as a parametric convex optimization problem. We start with a baseline composed by a two-stage offline approach paired with the online greedy heuristic. We then propose multiple methods to tighten the offline/online integration, leading to significant quality improvements, at the cost of an increased computation effort either in the offline or the online phase. Overall, our methods provide multiple options to balance the solution quality/time trade-off, suiting a variety of practical application scenarios. To test our methods, we ground our approaches on two real cases studies with both offline and online decisions: an energy management problem with uncertain renewable generation and demand, and a vehicle routing problem with uncertain travel times. The application domains feature respectively continuous and discrete decisions. An extensive analysis of the experimental results shows that indeed offline/online integration may lead to substantial benefits.

1. Introduction

Optimization under uncertainty arises in many application areas, such as project scheduling, transportation systems, financial systems, and energy management: fuel prices, electrical power, activity durations, travel times, etc. are effectively stochastic in the real world. Optimization problems in this class can be seen as a sequence of multiple stages, such that at each stage part of the uncertainty is revealed and some decisions must be made. Such decisions are irrevocable and made without full knowledge of the future: they should therefore account for multiple (ideally all) possible outcomes, and optimize a *probabilistic* performance measure (e.g.the expected value of a relevant cost metric).

The need to account for multiple future developments makes stochastic optimization incredibly challenging, which explains how approximate (sampling-based) methods and heuristics are the most popular solution techniques (Powell, 2016; Van Hentenryck & Bent, 2009). Due to such a complexity, the applicable approaches depend on the temporal granularity of the decisions to be made. Long-term "strategic" decisions (which are often very impactful) are typically solved via expensive, but more accurate, sampling-based approaches. Short-term "operational" decisions often need to be made over multiple steps, within a short time frame: they are commonly addressed via polynomial-time heuristics, while more advanced sampling-based methods are applicable only if their computational cost is carefully managed. We will broadly refer to the first class of problems (and solution approaches) as *offline* and to the second as *online*.

In this paper, we move from the observation that many practical application scenarios require to make *interdependent offline and online decisions*. For example, we may need to define a daily production schedule for an industrial plant, and then manage its power supply on a hour by hour basis; or we may assign customers to vehicles for delivering goods, and then adjust their routes dynamically as the traffic conditions reveal themselves over time. The simplest approach to tackle such problems is to deal with the offline and online phase separately, (e.g.) via a sampling-based method and a heuristic respectively. However, we will show that *substantial improvements can be obtained by treating the two phases in an integrated fashion*.

This paper builds over and substantially extends our previous work (De Filippo, Lombardi, & Milano, 2018b, 2018a). In particular, the first and the fourth methods (respectively ANTICIPATE and ACTIVE) are already presented in the two mentioned conference papers, while the second and the third (i.e. TUNING and ACKNOWLEDGE) are new methods proposed in this work to fill the gap between online and offline phases in optimization under uncertainty. The main focus of these new methods is how to manage the (high) offline computational effort of ACTIVE by maintaining an efficient online heuristic by ensuring robust solutions. The results of our previous cited works (related to VPP and VRP cases study for ANTICIPATE and ACTIVE) are presented in this paper to compare them with the results of the new proposed methods and to make a more extensive overview on integrated offline/online optimization methods under uncertainty. The paper is structured as follows: Section 2 provides a review of methods for optimization under uncertainty, both in an offline and online setting, and then focuses on motivating examples for offline/online integration. Section 3 describes the starting baseline model which is designed to be representative of this state of the art. Section 4 describes in details our proposed methods (as improvements of the baseline model) by pointing out the importance of both the offline and the online part for each method. Finally, in Section 5 and Section 6 we ground our methods on two real cases study and we provide an exhaustive analysis and discussion of the results. Section 7 concludes with final remarks and future work.

1.1 Contribution

In pure offline and online methods for optimization under uncertainty, if we fail to realize that we have a tight integration between these two phases, we may often have undesirable side effects and missed opportunities. For example, the solutions produced by pure offline methods may conflict with the behavior of the approach used for making online decisions; conversely, an online solver may struggle with producing solutions within a reasonable time budget, not realizing that some degree of offline preparation could greatly simplify the issue. For these reasons, our contribution is focused on how to manage the delicate tradeoff between the solution quality and the computational effort, either via approximations or by shifting part of the computational load to the offline phase. Generally, increasing the quality of the solution by making it robust, for example to unexpected events, requires a higher computational cost. By integrating the offline and online phases and taking into account the information available in advance, it is possible to choose where to move the computational load for high-quality solutions (De Filippo, Lombardi, & Milano, 2020b, 2020a).

To compare our methods with existing literature, we consider a baseline composed by an approach that deals with offline decisions via a sampling-based method (e.g. a two-stage stochastic optimization model is used for the offline phase), and with online decisions via a greedy heuristic, that is representative of best practices in real problems. This baseline is not problem specific, instead

we simply assume that: 1) the uncertainty is exogenous; 2) the online heuristic can be stated as convex optimization problem. We then show how to improve the baseline in different directions, each altering either the offline or the online component of the solution process, so that the two play better together. All our methods are applicable under the same (general) assumptions as the baseline.

We believe our techniques represent a significant step toward integrated offline/online optimization: we propose four methods to improve the baseline in different directions. First, we try to improve the online solver by adding some anticipatory capabilities: this is the key idea in our ANTICIPATE method. Second, we try to make the offline solver explicitly aware of the limitations of the online approach: we mitigate the discrepancy by translating the online greedy heuristic as a set of constraints, which can be injected in the offline model. When stringent time constraints exist, it may be better to improve the greedy heuristic by simply adjusting its parameters (this is the main idea in the TUNING approach). Then, shifting our attention to the offline decisions, this technique (of injecting constraints in the offline model to translate the online heuristic) leads to our ACKNOWLEDGE method and, by combining this approach with an offline parameter tuning to achieve even deeper integration, to our ACTIVE method.

To test our methods, we ground them on two case studies, matching the examples mentioned earlier: 1) an energy system management problem, where load shifts are planned offline (the day ahead) and power flows must be controlled online (e.g. hour by hour); and 2) a Vehicle Routing Problem where customer are assigned offline, but the routes can be chosen online (i.e. based on the uncertain travel times). The first problem features a continuous (and hence non-enumerable) decision space, while the second has pure discrete decisions. In our experiments, all the proposed methods significantly improve over the baseline in terms of solution quality. While the computation cost is always higher than the baseline, each approach hits a different trade-off in terms of offline and online solution time. It is important to stress that the first problem features a decision space that is not enumerable (e.g. with continuous decision variables) while the second has the possibility to enumerate the decision space since it features discrete variables and its results (for the method that presents online anticipatory algorithm) can be compared with existing algorithms in literature.

2. Background and Motivation

In this section we provide a brief overview of methods for optimization under uncertainty, both in an offline and online setting, and then provide motivating examples to show how the two are often interrelated in real-world applications. The distinction between offline and online problems in the literature is somewhat blurry: in this paper, we will refer as "online" to *problems that need to be solved repeatedly over time*, with the outcome of each solution attempt affecting the subsequent ones. In practice, online problems often need to be solved within strict time limits, while this requirement is relaxed for offline problems.

2.1 Optimization Under Uncertainty: General Concepts

A key difficulty in optimization under uncertainty is in dealing with an uncertainty space that is huge and frequently leads to very large-scale optimization models, and decision-making process is often further complicated by the presence of decisions in a multi-period or multi-stage setting. Due to their complexity and the need of complete knowledge of the future, stochastic problems are traditionally solved via offline methods. There is however a growing interest in online algorithms to make decisions over time (without complete knowledge of the future), so as to take advantage of the

information that is slowly revealed. Generally, in optimization under uncertainty, both approaches make sense: "strategic" decisions can be taken off-line, while "operational" decisions are better left to an online approach.

The need to make decisions without complete knowledge about the problem data is an extremely common situation, and there is a growing realization that dealing with uncertainty in optimization is necessary to achieve real-world impact in many domains. The field has been extensively investigated, and we refer the reader to Kall and Wallace (1994); Birge and Louveaux (1997); Shapiro and Philpott (2007); Sahinidis (2004); and Powell (2016) for a historical perspective and comprehensive overview.

Data subject to uncertainty is usually represented via *random variables* (see, e.g., Birge & Louveaux, 1997; Kall & Wallace, 1994). A random variable ξ does not take value, but is instead *sampled* to obtain realizations, from a continuous or discrete set called *support* (i.e. the variable domain). A *probability distribution* defines how likely each value in the support is to be sampled. As mentioned in Section 1, stochastic optimization problems can be viewed as composed of multiple stages. At each stage, some uncertain elements are observed (i.e. one or more random variables are sampled), and some decisions must be made (i.e. some decision variables need to be assigned). The uncertainty is said to be exogenous if the distribution of the random variables in a stage does not depend on the decisions made in previous stages (e.g. weather conditions), and endogenous in the opposite case (e.g. recovering from an illness, while receiving cures).

The goal is to optimize a probabilistic performance measure (e.g. the expected problem cost), subject to both deterministic and probabilistic constraints. Probabilistic constraints can be further divided on constraints over expectation (e.g. the expected stock of certain goods in a warehouse should be above a given level) or *chance* constraints (e.g. the probability that the stock is above a given level should be higher than a threshold). For more details, see, e.g., Zhang and Li (2011); Li, Arellano-Garcia, and Wozny (2008); and Sahinidis (2004).

Robust Optimization Sometimes, when a deterministic model is clearly inappropriate, and there are few clues to the probabilities to use a stochastic model, it is useful to work with ranges of uncertainty. Uncertain parameters, in this case, are assumed as restricted to particular intervals, without an associated probability distribution. This is the key idea in so-called *robust optimization* and has the additional benefit of reducing even further the computational costs (Bertsimas & Sim, 2004; Bertsimas, Brown, & Caramanis, 2011; Zheng, Wang, & Liu, 2015). Instead of minimizing the total expected cost as in stochastic optimization, robust optimization reduces the worst-case costs for all possible results of uncertain parameters.

Hybrid stochastic/robust models have been proposed in recent years (see, e.g., Zhao & Guan, 2013) to combine the advantages and compensate for the disadvantages of pure robust and stochastic approaches to make better decisions in complex domains under uncertainty.

Sampling and the Sample Average Approximation (SAA) With a few exceptions, the probability distributions of the random variables are approximated by drawing a finite number of samples (Shapiro, 2013): this yields a collection of realizations referred to as *scenarios*. This sampling step can be done prior to the solution process in case of exogenous uncertainty, but must be performed at search time for endogenous uncertainty. Sampling and scenarios allow to tackle stochastic optimization via the Sample Average Approximation (Shapiro & Philpott, 2007; Shapiro, 2013). In this approach, a set (i.e. a copy) of deterministic decisions is associated to each scenario, which allows to deal with expected values and chance constraints via summations and averages. To ensure

meaningful solutions, it is important to add so-called *non-anticipativity* constraints to ensure that no decision is made with perfect knowledge of the future. Indeed, a model with non-anticipativity constraints has no knowledge (information) on how the uncertainty will resolve in the (future) remaining stages and they only know the inputs and the state of the system in the current stage. In practice, if two scenarios share the same realization for the random variables in stages 1 to k, then their decisions for the stages 1 to $k + 1$ must be identical.

The SAA is a powerful and general method, but also very expensive from a computational point of view (Walsh, 2002). For this reason, its application has been historically limited to offline, two-stage, problems. This is in fact the context for which the SAA was originally developed, and the focus of many acceleration techniques such as the classical L-shaped method (Laporte & Louveaux, 1993). Applying the SAA to a two stage problem requires to determine a single set of decisions for stage 1, and one set of decisions (the so-called *recourse actions*) per scenario for stage 2 (Shapiro, 2008). This structure is depicted in Figure 1.

Robust and stochastic optimization methods have traditionally focused on strategic (*offline*) optimization, under relaxed time constraints. Indeed, offline approaches are concerned with making decisions before the uncertainty is revealed: they can obtain high quality and robust solutions, but they have a significant computational cost.

Online Stochastic Optimization When dealing with a problem with more than two stages, optimizing at run time provides the opportunity to adapt the solutions to unexpected events (since those can be observed) and to reduce the computational cost (since there is no need to plan for every possible outcome). This line of reasoning is at the basis of stochastic online optimization that, due to the frequent presence of tight time limit on the solution process, has been traditionally tackled via heuristics.

In recent years, the availability of improved algorithms and computing power has enabled the application of sampling-based algorithm also in an online setting: these are often referred to as *anticipatory algorithms*, many of which received excellent coverage in Van Hentenryck and Bent (2009). Generally speaking, these algorithms need to be run at each stage and rely on scenarios to obtain approximate information about the future: this enable significant improvements in terms of quality, but comes with a substantial computational cost that must be carefully managed.

For example, the EXPECTATION algorithm (Bent & Van Hentenryck, 2004a) attempts to reduce the solution time by optimizing each scenario independently (and therefore as a deterministic problem), for all possible decisions; the method then selects the decision which maximizes the expected profit. The CONSENSUS algorithm (Bent & Van Hentenryck, 2004c) improves over this scheme by solving a deterministic problem per scenario. Every time a decision for the current stage is picked as optimal in one of those problems it receives a votes; once the process ends, the algorithm chooses the decision with the highest number of "votes".

The technique employed by CONSENSUS has some adverse effects on the solution quality, which are addressed in REGRET algorithm (Bent & Van Hentenryck, 2004b) by extracting more information from each solved problem; this leads to a more reliable selection of the optimal decision for the current stage. The AMSAA method (Mercier & Van Hentenryck, 2008) instead hybridizes SAA and Markov Decision Processes techniques to improve the solution quality at the expense of the computational cost. All the anticipatory online algorithms mentioned so far are applicable only to problems with discrete, enumerable, decisions.

Figure 1: Two-Stage Stochastic Programming (Van Hentenryck & Bent, 2009)

There is always a trade-off between the computation cost and the quality and robustness of the provided solution. This trade-off is the primary object of investigation in Mercier and Van Hentenryck (2007), and can be tuned by adjusting the number scenarios and the so-called look ahead horizon, i.e. the number of future stages that are taken into account in each scenario. More in general, the method for generating the scenarios can be adapted to the given problem and the user goals, as described for example in Kaut and Wallace (2003).

2.2 Work Position in the Current Literature

In previous sections we overviewed purely offline and online methods. Offline approaches are concerned with making decisions before the uncertainty is revealed: they can obtain high quality and robust solutions, but they have a significant computational cost. Online algorithms focus on reacting to unexpected events once they are observed, but often run under strict time constraints, making it challenging to compute high-quality solutions.This distinction in offline and online optimization has led to consider the two modes separately in recent literature. However, in many cases multi-stage optimization problems under uncertainty can be considered composed of both an offline "strategic" phase and an "operational" online phase, as shown in the following Section 2.3. Strict constraints on the available decision time-to-solution are often present in the online phase, but are absent (or very relaxed) on the offline one. Since the available solution methods in literature tend to be polarized between offline and online approaches, all our integrated offline/online proposed methods are not directly related to the current literature. Below we list all the points in common with the current literature. We consider the usual setup for multi-stage stochastic problems: 1) the number of stages is priori-known; 2) uncertainty is represented via random variables with known distributions; 3) uncertainty is exogenous (i.e. the distributions do not depend on the decisions). These assumptions hold in a great variety of applications and have significant computational advantages, as shown in the cited references. Moreover, our work is related to all the general concepts presented in Section 2.1: 1) we deal with uncertainty via sampling (i.e. we sample realizations by assuming that the error for our forecasts can be considered as an independent random variable.); 2) we focus on a number of extreme scenarios, thus making our model somewhat close to robust optimization; 3) online stochastic optimization literature is cited and analyzed to introduce (and compare) our online anticipatory algorithm. However, all our other methods are not directly comparable because they focus on problems that feature both strategic and operational decisions and the objective is to control offline decisions in such a way (e.g. parameter tuning, optimality conditions) so that they synergize with the online solver. To the best of our knowledge, these methods have been a first attempt in the direction of a real integration between offline (strategic) and online (operational) decision phases. In details, related to the previous works cited in Section 2.1, the only method that is suitable for a

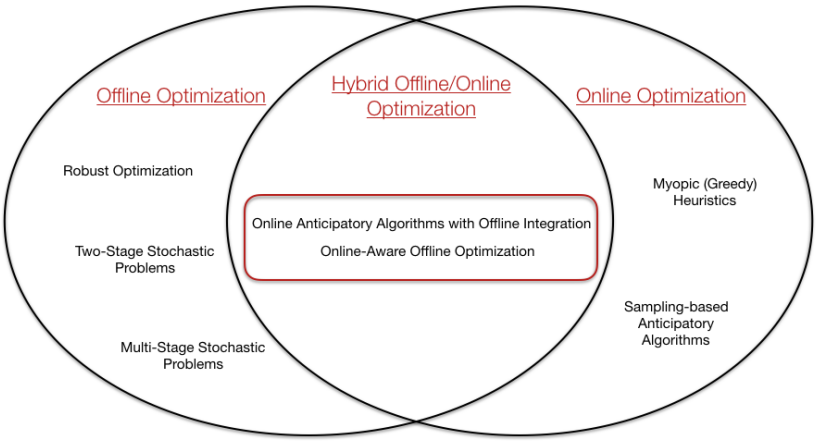

Figure 2: Position of our work in the current state of the art in Optimization under Uncertainty

comparison is ANTICIPATE. In particular, the online phase of our method has the same semantic of EXPECTATION which obtains the same results by enumerating the feasible decisions for the current stage, and evaluating the expected cost by solving the problem on each scenario. However, when the decision space is not enumerable (e.g. for continuous variables), EXPECTATION, REGRET (and even CONSENSUS and AMSAA) cannot be applied in a straightforward fashion, while our method is applicable also in the case of numeric decisions. It is important to consider also that when each online stage requires to take multiple decisions, enumeration may be expensive and associating an expected cost to each decision in isolation may lead to an underestimation of the expected cost. Moreover, the most important improvement given by our method is the integration of the online anticipatory algorithm with offline decisions (i.e. customer assignment or load shift planning in the offline phase of ANTICIPATE).

2.3 Motivating Examples for Offline/Online Decision-Making

In this section, we review some real world use cases that are typically solved via either offline or online models, while in fact they are *integrated offline/online problems*. In the model descriptions in the following section, y will represents the offline decisions; x^k will represent the online decisions for stage k; s^k (resp. ξ^k) will represent the system state (resp. the uncertainty) revealed at the beginning (resp. the end) of stage k. We will show them in the following motivating examples.

- *Energy Management Systems (EMS)* are key components of the electrical grid that maintain its stability both by shifting consumption (over time) and routing power flows from the available generators. EMS need to tackle a very challenging problem, due to the progressive shift towards decentralized generation, the strong penetration of (uncontrollable and stochastic) Renewable Energy Sources (RES) that represent the uncertainty for each stage ξ^k, and the integration of flexible (deterministic) energy systems. In practice, the load shifts must be planned offline with offline decisions y (the day ahead) and the power flow balance should be maintained online (e.g. hour by hour) x^k, so as to minimize the costs (see, e.g., Morales, Conejo, Madsen, Pinson, & Zugno, 2013; Clavier, Bouffard, Rimorov, & Jos, 2015).

- In *transportation systems*, a central role is played by the Vehicle Routing Problem and its variants (Toth & Vigo, 2002), which consists in establishing the paths for a set of vehicles to serve a set of customers. In a real world setting, many aspects (e.g. customer demands and travel times) are also subject to uncertainty ξ^k (Manzo, Nielsen, & Prato, 2014). Several transportation companies focus on assigning customers to smaller scale operators (offline y), which are then in charge of choosing the routes (online x^k). In the next section we will use this case study to build a running example to present and clarify our methods.

- In *project scheduling* the goal is to generate a feasible schedule that optimizes some performance metric (i.e. the project duration), in presence of limited resources. This schedule can serve as a basis for planning external activities such as material procurement, preventive maintenance and delivery of orders to external or internal customers. During execution, project activities are subject to considerable uncertainty ξ^k that may lead to schedule disruptions (Herroelen & Leus, 2005). A disrupted schedule incurs higher costs due to missed deadlines, resource idleness, higher work-in-process inventory and possible frequent rescheduling. Like in the previous examples, it is possible to plan project activities offline (y) and then to use online algorithms to improve (online x^k) solutions as the elements of uncertainty reveal themselves.

- In *reservation systems* requests arrive online and must be dynamically allocated to limited resources in order to maximize profit (Van Hentenryck, Bent, & Vergados, 2006). Example include hotel of flight booking systems, which are both subject to considerable uncertainty ξ^k in the real world. Once again, a base reservation plan is usually devised offline (y), but it then needs to be integrated with an online dynamic system (x^k) to cope with unexpected disruptions.

All such problems feature both offline and online phases, which are typically solved in isolation, despite being strongly interconnected. In this paper, we will show that a tighter integration between the two phases can lead to substantial improvements: this will be done via an empirical evaluation using the first two examples (energy management and transportation system) as case studies.

Generally, increasing the quality of the solution and the robustness (e.g.) to unexpected events requires a higher computational cost. The integration of the offline and online phases allows to obtain higher quality solutions compared to the Baseline. In general this is achieved by making sure that the offline decisions synergize with the online solver, by tuning the behavior of the online solver itself, or by making the online solver anticipatory. In practice, as shown in the final experiment sections, our method based on online anticipatory algorithms (i.e. ANTICIPATE) yields substantially high-quality solutions but, being also MILP-based, it takes non-negligible time during the online phase. The remaining methods are based on the idea of controlling offline decisions so that they synergize with the online (fast) solver. They obtain solutions whose quality matches or beats that of ANTICIPATE, at the cost of a higher offline computation time, though it has the same online computational cost of the baseline.

3. Baseline Offline/Online Model

We can now proceed to describe our baseline method, which will bee improved in Section 4 via three broad ideas: 1) improving the online heuristic by adding an anticipatory component; 2) making

the offline solver aware of the online heuristic and its limitations; 3) tuning the parameters of the online heuristic to alter its behavior. The first idea is closely related to existing online anticipatory algorithms (e.g. EXPECTATION); the second and third ideas exploit the mixed nature of the problem to enable improvements via a deeper integration of the offline and online phases. We formalize our methods to propose general approaches that can be applied to different real world use cases, as long as a few basic assumptions are satisfied.

Historically, methods such as stochastic optimization (see, e.g., Shapiro & Philpott, 2007; Birge & Louveaux, 1997; Kall & Wallace, 1994) have been used for the offline phase, while the online phase has often been tackled via simple, non-anticipatory, heuristics. Our baseline method is designed to be representative of this state of the art. In particular, we use a sampling-based model for the offline decision that is already capable of taking into account the existence of the online phase, albeit in a limited fashion. For the online phase itself, we use instead a fast greedy heuristic.

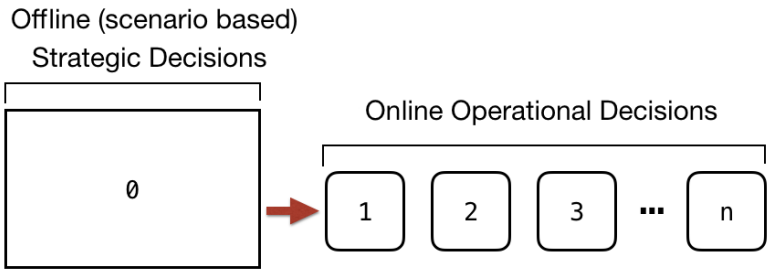

Figure 3: Baseline Offline and Online Integration (De Filippo et al., 2018a)

We assume *exogenous uncertainty*, and that the overall management system is composed by two macro steps: *the offline decisions are made by a two-stage stochastic optimization model*, based on sampling and scenarios. The second step is an online algorithm, implemented within a simulator, that tries to make optimal online choices, by building over the offline decisions. We make the assumption that *the online algorithm is based on a convex parametric optimization model*. Since our methods use (offline) parameter tuning techniques, we consider an online parametric heuristic: the heuristic parameters could be costs, resource availability or weights, depending on the semantic used. In our case studies (i.e. an Energy Management System and a Vehicle Routing Problem), we use parametric costs associated to the storage system for the EMS, and travel times for the VRP.

Formal Problem Description We view mixed offline/online problems as on n-stage problems where the first-stage decisions are *strategic* (and can be taken with relative leisure), while the remaining n - 1 stages involve *operational* decisions (with tighter temporal constraints).

In the model descriptions, y will represents the offline decisions; x^k will represent the online decisions for stage k; s^k (resp. ξ^k) will represent the system state (resp. the uncertainty) revealed at the beginning (resp. the end) of stage k. All variables are assumed to be vector-valued; they can be either continuous or discrete, and have either finite or infinite domain.

We will refer to $\mathcal{F}(y, x^k, s^k)$ as the cost incurred at stage k for taking decisions x^k. The cost directly associated to the offline decisions is instead referred to as $\mathcal{F}_o(y)$. Therefore, the total cost for a single run over all the stages is given by:

$$\mathcal{F}_o(y) + \sum_{k=1}^{n} \mathcal{F}(y, x^k, s^k) \tag{1}$$

The transition from the state in stage k to the state in stage $k+1$ is defined by means of a *transition function* T, i.e.:

$$s^{k+1} = T(y, x^k, s^k, \xi^k)$$

where it can be seen that the effect of the uncertainty (i.e. the random variable) is encoded in the state.

Flattened Problem Before introducing the model for the offline phase, it is useful to discuss a common approximation technique employed to reduce the computational cost of solving a multi-stage problem.

Let Ω be a set of scenarios ω for $\xi = (\xi^0, \dots \xi^{n-1})$. Given a single scenario ω, it is possible to collapse the constraint and cost of each stage to obtain a *flattened (online) problem*:

$$\min \sum_{k=1}^{n} \mathcal{F}(y, x^k_\omega, s^k_\omega) \qquad (\mathbf{PF})$$

$$\text{s.t. } e(y, x^k_\omega, s^k_\omega) = 0 \qquad \forall k = 1..n \tag{2}$$

$$g(y, x^k_\omega, s^k_\omega) \leq 0 \qquad \forall k = 1..n \tag{3}$$

$$s^{k+1}_\omega = T(y, x^k_\omega, s^k_\omega, \xi^k_\omega) \qquad \forall k = 1..n-1 \tag{4}$$

where x^k_ω/s^k_ω/ξ^k_ω are the online decisions/state/realizations for stage k in scenario ω. Functions e and g are vector-valued in general and define the constraints for each stage.

Since **PF** assumes the availability of all ξ^k_ω values, it is effectively a clairvoyant approach, due to the lack of non-anticipativity constraints. In the online optimization literature the flattened problem is better known as the offline problem (Van Hentenryck & Bent, 2009): we adopt a different name to avoid ambiguity with the actual offline phase.

Note that the flattened problem is obtained by collapsing *online* stages, for which we have made a convexity assumption. This implies that g must be convex and e linear. From a computational standpoint, this also means that **PF** is largely convex itself, and that its complexity depends heavily on the properties of the state transition function. If T is linear, then the flattened problem will be convex and relatively easy to solve. Non-linear transition functions are conversely much harder to handle.

Offline Problem As a baseline to deal with the offline decisions we consider a two-stage stochastic optimization problem obtained by instantiating **PF** once per scenario:

$$\min \mathcal{F}_o(y) + \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{k=1}^{n} \mathcal{F}(y, x^k_\omega, s^k_\omega) \qquad (\mathbf{PO})$$

$$\text{s.t. Eq. (2)} - \text{(4)} \qquad \forall \omega \in \Omega$$

$$s^1_\omega = T_o(y, \xi^0_\omega) \qquad \forall \omega \in \Omega \tag{5}$$

$$y \in \mathcal{Y} \tag{6}$$

where we recall that $\mathcal{F}_o(y)$ represents the cost that depends directly on the offline decisions. The remainder of the cost function is given by the Sample Average Approximation of the expected cost of the subsequent stages. The function $T_o(y, \xi^0_\omega)$ determines the initial state for the online stages, based on the value of y and on the uncertainty revealed at the end of the offline stage (i.e. ξ^0_ω).

Finally, $\mathcal{Y}$ is the feasible space for the offline decision variables y. We make no special assumption on $\mathcal{Y}$, $\mathcal{F}_o(y)$, and $T_o(y, \xi_\omega^0)$, meaning that even when the flattened problem is convex the offline problem may be NP-complete (or harder). Still, the fact that the problem is solved offline makes its complexity less critical.

Online Heuristic Since we assume that the online heuristic can be modeled as a parametric convex optimization problem, we have that:

$$\begin{aligned} &\min f(y, x^k, s^k; \alpha^k) && (\mathbf{PH}) \\ &\text{s.t. } e(y, x^k, s^k) = 0 && (7) \\ &\quad g(y, x^k, s^k) \leq 0 && (8) \end{aligned}$$

where f is the cost function with parameter vector α^k, while e and g are the same constraint functions appearing in **PF**.

Note that the objective function f is not in general the same as the actual cost $\mathcal{F}(y, x^k, s^k)$ incurred at stage k: using a modified cost function is actually a common technique employed by domain experts to control the behavior of a heuristic.

Problem **PH** is general enough to capture heuristics of practical interest, such as shortest link selection in routing, or Priority Rule Based scheduling (aka List Scheduling): in this cases, the constraints define the available actions and the cost function allows to rank them.

3.0.1 RUNNING EXAMPLE ON VRP FOR THE BASELINE

From here we will use a running example on a mini use case (related to a Vehicle Routing Problem) to clarify the methods, their offline/online decisions and the effectiveness of their integration.

We consider a representative instance of a VRP composed by 5 clients $\in C$, 2 vehicles $\in V$ and 1 Depot. Each vehicle must start/end its route from/to the Depot. Each client has a demand $\in D$ and each vehicle is characterized by its capability $\in CP$. Each route (arc) between two clients (nodes) has a cost (travel time) that is represented with a cost matrix M for all routes. Each cell (i.e. c_{ij}) of the cost matrix represents the travel time from client i to client j. These costs are considered as a runtime instance (i.e. the online realization to be solved). The offline decisions (based on scenarios) are represented by the assignment of the clients (based on their demand) to the available vehicles (based on their capability). Each step of the online routing decisions is represented by a route from client i to client j. We will show, based on the chosen offline assignment for each method, the different online routing decisions for all our proposed methods.

We define:

$$C = [C1, C2, C3, C4, C5]$$
$$D = [10, 5, 10, 5, 8]$$
$$V = [V1, V2]$$
$$CP = [30, 20]$$
$$T = \begin{pmatrix} 0 & 2.5 & 2 & 2 & 3 & 2 \\ 3 & 0 & 1 & 4 & 3 & 3 \\ 6 & 1.5 & 0 & 3.5 & 2 & 4 \\ 2.5 & 2 & 1 & 0 & 1 & 4 \\ 4 & 1 & 2 & 3 & 0 & 2 \\ 2 & 3 & 2.5 & 5 & 3 & 0 \end{pmatrix}$$

Each cell of T represents an online parameter cost (travel time) $t_{i,j}$ (e.g. $t_{0,1}$ = 2.5 is the travel time from 0 (the depot) to client 1, $t_{4,2}$ = 2 is the travel time from client 4 to client 2

The offline decisions y are the client assignments to the available vehicles. In the Baseline, these decisions are taken by the two-stage stochastic model. Let us assume that such model assigns clients C1, C2 and C4 to vehicle V1 and clients C3 and C5 to vehicle V2 to satisfy the capability constraints and to minimize the expected total cost over all the scenarios:

$$\begin{pmatrix} \textbf{V1:} & C1, C2, C4 \\ \textbf{V2:} & C3, C5 \end{pmatrix} \tag{9}$$

The online decisions x_k are the online routings that minimize the total cost (travel time) for all vehicles. In the Baseline, these decisions are taken by a greedy heuristic.

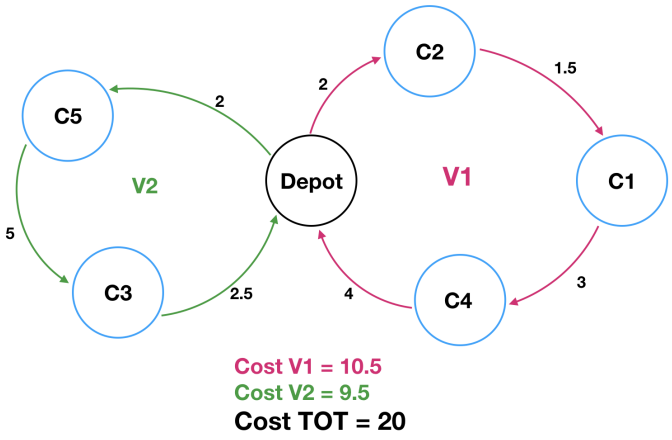

Figure 4: Baseline Offline and Online Integration

We can observe that the online stage of the Baseline method makes online routing decisions with a cost of 10.5 for vehicle 1 (i.e. 2+1.5+3+4 by considering the cost matrix T) and 9.5 for vehicle 2 (i.e. 2+2.5+5 by considering the cost matrix T) for a total cost of 20.

4. Offline/Online Integration Methods

The biggest drawback of the approach from Section 3 is that using the flattened problem to estimate the effect of the offline decision on the future is equivalent to assuming the availability of perfect information. However, the greedy heuristic employed for the online phase is instead completely myopic. *This creates a discrepancy between the estimates made by the offline solver and the capabilities of the online solver*, which intuitively should have an adverse effect on the cost/quality tradeoff on the overall problem.

Such a discrepancy can be addressed by following two strategies. First, we can *improve the online solver by adding some anticipatory capabilities*. Second, we can *make the offline solver explicitly aware of the limitations of the online approach*. Both methods have the effect of bridging the gap between the tools used in the offline and online phase. They are also not mutually exclusive, and in fact one of the approaches proposed here acts in both directions.

Probably the most natural way to improve online decision making consists in replacing the greedy heuristic with a sampling-based anticipatory algorithm: this is the key idea in our ANTICIPATE method (see Section 4.1). However, increasing the computational load of the online phase may not a good idea when stringent time constraints exist. In such a situation, it may be better to improve the greedy heuristic by simply adjusting its parameters. This is the main idea in the TUNING approach: this maintains the efficiency of the original greedy heuristic, at the price of a computationally expensive parameter tuning process, which is however performed offline (see Section 4.2).

Shifting our attention to the offline decision, we can mitigate the discrepancy by translating the online greedy heuristic as a set of constraints, which can be injected in the offline model **PO**. This techniques leads to our ACKNOWLEDGE method (see Section 4.3). Interestingly, we show in Section 4.4 that the approach can be combined with parameter tuning to achieve even deeper integration: this idea is explored in our ACTIVE method (see Figure 5).

4.1 Anticipate

We can derive a sampling-based anticipatory algorithms for the online phase via the same method employed for the offline problem in our baseline, i.e. instantiating **PF** for the remaining stages (and for all samples). Formally, let Ω be a set of scenarios ω for $\xi = (\xi^0, \dots \xi^{n-1})$, and h be the index of the current stage, then we consider:

$$\min \mathcal{F}(y, x^h, s^h) + \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{k=h+1}^{n} \mathcal{F}(y, x^k_\omega, s^k_\omega) \qquad (\mathbf{PA})$$

$$\begin{array}{lll} \text{s.t. Eq. } (7,8) & \text{– } \textit{online problem constraints} \text{ –} & \text{for stage } h \\ \text{Eq. } (2,3) & \text{– } \textit{flattened problem constraints} \text{ –} & \text{for } k > h \\ \text{Eq. } (4) & \text{– } \textit{state transitions} \text{ –} & \text{for } k \geq h, \text{ with } s^h_\omega = s^h \text{ and } x^h_\omega = x^h \end{array}$$

The offline decisions are taken like in the baseline, i.e. by using **PO**. This first approach, referred to as ANTICIPATE, improves the accuracy of the online component at the expense of its solution time. In particular, the need to take into account state transitions may make **PA** NP-hard even if the constraints for each online stage are convex.

PA has the same semantic as the EXPECTATION algorithm, except that this is done by solving a single optimization problem rather than one problem for each scenario and for each possible

Technique	What to improve?	Where is the effort?
Anticipatory Heuristic	Online	Online
Tuning Heuristic	Online	Offline
Heuristic Aware Offline Decisions	Online	Offline

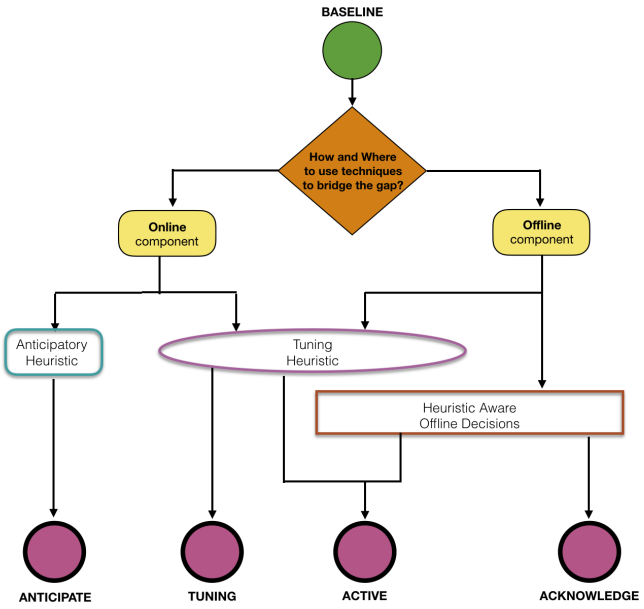

Figure 5: (left) General used techniques. (right) Techniques and component to generate our methods

decision in the current stage. As a main drawback, the problem that our method needs to solve may be considerably larger (as it takes into account all scenarios simultaneously): in many cases, this trait makes our approach less efficient than EXPECTATION (and therefore than CONSENSUS and REGRET).

However, there are two important practical cases where our approach has a substantial advantage. First, when at each stage we have to take multiple decisions (e.g. choosing subsets of items in a knapsack problem) the number of potential alternatives may grow very large. As an example, Chisca, Lombardi, Milano, and O'Sullivan (2018) show that problems such as the online Kidney Exchange (i.e. a multi-stage stochastic problem) have clear scalability issues and their exact solution is a policy tree that recursively specifies the best matching for each step and every possible uncertain outcome. In such a situation, EXPECTATION algorithm may become rather costly (due to the need to enumerate all subsets), while CONSENSUS and REGRET may have difficulties in obtaining a valid estimate of the expected impacts (since costs cannot be readily ascribed to individual items). Second (and more importantly), when the decision space is not enumerable (e.g. for continuous x^k variables), EXPECTATION, REGRET (and even CONSENSUS and AMSAA) cannot be applied directly, while our method is still viable with no modification.

4.1.1 RUNNING EXAMPLE ON VRP FOR ANTICIPATE

Given the same setting of Section 3.0.1, with C, D, V, CP, and T, we show the behavior of ANTICIPATE (i.e. offline decisions, online decisions and total costs):

The offline decisions y are the client assignments and they are decided by the same two-stage stochastic model as the Baseline. Since ANTICIPATE use the same offline step of the Baseline,

it produces the same offline assignments:

$$\begin{pmatrix} \textbf{V1:} & C1, C2, C4 \\ \textbf{V2:} & C3, C5 \end{pmatrix} \tag{10}$$

The online decisions x_k are the routings to minimize the total cost (travel time) for all vehicles (we recall that ANTICIPATE is composed by an *online anticipatory algorithm*:

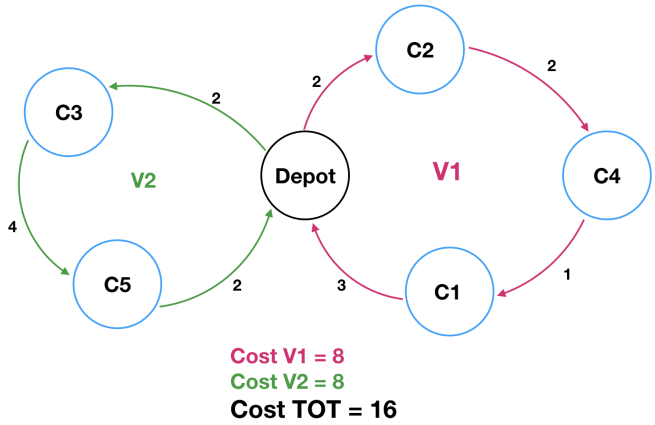

Figure 6: ANTICIPATE Offline and Online Integration

We can notice different online decisions compared to the Baseline (see Figure 4). In particular, V1 moves from C2 to C4 (instead of C1) because it *anticipates* that it will have two minor costs (i.e. $c_{4,1}$ = 1 $c_{1,0}$ = 3). Similar decisions are taken for V2 that moves from C3 to C5 because, even if costs $c_{0,3}$ and $c_{0,5}$ are the same, by choosing C3 first it will have a total lower cost to reach the depot (node 0). This leads to an improvement in the solution quality (i.e. in terms of lower total travel times of 16 compared to 20 of the Baseline) with the same offline assignment.

4.2 Tuning

Our second technique for improving the online decision making consists in applying a parameter tuning phase (computed offline) to the greedy heuristic. In principle, this could be done by any suitable algorithm available from the literature, such as those from López-Ibáñez et al. (2016) or Hutter, Hoos, Leyton-Brown, and Stützle (2009). However, we can take advantage of the convexity of **PH** to tackle the tuning problem in a principled fashion and obtain a guaranteed optimal parameters.

In particular, any decision made a stage k by the heuristic is a global optimum for **PH**. Now, convexity implies that any local minimum must be a global minimum. Local minima can be characterized in terms of the Karush-Kuhn-Tucker (KKT) optimality conditions (Winston, 2004). Essentially, *those conditions give us a set of constraints that must be satisfied by any solution that is compatible with the behavior of the greedy heuristic*. We can exploit this property to formulate the tuning problem as a Mathematical program. In our case, we add the KKT conditions for the greedy heuristic as constraints in the offline model. Once the KKT conditions have been added, the online decisions are forced to take the values that the greedy heuristic would actually assign to them. The

optimality conditions are specified through the use of a Lagrangian Function by introducing a multiplier $\mu_i \geq 0$ for each inequality constraint and a multiplier λ_i for each equality constraint. Then, if x^* is an optimal solution, there are corresponding values of the multipliers $\mu^* = (\mu_1^*, ..., \mu_g^*) \geq 0$ and $\lambda^* = (\lambda_1^*, ..., \lambda_e^*)$ that: 1) cancel out the gradient of the Lagrangian Function, and additionally 2) satisfy the so-called complementary slackness conditions.

As a first step, we need to consider the form of the KKT conditions for **PH** in a given scenario ω. Those are given by:

$$-\nabla_{x_\omega^k} f(y, x_\omega^k, s_\omega^k; \alpha_k) = \sum_{i=1}^{|e|} \lambda_{\omega,i}^k \nabla_{x_\omega^k} e_i + \sum_{i=1}^{|g|} \mu_{\omega,i}^k \nabla_{x_\omega^k} g_i \tag{11}$$

$$\mu_{\omega,i}^k g_i = 0 \qquad \forall i = 1..|g| \tag{12}$$

$$\mu_{\omega,i}^k \geq 0 \qquad \forall i = 1..|g| \tag{13}$$

$$\text{Eq. } (7, 8) \qquad \textit{– online problem constraints –}$$

where, for sake of readability, $f(y, x_\omega^k, s_\omega^k; \alpha_k)$ has been shortened to f, the i-th component (out of $|e|$) of $e(y, x_\omega^k, s_\omega^k)$ to e_i, and the i-th component (out of $|g|$) of $g(y, x_\omega^k, s_\omega^k)$ to g_i. The $\lambda_{\omega,i}^k$ and $\mu_{\omega,i}^k$ variables represent dual multipliers. These are dual multipliers as in classical duality theory (Diewert, 1974): weights associated to the constraints that must satisfy specific conditions in optimal solutions. Eq. (11) corresponds to the gradient cancellation condition, Eq. (12) to complementary slackness, Eq. (13) to dual feasibility ($\lambda_{\omega,i}^k$ is free), and Eq. (7), (8) to primal feasibility. Note that *here we use the heuristic cost function* f, rather than the "real" cost $\mathcal{F}$.

Then, we rely on the KKT conditions to define a model for an additional offline processing step, whose goal is to find the optimal values of the α_k parameters for a given set of scenarios. Such model is given by:

$$\min \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{k=1}^{n} \mathcal{F}(y, x_\omega^k, s_\omega^k) \qquad (\mathbf{PT})$$

$$\text{s.t. Eq. } (2) - (4) \qquad \textit{– flattened problem and state transitions –} \qquad \forall \omega \in \Omega$$

$$\text{Eq. } (5), (6) \qquad \textit{– initial state and feasible space for } y \textit{ –}$$

$$\text{Eq. } (11) - (13) \qquad \textit{– KKT conditions –} \qquad \forall \omega \in \Omega, \forall k = 1 \dots n$$

This is a stochastic two-stage model where the first stage variables are the α_k parameters (appearing in the equations for the KKT conditions), and the recourse actions (i.e. second stage variables) are the decisions x_ω^k that the heuristic would make in the considered scenarios – plus the related states s_ω^k and the $\lambda_{\omega,i}^k$ and $\mu_{\omega,i}^k$ multipliers. The problem goal is to minimize the expected cost over all stages and scenarios.

Solving **PT** yields an optimal parameter vector for the considered scenario set Ω. The offline decisions y are still made using **PO**, while the online decisions are made via **PH**, *with the optimized parameters*. We refer to this method as TUNING. Intuitively, this approach should allow to retain some of the benefits of ANTICIPATE, without increasing the online computational cost. The price to pay is a considerably larger offline cost.

4.2.1 RUNNING EXAMPLE ON VRP FOR TUNING

Given the same setting of Section 3.0.1, with C, D, V, CP, and T, we show the behavior of TUNING (i.e. offline decisions, online decisions and total costs).

As explained above, the offline phase of TUNING is split in two different step: the first part decides the client assignment and the second one produces virtual travel times α_k to guide the online heuristic. The first offline step is the same of the Baseline, while the second one applies the KKT conditions to make parameter tuning. The offline client assignment will be the same of the Baseline but we have the ability to guide online routing decisions with α_k.
The offline decisions y represented below are the client assignments:

$$\begin{pmatrix} \textbf{V1:} & C1, C2, C4 \\ \textbf{V2:} & C3, C5 \end{pmatrix} \tag{14}$$

We can notice the same client assignment of the Baseline (see Figure 4). The virtual travel times α_k are a cost matrix as T composed by virtual cost. We represent (in yellow) in Figure 7 only the α_k elements needed for the TUNING online routing decisions.

The online decisions x_k are always the routings to minimize the total cost (travel time) for all vehicles, and we recall that TUNING is composed by *the same online greedy heuristic as the Baseline)*:

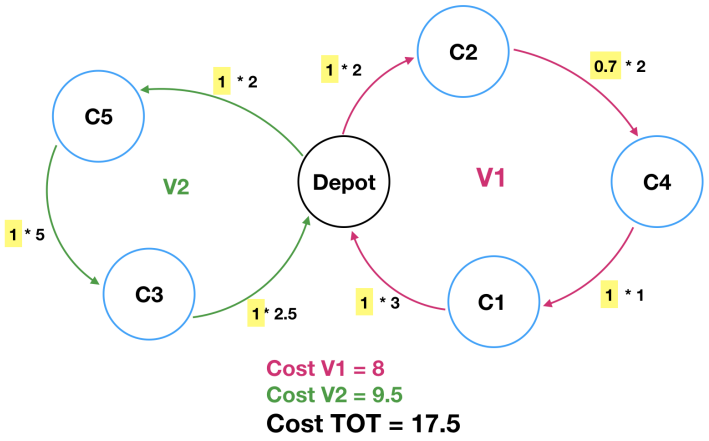

Figure 7: TUNING Offline and Online Integration

We can notice an improvement in the solution quality (i.e. in terms of lower total travel times compared to the Baseline) with the same online heuristic as the Baseline. However, the use of α_k allows the online heuristic to choose different routes in order to obtain a solution relatively close to ANTICIPATE (see Figure 6) but reached with a (fast) online heuristic. Indeed, the α_k are able to guide the online greedy solver to move from C2 to C4 (as ANTICIPATE) by improving the travel cost of V1 (from 10.5 of the Baseline to 8).

4.3 Acknowledge

We now move to explore the second improvement direction: rather than trying to overcome the limitations of the online approach, we make the offline solver aware of the online heuristic.

We achieve this by simply injecting the KKT conditions from Eq. (11) - (13) as constraints in **PO**. Similarly to what done to **PT**, this forces all x^k_ω variables in the offline problem to take the values that would be actually assigned by the heuristic. Overall, we get the following problem:

$$\min f_o(y) + \frac{1}{|\Omega|}\sum_{\omega\in\Omega}\sum_{k=1}^{n}\mathcal{F}(y, x^k_\omega, s^k_\omega) \qquad (\mathbf{PACK})$$

$$\text{s.t. Eq. } (2)-(4) \quad \textit{– flattened problem and state transitions –} \quad \forall\omega\in\Omega$$

$$\text{Eq. } (5), (6) \quad \textit{– initial state and feasible space for } y \textit{ –}$$

$$\text{Eq. } (11)-(13) \quad \textit{– KKT conditions –} \quad \forall\omega\in\Omega, \forall k = 1\dots n$$

Similarly to **PO**, this is a two-stage stochastic program. The first stage variables are the offline decisions y, while the recourse actions are x^k_ω – plus s^k_ω, $\lambda^k_{\omega,i}$, and $\mu^k_{\omega,i}$.

Once an offline decision vector has been found via **PACK**, the online decisions can be made via the original heuristics. We refer to this method as ACKNOWLEDGE. The method achieves integration at the cost of offline solution time, because of the additional variables in **PACK** and the presence of non-linearities in Eq. (12).

4.3.1 RUNNING EXAMPLE ON VRP FOR ACKNOWLEDGE

Given the same setting of Section 3.0.1, with C, D, V, CP, and T, we show the behavior of ACKNOWLEDGE (i.e. offline decisions, online decisions and total costs).

As explained above, the applicability of KKT conditions to the offline phase can produce different offline assignment wrt the Baseline (see Figure 4) and ANTICIPATE (see Figure 6) to guide the online heuristic.
The offline decisions y are the client assignments:

$$\begin{pmatrix} \textbf{V1:} & C1, C3, C4, C5 \\ \textbf{V2:} & C2 \end{pmatrix} \tag{15}$$

We can notice a different offline assignment (compared to all the previous methods based on an offline phase such as the Baseline for the offline assignment) with the use of KKT optimality conditions in the offline phase: C1, C3, C4 and C5 are assigned to V1 and only C2 is assigned to V2. These decisions are different from those of the other models because the offline solver of ACKNOWLEDGE is aware of the limitations of the online greedy heuristic by translating it as a set of constraints injected into the offline model.

The online decisions x_k are the routings to minimize the total cost (travel time) for all vehicles (we recall that ACKNOWLEDGE is composed by the same greedy heuristic of the Baseline in the online phase:

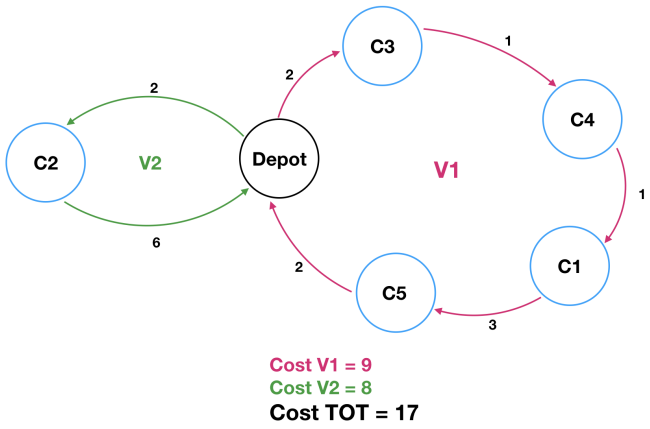

Figure 8: ACKNOWLEDGE Offline and Online Integration

We can notice in Figure 7 an improvement in the solution quality (i.e. in terms of lower total travel times) with the same online heuristic but with a different offline assignment wrt the Baseline. Indeed, since the offline assignment is made by a solver aware of the limitation of the following online greedy heuristic, the offline assignment allows the greedy heuristic to improve the total travel time by always choosing the lower arc.

4.4 Active

Finally, we can combine these two methods to obtain the ACTIVE method that is composed by an offline part with KKT conditions that optimizes the offline decisions y and α^k (i.e. **PACT**).

$$\min f_o(y) + \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{k=1}^{n} \mathcal{F}(y, x^k_\omega, s^k_\omega) \qquad (\textbf{PACT})$$

$$\begin{aligned}
&\text{s.t. Eq. } (2) - (4) && \textit{– flattened problem and state transitions –} && \forall \omega \in \Omega \\
&\quad \text{Eq. } (5), (6) && \textit{– initial state and feasible space for } y \textit{ –} && \\
&\quad \text{Eq. } (11) - (13) && \textit{– KKT conditions –} && \forall \omega \in \Omega, \forall k = 1 \dots n
\end{aligned}$$

The decision variables of **PACT** are, in this case, y, x^k_ω, s^k_ω, $\lambda^k_{\omega,i}$, $\mu^k_{\omega,i}$ and crucially α^k. The online decisions are then taken using the original heuristics, but its behavior will be affected also in this case by the "parameter schedule" $\alpha^1, \dots \alpha^n$ produced by solving **PACT**. The difference in this case is that y and α^k are both optimized at the same time and by considering the KKT. The computational time will be higher but with, hopefully, better solution quality to steer online heuristic behavior.

4.4.1 RUNNING EXAMPLE ON VRP FOR ACTIVE

Given the same setting of Section 3.0.1, with C, D, V, CP, and T, we show the behavior of ACTIVE (i.e. offline decisions, online decisions and total costs).

As explained above, the applicability of KKT conditions to the offline phase can produce different offline assignment wrt the Baseline and ANTICIPATE. Moreover, in the same offline phase, ACTIVE produces also virtual travel times α_k (in yellow) to guide the online heuristic. Differently from TUNING, the offline is phase not split in two different steps but the client assignment and the virtual travel times are optimized in one offline step.
The offline decisions are the client assignments:

$$\begin{pmatrix} \textbf{V1:} & C1, C3, C4, C5 \\ \textbf{V2:} & C2 \end{pmatrix} \tag{16}$$

We can notice the same offline assignment as ACKNOWLEDGE (see Figure 8).

The online decisions x_k are the online routings to minimize the total cost (travel time) for all vehicles (we recall that ACTIVE is composed by an efficient greedy heuristic in the online phase guided also by the α_k parameters):

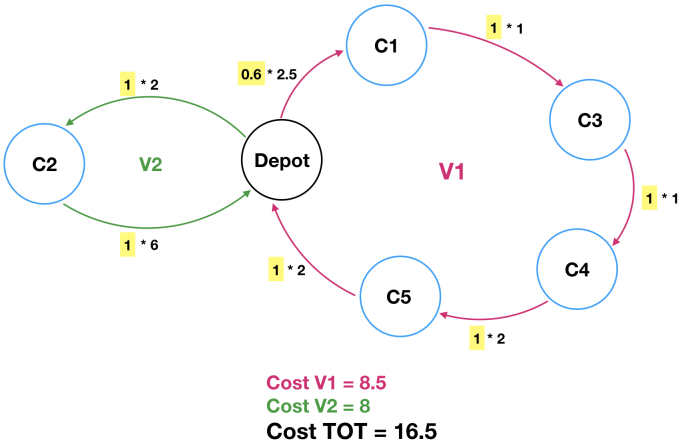

Figure 9: ACTIVE Offline and Online Integration

We can notice an improvement in the solution quality (i.e. in terms of lower total travel times) with the same online heuristic w.r.t. the Baseline, and ACKNOWLEDGE with the same offline client assignment. In particular, differently from ACKNOWLEDGE, ACTIVE chooses a different routing for V1 due to the α virtual travel times (e.g. C1 instead of C3 as first served client). This online routing allows to reach a cost very close to ANTICIPATE (i.e. costTOT = 16.5 w.r.t. costTOT = 16) but with a fast greedy heuristic in the online phase (instead of an online anticipatory algorithm).

4.5 Wrap-Up

In this paper, we start from the observation that many practical applications require to make interdependent offline and online decisions. The simplest and most common approach to tackle such problems is to deal with the offline and online phase separately, respectively (e.g.) via a sampling-based method and a heuristic: we consider these methods as components of our baseline. However,

we will show that substantial improvements can be obtained by treating the two phases in an integrated fashion.

We propose four methods to improve the baseline in different directions, each altering either the offline or the online component of the solution process, so that the two play better together. Our methods are applicable provided that some simple but important assumptions are satisfied: 1) the uncertainty is exogenous; 2) in the baseline, a two-stage stochastic optimization model is used for the offline phase; 3) in the baseline, the online heuristic can be stated as convex optimization problem.

Selecting the suitable technique requires to consider the available time constraints for all offline and online decisions to use the most suitable method. Generally, we can define some basic guidelines on time constraints, all based on the assumption of reaching high quality solutions: 1) if the problem does not present stringent time constraints in the online phase, ANTICIPATE is the recommended method; 2) if the problem presents stringent time constraints in the online phase and no time limits in the offline phase, ACTIVE is the recommended method; 3) if the problem presents stringent time constraints in the online phase and time limits in the offline phase, we can choose between two methods based on the more or less stringent time limits in the offline phase (i.e. TUNING is slightly faster in the offline phase, with a slight loss of solution quality compared to ACKNOWLEDGE). We believe our techniques represent a significant step toward integrated offline/online optimization in complex systems.

Figure 10 summarizes the design of our methods, highlighting the techniques used in each phase, their decision variables and the output:

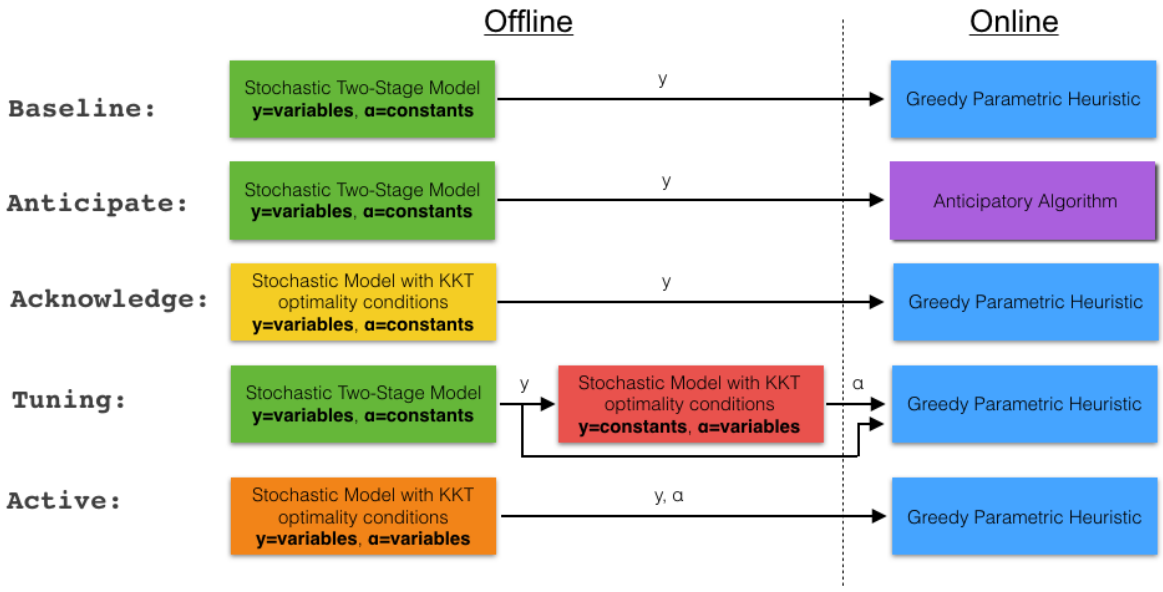

Figure 10: Proposed methods for Offline and Online integration

The following sections show the case studies (i.e. VPP system and VRP) that we used to ground and test our methods.

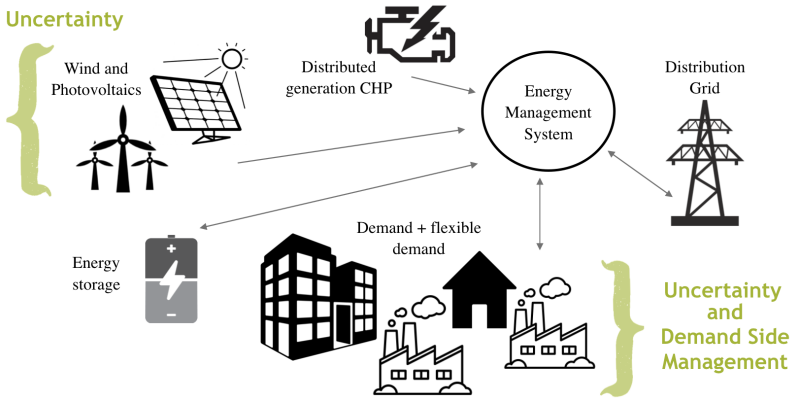

Figure 11: A typical Virtual Power Plant

5. Virtual Power Plant Case Study

In this section we present our first case study, an energy management system, that was originally considered in De Filippo, Lombardi, Milano, and Borghetti (2017): since it features continuous online decision variables, it is not amenable to existing approaches such as EXPECTATION or REGRET.

5.1 Energy Management System

The progressive shift towards decentralized generation in power distribution networks has made the problem of optimal Distributed Energy Resources (DERs) operation increasingly constrained. This is due to the integration of flexible (deterministic) energy systems with the strong penetration of (uncontrollable and stochastic) Renewable Energy Sources (RES). The integration of these resources into power system operation requires a major change in the current network control structure. This challenge can be met by using the Virtual Power Plant (VPP) concept, which is based on the idea of aggregating the capacity of many DERs, (i.e. generation, storage, or demand) to create a single operating profile and manage the uncertainty. A typical VPP is a large plant with high (partially shiftable) electric and thermal loads, renewable and Combined Heat and Power (CHP) generators and electric and thermal storages (see Figure 11).

Making decisions under uncertainty pervades the planning and operation of energy systems (Wallace & Fleten, 2003). Optimization techniques have a long tradition in supporting planning and operational decisions in the energy sector, but the recent literature highlights the need for increasing both the scope and the granularity of the decisions, including new factors like distributed generation by renewable sources and smart grids (Morales et al., 2013).

Both the most popular methods to deal with uncertainty in mathematical programming (i.e. robust optimization and stochastic programming) have been widely applied in energy systems (Reddy, Sandeep, & Jung, 2017; Zhou et al., 2013; Jurković, Pandšić, & Kuzle, 2015). One of the most used assumption is that the distribution of future uncertainty is available for sampling, e.g. thanks to historical data and/or predictive models. In particular, the assumption that the distribution of future (exogenous) uncertainty is independent of current decisions is present in a variety of applications (Van Hentenryck & Bent, 2009).

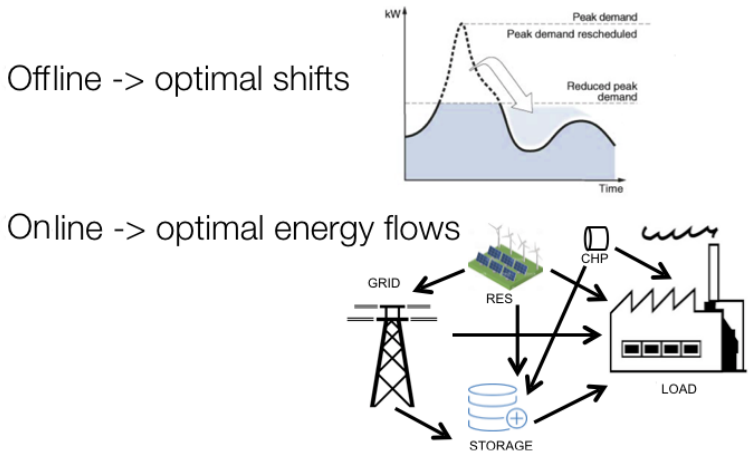

Figure 12: Offline/Online Decision Making in VPP

We consider a VPP Energy Management System (EMS) (see, e.g., Morales et al., 2013) with partially shiftable loads, renewable energy generators, storage systems, and grid-connection. The goal is to decide the minimum-cost energy flows at each online stage (see, e.g., Clavier et al., 2015). The uncertainty stems from uncontrollable deviations from the planned shifts and from the presence of Renewable Energy Sources (RES) (see, e.g., Palma-Behnke et al., 2011; Bai, Miao, Ran, & Ye, 2015). We assume that the RES production forecast is good enough that its error in each stage can be considered an independent random variable, that is the most common assumption in literature (Palma-Behnke et al., 2011; Bai et al., 2015). This is also a reasonable hypothesis, provided that the predictor is good enough, and this allows us to define our uncertainty ranges based on confidence intervals (Gamou, Yokoyama, & Ito, 2002). Based on actual energy prices and on the availability of DERs, the EMS decides: 1) how much energy should be produced; 2) which generators should be used for the required energy; 3) whether the surplus energy should be stored or sold to the energy market; 4) the load shifts planned offline. Optimizing the use of energy can lead to significant economic benefits, and improve the efficiency and stability of the electric system (see, e.g., Palma-Behnke et al., 2011). Unlike in most of the existing literature, we acknowledge that in many practical cases, *the load shifts can be planned offline*, while the energy balance should be maintained online by managing energy flows among the grid, the renewable and traditional generators, and the storage systems. This is depicted in Section 2.3. Intuitively, handling these two phases in an integrated fashion should lead to some benefits, thus making the VPP EMS a good candidate for grounding our approach (see Figure 12): integrating the two is likely to lead to better results, as it is often the case when a complex problem is partitioned. Whenever distinct offline and online phases are present, as in this case study, a tighter integration can lead to substantial improvements in terms of both solution quality and computational costs. The idea is that by injecting knowledge of offline load decisions in a greedy online simulator, we can achieve better solution quality by maintaining a fast (greedy) heuristic in the online phase.

5.1.1 BASELINE OFFLINE PROBLEM

Generally, a typical distributed energy system like the one considered in this first case study is composed by several distributed nodes (our set represented by G). Each of these nodes (i.e. g) can

be a generator (traditional generator such as a CHP or renewable generator such as a photovoltaic system); a battery-based system (storage or electric vehicles); or the external grid which, in these cases, can be associated with the energy market for buying and selling energy. All these generators have an associated cost (e.g. the diesel cost for the CHP generator) and this is the meaning of c_g^k (i.e. the generation cost for each stage k and for each generator g). The state of the system for each stage k is represented by the charge state of the storage system γ^k. The system tries to minimize the total costs obtained by the energy flow from each generator g (i.e. the online decisions x_g^k) associated to the relative cost c_g^k. The system tries also to maintain the power balance in each step k by satisfying the requested (offline) shifted load $\tilde{L}^k$ with the available resources. The y^k variable represents the (offline planned) shift from the estimated load. Eq. (23) ensures that the shifts respect a local balance. The initial battery charge γ_ω^0 is identical for all scenarios. n is the number of stages, and x_g^k represents the flow from g to the VPP (if positive) or in the reverse direction (if negative). All flows must respect the physical bounds $\underline{x}_g$ and $\overline{x}_g$. The state variables are the RES energy flow x_1^k, the load to be satisfied $\tilde{L}^k$, and the battery charge γ^k. The battery upper limit is Γ and η is the charging efficiency.

The offline problem is modeled via Mixed Integer Programming (MILP) and it is given by:

$$\min \frac{1}{|\Omega|} \sum_{\omega\in\Omega} \sum_{g\in G} \sum_{k=1}^{n} c_g^k x_{g,\omega}^k \qquad (\mathbf{P1.2})$$

$$\text{s.t. } \tilde{L}_\omega^k = \sum_{g\in G} x_{g,\omega}^k \qquad \forall\omega\in\Omega, \forall k = 1,\dots n \tag{17}$$

$$\underline{x}_g \le x_{g,\omega}^k \le \overline{x}_g \qquad \forall\omega\in\Omega, \forall k = 1,\dots n \tag{18}$$

$$0 \le \gamma_\omega^k \le \Gamma \qquad \forall k = 1,\dots n \tag{19}$$

$$\gamma_\omega^{k+1} = \gamma_\omega^k + \eta x_{0,\omega}^k \quad \forall\omega\in\Omega, \forall k = 1,\dots n-1 \tag{20}$$

$$x_{1,\omega}^{k+1} = \hat{R}_k + \xi_{R,\omega}^k \qquad \forall\omega\in\Omega, \forall k = 1,\dots n \tag{21}$$

$$\tilde{L}_\omega^{k+1} = \hat{L}_k + y_k + \xi_{L,\omega}^k \quad \forall\omega\in\Omega, \forall k = 1,\dots n \tag{22}$$

$$\sum_{k=t}^{t+m} y_k = 0 \qquad \forall t = 1,\dots n-m \tag{23}$$

$$\underline{y}^k \le y_k \le \overline{y}^k \qquad \forall k = 1,\dots n \tag{24}$$

where Eq.(17) $-$ (22) define the flattened problem, and Eq. (23) $-$ (24) the feasible space for the offline variables y. In particular, Eq.(20) $-$ (22) represent the transition function, where $\hat{R}_k$ and $\hat{L}^k$ are the estimated RES production and load, and ξ_R^k and ξ_L^k are the corresponding errors (random variables). We assume that the errors follow roughly a Normal distribution $N(0,\sigma^2)$, and that the variance σ^2 is such that the 95% confidence interval corresponds to $\pm 20\%$ of the estimated value (Gamou et al., 2002).

5.1.2 BASELINE HEURISTIC

Based on the shifts produced by the offline step, and taking into account the uncertainty, the online heuristic minimizes the operational cost and covers the energy demand by manipulating flows between nodes in $g \in G$. We assume the index 0 refers to the storage system and index 1 to the RES

generators. The stages represent periods long enough to treat the corresponding flow decisions as independent. The heuristic can be formulated as an LP model:

$$\min \sum_{k=1}^{n} \sum_{g \in G} c_g^k x_g^k \qquad (\mathbf{P1.1})$$

$$\text{s.t. } \tilde{L}^k = \sum_{g \in G} x_g^k \tag{25}$$

$$0 \leq \gamma_k + \eta x_0^k \leq \Gamma \tag{26}$$

$$\underline{x}_g \leq x_g^k \leq \overline{x}_g \tag{27}$$

where n is the number of online stages, and the flow costs c_g^k correspond to the problem parameters α^k in **PH**.

5.1.3 INSTANTIATING ANTICIPATE FOR THE VPP

A model for the ANTICIPATE approach can be obtained by applying in an almost straightforward fashion the definitions from Section 4.1:

$$\min \sum_{g \in G} c_g^h x_g^h + \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{k=h+1}^{n} \sum_{g \in G} c_g^k x_{g,\omega}^k \qquad (\mathbf{P1.3})$$

s.t. Eq. (25) – (27) *– online problem constraints –*

Eq. (17) – (19) $\forall k > h$ *– flattened problem constraints –*

Eq. (20) – (22) $\forall k \geq h$, with $s_\omega^h = s^h$ and $x_\omega^h = x^h$ *– state transitions –*

Note that **P1.3**, although potentially large, is a Linear Program and can be solved in polynomial time.

5.1.4 INSTANTIATING ACTIVE/TUNING/ACKNOWLEDGE FOR THE VPP

We start by formulating the KKT conditions for the online heuristic in a single scenario, thus obtaining:

$$-c_g^k = \lambda_\omega^k + \mu_{g,\omega}^k - \nu_{g,\omega}^k \qquad \forall g \in G \tag{28}$$

$$\mu_{g,\omega}^k (x_{g,\omega}^k + \overline{x}_g) = 0 \qquad \forall g \in G \tag{29}$$

$$\nu_{i,\omega}^k (\underline{x}_g - x_{g,\omega}^t) = 0 \qquad \forall g \in G \tag{30}$$

$$\hat{\mu}_\omega^k (\eta x_{0,\omega}^k + \gamma^k - \Gamma) = 0 \tag{31}$$

$$\hat{\nu}_\omega^k (\eta x_{0,\omega}^k + \gamma^k) = 0 \tag{32}$$

$$\mu_{g,\omega}^k, \nu_{g,\omega}^k \geq 0 \qquad \forall g \in G \tag{33}$$

$$\hat{\mu}_\omega^k, \hat{\nu}_\omega^k \geq 0 \tag{34}$$

where $\mu_{g,\omega}^k$ and $\nu_{g,\omega}^k$ are the multipliers associated to the physical flow bounds, while $\hat{\mu}_\omega^k$ and $\hat{\nu}_\omega^k$ are associated to the battery capacity bounds. The multiplier λ_ω^k is associated to the balancing

constraint, i.e. Eq. (25), and can be eliminated with a few algebraic transformations. Injecting the conditions in the offline model yields:

$$\min \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{g \in G} \sum_{k=1}^{n} c_g^k x_{g,\omega}^k \qquad (\mathbf{P1.4})$$

$$\text{s.t. Eq. } (17) - (24) \qquad \textit{– offline problem constraints –}$$

$$\text{Eq. } (28) - (34) \qquad \forall \omega \in \Omega, \forall k = 1, \dots n \qquad \textit{– KKT conditions –}$$

where the decision variables are y^k, $x_{g,\omega}^k$, $\mu_{g,\omega}^k$, $\nu_{g,\omega}^k$, $\hat{\mu}_\omega^k$, $\hat{\nu}_\omega^k$, based on the considered method. To those, we add the cost c_0^k associated to the flow between the VPP and the storage system (the only parameter that we allow the solver to adjust). Normally, since the online (myopic) solver has the ability to sell energy on the market and storing energy has no profit, it ends up in always selling unless the virtual cost is employed.

We recall that in ACKNOWLEDGE we consider y^k as decision variables and c_0^k as constant parameters using **P1.4** as offline phase; in TUNING we use **P1.2** to solve the y^k variables and then we use **P1.4** for c_0^k (i.e. our offline phase is divided in two parts); finally in ACTIVE we use **P1.4** with both y^k and c_0^k as decision variables of the *same offline phase*. As a side effect, the naive **P1.1** heuristic will always choose to sell the surplus energy. ACTIVE allows the offline solver to associate a "virtual cost" to storing energy, which enables addressing the original limitation at no online computational cost. In details, the virtual cost is a parameter and in particular a cost associated to the use of the storage system. We called it "virtual" because it does not correspond to a real economic cost: it is introduced to encourage or discourage the use of the storage system. In the energy system, since the online solver has the ability to sell energy on the market and storing energy has no profit, it ends up always selling unless the virtual cost is employed. By using the offline phase to choose different virtual costs for each online stage, we are able to "guide" the online greedy heuristic to store energy (e.g.) in view of an increase in energy market prices in the future stages. By changing these scores, an offline method can effectively alter (hopefully for the better) the behavior of the online heuristic.

5.2 Results

We performed an experimentation to compare the solution quality and run times of our methods. As references for comparison we use the baseline approach, plus an optimal solver operating under perfect information.

5.2.1 EXPERIMENTAL SETUP

Our methods are evaluated over different uncertainty realizations, obtained by sampling the random variables for the loads and RES generation. We consider a sample of 100 realizations for six different instances of each problem. We then run each approach on each realization and measure the cost and run time. The scenarios in our models, conversely, are not sampled, but programmatically chosen: we consider four "extreme" scenarios where (resp.) the load and the RES generation are at low/high values. Concerning the uncertainty in our models, we deal with: 1) uncertainty via sampling (i.e. we sample realizations by assuming that the error for our forecasts can be considered an independent random variable); 2) a number of "edge" scenarios, thus making our model somewhat close to robust optimization. The problem has 24 online stages.

We solve our LPs and MILPs using Gurobi, while for the non-linear problems we use BARON via the GAMS modeling system on the Neos server for optimization. The time limit is 100 seconds and we use data from two public datasets to define problem instances: a residential plant (Espinosa & Ochoa, 2015) [1] with only PV energy production for renewable sources and an industrial plant [2] with wind and PV production. We modify these datasets to obtain use cases for testing our models (see Table 1):1) Use Case 1 (RB) is the baseline residential dataset (with no modifications); 2) RR is the residential dataset with an increase of 15% (for each t) of the renewable production (i.e. PV production in this dataset); 3) RP is dataset RB where the market prices are different for the sale/purchase of energy from/to the grid with an increase of 10% (for each); 4) IB is the industrial dataset (with no modifications), differently from RB, this dataset presents also eolic renewable production; 5) in IR we increase the renewable production as in RR (by increasing also eolic production by a 10% for each t); 6) in IP we consider RB with different market prices exactly as in RP.

Methodologies for the estimation of hourly global solar radiation have been proposed by many researchers and in this work, we consider as a prediction the average hourly global solar radiation from Kaplanis and Kaplani (2007) and we use assumption for wind prediction from Hodge et al. (2012). We then assume that the prediction errors in each timestamp can be modeled again as random variables. Specifically, we assume normally distributed variables with a variance such that the 95% confidence interval corresponds to $\pm 10\%$ of the prediction value. We assume physical bounds on CHP due to its Electrical Capability based on real generation data (Bai et al., 2015; Espinosa & Ochoa, 2015). The initial battery states and the efficiency values are based on real generation data (Bai et al., 2015; Espinosa & Ochoa, 2015) and we assume there are physical bounds for storage system based on real data (Bai et al., 2015; Espinosa & Ochoa, 2015).

Load demand	Baseline dataset	Renewable peak	Different market Prices
Residential	RB	RR	RP
Industrial	IB	IR	IP

Table 1: Different use cases

5.2.2 RESULT DISCUSSION

In Tables 2 and 3 we show the average costs and run times over the 100 input realizations for each approach. Online times refer to the sum of the stages. We show the solution quality (cost(K€)) for each method and for the baseline and the oracle. In the other columns, we show (respectively) the computation total time and then the two computation times for the offline and online optimization. This information allows to understand the most suitable method to use by evaluating the cost/quality tradeoff for each considered case study. The baseline model (being an LP) appears to be rather efficient in terms of computation time, but yields solutions of limited quality. The ANTICIPATE method comes much closer to the oracle solver, at the cost of a higher, but still reasonable, online run time. The ACTIVE method incurs substantially larger offline solution times, but it manages to beat or match the ANTICIPATE solution quality by making use of the original, straightforward,

1. Available at `https://www.enwl.co.uk/lvns`)
2. Available at `https://data.lab.fiware.org/dataset/smart_energy_data-_aachen_cologne_virtual_power_plant`

Instance	Oracle	Baseline				ANTICIPATE			
	c(K€)	c(K€)	tot t(s)	offline t(s)	online t(s)	c(K€)	tot t(s)	offline t(s)	online t(s)
RB	**331.36**	404.62	0.962	0.184	0.778	**342.06**	5.195	0.184	5.011
RR	**247.21**	311.14	0.962	0.190	0.772	**265.32**	5.207	0.190	5.017
RP	**393.81**	462.57	0.960	0.185	0.775	404.32	5.194	0.185	5.009
IB	**798.38**	923.24	1.185	0.346	0.839	822.24	5.776	0.346	5.430
IR	**565.60**	684.19	1.173	0.341	0.832	580.17	5.764	0.341	5.423
IP	**856.95**	984.90	1.183	0.348	0.835	**874.58**	5.768	0.348	5.420

Table 2: Cost/computation time for the Oracle, Baseline and ANTICIPATE

Instance	TUNING				ACKNOWLEDGE				ACTIVE			
	c(K€)	tot t(s)	offline t(s)	online t(s)	c(K€)	tot t(s)	offline t(s)	online t(s)	c(K€)	tot t(s)	offline t(s)	online t(s)
RB	391.18	11.758	10.980	0.778	382.44	11.231	10.453	0.778	346.60	28.662	27.884	0.778
RR	294.75	10.245	9.473	0.772	297.77	10.768	9.996	0.772	266.80	32.764	31.992	0.772
RP	422.92	11.996	11.221	0.775	435.11	11.719	10.944	0.775	**408.72**	31.547	30.772	0.775
IB	883.99	14.305	13.466	0.839	894.33	16.276	15.437	0.839	**817.11**	39.752	38.913	0.839
IR	609.81	13.826	12.994	0.832	625.83	17.609	16.777	0.832	**577.93**	40.016	39.184	0.832
IP	901.27	16.278	15.443	0.835	950.81	16.603	15.768	0.835	888.76	38.612	37.777	0.835

Table 3: Cost/computation time for TUNING, ACKNOWLEDGE and ACTIVE

online heuristic. Tables 2 and 3 also show a comparison among the computational times of all the proposed method to help understanding the potential of each method both in terms of offline and online computational cost.

We show the average values of each hourly optimized flow over the 100 realizations for each proposed model in instance RR. We use a barchart representation where the positive values represent all the energy generation flows (energy production by the internal sources of the VPP, thus outgoing from the VPP), and the negative values represent the energy stored in the storage system or sold to the grid, thus incoming to the VPP. Each barchart shows the average values over the 100 realizations for a tested model (e.g. the first barchart in Figure 13 shows the results of the Oracle (on the left) and of the Baseline model (on the right). Each color represents a component of the VPP, the blue line is the shifted load to be satisfied from the VPP, and the red dot line represents the market energy prices. We remember that the shifted load is optimized in the offline phase in the Demand Side Management step. With this representation, it is possible to see when we have a surplus of energy PV production (e.g. at midday, as expected) and we can note that the system tends to sell energy to the grid or (if it is able to have a certain degree of anticipation) to store energy into the storage system. Moreover, with this representation, we can clearly observe, for each timestamp, the contribution of each component to satisfy the load requested to the system. The aim is also to show the different (sometimes absent) use of the storage system in models with no degree of anticipation. In Figure 13, the limits of using a non anticipatory algorithm, compared (e.g.) to the Oracle optimization, are clearly represented: it is not possible to acquire energy from the grid in advance (i.e. when the cost is lower) and/or to sell energy to the grid in periods of highest price on the market or when more energy is available from renewable sources. In order to stress this point, we also add (Figure 16) that represents only the flows (the incoming and outgoing flows) of the storage system in the two most representative cases (i.e. Baseline and ACTIVE).

Moreover the exchange of energy with the storage system is almost never used, i.e. to store RES energy. In Figure 15, it is possible to see that, near the peak of renewable energy production, the ACTIVE model accumulates energy in the storage and uses in a more balanced way the energy present in the storage system compared to the baseline model represented in Figure 13 (that never uses the storage system). Furthermore, still looking at Figure 15, it can be seen that ANTICIPATE

has peaks of energy sold on the network near the increase in electricity prices on the market. In Figure 14 and Figure 15 it is possible to notice the more consistent use of the storage system. We can see that, by optimizing the virtual storage cost in the offline phase, we can improve solution quality in term of cost (see Tables 2 and 3) by using the storage system. Since the online solver has the ability to sell energy on the market, and storing energy has no profit, it ends up in always selling unless the virtual cost is employed.

Figure 16 shows the comparison between the optimized α in the two methods ACTIVE and TUNING. Moreover, we show also the market price trend to compare the trends of the two methods. It is interesting to notice how both the methods try to increase or decrease the virtual cost of the storage to promote it in anticipation of future increases of energy price. In Figure 17 it is possible to observe the different use of the storage system due to the optimized parameters injected in the online solver to guide the heuristic decisions. We recall, as explained for Figure 13, that the storage *flow* is negative (for convenience) when the storage system is in charging mode, while it is positive when the storage system is in discharging mode. Since the online (baseline) solver has the ability to sell energy on the market and storing energy has no profit, it ends up in always selling unless the virtual cost is employed (ACTIVE).

6. Vehicle Routing Problem Case Study

The second use case (a Vehicle Routing Problem variant) is meant to provide a realistic, dramatically different, example of how the methods can be instantiated: it features discrete online decisions, and allows a quality comparison with classical algorithms because in such cases ANTICIPATE leads to the same results as EXPECTATION (although with different solution times). The case studies have been chosen to show that our methods work with both discrete and numerical decision variables.

6.1 VRP Problem Description

We consider a variant of the Capacitated VRP with uncertain travel times (see, e.g., Toth & Vigo, 2002; Bertsimas & Simchi-Levi, 1996; Lee, Lee, & Park, 2012; Tacs, Dellaert, Van Woensel, & De Kok, 2013). The problem consists in establishing the paths of a set of vehicles to serve a set of customers. All vehicles have a finite capacity, and customers have a known demand and can be

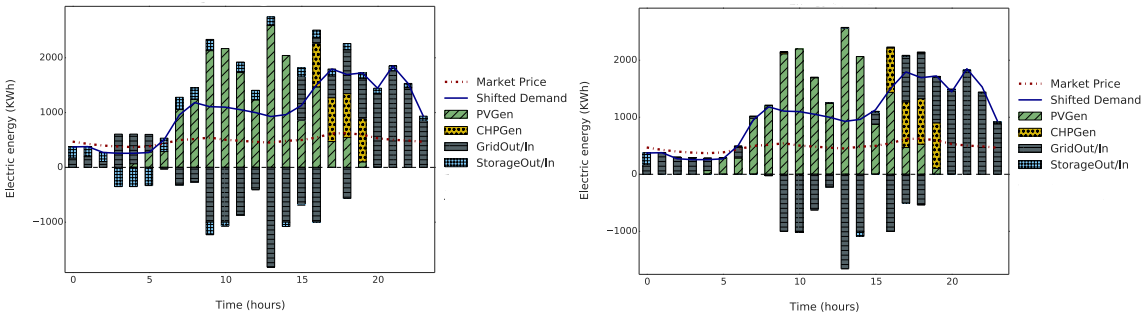

Figure 13: Oracle (left) and Baseline (right) optimal power flows for instance RR

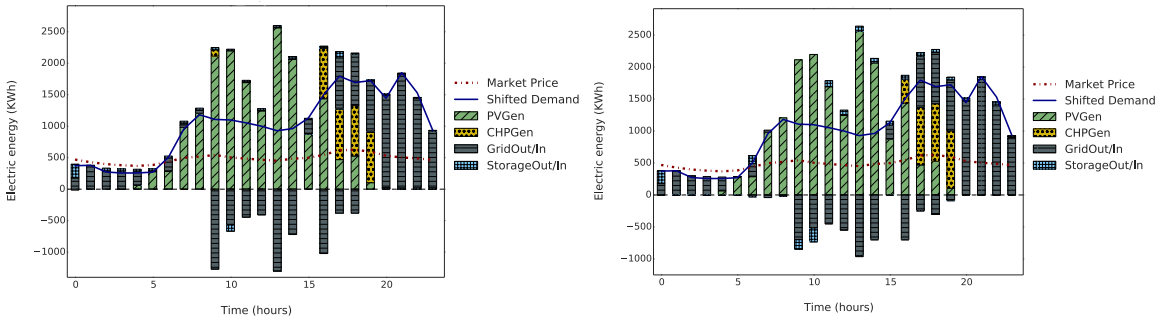

Figure 14: ACKNOWLEDGE (left) and TUNING (right) optimal power flows for instance RR

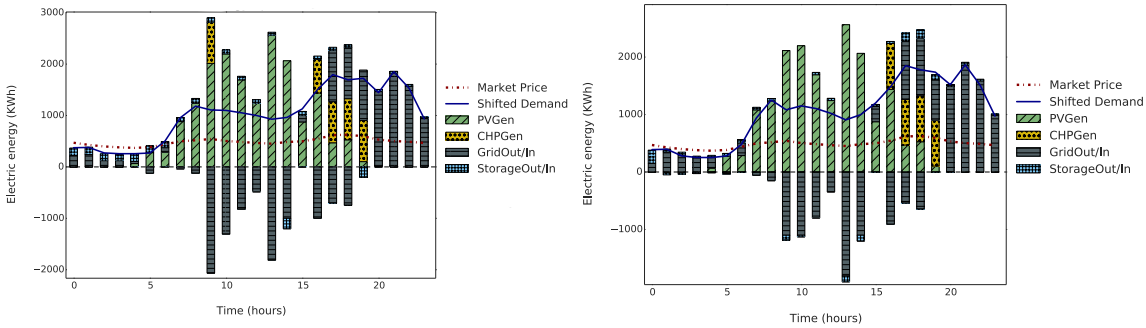

Figure 15: (left) ANTICIPATE and ACTIVE (right) optimal power flows for instance RR

visited by a single vehicle. There are n fully connected customers/nodes, with node 0 being the (single) depot.

Customer assignments must be done offline, while the vehicle routes are chosen online. In our experimental setup, whenever a node is reached, its binary "state" becomes known, and with that the (uniform) distributions followed by the travel times of all its outgoing arcs.

Formally, this translates in bi-modally distributed, statistically dependent, travel times. The objective is to minimize the total travel time.

6.1.1 BASELINE HEURISTIC AND TRANSITION FUNCTION

The online heuristic consists in simply picking the next node to be visited based on the outgoing arc with the shortest travel time. This can be modeled also as a simple Integer Program. Let h be the

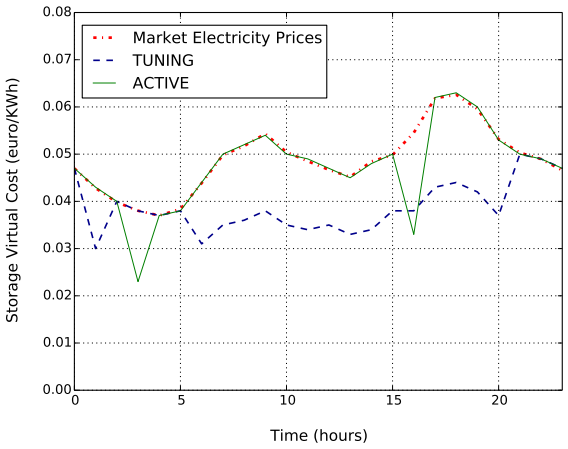

Figure 16: Market prices and optimized alpha in ACTIVE and TUNING for instance RB

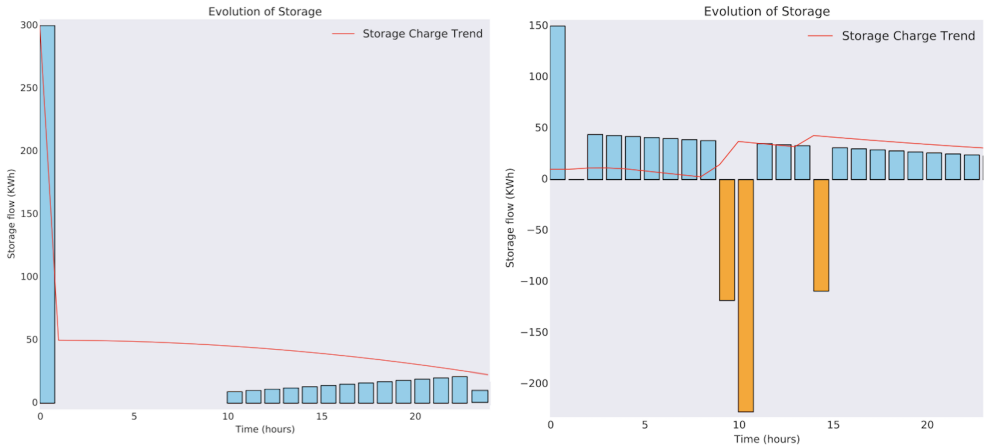

Figure 17: Baseline (left) and ACTIVE (right) online storage flow for instance RB

current node, then we have:

$$\min \sum_{j \in V_h} c_{hj} x_{hj} \qquad (\mathbf{P2.1})$$

$$\text{s.t.} \sum_{j \in V_h} x_{h,j} = 1 \qquad (35)$$

$$x_{h,j} \in \{0,1\} \qquad \forall j \in V \qquad (36)$$

where $x_{hj} = 1$ iff we choose to move from h to j, V_h is the set of nodes that still needs to be visited (and it always include the depot), and the travel times c_{hj} are the heuristic parameters: they represent the cost associated to each route in terms of travel times. In some cases we refer to them as virtual, since they may be modified by the offline solver. **P2.1** does not apparently satisfy our assumptions, due to the integer variables. However, *its LP relaxation has always an integer solution, banning degenerate cases* (i.e. arcs with the same cost). We can therefore relax the integrality requirement

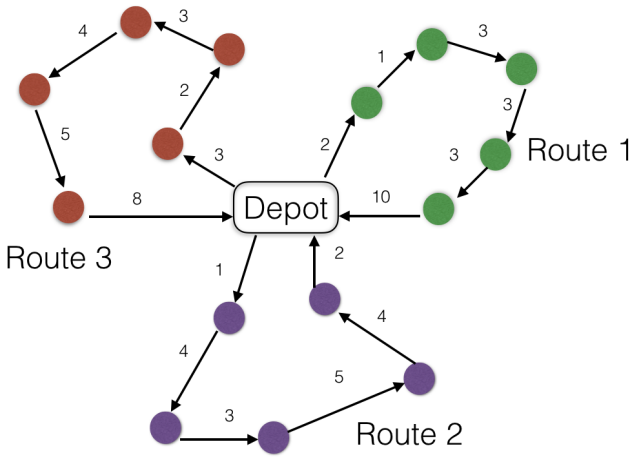

Figure 18: An Example of Vehicle Routing Problem

without loss of generality. The transition function is given by:

$$V_{h^*} = V_h \setminus \{h^*\} \tag{37}$$

$$c_{h^*,j} = \xi_{h^*,j} \tag{38}$$

where h^* is the index of the next node selected by the heuristic and $\xi_{h^*,j}$ is the travel time from h^* to j (a random variable). Note also that in this case *the index of the online stage is implicitly given by* h. We take advantage of this and reduce the notation clutter by moving the ω index to apex position.

6.1.2 BASELINE OFFLINE PROBLEM

We tackle the offline problem via Mixed Integer Linear Programming, which forbids us to directly embed the non-linear Eq. (37) in the model. In practice, however, the equation states that 1) each vehicle should serve only its assigned customers, and 2) the visit should form a single loop. Both are well known VRP constraints and can be linearized. In the VRP model, K is the set of vehicles indexed with k, V is set of arrival nodes for each route (with index j), while in the online problem V_h is the set of nodes that still needs to be visited (with additional index h for representing the current node). y_{kj} is a binary variable (between k and j nodes) that represents the offline assignment decisions: it is $y_{kj} = 1$ iff node j should be visited by vehicle k. C_k is the upper bound for the capacity of each vehicle k, and q_i is the quantity requested by each customer i. In particular, we use

the model:

$$\min \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{k \in K} \sum_{i,j \in V} \xi_{i,j}^{\omega} x_{k,i,j}^{\omega} \qquad (\mathbf{P2.2})$$

$$\text{s.t.} \sum_{j \in V} x_{k,i,j}^{\omega} = y_{k,i} \qquad \forall k \in K, \forall i \in V \tag{39}$$

$$\sum_{i \in V} x_{k,i,j}^{\omega} = y_{k,j} \qquad \forall k \in K, \forall j \in V \tag{40}$$

$$y_{k,0} = 1 \qquad \forall k \in K \tag{41}$$

$$t_{k,j}^{\omega} \geq t_{k,i}^{\omega} - M + (M+1) x_{k,i,j}^{\omega} \qquad \begin{matrix} \forall k \in K, \\ \forall i,j \in V, V^+ \end{matrix} \tag{42}$$

$$t_{k,0}^{\omega} = 0 \qquad \forall k \in K \tag{43}$$

$$\sum_{i \in V} q_i y_{k,i} \leq C_k \qquad \forall k \in K \tag{44}$$

$$\sum_{k \in K} y_{k,i} = 1 \qquad \forall i \in V^+ \tag{45}$$

where all constraints where an ω apex refers to as $\forall \omega \in \Omega$. All x and y variables are binary. We have $M = |V|$, and $V^+ = V \setminus \{0\}$. Eq.(39) – (43) define the flattened problem, and Eq. (44) – (45) define the feasible space of the offline decision variables. For sake of simplicity, we eliminate sub-loops by keeping track of the visiting order t_{ki}^{ω} of each node for each vehicle: this is a simple, but not particularly effective method, because it relies on big-Ms and reduces the quality of the LP bound. We eliminate sub-loops in our VRP problem (Equation (42)) by keeping track of the visiting node order by adding a variable t and by imposing that node i must be visited before node i for each node and for each vehicle: this is a simple and well-known method that relies on big-Ms (Miller, Tucker, & Zemlin, 1960) but that also reduces the quality of the LP bound. We used it to avoid too complex constraints and to give priority to the insertion and applicability of KKT optimality conditions in order to better understand their effective computational cost.

6.1.3 INSTANTIATING ANTICIPATE FOR THE VRP

The ANTICIPATE method can be instantiated for each vehicle k separately, by first restricting the focus to the set of nodes V_h, and then by applying the definition from Section 4.1 and linearizing Eq. (37) in the baseline offline problem, we get:

$$\min \sum_{j \in V_h} c_{h,j} x_{k,i,j} + \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{i,j \in V_h} \xi_{i,j}^{\omega} x_{k,i,j}^{\omega} \qquad (\mathbf{P2.3})$$

$$\begin{aligned} \text{s.t. Eq. (35)} & \qquad \textit{– online problem constraints –} \\ \text{Eq. (39) restricted to } V_h \setminus \{0\} & \qquad (46) \\ \text{Eq. (40) – (42) restricted to } V_h & \qquad \textit{– state transition –} \\ t_{k,h}^{\omega} = 0 & \qquad (47) \end{aligned}$$

where Eq. (47) means that the vehicle path should start from the current node h (and end as usual in the depot).

6.1.4 INSTANTIATING ACTIVE/TUNING/ACKNOWLEDGE FOR THE VRP

As usual, the first step in the ACTIVE is formulating the KKT conditions for **P2.1**. In this case after some algebraic transformations, for a given vehicle k, node h, and scenario ω we obtain:

$$(c_{hj} + \lambda^{\omega}_{k,h})x^{\omega}_{k,h,j} = 0 \qquad \forall j \in V_h \tag{48}$$

$$(c_{hj} + \lambda^{\omega}_{k,h}) \geq 0 \qquad \forall j \in V_h \tag{49}$$

where $\lambda^{\omega}_{k,h}$ is the multiplier for Eq. (35): it is derived from the equality constraint in Eq. (30) following the definition of KKT conditions (Winston, 2004). Since this problem features no inequality constraints, the associated multipliers are simply absent. The main difficulty is again dealing with the set V_h, which is part of the state and should be constructed dynamically in the offline problem. Here, we handle V_h by introducing fresh variables r^{ω}_{kji} such that $r^{\omega}_{kji} = 1$ iff node i has been visited when node j is reached. The semantic is enforced via additional non-linear constraints in the offline model. The latter is given by:

$$\min \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \sum_{k \in K} \sum_{i,j \in V} \xi^{\omega}_{i,j} x^{\omega}_{k,i,j} \qquad (\mathbf{P2.4})$$

$$\text{s.t. Eq. (39)} - \text{(45)} \qquad \textit{– offline problem constraints –}$$

$$(c_{ij} + \lambda^{\omega}_{k,i})x^{\omega}_{k,i,j}(1 - r_{kji}) = 0 \quad \forall k \in K, \forall i,j \in V$$

$$(c_{ij} + \lambda^{\omega}_{k,i})(1 - r_{kji}) \geq 0 \qquad \forall k \in K, \forall i,j \in V$$

$$r^{\omega}_{k,i,i} = y_{k,i} \qquad \forall i \in V$$

$$r^{\omega}_{k,j,i} = r^{\omega}_{k,h,i} x^{\omega}_{k,h,j} \qquad \forall i \in V, \forall h \in V, \forall j \in V$$

$$\underline{c}_{ij} \leq c_{ij} \leq \overline{c}_{ij} \qquad \forall i,j \in V$$

The decision variables are y_{ki}, x^{ω}_{kij}, λ^{ω}_{ki}, r^{ω}_{kji}, plus the "virtual travel times" c_{ij}, i.e. the parameters for the online heuristic, always based on the method considered: they represent the cost associated to each route in terms of travel times. In some cases we refer to them as "virtual", since they may be modified by the offline solver and they do not correspond to a real travel time: as already discussed in Section 5.1, this is done to encourage or discourage the use of a most promising route. Concerning the meaning of the underbar and overbar versions of c_{ij}, we underline that bounding the virtual travel times is necessary to prevent the solver from building degenerate parameterizations for **P2.1** on purpose. This means, for example, that the offline solver may trivially satisfy the KKT constraints by setting all the travel times to 0. Finally, the constraints on the r^{ω}_{kji} variables enforce the transitive property on the set of visited nodes.

Also here, we recall that in ACKNOWLEDGE we consider y_{ki} as decision variables and c_{ij} as parameters using **P2.4** as offline phase; in TUNING we use **P2.2** to solve the y_{ki} variables and then we use **P2.4** for c_{ij} (i.e. our offline phase is divided in two parts); finally in ACTIVE we use **P2.4** with both y_{ki} and c_{ij} as decision variables of the *same offline phase*.

6.2 Results

Also in this case, as references for comparison we use the baseline approach, plus an optimal solver operating under perfect information. Additionally, we underline that, on the VRP (discrete decisions), ANTICIPATE obtains the same solution quality as EXPECTATION, which gives a third term of comparison.

6.2.1 EXPERIMENTAL SETUP

Our methods are evaluated over different uncertainty realizations, obtained by sampling the random variables for the travel times in the VRP model. We consider a sample of 100 realizations for six different instances of each problem. We then run each approach on each realization and measure the cost (total travel time) and run time.

We solve our LPs and MILPs using Gurobi, while for the non-linear problems we use BARON via the GAMS modeling system on the Neos server for optimization. The time limit is 500 seconds. We use classical SolomonTSPTW instances[3]. In particular, from the cited dataset, we chose instance I1 with 4 customers, I2 with 10 customers, I3 with 14 customers, I4 with 20 customers, I5 with 25 customers, and the last instance I6 with 30 customers (see Table 5). We always consider 2 available vehicles.

Concerning the scenarios, we assume that, whenever a node is reached, its binary state becomes known, and with that the (uniform) distributions followed by the travel times of all its outgoing arcs. Formally, this results in bimodally distributed, statistically dependent, travel times. The number of stages depends on how many customers are assigned to each vehicle.

Inst.	# of Customers
I1	4
I2	10
I3	14
I4	20
I5	25
I6	30

Table 4: Different instances for VRP

6.2.2 RESULT DISCUSSION

Tables 5 and 6 report the results in terms of costs (i.e. the total travel time t(s)) and computation time for the VRP. Here the online times are summed over all the vehicles. The original online heuristic is very efficient, but coupled with the baseline offline model it does not come close to the oracle quality. The offline model (a Mixed Integer Linear Program) takes also considerably more time to be solved. ANTICIPATE, which in this case yields the same results as EXPECTATION with no time limit, yields substantially better solutions, but, being also MILP-based, it takes non-negligible time during the online phase. The online phase of ANTICIPATE has the same semantic of EXPECTATION, which obtains the same results by enumerating the feasible decisions for the current stage, and evaluating the expected cost by solving the problem on each scenario. Since each scenario is considered in isolation, EXPECTATION is arguably much more efficient than (the online phase of) ANTICIPATE whenever the current stage decisions can be enumerated reasonably fast. The same argument applies to the REGRET algorithm, an efficient approximation of EXPECTATION. The improvement given by our methods is the integration with offline decisions (i.e. customer assignment in the offline phase of ANTICIPATE). Moreover, when each online stage requires to take multiple decisions, enumeration may be expensive and associating an expected cost to each decision in isolation leads to underestimations if the costs are not additive (Awasthi & Sandholm, 2009).

3. https://myweb.uiowa.edu/bthoa/TSPTWBenchmarkDataSets.htm

Instance	Oracle	Baseline				ANTICIPATE			
	c(t(s))	c(t(s))	tot t(s)	offline t(s)	online t(s)	c(t(s))	tot t(s)	offline t(s)	online t(s)
I1	**146.10**	165.83	1.954	1.699	0.255	**151.23**	8.833	1.699	7.134
I2	**278.37**	347.28	2.646	2.477	0.169	299.67	17.699	2.477	15.222
I3	**372.82**	561.66	3.086	2.532	0.554	**477.16**	20.556	2.532	18.024
I4	**321.57**	381.45	190.242	186.798	3.444	342.94	442.73	186.798	255.932
I5	**503.65**	670.86	248.578	243.330	5.248	559.22	556.986	243.330	313.656
I6	**448.53**	871.87	366.879	361.537	5.342	**470.99**	778.182	361.537	416.645

Table 5: Cost/computation time for the Oracle, Baseline and ANTICIPATE.

Instance	TUNING				ACKNOWLEDGE				ACTIVE			
	c(t(s))	tot t(s)	offline t(s)	online t(s)	c(t(s))	tot t(s)	offline t(s)	online t(s)	c(t(s))	tot t(s)	offline t(s)	online t(s)
I1	158.01	4.611	4.356	0.255	162.88	3.697	3.442	0.255	148.84	6.510	6.255	0.255
I2	312.33	12.546	12.377	0.169	320.43	10.398	10.229	0.169	**295.43**	17.614	17.445	0.169
I3	522.32	19.877	19.323	0.554	530.43	16.553	15.999	0.554	507.80	26.492	25.938	0.554
I4	355.32	298.691	295.247	3.444	368.94	291.788	288.344	3.444	**340.85**	342.442	338.998	3.444
I5	632.33	317.470	312.222	5.248	659.22	305.48	300.232	5.248	**543.92**	362.791	357.543	5.248
I6	584.33	427.642	422.300	5.342	605.88	410.575	405.233	5.342	504.82	496.198	490.856	5.342

Table 6: Cost/computation time for TUNING, ACKNOWLEDGE and ACTIVE.

The ACTIVE results follow the same trend as the VPP: the solution quality matches or beats that of ANTICIPATE, at the cost of a higher offline computation time, though the gap wrt the baseline is now much smaller. We want to stress again that our emphasis is on comparing different offline/online integration methods. EXPECTATION and REGRET all achieve a small amount of integration by relying on the same basic idea (making the online solver anticipatory). This is the same strategy used by our ANTICIPATE method which (for convenience reasons) was chosen as representative for this class of approaches. Additionally ANTICIPATE has the same semantic of EXPECTATION on the VRP problem, but it is applicable also in the VPP use case.

From Figure 19 to Figure 20 we show the average online routing decisions over the 100 realizations for the same instance I2 (10 customers, one depot and two vehicles). The heat maps shown below represent different colors for each vehicle and different color intensity for the number of times that each route has been chosen over the 100 realizations. We show only the heat maps for the two references (i.e. the Oracle and the Baseline) and for ANTICIPATE and ACTIVE that are the most significant results in terms of differences in offline/online decisions. We remember that our models make first offline decisions (i.e. assignment of clients for each vehicle) and then make routing (online) decisions. We remember also that in ACTIVE model we can have different offline decisions (compared to those made from the other three models) since we inject KKT conditions in the offline part. We therefore propose an instance as example where the offline decisions are the same for all the models with the aim to observe the different online routing decisions. Indeed, we have different trends with the same offline decisions. In particular, the Baseline model makes different routing decisions compared to the Oracle decisions: routes 2 -> 10, 3 -> 2, 6 -> 5 are never considered in the Baseline decisions while they are (with a certain probability) considered in the ANTICIPATE decisions. We can also notice that, the Baseline and ANTICIPATE models assume a (low) probability also for different routing decisions of vehicle 0 and this is not present in ACTIVE routing decisions. The ACTIVE routing decisions are equals to the Oracle ones for vehicle 0 in terms of probability and, for the other vehicle, they present routes (with the relative probability) never used by ANTICIPATE (e.g. 0 -> 6, 5 -> 7). Moreover, we can notice that ANTICIPATE presents online decisions with a higher probability but, in general, different from the most frequent decisions of the Oracle. Instead,

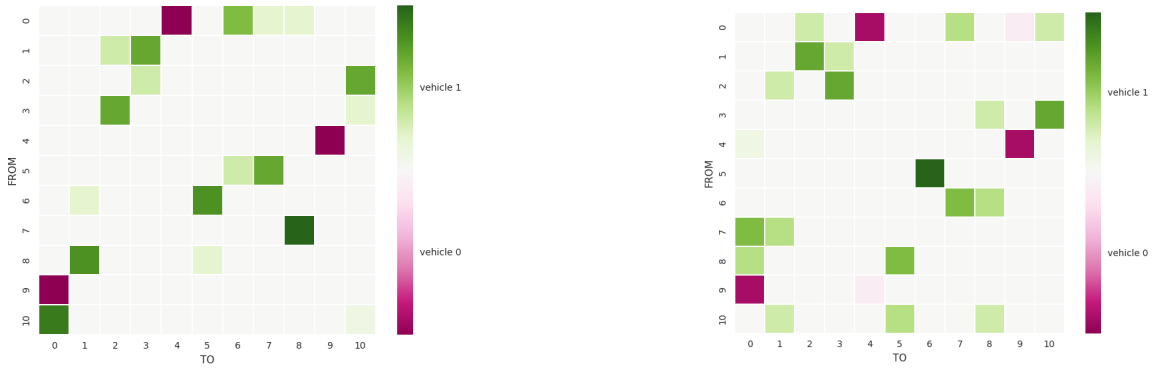

Figure 19: (left) Oracle and (right) Baseline routing decisions for instance I2

ACTIVE makes more decisions with lower probability than ANTICIPATE, but considering more often decisions similar to the Oracle.

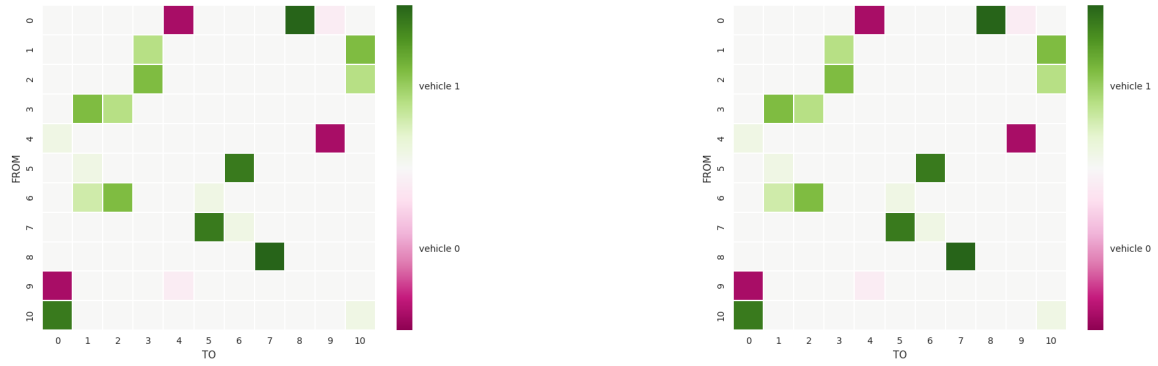

Figure 20: (left) ANTICIPATE and (right) ACTIVE routing decisions for instance I2

7. Conclusion and Future Work

This paper makes a first significant step toward generic integrated offline/online optimization. We propose four alternative approaches, based on the idea of making the offline and online solvers operate synergistically. In the ANTICIPATE method this is done by providing the online solver with the approximation of an oracle (i.e. replacing the greedy heuristic with a sampling-based anticipatory algorithm). However, increasing the computational load of the online phase may not a good idea when stringent time constraints exist. In such a situation, it may be better to improve the greedy heuristic by simply adjusting its parameters. This is the main idea in the TUNING approach which maintains the efficiency of the original greedy heuristic, at the price of a computationally expensive parameter tuning process, which is however performed offline. In the remaining methods, we instead make the offline solver aware of the limitations of the online one (i.e. ACKNOWLEDGE), and capable of controlling its behavior by adjusting parameters (i.e. ACTIVE). Indeed, shifting our attention to the offline decision, we can mitigate the discrepancy by translating the online greedy heuristic as a set of constraints, which can be injected in the offline model.

All the proposed techniques yield substantially improved solutions: ANTICIPATE matches the quality level of EXPECTATION, but it is applicable under more general assumptions. Unfortunately, the method is also less efficient. ACTIVE often manages to beat ANTICIPATE (and therefore EXPECTATION) in terms of solution quality. While this comes at the price of a substantially increased offline computation time, the method achieves these results by using naive and very efficient online heuristics.

Whenever distinct offline and online phases are present, a tighter integration can lead to substantial improvements in terms of both solution quality and computational costs. It would be very useful for the applicability of our methods to define a set of guidelines in order to choose one over another method, based on the problem under consideration. For this reason, we are working as current research direction, on the automatic recommendation of an approach based on the characteristics of the problem under consideration.

Regarding the limitations of our methods, we could make some discussion about their scalability. In particular, the scalability of several of our methods is adversely affected by the need to include additional variables and constraints to model the KKT conditions. Similarly to other stochastic optimization problems, large scenario sizes or look-ahead horizons result in better estimates, but also in more and/or bigger (possibly NP-hard) problems to be solved. However, this scalability issue is present mostly in the offline phase where computational resources are more available. We are working on research directions to improve the scalability of our methods by replacing sampling with (e.g.) ML predictors.

We also believe there is room for improving the efficiency of our methods (similarly to how EXPECTATION was improved in REGRET), and achieving this goal is part of our current research directions (De Filippo, Lombardi, & Milano, 2019). We also plan to apply our approaches to different problems, such as resource allocation and scheduling with Simple Temporal Networks under Uncertainty.

Acknowledgments

This work has been partially supported by the H2020 Project AI4EU (g.a. 825619), the VIRTUS Project (CCSEB-00094) and the Emilia-Romagna Region.

References

Awasthi, P., & Sandholm, T. (2009). Online stochastic optimization in the large: Application to kidney exchange.. In *IJCAI*, Vol. 9, pp. 405–411.

Bai, H., Miao, S., Ran, X., & Ye, C. (2015). Optimal dispatch strategy of a virtual power plant containing battery switch stations in a unified electricity market. *Energies*, *8*(3), 2268–2289.

Bent, R., & Van Hentenryck, P. (2004a). Online stochastic and robust optimization. In *Annual Asian Computing Science Conference*, pp. 286–300. Springer.

Bent, R., & Van Hentenryck, P. (2004b). Regrets only! online stochastic optimization under time constraints. In *AAAI*, Vol. 4, pp. 501–506.

Bent, R., & Van Hentenryck, P. (2004c). Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, *52*(6), 977–987.

Bertsimas, D., Brown, D. B., & Caramanis, C. (2011). Theory and applications of robust optimization. *SIAM review*, *53*(3), 464–501.

Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations research*, *52*(1), 35–53.

Bertsimas, D. J., & Simchi-Levi, D. (1996). A new generation of vehicle routing research: robust algorithms, addressing uncertainty. *Operations Research*, *44*(2), 286–304.

Birge, J. R., & Louveaux, F. (1997). *Introduction to Stochastic Programming. Series in Operations Research and Financial Engineering*. Springer.

Chisca, D., Lombardi, M., Milano, M., & O'Sullivan, B. (2018). From offline to online kidney exchange optimization. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 587–591. IEEE.

Clavier, J., Bouffard, F., Rimorov, D., & Jos, G. (2015). Generation dispatch techniques for remote communities with flexible demand. *IEEE Transactions on Sustainable Energy*, *6*(3), 720–728.

De Filippo, A., Lombardi, M., & Milano, M. (2018a). Methods for off-line/on-line optimization under uncertainty. In *IJCAI*.

De Filippo, A., Lombardi, M., & Milano, M. (2018b). Off-line and on-line optimization under uncertainty: A case study on energy management. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research - 15th International Conference, CPAIOR 2018, Delft, The Netherlands, June 26-29, 2018, Proceedings*, pp. 100–116.

De Filippo, A., Lombardi, M., & Milano, M. (2019). How to tame your anticipatory algorithm. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 1071–1077. AAAI Press.

De Filippo, A., Lombardi, M., & Milano, M. (2020a). The blind men and the elephant: Integrated offline/online optimization under uncertainty. In Bessiere, C. (Ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pp. 4840–4846. ijcai.org.

De Filippo, A., Lombardi, M., & Milano, M. (2020b). Hybrid offline/online optimization under uncertainty. In Giacomo, G. D., Catalá, A., Dilkina, B., Milano, M., Barro, S., Bugarín, A., & Lang, J. (Eds.), *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain*, Vol. 325 of *Frontiers in Artificial Intelligence and Applications*, pp. 2899–2900. IOS Press.

De Filippo, A., Lombardi, M., Milano, M., & Borghetti, A. (2017). Robust optimization for virtual power plants. In *Conference of the Italian Association for Artificial Intelligence*, pp. 17–30. Springer.

Diewert, W. E. (1974). *Applications of duality theory*. Stanford Institute for Mathematical Studies in the Social Sciences.

Espinosa, A. N., & Ochoa, L. (2015). Dissemination document low voltage networks models and low carbon technology profiles. *The University of Manchester*.

Gamou, S., Yokoyama, R., & Ito, K. (2002). Optimal unit sizing of cogeneration systems in consideration of uncertain energy demands as continuous random variables. *Energy Conversion and Management*, *43*(9), 1349 – 1361.

Herroelen, W., & Leus, R. (2005). Project scheduling under uncertainty: Survey and research potentials. *European Journal of Operational Research*, *165*(2), 289 – 306. Project Management and Scheduling.

Hodge, B. M., et al. (2012). Wind power forecasting error distributions: An international comparison. In *11th Annual International Workshop on Large-Scale Integration of Wind Power into Power Systems as well as on Transmission Networks for Offshore Wind Power Plants Conference*.

Hutter, F., Hoos, H. H., Leyton-Brown, K., & Stützle, T. (2009). Paramils: an automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, *36*, 267–306.

Jurković, K., Pandšić, H., & Kuzle, I. (2015). Review on unit commitment under uncertainty approaches. In *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2015 38th International Convention on*, pp. 1093–1097. IEEE.

Kall, P., & Wallace, S. W. (1994). *Stochastic programming*. Springer.

Kaplanis, S., & Kaplani, E. (2007). A model to predict expected mean and stochastic hourly global solar radiation i(h;nj) values. *Renewable Energy*, *32*(8), 1414 – 1425.

Kaut, M., & Wallace, S. W. (2003). *Evaluation of scenario-generation methods for stochastic programming*. Humboldt-Universität zu Berlin, Mathematisch-Naturwissenschaftliche Fakultät II, Institut für Mathematik.

Laporte, G., & Louveaux, F. V. (1993). The integer l-shaped method for stochastic integer programs with complete recourse. *Operations research letters*, *13*(3), 133–142.

Lee, C., Lee, K., & Park, S. (2012). Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *Journal of the Operational Research Society*, *63*(9), 1294–1306.

Li, P., Arellano-Garcia, H., & Wozny, G. (2008). Chance constrained programming approach to process optimization under uncertainty. *Computers & chemical engineering*, *32*(1-2), 25–45.

López-Ibáñez, et al. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, *3*, 43–58.

Manzo, S., Nielsen, O. A., & Prato, C. G. (2014). *Uncertainty calculation in transport models and forecasts*. Ph.D. thesis, PhD thesis, DTU Copenhagen.

Mercier, L., & Van Hentenryck, P. (2007). Performance analysis of online anticipatory algorithms for large multistage stochastic integer programs. In *IJCAI*.

Mercier, L., & Van Hentenryck, P. (2008). Amsaa: A multistep anticipatory algorithm for online stochastic combinatorial optimization. In *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming*, pp. 173–187. Springer.

Miller, C., Tucker, A., & Zemlin, R. (1960). Integer programming formulation of traveling salesman problems. *J. ACM*, *7*(4), 326–329.

Morales, J., Conejo, A., Madsen, H., Pinson, P., & Zugno, M. (2013). *Integrating renewables in electricity markets: operational problems*, Vol. 205. Springer Science & Business Media.

Palma-Behnke, R., et al. (2011). Energy management system for a renewable based microgrid with a demand side management mechanism. In *2011 IEEE Symposium on Computational Intelligence Applications In Smart Grid (CIASG)*, pp. 1–8.

Powell, W. B. (2016). *A Unified Framework for Optimization Under Uncertainty*, chap. 3, pp. 45–83.

Reddy, S. S., Sandeep, V., & Jung, C.-M. (2017). Review of stochastic optimization methods for smart grid. *Frontiers in Energy*, 1–13.

Sahinidis, N. V. (2004). Optimization under uncertainty: state-of-the-art and opportunities. *Computers & Chemical Engineering*, *28*(6), 971 – 983. FOCAPO 2003 Special issue.

Shapiro, A. (2008). Stochastic programming approach to optimization under uncertainty. *Mathematical Programming*, *112*(1), 183–220.

Shapiro, A. (2013). Sample average approximation. In *Encyclopedia of Operations Research and Management Science*, pp. 1350–1355. Springer.

Shapiro, A., & Philpott, A. (2007). A tutorial on stochastic programming. *Manuscript. Available at www2. isye. gatech. edu/ashapiro/publications. html*, *17*.

Tacs, D., Dellaert, N., Van Woensel, T., & De Kok, T. (2013). Vehicle routing problem with stochastic travel times including soft time windows and service costs. *Computers and Operations Research*, *40*(1), 214–224.

Toth, P., & Vigo, D. (2002). *The vehicle routing problem*. SIAM.

Van Hentenryck, P., & Bent, R. (2009). *Online stochastic combinatorial optimization*. The MIT Press.

Van Hentenryck, P., Bent, R., & Vergados, Y. (2006). Online stochastic reservation systems. In *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming*, pp. 212–227. Springer.

Wallace, S. W., & Fleten, S.-E. (2003). Stochastic programming models in energy. In *Stochastic Programming*, Vol. 10 of *Handbooks in Operations Research and Management Science*, pp. 637 – 677. Elsevier.

Walsh, T. (2002). Stochastic constraint programming. In *ECAI*, Vol. 2, pp. 111–115.

Winston, W. L. (2004). *Operations research: applications and algorithms*, Vol. 3.

Zhang, H., & Li, P. (2011). Chance constrained programming for optimal power flow under uncertainty. *IEEE Transactions on Power Systems*, *26*(4), 2417–2424.

Zhao, C., & Guan, Y. (2013). Unified stochastic and robust unit commitment. *IEEE Transactions on Power Systems*, *28*(3), 3353–3361.

Zheng, Q. P., Wang, J., & Liu, A. L. (2015). Stochastic optimization for unit commitment, a review. *IEEE Transactions on Power Systems*, *30*(4), 1913–1924.

Zhou, Z., et al. (2013). A two-stage stochastic programming model for the optimal design of distributed energy systems. *Applied Energy*, *103*, 135–144.